LNBI 4453

Terry Speed
Haiyan Huang (Eds.)

Research in Computational Molecular Biology

11th Annual International Conference, RECOMB 2007
Oakland, CA, USA, April 2007
Proceedings

Springer

Terry Speed Haiyan Huang (Eds.)

Research in Computational Molecular Biology

11th Annual International Conference, RECOMB 2007
Oakland, CA, USA, April 21-25, 2007
Proceedings

Springer

Series Editors

Sorin Istrail, Brown University, Providence, RI, USA
Pavel Pevzner, University of California, San Diego, CA, USA
Michael Waterman, University of Southern California, Los Angeles, CA, USA

Volume Editors

Terry Speed
Department of Statistics and Program in Biostatistics
University of California
Berkeley, CA 94720-3860, USA
E-mail: terry@stat.Berkeley.EDU

Haiyan Huang
Department of Statistics and Program in Biostatistics
University of California
Berkeley, CA 94720-3860, USA
E-mail: hhuang@stat.Berkeley.EDU

Library of Congress Control Number: 2007923790

CR Subject Classification (1998): F.2.2, F.2, E.1, G.2, H.2.8, G.3, I.2, J.3

LNCS Sublibrary: SL 8 – Bioinformatics

ISSN 0302-9743
ISBN-10 3-540-71680-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-71680-8 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12043330 06/3142 5 4 3 2 1 0

Preface

This volume contains the papers presented at the 11th Annual International Conference on REsearch in COMputational Biology, RECOMB 2007, held in the San Francisco Bay Area, USA, April 21-25, 2007. The RECOMB conference series was started in 1997 by Sorin Istrail, Pavel Pevzner and Michael Waterman, and the history is summarized in a table below. RECOMB 2007 was hosted by the California Institute for Quantitative Biomedical Research (QB3), and took place at the Oakland Marriott City Center in Oakland, California, USA. It was organized by a committee chaired by Sandrine Dudoit. The papers presented at the conference were selected by a Program Committee (PC) consisting of 40 members, ably assisted by numerous external reviewers. Thirty-seven papers were chosen by the PC from just under 170 submissions, and these appear in this volume. Each paper was reviewed by three members of the PC, or by one of the external reviewers acting as a sub-reviewer to a member of the PC. Following the initial reviews, there was a Web-based discussion leading to the final decisions. Selected papers from the proceedings will also be published in a special issue of the *Journal of Computational Biology.*

An important aspect of RECOMB is the presence of distinguished scientists presenting keynote addresses. This year we were honored to have with us at the meeting Elizabeth H. Blackburn (University of California, San Francisco, USA), Patrick O. Brown (Stanford University, USA), Abby Dernburg (University of California, Berkeley, USA). Jennifer Marshall Graves (Australian National University, Canberra, Australia), Yishi Jin (University of California, San Diego, USA), Jay D. Keasling (University of California, Berkeley, USA), Harry F. Noller (University of California, Santa Cruz, USA), and Aviv Regev (Broad Institute of Harvard and MIT, USA). Dr. Dernburg gave the Distinguished Biology Lecture, Dr. Brown gave the Distinguished Technology Lecture, and Dr. Regev gave the Stanislav Ulam Memorial Computational Biology Lecture. We also had close to 200 posters on display.

RECOMB 2007 was only possible through the support, dedication and hard work by many individuals and organizations. The conference was overseen by the RECOMB Steering Committee, while the PC and the external reviewers helped create the program. As well as an Organizing Committee whose members gave freely of their time, we were indebted to QB3 for hosting the conference and providing administrative, logistic and financial support, including the time of several dedicated QB3 staff members. In addition we received financial and other assistance from the University of California, the US Department of Energy, the US National Science Foundation, the International Society of Computational Biology, and our Sponsors. Finally, it is important to give special thanks to all those who contributed papers or posters, and those who attended the conference. Their enthusiastic participation is what makes RECOMB conferences such fun.

April 2007 Terry Speed

Organization

Program Committee

Tatsuya Akutsu	Kyoto University, Japan
Vineet Bafna	University of California, San Diego, USA
Serafim Batzoglou	Stanford University, USA
Gill Bejerano	University of California, Santa Cruz, USA
Mathieu Blanchette	McGill University, Canada
Ting Chen	University of Southern California, USA
Andrew Clark	Cornell University, USA
Dannie Durand	Carnegie Mellon University, USA
Eleazar Eskin	University of California, Los Angles, USA
Nir Friedman	Hebrew University, Jerusalem, Israel
Dan Geiger	Technion, Haifa, Israel
Dan Gusfield	University of California, Davis, USA
Alexander Hartemink	Duke University, USA
Ian Holmes	University of California, Berkeley, USA
Liisa Holm	University of Helsinki, Finland
Susan Holmes	Stanford University, USA
Rafael Irizarry	Johns Hopkins University, USA
Sorin Istrail	Brown University, USA
Manolis Kellis	MIT and Broad Institute, USA
Sung Wing Kin	National University of Singapore
Jens Lagergren	Royal Institute of Technology, Sweden
Thomas Lengauer	Max Planck Institute for Informatics, Germany
Michael Levitt	Stanford University, USA
Michal Linial	Hebrew University, Israel
Jacek Majewski	McGill University, Canada
Richard Mott	Oxford University, UK
William Noble	University of Washington, USA
Lior Pachter	University of California, Berkeley, USA
Pavel Pevzner	University of California, San Diego, USA
Marie-France Sagot	INRIA, Lyon, France
Sophie Schbath	INRA, Jouy-en-Josas, France
Ron Shamir	Tel Aviv University, Israel
Mona Singh	Princeton University, USA
Terry Speed (Chair)	University of California, Berkeley, USA
Simon Tavaré	Cambridge University, UK
Esko Ukkonen	University of Helsinki, Finland
Martin Vingron	Max Planck Institute for Molecular Genetics, Germany
Wing Hung Wong	Stanford University, USA

Zohar Yakhini	Agilent Labs, USA and Technion, Israel
Golan Yona	Cornell University, USA and Technion, Israel

Steering Committee

Serafim Batzoglou	Stanford University, USA
Sorin Istrail	RECOMB Vice-Chair, Brown University, USA
Thomas Lengauer	Max Planck Institute for Informatics, Germany
Michal Linial	Hebrew University, Israel
Pavel A. Pevzner	RECOMB General Chair, UC San Diego, USA
Ron Shamir	Tel Aviv University, Israel
Terry Speed	University of California, Berkeley, USA
Martin Vingron	Max Planck Institute for Molecular Genetics, Germany

Organizing Committee

Inna Dubchak	Lawrence Berkeley National Laboratory
Serafim Batzoglou	Stanford University
Susan Holmes	Stanford University
Daphne Koller	Stanford University
Michael Levitt	Stanford University
Wing Hung Wong	Stanford University
Adam Arkin	University of California, Berkeley
Sandrine Dudoit (Chair)	University of California, Berkeley, USA
Michael Eisen	University of California, Berkeley
Ian Holmes	University of California, Berkeley
Haiyan Huang	University of California, Berkeley
Nick Jewell	University of California, Berkeley
Lior Pachter	University of California, Berkeley
Kimmen Sjölander	University of California, Berkeley
Terry Speed	University of California, Berkeley
Craig Benham	University of California, Davis
Dan Gusfield	University of California, Davis
Dawei Lin	University of California, Davis
David Rocke	University of California, Davis
Patricia Babbitt	University of California, San Francisco
Tanja Kortemme	University of California, San Francisco
Hao Li	University of California, San Francisco
Mark Segal	University of California, San Francisco
Chris Voigt	University of California, San Francisco
Ru-Fang Yeh	University of California, San Francisco
David Haussler	University of California, Santa Cruz
Todd Lowe	University of California, Santa Cruz
Raquel Prado	University of California, Santa Cruz
Josh Stuart	University of California, Santa Cruz

Previous RECOMB Meetings

Date/Location	Hosting Institution	Program Chair	Conference Chair
January 20-23, 1997 Santa Fe, NM, USA	Sandia National Lab	Michael Waterman	Sorin Istrail
March 22-25, 1998 New York, NY, USA	Mt. Sinai School of Medicine	Pavel Pevzner	Gary Benson
April 22-25, 1999 Lyon, France	INRIA	Sorin Istrail	Mireille Regnier
April 8-11, 2000 Tokyo, Japan	University of Tokyo	Ron Shamir	Satoru Miyano
April 22-25, 2001 Montréal, Canada	Université de Montréal	Thomas Lengauer	David Sankoff
April 18-21, 2002 Washington, DC, USA	Celera	Gene Myers	Sridhar Hannenhalli
April 10-13, 2003 Berlin, Germany	German Federal Ministry for Education and Research	Webb Miller	Martin Vingron
March 27-31, 2004 San Diego, USA	UC San Diego	Dan Gusfield	Philip E. Bourne
May 14-18, 2005 Boston, MA, USA	Broad Institute of MIT and Harvard	Satoru Miyano	Jill P. Mesirov and Simon Kasif
April 2-5, 2006 Venice, Italy	University of Padova	Alberto Apostolico	Concettina Guerra

The RECOMB 2007 Program Committee Gratefully Acknowledges the Valuable Input Received from the Following External Reviewers:

Örjan Åkerborg
Max Alekseyev
Julien Allali
Peter Arndt
Lars Arvestad
George Asimenos
Yonatan Aumann
Erik Aurell
David Balding
Nuno L. Barbosa-Morais
Ziv Bar-Joseph
Chuong B Do
Asa Ben-Hur
Anne Bergeron
Gilles Bernot
Yonatan Bilu
Etienne Birmele
David Bryant
Jeremy Buhler
Peter Calabrese
Jose Carlos Nacher
Mathilde Carpentier
Anat Caspi
Natalie Castellano
Dominique Cellier
Sourav Chatterji
Gal Chechik
Ho-Ryun Chung
J. Coulombe-Huntington
Jean-Jacques Daudin
Eugene Davydov
Minghua Deng
Ameya Deoras
Rick Desper
Nuno Dias Mendes
Zihong Ding
Andreas Doering
Francisco Domingues
Banu Dost
Gideon Dror
Samuel Druhle
Debo Dutta
Eran Eden
Tal El-Hay
Isaac Elias
Rani Elkon
Olof Emanuelsson
Nicholas Eriksson
Lene Monrad Favrholdt
Vladimir Filkov
Jason Flannick
Ari Frank
Eugene Fratkin
Menachem Fromer
Daniel Gaffney
Irit Gat-Viks
Josh Grochow

Samuel Gross
Laurent Gueguen
Stefan Haas
Naomi Habib
Yonit Halperin
Christoph Hartmann
Chris Holmes
Peter Huggins
Alex Inberg
Rui Jiang
Öjvind Johansson
Daniel Kahn
Lukas Kall
Tommy Kaplan
Pouya Kheradpour
Gad Kimmel
Bonnie Kirkpatrick
Teemu Kivioja
Aaron Klammer
Ina Koch
Patrice Koehl
Mikko Koivisto
Michel Koskas
Dennis Kostka
Mehmet Koyuturk
Gert Lanckriet
Hyunju Lee
Nolwenn Le Meur
Christina Leslie
Yaviv Lewinstein
Mike Lin
Chaim Linhart
Doron Lipson
Alair Pereira do Lago
Hannes Luz
Xiaotu Ma
Vladimir Makarenkov
Veli Mäkinen
Thomas Manke
Tobias Mann
John Marioni
Catherine Matias
Michael McCoss
Christian Merkwirth
Ofer Meshi
Shipra Metah Hyun
Min Kang
Pierre Nicolas
Matan Ninio
Antal Novak
Guillaume Obozinski
Arlindo Oliveira
Sean O'Rourke
Michal Ozery-Flato
Kimmo Palin
Pierre Peterlongo
Yanjun Qi
Jian Qiu
Matthias Rarey
Matt Rasmussen
Stephane Robin
Helge Roider
Ingo Ruczinski
Larry Ruzzo
Oliver Sander
Mike Sanderson
Sriram Sankararaman
Alain Schenkel
Thomas Schiex
Alexander Schliep
Ariel Schwartz
Bengt Sennblad
Roded Sharan
Noa Shefi
Alexandra Shulman-Peleg
Brian Smith
Yun Song
Srinath Sridhar
Alex Stark
Kristian Stevens
Katherine St John
Maureen Stolzer
Andreas Sundquist
Stephen Tanner
Eric Tannier
Andrew Teschendorff
Alexander Thielen
Robert Thurman
Ali Tofigh
Petri Toronen
Olga Troyanskaya
Anya Tsalenko
Dekel Tsur
Zhidong Tu
Igor Ulitsky
Jacques Van Helden
Jean-Stéphane Varre
Roy Varshavsky
Balaji Venkatuachalam
Benjamin Vernot
Jean-Philippe Vert
Inna Weiner
Yufeng Wu
Benny Yakir
Moran Yasour
Nir Yosef
Noah Zaitlen
Tomasz Zemojtel
Yu Zhang
Michal Ziv-Ukelson

RECOMB 2007 Venue: Oakland Marriott City Center

Sponsors

AFFYMETRIX®

bioNovo

Genentech
IN BUSINESS FOR LIFE

iSCB INTERNATIONAL SOCIETY FOR COMPUTATIONAL BIOLOGY

Helicos
BioSciences Corporation

MERCK
Research Laboratories

Roche

Wyeth

Hosted by qb3
ucb·ucsc·ucsf
California Institute
for Quantitative Biomedical Research

Table of Contents

QNet: A Tool for Querying Protein Interaction Networks

Banu Dost[1,*], Tomer Shlomi[2,*], Nitin Gupta[1], Eytan Ruppin[2,3], Vineet Bafna[1], and Roded Sharan[2]

[1] Computer Science and Engineering,
Univ. of California, San Diego, CA 92093, USA
{bdost,ngupta,vbafna}@cs.ucsd.edu
[2] School of Computer Science, Tel Aviv University, 69978 Tel Aviv, Israel
{shlomito,ruppin,roded}@post.tau.ac.il
[3] School of Medicine, Tel Aviv University, 69978 Tel Aviv, Israel

Abstract. Molecular interaction databases can be used to study the evolution of molecular pathways across species. Querying such pathways is a challenging computational problem, and recent efforts have been limited to simple queries (paths), or simple networks (forests). In this paper, we significantly extend the class of pathways that can be efficiently queried to the case of trees, and graphs of bounded treewidth. Our algorithm allows the identification of non-exact (homeomorphic) matches, exploiting the color coding technique of Alon et al. We implement a tool for tree queries, called QNet, and test its retrieval properties in simulations and on real network data. We show that QNet searches queries with up to 9 proteins in seconds on current networks, and outperforms sequence-based searches. We also use QNet to perform the first large scale cross-species comparison of protein complexes, by querying known yeast complexes against a fly protein interaction network. This comparison points to strong conservation between the two species, and underscores the importance of our tool in mining protein interaction networks.

1 Introduction

The study of biological networks has gained substantial interest in recent years. In particular, technological advances, such as the yeast two-hybrid [11] and co-immunoprecipitation assays [15], have enabled the large-scale mapping of protein-protein interactions (PPIs) across many model species. The newly available PPI networks present a host of new challenges in studying protein function and evolution. Key to addressing these challenges is the development of efficient tools for network database searches, much the same as sequence searches have been instrumental in addressing similar problems at the genome level.

Network queries call for searching a "template" subnetwork within a network of interest. Commonly, the query is a known pathway, and the network is searched for subnetworks that are similar to the query. Similarity is measured

* These authors contributed equally to this work.

both in terms of protein sequence similarity and in terms of topological similarity. The hardness of the problem stems from the non-linearity of a network, making it difficult to apply sequence alignment techniques for its solution.

Several authors have studied the network querying problem, mostly focusing on queries with restricted topology. Kelley et al. [13] devised an algorithm for querying linear pathways in PPI networks. While the problem remains NP-hard in this case as well (as, e.g., finding the longest path in a graph is NP-complete [7]), an efficient algorithm that is polynomial in the size of the network and exponential in the length of the query was devised for it. Pinter et al. [17] enable fast queries of more general pathways that take the form of a tree. However, their algorithm is limited to searching within a collection of trees rather than within a general network. Sohler and Zimmer [6] developed a general framework for subnetwork querying, which is based on translating the problem to that of finding a clique in an appropriately defined graph. Due to its complexity, their method is applicable only to very small queries. Recently, some of us have provided a comprehensive framework, called QPath, for linear pathway querying. QPath is based on an efficient graph theoretic technique, called color coding [1], for identifying subnetworks of "simple" topology in a network. It improves upon [13] both in speed and in higher flexibility in non-exact matches.

In this paper, we greatly extend the QPath algorithm to allow queries with more general structure than simple paths. We provide an algorithmic framework for handling tree queries under non-exact (homeomorphic) matches (Section 3.1). In this regard, our work extends [17] to querying within general networks, and the results in [1] to searching for homeomorphic rather than isomorphic matches. More generally, we provide an algorithm for querying subnetworks of bounded treewidth (Section 3.2). We implemented a tool for tree queries which we call QNet. We demonstrate that QNet performs well both in simulation of synthetic pathway queries, and when applied to mining real biological pathways (Section 5). In simulations, we show that QNet can handle queries of up to 9 proteins in seconds in a network with about 5,000 vertices and 15,000 interactions, and that it outperforms sequence-based searches. More importantly, we use QNet to perform the first large scale cross-species comparison of protein complexes, by querying known yeast complexes in the fly protein interaction network. This comparison points to strong conservation of protein complexes structures between the two species. For lack of space some algorithmic details are omitted in the sequel.

2 The Graph Query Problem

Let $G = (V, E, w)$ be an undirected weighted graph, representing a PPI network, with a vertex set V of size n, representing proteins, an edge set E of size m, representing interactions, and a weight function $w : E \to \mathcal{R}$, representing interaction reliabilities.

Let $G_Q = (V_Q, E_Q)$ denote a query graph with k vertices. We reserve the term *node* for vertices of G_Q and use the term *vertex* for vertices of G.

Let $h(q,v)$ denote a similarity score between query node $q \in V_Q$ and vertex $v \in V$. In our context, vertices correspond to proteins, and their similarity score is a function of their sequence similarity. A query node q is referred to as *homologous* to a graph vertex v, if the corresponding similarity score $h(q,v)$ exceeds a predefined threshold.

A *subdivision* of an edge (u,v) in a graph $H=(U,F)$ replaces it with two edges (u,w) and (w,v), where $w \notin U$, i.e., creating a new graph $H' = (U \cup \{w\}, F \cup \{(u,w),(w,v)\} \setminus \{u,v\})$. H is considered *extendable* to a graph G, if G can be obtained from H by a series of subdivisions. In particular, H is then homeomorphic to G.

An *alignment* of the query graph G_Q to G is defined as a pair of: (i) a subgraph $G_A = (V_A, E_A)$ of G, referred to as the *alignment subgraph*; and (ii) a bijection, $\sigma : V_Q^S \to V_A^S$, between a subset of query nodes, $V_Q^S \subseteq V_Q$, and homologous vertices in the alignment subgraph, $V_A^S \subseteq V_A$. The vertices in $V_Q^S \cup V_A^S$ are called *skeleton* vertices. Pairs of associated vertices $(q, \sigma(q)) \in V_Q^S \times V_A^S$ are called *aligned*.

An alignment is *proper* if there exists a pair of *skeleton* graphs $S_Q = (V_Q^S, E_Q^S)$ and $S_A = (V_A^S, E_A^S)$ that satisfy the following conditions: (i) there is an isomorphism between S_Q and S_A which respects the alignment (i.e., there is an edge $(u,v) \in E_Q^S$ iff there is an edge $(\sigma(u), \sigma(v)) \in E_A^S$); and (ii) S_Q is extendable to G_Q and S_A is extendable to G_A. In particular, this means that G_Q and G_A are required to be homeomorphic. In the rest of the paper we discuss proper alignments only. An example of such an alignment is given in Figure 1a.

Query nodes that are not aligned with vertices in the alignment subgraph are considered to be *deleted*. Conversely, vertices in the alignment subgraph that are not aligned with query nodes are considered to be *inserted*. Insertions and deletions are also referred to as *indels*. From the above definitions, inserted and deleted vertices must be of degree 2 in their respective graphs. An alignment which involves no insertions or deletions is considered *simple*. The weight of an alignment is the sum of: (i) similarity scores of aligned vertices, (ii) weights of edges in the aligned subgraph, (iii) a penalty score, δ_d, for each node deletion, and (iv) a penalty score, δ_i, for each vertex insertion.

The *graph query problem* is formally defined as follows: Given a query graph G_Q, a graph G, a similarity score h, and penalty scores for insertions and deletions, find a proper alignment of G_Q in G with maximal weight. In practice, we would also like to limit the number of insertions and deletions in the alignment, to control the evolutionary distance between the two subnetworks. To this end, we also consider a variant of the problem in which the number of insertions is limited by N_{ins}, and the number of deletions is limited by N_{del}.

3 Graph Query Algorithms

The complexity of the graph query problem depends on the topology of the query graph G_Q, the topology of the graph G, and the similarity function h. In the general case, the problem of finding simple alignments is in general equivalent to subgraph isomorphism [8], which is computationally hard. In this paper, we focus on efficient query algorithms by exploiting the underlying biological constraints. Specifically, motivated by known pathways in KEGG [12], we consider restricted query topologies, i.e., the query graph being a *tree*, and a graph of *bounded treewidth* (see also [17]). For these special structures, we adapt the color coding method of Alon et al. [1] to make the problem tractable.

Color coding is a randomized technique for finding simple paths and simple cycles of a specified length k within a given graph of size n. The basic idea is to randomly assign k colors to the vertices of the graph and then search for *colorful* paths in which each color is used exactly once. Thus, rather than having to maintain a list of vertices visited so far (of size $O(n^k)$), one can maintain a list of colors at considerably lower complexity ($O(2^k)$).

The use of the color coding technique within a query algorithm is intuitively similar. We construct an optimal alignment by extending optimal sub-alignments using dynamic programming. Adding a network vertex to the optimal alignment can be done only if this vertex is not already contained in the sub-optimal alignment. Thus, naively, each potential sub-optimal alignment should maintain the list of at most k vertices already matched. This yields $O(n^k)$ potential alignments. In color coding, we apriori color each network vertex randomly with one of k colors, looking for a colorful alignment. Consequently, we only need to maintain a list of used colors (of size $O(2^k)$), which significantly reduces the computation time. However, the computation returns a correct answer only if the optimum alignment is colorful, which happens with probability $\frac{k!}{k^k} \simeq e^{-k}$. Therefore, if we repeat the experiment $\ln(\frac{1}{\epsilon})e^k$ times, we get the optimum alignment with probability at least $1 - \epsilon$ for any desired value of ϵ.

3.1 Tree Query

We describe an algorithm for solving the graph query problem assuming that the query graph is a tree. For ease of presentation, we start by presenting a simplified version of the algorithm that limits the number of insertions only. The proper treatment of limiting both the number of insertions and deletions is deferred to the end of the section.

First, we root G_Q arbitrarily at a node r with degree 1. For each query node q, denote its children by $q_1, \ldots, q_{n_q}$, where n_q denotes their number. Let $T_{q,j}$ denote the tree that includes q and the subtrees rooted at each of its first j children, for $1 \leq j \leq n_q$. The algorithm proceeds in a series of trials in which every vertex $v \in V$ is independently assigned a color $c(v)$ drawn uniformly at random from the set $C = \{1, 2, \ldots, k + N_{ins}\}$. Given the random vertex colors, we employ dynamic programming to identify an optimal colorful alignment. Let $W^M(q, v, S, j)$ denote the maximal score of an alignment of $T_{q,j}$ in G, such that query node q is aligned with graph vertex v, with the aligned subgraph receiving distinct colors from $S \subseteq C$. The recursion is initialized by setting $W^M(q, v, S, 0) = h(q, v)$ for leaf nodes q, and is formulated as follows:

$$W^M(q,v,S,j) = \max_{\substack{u:(u,v)\in E \\ S'\subset S}} \begin{cases} \text{(* \textit{Match, child j} *)} \\ W^M(q,v,S',j-1)+W^M(q_j,u,S-S',n_{q_j})+w(u,v), \\ \\ \text{(* \textit{Insertion, vertex u} *)} \\ W^M(q,v,S',j-1)+W^I(q_j,u,S-S')+w(u,v), \\ \\ \text{(* \textit{Deletion, child j} *)} \\ W^M(q,v,S',j-1)+W^D(q_j,v,S-S') \end{cases}$$

Here $W^I(q,v,S)$ denotes the optimal score of an alignment of T_{q,n_q} in G, such that q is aligned with some vertex u that is a descendant of v in the aligned subgraph. $W^D(q,v,S)$ denotes the optimal score of the alignment of $T_{q,1}$ in G, such that q is deleted and v is aligned with an ancestor of q. The recursions for the insertion and deletions cases are given below. For query nodes q of degree other than 2, we set $W^D(q,v,S) = -\infty$.

$$W^I(q,v,S) = \max_{u:(u,v)\in E} \begin{cases} W^M(q,u,S-\{c(v)\},n_q)+w(u,v)+\delta_i, \\ W^I(q,u,S-\{c(v)\})+w(u,v)+\delta_i \end{cases}$$

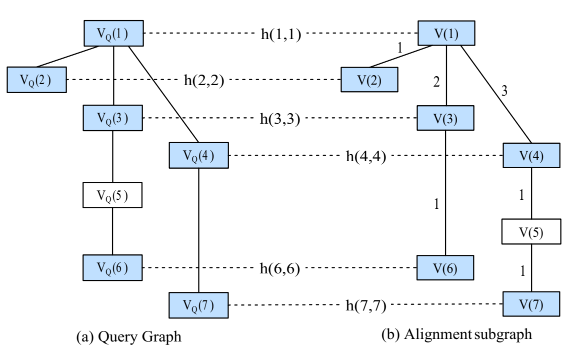
$$W^D(q,v,S) = \max_{u:(u,v)\in E} \begin{cases} W^M(q_1,u,S,n_{q_1})+w(u,v)+\delta_d, \\ W^I(q_1,u,S)+w(u,v)+\delta_d, \\ W^D(q_1,v,S)+\delta_d \end{cases}$$

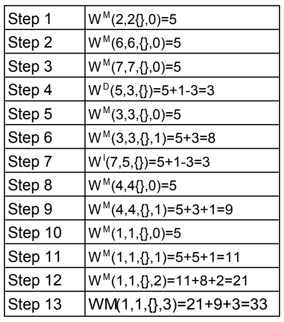
The maximal score of the alignment is $\max_{v,S} W^M(r,v,S,1)$. The optimal alignment is obtained through standard dynamic programming backtracking. An application of the dynamic programming recursions to a sample query is demonstrated in Figure 1.

The running time of each trial is $2^{O(k+N_{ins})}m$. The probability of receiving distinct colors for the vertices of the optimal matching tree is at least $e^{-k-N_{ins}}$. Thus, the running time of the algorithm is $2^{O(k+N_{ins})}m\ln(\frac{1}{\epsilon})$ for any desired success probability $1-\epsilon$ (where $\epsilon > 0$). We note that it is straightforward to limit the number of deletions to N_{del} by incorporating an additional variable in the recursions to count the number of deletion in the optimal sub-alignment. The cost in terms of running time is multiplicative in N_{del}. When incorporating such a variable, it is also easy to limit the number of insertions to N_{ins} by choosing the optimum solution based on its number of deletions and the cardinality of its color set.

3.2 Bounded Treewidth Graph Query

The algorithm for matching trees can be extended to subgraphs that have tree-like properties. We present an algorithm for the simpler case where no indels are allowed and defer the description of an algorithm for the general case to the appendix. Intuitively, the treewidth of a graph indicates how close the graph is to being a tree, where a tree has treewidth 1. The maximal treewidth value for





Step 1	$W^M(2,2\{\},0)=5$
Step 2	$W^M(6,6,\{\},0)=5$
Step 3	$W^M(7,7,\{\},0)=5$
Step 4	$W^D(5,3,\{\})=5+1-3=3$
Step 5	$W^M(3,3,\{\},0)=5$
Step 6	$W^M(3,3,\{\},1)=5+3=8$
Step 7	$W^I(7,5,\{\})=5+1-3=3$
Step 8	$W^M(4,4\{\},0)=5$
Step 9	$W^M(4,4,\{\},1)=5+3+1=9$
Step 10	$W^M(1,1,\{\},0)=5$
Step 11	$W^M(1,1,\{\},1)=5+5+1=11$
Step 12	$W^M(1,1,\{\},2)=11+8+2=21$
Step 13	$WM(1,1,\{\},3)=21+9+3=33$

(c) Dynamic Programming steps

Fig. 1. (a) An example of a tree query graph and the corresponding alignment subgraph. Numbers on the query graph's edges represent an arbitrary ordering of children nodes. Aligned query nodes and graph vertices are connected with dashed lines. Nodes in the skeleton graphs appear in gray. (b) A simulation of the dynamic programming recursions. For simplicity, we denote color sets as $\{\}$. Matched vertices are awarded by $+5$, insertions and deletions are penalized by -3 and edge weights are as shown.

a graph with n vertices is $n-1$ and this value is attained by an n-vertex clique. A formal definition of a treewidth and the associated tree-like structure follows.

A *tree decomposition* (X,T) of the query graph $G_Q=(V_Q,E_Q)$ is defined as follows (see, e.g., [14]): $T=(I,F)$ is a rooted binary tree, and $X=\{X_i\subseteq V_Q : i\in I\}$ is a collection of subsets of V_Q, such that $\bigcup_{i\in I}X_i=V_Q$ and the following conditions are satisfied:

1. For each edge $(u,v)\in E_Q$ there exists $i\in I$ such that $u,v\in X_i$.
2. If $i,j,k\in I$ and j is on the path from i to k in T, then $X_i\bigcap X_k\subseteq X_j$.

The *treewidth* of the tree decomposition is $max_{i\in I}|X_i|-1$. An example of a graph and its tree decomposition is given in Figure 2a,b.

Let t denote a bound on the treewidth of G_Q. We add a dummy node d as a parent of the root of T, with $X_d=\emptyset$. To avoid confusion, we call the nodes of T, *super-nodes*. For a non-leaf tree super-node $X_i\in X$, denote its two children by X_{i_1} and X_{i_2}. Let T_i denote the subtree of T that is rooted at X_i. The algorithm proceeds in a series of trials in which every vertex $v\in V$ is independently assigned a color $c(v)$ drawn uniformly at random from the set $\{1,2,\ldots,k\}$. Given the random vertex colors, we employ dynamic programming to identify an optimal colorful alignment.

The properties of the tree decomposition enable us to identify the optimal alignment by recursing on T and maintaining sub-optimal alignments of query nodes spanned by subtrees of T, similar to the tree query algorithm described above. However, there are two main difficulties to tackle: (i) A set of query nodes, X_i, may have an arbitrary topology (e.g., forming a clique), potentially requiring an exhaustive $O(n^{t+1})$-time search of an alignment subgraph for it. (ii) A query node v may appear in more than a single super-node.

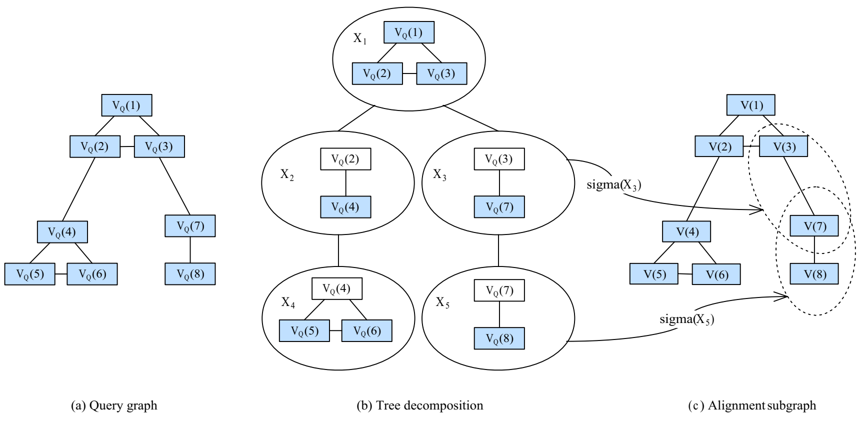


Fig. 2. (a) An example of a query graph with a treewidth of 2. (b) A tree decomposition of the query graph such that each super-node has no more than 3 query nodes associated with it. Non-active query nodes are grayed. (c) An alignment subgraph. $\sigma(X_3)$ and $\sigma(X_5)$ are mappings of the query nodes in X_3 and X_5 to graph vertices, respectively, that identify on the active query node $V_Q(6)$ in X_5.

For the first issue, we exploit the fact that the treewidth is bounded by t. Large values of t would make the algorithm impractical. To cope with the second difficulty, we note that by definition, if $v \in X_{i_j}$ and $v \notin X_i$, then $v \notin X_l$ for all super-nodes X_l that are not descendants of X_i in the tree. Thus, when visiting a certain super-node X_{i_j}, it contains *active* query nodes $X^A_{i_j} = X_i \cap X_{i_j}$ that are yet to be handled, and *non-active* nodes $X^N_{i_j}$ that can be removed from consideration when traversing up the tree (Figure 2b). We define a *non-active* edge at a super-node X_i, as a query edge touching a non-active node in X_i. We let E^N_i denote the set of non-active edges in super-node X_i.

We need some more notation before giving the main recurrence of the algorithm. For each $X_i \in X$, let Σ_i denote the $O(n^{t+1})$-size set of all mappings $\sigma : X_i \to V$ such that: (i) for all distinct $q_1, q_2 \in X_i$, $c(\sigma(q_1)) \neq c(\sigma(q_2))$; and (ii) if $(q_1, q_2) \in E_Q$ then $(\sigma(q_1), \sigma(q_2)) \in E$. Figure 2b,c shows an example of mappings between query nodes and graph vertices.

For computing the weight of an alignment, it is convenient to credit each super-node i (when traversing up the tree) with the similarity scores associated with its non-active nodes and the edge weights corresponding to its non-active edges. The node term is $W^S(i,\sigma) = \sum_{u \in X^N_i} h(u, \sigma(u))$. The edge term is $W^E(i,\sigma) = \sum_{(u_1,u_2) \in E^N_i} w(\sigma(u_1), \sigma(u_2))$.

Let $W(i, \sigma, S)$ be the maximum weight of an alignment of a subgraph of G_Q that includes all super-nodes in $T_i - X_i$, identifies on the active query nodes in super-node i with the assignment $\sigma \in \Sigma_i$, and uses the colors in $S \subseteq C$. $W(i, \sigma, S)$ can be recursively computed as follows. For a leaf i, $W(i, \sigma, S) = 0$. For all other super-nodes:

$$W(i,\sigma,S) = \max_{\substack{S_1 \uplus S_2 = S \\ \sigma_1,\sigma_2}} \sum_{j=1}^{2} \left[W(i_j,\sigma_j,S_j) + W^S(i_j,\sigma_j) + W^E(i_j,\sigma_j) \right]$$

where σ is consistent with $\sigma_1 \in \Sigma_{i_1}$ and $\sigma_2 \in \Sigma_{i_2}$.

The score of an optimal alignment of G_Q is thus $\max_S W(d,\emptyset,S)$. The total running time is $2^{O(k)}n^{t+1}$.

4 Implementation Notes

We implemented a tool, QNet, for querying a given network with a tree subnetwork, following the algorithm given in Section 3.1. Bounded treewidth queries will be supported in future versions. To allow higher flexibility in matching a query, we slightly generalized the tree query algorithm to enable also deletions of query nodes of degree 1 (leaves of the tree). We also included in QNet a heuristic that exploits the structure of the homology function to reduce the number of color coding iterations needed. In the following we describe this heuristic and the parameter setting employed in QNet.

Restricted Color Coding. We present a heuristic approach to color coding that tries to take advantage of queries whose protein members tend to have non-overlapping sets of homologs. First, we assign each query node a distinct *match color*, and choose N_{ins} additional *insertion colors*. Now, we color the network vertices using the following rule: For each network vertex v, if v is not homologous to any query protein, then assign it with a random insertion colors. Otherwise, toss a coin with probability $p_t = \frac{N_{ins}}{k+N_{ins}}$. If HEADS, choose a random insertion color for it, else if TAILS, assign it with a random color from the set of query nodes it is homologous to.

The probability P_s to obtain a colorful alignment subgraph is at least the probability that: (i) each aligned vertex is given a match color, and each inserted vertex is given an insertion color; and (ii) all colors are distinct. Let p_m be the probability that aligned vertices are colorful, and p_i be the probability that insertion vertices are colorful. Then

$$P_s = (1-p_t)^k p_t^{n_i} p_i p_m = \left(\frac{k}{N_{ins}+k}\right)^k \left(\frac{N_{ins}}{k+N_{ins}}\right)^{N_{ins}} p_i p_m$$

where $p_i \geq \frac{N_{ins}!}{{N_{ins}}^{N_{ins}}}$. It remains for us to compute a lower bound for p_m. To this end, we form a graph on the set of query nodes, in which for every pair q, q' of query nodes, we add the edge (q,q') if there exists a network vertex v that is homologous to both. We then partition the query vertices into connected components $Q_1, Q_2, \ldots, Q_{k'}$, and use the following bound: $p_m \geq \prod_{u=1}^{k'} \frac{|Q_u|!}{|Q_u|^{|Q_u|}}$. We expect p_m to be high since often query nodes are homologous to a single vertex. When the probability of success with restricted coloring is greater than the probability of success with the standard color coding (i.e., $\frac{(k+N_{ins})!}{(k+N_{ins})^{k+N_{ins}}}$), we use this procedure, and otherwise we use the standard color coding.

Parameter Setting. QNet involves several parameters controlling sequence similarity, insertion/deletion penalties, and the relative weights of edge- and node-terms. The current settings are as follows: we used blastp with an E-value threshold of 10^{-7} to compute sequence similarity, and set $h(q, v) = -log(\texttt{E-value})$. Interaction reliabilities $p(u, v)$ are assigned using a logistic regression scheme based on the experimental evidences for the interactions, as described in [18]. We use $w(u, v) = c \cdot r(u, v)$, where c is chosen to ensure the same scale for the reliability and homology values. We allow at most two insertions and two deletions per query, i.e., $N_{ins} = N_{del} = 2$. Indel penalties are set to $\delta_d = \delta_i = -100$. We empirically tested a range of penalties by querying perturbations of subtrees in the yeast network (see Section 5.1). A small set of queries were examined and the results did not change over the range as long as the net influence of a deletion or insertion were kept negative. In all runs reported below, the number of color coding iterations was set to ensure success probability ≥ 0.99.

5 Experimental Results

To evaluate the performance of QNet we measure its running time and accuracy under various configurations. We start by applying QNet to query a set of synthetic trees in the PPI network of yeast, measuring its running time and accuracy. Next, we show examples of querying known yeast and human signal transduction pathways in the PPI network of fly. Finally, we apply QNet to query known yeast complexes in fly.

Protein-protein interaction data for yeast *S. cerevisiae* and fly *D. melanogaster* were obtained from the Database of Interacting Proteins (DIP) [20] (April 2005 download). The fly data was complemented by PPI interactions from [19] and by genetic interactions from FlyGRID (see also [18]). Altogether, the yeast network consists of 4,738 proteins and 15,147 interactions, and the fly network consists of 7,481 proteins and 26,201 interactions.

5.1 Synthetic Query Trees

To measure the running time and estimate the accuracy of QNet, we applied it to query the PPI network of yeast with a set of synthetic query trees. This set consists of 20 randomly chosen subtrees of sizes ranging from $k = 5$ to $k = 9$ from the yeast PPI network. Each query tree was perturbed with up to 2 node insertions and deletions, and by a pre-specified amount of point mutations in its proteins' sequences of average length ~ 500. QNet was applied to identify a match for each query tree.

The running time measurements were performed on a standard PC (2GHz, 1Gb). We find that the running time of QNet is a few seconds in all cases, reaching an average of 11 seconds for the largest tree queries with 9 nodes (Table 1). To measure the improvement in running time introduced by the restricted color coding heuristic, we applied QNet also without this heuristic. We find that restricted color coding significantly reduces the number of iterations required to

Table 1. Number of color coding iterations and timing statistics for QNet. The last two columns show the average time per query. The algorithm's parameters are set as follows: $N_{ins} = 2$, $N_{del} = 2$, and the probability of success is set to 0.99.

	#Iterations		Avg. time (sec.)	
Query size (k)	Standard color coding	Restricted color coding	Standard color coding	Restricted color coding
5	752	603	1.71	1.58
6	1916	917	6.36	4.73
7	4916	1282	20.46	6.24
8	12690	1669	61.17	9.08
9	32916	2061	173.88	11.03
10	85720	2509	1463	21.74
11	223990	2987	5501	41.39
12	1891868	4623	50455	97.93

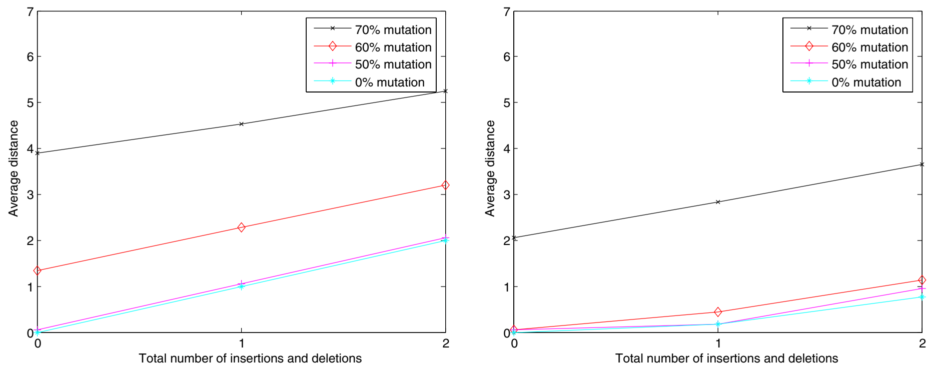
identify the optimal match, while the running time of each iteration remains similar. Overall, restricted color coding reduces the running time by an order of magnitude on average (Table 1). The running time of the algorithm is significantly affected by the number of insertions allowed. If no insertions are allowed, the average number of iterations required for queries of size 9 is less than 100. When increasing the number of allowed insertions to above 2, the restricted color coding heuristic becomes less effective (data not shown).

To evaluate the accuracy of the matched trees, we computed the symmetric difference between the protein set of a query and its match, termed their *distance* herein. The results show that when perturbing protein sequences in up to 60% of the residues, the average distance between the matched tree and the original tree is lower than 1 (Figure 3b). Moreover, we compared the accuracy of matches obtained by QNet to matches that are based only on best BLAST hits. We found that matches obtained by QNet are markedly more accurate than purely sequence-based matches, showing that the topology of the query tree carries important signal (Figure 3a). Evidently, the advantage of QNet over a sequence-based approach becomes more pronounced when the mutation rate increases.

5.2 Cross-Species Comparison of MAPK Pathways

The mitogen-activated protein kinase (MAPK) pathways are a collection of related signal transduction pathways, which play a critical role in mediating the cellular response to various toxic stresses [5]. The pathways are known to be conserved across species and, hence, serve as controlled tests to QNet.

We queried MAPK pathways from the KEGG database [12] in the PPI network of fly. The first pathway is a classical human MAPK pathway involved in cell proliferation and differentiation. Querying this pathway in fly resulted in detecting a known MAPK pathway involved in dorsal pattern formation (Figure 4a). Specifically, 6 out of the 8 matched proteins in the target are members of the known MAPK pathway in fly. Similar results were obtained by querying the yeast MAPK pathways from KEGG against the fly network. As an example, the top output for



(a) (b)

Fig. 3. The average distance of the matched tree from the original tree is plotted against the total number of insertions and deletions introduced to the query for 4 different mutation levels. (a) Performance of a sequence-based approach. (b) Performance of QNet.

the starvation response pathway query (Figure 4b) is a fly MAPK pathway with a putative MAPK cascade (fray,Dsor1,rl), which includes the GTPases Cdc42, Ras64b that are homologous to the two GTPases in the query. These results support the fidelity of QNet.

5.3 Cross-Species Comparison of Protein Complexes

As a large-scale validation of QNet we systematically queried known yeast protein complexes, obtained from the MIPS database [16,9], in the fly network, and tested the biological plausibility of the identified matches. We included all hand curated complexes in MIPS, which are considered a reliable data source, excluding complexes that were identified via high throughput measurements (category 550 in MIPS). Overall, we considered 94 complexes consisting of at least 4 proteins each. As MIPS does not contain information on the topology of the complexes, we mapped each complex to the yeast network and used the induced subnetworks as queries. More accurately, for each complex, we extracted an average of 40 random query trees of size in the range $3 - 8$ from its induced subnetwork. We applied QNet to systematically query all of the induced query trees in fly. The resulting query matches were used to construct a *consensus match*, consisting of all proteins that appeared in at least half of the matches.

The biological plausibility of an obtained consensus matches was tested based on functional enrichment of their member proteins w.r.t. the fly gene ontology (GO) process annotation [2]. Specifically, let $n(t)$ denote the number of genes in the consensus match that are annotated with term t. We compute the probability $p(t)$ of obtaining a random set of genes, of the same size as the original pathway, with at least $n(t)$ genes annotated with term t, assuming a hypergeometric

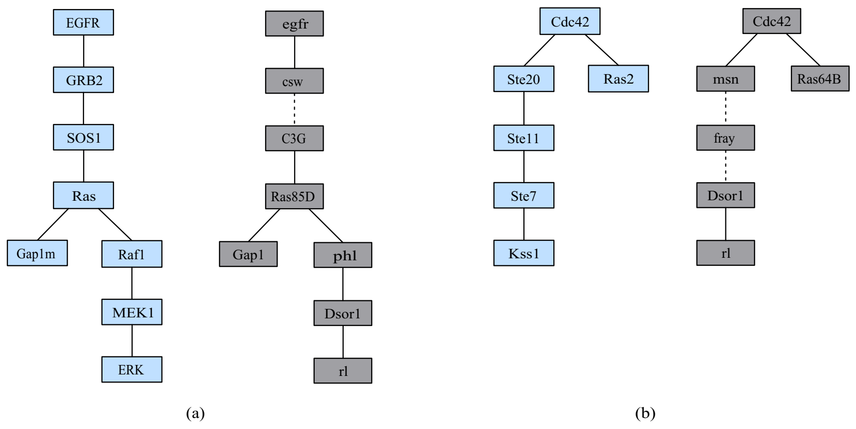


Fig. 4. Querying the fly network using (a) a human MAPK pathway, and (b) a yeast MAPK pathway induced by starvation, taken from the KEGG database [12]. Matched nodes appear on the same horizontal line. A dotted edge represents inserted proteins (not shown).

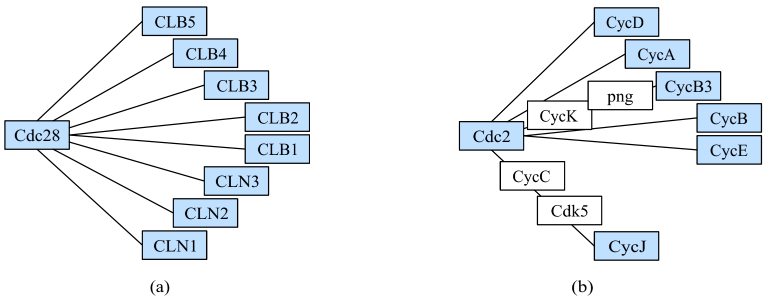


Fig. 5. (a) The MIPS Cdc28p complex. (b) The consensus match in fly. Matched nodes appear on the same horizontal line. Inserted proteins appear in white.

distribution. Having found a term t_0 with minimal probability $p(t_0)$, we compute a p-value for the enrichment under term t_0 by comparing $p(t_0)$ with similar values computed for 10,000 random sets of genes. The latter p-values are further corrected for multiple match testing via the false discovery rate procedure [3].

36 of the yeast complexes resulted in a consensus match with more than one protein in fly. We find that 72% of these consensus matches are significantly functionally enriched ($p < 0.05$). For comparison, we computed the functional enrichment of randomly chosen trees from the fly PPI network that have the same distribution of sizes and interactions scores as the consensus matches. We find that only 17% of the random trees are functionally enriched, and that the mean enrichment p-values is significantly lower for the true consensus matches (Wilcoxon rank test p-value$< 6.5e-9$).

Figure 5 illustrates the result of querying the Cdc28p complex. This complex is composed of cyclin-dependent kinases involved in regulating the cell cycle in yeast. The consensus match obtained in fly consists solely of cyclin-dependent kinases and significantly overlaps the cyclin-dependent protein kinase holoenzyme complex (GO:0000307).

6 Conclusion

Data sets of protein-protein interactions are increasingly common, and will continue to increase in number and complexity. In this paper, we address the problem of searching such data for specific pathways of interest. We provide efficient algorithms for querying trees and graphs of bounded treewidth within PPI networks. We implement the tree query algorithm, QNet, and demonstrate its efficiency and accuracy. QNet can handle queries of up to 9 proteins in seconds on current networks, and is shown to outperform sequence-based homology searches. More importantly, we use QNet to perform a large scale cross-species comparison of protein complexes, by querying known yeast complexes in the fly network. This comparison points to strong conservation between the two species.

While our work has helped in clarifying some algorithmic questions regarding efficient querying of biological networks, and has shown promising results in practice, it leaves many aspects open for future research. One important direction is the development of appropriate score functions to better identify conserved pathways. Research in this direction could gain from probabilistic models of network evolution [4,10]. A second important direction is the application of the methods developed here to queries of more general structure. This entails both the implementation and testing of a tool for querying bounded treewidth graphs, and the use of such a tool for querying arbitrary structures, perhaps in a way similar to that presented in Section 5.2.

Acknowledgments

We thank Eyal Kaplan for critical reading of the manuscript. B.D. and V.B. were supported in part by the NSF grant CCF-0425926, T.S. was supported by the Tauber fund, and R.S. was supported by an Alon fellowship. This research was supported in part by a research grant from the Ministry of Science and Technology, Israel.

References

1. N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM*, 42(4):844–856, 1995.
2. M. Ashburner et al. The gene onthology consortium. gene onthology: Toll for the unification of biology. *Nature Genetics*, 25:25–29, 2000.
3. Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc. B*, 57:289–300, 1995.

4. J. Berg, M. Lassig, and A. Wagner. Structure and evolution of protein interaction networks: A statistical model for link dynamics and gene duplications. *Bio. Med. Center Evolutionary Biology*, 4:51, 2001.
5. P. Dent, A. Yacoub, P. B. Fisher, M. P. Hagan, and S. Grant. Mapk pathways in radiation responses. *Oncogene*, 22(37):5885–5896, Sep 2003.
6. S. F and Z. R. Identifying active transcription factors and kinases from expression data using pathway queries. *Bioinformatics*, 21(Suppl 2):ii115–ii122, Sep 2005.
7. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Co., San Francisco, 1979.
8. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W. H. Freeman and Company, 1979.
9. U. Guldener, M. Munsterkotter, M. Oesterheld, P. Pagel, A. Ruepp, H.-W. Mewes, and V. Stumpflen. MPact: the MIPS protein interaction resource on yeast. *Nucleic Acids Res*, 34(Database issue):436–441, Jan 2006.
10. E. Hirsh and R. Sharan. Identification of conserved protein complexes based on a model of protein network evolution. In *Fifth European Conference on Computational Biology (ECCB'06)*, 2006. To appear.
11. T. Ito, T. Chiba, and M. Yoshida. Exploring the yeast protein interactome using comprehensive two-hybrid projects. *Trends Biotechnology*, 19:23–27, 2001.
12. M. Kanehisa, S. Goto, S. Kawashima, Y. Okuno, and M. Hattori. The KEGG resource for deciphering the genome. *Nucleic Acids Res*, 32(Database issue):277–280, Jan 2004.
13. B. P. Kelley, R. Sharan, R. M. Karp, T. Sittler, D. E. Root, B. R. Stockwell, and T. Ideker. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc Natl Acad Sci U S A*, 100(20):11394–9, 2003.
14. T. Kloks. *Treewidth: computations and approximations.* Springer-Verlag, 1994.
15. M. Mann, R. Hendrickson, and A. Pandey. Analysis ures of proteins and proteomes by mass spectrometry. *Annu. Rev. Biochem*, 70:437–473, 2001.
16. H. W. Mewes, D. Frishman, K. F. Mayer, M. Munsterkotter, O. Noubibou, P. Pagel, T. Rattei, M. Oesterheld, A. Ruepp, and V. Stumpflen. MIPS: analysis and annotation of proteins from whole genomes in 2005. *Nucleic Acids Res*, 34(Database issue):169–172, Jan 2006.
17. R. Y. Pinter, O. Rokhlenko, E. Yeger-Lotem, and M. Ziv-Ukelson. Alignment of metabolic pathways. *Bioinformatics*, 21(16):3401–8, 2005.
18. T. Shlomi, D. Segal, E. Ruppin, and R. Sharan. QPath: A Method for Querying Pathways in a Protein-Protein Interaction Network. *BMC Bioinformatics*, 7(199), 2006.
19. C. A. Stanyon, G. Liu, B. A. Mangiola, N. Patel, L. Giot, B. Kuang, H. Zhang, J. Zhong, and J. Finley, R. L. A Drosophila protein-interaction map centered on cell-cycle regulators. *Genome Biol*, 5(12):R96, 2004.
20. I. Xenarios, D. W. Rice, L. Salwinski, M. K. Baron, E. M. Marcotte, and D. Eisenberg. DIP: the database of interacting proteins. *Nucleic Acids Res*, 28(1):289–91, 2000.

Appendix: A General Alignment Algorithm for Bounded Treewidth Queries

In Section 3.2 we described an algorithm for identifying optimal simple alignments of a bounded treewidth query graph. To generalize the algorithm to support deletions, we modify the mapping σ to allow mapping to '0'. To support insertions, we allow σ to map connected query nodes to non-connected graph vertices, and use additional N_{ins} color (as in Section 3.1).

Given the new definition of σ, the node term is modified as follows:

$$W^S(i,\sigma) = \delta_d |\{u \in X_i^N : \sigma(u) = 0\}| + \sum_{u \in X_i^N, \sigma(u) \neq 0} h(u, \sigma(u))$$

The edge term is more problematic as it depends on the subset of colors used for insertions, and requires some preprocessing. For a pair of vertices $u, v \in V$ and a set of colors $S \subseteq C - \{c(u), c(v)\}$, we denote by $W_P(u, v, S)$ the maximum weight of a path between u and v that visits the colors in S. Given a set of vertex pairs $R = R(l) = \{(r_1^1, r_2^2), \ldots, (r_l^1, r_l^2)\}$, we define $W_P(R, S)$ as the maximum weight of $|R|$ simple paths between all vertex pairs that visit distinct colors from S:

$$W_P(R,S) = \max_{\substack{S^1, S^2, \ldots S^q \\ \biguplus S^l = S}} \sum_{l=1}^{q} W_P(r_l^1, r_l^2, S^l)$$

In order to compute $W_P(R, S)$ efficiently, we use the following recurrence:

$$W_P(R(l), S) = \max_{S' \subset S} \left[W_P((r_i^1, r_i^2), S') + W_P(R(l-1), S - S') \right]$$

Define $E_i(\sigma)$ as the set of graph vertex pairs that are mapped from non-active edges in super-node i:

$$E_i(\sigma) = \{(u, v) \in E : (u', v') \in E_i^N, \sigma(u') = u, \sigma(v') = v\}$$

The edge term for super-node i under the mapping σ and colors S, is:

$$W^E(i, \sigma, S) = W_P(E_i(\sigma), S)$$

Finally, we modify the main recursion as follows:

$$W(i,\sigma,S) = \max_{\substack{S_1 \uplus S_2 = S, \\ S_1' \subset S_1, S_2' \subset S_2, \\ \sigma_1, \sigma_2}} \sum_{j=1}^{2} \left[W(i_j, \sigma_j, S_j - S_j') + W^S(i_j, \sigma_j) + W^E(i_j, \sigma_j, S_j') \right]$$

To compute the running time of the preprocessing stage, note that $W_P((u, v), S))$ can be pre-computed for all S in $O(n^2 2^k)$ time. Therefore, $W_P(E_i(\sigma), S)$ can be pre-computed in $2^{O(k)} n^{t+1}$ time, and hence the total running time is $2^{O(k)} n^{t+1}$.

Pairwise Global Alignment of Protein Interaction Networks by Matching Neighborhood Topology

Rohit Singh[1], Jinbo Xu[2], and Bonnie Berger[1,⋆]

[1] Computer Science and AI Lab., Massachusetts Institute of Technology
rsingh@mit.edu, bab@mit.edu
[2] Toyota Technological Institute, Chicago, USA
j3xu@tti-c.org

Abstract. We describe an algorithm, IsoRank, for global alignment of two protein-protein interaction (PPI) networks. IsoRank aims to maximize the overall match between the two networks; in contrast, much of previous work has focused on the local alignment problem— identifying many possible alignments, each corresponding to a local region of similarity. IsoRank is guided by the intuition that a protein should be matched with a protein in the other network if and only if the neighbors of the two proteins can also be well matched. We encode this intuition as an eigenvalue problem, in a manner analogous to Google's PageRank method. We use IsoRank to compute the first known global alignment between the *S. cerevisiae* and *D. melanogaster* PPI networks. The common subgraph has 1420 edges and describes conserved functional components between the two species. Comparisons of our results with those of a well-known algorithm for local network alignment indicate that the globally optimized alignment resolves ambiguity introduced by multiple local alignments. Finally, we interpret the results of global alignment to identify functional orthologs between yeast and fly; our functional ortholog prediction method is much simpler than a recently proposed approach and yet provides results that are more comprehensive.

1 Introduction

A fundamental goal of biology is to understand the cell as a system of interacting components and, in particular, how proteins in the cell interact with each other. Towards this goal, high-throughput experimental techniques (e.g., yeast two-hybrid [12,14] and co-immunoprecipitation [11]) to discover protein-protein interactions (PPIs) are being used. These techniques have also been supplemented by promising new computational approaches [27,24,23,26,17,29,9] to PPI prediction, resulting in an explosive growth in available PPI data. A powerful way of representing and analyzing all this data is the PPI network: a network where each node corresponds to a protein and an edge indicates a direct physical

⋆ Corresponding author. Also with the Department of Mathematics, MIT.

interaction between the proteins. Computational analyses of these networks has already yielded valuable insights: the scale-free character of these networks and the disproportionate importance of "hub" proteins [30]; the combination of these networks with gene expression data to discern some of the dynamic character of the cell [8]; the use of PPI networks for inferring biological function [20], etc.

As more PPI data becomes available, comparative analysis of PPI networks (across species) is proving to be a valuable tool. Such analysis is similar in spirit to traditional sequence-based comparative genomic analyses; it also promises commensurate insights. Such an analysis can identify conserved functional components across species [15]. As a phylogenetic tool, it offers a function-oriented perspective that complements traditional sequence-based methods. It also facilitates annotation transfer between species. Indeed, Bandyopadhyay *et al.* [3] have demonstrated that the use of PPI networks in computing orthologs produces orthology mappings that better conserve protein function across species.

In this paper, we explore a new approach to comparative analysis of PPI networks. Specifically, we consider the problem of finding the optimal *global* alignment between two PPI networks, aiming to find a correspondence between nodes and edges of the input networks that maximizes the overall match between the two networks. For this problem, we propose a novel pairwise global alignment algorithm, IsoRank.

1.1 Contributions

In this paper, we draw attention to the global network alignment problem and its biological importance (as distinct from local network alignment, see Sec. 1.2). We propose IsoRank— an algorithm for pairwise global network alignment of PPI networks; to the best of our knowledge, it is the first such algorithm of its kind. It simultaneously uses both PPI network data and sequence similarity data to compute the alignment, the relative weights of the two data sources being a free parameter (existing *local* network alignment algorithms have typically not provided such direct control over the relative weights). The algorithm is intuitive: a node i in G_1 is mapped to a node j in G_2 if the neighborhood topologies of i and j are similar, i.e., the neighbors of i can be well-mapped to the neighbors of j. This approach has parallels to Google's PageRank technique; like the latter, we formalize our intuition as an eigenvalue problem (see Sec. 3). IsoRank is, by design, tolerant to errors in the input (e.g., missing or spurious edges) and takes advantage of edge confidence scores as well as other biological signals (e.g. sequence similarity scores), when available. We use the algorithm to compute a global alignment of the *S. cerevisiae* and *D. melanogaster* PPI networks and describe the conserved subgraph (possibly disconnected) between them. The conserved subgraph immediately suggests functions for some hitherto unannotated proteins. It also suggests sets of functional orthologs between the two species; these predictions are consistent with those of Bandyopadhyay *et al.* [3], and, in some cases, are more precise and accurate.

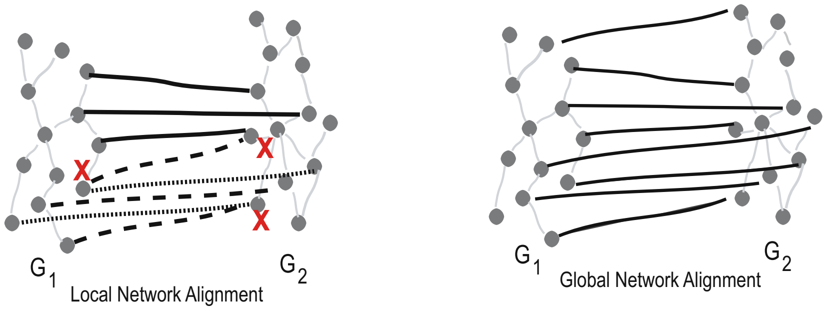


Fig. 1. Cartoon comparing global and local network alignments: The local network alignment between G_1 and G_2 specifies three different alignments; the mappings for each are marked by a different kind of line (solid, dashed, dotted). Each alignment describes a small common subgraph. Local alignments need not be consistent in their mapping— the points marked with 'X' each have ambiguous/inconsistent mappings under different alignments. In global network alignment, the maximum common subgraph is desired and it is required that the mapping for a node be unambiguous. In both cases, there are 'gap' nodes for which no mappings could be predicted (here, the nodes with no incident black edges are such nodes).

1.2 Related Work: The Distinction Between Local and Global Alignment

The network alignment problem has been formulated previously [6,18,15], with some variations. To place our work in that context, we first distinguish between global and local network alignment.

Each input network can be represented as an undirected graph $G = (V, E)$ where V is the set of nodes and E is the set of edges. Furthermore, G may be a weighted graph, i.e., a confidence measure $w(e)$ may be associated with each edge e in E. In this paper, we consider graphs of arbitrary structure; when graphs have specific structures (e.g., trees) other efficient methods are available [13,28]. The goal in network alignment is to identify one or multiple possible mappings between the nodes of the input networks and, for each mapping, the corresponding set of conserved edges. Mappings may be partial, i.e., they need not be defined for all the nodes in the networks. Each mapping implies a common subgraph between the two networks: when protein a_1 from network G_1 is mapped to protein a_2 from network G_2, then a_1 and a_2 refer to the same node in the common subgraph; the edges in the common subgraph correspond to the conserved edges. Based on the kind of mapping(s) sought, we distinguish between the local and global network alignment (in analogy with sequence alignment).

Local Network Alignment (LNA): The goal in LNA is to find local regions of isomorphism (i.e. same graph structure) between the input networks, each region implying a mapping independently of others. Many independent, high-scoring local alignments are usually possible between two input networks; in fact, the corresponding local alignments need not even be mutually consistent (i.e., a protein might be mapped differently under each, see Fig 1). This may not be undesirable (e.g., it may indicate gene duplication); however, in some cases LNA algorithms offer implausibly numerous matches for a single protein. The motivations behind local sequence alignment and local network alignment are analogous— the former is often used to find conserved sequence motifs; the latter for finding conserved functional components (e.g., pathways, complexes, etc.).

Previous work on PPI network alignment has almost exclusively focused on this problem: the pioneering work of Kelley *et al.* [6] described how BLAST similarity scores and PPI network information could be used to identify conserved functional motifs. Koyuturk *et al.* [18] proposed another method, motivated by biological models of duplication and deletion. Recently, Flannick *et al.* [15] proposed a new approach, using modules of proteins to infer the alignment. The approach is efficient and is the first LNA method to align multiple species simultaneously. In contrast to these methods, our work targets the global network alignment problem (see Footnote 1).

Global Network Alignment (GNA): The aim in GNA is to find the best overall alignment between the input networks. A GNA algorithm must define a single mapping across all parts of the input (see Fig 1), even if it were locally suboptimal in some regions of the networks. In contrast, an LNA algorithm has the freedom to choose the locally optimal mapping for each local region of similarity, even if this results in overlapping — and mutually inconsistent — local alignment. We avoid this in GNA by requiring that for any global alignment to be valid the corresponding mapping be *comprehensive*: each node in an input network is either matched to some node in the other network or explicitly marked as a gap node (i.e., with no match in the other network). Our goal in GNA then is to find a comprehensive mapping such that the size of the corresponding common subgraph is maximized. The motivations behind global sequence alignment and GNA are again analogous: the former is often used for comparing genomic sequences to understand variations between species; the latter may be used to compare interactomes, and to understand cross-species variations. Also, the GNA problem is related to the detection of functional orthologs, as we discuss in Sec. 4.

The GNA problem, as we describe it here, is the focus of this paper. It has previously received little attention in the literature; much of existing work has focused on the LNA problem[1]. One can imagine using results of an LNA to estimate a global alignment: use LNA methods to compute possible matches for

[1] We note that in some previous works on network alignment, the distinction between "global" and "local" network alignment has centered on the relative input sizes for each. There, the term "global network alignment" is used when the input consists of roughly equal-sized networks (e.g., two species-wide networks) while "local network alignment" is used when one input is a small query network and the other is a large species-wide network. In both instances, however, the output consists of multiple local subgraphs (and corresponding local alignments). As such, we believe that both these instances are best characterized as local network alignments, regardless of input sizes.

each protein. Then, for each protein select the mapping best supported overall by the alignment results. Banydopadhyay *et al.* have used a similar approach for functional ortholog detection. Unfortunately, this approach is somewhat complex and, more importantly, ignores inconsistencies across local alignments so that the node matches in the final alignment might not even be mutually consistent. Instead, we propose a simpler, yet powerful algorithm.

2 Problem Formulation

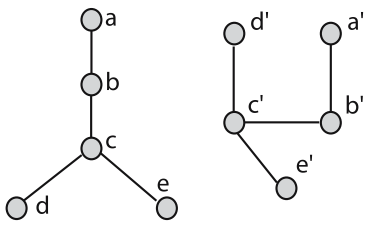
The input to the algorithm consists of two PPI networks G_1 and G_2. Each edge e may have an associated edge weight $w(e)$ $(0 < w(e) \leq 1)$. In addition, other measures of similarity between the nodes may be available. In this paper, we use BLAST similarity scores, but additional measures (e.g., synteny-based scoring, functional similarity) can be incorporated.

The desired output, given only PPI network data, is the maximum common subgraph (MCS) between G_1 and G_2 (i.e., the largest graph that is isomorphic to subgraphs of both) and the corresponding node-mapping such that each node is mapped to at most one node in the other network. Nodes not mapped to any other node are referred to as gap nodes. MCS is an NP-complete problem and thus approximate solutions, especially for the large-sized PPI networks, are essential. Also, when incorporating sequence data, the global alignment problem is no longer a pure MCS problem. To address these issues, we formulate an eigenvalue problem that approximates the desired objective.

The "at most one match per node" constraint is motivated by analogy with two-way global *sequence* alignment where any position in a sequence can be matched to at most one position in the other sequence. When performing LNA, Kelley *et al.* [6] have imposed a similar constraint. The benefits of imposing this constraint are: (1) we simplify the alignment problem, and (2) we can *unambiguously* identify the closest functional equivalent of a protein in the other species; this is related to the discovery of functional orthologs (see Sec. 4). On the other hand, in instances of gene duplication across species this constraint requires that a protein cannot be matched to multiple proteins in another species. In future work, we plan to relax this constraint.

3 Algorithm: IsoRank

The key problem that our algorithm (IsoRank) targets is identifying the node mappings between the input networks; given such a mapping, the set of conserved edges can be easily computed. The algorithm works in two stages. It first associates a score with each possible match between nodes of the two networks. Let R_{ij} be the score for the protein pair (i, j) where i is from network G_1 and j is from network G_2. Given network and sequence data, we construct an eigenvalue problem and solve it to compute R (the vector of all R_{ij}s). The second stage constructs the mapping for the GNA by extracting from R high-scoring, pairwise, mutually-consistent matches.



R

	a'	b'	c'	d'	e'
a	0.0312		0.0937		
b		0.1250		0.0625	0.0625
c	0.0937		0.2812		
d		0.0625		0.0312	0.0312
e		0.0625		0.0312	0.0312

$R_{aa'} = \frac{1}{4} R_{bb'}$ $R_{bb'} = \frac{1}{3} R_{ac'} + \frac{1}{3} R_{a'c} + R_{aa'} + \frac{1}{9} R_{cc'}$ $R_{dd'} = \frac{1}{9} R_{cc'}$

$R_{cc'} = \frac{1}{4} R_{bb'} + \frac{1}{2} R_{be'} + \frac{1}{2} R_{bd'} + \frac{1}{2} R_{eb'} + \frac{1}{2} R_{db'} + R_{ee'} + R_{ed'} + R_{de'} + R_{dd'}$

Fig. 2. Intuition behind the algorithm: Here we show, for a pair of small, isomorphic graphs how the vector of pairwise scores (R) is computed. For each possible pairing (i, j) between nodes of the two graphs, we compute the score R_{ij}. The scores are constrained to depend on the scores from the neighborhood as described by Eqn. 1. Only a partial set of constraints is shown here. The scores R_{ij} are computed by starting with random values for R_{ij} and using the methods described below to find values that satisfy these constraints; here we show the vector R reshaped as a table for ease of viewing (empty cells indicate a value of zero). The second stage of our algorithm uses R to extract likely matches. One strategy could: choose the highest-scoring pair, output it, remove the corresponding row and column from the table, and repeat. This strategy will return the correct mapping: $\{(c, c'), (b, b'), (a, a'), (d, d'), (e, e')\}$. The $\{d, e\} \rightarrow \{d', e'\}$ mapping is ambiguous; using sequence information, such ambiguities can be resolved.

Computing R (setting up the constraints): To compute R_{ij} we pursue the intuition that (i, j) is a good match if i and j's respective neighbors also match well with each other. More precisely, we require the following equality to hold for all possible pairs (i, j):

$$R_{ij} = \sum_{u \in N(i)} \sum_{v \in N(j)} \frac{1}{|N(u)||N(v)|} R_{uv} \quad i \in V_1,\ j \in V_2 \tag{1}$$

where $N(a)$ is the set of neighbors of node a; $|N(a)|$ is the size of this set; and V_1 and V_2 are the sets of nodes in networks G_1 and G_2, respectively.

These equations require that the score R_{ij} for any match (i, j) be equal to the total support provided to it by each of the $|N(i)||N(j)|$ possible matches between the neighbors of i and j. In return, each match (u, v) must distribute back its entire score R_{uv} equally among the $|N(u)||N(v)|$ possible matches between its neighbors. We note that these equations also capture non-local influences on R_{ij}: the score R_{ij} depends on the score of neighbors of i and j and the latter, in turn, depend on the neighbors of the neighbors and so on. The extension to the weighted-graph case is intuitive: the support offered to neighbors is now in proportion to the edge weights:

$$R_{ij} = \sum_{u \in N(i)} \sum_{v \in N(j)} \frac{w(i,u)w(j,v)}{\sum_{r \in N(u)} w(r,u) \sum_{q \in N(v)} w(q,v)} R_{uv} \quad i \in V_1,\ j \in V_2 \tag{2}$$

Clearly, Eqn. 1 is a special case of Eqn. 2 when all the edge weights are 1. We can rewrite Eqn. 1 in matrix form (Eqn. 2 can be similarly rewritten):

$$\begin{aligned} R &= AR \\ A[i,j][u,v] &= \begin{cases} \frac{1}{|N(u)||N(v)|} & \text{if } (i,u) \in E_1 \text{ and } (j,v) \in E_2 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{3}$$

where A is a $|V_1||V_2| \times |V_1||V_2|$ matrix and $A[i,j][u,v]$ refers to the entry at the row (i,j) and column (u,v) (the row and column are doubly-indexed).

Another interpretation of the above equations is that they describe a random walk on the product graph of $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. We define $G^* = (V^*, E^*)$ where $V^* = V_1 \times V_2$ and $E^* = \{(\,(i,j),(u,v)\,) \mid (i,u) \in E_1,\ (j,v) \in E_2\}$. Also, if G_1 and G_2 are weighted, so is G^*: $w(\,(i,j),(u,v)\,) = w(i,u)w(j,v)$. We now specify a random walk among the nodes of G^*: from any node we can move to one of its neighbors, with a probability proportional to the edge weight:

$$P(s_t = (i,j) \mid s_{t-1} = (u,v)) = \frac{w(i,u)w(j,v)}{\sum_{r \in N(u)} w(r,u) \sum_{q \in N(v)} w(q,v)} \tag{4}$$

where s_t is the node occupied at time t. Eqns. 1, 2 and 3 can now be interpreted as defining R to be the stationary distribution of this random walk (its transition matrix is A). Thus, a high R_{ij} implies that the node (i,j) of G^* has a high probability of being occupied in the stationary distribution.

The vector R is determined by finding a non-trivial solution to these equations (a trivial solution is to set all R_{ij}s to zero). In Fig 2, we illustrate, on a pair of small graphs, how the equations capture the graph topology; their solution also confirms our intuition: node pairs that match well have higher R_{ij} scores.

Computing R (solving the constraints): In general, to solve the above equations, we observe that these equations describe an eigenvalue problem (see Eqn. 3). The value of R we are interested in is the principal eigenvector of A. Note that A is a stochastic matrix (i.e., each of its columns sums to 1) so that the principal eigenvalue is 1. Also, for numerical stability purposes we require that R be normalized, i.e., $|R|_1 = 1$. In the case of biological networks, A is typically a very large matrix (about $10^8 \times 10^8$ for fly-vs.-yeast GNA); however, A and R are both very sparse, so R can be efficiently computed by iterative techniques. We use the *power method* [16], an iterative technique often used for large eigenvalue problems. The power method repeatedly updates R as per the update rule: $R(k+1) \leftarrow AR(k)/|AR(k)|$, where $R(k)$ is the value of the vector R in the k-th iteration and has unit norm. In case of a stochastic matrix (like A), the power method will provably converge to the principal eigenvector; the convergence can be sped up significantly by a judicious choice of the initial value $R(0)$ [16]. As we describe shortly, a good initial value $R(0)$ is often available in our case.

The incorporation of other information, e.g. BLAST scores, into this model is straightforward. Let B_{ij} denote the score between i and j; for instance, B_{ij} can be the Bit-Score of the BLAST alignment between sequences i and j. B_{ij}s need not even be numeric— they can be binary. Let B be the vector of B_{ij}s. We first normalize B: $E = B/|B|$. The eigenvalue equation is then modified to

$$R = \alpha AR + (1-\alpha)E \quad \text{where} \quad 0 \le \alpha \le 1. \tag{5}$$

Eqn. 5 is solved by similar techniques as Eqn. 3. Also, node matches based purely on sequence similarity are an approximation to the node mappings desired; hence, the vector E is a good choice for the initial value $R(0)$ in the power method. We emphasize that this choice of starting value does not change the final value of R— it just speeds up the computation.

In this computation, α controls the weight of the network data (relative to sequence data), e.g., $\alpha = 0$ implies no network data will be used, while $\alpha = 1$ indicates only network data will be used. Tuning α allows us to analyze the relative importance of PPI data in finding the optimal alignment.

Extracting the mapping from R: Once R has been computed, we extract the node mappings from it. An appealing approach is to extract the set of mutually-consistent, pairwise matches (p, q) such that the sum of their scores is maximized. The optimal solution can thus be found efficiently by interpreting R as encoding a bipartite graph and finding the maximum-weight bipartite matching [22] for this graph. Each side of the bipartite graph contains all the nodes from one network. The weight of the edge (i, j) is then set to R_{ij}. We compute the maximum-weight matching in this bipartite graph and output the paired nodes. Any remaining unpaired nodes are designated as gap nodes. This algorithm guarantees the set of matches that satisfy our criterion.

While this principled algorithm does give good results, in practice we found that the following greedy algorithm sometimes performs even better: identify the highest score R_{pq} and output the pairing (p, q). Then, remove all scores involving p or q. We then repeat this process until the list is empty. In the bipartite graph, this strategy corresponds to removing, at each step, the maximum weight edge and the incident nodes. In future work, we plan to investigate whether this heuristic's better performance is related to the structure of R.

Once a comprehensive alignment has been computed, the corresponding subgraph in the GNA can be identified relatively easily. For example, if a_1 is aligned to a_2, and b_1 is aligned to b_2, the output subgraph should contain an edge between (a_1, a_2) and (b_1, b_2) if and only if both the input networks contain supporting edges (i.e., (a_1, b_1) in G_1 and (a_2, b_2) in G_2). When edges also have associated weights, formalizing the intuition depends on how the edge weights are being interpreted; for example, we could require that the combined weight be higher than a threshold or that the minimum of the two be greater than a threshold.

4 Results: GNA of Yeast and Fly PPI Networks

We now describe the results of two-way global alignment of the *S. cerevisiae* and *D. melanogaster* PPI networks, the two species with the most available network data. The PPI network data for the species was retrieved from the GRID [4] and DIP [7] databases, and the sequence data was retrieved from Ensembl [2]. The edges in the PPI networks did not have associated weights. We applied IsoRank to this pair of networks, using it to identify the common subgraph.

The common subgraph corresponding to the global alignment between the yeast and fly PPI networks has 1420 edges (where $\alpha = 0.6$; the criterion for choosing α is described later in this section). While this indicates a relatively low overlap between the yeast and fly networks (both the networks have more than 25000 edges each), it is not surprising: firstly, currently available PPI data is known to contain many false-positives, and the number of true interactions in the current networks is expected to be significantly lower [27,25]. Secondly, current PPI data is far from comprehensive; e.g., the fly network has no known PPIs for about 6500 proteins (almost 50% of the genome). As these issues get resolved, we expect the size of the global alignment to grow substantially. Nevertheless, the current global alignment already provides many valuable insights.

The alignment subgraph consists of many disconnected components, with the largest component having 35 edges (Fig. 3). The component's size may seem low but is directly related to the poor connectivity of the alignment subgraph. The poor connectivity is, we believe, because of the poor quality and coverage of current PPI networks; as the datasets improve, so will the connectivity. Even

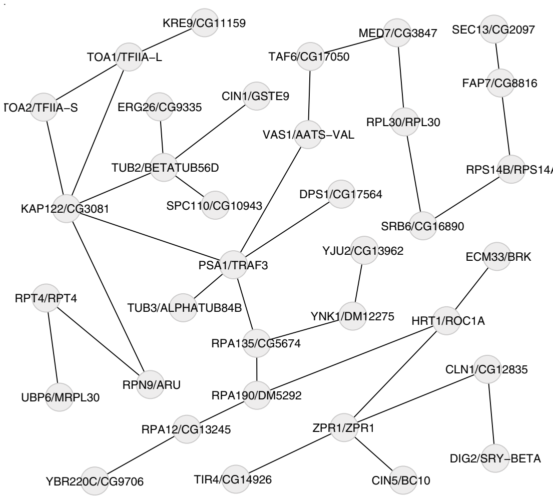
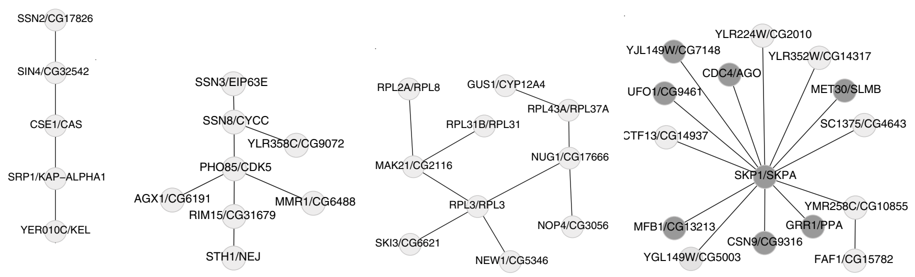


Fig. 3. Largest connected component of the yeast-fly Global Network Alignment: The node labels indicate the corresponding "yeast/fly" proteins (the two separated by a "/"). The proteins in this graph span a variety of functions: metabolic, signaling, transcription etc. For a discussion of this subgraph's size, see text.



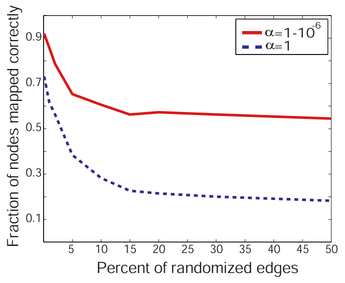
(a) Pathway (b) Kinases (c) Ribosome Complex (d) Ubiquitin Ligase

Fig. 4. Selected subgraphs of the yeast-fly GNA: The node labels indicate the corresponding "yeast/fly" proteins (the two separated by a "/"). The subgraphs span a variety of topologies and are often enriched in specific functions (c) and (d). In (d), the nodes for which at least one of the corresponding proteins is known to be involved in ubiquitin ligase activity are shaded.

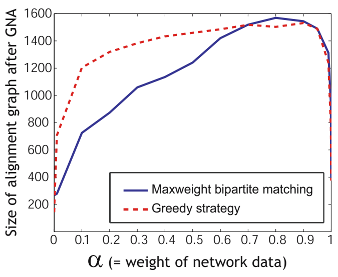
now, however, the subgraph in Fig. 3 is significantly larger than any common subgraph we could identify using Pathblast [6], a LNA method. The longest pathway-like component identified by the latter had 4 nodes, and the largest complex-like component had 16 nodes. Also, the components of the global alignment span various topologies, from linear pathways (Fig. 4(a)) to components corresponding to protein complexes (Fig 4(d)); in contrast, some of the local network alignment methods [6,18] are tailored to search only for specific topologies. We emphasize that our components were discovered simultaneously— they are just subgraphs of the larger alignment graph. Many of our discovered components are de-facto *functional modules* (though not in the sense Flannick *et al.* [15] use the term): they are enriched in proteins involved in a single biological process (e.g., see Fig 4(d)). These functions range from various signaling cascades (Fig. 4(b)) to core cellular functions like ribosomal synthesis and function (Fig. 4(c)), DNA transcription and translation, cell division etc. The preponderance of core cellular functions in the conserved subgraph is not too surprising— it is exactly these mechanisms that are likely to be highly conserved across species.

The global alignment may be used to predict protein function. For example, Fig 4(d) shows a subgraph of the global alignment, most of the proteins in which are involved in SCF ubiquitin ligase activity. Hence, we predict the function of two hitherto-unannotated fly proteins CG7148 and CG13213 as being involved in ubiquitin protein ligase activity. In support of this, we note that the FlyBase database [5] indicates that the involvement of these proteins in ubiquitin ligase activity has been postulated before in the literature. Of course, more sophisticated methods to transfer annotation may perform even better at elucidating function of such proteins [20].

Evaluating the algorithm's error tolerance: Our simulations indicate that the algorithm is tolerant to error in the input (Fig 5(a)); this is valuable since PPI networks have high false positive and false negative rates. To evaluate the



(a) Error-Tolerance



(b) α

Fig. 5. (a) Effect of error on algorithm's performance: We believe the solid (red) curve slightly overestimates the algorithm's performance, while the dashed (blue) curve grossly underestimates it. See the discussion in text below. **(b) Impact of α on the size of the alignment graph.**

algorithm's error-tolerance, we first extracted a 200-node subgraph of the yeast PPI network. We then randomized a fraction p of its edges using the Maslov-Sneppen trick that preserves node degrees [19]: we randomly choose two edges (a, b) and (c, d), remove them, and introduce new edges (a, d) and (c, b). We then computed a GNA between these two graphs, with $\alpha = 1$ and $\alpha = 1 - 10^{-6}$. For each choice of p, we created 5 such randomized graphs and computed the average fraction of nodes that are mapped to themselves in the original graph after a GNA. Using $\alpha = 1$ results in a significant underestimate because there often are multiple possible isomorphism-preserving mappings between two isomorphic graphs (e.g., see Fig 2) and our algorithm— even if working correctly— might choose a mapping that does not preserve node labels. Adding a very small amount of sequence information ($\alpha = 1 - 10^{-6}$) helps avoid this, but also results in a slight overestimate. We believe the true curve (for Fig 5(a)) is closer to the top curve than the bottom one. Clearly, the algorithm makes very few mistakes when the error rate p is low and even for fairly high error rates (20-50%), its performance degrades smoothly and very slowly. When computing the yeast-fly GNA, we assigned a significant weight to sequence information ($\alpha = 0.6$); these simulations suggest our results are quite robust to errors in PPI data.

Evaluating the influence of α: As α increases, so does the importance of network data in the alignment process, for both the greedy strategy and the maximum weight bipartite matching strategy (Fig 5(b)). In line with our expectations, the size of the common subgraph depends on this parameter: $\alpha = 0$ results in a graph with 266 edges, while $\alpha = 0.9$ results in 1544 edges (for the greedy strategy). Intriguingly, as α gets very close to 1, the common graph's size *decreases*. We believe that this discrepancy is an artifact of the current PPI data sets being noisy and covering the interactome only partially, resulting in a relatively small overlap between the yeast and fly PPI networks. Consequently, in absence of any other information a random mapping of nodes between the two networks might satisfy Eqn.1 better than the one corresponding to the "true" alignment. The use of sequence-based scores helps mitigate this, by directing the algorithm towards the true alignment.

When choosing the most appropriate value of the free parameter α, we rejected the choice corresponding to the largest common subgraph size— the input networks are noisy and conserved edges may be simply due to noise; thus, the α leading to the largest-size subgraph may not be a biologically appropriate choice. Instead, for each choice of α, we compared the resulting node mappings to sequence-based ortholog predictions from the Inparanoid database [21] and chose the α ($= 0.6$) that resulted in the greatest overlap with these. While this approach is conservative and might undervalue the network component during the alignment, it also lowers the adverse impact of noise in the PPI data.

The differences between the node pairings found by our algorithm and those from Inparanoid broadly fall into two categories: (1) those corresponding to low R_{ij} values indicating low confidence of our approach in that mapping, and (2) functional orthologs where the use of network data genuinely changes the node mapping. We discuss the latter in more detail later in this section.

Comparing global and local alignment results: Our global alignment results compare favorably to the those of NetworkBlast [1] (an implementation of PathBlast) and sequence-only approaches. We compared the aggregate set of local alignments from NetworkBlast with our global alignment. Each local alignment defines one-to-one matches between some yeast and fly proteins. Many of the matches from our global alignment are seen in these local alignments: of the 701 matched protein-pairs in the former that consist of proteins seen in at least one local alignment, 83% (582) of the pairs are also observed in one or more local alignments. However, there are many overlapping local alignments, resulting in ambiguity and inconsistency: averaged across the entire set of local alignments, a yeast protein is aligned to 5.36 different fly proteins. Sometimes, such ambiguity may be biologically meaningful, e.g., in instances of gene duplication. However, the degree of ambiguity in some of the PathBlast results is clearly implausible. For example, the yeast protein SNF1, a Serine-Threonine Kinase (STK), is matched to 71 different fly proteins. In fact, PathBlast results for many of the yeast STKs are very ambiguous– over the set of 72 yeast proteins annotated as STKs, the average number of matching fly proteins per yeast STK is 29.3. STKs are part of many important signaling pathways, e.g, the MAPK, JNK and AKT cascades. Sequence-only approaches. (e.g. Inparanoid) too have performed poorly at ascertaining the correspondence between yeast and fly STKs: Inparanoid does not predict any fly orthologs for 58 of the 72 yeast STKs. Thus the use of GNA to resolve this ambiguity in correspondence is particularly valuable.

GNA and functional orthologs: In analogy with sequence-based comparative genomics methods [10], we apply IsoRank to the detection of functional orthologs (i.e., sets of proteins that perform the same function in two or more species) by exploiting the strong connection between these two problems: proteins that are aligned together in the global alignment should have similar interaction patterns in their respective species and are thus likely to be functional orthologs. There has been a lot of recent interest in the discovery of functional orthologs (FO). In particular, Bandyopadhyay *et al.* [3] took a fairly complex approach to FO detection between yeast and fly through local network alignment (LNA): first, possible FOs for a protein are short-listed using a sequence-only approach; then, using a probabilistic technique (based on Markov Random Fields) and the results of a LNA of the yeast and fly networks (performed using PathBlast), the probability of each short-listed pair of proteins being true FOs is computed.

Table 1. Interpreting two-way global alignment results as functional orthologs (FOs): Comparison of our results with Bandyopadhyay *et al.*'s results [3]. Our method is often consistent with their results and, moreover, often resolves the ambiguity in their predictions. [1]Our predicted FO for the protein matches Bandyopadhyay *et al.*'s predicted FO, or the most likely FO if their method predicted multiple FOs. [2]Our predicted FO for the protein is one of the likely FOs predicted by Bandyopadhyay *et al.* (but not the most likely one).

Protein (species)	**Predicted Functional Ortholog by Our Method**	**Related Predictions from (Bandyopadhyay et al.)**		**Remarks**
		Yeast/Fly pair	**Prob.**	
Gid8 (yeast)	CG6617	Gid8/CG6617 Gid8/CG18467	76.51% -	Our predictions consistent with Bandyopadhyay et al.[1]
Tpm2 (yeast)	Tm1	Tpm2/Tm1	-	Consistent predictions.[1]
Tpm1 (yeast)	Tm2	Tpm1/Tm2	43.98%	Consistent predictions.[1]
Gpa1 (yeast)	G-oα47a	Gpa1/G-oα47a Gpa1/G-ia65a	41.53% -	Consistent predictions.[1]
Rpl12 (fly)	Rpl12a	Rpl12a/Rpl12 Rpl12b/Rpl12	48.39% -	Consistent predictions.[1]
Btt1 (yeast)	CG11835	Btt1/CG11835 Btt1/Bcd	70.5% 40.86%	Consistent predictions.[1]
CG18617 (fly)	Vph1	Vph1/CG18617 Stv1/CG18617	43.53% 38.44%	Consistent predictions.[1]
Kap104 (fly)	Trn	Kap104/Trn Kap104/CG8219	40.64% 46.78%	Partially consistent predictions.[2]
Act1 (yeast)	Act5c	Act1/Act5c Act1/Act42a Act1/Act87e Act1/Act88f Act/CG10067	39.56% 39.24% 43.53% 40.17% 38.20%	Partially consistent predictions.[2]
Kel2 (yeast)	CG12081	Kel2/CG12081 Kel1/CG12081	- 45.41%	Partially consistent predictions.[2]
Cmd1 (yeast)	Cam	Cmd1/Cam Cmd1/And	35.90% 44.39%	Partially consistent predictions.[2]
Hsc70-4 (fly)	Ssa3	Hsc70-4/Ssa3	-	Partially consistent predictions.[2]

The results of IsoRank compare favorably with Bandyopadhyay *et al.*'s. Our method has the advantage that it guarantees the predicted sets of FOs will be mutually consistent and achieves higher genome coverage— PathBlast's yeast-vs.-fly local alignments cover only 20.56% of the genes covered by our global alignment. In many cases the FO predictions between the two methods are partially or fully consistent (see Table 1), i.e, FOs predicted by our method are also the likely FOs predicted by their method. Furthermore, their method often proposes multiple FOs for a protein, and our method resolves the ambiguity in their results. In a few other cases, predictions of the two methods differ. At least in some such cases, our method's predictions are better supported by evidence. For example, our method predicts *Bic* (in fly) as the FO of *Egd* (in yeast). Bandyopadhyay *et al.*'s method is ambiguous here as *Bcd*, its predicted FO of *Egd*, is also predicted as a FO of *Btt1*. Furthermore, there is experimental evidence that both *Egd* and *Bic* are components of the Nascent Polypeptide-Associated Complex (NAC) in their respective species, lending support to our prediction; in contrast, *Bcd* does not seem to be involved in NAC.

5 Conclusion

In this paper, we focus on the global network alignment problem, and describe an intuitive yet powerful algorithm for computing the global alignment of two PPI networks; in contrast, much of the previous work has been focused on the local alignment problem. Our algorithm, IsoRank, simultaneously uses network and sequence information and is tolerant of noise in the inputs; furthermore, it is easy to control the relative weights of the network and sequence information in the alignment. We use IsoRank to compute a global alignment of the *S. cerevisiae* and *D. melanogaster* PPI networks. The results provide valuable insights about the conserved functional components between the two species. They also allow us to predict functional orthologs between the fly and yeast; the quality of our predictions compare favorably with previous work.

Our algorithm is similar— in spirit— to Google's PageRank algorithm, which ranks web-pages in the order of their "authoritativeness". The intuition behind the two algorithms has a similar flavor: in PageRank, a page has a high score if many pages with high scores link to it. The intuitions are also formalized similarly– by constructing an eigenvalue problem. Our actual algorithm is quite distinct from PageRank: in our case the input is a pair of undirected, weighted graphs and the output is an alignment; PageRank's input is a directed, unweighted graph (where the nodes indicate web-pages and directed edges, hypertext links), and it outputs node rankings.

We have already extended IsoRank to perform global alignment of multiple networks, but this is beyond the scope of this paper. In future work, we plan to improve the algorithm, better characterize its theoretical behavior, and identify other applications for it. Since PPI data is noisy, it might be useful to generate multiple near-optimal alignments and rank them by their significance. Also, the algorithm can be applied to other biological and non-biological data.

It might also be possible to extend such an eigenvalue approach to perform local network alignment; as noted before, the use of an eigenvalue approach removes the restriction of being able to find subgraphs with only certain topologies– a limitation of some of the existing local network alignment methods.

References

1. *http://chianti.ucsd.edu/NetworkBlast.*
2. *http://www.ensembl.org.*
3. S. Bandyopadhyay, R. Sharan, and T. Ideker. Systematic identification of functional orthologs based on protein network comparison. *Genome Res*, 16(3):428–35, 2006.
4. B.J. Breitkreutz, C. Stark, and M. Tyers. The GRID: the general repository for interaction datasets. *Genome Biology*, 4(3):R23, 2003.
5. FlyBase Consortium. The FlyBase database of the drosophila genome projects and community literature. *Nucleic Acids Res*, 31(1):172–175, 2003.
6. B.P. Kelley et al. Pathblast: a tool for alignment of protein interaction networks. *Nucleic Acids Res*, 32(Web Server issue):W83–8, 2004.
7. I. Xenarios et al. DIP, the database of interacting proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Res*, 30(1): 303–305, 2002.
8. J.D. Han et al. Evidence for dynamically organized modularity in the yeast protein-protein interaction network. *Nature*, 430(6995):88–93, 2004.
9. J.P. Miller et al. Large-scale identification of yeast integral membrane protein interactions. *Proc Natl Acad Sci USA*, 102(34):12123–12128, 2005.
10. M. Kellis et al. Methods in comparative genomics: genome correspondence, gene identification and regulatory motif discovery. *J of Computational Biology*, 11(2-3):319–355, 2004.
11. N.J. Krogan et al. Global landscape of protein complexes in the yeast saccharomyces cerevisiae. *Nature*, 440(7084):637–43, 2006.
12. P. Uetz et al. A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae. *Nature*, 403(6770):623–7, 2000.
13. R.Y. Pinter et al. Alignment of metabolic pathways. *Bioinformatics*, 21(16): 3401–3408, 2005.
14. T. Ito et al. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc Natl Acad Sci USA*, 98(8):4569–74, 2001.
15. J. Flannick, A. Novak, B.S. Srinivasan, H.H. McAdams, and S. Batzoglou. Graemlin: general and robust alignment of multiple large interaction networks. *Genome Res*, 16(9):1169–81, 2006.
16. G.H. Golub and C. Van Loan. Matrix computations. *Johns Hopkins University Press)*, 2006.
17. I. Gat-Viks, A. Tanay, D. Raijman, and R. Shamir. A probabilistic methodology for integrating knowledge and experiments on biological networks. *J of Computational Biology*, 13(2):165–181, 2006.
18. M. Koyuturk, A. Grama, and W. Szpankowski. Pairwise local alignment of protein interaction networks guided by models of evolution. *Proc of the 9th International Conference on Research in Computational Molecular Biology (RECOMB)*, 2005.
19. S. Maslov and K. Sneppen. Specificity and stability in topology of protein networks. *Science*, 296(5569):910–913, 2002.

20. E. Nabieva, K. Jim, A. Agarwal, B. Chazelle, and M. Singh. Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. *Bioinformatics*, 21 Suppl 1:i302–10, 2005.
21. K.P. O'Brien, M. Remm, and E.L. Sonnhammer. Inparanoid: a comprehensive database of eukaryotic orthologs. *Nucleic Acids Res*, 33(Database issue):D476–80, 2005.
22. C. Papadimitriou and K. Steiglitz. Combinatorial optimization: algorithms and complexity. *Dover)*, 1998.
23. Y. Qi, J. Klein-Seetharaman, and Z. Bar-Joseph. Random forest similarity for protein-protein interaction prediction from multiple sources. *Proc of the Pacific Symposium on Biocomputation*, 2005.
24. R. Singh, J. Xu, and B. Berger. Struct2net: Integrating structure into protein-protein interaction prediction. *Proceedings of the Pacific Symposium on Biocomputation*, 2006.
25. D. Sontag, R. Singh, and B. Berger. Probabilistic modeling of systematic errors in yeast two-hybrid experiments. *To Appear. Proceedings of the Pacific Symposium on Biocomputation*, 2007.
26. B.S. Srinivasan, A. Novak, J. Flannick, S. Batzoglou, and H. McAdams. Integrated protein interaction networks for 11 microbes. *Proc of the 10^{th} International Conference on Research in Computational Molecular Biology(RECOMB)*, 2006.
27. C. von Mering et al. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417(6887):399–403, 2002.
28. M.Y. Yao, T.W. Lam, and H.F. Ting. An even faster and more unifying algorithm for comparing trees via unbalanced bipartite matchings. *J of Algorithms*, 40:212, 2006.
29. C.H. Yeang and M. Vingron. A joint model of regulatory and metabolic networks. *BMC Bioinformatics*, 7:332, 2006.
30. S.H. Yook, Z.N. Oltvai, and A.L. Barabasi. Functional and topological characterization of protein interaction networks. *Proteomics*, 4(4):928–42, 2004.

Reconstructing the Topology of Protein Complexes

Allister Bernard*, David S. Vaughn*, and Alexander J. Hartemink

Department of Computer Science, Duke University, Durham, NC 27708-0129
{allister,dsv,amink}@cs.duke.edu

Abstract. Recent advances in high-throughput experimental techniques have enabled the production of a wealth of protein interaction data, rich in both quantity and variety. While the sheer quantity and variety of data present special difficulties for modeling, they also present unique opportunities for gaining insight into protein behavior by leveraging multiple perspectives. Recent work on the modularity of protein interactions has revealed that reasoning about protein interactions at the level of domain interactions can be quite useful. We present PROCTOR, a learning algorithm for reconstructing the internal topology of protein complexes by reasoning at the domain level about both direct protein interaction data (Y2H) and protein co-complex data (AP-MS). While other methods have attempted to use data from both these kinds of assays, they usually require that co-complex data be transformed into pairwise interaction data under a spoke or clique model, a transformation we do not require. We apply PROCTOR to data from eight high-throughput datasets, encompassing 5,925 proteins, essentially all of the yeast proteome. First we show that PROCTOR outperforms other algorithms for predicting domain-domain and protein-protein interactions from Y2H and AP-MS data. Then we show that our algorithm can reconstruct the internal topology of AP-MS purifications, revealing known complexes like Arp2/3 and RNA polymerase II, as well as suggesting new complexes along with their corresponding topologies.

1 Introduction

Protein complexes serve as cellular building blocks, signal transducers, and machines. Protein complexes are assembled and held together by the direct interactions of their constituent proteins with one another. Direct interactions between pairs of proteins occurs in a modular fashion when some domain of one protein comes into sufficiently close proximity with some domain of the other such that the domains mediate an interaction between the two proteins.[1]

The structure of a macromolecular protein complex can be characterized at increasing levels of refinement: 1) identify its constituent proteins, 2) reveal its topology in terms of which proteins are directly interacting with which others, 3) determine the domain-domain interactions (DDIs) that mediate the direct protein-protein interactions (PPIs), and 4) specify the complete 3D atomic structure. While the fourth has become

* These authors contributed equally to this work.

[1] This interaction can be mediated by domains, motifs, or other features of the surface of a protein. Although not entirely precise, to simplify the presentation we make no distinction between any of these terms, but rather use the term 'domain' to refer to them interchangeably.

easier for individual proteins, in the case of protein complexes, technical difficulties have posed significant obstacles to accomplishing this task [1]. Approaches are therefore needed at all these levels to help elucidate the architecture of complexes and how their individual subunits interact with each other [2].

1.1 Related Work

Recently, high-throughput PPI assays like yeast two-hybrid (Y2H) or affinity purification-mass spectrometry (AP-MS) have been conducted to partially bridge this gap. Y2H assays use a gene reporter construct to screen for direct interactions between pairs of proteins. AP-MS assays, on the other hand, provide a list of 'prey' proteins that are identified by mass-spectrometry to have been co-complexed with a given 'bait' protein during an affinity purification process.

Y2H and AP-MS assays have critically different semantics in terms of the kind of evidence they provide about PPIs. In contrast to Y2H, AP-MS assays do not provide direct evidence of PPIs but only indirect evidence. This means that correctly reasoning about both kinds of data together is challenging. Most methods tend to use either one kind of data or the other, in each case possibly supplementing the data with information from small scale experiments made available by databases like MIPS [27], DIP [38], and BIND [3]. Almost all existing methods for reasoning about both kinds of data do so by simply transforming the list of co-complexed proteins in an AP-MS purification into a list of direct protein interactions using either a 'spoke' or 'clique' model.

In the spoke model, a PPI is assumed to occur only between the bait and each prey protein. In the clique model, a PPI is assumed to occur between each pair of proteins (bait or prey). In general, a spoke model results in a lower false positive rate for PPIs within a complex at the expense of a higher false negative rate. On the other hand, a clique model results in a zero false negative rate for PPIs at the expense of a high false positive rate. If, however, the topology of PPIs explaining the AP-MS purification could be ascertained more accurately, then more accurate DDI estimates could be obtained as well. To the best of our knowledge, our approach is the first attempt to integrate AP-MS and Y2H assays by taking into account the different possible topologies of protein complexes that might explain a given AP-MS purification.

While Y2H and AP-MS assays differ importantly in terms of their semantics, they are similar in one distressing respect. For a range of published Y2H datasets and AP-MS datasets transformed by a spoke or clique model, error rates have been estimated to be significant: many true PPIs are not reported (70–98%) and many reported PPIs are not true (46–90%). Numerous computational methods have been developed to improve our knowledge of PPIs. Such methods have used a variety of techniques such as Bayesian networks [22], probabilistic decision trees [39], or kernel methods [5, 26, 18]. All of these improve prediction by incorporating multiple sources of additional information such as gene expression, Gene Ontology (GO) annotations, or interacting homologues of other species. Recent work has also examined the orthogonal problem of predicting co-complexed proteins from noisy AP-MS data [31, 8].

An alternative approach models the interactions between proteins' constituent domains to improve predictions. Here, the intuition is to use overrepresentation of certain DDIs among observed PPIs to more accurately predict both DDIs and PPIs. Such an approach was first outlined by Sprinzak and Margalit [32]. Deng and colleagues presented a probabilistic formulation in which an iterative EM algorithm was used to (locally) maximize a likelihood function [10]. Wang and colleagues extended this maximum likelihood formulation by incorporating the notion of 'active' interacting motifs [36]. Other approaches have explored Markov chain Monte Carlo methods [19] as well as the use of protein structure information [29, 30].

Here, we introduce a new statistical learning approach, PROCTOR (PROtein Complex TOpology Reconstruction), to elucidate the architecture of protein complexes, enabling effective use of the information available from both Y2H and AP-MS assays for reasoning about PPIs and DDIs. Our method provides a framework for determining the underlying topology of a protein complex and understanding how constituent proteins interact with each other to form a complex. Using available AP-MS datasets, including two recently published landmark datasets [16, 24], we determine topologies for macromolecular complexes containing essentially all of the yeast proteome. In addition, PROCTOR serves as a tool for predicting the DDIs that mediate PPIs. It accurately estimates DDIs across almost the entire yeast proteome and is efficient, displaying rapid convergence properties.

2 Motivation and Intuition

Our current knowledge of PPIs and DDIs is both very noisy and very incomplete, which means that current estimates of the probability of a given PPI or DDI can often be wildly inaccurate. As explained earlier, approaches have been developed to increase the accuracy of our estimates by modeling interactions between a proteins' constituent domains. In most of these cases, the data available for estimation comes from experimental assays that provide direct evidence of PPIs, such as Y2H assays.

In contrast, AP-MS assays do not provide direct evidence of PPIs. First, each prey protein appearing in a given AP-MS purification is either a true positive or a false positive in terms of whether it actually co-complexes with the bait protein *in vivo*. In addition, although each true positive prey must somehow co-complex with the bait, 1) not all true positive preys must co-complex with one another (the bait may participate in multiple complexes), and 2) not all true positive preys must interact directly with the bait (many interactions may be indirect, via other intermediating preys).

Of course, we do not know which preys are true positives; nor do we know which true positive preys interact directly with the bait; nor do we know, in the case of true positive preys that interact only indirectly with the bait, which other preys intervene. Another way of saying this is that although many possible 'explanations' of an AP-MS purification are possible (as shown in Figure 1), we are not sure which explanation is true. If we knew which explanation was true, we could use this information across many AP-MS purifications to robustly estimate both the DDIs that mediate all the observed PPIs, as well as the internal topology of protein complexes *in vivo*.

One way to approach this problem would be to assume in advance which one particular explanation is to be taken as true: many methods do this when they explain the results of an AP-MS purification by stipulating the existence of direct PPIs among the bait and preys via either a spoke or a clique model. In addition to the fact that these

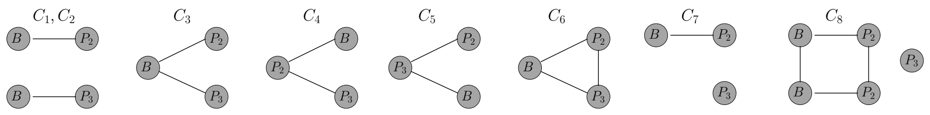


Fig. 1. Topologies of possible protein complexes when bait B is purified along with two other proteins P_2 and P_3. Edges indicate PPIs. A spoke model assumes C_3, whereas a clique model assumes C_6. In C_7 and C_8, P_3 is a false positive. If multiple copies of proteins are permitted, many additional topologies are possible (e.g., C_8). In the absence of stoichiometric information, it is impossible to distinguish the two-complex case C_1, C_2 from the one-complex case C_3.

methods lead to poor estimates of the DDIs (because the stipulated PPIs are incorrect), such methods are incapable of estimating the topology of protein complexes because they assume the topology in advance.

A more sensible way to approach this problem is to estimate the probability of each possible explanation of the data. If this is done, a model selection perspective can be employed to choose the most likely explanation of the data, or alternatively, a more Bayesian model averaging perspective can be employed to marginalize over possible explanations of the data.

Here, we demonstrate how to compute the probability of each possible explanation of an AP-MS purification. We adopt a model averaging perspective to marginalize over possible explanations of the data in computing estimates for PPIs, and for the DDIs that mediate them. We also use this framework to reveal a model-averaged view of the internal topology of the protein complexes that give rise to the lists of proteins that appear in AP-MS purifications. In what follows, we first formulate the estimation problem for PPIs, DDIs, and unknown internal topologies, and then present a sampling algorithm for approximating the solution.

3 Mathematical Formulation

Let θ_{mn} be the probability of domains m and n interacting and let $\Theta = \{\theta_{mn}\}$ represent the set of all DDI probabilities. Likewise, let w^{ij} denote the probability of proteins i and j interacting. Since we assume that a direct interaction between two proteins i and j is mediated by at least one interaction between some domain of i and some domain of j, we can use a 'noisy-or' formulation to write:

$$w^{ij} = 1 - \prod_{m,n} (1 - \theta_{mn})$$

where m and n index over the constituent domains of proteins i and j, respectively.

Define the false positive rate ϕ_p as the probability of observing a prey protein in an AP-MS purification even though that prey does not actually co-complex with the bait *in vivo*; define the false negative rate ϕ_n conversely: the probability of not observing a prey protein that actually co-complexes with the bait *in vivo*.[2] We allow different

[2] Of course, complexes are dynamic and might exist only under certain conditions, so the concept 'actually co-complexes with' is ambiguous, but this subtlety is generally ignored in the literature—nor is the data available for addressing it—so we proceed to ignore it as well.

AP-MS datasets to have different error rates, but assume within a single dataset that the false positive rate ϕ_{p} is the same for all purifications, and also that the false negative rate ϕ_{n} is the same for all purifications.

3.1 Model for an AP-MS Purification Observation

Denote the set of observed proteins in a single AP-MS purification as $\mathcal{O}$ (i.e., the bait protein plus all prey proteins). Consider the simplest (non-trivial) case, wherein bait i is purified with only a single prey j. In this case, only two explanations are possible—either the two proteins interact via one of their domains to form a two-protein complex or the proteins do not interact and the observed prey is a false positive. Given the DDI values Θ and the appropriate error rates, the complete probability of the observation is:

$$\Pr(\mathcal{O}|\Theta, \phi_{\text{n}}, \phi_{\text{p}}) = (1 - \phi_{\text{n}})w^{ij} + \phi_{\text{p}}(1 - w^{ij}) \tag{1}$$

Now consider the case when an AP-MS purification contains more than two proteins. We first make the simplifying assumption that any complex topology for an AP-MS purification can only be a tree c that spans over the complete graph induced over a subset $\mathcal{I}_c$ of the proteins $\mathcal{O}$. We require that the bait protein always be in $\mathcal{I}_c$. The remaining proteins $\mathcal{O} - \mathcal{I}_c$ are then treated as false positives of the AP-MS purification. The set of proteins $\mathcal{I}_c$ represent the true positives of the AP-MS purification and the edges $\mathcal{E}_{\mathcal{I}_c}$ that span over these true positives define an underlying topology for the complex(es) represented by this purification. We permit self-edges to be considered for proteins in $\mathcal{I}_c$ (true positives). For those concerned that self-edges violate the definition of a tree simply imagine that every protein is duplicated and a self-edge connects a protein i to its duplicate protein. We call the tree c, consisting of interaction edges $\mathcal{E}_{\mathcal{I}_c}$ that span over $\mathcal{I}_c$ and the remaining false positive proteins $\mathcal{O} - \mathcal{I}_c$, a 'complex topology tree'. Biologically, such a tree represents the underlying interaction backbone for the set of complexes represented by this AP-MS purification. Note that throughout we use the terminology 'complex topology tree' but strictly speaking this represents the topology of multiple complexes represented by this purification. The probability of any complex topology tree c is given by:

$$\Pr(c) = \left[(1 - \phi_{\text{n}})^{|\mathcal{I}_c|-1}(\phi_{\text{p}})^{|\mathcal{O}-\mathcal{I}_c|}\right] \left[\prod_{e^{ij} \in \mathcal{E}_{\mathcal{I}_c}} w^{ij}\right] \left[\prod_{e^{ij} \notin \mathcal{E}_{\mathcal{I}_c}} (1 - w^{ij})\right] \tag{2}$$

This has three terms—the first term models the true and false positives represented by the tree, the second term models the interactions spanned by the tree, and the final term ensures that this is the probability of c and only c (and not graphs for which c is a sub-graph). The probability of observing an AP-MS purification is the sum of the probabilities of all possible complex topology trees $c \in \mathcal{C}$:

$$\Pr(\mathcal{O}|\Theta, \phi_{\text{n}}, \phi_{\text{p}}) = \sum_{c \in \mathcal{C}} \Pr(c) \tag{3}$$

3.2 Incorporating Negative Information into the Observation Model

The model for observing an AP-MS purification in (3) ignores the possibility of false negative complexes or false negative errors due to mass spectrometry. It also fails to recognize that proteins which do not participate in a complex with a bait provide negative evidence for DDIs. Let $\overline{\mathcal{O}}$ be the set of proteins that are neither the bait nor observed as prey proteins in some purification. Modeling false negative complexes would require an expression for the probability of $\overline{\mathcal{O}}$ analogous to that in (3), but this is not practical since the size of $\overline{\mathcal{O}}$ is many orders of magnitude larger than that of $\mathcal{O}$. Instead, we only incorporate negative evidence for pairs of proteins in the set $\mathcal{O} \times \overline{\mathcal{O}}$. This results in the following negative observation model:

$$\Pr(\overline{\mathcal{O}}|\Theta, \phi_{\mathrm{n}}, \phi_{\mathrm{p}}) = \prod_{i \in \overline{\mathcal{O}}} \left[(1-\phi_{\mathrm{p}}) \prod_{j \in \mathcal{O}} (1 - w^{ij}) + \phi_{\mathrm{n}} \sum_{j \in \mathcal{O}} w^{ij} \prod_{\substack{k \in \mathcal{O} \\ k \neq j}} (1 - w^{ik}) \right] \quad (4)$$

For each protein $i \in \overline{\mathcal{O}}$, the first term in the product models the possibility of i being a true negative (and thus not interacting with any of the proteins in $\mathcal{O}$), while the second term models the possibility of i being a false negative (in this case, we make a further simplification by assuming i interacts with only one protein $j \in \mathcal{O}$).

4 Joint Learning Using Proteomic Data

In the preceding section, we developed a model for explaining an AP-MS purification. We note this is easily extended to incorporate data from Y2H assays: we simply treat every observed Y2H interaction as a two-protein complex whose probability of interaction is thus given by (1). For Y2H datasets, we do not need to worry about incorporating negative evidence as every unobserved interaction $\overline{\mathcal{O}}$ is treated as a negative.

4.1 Joint Model for Y2H and AP-MS Data

Suppose we have K Y2H datasets Y_k and L AP-MS datasets A_l. Let us denote all the Y2H datasets by $\mathcal{Y} = \{Y_k.\mathcal{O}, Y_k.\overline{\mathcal{O}}, Y_k.\phi_{\mathrm{p}}, Y_k.\phi_{\mathrm{n}}\}_{k=1}^{K}$ and all the AP-MS datasets by $\mathcal{A} = \{\{A_l.\mathcal{O}, A_l.\overline{\mathcal{O}}\}, A_l.\phi_{\mathrm{p}}, A_l.\phi_{\mathrm{n}}\}_{l=1}^{L}$. In the definition of $\mathcal{A}$, we have used set notation (curly brackets) around the first two elements to remind the reader that each AP-MS dataset consists of a (large) set of purifications. We can now construct a full joint probability model of all these assays as shown:

$$\Pr(\mathcal{Y}, \mathcal{A}|\Theta, \Phi) = \prod_{k=1}^{K} \Big(\prod \Pr(Y_k.\mathcal{O}|\Theta, \Phi) \prod (1 - \Pr(Y_k.\mathcal{O}|\Theta, \Phi)) \Big)$$
$$\times \prod_{l=1}^{L} \Big(\prod \Pr(A_l.\mathcal{O}|\Theta, \Phi) \Pr(A_l.\overline{\mathcal{O}}|\Theta, \Phi) \Big) \quad (5)$$

The first term represents the Y2H model and is fully specified by (1). Here, the inner product terms are over all PPIs observed and unobserved in the Y2H dataset Y_k, respectively. The second term represents the AP-MS model and is fully specified by (3) and (4). Here, the inner product is over all the purifications of the AP-MS dataset A_l. We point out that in the absence of AP-MS information, this model is equivalent to a previously published method [10]. The assumption of independence is strictly not correct but has been introduced to avoid the complicated dependency structure arising with AP-MS data. The independence assumption can be relaxed for Y2H data and leads to a more efficient algorithm. For lack of space, to simplify presentation, and since we observed only a marginal improvement in results, we present these ideas in the Supplementary Material.

4.2 Inference Using a Monte Carlo EM Algorithm

We generalize the EM algorithm of [10] to incorporate our AP-MS observation model. As in [10], we assume the error rates Φ are given, so we only need to estimate the DDI probabilities Θ. This is not a serious problem in practice, as estimates of error rates for a number of Y2H and AP-MS datasets have been published [13, 35, 11, 17].

Exact computation of (3) requires the enumeration of all possible spanning trees in a complete graph which is $O(|\mathcal{O}|^{|\mathcal{O}|-2})$ [25]. This is clearly not tractable. Instead, we approximate (3) using a Monte Carlo approach by generating random trees from the uniform distribution and then calculate (3) using these trees. This can be implemented efficiently by performing a simple random walk over a complete graph with vertex set $\mathcal{O}$ [7, 37, 25]. Thus development of an efficient sampling algorithm requires that complex topologies only be trees. This is why we have made the simplifying assumption that the space of possible complex topologies can only be trees. Note that this assumption is not unduly restrictive as a complex topology that is a graph can be represented as a combination of a set of trees. To model false positives, we randomly select a set of vertices from $\mathcal{O}$ as false positives and then use the simple random walk on the remaining set of vertices. Care has to be taken that the selected set of trees are unique. This can be done efficiently by maintaining a hashtable indexed by a hash function dependent on the structure of the tree. The number of trees (samples) generated for an AP-MS purification is initially selected as a function of $A_l.\mathcal{O}$ to reflect the fact that the number of possible trees is a function of the size of the purification. This number is then uniformly increased at each iteration of the EM algorithm to increase the accuracy of the approximation over time.

We now outline the extensions needed to the EM algorithm of [10]. The E-step requires the calculation of the binomial sufficient statistics for the binary latent variables D^{ij}_{mn} representing the presence of an interaction between domains m and n in proteins i and j. For an AP-MS purification, we need to consider two cases.

a. When $(i,j) \in \mathcal{O} \times \mathcal{O}$, the E-step calculations are shown below, where $\mathbf{I}_{\{e^{ij} \in \mathcal{E}_{\mathcal{I}_c}\}}$ is an indicator variable which returns 1 if the edge e^{ij} is in the edge set $\mathcal{E}_{\mathcal{I}_c}$ of tree c and 0 otherwise.

$$E(D^{ij}_{mn}|\Theta^{(t-1)},\Phi) = \frac{\theta^{(t-1)}_{mn}\Pr(\mathcal{O}|D^{ij}_{mn},\Theta^{(t-1)},\Phi)}{\Pr(\mathcal{O}|\Theta^{(t-1)},\Phi)} \quad (6)$$

$$\Pr(\mathcal{O}|D^{ij}_{mn},\Theta^{(t-1)},\Phi) = \sum_{c\in\mathcal{C}} \frac{\Pr(c)\mathbf{I}_{\{e^{ij}\in\mathcal{E}_{\mathcal{I}_c}\}}}{w^{ij}}$$

b. When $(i, j) \in \overline{\mathcal{O}} \times \mathcal{O}$, the E-step calculations are shown below.

$$E(D_{mn}^{ij}|\Theta^{(t-1)}, \Phi) = \frac{\theta_{mn}^{(t-1)} \Pr(\overline{\mathcal{O}}|D_{mn}^{ij}, \Theta^{(t-1)}, \Phi)}{\Pr(\overline{\mathcal{O}}|\Theta^{(t-1)}, \Phi)} \tag{7}$$

$$\Pr(\overline{\mathcal{O}}|D_{mn}^{ij}, \Theta^{(t-1)}, \Phi) = \phi_{\mathrm{n}} \prod_{\substack{k \in \mathcal{O} \\ k \neq j}} (1 - w^{ik})$$

If the total number of possible interactions between domains m and n across all Y2H and AP-MS datasets is T_{mn}, then the M-step involves a simple recursive formula for θ_{mn}, where the summation is over all pairs of proteins i and j that contain domains m and n, respectively.

$$\theta_{mn}^{(t)} = \frac{1}{T_{mn}} \sum E(D_{mn}^{ij}|\Theta^{(t-1)}, \Phi) \tag{8}$$

We now briefly examine the time and space requirements for each iteration of the EM algorithm. Let P be the size of the proteome, D be the total number of unique domains across all proteins, S be the maximum number of samples (trees) generated for a particular EM iteration, R be the total number of AP-MS purifications (this is the sum total of all purifications across the L AP-MS datasets), H be the size of the AP-MS purification with the maximum number of prey proteins, and T be the maximum number of protein-protein pairs with the same domain-domain combination (terms summed over in (8)). Y2H and AP-MS datasets require $O(KP + RH)$ space (we assume that the number of observed protein-protein interactions in any Y2H assay is orders of magnitude less than the total number of possible interactions). Each EM iteration involves maintaining $O(P^2)$ and $O(D^2)$ matrices for updating the sufficient statistics. Selecting a unique set of trees generated from the simple random walk requires $O(SH)$ space. Thus, the total space usage at each EM iteration is $O(KP + RH + P^2 + D^2 + SH)$. In practice, H is usually orders of magnitude smaller than P and D. For the rare AP-MS purifications with many prey proteins, we can set an upper bound on the value of SH without adversely affecting the algorithm. Updating the sufficient statistics requires $O(P^2 + D^2T)$ time. Generating a sample of random trees requires $O(SH^2)$ time (checking whether a tree is unique can be made $O(1)$ with a good hash function, the simple random walk requires $O(H \log H)$, and computing the cost of a tree requires $O(H^2)$). Thus, the time of each EM iteration is bounded by $O(P^2 + D^2T + RH + SH^2)$. In practice, T is not prohibitively large. Hence, both time and space requirements are dominated by the number of samples S used in each iteration, the size of the proteome P, and the number of unique domains D.

4.3 Recovering the Topology of a Protein Complex

Our Monte Carlo EM algorithm provides estimates for the DDI probabilities Θ. Given Θ, we can compute the probabilities of different complex topologies. If we assume the topology is tree, we can determine the most probable such tree using standard minimum spanning tree (MST) algorithms over the complete graph with vertex set $\mathcal{O}$. The MST is essentially a maximum likelihood estimate of the complex topology, restricted to a tree. One might also wish to determine the most probable DDIs that mediate the PPIs in the graph. In this case, one could construct an MST in an annotated graph over the set of proteins $\mathcal{O}$: instead of having a single edge between proteins i and j, the annotated graph would have as many edges as the number of possible DDIs between i and j. A natural extension to this would be to then determine a complex topology for the proteins in $\mathcal{O}$ such that edges selected for the spanning tree never re-use any of a protein's domains. We call this the *Domain Re-use Constraint* problem and have proved that it is NP-Hard (see Supplementary Material).

Using a maximum likelihood estimate as described above would prevent us from noticing if a different tree were nearly equally probable. We overcome this by using Bayesian model averaging instead, employing our sampling algorithm to generate a huge number of random trees and then computing marginal probability estimates for all the edges of the graph. Using this Bayesian model averaging approach, we are not restricted to complex topology trees but can determine complex topology graphs (i.e., with cycles). Note also that such topology graphs can represent the topology of more than one complex since proteins can participate in more than one complex.

5 Results

We applied our algorithm to publicly available high-throughput proteomic interaction data for *S. cerevisiae*. We obtained a total of 6,864 purifications from five different AP-MS datasets. Four datasets were tandem affinity purification (TAP) assays with 589 purifications [15], 294 purifications [23], 1,993 purifications [16], and 3,436 purifications [24], respectively. The fifth dataset was a high-throughput mass spectrometric protein complex identification (HMS-PCI) assay with 552 purifications [20]. The average size of the number of prey proteins across all 6,864 AP-MS purifications was 13, with a maximum size of 332.

We also used two Y2H datasets. The first included two Y2H screens, one where all transformants were allowed to mate with each other (6,000 $\times$ 6,000) and another where all transformants were allowed to mate with only 192 (6,000 $\times$ 192) [34]. Ideally, we would treat these two screens as separate datasets, but the identities of the 192 transformants used in the second screen were not recorded (Peter Uetz, personal communication) so we pooled the two screens into one dataset containing 957 interactions. The second Y2H dataset included one high-throughput screen (6,000 $\times$ 6,000) producing a total of 4,549 interactions, 841 of which were detected more than three times and labeled as 'core' interactions [21]. We treated the core and full sets as separate Y2H datasets to reflect the difference in the confidence of their observations.

All the data taken together represent 5,925 yeast proteins, covering essentially the entire yeast proteome. We used protein-domain information from two different sources—Pfam [4] and InterPro [28]—and evaluated all our results with each of these sources independently. The average number of domains per protein was 2 for both Pfam and InterPro, with the maximum number of domains in a protein being 30 and 18, respectively.

For AP-MS assays, we used previously reported false positive and false negative error estimates [17]. To estimate error rates for direct protein interaction assays like Y2H (or AP-MS assays converted using spoke or clique models), we used a published method for simultaneously estimating the error rates of two datasets given a gold standard set of known interactions [11]. By comparing two independent datasets to a gold standard, the error rates are more trustworthy than those obtained by comparing each dataset to the gold standard by itself. To further increase our confidence in these values, we paired each dataset with several others, ran the method for each pair, and averaged all the results (see Supplementary Material for values).

Although the space and time complexity of our learning algorithm increases as the number of samples increase with each EM iteration, in practice the algorithm converged rapidly within 25–30 iterations, with each iteration taking an average of 30 minutes on a machine with 4GB RAM. Convergence was assessed as change in log likelihood being less than 0.01%. We compared four algorithms: 1) a previously published algorithm [10] using Y2H datasets only (called Y2H-ONLY), 2) the same algorithm as 1 but including AP-MS data transformed using a spoke model (called SPOKE), 3) the same algorithm as 1 but including AP-MS data transformed using a clique model (called CLIQUE) and 4) our algorithm using separate observation models for Y2H and AP-MS datasets (called PROCTOR). We experimented with random initializations for θ_{mn} but consistently observed the best results by initializing θ_{mn} to a constant less than 0.01. None of these algorithms displayed significant differences for small variations in the error rate estimates, as has been observed by others as well [10]. Also, the observed results were virtually indistinguishable whether we used protein-domain information from Pfam or Interpro, so we only show results using Pfam.

5.1 Evaluation of Domain-Domain Interaction Predictions

We used structural information from 3DID [33] and iPfam [14] to define positive DDIs for our evaluation set. As suggested previously [30], we pruned out those DDIs having no evidence in the training data because none of the algorithms will be able to do better than random for these, leaving us with a total of 1,501 positive DDIs for evaluation. We selected 40 times as many negative DDIs by randomly sampling from DDIs not in our positive set. Figure 2 shows the resulting precision-recall curves; we use precision-recall curves instead of ROC curves since they are more informative when using highly skewed evaluation sets [9]. The SPOKE approach does a little better than the CLIQUE approach, while Y2H-ONLY fares the worst. This is presumably due to the fact that it has the least available training data. Overall, the area under the DDI precision-recall curve increases at least 0.36 with PROCTOR. On examining the curve for the CLIQUE approach more closely, we discovered its poor performance was partly due to its inability to predict self-interacting domains. We then modified CLIQUE to include self-interacting proteins (a protein interacts not only with all other proteins but also with itself). This modification improved the precision-recall curve for CLIQUE to slightly better than that of SPOKE and Y2H-ONLY (results not shown).

5.2 Evaluation of Protein-Protein Interaction Predictions

We obtained positive PPIs for our evaluation set from the small scale experiments contained in DIP [38]. We again pruned out PPIs having no evidence in the training data, leaving us with a total of 3,144 positive PPIs for evaluation. We selected 40 times as

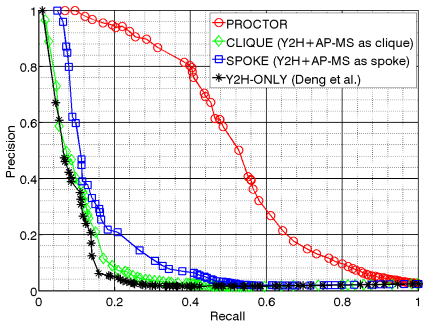


Fig. 2. DDI: precision vs. recall

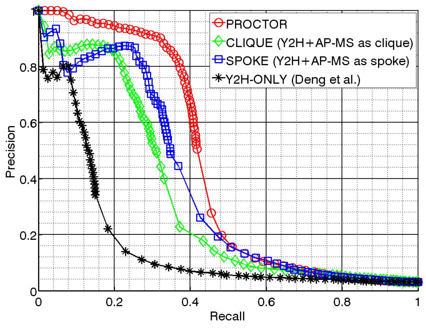


Fig. 3. PPI: precision vs. recall

many negative PPIs by randomly sampling from PPIs not in our positive set. As an alternative negative evaluation set, we also obtained the localization of 1,119 proteins [12] and randomly sampled from pairs of proteins with different localizations. These two negative evaluation sets showed no discernible difference (results not shown), so we only present results using the former (without localization). Somewhat surprisingly, the PPI precision-recall curves, shown in Figure 3, are not as dramatically different as the DDI precision-recall curves. Although PROCTOR still performs the best, the area under the PPI precision-recall curve increases only 0.08 compared to the next best algorithm (SPOKE). Upon close examination, we determined that this difference between DDI and PPI precision-recall curves is due to a combination of the inherently non-uniform contributions of different DDIs to the PPI network, the low prevalence of PPIs (the ratio of non-PPIs to PPIs in the proteome is estimated to be at least 1,200), and differences in the evaluation sets. Both SPOKE and CLIQUE predict many more DDIs than PROCTOR. However, these DDIs are distributed across a relatively small fraction of the PPIs. Both SPOKE and CLIQUE also predict many more PPIs than PROCTOR, but these additional PPIs are extremely small in number when compared to the total number of possible PPIs. As a result of all these factors, the improvement observed in Figure 3 is not as visually dramatic as that seen in Figure 2.

5.3 Prediction of Protein Complex Topologies

Finally, we examine PROCTOR's ability to reconstruct the topology of protein complexes in an AP-MS purification. To construct a complex topology graph explaining a purification, we used our sampling algorithm to compute an average confidence for each edge in the graph. For two well-studied complexes, we compared the known crystal structure in PDB [6] with the topology predicted by PROCTOR. The first complex we studied is part of the core component of RNA polymerase II (PDBid: `1sfo`) and the reconstructed complex topology is shown in Figure 4. Tfg1 was used as the bait in this purification and is involved in both transcription initiation and elongation of RNA polymerase II. Besides the five members of the core component (Rpb1, Rpb2, Rpb3, Rpb6, Rpb8), the Tfg1 purification pulled down two unrelated proteins, Hst4 and Msy1, both of which were correctly identified as false positives. The second complex we studied

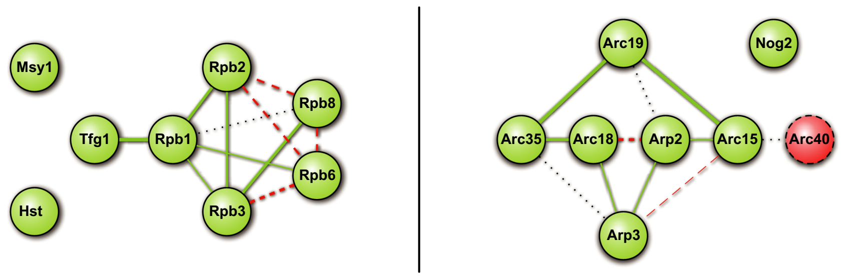


Fig. 4. Predicted complex topology of Arp2/3 complex and a portion of RNA polymerase II. The width of each edge reflects our relative confidence in the corresponding PPI. Dashed red nodes indicate nodes wrongly labeled as a false positive. Dashed red edges indicate PPIs wrongly labeled as interactions. Dotted black edges indicate PPIs wrongly labeled as non-interactions. All solid green nodes and solid green edges are correct.

is the Arp2/3 complex (PDBid: `1k8k`) which is a highly conserved actin nucleation center consisting of seven proteins. We had AP-MS purifications with both Arp2 and Arp3 as bait. We reconstructed the same topology from both purifications, one of which is shown in Figure 4. Only Arc40 was not selected as part of the complex (incorrectly). We suspect this is because Arc40 appeared in the Arp2 purification, but not in the Arp3 purification. On the other hand, Nog2 was correctly identified as a false positive. Observe that the topology of neither complex can be accurately characterized by the spoke or clique models. In addition to reconstructing protein complex topologies, PROCTOR also provides an estimate of the DDIs that mediate the PPIs in these topologies, but this has been left out of the figure to reduce clutter.

6 Discussion

We demonstrate the accuracy of our learning algorithm PROCTOR for predicting both PPIs and the DDIs that mediate them, and more importantly, in reconstructing the topologies of protein complexes. These topologies are useful in providing a better understanding of the architecture of cellular protein complexes. This has been achieved by careful modeling of the different semantics of Y2H assays (direct protein interaction evidence) and AP-MS assays (protein co-complex evidence). Our implementation of PROCTOR also permits variations within one kind of assay to be modeled with different error rates: e.g., TAP vs. HMS-PCI protocols used for AP-MS, or core vs. full for Y2H. Our results demonstrate that sizable improvements in accuracy can be obtained by careful modeling of the semantics of AP-MS data; if the distinct semantics of AP-MS data are ignored, algorithm performance degenerates rapidly. Our implementation of PROCTOR is also the first domain-domain model for protein-protein interactions to cover essentially all of the yeast proteome. Previously reported results [10, 36, 30] have less than half this coverage, and in no case cover more than 42% of the proteome.

The most important feature of PROCTOR is its ability to further our understanding of the architecture of cellular macromolecules through the reconstruction of complex topology graphs (along with the corresponding DDIs that mediate them). We plan to set up a database of PROCTOR's protein complex topology reconstructions as a useful tool for others in the community.

Currently, PROCTOR reports a topology for explaining each individual purification, ignoring potential overlap between purifications either due to repeated experiments or the use of different members of a complex as bait proteins. It would be useful to extend our algorithm to use such information for partitioning AP-MS purifications into their constituent complexes. We might be able to use ideas developed by others for predicting co-complex membership [31, 8] to assist in this. In addition, some of the available AP-MS datasets may soon be enhanced to include stoichiometric information for their purifications (Anne-Claude Gavin, personal communication). We would like to explore the use of such information in further elucidating the topology of protein complexes.

Acknowledgments. The authors would like to thank Ed Iversen and Kamesh Munagala for useful discussions and suggestions, and Carlo Tomasi for suggesting the name PROCTOR. The research presented here was supported by a National Science Foundation CAREER award and an Alfred P. Sloan Fellowship to A.J.H.

Supplementary Material can be found at http://www.cs.duke.edu/∼amink/.

References

[1] P. Aloy, B. Bottcher, H. Ceulemans, C. Leutwein, C. Mellwig, S. Fischer, A.C. Gavin, P. Bork, G. Superti-Furga, L. Serrano, and R.B. Russell. Structure-based assembly of protein complexes in yeast. *Nature*, 303:2026–2029, 2004.

[2] P. Aloy and R.B. Russell. Structural systems biology: modeling protein interactions. *Nature Reviews in Molecular Cell Biology*, 7:188–197, 2006.

[3] G.D. Bader, I. Donaldson, C. Wolting, B.F. Ouellette, T. Pawson, and C.W. Hogue. BIND - the biomolecular interaction network database. *Nucleic Acids Research*, 29:242–245, 2001.

[4] A. Bateman et al. The pfam protein families database. *Nucleic Acids Research*, 32: D138–D141, 2004.

[5] A. Ben-Hur and W.S. Noble. Kernel methods for predicting protein-protein interactions. In *ISMB*. ISCB, June 2005.

[6] H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, and P.E. Bourne. The protein data bank. *Nucleic Acids Research*, 28: 235–242, 2000.

[7] A.Z. Broder. Generating random spanning trees. In *Foundations of Computer Science*, pages 442–447. IEEE, 1989.

[8] W. Chu, Z. Ghahramani, R. Krause, and D.L. Wild. Identifying protein complexes in high-throughput protein interaction screens using an infinite latent feature model. In *PSB*, pages 231–242, 2006.

[9] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proc. 23rd International Conference on Machine Learning*, pages 233–240, 2006.

[10] M. Deng, S. Mehta, F. Sun, and T. Chen. Inferring domain-domain interactions from protein-protein interactions. In *RECOMB '02: Proceedings of the sixth annual international conference on Computational biology*, pages 117–126. ACM Press, 2002.

[11] P. D'haeseleer and G.M. Church. Estimating and improving protein interaction error rates. In *CSB*. IEEE, August 2004.

[12] A. Drawid and Mark Gerstein. A Bayesian system integrating expression data with sequence patterns for localizing proteins: comprehensive application to the yeast genome. *Journal of Molecular Biology*, 301:1059–1075, 2000.

[13] R. Edwards and L. Glass. Combinatorial explosion in model gene networks. *Chaos*, 10:691–704, September 2000.

[14] R. Finn, M. Marshall, and A. Bateman. ipfam: visualization of protein-protein interactions in pdb at domain and amino acid resolutions. *Bioinformatics*, 21:410–412, 2005.

[15] A.-C. Gavin et al. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 415:141–147, 2002.

[16] A.-C. Gavin et al. Proteome survey reveals modularity of the yeast cell machinery. *Nature*, 440:631–636, 2006.

[17] M.A. Gilchrist, L.A. Salter, and A. Wagner. A statistical framework for combining and interpreting proteomic datasets. *Bioinformatics*, 20:689–700, 2004.

[18] S.M. Gomez, W.S. Noble, and A. Rzhetsky. Learning to predict protein-protein interactions. *Bioinformatics*, 19:1875–1881, 2003.

[19] S.M. Gomez and A. Rzhetsky. Towards the prediction of complete protein-protein interaction networks. In *PSB*, volume 7, pages 413–424, 2002.

[20] Y. Ho et al. Systematic identification of protein complexes in Saccharomyces cerevisiae by mass spectrometry. *Nature*, 415:123–124, 2002.

[21] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *PNAS*, 98:4569–4574, 2001.

[22] R. Jansen, H. Yu, D. Greenbaum, Y. Kluger, Krogan N.J., Chung S., Emili A., Snyder M., Greenblatt J.F., and Gerstein M. A Bayesian networks approach for predicting protein-protein interactions from genomic data. *Science*, 302:449–453, 2003.

[23] N.J. Krogan et al. High-definition macromolecular composition of yeast rna-processing complexes. *Molecular Cell*, 13:225–39, 2004.

[24] N.J. Krogan et al. Global landscape of protein complexes in the yeast saccharomyces cerevisiae. *Nature*, 440:637–643, 2006.

[25] R. Lyons and Y. Peres. *Probability on trees and networks*. Cambridge University Press, in progress, 2005.

[26] S. Martin, D. Roe, and J.-L. Faulon. Predicting proteinprotein interactions using signature products. *Bioinformatics*, 21:218–226, 2005.

[27] H.W. Mewes, D. Frishman, U. Guldener, G. Mannhaupt, K. Mayer, M. Mokrejs, B. Morgenstern, M. Munsterkotter, S. Rudd, and B. Weil. MIPS: a database for genomes and protein sequences. *Nucleic Acids Research*, 30:31–34, 2002.

[28] N.J. Mulder et al. Interpro, progress and status in 2005. *Nucleic Acids Research*, 33:D201–D205, 2005.

[29] T.M. Nye, C. Berzuini, W.R. Gilks, M.M. Babu, and S.A. Teichmann. Statistical analysis of domains in interacting protein pairs. *Bioinformatics*, 21:993–1001, 2005.

[30] T.M. Nye, C. Berzuini, W.R. Gilks, M.M. Babu, and S.A. Teichmann. Predicting the strongest domain-domain contact in interacting protein pairs. *Statistical Applications in Genetics and Molecular Biology*, 5, 2006.

[31] D. Scholtens, M. Vidal, and R. Gentleman. Local modeling of global interactome networks. *Bioinformatics*, 21:3548–3557, 2005.

[32] E. Sprinzak and H. Margalit. Correlated sequence-signatures as markers of protein-protein interaction. *J. Mol. Biol.*, 311:681–692, 2001.

[33] A. Stein, R. Russell, and P. Aloy. 3did: interacting protein domains of known three-dimensional structure. *Nucleic Acids Research*, 33:D413–D417, 2005.

[34] P. Uetz, L. Giot, G. Cagney, T.A. Mansfield, R.S. Judson, V. Narayan, D. Lockshon, M. Srinivasan, P. Pochart, A. Qureshi-Emili, Y. Li, B. Godwin, D. Conover, T. Kalbfleisch, G. Vijayadamodar, M. Yang, M. Johnston, S. Fields, and J.M. Rothberg. A comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae*. *Nature*, 403(6770):623–627, 2000.

[35] C. von Mering, R. Krause, B. Snel, M. Cornell, S.G. Oliver, S. Fields, and P. Bork. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417: 399–403, 2002.

[36] H. Wang, E. Segal, A. Ben-Hur, D. Koller, and D. Brutlag. Identifying protein-protein interaction sites on a genome-wide scale. In *Advances in Neural Information Processing Systems (NIPS 2004)*, Vancouver, Canada, 2004.

[37] D.B. Wilson. Generating random spanning trees more quickly than the cover time. In *Symposium on Theory of Computing*, pages 396–303. ACM, 1996.

[38] I. Xenarios, L. Salwinski, X.J. Duan, P. Higney, S.M. Kim, and D. Eisenberg. DIP, the database of interacting proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Research*, 30:303–305, 2002.

[39] L.V. Zhang, S.L. Wong, O.D. King, and F.P. Roth. Predicting co-complexed protein pairs using genomic and proteomic data integration. *BMC Bioinformatics*, 5:1–15, 2004.

Network Legos: Building Blocks of Cellular Wiring Diagrams

T.M. Murali and Corban G. Rivera

660 McBryde Hall, Department of Computer Science,
Virginia Polytechnic Institute and State University, Blacksburg VA 24061

Abstract. Publicly-available data sets provide detailed and large-scale information on multiple types of molecular interaction networks in a number of model organisms. These multi-modal universal networks capture a static view of cellular state. An important challenge in systems biology is obtaining a dynamic perspective on these networks by integrating them with gene expression measurements taken under multiple conditions.

We present a top-down computational approach to identify building blocks of molecular interaction networks by

(i) integrating gene expression measurements for a particular disease state (e.g., leukaemia) or experimental condition (e.g., treatment with growth serum) with molecular interactions to reveal an *active network*, which is the network of interactions active in the cell in that disease state or condition and
(ii) systematically combining active networks computed for different experimental conditions using set-theoretic formulae to reveal *network legos*, which are modules of coherently interacting genes and gene products in the wiring diagram.

We propose efficient methods to compute active networks, systematically mine candidate legos, assess the statistical significance of these candidates, arrange them in a directed acyclic graph (DAG), and exploit the structure of the DAG to identify true network legos. We describe methods to assess the stability of our computations to changes in the input and to recover active networks by composing network legos.

We analyse two human datasets using our method. A comparison of three leukaemias demonstrates how a biologist can use our system to identify specific differences between these diseases. A larger-scale analysis of 13 distinct stresses illustrates our ability to compute the building blocks of the interaction networks activated in response to these stresses.

1 Introduction

Rapid advances in high-throughput and large-scale biological experiments are inspiring the study of properties of sets of molecules that act in concert [13], how these sets interact with each other, and how these interactions change dynamically in response to perturbations. Such groups of molecules have been dubbed various names such as gene modules [5,30,34], module networks [31] and gene sets [35]. One of the fundamental challenges of systems biology is to automatically compute such modules and the relationships between them by integrating multiple types of data and discovering patterns of coordinated activity contained in these data sets.

In this paper, we present a top-down computational approach that identifies building blocks of cellular networks by

(i) integrating gene expression measurements for a particular disease state (e.g., leukaemia) or experimental condition (e.g., treatment with growth serum) with molecular interactions to reveal an *active network*, which is the network of interactions active in the cell in that disease state or condition and
(ii) systematically combining active networks computed for different experimental conditions using set-theoretic formulae to reveal *network legos*, which are modules of coherently interacting genes and gene products in the wiring diagram. These network legos are potential building blocks of the wiring diagram, since we can express each active network as a composition of network legos.

We illustrate the essence of our method using an example. Armstrong et al. [2] demonstrated that lymphoblastic leukaemias involving translocations in the *MLL* gene constitute a disease different from conventional acute lymphoblastic (ALL) and acute myelogenous leukaemia (AML). The authors based their analysis on the comparison of gene expression profiles from individuals diagnosed with ALL, AML, and MLL. We reasoned that the networks of molecular interactions activated in these diseases may also show distinct differences. First, we computed networks of molecular interactions activated in each leukaemia, as described in Section 3.2. Next, we systematically combined these active networks in multiple ways into network legos using graph intersections and graph differences, using the method presented in Section 3.3. Our system generated all the possible 19 ($3^3 - 2^3$) combinations involving the ALL, AML, and MLL active networks and their complements and connected them in the directed acyclic graph (DAG) displayed in Figure 1. In this DAG, each node represents a single combination, e.g., the leftmost node on the top row represents the MLL active network while the leftmost node in the middle row represents the interactions activated in AML but not in MLL (the "formula" AML$\cap$!MLL). A solid blue edge directed from a child to a parent indicates that the formula for the child (e. g., MLL) appears as a part of the formula for the parent (e.g., MLL$\cap$!AML), while a green edge indicates that the child's formula (e.g., MLL) appears negated in the parent's formula (e.g., AML$\cap$!MLL). The DAG is a concise representation of all the formulae we compute and the subset relationships between the formulae. We did not consider complementation-only formulae such as !ALL$\cap$!AML since the resulting networks are unlikely to be biologically useful. In Section 4.1, we describe how function and pathway enrichment of these networks suggests differences and similarities between ALL, AML, and MLL. For instance, Figure 1 displays an example of the enrichment of the interactions in the KIT pathway in the computed network legos. Interactions in this pathway are significantly

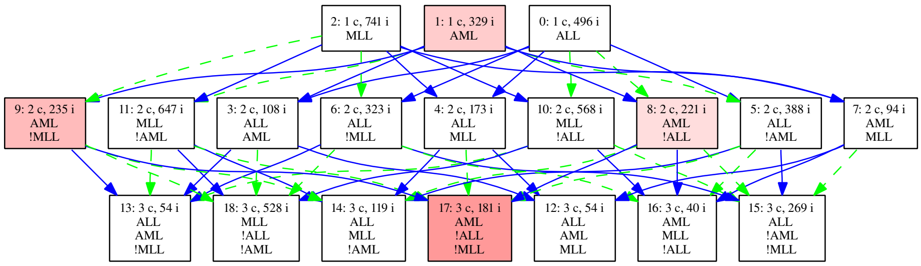


Fig. 1. The lattice connecting combinations of ALL, AML, and MLL active networks. Each node contains an index, the number of 'c'onditions, the number of 'i'nteractions and the active networks participating in the formula, with '!' indicating complementation. Colours indicate differential enrichment of the interactions in the KIT pathway in the computed combinations. Darker colours denote more significant enrichment values.

enriched only in formulae that involve the AML active network; the most significant enrichment (3.5×10^{-7}) occurs in the formula AML $\cap$!ALL $\cap$!MLL.

Given a wiring diagram and the transcriptional measurements for a particular condition, we use the gene expression data to induce edge weights in the wiring diagram. We find dense subgraphs [8] in this weighted graph to compute the active network for that condition. Given the active networks for a number of different conditions, we first represent the active networks in an appropriately-defined binary matrix and compute closed itemsets [1,40] in the matrix. Each itemset simultaneously represents a set-theoretic combination of particular active networks and a subgraph of the wiring diagram; we call such a subgraph a "network block". We exploit the subset structure between blocks to arrange them in a DAG. When the number of active networks is large, we may compute a very large number of highly-similar blocks. Not all these blocks are likely to be network legos. We assess the statistical significance of each block by simulation and identify those that are maximally significant, i.e., more significant than any descendant or an ancestor in the DAG. We deem these blocks to be network legos.

We develop two measures to assess the quality of the network legos we compute. *Stability* measures to what degree we can recompute the same legos when we remove each active network in turn from the input. *Recoverability* measures to what extent we recoup the original active networks when we combine network legos. These two notions test two different aspects of network lego computation. Considering active networks to be the inputs and network legos to be the outputs, stability measures how much the outputs change when we perturb the inputs by removing one of the inputs at a time. In contrast, recoverability asks whether we can reclaim the inputs by combining the outputs; thus recoverability is a measure of how well the network legos serve as building blocks. To assess the biological content of network legos, we measure the functional enrichment of the genes and interactions that belong to a network lego. For each function, we track its degree of enrichment in the DAG to visually highlight differences among the active networks, as displayed in Figure 1.

In addition to the ALL-AML-MLL analysis, we apply our approach to a collection of 178 arrays measuring the gene expression responses of HeLa cells and primary human lung fibroblasts to cell cycle arrest, heat shock, endoplasmic reticulum stress, oxidative stress, and crowding [21]. Overall, the dataset contains 13 distinct stresses over the two cell types. Our method computes 143 network legos. In this paper, we focus on a structural analysis of the network legos. We carefully examine the compositions of these network legos to demonstrate that they are true building blocks of the active networks for these 13 stresses. We demonstrate that our algorithm to construct network legos is stable: when we remove each active network and recompute network legos, we are able to recompute most network legos at least 95% fidelity. We also demonstrate that we can recover active networks with almost perfect accuracy by composing network legos. Further analysis of the network legos reveals that the active networks corresponding to cell cycle arrest contain interactions that are quite distinct from the network of interactions activated by the other stresses. When we remove the two cell cycle arrest data sets, we compute only 15 network legos. Of the 11 remaining active networks, we recover five with complete accuracy and one with 99.9% accuracy. We recover the other five active networks with accuracies ranging from 71% to 92%. Taken together, these statistics indicate that the network legos we detect are indeed building blocks of the networks activated in response to the stresses studied by Murray et al. [21].

There are two ways in which a biologist can use our system. In the first, our system allows the systematic comparison of responses to a small number of different conditions, diseases, or perturbations tested in the same lab. The ALL-AML-MLL comparison we presented earlier and discuss further in Section 4.1 is such an application. In the second, a biologist can analyse a specific condition of interest in the context of a large compendium of other conditions, compute the building blocks of the networks activated in these conditions, and ask how the building blocks compose the active network for the specific condition of interest to the biologist. In Section 4.2, we analyse 13 distinct stresses imparted to human cells to illustrate this application. In this respect, our work is similar to the approach developed by Tanay et al. [38]. They integrate a diverse collection of datasets into a bipartite graph representing connections between genes and gene properties. Their modules are statistically-significant biclusters [37] in this graph. They represent a target gene expression dataset as a bipartite graph and compute which already-computed modules respond in the target data set. Our approach differs from theirs in two respects. First, we represent differences and similarities between multiple conditions explicitly as a set theoretic formula involving the interaction network activated in each condition. Two, when we analyse a large compendium of gene expression data sets, we exploit the subset structure between these formulae to detect network legos, statistically-significant building blocks of these active networks.

The success of our approach stems from a number of factors. First, unlike other approaches that simultaneously integrate multiple gene expression data sets in the context of the network scaffold [5,22], we compute individual active networks for each data set and associate the active network with the corresponding disease or perturbation. This approach allows us to explicitly compare and contrast different conditions. Second, we treat interactions (rather than genes or proteins) as the elementary objects of our analysis. Therefore, different network legos may share genes, allowing for the situation when a gene participates in multiple biological processes and is activated differently in these processes. Finally, we develop a simple but effective recursive method to assess the statistical significance of a network lego and to weed out sub-networks that masquerade as building blocks but contain true network legos. Taken together, formulae and network legos provide a dynamic and multi-dimensional view of cell circuitry obtained by integrating molecular interaction networks, gene expression data, and descriptions of experimental conditions.

2 Previous Work

A number of approaches, recently surveyed by Joyce and Palsson [17], have been developed to integrate diverse types of biological data and "mine" these datasets to find groups of molecules (usually genes and/or proteins) that act in concert to perform a specific biological task. Integrating information on available molecular interactions such as protein-protein, protein-DNA, protein-metabolite, and genetic interactions yields a multi-modal wiring diagram [32]. However, such a network typically provides a static view of the underlying cellular circuitry. A number of techniques attempt to obtain a dynamic view of cell state by overlaying measurements of molecular profiles (usually in the form of gene expression data) obtained under multiple conditions on the wiring diagram [10,12,15,17,20]. For instance, Han et al. [12] categorised hubs in *S. cerevisiae* protein interaction networks into "party" hubs, which interact with most of their partners simultaneously, and "date" hubs, which bind their different partners at different times or locations. Luscombe et al. [20] characterise topological changes in the structure of the *S. cerevisiae* transcriptional regulatory network under different conditions. The SAMBA algorithm [36] integrates a wide variety of data types in *S. cerevisiae* to identify gene modules with statistically significant correlated behaviour across diverse data sources. The bioPIXIE system [22] probabilistically integrates diverse genome-wide datasets and computes pathway-specific networks that include query genes input by a biologist.

Other methods have computed gene modules by focusing solely on gene expression data collected across multiple cellular conditions; they analyse large compendia of such data to reveal similarities and differences between multiple cellular conditions [30] or between organisms [7,34], predict functional annotations [14,18], reconstruct regulatory networks [41] and networks activated in diseases [6], zero in on biomarkers for diseases [25,26], and identify the gene products and associated pathways that a drug compound targets [10].

3 Algorithms

We describe the main computational ingredients of our approach in stages. We first define some useful terminology. Next, we present our method to integrate a cellular wiring diagram with the gene expression data for a single condition to compute the active network for that condition. Third, we describe how we combine active networks for different conditions to form blocks. Fourth, we discuss how we compute the statistical significance of blocks, arrange them in a DAG, and exploit the DAG to identify network legos, which are the most statistically-significant blocks in the DAG. Finally, we present our methods to measure the stability of network legos and assess how well we can recover active networks from the network legos.

3.1 Definitions

Our method takes as input (i) a cellular wiring diagram W representing known physical and/or genetic interactions between genes or gene products in an organism and (ii) a compendium of transcriptional measurements in the same organism obtained under various conditions such as diseases (e.g., breast cancer), stimuli (e.g., heat shock), or other perturbations (e.g., gene knock-out or over-expression). We assume that each gene expression dataset in the compendium contains measurements for multiple gene chips. For instance, a breast cancer dataset might include data from multiple patients while a heat shock dataset may measure gene expression at different time points.

Given a gene expression data set D_c for a condition c, we say that a gene *responds in* c if the expression values of the gene in D_c vary by more than an input threshold. Let g and h be two genes that respond in c and let $e = (g, h)$ be an interaction in W. We say that e is *active in* c if the expression profiles of g and h in D_c are correlated to a statistically-significant extent. The *active network* A_c *in* c is the sub-network of interactions in W that are active in c. We describe the details of how we detect responding genes, active interactions, and active networks in Section 3.2.

Let $\mathcal{A}$ denote the set of active networks for each of the conditions in the input compendium. We define a *block* to be a triple $(G, \mathcal{P}, \mathcal{N})$, where G is a subgraph of W; $\mathcal{P}$ and $\mathcal{N}$ are subsets of $\mathcal{A}$; $\mathcal{P} \neq \emptyset$; and $\mathcal{P} \cap \mathcal{N} = \emptyset$ such that

1. for each *positive* active network $P \in \mathcal{P}$, $G \subseteq P$,
2. for each *negative* active network $N \in \mathcal{N}$, $G \cap N = \emptyset$,
3. G is maximal, i.e., adding an edge to G violates at least one of the first two properties,
4. $\mathcal{P}$ is maximal, i.e., there is no $P \in \mathcal{A} - \mathcal{P}$ such that $G \subseteq P$, and
5. $\mathcal{N}$ is maximal, i.e., there is no $N \in \mathcal{A} - \mathcal{N}$ such that $G \cap N = \emptyset$.

Intuitively, we can form G by taking the intersection of all the active networks in $\mathcal{P}$ and removing any edge that appears in any of the active networks in $\mathcal{N}$. In other words,

$$G = \Big(\bigcap_{P \in \mathcal{P}} P\Big) \bigcap \Big(\bigcap_{N \in \mathcal{N}} !N\Big) = \Big(\bigcap_{P \in \mathcal{P}} P\Big) - \Big(\bigcup_{N \in \mathcal{N}} N\Big),$$

where "$\cap$" (respectively, "$\cup$") denotes the intersection (respectively, union) of the edge sets of two graphs and "!" denotes the complementation (with respect to W) of the edge set of a graph. We require that $\mathcal{P}$ contain at least one active network so that G is not formed solely by the intersection of the networks in $\mathcal{N}$; such a block is unlikely to be biologically interesting. We also require that $\mathcal{P}$ and $\mathcal{N}$ be disjoint so that G is not the empty graph. Requiring $\mathcal{P}$ and $\mathcal{N}$ to be maximal ensures that we include all the relevant active networks in the construction of G. These criteria imply that it is enough to specify $\mathcal{P}$ and $\mathcal{N}$ to compute G uniquely; we include G in the notation for a block for convenience and drop $\mathcal{P}$ and $\mathcal{N}$ when they are understood from the context. We refer to $\left(\bigcap_{P \in \mathcal{P}} P\right) - \left(\bigcup_{N \in \mathcal{N}} N\right)$ as the *formula* for the block.

Let $\mathcal{B}$ be a set of blocks. Given two blocks $(G_1, \mathcal{P}_1, \mathcal{N}_1)$ and $(G_2, \mathcal{P}_2, \mathcal{N}_2)$ in $\mathcal{B}$, we say that $G_1 \prec G_2$ if

(i) $\mathcal{P}_1 \subseteq \mathcal{P}_2$ and $\mathcal{N}_1 \subseteq \mathcal{N}_2$ or
(ii) $\mathcal{P}_1 \subseteq \mathcal{N}_2$ and $\mathcal{N}_1 \subseteq \mathcal{P}_2$.

We say that $G_1 < G_2$ if there is no block $G_3 \in \mathcal{B}$ such that $G_1 \prec G_3 \prec G_2$. The operators $<$ and $\prec$ represent partial orders between blocks, with $\prec$ being the transitive closure of $<$. Given a set $\mathcal{B}$ of blocks, let $\mathcal{D}_{\mathcal{B}}$ denote the directed acyclic graph representing the partial order $<$: each node in $\mathcal{D}_{\mathcal{B}}$ is a block in $\mathcal{B}$ and an edge connects two blocks related by $<$. For a block G, let $\sigma_G \in [0, 1]$ denote the statistical significance of G. We describe a method to compute this value in Section 3.4. We define a *network lego* to be a block $(G, \mathcal{P}, \mathcal{N}) \in \mathcal{B}$ such that $\sigma_G < \sigma_H$, for every $H \in \mathcal{B}$ where $G \prec H$ or $H \prec G$. In other words, $(G, \mathcal{P}, \mathcal{N})$ is a network lego if it is more statistically significant that blocks formed by combining any subset of $\mathcal{P}$ and $\mathcal{N}$ or by combining any superset of $\mathcal{P}$ and $\mathcal{N}$. In this sense, we claim that G is a building block of the active networks in $\mathcal{A}$.

3.2 Computing Active Networks

Given a gene expression dataset for a disease state or an experimental condition c, we use a variational filter to remove all genes whose expression profile has a small dynamic range from the wiring diagram W. More specifically, we log-transform and zero centre each gene's expression values. We discard a gene and all its interactions in the wiring diagram W if all the transformed expression values of the gene lie between -1 and 1 [30]. We deem the remaining genes to have responded in the condition. To each interaction $e = (g, h)$ remaining in W, we assign a weight equal to the absolute value of the Pearson's correlation coefficient of the expression profiles of the genes g and h, reasoning that this weight indicates how "active" the interaction is in the experimental condition. We discard edges whose weights are not statistically significant (based on a permutation test) at the 0.01 level. Let W_c be the resulting weighted interaction network. To mitigate the effect of isolated interactions in W_c, we search for pockets of concerted activity in W_c as follows. We define the *density* of a graph to be the total weight of the edges in the graph divided by the number of nodes in the graph. It is possible to find the subgraph of largest density using linear programming or parametric network flows [8]. We use a simpler greedy algorithm that finds a subgraph whose density is at least half the maximum density [8]. We repeatedly apply this approximation algorithm, remove the edges of the subgraph it computes, and re-invoke the algorithm on the remaining graph until the density of the remaining graph is less than the density of W_c. We deem the union of the computed dense subgraphs to be the active network A_c for the condition.

3.3 Computing Blocks

We reduce the problem of computing blocks to the problem of computing closed itemsets in a binary matrix [1,40]. We construct a binary matrix M where each column corresponds to an interaction in the wiring diagram W. The matrix M contains two rows for each active network $A \in \mathcal{A}$: the *positive* row corresponds to the interactions in A and the *negative* row to the interactions in $W - A$. In the positive row corresponding to A, we set a cell to be one if and only if the corresponding interaction belongs to A; this cell is zero in the negative row for A. Thus, M is a qualitative representation of which interactions are present in each active network and which are present in its complement.

In a binary matrix such as $\mathcal{B}$, an *itemset* (R, C) is a subset R of rows and a subset C of columns such that the sub-matrix spanned by these rows and columns only contains ones [1]. A *closed* itemset [40] is an itemset with the property that each row (respectively, column) not in the itemset contains a zero in at least one column (respectively, row) in the itemset. Therefore, it is not possible to add a row or a column to the itemset without introducing a zero into the corresponding sub-matrix. We can partition R into two subsets R_P and R_N where R_P (respectively, R_N) consists of all the positive (respectively, negative) rows in R. There is a natural mapping from a closed itemset (R, C) to a block $(G, \mathcal{P}, \mathcal{N})$:

1. G is the subgraph of W induced by the interactions corresponding to the columns in C;
2. $\mathcal{P}$ is the set of active networks corresponding to the rows in R_P; and
3. $\mathcal{N}$ is the set of active networks corresponding to the rows in R_N.

We compute closed itemsets in $\mathcal{B}$ to satisfy the maximality requirements in the definition of a block. We do not compute any itemsets where all rows correspond to complements of active networks, since such itemsets are unlikely to be biologically relevant (they correspond to blocks where $\mathcal{P} = \emptyset$). To construct closed itemsets, we use our implementation of the *Apriori* algorithm [1]. We have modified the original *Apriori* algorithm to construct closed itemsets. We convert each itemset to the corresponding block and formula. Finally, we connect the resulting set of blocks $\mathcal{B}$ in the DAG $\mathcal{D}_{\mathcal{B}}$ as per the partial order $<$.

3.4 Statistical Significance of a Block

To measure the statistical significance of a block, we construct an empirical distribution of block sizes. We repeatedly select a subset of rows uniformly at random from the binary matrix M, compute the columns common to these rows, and convert the resulting itemset into a block. We ensure that the random subset of rows does not contain an active network and its complement, since such a subset will trivially result in an itemset with zero columns. Given a block $(G, \mathcal{P}, \mathcal{N})$ computed in the real dataset, let m be the number of interactions in G. To estimate the statistical significance σ_G of $(G, \mathcal{P}, \mathcal{N})$, we only consider the distribution formed by random blocks $(H, \mathcal{P}', \mathcal{N}')$ where $|\mathcal{P}| = |\mathcal{P}'|$ and $|\mathcal{N}| = |\mathcal{N}'|$. We set σ_G to be the fraction of such blocks that have more than m interactions. Since the number of interactions in a block will decrease with an increase in $|\mathcal{P}|$ or in $|\mathcal{N}|$, these constraints ensure that we compare G with appropriate random blocks in order to estimate σ_G. We only retain blocks that are significant at the 0.01 level. We compute the DAG defined by these blocks. We perform two topological traversals of this DAG, one from the roots to the leaves and the other from the leaves to the roots, to identify the maximally-significant blocks. The resulting set of blocks are the network legos we desire to compute. Let $\mathcal{L}$ denote the set of network legos.

3.5 Stability and Recoverability Analysis

It is clear that the set $\mathcal{L}$ of network legos we compute depend on the active networks in $\mathcal{A}$. To assess this dependence, we modify a method for suggested by Segal et al. [30]. We remove each network $N \in \mathcal{A}$ in turn and recompute network legos from the set $\mathcal{A} - \{N\}$. Let $\mathcal{L}_N$ denote the resulting set of network legos. For each network lego L in $\mathcal{L}$, we compute the most similar network lego L' in $\mathcal{L}_N$ using the set-similarity measure $(|L \cap L'|/|L \cup L'|)$ and store this measure as $s_{L,N}$. Given a similarity threshold t, for each network lego L in $\mathcal{L}$, we compute the fraction of networks in $\mathcal{A}$ such that $s_{L,N} \geq t$. The higher this fraction is, the more resilient L is to perturbations in the input.

If the network legos in $\mathcal{L}$ are true building blocks of the active networks in $\mathcal{A}$ that they spring from, it should be possible to recover each active network in $\mathcal{A}$ from the network legos in $\mathcal{L}$. For each active network A, we define

$$\mathcal{L}_A = \{(G, \mathcal{P}, \mathcal{N}) \in \mathcal{L} | A \in \mathcal{P}\},$$

the set of network legos in $\mathcal{L}$ where A does not appear negated in the network lego. We compute the union of the network legos in $\mathcal{L}_A$ and the fraction of A's edge set that appears in the union. The larger this fraction is, the more "recoverable" A is from the computed network legos.

4 Results

We applied the algorithm described in the previous section to human data sets. We obtained a network of 31108 molecular interactions between 9243 human gene products by integrating the interactions in the IDSERVE database [24], the results of large scale yeast two-hybrid experiments [27,33], and 20 immune and cancer signalling pathways in the Netpath database (http://www.netpath.org). The IDSERVE database includes human curated interactions from BIND [4], HPRD [23], and Reactome [16], interactions predicted based on co-citations in article abstracts, and interactions that transferred from lower eukaryotes based on sequence similarity [19]. We derived functional annotations for the genes in our network from the Gene Ontology (GO) [3] and from MSigDB [35]. In addition, we annotated each Netpath interaction in our network with the name of the pathway it belonged to. We used these annotations to compute the functional enrichment of the nodes and edges in the network legos using the hypergeometric distribution with FDR correction.

4.1 ALL, AML, and MLL

We continue the analysis of ALL, AML, and MLL that we started in Section 1. Since the three leukaemias induce only 19 blocks, we did not compute the statistical significance of the blocks. Instead, we treated every block as a network lego. To assess the biological content of the results and to illustrate one type of analysis our approach facilitates, we computed functions enriched in the genes and interactions in the networks corresponding to the 19 formulae. Figure 1 demonstrates that the interactions in the KIT pathway are differentially enriched in the 19 networks. The darker the colour of a node, the more statistically significant is the enrichment of this pathway in the corresponding network. We first note that the only formulae enriched in this pathway are the ones that involve AML (and not the complement of AML). The statistical significance is the lowest (FDR-corrected p-value 3.5×10^{-7}) for the formula AML $\cap$!ALL $\cap$!MLL, indicating that this pathway may be activated in AML and not in ALL or in MLL. Evidence in the literature supports this conclusion. The c-KIT receptor is activated in almost all subtypes of AML [29]. Similarly, Schnittger et al. [28] report that "mutations in codon D816 of the KIT gene represent a recurrent genetic alteration in AML". We note that gain-of-function mutations in c-Kit have been observed in many human cancers [9]. Our analysis only suggests that in the context of ALL, AML, and MLL, the KIT pathway may be activated only in AML.

4.2 Human Stresses

We computed network legos by applying our methods to the human interaction network and the gene expression responses of HeLa cells and primary human lung fibroblasts to heat shock, endoplasmic reticulum stress, oxidative stress, and crowding [21]. The dataset we analysed includes transcriptional measurements obtained by Whitfield et al. [39] for studying cell cycle arrest by using a double thymidine block or with a thymidine-nocodazole block. Overall, the dataset contains 13 distinct stresses over the two cell types. The authors note that each type of stress resulted in a distinct response and that there was no general stress response unlike in the case of *S. cerevisiae* [11]. Therefore, this dataset poses a challenge to our system. Can we find network legos that combine active networks for multiple stresses? In this paper, we focus on the topological and quantitative aspects of our results.

The number of genes in the 13 active networks we computed ranged from 165 (for crowding of WI38 cells) to 1148 (for the thymidine-nocodazole block) with an average of 684 genes per active network. The number of interactions ranged from 257 to 3667 with an average of 1874 interactions per active network. Theoretically, we can compute 1586131 ($3^{13} - 2^{13}$) blocks involving 13 distinct active networks. Our method computed 444201 blocks, indicating that the remaining combinations of active networks are not closed or yield blocks without any interactions. We computed a null distribution of block sizes using a million random samples. Of the 444201 blocks, 12386 blocks were statistically significant at the 0.01 level. We identified 143 network legos in the DAG induced by the relation $<$ on these blocks. We observed that all but one of the 143 network legos involved at least six distinct active networks, indicating that these network legos are not the result of combining a small number of active networks. The following table displays the distribution of the number of legos involving k conditions, where $5 \leq k \leq 12$. Interestingly, no network lego involved all 13 active networks.

#conditions	5	6	7	8	9	10	11	12
#legos	1	6	10	36	34	20	28	8

In light of the statement by Murray et al. [21] that each type of stress resulted in a distinct response, it is important to ask whether most of our network legos primarily involve complemented active networks. Over all network legos $(G, \mathcal{P}, \mathcal{N})$, we counted the total size of the positive active networks (those in the sets $\mathcal{P}$) and the total size of the negative active networks (those in the sets $\mathcal{N}$). Interestingly, more than 40% of the active networks appeared in the positive sets, indicating that the network legos we found were not primarily focussed on what made the stresses unique. Rather, a large fraction of the network legos represented features common to multiple stresses. The active networks that appeared most often in the positive form were the two treatments that resulted in cell cycle arrest. Each participated in as many as 119 network legos. In most of these network legos, almost all the other active networks appeared in complemented form. The complements of the cell cycle arrest active networks did not participate in any network legos. This observation indicates that the interactions activated by cell cycle arrest are quite distinct from the network of interactions activated by the other stresses.

We obtained very good stability and recovery results. Upon the removal of each active network, we were able to recompute each network lego with at least 95% fidelity. We were also able to recover 11 active networks with 100% accuracy by composing network legos. The two active networks we could not recover completely were the double thymidine network (97% recovery) and the thymidine-nocodazole network (86% recovery). When we tested the recoverability of active networks using the blocks at the roots of the DAG connecting statistically-significant blocks, the recovery for these two active networks dropped to 85% and 75% respectively. This result underscores the fact that identifying network legos as those that are maximally statistically-significant in the DAG of blocks is a useful concept.

Since the cell-cycle treatments resulted in active networks that were quite distinct from those for the other stresses, we repeated the analysis after removing the double thymidine and thymidine-nocodazole active networks. The 11 remaining active networks yielded only 77117 blocks (out of the 175099 possible). Of these, 1629 blocks were statistically significant. These blocks yielded 15 network legos. This much smaller set of network legos suggests that a number of the 143 network legos in the complete analysis were needed to capture unique aspects of the cell cycle active networks. Each network lego involved at least seven active networks. No network lego involved all 11 stresses. The ratio of total size of the positive active networks and the negative active networks in the 15 network legos was 1:2. As many as eight network legos had only one active network in $\mathcal{P}$—the fibroblast active network upon treatment with menadione—indicating that this stress results in an active network that is quite unique compared to the other 10 active networks. Of the 11 active networks, we recovered five with complete accuracy and one with 99.9% accuracy. We recovered the remaining with accuracies ranging from 71% to 92%. Taken together, these statistics indicate that the network legos we detect are indeed building blocks of the networks activated in response to the stresses studied by Murray et al. [21].

5 Discussion

We have presented a novel approach for combining gene expression data sets with multi-modal interaction networks. This combination provides a dynamic view of the interactions that are activated in the wiring diagram under different conditions. We represent similarities and differences between the network of interactions activated in response to different cell states both as a set theoretic formula involving cell states and as a network lego, a functional module of co-expressed molecular interactions. A novel contribution of our work is the DAG that relates all cell states (and the active networks corresponding to the cell states). This DAG provides a high-level abstract view of the similarities and differences between cell states.

Acknowledgments. We thank Naren Ramakrishnan for useful suggestions on the algorithm for computing blocks. We thank Aravind Subramanian for providing us an updated version of the gene sets in MSigDB.

References

1. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the Twentieth International Conference on Very Large Databases*, pages 487–499, Santiago, Chile, 1994.
2. S. Armstrong, J. Staunton, L. Silverman, R. Pieters, M. den Boer, M. Minden, S. Sallan, E. Lander, T. Golub, and S. Korsmeyer. MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nat Genet*, 30(1):41–7, 2002.

3. M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene Ontology: tool for the unification of biology. the Gene Ontology Consortium. *Nat Genet*, 25(1):25–9, 2000.
4. G. D. Bader, D. Betel, and C. W. V. Hogue. BIND: the Biomolecular Interaction Network Database. *Nucleic Acids Res*, 31(1):248–50, 2003.
5. Z. Bar-Joseph, G. K. Gerber, T. I. Lee, N. J. Rinaldi, J. Y. Yoo, F. Robert, D. B. Gordon, E. Fraenkel, T. S. Jaakkola, R. A. Young, and D. K. Gifford. Computational discovery of gene modules and regulatory networks. *Nat Biotechnol*, 21(11):1337–42, 2003.
6. K. Basso, A. Margolin, G. Stolovitzky, U. Klein, R. Dalla-Favera, and A. Califano. Reverse engineering of regulatory networks in human B cells. *Nat Genet*, 37(4):382–90, 2005.
7. S. Bergmann, J. Ihmels, and N. Barkai. Similarities and differences in genome-wide expression data of six organisms. *PLoS Biol*, 2(1):E9, 2003.
8. M. Charikar. Greedy approximation algorithms for finding dense components in graphs. In *Proceedings of APPROX*, 2000.
9. D. Cozma and A. Thomas-Tikhonenko. Kit-Activating Mutations in AML: Lessons from PU.1-Induced Murine Erythroleukemia. *Cancer Biol Ther*, 5(6):579–81, 2006.
10. D. di Bernardo, M. J. Thompson, T. S. Gardner, S. E. Chobot, E. L. Eastwood, A. P. Wojtovich, S. J. Elliott, S. E. Schaus, and J. J. Collins. Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks. *Nat Biotechnol*, 23(3):377–83, 2005.
11. A. P. Gasch, P. T. Spellman, C. M. Kao, M. B. Eisen, G. Storz, D. Botstein, and P. O. Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Mol Biol Cell*, 11(12):4241–57, 2000.
12. J. Han, N. Bertin, T. Hao, D. Goldberg, G. Berriz, L. Zhang, D. Dupuy, A. Walhout, M. Cusick, F. Roth, and M. Vidal. Evidence for dynamically organized modularity in the yeast protein-protein interaction network. *Nature*, 430(6995):88–93, 2004.
13. L. Hartwell, J. Hopfield, S. Leibler, and A. Murray. From molecular to modular cell biology. *Nature*, 402(6761 Suppl):C47–52, 1999.
14. H. Hu, X. Yan, Y. Huang, J. Han, and X. J. Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics*, 21 Suppl 1:i213–i221, 2005.
15. T. Ideker, O. Ozier, B. Schwikowski, and A. F. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics*, 18 Suppl 1:S233–40, 2002.
16. G. Joshi-Tope, M. Gillespie, I. Vastrik, P. D'Eustachio, E. Schmidt, B. de Bono, B. Jassal, G. Gopinath, G. Wu, L. Matthews, S. Lewis, E. Birney, and L. Stein. Reactome: a knowledgebase of biological pathways. *Nucleic Acids Res*, 33(Database issue):D428–32, 2005.
17. A. R. Joyce and B. O. Palsson. The model organism as a system: integrating 'omics' data sets. *Nat Rev Mol Cell Biol*, 7(3):198–210, 2006.
18. H. Lee, A. Hsu, J. Sajdak, J. Qin, and P. Pavlidis. Coexpression analysis of human genes across many microarray data sets. *Genome Res*, 14(6):1085–94, 2004.
19. B. Lehner and A. G. Fraser. A first-draft human protein-interaction map. *Genome Biol*, 5(9):R63, 2004.
20. N. Luscombe, M. Babu, H. Yu, M. Snyder, S. Teichmann, and M. Gerstein. Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, 431(7006):308–12, 2004.

21. J. I. Murray, M. L. Whitfield, N. D. Trinklein, R. M. Myers, P. O. Brown, and D. Botstein. Diverse and specific gene expression responses to stresses in cultured human cells. *Mol Biol Cell*, 15(5):2361–74, 2004.
22. C. L. Myers, D. Robson, A. Wible, M. A. Hibbs, C. Chiriac, C. L. Theesfeld, K. Dolinski, and O. G. Troyanskaya. Discovery of biological networks from diverse functional genomic data. *Genome Biol*, 6(13):R114, 2005.
23. S. Peri, J. Navarro, R. Amanchy, T. Kristiansen, C. Jonnalagadda, V. Surendranath, V. Niranjan, B. Muthusamy, T. Gandhi, M. Gronborg, N. Ibarrola, N. Deshpande, K. Shanker, H. Shivashankar, B. Rashmi, M. Ramya, Z. Zhao, K. Chandrika, N. Padma, H. Harsha, A. Yatish, M. Kavitha, M. Menezes, D. Choudhury, S. Suresh, N. Ghosh, R. Saravana, S. Chandran, S. Krishna, M. Joy, S. Anand, V. Madavan, A. Joseph, G. Wong, W. Schiemann, S. Constantinescu, L. Huang, R. Khosravi-Far, H. Steen, M. Tewari, S. Ghaffari, G. Blobe, C. Dang, J. Garcia, J. Pevsner, O. Jensen, P. Roepstorff, K. Deshpande, A. Chinnaiyan, A. Hamosh, A. Chakravarti, and A. Pandey. Development of human protein reference database as an initial platform for approaching systems biology in humans. *Genome Res*, 13(10):2363–71, 2003.
24. A. K. Ramani, R. C. Bunescu, R. J. Mooney, and E. M. Marcotte. Consolidating the set of known human protein-protein interactions in preparation for large-scale mapping of the human interactome. *Genome Biol*, 6(5):R40, 2005.
25. D. Rhodes, J. Yu, K. Shanker, N. Deshpande, R. Varambally, D. Ghosh, T. Barrette, A. Pandey, and A. Chinnaiyan. Large-scale meta-analysis of cancer microarray data identifies common transcriptional profiles of neoplastic transformation and progression. *Proc Natl Acad Sci U S A*, 101(25):9309–14, 2004.
26. D. R. Rhodes, S. Kalyana-Sundaram, V. Mahavisno, T. R. Barrette, D. Ghosh, and A. M. Chinnaiyan. Mining for regulatory programs in the cancer transcriptome. *Nature Genetics*, 37(6):579–583, May 2005.
27. J. Rual, K. Venkatesan, T. Hao, T. Hirozane-Kishikawa, A. Dricot, N. Li, G. Berriz, F. Gibbons, M. Dreze, N. Ayivi-Guedehoussou, N. Klitgord, C. Simon, M. Boxem, S. Milstein, J. Rosenberg, D. Goldberg, L. Zhang, S. Wong, G. Franklin, S. Li, J. Albala, J. Lim, C. Fraughton, E. Llamosas, S. Cevik, C. Bex, P. Lamesch, R. Sikorski, J. Vandenhaute, H. Zoghbi, A. Smolyar, S. Bosak, R. Sequerra, L. Doucette-Stamm, M. Cusick, D. Hill, F. Roth, and M. Vidal. Towards a proteome-scale map of the human protein-protein interaction network. *Nature*, 437(7062):1173–8, 2005.
28. S. Schnittger, T. M. Kohl, T. Haferlach, W. Kern, W. Hiddemann, K. Spiekermann, and C. Schoch. KIT-D816 mutations in AML1-ETO-positive AML are associated with impaired event-free and overall survival. *Blood*, 107(5):1791–9, 2006.
29. S. Schwartz, A. Heinecke, M. Zimmermann, U. Creutzig, C. Schoch, J. Harbott, C. Fonatsch, H. Loffler, T. Buchner, W. D. Ludwig, and E. Thiel. Expression of the C-kit receptor (CD117) is a feature of almost all subtypes of de novo acute myeloblastic leukemia (AML), including cytogenetically good-risk AML, and lacks prognostic significance. *Leuk Lymphoma*, 34(1-2):85–94, 1999.
30. E. Segal, N. Friedman, D. Koller, and A. Regev. A module map showing conditional activity of expression modules in cancer. *Nat Genet*, 36(10):1090–8, 2004.
31. E. Segal, M. Shapira, A. Regev, D. Botstein, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat Genet*, 34(2):166–76, 2003.
32. R. Sharan and T. Ideker. Modeling cellular machinery through biological network comparison. *Nat Biotechnol*, 24(4):427–33, 2006.

33. U. Stelzl, U. Worm, M. Lalowski, C. Haenig, F. Brembeck, H. Goehler, M. Stroedicke, M. Zenkner, A. Schoenherr, S. Koeppen, J. Timm, S. Mintzlaff, C. Abraham, N. Bock, S. Kietzmann, A. Goedde, E. Toksoz, A. Droege, S. Krobitsch, B. Korn, W. Birchmeier, H. Lehrach, and E. Wanker. A human protein-protein interaction network: a resource for annotating the proteome. *Cell*, 122(6):957–68, 2005.
34. J. M. Stuart, E. Segal, D. Koller, and S. K. Kim. A gene-coexpression network for global discovery of conserved genetic modules. *Science*, 302(5643):249–55, 2003.
35. A. Subramanian, P. Tamayo, V. Mootha, S. Mukherjee, B. Ebert, M. Gillette, A. Paulovich, S. Pomeroy, T. Golub, E. Lander, and J. Mesirov. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proc Natl Acad Sci U S A*, 2005.
36. A. Tanay, R. Sharan, M. Kupiec, and R. Shamir. Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genomewide data. *Proc Natl Acad Sci U S A*, 101(9):2981–6, 2004.
37. A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. In *Proceedings of ISMB 2002*, pages S136–S144, 2002.
38. A. Tanay, I. Steinfeld, M. Kupiec, and R. Shamir. Integrative analysis of genome-wide experiments in the context of a large high-throughput data compendium. *Molecular Systems Biology*, 1(1):msb4100005–E1–msb4100005–E10, March 2005.
39. M. L. Whitfield, G. Sherlock, A. J. Saldanha, J. I. Murray, C. A. Ball, K. E. Alexander, J. C. Matese, C. M. Perou, M. M. Hurt, P. O. Brown, and D. Botstein. Identification of genes periodically expressed in the human cell cycle and their expression in tumors. *Mol Biol Cell*, 13(6):1977–2000, 2002.
40. M. J. Zaki and C.-J. Hsiao. CHARM: An efficient algorithm for closed itemset mining. In *SIAM International Conference on Data Mining*, pages 457–473, 2002.
41. X. J. Zhou, M. C. Kao, H. Huang, A. Wong, J. Nunez-Iglesias, M. Primig, O. M. Aparicio, C. E. Finch, T. E. Morgan, and W. H. Wong. Functional annotation and network reconstruction through cross-platform integration of microarray data. *Nat Biotechnol*, 23(2):238–43, 2005.

An Efficient Method for Dynamic Analysis of Gene Regulatory Networks and *in silico* Gene Perturbation Experiments

Abhishek Garg, Ioannis Xenarios[1], Luis Mendoza[2], and Giovanni DeMicheli

Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland
[1] Merck Serono, Geneva, Switzerland
[2] Instituto de Investigaciones Biomédicas, UNAM, Mexico
{abhishek.garg,giovanni.demicheli}@epfl.ch,
ioannis.xenarios@merckserono.net, lmendoza@biomedicas.unam.mx

Abstract. With the increasing availability of experimental data on gene-gene and protein-protein interactions, modeling of gene regulatory networks has gained a special attention lately. To have a better understanding of these networks it is necessary to capture their dynamical properties, by computing its steady states. Various methods have been proposed to compute steady states but almost all of them suffer from the state space explosion problem with the increasing size of the networks. Hence it becomes difficult to model even moderate sized networks using these techniques. In this paper, we present a new representation of gene regulatory networks, which facilitates the steady state computation of networks as large as 1200 nodes and 5000 edges. We benchmarked and validated our algorithm on the T helper model from [8] and performed *in silico* knock out experiments: showing both a reduction in computation time and correct steady state identification.

1 Introduction

The face of biological research has evolved at an alarming rate over the last two decades. From a one-gene/one-protein analysis it has borne witness to a multitude of technologies that allows us to capture and integrate a vast amount of information generated by high throughput methods such as DNA microarrays, siRNA knock-down and protein-protein interactions. While a wealth of information is present on the interaction of the genes and proteins, the exact stoichiometry and precise kinetics still evades our technologies and understanding. In such situation, one could either wait to gather the crucial information on the precise biochemical processes or choose to model the flow of information in genetic regulatory networks. We chose the latter as we think that the information is sufficient already to identify qualitative behavior of the studied biological system. We also claim that enabling such kind of approaches should further the understanding of the design and identifications of keys elements that dictate cell fate.

The methodology presented here is an improvement of the methods described by [8] to model dicrete regulatory networks and proposes to use a data structure called binary decision diagram (BDD) to represent and manipulate Boolean networks. This data transformation enables the compact representation of the state space of the network and their efficient dynamic analysis. BDDs have primarily been used in several other applications like logic synthesis and testing in the field of Electronic Design Automation [21,22] and model checking [23,24]. In this paper, we show their application on biological regulatory networks. Some work on modelling the gene regulatory networks using the formal methods have been introduced in [27,28,29].

We use the already published *T helper cell* regulatory network [8,7] as a framework to validate our approach and show that our software, **GenYsis** finds all steady states and correctly identifies the outcome of gene perturbation experiments. We show that GenYsis scales well with the size of the network and can compute cell states in the network with size over 1000 nodes in reasonable time using modest computing resources.

2 Binary Decision Diagrams

2.1 Introduction

A Binary Decision Diagram(BDD) is a directed acyclic graph consisting of a root node, intermediate decision nodes and two terminal nodes, namely **0**-terminal and **1**-terminal. BDDs can be used for representing Boolean functions. Each variable of the function is represented as a decison node of the graph. Each decision node has two outgoing edges to represent evaluation of variable to **1** and **0**. All paths from root node to **1**-terminal gives the variable evaluations for which the function is true. There might be some variables missing in some of the paths. These variables have a “Don’t Care” evaluation, i.e. they can take either 0 or 1 value.

A simple BDD that represents the Boolean function $f = (a$ **AND** $b)$ **OR** c is shown in Figure 1.

It has three paths from root node (node f) to **1**-terminal node. For the path, $a \xrightarrow{0} c \xrightarrow{1} 1$, two possible assignments $(a = 0, b = 1, c = 1)$ and $(a = 0, b = 0, c = 1)$ leads to 1-terminal from root node f. Similarly, the path $a \xrightarrow{1} b \xrightarrow{1} 1$,

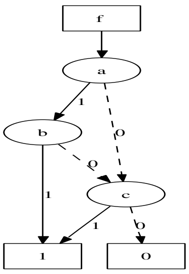


Fig. 1. BDD for the function $f = (a \wedge b) \vee c$

represents the assignments $(a = 1, b = 1, c = 1)$ and $(a = 1, b = 1, c = 0)$. Finally the last path $a \xrightarrow{1} b \xrightarrow{0} c \xrightarrow{1} 1$ representing $(a = 1, b = 0, c = 1)$ complete the five possible TRUE evaluations for the Boolean formula f.

Here, we use Reduced Ordered BDDs (ROBDDs), which are the compact reduced form of BDDs. For the sake of brevity whenever we say BDD in this paper, we are referring to ROBDDs.

The representation of Boolean functions as BDDs is memory efficient as isomorphic subgraphs can be shared by multiple nodes. The size of the BDD scales well in most cases with the size of the Boolean function and all the logic operations like AND, OR, Existential quantification, Universal quantification, etc. can be performed in polynomial (with the size of the BDD) time on this data structure [1]. This implicit representation does not require the explicit construction of a truth table and can be directly constructed from the Boolean function. Further details on BDD construction and logic operations on them is outside the scope of this paper, interested readers can find details in [1,2,3].

There are many existing packages that can be used for working with BDDs like CUDD, CMUBDD, etc. In this paper we use the CUDD package [20], which is the most efficient package for BDD representation and evaluation.

2.2 Representation of Gene Regulatory Networks

In this section we show how gene regulatory networks can be mapped to BDDs. We start with the representation of regulatory networks as Boolean functions and then we use these functions to construct corresponding BDD representation.

Given a gene regulatory network, the state of a node (or gene) i at time t is represented with the Boolean variable $x_i(t)$. To evaluate the evolution in time of each node, it is necessary to describe the state of each node at time $t + 1$ as a function of state of those nodes acting as input at time t [8]:

$$x_i(t+1) = \left(\bigvee_{j=1}^{m} x_j^a(t) \right) \wedge \neg \left(\bigvee_{j=1}^{n} x_j^{in}(t) \right) \tag{1}$$

$$x_i(t+1) = \left(\bigvee_{j=1}^{m} x_j^a(t) \right) \tag{2}$$

$$x_i(t+1) = \neg \left(\bigvee_{j=1}^{n} x_j^{in}(t) \right) \tag{3}$$

$$x_j \;\in\; \{0,1\}$$
$$x_m^a \textit{ and } x_n^{in} \textit{ are the set of activators and inhibitors of } x_i$$
$$\wedge \textit{ and } \vee \textit{ represent logical AND and OR}$$

Equation 1 is used if the gene i has both activators and inhibitors. Equation 2 is used if the gene i has only activators and equation 3, if there are only inhibitors.

In equation 1, inhibitors are strong enough to change the state of a gene from 1 to 0, while activators can change the state from 0 to 1 if and only if there are no inhibitors acting on that gene.

A snapshot of the activity level of all the genes in the network at a time t is called the state of the network. The state of the network is represented by a Boolean vector of size N (number of genes in the network). Each bit of this vector represents whether the gene is active or inactive. Another Boolean vector of size N is used to represent the status of the genes at next step. We call the previous vector as *present state*(V_t) and latter one as *next state* (V_{t+1}).

The transition between states of the network can be either *synchronous* and *asynchronous*. If the transitions are synchronous, all the genes change their state at the same time point. If the transition is asynchronous, atmost one gene can change its state between two consecutive states. Biologically, it is more realistic to assume that genes have different response times, and hence an asynchronous network might seem more realistic. In this paper, we use the asynchronous model to represent state transition of the regulatory network and assume that time points are close enough, so that only one gene can make a transition at each time point.

Now we shall see how the Boolean functions in equations 1-3 can be used to construct a BDD representation. Let $T_i(V_t, V_{t+1})$ be the BDD representing transition of gene i from V_t to V_{t+1} and $T(V_t, V_{t+1})$ be the BDD representing the transition from state of the network at time t to state at time $t+1$. The relation between $T_i(V_t, V_{t+1})$ and $T(V_t, V_{t+1})$ is given by equation 4. Equation 4 says that all genes make asynchronous transitions and state of the network at time t can have multiple successor states.

$$T(V_t, V_{t+1}) = T_0(V_t, V_{t+1}) \vee T_1(V_t, V_{t+1}) \vee ... \vee \ T_N(V_t, V_{t+1}) \tag{4}$$

To impose the constraint that two consecutive states differ in atmost one gene evaluation, we define $T_i(V_t, V_{t+1})$ in equation 5, which states that for gene i, its evaluation at the next time step $v_i'(\in V_{t+1})$ and function $f_i(V_t)(= x_i(t+1))$ have the same value, and all the other genes remain at their activation level from the previous time step.

$$T_i(V_t, V_{t+1}) = (v_i' \leftrightarrow f_i(V_t)) \wedge \bigwedge_{j \neq i} (v_j' \leftrightarrow v_j) \tag{5}$$

Let us go through a small example to have a better understanding of the process.

Example 1. In figure 2, gene 'A' has an auto-activation and is inhibited by the presence of gene 'C'. Gene 'B' is activated by the presence of gene 'A' and presence of gene 'B' inhibits gene 'C'. The present state and next state vectors are given by $V_t = \{a, b, c\}$ and $V_{t+1} = \{a', b', c'\}$ respectively. Boolean functions describing this small network are given by:

$$a' = a \wedge \neg c \tag{6}$$

$$b' = a \tag{7}$$

$$c' = \neg b \tag{8}$$

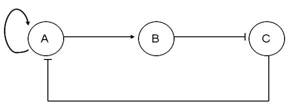


Fig. 2. An example of Gene Regulatory Network

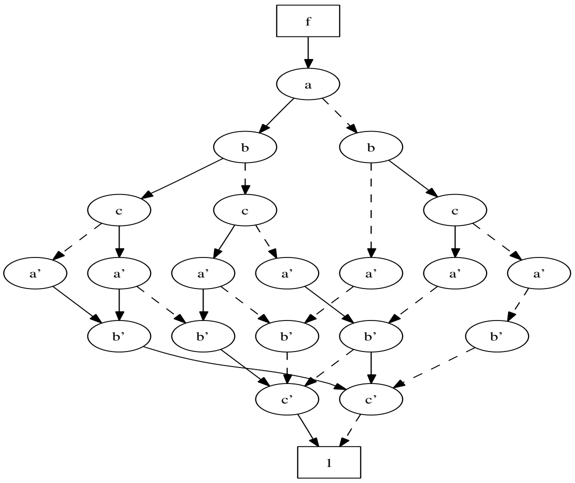


Fig. 3. BDD representing the state space of example in figure 2. The dashed edges represent **0** evaluation of the variables and the solid edges represent the **1** evaluation. For clarity, edges going to **0**-terminal are not shown in this figure.

Corresponding transition relations for each gene is then given by:

$$T_0(V_t, V_{t+1}) = a' \leftrightarrow (a \wedge \neg c) \wedge b' \leftrightarrow (b) \wedge c' \leftrightarrow (c) \tag{9}$$

$$T_1(V_t, V_{t+1}) = b' \leftrightarrow (a) \wedge a' \leftrightarrow (a) \wedge c' \leftrightarrow (c) \tag{10}$$

$$T_2(V_t, V_{t+1}) = c' \leftrightarrow (\neg b) \wedge a' \leftrightarrow (a) \wedge b' \leftrightarrow (b) \tag{11}$$

The BDD representation for $T(V_t, V_{t+1})$ by using equations 9-11 in equation 4 is shown in figure 3. For clarity in figure 3, edges pointing to **0**-terminal are removed. This BDD represents all the possible state transitions of the network. To find the immediate successor states of a given state of the network(say for example $a = 1, b = 0, c = 1$), the following steps can be performed on BDD $T(V_t, V_{t+1})$:

1. Assign initial activity levels to the genes a, b,and c (i.e. $v_i \in V_t$).
2. Remove all outgoing edges which do not satisfy the evaluations in step 1.
3. Find all the paths from root node to **1**-terminal and for each path only print variables in set V_{t+1}.
4. Swap variable names from the set V_{t+1} with the corresponding variable names in the set V_t, on all the printed paths. □

The steps given in above example can be implemented by using efficient logic functions such as "AND" and "Existential Quantify" along with the expressive power of BDDs [1,2], as follows:

1. Construct a BDD 'X' which represents the initial state.
2. Take logical 'AND' of BDD 'X' with the BDD $T(V_t, V_{t+1})$.
3. Existentially quantify out variables in V_t from the resulting BDD.
4. Swap variables $v'_i \in V_{t+1}$ with $v_i \in V_t$ in the BDD got from the last step.

The BDD formed after executing step 4 represents all the immediate successor states from a given initial state.

3 Computing the Steady States

In this section, we will see how to efficiently compute steady states on the BDD representation of the gene regulatory networks. But first we shall give some definitions that we are going to use in the rest of the section. Let, f be the state transition function.

Definition 1. *Forward image, $I_f(S(V_t))$ is the set of immediate successors of the states in $S(V_t)$ on the state transition graph.*

Definition 2. *Backward image, $I_b(S(V_t))$ is the set of immediate predecessors of the states in $S(V_t)$ on the state transition graph.*

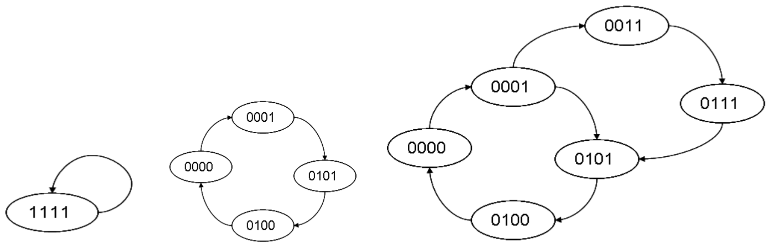
Definition 3. *Forward reachable states $FR(S_0)$ from the states S_0 are all the states that can be reached from S_0 by iteratively computing forward image in the transition relation $T(V_t, V_{t+1})$ until no new states are reachable.*

Definition 4. *Backward reachable states, $BR(S_0)$, are all the states in $T(V_t, V_{t+1})$ whose forward reachable states contain S_0.*

Definition 5. *Steady State is the set of states $SS(V_t)$ having the following two properties:*

1. *Forward image $I_f(SS(V_t))$ is same as $SS(V_t)$.*
2. *For all the states in $SS(V_t)$, if that state is reached once, then the probability of revisiting that state is one. [19]*

The first property of a steady state implies that there are only three possible variants of steady states as shown in figure 4. The second property of steady state ensures that there are only simple cycles(figure 4(a) and 4(b)) in the set $SS(V_{ps})$ and invalidates the third kind of steady state(figure 4(c)) with complex loops as some of the states in this loop might not be revisited. The first property is contained in the second property of steady states. An efficient algorithm can be designed for finding steady states that satisfy the first property, though any efficient algorithm satisfying property 2 is not known yet. Here we use the modified algorithm by [18] for computing the set of steady states satisfying property 1 and then remove the false steady states of type III (in figure 4) from that set.



(a) Self Loop (b) Simple Loop (c) Complex Loop(false steady state)

Fig. 4. Different types of steady states

Algorithm [18] for computing steady states is based on two main theorems. For the sake of completeness of the paper, we present these theorems and algorithms here again. Proof of these theorems can be found in [18].

Theorem 1. *A state $i \in S$ is a steady state if and only if $FR(i) \subseteq BR(i)$. State i is transient otherwise.*

Theorem 2. *If state $i \in S$ is transient, then states in $BR(i)$ are all transient. If state i is steady, then all the states in $FR(i)$ are steady states. In the latter case set $\{BR(i) - FR(i)\}$ are all transient.*

Based on these two theorems, the algorithm for steady state computation is given in Algorithm 2. Algorithm 2 uses the functions *forward_set()* and *backward_set()* for computing forward reachable ($FR(S)$) and backward reachable ($BR(S)$) states respectively. These functions are given in Algorithm 1. In Algorithm 1, FS^k and RS^k, are the frontier set and reachable set respectively in the k^{th} iteration of the while loop. Frontier set (Backward set) in iteration $k+1$, is computed by taking the forward (backward) image of the frontier (backward) set in the k^{th} iteration and removing from this image set the states that have already been explored in previous iterations (which are stored in Reached Set). Reached Set is updated by adding the new states from frontier(backward) set. This process is iterated until no new states can be added to Reached Set. The final Reached Set represents the forward (backward) reachable set from the set of initial states S_0.

The Algorithm 2 uses Theorems 1 and 2. In line 5 of Algorithm 2, a prospective steady state is selected from the state space T' and forward and backward reachable sets from this seed state are computed in lines 6 and 7. Then Theorem 1 as implemented in line 8, checks if the seed state is a steady state. If the seed state is indeed a steady state then using Theorem 2 (as implemented in lines 9-12), all the states in forward reachable set are declared steady states in line 9 and rest of the states in backward reachable set are declared transient states in line 10. Otherwise, the seed state and all the other states in backward reachable set are declared transient in line 12. In line 13, state space is reduced by removing the states that have already been tested for reachability and the process is repeated to find another steady state on the reduced state space. This process is

Algorithm 1. Computing Forward and Backward reachable sets

```
1  $forward\_set(S_0, T)$
2  /* $backward\_set(S_0, T)$ */
3  begin
4      $RS^{(0)} \longleftarrow \emptyset, FS^{(0)} \longleftarrow \{S_0\}$
5      $k \longleftarrow 0$
6      while $FS^{(k)} \neq \emptyset$ do
7          $FS^{(k+1)} = I_f(FS^{(k)})(V_{t+1} \leftarrow V_t) \wedge \overline{RS^{(k)}}$
8          /* $FS^{(k+1)} = I_b(FS^{(k)})(V_t \leftarrow V_{t+1}) \wedge \overline{RS^{(k)}}$ */
9          $RS^{(k+1)} = RS^{(k)} \vee FS^{(k+1)}$
10         $k \longleftarrow k + 1$
11     return $(FR(S_0) \longleftarrow RS^{(k)})$
12     /* return $(BR(S_0) \longleftarrow RS^{(k)}(V_{t+1} \leftarrow V_t))$ */
13 end
```

iterated until the whole state space is explored (i.e. until $T \neq \emptyset$). Since in each iteration, states in backward reachable set are removed from the state space, the size of the state space reduces in each iteration. The number of iterations also depends upon how the seed state is selected.

Function initial_state() in Algorithm 2 selects a prospective steady state from the given state space T'. In this function (implemented in lines 17-25), a BDD representing a random path from the root node to 1-terminal, is selected in line 17. The variables $v_i \in V_{t+1}$ on this path P are removed (line 18) and the resulting BDD is called the intial state,s. Forward reachable set from this random initial state is then computed in lines 19-24. During the forward set computation, when the frontier set evaluates to $\emptyset$ in iteration k, a random state is taken from the frontier set in iteration $k - 1$ and returned as the seed state. The motivation behind this function is that a state in the last frontier set is more likely to be a steady state then a random state in the state space T. This function differs from the one given in [18], in which the authors propose to do forward reachability until a user-defined depth (i.e. k is taken as input). But in our experience the number of iterations of while loop in line 4 of Algorithm 2 can be reduced by a large factor if we do complete forward reachability and select a state from the frontier set in the last iteration as compared to selecting from one of the intermediate iterations. This is because any state from frontier set of k^{th} iteration should have a larger backward reachable set then any other state in previous $k - 1$ iterations. And larger the size of backward reachable set, smaller the number of iterations required to exhaust the state space.

The Algorithm 2 gives the set of steady states satisfying property 1 as mentioned in the definition of steady states. Pseudocode for doing the complete dynamic analysis is given in Algorithm 3, which uses the function *isFalseLoop()* to check for false steady states. In the function *isFalseLoop()*, given a steady state S, a random state s_0 is selected from this set S and the image of s_0 is computed in line 14. In lines 15-19, we test if the immediate successor state of

Algorithm 2. Steady State Algorithm

```
1  $all\_Steady\_States(T)$
2  begin
3      $T' \longleftarrow T$
4      while $T' \neq \emptyset$ do
5          $s \longleftarrow initial\_state(T')$
6          $FR(s) \longleftarrow forward\_set(s, T')$
7          $BR(s) \longleftarrow backward\_set(s, T')$
8          if $FR(s) \wedge \overline{BR(s)} = \emptyset$ then
9              report $FR(s)$ as a steady state
10             report $BR(s) \wedge \overline{FR(s)}$ as all transient states
11         else
12             report $s \vee BR(s)$ as all transient states
13         $T' \longleftarrow T' \wedge \overline{s \vee BR(s)}$
14 end
15 $initial\_state(T)$
16 begin
17     $P = random\_path\_to\_1\_node(T)$
18     $s(V_t) = \exists_{v \in V_t} P$
19     $RS^{(0)} \longleftarrow \emptyset, FS^{(0)} \longleftarrow \{s\}$
20     $k \longleftarrow 0$
21     while $FS^{(k)} \neq \emptyset$ do
22         $FS^{(k+1)} = I_f(FS^{(k)})(V_{t+1} \leftarrow V_t) \wedge \overline{RS^{(k)}}$
23         $RS^{(k+1)} = RS^{(k)} \vee FS^{(k+1)}$
24         $k \longleftarrow k + 1$
25     $s \longleftarrow random\_path\_to\_1\_node(FS^{(k-1)})$
26     return $s$
27 end
```

this initial state is a single state or a set of state. To do this, a random path(or the state) from the image set is computed(line 15) and removed from this set in line 16. If the resulting set is not empty (line 17), then the given steady state is declared false. Otherwise the lines 13-21 are iterated with the frontier set and the reached set being updated as in line 20 and 21. Function *isFalseLoop()* removes all the type III steady states, because these steady states will always contain one or more states with two possible immediate successors. All the other steady states are simple loops and are reported genuine by this function.

3.1 Results

We have implemented our software, **GenYsis** in C++ using the CUDD software package for BDD manipulation. To analyse the computational efficiency of our methodology, we have tested GenYsis on a range of biological networks of varying complexity. In Table 1 we report the time taken by GenYsis on these

Algorithm 3. Computing all genuine Steady States

```
Input  : Transition function $T$
Output: Steady state array $SS[]$
1  $comp\_steady\_states()$
2  begin
3      $SS[] = all\_Steady\_States(T)$
4      for $i = 0$ to $SS[].size()$ do
5          if $isFalseLoop(SS[i], T) == false$ then
6              report $SS[i]$ as a genuine steady state
7  end
8  $isFalseLoop(S, T)$
9  begin
10     $s_0 = random\_path\_to\_1\_node(S)$
11     $RS^{(0)} \longleftarrow \emptyset, FS^{(0)} \longleftarrow \{s_0\}$
12     $k \longleftarrow 0$
13     while $FS^{(k)} \neq \emptyset$ do
14         $FS^{(k+1)} = I_f(FS^{(k)})(V_{t+1} \leftarrow V_t)$
15         $s' = random\_path\_to\_1\_node(FS^{(k+1)})$
16         $FS_{temp} = FS^{(k+1)} \wedge \overline{s'}$
17         if $FS_{temp} \neq \emptyset$ then
18             /* false steady state */
19             return true
20         $FS^{(k+1)} = FS^{(k+1)} \wedge \overline{RS^{(k)}}$
21         $RS^{(k+1)} = RS^{(k)} \vee FS^{(k+1)}$
22         $k \longleftarrow k + 1$
23     /* genuine steady state */
24     return false
25 end
```

Table 1. Computational results on some gene regulatory networks

Network	Nodes	Edges	Steady States	Number of Iterations	Memory Usage	Time taken (in sec)		
						BDD const.	Steady State	Total
Th network	23	34	3	3	< 15 MB	0.001	0.04	0.041
network2	114	129	1	1	< 15 MB	0.03	0.001	0.031
network3	669	2710	4	10	< 17 MB	0.07	3.15	3.22
network4	1263	5031	1	1	< 57 MB	0.95	314.55	315.55

sample networks. The run time is divided into two parts: time taken to construct BDD and time taken to compute steady states. We also measure the memory requirements for each sample network when analysed by GenYsis. All the results are reported on a 1.6 GHz machine running on linux operating system.

In Table 1, we see that the networks with a size as big as 1263 nodes and 5235 edges can be analysed in less then 6 minutes by using GenYsis. Finding all possible steady states for large network was not feasible with the previous methodologies based on finding the characteristic state of all the feedback loops in the network. Also, GenYsis follows a very intutive way to explore the state space of the network rather then an indirect and difficult to comprehend way of computing the characteristic state.

4 T Helper Cell Differentiation

The vertebrate immune system is constituted by diverse cell populations. Here, we will focus in the CD4+ T lymphocytes known as T helper (Th) cells. These cells conform a particularly suitable differentiation model, because there is a type of precursors cells (Th0), which upon receiving an appropriate antigenic stimulus in vitro, can be further differentiated into cytokine-secreting effector cells, either Th1 or Th2 cells. At the molecular level, Th1 and Th2 cells can be distinguished by their pattern of cytokine secretion, which are responsible for their central role in cell mediated immunity (Th1 cells) and humoral responses (Th2 cells). Understanding the molecular mechanisms that regulate the differentiation process from Th0 towards either Th1 or Th2 is very important, since an immune response biased towards the Th1 phenotype result in the appearance of autoimmune diseases, and an enhanced Th2 response can originate allergic reactions [4,10].

There are several factors at the cellular and molecular levels that determine the differentiation of T helper cells. Importantly, the cytokines present in the cellular milieu play a key role in directing Th cell polarization. On the one hand, IFN-γ, IL-12, IL-18 and IL-27 are the major cytokines that promote Th1 development [11].And on the other hand, IL-4 is the major cytokine responsible for driving Th2 responses. Besides this positive roles of cytokines in the differentiation process, there exist also a mutual inhibitory mechanism. Specifically, IFN-γ play a role in inhibiting the development of Th2 cells, whereas IL-4 inhibits the appearance of Th1 cells. This interplay of positive and negative signals, at both the cellular and molecular levels, creates a complexity that is very suitable for analysis by the modeling approach.

Due to its physiological relevance, there are various mathematical models that have been proposed for describing the differentiation, activation and proliferation of T helper lymphocytes. Most of these models, however, focus on interactions established among the diverse cell populations that somehow modify the differentiation of Th cells [5,17]. Also, other modeling efforts have been aimed at understanding the mechanism of the generation of antibody and T-cell receptor diversity, as well as the molecular networks of cytokine or immunoglobulin interactions [6,16].

Recently we published the first analyses on the gene regulatory network that controls the differentiation process from Th0 to either Th1 or Th2 cells [8,7]. The network (Fig 5) is made of 23 nodes, 26 positive and 8 negative interactions. Importantly, the model does not need to be seen as metabolic pathway, or a reaction network, but rather as an information processing network.

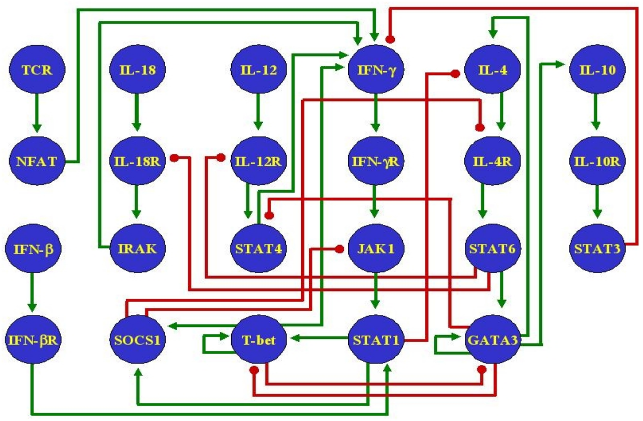


Fig. 5. Th network. The regulatory network that controls the differentiation process of t helper cells. Positive regulatory interactions are with pointed arrow head and negetive interactions with round arrow head.

We already studied the dynamical behavior of the Th network using both discrete and continuous approaches. Such studies permitted the identification of all the stable states of the system. Specifically, the dynamical system obtained from the network has three stable fixed points, which correspond to the patterns of activation observed in normal Th0, Th1 and Th2 cells. Moreover, we were able to modify the model so as to describe the patterns of expression of null mutants, as well as constitutive-expression variants.

Central to our previous analyses is the use of the generalized logical analysis [14,15] for the qualitative analysis of the dynamical properties of the system by focusing on the feedback loops present in the network. Besides helping to understand the Th network, the generalized logical analysis has been applied to other regulatory networks, including those involved in organ differentiation control in the flowers of *Arabidopsis thaliana* [9], and in the initiation of segmentation during *Drosophila melanogaster* embryogenesis [12,13]. Despite its usefulness, the generalized logical analysis has two main drawbacks. First, the computational time needed to analyze all possible feedback loops in a network grows very fast, so that it is not feasible to study large networks. And second, to study the behavior of mutants, it is necessary to create alternative models where the parameters reflect the intended mutation, so that the number of models multiplies by the number of intended mutants. Hence, an alternative, faster and more easily scalable methodology is required for the study of the dynamical properties of biological networks. Our new approach of BDD representation for gene regulatory networks, can provide an alternate way for efficiently analyzing feedback loops in the network and perform *in silico* gene perturbation experiments. When we apply GenYsis on the T helper cell network of Figure 5, we get the three wild type steady states as listed in Table 2.

Table 2. Steady state of the wild type and virtual knock-out of IFN-γ and IFN-γR

Knocked Genes	Active genes in steady states
wild type	IFN-γ Tbet SOCS-1 IFN-γR
	All the genes are inactive
	GATA-3 IL-10 IL-10R IL-4 IL-4R STAT3 STAT6
IFN-γ^-	Tbet SOCS-1
	All the genes are inactive
	GATA-3 IL-10 IL-10R IL-4 IL-4R STAT3 STAT6
IFN-γR$^-$	IFN-γ Tbet SOCS-1
	All the genes are inactive
	GATA-3 IL-10 IL-10R IL-4 IL-4R STAT3 STAT6

These steady states correspond to the molecular profiles observed in Th0, Th1 and Th2 cells respectively. The first steady state reflects the pattern of Th0 cells, which are precursor cells that do not produce any of the cytokines included in the model (IFN-β, IFN-γ, IL-10, IL-12, IL-18 and IL-4). The second steady state represents Th1 cells with high activation of IFN-γ, IFN-γR, T-bet and SOCS1. Finally the third steady state corresponds to the activation observed in th2 cells, with high level of activation of GATA-3, IL-10, IL-10R, IL-4, IL-4R, STAT3 and STAT6. These results also match those published in [8]. GenYsis took only 0.04 seconds to compute these steady states.

In the literature, modeling of Th cell differentiation at the molecular level has been shown to be very useful to bring insight into the origin of the unexpected phenotypes. Previously [7], we made an explanation for the unexpected phenotypic similarity between IFN-γ and IFN-γR loss-of-function mutants (figure 5) [26] [25]. Similarly, using our new BDD based methods we performed virtual knock-out on both IFN-γ and IFN-γR (see Table 2) and compared it to the unperturbed system (wild type). In the case of the IFN-γ knock-out, both the IFN-γ and its receptor are removed from the identified steady state. However when IFN-γR is knocked out, the steady state observed still contains the production of IFN-γ. This is similar to what was obtained with the standard GLA approach from [7], but GenYsis is 100x faster then latter.

5 Conclusion

This paper gives an efficient way for modeling the gene regulatory networks and perform dynamic analysis on them. This new approach can model very efficiently even the biggest regulatory networks available to the modeling community and provide means to perform *in silico* experiments on them. The proposed method has been applied on a T helper cell regulatory network. From the whole range of experiments that were tested with GenYsis, we have reported in this paper, two very interesting knock-outs which have been studied extensively by the modelling community for a long time. In future, we will be extending GenYsis to perform whole suite of *in silico* gene perturbation experiments including gene over-expression and multiple perturbations.

Acknowledgements

This work was supported in part by ENFIN, a Network of Excellence funded by the European Commission within its FP6 Programme, under the thematic area "Life Sciences, genomics and biotechnology for health", contract number LSHG-CT-2005-518254.

Abhishek Garg would like to thank Luca Benini for his help and discussion on the technical details of this paper.

References

1. Bryant, R.E., 'Graph-Based Algorithms for Boolean Function Manipulation'. *IEEE Trans. on Computers*,Vol. 35 (1986), 677–691.
2. Burch, J.R. and Clarke, E.M. and Long, D.E. and MacMillan, K.L. and Dill, D.L., 'Symbolic Model Checking for Sequential Circuit Verification'. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 13 (1994), 401–424.
3. Touati, H.J, Savoj, H., Lin, B., Brayton, R.K., Sangiovanni-Vincentelli, A., 'Implicit state enumeration of finite-state machines using BDDs'. *Proc. of ICCAD*, 1990.
4. Agnello, D., Lankford, C.S.R., Bream, J., Morinobu, A., Gadina, M., OShea, J. and Frucht, D.M., 'Cytokines and transcription factors that regulate T helper cell differentiation: new players and new insights'. *J. Clin. Immun.*, Vol. 23 (2003), 147–162
5. Bergmann, C. and van Hemmen, J.L., 'Th1 or Th2: how an appropriate T helper response can be made'. *Bull. Math. Bio.*, Vol. 63 (2001), 405-430.
6. Krueger, G.R., Marshall, G.R., Junker, U., Schroeder, H., Buja, L.M. and Wang, G., 'Growth factors, cytokines, chemokines and neuropeptides in the modeling of T-cells'. *In Vivo*, Vol. 16(2002), 365-586.
7. Mendoza, L., A network model for the control of the differentiation process in Th cells. BioSystems, Vol 84 (2006), 101-114.
8. Mendoza, L. and Xenarios, I., 'A method for the generation of standardized qualitative dynamical systems of regulatory networks'. *Theoretical Biology and Medical Modelling*, Vol. 3 (2006).
9. Mendoza, L., Thieffry, D., Alvarez-Buylla, E.R., 'Genetic control of flower morphogenesis in Arabidopsis thaliana: a logical analysis.' *BioInfo.*, Vol. 15 (1999), 593-606.
10. Murphy, K.M. and Reiner, S.L., 'The lineage decisions on helper T cells'. *Nat. Rev. Immun.*, 2002, 933-944.
11. Szabo,S.J., Sullivan, B.M., Peng, S.L. and Glimcher, L.H., 'Molecular mechanisms regulating Th1 immune responses'. *Ann. Rev. Immun.*, Vol. 21 (2003), 713-758.
12. Thieffry, D. and Sánchez, L., 'Alternative epigenetic states understood in terms of specific regulatory structures'. *Ann. N.Y. Acad. Sci.*, Vol. 981 (2002), 135-153.
13. Sánchez, L. and Thieffry, D., 'Segmenting the fly embryo: a logical analysis of the pair-rule cross-regulatory module'. *Jour. Theo. Bio.*, Vol. 224 (2003), 517-537.
14. Thomas, R., 'Regulatory networks seen as asynchronous automata: a logical description'. *Jour. Theo. Bio.*, Vol. 153 (1991), 1-23.
15. Thomas, R., Thieffry, D. and Kaufman, M., 'Dynamical behaviour of biological regulatory networks-I. Biological role of feedback loops and practical use of the concept of the loop-characteristic state'. *Bull. Math. Biology*, Vol. 57 (1995), 247-276.

16. Weisbuch, G., DeBoer, R.J. and Perelson, A.S., ‘Localized memories in idiotypic networks’. *Jour. Theo. Bio.*, Vol. 146 (1990), 483-499.
17. Yates, A., Bergmann, C., van Hemmen, J.L., Stark, J. and Callard, R., ‘Cytokine-modulated regulation of helper T cell populations’. *Jour. Theo. Bio.*, Vol. 206 (2000), 539-560.
18. Xie, A. and Beerel, P.A., ‘Efficient State Classification of Finite State Markov Chains’. *Proc. of DAC*, 1998.
19. Hachtel, G. Macii, E., Pardo, A. and Somenzi, F., ‘Markovian analysis of large finite state machines’. *IEEE Trans. on CAD*, Vol. 15 (1996), 1479-1493.
20. Somenzi, F, ‘CUDD: CU Decision Diagram Package Release 2.4.1.’. *University of Colorado at Boulder*. 2005.
21. Brayton, R.K., Sangiovanni-Vincentelli, A.L., McMullen, C.T., and Hachtel, G.D., *Logic Minimization Algorithms for VLSI Synthesis*. Kluwer Academic Publishers, 1984.
22. DeMicheli, G., *Synthesis and Optimization of Digital Circuits*. McGraw-Hill Higher Education, 1994.
23. Burch, J.R., Clarke, E.M., MacMillan, K.L., Dill, D.L. and Hwang, L.H., ‘Symbolic Model Checking:: 10^{20} States and Beyond’. *In Proc. of the IEEE Symp. on Logic in Computer Science*, 1990.
24. Alur, R., Henzinger, T.A., Mang, F.Y.C., Qadeer, S., Rajamani, S.K. and Tasiran, S., ‘MOCHA: Modularity in Model Checking’. *CAV*, 1998.
25. Diehl, S., Anguita, J., Hoffmeyer, A., Zapton, T., Ihle, J.N., Fikrig, E. and Rincón, M., ‘Inhibition of Th1 differentiation by IL-6 is mediated by SOCS1’. *Immunity*, Vol. 13 (2000), 805-815.
26. Tang, H., Sharp, G.C., Peterson, K.P. and Braley-Mullen, H., ‘IFN-g-deficient mice develop severe granulomatous experimental autoimmune thyroiditis with eosinophil infiltration in thyroids’. *Jour. Immun.*, Vol. 160 (1998), 5105-5112.
27. Bernot, G., Comet, J.P., Richard, A. and Guespin, J., ‘Application of formal methods to biological regulatory networks: extending Thomas’ asynchronous logical approach with temporal logic’. *Jour. Theo. Bio.*, Vol. 229 (2004), 339-347.
28. Devloo,V., Hansen, P. and Labb, M., ‘Identification Of All Steady States In Large Biological Systems By Logical Analysis’. *Bull. Math. Bio.*, Vol. 65 (2003), 1025-1051.
29. Chabrier, N., Fages, F. and Soliman, S., ‘BIOCHAM’. *Proc. of CMSB*, May 2004.

A Feature-Based Approach to Modeling Protein-DNA Interactions

Eilon Sharon and Eran Segal

Department of Computer Science,
Weizmann Institute of Science,
Rehovot, 76100, Israel
{eilon.sharon,eran.segal}@weizmann.ac.il
http://genie.weizmann.ac.il

Abstract. Transcription factor (TF) binding to its DNA target site is a fundamental regulatory interaction. The most common model used to represent TF binding specificities is a *position specific scoring matrix* (PSSM), which assumes independence between binding positions. In many cases this simplifying assumption does not hold. Here, we present *feature motif models* (FMMs), a novel probabilistic method for modeling TF-DNA interactions, based on *Markov networks*. Our approach uses sequence *features* to represent TF binding specificities, where each feature may span multiple positions. We develop the mathematical formulation of our models, and devise an algorithm for learning their structural features from binding site data. We evaluate our approach on synthetic data, and then apply it to binding site and ChIP-chip data from yeast. We reveal sequence features that are present in the binding specificities of yeast TFs, and show that FMMs explain the binding data significantly better than PSSMs.

Keywords: transcription factor binding sites, DNA sequence motifs, probabilistic graphical models, Markov networks, motif finder.

1 Introduction

Precise control of gene expression lies at the heart of nearly all biological processes. An important layer in such control is the regulation of transcription. This regulation is preformed by a network of interactions between transcription factor proteins (TFs) and the DNA of the genes they regulate. To understand the workings of this network, it is thus crucial to understand the most basic interaction between a TF and its target site on the DNA. Indeed, much effort has been devoted to detecting the TF-DNA binding location and specificities.

Experimentally, much of the binding specificity information has been determined using traditional methodologies such as footprinting, gel-shift analysis, Southwestern blotting, or reporter constructs. Recently, a number of high-throughput technologies for identifying TF binding specificities have been developed. These methods can be classified to two major classes, in vitro and in vivo methods. In vitro methods can further be classified to methods that select high-affinity binding sequences for a protein of interest (review in Elnitski *et al.*[1]), and high-throughput methods that measure the affinities of specific proteins to multiple DNA sequences. Examples of the latter class of methods include protein binding microarrays [2] and microfluidic platforms [3], which claim to achieve better measurement of transient low affinity interactions. The in vivo methods are mainly based on microarray readout of either DNA adenine methyltransferase fusion proteins (DamID) or of chromatin immunoprecipitation DNA-bound proteins (ChIP-chip) [2]. However, despite these technological advances, distilling the TF binding specificity from these assays remains a great challenge, since in many cases the in vivo measured targets of a TF do not have common binding sites, and in other cases genes that have the known and experimentally determined site for a TF are not measured as its targets. For these reasons, the problem of identifying transcription factor binding sites (TFBSs) has also been the subject of much computational work [1].

The experimental and computational approaches above revealed TFBSs are short, typically 6-20 base pairs, and that some degree of variability in the TFBSs is allowed. For these reasons, the binding site specificities of TFs are described by a sequence *motif*, which should represent the set of multiple allowed TFBSs for a given TF. The most common representation for sequence motifs is the *position specific scoring matrix* (PSSM), which specifies a separate probability distribution over nucleotides at each position of the TFBS. The goal of computational approaches is then to identify the PSSM associated with each TF.

Despite its successes, the PSSM representation makes the very strong assumption that the binding specificities of TFs are position-independent. That is, the PSSM assumes that for any given TF and TFBS, the contribution of a nucleotide at one position of the site to the overall binding affinity of the TF to the site does not depend on the nucleotides that appear in other positions of the site. In theory, it is easy to see where this assumption fails. Consider for example the models described in Figure 1, if instead of the PSSM representation, we allowed ourselves to assign probabilities to multiple nucleotides at multiple positions, then we could use the same number of parameters to specify the desired TF binding specificities. This observation lies at the heart of our approach.

From the above discussion, it should be clear that the position-independent assumption of PSSMs is rather strong, and that relaxing this assumption may lead to a qualitatively better characterization of TF motifs. Indeed, recent studies revealed specific cases in which dependencies between positions may exist, [3]. In a more comprehensive study, Barash *et al.*[4] developed a Bayesian network approach to represent higher order dependencies between motif positions, and showed that these models predict putative TFBSs in ChIP-chip data with higher accuracy than PSSMs. However, the Bayesian network representation, due to its acyclicity constraints, imposes unnecessary restrictions on the motif structure, and its conditional probability distributions limit the number of dependencies that can be introduced between positions in practice, due to the exponential increase in the number of parameters introduced with each additional

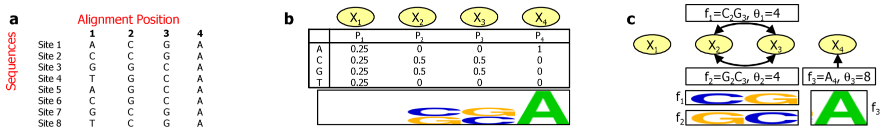


Fig. 1. Comparison between FMMs and PSSMs in a toy example of a TFBS with 4 positions. (a) Eight input TFBSs that the TF recognizes. (b) A PSSM for the input data in (a), showing its Markov network representation, probability distributions over each position, and sequence logo. Note that the PSSM assigns a high probability to CG and GC in positions 2 and 3 as expected by the input data, but it also undesirably assigns the same high probability to CC and GG in these positions. (c) An FMM for the input data in (a), showing the associated Markov network, with 3 features, and sequence logo. Note that features f_1 and f_2 assign a high probability to CG and GC in positions 2 and 3 but not to CC and GG in these positions, as desired.

dependency. While some of these issues may be addressed, e.g., using sparse conditional probability distribution representations, Bayesian networks are not the ideal and most intuitive tool for the task.

Here, we propose a novel approach to modeling TFBS motifs, termed *feature motif models* (FMMs). Our approach is based on describing the set of sequence properties, or *features*, that are relevant to the TF-DNA interactions. Intuitively, the binding affinity of a given site to the TF increases as it contains more of the features that are important for the TF in recognizing its target site. In our framework, features may be binary (e.g., “C at position 2, and G at position 3”) or multi-valued (e.g., “the number of G or C nucleotides at positions 1-4”), and global features are also allowed (e.g., “the sequence is palindromic”). Each feature is assigned a statistical weight, representing the degree of its importance to the TF-DNA interaction, and the overall strength of a TFBS can then be computed by summing the contribution of all of its constituent features. We argue that this formulation captures the essence of the TF-DNA interaction more explicitly than PSSMs and other previous approaches. It is easy to see that our FMMs contains in it the PSSM description, since a PSSM can be described within our framework using four single nucleotide features per position.

In what follows, we provide the mathematical formulation of FMMs, and devise an algorithm for learning FMMs from TFBSs data. This problem is quite difficult, as it reduces to learning structure in Markov networks, a paradigm that is still poorly developed. We evaluate our approach in a controlled synthetic data setting, and demonstrate that we can learn the correct features even from a relatively small number of positive examples. Finally, we apply our method to real TFBSs for yeast TFs [5,6], and show several cases where our method better explains the observed TFBS data and identifies motif sequence features that span multiple positions. We identify global properties that are common to the DNA sequence specificities of most TFs: TFBSs have strong dependencies between positions; these dependencies mostly occur in the center of the site; and dependencies typically exist between nearby positions in the site.

2 The Feature Motif Model

We now present our approach for representing TF binding specificities. Much like in the PSSM representation, our goal is to represent commonalities among the different TFBSs that a given TF can recognize, and assign a different strength to each potential site, corresponding to the affinity that the TF has for it. The key difference between our approach and a PSSM is that we want to represent more expressive types of motif commonalities compared to the PSSM representation, in which motif commonalities can only be represented separately for each position of the motif. Intuitively, we think of a TF-DNA interaction as one that can be described by a set of sequence *features*, such as pairs or triplets of nucleotides at key positions, that are important for the interaction to take place: the more important features a specific site has, the higher affinity it will have for the TF.

One way to achieve the above task is to represent a probability distribution over the set of all sequences of the length recognized by the given TF. That is, for a motif of length L, we represent a probability distribution over all 4^L possible L-mer sequences. Formally, we wish to represent a joint probability distribution $P(X_1, \ldots, X_L)$, where X_i is a random variable with domain $\{A, C, G, T\}$ corresponding to the nucleotide at the i-th position of the sequence. However, rather than representing this distribution using the prohibitively large number of $4^L - 1$ independent parameters, our goal is to represent this joint distribution more compactly in a way that requires many fewer parameters but still captures the essence of TF-DNA interactions. The PSSM does exactly this, but it forces the form of the joint distribution to be decomposable by positions. Barash *et al.*[4] presented alternative representations to the PSSM, using Bayesian networks, that allow for dependencies to exist across the motif positions. However, as discussed above, the use of Bayesian networks imposes unnecessary restrictions and is not natural in this context.

A more natural approach that can easily capture our above desiderata is the framework of undirected graphical models, such as Markov networks or log-linear models, which have been used successfully in an increasingly large number of settings. As it is more intuitive for our setting, we focus our presentation on log-linear models. Let $\mathcal{X} = \{X_1, \ldots, X_L\}$ be a set of discrete-valued random variables. A *log-linear model* is a compact representation of a probability distribution over assignments to $\mathcal{X}$. The model is defined in terms of a set of *feature functions* $f_k(\boldsymbol{X}_k)$, each of which is a function that defines a numerical value for each assignment $\boldsymbol{x}_k$ to some subset $\boldsymbol{X}_k \subset \mathcal{X}$. Given a set of feature functions $F = \{f_k\}$, the parameters of the log-linear model are weights $\boldsymbol{\theta} = \{\theta_k : f_k \in F\}$. The overall joint distribution is then defined as:

$$P(\boldsymbol{x}) = \frac{1}{Z} \exp\left(\sum_{f_k \in F} \theta_k f_k(\boldsymbol{x}_k)\right), \text{where } Z = \sum_{\boldsymbol{x} \in \boldsymbol{X}} \exp\left(\sum_{f_k \in F} \theta_k f_k(\boldsymbol{x}_k)\right) \quad (1)$$

is the *partition function* that ensures that the distribution P is properly normalized (i.e., $\sum_{\boldsymbol{x} \in \boldsymbol{X}} P(\boldsymbol{x}) = 1$), and $\boldsymbol{x}_k$ is the assignment to $\boldsymbol{X}_k$ in $\boldsymbol{x}$. Although we chose the log-linear model representation, we note that it is in fact equivalent to the Markov network representation, and the mapping between the two is straightforward. We now demonstrate how we can use this log-linear model representation in our setting, to represent feature-based motifs. We start by showing how PSSMs can be represented within this framework.

Representing PSSMs. Recall that a PSSM defines independent probability distributions over each of the L positions of the motif. To represent PSSMs in our model, we define 4 features f_{iJ} for each position that indicate whether a specific nucleotide $J \in \{A, C, G, T\}$ exists at a specific position $1 \leq i \leq L$ of the TFBS. We associate each feature with a weight θ_{iJ} that is equal to its marginal log probability over all possible TFBSs. It is easy to show that putting this into Equation 1 defines the exact same probability distribution as of the PSSM, and that the *partition function* as defined in Equation 1 is equal to 1 in this case.

Representing Feature Motifs. Given a TF that recognizes TFBSs of length L, our feature-based model represents its motif using the log-linear model of Equation 1, where each feature f_k corresponds to a sequence property that may be defined over multiple positions. As an example for a feature, consider the indicator function: 'C' at position 2 and 'G' at position 3, as in Figure 1c. This feature illustrates our ability to define features over multiple positions. Although our results focus on indicators of single or pair of nucleotide features, we note that continuous and even global features (such as G/C content) can easily be defined within our model. We then associate each feature with a weight, θ_k, that defines its importance to the TF-DNA binding affinity. Given a sequence, we can now compute its probability using Equation 1, which boils down to summing the value of all the features present in the sequence, each multiplied by its respective weight parameter, and exponentiating and normalizing this resulting sum. Intuitively, this model corresponds to identifying which of the features that are important for the TF-DNA interaction are present in the sequence, and summing their contributions to obtain the overall affinity of the TF to the site. This intuitive model is precisely the one we set out to obtain.

3 Learning Feature Motif Models

In the previous section, we presented our feature-based model for representing motifs. Given a collection of features F, our method uses the log-linear model to integrate them, as in Equation 1. As we showed, the standard PSSM model can be represented in our framework. However, our motivation in defining the model was to allow for integration of other features, that may span multiple positions. A key question is how to select the set of features for a given model. In this section, we address this problem. Since log-linear models are equivalent to Markov networks, our problem essentially reduces to structure learning in Markov networks. This problem is quite difficult, since even the simpler problem of estimating the parameters of a fixed model does not have an analytical closed form solution. Thus, the solutions proposed for this problem have been various heuristic searches, which incrementally modify the model by adding and deleting features to it in some predefined scheme [7,8].

We now present our algorithm for learning a feature-based model from TFBSs data. Our approach follows the Markov network structure learning method of Lee *et al.*[8]. It incrementally introduces (or selects) features using the *grafting* method of Perkins *et al.*[9]. We first present the simpler task of estimating the parameters of a given model, as this is a sub-problem that we need to solve when searching over the space of possible network structures.

3.1 Parameter Estimation

For the parameter estimation task, we assume that we are given as input a dataset $D = \{\boldsymbol{x}[1], \ldots, \boldsymbol{x}[N]\}$ of N *aligned* i.i.d TFBSs, each of length L, and a model $\mathcal{M}$ defined by a set of sequence features $F = \{f_1, \ldots, f_k\}$. Our goal is to find the parameter vector $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_k\}$ that specifies a weight for each feature $f_i \in F$, and maximizes the log-likelihood function:

$$\log P(D \mid \boldsymbol{\theta}, \mathcal{M}) = \sum_{i=1}^{N} \log P(\boldsymbol{x}[i] \mid \boldsymbol{\theta}, \mathcal{M}) = \sum_{i=1}^{N} \sum_{f_k \in F} \theta_k f_k(\boldsymbol{x}[i]_k) - N \log Z \quad (2)$$

where $\boldsymbol{x}[i]_k$ corresponds to the nucleotides of the i-th TFBS at the positions relevant to feature k, and Z is the partition function as in Equation 1. It can easily be shown that the gradient of Equation 2 is:

$$\frac{\partial \log P(D \mid \boldsymbol{\theta}, \mathcal{M})}{\partial \theta_k} = \sum_{i=1}^{N} f_k(\boldsymbol{x}[i]_k) - N \frac{1}{Z} \frac{\partial Z}{\partial \theta_k} \quad (3)$$

Although no closed-form solution exists for finding the parameters that maximize Equation 2, the objective function is concave, and we can thus find the optimal parameter settings using numerical optimization procedures such as gradient ascent or conjugate gradient [10]. We now deal with optimizing Equation 2.

3.2 Optimization of the Objective Function

Applying numerical optimization procedures such as gradient ascent requires the computation of the objective function and the gradient with respect to any of the θ_k parameters. Although the fact that the objective function is concave, and that both the function and its gradient have simple closed forms may make the parameter estimation task look simple, in practice the computing them may be quite expensive. The reason is that the second terms of both the function and the gradient involve evaluating the partition function, which requires, in a naive implementation, summing over 4^L possible TFBSs sequences.

Since algorithms for learning Markov networks usually require computation of the partition function, this problem was intensively researched. Although in some cases the structure of the features may be such that we can decompose the computation to achieve efficient computation, in the general case it can be shown to be a NP-hard problem and hence requires approximation. Here we suggest a novel strategy of optimizing the objective function. We first use the (known) observation that the gradient of Equation 2 can also be expressed in terms of features expectations. Specifically, since

$$\frac{1}{Z}\frac{\partial Z}{\partial \theta_k} = \frac{\sum_{\boldsymbol{x}\in\boldsymbol{X}} f_k(\boldsymbol{x}_k)\exp\left(\sum_{f_k\in F}\theta_k f_k(\boldsymbol{x}_k)\right)}{\sum_{\boldsymbol{x}\in\boldsymbol{X}}\exp\left(\sum_{f_k\in F}\theta_k f_k(\boldsymbol{x}_k)\right)} = E_{P\sim\theta}(f_k(\boldsymbol{x}_k)), \tag{4}$$

we can rewrite Equation 3 as:

$$\frac{\partial \log P(D \mid \boldsymbol{\theta}, \mathcal{M})}{\partial \theta_k} = \sum_{i=1}^{N} f_k(\boldsymbol{x}[i]_k) - N E_{P\sim\theta}(f_k(\boldsymbol{x}_k)). \tag{5}$$

We further observed that since Equation 2 is a concave function, its absolute directional derivative along any given line in its domain is also a concave function. We used this observation to use the conjugate gradient function optimization algorithm [10] in a slightly modified version: Although the gradient that was given to the algorithm was indeed as in Equation 5, the function value along every line search step of the algorithm was the absolute directional derivative along this line. For example, at the line search step along direction $\boldsymbol{y}$ our function $F^{\star}(\boldsymbol{\theta}, \boldsymbol{y})$ value is: $F^{\star}(\boldsymbol{\theta}, \boldsymbol{y}) = | < \nabla \log P(D \mid \boldsymbol{\theta}, \mathcal{M}), \boldsymbol{y} > |$.

Following the above strategy allows us to optimize Equation 2 without computing its actual value. Specifically, it means that we can optimize our objective without computing the partition function. Instead, the problem reduces to evaluating feature expectations, a special case of inference in Markov networks, that can be computed using algorithms such as *loopy belief propagation* [11]. The ability of these algorithms to give an exact result depends on the underlying network structure. As the network structure becomes more complex, the algorithms need to use approximations. Since this family of algorithms can also approximate the partition function, our method will be similar to methods that evaluate the partition function when the network structure allows for exact inference. However, as the error bounds for approximate inference are better characterized then the error bounds of partition function estimations, it is possible that our approach may work better under conditions that require approximation.

3.3 Learning the Features

In Section 3.1, we developed our approach for estimating the feature parameters for a fixed model in which the feature set F is defined. We now turn to the more complex problem of automatically learning the set of features from aligned TFBSs data. This problem is an instance of the more general problem of learning the structure of Markov networks from data. However, quite surprisingly, although Markov networks are used in a wide variety of applications, there are very few effective algorithms for learning Markov network structure from data.

In this paper we followed the Markov network structure learning approach suggested by Lee *et al.*[8]. This approach extends the learning approach of Perkins *et al.*[9] to learning the structure of Markov networks using the L_1-Regularization over the model parameters. To incorporate the L_1-Regularization into our model we need to introduce a *Laplacian* parameter prior over each feature, leading to the modified objective function:

$$\log P(D, \boldsymbol{\theta} \mid \mathcal{M}) = \log P(D \mid \boldsymbol{\theta}, \mathcal{M}) + \log P(\boldsymbol{\theta} \mid \mathcal{M}) \tag{6}$$

where $P(\boldsymbol{\theta} \mid \mathcal{M}) = \left(\frac{\alpha}{2}\right)^{|F|} exp\left(-\sum_{f_k \in F} \alpha|\theta_k|\right)$ and $\log P(D \mid \boldsymbol{\theta}, \mathcal{M})$ is the data likelihood function as in Equation 2. Taking the log of this parameter prior and eliminating constant terms, we arrive at the final form of our objective function:

$$\log P(D, \boldsymbol{\theta} \mid \mathcal{M}) = \sum_{i=1}^{N} \sum_{f_k \in F} \theta_k f_k(\boldsymbol{x}[i]_k) - N \log Z - \alpha \sum_{f_k \in F} |\theta_k| \tag{7}$$

It is easy to see that this modified objective function is also concave in the feature parameters $\boldsymbol{\theta}$ and we can thus optimize it using the same conjugate gradient procedure described in Section 3.1. We then follow the *grafting* approach of adding features in a stepwise manner. In each step, the algorithm first optimizes the objective function relative to the current set of active features F, and then adds the inactive feature $f_i \notin F$ with the maximal gradient at $\theta_i = 0$. Using an L_1-Regularized concave function provides a stopping criteria to the algorithm that leads to the global optimum [9]. The L_1-Regularization has yet another desirable quality for our purpose, as it has a preference for learning sparse models with a limited number of features [8]. It has long been known to have a tendency towards learning sparse models, in which many of the parameters have weight zero [12] and theoretical results show that it is useful in selecting the features that are most relevant to the learning task [13]. Since the *grafting* feature addition method is a heuristic, it seems reasonable that features that were added at an early stage may become irrelevant at later stages, and hence get a zero weight. We thus introduce an important difference from the method of Lee *et al.*, by allowing the removal of features that become irrelevant.

4 Experimental Results

We now present an experimental evaluation of our approach. We first use synthetic data to test whether our method can reconstruct sequence features that span multiple positions when these are present, and then compare the ability of our approach to learn binding specificities of yeast TFs to that of PSSMs.

4.1 Synthetic Data

To evaluate our models in a controlled setting, we manually created three FMMs (Figure 2) of varying weights and features, and learned both PSSM and FMMs from TFBSs that we sampled from them. We evaluated the learned models by computing the log-likelihood that the learned models assign to a test set of 10,000

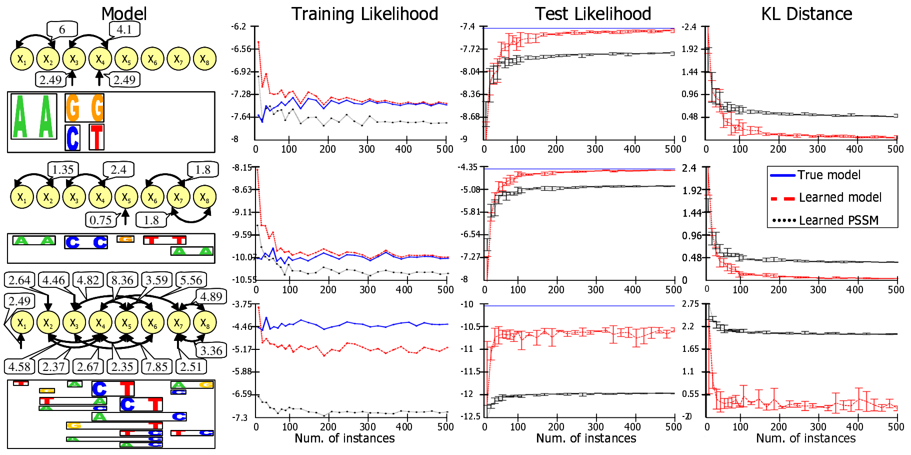


Fig. 2. Evaluation of our approach on synthetic data. Results are shown for three manually constructed model, from which we drew samples and constructed FMMs and PSSMs. For each model, shown is its Markov network and sequence logo (left), training and test log-likelihood (average per instance for the true model, and learned FMM and PSSM) and KL-distance of the learned FMM and PSSM models from the true model (train likelihood error bars were excluded for clarity).

unseen TFBSs sampled from the true model, and by computing the Kullback Leibler (KL) distance between distributions of the true and learned models.

We evaluated two specific aspects of our approach: the minimum number of samples needed for learning FMMs, and the dependency of the learning on the prior weighting parameter, α. In all experiments, we limited the FMM to structures that allow exact inference using belief propagation algorithm [11]. While this poses constraints on the underlying network, learning more complex models also gave good performance, since the most important features were still learned. We repeated each experiment setting 3 times.

We first tested the effect of the prior weight parameter α on the quality of the learning reconstruction. To this end, we varied α in the range of 10^{-6} to 100, while using a fixed number of 500 input sequences. The results showed that in the range tested, the best reconstruction performance was achieved for $\alpha = 0.1$. While smaller values tend to allow over fitting, higher values pose harsh constraints on the learned model.

Second, we estimated the minimum number of samples needed for learning FMMs, by sampling different training set sizes in the range of 10-500. As can be seen in Figure 2, for all three cases, our model reconstructs the true model with high accuracy even with a modest number of 50 input TFBSs, reconstructs the true model nearly perfectly with 100 or more samples. As expected, since the true model includes dependencies between positions, our model significantly outperforms the PSSM in these cases even when only 20 input sites were used. In these experiments, we fixed the prior weight parameter to 0.1. Examining the learned features, we found that for a sample size of 20 or more, only features that

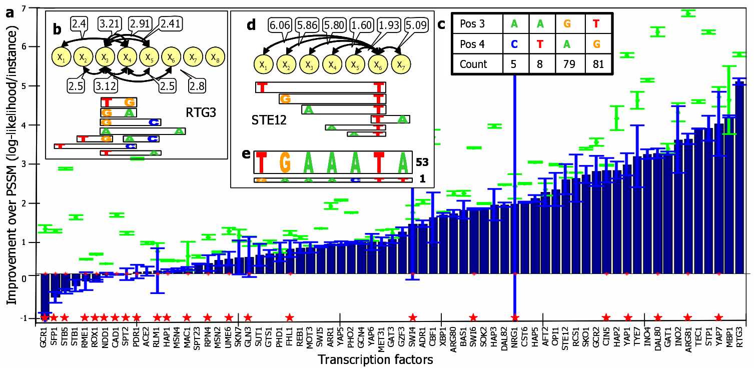


Fig. 3. Evaluating our approach on real TFBSs from yeast. (a) Train (green points) and test log-likelihood (blue bars), shown as the mean and standard deviation improvements in the average log-likelihood per instance compared to a PSSM. Each model was learned from the TFBSs reported by MacIsaac06 et al. in a 5-fold cross validation scheme. Models that were constrained to allow exact inference are marked with a red star. (b) Markov network representation of the dominant features of the FMM model learned for RTG3. (c) Sequence counts for positions 3 and 4 of the input TFBSs of RTG3. (d,e) Same as (b,c), for strong feature relations learned for the STE12 TF.

appeared in the true model were learned with significant weight. Our results thus show that we can successfully learn FMMs, even in a realistic setting in which only a limited number of input TFBSs is available.

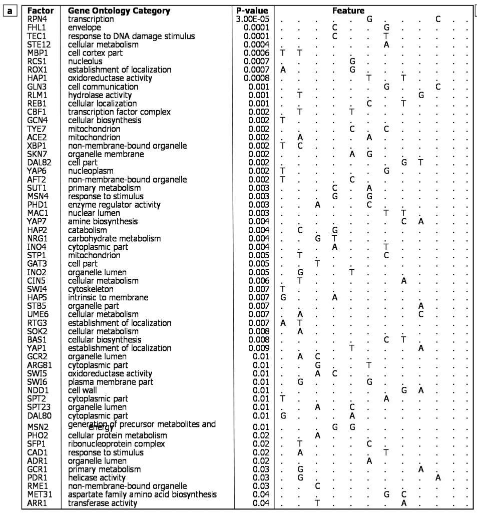
4.2 Identifying Binding Features of Yeast TFs

Having validated our approach on synthetic data, we next applied it to TFBSs data for yeast TFs. Our goal is to identify whether FMMs can better describe the sequence specificities of yeast TFs. As input to our method, we used the high-quality TFBSs data reported by MacIsaac *et al.*[6]. This dataset consists of 16371 regulatory TF-binding site interactions, where each interaction reported is one in which the TF is bound to the promoter region containing the TFBS as determined by the ChIP-chip assays of Harbison *et al.*[5], and the TFBS has a good match to the PSSM reported for the corresponding TF. While this dataset is quite comprehensive, it is in fact a very stringent test for our method, since each reported TFBS is required to have a relatively good match to the PSSM, a property that we do not necessarily expect from sequences that are well explained by our feature motif models.

We used a five fold cross validation scheme to test whether FMMs can better explain yeast TFBSs. We took 69 TFs of length ≤ 12 and at least 30 TFBSs and learned for each a model from the training set. Models of length greater then 8 were constrained to allow exact inference as in Section 4.1. As a measure of success, we computed for each motif, the average and standard deviation of the five test sets average log likelihood. Using this criterion, we compared the results of applying our model to that of applying the PSSM model to the same input data. The results are shown in Figure 3(a). As can be seen, FMMs better explained the TFBSs data of 60 of the 69 TFs (86%). In 34 of the 69 (49%) cases, the probability that our model gave to each TFBS in the test data was, on average, more than twice the probability assigned by the PSSM. We note that although the results of the constrained model were slightly weaker (66%, and 33% respectively) they are still relatively good. Taken together, these results demonstrate that TFBSs data can be better characterized by feature motif models compared to PSSMs, and that the position independent assumption of the PSSM model does not hold in many cases and can thus poorly represent the binding affinities of many TFs.

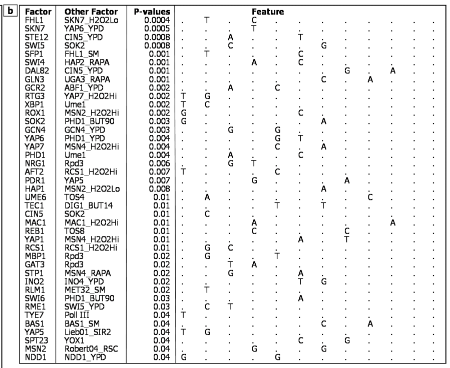
We next turned to examine the actual features that we identified and test their biological significance. To this end, we first examined the models learned for each of the 69 TFs, by extracting the dominant features learned and observing the counts of these features in the original input TFBS data. Two examples of such a model examination are shown in Figure 3(b-e). The leucine zipper TF RTG3, an activator of the TOR growth pathway, represents one case in which our model provides insight into its binding specificity, and in which we can clearly understand why the PSSM model fails. For this TF, our model assigns a probability that, on average per test-set TFBS, is more than 20 times greater than the corresponding probability assigned by the PSSM. Examining the dominant features of the model reveals that the two most dominant features were defined over positions 3 and 4. Each one of these features gives high weight to either "GA" or "TG" at thess positions. Strikingly, the counts of these two features in the original input data were 79 and 81 (out of 173 TFBSs), respectively. Clearly, the PSSM model completely misses this. These results suggest that RTG3 may have two distinct types of TFBSs, one with a "TG" in positions 3 and 4 and another with "GA" in these positions. This hypothesis is consistent with a study by Rothermel *et al.*[14] showing that RTG3 contains at least two independent activation domains, which may interact with different co-factors, leading to two different binding modes.

The STE12 TF, an activator of the mating or pseudohyphal growth pathways, is another intriguing example where our model provides insight into the specificity of the corresponding TF. Of all the 994 TFBSs of STE12 in the input data, 54 have a 'T' in position 6. Of these, 53 have the exact full TFBS of 'TGAAATA'. In other words, if a 'T' appears in position 6 of the TFBS, it fully determines the remaining basepairs of the site. As can be seen in Figure 3(d,e), our model captures this property, by learning six features with high weights that each contained a 'T' in position 6, and one of the other positions as the second position. This result is consistent with reports in the literature that the specificity of STE12 can change, depending on its interaction with other regulators [15]. This TFBS is also an example where a simple Bayesian network representation of the site would not be able to compactly represent the site, since position 6 would have to be a parent of each of the other positions, thereby placing constraints (due to acyclicity) on the types of features that could be learned between the positions when 'T' is not present in position 6, and in any case requiring many parameters for the representation. A



a

Factor	Gene Ontology Category	P-value	Feature											
RPN4	transcription	3.00E-05	.	.	.	.	.	G	.	.	.	C	.	.
FHL1	envelope	0.0001	.	.	.	C	.	.	G	.	.	.	.	.
TEC1	response to DNA damage stimulus	0.0001	.	.	.	C	.	.	T	.	.	.	.	.
STE12	cellular metabolism	0.0004	.	.	.	.	.	.	A	.	.	.	.	.
MBP1	cell cortex part	0.0006	T	T	.	.	.	.	.	.	.	.	.	.
RCS1	nucleolus	0.0007	.	.	.	.	G	.	.	.	.	.	.	.
ROX1	establishment of localization	0.0007	A	.	.	.	G	.	.	.	.	.	.	.
HAP1	oxidoreductase activity	0.0008	.	.	.	.	.	T	.	T	.	.	.	.
GLN3	cell communication	0.001	.	.	.	.	.	.	G	.	.	C	.	.
RLM1	hydrolase activity	0.001	.	T	.	.	.	.	.	.	G	.	.	.
REB1	cellular localization	0.001	.	.	.	.	.	C	.	T	.	.	.	.
CBF1	transcription factor complex	0.002	.	T	.	.	T	.	.	.	.	.	.	.
GCN4	cellular biosynthesis	0.002	T	.	.	.	.	.	.	.	.	.	.	.
TYE7	mitochondrion	0.002	.	.	.	.	C	.	C	.	.	.	.	.
ACE2	mitochondrion	0.002	.	A	.	.	.	A	.	.	.	.	.	.
XBP1	non-membrane-bound organelle	0.002	T	C	.	.	.	.	.	.	.	.	.	.
SKN7	organelle membrane	0.002	.	.	.	.	A	G	.	.	.	.	.	.
DAL82	cell part	0.002	.	.	.	.	.	.	.	G	T	.	.	.
YAP6	nucleoplasm	0.002	T	.	.	.	.	.	G	.	.	.	.	.
AFT2	non-membrane-bound organelle	0.002	T	.	.	.	C	.	.	.	.	.	.	.
SUT1	primary metabolism	0.003	.	.	.	C	.	A	.	.	.	.	.	.
MSN4	response to stimulus	0.003	.	.	.	G	.	G	.	.	.	.	.	.
PHD1	enzyme regulator activity	0.003	.	.	A	.	.	C	.	.	.	.	.	.
MAC1	nuclear lumen	0.003	.	.	.	.	.	.	T	T	.	.	.	.
YAP7	amine biosynthesis	0.004	.	.	.	.	.	.	.	C	A	.	.	.
HAP2	catabolism	0.004	.	C	.	G	.	.	.	.	.	.	.	.
NRG1	carbohydrate metabolism	0.004	.	.	G	T	.	.	.	.	.	.	.	.
INO4	cytoplasmic part	0.004	.	.	.	A	.	.	T	.	.	.	.	.
STP1	mitochondrion	0.005	.	T	.	.	.	.	C	.	.	.	.	.
GAT3	cell part	0.005	.	.	T	.	.	.	.	.	.	.	.	.
INO2	organelle lumen	0.005	.	G	.	.	T	.	.	.	.	.	.	.
CIN5	cellular metabolism	0.006	.	T	.	.	.	.	.	A	.	.	.	.
SWI4	cytoskeleton	0.007	T	.	.	.	.	.	.	.	.	.	.	.
HAP5	intrinsic to membrane	0.007	G	.	.	A	.	.	.	.	.	.	.	.
STB5	organelle part	0.007	.	.	.	.	.	.	.	.	A	.	.	.
UME6	cellular metabolism	0.007	.	A	.	.	.	.	.	.	C	.	.	.
RTG3	establishment of localization	0.007	A	T	.	.	.	.	.	.	.	.	.	.
SOK2	cellular metabolism	0.008	.	A	.	.	.	.	.	.	.	.	.	.
BAS1	cellular biosynthesis	0.008	.	.	.	.	.	.	C	T	.	.	.	.
YAP1	establishment of localization	0.009	.	.	.	.	T	.	.	.	A	.	.	.
GCR2	organelle lumen	0.01	.	A	C	.	.	.	.	.	.	.	.	.
ARG81	cytoplasmic part	0.01	.	.	G	.	.	T	.	.	.	.	.	.
SWI5	oxidoreductase activity	0.01	.	.	A	C	.	.	.	.	.	.	.	.
SWI6	plasma membrane part	0.01	.	G	.	.	.	G	.	.	.	.	.	.
NDD1	cell wall	0.01	.	.	.	.	.	.	.	G	A	.	.	.
SPT2	cytoplasmic part	0.01	T	.	.	.	.	.	A	.	.	.	.	.
SPT23	organelle lumen	0.01	.	.	A	.	C	.	.	.	.	.	.	.
DAL80	cytoplasmic part	0.01	G	.	.	.	A	.	.	.	.	.	.	.
MSN2	generation of precursor metabolites and energy	0.01	.	.	.	G	G	.	.	.	.	.	.	.
PHO2	cellular protein metabolism	0.02	.	.	A	.	.	.	.	.	.	.	.	.
SFP1	ribonucleoprotein complex	0.02	.	T	.	.	.	C	.	.	.	.	.	.
CAD1	response to stimulus	0.02	.	A	.	.	.	.	T	.	.	.	.	.
ADR1	organelle lumen	0.02	.	.	.	.	.	A	.	.	.	.	.	.
GCR1	primary metabolism	0.03	.	G	.	.	.	.	.	.	A	.	.	.
PDR1	helicase activity	0.03	.	G	.	.	.	.	.	.	.	A	.	.
RME1	non-membrane-bound organelle	0.03	.	.	C	.	.	.	.	.	.	.	.	.
MET31	aspartate family amino acid biosynthesis	0.04	.	.	.	.	.	.	G	C	.	.	.	.
ARR1	transferase activity	0.04	.	.	T	.	.	.	.	A	.	.	.	.



b

Factor	Other Factor	P-values	Feature											
FHL1	SKN7_H2O2Lo	0.0004	.	T	.	C	.	.	.	.	.	.	.	.
SKN7	YAP6_YPD	0.0005	.	.	.	T	.	.	.	.	.	.	.	.
STE12	CIN5_YPD	0.0008	.	.	A	.	.	T	.	.	.	.	.	.
SWI5	SOK2	0.0008	.	.	C	.	.	.	G	.	.	.	.	.
SFP1	FHL1_SM	0.001	.	T	.	.	.	C	.	.	.	.	.	.
SWI4	HAP2_RAPA	0.001	.	.	.	A	.	C	.	.	.	.	.	.
DAL82	CIN5_YPD	0.001	.	.	.	.	.	.	.	G	.	A	.	.
GLN3	UGA3_RAPA	0.001	.	.	.	.	.	.	C	.	A	.	.	.
GCR2	ABF1_YPD	0.002	.	.	A	.	C	.	.	.	.	.	.	.
RTG3	YAP7_H2O2Hi	0.002	T	G	.	.	.	.	.	.	.	.	.	.
XBP1	Ume1	0.002	T	C	.	.	.	.	.	.	.	.	.	.
ROX1	MSN2_H2O2Hi	0.002	G	.	.	.	.	C	.	.	.	.	.	.
SOK2	PHD1_BUT90	0.003	G	.	.	.	.	.	A	.	.	.	.	.
GCN4	GCN4_YPD	0.003	.	.	G	.	G	.	.	.	.	.	.	.
YAP6	PHD1_YPD	0.004	.	.	.	.	G	T	.	.	.	.	.	.
YAP7	MSN4_H2O2Hi	0.004	.	.	.	.	C	.	A	.	.	.	.	.
PHD1	Ume1	0.004	.	.	A	.	.	C	.	.	.	.	.	.
NRG1	Rpd3	0.006	.	.	G	T	.	.	.	.	.	.	.	.
AFT2	RCS1_H2O2Hi	0.007	T	.	.	.	C	.	.	.	.	.	.	.
PDR1	YAP5	0.007	.	.	.	G	.	.	.	A	.	.	.	.
HAP1	MSN2_H2O2Lo	0.008	.	.	.	.	.	.	A	.	.	.	.	.
UME6	TOS4	0.01	.	A	.	.	.	.	.	.	C	.	.	.
TEC1	DIG1_BUT14	0.01	.	.	.	.	T	.	T	.	.	.	.	.
CIN5	SOK2	0.01	.	C	.	.	.	.	.	.	.	.	.	.
MAC1	MAC1_H2O2Hi	0.01	.	.	.	A	.	.	.	.	.	A	.	.
REB1	TOS8	0.01	.	.	.	C	.	.	.	C	.	.	.	.
YAP1	MSN4_H2O2Hi	0.01	.	.	.	.	.	A	.	T	.	.	.	.
RCS1	RCS1_H2O2Hi	0.01	.	G	C	.	.	.	.	.	.	.	.	.
MBP1	Rpd3	0.02	.	G	.	.	T	.	.	.	.	.	.	.
GAT3	Rpd3	0.02	.	.	T	A	.	.	.	.	.	.	.	.
STP1	MSN4_RAPA	0.02	.	.	G	.	.	A	.	.	.	.	.	.
INO2	INO4_YPD	0.02	.	.	.	.	.	T	G	.	.	.	.	.
RLM1	MET32_SM	0.02	.	T	.	.	.	.	.	.	.	.	.	.
SWI6	PHD1_BUT90	0.03	.	.	.	.	.	A	.	.	.	.	.	.
RME1	SWI5_YPD	0.03	.	C	T	.	.	.	.	.	.	.	.	.
TYE7	Poll III	0.04	T	.	.	.	.	.	.	.	.	.	.	.
BAS1	BAS1_SM	0.04	.	.	.	.	.	.	C	.	A	.	.	.
YAP5	Lieb01_SIR2	0.04	T	G	.	.	.	.	.	.	.	.	.	.
SPT23	YOX1	0.04	.	.	.	.	.	C	.	G	.	.	.	.
MSN2	Robert04_RSC	0.04	.	.	.	G	.	.	G	.	.	.	.	.
NDD1	NDD1_YPD	0.04	G	.	.	.	G	.	.	.	.	.	.	.

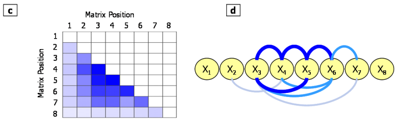


Fig. 4. Biological significance of FMMs. (a) TFBSs of yeast TFs with particular features are enriched for specific GO functional categories. (b) Same as (a) for enrichment in protein-DNA interactions that we compiled from 10 different studies. (c) Average weight of features that span 2 positions, across FMM models learned for all yeast TFs with $L = 8$ (d) 'Consensus' properties of correlations between positions in the sequence specificities of TFs, compiled based on (c).

mixture model, which is one of the options presented Barash *et al.*[4] would work here, but learning it from the data might be challenging.

To further and globally characterize the biological significance of the feature motif models learned, we took the dominant features of each of the 69 models learned, and partitioned the TFBSs into two sets, based on the presence of each of the features. By mapping the sites back to the promoters in which they were identified, we could partition the genes regulated by each TF into genes that have TFBS of the TF and have the examined feature, and genes that don't have TFBS that have the feature. We used the hypergeometric distribution to compute a p-value for an enrichment of the partition to various features. In all enrichment tests we took $p < 0.01$ to be significant, corrected the results by FDR, and presented the best enrichment for each TF. We first tested for enrichment in functional categories from the Gene Ontology (GO) database. The results are shown in Figure 4(a). These results suggest that particular features of the TFBS of each TF may be important for its ability to regulate one specific class of genes. Second, we ran the same enrichment tests using a database of 346 protein-DNA interactions that we compiled from 10 different ChIP-chip studies. The top enrichments in this case, shown in Figure 4(b), suggest hypotheses on the cooperation between other proteins and specific types of the TFBS of the TF as characterized by the enriched feature. Since the data include protein-DNA interactions measured in various conditions [5], some enrichments represent TFBSs that are bound by the corresponding TF only in some conditions.

Finally, we used our resulting models to gain insights into the global properties of binding specificities of all yeast transcription factors. To this end, we collected all the dominant features that we learned across all 8 length models, and computed the average weight of features that were learned between each pair of positions of the TFBS. The comparison of this average weight for each combination of positions is shown in Figure 4(c). Intriguingly, although this average represents many different TFs, two prominent signals emerge. First, the strongest dependencies between features exist between features positioned in the center of the site. Second, nearby positions tend to have a higher dependency compared to dependencies that exist between distant positions. From these results, we compiled a general 'consensus' model for representing the dependencies between positions in the TFBSs of the yeast transcription factors, shown in Figure 4(d). Thus, our model provides insights into global properties that are characteristics of TFBS specificities across all yeast TFs.

4.3 Application of FMM to Motif Finder

As a natural extension of our FMM approach, we integrated our FMM model into a basic motif finder application. Our motif sampler takes as input a set of positive sequences, and a set of negative sequences. The algorithm searches for a motif of length L that maximizes the sum of the log-probabilities of the best TFBSs for each positive input sequence. The algorithm works in an iterative manner. It first searches for a sequence of length L that maximizes the ratio between the fractions of positive sequences and of negative sequences that contain it up to one mismatch. It then initializes a model from these L-length sequences that appear in the positive set. Following this initialization, we then use the Expectation Maximization (EM) algorithm to optimize the model. In the "E" step the motif finder selects the maximum likelihood TFBS from each positive sequence, while in the "M" step it learns a new model from these selected sequences. The algorithm stops after convergence is reached or after a maximal number of "EM" steps. After finding a motif, the algorithm removes from each sequence the highest likelihood TFBS, and then searches for a new motif.

Although our motif finder does not yet integrate all the state of the art methodologies for motif finding, we use it to provide an example for the potential of using FMMs instead of PSSMs for the motif finding task. Specifically, we took the 177 sets of at least 25 sequences each, that bind a TF under a specific condition according to the data of Harbison *et al.*[5] as positive sets, and the rest of the sequences as negative set. We used a 5-fold cross validation scheme to evaluate the motif finder using either FMM or PSSM as the motif model. In these runs we used half of the background as input and half for evaluation. We evaluated the performance of the results by evaluating the sum of log-probabilities of the best TFBS for both the positive sequence test set, and for the held out set of background sequences, and compared the difference of the two. For this evaluation, we used the best of motifs number 2 to 5 that the motif finder outputs. As the results presented in Figure 5 show, even with this relatively basic motif finder, in 133 of the 177 (75%) positive sets tested, we

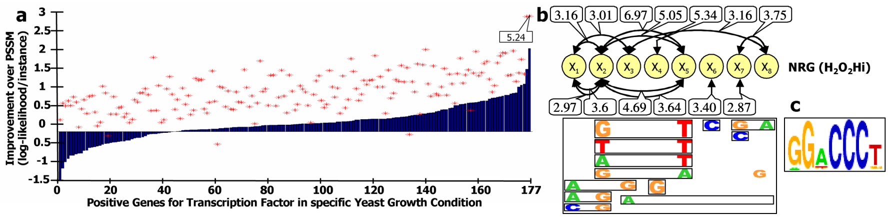


Fig. 5. Motif finder results. (a) The difference between the test average log-likelihood and the background average log-likelihood for the best FMM model (stars), and the difference between this value and the similar value using the best PSSM model (bars). (b) Markov network representation of the dominant features of the best FMM model learned for NRG in High H_2O_2 conditions. (c) PSSM representation of NRG TFBS from Harbison *et al.*

found motifs that gave better average likelihood on the held out positive test set compared to the PSSMs that were learned. Since we used the same framework for learning both the FMM and PSSM models, these results show the potential of our FMM models for the motif finding task, and suggest that combining them within advanced motif finding schemes may yield improved results. An example of FMM motif learned is presented in Figure 5 (b). Comparing it to the PSSM of Harbison *et al.* reveals many similarities (the FMM resembles the complementary of the PSSM), though the FMM also describes some multi-positional features.

5 Conclusion

In this paper we presented *feature motif models* (FMMs), a novel probabilistic method for modeling the binding specificities of TFs. We presented the mathematical foundations of FMMs and showed their advantage over PSSMs in learning motifs from both synthetic and real data. We demonstrated the benefits of using undirected graphical models (Markov networks) for representing important features of TF binding specificities, and suggested a methodology to learn such features from both aligned and unaligned input sequences. We also suggested a methodology for optimizing the objective function, that may give better performance under settings that require approximation.

There are several directions for refining and extending our approach. First, expanding the network structure in which we preform exact inference, and improving our approximate inference abilities, will greatly increase the power of our models. Second, integrating our model into a state of the art (rather than basic) motif finder algorithms may allow us to improve upon existing approaches to the task. Finally, using our models as an improved basic building block, we can integrate it into higher level regulatory models (e.g., [16]) and obtain a much better quantitative understanding of the underlying principles of transcriptional regulatory networks.

Acknowledgments. We thank Tali Sadka for useful discussions. This research was supported by the Israel Science Foundation (Grant No. 645/06), and by ENFIN, a Network of Excellence funded by the European Commission within its FP6 Programme, contract number LSHG-CT-2005-518254.

References

1. Elnitski, L. *et al.*: Locating mammalian transcription factor binding sites: A survey of computational and experimental techniques. Genome Res **16**(12) (2006) 1455–64
2. Bulyk, M.L.: Dna microarray technologies for measuring protein-dna interactions. Current Opinion in Biotechnology **17** (2006) 1–9
3. Maerkl, S.J., Quake, S.R.: A systems approach to measuring the binding energy landscapes of transcription factors. Science **315**(5809) (2007) 233–236
4. Barash, Y., Elidan, G., Friedman, N., Kaplan, T.: Modeling dependencies in protein-dna binding sites. In: RECOMB. (2003)
5. Harbison, C.T. *et al.*: Transcriptional regulatory code of a eukaryotic genome. Nature **431**(7004) (2004) 99–104
6. MacIsaac, K. *et al.*: An improved map of conserved regulatory sites for saccharomyces cerevisiae. BMC Bioinformatics **7** (2006) 113
7. Della Pietra, S. *et al.*: Inducing features of random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence **19**(4) (1997) 380–393
8. Lee, S.I., Ganapathi, V., Koller, D.: Efficient structure learning of Markov networks using L1-regularization. In: NIPS. (2007)
9. Perkins, S., Lacker, K., Theiler, J.: Grafting: fast, incremental feature selection by gradient descent in function space. J. Mach. Learn. Res. **3** (2003) 1333–1356
10. Minka, T.P.: Algorithms for maximum-likelihood logistic regression. Technical Report 758, Carnegie Mellon University (2001)
11. Yedidia, J.S. *et al.*: Generalized belief propagation. In: NIPS. (2000) 689–695
12. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. Royal. Statist. Soc B. **58**(1) (1996) 267–288
13. Ng, A.: Feature selection, l1 vs. l2 regularization, and rotational invariance. In: ICML.(2004)
14. Rothermel, B., Thornton, J., Butow, R.: Rtgp3, a basic helix-loop-helix/leucine zipper protein that functions in mitochondrial-induced changes in gene expression, contains independent activation domains. J Biol Chem. **272** (1997) 19801–7
15. Zeitlinger, J. *et al.*: Program-specific distribution of a transcription factor dependent on partner transcription factor and mapk signaling. Cell **113(3)** (2003) 395–404
16. Segal, E. *et al.*: Genome-wide discovery of transcriptional modules from DNA sequence and gene expression. Bioinformatics **19**(Suppl 1) (2003) S273–82

Network Motif Discovery Using Subgraph Enumeration and Symmetry-Breaking

Joshua A. Grochow and Manolis Kellis

Computer Science and AI Laboratory, M.I.T.
Broad Institute of M.I.T. and Harvard
joshuag@cs.uchicago.edu, manoli@mit.edu

Abstract. The study of biological networks and network motifs can yield significant new insights into systems biology. Previous methods of discovering network motifs – network-centric subgraph enumeration and sampling – have been limited to motifs of 6 to 8 nodes, revealing only the smallest network components. New methods are necessary to identify larger network sub-structures and functional motifs.

Here we present a novel algorithm for discovering large network motifs that achieves these goals, based on a novel symmetry-breaking technique, which eliminates repeated isomorphism testing, leading to an exponential speed-up over previous methods. This technique is made possible by reversing the traditional network-based search at the heart of the algorithm to a motif-based search, which also eliminates the need to store all motifs of a given size and enables parallelization and scaling. Additionally, our method enables us to study the clustering properties of discovered motifs, revealing even larger network elements.

We apply this algorithm to the protein-protein interaction network and transcription regulatory network of *S. cerevisiae*, and discover several large network motifs, which were previously inaccessible to existing methods, including a 29-node cluster of 15-node motifs corresponding to the key transcription machinery of *S. cerevisiae*.

1 Introduction

1.1 Network Motifs

In the past decade, new technologies have enabled the observation and study of networks of thousands and millions of nodes, such as social networks, computer networks, and, notably, *biological networks*, including protein-protein interaction networks [4,5,6], genetic regulatory networks [12,18], and metabolic networks [7]. In order to extract meaningful information from these vast and sometimes noisy datasets, it is necessary to develop methods of computational analysis that are both efficient and robust to errors in the underlying data.

Network motifs – patterns of connectivity that occur significantly more frequently than expected – were introduced by Milo *et al.* [18] and provide one such robust property of biological networks. Network motifs also provide an important tool for understanding the modularity and the large-scale structure of networks

[8,13,20,25]. The importance of network motifs as information-processing modules has been modeled theoretically [12,21] and verified experimentally [8,13,20,25]. Network motifs also have numerous other applications: they have been used to classify networks into "superfamilies" [17], they have been used in combination with machine learning techniques to determine the most appropriate network model for a given real-world network [16], and they have been used to determine which properties to use in parsimony models of phylogeny [19].

Unfortunately, all of these applications are hampered by the limited size of motifs discoverable by current methods. Exact counting methods have only been reported to find motifs up to 4 nodes [18] and motif generalizations up to 6 nodes [10]. Subgraph sampling methods have found motifs up to 7 [9] and 8 nodes [1,16]. The statistical measures developed by Ziv *et al.* [26] are an important step towards larger network structures, but unfortunately lack a one-to-one correspondence with subgraphs, making them potentially difficult to interpret. Motif generalizations [10] are another important step towards these goals, although current methods are still limited to finding motif generalizations of only 6 nodes.

This current size limitation leaves many fundamental questions unanswered, and significant additional insight could be gained by exploring larger subgraphs and finding larger motifs. [1,10]. We should not expect *a priori* that the building blocks of complex networks are as small as 4 nodes, or that the largest significant structures and pathways contain only 8 nodes. What are the fundamental building blocks? How do they combine to form larger structures? [1,10] Do networks which share the same building blocks also share the same combinations of these blocks? [10] How can larger structures be used to distinguish between networks of different types, or between proposed models for a given network? [1]

In this paper, we present a new approach for discovering network motifs. The heart of our algorithm exhaustively assesses the significance of **a single query subgraph** as a potential motif. This can then be applied to all subgraphs of a given size to emulate the behavior of previous exhaustive algorithms, but with an exponential speed-up due to a **novel symmetry-breaking technique** (which is not feasible with previous methods). The symmetry-breaking technique also allows us to write instances of a subgraph to disk as they are found, **further eliminating limitations due to memory usage**. We are thus able to find motifs of up to 15 nodes, to find all instances of subgraphs of 31 nodes, and potentially even larger subgraphs. Although this work is motivated by biological networks, and this paper focuses on the protein-protein interaction (PPI) network and the transcription network of *S. cerevisiae*, our methods are applicable to any network – directed or undirected – and thus to many different fields, even outside the realm of biology.

In this section, we review previous work and give an overview of our algorithm, outlining several novel techniques which apply both to our approach and to previous approaches. In Sec. 2 we present our algorithm in detail. In Sec. 3 we present benchmarks comparing our approach to previous approaches. Additionally, we present data as to the effectiveness of the resulting improvements as applied to both the transcription and PPI networks of *S. cerevisiae*. In Sec. 4, we present some of the larger subgraphs we have discovered. Finally, in Sec. 5 we discuss the significance of these contributions for the understanding of networks in general.

1.2 Limitations of Network-Centric Approaches for Motif Discovery

Two basic techniques have been proposed for identifying network motifs: exact counting [18] and subgraph sampling [1,9,16]. These methods attempt to determine the significance of all or many subgraphs of a given size by comparing their frequency in a given network to their frequency in a random ensemble of networks with similar properties to the original. To determine which subgraphs are motifs, subgraph sampling [1] is an effective and efficient approach, and has been used to evaluate the significance of larger subgraphs than can be evaluated by the exact counting method.

Most methods for finding DNA *sequence motifs* scan or sample a sequence pattern from a genome. Similarly, previous techniques for finding network motifs scan or sample subgraphs from a network, and count the number of occurrences of each subgraph encountered. (This process is then repeated for each network in a random ensemble resembling the initial network, and the counts are compared.) For the discovery of DNA sequence motifs, this general methodology is very efficient, because sequence motifs can be efficiently hashed based on their content. Thus a single linear scan of the genome suffices to exhaustively count all possible substrings of a given size, *regardless of the size of the substrings.*

In contrast, for the discovery of network motifs, enumerating all subgraphs of a given size is in general *exponential in the number of nodes of the subgraphs.* Additionally, there is no known efficient algorithm that correctly identifies two graphs as isomorphic or not. (The *graph isomorphism problem* is not known to be either in P or to be NP-complete.) This intrinsic difference in complexity between discovering *sequence* motifs and discovering *network* motifs makes traditional network-scanning methodologies inefficient for network motif discovery.

1.3 Distinguishing Features of the New Algorithm

To avoid these limitations of the traditional network-centric approaches, we have taken a motif-centric approach which has several attractive features, outlined here. Features 1-3 are specific to motif-centric methods, while features 4 and 5 can also benefit traditional network-centric methods.

(1) Searching for a single query graph. To avoid the increased complexity of subgraph enumeration (in the absence of an appropriate hashing scheme) our algorithm works by exhaustively searching for the instances of a *single query graph* in a network. (To find all motifs of a given size we couple this search with subgraph enumeration, using McKay's `geng` and `directg` tools [15]). Even though the *subgraph isomorphism problem* – finding a given graph as a subgraph of a larger network – is known to be NP-complete, several algorithmic improvements

enable this search to be carried out effectively in practice, even for subgraphs up to 31 nodes (and potentially even more).

(2) Mapping instead of enumerating. Rather than enumerating all connected subgraphs of a given size and testing to see whether each is isomorphic to the query graph, our algorithm attempts to map the query graph onto the network in all possible ways. We developed this technique for subgraph isomorphism independently, but subsequently identified a prior use [23].

(3) Taking advantage of subgraph symmetries. We introduce a technique that *avoids spending time finding a subgraph more than once* due to its symmetries. This technique improves the speed of our method by a factor exponential in the size of the query subgraph (Table 1). Moreover, since each instance is discovered exactly once, our algorithm can write instances to disk as they are found, greatly improving memory usage.

(4) Improved isomorphism testing. Our isomorphism test takes into account the degree of each node, and the degrees of each node's neighbors, leading to marked improvements over current motif-finding algorithms, which use exhaustive graph isomorphism tests.

(5) Subgraph hashing. When examining all subgraphs of a given size we hash the graphs based on their degree sequences, which leads to a significant improvement in the number of isomorphism tests needed. In a *directed* network, we group the query graphs based on their *undirected* isomorphism types, find all instances, and then go back to the directed network and divide these instances into their directed isomorphism types.

2 Description of the Algorithm

For clarity, we first present the basic mapping algorithm for subgraph isomorphism, without taking into account the symmetries of the query graph. In Sec. 2.2 we incorporate our symmetry-breaking technique into the algorithm. In the pseudo-code, we identify statements used solely for symmetry-breaking by enclosing them in square brackets. Finally, In Sec. 2.3 we incorporate our technique into two new methods of finding motifs.

Throughout this section, G will denote the network being searched and H will denote the query subgraph. We say that a node g of G can *support* a node h of H if we cannot rule out a subgraph isomorphism from H into G which maps h to g based on the degrees of h and g and the degrees of their neighbors. (Other constraints could also be used here, but these two proved effective and simple to implement.) This notion of support is used to exclude inconsistent candidate maps during the backtracking search.

2.1 Finding a Given Subgraph (Subgraph Isomorphism)

We start by presenting the algorithm without symmetry-breaking. Note that symmetry-breaking is not required for correctness of the algorithm.

```
FindSubgraphInstances(H,G):
Finds all instances of query graph $H$ in network $G$
Start with an empty set of instances.
[Find Aut($H$). Let $H_E$ be the equivalence representatives of $H$.]
[Find symmetry-breaking conditions $C$ for $H$ given $H_E$ and Aut($H$).]
Order the nodes of $G$ by increasing degree and
 then by increasing neighbor degree sequence.
For each node $g$ of $G$
 For each node $h$ of $H$ [$H_E$] such that $g$ can support $h$
  Let $f$ be the partial map associating $f(h) = g$.
  Find all isomorphic extensions of $f$ [up to symmetry]
   i.e. call IsomorphicExtensions($f$,$H$,$G$[,$C(h)$]).
  Add the images of these maps to the set of all instances.
 Remove $g$ from $G$.
Return the set of all instances.
```

FindSubgraphInstances includes the *images* of the maps in the list of instances, thus merging all maps which differ only by a symmetry of H. (Without symmetry-breaking, the algorithm spends additional time finding several distinct maps to a single subgraph.)

IsomorphicExtensions is a backtracking search to find all isomorphisms from H into G. As is standard in backtracking searches, the algorithm uses the most constrained neighbor to eliminate maps that cannot be isomorphisms: that is, the neighbor of the already-mapped nodes which is likely to have the fewest possible nodes it can be mapped to. First we select the nodes with the most already-mapped neighbors, and amongst those we select the nodes with the highest degree and largest neighbor degree sequence.

For each call to IsomorphicExtensions, f is extended by a single node. Each time an extension is made, the algorithm ensures that the newly mapped node is appropriately connected to the already-mapped nodes. Thus when IsomorphicExtensions returns a map, it is guaranteed to be an isomorphism.

We have effectively pushed the isomorphism testing of previous exhaustive methods into IsomorphicExtensions, which allows the isomorphism test to abort early. The ability to abort early when finding instances of a particular query graph presents significant savings over previous methods.

2.2 Exploiting Subgraph Symmetries to Speed Up the Search

Due to symmetries, a given subgraph of G may be mapped to a given query graph H multiple times. For example, the subgraph in Fig. 1 can be mapped to the same 6 nodes in 8 different ways. Thus a simple mapping-based search for a query graph will find each instance of the query graph as many times as the graph has symmetries. To avoid this, we compute and enforce several symmetry-breaking conditions, which ensure that there is a *unique* map from the query graph H to each instance of H in G, so that our search only spends time finding each instance once.

```
IsomorphicExtensions(f,H,G[,C(h)]:
Finds all isomorphic extensions of partial map $f : H \rightarrow G$ [satisfying C(h)]
Start with an empty list of isomorphisms.
Let $D$ be the domain of $f$.
If $D = H$, return a list consisting solely of $f$. (Or write to disk.)
Let $m$ be the most constrained neighbor of any $d \in D$
 (constrained by degree, neighbors mapped, etc.)
For each neighbor $n$ of $f(D)$
 If there is a neighbor $d \in D$ of $m$ such that $n$ is not neighbors with $f(d)$,
  or if there is a non-neighbor $d \in D$ of $m$ such that $n$ is neighbors with $f(d)$
  [or if assigning $f(m) = n$ would violate a symmetry-breaking condition in $C(h)$],
 then continue with the next $n$.
 Otherwise, let $f' = f$ on $D$, and $f'(m) = n$.
 Find all isomorphic extensions of $f'$.
 Append these maps to the list of isomorphisms.
Return the list of isomorphisms.
```

The symmetries of a graph H are known as automorphisms (self-isomorphisms), and the group of automorphisms of H is denoted $\mathrm{Aut}(H)$. For a set A of automorphisms, two nodes are said to be "A-equivalent" if there is some automorphism in A which maps one to the other, or simply "equivalent" if $A = \mathrm{Aut}(H)$. We denote the A-equivalence of two nodes n_1, n_2 by $n_1 \sim_A n_2$. This equivalence relation partitions the nodes of H into equivalence classes. Since starting a map from two equivalent nodes is unnecessary and wasteful, FindSubgraphInstances uses a set consisting of one representative from each equivalence class of H.

The symmetry-breaking conditions are based on labellings of the nodes of H by integers, represented as maps from $H \rightarrow \mathbf{Z}$. Let $\ell : G \rightarrow \mathbf{Z}$ be a labelling of the nodes of G by *distinct* integers. Then each map $f : H \rightarrow G$ generates a labelling $L : H \rightarrow \mathbf{Z}$, given by $L(n) = \ell(f(n))$ for nodes $n \in H$. Thus, conditions on labellings of H translate into restraints on maps from H into G.

Given a set of conditions C, we say an automorphism α *preserves the conditions* C if, given a labelling L_1 of H which satisfies C, the corresponding labelling $L_2 : H \rightarrow \mathbf{Z}$ given by $L_2(n) = L_1(\alpha(n))$ also satisfies C. We are thus searching for conditions C such that the only automorphism which preserves C is the identity. This ensures there will be exactly one map from H onto each of its instances in G which satisfies the conditions.

To find these conditions, we pick an $\mathrm{Aut}(H)$-equivalence class $\{n_0, \ldots, n_k\}$ of nodes of H, and we impose the condition $L(n_0) < \textsc{min}(L(n_1), \ldots, L(n_k))$. Any automorphism must send n_0 to one of the n_i, since these are all of the nodes equivalent to n_0. But to preserve this condition, an automorphism must send n_0 to itself. Then we continue recursively, replacing $\mathrm{Aut}(H)$ with the set A of automorphisms which send n_0 to itself. For example, see Fig. 1.

Because FindSubgraphInstances starts with a particular node, we can consider that node already fixed. (Note that the version of FindSubgraphInstances which uses symmetry-breaking only iterates over a set of equivalence class representatives, and not over all nodes of H.) Thus for each representative

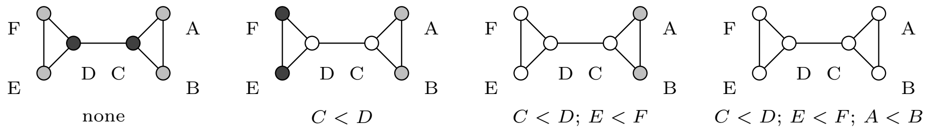


Fig. 1. Finding conditions that will break all the symmetries of a 6-node graph. White nodes are fixed by any automorphism preserving the indicated conditions, and other nodes are shaded according to their equivalence class under the automorphisms which preserve the indicated conditions.

used by FindSubgraphInstances, SymmetryConditions must generate a series of symmetry-breaking conditions which start by fixing that node.

To find the automorphisms of H, we use IsomorphicExtensions *without* symmetry-breaking, which returns an exhaustive list of all isomorphisms from H to itself. To find the automorphisms which fix a node or a set of nodes, the algorithm filters this list in a single pass.

Finding the automorphisms of a graph is thought to be computationally expensive[1], but in practice we have found this is far from the bottleneck in motif-finding algorithms. We were able to *exhaustively* find the automorphisms of all 11,117 8-node undirected graphs in under 30 seconds on a standard laptop, and McKay's tools [14] can find all the automorphisms of very large graphs very rapidly (e.g. some graphs with thousands of nodes and millions of automorphisms, in less than one second on a standard laptop).

```
SymmetryConditions:
Finds symmetry-breaking conditions for $H$ given $H_E, \mathbf{Aut}(H)$
Let $M$ be an empty map from equivalence representatives to sets of conditions.
For each $n \in H_E$
  Let $C$ be an empty set of conditions.
  $n' \leftarrow n$, and $A \leftarrow \mathrm{Aut}(H)$.
  Do until $|A| = 1$:
    Add "$\textsc{label}(n') < \textsc{min}\{\textsc{label}(m) | m \sim_A n'$ and $m \neq n'\}$" to $C$.
    $A \leftarrow \{f \in A | f(n') = n'\}$.
    Find the largest $A$-equivalence class $E$.
    Pick $n' \in E$ arbitrarily.
  Let $M(n) = C$.
Return $M$.
```

2.3 Subgraph Evaluation and Network Motif Discovery

To find network motifs we enumerate candidate subgraphs H (exhaustively or by sampling), and evaluate each candidate based on its instances.

[1] Finding graph automorphisms is at least as hard as determining if two graphs are isomorphic. Like the graph isomorphism problem, the graph automorphism problem is not known to be either in P or to be NP-complete.

Evaluating candidate subgraphs. We find all instances of a query graph H in the network G, as well as in a random ensemble of networks with the same degree distribution and same distribution of 3-node subgraphs as G.[2] We evaluate the overrepresentation of the query graph H based on the z-score of its abundance in G against the distribution of its abundance in the random ensemble, as in [18,21].

Exhaustive subgraph enumeration. Our method can be used to find all instances of subgraphs of a given size, similar to previous exhaustive methods. To do this, we generate all non-isomorphic graphs of a particular size using McKay's `geng` and `directg` tools [15]. Then for each graph, we evaluate its significance as above.

Subgraph sampling. Our method can also be used in combination with subgraph sampling. We sample connected subgraphs (usually relatively large, compared to previous network motifs: 10, 15, or 20 nodes) by picking a node at random, and taking a random walk until we have as many nodes as desired [16]. Then we assess the significance of this subgraph as above.

Sampling subgraphs to find anti-motifs. Some studies have also considered *anti*-motifs: subgraphs which are significantly *under*represented compared to randomized versions of the network. To use a sampling method to find anti-motifs, it might be more fruitful to sample initial subgraphs from the random ensemble rather than the network being studied. Anti-motifs will be more prevalent in the ensemble than in the target network, and thus are more likely to be discovered by sampling from the ensemble.

3 Results and Evaluation

We applied our algorithm to the PPI network (1379 nodes, 2493 edges) [4] and transcription network (685 nodes, 1052 edges) [2] of *S. cerevisiae* and compared its performance to previous methods of motif disccovery.

Comparison with previous methods: time. We compare the time requirements of our method to those of Milo *et al.* [18] (Fig. 2). We make this comparison on the undirected PPI network of *S. cerevisiae* [4], by exhaustively counting subgraphs up to 7 nodes.

We implemented both our algorithm and two versions of the Milo *et al.* algorithm [18]: both as originally presented [18], and also by additionally hashing subgraphs by their degree sequence (Sec. 1.3). Fig. 2 shows that our algorithm provides an *exponential* improvement in time, even compared to the modified version of the previous algorithm [18].

[2] Although Shen-Orr *et al.*[21] use a model in which the distribution of $(n-1)$-node subgraphs is preserved when looking for n-node motifs, they only applied this to the case $n = 4$, and we have found it computationally infeasible to preserve this distribution for $n > 4$. Nonetheless, we have found it fruitful to preserve the distribution of 3-node subgraphs, regardless of n.

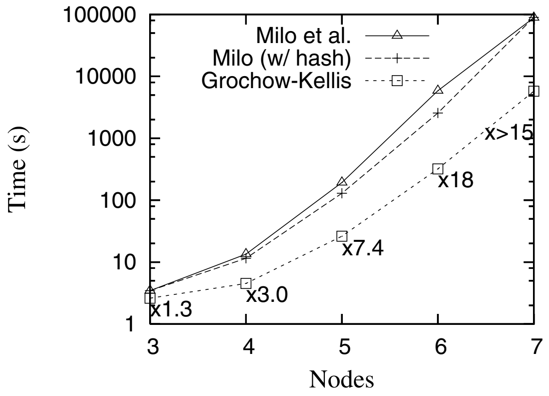


Fig. 2. The runtimes of the original algorithm of Milo *et al.* [18], an improved version of their algorithm, and our new algorithm, as applied to the undirected PPI network of *S. cerevisiae* [4]. The speed-up from the original algorithm of Milo *et al.* [18] to our algorithm is indicated. (Note: the values for 7 nodes for the two variants of Milo *et al.*'s algorithm are underestimates: the program ran out of memory before finishing.)

Comparison with previous methods: space. Our method gains an exponential memory advantage over previous exhaustive methods by not keeping a list of previously visited subgraphs. In the previous exact counting method [18], a list of the subgraphs encountered at each node is necessary in order to avoid duplication, even when the instances of the subgraphs are not desired as output. Thus the space required by the previous method is proportional to the number of subgraphs of a given size going through a given node, which can be exponential in the size of the subgraphs. Because our method does not need to keep such a list, its asymptotic memory requirements are determined by the maximum depth of recurion of IsomorphicExtensions, which is linear in the size of the query graph. Our method thus uses exponentially less space than previous exhaustive methods.

Disk usage. Furthermore, our algorithm uses no more memory to find a list of all instances than to simply count the instances. Since each instance is encountered exactly once, it can be written to disk and removed from active memory as soon as it is encountered, using effectively no additional memory.

Improvement due to symmetry-breaking. The main reason for these improvements is our novel symmetry-breaking technique. Symmetry-breaking ensures that each instance is discovered exactly once, so our method does not have to check a list of the subgraphs previously encountered at a node in order to avoid duplicate counting, while the previous method of exact counting does. Table 1 quantifies this contribution explicitly, showing that the average number of automorphisms of graphs weighted by their occurences in the PPI network and regulatory network of yeast – i.e. the savings gained by symmetry-breaking – appears to grow exponentially.

Table 1. The number of subgraphs encountered by our algorithm with and without symmetry-breaking (including multiple encounters for the version without symmetry-breaking). The improvement factor is exactly the average number of automorphisms of subgraphs of the associated size.

Nodes	Undirected PPI Network			Directed Regulatory Network		
	Total Subgraphs Searched	With Symmetry-Breaking	Improvement	Total Subgraphs Searched	With Symmetry-Breaking	Improvement
3	3.7×10^4	1.1×10^4	$\times3.13$	2.6×10^4	1.3×10^4	$\times2.02$
4	4.0×10^5	7.0×10^4	$\times5.77$	9.7×10^5	1.8×10^5	$\times5.41$
5	4.4×10^6	4.1×10^5	$\times10.9$	4.4×10^7	2.5×10^6	$\times18.0$
6	5.1×10^7	2.3×10^6	$\times22.2$	2.3×10^9	3.2×10^7	$\times73.3$
7	5.7×10^8	1.2×10^7	$\times46.3$	1.3×10^{11}	4.0×10^8	$\times334$
8	6.4×10^9	6.6×10^7	$\times96.2$	—	—	—

4 Discovered Motifs and Their Biological Significance

Discovered motifs. We exhaustively evaluated all candidate motifs and anti-motifs up to 7 nodes in the PPI network of *S. cerevisiae*[4] (1379 nodes, 2493 edges). We used a random ensemble of networks with the same degree distribution and the same distribution of 3-node subgraphs as the PPI network.[3] The most significant subgraphs tend to be motifs rather than anti-motifs: using a z-score cutoff of 4.0, only 3 of the 54 significant subgraphs of size at most 7 were anti-motifs. Two of the motifs were trees, and the most dense motif had 18 edges. Most of the significant graphs were of moderate density: the mean number of edges for 7-node motifs and anti-motifs is 11.49 ± 2.89.

Large discovered motifs. We also discovered larger motifs by first sampling connected subgraphs from the PPI network of *S. cerevisiae*, and then assessing their significance using our method. We sampled approximately 100 connected subgraphs of 15 and 20 nodes, and found several motifs. One such 15-node motif (Fig. 3) represents a common connectivity pattern found within the transcriptional machinery of *S. cerevisiae* (see discussion below).

Clustering of discovered motifs and larger network structures. We noted that almost all of the larger subgraphs we evaluated have large numbers of overlapping instances, which become apparent since our method reports all network instances of a discovered motif. To quantify this property, we developed a subgraph clustering score, based on the number of subgraph instances overlapping a given node, averaged over all nodes in any subgraph instance. We applied this score to evaluate the clustering properties of all discovered motifs, and we found that indeed some of the most abundant motifs show striking clustering properties.

[3] See footnote 2.

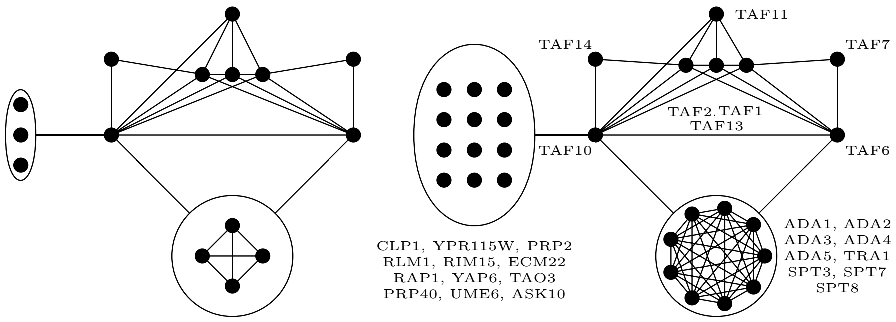


Fig. 3. A motif of 15 nodes and 34 edges (left). An edge from a group of nodes to a node n indicates that each node in the group is connected to n. This motif appears 27,720 times in the PPI network of *S. cerevisiae*[4], and does not appear at all in random ensembles which preserve the degree distribution and the distribution of 3-node subgraphs. All 27,720 instances are clustered into a total of 29 nodes (right), corresponding to the cellular transcription machinery.

The clustered instances frequently reveal important larger network structures. For example, the 15-node motif of Fig. 3 occurs 27,720 times in a single subnetwork of 29 nodes, part of the core transcription machinery of *S. cerevisiae.* This includes a complete 11-node graph (including the two central hubs) corresponding to the SAGA complex, and consisting almost entirely of chromatin modification and histone acetylation factors an 8-node core (shared by all instances of the 15-node motif) corresponding to the TFIID complex, and 12 attachments, which are known activators and suppressors of these two complexes [11]. Similarly, the subgraph of 20 nodes shown in Fig. 4 occurs 5,020 times in a total of 31 nodes, enriched in cell-cycle regulation.

The role of combinatorial effects. The extreme clustering properties of the most abundant motifs appear to result from combinatorial connectivity patterns prevalent in larger network structures. For example, all 27,720 instances of the 15-node motif in Fig. 3 result by choosing 3 attachments from the left and 4 attachments from the the bottom of Fig. 3 ($\binom{12}{3}\binom{9}{4} = 27,720$), and similarly for the 5,020 instances of the 20-node subgraph in Fig. 4. Additionally, in the random ensemble, these combinatorially appearing structures occur either thousands of times, or not at all – they almost never occur just a few or a few hundred times.

Although motif clustering has previously been observed [3] and demonstrated analytically [24], previous motifs studied do not have enough nodes to exhibit the extreme combinatorial clustering we observed for large subgraphs (at least 15 nodes). The magnitude of this combinatorial clustering effect brings into question the current definition of network motif, when applied to larger structures. We propose that additional statistics, either alone or in combination, might be well-suited to identify larger meaningful network structures: our subgraph clustering score, the total number of nodes covered by all instances of the query graph, the

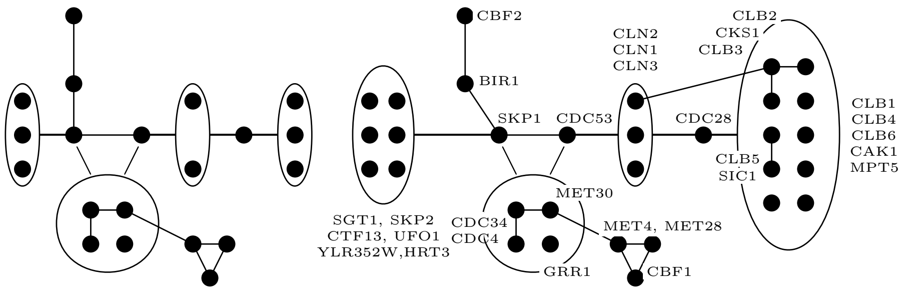


Fig. 4. A subgraph of 20 nodes and 27 edges (left). An edge from a group of nodes to a node n indicates that each node in the group is connected to n. This subgraph appears 5,020 times in the PPI network of *S. cerevisiae* [4]. All 5,020 instances are clustered into a total of 31 nodes (right), enriched in cell-cycle regulation.

total number of edges, and the weighting of the number of nodes/edges based on the number of overlapping instances. All of these statistics can be easily calculated using our algorithm, since it finds and stores all motif instances, and these will be the subject of future studies.

5 Discussion

We presented a novel approach to the discovery of network motifs, based on a solution to the subgraph isomorphism problem that uses a new symmetry-breaking technique, an improved isomorphism test, and hashing based on degree sequences. Several of the techniques presented in our algorithm can also be used in previous algorithms, and lead to significant improvements.

We implemented our algorithm and used it to find significant structures of 15 and 20 nodes in the PPI network and the regulatory network of *S. cerevisiae*, where previous methods had been limited to motifs of 6 and 8 nodes. Using our approach to motif-finding, we re-discovered the cellular transcription machinery – as a 29-node cluster of 15-node motifs – based solely on the structure of the protein interaction network.

Previous methods of motif discovery were network-centric, and could therefore not take advantage of subgraph symmetries. By using a motif-centric algorithm instead, we are able to use symmetry-breaking to get an exponential improvement.

5.1 Applications and Advantages of the New Method

(1) Finding larger motifs. Our improvements have enabled the exhaustive discovery of motifs up to 7 nodes. To find even larger motifs, we sample a connected subgraph as in [16], and then find *all* its instances and assess its significance using our method. This technique has enabled us to find motifs up to 15 nodes and examine subgraphs up to 31 nodes.

(2) Querying a particular subgraph. Our method can be used to query whether a particular subgraph is significant, whereas previous methods can only do this

by examining all subgraphs of the same size, which quickly becomes prohibitive for even moderate sizes. This technique could be used to explore *in silico* the prevalence of a subgraph of interest, identified experimentally (e.g. known pathways), computationally (e.g. motif generalizations [10]), or genetically.

(3) Exploring motif clustering. Because our algorithm finds all instances of a given subgraph, it can be used to explore how these instances cluster together to form larger structures. For example, after finding a 15-node motif, we were able to determine that all of its 27,720 instances clustered in 29 nodes (Fig. 3).

(4) Time and space. Our method, applied to all subgraphs of a given size, takes exponentially less time than previous methods, even when we implement the previous method with our hashing scheme (Sec. 3). Additionally, there are essentially no space limitations on our method: since each instance is found exactly once due to our symmetry-breaking technique, it can be written to disk and removed from active memory as soon as it is found.

(5) Parallelization. Our method is more easily parallelizable than previous motif-finding methods, since each subgraph can be counted on a separate processor. We have found this attribute to be very useful, and we believe other researchers will as well, as cluster computing becomes commonplace in the computational biology community.

5.2 Clustering Properties of Large Subgraphs and Motifs

We revealed that larger subgraphs tend to cluster together *combinatorially* – that is, all instances share a significant core of nodes, and each instance represents a choice of attachments to these core nodes. This combinatorial clustering brings into question the relevance of the standard definition of network motif for large subgraphs of 15 nodes or more. We proposed several statistics which may be more appropriate in this domain.

Finally, we mention that the statistics of Ziv *et al.* [26] may not suffer from these combinatorial effects. The main drawback of these statistics is their lack of one-to-one correspondence with subgraphs. In combination with our algorithm, however, the large subgraphs encompassed by these statistics could be further explored, allowing for a clearer interpretation of the most significant statistics.

Moving forward, we expect the network motifs and methodology presented here will open a window into the large structures and global organization of biological and other networks.

Acknowledgements. The authors would like to thank Pouya Kheradpour, Mike Lin, Matt Rasmussen, Alex Stark, and Radek Szklarczyk (all at the M.I.T. Computer Science and AI Laboratory) for many useful and interesting discussions.

All algorithms were implemented in Java using the Java Universal Networks and Graphs (JUNG) framework [22]. Our software is available on request. McKay's `geng` and `directg` tools [15] were used to enumerate all graphs of a given size. Much of the computational work was carried out on the compute cluster at the Broad Institute of M.I.T. and Harvard.

This work was supported in part by startup funds from Professor Kellis.

References

1. K. Baskerville and M. Paczuski. Subgraph ensembles and motif discovery using a new heuristic for graph isomorphism, 2006. arxiv.org:q-bio/0606023.
2. M. C. Costanzo, M. E. Crawford, J. E. Hirschman, J. E. Kranz, P. Olsen, L. S. Robertson, M. S. Skrzypek, B. R. Braun, K. L. Hopkins, P. Kondu, C. Lengieza, J. E. Lew-Smith, M. Tillberg, and J. I. Garrels. Ypd(tm), pombepd(tm), and wormpd(tm): model organism volumes of the bioknowledge(tm) library, an integrated resource for protein information. *Nucleic Acids Res.*, 29:75–79, 2001.
3. R. Dobrin, Q. K. Beg, A.-L. Barabási, and Z. N. Oltvai. Aggregation of topological motifs in the *Escherichia coli* transcriptional regulatory network. *BMC Bioinformatics*, 5:10, Jan 2004.
4. J.-D. J. Han, N. Bertin, T. Hao, D. S. Goldberg, G. F. Berriz, L. V. Zhang, D. Dupuy, A. J. M. Walhout, M. E. Cusick, F. P. Roth, and M. Vidal. Evidence for dynamically organized modularity in the yeast protein-protein interaction network. *Nature*, 430(6995):88–93, Jul 2004.
5. A. Jaimovich, G. Elidan, H. Margalit, and N. Friedman. Towards an integrated protein-protein interaction network: a relational markov network approach. *J. Comp. Bio.*, 13:145–164, 2006.
6. H. Jeong, S. Mason, A.-L. Barabási, and Z. N. Oltvai. Centrality and lethality of protein networks. *Nature*, 411, 2001.
7. H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabási. The large-scale organization of metabolic networks. *Nature*, 407, 2000.
8. S. Kalir, J. McClure, K. Pabbaraju, C. Southward, M. Ronen, S. Leibler, M. G. Surette, and U. Alon. Ordering genes in a flagella pathway by analysis of expression kinetics from living bacteria. *Science*, 292(5524):2080–2083, Jun 2001.
9. N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, 20(11):1746–1758, Jul 2004. Evaluation Studies.
10. N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. Topological generalizations of network motifs. *Phys. Rev. E*, 70:031909, 2004.
11. T. I. Lee and R. A. Young. Transcription of eukaryotic protein-coding genes. *Annu. Rev. Genet.*, 34:77–137, 2000.
12. S. Mangan and U. Alon. Structure and function of the feed-forward loop network motif. *PNAS*, 100(21):11980–11985, Oct 2003.
13. S. Mangan, A. Zaslaver, and U. Alon. The coherent feedforward loop serves as a sign-sensitive delay element in transcription networks. *J. Mol. Biol.*, 334(2):197–204, Nov 2003.
14. B. D. McKay. Practical graph isomorphism. In *Proceedings of the Tenth Manitoba Conference on Numerical Mathematics and Computing, Vol. I (Winnipeg, Man., 1980)*, volume 30, pages 45–87, 1981. http://cs.anu.edu.au/~bdm/nauty/.
15. B. D. McKay. Isomorph-free exhaustive generation. *J. Algorithms*, 26:306–324, 1998.
16. M. Middendorf, E. Ziv, and Chris H. Wiggins. Inferring network mechanisms: the Drosophila melanogaster protein interaction network. *PNAS*, 102(9):3192–3197, Mar 2005.
17. R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, and U. Alon. Superfamilies of evolved and designed networks. *Science*, 303(5663):1538–1542, Mar 2004.

18. R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, Oct 2002.
19. T. M. Przytycka. An important connection between network motifs and parsimony models. In *RECOMB 2006*, pages 321–335, 2006.
20. M. Ronen, R. Rosenberg, B. I. Shraiman, and U. Alon. Assigning numbers to the arrows: parameterizing a gene regulation network by using accurate expression kinetics. *Proc Natl Acad Sci U S A*, 99(16):10555–10560, Aug 2002.
21. S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of Escherichia coli. *Nature Genetics*, 31(1):64–68, May 2002.
22. JUNG Framework Development Team. Jung: The java universal network/graph framework, 2005.
23. J. R. Ullman. An algorithm for subgraph isomorphism. *J. Assoc. Comp. Mach.*, 23(1):31–42, Jan 1976.
24. A. Vazquez, R. Dobrin, D. Sergi, J.-P. Eckmann, Z. N. Oltvai, and A.-L. Barabasi. The topological relationship between the large-scale attributes and local interaction patterns of complex networks. *PNAS*, 101(52):17940–17945, Dec 2004.
25. A. Zaslaver, A. E. Mayo, R. Rosenberg, P. Bashkin, H. Sberro, M. Tsalyuk, M. G. Surette, and U. Alon. Just-in-time transcription program in metabolic pathways. *Nature Genetics*, 36(5):486–491, May 2004.
26. E. Ziv, R. Koytcheff, M. Middendorf, and C. Wiggins. Systematic identification of statistically significant network measures. *Phys. Rev. E*, 71:016110, 2005.

Nucleosome Occupancy Information Improves *de novo* Motif Discovery

Leelavati Narlikar*, Raluca Gordân*, and Alexander J. Hartemink

Department of Computer Science, Duke University, Durham, NC 27708-0129
{lee,raluca,amink}@cs.duke.edu

Abstract. A complete understanding of transcriptional regulatory processes in the cell requires identification of transcription factor binding sites on a genome-wide scale. Unfortunately, these binding sites are typically short and degenerate, posing a significant statistical challenge: many more matches to known transcription factor binding sites occur in the genome than are actually functional. Chromatin structure is known to play an important role in guiding transcription factors to those sites that are functional. In particular, it has been shown that active regulatory regions are usually depleted of nucleosomes, thereby enabling transcription factors to bind DNA in those regions [1]. In this paper, we describe a novel algorithm which employs an informative prior over DNA sequence positions based on a discriminative view of nucleosome occupancy; the nucleosome occupancy information comes from a recently published computational model [2]. When a Gibbs sampling algorithm with our informative prior is applied to yeast sequence-sets identified by ChIP-chip [3], the correct motif is found in 50% more cases than with an uninformative uniform prior. Moreover, if nucleosome occupancy information is not available, our informative prior reduces to a new kind of prior that can exploit discriminative information in a purely generative setting.

1 Introduction

Finding functional DNA binding sites of transcription factors (TFs) on a genome-wide scale is a crucial step in understanding transcriptional regulation. Despite an explosion of data about TF binding from high-throughput technologies like ChIP-chip [3, 4, and many more], DIP-chip [5], PBM [6], and gene-expression arrays [7, 8, and many more], *de novo* motif finding remains a difficult problem. The fundamental reason for this is that the binding sites of most TFs are short, degenerate sequences which occur frequently in the genome by chance. The ‘signal’ of functional sites (which are bound *in vivo*) is overwhelmed by the ‘noise’ due to the non-functional sites (which are not bound *in vivo*). Distinguishing functional sites from non-functional ones, and inferring the true motif recognized by the TF, is thus a challenge.

Many probabilistic motif discovery methods have been developed to tackle the problem of motif discovery [9, 10]. The standard approach is to look for a pattern common

* These authors contributed equally to this work.

to the bound sequences that is statistically enriched with respect to the background distribution of all intergenic sequences. If, in addition to the set of bound sequences, a set of unbound sequences is available, a stronger criterion might insist that the pattern be able to discriminate between the two sets [11, 12, 13, 14, 15]. Unfortunately, due to the low signal-to-noise ratio of binding sites mentioned earlier, these methods generally suffer from low specificity and sensitivity [16].

Often, DNA sequences that match known TF motifs do not appear to be functional *in vivo* TF binding sites. One explanation is that not all parts of the genome are equally accessible to TFs *in vivo*. In particular, since DNA is wound over histone octamers called nucleosomes, the positioning of these nucleosomes provides a possible mechanism for differential access of TFs to potential binding sites [1, 2, 17, 18, 19, 20]. Our goal in this paper is to leverage knowledge about nucleosome positioning to improve *de novo* motif finding.

If we knew exactly what parts of the genome were occupied by nucleosomes in the exact environmental conditions for which we have *in vivo* TF binding data, we could bias our search for TF binding sites to the areas that are free of nucleosomes. Unfortunately, no high-resolution nucleosome occupancy data is available for any organism on a whole-genome scale. In the case of yeast, Yuan *et al.* [20] have reported high-resolution nucleosome occupancy data using tiling arrays, but only for chromosome III. On the other hand, Lee *et al.* [1] have published occupancy data for the whole genome, but it is of low resolution: they report only the average occupancy over each intergenic region.

Recently, Segal *et al.* [2] developed a computational model based on high-quality experimental nucleosome binding data to calculate the average nucleosome occupancy at each nucleotide position in the yeast and chicken genomes. This occupancy is purported to be intrinsic to the DNA sequence, and hence independent of *in vivo* conditions. The authors claim that their predictions explain about 50% of observed *in vivo* nucleosome positions. Here, we use predictions from their model to build informative priors over DNA sequence positions that can be used to improve the accuracy of motif finding. We formulate two different nucleosome occupancy priors: the first is based directly on the predictions of Segal *et al.*, while the second adopts a discriminative perspective, comparing nucleosome occupancy in bound versus unbound sequences. When nucleosome occupancy information is not available, the first prior simplifies to an uninformative uniform prior, but the second simplifies to a new kind of informative prior that can exploit discriminative information in a purely generative setting. This represents a novel approach to discriminative motif discovery that retains the computational benefits of a generative formulation. As we shall see, each of our three informative priors improves upon the uninformative uniform prior.

We choose Gibbs sampling as the search method in our algorithms, but in principle, the priors can be used with any search strategy. Our choice of a position specific scoring matrix (PSSM) [21] as a model for the motif is also arbitrary since our priors can be applied while learning any type of motif model. The purpose of this paper is not to demonstrate the benefits of one search strategy over another or one motif model over another, but to demonstrate the utility of nucleosome occupancy data in constructing informative priors for motif discovery.

2 Motif Discovery

In this section, we describe the popular generative formulation of the problem of motif discovery, derive the objective function we seek to optimize, and explain the search methodology that we use to optimize this objective function.

2.1 Sequence Model and Objective Function

Assume we have n DNA sequences $\boldsymbol{X}_1$ to $\boldsymbol{X}_n$ believed to be commonly bound by some TF. Although in reality a sequence might have multiple binding sites, for simplicity we model only one binding site in each sequence. Because the experimental data might be erroneous, we also model the possibility of some sequences not having any binding site. This is analogous to the zero or one occurrence per sequence (ZOOPS) model in MEME [22]. Let $\boldsymbol{Z}$ be a vector of length n denoting the starting location of the binding site in each sequence: $Z_i = j$ if there is a binding site starting at location j in $\boldsymbol{X}_i$ and we adopt the convention that $Z_i = 0$ if there is no binding site in $\boldsymbol{X}_i$. We assume that the TF motif can be modeled as a PSSM of length W while the rest of the sequence follows some background model parameterized by $\boldsymbol{\phi}_0$. The PSSM can be described by a matrix $\boldsymbol{\phi}$ where $\phi_{a,b}$ is the probability of finding base b at location a within the binding site for $1 \leq b \leq 4$ and $1 \leq a \leq W$.

Thus if the sequence $\boldsymbol{X}_i$ is of length m_i, and $\boldsymbol{X}_i$ contains a binding site at location Z_i, we can compute the probability of the sequence given the model parameters as:

$$P(\boldsymbol{X}_i \mid \boldsymbol{\phi}, Z_i > 0, \boldsymbol{\phi}_0) = P(X_{i,1}, \ldots X_{i,Z_i-1} \mid \boldsymbol{\phi}_0) \times \left(\prod_{k=1}^{W} \phi_{k,X_{i,Z_i+k-1}} \right) \\ \times P(X_{i,Z_i+W}, \ldots X_{i,m_i} \mid \boldsymbol{\phi}_0)$$

and if it instead does not contain a binding site as:

$$P(\boldsymbol{X}_i \mid \boldsymbol{\phi}, Z_i = 0, \boldsymbol{\phi}_0) = P(X_{i,1}, X_{i,2} \ldots X_{i,m_i} \mid \boldsymbol{\phi}_0)$$

We wish to find $\boldsymbol{\phi}$ and $\boldsymbol{Z}$ that maximize the joint posterior distribution of all the unknowns given the data. Assuming priors $P(\boldsymbol{\phi})$ and $P(\boldsymbol{Z})$ over $\boldsymbol{\phi}$ and $\boldsymbol{Z}$ respectively, our objective function is:

$$\arg\max_{\boldsymbol{\phi},\boldsymbol{Z}} P(\boldsymbol{\phi}, \boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\phi}_0) = \arg\max_{\boldsymbol{\phi},\boldsymbol{Z}} \big(P(\boldsymbol{X} \mid \boldsymbol{\phi}, \boldsymbol{Z}, \boldsymbol{\phi}_0) P(\boldsymbol{\phi}) P(\boldsymbol{Z}) \big) \tag{1}$$

2.2 Optimization Strategy and Scoring Scheme

As others before us have done, we use Gibbs sampling to sample repeatedly from the posterior over $\boldsymbol{\phi}$ and $\boldsymbol{Z}$ with the hope that we are likely to visit those values of $\boldsymbol{\phi}$ and $\boldsymbol{Z}$ with the highest posterior probability. Gibbs sampling is a Markov chain Monte Carlo (MCMC) method that approximates sampling from a joint posterior distribution by sampling iteratively from individual conditional distributions [23]. Applying the collapsed Gibbs sampling strategy developed by Liu [24] for faster convergence, we can integrate out $\boldsymbol{\phi}$ and sample only the Z_i. This results in the following expression for sampling Z_i from its conditional distribution assuming the prior on $\boldsymbol{Z}$ to be independent of the PSSM parameters $\boldsymbol{\phi}$:

$$P(Z_i \mid \boldsymbol{Z}_{[-i]}, \boldsymbol{X}, \boldsymbol{\phi}_0) = \frac{P(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\phi}_0)}{P(\boldsymbol{Z}_{[-i]} \mid \boldsymbol{X}, \boldsymbol{\phi}_0)} = \frac{P(\boldsymbol{Z}) \int\limits_{\boldsymbol{\phi}} P(\boldsymbol{X} \mid \boldsymbol{\phi}, \boldsymbol{Z}, \boldsymbol{\phi}_0) P(\boldsymbol{\phi}) \mathrm{d}\boldsymbol{\phi}}{P(\boldsymbol{Z}_{[-i]}) \int\limits_{\boldsymbol{\phi}} P(\boldsymbol{X} \mid \boldsymbol{\phi}, \boldsymbol{Z}_{[-i]}, \boldsymbol{\phi}_0) P(\boldsymbol{\phi}) \mathrm{d}\boldsymbol{\phi}}$$

where $\boldsymbol{Z}_{[-i]}$ is the vector $\boldsymbol{Z}$ without Z_i. Proceeding analogously to the derivation of Liu [24], we compute the integrals using a Dirichlet prior on $\boldsymbol{\phi}$. We further simplify the sampling expression by dividing it by $P(Z_i = 0, \boldsymbol{X}_i \mid \boldsymbol{\phi}_0)$ which is a constant at a particular sampling step. This results in the following sampling distribution for a particular location j within sequence $\boldsymbol{X}_i$, similar to the predictive update formula as described in [25]:

$$P(Z_i = j \mid \boldsymbol{Z}_{[-i]}, \boldsymbol{X}, \boldsymbol{\phi}_0) = \frac{P(Z_i = j) \times \left(\prod_{a=1}^{W} \phi_{a, X_{i,j+a-1}} \right)}{P(Z_i = 0) \times P(X_{i,j}, \ldots, X_{i,j+W-1} \mid \boldsymbol{\phi}_0)} \tag{2}$$

for $1 \leq j \leq m_i - W + 1$, and

$$P(Z_i = j \mid \boldsymbol{X}, \boldsymbol{\phi}_0) = 1 \tag{3}$$

for $j = 0$, where $\boldsymbol{\phi}$ is calculated from the counts of the sites contributing to the current alignment $\boldsymbol{Z}_{[-i]}$, plus the pseudocounts as determined by the Dirichlet prior. More details are provided in [26].

The joint posterior distribution after each iteration can be calculated as:

$$P(\boldsymbol{\phi}, \boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\phi}_0) \propto P(\boldsymbol{X} \mid \boldsymbol{\phi}, \boldsymbol{Z}, \boldsymbol{\phi}_0) \times P(\boldsymbol{\phi}) \times P(\boldsymbol{Z}) \tag{4}$$

To simplify computation, we divide the above expression by the constant probability $P(\boldsymbol{X} \mid \boldsymbol{Z} = \boldsymbol{0}, \boldsymbol{\phi}_0)$ and use the logarithm of the resulting value as a score for the motif.

To maximize the objective function and hence the score, we run the Gibbs sampler for a predetermined number of iterations after apparent convergence to the joint posterior and output the highest scoring PSSM at the end. We report only a single motif $\boldsymbol{\phi}$ to enable us to evaluate the algorithm and compare it with other popular methods. In principle, however, since we are using an MCMC sampling method, we could instead perform Bayesian model averaging over many samples from the posterior and report a mean motif (or multiple motifs if there are multiple modes in the distribution).

3 Informative Positional Priors for Motif Discovery

The basic Gibbs sampling approach mentioned above has been used in several motif finders, often with additional parameters and heuristics [27, 28, 29]. However, all these methods use an uninformative prior over the locations $\boldsymbol{Z}$ at which the TF is supposed to bind within the DNA sequences. In a recent paper, we showed how information about the TF's structural class could be leveraged to produce informative priors over $\boldsymbol{Z}$ that significantly help motif discovery [26]. Here, we describe other informative choices for

$P(\boldsymbol{Z})$ which we will henceforth refer to as 'positional priors'. We introduce a prior $\mathcal{N}$ based solely on nucleosome occupancy, a prior $\mathcal{DN}$ incorporating nucleosome occupancy information from both bound as well as unbound sequences, and a discriminative prior $\mathcal{D}$, which is a special case of $\mathcal{DN}$ when nucleosome occupancy information is unavailable. To assess the utility of these priors, we compare their performance to the performance of an uninformative uniform prior $\mathcal{U}$, keeping all other aspects of the algorithm identical.

3.1 Building a Positional Prior

The four positional priors mentioned above can be constructed in a similar fashion from different probabilistic scores. We use the term 'probabilistic score' in the remainder of the paper to denote the probability of a particular W-mer being a binding site of transcription factor T: $S_{i,j} = P(X_{i,j}^W$ is a binding site of $T)$, where $X_{i,j}^W$ denotes the W-mer $X_{i,j}X_{i,j+1}\cdots X_{i,j+W-1}$.

For each sequence $\boldsymbol{X}_i$, we wish to define a prior probability distribution over all possible starting locations j of a binding site in that sequence, i.e. $P(Z_i = j)$. We notice that the values $S_{i,j}$ themselves do not define a probability distribution over j, because they may not sum to 1. As mentioned in Section 2.1, we model each sequence $\boldsymbol{X}_i$ as containing at most one binding site of T. If $\boldsymbol{X}_i$ has no binding site, none of the positions of $\boldsymbol{X}_i$ can be the starting location of a binding site of T so it must be that:

$$P(Z_i = 0) \propto \prod_{u=1}^{m_i-W+1} (1 - S_{i,u}) \tag{5}$$

On the other hand, if $\boldsymbol{X}_i$ has one binding site at position j, not only must a binding site start at location j but also no binding site should start at any of the other locations in $\boldsymbol{X}_i$. Formally, we write:

$$P(Z_i = j) \propto S_{i,j} \prod_{\substack{u=1 \\ u \neq j}}^{m_i-W+1} (1 - S_{i,u}) \qquad \text{for} \quad 1 \leq j \leq m_i - W + 1 \tag{6}$$

We then normalize $P(Z_i)$ assuming the same proportionality constant in (5) and (6), so that under the assumptions of our model we have:

$$\sum_{j=0}^{m_i-W+1} P(Z_i = j) = 1 \qquad \text{for} \quad 1 \leq i \leq n \tag{7}$$

3.2 Uniform Prior ($\mathcal{U}$)

This is the simplest form of positional prior. It is built using a uniform probabilistic score $U_{i,j}$ which assigns equal probabilities to a W-mer $X_{i,j}^W$ being a binding site of T or not:

$$U_{i,j} = 1 - U_{i,j} = 0.5 \qquad \text{for} \quad 1 \leq j \leq m_i - W + 1 \tag{8}$$

If we substitute $S_{i,j}$ with $U_{i,j}$ in equations (5) and (6) and normalize $P(Z_i = j)$, we get a uniform prior $\mathcal{U}$:

$$P(Z_i = j) = \frac{1}{m_i - W + 2} \qquad \text{for} \quad 0 \leq j \leq m_i - W + 1 \tag{9}$$

3.3 Nucleosome Occupancy Prior ($\mathcal{N}$)

A uniform prior is a common choice for a positional prior and most motif finding algorithms implicitly use such a prior. In reality though, as mentioned earlier, certain DNA regions are inaccessible to TFs due to the presence of nucleosomes at those locations.

We would like to bias the search in a probabilistic manner towards nucleosome-free areas. For this purpose, we use the nucleosome occupancy predicted by the computational model developed by Segal *et al.* [2]. This model outputs the probability $N_{i,j}$ of each nucleotide $X_{i,j}$ in the input sequences being occupied by nucleosomes (for now the model is only designed for sequences in yeast or chicken). Assuming nucleosome occupancy indicates inaccessibility, we calculate the average probability of the W-mer $X_{i,j}^W$ being accessible to the TF as:

$$A_{i,j} = 1 - \frac{1}{W} \sum_{k=0}^{W-1} N_{i,j+k} \tag{10}$$

Alternatively one could use the maximum instead of the average occupancy over the W nucleotides when computing $A_{i,j}$, but averaging reduces the effect of outliers. Having defined $A_{i,j}$, we can now build the positional prior $\mathcal{N}$ as described in Section 3.1, using $A_{i,j}$ as the probabilistic score $S_{i,j}$.

3.4 Discriminative Nucleosome Occupancy Prior ($\mathcal{DN}$)

The formulation of the above probabilistic score $A_{i,j}$ has a drawback: What if a particular W-mer is prone to be highly accessible throughout the genome? For instance, certain promoter elements which are required for the assembly of general TFs and are not related to the specific TF in question, might be depleted of nucleosomes. The prior $\mathcal{N}$, in that case, could indicate a high prior belief in that W-mer being a TF binding site regardless of the fact that the W-mer is equally accessible in the rest of the genome as in the bound set $\boldsymbol{X}$.

Most large-scale high-throughput experimental methods like ChIP-chip, DIP-chip, and PBM give rise to two sets of DNA sequences: those bound by the profiled transcription factor T (positive sequences $\boldsymbol{X}$) and those not bound (negative sequences which we denote as $\boldsymbol{Y}$). The use of the negative set along with the positive set to enrich the motif signal has been shown previously to be beneficial in improving specificity [12, 14, 15]. In the referenced methods, if a W-mer is present in the negative set for transcription factor T, it is generally treated as an instance of a non-binding site and hence, penalized. However, in an *in vivo* situation, a W-mer matching the true motif of T might occur in the negative set but be inaccessible for T due to the presence of a nucleosome at that position. In that case, it should not be treated as a negative data point.

Here, we devise a new discriminative prior which takes into account both these issues. For each W-mer $X_{i,j}^W$, we ask the following question: “Of all the accessible occurrences of this word, how many occur in the positive set?” To answer this question, we subject each accessible W-mer to a Bernoulli trial. Unfortunately, we cannot tell for sure whether a particular location is accessible or not, because we only know the probability that each location is accessible. Thus, we count the number of accessible sequences in expectation, by weighing each occurrence of the W-mer according to how accessible it is. For this purpose, we introduce two functions $r_{k,l}$ and $r'_{k,l}$ defined on the set of all possible W-mers σ:

$$r_{k,l}(\sigma) = \begin{cases} 1 & : \quad X^W_{k,l} = \sigma \\ 0 & : \quad X^W_{k,l} \neq \sigma \end{cases} \quad \text{and} \quad r'_{k,l}(\sigma) = \begin{cases} 1 & : \quad Y^W_{k,l} = \sigma \\ 0 & : \quad Y^W_{k,l} \neq \sigma \end{cases} \tag{11}$$

We now define a new probabilistic score $C_{i,j}$ as:

$$C_{i,j} = \frac{\sum\limits_{k,l} A_{k,l} r_{k,l}(X^W_{i,j})}{\sum\limits_{k,l} A_{k,l} r_{k,l}(X^W_{i,j}) + \sum\limits_{k,l} A'_{k,l} r'_{k,l}(X^W_{i,j})} \tag{12}$$

where $A'_{i,j}$ is the accessibility score calculated for the set $\boldsymbol{Y}$ analogous to the calculation of $A_{i,j}$ for $\boldsymbol{X}$ in (10). Using $C_{i,j}$ as our probabilistic score $S_{i,j}$, we can now build the positional prior $\mathcal{DN}$ as described in Section 3.1. In practice, we notice that $C_{i,j}$ can have some false peaks due to W-mers that occur very rarely in the genome. In such cases, when the W-mer occurs in $\boldsymbol{X}_i$ at some position j, $C_{i,j}$ becomes large due to a small denominator. This effect can be alleviated by adding pseudocounts to the expression in (12).

3.5 Simple Discriminative Prior ($\mathcal{D}$)

To assess the importance of incorporating nucleosome occupancy information in discriminative motif discovery, we now consider a special case of $\mathcal{DN}$. We assume we have no nucleosome occupancy information, i.e., each $A_{i,j} = c$ and $A'_{i,j} = c$, where c is some arbitrary constant. Equation (12) then reduces to a new probabilistic score $D_{i,j}$:

$$D_{i,j} = \frac{\sum\limits_{k,l} r_{k,l}(X^W_{i,j})}{\sum\limits_{k,l} r_{k,l}(X^W_{i,j}) + \sum\limits_{k,l} r'_{k,l}(X^W_{i,j})} \tag{13}$$

In other words, we calculate the probability $D_{i,j}$ of $X^W_{i,j}$ being a binding site of T as the number of occurrences of $X^W_{i,j}$ in $\boldsymbol{X}$ relative to the total number of occurrences of $X^W_{i,j}$ in both sets $\boldsymbol{X}$ and $\boldsymbol{Y}$ without looking at accessibility. Again, we add pseudocounts while computing $D_{i,j}$ and then calculate a positional prior $P(Z_i = j)$ as described in Section 3.1 by substituting $D_{i,j}$ for $S_{i,j}$. We refer to this positional prior as $\mathcal{D}$.

Note that in computing $\mathcal{D}$ we use only the datasets $\boldsymbol{X}$ and $\boldsymbol{Y}$ and not any nucleosome occupancy information. Other motif discovery algorithms that make use of both $\boldsymbol{X}$ and $\boldsymbol{Y}$ formulate the problem in a discriminative manner, and attempt to learn a motif that appears more often in the positive set than in the negative set. Since these models optimize a discriminative objective function over the sets $\boldsymbol{X}$ and $\boldsymbol{Y}$, they have to deal with a large search space and typically are prone to many local optima. Such methods often require an ‘intelligent guess’ as a seed matrix to initialize the search so as to avoid poor local optima. In addition, at every step of the search algorithm, they have to evaluate the parameters of the model on each sequence in *both* sets. Hence, the time complexity of these algorithms is much worse compared to generative models which iterate only over the positive set. Here, however, our generative model framework remains generative and all the discriminative information is captured in our prior.

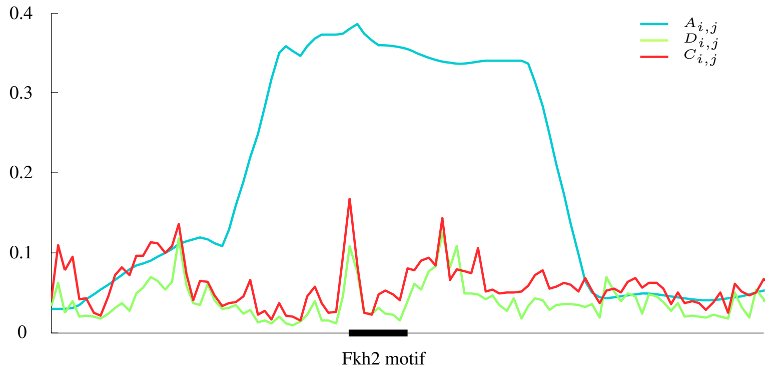


Fig. 1. Plot of $A_{i,j}$, $D_{i,j}$, and $C_{i,j}$ used to compute the priors $\mathcal{N}$, $\mathcal{D}$, and $\mathcal{DN}$, respectively. The x-axis represents part of an intergenic DNA region from a sequence-set for Fkh2 profiled under the YPD condition in a ChIP-chip experiment [3]. The intergenic region spans positions 770845 to 770945 in Chromosome XVI. The Fkh2 binding site shown in the figure starts at position 770887.

3.6 Informative Priors in Action

To visualize how informative priors might be helpful in identifying TF binding sites, we show in Figure 1 the values of $A_{i,j}$, $D_{i,j}$, and $C_{i,j}$ used to compute the priors $\mathcal{N}$, $\mathcal{D}$, and $\mathcal{DN}$ over a portion of a DNA sequence obtained from an Fkh2 ChIP-chip experiment. As can be seen from the figure, in this instance all three priors give a good indication of where a Fkh2 binding site is likely to exist, even before information from the likelihood is taken into account. Of course, this may not happen all the time so we use the remainder of the paper to assess more precisely the relative utility of these priors.

4 Results

We compiled ChIP-chip data published by Harbison *et al.* [3], who profiled the intergenic binding locations of 203 yeast TFs under various environmental conditions: YPD, and one or more of Alpha, But14, But90, H202Hi, H202Lo, Pi-, RAPA, or SM over 6140 intergenic regions. These intergenic regions range from 48 to 1553 nucleotides and have an average length of 433 nucleotides. For each TF profiled under each condition, we define its bound sequence-set to be those intergenic sequences reported to be bound with p-value < 0.001. We restrict our attention to sequence-sets of size at least 10, which yields 242 sequence-sets, encompassing 148 TFs. Of these sequence-sets, 156 correspond to the 80 TFs with a consensus binding motif in the literature (as summarized by Harbison *et al.* at the time their paper was published, or as earlier reported by Dorrington and Cooper [30] or Jia *et al.* [31]), and these 156 are used throughout the remainder of the paper to compare the performance of various motif finding algorithms.

We incorporate the $\mathcal{U}$, $\mathcal{N}$, $\mathcal{D}$, and $\mathcal{DN}$ priors into our Gibbs sampling framework—implemented in PRIORITY [26]—and refer to the resulting algorithms as PRI-$\mathcal{U}$, PRI-$\mathcal{N}$, PRI-$\mathcal{D}$, and PRI-$\mathcal{DN}$, respectively. For evaluation purposes, we fix the motif-width W to 8 in all our runs, although in practice one could certainly explore more values of W. As a background model, we use a third order Markov model trained on all intergenic regions in yeast. We run each algorithm 10 times from different random starting points for each sequence-set for 10,000 sampling iterations and report the top-scoring motif among the 10 runs. We consider an algorithm to be successful for a sequence-set only if the top-scoring motif matches the literature consensus for the corresponding TF. We use a variation of the inter-motif distance measure described by Harbison *et al.* and consider a motif learned by an algorithm to be correct if it is at a distance less than 0.25 from the literature consensus.[1] Different distance cut-offs give different results, but we notice the general trend across all programs remains the same.

Because we are primarily interested in quantifying the extent to which these new informative priors improve *de novo* motif discovery, the results presented in the main portion of the manuscript are limited to a comparison of PRI-$\mathcal{N}$, PRI-$\mathcal{D}$, and PRI-$\mathcal{DN}$ versus PRI-$\mathcal{U}$. However, to ensure that PRI-$\mathcal{U}$ is not simply a 'straw man', but represents a reasonable point of comparison, we have also compiled results from three other popular motif discovery programs as reported by Harbison *et al.*: AlignACE [27], MEME [22], and MDscan [32] (see Supplementary Material). Using the same criterion for success (the top-scoring motif should match the literature consensus), AlignACE is successful in 16 of the 156 sequence-sets, MEME in 35, MDscan in 54, and PRI-$\mathcal{U}$ in 46. AlignACE has one disadvantage over the others in that it uses a first-order Markov model of the background, but each of the three existing methods has advantages over PRI-$\mathcal{U}$: AlignACE considers many motif widths; MEME considers many motif widths, uses sophisticated heuristics to initialize its search, and uses a fifth-order Markov model of the background; and MDscan makes significant use of the p-values from the ChIP-chip experiments. Despite these disadvantages, PRI-$\mathcal{U}$ performs admirably, even without an informative prior, and therefore represents a reasonable point of comparison. Since everything about the algorithm is the same apart from the choice of prior, PRI-$\mathcal{U}$ permits the most accurate quantification of the utility of our new informative priors, and so we use it in the remainder of the paper as a baseline when comparing the performance of PRI-$\mathcal{N}$, PRI-$\mathcal{D}$, and PRI-$\mathcal{DN}$.

Figure 2 summarizes the results of the four algorithms on 156 sequence-sets. Overall, while PRI-$\mathcal{U}$ finds the correct motif in 46 sequence-sets, PRI-$\mathcal{DN}$ finds the correct motif in 69 sequence-sets, resulting in an improvement of 50% over baseline. To break down these results more carefully, we divide the sequence-sets into four groups based on the success/failure of PRI-$\mathcal{U}$ and PRI-$\mathcal{DN}$ (corresponding to the four quadrants in Figure 2). This grouping reveals that the $\mathcal{DN}$ prior never performs worse than the $\mathcal{U}$ prior, a claim that is also true of the $\mathcal{D}$ prior, but not of the $\mathcal{N}$ prior. To better understand the performance of these two priors in relation to the $\mathcal{DN}$ prior, we now consider each group in detail:

[1] The distance is normalized to lie between 0 and 1; see Supplementary Material for details about the distance calculation.

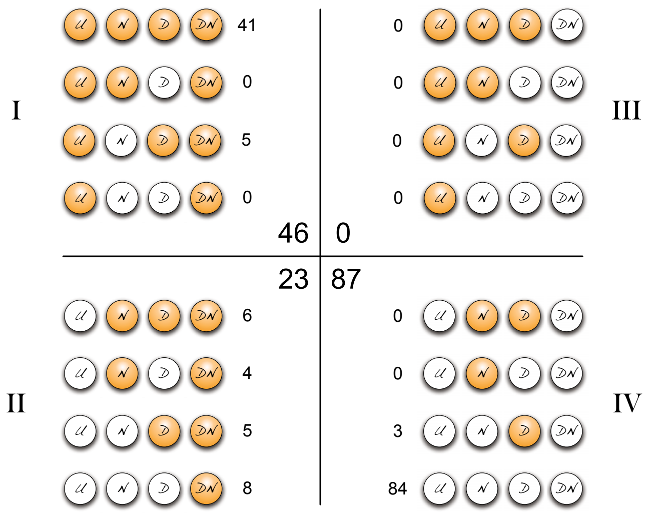


Fig. 2. Results of the four algorithms on 156 yeast sequence-sets produced by ChIP-chip experiments [3]. Each row of four balls corresponds to the four positional priors $\mathcal{U}$, $\mathcal{N}$, $\mathcal{D}$, and $\mathcal{DN}$. A filled ball indicates the situation where the respective prior succeeds in finding the true motif. There are $2^4 = 16$ possible combinations of successes/failures of the four priors shown by 16 rows of filled/empty balls. The number of cases resulting in each combination is indicated next to the respective row. The 16 combinations are divided into four quadrants, conditioned on the success/failure of $\mathcal{U}$ and $\mathcal{DN}$. The central numbers indicate the cardinality of each quadrant. As can be seen, some combinations like those in quadrant III do not occur.

Group I: PRI-$\mathcal{U}$ succeeds and PRI-$\mathcal{DN}$ succeeds.

This group corresponds to the upper-left quadrant of Figure 2 and it contains 46 sequence-sets corresponding to 31 TFs. For most sequence-sets in this group (41 of 46) all four algorithms find motifs matching the literature consensus. For the other 5 sequence-sets (Cin5_H202Lo, Ste12_Alpha, Ste12_YPD, Hsf1_H202Lo, and Skn7_YPD) PRI-$\mathcal{N}$ is the only algorithm that fails.

Let us look at the case of the TF Ste12 in more detail. In theory, the way the priors are formulated, PRI-$\mathcal{N}$ should work on TFs for which the nucleosome occupancy over the functional binding sites is lower, in general, than the nucleosome occupancy over the rest of the sequences in the set. For PRI-$\mathcal{DN}$ to succeed though, the nucleosome occupancy over the functional sites must be lower than the occupancy over the non-functional sites (that is, sites in the negative set). In both the Alpha and YPD conditions, the average nucleosome occupancy in the sequence-sets is lower than the nucleosome occupancy at the functional binding sites of Ste12. This explains why PRI-$\mathcal{N}$ fails. But according to the analysis of Segal *et al.* [2, Supplemental Figure 36], the average nucleosome occupancy at the functional sites of Ste12 is lower than the average occupancy at the non-functional sites. This clarifies why in spite of using the same nucleosome occupancy data, PRI-$\mathcal{DN}$ succeeds in finding the true motif of Ste12 in both conditions, although PRI-$\mathcal{N}$ does not. This suggests the importance of using nucleosome occupancy information in a discriminative setting.

Group II: PRI-$\mathcal{U}$ fails and PRI-$\mathcal{DN}$ succeeds.

This group corresponds to the lower-left quadrant of Figure 2 and it contains 23 sequence-sets corresponding to 19 TFs. In eight cases, PRI-$\mathcal{DN}$ is the only algorithm that succeeds in finding the true motif. This implies that neither $\mathcal{D}$ nor $\mathcal{N}$ alone is strong enough to identify the true motif, but the combination $\mathcal{DN}$ succeeds. In 9 other cases in this group, in addition to $\mathcal{DN}$, exactly one of $\mathcal{D}$ and $\mathcal{N}$ is successful. This suggests that in those cases, the improvement in $\mathcal{DN}$ comes mainly from the respective prior.

Group III: PRI-$\mathcal{U}$ succeeds and PRI-$\mathcal{DN}$ fails.

This group, corresponding to the upper-right quadrant of Figure 2, is empty. This implies that whenever the uniform prior succeeds, the $\mathcal{DN}$ prior also succeeds. Thus using this informative prior does not worsen the performance of the algorithm for any sequence-set.

Group IV: PRI-$\mathcal{U}$ fails and PRI-$\mathcal{DN}$ fails.

This group corresponds to the lower-right quadrant of Figure 2 and contains 87 sequence-sets corresponding to 50 TFs. For 84 of these 87 sequence-sets, none of the four algorithms finds motifs matching the literature consensus. For the remaining three cases (Msn2_H202Hi, Skn7_H202Lo, and Tec1_YPD) although PRI-$\mathcal{D}$ succeeds, PRI-$\mathcal{DN}$ seems to fail to find the true motif. However, the failure of PRI-$\mathcal{DN}$ seems to be the result of the program getting stuck in a local optimum in each case. When we score the three motifs found by PRI-$\mathcal{D}$ according to the posterior score obtained using the $\mathcal{DN}$ prior, we get a significantly higher score than the score reported by PRI-$\mathcal{DN}$ for the respective top motifs it learns (which do not score well according to the distance metric). The same reasoning applies for the failure of PRI-$\mathcal{N}$ for the sequence-sets of Msn2_H202Hi and Skn7_H202Lo. In the Tec1_YPD sequence-set, however, Tec1 binding sites have an average nucleosome occupancy of $\sim$89% which is higher than the average occupancy over all intergenic regions ($\sim$85%) causing the $\mathcal{N}$ prior to fail.

5 Discussion

Although it has been known for a while that nucleosomes control the binding activity of TFs by providing differential access to DNA binding sites [1, 2, 17, 18, 19, 20], we believe we are the first to use nucleosome occupancy information to more accurately predict *de novo* binding sites of TFs.

Our results show that direct use of the nucleosome occupancy predictions of Segal *et al.* [2] as a positional prior does not help motif discovery much: PRI-$\mathcal{N}$ finds 51 correct motifs compared to the 46 found by PRI-$\mathcal{U}$. Motifs of some TFs are more prone to be occupied by nucleosomes than others. The example of Ste12 in Group I illustrates how the prior $\mathcal{N}$ can fail because of the high nucleosome occupancy at Ste12 functional sites. However, when we adopt a discriminative perspective on nucleosome occupancy, the prior $\mathcal{DN}$ succeeds in finding the true Ste12 motif. In fact, there is no sequence-set on which PRI-$\mathcal{N}$ succeeds, but PRI-$\mathcal{DN}$ fails. Overall, our results show that discriminative use of nucleosome occupancy information is extremely useful: PRI-$\mathcal{DN}$ finds 69 true motifs, 50% more than PRI-$\mathcal{U}$. Although in this paper we focus on the usefulness of nucleosome occupancy information, the $\mathcal{D}$ prior also improves motif discovery noticeably without this information: PRI-$\mathcal{D}$ finds 60 true motifs, 30% more than PRI-$\mathcal{U}$. In addition to the three programs AlignACE, MEME, and MDscan discussed earlier, Harbison *et al.* use three conservation-based algorithms to discover motifs: MEME_c, CONVERGE [3], and a method by Kellis *et al.* [33] which find 49, 56, and 50 correct motifs respectively (see Supplementary Material). Not only does PRI-$\mathcal{DN}$ perform much better than these programs, even PRI-$\mathcal{D}$ finds more correct motifs than the best of these programs. This suggests that our prior $\mathcal{D}$ will be quite useful in motif discovery problems even when nucleosome occupancy information is unavailable.

Our discriminative priors (both $\mathcal{D}$ and $\mathcal{DN}$) are novel in the way they incorporate discriminative information in a generative setting. Note that in a specific genome, for a particular W-mer σ starting at position j in $\boldsymbol{X}_i$, the denominator of (13) remains the same regardless of the sequences in $\boldsymbol{X}$, since it is nothing but the total number of occurrences of σ in the whole genome. Similarly, for a particular nucleosome occupancy dataset (experimental or computational), the weighted sum of all accessible sites in the denominator of (12) remains the same for all possible sequence sets $\boldsymbol{X}$. Hence these numbers can be precomputed and stored in a table of size 4^W. Then, for a particular sequence-set $\boldsymbol{X}$, computing the prior involves one pass (linear-time) over just the sequences in $\boldsymbol{X}$. No information needs to be explicitly computed from the negative set $\boldsymbol{Y}$, which is good because it changes as the positive set changes. In addition, since the actual algorithm only needs to sample over the positive set, the overall time and space complexities of the search are much less than the complexities of other discriminative approaches. In fact, it is practically impossible to compare the performance of PRI-$\mathcal{D}$ with these approaches since the size of the intergenic regions in yeast is about 3 megabases (and larger for metazoan genomes).

In this study, we have fixed W to be 8. In the case of longer motifs, we could post-process the short motif learned by the algorithm and expand it appropriately on either side. Alternatively, we could build priors for multiple values of W and, like most motif finders, run the algorithm with different motif lengths. A larger value for W has certain consequences, however. First, the space required to store priors over W-mers is exponential in W. Second, as W grows, the average probability of seeing a W-mer in the genome decreases, implying that pseudocounts used to smooth the prior become increasingly important (of course, this effect will be mitigated somewhat in larger genomes).

Throughout the paper, we have used PSSMs to model motifs. Although the PSSM is currently a popular choice for a motif model, recent biological [34] and computational [35, 36] findings indicate that more expressive (and hence, more complex) models might be more appropriate. Since our method assigns a prior on the locations within each sequence and not on any specific form of the motif model, it can be used to learn any motif model.

The nucleosome occupancy predictions from the model of Segal *et al.* attempt to capture the static, intrinsic nucleosome binding properties of the DNA. In reality, however, the positioning of nucleosomes changes dynamically as the environmental conditions change or even as the cell progresses through its cell-cycle. Nucleosomes covering certain functional sites might be displaced under specific conditions by other mechanisms to permit access to TFs. It is thus not surprising that Segal *et al.* note that according to their computational model, certain TFs have higher nucleosome occupancy at their functional sites than non-functional sites. If nucleosome occupancy data collected under the same environmental conditions in which the TFs are profiled were available, we would expect to get better results. Unfortunately, at this time high-resolution nucleosome occupancy data is limited. But as more data becomes available, we can incorporate it usefully into our approach.

In closing, we stress that incorporating informative priors over sequence positions is of great benefit to motif discovery algorithms. Low signal-to-noise ratio, especially in higher organisms, makes it difficult to successfully use algorithms based only on statistical overrepresentation. Narlikar *et al.* [26] have shown that using informative priors based on structural classes of TFs improves motif discovery and this paper shows that other kinds of informative priors improve motif discovery as well. Algorithms using conservation information across species [3, 33, 37, 38] are another example of successful incorporation of additional information for motif discovery. We note that although PRI-$\mathcal{DN}$ does better overall than the conservation based methods described earlier, there are certain motifs that one or more of these methods find but PRI-$\mathcal{DN}$ does not. This suggests that combining conservation and nucleosome occupancy might further improve the performance of motif finders. We are currently working toward a unified framework of informative priors based on nucleosome occupancy, TF structural class, and conservation.

Acknowledgments. The authors thank Eran Segal for sending them the nucleosome occupancy model, Jason Lieb for sharing unpublished data, and Uwe Ohler for useful discussions and suggestions. The research presented here was supported by a National Science Foundation CAREER award and an Alfred P. Sloan Fellowship to A.J.H.

Supplementary Material can be found at http://www.cs.duke.edu/∼amink/.

References

[1] Lee,C., Shibata,Y., Rao,B., Strahl,B., Lieb,J. (2004) Evidence for nucleosome depletion at active regulatory regions genome-wide, *Nature Genetics*, 36(8): 900–905.

[2] Segal,E., Fondufe-Mittendorf,Y., Chen,L., Thastrom,A., Field,Y., Moore,I., Wang,J., and Widom,J. (2006) A genomic code for nucleosome positioning, *Nature*, 442(7104):772–778.

[3] Harbison,C., *et al.* (2004) Transcriptional regulatory code of a eukaryotic genome, *Nature*, 431:99–104.

[4] Lee,T., *et al.* (2002) Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, 298:799–804.

[5] Liu,X., Noll,D., Lieb,J., and Clarke,N. (2005) DIP-chip: Rapid and accurate determination of DNA binding specificity, *Genome Research*, 15(3):421–427.

[6] Mukherjee S., Berger M., Jona G., Wang X., Muzzey D., Snyder M., Young R., and Bulyk M. (2004) Rapid analysis of the DNA binding specificities of transcription factors with DNA microarrays, *Nature Genetics*, 36(12):1331–1339.

[7] Spellman,P., Sherlock,G., Zhang,M., Iyer,V., Anders,K., Eisen,M., Brown,P., Botstein,D., and Futcher,B. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization, *Molecular Biology of the Cell*, 9:3273–3297.

[8] Kim,S., Lund,J., Kiraly,M., Duke,K., Jiang,M., Stuart,J., Eizinger,A., Wylie,B., and Davidson,G. (2001) A gene expression map for *Caenorhabditis elegans*, *Science*, 293:2087–2092.

[9] Wasserman,W. and Sandelin,A. (2004) Applied bioinformatics for the identification of regulatory elements, *Nat Rev Genet*, 5(4):276–287.

[10] Siggia,E. (2005) Computational methods for transcriptional regulation, *Current Opinion in Genetics and Development*, 15:214–221.

[11] Workman,C. and Stormo,G. (2000) ANN-Spec: A method for discovering transcription factor binding sites with improved specificity, *Pac. Symp. Biocomput.*, 467–478.

[12] Segal,E., Barash,Y., Simon,I., Friedman,N., and Koller,D. (2002) From sequence to expression: A probabilistic framework, *RECOMB '02*.

[13] Sinha,S. (2002) Discriminative motifs, *RECOMB '02*.

[14] Hong,P., Liu,X., Zhou,Q., Lu,X., Liu,J., and Wong,W. (2005) A boosting approach for motif modeling using ChIP-chip data, *Bioinformatics*, 21(11):2636–2643.

[15] Sinha,S. (2006) On counting position weight matrix matches in a sequence, with application to discriminative motif finding, *Bioinformatics*, 22(14):e454–463.

[16] Tompa,M. *et al.* (2005) Assessing computational tools for the discovery of transcription factor binding sites, *Nat. Biotechnol.*, 23(1):137–144.

[17] Almer,A., Rudolph,H., Hinnen,A., and Horz,W. (1986) Removal of positioned nucleosomes from the yeast PHO5 promoter upon PHO5 induction releases additional upstream activating DNA elements, *Embo. J.*, 5:2689–2696.

[18] Mai,X., Chou,S., and Struhl,K. (2000) Preferential accessibility of the yeast *his3* promoter is determined by a general property of the DNA sequence, not by specific elements, *Cell Biol.*, 20:6668:6676.

[19] Sekinger,E., Moqtaderi,Z., and Struhl,K. (2005) Intrinsic histone-DNA interactions and low nucleosome density are important for preferential accessibility of promoter regions in yeast, *Mol. Cell*, 18:735–748.

[20] Yuan,G., Liu,Y., Dion,M., Slack,M., Wu,L., Altschuler,S., and Rando,O. (2005) Genome-scale identification of nucleosome positions in *S. cerevisiae*, *Science*, 309:626–630.

[21] Staden,R. (1984) Computer methods to locate signals in nucleic acid sequences, *Nucleic Acids Research*, 12:505–519.

[22] Bailey,T. and Elkan,C. (1994) Fitting a mixture model by expectation maximization to discover motifs in biopolymers, *ISMB '94*, AAAI Press, Menlo Park, California, pp. 28–36.

[23] Gelfand,A. and Smith,A. (1990) Sampling based approaches to calculating marginal densities, *Journal of the American Statistical Association*, 85:398–409.

[24] Liu,J. (1994) The collapsed Gibbs sampler with applications to a gene regulation problem, *Journal of the American Statistical Association*, 89:958–966.

[25] Liu,J., Neuwald,A., and Lawrence,C. (1995) Bayesian models for multiple local sequence alignment and Gibbs sampling strategies, *Journal of the American Statistical Association*, 90:1156–1170.

[26] Narlikar,L., Gordân,R., Ohler,U., and Hartemink,A. (2006) Informative priors based on transcription factor structural class improve *de novo* motif discovery, *Bioinformatics*, 22(14):e384–e392.

[27] Roth,F., Hughes,J., Estep,P., and Church,G. (1998) Finding DNA regulatory motifs within unaligned non-coding sequences clustered by whole-genome mRNA quantitation, *Nature Biotech.*, 16:939–945.

[28] Liu,X., Brutlag,D., and Liu,J. (2001) BioProspector: Discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes, *Pac Symp Biocomput.*, 127–138.

[29] Thijs,G., Marchal,K., Lescot,M., Rombauts,S., De Moor,B., Rouze,P., and Moreau,Y. (2002) A Gibbs sampling method to detect over-represented motifs in the upstream regions of coexpressed genes, *Journal of Computational Biology*, 9:447–464.

[30] Dorrington,R.A. and Cooper,T.G. (1993) The DAL82 protein of *Saccharomyces cerevisiae* binds to the DAL upstream induction sequence (UIS), *Nucleic Acids Research*, 21(16):3777-3784.

[31] Jia,Y., Rothermel,B., Thornton,J. and Butow,R.A. (1993) A basic helix-loop-helix-leucine zipper transcription complex in yeast functions in a signaling pathway from mitochondria to the nucleus, *Molecular and Cellular Biology*, 17: 1110–1117.

[32] Liu,X., Brutlag,D., and Liu,J. (2002) An algorithm for finding protein-DNA binding sites with applications to chromatin immunoprecipitation microarray experiments, *Nature Biotech.*, 20:835–839.

[33] Kellis,M., Patterson,N., Endrizzi,M., Birren,B., and Lander,E. (2003) Sequencing and comparison of yeast species to identify genes and regulatory elements, *Nature*, 432:241–254.

[34] Bulyk,M., Johnson,P., and Church,G. (2002) Nucleotides of transcription factor binding sites exert interdependent effects on the binding affinities of transcription factors, *Nucleic Acids Research*, 30:1255–1261.

[35] Agarwal,P. and Bafna,V. (1998) Detecting non-adjacent correlations within signals in DNA, *RECOMB '98*

[36] Barash,Y., Elidan,G., Friedman,N., and Kaplan,T. (2003) Modeling dependencies in protein-DNA binding sites, *RECOMB '03.*

[37] Miller,W., Makova,K., Nekrutenko,A., and Hardison,R. (2004) Comparative Genomics, *Annu. Rev. Genom. Human. Genet.*, 5:15–56.

[38] Siddharthan,R., Siggia,E., and Nimwegen,E. (2005) PhyloGibbs: A Gibbs Sampling Motif Finder That Incorporates Phylogeny, *PLoS Comput. Biol.*, 1(7):e67.

Framework for Identifying Common Aberrations in DNA Copy Number Data

Amir Ben-Dor[1,⋆], Doron Lipson[2], Anya Tsalenko[1], Mark Reimers[3], Lars O. Baumbusch[4], Michael T. Barrett[1,5], John N. Weinstein[3], Anne-Lise Børresen-Dale[4], and Zohar Yakhini[1,2]

[1] Agilent Laboratories, Santa-Clara, CA
[2] Computer Science Dept., Technion, Haifa
[3] National Cancer Institute, Bethesda, MD
[4] Department of Genetics, Institute for Cancer Research, Rikshospitalet-Radiumhospitalet Medical Center
[5] Translational Genomics Research Institute, Phoenix, AZ
`amir_ben-dor@agilent.com`

Abstract. High-resolution array comparative genomic hybridization (aCGH) provides exon-level mapping of DNA aberrations in cells or tissues. Such aberrations are central to carcinogenesis and, in many cases, central to targeted therapy of the cancers. Some of the aberrations are sporadic, one-of-a-kind changes in particular tumor samples; others occur frequently and reflect common themes in cancer biology that have interpretable, causal ramifications. Hence, the difficult task of identifying and mapping common, overlapping genomic aberrations (including amplifications and deletions) across a sample set is an important one; it can provide insight for the discovery of oncogenes, tumor suppressors, and the mechanisms by which they drive cancer development.

In this paper we present an efficient computational framework for identification and statistical characterization of genomic aberrations that are common to multiple cancer samples in a CGH data set. We present and compare three different algorithmic approaches within the context of that framework. Finally, we apply our methods to two datasets – a collection of 20 breast cancer samples and a panel of 60 diverse human tumor cell lines (the NCI-60). Those analyses identified both known and novel common aberrations containing cancer-related genes. The potential impact of the analytical methods is well demonstrated by new insights into the patterns of deletion of CDKN2A (p16), a tumor suppressor gene crucial for the genesis of many types of cancer.

Keywords: CGH, cancer, microarray data analysis, common aberrations, breast cancer, NCI-60.

1 Introduction

Alterations in DNA copy number are characteristic of many cancer types and drive some cancer pathogenesis processes as well as several developmental disorders.

⋆ Corresponding author.

These alterations include large chromosomal gains and losses as well as smaller scale amplifications and deletions. Genomic instability can often trigger the over-expression or activation of oncogenes and the silencing of tumor suppressors. Mapping regions of common genomic aberrations can therefore provide insight to cancer pathogenesis and lead to discovery of cancer-related genes and the mechanisms by which they drive the disease. Genomic aberrations are also routinely used for diagnosis and clinical practice. For example, Erbb2 amplification is a strong predictor of Herceptin activity in breast cancer patients [1]. Similarly, amplifications of MDM2 and CDK4 genes on chromosome 12q13-15 are useful in distinguishing well-differentiated liposarcomas from benign adipose tumors [2].

Technologies for measuring alterations in DNA copy number include local fluorescence in situ hybridization-based techniques, Comparative Genomic Hybridization (CGH [3,4,5]) and the advanced method termed array CGH (aCGH). In aCGH differentially labeled tumor and normal DNA are co-hybridized to a microarray of thousands to hundreds of thousands of genomic BAC clones, cDNA or oligonucleotide probes [6,7,8,9,10,11,12]. The use of oligonucleotide aCGH allows the determination of changes in DNA copy number of relatively small chromosomal regions. Using high density arrays allows very high DNA copy number resolution, in terms of genomic distances, down to single Kb and less.

A common first step in analyzing DNA copy number alterations (CNAs) data consists of identifying aberrant (amplified or deleted) regions in each individual sample. Aberration calling is the subject of extensive literature [13,14,15,16,17]. We briefly address this step of the process in Section 2.1.

To realize the full power of multi-sample, high-resolution oligo-aCGH studies, we are interested in efficient computational methods that enable the automatic elucidation of more complex structures. The focus of this paper is the discovery of common genomic aberrations, either in a fixed set of samples or in a significant subset of the samples. To date, little attention has been given in the literature to formal treatments of this task. Two important exceptions are the work of Disking et al [18] and Rouveirol et al [19]. In [18] the authors developed a method called Significance Testing for Aberrant Copy number (STAC) to address the detection of DNA copy number aberrations across multiple aCGH experiments. STAC uses two complementary statistical scores in combination with a heuristic search strategy. The significance of both statistics is assessed, and p-values are assigned to each location in the genome by using a permutation approach. In the work of Rouveirol et al [19] the authors propose a formal framework for the task of detecting commonly aberrant regions in CGH data, and present two algorithms (MAR and CMAR) for this task. The framework requires, however, a segmentation algorithm that categorize each data point as being gained/lost/normal. Therefore, this approach requires setting an arbitrary threshold for the discretization step, and is not sensitive to the actual copy number change. In addition, the methods of Lipson et al [20], based on optimizing a statistically motivated score function for genomic intervals can be adapted to automatic identification of aberrations that are common in subsets of the sample set. Despite the lack of formal approaches to identifying common aberrations many studies do report common aberrations and their locations. Typically these aberrations are determined by counting and applying human judgment to single sample calls.

In this paper we present an efficient computational framework for identification and statistical characterization of common genomic aberrations. In Section 2 we start with a description of the overall structure of the framework. The first step, aimed at per-sample aberration calling is described in Section 2.1. The rest of Section 2 is devoted to detailed description of three specific approaches for detecting common aberrations. In Section 2.2 we present the commonly used penetrance method, and its weighted version. We introduce a context-corrected version of penetrance in Section 2.3. We conclude the methods section in Section 2.4 with the CoCoA algorithm, that extends the context-corrected statistical approach to multi-probe intervals.

In Section 3 we apply our methods to two DNA copy number cancer datasets, one derived from a collection of 20 breast cancer samples, and the other a set of 60 cell lines. We compare the results of the three approached using the breast cancer dataset, and highlight several interesting significant aberrations that contain cancer related genes.

2 Framework

In this section we describe a framework for identifying and statistically scoring aberrations that are reoccurring in multiple samples. In a nutshell, the process consists of four steps.

1. **Aberration Calling** – Each of the samples' data vector is analyzed independently, and a set of aberrations (amplifications and deletions) is identified.
2. **Listing candidate intervals -** Given the collection of aberration sets called for all samples, we construct a list of genomic intervals that will be evaluated. We refer to these intervals as *candidate intervals*.
3. **Scoring $\langle$candidate interval, sample$\rangle$** – In this step, we calculate a statistical significance score for each candidate interval with respect to each sample.
4. **Scoring candidate intervals** – For each candidate interval, we combine the per-sample scores derived in the previous step into a comprehensive score for the candidate interval and estimate its statistical significance. In addition, we also identify for each candidate interval the set of samples that supports it.

At the end of the process, we list the top-scoring candidate intervals together with their support sets.

The framework is modular in nature, in the sense that different algorithms and statistical models and methods can be used in each of the different steps. For example, alternative algorithms can be used to call aberrations in the first step. Similarly, alternative approaches may be employed to define candidate intervals and interval scores.

In the rest of this section we will describe several specific embodiments of the framework. We begin (Section 2.1) by discussing single sample aberration calling, which may be viewed as the input of the common aberration analysis procedure. In Sections 2.2-2.4 we describe three different algorithms based on the framework. For simplicity, we will describe only scores related to common amplifications, although it is clear that symmetric scores apply to common deletions.

2.1 Single Sample Aberration Calling

The starting point of the procedure of identifying statistically significant common aberrations is a set of aberrant segments for each sample. In this paper we assume that, independent of the particular aberration-calling algorithm, the set of aberration calls for a particular sample and a particular chromosome can be represented by a *step-function*. The latter consists of discrete segments parallel to the x-axis, that together span the entire chromosome. Formally, for a sample s, denote the length (in Mb) of the chromosome by ℓ. A step-function $\mathcal{F}_s : [0, \ell] \longrightarrow \mathcal{R}$ contains a segment for each aberration call (with the appropriate boundaries and height). In addition, segments of height zero are used to represent non-aberrant regions of the chromosome. See Figure 1 for an example of a step-function.

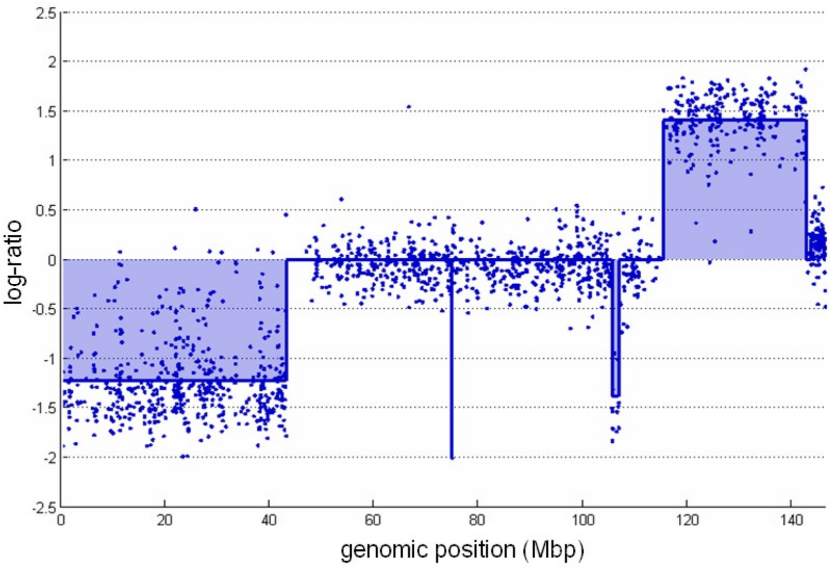


Fig. 1. Step-function derived from chromosome 8 data for colon carcinoma cell line HT29, data from Agilent 44K aCGH array. Solid blue line shows the step-function $\mathcal{F}_s$.

In this study we used the *StepGram* algorithm for single sample aberration calling. StepGram runs in subquadratic time in terms of the number of probes on the chromosome. That translates to < 1 sec for 44K probes, and 3 sec for

185K probes with current the implementation. StepGram is therefore particularly suitable for analysis of large datasets and useful in the context of looking for common aberrations. The details of StepGram were described previously by Lipson et al [20], and an overview is provided here for completeness.

StepGram. Given a vector of real values $V = (v_1, \ldots, v_n)$ (corresponding to normalized log-ratio measurements along a particular chromosome) the optimization problem solved by StepGram involves identifying the interval $I \subseteq [1, n]$ that maximizes the score $|\sum_{i \in I} v_i / \sqrt{I}|$. A branch-and-bound approach allows solving this problem in $O(n^{1.5})$ time complexity in practice. Following identification of the maximal scoring interval the analysis is repeated by recursion to the left, to the right, and within the identified interval until some lower threshold score is attained. A stand-alone implementation of the StepGram algorithm is publicly available at `http://bioinfo.cs.technion.ac.il/stepgram/`.

Other aberration-calling algorithms. Several other algorithms for identifying aberrations in DNA copy number data have been described. These include CBS [15] based on binary segmentation, CLAC [16] based on clustering, aCGH [13] based on HMM, ACE [21] based on FDR, and others. Comparison studies of several of these algorithms were conducted by Lai et al [14] and by Willenbrock et al [17]. Note that many of them are *segmentation* algorithms in the sense that they partition the chromosome into segments of equal copy number but do not attempt to identify which of those segments are aberrant. For the purpose of identifying common aberrations the segmentation output is typically sufficient.

2.2 Weighted and Unweighted Penetrance

We begin by describing the commonly-used penetrance score and its role within the common aberrations analysis framework. Although the penetrance score is not a measure of statistical significance, it does exemplify the different steps of the process.

Candidate intervals. In the case of the penetrance score, the candidate intervals are defined simply as the positions of the probes in the aCGH array. Similar definitions, such as uniformly-spaced pseudo-probes, are also possible. In either case, for a particular chromosome the candidate intervals can be formally defined as a set of non-overlapping intervals $\mathcal{I} = \{[x_i - \epsilon, x_i + \epsilon]\}$. Here ϵ is an arbitrary constant smaller then the minimum distance between any two probes on the array.

Scoring ⟨interval, sample⟩. For a given interval $I = [x_i - \epsilon, x_i + \epsilon]$ and sample s the unweighted amplification penetrance score is defined as a binary score $\alpha(I, s) = 1_{\mathcal{F}_s(x_i) > t}$ for some threshold t. The weighted penetrance scores take into account also the height of the aberration: $\alpha'(I, s) = 1_{\mathcal{F}_s(x_i) > t} \cdot \mathcal{F}_s(x_i)$.

Scoring candidate intervals. The overall penetrance score for a given candidate interval I is defined simply as $\alpha(I) = \sum_s \alpha(I, s)$. As noted before, this score does not reflect any measure of statistical significance.

2.3 Context-Corrected Penetrance

A variant of the penetrance score provides a measure of statistical significance of the common aberration at the specified probe. The significance is defined with respect to the genomic background of each sample, as represented by the pattern of aberrations over each of the samples. In other words, given the specific set of aberration calls for each sample, we wish to describe our “surprise” at seeing a specific set of aberrations co-localized at the same genomic position. Note that the context provided for the score may be either genomic or chromosomal.

Candidate intervals. As was in the case of the penetrance score, the candidate intervals are defined as as a set of non-overlapping intervals at specific genomic positions: $\mathcal{I} = \{[x_i - \epsilon, x_i + \epsilon]\}$.

Scoring ⟨interval, sample⟩. For the context-corrected score, we wish the score of a given interval $I = [x_i - \epsilon, x_i + \epsilon]$ and sample s to reflect the probability of finding an interval of similar (or higher) amplitude given the context of the sample. The score is therefore defined as

$$p(I, s) = \frac{|\{x_j \in \mathcal{I} : \mathcal{F}_s(x_j) \geq \mathcal{F}_s(x_i)\}|}{|\mathcal{I}|}.$$

Scoring candidate intervals. Let S be the set of samples, with $m = |S|$. For a given interval I we now have m scores. Note that the interval I might be aberrant in only a subset of the samples, we therefore seek the subset of samples that will provide maximal significance. Assume, w.l.o.g., that $p(I, 1) \leq p(I, 2) \leq \ldots \leq p(I, m)$. Looking at the first k samples, the probability of concurrently observing k or more scores of probability $p = p(I, k)$ or lower is provided by the Binomial distribution:

$$\rho_k(I) = \text{Binom}(k, m, p) = \sum_{i=k}^{m} \binom{m}{i} p^i (1-p)^{m-i}$$

Since we are interested in identifying aberrations that occur in at least two samples, and to address multiple testing concerns, we define a more conservative score that ignores the first success in the computation,

$$\rho_k^{'}(I) = \text{Binom}(k-1, m-1, p)$$

We define the score of I, to be the minimum of these scores over all values of k, namely,

$$\rho(I) = \min_{k=1,\ldots,m-1} \rho_k^{'}(I).$$

2.4 Context-Corrected Common Aberrations (CoCoA)

Although the context-corrected penetrance algorithm will clearly detect statistically significant common aberrations that are affecting a single probe, its ability to detect larger significant aberrations is not guaranteed. In some cases, a multi-probe common aberration may be significant as a whole, although the score of each single probe contained in the aberration may not show statistical significance. For example, consider the case in which each of many samples contains many random high-amplitude single-probe amplifications and a common large moderate-amplitude amplification. In that case, the size of the aberration may help us to determine its significance, since not many random aberrations of the same size will be detected in the background.

The third, most sophisticated, algorithm for identifying significant common aberrations expands the concept of a context-corrected significance score to intervals that are larger than a single probe.

Candidate intervals. Consider a particular chromosome, c, and denote by $T = \{[b_1, e_1], \dots, [b_k, e_k]\}$ the set of all genomic intervals in c that are called as aberrant in any of the samples. The set of candidate intervals in c is defined to be all genomic intervals that starts at the left side of one interval from T and end at the right side of another. That is, $\mathcal{I} = \{[b_i, e_j] : 1 \leq i, j \leq k, and\ \ b_i \leq e_j\}$. Note that the size of $\mathcal{I}$ is quadratic in k, the number of called aberrations. A smaller list of candidate intervals can be constructed by considering only intervals in T and intersections thereof. that is $\mathcal{I} = T \cup \{t \cap s : t, s \in T\}$. The size of $\mathcal{I}$ is typically $o(k^2)$, and can be constructed in linear time (proof omitted).

Scoring ⟨interval, sample⟩. Applying the same reasoning as for the Context-Corrected Penetrance, we wish the score of a given interval $I = [b, e]$ and sample s to reflect the probability of finding an interval of the same length with a similar (or higher) amplitude given the context of the sample. More specifically, assume we pick a random interval J of the same size as I in the context (that is, in the same chromosome, or in the entire genome). The score is defined as the probability that the average height of J would be as high (or higher) as the height of I,

$$p(I, s) = \mathrm{Pr}_{J:|J|=|I|}\left(h_s(J) \geq h_s(I)\right).$$

where $|I|$ denotes the genomic size I, and $h_s(J)$ denotes the average height of the step-function $\mathcal{F}_s$ over the interval J. We outline now how to computed $p(I, s)$ efficiently (in linear time). Denote by $\mathcal{F}_{s,\ell}(\cdot)$ the ℓ-window moving average of $\mathcal{F}_s$. The score $p(I, s)$, can now be expressed as a function of $\mathcal{F}_{s,\ell}$,

$$p(I, s) = \frac{|x : \mathcal{F}_{s,\ell}(x) \geq h_s(I)|}{c - \ell}.$$

where c denoted the length of the chromosome. Since $\mathcal{F}_s$ is a step-function, its moving average $\mathcal{F}_{s,\ell}$ is a piecewise-linear function. Thus, we can efficiently identify the regions where $\mathcal{F}_{s,\ell}(x) \geq h_s(i)$, and compute $p(I, s)$.

Scoring candidate intervals. After computing context-corrected per-sample scores for I, we combine them into a statistical score for I using the same binomial distribution calculation as detailed in Section 2.3.

3 Results

In this section we demonstrate the application of the above methods to DNA copy number data from two datasets, both measured using Agilent 44K aCGH arrays:

1. A set of 20 primary breast tumor samples were included in this study. These samples are part of a larger patient cohort consisting of 920 breast cancer patients stage I and II referred for surgical treatment and where detection of isolated tumor cells in bone marrow was performed (The Oslo Micrometastases Study) [22]. Tumor material were fresh frozen immediately after surgery and stored at -80C until use. Although the sample set includes several distinct subtypes that had previously been characterized [23,24], due to its relative homogeneity, we expected to encounter common aberrations typical of breast cancer.
2. A diverse set of 60 cancer cell-lines known as the NCI-60 cell line panel [25]. The NCI-60 panel has been used by the Developmental Therapeutics Program (DTP) of the U.S. National Cancer Institute (NCI) to screen $> 100,000$ chemical compounds and natural product extracts for anticancer activity since 1990 [26,27,25]. The NCI-60 panel is comprised of cell lines from diverse human cancers, including leukemias, melanomas, and cancers of renal, ovarian, lung, colon, breast, prostate, and central nervous system origin. The NCI-60 have been profiled more comprehensively at the DNA, RNA, protein, and functional levels than any other set of cells in existence. The resulting information on molecular characteristics and their relationship to patterns of drug activity have proven fruitful for studies of drug mechanisms of action and resistance [28,29,30,31,25]. Because of its diversity, we expected to find mostly aberrations common only to specific tissue of origin, and possibly some that were found more generally in the panel.

We first compared three algorithms – simple unweighted penetrance, context-corrected penetrance, and CoCoA – on the breast tumor dataset. Overall, the three algorithms detected similar patterns, although the specific output contained obvious differences. In Figure 2 we show the output of the three algorithms for chromosome 9 of the breast tumor dataset. The top panel (a) depicts the aberration calls made on that set of samples, using the StepGram algorithm [20][1]. Several common aberrations, detectable by visual inspection, are indicated at the top of the panel by green and red arrows (deletions and amplifications, respectively). The lower three panels (b-d) depict the output of the three algorithms for the chromosome, aligned by genomic position along the x-axis. Output for the simple penetrance method is expressed in fraction of affected samples, whereas the output for the remaining algorithms is expressed in units of $-\log_{10}\rho(I)$. Note that while the output of the two penetrance algorithms (b,c) is simple to plot in genomic coordinates (by probe location), the output

[1] The data points were first centered by most common ploidy. StepGram was then applied with a threshold parameter of 5 stds.

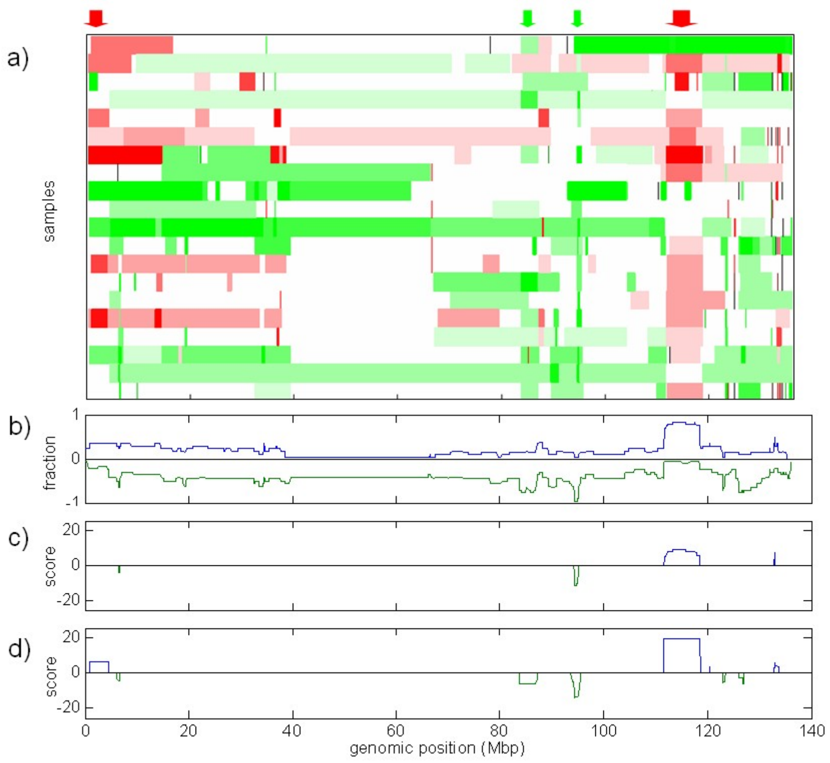


Fig. 2. Common aberrations in a panel of 20 breast tumor samples, chromosome 9: a) Aberration calls in each of the tumor samples (amplifications noted in red, deletion in green). Aberrations were called using StepGram algorithm [20] on centered data, with threshold of 5 stds; b) unweighted penetrance (fraction of samples), c) context-corrected penetrance, d) context-corrected common aberrations (CoCoA), where each probe was scored according to the maximal-scoring interval containing it. Positive values denote amplifications, negative values — deletions. Scores for last two methods are given in $-\log_{10}\rho(I)$ units, only aberrations with score $\rho(I) < 10^{-3}$ and larger than one probe are denoted. Some specific common aberrations in the data are highlighted by arrows at the top of the figure.

of the CoCoA algorithm was transformed into a genomic plot by setting the value of each probe to the score of the maximally-scoring common interval that contains it.

The most prominent common aberrations in the chromosome shown are clearly the large amplification between 110-120Mb and the smaller deletion at 95Mb, both of which were detected by all algorithms. The results of the simple penetrance method, which is a non-statistical method, can be interpreted loosely based on setting of some arbitrary threshold. It is clear that a significant part of the genome can be considered to contain common aberrations if that method is used. The context-corrected penetrance method gives improved output in the

Table 1. Number of common aberrations in the breast cancer data

	Amplifications	Deletions	Total
$<$ 200Kb	160	118	278
$\geq$ 200Kb	86	32	118
Total	246	150	396

a) b)

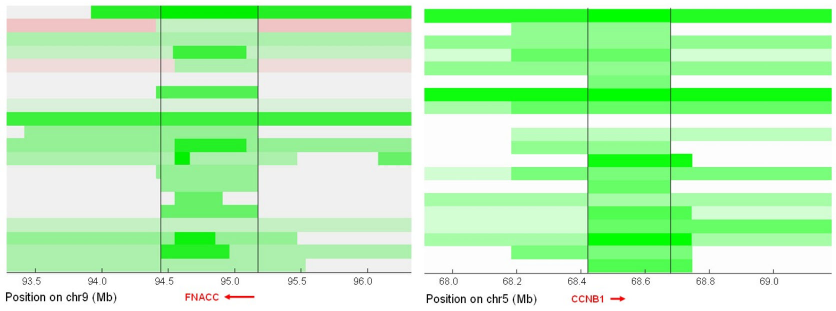


Fig. 3. Two common focal deletions identified in a panel of 20 breast tumor samples: a) Common deletion in 9q22.32 disrupting FANCC – a gene that encodes a DNA repair protein (11/20 samples, $\rho(I) = 10^{-21}$), b) Common deletion in 5q13.2 disrupting a cyclin gene CCNB1 (8/20 samples, $\rho(I) = 10^{-11.8}$)

sense that only very specific parts of the chromosome are deemed to contain common aberrations, based on a very modest threshold $\rho(I) < 10^{-3}$. Clearly, from the biological point of view, specific output of this type, a result of the correction for the chromosomal context, is highly preferable.

The superiority of the common aberrations method (CoCoA) lies in the higher significance that it gives common aberrations that are longer than one probe. This feature allows higher sensitivity for lower-amplitude common aberrations without loss of specificity. An example of the increased sensitivity is the common amplification detected between 1-5Mb. That aberration is not clearly visible in the outputs of the two methods based on single probe.

Overall, CoCoA identified 396 disjoint common aberrations with score $\rho(I) < 10^{-3}$ in the breast tumor dataset (see Table 1). The range of sizes of the common aberrations identified on the basis of more than a single probe is 1.7Kb - 60Mb. The aberrations are supported by 3-17 samples each. Two specific common focal deletions that were identified in the data set are depicted in Figure 3. The two

Table 2. Number of common aberrations in the NCI-60 data

	Amplifications	Deletions	Total
$<$ 200Kb	216	145	361
$\geq$ 200Kb	60	50	110
Total	276	195	471

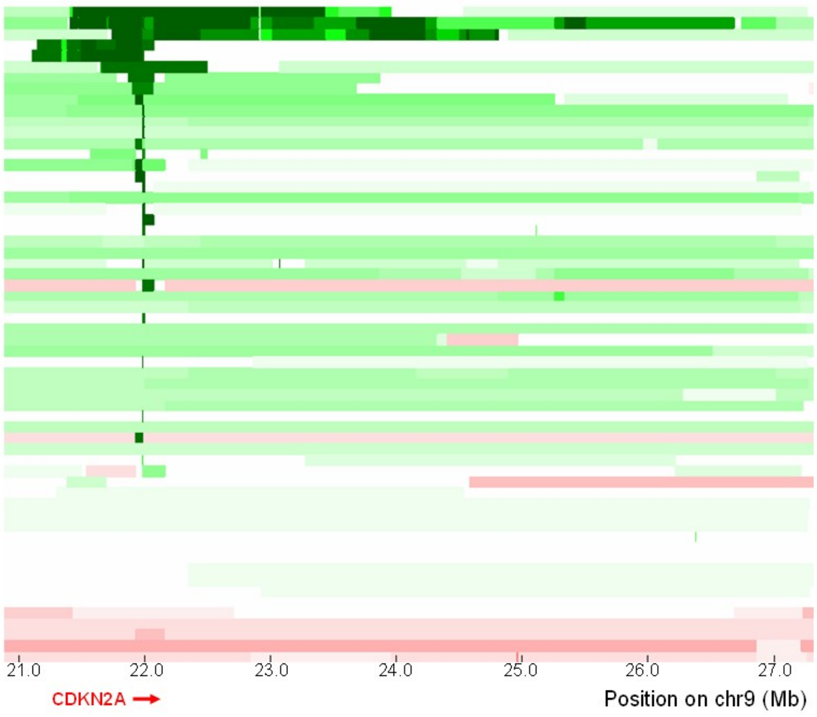


Fig. 4. A common deletion in 9p that reoccurs in a large fraction (20/60) of the cell-lines of the NCI-60 panel. Common aberration analysis points to the focus of the deletion as being the known tumor suppressor gene CDNK2A (p16), with $\rho(I) = 10^{-54}$.

deletions, identified in 5q13.2 and 9q22.32, appear to be disrupting two genes with direct involvement in tumor development – CCNB1 (a cyclin gene) and FANCC (a gene encoding a DNA repair protein), respectively. Slightly larger intervals are also aberrant in many samples. The highlighted intervals, however, have the strongest statistical significance.

In the NCI-60 cell line panel CoCoA identified 471 common aberrations (see Table 2). The range of sizes of the common aberrations identified on the basis of more than a single probe is 0.5kb - 100Mb, and aberrations are supported by 3-38 samples each.

One striking common aberration detected in the NCI-60 dataset was a deletion of CDKN2A (p16), a well-characterized tumor-suppressor gene (Figure 4). Clearly the deletion of this gene is a common feature of many of the cell-lines (20/60 of the samples), crossing the boundaries of cell-line subtype. Note also that even though some samples have deletions over larger regions, they all overlap at the genomic location of the p16 gene itself. This observation indicates that a selective pressure to delete p16 was part of the development of all 20 cell line populations and represents a very common feature of cancer development.

4 Discussion

In this paper we propose a computational framework for identifying and analyzing copy number aberrations (amplifications and deletions) that occur across multiple samples and for assessing their statistical significance. The framework allows using different aberration calling algorithms as input, independent of their statistical modeling.

Two central features of our methods are: A)When assessing the significance of a particular aberration, we use the height of the aberration, as opposed to requiring an additional threshold to discretize the aberration calls. B) The ability to address the context of the aberration structures in the individual samples. Given a candidate interval, its significance at a particular sample depends not only on the average height of the candidate interval, but also on the overall prevalence of aberrations in that sample. We describe two methods that have those important features. The CoCoA method scores intervals while the context-corrected penetrance method scores individual loci. In theory, there is a larger statistical power in considering multi-loci aberrations as both a supporting sample set and a genomic interval are identified together. Another difference between probe level and interval level analysis, is that in probe level analysis an additional thresholding step is required to determine the boundaries of the common aberrations. Note that for any single locus penetrance based method intervals with consistent high scoring can theoretically arise from aberrations in different sets of samples. In practice this is usually not the case. When scoring intervals, as CoCoA does, sample integrity is always preserved: the set of samples over which an interval is reported as a common aberration is the same for all loci spanned by said interval.

Our framework is very efficient. When run on the NCI60 sample set our process takes under 1 minute, including the first step of single sample aberration calling, using StepGram. This enables interactive data analysis that is not possible for less efficient approaches. This combined approach will scale up to larger datasets and to denser arrays that allow for much finer mapping of aberrant regions. We emphasize that this requires not only an efficient approach to common aberrations but also a very efficient aberration-calling methods.

One important previous formal treatment of calling common aberrations in CGH data is described in [18]. The method described therein, called STAC, is based on a heuristic search seeking to optimize statistical scores assigned to candidate regions of common aberrations. STAC's search is computationally intensive and performance is further limited by relying on permutations and simulations to obtain significance estimates. According to the paper's Supplementary material STAC implementation takes days to run on relatively small datasets of 42 and 47 samples, measured using a low resolution (approximately 1Mb) technology. STAC treats gains and losses as binary and does not takes into account the exact amplitude of the measured signal.

We have shown examples of applying the framework on a set of breast cancer samples that identify both known and novel cancer related genes. It is interesting to note p16 as a universal deletion in the NCI60 panel. FANCC, a gene from the Fanconi anemia group of genes (FA), which codes to a DNA repair protein is deleted in 11 out of the 20 breast cancer samples. FA genes are known to be co-factors interacting with BRCA2 in breast cancer pathogenesis. In a recent study [32] the authors demonstrate a role for the FA pathway in interstrand cross-link repair which is independent from that of BRCA2 in the same process. This finding and our implication of FANCC as a fairly focal common breast cancer deletion together suggest an important role for FANCC under-functioning in cancer pathogenesis.

Lastly, we note that the methods herein presented can be extended to identify differential aberrations in DNA copy number data coming from several phenotypic classes. A more detailed investigation of this application will be the topic of future work.

References

1. Kauraniemi, P., Hautaniemi, S., Autio, R., Astola, J., Monni, O., Elkahloun, A., Kallioniemi, A.: Effects of Herceptin treatment on global gene expression patterns in HER2-amplified and nonamplified breast cancer cell lines. Oncogene **23**(4) (2004) 1010–1013
2. Binh, M., Sastre-Garau, X., Guillou, L., de Pinieux, G., Terrier, P., Lagace, R., Aurias, A., Hostein, I., Coindre, J.: MDM2 and CDK4 immunostainings are useful adjuncts in diagnosing well-differentiated and dedifferentiated liposarcoma subtypes: A comparative analysis of 559 soft tissue neoplasms with genetic data. American Journal of Surgical Pathology **29**(10) (2005) 1340–1347
3. Balsara, B., Testa, J.: Chromosomal imbalances in human lung cancer. Oncogene **21**(45) (2002) 6877–83
4. Kallioniemi, O., Kallioniemi, A., Sudar, D., Rutovitz, D., Gray, J., Waldman, F., Pinkel, D.: Comparative genomic hybridization: a rapid new method for detecting and mapping DNA amplification in tumors. Semin Cancer Biol **4**(1) (1993) 41–46
5. Mertens, F., Johansson, B., Hoglund, M., Mitelman, F.: Chromosomal imbalance maps of malignant solid tumors: a cytogenetic survey of 3185 neoplasms. Cancer Research **57**(13) (1997) 2765–80
6. Barrett, M., Scheffer, A., Ben-Dor, A., Sampas, N., Lipson, D., Kincaid, R., Tsang, P., Curry, B., Baird, K., Meltzer, P., Yakhini, Z., Bruhn, L., Laderman, S.: Comparative genomic hybridization using oligonucleotide microarrays and total genomic DNA. PNAS **101**(51) (2004) 17765–17770
7. Bignell, G., Huang, J., Greshock, J., Watt, S., Butler, A., West, S., Grigorova, M., Jones, K., Wei, W., Stratton, M., Futreal, P., Weber, B., Shapero, M., Wooster, R.: High-resolution analysis of DNA copy number using oligonucleotide microarrays. Genome Research **14**(2) (2004) 287–95
8. Brennan, C., Zhang, Y., Leo, C., Feng, B., Cauwels, C., Aguirre, A., Kim, M., Protopopov, A., Chin, L.: High-resolution global profiling of genomic alterations with long oligonucleotide microarray. Cancer Research **64**(14) (2004) 4744–8
9. Hedenfalk, I., Ringner, M., Ben-Dor, A., Yakhini, Z., Chen, Y., Chebil, G., Ach, R., Loman, N., Olsson, H., Meltzer, P., Borg, A., Trent, J.: Molecular classification of familial non-BRCA1/BRCA2 breast cancer. PNAS **100**(5) (2003) 2532–7
10. Pinkel, D., Segraves, R., Sudar, D., Clark, S., Poole, I., Kowbel, D., Collins, C., Kuo, W., Chen, C., Zhai, Y., Dairkee, S., Ljung, B., Gray, J., Albertson, D.: High resolution analysis of DNA copy number variation using comparative genomic hybridization to microarrays. Nature Genetics **20**(2) (1998) 207–211

11. Pollack, J., Perou, C., Alizadeh, A., Eisen, M., Pergamenschikov, A., Williams, C., Jeffrey, S., Botstein, D., Brown, P.: Genome-wide analysis of DNA copy-number changes using cDNA microarrays. Nature Genetics **23**(1) (1999) 41–6
12. Sebat, J., Lakshmi, B., Troge, J., Alexander, J., Young, J., Lundin, P., Maner, S., Massa, H., Walker, M., Chi, M., Navin, N., Lucito, R., Healy, J., Hicks, J., Ye, K., Reiner, A., Gilliam, T., Trask, B., Patterson, N., Zetterberg, A., Wigler, M.: Large-scale copy number polymorphism in the human genome. Science **305**(5683) (2004) 525–8
13. Fridlyand, J., Snijders, A., Pinkel, D., Albertson, D., Jain, A.: Hidden markov models approach to the analysis of array cgh data. Journal of Multivariate Analysis **90** (2004) 132–153
14. Lai, W., Johnson, M., Kucherlapati, R., Park, P.: Comparative analysis of algorithms for identifying amplifications and deletions in array CGH data. Bioinformatics **21**(19) (2005) 3763–70
15. Olshen, A., Venkatraman, E., Lucito, R., Wigler, M.: Circular binary segmentation for the analysis of array-based dna copy number data. Biostatistics **5** (2004) 557–72
16. Wang, P., Kim, Y., Pollack, J., Narasimhan, B., Tibshirani, R.: A method for calling gains and losses in array CGH data. Biostatistics **6** (2005) 45–58
17. Willenbrock, H., Fridlyand, J.: A comparison study: applying segmentation to array CGH data for downstream analyses. Bioinformatics **21**(22) (2005) 4084–91
18. Diskin, S., Eck, T., Greshock, J., Mosse, Y., Naylor, T., Stoeckert, C., Weber, B., Maris, J., Grant, G.: STAC: a method for testing the significance of DNA copy number aberrations across multiple array-CGH experiments. Genome Research **16** (2006) 1149–1158
19. Rouveirol, C., Stransky, N., Hupe, P., Rosa, P.L., Viara, E., Barillot, E., Radvanyi, F.: Computation of reccurant minimla genomic alterations from array-cgh data. Bioinformatics (2006) 849–856
20. Lipson, D., Aumann, Y., Ben-Dor, A., Linial, N., Yakhini, Z.: Efficient calculation of interval scores for DNA copy number data analysis. Journal of Computational Biology **13**(2) (2006) 215–28
21. Lingjarde, O.C., Baumbusch, L.O., Liestol, K., Glad, I.K., Borresen-Dale, A.L.: Cgh-explorer: a program for analysis of array-cgh data. Bioinformatics **21(6)** (2005) 821–822
22. Wiedswang, G., Borgen, E., Kvalheim, R.K.G., Nesland, J., Qvist, H., Schlichting, E., Sauer, T., Janbu, J., Harbitz, T., Naume, B.: Detection of isolated tumor cells in bone marrow is an independent prognostic factor in breast cancer. Journal of Clinical Oncology **21** (2003) 3469–3478
23. Sorlie, T., Perou, C., Tibshirani, R., Aas, T., Geisler, S., Johnsen, H., Hastie, T., Eisen, M., van de Rijn, M., Jeffrey, S., Thorsen, T., Quist, H., Matese, J., Brown, P., Botstein, D., Lonning, P.E., Borresen-Dale, A.: Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. PNAS **98**(10) (2001) 10869–74
24. Sorlie, T., Wang, Y., Xiao, C., Johnsen, H., Naume, B., Samaha, R., Borresen-Dale, A.L.: Distinct molecular mechanisms underlying clinically relevant subtypes of breast cancer: Gene expression analyses across three different platforms. BMC Genomics **7** (2006) 127
25. Weinstein, J., Myers, T., O'Connor, P., Friend, S., Jr., A.F., Kohn, K., Fojo, T., Bates, S., Rubinstein, L., Anderson, N., Buolamwini, J., van Osdol, W., Monks, A., Scudiero, D., Sausville, E., Zaharevitz, D., Bunow, B., Viswanadhan, V., Johnson, G., Wittes, R., Paull, K.: An information-intensive approach to the molecular pharmacology of cancer. Science **275**(10) (1997) 343–49

26. Monks, A., Scudiero, D., Skehan, P., Shoemaker, R., Paull, K., Vistica, D., Hose, C., Langley, J., Cronise, P., et al., A.V.W.: Feasibility of a high-flux anticancer drug screen using a diverse panel of cultured human tumor cell lines. Journal of the National Cancer Institute **83** (1991) 757–66
27. Shoemaker, R., Monks, A., Alley, M., Scudiero, D., Fine, D., McLemore, T., Abbott, B., Paull, K., Mayo, J., Boyd, M.: Development of human tumor cell line panels for use in disease-oriented drug screening. Progress in Clinical and Biological Research **276** (1988) 265–86
28. Nishizuka, S., Charboneau, L., Young, L., Major, S., Reinhold, W., Waltham, M., Kouros-Mehr, H., Bussey, K., Lee, J., Espina, V., Munson, P., 3rd, E.P., Liotta, L., Weinstein, J.: Proteomic profiling of the nci-60 cancer cell lines using new high-density reverse-phase lysate microarrays. PNAS **100** (2003) 14229–34
29. Paull, K., Shoemaker, R., Hodes, L., Monks, A., Scudiero, D., Rubinstein, L., Plowman, J., Boyd, M.: Display and analysis of patterns of differential activity of drugs against human tumor cell lines: development of mean graph and compare algorithm. Journal of the National Cancer Institute **81** (1989) 1088–92
30. Shi, L., Fan, Y., Lee, J., Waltham, M., Andrews, D.T., Scherf, U., Paull, K., Weinstein, J.: Mining and visualizing large anticancer drug discovery databases. Journal of Chemical Information and Computer Sciences **40** (2000) 367–79
31. Staunton, J., Slonim, D., Coller, H., Tamayo, P., Angelo, M., Park, J., Scherf, U., Lee, J., Reinhold, W., Weinstein, J., Mesirov, J., Lander, E., Golub, T.: Chemosensitivity prediction by transcriptional profiling. PNAS **98** (2001) 10787–92
32. Kitao, H., Yamamoto, K., Matsushita, N., Ohzeki, M., Ishiai, M., Takata, M.: Functional interplay between brca2/fancd1 and fancc in dna repair. Journal of Biological Chemistry **281(30)** (2006) 21312–21320

Estimating Genome-Wide Copy Number Using Allele Specific Mixture Models

Wenyi Wang[1], Benilton Carvalho[2], Nate Miller[3], Jonathan Pevsner[4], Aravinda Chakravarti[5], and Rafael A. Irizarry[6,⋆]

[1] Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health, 615 North Wolfe Street, Baltimore, MD 21205, USA
wwang2@jhsph.edu

[2] Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health, 615 North Wolfe Street, Baltimore, MD 21205, USA
bcarvalh@jhsph.edu

[3] Department of Neurology, Kennedy Krieger Institute, 707 North Broadway, Baltimore, MD 21205, USA
natemiller@jhmi.edu

[4] Department of Neurology, Kennedy Krieger Institute, 707 North Broadway, Baltimore, MD 21205, USA
pevsner@kennedykriger.org

[5] McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University School of Medicine Broadway Research Building, Suite 579, 733 N. Broadway, Baltimore, MD 21205,USA
aravinda@jhmi.edu

[6] Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health, 615 North Wolfe Street, Baltimore, MD 21205, USA
rafa@jhu.edu

Abstract. Genomic changes such as copy number alterations are thought to be one of the major underlying causes of human phenotypic variation among normal and disease subjects [23,11,25,26,5,4,7,18]. These include chromosomal regions with so-called copy number alterations: instead of the expected two copies, a section of the chromosome for a particular individual may have zero copies (homozygous deletion), one copy (hemizygous deletions), or more than two copies (amplifications). The canonical example is Down syndrome which is caused by an extra copy of chromosome 21. Identification of such abnormalities in smaller regions has been of great interest, because it is believed to be an underlying cause of cancer.

More than one decade ago comparative genomic hybridization (CGH) technology was developed to detect copy number changes in a high-throughput fashion. However, this technology only provides a 10 MB resolution which limits the ability to detect copy number alterations spanning small regions. It is widely believed that a copy number alteration as small as one base can have significant downstream effects, thus microarray manufacturers have developed technologies that provide much higher resolution. Unfortunately, strong probe effects and variation introduced by sample preparation procedures have made single-point copy number

⋆ Corresponding author.

estimates too imprecise to be useful. CGH arrays use a two-color hybridization, usually comparing a sample of interest to a reference sample, which to some degree removes the probe effect. However, the resolution is not nearly high enough to provide single-point copy number estimates.

Various groups have proposed statistical procedures that pool data from neighboring locations to successfully improve precision. However, these procedure need to average across relatively large regions to work effectively thus greatly reducing the resolution. Recently, regression-type models that account for probe-effect have been proposed and appear to improve accuracy as well as precision. In this paper, we propose a mixture model solution specifically designed for single-point estimation, that provides various advantages over the existing methodology. We use a 314 sample database, constructed with public datasets, to motivate and fit models for the conditional distribution of the observed intensities given allele specific copy numbers. With the estimated models in place we can compute posterior probabilities that provide a useful prediction rule as well as a confidence measure for each call. Software to implement this procedure will be available in the Bioconductor `oligo` package (http://www.bioconductor.org).

1 Introduction

The demand for technologies that provide high-resolution measurements for copy number estimates has driven microarray manufacturers such as Illumina and Nimblegen to develop CGH-SNP microarrays [20,6,24]. Although Affymetrix has not yet developed a product specific for copy number analysis, various groups, including Affymetrix, have described statistical methodology that make use of their SNP chips, originally developed for genotyping [28,1,9,19,13,15,8,16], to provide successful copy number estimation algorithms. An advantage of the SNP chip technology is that, given the popularity of the genotyping application, the protocols used to prepare the genomic DNA samples is well developed and tested. Another advantage is that we can obtain genotype calls which permits allele specific copy number estimation which in turn can be used to predict parent specific copy number [16].

The genotyping platform provided by Affymetrix interrogates hundreds of thousands of human single nucleotide polymorphisms (SNPs) on a microarray. DNA is obtained and fragmented at known locations so that the SNPs are far from the ends of these fragments, the fragmented DNA is amplified with a polymerase chain reaction (PCR), and the sample is labeled and hybridized to an array containing probes designed to interrogate the resulting fragments. We refer to the measurements obtained from these probes as the *feature intensities.* There are currently three products available from Affymetrix: an array covering approximately 10,000 SNPs (GeneChip Human Mapping 10K), a pair of arrays covering approximately 100,000 SNPs (GeneChip Human Mapping 50K Xba and Hind Array), and a pair of arrays covering approximately 500,000 SNPs (GeneChip Human Mapping 250K Nsp Array and Sty Array). These are referred to as the 10K, 100K, and 500K chips respectively.

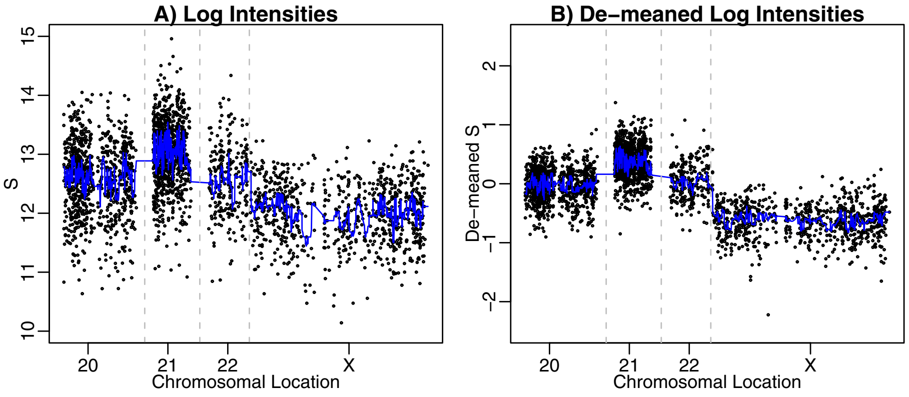


Fig. 1. Log (base 2) intensity, $S_{i,j}$ plotted against chromosomal position of SNP i. Chromosomes 20, 21, 22, and X are shown for male with Down syndrome. Notice that we expect values from chromosome 21 and X to be higher (3 copies) and lower (1 copy) respectively.

To motivate the model and estimation procedures described here we need to understand the basics of the feature-level data. We provide the essential details here and refer readers to Kennedy et al. [14] for a complete description. Each SNP on the array is represented by a collection of probe quartets. As with Affymetrix expression arrays, the probes are defined by 25-mer oligonucleotide molecules referred to as perfect match (PM) probes[1]. A difference with expression arrays is that PM probes differ in three important ways. First, two alleles are interrogated (for most SNPs only two alleles are observed in nature). These are denoted by A and B and divide the probes into two groups of equal size. For each PM probe representing the A allele there is an allele B that differs by just one base pair (the SNP). Second, features are included to represent the sense and antisense strands. This difference divides the probes into two groups that are not necessarily of the same size. Finally, for each allele/strand combination, various features are added by shifting the position of the SNP within the probe. The position shift ranges only from -4 to 4 bases, therefore within each strands the probes are relatively similar.

Most copy number algorithms can be divided into three main steps which we refer to as 1) the *preprocessing* step, 2) the copy number *estimation* step, and 3) the *smoothing* across the chromosome step. In the preprocessing step we summarize feature intensities into two quantities, representative of allele A and B. We refer to this step as *preprocessing.* In this paper, we use the following notation to denote the preprocessed data: $\theta_{A,i,j}$ and $\theta_{B,i,j}$ are the logarithms (base 2) of quantities proportional to the amount of DNA in target sample j associated with alleles A and B for SNP i. In the estimation step, we use these θs to estimate the true copy number, which we denote with $C_{i,j}$. The allele specific copy number are denoted with $C_{A,i,j}$ and $C_{B,j,i}$. Notice that the total copy

[1] There are also mismatch probes MM which we completely ignore because the manufacture has plans of no longer using them.

number is the sum of the allele specific copy numbers, i.e. $C_{i,j} = C_{A,i,j} + C_{B,i,j}$. As we demonstrate here (see Figure 1), estimates of θ_A and θ_B are, in general, not precise enough to provide useful copy number calls. Therefore, most copy number estimation algorithms include the smoothing step in which copy number estimates from neighboring regions are averaged to improve the signal to noise ratio. These techniques range from the simple methods such as running median to more complicated ones such as hidden Markov models (HMM). In Section 3 we review some of the existing methods and motivate our mixture model approach. In Section 4 we describe the mixture model approach. In Sections 5 we present results and discussion respectively. Throughout this extended abstract we use data obtained from collaborators and public repositories which we briefly describe in Section 2.

2 Control Data

In Section 4 we describe a model that is trained using a reference set of 314 normal samples hybridized to Affymetrix's 100K array. We screened out samples not achieving the quality standard described by Carvalho et al [2]. Our reference set consists of 86 Hapmap samples, 124 samples from the Coriell Repositories (42 African American, 20 Asians, 40 Caucasians and 22 samples from the polymorphisms discovery panel) [3], and 104 from Chakravarti's lab. The test data was sampled from 20 Trisomy samples from Pevsner's lab.

3 Previous Work and Motivation

The first algorithms we describe do not provide allele specific results. We therefore define the total copy number quantity $S_{i,j} = \log_2(2^{\theta_{A,i,j}} + 2^{\theta_{B,i,j}})$. Ideally the $S_{i,j}$ are proportional to the true log-scale copy number. Figure 1A shows data for a male with Down syndrome. The $S_{i,j}$ are highly noisy and differences between chromosome 21 and X are hard to detect unless we smooth along the chromosome (in Figure 1 we show the results of running median). The lessons learned from expression arrays help understand this problem. Various authors have proposed an additive background/multiplicative model for gene expression microarray [22,10,27]. Furthermore, for Affymetrix arrays, various authors [12,17] have clearly shown the existence of a strong multiplicative probe-specific effect. Probe-specific background noise, attributed to non-specific binding, have also been described [27]. Others [21,16,2] demonstrate that similar sized effects are seen with SNP chips. These effects are strong enough to be clearly seen even after averaging the various feature intensities associated with each SNP. Extending these findings to the copy number case results in the following model:

$$\theta_{a,i,j} = \log\left(\beta_{a,i} + \phi_{a,i} c_a \varepsilon_{a,i,j}\right) \qquad (1)$$

with $a = A, B$ denoting allele, i identifies the SNP, j identifies the sample, β represents a positive valued SNP-specific background level, ϕ represents a

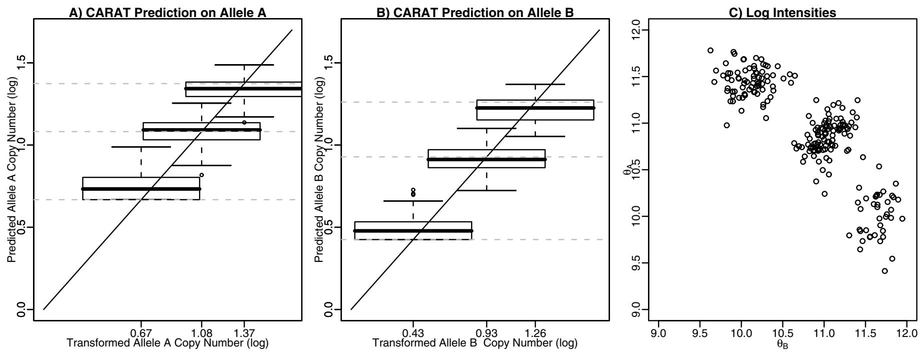


Fig. 2. CARAT regression prediction on a SNP from Chromosome X. The dashed grey lines indicate allele specific copy number = 0,1,2 (from low to high). Figure A) and B) shows the allele specific prediction of copy number (log base 2) as compared to the real copy number (log base 2). We can see that even though the middle of each boxplot lies closely to the intersection of the dashed lines and the diagonal line, the ranges of boxplots overlap. Figure C) shows a scatterplot of preprocessed log intensities for allele A and B.

SNP-specific probe-effect, and ε is multiplicative measurement error. Assuming this model holds, relatively simple calculations demonstrate that large values of $\beta_{a,i}$ result in attenuation of real differences and that large variability of $\phi_{a,i}$ across SNPs explains the large variance seen in Figure 1A.

Affymetrix's Copy Number Analysis Tool (CNAT) [1,9] deals with the probe-effect using a simple yet effective technique. CNAT does not provide allele specific results and concentrates on estimating the overall copy number. For the preprocessing step, all feature intensities related to the SNP are therefore averaged to form $S_{i,j}$. Using dozens of control subjects CNAT defines a SNP specific average $\bar{S}_{ig}$, standard deviation $\hat{\sigma}_{ig}$, for each genotype g=AA,AB,BB. Values $S_{i,j'}$, from any new sample j' that are called genotype g are standardized in the usual manner: $(S_{i,j'} - \bar{S}_{ig})/\hat{\sigma}_{ig}$. A predefined regression equation is then used to transform these standardized values to the copy number scale. The standardized S values are used to obtain p-values from the null hypothesis the $S = 0$ $(C = 2)$. Figure 1B shows de-meaned values for the same data shown in Figure 1A. The improvement is clear and it is due to the fact that ϕ is partially removed from the de-mean-ed values. However, notice that the signal to noise ratio still appears to be large: the separation between chromosomes with known differences is far from perfect. To avoid false positives, the third step in CNAT involves looking for strings of consecutive p-values that are smaller than some predefined cut-off. Other smoothing approaches have been used. For example, Zhao et al. [28] proposes the use Hidden Markov models (HMM) to define the procedure implemented by dChip.

Other authors have noted that further improvements can be obtained by reducing the variance at the preprocessing step. For example, several groups

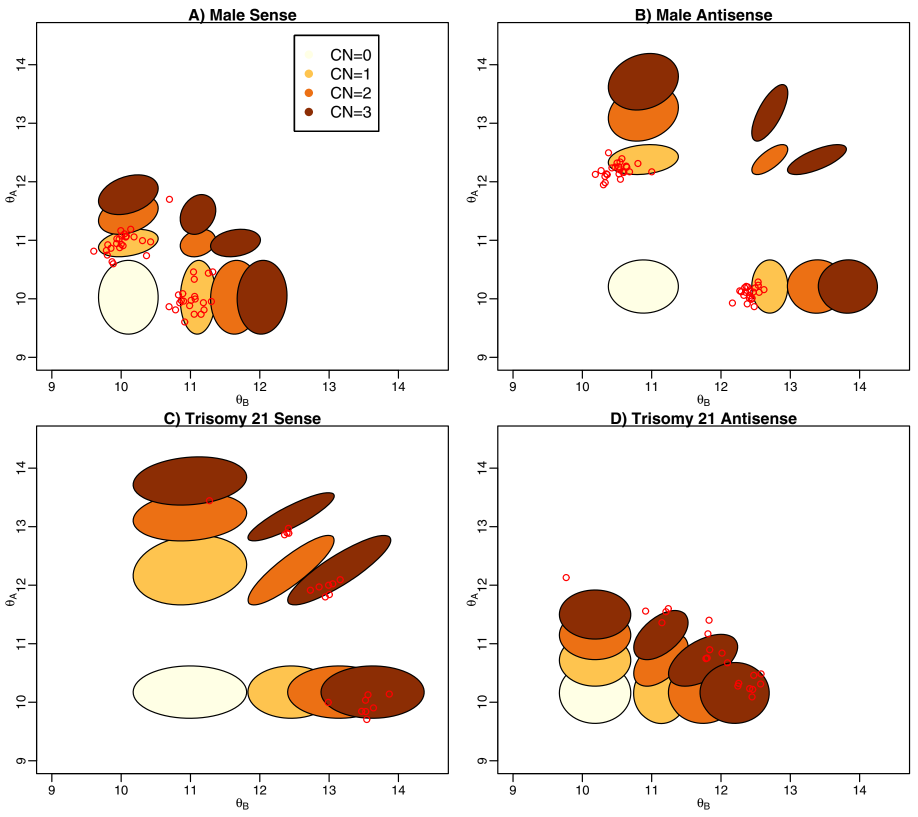


Fig. 3. Conditional joint distributions of $[\boldsymbol{\theta}_{i,j}|\mathbf{C}_{i,j} = \mathbf{c}]$ for a SNP on chromosome X and a SNP on chromosome 21. The X-axis is the log base 2 intensity for allele B. The Y-axis is the log base 2 intensity for allele A. The red dots are from 45 new samples of male normal and 20 Down syndrome patients, respectively. The ellipses shows the 95% critical region around the centers. The brown ellipses represent $C = 3$, $\mathbf{C}_{i,j} = (0,3),(1,2),(2,1),(3,0)$. The orange ellipses represent $C = 2$, $\mathbf{C}_{i,j} = (0,2),(1,1),(2,0)$. The tan ellipses represent $C = 1$, $\mathbf{C}_{i,j} = (0,1),(1,0)$. The yellow ellipses represent $C = 0$, $\mathbf{C}_{i,j} = (0,0)$.

[19,13,15] have used probe-sequence information, mainly GC content, and fragment length to predict and remove some of the probe-effect related variability. However, even after accounting for such factors the signal to noise ratio remains too low to make single-point cop number calls, thus these authors propose their own versions of the smoothing step.

Huang et al. [8] noted that CNAT's mean removal approach does not fully remove the probe effect because it does not properly deal with the additive background effect β. They propose the Copy Number Analysis with Regression And Tree (CARAT) algorithm which uses a non-linear regression model, based on model (1), to account for the probe-specific effects. To estimate model parameters they use a control dataset composed of dozens of arrays. First, genotype calls are obtained and treated as known. This permits estimation of allele specific parameter estimates. For example, for allele A we have known values of $c_A = 0, c_B = 2$ (BB genotype), $c_A = 1, c_B = 1$ (AB genotype), and ($c_A = 2, c_B = 0$ (AA genotype) and thus we can estimate $\beta_{a,i}$ and $\phi_{a,i}$ with, for example, least squares, for each SNP i and each allele $a = A, B$. For a new sample, we can predict c_A, c_B using the fitted parameters and equation (1). Calls can then be based on cut-offs for the prediction of $\hat{c}_A + \hat{c}_B$. Huang et al. [8] suggest using $[0, 1.5)$, $[1.5, 2.5]$, $(2.5, \infty)$ for total copy number $< 2, = 2, > 2$ respectively. Figure 2A shows the data used in the regression for allele A from a randomly chosen SNP. The figure demonstrates that the model works reasonably well but that the signal to noise ratio is not large enough to provide perfect accuracy (the boxplots overlap). CARAT utilized a regression tree approach in the smoothing step.

The Probe-level allele-specific quantitation (PLASQ) [16] procedure is similar to CARAT. Two major difference is that PLASQ fits model (1) to the feature-level data and that PLASQ does not rely on external genotype calls. Although, in our opinion, PLASQ provides a superior model-based framework than any other approach, it is computationally challenging to implement. This is because a non-linear estimation procedure is performed at the feature-level for every SNP. Furthermore, it is difficult to adapt it to be robust to outliers and to take probe-sequence and fragment size into account.

Model based approaches such as CARAT and PLASQ provide a great advantage over previous ones: reliable confidence intervals can be computed for single-point copy number estimates. Huang et al. [8] point out that their uncertainty assessment permits one to call a relatively large group of SNPs and keep the false positive rate relatively low. We now briefly describe a simple adaptation of these methods that provides further improvements.

Notice that all of the above described algorithms use regression-type approaches to give a continuous prediction of copy number. The current approaches rely on three assumptions that we believe are not exactly true. The first is that the linear relationship predicted by model 1. Figure 2A and 2B show that there are small but significant deviations from these models. Other SNPs (not shown) show slightly larger deviations. The second is that θ_A and θ_B are independent. This assumption is clearly not true as demonstrated by Figures 2C and 3. The third assumption is that the variance of the measurement error term does not depend on allele-specific copy number values. Figure 3 also shows this is not the case. In general, we are making convenience assumptions regarding the conditional probabilities of $(\theta_A, \theta_B)'$ given allele-specific copy number that might be hurting our bottom-line results.

Theoretically, one can show that the best predictor of discrete classes given continuous covariates is Bayes classifier. Bayes classifier is a function of the conditional distribution of the predictors given the classes which we can not always estimate effectively. In the next section we describe how we can use the large amount of public data and equation (1) to obtain useful estimates of these conditional distributions and therefore improved copy number calls.

Another problem with the above mentioned approaches is that they do not take into account the strong lab/study effects that are seen in SNP chip data [2]. Huang et al. [8] notice this problem and propose an ad-hoc solution that relies on having genotype calls for many samples from each study. This approach is clearly not usable in cases where we wish to obtain copy number calls for a small batch of new samples. Recently, Carvalho et al. [2] proposed a preprocessing algorithm, SNP-RMA, that remove much of the lab-effect. We use SNP-RMA preprocessed values throughout this extended abstract. These authors also describe the CRLMM procedure which we use to obtain genotype calls for the control dataset.

4 Allele-Specific Mixture Model

Figure 2C shows a scatterplot of θ_A vs. θ_B values from hundreds of control arrays for a particular SNP. The three genotypes are clearly seen. Furthermore each of the three “clouds” looks bivariate normal with the AB genotype showing signs of correlation. Most SNPs show very similar plots and motivate the following model:

$$[\boldsymbol{\theta}_{i,j}|\mathbf{C}_{i,j} = \mathbf{c}] = \begin{pmatrix} \gamma_{A,c_A,i} \\ \gamma_{B,c_B,i} \end{pmatrix} + \begin{pmatrix} \epsilon_{A,c_A,i,j} \\ \epsilon_{A,c_B,i,j} \end{pmatrix} \tag{2}$$

with $\boldsymbol{\theta}_{i,j} = (\theta_A, \theta_B)'$, $\mathbf{C}_{i,j} = (C_{A,i,j}, C_{B,i,j})'$ representing the un-observed true copy number of alleles A and B for SNP i on sample j, $\mathbf{c} = (c_A, c_B)'$ are the possible values $\mathbf{C}_{i,j}$ can take, $(\gamma_{A,c_A,i}, \gamma_{B,c_B,i})'$ accounts for the shifts in location caused by the probe-effect, and $(\epsilon_{A,c_A,i,j}, \epsilon_{B,c_B,i,j})'$ is a bivariate normal error with mean 0 and copy-number-specific covariance matrix $\Sigma_{c,i}$ which is defined by the allele-copy-number-specific standard deviations $\sigma_{c_A,i}, \sigma_{c_B,i}$ and copy-number-pair-specific correlation $\rho_{c_A,c_B,i}$.

Remember that the sequence composition of the sense and antisense probes are quite different. Carvalho et al. [2] point out that, for a few hundred SNPs, one of the two strands does not appear to be sensitive to genotype changes. This difference is also observed with probes within the same strand. Huang et al. [8] used arbitrary cutoffs to select probes with strong allelic dosage response for further analysis. For this reason, we propose fitting the above model for probe-level data. Currently, the SNP-RMA preprocessing provides, by default, separate values of θ_A and θ_B for sense and antisense. However, without loss of generality we describe the procedure as if there was only one probe.

For a large set of control data, described in Section 2, we obtain genotype calls, act as if these are known. This implies we know $\mathbf{C}$ for all these samples and we can estimate the γs by simply using:

$$\hat{\gamma}_{A,c_A,i} = N_{c_A,i}^{-1} \sum_{\{j:\ C_{A,i,j}=c_A\}} \theta_{A,i,j}, \quad c_A = 0,1,2 \tag{3}$$

with $N_{c_A,i}$ the number of samples with genotypes implying $C_{A,i,j} = c_A$. The covariance matrix $\Sigma_{\mathbf{c},i,s}$ is computed in a similar way, namely using the sample covariance matrix of $\boldsymbol{\theta}_{i,j}$ for samples j implying $\mathbf{C}_{i,j} = \mathbf{c}$.[2] Because we assume the ϵs are normal, these sets of parameter estimates define the conditional distributions for $\mathbf{C} = (2,0),(1,1)$ and $(0,2)$. Notice that model (1) is not used to form any of these conditional distributions. Next we assume that the behavior of the $\boldsymbol{\theta}$s for $C_A = c_A$ is similar for all values of C_B and vice-versa. We then infer the conditional means for $\mathbf{C} = (0,0),(0,1),(1,0)$. For example the conditional mean for SNP i when $\mathbf{C} = (0,1)$ will be $(\gamma_{A,0,i}, \gamma_{B,1,i})$. The covariance matrix is inferred in a similar way (described below). To predict $(\gamma_{A,c_A,i}, \gamma_{B,c_B,i})'$ for $\mathbf{C} = (3,0)$,$(2,1)$,$(1,2)$, $(0,3)$,$(4,0)$,$(3,1)$,..., we use model 1. For example, we first use the estimates of $\gamma_{A,0,i}$, $\gamma_{A,1,i}$, $\gamma_{A,2,i}$ as outcomes in model (1) for values of $C_A = 0,1,2$ respectively, fit the model, and obtain estimates of $\beta_{A,i}$ and $\phi_{A,i}$, which permit us to predict $\gamma_{A,3,i}$, for $C_A = 3$.

We now describe how we infer $\Sigma_{\mathbf{c},i}$ for cases other than $\mathbf{C} = (2,0)$, $(1,1)$,$(0,2)$ using the estimates we already have. For the A and B variance components (the diagonal entry) of the covariance matrix, we simply assume they depend only on c_A and c_B respectively. For $c_A > 2$ and $c_B > 2$ we assume the same variance as $c_A = 2$ and $c_B = 2$ respectively. We therefore use the estimates of the six parameters: $\sigma_{A,c_A,i}, c_A = 0,1,2$, $\sigma_{B,c_B,i}, c_B = 0,1,2$ and do not need to predict any new values. The correlation component is a bit more difficult. We assume that the correlation coefficient when $C_A > 0$ and $C_B > 0$ is the same as $\mathbf{C} = (1,1)$. The rationale for this is that correlations are due to PCR effects being different from sample to sample. Thus if both allele fragments are present, the resulting quantities will be similar regardless of the starting quantities. When one of the two alleles is not present (PCR no longer makes it grow) we assume that the correlation for case where $C_A > 0$ but $C_B = 0$ is the same as $\mathbf{C} = (2,0)$ and $C_A = 0$ but $C_B > 0$ the same as $\mathbf{C} = (0,2)$. For $\mathbf{C} = (0,0)$ we simply assume independence. With this assumption in place we can produce conditional expectations for any value of $\mathbf{C}$ given the observed $\boldsymbol{\theta}$s, described as follows.

With the model parameter estimates in place we are able to provide posterior probabilities for allele specific copy number. Furthermore, we can compute these posterior probabilities for total copy number:

$$[C_{A,i,j} + C_{B,i,j} = c | \boldsymbol{\theta}_{i,j}] \propto \sum_{\{\mathbf{c}:\ c_A + c_B = c\}} [\boldsymbol{\theta}_{i,j} | \mathbf{C}_{i,j} = \mathbf{c}] \times [\mathbf{C}_{i,j} = \mathbf{c}]$$

where $[\boldsymbol{\theta}_{i,j} | \mathbf{C}_{i,j} = \mathbf{c}]$ is the bivariate normal distribution defined by model (3). The marginal probability of the $\mathbf{C}$ pair can be pre-specified and used to control specificity and sensitivity for any copy number value. We can obtain meaningful values by decomposing the probability into: $\Pr(C_{A,i,j} = c_A, C_{B,i,j} = c_B) = \Pr(C_{A,i,j} = c_A, C_{B,i,j} = c_B | C_{A,i,j} + C_{B,i,j} = c) \Pr(C_{A,i,j} + C_{B,i,j} = c)$. The first component relates to the proportion of each genotype in the population and can be computed using the Hardy-Weinberg Equilibrium for diploids

[2] In this extended abstract we actually use robust (to outliers) versions of these sample means and covariances.

($C_{A,i,j} + C_{B,i,j} = 2$). The second component relates to the probability of each alteration $c = 0, 1, 3, 4, \ldots$ which is unknown. We recommend the user define these probabilities to control specificity and sensitivity. For the examples shown in this extended abstract we assigned equal probabilities to $C_{A,i,j} + C_{B,i,j} = 0, 1, \ldots, 6$.[3] Once we have calculated the probabilities above we can provide estimates of copy number by, for example, computing the expected value of $C = C_A + C_B$.

A summary of the algorithm:

1. For each array, we obtain the pre-processed probe-level log intensities from snpRMA, the pre-processing algorithm used by CRLMM. These resulting measurements are $\theta_{A,+}$,$\theta_{A,-}$,$\theta_{B,+}$,$\theta_{B,-}$ for each SNP.
2. We estimate the conditional probability of these measurements, given allele specific copy number. We assume a bivariate normal for the A and B alleles at each copy number pair. This reduces the number of parameters greatly and we can estimate them precisely using a large training set. We use genotype calls to treat the allele specific copy number as known. We do this independently for sense (+) and antisense (-). More specifically:
3. We assume the prior probability for the joint distribution of C_A and C_B is a uniform distribution.
4. For a new dataset, we use the above estimates to calculate the posterior probability for C_A and C_B being $0, 1, 2, \ldots, K$ (K is the maximum copy number permitted). We average the sense and antisense results.
5. Finally, we compute the posterior probability of $C_A + C_B$ being $0, 1, 2, \ldots, K$.

5 Results

We now describe some of the applications of the hierarchical model described above. In general we refer to our procedure as the Copy Number-Robust Linear Model and Mixture Model (CN-RLMM) procedure.

Figure 3 gives the SNP-specific bivariate normal distribution of $\boldsymbol{\theta}$ for $\mathbf{C} = (0,0), (0,1), (1,0), (0,2), (1,1), (2,0), (0,3), (1,2), (2,1), (3,0)$, depicted in ellipses (95% confidence regions). These are estimated from the control data as described in Section 2. Figure 3A and 3B give the sense and antisense-specific distributions for a SNP on chromosome X. We observe the extrapolated distribution of $\boldsymbol{\theta}$s given $\mathbf{C} = (0,1), (1,0)$ coincide with the observed $\boldsymbol{\theta}$s from 45 male samples that were not used in training. Similarly, figure 3C and 3D give the sense and antisense-specific distributions for a SNP on chromosome 21 and we observe that our extrapolated distributions coincide with the observed $\boldsymbol{\theta}$s from 20 Trisomy samples. This demonstrates that our assumptions seem to provide reasonable estimates of the conditional distribution of copy number the cases predicted with mode (1) ($\mathbf{C} = (0,0), (0,1), (1,0), (0,3), (1,2), (2,1), (3,0)$).

In Figure 4A and 4B we demonstrate how our results have much better precision than CNAT and values with and without probe- sequence and fragment

[3] Remember that we perform the above calculation separately for the sense and antisense values. A final estimate of the posterior probability simply average these two values.

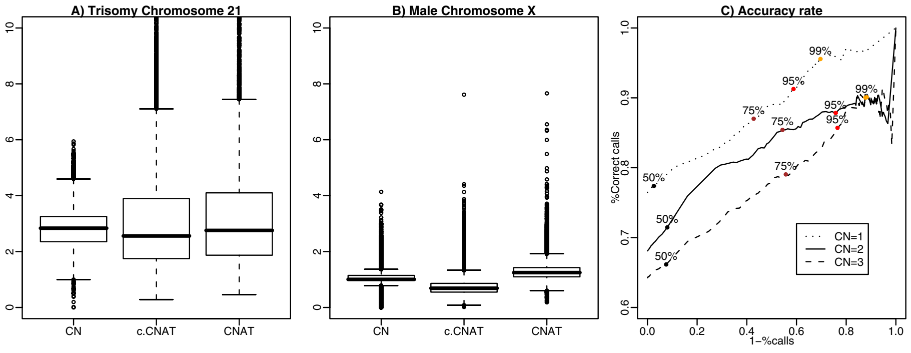


Fig. 4. CN-RLMM results. CN is abbreviation for CN-RLMM, c.CNAT is CRLMM preprocessed probes + CNAT. Figure A) shows the expected copy number given preprocessed log intensities for 817 SNPs on Chromosome 21 of 20 Down syndrome patients (with identified Trisomy 21). Figure B) shows the expected copy number given preprocessed log intensities for 807 SNPs on Chromosome X of 45 male trio samples. Figure C) shows the average true positive rate versus 1-average percentage of call rate for 2 Down syndrome patients. The points demonstrate some of the corresponding posterior probability values used as cut-offs.

length corrections. We achieve this precision without any loss of accuracy. Note that we could not get PLASQ to work with our data and no software is available to implement CARAT. The preprocessing used by dChip is very similar to CNAT and thus we expect results to be the same. Keep in mind the smoothing step is not being assessed. We observe that the degree of improvement is not equivalent for copy numbers 3 and copy number 1. This is expected because it is easier to detect a 2 times difference (copy number 2 versus 1) than to detect a 1.5 times difference (copy number 3 versus 2).

The most useful application of our results is that we provide improved single-point copy number estimates with reliable uncertainty assessment without the need to re-calibrate for new samples. Note that we can easily control our false positive rate by simply restricting calls to SNPs with posterior probabilities close to 1. Figure 4C demonstrates that we can get usable single-point copy number estimates for a large amount of SNPs. Notice that the worst performance is observed for CN=3. This is likely due to the fact that we used model (1) to extrapolate (as done by CARAT).

6 Discussion

We have presented a mixture model approach that permits us to obtain improved copy number estimate as well as reliable single-point copy number calls. A major advantage of our methodology over the best existing one, e.g. CARAT and PLASQ, is that we explicitly model the conditional joint distribution of the intensities given the copy number values. This permits us to model the strong correlation that sometimes exists between A and B and exploit this information to improve bottom-line results. This advantage is best exemplified by Figure 3C where the $\mathbf{C} = (2,1), (1,1)$, and $(1,2)$ are usefully separable only if we take this correlation into account. Furthermore, avoiding the linearity assumption made by these other procedures seems to help as well. This is best demonstrated by the fact that we perform worst in cases where we rely on this assumption, i.e. making calls for CN = 3. Finally, because we use training data to fit the mixture models, the procedure is entirely linear. Other procedures, such as CARAT and PLASQ rely on non-linear algorithms that present many practical problems.

We have plans to extend and improve our approach in various ways. First, we plan to implement it for the 500K chips. Second, we believe this approach can be used with Illumina's SNP array and thus plan to try it with data from this platform. Third, we plan to add another level to the model that will permit us to borrow strength across the thousands of SNPs to better estimate the parameters of the conditional probabilities. We plan to use an approach similar to that of CRLMM. Fourth, we plan to look for ways to avoid using the linearity assumption to infer the parameter of conditional distributions when $C > 2$. We plan to use general regression approaches that predict these parameters from the known parameters $C <= 2$. We can train this regression model with Trisomy data ($C = 3$) and design experiments to be able to train for $C > 3$. Fifth, we plan to look for better ways of combining the results form sense and antisense probes. It is desirable to detect and ignore misbehaving strands. Finally, we have observed correlation between parameter estimates coming for proximal locations on the chromosome. This could be due to the fact that various SNPs are on each of the fragments that are amplified. We will explore ways to exploit this finding.

It is possible that the reference set we use has an influence on our results. We plan to study this problem in more detail in the near future. We also plan to substantially increase the size of the reference set to reduce the effect of outlier samples. By combining various publicly available assessment experiments, we plan to develop an comparison protocol for analysis methods. This will help us determine not only which methods work better, but to explore if subsets of the reference set provide better results.

Notice that we did not offer any solutions for the smoothing step as we are more interested in developing techniques for single-point estimates. We expect some of the existing techniques to work well when applied to our estimates of copy number. However, because we explicitly model the conditional probabilities it is possible to develop new methods that impose the across-chromosome correlation through those probabilities instead of the actual copy number estimates.

References

1. G. R. Bignell, J. Huang, J. Greshock, S. Watt, A. Butler, S. West, M. Grigorova, K. W. Jones, W. Wei, M. R. Stratton, P. A. Futreal, B. Weber, M. H. Shapero, and R. Wooster. High-resolution analysis of DNA copy number using oligonucleotide microarrays. *Genome Res*, 14(2):287–95, Feb 2004.

2. B. Carvalho, T. P. Speed, and R. A. Irizarry. Exploration, normalization, and genotype calls of high density oligonucleotide snp array data. *Johns Hopkins University, Dept. of Biostatistics Working Papers*, (111), 2006.
3. F. S. Collins, L. D. Brooks, and A. Chakravarti. A DNA polymorphism discovery resource for research on human genetic variation. *Genome Res*, 8(12):1229–31, Dec 1998.
4. D. F. Conrad, T. D. Andrews, N. P. Carter, M. E. Hurles, and J. K. Pritchard. A high-resolution survey of deletion polymorphism in the human genome. *Nat Genet*, 38(1):75–81, Jan 2006.
5. L. Feuk, A. R. Carson, and S. W. Scherer. Structural variation in the human genome. *Nat Rev Genet*, 7(2):85–97, Feb 2006.
6. S. M. Gribble, D. Kalaitzopoulos, D. C. Burford, E. Prigmore, R. R. Selzer, B. L. Ng, N. S. W. Matthews, K. M. Porter, R. Curley, S. J. Lindasy, J. Baptista, T. A. Richmond, and N. P. Carter. Ultra-high resolution array painting facilitates breakpoint sequencing. *J Med Genet*, Sep 2006.
7. D. A. Hinds, A. P. Kloek, M. Jen, X. Chen, and K. A. Frazer. Common deletions and SNPs are in linkage disequilibrium in the human genome. *Nat Genet*, 38(1):82–5, Jan 2006.
8. J. Huang, W. Wei, J. Chen, J. Zhang, G. Liu, X. Di, R. Mei, S. Ishikawa, H. Aburatani, K. W. Jones, and M. H. Shapero. CARAT: a novel method for allelic detection of DNA copy number changes using high density oligonucleotide arrays. *BMC Bioinformatics*, 7:83, 2006.
9. J. Huang, W. Wei, J. Zhang, G. Liu, G. R. Bignell, M. R. Stratton, P. A. Futreal, R. Wooster, K. W. Jones, and M. H. Shapero. Whole genome DNA copy number changes identified by high density oligonucleotide arrays. *Hum Genomics*, 1(4):287–99, May 2004.
10. W. Huber, A. von Heydebreck, H. Sueltmann, A. Poutska, and M. Vingron. Variance stabilization applied to microarray data calibration and to the quantification of differential expression. *Bioinformatics*, 1, 2002.
11. A. Iafrate, L. Feuk, M. Rivera, M. Listewnik, P. Donahoe, Y. Qi, S. Scherer, and C. Lee. Detection of large-scale variation in the human genome. *Nature Genetics*, 36(9):949–951, 2004.
12. R. Irizarry, F. C. B. Hobbs, Y. Beaxer-Barclay, K. Antonellis, U. Scherf, and T. Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4:249–264, 2003.
13. S. Ishikawa, D. Komura, S. Tsuji, K. Nishimura, S. Yamamoto, B. Panda, J. Huang, M. Fukayama, K. W. Jones, and H. Aburatani. Allelic dosage analysis with genotyping microarrays. *Biochem Biophys Res Commun*, 333(4):1309–14, Aug 2005.
14. G. C. Kennedy, H. Matsuzaki, S. Dong, W. min Liu, J. Huang, G. Liu, X. Su, M. Cao, W. Chen, J. Zhang, W. Liu, G. Yang, X. Di, T. Ryder, Z. He, U. Surti, M. S. Phillips, M. T. Boyce-Jacino, S. P. Fodor, and K. W. Jones. Large-scale genotyping of complex DNA. *Nature Biotechnology*, 21:1233–1237, 2003.
15. D. Komura, K. Nishimura, S. Ishikawa, B. Panda, J. Huang, H. Nakamura, S. Ihara, M. Hirose, K. W. Jones, and H. Aburatani. Noise Reduction from genotyping microarrays using probe level information. *In Silico Biol*, 6(1-2):0009, Feb 2006.
16. T. Laframboise, D. Harrington, and B. A. Weir. PLASQ: A Generalized Linear Model-Based Procedure to Determine Allelic Dosage in Cancer Cells from SNP Array Data. *Biostatistics*, Jun 2006.
17. C. Li and W. Wong. Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection. *Proceedings of the National Academy of Science U S A*, 98:31–36, 2001.

18. S. A. McCarroll, T. N. Hadnott, G. H. Perry, P. C. Sabeti, M. C. Zody, J. C. Barrett, S. Dallaire, S. B. Gabriel, C. Lee, M. J. Daly, D. M. Altshuler, and I. H. Consortium. Common deletion polymorphisms in the human genome. *Nat Genet*, 38(1):86–92, Jan 2006.
19. Y. Nannya, M. Sanada, K. Nakazaki, N. Hosoya, L. Wang, A. Hangaishi, M. Kurokawa, S. Chiba, D. K. Bailey, G. C. Kennedy, and S. Ogawa. A robust algorithm for copy number detection using high-density oligonucleotide single nucleotide polymorphism genotyping arrays. *Cancer Res*, 65(14):6071–9, Jul 2005.
20. D. A. Peiffer, J. M. Le, F. J. Steemers, W. Chang, T. Jenniges, F. Garcia, K. Haden, J. Li, C. A. Shaw, J. Belmont, S. W. Cheung, R. M. Shen, D. L. Barker, and K. L. Gunderson. High-resolution genomic profiling of chromosomal aberrations using Infinium whole-genome genotyping. *Genome Res*, 16(9):1136–48, Sep 2006.
21. N. Rabbee and T. P. Speed. A genotype calling algorithm for affymetrix SNP arrays. *Bioinformatics*, 22(1):7–12, Jan 2006.
22. D. M. Rocke and B. Durbin. A model for measurement error for gene expression arrays. *J Comput Biol*, 8(6):557–569, 2001.
23. J. Sebat, B. Lakshmi, J. Troge, J. Alexander, J. Young, P. Lundin, S. Maner, H. Massa, M. Walker, M. Chi, N. Navin, R. Lucito, J. Healy, J. Hicks, K. Ye, A. Reiner, T. Gilliam, B. Trask, N. Patterson, A. Zetterberg, and M. Wigler. Large-scale copy number polymorphism in the human genome. *Science*, 305:525–528, 2004.
24. A. J. Sharp, S. Hansen, R. R. Selzer, Z. Cheng, R. Regan, J. A. Hurst, H. Stewart, S. M. Price, E. Blair, R. C. Hennekam, C. A. Fitzpatrick, R. Segraves, T. A. Richmond, C. Guiver, D. G. Albertson, D. Pinkel, P. S. Eis, S. Schwartz, S. J. L. Knight, and E. E. Eichler. Discovery of previously unidentified genomic disorders from the duplication architecture of the human genome. *Nat Genet*, 38(9):1038–42, Sep 2006.
25. A. J. Sharp, D. P. Locke, S. D. McGrath, Z. Cheng, J. A. Bailey, R. U. Vallente, L. M. Pertz, R. A. Clark, S. Schwartz, R. Segraves, V. V. Oseroff, D. G. Albertson, D. Pinkel, and E. E. Eichler. Segmental duplications and copy-number variation in the human genome. *Am J Hum Genet*, 77(1):78–88, Jul 2005.
26. E. Tuzun, A. J. Sharp, J. A. Bailey, R. Kaul, V. A. Morrison, L. M. Pertz, E. Haugen, H. Hayden, D. Albertson, D. Pinkel, M. V. Olson, and E. E. Eichler. Fine-scale structural variation of the human genome. *Nat Genet*, 37(7):727–32, Jul 2005.
27. Z. Wu, R. Irizarry, R. Gentleman, F. Martinez-Murillo, and F. Spencer. A model based background adjustement for oligonucleotide expression arrays. *Journal of the America Statistical Association*, 2004.
28. X. Zhao, C. Li, J. G. Paez, K. Chin, P. A. Jeanne, T.-H. Chen, L. Girard, J. Minna, D. Christiani, C. Leo, J. W. Gray, W. R. Sellers, and M. Meyerson. An integrated view of copy number and allelic alterations in the cancer genome using single nucleotide polymorphism arrays. *Cancer Res*, 64(9):3060–71, May 2004.

GIMscan: A New Statistical Method for Analyzing Whole-Genome Array CGH Data

Yanxin Shi[1], Fan Guo[1], Wei Wu[2], and Eric P. Xing[1,⋆]

[1] School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 15213
epxing@cs.cmu.edu

[2] Division of Pulmonary, Allergy, and Critical Care Medicine, University of Pittsburgh, Pittsburgh, PA, 15213

Abstract. Genetic instability represents an important type of biological markers for cancer and many other diseases. Array Comparative Genome Hybridization (aCGH) is a high-throughput cytogenetic technique that can efficiently detect genome-wide genetic instability events such as chromosomal gain, loss, and more complex aneuploidity, collectively known as genome imbalance (GIM). We propose a new statistical method, Genome Imbalance Scanner (GIMscan), for automatically decoding the underlying DNA dosage states from aCGH data. GIMscan captures both the intrinsic (nonrandom) spatial change of genome hybridization intensities, and the prevalent (random) measurement noise during data acquisition; and it simultaneously segments the chromosome and assigns different states to the segmented DNA. We tested the proposed method on both simulated data and real data measured from a colorectal cancer population, and we report competitive or superior performance of GIMscan in comparison with popular extant methods.

1 Introduction

A hallmark of the defective cells in precancerous lesions, transformed tumors, and metastatic tissues, is the abnormality of gene dosage caused by regional or whole chromosomal amplification and deletion in these cells [1]. Cytogenetic and molecular analysis of a wide range of cancers have suggested that amplifications of proto-oncogenes and deletions or loss of heterozygosity (LOH) of tumor suppressor genes can seriously compromise key grow-limiting functions (e.g., cell-cycle checkpoints), cell-death programs (e.g., apoptotic pathways), and self-repair abilities (e.g., DNA repair systems) of injured or transformed cells that are potentially tumorigenetic [2]. Thus DNA copy number aberrations are crucial biological markers for cancer and possibly other diseases. The development of fast and reliable technology for detecting (the presence of) and pinpointing (the location of) such aberrations has become an important subject in biomedical research, with important applications to cancer diagnosis, drug development and molecular therapy.

⋆ Corresponding author.

Array comparative genomic hybridization (array CGH, or, aCGH) assay offers a high-throughput approach to measure the DNA copy numbers across the whole genome [3]. The outcome of an array CGH assay is a collection of log-ratio (LR) values reflecting the relative DNA copy number of test (e.g., tumor cells) versus control (e.g., normal cells) samples at all examined locations in the genome. Ideally, for diploid cells, assuming no copy-number aberration in the control and perfect measurement in the assay, the LRs of clones with k copies in the test sample can be exactly computed. It is noteworthy that among all possible magnitudes of k, usually only a few need to be distinguished, such as 0, 1, 2, 3, and collectively all integers that are greater (often significantly greater) than 3. These are typical copy numbers that reflect distinct cytogenetic mechanisms of chromosome alteration and rearrangements, and hence they are commonly referred to as *gene dosage states*: deletion, loss, normal, gain, and amplification.

Manual annotation of gene dosage tedious and inaccurate due to various reasons, such as impurity of the test sample (e.g., normal cell contaminations), intrinsic inhomogeneity of copy numbers among defective cells, variations of hybridization efficiency, and measurement noises arising from the high-throughput method [4]. Numerous computational methods have been developed for efficient and automated interpretation of array CGH data. Earlier methods used value-windows defined by hard thresholds to determine gene dosage state for each clone based on noisy LR measurement (e.g. [5]). However, these methods suffer from high false positive rate and low coverage (see Sec. 3.1). Recent developments resort to more sophisticated statistical modeling and inference techniques to interpret aCGH data. Based on the underlying statistical assumptions on signal distribution adopted by these methods, they largely fall into four categories: mixture models, regression models, segmentation models and spatial dynamic models. *Mixture models* [6] assume that the LR measurements of all the clones in an aCGH assay are *independent* samples from an underlying distribution consisting of multiple components, each corresponding to a specific gene dosage state. *Regression models* [7,8] try to fit the noisy LRs with a smooth intensity curve over the chromosome to facilitate detection of gene dosage change via visual inspection which are only suitable for data denoising and visualization, rather than explicitly predicting the discrete dosage state underlying the LR signals. *Segmentation models* [9,10,11,12,13,14,15] directly search for breakpoints in sequentially ordered LR signals so that the resulting LR segments have the minimum within-segment signal variations. However, this segmented clone sequences suffer from state "over-representation", in which numerous spurious states without apparent biological meanings are uncovered for the segments. *Spatial dynamic models* solve the problems of dosage-state annotation and clone-sequence segmentation under a unified model for array CGH data. Fridlyand *et al.* [4] proposed a spatial dynamic framework that models the LR sequence as the output of a hidden Markov model (HMM) that governs the distribution of the dosage-states along the chromosome. Marioni *et al.* [16] extended this model by considering the distances between adjacent clones when modeling the transition matrix in HMM. Broet and Richardson [17] developed a Bayesian HMM by allowing the mixture

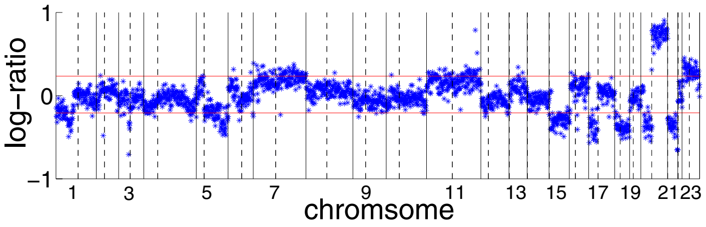


Fig. 1. The LR values (blue dots) of genome X77 from Nakao *et al.* [5]. The solid vertical lines delineate the boundaries between chromosomes; and the dashed vertical lines indicate the position of the centromere of each chromosome. The red horizontal lines indicate the thresholds used by Nakao *et al.* [5] to decide clones dosage states. Clones within these two lines are predicted to be normal state.

weights to be correlated for neighboring genomic sequences on a chromosome. More recently, Shah *et al.* [18] proposed a new Bayesian HMM model that integrates prior knowledge of DNA copy number polymorphisms (CNPs).

This progress notwithstanding, the computational methods for aCGH analysis developed so far are still limited in their accuracy, robustness and flexibility for handling complex aCGH data, and are inadequate for addressing some of the deep biological and experimental issues underlying aCGH assay. Take the whole-genome aCGH data displayed in Fig. 1 as an example. Overall, the LR signals are highly fluctuating, but exhibit visible spatial auto-correlation patterns within the chromosomes. A caveat of the mixture-model-based or threshold-based methods is that they are very sensitive to such random fluctuations of the LR signals because they treat each measurement as an independent sample and ignore spatial relationships among clones. This could lead to highly frequent dosage-state switching (e.g., alternating back and forth between gain and loss, as we will show in our results) within short genetic distances, which is biologically implausible. A number of recent methods, particularly the spatial dynamic models based on HMM, have offered various ways to address this issue, which have significantly improved the performance of computational array CGH analysis.

Nevertheless, a key limitation of the HMM-based methods is that they all assume invariance of the true hybridization signal intensity alone chromosome for each dosage state, which is not always satisfied in real data. As shown in Fig. 1, an outstanding feature of the spatial pattern of the LR signals is that, within each chromosome, there exists both *segmental patterns* that are likely due to change of the copy number of the corresponding region, and *spatial drift* of the overall trend of the LR intensities along the chromosome. For example, in chromosome 4, the LR signals along the sequence of clones are not fluctuating around a baseline (presumably corresponding to a certain dosage state) that is invariant along the chromosome; instead, it is apparent that the baseline itself first has an increasing trend from left to right on 4p and into 4q, and then turns to a decreasing trend along the rest of 4q. Visually, there is not many abrupt breakage points that would signal a dosage-state alteration alone this continuously evolving sequence of LRs. But an HMM approach, which models spatially-dependent choices among different copy-number state, each associated with an invariant distribution of LR values, can fail to capture the spatial drift of LRs over chromosome region with the same copy number as shown in Sec. 3.

Rather than reflecting the discrete change of copy numbers of the clones, the non-random spatial trend of LR signals possibly reflects a continuous change of the biophysical properties and hybridization quality along the chromosome. As discussed in [2], the intensity of the hybridization signal of each clone is affected by a number of factors such as base compositions of different probes, the proportion of repetitive content in sequence, the saturation of array, divergent sequence lengths of the clones, reassociation of double-stranded nucleic acids during hybridization, and the amount of DNA in the array element available for hybridization. These factors may further contribute random or correlated stochasticity of the LR values on top of the content-derived spatial drift. Pinkel and Albertson [2] reported that signal intensity may vary by a factor of 30 or more among array elements even if there are no copy-number changes. These complexities present in real aCGH data render extant models based on fixed state-specific LR distributions, such as an HMM, incapable of making accurate or robust state prediction.

Another problem that affects all the approaches discussed above lies in the calibration of signals across chromosomes and across individuals. As observed from Fig. 1, the mean and the variance of the LRs, and their spatial trends vary significantly from chromosome to chromosome, and more so from individual to individual (not displayed in the Figure), due to reasons possibly beyond copy number differences. This makes measurements from different individuals and/or for different chromosomes difficult to compare. Engler *et al.* [19] recently proposed a parameter sharing scheme for a Gaussian mixture model for genetic variability between and within chromosomes. In the new statistical model for aCGH data presented below, we introduce more careful treatments, which employ different parameter sharing scheme for effects shared among different chromosomes in the same individual (e.g., state baselines) and effects common to the same chromosomes in different individuals (e.g., signal dynamics).

In this paper, we introduce a new method *Genome Imbalance scanner* (GIMscan) for computational analysis of aCGH data. GIMscan employs a more powerful spatial dynamic model, known as switching Kalman filters (SKFs) [20], to jointly capture the spatial-trends of evolving LR signals along chromosomes, and spatially dependent configuration of gene dosage states along chromosomes. Unlike an HMM, which captures all the stochasticities in LRs with invariant dosage-specific distributions, an SKF breaks the accumulation of the stochasticities into two stages: 1) the *hybridization stage*, which involves physical sensory of clone-copies from the digested chromosomes, during which the spatial trend of DNA content and its biophysical properties, saturation effects, etc., can cause stochastic spatial drift of the mass of the hybridized material; 2) the *measurement stage*, which involves acquisition of the readings of fluorescence intensity of each clone, during which errors from reagents, instruments, environment, personal effects, etc., can cause another layer of random noises on top of the hybridization signal. Under the SKF, we model the variations in the hybridization stage using dosage-state-specific continuous dynamic processes, akin to the regression approach discussed above. These hybridization intensities can be understood as the "true" *sensory signals* in an aCGH assay, which are unobservable to the examiner. We refer the sequence of hybridization intensities following such a linear dynamic model as a *hybridization trajectory.* Given the hybridization trajectory, we model the random noise from the measurement stage by a conditional Gaussian distribution whose mean is set by the sensory signal which evolves over each clone according to the trajectory. Overall, for each dosage state, we have a unique linear dynamic model for the sensory signals and a Gaussian emission model for their corresponding noising measurements. This model is known as a Kalman filter. To model changes of dosage-state along the chromosome, we follow the HMM idea to set up a hidden Markov state-transition process, but in our case not over state-specific distributions of LRs with fixed means, but over state-specific Kalman filters over both the observed LR measurements and the unobserved sensory signals for each clone.

On both simulated and experimental aCGH data, GIMscan has shown superior performance over other approaches such as HMM or mixture-model based threshold methods, being able to handle a number of complex LR patterns beyond the recognition power of reference models. We applied our methods to a whole-genome aCGH assay of 125 primary colorectal tumors [5], and constructed a high-quality genome-level gene dosage alteration map for colon cancer.

2 SKF Model and Adaptation to aCGH Analysis

For each specific gene dosage state, we model the spatial drift of its hybridization signal intensities using a hidden trajectory and model the uncertainty in LR measurements using a zero-mean Gaussian noise. This corresponds to a standard dynamic model named Kalman filter (KF). Observed LR values arise as a mixture of the outputs of state-specific Kalman filters. The mixing proportion, modeled as latent variables indicating gene dosage states, is also spatially dependent as captured by a Markov state-transition process (or switching process). Now we have multiple Kalman filters controlled by a dynamic switching process, which can be formulated as a factored switching Kalman filters (SKF). Our proposed method, GIMscan (Genome IMbalance SCANner), adopts the SKF model to whole-genome analysis of aCGH data by allowing a parameter sharing scheme among multiple chromosomes and multiple individuals which makes best use of data. In this section, we first introduce the SKF model and its parameters, then discuss the approximate inference algorithm for joint dosage-state annotation and clone-sequence segmentation. Model selection and further extension of the model are covered briefly at the end of this section.

Figure 2(a) illustrates the Kalman filter for a specific dosage state m, which is a linear chain graphical model with a backbone of hidden real-valued variables (denoted by $X_{1:T}^{(m)}$) emitting a series of real-valued observation (denoted by $Y_{1:T}^{(m)}$). The trajectory of hidden variables is linear and subject to Gaussian

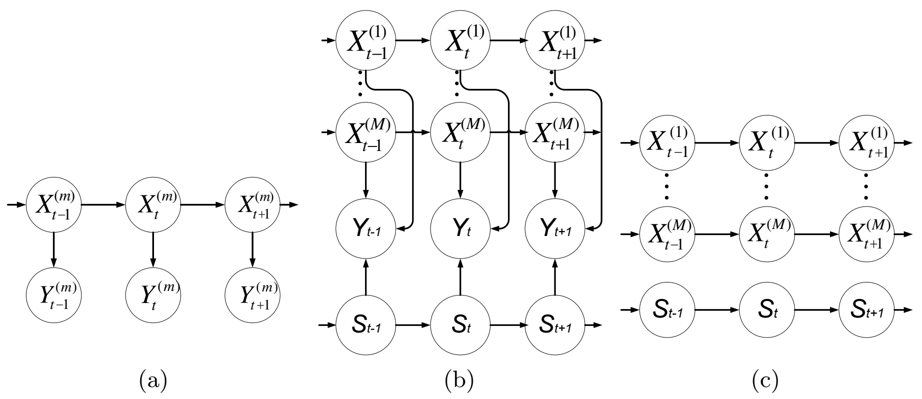


Fig. 2. (a) Graphical structure of dosage-state-specific Kalman filter for dosage state m. $X_t^{(m)}$ is the hidden variable at clone t on the trajectory, and $Y_t^{(m)}$ representing the corresponding observed variable of the Kalman filter. (b) Graphical structure of the switching Kalman filter (SKF) model. The model consists of M linear chains from Kalman filters ($X_{1:T}^{(1:M)}$), a Markov chain of switching processes ($S_{1:T}$) and a series of observed variables ($Y_{1:T}$). (c) Graphical structure of the uncoupled model which represents the tractable subfamily of distributions to approximate the true posterior of the SKF model.

noise which reflects the evolving hybridization signal intensities. The emission model imposes a Gaussian noise arising in the measurement stage on each hidden variable to generate the LR ratio at each position (clone). This model for a specific dosage state m can be formulated as $P(X_t^{(m)}|X_{t-1}^{(m)}) \sim \mathcal{N}(a^{(m)}X_{t-1}^{(m)}, b^{(m)})$, $P(Y_t^{(m)}|X_t^{(m)}) \sim \mathcal{N}(X_t^{(m)}, r)$.

The parameters $a^{(m)}, b^{(m)}, r$ are all position-invariant; r determines the degree of uncertainty in observation measurements. We also assume the initial value of the hidden trajectory, $X_1^{(m)}$, is distributed normally: $P(X_1^{(m)}) \sim \mathcal{N}(\mu^{(m)}, \sigma^{(m)})$. All the variables and parameters are univariate. The computation of posterior distributions of the hidden variables given the observation is tractable because of the conjugacy of the normal distribution to itself. This computation will be part of the inference procedure discussed later in which we decouple the SKF model to a number of tractable linear chains.

Given the dosage-state-specific Kalman filter for M dosage states, a switching Kalman filters generates the LR value at each position from one of the outputs: $Y_t = \sum_{m=1}^{M} Y_t^{(m)} S_t^{(m)}$, where S_t is the M-dimensional multinomial switching variables for clone t following $1 \times M$ binary coding scheme. The discrete switching process $S_{1:T}$ evolves according to Markov dynamics, with initial state distribution parameterized by π and state transition matrix Φ: $S_1 \sim \text{Multinomial}(1, \pi), P(S_t^{(m)} = 1|S_{t-1}^{(n)} = 1) = \phi_{mn}$. We could save the variables $Y_t^{(1:M)}$ and generate the observation directly from the M hidden linear Gaussian trajectories as $P(Y_t|X_t^{(1:M)}, S_t) \sim \mathcal{N}\left(\sum_{m=1}^{M} X_t^{(m)} S_t^{(m)}, r\right)$. The graphical structure of the SKF model is shown in Fig. 2(b).

To facilitate dosage state annotation and clone-sequence segmentation, the posterior distribution $P(S_t|Y_{1:T})$ need to be computed for $t = 1, \ldots, T$. However, exact computation of this posterior probability is intractable. We employed an algorithm [21] which approximates the posterior distribution $\mathcal{P}$ with a parameterized distribution $\mathcal{Q}(\mathbf{v})$ from some tractable subfamily of distributions. It iteratively updates the values of variational parameters $\mathbf{v}$ to minimize the KL divergence between the approximate posterior distribution and the true posterior distribution. The choice of the tractable subfamily for the SKF model is a discrete Markov chain and M uncoupled KFs (Fig. 2(c)). Two sets of variational parameters are introduced for the Markov chain and KFs respectively. Their updates can be carried out using fix-point equations [21], which maintain or increase a lower bound of log likelihood of the model and usually converge in a few iterations. Fast rate of convergence is mainly due to low data dimension.

Parameter estimation is performed under the EM framework. The E step employs the variational inference algorithm to find the best approximate posterior via iterative updates of the variational parameters. The M step reestimates the model parameters Θ to maximize the same lower bound of log-likelihood in variational inference. This reestimation can be performed exactly by zeroing the derivatives with respect to the model parameters. Parameter estimation is implemented by a coordinate ascent procedure.

Now we have introduced the SKF model and its parameters $\mu^{(m)}, \sigma^{(m)}, a^{(m)}$, $b^{(m)}, r, \pi$ and ϕ, each of which delineates one property behind the aCGH data. $\mu^{(m)}$ and $\sigma^{(m)}$ are the mean and variance of Gaussian distribution of the starting clone on hidden trajectory for dosage state m. $a^{(m)}$ and $b^{(m)}$ determine the transition model of that trajectory which dictate the spatial drift of the signal intensities. r is the variance of the Gaussian accounting for the noise introduced in the experiment stage, and is independent of the hidden dosage-state. Lastly, π and Φ are initial state parameters and transition matrix for the discrete switching process between different dosage states.

In the settings for a whole-genome analysis, the aCGH dataset are collected from experimental data of J individuals, the genome of which consists of K chromosomes, and chromosome k contains T_k clones. The LR values $Y_{1:T,j,k}$ on individual j, chromosome k are generated by an SKF model with hidden trajectory $X_{1:T,j,k}$ and switching states $S_{1:T,j,k}$.

We are now ready to describe the parameter sharing scheme in GIMscan for the analysis of whole genome aCGH data. We consider two groups of parameters. Firstly, we let $\mu^{(m)}$, $\sigma^{(m)}$, r, π and Φ be shared across all chromosomes of one particular individuals. Mainly due to the normal cell contamination, the magnitude of starting value for the trajectory of one particular state varies across different individuals. Different $\mu^{(m)}$ and $\sigma^{(m)}$ for different individuals can account for this "un-normalized" starting value of the trajectory of one state. r is also shared by chromosomes from one individual because one individual corresponds to one experiment, and different experiments may have different noise levels. π and Φ are shared by one individual because the number of dosage states are individual-specific: different individuals may have different number of dosage states. The second group of parameters, $a^{(1:M)}$ and $b^{(1:M)}$, is assumed to be shared by one particular chromosome of all individuals, because the physical-chemical properties (e.g. the base composition) of one particular chromosome of different individuals (the same tumor cell line of same species) are very similar. This similarity leads to similar hybridization signal intensity over a chromosome.

The maximum number of dosage states M one individual can have remains to be determined. we employ Gaussian mixture model using penalized likelihood criteria such as AIC to select the number of states for each individual.

3 Experiments and Results

We tested the performance of GIMscan on simulated aCGH data and real data with complex aCGH patterns to demonstrate the working principle and general trends of our method in gene dosage prediction, and to evaluate our prediction quality under nontrivial genome imbalance and hybridization scenarios. The benefit of applying a sophisticated hybrid stochastic model to capture both discrete (e.g., changing DNA copy number) and continuous (e.g., varying hybridization efficiency) latent spatial trajectory underlying noisy aCGH measurements is evidenced in each level of genetic scales we have analyzed.

3.1 Simulated aCGH Data

We first validate GIMscan on simulated aCGH datasets, which mimic typical spatial patterns of LR sequences in real aCGH assays, and allow a quantitative assessment of model performance based on known underlying gene dosage states in the simulation.

In our simulation experiments, three methods—threshold, HMM as in [4], and GIMscan—were tested on 12 datasets simulated with different settings of two parameters, the Gaussian emission variance r and the KF transitional variance b (see Section 2). These two parameters represent the two sources of the overall noise in the data: r reflects the quality the LR measurements in an aCGH experiment, whereas b reflects the variability of the hybridization signal intensity along the chromosome. Our datasets correspond to three different values of r, ranging from low, to medium, high; and four values of b also spanning a significant range (see Fig. 3). For each combination of r and b, a total of 100 LR sequences each containing 100 clones were generated. For each sequence, we simulated a random 5×5 stochastic matrix, T, for modeling transitions between gene dosage states, and T was set to allow both short and long stretch of gene dosage alterations, but not high-frequency oscillations between different states. All three methods were applied to each dataset to infer the gene dosage states underlying the simulated LRs, and the experiments were repeated 100 times. Fig. 3 summarizes the medians, quantiles and ranges of the prediction error rates by different methods under various parameter settings. Consistently, GIMscan outperformed the other two methods by a significant margin.

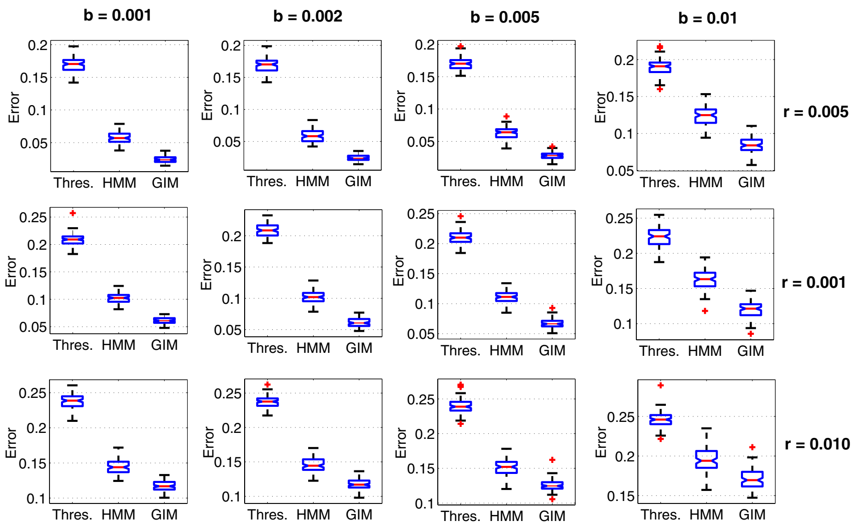


Fig. 3. Performance of gene dosage state prediction on simulated aCGH datasets. Each row corresponds to an emission noise, and each column corresponds to a Kalman filter transitional variance. In each case, we plot the results of threshold method (Thres.), HMM and GIMscan (GIM). The red line represents the median, and the blue box indicates upper and lower quantiles. The black bars are the range of the error rate. Outliers are plotted by "+".

As an illustration of the advantage offered by the SKF model adopted by GIMscan, and the effectiveness of our inference algorithm, Fig. 4 and Fig. 5 show two examples of GIMscan's performance in the simulated datasets. The first example concerns "high-quality" aCGH records simulated with low measurement noise ($r = 0.001$) over 100 clones switching between two gene dosage states both with low spatial drift in their corresponding true hybridization intensities ($b = 0.001$) (Fig. 4(a)). Figure 4(b) presents the inferred gene dosage state and the inferred dosage-state-specific "trajectories" (i.e., the latent dynamical trend captured by each KF) of the latent true hybridization intensities underlying the observed LR sequence shown in Fig. 4(a). As shown in this illustration, each inferred latent trajectory indeed represents a smoothed and spatially changing baseline of the LR signals corresponding to a particular dosage state; and all inferred trajectories agree well with the true trajectories of hybridization intensities used for simulating the observed LR signals. As a result, the inferred switching process over these trajectories gives a highly accurate prediction of the gene dosage states underlying the LR sequence. GIMscan can also estimate the confidence intervals (i.e., standard deviation) of the inferred hybridization trajectories, as shown in Fig. 4(c).

Another example shown in Fig. 5 concerns low-quality, arguably more realistic aCGH records simulated with high measurement noise ($r = 0.01$) and severer spatial change ($b = 0.01$) in the true hybridization trajectories. The combined

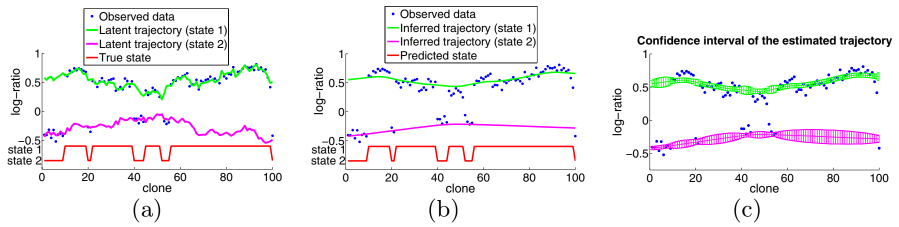


Fig. 4. (a) Simulated data (blue dots), two latent trajectory (green and pink), and switching process (red). The length of the simulated data is 100 clones. (b) Simulated data (blue dots), inferred trajectory (green and pink) and inferred switching process (red). (c) The confidence intervals of the inferred trajectories.

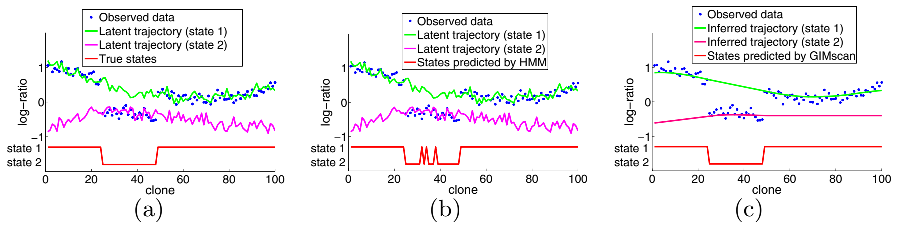


Fig. 5. (a) Simulated data (blue dots), two latent trajectories (green and pink), and switching process (red). The length of the simulated data is 100 clones. (b) Given the observed data in (a), the red line is the state predicted by HMM. (c) Given the observed data in (a), the green and pink curves are trajectories inferred by SKF, and the red line is the switching process inferred by SKF.

effects of high measurement noise and high spatial variance of the hybridization trajectories are excepted to lead to misassignment of gene dosage state due to inaccurate estimation of the dosage-state-specific hybridization intensities when spatial trajectory of the hybridization intensities is ignored. Note that the trajectories in Fig. 5(a) of both dosage states are not flat, which reflect severe spatial drift of hybridization signal intensity within each state. When assuming spatial invariance of dosage-state-specific signal distribution, the unflatness of both trajectories can cause the estimated mean of LR signals to be highly biased (e.g., higher for state 1, and lower for state 2), and their variances to be significantly greater than the actual fluctuation. Consequently, the estimated dosage-specific signal distributions can be seriously overlapping, causing the LR signals from two states hard to distinguish. Fig. 5(b) shows exactly this effect, on the quality of state estimation by an HMM model. Whereas the SKF model underlying GIMscan readily mitigates this effect, and produces the correct estimation.

3.2 Real aCGH Data with Diverse Spatial Patterns

Now we present case studies of selected real aCGH data with a diverse spectrum of spatial patterns. Our dataset was obtained from an online repository of whole-genome aCGH profiles of 125 colorectal tumors originally studied in [5]. This dataset was found to contain highly stochastic LR measurements with severe spatial variance and drifts along the chromosomes, and bear rich cohorts of genome imbalance patterns. Such complications present a great challenge to naive algorithms for gene dosage inference, and are thus particularly suitable for evaluating our proposed method.

Given an aCGH profile, GIMscan first employs a k-nearest neighbor regression procedure (e.g., $k = 3$) to impute the missing values in the LR records. Then it fits the processed data with a Gaussian mixture based on maximum likelihood estimation, and performs model selection based on AIC to determine the total number of gene dosage states, M (which is constrained between 1 to 5), for each individual. Afterwards, the number of component KFs (i.e., dosage-state-specific hybridization trajectories) in GIMscan is set to be M, and the mean of the starting clone of each KF takes on the mean of a component in the estimated Gaussian mixture as initial value. Note that with this setup, we still need to establish the exact mapping between the KFs inferred by GIMscan and the possible gene dosage states, namely deletion, loss, normal, gain, and amplification. Since GIMscan provides estimations of the hybridization trajectories of each KF, we follow a straightforward statistical and biological argument and determine the corresponding dosage-state of each trajectory based on the relative mean-values of the estimated true hybridization intensities of all clones.

For comparison, we re-implemented the HMM methods according to Fridlyand *et al.* [4], with modest extension (i.e., parameter sharing) so that it can be applied to whole genome CGH profiles covering multiple chromosomes. Following [4] AIC is also used for model selection for the HMM.

The dataset we studied contains a total of $\sim 2.75 \times 10^5$ LR measurements from 23×125 chromosomes (i.e., 125 human genomes). Here we first present a small-scale case study of three representative chromosomes, each containing a typical spatial pattern for the LR sequence that was found to be difficult to analyze by conventional methods. For convenience, we refer to these patterns as, flat-arch, step, and spike, respectively, according to their shapes in the LR intensity plots (Fig. 6).

Pattern I: Flat-Arch Figure 6(a)(b) displays the LR measurements from chromosome 4 of individual X77, this pattern is marked by lower magnitudes of LRs at the two telomere regions of the chromosome and elevated magnitudes in the central region. Locally (i.e., along the plotted chromosomal region), there is a continuous trend of spatially evolving hybridization intensity along the chromosome, and there are few abrupt breakage points that would signal a dosage-state alteration. But due to the high dispersion of LR values as a result of such a spatial drift, methods based on invariant state-specific hybridization intensity, such as the HMM, would either fit the observed LR values with one biased and high-variance Gaussian distribution, or split the LRs with two highly overlapping Gaussians. These caveats could seriously compromise the quality of gene dosage state estimation. Figure 6(a) shows the dosage estimation by an HMM fitted on this chromosome. The outcome suggests heavy oscillations between two dosage

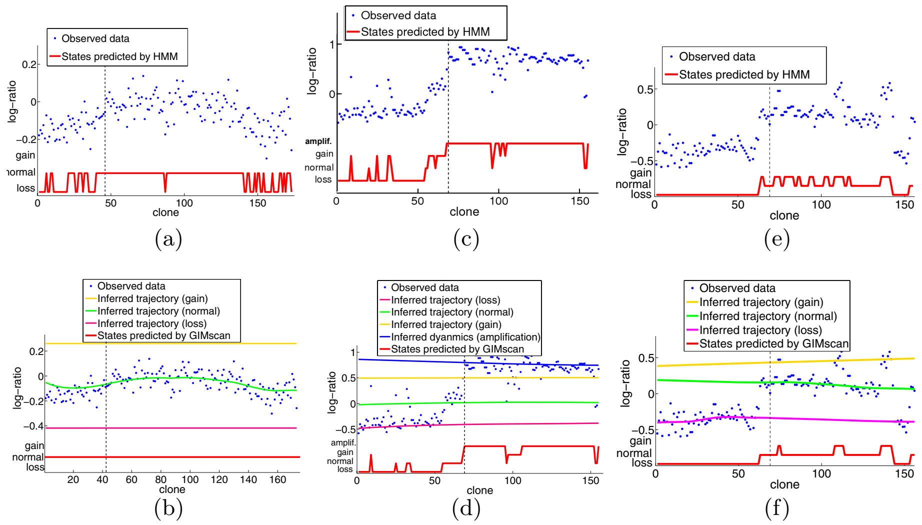


Fig. 6. Three typical spatial patterns for the LR sequence which were found to be difficult to analyze by convenctional methods: (a)(b)Flat-Arch pattern; (c)(d)Step pattern; (e)(f) Spikes pattern. (a)(c)(e) shows states predicted by HMM (red). In (b)(d)(f) the pink, green and yellow curves are inferred trajectories for loss, normal and gain, respectively. Red solid line indicates states predicted by GIMscan. Centromere position is indicated by dashed vertical line in these two plots.

states throughout the chromosome, which is biologically implausible. Figure 6(b) shows the dosage state sequence and dosage-state-specific trajectories underlying chromosome 4 of individual X77 inferred by GIMscan. A whole-genome fitting resulted in three estimated dosage-states. On this particular chromosome, the trajectories of the loss and gain states (the pink and yellow curves, respectively) were not matched to any observations, and the entire region is determined to be corresponding to a normal state whose hybridization intensity varies along the chromosome (the green curve). Indeed, a more global visual inspection of these Flat-Arch patterns in the context of whole aCGH profile often reveals that the flat-arch shape in the LR-plots often merely reflects modest (but spatially correlated) change of the LR magnitude most likely within a single dosage state.

Pattern II: Step This pattern is typical when there appears to be a quantum change of LR magnitudes from one to the other end of the chromosome, but the boundary of the change is not sharp and the overall sequence is moderately noisy, such as shown in Fig. 6(c)(d) which is taken from chromosome 8 from individual X265. In addition to the step, this sample also harbors a number of local spikes and short regions potentially implying dosage-state alterations. Via AIC model-selection, the HMM adopted four dosage state when processing this data. The states predicted by HMM are shown in Fig. 6(c). As can be seen, the results are reasonable, except that several positions near clone 100 contains highly frequent switching between states. The dosage state sequence and dosage-state-specific trajectories inferred by GIMscan are shown in Fig. 6(d). Note that there is a slightly decreasing trend in the trajectory corresponding to the amplification state. While the gain and normal trajectories correspond to only a few clones on this chromosome, a genome-level parameter sharing scheme adopted by GIMscan enables them to be reliably estimated, and thereby leads to plausible prediction of point changes on isolated clones (e.g., clone 97 and 152).

Pattern III: Spikes Spikes are a typical pattern often accompany other patterns, such as steps. It is marked by short sequences, sometimes singletons, of elevated or attenuated LR measurements along the chromosomes. Figure 6(e)(f) shows such an example from chromosome 8 of individual X318. In this chromosome, the copy-number loss was apparent on 8p arm, while three spikes (around clone 75, 110 and 140) were visible on 8q arm. These spikes correspond to the gain state with a large measurement variance. Figure 6(e) shows the states predicted by HMM. Although HMM correctly predicted the states on 8p, it predicted more clones on 8q to be gain state. However, by our visual check, some clones (e.g. around clone 79, 96) should have been classified to be normal state. The possibly faulty predictions of gain states resulted from the large variance of the spikes estimated by the HMM. GIMscan correctly detected and annotated the spikes, as well as giving convincing predictions on other clones (Fig. 6(f)). Compared to the case for the same chromosome (i.e., no. 8) from individual X265 shown in Fig. 6(c)(d), where four dosage-state-specific trajectories were determined, here we uncovered only three states for chromosome 8. This is because model selection for SKF in this individual based on the whole-genome aCGH only identifies three states—normal, loss and gain. Parameter-sharing was adopted by GIMscan for all chromosomes in this individual, and leads to three common trajectories. Comparing Fig. 6(d) with Fig. 6(f), one can notice that the elevates of the trajectories corresponding to the same dosage state (e.g., normal) can be quite different across individual, which is likely due to some unidentified systematic error or hybridization-efficiency difference across individuals. The parameter-sharing scheme adopted by GIMscan (i.e., sharing dosage-state-specific trajectories across chromosomes within individual, but not across individual) provides a reasonable strategy to tackle such variations.

We finally used GIMscan for populational analysis of Nakao *et al.*'s [5] dataset. Overall, over the 125 genomes each examined at $\sim$2200 clones uniformly distributed in the genome, on average each genome have 19.18% (or 407) of the clones suffered either gain or loss (9.25% and 9.94%, respectively), and another 1.33% of the clones were hit by amplification or deletion (0.93% and 0.4%, respectively). The whole-genome spatial spectrum of GIM rates over the entire study population is displayed in Fig. 7. As can be seen, the population rates of gain and amplification of clones in chromosome 7, 8q, 13q, 20q and 23 were significantly higher than those of the other regions, suggesting possible presence of proto-oncogenes in these regions. Likewise, the population rates of loss and

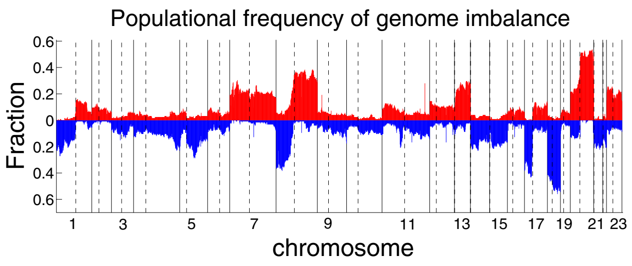


Fig. 7. Overall frequency of DNA dosage state alteration over entire genomes of 125 individuals. Blue bars represent clones with DNA copy number loss or deletion, whereas red bars for DNA copy number gain or amplification. Solid vertical lines show boundaries between chromosomes; dashed vertical lines show centromeres of chromosomes.

deletion in chromosome 1p, the distal-end of 4q, 5q, 8p, 14, 15, 17p, 18, and 21, were significantly higher than those of the other regions, suggesting possible presence of tumor suppressor genes in these regions.

4 Discussion

An important issue for the success of GIMscan is the parameters initialization. Our experience with GIMscan shows that the initial values for π and ϕ may be fairly arbitrary, while the initial values for μ and r are more essential. We can employ the Gaussian mixture to cluster the data points into M clusters. The mean LR value of one cluster is used as the initial value for the starting mean of the corresponding trajectory. The initial value of r can be determined similarly. We initialized a and b with some constants: a was fixed to 1 and b was fixed to 10^{-2}. σ was given the same initial value as r.

Acknowledgments. This material is based upon work supported by the National Science Foundation under Grant No. 0523757, and and by the Pennsylvania Department of Health's Health Research Program under Grant No. 2001NF-Cancer Health Research Grant ME-01-739. E.P.X. is also supported by a NSF CAREER Award under Grant No. DBI-0546594.

References

1. Diep, C.B. *et al.*: Genome characteristics of primary carcinomas, local recurrences, carcinomatoses, and liver metastases from colorectal cancer patients. Mol. Cancer. **3(1)** (2004) 6
2. Pinkel, D., Albertson, D.G.: Array comparative genomic hybridization and its applications in cancer. Nat. Genet. **37** (2005) S11–S17
3. Pinkel, D. *et al.*: High resolution analysis of DNA copy number variation using comparative genomic hybridization to microarrays. Nat. Genet. **20** (1998) 207–211

4. Fridlyand, J., Snijders, A.M., Pinkel, D., Albertson, D.G., Jain, A.N.: Hidden Markov models approach to the analysis of array CGH data. J. Multivariate Anal. **90(1)** (2004) 132–153
5. Nakao, K. *et al.*: High-resolution analysis of DNA copy number alterations in colorectal cancer by array-based comparative genomic hybridization. Carcinogenesis. **25(8)** (2004) 1345–1357
6. Hodgson, G. *et al.*: Genome scanning with array CGH delineates regional alterations in mouse islet carcinomas. Nat. Genet. **29** (2001) 459–464
7. Eilers, P., De Menezes, R.: Quantile smoothing of array CGH data. Bioinformatics. **21(7)** (2005) 1146–1153
8. Hsu, L., Self, S., Grove, D., Randolph, T., Wang, K., Delrow, J., Loo, L., Porter, P.: Denoising array-based comparative genomic hybridization data using wavelets. Biostatistics. **6(2)** (2005) 211–226
9. Myers, C.L., Dunham, M.J., Kung, S.Y., Troyanskaya, O.G.: Accurate detection of aneuploidies in array CGH and gene expression microarray data. Bioinformatics. **20(18)** (2004) 3533–3543
10. Olshen, A.B., Venkatraman, E.S., Lucito, R., Wigler, M.: Circular binary segmentation for the analysis of array based DNA copy number data. Biostatistics. **5(4)** (2004) 557–572
11. Jong, K., Marchiori, E., Meijer, G., Vaart, A.V., Ylstra, B.: Breakpoint identification and smoothing of array comparative genomic hybridization data. Bioinformatics. **20(18)** (2004) 3636–3637
12. Picard, F., Robin, S., Lavielle, M., Vaisse, C., Daudin, J.J.: A statistical approach for array CGH data analysis. BMC Bioinformatics. **6(1)** (2005) 27.
13. Hupe, P., Stransky, N., Thiery, J.P., Radvanyi, F., Barillot, E.: Analysis of array CGH data: from signal ratio to gain and loss of DNA regions. Bioinformatics. **20(18)** (2004) 3413–3422
14. Wang, P., Kim, Y., Pollack, J., Narasimhan, B., Tibshirani, R.: A method for calling gains and losses in array CGH data. Biostatistics. **6(1)** (2005) 45–58
15. Daruwala, R.S., Rudra, A., Ostrer, H., Lucito, R., Wigler, M., Mishra, B.: A versatile statistical analysis algorithm to detect genome copy number variation. PNAS. **101** (2004) 16292–16297
16. Marioni, J.C., Thorne, N.P., Tavare, S.: BioHMM: a heterogeneous hidden Markov model for segmenting array CGH data. Bioinformatics. **22(9)** (2006) 1144–1146
17. Broet, P., Richardson, S.: Detection of gene copy number changes in CGH microarrays using a spatially correlated mixture model. Bioinformatics. **22(8)** (2006) 911–918
18. Shah, S.P., Xuan, X., De Leeuw, R., Khojasteh, M., Lam, W., Ng, R., Murphy, K.P.: Integrating copy number polymorphisms into array CGH analysis using a robust HMM. Bioinformatics. **22(14)** (2006) e431–e439
19. Engler, D.A., Mohapatra, G., Louis, D.N., Betensky, R.A.: A pseudolikelihood approach for simultaneous analysis of array comparative genomic hybridizations. Biostatistics. **7(3)** (2006) 399–421
20. Murphy, K.P.: Learning switching Kalman filter models. Compaq Cambridge Research Lab Tech Report 98-10. (1998)
21. Ghahramani, Z., Hinton, G.E.: Variational learning for switching state-space models. Neural Comput. **12(4)** (1998) 963–996

Production-Passage-Time Approximation: A New Approximation Method to Accelerate the Simulation Process of Enzymatic Reactions*

Hiroyuki Kuwahara[1] and Chris Myers[2]

[1] School of Computing, University of Utah
Salt Lake City, UT 84112, U.S.A.
kuwahara@cs.utah.edu

[2] Department of Electrical and Computer Engineering, University of Utah
Salt Lake City, UT 84112, U.S.A.
myers@ece.utah.edu

Abstract. Given the substantial computational requirements of stochastic simulation, approximation is essential for efficient analysis of any realistic biochemical system. This paper introduces a new approximation method to reduce the computational cost of stochastic simulations of an enzymatic reaction scheme which in biochemical systems often includes rapidly changing fast reactions with enzyme and enzyme-substrate complex molecules present in very small counts. Our new method removes the substrate dissociation reaction by approximating the passage time of the formation of each enzyme-substrate complex molecule which is destined to a production reaction. This approach skips the firings of unimportant yet expensive reaction events, resulting in a substantial acceleration in the stochastic simulations of enzymatic reactions. Additionally, since all the parameters used in our new approach can be derived by the Michaelis-Menten parameters which can actually be measured from experimental data, applications of this approximation can be practical even without having full knowledge of the underlying enzymatic reaction. Furthermore, since our approach does not require a customized simulation procedure for enzymatic reactions, it allows biochemical systems that include such reactions to still take advantage of standard stochastic simulation tools. Here, we apply this new method to various enzymatic reaction systems, resulting in a speedup of orders of magnitude in temporal behavior analysis without any significant loss in accuracy.

1 Introduction

This paper considers a well-stirred chemically reacting system with the following enzymatic reaction scheme:

$$\mathrm{E} + \mathrm{S} \underset{k_{-1}}{\overset{k_1}{\rightleftarrows}} \mathrm{C} \xrightarrow{k_2} \mathrm{E} + \mathrm{P} \tag{1}$$

* This material is based upon work supported by the National Science Foundation under Grant No. 0331270.

where E, S, C, and P represent an enzyme, a substrate, an enzyme-substrate complex, and a product, respectively, and k_1, k_{-1}, and k_2 represent non-zero rate constants for the reaction channels, R_1, R_{-1} and R_2, respectively. This enzymatic reaction scheme specifies the transformation of S into P catalyzed by E where E has one active site to which S can bind to form C. These types of enzymatic reactions can be found in many biochemical pathways such as metabolic pathways, and therefore, abstracting away low-level details found in enzymatic reaction schemes may have a significant computational benefits.

Traditionally, biochemical systems—including the enzymatic reaction system that is considered in this paper—are modeled and analyzed typically within the continuous-deterministic, *classical chemical kinetics* (CCK) framework based on the *law of mass action* where the dynamics of a well-stirred system is described by a set of ordinary differential equations (ODEs). However, the limitations of the CCK analysis have been broadly accepted [1,2,3,4]. In particular, given the same initial condition, the CCK analysis of biochemical systems always produces the same results as it neglects fluctuations. Such treatment, nevertheless, can be justified when the molecular populations are very large, and hence a CCK analysis may provide the most efficient approach to determine the time evolution of a system in such cases. However, many regulatory components in biological systems are often present in amounts too small to simply neglect the effects of inherent fluctuations [5,6,7,8]. Moreover, if a system being analyzed has multiple steady states, the traditional ODE approach may not be able to provide an accurate time evolution of a system since it cannot capture spontaneous transitions between steady states [9,10].

In order to more accurately predict the temporal behavior of biochemical systems without acquiring more information on a biological system such as the positions and the velocities of every molecule, the *stochastic chemical kinetics* (SCK) framework can be used [11]. SCK describes the time evolution of a well-stirred biochemical system as a discrete-state jump Markov process that is analytically governed by the *chemical master equation* (CME) [12]. The CME is derived from the *state-change vector*, specifying the change in each molecular species population for each reaction, and a *propensity function* for each reaction. For example, the enzymatic reaction scheme (1) contains the following propensity functions for each reaction R_i:

$$\mathrm{R}_1 : a_1(\mathbf{x}) = k_1 x_E x_S, \;\; \mathrm{R}_{-1} : a_{-1}(\mathbf{x}) = k_{-1} x_C, \;\; \mathrm{R}_2 : a_2(\mathbf{x}) = k_2 x_C$$

where $\mathbf{x} = (x_E, x_S, x_C, x_P)$, and each x_* is the value of random variable $X_*(t)$ representing the molecular population of the species subscripted. Thus, the vector of these random variables: $\mathbf{X}(t) = (X_E(t), X_S(t), X_C(t), X_P(t))$ represents the system state at time t. Assuming that the system is spatially homogeneous, this SCK approach describes the time evolution of a biochemical system at the individual reaction level by exactly tracking the quantities of each molecular species and by treating each reaction as a separate random event. However, directly obtaining the solution of the CME of any realistic system, either analytically or numerically, is not feasible due to its intrinsic complexity. Note that, though it is possible to numerically solve the CME of the enzymatic reaction scheme (1) as the system state is bounded albeit with potentially substantial computational demands, if systems also contain other reactions and species as is the case for many realistic biological systems, then the space complexity of CMEs of such systems often inevitably becomes too large to be tractable, making the numerical solutions of such CMEs infeasible. Consequently, the time evolution of moments $\langle \mathbf{X}^n(t) \rangle$ is also generally infeasible to compute from the CME.

To overcome this, several methods have been introduced to approximate the time evolution of moments of the process $\mathbf{X}(t)$ without solving the CME [9,13]. Such approximations are very useful to efficiently understand the mean behavior, standard deviation, skewness, etc. of $\mathbf{X}(t)$, as well as to potentially characterize the time evolution of the asymptotic probability distribution of the system states. However, utilizing such methods alone may encounter difficulties in quantitative analyses of some biologically relevant properties based on stochastic competition such as probabilistic analysis of lysis/lysogeny developmental pathways in bacteriophage λ-infected *Escherichia coli* [1]. Furthermore, since the complexity of the moment evolution equations may significantly increase as that of a system increases [9], such approach may be unwieldy for a large-scale biological system.

Instead of attempting to solve the CME, exact discrete-stochastic numerical realizations of a system via Gillespie's *stochastic simulation algorithm* (SSA) [14], which is derived from the same premise as the CME, are often used to infer the temporal system behavior with a much smaller memory footprint. This Monte Carlo simulation approach is useful to intuitively observe the trend of system dynamics, which may be possible with as few as tens of numerical realizations. Furthermore, *in silico* experiments via Monte Carlo simulation come with potentially unlimited controlling capabilities and abilities to capture virtually any dynamical properties of the system, making a number of qualitative and quantitative analyses which cannot be done in *wet-lab* experiments possible. Unfortunately, the computational requirements of the SSA—even with the Gibson and Bruck optimization [15], which, among other things, reduces the generations of the random numbers by reusing them—can be substantial. This is due largely to the fact that it not only requires a potentially large number of simulation runs in order to estimate the system behavior at a reasonable degree of statistical confidence, but it also requires every single reaction event to be simulated one at a time. For example, if $k_2 \ll k_{-1}$ in the enzymatic reaction scheme (1), then C dissociates into S much more often than into P, and thus, much of the computation time is allocated for this *substrate-complex loop.*

Several approximation methods have been proposed to accelerate the simulation process of the SSA by sacrificing exactness. For example, the explicit τ-leaping method approximates the number of firings of each reaction in a predefined interval rather than executing each reaction individually [16]. While this and similar methods [17,18,19] are very promising, they may not perform well for an enzymatic reaction which bears rapidly changing fast reactions driven by the enzyme and enzyme-substrate complex molecules present in very small counts because the *leaping condition* may not be satisfied in such situations.

Some acceleration methods for the stochastic simulations of enzymatic reactions have been proposed that perform well even when the enzyme is present in very small amounts by eliminating the undesirable substrate-complex loop in the enzymatic reaction scheme. For example, Rao and Arkin have performed model abstraction by using biochemical insight in combination with the *quasi-steady-state approximation* (QSSA) to remove the expensive substrate-complex loop, and then applied a modified version of SSA to the simplified model [20]. Cao et al. [21] have demonstrated how the substrate-complex loop can be removed by applying the enzyme substrate reaction system to their slow-scale SSA approach which explicitly simulates the firings of only the slow reaction events [22]. Both approximation methods require the use of special simulation procedures. Thus, there might be cases where one finds the use of these approximation methods inconvenient when it comes to the analysis of a system containing enzymatic reactions along with other types of reactions. Such cases occur, for example, when a biochemical system is represented in the *Systems Biology Markup Language* (SBML), the emerging standard format to represent models of biochemical reaction networks [23]. SBML level 2 version 1 contains reactions only in the generic type, and it cannot specify any specific reaction types without using proprietary annotations. Thus, in order for SBML compliant SSA tools to know when to use a specially tailored Monte Carlo simulation procedure for enzymatic reactions, the tools must either understand the semantics of proprietary fields that specify enzymatic reactions or perform structural analysis to find enzymatic reactions.

This paper introduces a new approximation approach to accelerate the process of stochastic simulations of enzymatic reactions. Our new approach, which we call *production-passage-time approximation* (PPTA), approximates the passage time of the complex C which is destined to turn into the product P, and only tracks such instances of C. Thus, this approach eliminates the substrate-complex loop by removing R_{-1}, allowing a substantial acceleration in stochastic simulations of enzymatic reactions. Furthermore, since our approach does not require a customized simulation procedure for enzymatic reactions, it allows a biochemical system comprising the PPTA reactions along with other types of reactions to still be modeled using a SBML modeling tool such as PathwayBuilder from BioSPICE [24], and analyzed by using any SBML compliant SSA tools.

This paper describes the PPTA method in Section 2. Section 3 demonstrates how our approach can help analyze the temporal behaviors of enzymatic one-substrate reaction models efficiently with reasonable accuracy. This is shown by applying our new approximation method to various systems and comparing the full models with the corresponding PPTA models in terms of the accuracy—by calculating means and standard deviations—as well as runtime. Finally, this paper concludes in Section 4 by discussing the benefits gained by the PPTA.

2 Production-Passage-Time Approximation

To describe the PPTA method, the enzymatic reaction scheme (1) is first considered to have the initial condition: $\mathbf{X}(t_0) = \mathbf{x_{t_0}}$, where $\mathbf{x_{t_0}} = (e_{tot}, s_{tot}, 0, 0)$, $e_{tot} \geq 1$, and $s_{tot} \geq 1$. Let $\mathbf{x}_\infty = (e_{tot}, 0, 0, s_{tot})$, then the probability that

$\mathbf{X}(t) = \mathbf{x}_\infty$ given $\mathbf{X}(t_0) = \mathbf{x_{t_0}}$ approaches 1, as $t \to \infty$. In other words, in any simulation runs, the enzymatic reaction process always reaches $\mathbf{x}_\infty$ eventually. In order for each numerical realization of $\mathbf{X}(t)$ to transition from $\mathbf{x_{t_0}}$ to $\mathbf{x}_\infty$, S must be transformed into C at least s_{tot} times and C must be converted into P exactly s_{tot} times. Thus, let $\mathbf{x}^{(i)}(t)$ be the i-th sample trajectory of $\mathbf{X}(t)$ given that $\mathbf{X}(t_0) = \mathbf{x_{t_0}}$ and $\mathbf{T_i}$ be a set of time instances such that each time instance t^i_j represents the time point where the j-th reaction event occurs in $\mathbf{x}^{(i)}(t)$. Then, the statement $\forall i.\ |\mathbf{T_i}| \in [2s_{tot}, \infty)$ must be true. Intuitively, if $k_{-1} \ll k_2$, then C tends to be consumed by R_2 rather than R_{-1}, making the size of each $\mathbf{T_i}$ close to the lower bound $2s_{tot}$. On the other hand, if $k_{-1} \gg k_2$, then C is more likely to be consumed by R_{-1}, and in consequence each $|\mathbf{T_i}|$ is very likely much greater than $2s_{tot}$, making the computational cost of simulations significantly higher.

Our new PPTA approach minimizes the number of reaction events that fire through the passage of each $\mathbf{x}^{(i)}(t)$ to $\mathbf{x}_\infty$ by preventing each $\mathbf{x}^{(i)}(t)$ from revisiting the same state. Thus, it guarantees that $\forall i.\ |\mathbf{T_i}| = 2s_{tot}$. This is achieved by eliminating R_{-1} and approximating transitions of each $\mathbf{x}^{(i)}(t)$ using only complex-formation and production reactions. In other words, the PPTA approximates the passage time of the formation of each C molecule which leads to a production of P, and only keeps track of such instances of formation of C, rather than explicitly also simulating the formation of C molecules that are destined to dissociate into E and S molecules. Therefore, the PPTA can accelerate the stochastic simulations of the enzymatic reaction scheme (1), especially when $k_{-1} \gg k_2$ where the reduction in each $|\mathbf{T_i}|$ by this new approach is substantial.

Let us first consider the special case where the total molecular count of the enzyme is 1 (i.e., $e_{tot} = 1$), and describe the derivation of the PPTA model. This section then extends this special case to more general cases where the total molecular count of the enzyme is greater than 1 (i.e., $e_{tot} > 1$).

When e_{tot} is 1, the enzyme state for all $t \geq 0$ is defined by $X_E(t) = 1 - X_C(t)$. Also, R_1 is only enabled when E is active (i.e., $X_E(t) = 1$), and R_{-1} and R_2 are only enabled when C is active (i.e., $X_C(t) = 1$). In this case, $\mathbf{X}(t)$ can be seen as a temporal-homogeneous birth-death Markov process $\mathbf{Y}(t)$ with $2s_{tot} + 1$ states as shown in Figure 1. Each state $s \in [0, 2s_{tot}]$ of $\mathbf{Y}(t)$ can then be mapped onto a system state $\mathbf{x_s}$ of $\mathbf{X}(t)$ by the relationship: $\mathbf{x_s} \equiv ((s+1) \bmod 2, s_{tot} - \lceil s/2 \rceil, s \bmod 2, \lfloor s/2 \rfloor)$. Thus, for all $t > t_0$, the probability that $\mathbf{Y}(t) = s$ given that $\mathbf{Y}(t_0) = 0$ is the same as the probability that $\mathbf{X}(t) = \mathbf{x_s}$ given that $\mathbf{X}(t) = \mathbf{x_{t_0}}$, and with the initial condition $\mathbf{X}(t_0) = \mathbf{x_{t_0}}$, each simulation run of $\mathbf{Y}(t)$ starts in state 0, and eventually ends up in state $2s_{tot}$. Since E is active only in even number states in this process, R_1 can fire only in these states except in state $2s_{tot}$. Similarly, C is active only in odd number states, so R_{-1} and R_2 can fire in these states. Thus, let $\mathbf{S_e}$ be a set of even number states $\{2m \mid 0 \leq m \leq s_{tot}\}$, $\mathbf{S_{e'}}$ be a set of states $\mathbf{S_e} \setminus \{2s_{tot}\}$, and $\mathbf{S_o}$ be a set of odd number states $\{2m+1 \mid 0 \leq m < s_{tot}\}$. Then, the $s \to s+1$ transition rate λ_s is $a_1(\mathbf{x_s})$ if $s \in \mathbf{S_{e'}}$, and $a_2(\mathbf{x_s})$ if $s \in \mathbf{S_o}$, whereas the $s \to s-1$ transition rate μ_s is $a_{-1}(\mathbf{x_s})$ if $s \in \mathbf{S_o}$ and 0 if $s \in \mathbf{S_e}$.

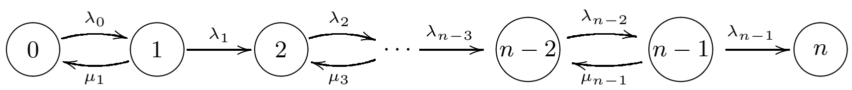

$$\begin{tikzcd} 0 \arrow[r, bend left, "\lambda_0"] & 1 \arrow[l, bend left, "\mu_1"] \arrow[r, "\lambda_1"] & 2 \arrow[r, bend left, "\lambda_2"] & \cdots \arrow[l, bend left, "\mu_3"] \arrow[r, "\lambda_{n-3}"] & n-2 \arrow[r, bend left, "\lambda_{n-2}"] & n-1 \arrow[l, bend left, "\mu_{n-1}"] \arrow[r, "\lambda_{n-1}"] & n \end{tikzcd}$$

Fig. 1. The state graph of the birth-death process of the enzymatic reaction scheme (1) when $e_{tot} = 1$. This birth-death process has $n + 1$ states where $n = 2s_{tot}$, and each state s can be mapped onto a system state of $\mathbf{X}(t)$ by the relationship $\mathbf{x_s} \equiv ((s + 1) \bmod 2, s_{tot} - \lceil s/2 \rceil, s \bmod 2, \lfloor s/2 \rfloor)$. Transition rate λ_s is $a_1(\mathbf{x_s})$ if s is an even number, and $a_2(\mathbf{x_s})$ if s is an odd number. Transition rate μ_s is $a_{-1}(\mathbf{x_s})$ if s is an odd number and 0 otherwise.

Suppose $\mathbf{Y}(t)$ starts in state s_0 where $s_0 \in \mathbf{S_{e'}}$. Then, the average waiting time that $\mathbf{Y}(t)$ spends in states s_0 and $s_0 + 1$ for each simulation run are equivalent to $t(s_0; s_0 \to s_0 + 2)$ and $t(s_0 + 1; s_0 \to s_0 + 2)$, respectively, where $t(s_j; s_i \to s_k)$ is the mean time that $\mathbf{Y}(t)$ spends in state s_j in the course of a (first) passage from s_i to s_k. In other words, using the variable $t(s_j; s_i \to s_k)$,

$$t(s_0; s_0 \to s_0 + 2) \equiv t(s_0; 0 \to 2s_{tot}),$$
$$t(s_0 + 1; s_0 \to s_0 + 2) \equiv t(s_0 + 1; 0 \to 2s_{tot}),$$

since the transitions: $s_0 \to s_0 - 1$ and $s_0 + 2 \to s_0 + 1$ are not allowed in $\mathbf{Y}(t)$. To find out the mean waiting times in states s_0 and $s_0 + 1$ using the *pedestrian approach* [9], then, variables: $v(s)$ and $v_+(s)$ are defined. The variable $v(s)$ is defined as the average number of visits by $\mathbf{Y}(t)$ to state s in the course of a first passage from state 0 to state $2s_{tot}$ while $v_+(s)$ is defined as the average number of transitions $s \to s+1$ taken by $\mathbf{Y}(t)$ in the course of a first passage from state 0 to state $2s_{tot}$. Using these variables, the probability that $\mathbf{Y}(t)$ moves to state s_0+2 from state s_0+1 at the very next jump can be expressed as $v_+(s_0+1)/v(s_0+1)$. Since this probability can also be expressed as $\lambda_{s_0+1}/(\lambda_{s_0+1} + \mu_{s_0+1})$, and since $v_+(s_0 + 1)$ is 1, we can say

$$v(s_0 + 1) = \frac{(\lambda_{s_0+1} + \mu_{s_0+1})}{\lambda_{s_0+1}}.$$

Because state $s_0 + 1$ can only be visited from state s_0 in $\mathbf{Y}(t)$, $v_+(s_0)$ must be equal to $v(s_0 + 1)$. Furthermore, since the transition from state s_0 to state $s_0 - 1$ cannot occur in $\mathbf{Y}(t)$, $v(s_0)$ must be equivalent to $v(s_0 + 1)$. Therefore,

$$v(s_0) = \frac{(\lambda_{s_0+1} + \mu_{s_0+1})}{\lambda_{s_0+1}}.$$

Now, let $T(s)$ be a random variable which represents the pausing time in state s in $\mathbf{Y}(t)$. Then, since $\mathbf{Y}(t)$ is a temporally homogeneous birth-death Markov process, $T(s)$ must be a random variable which is necessarily exponentially distributed with parameter $(\lambda_s + \mu_s)$. Then, the mean pausing times in states s_0 and $s_0 + 1$ can be expressed, respectively, as:

$$\begin{aligned}
\langle T(s_0)\rangle &= \int_0^\infty t\lambda_{s_0}\exp(-\lambda_{s_0}t)dt = \frac{1}{\lambda_{s_0}}, \\
\langle T(s_0+1)\rangle &= \int_0^\infty t\left(\lambda_{s_0+1}+\mu_{s_0+1}\right)\exp\left(-\left(\lambda_{s_0+1}+\mu_{s_0+1}\right)t\right)dt \\
&= \frac{1}{\lambda_{s_0+1}+\mu_{s_0+1}}.
\end{aligned}$$

Since $t(s_j; s_i \to s_k)$ can be formulated as the product of $\langle T(s_j)\rangle$ and $v(s_j)$, the mean waiting times that $\mathbf{Y}(t)$ spends in states s_0 and $s_0 + 1$ can be expressed as:

$$\begin{aligned}
t(s_0; 0 \to 2s_{tot}) &= \frac{\lambda_{s_0+1}+\mu_{s_0+1}}{\lambda_{s_0+1}\lambda_{s_0}} = \frac{a_2(\mathbf{x_{s_0+1}}) + a_{-1}(\mathbf{x_{s_0+1}})}{a_2(\mathbf{x_{s_0+1}})a_1(\mathbf{x_{s_0}})}, \\
t(s_0+1; 0 \to 2s_{tot}) &= \frac{1}{\lambda_{s_0+1}} = \frac{1}{a_2(\mathbf{x_{s_0+1}})}.
\end{aligned}$$

Using this information, $\mathbf{Y}(t)$ can be approximated by creating a temporally homogeneous birth Markov process $\mathbf{Y}'(t)$ with the same state space where the mean waiting time in each state s is $t(s; 0 \to 2s_{tot})$ derived from $\mathbf{Y}(t)$. Figure 2 shows the state graph of $\mathbf{Y}'(t)$. Since the waiting time in each state s in $\mathbf{Y}'(t)$ is exponentially distributed, the $s \to s+1$ transition rate λ'_s is the reciprocal of $t(s; 0 \to 2s_{tot})$. Thus, λ'_s is $a_1(\mathbf{x_s})a_2(\mathbf{x_{s+1}})/(a_{-1}(\mathbf{x_{s+1}})+a_2(\mathbf{x_{s+1}}))$ if $s \in \mathbf{S_o}$ and $a_2(\mathbf{x_s})$ if $s \in \mathbf{S_{e'}}$. Therefore, using the PPTA, the enzymatic reaction scheme (1) with e_{tot} being 1 is approximated by a new reaction scheme:

$$\mathrm{E} + \mathrm{S} \xrightarrow{k_{1'}} \mathrm{C} \xrightarrow{k_2} \mathrm{E} + \mathrm{P} \tag{2}$$

where $k_{1'} = k_1k_2/(k_{-1} + k_2)$.

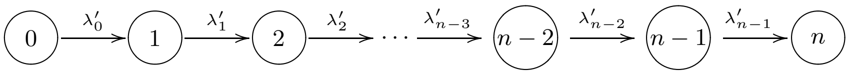


Fig. 2. The state graph of the pure birth process of the PPTA model when $e_{tot} = 1$. This birth process has the same state space as the birth-death process in Figure 1. Transition rate λ'_s is $a_1(\mathbf{x_s})a_2(\mathbf{x_{s+1}})/(a_{-1}(\mathbf{x_{s+1}}) + a_2(\mathbf{x_{s+1}}))$ if $s \in \mathbf{S_o}$ and $a_2(\mathbf{x_s})$ if $s \in \mathbf{S_{e'}}$.

When $e_{tot} > 1$, the enzymatic reaction scheme (1) is considered as a set of the enzymatic reactions as follows:

$$\mathrm{E_i} + \mathrm{S} \underset{k_{-1}}{\overset{k_1}{\rightleftarrows}} \mathrm{C_i} \xrightarrow{k_2} \mathrm{E_i} + \mathrm{P}, \quad 1 \le i \le e_{tot}$$

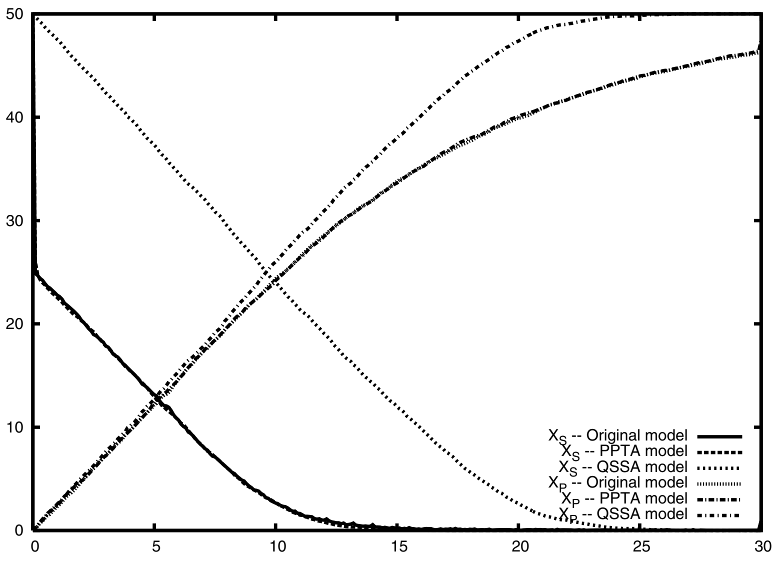


Fig. 3. Comparison of the average time evolutions of the original enzymatic reaction model, its PPTA model, and its QSSA model with initial conditions: $\mathbf{x_{t_0}} = (25, 50, 0, 0)$ and the rate constants: $k_1 = 100.0,\ k_{-1} = 10.0,\ k_2 = 0.01$

where $X_{E_i}(t_0) = 1$ and $X_{C_i}(t_0) = 0$ for each i. Although simulations of this process is definitely slower than that of $\mathbf{X}(t)$, this transformation itself does not require any approximation as, when $\mathbf{X}(t) = \mathbf{x}$, $k_1 x_E x_S \equiv \sum_{i=1}^{e_{tot}} k_1 x_{E_i} x_S$, $k_{-1} x_C \equiv \sum_{i=1}^{e_{tot}} k_{-1} x_{C_i}$, and $k_2 x_C \equiv \sum_{i=1}^{e_{tot}} k_2 x_{C_i}$. Thus, by applying the PPTA to each of the transformed enzymatic reactions, the enzymatic reaction scheme (1) can be approximated by

$$\mathrm{E_i} + \mathrm{S} \xrightarrow{k_{1'}} \mathrm{C_i} \xrightarrow{k_2} \mathrm{E_i} + \mathrm{P}, \quad 1 \leq i \leq e_{tot},$$

which can now be represented using reaction scheme (2). This implies that the accuracy of the PPTA of the $e_{tot} > 1$ case is based on that of the PPTA of the $e_{tot} = 1$ case, and that the PPTA model provides the most accurate results if $e_{tot} = 1$.

The two parameters in a PPTA model: $k_{1'}$ and k_2 can be derived from K_M, and V_{max}, the maximal reaction rate. Unlike the parameters: k_1 and k_{-1}, the parameters K_M and V_{max} can actually be measured experimentally. Thus, a PPTA model can be constructed and simulated even when full knowledge of the underlying enzymatic reaction is not available and the enzymatic reaction cannot be analyzed quantitatively at that level of detail. This is also true for a QSSA model as its MM form only requires K_M and V_{max} parameters; however, since a PPTA model does not assume that the intermediate species is in quasi-steady state, a PPTA model may perform better than a QSSA model in terms of accuracy, especially in the pre-steady state phase. For example, suppose the enzymatic reaction scheme (1) has the conditions:

$$\mathbf{x_{t_0}} = (25, 50, 0, 0), \ k_1 = 100.0, \ k_{-1} = 10.0, \ k_2 = 0.1.$$

Then, since e_{tot} is arguably much smaller than s_{tot}, the QSSA *could* be applied to safely approximate the temporal behavior of the underlying enzymatic reaction. However, in this system, the propagation effects of the pre-steady state dynamics are rather important, making any QSSA-based models unable to describe the temporal behavior well. Therefore, as shown in Figure 3, the estimated mean time evolution of this enzymatic reaction model is captured more accurately by the PPTA model than by the QSSA model.

3 Case Studies

This section describes the benefits gained by the PPTA method by applying it to various systems containing enzymatic reaction scheme. This section first considers two model of the enzymatic reaction scheme (1) which are used to help illustrate the application of the slow-scale SSA in [21]. It then considers the *enzymatic futile cycle* motif which can be ubiquitously seen in biological systems including GTPase cycles, mitogen-activated protein kinase cascades, and glucose mobilization [4]. Finally, it considers a more complex competitive enzymatic reaction. Each model is encoded in SBML [23] and simulated for 1,000 runs using the same stochastic simulator, an optimized SSA implementation within our modeling and analysis tool `reb2sac` [25]. Accuracy of a PPTA model is measured by comparing the time evolution of means and standard deviations.

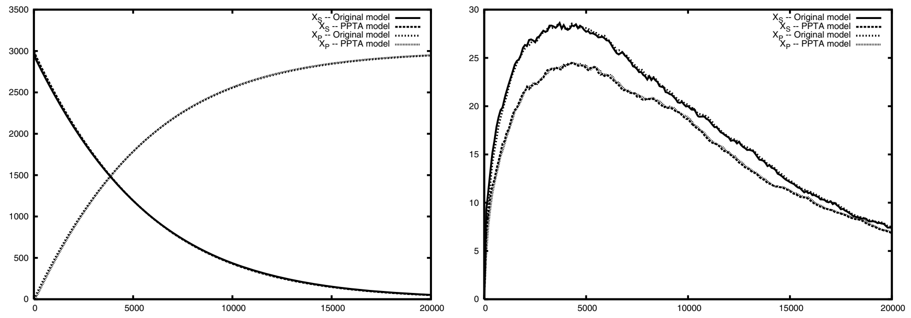
3.1 Single Enzymatic Reaction

The first model of the enzymatic reaction scheme (1) has the following initial condition and the reaction rate constants:

$$\mathbf{x_{t_0}} = (220, 3000, 0, 0), \ k_1 = 0.01, \ k_{-1} = 100.0, \ k_2 = 0.01.$$

This system is simulated for 20,000 time units and each data point is plotted every 100 time units. Figure 4 shows the results from the original model and the PPTA model of this system. The estimated means and standard deviations of X_S and X_P are shown in Figures 4(a) and (b), respectively.

The results from the PPTA model are in a very close agreement with those from the original model, yet the speedup gained by the PPTA model is significant. While the entire simulation of the original model takes 68.58 hours, that of the PPTA model only takes 22.8 seconds, achieving more than 10,800 times speedup. Furthermore, since the speedup gained by the slow-scale SSA is about 950 on this system, the PPTA method is able to outperform the slow-scale SSA by an order of magnitude.



(a) (b)

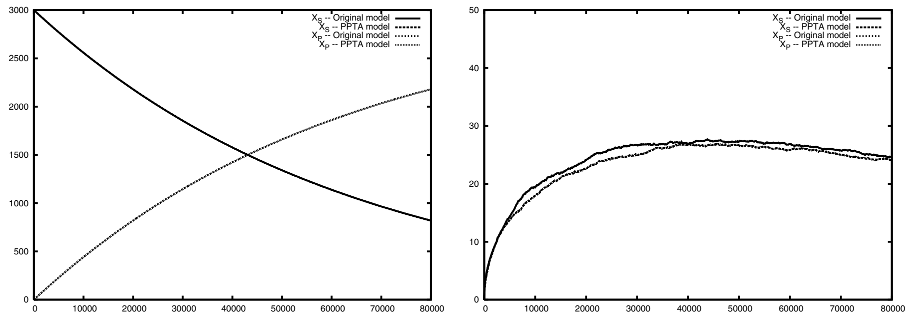
Fig. 4. Comparison of the original enzymatic reaction model and its PPTA model with initial conditions: $\mathbf{x_{t_0}} = (220, 3000, 0, 0)$ and the rate constants: $k_1 = 0.01,\ k_{-1} = 100.0,\ k_2 = 0.01$. (a) Means of X_S and X_P. (b) Standard deviations of X_S and X_P.

The second enzymatic reaction system has the following initial conditions and the specification of the reaction rate constants:

$$\mathbf{x_{t_0}} = (10, 3000, 0, 0)\,,\ k_1 = 0.01,\ k_{-1} = 600.0,\ k_2 = 0.1.$$

This system illustrates a case where the average of $X_C(t)$ remains less than 1 as the maximum reaction rate of R_1 (i.e., $k_1 e_{tot} s_{tot}$) is less than k_{-1}. This system is simulated for 80,000 time units and each data point is again plotted every 100 time units. Figure 5(a) shows the estimated means of X_S and X_P, and Figure 5(b) shows the estimated standard deviations.

Both the means and the standard deviations from the PPTA model track those from the original model very well while, at the same time, the simulation time of the PPTA is substantially reduced compared with that of the original model.



(a) (b)

Fig. 5. Comparison of the original enzymatic reaction model and its PPTA model with initial conditions: $\mathbf{x_{t_0}} = (10, 3000, 0, 0)$ and the rate constants: $k_1 = 0.01,\ k_{-1} = 600.0,\ k_2 = 0.1$. (a) Means of X_S and X_P. (b) Standard deviations of X_S and X_P.

Whereas the simulation of the original model takes 27.63 hours, that of the PPTA model only takes 17.9 seconds, improving the computation performance by a factor of more than 5,500. Since the speedup gained by the slow-scale SSA is 400 on this system, the PPTA method is able to outperform the slow-scale SSA by an order of magnitude.

3.2 Enzymatic Futile Cycle

The enzymatic futile cycle motif consists of two instances of the enzymatic reaction scheme (1) as follows:

$$\mathrm{E}_1 + \mathrm{S} \underset{k_{-1}}{\overset{k_1}{\rightleftarrows}} \mathrm{C}_1 \xrightarrow{k_2} \mathrm{E}_1 + \mathrm{P}, \quad \mathrm{E}_2 + \mathrm{P} \underset{k_{-3}}{\overset{k_3}{\rightleftarrows}} \mathrm{C}_2 \xrightarrow{k_4} \mathrm{E}_2 + \mathrm{S}\,. \tag{3}$$

One is to transform S into P catalyzed by E_1, and the other one is to transform P into S catalyzed by E_2. This motif is found in many biological systems [4], abstracting away low-level detail of the motif such as unproductive substrate-complex cycles may provide a significant improvement in performance of the overall system behavior analysis. With the PPTA method, unproductive dissociation reactions are removed, transforming the enzymatic futile cycle model into the following PPTA model:

$$\mathrm{E}_1 + \mathrm{S} \xrightarrow{k_{1'}} \mathrm{C}_1 \xrightarrow{k_2} \mathrm{E}_1 + \mathrm{P}, \quad \mathrm{E}_2 + \mathrm{P} \xrightarrow{k_{3'}} \mathrm{C}_2 \xrightarrow{k_4} \mathrm{E}_2 + \mathrm{S}, \tag{4}$$

where $k_{3'} = k_3 k_4/(k_{-3} + k_4)$.

The original enzymatic futile cycle model and its PPTA model are simulated for 300 time units with one time unit plot-interval to analyze the accuracy as well as the performance gain of the PPTA model with the initial conditions:

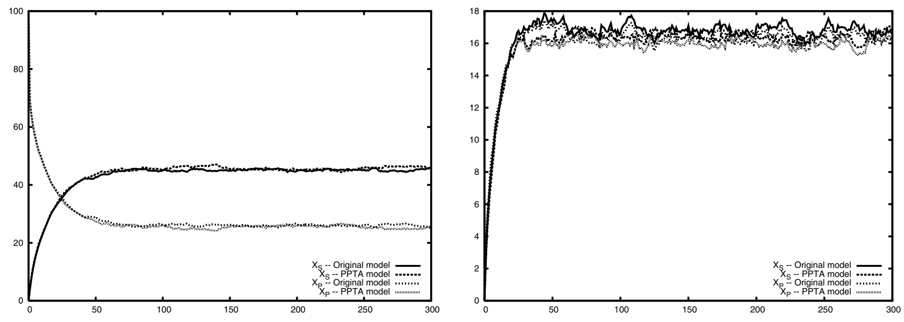
$$(X_S(0), X_P(0), X_{E_1}(0), X_{E_2}(0), X_{C_1}(0), X_{C_2}(0)) = (0, 100, 10, 20, 0, 0),$$

and the rate constants:

$$k_1 = 10^3; k_{-1} = 1.5 \times 10^3; k_2 = 2; k_3 = 10^3; k_{-3} = 5 \times 10^2; \text{ and } k_4 = 1.$$

Since each numerical simulation of the two models starts with no copies of S and 10 copies of E_1, this system illustrates a case where substrate is initially lower than the catalyzing enzyme. Furthermore, since $X_S(t) + X_P(t) + X_{C_1}(t) + X_{C_2}(t)$ is fixed at 100 for all $t \geq 0$, this enzymatic futile cycle system illustrates an applicability of the PPTA model when the numbers of both substrate and enzyme molecules are very low.

Figure 6 shows the results from the original model and the PPTA model of this enzymatic futile cycle system. The time evolutions of the estimated means and standard deviations of X_S and X_P are shown in Figures 6(a) and (b), respectively. From these figures, it is clear that both the means and the standard deviations of X_S and X_P from the PPTA model approximate those from the original model very well. The simulation of the original enzymatic futile cycle model takes 17.73 hours while that of the PPTA model only takes 87.51 seconds which represents a speedup of more than 729 times. Furthermore, this demonstrates that the PPTA can be applicable to systems with a low number of substrate molecules.



(a) (b)

Fig. 6. Comparison of the original enzymatic futile cycle model (3) and its PPTA model (4) with initial conditions: $X_S(0) = 0$, $X_P(0) = 100$, $X_{E_1}(0) = 10$, $X_{E_2}(0) = 20$, $X_{C_1}(0) = 0$, $X_{C_2}(0) = 0$, and the rate constants: $k_1 = 1.0 \times 10^3$, $k_{-1} = 1.5 \times 10^3$, $k_2 = 2.0$, $k_3 = 1.0 \times 10^3$, $k_{-3} = 5.0 \times 10^2$, $k_4 = 1.0$. (a) Means of X_S and X_P. (b) Standard deviations of X_S and X_P.

3.3 Competitive Enzymatic Reaction

To further demonstrate the usefulness of the PPTA, the following competitive enzymatic reaction scheme is considered:

$$\begin{aligned}
&\mathrm{S}_1 \xrightarrow{k_{b1}} \mathrm{P}_1, \quad \mathrm{E} + \mathrm{S}_1 \underset{k_{-1}}{\overset{k_1}{\rightleftarrows}} \mathrm{C}_1 \xrightarrow{k_2} \mathrm{E} + \mathrm{P}_1, \quad \xrightarrow{k_{p1}} \mathrm{S}_1, \quad \mathrm{S}_1 \xrightarrow{k_{d1}}, \\
&\mathrm{S}_2 \xrightarrow{k_{b2}} \mathrm{P}_2, \quad \mathrm{E} + \mathrm{S}_2 \underset{k_{-3}}{\overset{k_3}{\rightleftarrows}} \mathrm{C}_2 \xrightarrow{k_4} \mathrm{E} + \mathrm{P}_2, \quad \xrightarrow{k_{p2}} \mathrm{S}_2, \quad \mathrm{S}_2 \xrightarrow{k_{d2}} .
\end{aligned} \tag{5}$$

In this scheme, both S_1 and S_2 compete to bind to E to produce P_1 and P_2, respectively. Also, this scheme contains basal reactions to transform S_1 and S_2 into P_1 and P_2, respectively, without being catalyzed by E. Moreover, since substrates S_1 and S_2 are often produced and consumed via various reactions, reaction scheme (5) also contains reactions to model productions and consumptions of S_1 and S_2.

The PPTA model of the competitive enzymatic reaction model (5) removes the substrate-dissociation reactions from C_1 and C_2, resulting in the following model:

$$\begin{aligned}
&\mathrm{S}_1 \xrightarrow{k_{b1}} \mathrm{P}_1, \quad \mathrm{E} + \mathrm{S}_1 \xrightarrow{k_{1'}} \mathrm{C}_1 \xrightarrow{k_2} \mathrm{E} + \mathrm{P}_1, \quad \xrightarrow{k_{p1}} \mathrm{S}_1, \quad \mathrm{S}_1 \xrightarrow{k_{d1}}, \\
&\mathrm{S}_2 \xrightarrow{k_{b2}} \mathrm{P}_2, \quad \mathrm{E} + \mathrm{S}_2 \xrightarrow{k_{3'}} \mathrm{C}_2 \xrightarrow{k_4} \mathrm{E} + \mathrm{P}_2, \quad \xrightarrow{k_{p2}} \mathrm{S}_2, \quad \mathrm{S}_2 \xrightarrow{k_{d2}} .
\end{aligned} \tag{6}$$

To analyze the accuracy of this PPTA model, the following initial conditions:

$$(X_E(0), X_{S_1}(0), X_{S_2}(0), X_{P_1}(0), X_{P_2}(0), X_{C_1}(0), X_{C_2}(0)) = (10, 0, 0, 0, 0, 0),$$

and the rate constants:

$$k_{b1} = 2 \cdot 10^{-5}; k_1 = 10^2; k_{-1} = 10^2; k_2 = 0.1; k_{p1} = 10; k_{d1} = 0.2;$$
$$k_{b2} = 10^{-5}; k_3 = 200; k_{-3} = 10^2; k_4 = 0.15; k_{p2} = 10; \text{ and } k_{d2} = 0.2,$$

are used for the simulations. The values of rate constants for the productions and consumptions of S_1 and S_2 are chosen so that the consumption rate constants are relatively high to capture isolation of substrates from binding to the enzyme and that both substrates are present in low counts throughout the simulations (i.e., $\forall t \geq 0.\ \langle X_{S_1}(t)\rangle \leq 100 \wedge \langle X_{S_2}(t)\rangle \leq 100$). The values of basal transformation rate constants k_{b1} and k_{b2} are chosen so that basal transformation rates are much smaller than those from the catalyzed reactions when the substrates are present in low counts (i.e., $k_2 \cdot e_{tot} \gg 100k_{b1}$ and $k_4 \cdot e_{tot} \gg 100k_{b2}$).

Figure 7 shows the results from the simulations of the two models. The estimated means of X_{S_1} and X_{S_2} are shown in Figure 7(a), while the estimated standard deviations of X_{S_1} and X_{S_2} are shown in Figure 7(b). Once again, both the means and the standard deviations of X_{S_1} and X_{S_2} from the PPTA model track those from the original model very well with a substantial improvement in simulation time. While the simulation of the original model takes 65.16 minutes, that of the PPTA model only takes 35.78 seconds, achieving more than 109 times speedup.

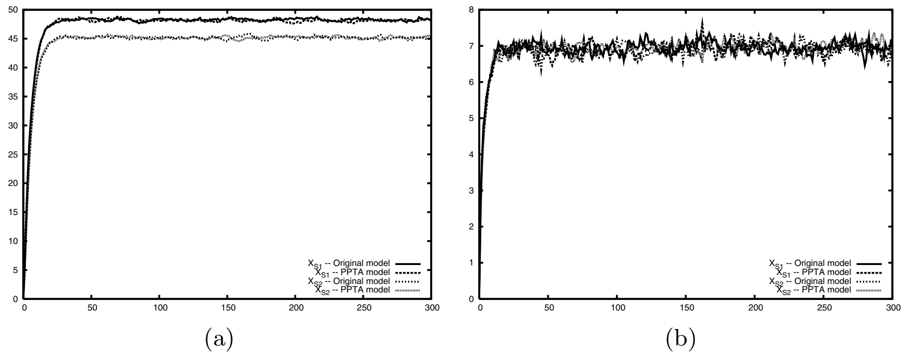


Fig. 7. Comparison of the original model of competitive enzymatic reaction model (5) and its PPTA model (6) with initial conditions: $X_E(0) = 10$ and 0 molecule for the rest of the species, and the rate constants: $k_{b1} = 2 \cdot 10^{-5}$, $k_1 = 10^2$, $k_{-1} = 10^2$, $k_2 = 0.1$, $k_{p1} = 10$, $k_{d1} = 0.2$, $k_{b2} = 10^{-5}$, $k_3 = 200$, $k_{-3} = 10^2$, $k_4 = 0.15$, $k_{p2} = 10$, $k_{d2} = 0.2$. (a) Means of X_{S_1} and X_{S_2}. (b) Standard deviations of X_{S_1} and X_{S_2}.

4 Conclusion

This paper introduces a new model abstraction method, *production-passage-time approximation* (PPTA), that can significantly improve the temporal behavior analysis time of enzymatic reaction systems. As a case study, we have applied the PPTA method to various systems, and compared the accuracy as well as the runtime between the original model and the PPTA model. The preliminary results are promising. This paper has shown that the PPTA model can make stochastic simulations orders of magnitude faster without any significant loss in accuracy. This paper has also shown that the PPTA method achieves an acceleration of an order of magnitude over the slow-scale SSA for the two enzymatic reaction systems from [21]. Moreover, this paper has demonstrated that the PPTA can be utilized to efficiently approximate more complex systems, exemplified here using an enzymatic futile cycle model and a competitive enzymatic reaction model. Additionally, our approach can also be used within a continuous, deterministic framework to remove the stiff condition often found in enzymatic reactions with $k_{-1} \gg k_2$ that gives significant computational challenges. Furthermore, since our approach does not require a customized simulation procedure for enzymatic reactions, it allows biochemical systems comprising such reactions along with other types of reactions to still take advantage of utilizing general stochastic simulation tools for the standard Gillespie stochastic simulation algorithm.

Future work includes comprehensive analysis of the PPTA errors under various conditions and analysis of efficiency-versus-accuracy among the PPTA and other approximations such as the QSSA based on the initial conditions, that is, a static and systematic approach to determine when to use the PPTA.

References

1. Arkin, A., Ross, J., McAdams, H.: Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected escherichia coli cells. Genetics **149** (1998) 1633–1648
2. Elowitz, M.B., Levine, A.J., Siggia, E.D., Swain, P.S.: Stochastic gene expression in a single cell. Science **297** (2002) 1183–1186
3. Rao, C.V., Wolf, D.M., Arkin, A.P.: Control, exploitation and tolerance of intracellular noise. Nature **420** (2002) 231–238
4. Samoilov, M., Plyasunov, S., Arkin, A.P.: Stochastic amplification and signaling in enzymatic futile cycles through noise-induced bistability with oscillations. Proceedings of the National Academy of Sciences US **102**(7) (2005) 2310–5
5. McAdams, H.H., Arkin, A.: Stochastic mechanisms in gene expression. Proceedings of the National Academy of Sciences USA **94**(3) (1997) 814–819
6. Pedraza, J.M., van Oudenaarden, A.: Noise Propagation in Gene Networks. Science **307**(5717) (2005) 1965–1969
7. Cai, L., Friedman, N., Xie, X.S.: Stochastic protein expression in individual cells at the single molecule level. Nature **440**(7082) (2006) 358–362
8. Newman, J.R.S., Ghaemmaghami, S., Ihmels, J., Breslow, D.K., Noble, M., DeRisi, J.L., Weissman, J.S.: Single-cell proteomic analysis of s. cerevisiae reveals the architecture of biological noise. Nature **441**(7095) (2006) 840–846
9. Gillespie, D.T.: Markov Processes An Introduction for Physical Scientists. Academic Press, Inc. (1992)
10. Gillespie, D.: The chemical langevin equation. J. Chem. Phys. **113**(1) (2000)
11. Gillespie, D.T.: 5.11. In: Handbook of Materials Modeling. Springer (2005) 1735–1752
12. Gillespie, D.T.: A rigorous derivation of the chemical master equation. Physica A **188** (1992) 404–425

13. Achimescu, S., Lipan, O.: Signal propagation in nonlinear stochastic gene regulatory networks. IEE Proc. Sys. Bio. **153** (2006) 120–134
14. Gillespie, D.T.: A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. Journal of Computational Physics **22** (1976) 403–434
15. Gibson, M., Bruck, J.: Efficient exact stochastic simulation of chemical systems with many species and many channels. J. Phys. Chem. **A 104** (2000) 1876–1889
16. Gillespie, D.T.: Approximate accelerated stochastic simulation of chemically reacting systems. J. Chem. Phys. **115**(4) (2001) 1716–1733
17. Rathinam, M., Cao, Y., Petzold, L., Gillespie, D.: Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method. J. Chem. Phys. **119** (2003) p12784–94
18. Gillespie, D., Petzold, L.: Improved leap-size selection for accelerated stochastic simulation. J. Chem. Phys. **119** (2003)
19. Cao, Y., Gillespie, D., Petzold, L.: Avoiding negative populations in explicit tau leaping. J. Chem. Phys. **123** (2005)
20. Rao, C.V., Arkin, A.P.: Stochastic chemical kinetics and the quasi-steady-state assumption: Application to the gillespie algorithm. J. Phys. Chem. **118**(11) (2003)
21. Cao, Y., Gillespie, D., Petzold, L.: Accelerated stochastic simulation of the stiff enzyme-substrate reaction. J. Phys. Chem. **123** (2005)
22. Cao, Y., Gillespie, D., Petzold, L.: The slow-scale stochastic simulation algorithm. J. Chem. Phys. **122** (2005)
23. Finney, A., Hucka, M.: Systems Biology Markup Language (SBML) level 2: Structures and facilities for model definitions (2003)
24. BioSPICE. (`http://www.biospice.org/`)
25. Kuwahara, H., Myers, C., Barker, N., Samoilov, M., Arkin, A.: Automated abstraction methodology for genetic regulatory networks. Trans. on Comput. Syst. Biol. VI (2006)

Shift-Invariant Adaptive Double Threading: Learning MHC II - Peptide Binding

Noah Zaitlen[1], Manuel Reyes-Gomez[2], David Heckerman[2], and Nebojsa Jojic[2,⋆]

[1] University of California San Diego, La Jolla CA 92093, USA
[2] Microsoft Research, Redmond WA 98052, USA
jojic@microsoft.com

Abstract. Specificity of MHC binding to short peptide fragments from cellular as well as pathogens' proteins has been found to correlate with disease outcome and pathogen or cancer evolution. The large variation in MHC class II epitope length has complicated training of predictors for binding affinities compared to MHC class I. In this paper, we treat the relative position of the peptide inside the MHC protein as a hidden variable, and model the ensemble of different binding configurations. The training procedure iterates the predictions with re estimation of the parameters of a binding groove model. We show that the model generalizes to new MHC class II alleles, which were not a part of the training set. To the best of our knowledge, our technique outperforms all previous approaches to MHC II epitope prediction. We demonstrate how our model can be used to explain previously documented associations between MHC II alleles and disease.

1 Introduction

The open binding pocket of the MHC class II molecules allow for a greater variation in peptide length relative to the closed pocket of the MHC class I molecules. This difference combined with the relative lack of sequence similarity across binding peptides makes MHC binding prediction significantly more challenging for the class II molecules. Recent efforts for MHC class II binding have been focused on methods to identify a nine amino acid binding core of the peptide, which is widely believed to be responsible for a majority of the binding. This is then combined with one of numerous existing methods for predicting MHC class I binding over the derived nonamers.

Several approaches to binding core identification have been explored. Many of these search for an optimal alignment of nonamers across the binding peptides. [2] and [1] use MEME [16] to identify and align the over represented nonamers. Gibbs sampling is used by [3]. The Linear Programming method of [5] effectively produces an alignment or choice of nonamers during training.

The alignment can be a pre-processing step as in [2] and [1] who use the set of nonamers in the alignment as direct input into their MHC class I predictors.

⋆ Corresponding author.

The Gibbs sampling method of [3] uses the PSSM of the alignment as input into the binding prediction method. The method closest to our work in the Linear Programming model proposed by [5]. They use a sliding window over each peptide and a set of LP constraints which attempt to identify the window for each peptide that maximizes their ability to separate binders from non-binders with a PSSM model.

Several MHC class I tools have been applied to these nonamer alignments. Motif based methods such as RANKPEP [2], attempt to identify amino acids at particular positions that are characteristic of binding for a given allele. A variety of machine learning base methods such as neural networks, support vector machines, and hidden Markov Models have been applied. Structure based methods such as ours and [4] attempt to model the physics of MHC binding using the growing number of MHC class I and II molecules that have been solved by X-ray crystallography.

In this paper, we demonstrate a new method for predicting binding to arbitrary MHC class II alleles. Our binding model is based on the protein structure of the molecules and treats the possible peptide alignments as an ensemble of possible configurations. Rather than assuming simply that any peptide alignment is equally possible, or turning to separate methodology to provide the best alignment, we infer the distribution over possible states for each peptide-MHC combination based on the predicted state energy. This is not treated as a distribution over a variable with mutually exclusive and exhaustive states, but as population frequencies in the thermodynamics sense, and the equivalent total binding energy is estimated accordingly. This is the key difference between our approach and previous approaches to MHC class II binding prediction, which enabled us to outperform, to the best of our knowledge, all previously published techniques.

2 Modeling Variable Peptide Position in the MHC Groove

Since a longer peptide (15-30 amino acids, for example) has only part of it in the groove of the MHC class II molecule, we introduce a hidden random integer variable ℓ that represents the unknown alignment of the peptide with the groove. The largest difference in the binding of peptides to the same MHC II allele is in where the bound part of the peptides start. We represent the starting index of this segment with the variable $\ell \in [1, N-8]$, where N is the length of the peptide, and we assume the segment that is inside the groove is 9 amino acids long (Fig. 1). There are, of course, other hidden variables that describe the binding configuration, such as the particular geometric configuration m of the amino acids in the groove of the MHC molecule from the available crystal structures. In this section, we will denote all such hidden variables with $\mathbf{h}$, and in the next section, we will define $\mathbf{h} = (\ell, m)$ as the hidden variables in our shifted adaptive double threading model.

For now, we simply assume the existence of a model $E(\mathbf{s}, \mathbf{e}, \mathbf{h})$, where $\mathbf{s}$ denotes a particular MHC allele, and $\mathbf{e}$ denotes a particular peptide, such that if the

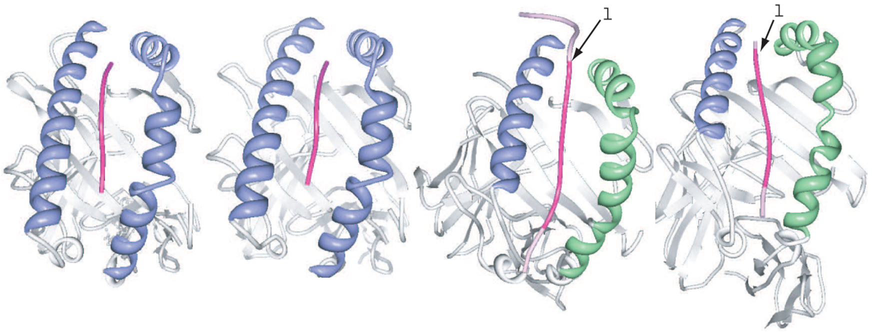


Fig. 1. Examples of binding configurations of MHC class I molecules (left) and MHC class II (right) bound to different peptides. The class I molecules are gray, except for the alpha helices forming the groove, which are shown in blue to accentuate the peptide (pink) sitting snugly inside it. The class II molecules consist of two separate chains. Their alpha helices (blue and green) form a similar groove to that of class I molecules. A class II molecule can bind to peptides of more variable length, with only a short segment (dark pink) of the bound peptide captured in its groove, and the peptide tails (light pink) sticking out and having a smaller effect. The start of the segment that fits the groove is modeled by the random variable ℓ in Section 2.

setting for the hidden variables $\mathbf{h}$ are provided, the model can produce a good estimate of the binding energy for the pair $\mathbf{s}, \mathbf{e}$. In many past approaches to MHC binding, the settings of $\mathbf{h}$ (in particular the alignment analogous to our variable ℓ), were provided by a separate routine, unrelated to the energy model E.

We index energy states $E(\mathbf{s}, \mathbf{e}, \mathbf{h})$ of a MHC-peptide complex by $\mathbf{h}$ with the partition function $Z = \sum_h e^{-E(\mathbf{s},\mathbf{e},\mathbf{h})}$, and the free energy per particle of the system of such particles is $F = -\log Z$, where the kT factors are omitted, as the reported measured binding energies are dimensionless $\log IC50$ values. Thus, we can model the measured binding energy $\log IC50$ as

$$E(\mathbf{s}, \mathbf{e}) = -\log \sum_{\mathbf{h}} e^{-E(\mathbf{s},\mathbf{e},\mathbf{h})}. \tag{1}$$

In particular, for the case of a shift as the hidden variable $\mathbf{h} = \ell$, the energy of a particular configuration $E(\mathbf{s}, \mathbf{e}, \ell)$, can be derived from a model $E_{mod}(\mathbf{s}, \mathbf{e})$ that does not deal with peptide shifts, and requires $\mathbf{e}$ to be a known k-mer sitting in the MHC groove, if the assumption is that k amino acids are in the pocket. In this case, $E(\mathbf{s}, \mathbf{e}, \ell) = E_{mod}(\mathbf{s}, \mathbf{e}_{\ell:\ell+k-1})$. Choices for models of binding given a known alignment include most previous MHC I and MHC II binding models (e.g. pssm, logistic regression, support vector machine, motif search), although they may have to be retrained in this new context. In what follows, we describe how this retraining may be done, on the example of the adaptive double-threading model [18], into which we add hidden variables according to the above recipe, and then derive an EM-like learning algorithm that can fit the parameters of the model.

An important feature of our treatment of variable peptide alignment with the groove is that the distribution over possible alignments is effectively determined by the model's energy predictions alone, rather than by the fit of these predictions to the energy data. This means that the proper alignment can be inferred not only in training, but also in testing on new peptides for which the true (measured) binding energy is not provided to the predictor. Since the energy in (4) is dominated by the minimum energy state $E(\mathbf{s}, \mathbf{e}, \mathbf{h})$, the preferred alignments will have lower energy, rather than better fit to the data.

3 Shift-Invariant Double Threading Model

Our basic binding energy model is based on the geometry of MHC-peptide complexes, and is motivated by the *threading* approach [20]. As in [18], its implementation in [19] is here augmented by including learnable parameters. The parameters are estimated from the experimental data.

Assuming that energy is additive, and that the pairwise potentials depend only on the amino acids themselves — and not on their context in the molecule — the energy becomes a sum of pairwise potentials taken from a symmetric 20×20 matrix of pairwise potentials between amino acids. These parameters are computed based on the amino acid binding physics, or from statistical analyses of amino acid pair contact preferences in large sets of available protein structures. Several sets of pairwise potentials have been described in the literature, each derived in a different way (for review see [21]. The choice of pairwise potential matrix can dramatically alter performance of the energy predictor [19].

In the adaptive double threading model of MHC I - peptide binding, the binding energy is estimated as

$$E(m, \mathbf{s}, \mathbf{e}) \approx \sum_i \sum_j w_{i,j}^m \phi_{\mathbf{s}_i, \mathbf{e}_j} h(d_{i,j}^m), \tag{2}$$

where MHC-specific weights $w_{i,j}^m$ and a trainable soft threshold function h provide added parameters whose role is to correct for the drastic approximations in the original threading approach. The adaptive soft step function and the addition of the weights w are meant to absorb the errors of the model assumptions [18].

The basic idea behind threading approaches is that, even though the structure information d is inferred from a known binding configuration of a *particular* peptide-MHC I combination, substituting *a different* peptide of the same length (or even another MHC molecule, as in our previous work) in the above equations still lead to a reasonable estimate of the binding energy for the new MHC-peptide combination. This is due to the fact that relative positions and the basic chemistry of the amino acid-amino acid interactions are fixed. Even the light changes over different geometries of peptide-groove configurations (indexed by m) have a small (though measurable) effect on the accuracy of the model. The success of the previous work on MHC I binding energy prediction attests that this main assumption holds well for MHC I molecule.

The same basic modeling strategy can be used for modeling MHC class II with one very important difference. While the fixed chemistry of the amino acid interactions and the fixed overall geometry of the MHC molecule are still relatively mild assumptions, the fixed relative position of the peptide is a gross over-approximation. Thus, the model needs to be extended to account for *variable position* of the peptide, as discussed above.

To estimate the energy of the binding configuration for a particular shift ℓ, we update our model in the following way:

$$E(m, \mathbf{s}, \mathbf{e}, \ell) \approx \sum_i \sum_{j=1+\ell}^{N+\ell} w^m_{i,j-\ell} \boldsymbol{\phi}_{\mathbf{s}_i, \mathbf{e}_{j-\ell}} h(d^m_{i,j-\ell}), \tag{3}$$

In order to fit the model to experimental binding essays, we need to express the total affinity of the peptide by summing over all the binding configurations. The binding energy is usually reported in terms of an IC50 value, which approximates the dissociation constant. The energy is assumed to be proportional to the negative log of this value, and so energy estimators are typically trained on the $E = -\log n_{IC50}$ values. When many copies of the same longer peptide are mixed with many copies of the same MHC class II molecule, binding configurations with all different shifts ℓ may form. Therefore, according to (1), we sum over the two unknown variables that meaningfully affect the energy used in (3):

$$E(\mathbf{s}, \mathbf{e}) = -\log \sum_{m,\ell} e^{-E(m,\mathbf{s},\mathbf{e},\ell)}. \tag{4}$$

Variable m, as in the case of the MHC class I molecule (2), represents the geometry of the configuration of the MHC molecule and the peptide's segment that is in the groove. The variable m influences the energy estimate through the distance matrix $d^m_{i,j}$. As the variability in the binding configurations of the groove is low, the influence of variable m is existent, but mild. In case of MHC class II molecule, this variability has a much smaller effect on the energy estimate than the shift variable ℓ – upon 3D alignment of different MHC structures, the relative positions of molecules close to the binding grooves change very little. While the slight geometry changes in the groove have an effect on the prediction, the shift variable ℓ influences the prediction much more dramatically as it alters the predicted amino acid composition of the peptide's segment sitting in the groove.

Short inspection (or simulation) of (4) reveals that the energy estimate is indeed dominated by the state (m, ℓ) with the smallest energy. However, as we will discuss later, it is typically dangerous to assume that the observed energies are equal to the minimum among the estimated energies for different states (m, ℓ). The reason for this is that the predictors are inherently noisy, and the more states we consider, and the more predicted variability across the states we find, the more likely it becomes that the wrong minimum energy state will be picked with a dramatically wrong predicted energy value. Taking more states into account in the estimate, on the other hand will lead to more robust estimates.

Parameter estimation and binding configuration inference. In our training and testing procedures, we assume that the data is given in a form of a list of triples, each consisting of an MHC class II sequence $\mathbf{s}$, a peptide $\mathbf{s}$ and the measured binding energy $E(\mathbf{s}, \mathbf{e})$. During training, we wish to determine the model parameters w, phi, d_{thr}, a which minimize the error of approximation in (4). Any number of optimization or search algorithms can be used for this. Since the error of approximation in (4) depends on the parameters in a highly nonlinear way, in our implementation, we introduce new auxiliary variables for each training case, in order to simplify the optimization criterion into a simple quadratic form. The price to pay is the EM-style iteration the parameter optimizations step with re-estimation of the case-specific auxiliary variables. To derive the algorithm, we first introduce an auxiliary probability distribution over states $q(m,\ell)$, $0 \leq q(m,\ell) \leq 1$, $\sum_{m,\ell} q(m,\ell) = 1$. As log is a concave function, we have

$$\begin{aligned} E(\mathbf{s},\mathbf{e}) =& - \log \sum_m \sum_{\ell=1}^{N-8} q(m,\ell) \frac{e^{-E(m,\mathbf{s},\mathbf{e},\ell)}}{q(m,\ell)} \geq - \sum_m \sum_\ell q(m,\ell) \log \frac{e^{-E(m,\mathbf{s},\mathbf{e},\ell)}}{q(m,\ell)} \\ =& \sum_m \sum_\ell q(m,\ell) E(m,\mathbf{s},\mathbf{e},\ell) + \sum_m \sum_\ell q(m,\ell) \log q(m,\ell). \end{aligned}$$

Since for a given state m, ℓ, the energy depends on each subset of model parameters w and ϕ linearly, this bound on the energy is also bi-linear in model parameters, and the same iterative linear regression reported in our previous work can be used to minimize the approximation error. The above bound is true for any auxiliary probability distribution q, but it becomes tight (exact equality is accomplished) when

$$q(m,\ell) = \frac{e^{-E(m,\mathbf{s},\mathbf{e},\ell)}}{\sum_{m,\ell} e^{-E(m,\mathbf{s},\mathbf{e},\ell)}}, \tag{5}$$

i.e., the distribution q is the exact distribution over states according to the energy model. This distribution depends on the sequence content of both the MHC molecule $\mathbf{s}$ and the peptide $\mathbf{e}$, and so it has to be recomputed for each training or test case. It is important to note that this distribution is not treated as a distribution over a variable with mutually exclusive and exhaustive states, but rather as population frequencies in the thermodynamics sense. In the former case, the hidden shift variable could only be inferred from a given binding energy, and in prediction, energies of different possible shifts would have to be averaged. In the latter case, the distribution over shifts depends on the predicted energies for individual shifts, and not on the observed energies, and so it can be equally used in training and testing. To learn the model parameters, the configuration inference step has to be iterated with re-estimation of model parameters. Such an iterative learning algorithm consists of the following steps:

– Initialize model parameters (e.g., setting all weight w to one, $d_t hr$ and a so that the step function h is smooth and has a larger threshold, e.g. 6 or 7, and the ϕ matrix to either uniform or the one previously estimated for other purposes.
– Initialize $q^t(m,\ell)$ to uniform for each training sample $(\mathbf{e}^t, \mathbf{s}^t, E^t)$.
– Re-estimate the model parameters w, ϕ, d_{thr}, a so that $\sum_t (E(\mathbf{e}^t,\mathbf{s}^t) - E^t)^2$ is minimized, where

$$E(\mathbf{e}^t,\mathbf{s}^t) = \sum_{m,\ell} q^t(m,\ell) E(m,\mathbf{s}^t,\mathbf{e}^t,\ell) + \sum_{m,\ell} q^t(m,\ell) \log q^t(m,\ell). \tag{6}$$

Since the model is linear in w and linear in ϕ, iterative linear regression to solve for one set of parameters at a time is efficient. Step function parameters d_{thr}, a are updated every few steps by gradient descent.
– Using the new parameters, re-estimate the distribution

$$q^t(m,\ell) = \frac{e^{-E(m,\mathbf{s},\mathbf{e},\ell)}}{\sum_{m,\ell} e^{-E(m,\mathbf{s},\mathbf{e},\ell)}}. \tag{7}$$

– Iterate the last two steps until convergence.

This procedure has some similarity with transformation-invariant generative models developed primarily for vision applications in [17]. However, the important difference is that the possible shifts are not considered as equally likely a priori. In fact, they depend on the peptide and MHC sequences. Consequently, the distribution over states m, ℓ can be determined both for training and testing peptides, and in prediction, the state energies are not averaged. Rather, the possible binding configurations are considered as an ensemble with population frequencies defined by q. It is also different form the LP approach discussed in the introduction, which tries to infer a single best alignment for each peptide in training.

Using temperature to account for modeling errors during learning. The update of the position distribution in (7) and the estimate of the energy in (6) are highly sensitive to the errors in prediction due to the non-linearity of estimating the equivalent energy by summing over all configurations (4). This can cause local minima problems for the EM-like procedure described in the previous section, as the parameters, and therefore the predictions, are less reliable in the early iterations of learning.

To illustrate how the prediction errors may be propagated through (4), we present the following simple experiment. Assuming the total number of different shifts ℓ is 10, and that the true binding energy for fake MHC-peptide configurations $E_\ell$are drawn randomly form a uniform distribution on the interval $[0, 10]$, we computed total binding energies according to $E_{true} = -\log \sum_\ell e^{-E_\ell}$ for 100 such configurations. Then, we computed

$$E_{estimate} = -T \log \sum_\ell e^{-\frac{\tilde{E}_\ell}{T}}, \tag{8}$$

where $\tilde{E}_\ell = E_\ell + v_\ell$, and v_ℓ, a random variable drawn from a zero mean Gaussian distribution with some variance σ^2, simulates a modeling error. A choice of the auxiliary temperature parameter $T > 1$ leads to smoothing of the energy estimate in the following sense: By reducing the differences between the energies of different states, it becomes possible for more states to significantly influence the estimate. This is potentially useful as the wrong state may have the lowest energy due to the prediction errors, and the state with the lowest energy dominates the estimate at $T = 1$. For larger parameter T, on the other hand, the lowest energy state would contribute more to the estimate of the energy, but the other states would contribute, as well.

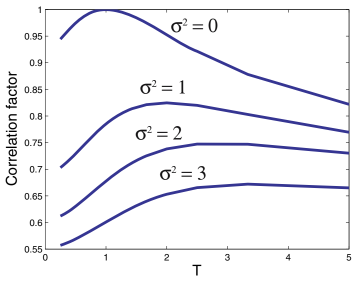


Fig. 2. The effect of the temperature T used in energy estimate (8) on the prediction accuracy, here measured in terms of the correlation between the estimate and the "true" energy in the synthetic experiment described in the text. The curves correspond to the variance of the modeling error σ^2 of 0,1,2, and 3. Higher error variance leads to lower Spearman correlation factors, and the best correlation is achieved at optimal temperatures which increase with the error variance.

We assume for the moment that the measurement procedure which would in practice provide a direct measurement of E_{true} is perfect, and that a potential inability of a predictor to match it is only due to predictor's errors in predicting the binding energy of the groove-peptide segment configurations for different shifts. In Figure 2 we show how well the tempered prediction $E_{estimate}$ using the noisy predictions E_ℓ correlate with the true energies E_{true}. In particular, for different levels of error variance σ^2, we show how the Spearman correlation factor between E_{true} and $E_{estimate}$ varies with the temperature T. The graph shows that a rise in modeling error σ^2 can, to some extent, be absorbed by raising temperature factor T.

Adding the temperature factor into (4) leads to the following change in (6) and (7) in the algorithm of the previous section:

$$E(\mathbf{e}^t, \mathbf{s}^t) = \sum_{m,\ell} q^t(m,\ell) E(m, \mathbf{s}^t, \mathbf{e}^t, \ell) + T \sum_{m,\ell} q^t(m,\ell) \log q^t(m,\ell). \tag{9}$$

$$q^t(m,\ell) = \frac{e^{-\frac{E(m,\mathbf{s},\mathbf{e},\ell)}{T}}}{\sum_{m,\ell} e^{-\frac{E(m,\mathbf{s},\mathbf{e},\ell)}{T}}}. \tag{10}$$

In training, rather than annealing the temperature according to some fixed training schedule, we *search* for the optimal temperature parameter after every few updates of the model parameters. Upon convergence of all model and auxiliary parameters, the temperature typically settles to a value close to 1, which might indicate that the physical measurement errors are higher than the modeling errors.

4 Experiments

We downloaded the complete set of MHC class II structures that contain an epitope of at least seven amino acids from the pdb [13]. The resulting set consisted of 12 HLA-DR and 3 HLA-DQ. Although the MHC class II allele HLA-DP are missing from this set, they share relatively high sequence similarity with HLA-DR alleles. These structures are used as exemplars m of the groove structures in the experiments. To evaluate the prediction accuracy, we used our method both as an epitope predictor and a binding energy predictor and tested it on the available epitope and energy data. In addition to comparisons with existing techniques for epitope prediction, we analyze the ability of our model to assist in association studies in immunology.

4.1 Energy and Epitope Prediction Experiments on Published Data Sets and Comparisons with other Methods

MHCPEP Dataset. The MHCPEP data set has recently been used to evaluate the performance of the MHC class II binding predictors *DistBoost* and RANKPEP. Following the procedure of [1] and [2], we downloaded the contents of the MHCPEP database [6] in order to compare the relative performance of our method. The data are peptide sequences paired with MHC alleles and binding affinities. As in [1] and [2], we removed all peptides classified as low binders or with unknown residues at some position. We removed peptides from all non human MHC alleles (although our method can be applied to these as well), leaving 1265 peptides from 17 MHC class II alleles. We verified via email correspondence that our data set matched the corresponding subset of [1]. Unlike [1] and [2] our method does not require an alignment step and was therefore omitted.

We compared our method to *DistBoost* and RANKPEP [2] by replicating the exact same experimental setup. The MHCPEP data set described above was used as the set of positive binders. Non-binders were taken from random protein sequence from the SwissProt database, so that there were twice as many non-binders as binders per allele. Training was performed using half of the binders for each allele with twice as many non-binders. Testing was performed on the remaining set. We used 5-fold cross validation over the training set to find an optimal set of parameters, and then evaluated the method on the test set. This setup was repeated 10 times to measure average performance and standard deviation.

We plotted ROC curves for our model and compared the AUC of our method with the published results of RANKPEP and *DistBoost*. Our method outperformed both *DistBoost* and RANKPEP on 15 out of the 17 data sets (p-value .00014 binomial) see Table 1. The average AUC for our method was .87 compared to .78 for *DistBoost* and .71 for RANKPEP. In addition, our average standard deviation was lower than either method, 0.04 compared to 0.044 and 0.05, showing our method is as robust or better.

MHCBench Dataset. The MHCBench dataset was constructed for the purpose of evaluating MHC class II binding predictors. Recently, [5] and [3] have evaluated their methods over this dataset after training on similar training data. In order to evaluate the relative performance of our method, we followed their training and testing procedures. We downloaded the set of HLA-DRB1*0401 binding peptides from the SYFPEITHI [12] database that were added before

1999. [3] does not require negative training examples for his method, so we followed the example of [5] and added the HLA-DRB1*0401 non-binders from the MHCBN database [7]. We followed their example and removed peptides that have a hydrophobic residue in the first position according to their model. Peptides that were more than 75% alanine were also removed. This left a dataset of 462 binding and 177 non-binding peptides and is the training data set. Our method also has the capability to incorporate information from other alleles in training. We therefore created another training data set which consists of that described above in addition to the set of non HLA-DRB1*0401 peptides contained at MHCBN. All peptides overlapping the test data (see below) with alignment over 90% were removed, leaving a set of 2997 peptides.

The test data sets used by [5] and [3] consist of the 8 data sets described in [9], the data set from [10], and the data set from [11]. In the [9] data set, any peptide with a non-zero value is considered a binder and is a non-binder otherwise. For the other data sets, any peptide with affinity of less than 1000nM was considered a binder, and a non-binder otherwise. Since there is a significant overlap between the peptides in the training and test data sets, we removed any peptide with $> 90\%$ sequence identity to a peptide in the training set. We verified via email correspondence that our training and test data sets matched those of [5] and [3].

Table 1. Comparison of RANKPEP,*DistBoost*, and our Shift Invariant Double Threading (SIDT) method over the MHCPEP data set. Best values shown in bold font. Columns A and B for *DistBoost* refer to training without and with negative constraints. Columns A and B for RANKPEP refer to PSSMs constructed using PROFILEWEIGHT and BLK2PSSM.

Allele	$RANKPEP_A$	$RANKPEP_B$	$\mathit{DistBoost}_A$	$\mathit{DistBoost}_B$	SIDT	#
QA10501x0201	0.87	0.88	**0.93**	**0.93**	0.87	31
QA10301x0302	0.7	0.7	0.75	0.77	**0.87**	52
PA10201x0901	0.8	**0.88**	0.75	0.74	**0.88**	18
RB10101	0.74	0.75	0.81	0.8	0.**87**	188
RB10102	0.72	0.72	0.9	0.83	**0.91**	21
RB10401	0.68	0.6	0.71	0.73	**0.87**	321
RB10402	0.7	0.72	0.74	0.69	**0.88**	72
RB10405	0.76	0.82	0.86	0.86	**0.89**	64
RB10404	0.7	0.61	0.74	0.7	**0.84**	44
RB10701	0.71	0.72	0.79	0.76	**0.89**	81
RB10901	0.8	0.78	0.89	0.91	**0.97**	39
RB11101	0.57	0.54	0.76	0.73	**0.85**	124
RB11501	0.61	0.6	0.73	0.75	**0.87**	35
RB50101	0.83	0.81	0.83	0.8	**0.87**	52
RB10801	0.52	0.52	0.67	0.65	**0.84**	42
RB11104	0.91	**0.92**	0.87	0.88	0.83	29
RB10301	0.54	0.52	0.54	0.62	**0.83**	52
AVERAGE	0.72	0.71	0.78	0.77	**0.87**	74.4

Table 2. Performance of our shift invariant double threading method (SIDT), the Gibbs sampler, TEPTITOPE, and the Linear Programming method over 10 homology reduced data sets. *This is our method trained with additional data for different alleles. It demonstrates the ability of our method take advantage of information across alleles.

Method	Set1	Set2	Set3a	Set3b	Set4a	Set4b	Set5a	Set5b	Geluk	Southwood	Average
SIDT	**0.76**	0.71	**0.73**	**0.79**	**0.77**	0.72	0.71	0.79	**0.78**	0.61	**0.737**
SIDT*	0.75	**0.73**	0.72	0.74	**0.77**	**0.73**	**0.83**	**0.85**	**0.78**	0.69	**0.759**
Gibbs	0.68	0.66	0.6	0.69	0.67	0.68	0.59	0.59	0.69	**0.88**	0.673
Tepi	0.6	0.65	0.6	0.7	0.59	0.66	0.66	0.68	0.66	0.49	0.629
LP2	0.67	0.7	0.67	0.76	0.65	0.7	0.73	0.76	0.66	0.84	0.714

Table 3. Description of data sets used in this paper. Train is the training set used for the MHCBench test set. Train2 is the same training set with the addition of peptides belonging to different alleles.

	Set_1	Set_2	Set_{3a}	Set_{3b}	Set_{4a}	Set_{4b}	Set_{5a}	Set_{5b}	So.	Gel_1	Trn	Trn_2	PEP
Bind	248	161	151	128	120	120	65	47	19	15	462	1782	1037
Non	283	255	204	197	283	255	45	37	80	6	177	121	2074
Total	531	416	355	325	403	375	110	84	99	21	639	2997	3111

We used 5 fold cross validation over the training set to estimate the optimal set of parameters for our model. ROC curves were generated for each test set and the AUC was computed for comparison with the published results of LP, Gibbs, and Tepitope. In addition, we trained on another training data set which contained peptides from other alleles to show how our method can incorporate other data to improve performance. The results are shown in Table 2. Our method has a higher average ROC than any other method, and it is further improved by adding non DRB1*0401 alleles to the training set. We beat the other methods on 8 out of 10 data sets (pvalue ¡ 0.017 binomial). In training our model we assume a different cutoff for good versus bad binders than the 1000 nM cutoff used for the Southwood and Geluk data sets in the test data. Using our cutoff of $e^{6.2}$ improves our performance on these data sets, but can not be compared with the above methods since the training set would be different.

4.2 Generalizing to New Alleles

One of the important features of our approach as opposed to most others is that after training, any MHC sequence may be threaded onto a structure and used for binding prediction[1]. This allows us to predict peptide binding for alleles with little or no experimental data. For MHC class I molecules there are hundreds of alleles. MHC class II molecules are polymers of two different molecules called the alpha and beta chains. HLA-DQ has several hundred alpha and beta chains, with thousands of possible combinations, each of which binds different peptides. Since

[1] Some techniques attempt something similar. For example, TEPITOPE learns individual binding pockets, allowing it some level of generalization.

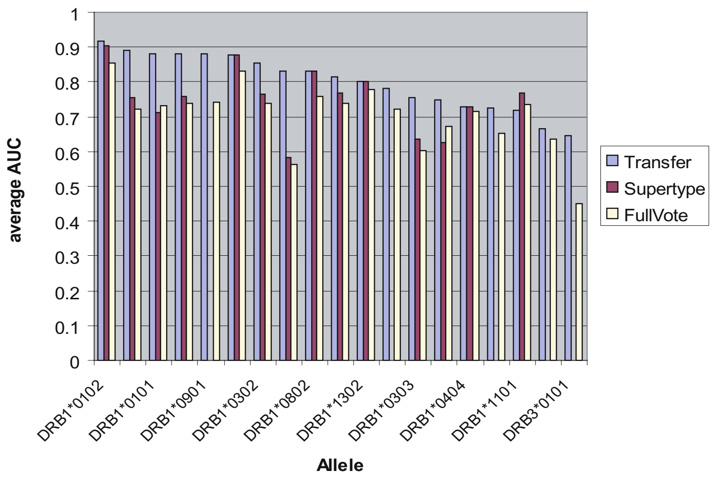


Fig. 3. The capability of generalizing epitope prediction to alleles not found in the training set allows our method to be applied to a much larger set of MHC molecules. This figure shows the significantly greater predictive power of our method over two voting based mechanisms for binding across alleles.

peptide binding experiments are currently costly and time consuming,the ability to predict binding for unseen alleles is an extremely useful feature of our method.

The IEDB [8] is a meticulously curated data set of peptide binding data-database. This resource maintains a hand curated list of epitopes, and carries continuous IC50 values. We downloaded the complete IEDB MHC and TCell binding data from IEDB, removing peptides from before 1993, and any peptide marked as a good binder with an IC50 of greater than 3000 and any peptide marked as a non-binder with IC50 less than 500. In order to guarantee an equal number of binding and non-binding peptides in each allele set, we added random human peptides from SwissProt until each allele was balanced. This data is described at `http://www.research.microsoft.com/jojic/hlaBinding.html`

Using the IEDB database described above, we created transfer data sets by removing all epitopes of each allele in turn. For each of these data sets, we trained the model using 5 fold cross validation to estimate the optimal parameters. We then threaded the MHC sequence of the allele that was left out onto the structure of the allele that had the closest sequence alignment. We then ran the model using this sequence structure combination over all of the alleles from the data set. Since there is significant overlap between peptides that bind to different alleles, we compared our transfer results to two different voting based methods for predicting binding of unseen alleles. We ran our standard trained model for all observed alleles in the training data over the set of peptides of the unobserved allele. We called a peptide a binder if the majority of the alleles called it a binder. In another voting setup, we called a peptide a good binder if a majority of the alleles in the supertype of the left out allele called it an good binder. We plotted a ROC curves for the performance of each method and calculated their average AUC. The results are show in Figure 3. As can be seen in the figures, our threading method significantly (p-value $< .00001$ binomial) outperforms either voting mechanism. We are able to predict peptide binding for MHC class II alleles having learned over both alpha and beta chains, a single alpha or beta chain, or without any previous exposure to either chain of the allele.

4.3 Myelin Binding

There are several auto-immune diseases in which the nerve insulating material called myelin is degraded. This degradation disrupts signal passage through the nervous system and can cause severe health problems. Myelin Basic Protein (MBP) has been shown to bind to the MHC class II allele HLA-DRB1*1501, and is a candidate autoantigen for multiple sclerosis (MS), an auto-immune disease of the central nervous system. We demonstrate how our MHC class II binding predictor can be used in autoimmune research by replicating several MS experimental results in silico. The HLA-DR2 supertype has been repeatedly shown to positively associate with MS [15]. We ran our method over the MBP using the HLA-DR2 allele HLA-DRB1*1501 and found four potential binders. Of these, the strongest signal was located at amino acid 91 of the MBP. The peptide consisting of residues 85-99 which contains our predicted binding site has been shown experimentally to be an immunodominant epitope for HLA-DRB1*1501 [14]. Furthermore, there is an approved drug to treat certain forms of MS that works by disrupting this binding, and there is active research to find new candidate peptides that will displace MBP 85-99 by competitively binding to the HLA-DRB1*1501 allele. These drugs have been shown to suppress relapse rates of certain forms of MS by 30% [14]. The drug and two other competitive binding peptides take the form of coplymers 1 poly(Y,E,A,K)n, 2 poly(F,Y,A,K)n, and 3 poly(V,W,A,K)n. These are peptide sequences of random combinations of each the amino acids inside the in the poly groups. We measured the number of predicted binders to HLA-DRB1*1501 over 20 random peptides of each of these polymers and found that in 20 polymers of length 50, there were 60, 80, and 155 predicted binders with a binding strength greater than that predicted for MBP 85-99, for polymers 1, 2, and 3 respectively. When 20 random SwissProt proteins of equivalent length were used, there were only 10 predicted stronger binders. This shows our method predicts the potential therapeutic uses of these coplymers. Recently, [14] examined the properties of the copolymers and synthesized non-random peptides of length 15. Three of these J2, J3, and J5 were experimentally found to suppress MBP 85-99 binding with the relative strength of suppression $J5 > J3 > J2$. We ran our method over each of these 15 amino acid long peptides and found that all three had predicted binding energies lower than MBP 85-99 (they form stronger bonds). Furthermore, the order of binding strength matched that of the relative levels of suppression. That is, J5 was the strongest binder followed by J3 and then J2.

5 Conclusion

We have developed a novel MHC class II binding model which can be trained on examples of measured binding affinities for a number of allele-peptide combinations, as well as lists of good and bad binders for various alleles.[2] To the best of our knowledge, our method outperforms significantly all previously published class II epitope prediction techniques, due to its unique treatment of the variable position of the peptide with respect to the binding groove. Our method is physics-based, and treats the binding configurations with different possible peptide positions as a statistical ensemble in a thermodynamic sense. However, as opposed to other structure-based techniques [4], our approach is both accurate in binding energy prediction *and* computationally efficient. For instance, due to the computational cost, [4] reports results for only six peptides. Our model, while guided by the known MHC II structures, is simplified and enriched with trainable parameters, which allows us to refine it using published binding data. Testing a new peptide takes a fraction of a second. One of the most appealing properties of our technique is that it naturally generalizes well to previously unseen MHC II alleles (or unseen combinations of alpha and beta chains). We illustrated the accuracy of our technique on a biological problem: identifying targets and drugs for an autoimmune disorder. We are also investigating the uses of the model to explain certain evolutionary trends in pathogens.

References

1. Hertz T, Yanover C. PepDist: a new framework for protein-peptide binding prediction based on learning peptide distance functions. BMC Bioinformatics. 2006 Mar 20;7 Suppl 1:S3.
2. Reche PA, Glutting JP, Zhang H, Reinherz EL. Enhancement to the RANKPEP resource for the prediction of peptide binding to MHC molecules using profiles. Immunogenetics. 2004 Sep;56(6):405-19. Epub 2004 Sep 3.
3. Nielsen M, Lundegaard C, Worning P, Hvid CS, Lamberth K, Buus S, Brunak S, Lund O. Improved prediction of MHC class I and class II epitopes using a novel Gibbs sampling approach. Bioinformatics. 2004 Jun 12;20(9):1388-97. Epub 2004 Feb 12.
4. Davies MN, Sansom CE, Beazley C, Moss DS. A novel predictive technique for the MHC class II peptide-binding interaction. Mol Med. 2003 Sep-Dec;9(9-12):220-5.
5. Murugan N, Dai Y. Prediction of MHC class II binding peptides based on an iterative learning model. Immunome Res. 2005 Dec 13;1:6.
6. Brusic V, Rudy G, Harrison LC. MHCPEP, a database of MHC-binding peptides: update 1997. Nucleic Acids Res. 1998 Jan 1;26(1):368-71.
7. Bhasin M, Singh H, Raghava GP. MHCBN: a comprehensive database of MHC binding and non-binding peptides. Bioinformatics. 2003 Mar 22;19(5):665-6.
8. Vita R, Vaughan K, Zarebski L, Salimi N, Fleri W, Grey H, Sathiamurthy M, Mokili J, Bui HH, Bourne PE, Ponomarenko J, de Castro R Jr, Chan RK, Sidney J, Wilson SS, Stewart S, Way S, Peters B, Sette A. Curation of complex, context-dependent immunological data. BMC Bioinformatics. 2006 Jul 12;7:341.
9. MHCBench http://www.imtech.res.in/raghava/mhcbench
10. Southwood S, Sidney J, Kondo A, del Guercio MF, Appella E, Hoffman S, Kubo RT, Chesnut RW, Grey HM, Sette A. Several common HLA-DR types share largely overlapping peptide binding repertoires. J Immunol. 1998 Apr 1;160(7):3363-73.

[2] In the latter case, good binders are given low and bad binders high nominal energy.

11. Geluk A, van Meijgaarden KE, Schloot NC, Drijfhout JW, Ottenhoff TH, Roep BO. HLA-DR binding analysis of peptides from islet antigens in IDDM. Diabetes. 1998 Oct;47(10):1594-601.
12. Hans-Georg Rammensee, Jutta Bachmann, Niels Nikolaus Emmerich, Oskar Alexander Bachor, Stefan Stevanovic: SYFPEITHI: database for MHC ligands and peptide motifs. Immunogenetics (1999) 50: 213-219 (access via : www.syfpeithi.de)
13. H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, P.E. Bourne: The Protein Data Bank. Nucleic Acids Research, 28 pp. 235-242 (2000).
14. Stern JN, Illes Z, Reddy J, Keskin DB, Fridkis-Hareli M, Kuchroo VK, Strominger JL. Peptide 15-mers of defined sequence that substitute for random amino acid copolymers in amelioration of experimental autoimmune encephalomyelitis. Proc Natl Acad Sci U S A. 2005 Feb 1;102(5):1620-5. Epub 2005 Jan 21.
15. Krogsgaard M, Wucherpfennig KW, Cannella B, Hansen BE, Svejgaard A, Pyrdol J, Ditzel H, Raine C, Engberg J, Fugger L. Visualization of myelin basic protein (MBP) T cell epitopes in multiple sclerosis lesions using a monoclonal antibody specific for the human histocompatibility leukocyte antigen (HLA)-DR2-MBP 85-99 complex. J Exp Med. 2000 Apr 17;191(8):1395-412. Erratum in: J Exp Med. 2003 Apr 7;197(7):947. Canella B [corrected to Cannella B].
16. Bailey TL, Williams N, Misleh C, Li WW. MEME: discovering and analyzing DNA and protein sequence motifs. Nucleic Acids Res. 2006 Jul 1;34(Web Server issue):W369-73.
17. Nebojsa Jojic and Brendan Frey, "Topographic transformation as a discrete latent varaible," Neural Information Processing Systems (NIPS) '99,November 1999, Denver, CO.
18. Nebojsa Jojic, Manuel Reyes-Gomez, David Heckerman, Carl Kadie, and Ora Schueler-Furman Learning MHC Ipeptide binding Bioinformatics 2006 22: e227-e235;
19. O. Schueler-Furman, Y. Altuvia, A. Sette and H. Margalit, "Structure-based prediction of binding peptides to MHC class I molecules:Application to a broad range of MHC alleles," Protein Science (2000) 9:1838-1846.
20. D.T. Jones, W.R. Taylor, and J.M. Thornton,"A new approach to protein fold recognition," Nature (1992) 358:86-89.
21. F. Melo, R. Sanches, and A. Sali, "Statistical potentials for fold assessment," Protein Science (2002) 11:430-448.

Reconstructing the Phylogeny of Mobile Elements

Sean O'Rourke[1], Noah Zaitlen[2], Nebojsa Jojic[3], and Eleazar Eskin[4]

[1] Dept. of Computer Science and Engineering, University of California, San Diego, La Jolla, CA 92092
sorourke@ucsd.edu
[2] Dept. of Bioinformatics, University of California, San Diego, La Jolla, CA 92092
nzaitlen@ucsd.edu
[3] Microsoft Research, 14870 NE 31st Way, Redmond, WA 98052
jojic@microsoft.com
[4] Dept. of Computer Science, Dept. of Human Genetics, University of California, Los Angeles, CA 90095
eeskin@cs.ucla.edu

Abstract. The study of mobile element evolution yields valuable insights into the mechanism and history of genome rearrangement, and can help answer questions about our evolutionary history. However, because the mammalian genome contains millions of copies of mobile elements exhibiting a complex evolutionary history, traditional phylogenetic methods are ill-suited to reconstructing their history. New phylogenetic reconstruction algorithms which exploit the unique properties of mobile elements and handle large numbers of repeats are therefore necessary to better understand both mobile elements' evolution and our own.

We describe a randomized algorithm for phylogenetic reconstruction that scales easily to a million or more elements. We apply our algorithm to human and chimpanzee *Alu* and L1 elements, and to SINE elements from 61 species, finding 32 new L1, 111 new SINE, and over 1000 new *Alu* subfamilies. Our results suggest that the history of mobile elements is significantly more complex than we currently understand.

1 Introduction

Nearly half our genome is the result of the activity of mobile elements, short sequences originating as exogenous viruses which copy and reinsert themselves into our genetic code. Of these mobile (or repeat) elements, a fifth are evolutionarily modern, found only in primates, and a few are still active today. Analysis of these mobile elements can help answer organism-level questions of primate phylogeny and human population structure [1]. Furthermore, a detailed picture of mobile element evolutionary history is crucial to understanding their role in genome rearrangement, protein evolution, and some genetic diseases [2,3]. Fundamental problems therefore include estimating the number and size of mobile element families, the relations between them, and their distribution over species.

Repeat element phylogeny is difficult for at least three reasons: First, a single major repeat family can contain over 1 million elements, presenting a computational challenge for traditional phylogeny methods. Second, the families exhibit complex evolutionary history, with a large number of related and highly similar subfamilies. Third, because many of the mobile elements are ancient and are located in non-coding regions, they are very degenerate due to mutation; it is not uncommon for a mobile element to have mutated in over 20% of its positions since insertion. Furthermore, truncation and partial insertion leave only fragments of most instances of the longer families. Although tools such as RepeatMasker [4] exist to identify repeat elements in the genome, few computational tools exist to identify repeat element subfamilies and construct their phylogeny.

The spread of repeat elements and their evolutionary and developmental roles have long aroused scientific interest, starting with McClintock's work on coloring differences in maize in the 1950's [5]. Much recent work has focused on the role and dynamics of the *Alu* repeat in humans and other primates. Eichler *et al.* [6] investigate *Alu*'s potential role in producing the relatively abundant segmental duplications seen in the human genome. Mishra *et al.* [7] propose that LINE-1 elements mediate some genome rearrangement in rat, as *Alus* do in primates. Han *et al.* [8] show that LINE elements may also play a role in genomic deletions in chimpanzee and human. Hedges *et al.* [3] review the putative role of several types of mobile element in genome growth, protein evolution, and human disease. Jurka [9] reviews the dynamics of *Alu* insertion and deletion, and discusses L1-mediated retrotransposition as a mechanism for their duplication.

Recent work has focused not just on the role of mobile elements in genomic evolution, but on the phylogeny of the elements themselves. Cordaux *et al.* [10] reconstruct a phylogenetic network over the *Alu Y* subfamily, showing that human *Alu* elements came from multiple active sources. Price *et al.* [11] reconstruct a phylogeny of all human *Alu* elements using a novel clustering method based on tests for correlated mutation. Salem *et al.* [12] use *Alu Ye* elements to provide evidence about species relations among human, chimpanzee, and gorilla. Their repeat phylogeny, reconstructed by traditional methods [13], gives evidence that hominids are monophyletic and more closely related to chimpanzee than to gorilla. Hedges *et al.* [3] argue that organism-level phylogenies should be used to provide further insight into mobile element population dynamics.

In this paper, we describe a method for recovering the most likely phylogeny of observed instances. Traditional phylogenetic methods are not appropriate for at least three reasons: First, since most are quadratic or worse in the number of sequences, they do not scale to the million or more repeat sequences we wish to analyze. Second, unlike in the case of species phylogeny, only a few repeat elements in the genome actively replicate, while the vast majority merely persist with gradual mutation. For this reason, repeat phylogenies are characterized by a few nodes in the tree each having a very large number of offspring and most nodes having none. Finally, since repeat subfamilies are highly overlapping in sequence space, the results of distance-based methods are uninformative (see "Why distance-based clustering fails" in Section 2.2).

Our method divides mobile elements into subfamilies via a novel clustering algorithm, Randomized Test and Split (RATS). The algorithm uses a statistical test of subfamily validity to recursively partition the set of elements, while ensuring that the final clustering is statistically well-motivated. A traditional approach to the subfamily identification problem would formulate a generative model for the data and apply the Expectation Maximization (EM) algorithm. However, due to the tremendous number of mobile elements and the fact that the subfamilies are closely related to each other, this is impractical due to slow convergence and numerous poor local optima. Using simulated data, we demonstrate that RATS approximately recovers the EM partition in a fraction of the time.

Our method is based on the following three-phase algorithm from Price *et al.* [11]: (1) Repeatedly compute the correlation between amino acids at every pair of positions in each subfamily and split it into two new subfamilies if any pair of positions fails a statistical test for independence. (2) When no such family exists, further split these initial subfamilies based on single-position deviations from a molecular clock estimated from the initial subfamilies. (3) Construct a minimum spanning tree over the subfamilies' consensus sequences.

We extend and improve upon this approach in four ways. First, we test a random subset of pairs of positions for correlation rather than testing all of them (Section 2.2). This reduces the time and space complexity from quadratic to linear in repeat sequence length, allowing us to analyze significantly larger data sets and longer elements. Second, we incorporate more partial sequence fragments. Together, these advances allow us to extend our analysis to longer families of repeats such as L1. Third, we apply a stronger statistical test between pairs to recover more subfamilies (many of Price's novel subfamilies were found not by correlation tests, but by single-position splits). Finally, we relate our statistical test to the (approximate) optimization of an underlying generative model, formalizing the assumed repeat generation process.

We apply RATS to repeat elements found in the ENCODE project database [14], and to repeats from the full genomes of human and chimpanzee, finding 32 new L1, 111 new SINE, and over 1000 new *Alu* subfamilies. We analyze the phylogeny of the SINE subfamilies, demonstrating its agreement with species phylogeny and with currently known repeat subfamilies.

2 Methods

Our goal is to recover the most likely phylogeny of the observed instances. Rather than directly computing a phylogeny over all individuals, we approach the problem in two independent steps, first identifying the most likely subfamilies and their (implicit) source elements, then constructing the best phylogeny of these predicted subfamilies. The phylogeny of individual elements is then specified by this subfamily phylogeny and the individuals' subfamily memberships, with all individuals in a subfamily being offspring of the subfamily source element. We present our generative model of mobile element replication in Section 2.1. We then discuss the subfamily identification problem and an efficient algorithm to solve it in Section 2.2, and finally discuss the simpler problem of subsequently constructing the phylogeny Section 2.3.

2.1 Subfamily Generation Model

In our model, mobile elements replicate by asexual reproduction and we assume a neutral mutation rate. Each individual sequence z generates a copy of itself at time t with probability $p_c(z,t)$, and each site mutates with some probability $p_m(z,t)$. However very few individuals create copies, and these copies are all created over a relatively short period of time. In other words, $p_c(z,t)$ is zero almost everywhere, and large relative to $p_m(z,t)$ where it is not. We can therefore make two simplifications: First, since almost no mutation occurs between copies of a single active element, we assume that all copies are initially identical. Second, since extant (inactive) individuals are simply the result of neutral mutation from these identical copies, we assume that a subfamily's elements are uniformly distributed (in sequence space) around its source element.

These considerations lead to the following view of mobile element generation: Each subfamily consists of a large number of inactive copies of a single active source element. (Source elements which themselves mutate are considered new distinct source elements if they are still active.) Each copy or instance undergoes independent point mutation over time, diverging from the sequence of the original source element. Each instance can, with some small probability, itself become a source element and generate its own distinct subfamily. Viewed as clusters of points in sequence space, the subfamilies form a highly-overlapping set of spheres whose radii reflect their ages.

More precisely, let $\Sigma = \{a_1, \ldots, a_m\}$ be an m-letter alphabet, and let $X = (X_1, \ldots, X_l)$ be a vector random variable over Σ^l. Let C be a scalar random variable over cluster source element indices $\{1, \ldots, k\}$, with $\mu(C) \in \Sigma^l$ being source element C's sequence. The k source elements are used to generate n sequences $Z = \{z\}$ under the following model: First select a cluster $c \in C$ with probability $p(C=c)$ and make sequence z a copy of $\mu(c)$. Then independently mutate each letter z_i to some symbol $s \neq \mu_i(c)$ with a small cluster- and position-dependent probability $r_{is}(c)$. Equivalently, let $p(X_i|C=c)$ define the conditional probability distribution of position i of a sequence in c, and let $\mu_i(c)$ be its consensus value, $\mu_i(c) = \operatorname{argmax}_x p(X_i=x|C=c)$. Then $p(X_i=s|C=c) = r_{is}(c)$ for $s \neq \mu_i(c)$ and $p(X_i = \mu_i(c)|C = c) = 1 - \sum_s r_{is}(c)$. The distribution over sequences X can then be modeled as a mixture of per-cluster distributions:

$$p(X) = \sum_{c \in C} p(C = c) \prod_{i=1}^{l} p(X_i|C = c)$$

2.2 Subfamily Identification

Given an n-element sample Z from $p(X)$, our goal is to recover the number of clusters k and, given k, to find the assignment $p(C|Z)$ of individuals to clusters maximizing the likelihood of the sample. The model's free parameters, representing the cluster probabilities $p(C)$ and discrete distributions of symbols occurring at each sequence position in each cluster $p(X|C)$, assume their maximum likelihood values.

This is an instance of hard clustering, a well-studied problem for which Expectation Maximization (EM) with soft assignment has been shown to perform well in many cases [15]. However, EM is not applicable to our problem for two reasons. First, our dataset is enormous: there are over one million elements in just the single largest repeat family in the human genome (*Alu*), representing a thousand or more distinct clusters. EM clustering, which requires $O\left(kl(m+n)\right)$ operations per iteration (or $\approx 10^{11}$ for $k = 10^3$, $n = 10^6$, and $l = 10^2$), is not computationally feasible at this scale. Second, since the objective function has many poor local optima, EM often requires many random restarts to find a good solution (see Section 3.1), further increasing runtime.

Limiting Model Complexity. Since data likelihood will always increase as the number of clusters increases, we need to control the tradeoff between smaller models and higher likelihood. One common solution to this well-known problem is to add a model complexity penalty to the objective function, with the Bayesian Information Criterion (BIC) being a popular choice [16]. The BIC for M model parameters and N data elements is $\frac{M}{2}\log N$.

However, we find on simulated data that the BIC penalty dominates our model score long before we have recovered the correct number of clusters (Section 3.1). For example, even on the small simulated dataset in Figure 4 with $k = 11$, the penalty is approximately 3×10^4, or almost ten times the improvement in likelihood over the random model. This happens because the BIC for our model, $\frac{k}{2}(1 + l(m-1))\log n$, depends linearly on the number of free parameters, and hence in our case on the sequence length. However, the number of parameters in our model is artificially high because we learn a set of independent clusters for data we assume are generated by a hierarchical model.

Our assumptions from Section 2.1 allow an alternative approach. Since we assume that mutations are independent within a subfamily, if the distribution of mutations at a pair of positions within a single cluster fails a statistical test for independence, we have evidence of further substructure. When no such pair exists, we have evidence of a candidate solution. Therefore instead of directly maximizing the likelihood as we would with EM, we instead search for a solution that satisfies this statistical test.

Specifically, we consider the estimated mutual information between pairs of positions [17]:

$$I(X_i; X_j|c) = \sum_{s_i, s_j \in \Sigma} p(s_i, s_j|c) \log \frac{p(s_i, s_j|c)}{p(s_i|c)p(s_j|c)}$$

Goebel *et al.* [18] show that the mutual information between two uncorrelated discrete random variables can be approximated by a gamma distribution

$$\Gamma\left(\frac{1}{2}(|X_1|-1)(|X_2|-1),\ \frac{1}{N\log 2}\right)$$

where N is the sample size and $|X_i|$ is the number of possible values of X_i. From this approximation we can derive a p-value for independence between two positions $1 \leq i < j \leq l$ and a corresponding threshold α for $I(X_i; X_j|c)$.

Rather than testing all $O(l^2)$ pairs of positions, we randomly sample enough pairs to detect correlation with high confidence. Assume that a single supposed cluster contains members of two true clusters differing in λ of l positions drawn from a binary alphabet, and that we want to test at least one pair of these λ positions with probability $1-p$. Then the probability of choosing a pair of correlated positions in a single draw is $\frac{\lambda(\lambda-1)}{l(l-1)}$ and probability of not detecting correlation after t trials is $p = \left(1 - \frac{\lambda(\lambda-1)}{l(l-1)}\right)^t$ Therefore choosing t to achieve our desired p yields

$$t = \frac{\log p}{\log\left(1 - \frac{\lambda(\lambda-1)}{l(l-1)}\right)}$$

For example, to detect a difference in 4 out of 300 positions with $p = 0.99$, we must sample 343 pairs. Mutation will increase the number of tests required by causing some of the $\lambda(\lambda-1)$ pairs of correlated positions to no longer be significantly correlated.

Fast Randomized Clustering. Conveniently, this same random pair sampling suggests an efficient top-down clustering algorithm. When positions i and j are correlated in cluster c, splitting c into two clusters c_1 and c_2 such that c_1 contains all individuals z with $(z_i, z_j) = \text{argmax}\, p(X_i = z_i, X_j = z_j|c)$, c_2 the rest, will tend to improve $p(Z|C)$. When no such positions exist, further cluster splits cannot significantly decrease the entropy of a cluster at multiple sequence positions at once. We have therefore found a reasonable approximation to the maximum likelihood solution.

While these tests ensure that each pair of split clusters is statistically justified, multiple splits and poor initial splits may create clusters which could be merged without creating correlated pairs. Such problematic pairs can be found by trying to merge sets of clusters and repeating the above correlation sampling. Because testing all subsets of clusters is computationally impossible, we instead test only pairs of clusters with similar consensus sequences.

Recursively applying this cluster splitting criterion yields the iterative algorithm in Figure 1. Starting with a single cluster, iteratively apply the following two steps: First, recursively test and split the current set of clusters until no cluster fails the correlation test. Second, merge neighboring pairs of clusters when doing so does not create detectable correlation. However, the split heuristic will poorly assign some elements, and splitting can only make large-scale changes to the current clustering. We therefore apply an additional fine-grained greedy improvement between split and join, assigning each individual z to the cluster

```
function split($C$)
    repeat $t$ times
        $(i, j) \leftarrow$ random($[1, l]$)
        if $I(X_i; X_j | C = c) \geq \alpha$
            $(s_i, s_j) \leftarrow \text{argmax}\, p(s_i, s_j | c)$
            $C' \leftarrow \{z | z_i = s_i, z_j = s_j, z \in C\}$
            return split($C'$) $\cup$ split($C - C'$)
        end if
    end repeat
    return $\{C\}$
function join($\mathcal{C}$)
    $\mathcal{C}' \leftarrow \{\}$
    while $|\mathcal{C}| > 1$
        $(C_1, C_2) \leftarrow \text{argmin}\, d_H\, (\mu(C_1), \mu(C_2))$
        $\mathcal{C} \leftarrow \mathcal{C} - \{C_1, C_2\}$
        if split($C_1 \cup C_2$) $= \{C_1 \cup C_2\}$
            $\mathcal{C}' \leftarrow \mathcal{C}' \cup \{C_1 \cup C_2\}$
        else
            $\mathcal{C}' \leftarrow \mathcal{C}' \cup \{C_1, C_2\}$
        end if
    end while
    return $\mathcal{C}' \cup \mathcal{C}$
function RATS ($Z$, $n$)
    $\mathcal{C}_0 \leftarrow \{Z\}$
    for $i = 1$ to $n - 1$
        $\mathcal{C}_{i+1} \leftarrow \text{join}\left(\bigcup_{C \in \mathcal{C}} \text{ split}(C)\right)$
    end for
    return $\mathcal{C}_n$
```

Fig. 1. The RATS algorithm. The optional greedy updates are performed between `split` and `join`, and again after the final iteration.

with the closest consensus sequence $\mu(c)$, i.e. $c = \text{argmin}_c \quad d_H(z, \mu(c))$ where d_H is the Hamming distance.

Why Distance-Based Clustering Fails. Consider two populations A and B of bitstrings of length n, where As are instances of $\bar{A} = 0X1$, Bs of $\bar{B} = 1X0$; X represents a sequence of $n-2$ random bits. The distance D_{AA} between an A and $\bar{A}$ follows a binomial distribution $\text{B}(n-2, 0.5)$, while the distance D_{AB} between it and $\bar{B}$ is $2+\text{B}(n-2, 0.5)$. So the probability $p(D_{AA} \leq D_{AB})$ of its being at least as close to $\bar{B}$ as to $\bar{A}$ is approximately $p\left(D_{AA} \leq E[D_{AB}]\right) = I_{0.5}(\frac{n}{2}+1, \frac{n}{2}-2)$, which for $n = 100$ equals 0.38. Since a substantial minority of the elements from each population are closer to the other's consensus sequence, a distance-based phylogeny will misleadingly relate elements from different populations, obscuring the two actual subpopulations. However, testing for correlated mutation can correctly identify $\bar{A}$ and $\bar{B}$.

2.3 Subfamily Phylogeny

Given a set of subfamilies, we next reconstruct the most likely phylogeny. The standard approach to this problem is to assume that the observed sequences are the tree's leaves, then to infer a maximum likelihood binary tree over them according to some mutation model (see e.g. [19]). Our problem is different in three ways: First, our subfamilies represent not just the leaves of the tree, but also its internal nodes, obviating the usual optimization over unobserved ancestral nodes. Second, the inferred $p(X|C)$ defines a fixed mutation model for each cluster. Third, the average divergence within a cluster defines its age, allowing us to constrain our tree so that a node must be younger than its parent.

We assume that the consensus sequence of each subfamily is generated from its parent family, and is therefore drawn from the parent family's distribution, though it does not have to be one of the actual observed members of either. For each cluster $c \in C$, we choose as its parent the most probable ancestor for its consensus sequence $\mu(c)$,

$$\underset{\{c'|\text{age}(c')>\text{age}(c)\}}{\text{argmax}} \quad p(c')p(\mu(c)|c')$$

where age(c) is the estimated age of cluster c. By assuming that mutation rate is constant over the genome at each point in time, we can approximate a subfamily's age by its average mutation rate or (equivalently) its entropy. Although this assumption is clearly not valid for estimating a subfamily's absolute age, our algorithm depends only on the ordering of the age estimates across different subfamilies. Since a genome containing elements of one subfamily will contain elements of all of its ancestors, it is highly unlikely that molecular clock variation will cause a repeat subfamily to appear younger than its ancestors.

3 Results

We first demonstrate that our method performs well relative to EM on simulated data similar to actual repeat elements. We then apply our method to repeat elements collected from over sixty organisms, finding subfamilies and phylogenies of the *Alu*, SINE (an evolutionary superset of *Alu*), and L1 families. We discuss a phylogeny of SINE elements across all ENCODE project species. Our repeat phylogeny's close agreement with known repeat family and species phylogenies validates our approach.

3.1 Simulated Data

A repeat phylogeny algorithm should obey two correctness properties: First, it should be approximately optimal: for any cardinality, the likelihood of the data given the model it finds is close to the optimum likelihood. Second, it should be conservative: the probability of the algorithm finding more than the true number of subfamilies should be acceptably small. Here we show that RATS exhibits these properties on simulated data similar to actual repeat subfamilies.

Each dataset consists of sequences in $\{0,1\}^{300}$ drawn from one of k subfamilies separated by d mutations with a uniform per-position mutation rate u. For each (d,k,u), we generate 5 sets of repeats with these parameters, then run RATS 20 times. We approximate the optimal likelihood for each number of subfamilies found by taking the best of 10 EM solutions.

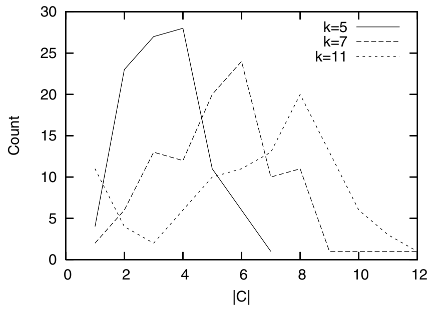


Fig. 2. Frequency of number of subfamilies found for different true numbers of subfamilies

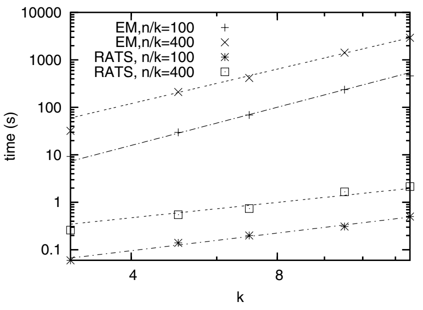


Fig. 3. Running time of the EM algorithm and RATS for different numbers of subfamilies k and different numbers of instances per subfamily n/k

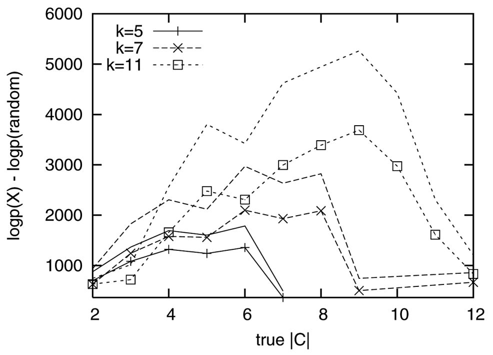


Fig. 4. Log-probability versus random clustering for RATS (lines with marks) and EM (lines without)

Figure 2 shows the distribution of the number of subfamilies found for $d = 6$ and $u = 0.1$ while varying k. As desired, RATS is conservative, finding more than the true number of subfamilies 1–12% of the time. Figure 4 compares the likelihood of the best solutions found by RATS and EM (with 10 restarts) to the average likelihood of a random clustering. (The random model likelihood is used instead of the generating model likelihood because, particularly for larger numbers of subfamilies with fewer members, the generating model can yield significantly lower likelihood than a learned model.) As expected, RATS consistently performs slightly worse than EM, with both approaching the random score as the number of subfamilies diverges from the true number.

Figure 3 shows the time for a single run of each algorithm as a function of the subfamily arity k and subfamily size n/k. Since the two algorithms are implemented in different languages, we cannot directly compare runtimes. However, EM's runtime grows at a faster rate, largely because the number of iterations to convergence grows with k. A least-squares fit of Ak^B to both curves shows EM is at least quadratic in k ($B = 2.7, 2.4$), where RATS is nearly linear ($B = 1.2, 1.1$).

3.2 Data Preparation

We obtained "full" genome sequences for *P. troglodytes* and *H. sapiens* from the most recent sequence builds available at UCSC as of March 1, 2006. We also downloaded accession numbers for orthologous regions of 64 vertebrate species from the NIH Intramural Sequencing Center (NISC) Comparative Sequencing Program (ENCODE) [14], obtaining the corresponding sequence data from GenBank. Sequences were joined together according to the position information specified within the GenBank files.

A data set of repeat elements was created for the complete set of sequences via RepeatMasker version 3.1.3 [4] using a library of repeat elements from RepBase [20], using the lowest sensitivity for the "full" genomes and standard sensitivity for the ENCODE regions. We generated multiple alignments for single repeat classes (e.g. *Alu*, L1) from pairwise alignments of each repeat instance to a single RepBase consensus sequence (e.g *Alu Sx*, HAL1#LINE/L1) as follows: Consensus sequences for repeat classes were created by Clustal W [21] multiple alignment of all constituent RepBase consensus sequences. Eleven SINE consensus sequences that aligned poorly to the others were excluded. Positions corresponding to gaps in the pairwise instance alignments were removed, and the aligned instances were threaded into the repeat class multiple alignment to yield a multiple alignment of instances. Interior gaps were treated as a separate symbol, leading and trailing gaps as missing data.

3.3 Novel Repeat Subfamilies

Table 1 compares the number of subfamilies we find to the number identified in RepBase. To ensure the correlation test's validity, RATS was constrained to test only subfamilies of more than 200 elements for L1, and more than 1000 for Alu

Table 1. Numbers of repeats in various families found in RepBase and by our method. *Alu* and L1 repeats are from the full Human and Chimpanzee genomes, while SINE repeats are from the ENCODE database of orthologous regions.

Family	RepBase	RATS	Elements	Source
Alu	35	1519	2 309 150	primate
L1	102	134	24 249	primate
SINE	197	308	381 248	ENCODE

and SINE. All runs used a p-value of 10^{-3} with a Bonferroni correction for testing multiple positions. The relatively small number of L1 subfamilies discovered may be due to the small number of available elements, and further L1 subfamilies may exist that cannot be statistically validated. The results, especially those for *Alu*, suggest that repeat phylogeny may exhibit much more fine-grained structure than is currently known.

3.4 SINE Phylogeny

Figure 5 presents a phylogeny of 381,248 SINE elements matching RepBase SINE elements found in 61 ENCODE species. Subfamilies are assigned a repeat subfamily (color) if at least 70% of their elements belong to that RepBase subfamily (further detail is available in the electronic version). These labels may partly reflect variation in ENCODE's coverage between species. However a phylogeny constructed from more complete data, while it may have higher resolution, will still be consistent with the one presented here. Representative subfamilies are labeled with the average percent divergence of their elements from the subfamily consensus. The phylogeny is taken from a single run of RATS. To show that the results are consistent between runs, we ran RATS 10 times on the *Alu* data, using the pairwise adjusted Rand index to measure consistency between runs [22]. The number of subfamilies ranged from 274 to 300 with a mean of 286, and the Rand index ranged from 0.154 to 0.193 with a mean of 0.173 and a p-value less than 10^{-8}.

Although our predicted phylogeny contains 111 novel repeat subfamilies, it still reflects many known aspects of SINE phylogeny. We recover the basic relationship between the oldest *Alu J*, intermediate *Alu S*, and recent *Alu Y* clades. Ages estimated from subfamily divergences are roughly in agreement with estimated family ages [6]. The *Alu Jo* subfamily is an ancestor of *Alu Jb* (Fig. 5, label 7). The 11%-diverged *Alu Sx* branch reported by Price [11] appears at (Fig. 5, label 6).

Our results are also consistent with primate clades: for example, the *Alu Jo* branch at (Fig. 5, label 1) separates strepsirrhins from the new- and old-world monkeys. Additionally, the large group of Galago-specific subfamilies we observe is consistent with the sequence analysis of Zietkiewicz *et al.* [23]. The *Alu Y* subfamily is (as expected) confined to old-world primates (Fig. 5, label 2), though without human sequence data only a subset of the currently-known *Alu Y* families are found. Finally, the divergence of the baboon-specific families dates them to after the divergence of old-world monkeys.

Further up the tree we find additional validation of our repeat phylogeny. Repeats from platypus and echidna, both monotremes, are concentrated in a subtree of MIRm elements at (Fig. 5, label 3). Similarly, the marsupial-specific MIR_Mars elements are correctly identified at (Fig. 5, label 4). Our analysis does not find a clade among marsupial, monotremic, and placental repeat elements. Monotremes diverged before marsupials, and Gilbert *et al.* [24] argue that the Ther-2 repeat family is found only in marsupials and placental mammals. However, while the species phylogeny is well-understood, there remains some

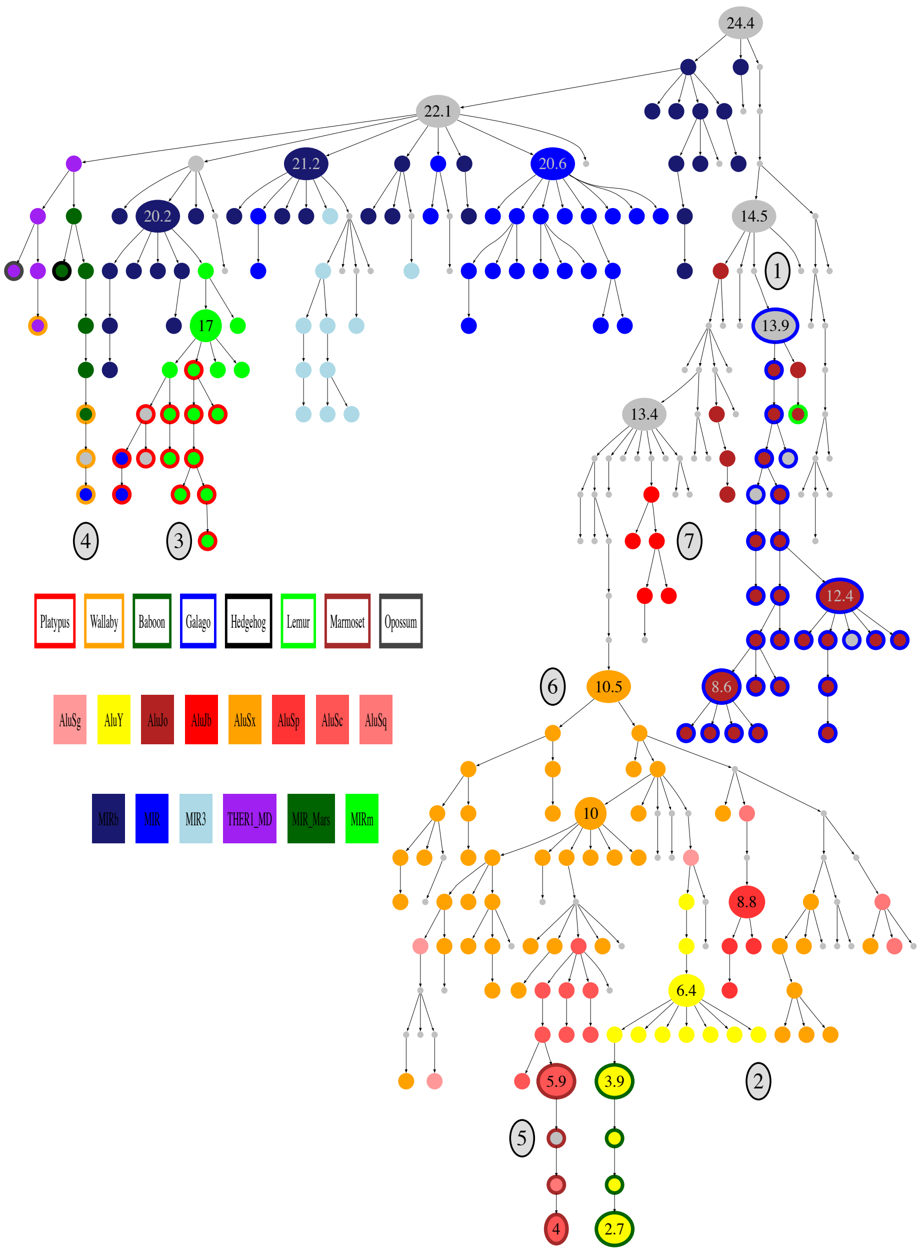


Fig. 5. SINE phylogeny from one run of RATS on all ENCODE species, showing sequences from (1) strepsirrhins, (2) old-world primates, (3) monotremes, (4) marsupials, and (5) new-world primates. Edges represent putative relations between families. Node colors represent known RepBase families. Numbers inside nodes indicate average percent sequence divergence from the subfamily consensus sequence. Unattached, numbered black squares correspond to labels in the text.

disagreement about MIR repeat phylogeny, and in some ways the marsupial genome is more similar to the monotreme than to the placental genome [25].

4 Discussion

We have described RATS, a randomized clustering algorithm for rapidly finding repeat subfamilies and a procedure for reconstructing phylogenies from sets of subfamilies. Because the divisions between subfamilies RATS finds are statistically validated, and because simulations show that it estimates the number of subfamilies conservatively, we can be confident that real substructure is being detected. We have applied our approach to SINE repeat data, yielding a phylogeny consistent with known repeat and species relationships both among and beyond primates. Our results demonstrate that mobile elements display complex family substructure and history, and suggest a number of areas for further exploration.

There remain a number of directions for algorithmic refinement. First, as noted above, inferring the phylogeny and clustering simultaneously would yield a more powerful model; an analogous randomized approach could again yield an efficient approximation. Second, including species phylogeny and repeat orthology information would improve sensitivity. Finally, large- and small-scale phylogeny could be handled simultaneously using an iterative approach combining clustering on inferred subfamilies with refinement of the guiding multiple alignment.

These methodological improvements, particularly the last, open up a number of experimental avenues. Multi-scale clustering would make it possible to distinguish clades among primates, while fine-grained distinctions between families would enable the detection of repeat homoplasy and the analysis of repeat insertion hotspots. Large-scale repeat phylogeny therefore has the potential to contribute in a number of ways to our understanding of evolutionary processes and history.

Acknowledgments. SO and EE are partially supported by National Science Foundation Grant 0513612. EE is partially supported by NIH grant K25HL080079. Part of this investigation was supported using the computing facility made possible by the Research Facilities Improvement Program Grant C06 RR017588 awarded to the Whitaker Biomedical Engineering Institute, and the Biomedical Technology Resource Centers Program Grant P41 RR08605 awarded to the National Biomedical Computation Resource, UCSD, from the National Center for Research Resources, National Institutes of Health. Additional computational resources were provided by the California Institute of Telecommunications and Information Technology. SO thanks David Heckerman and Alkes Price for valuable discussion, and the anonymous reviewers for their comments.

References

1. Watkins, W., et al: Genetic Variation Among World Populations: Inferences From 100 Alu Insertion Polymorphisms. Genome Res. **13**(7) (2003) 1607–18
2. Batzer, M., Deininger, P.: Alu repeats and the human genomic diversity. Nature rev. genet. **3** (May 2002) 370–9

3. Hedges, D., Batzer, M.: From the margins of the genome: mobile elements shape primate evolution. BioEssays **27**(8) (2005) 785–94
4. Smit, A., Hubley, R., Green, P.: RepeatMasker (2006) http://www.repeatmasker.org/.
5. Ostertag, E., Kazazian, H.: LINEs in mind. Nature **435** (2005) 890–1
6. Bailey, J., Liu, G., Eichler, E.: An Alu transposition model for the origin and expansion of human segmental duplications. Am J Hum Genet **73** (2003) 823–34
7. Zhou, Y., Mishra, B.: Quantifying the mechanisms for segmental duplications in mammalian genomes by statistical analysis and modeling. PNAS **102**(11) (2005) 4051–6
8. Han, K., Xing, J., Wang, H., Hedges, D., Garber, R., Cordaux, R., Batzer, M.: Under the genomic radar: the stealth model of Alu amplification. Genome Res. **15** (2005) 655–64
9. Jurka, J.: Evolutionary impact of human Alu repetitive elements. Current Opinion in Genetics & Development **14**(6) (December 2004) 603–8
10. Cordaux, R., Hedges, D., Batzer, M.: Retrotransposition of Alu elements: how many sources? Trends Genet. **20**(10) (October 2004) 464–7
11. Price, A., Eskin, E., Pevzner, P.: Whole genome analysis of Alu repeat elements reveals complex evolutionary history. Genome Res. **14** (2004) 2245–52
12. Salem, A., Ray, D., Xing, J., Callinan, P.A., Myers, J.S., Hedges, D.J., Garber, R.K., Witherspoon, D.J., Jorde, L.B., Batzer, M.A.: Alu elements and hominid phylogenetics. PNAS **100**(22) (2003) 12787–12791
13. Felsenstein, J.: PHYLIP (phylogeny inference package) version 3.6. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle (2004)
14. Thomas, J., Touchman, J., Blakesley, R., Bouffard, G., Beckstrom-Sternberg, S., Margulies, E., Blanchette, M., Siepel, A., Thomas, P., McDowell, J., et al.: Comparative analyses of multi-species sequences from targeted genomic regions. Nature **424** (2003) 788–93
15. Meila, M., Heckerman, D.: An experimental comparison of several clustering and initialization methods. Technical Report MSR-TR-98-06, Microsoft Research (1998)
16. Chickering, D., Heckerman, D.: Efficient approximations for the marginal likelihood of bayesian networks with hidden variables. Machine Learning **29**(2-3) (1997) 181–212
17. Cover, T., Thomas, J.: The elements of information theory. Plenum Press, New York (1991)
18. Goebel, B., Dawy, Z., Hagenauer, J., Mueller, J.: An approximation to the distribution of finite sample size mutual information estimates. In: IEEE International Conference on Communications (ICC), Seoul, South Korea (May 2005)
19. Friedman, N., Ninio, M., Pe'er, I., Pupko, T.: A structural EM algorithm for phylogentic inference. J. Comp. Biol. (2001)
20. Jurka, J.: Repbase update: A database and an electronic journal of repetitive elements. Trends Genet **9** (2000) 418–20
21. Thompson, J., Higgins, D., Gibson, T.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. Nucleic Acids Res. **22** (1994) 4673–80
22. Hubert, L., Arabie, P.: Comparing partitions. J Classification **2** (1985) 193–218

23. Zietkiewicz, E., Richer, C., Sinnett, D., Labuda, D.: Monophyletic origin of Alu elements in primates. J. Mol. Evol. **47**(2) (1998) 172–82
24. Gilbert, N., Labuda, D.: Evolutionary inventions and continuity of CORE-SINEs in mammals. J. Mol. Biol. **298** (2000) 365–77
25. Miller, W., Capy, P., eds.: Retrotransposon mapping in molecular systematics. In: Mobile genetic elements. Humana (2004)
26. Bashir, A., Ye, C., Price, A., Bafna, V.: Orthologous repeats and mammalian phylogenetic inference. Genome Res **15**(7) (2005) 998–1006

Beyond Galled Trees - Decomposition and Computation of Galled Networks

Daniel H. Huson and Tobias H. Klöpper

Center for Bioinformatics (ZBIT), Tübingen University,
Sand 14, 72076 Tübingen, Germany

Abstract. Reticulate networks are a type of phylogenetic network that are used to represent reticulate evolution involving hybridization, horizontal gene transfer or recombination. The simplest form of these networks are galled trees, in which all reticulations are independent of each other. This paper introduces a more general class of reticulate networks, that we call galled networks, in which reticulations are not necessarily independent, but may overlap in a tree-like manner. We prove a Decomposition Theorem for these networks that has important consequences for their computation, and present a fixed-parameter-tractable algorithm for computing such networks from trees or binary sequences. We provide a robust implementation of the algorithm and illustrate its use on two biological datasets, one based on a set of three gene-trees and the other based on a set of binary characters obtained from a restriction site map.

1 Introduction

Phylogenetic networks are graphs used for representing phylogenetic relationships between different taxa, and are usually employed when a tree representation does not suffice. There are many different types of phylogenetic networks and it is useful to distinguish between two main classes: *implicit* phylogenetic networks that provide tools to visualize and analyze incompatible phylogenetic signals, such as split networks [1,2], and *explicit* phylogenetic networks that provide explicit scenarios of reticulate evolution, such as hybridization networks [3,4,5,6,7], HGT networks [8] and recombination networks [9,10,11,12,13,14,15,16,17,18,19].

Although the latter three types of networks apply to quite different evolutionary scenarios, they share the common feature that they all contain *reticulation nodes* at which new sequences arise as a combination of sequences from two different predecessor sequences. Such networks are mathematically similar to each other and are collectively referred to as *reticulate networks*, which are the focus of this paper.

The different types of input to methods that compute these networks also share similarities and can be generally considered as *splits*, that is, bipartitionings of the underlying dataset, defined either by the edges of the input trees, in the case of hybridization or HGT networks [6], or by the non-constant columns of the alignments of binary sequences in the case of recombination networks [17].

Gusfield et al. [12,16] introduce the term *galled tree*, that can be defined as a reticulate network in which no two reticulation nodes are contained in a common unoriented cycle, and provide an algorithm for computing this type of networks. To be precise, their definition requires all cycles to be node-disjoint. However, the combinatorial analysis of galled trees also works if we require cycles to be only edge-disjoint, and so we prefer to require only the latter property.

Interest in galled trees is based on the fact that they are computationally tractable. However, there is little reason to believe that reticulate networks have this simple structure in practice and so an understanding and treatment of reticulate networks of a more general nature is desirable. In [6] we provide an algorithm for computing reticulate networks that goes slightly beyond galled trees and accommodates reticulate networks in which reticulations may overlap along paths. The goal of this paper is to introduce a more general class of reticulate networks that go substantially beyond galled trees.

Given an input dataset, such as a collection of trees, or a multiple alignment of binary sequences, the computational goal is to determine a reticulate network that "explains" the given dataset, in terms of mutations, speciations and reticulate events, such as recombinations, HGT or hybridizations. For any given dataset, many different networks may exist that can explain it. Always, at least one exists, as shown in [20]. Considering reticulations to be expensive evolutionary events, one will prefer a most parsimonious solution. In the context of phylogenetic trees and hybridization networks, one may attempt to minimize the number of reticulation nodes. In the context of recombination networks, an alternative optimization goal is to minimize the number of recombination cross-overs, but we do not address this here.

The associated computational problem is NP-complete in full generality [21,7]:

Problem 1 (Most Parsimonious Reticulate Network). *For a given input set Δ (consisting of trees, binary characters or splits, depending on the concrete application), determine a reticulate network N that explains Δ using a minimum number of reticulation nodes.*

Decomposition Theorems, which aim at dividing the task into independent subproblems that can be identified by the pattern of incompatibilities in the input [16,6], are an important tool for addressing Problem 1.

This paper makes five theoretical and practical contributions in the area of phylogenetic networks. Firstly, we introduce a natural generalization of galled trees, which we call *galled networks*, that go substantially beyond galled trees. This generalization allows for quite complicated configurations of reticulations, as present in real data. Secondly, we prove a Decomposition Theorem for galled networks that settles an open conjecture posed by Dan Gusfield at RECOMB 2005 [16] for this class of networks. Thirdly, we provide a fixed-parameter-tractable algorithm for computing galled networks from real data. Fourthly, we provide an implementation of our algorithm as a plug-in for SplitsTree4 [2], thus making the algorithm easily available to the community. Finally, we illustrate our results on two published datasets, one that uses three different gene trees to study the evolution of a set of fungal species [22] and the other that uses binary sequences from a restriction site map to study the evolution of a set of mosquitoes [23].

2 Splits, the Incompatibility Graph and Reticulate Networks

In this section we introduce a number of basic concepts.

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of taxa. A *split* $S = \frac{A}{B} = \frac{B}{A}$ of X is a bipartitioning of X into two *parts* $A \neq \emptyset$ and $B \neq \emptyset$, with $A \cap B = \emptyset$ and $A \cup B = X$. In the context of hybridization or HGT networks, the input usually is a set of phylogenetic trees, each leaf-labeled by X. Any edge e of such a tree T defines a split $S = \frac{A}{B}$ in which A and B are the label sets of the two subtrees T_A and T_B separated by e. The set $\Sigma(T)$ of all splits obtainable from T is called the *split encoding* of T. In the context of recombination networks, the input is usually a multiple alignment M of binary sequences, and any distinct (up to switching of the roles of 0 and 1), non-constant column i is said to *induce* a split $\frac{A}{B}$ in which all taxa whose i-th character has state 0 are placed in A, say, and all others are placed in B.

Usually, we will consider *rooted* trees and networks. To facilitate this, we will assume that the input contains an *outgroup taxon* o that attaches directly to the root node. As usual, we define two splits $S = \frac{A}{B}$ and $S' = \frac{A'}{B'}$ as *compatible*, if and only if one of the four possible intersections of split parts is empty: $A \cap A'$, $A \cap B'$, $B \cap A'$, or $B \cap B'$. Otherwise, they are called *incompatible*. For a set of splits Σ we define the *incompatibility graph* $IG(\Sigma)$ as the graph with node set Σ in which any two nodes $S, S' \in \Sigma$ are connected by an unoriented edge $\{S, S'\}$, if and only if they are incompatible. We will refer to the connected components of $IG(\Sigma)$ as *incompatibility components* of Σ. If $X' \subset X$, then we use $\Sigma|_{X'}$ to denote the set of all splits *induced* by X', i.e. obtained by removing X' from both parts of each split and then removing any split that has a empty part.

Definition 1. *Let X be a set of taxa. A* (rooted) reticulate network *N on X is a connected, directed acyclic graph (V, E, ϕ) with node set V and edge set E, together with a labeling $\phi : X \to V$, such that:*

- *precisely one node of indegree 0 exists, called the* root *and usually denoted by ρ;*
- *all other nodes are* tree nodes *of indegree 1 or* reticulation nodes *of indegree 2, and have arbitrary outdegrees ≥ 0;*
- *every edge is either a* tree edge *incident to precisely two tree nodes, or a* reticulation edge *leading to a reticulation node; and*
- *the set of* leaves *(nodes of outdegree 0) labelled by X.*

In the following we need to distinguish between oriented and unoriented paths and cycles. In the oriented case, all edges must be oriented in the same direction when traversing the path or cycle, whereas in the unoriented case, the orientation of the edges is irrelevant. We say that a node w is *reachable* from a node v, if there exists an oriented path from v to w in the network N.

It follows from the definition that each reticulation node (or *reticulation*, for short) r with reticulate edges p and q is contained in one or more unoriented cycles of the form $C=(r,p,v_1,e_1,\dots,e_{j-1},v_j,q,r)$, $C=(r,p,v_1,e_1,\dots,e_{j-1},v_j,r)$ or $C=(r,v_1,e_1,\dots,e_{j-1},v_j,q,r)$, with $v_i \in V$ and $e_i \in E \setminus \{p,q\}$ for all i. Any such unoriented cycle C is called a *reticulation cycle* and we define its *backbone* as $B(C)=(v_1,e_1,\dots,e_{j-1},v_j)$. Note that a reticulation $r \in R$ possesses at most one reticulation cycle C whose backbone B contains only tree edges and in this case we call C a *tree cycle*.

Following [12], we say that a reticulation r is a *gall*, if it is contained in precisely one unoriented cycle C, which then must be a tree cycle. If all reticulations in N are galls, then N is called a *galled tree*.

Reticulate networks are employed to *explain* a given set of observed data in terms of evolutionary events that include speciation, mutation and reticulation events. A precise definition of what an explaination is, depends on the nature of the input data.

First, we consider the situation that arises in the context of gene trees and hybridization. Let N be a reticulate network. We define $\mathcal{T}(N)$ as the set of all trees that can be *sampled* from N. A tree is sampled from N as follows: For each reticulation r with reticulation edges p and q, we select one reticulation edge and delete the other. After making a complete set of selections, superfluous nodes and edges are removed from the remaining network and the result is a sampled tree T. (For example, a node of outdegree 2 will become superfluous, if one of its outedges is a reticulation edge that is deleted.) In this way, a reticulate network with t reticulation nodes will give rise to $\leq 2^t$ different trees. If we are given a set of phylogenetic trees $\Delta=\{T_1,\dots,T_t\}$ on X, then we say that a reticulate network N *explains* Δ, if and only if $\Delta \subseteq \mathcal{T}(N)$. In this context, N is called a *hybridization network* and the reticulate nodes of N correspond to speciation events via hybridization. We treat HGT similarly to hybridization, but in this case reticulate nodes correspond to instances of horizontal gene transfer. (Note that throughout the paper we use Δ to denote the current input set, be it a set of trees, binary characters or splits.)

Second, we consider the situation that arises in the context of binary sequences and recombination [12]. Let N be a reticulate network on X and let M be a multiple alignment of binary sequences of length m on X. For a reticulate network to be able to explain a dataset such as M, we attach additional information to N and thus obtain what is called a *recombination network*. More precisely, we label each node of N by a binary sequence of length m such that each leaf v obtains the row of M that corresponds to the taxon that is associated with v. Moreover, each tree edge e is labeled by the set of columns in M at which the character states differ between the two sequences that label the source and target nodes of e. We say that N, together with these two labelings, *explains* M, if and only if each column occurs at most once as an edge label and the sequence at any given reticulation node r can be obtained by recombination of the ancestral sequences. The condition that mutations are allowed to occur at most once is known as the "infinite sites model" [24] and is necessary to avoid trivial solutions. (Without this condition, any set of sequences can be produced by the star phylogeny by inserting appropriate mutations along each of the leaf edges.)

Whether the initial input is a set of trees or a set of binary sequences, in both cases it is useful to consider the set of splits associated with the input. If the input is a set of phylogenetic trees $\{T_1, \ldots, T_t\}$, then the associated set of input splits is $\Delta = \bigcup_{i=1}^{t} \Sigma(T_i)$, where $\Sigma(T_i)$ denotes the split encoding of T_i. On the other hand, if the input is a multiple alignment M of binary sequences, then the associated set of input splits Δ consists of all splits induced by distinct, non-constant columns of M. Note, however, that the set of splits does not contain all relevant input information. For example, in the case of a multiple alignment of binary sequences, the order of the splits along the alignment is lost, whereas in the case of trees, we do not know whether two different splits ever occur together in the same tree or not.

For a reticulate network N on X, we define $\Sigma(N)$ as the set of all splits that can be *sampled* from N as follows: For each reticulation node r, select one of the two reticulation edges leading to r and delete the other, and then remove all superfluous nodes and edges. Any tree edge e contained in the remaining network N' separates two components N'_A and N'_B, defining a split $\frac{A}{B}$. The set of all splits generated for any fixed tree edge e is denoted by $\Sigma(e)$.

In graph theory, a 2-*connected component* of a graph G is any maximal subgraph G' with the property that any two nodes v and w of G' are connected by two paths P and P' that are node-disjoint except for v and w. (In [6], we refer to these as *netted components* and [16] introduced the term *blob*.)

3 Loose Galls and Galled Networks

Our goal is to go beyond galled trees and thus we introduce the following definition:

Definition 2. *Let N be a reticulation network. We say that a reticulation r in N is a* loose gall*, if it has a tree cycle. If all reticulations in N are loose galls, then N is called a* galled network *(or* loose galled tree*).*

Galled networks are substantially more general than galled trees. In contrast to the situation in galled trees where reticulation cycles are edge-disjoint, multiple reticulation cycles in a galled network may overlap along a common tree (see Figures 3 and 4).

To avoid configurations that may cause technical difficulties, we will assume that all galled networks N considered are *non-degenerate* in the sense that for each tree node v, an oriented path of tree edges exists which connects v to some leaf w of N.

A useful property of galled trees that is also valid for galled networks is the following result:

Lemma 1 (Isolation lemma). *Let $N = (V, E, \phi)$ be a galled network and r a reticulation in N. Let $R \subset V$ be the set of all nodes reachable from r. Then any unoriented path from any node $v \in V \setminus R$ to any node $w \in R$ must pass through r.*

Proof. By the properties of directed, acyclic graphs, any path leading from a node in $V \setminus R$ to a node in R must contain a reticulation edge p' leading to a reticulation node $r' \in R$. But then, any unoriented cycle through r' will pass through p' and one of the two reticulation edges of r, or through a reticulation edge of some other reticulation connecting $V \setminus R$ and R. In either case, the cycle is not a tree cycle, contradicting the assumption that N is a galled network. □

The following property is useful because it implies that we can build a galled network by first producing a phylogenetic tree and then adding reticulations to the tree by attaching them to *existing* tree nodes or to the mid-points of tree edges.

Lemma 2 (Node-attachment lemma). *Let N be a reticulate network that explains a set of splits Δ. For any reticulate node r, we can assume that each of the two* attachment nodes *(source nodes of the reticulate edges leading to r) are either the root node, or an internal tree node that is the source of at least two tree edges, or lies directly below such a node and lies directly above this type of a node or a leaf node.*

Proof. Assume that a path $P = (v, e_1, v_1, e_2, \ldots, e_j, w)$ exists so that each end node v, w is either the root node, a leaf or an internal tree node that is the source of at least two tree edges. Furthermore, assume that all other nodes in the path are tree nodes of degree three, each incident to precisely one reticulation edge. Let R denote the set of all such reticulation edges. The splits generated by any node in the path separates the set of all taxa below w (and a selection of the reticulate nodes attached to internal nodes of the path) from the outgroup o and other taxa. We can generate the same set of splits after replacing P by a new path $P' = (v, e, u, f, w)$ and attaching all reticulate edges in R to the node u. Now, any split sampled from an edge in P that does not separate any of the reticulation nodes from o can be sampled from e, whereas all other splits can be sampled from f, using an appropriate selection of the reticulation edges. (see Figure 1.) □

As mentioned above, the set of splits Δ derived from an input set of trees $\mathcal{T}$ does not contain the full information present in the input. In Lemma 2, we

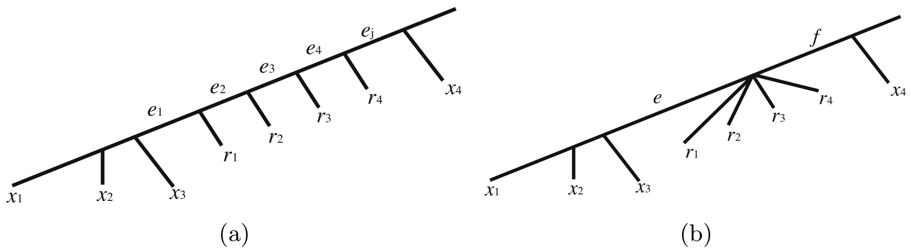


Fig. 1. (a) An unmodified path P, and (b) its modification P'. Any split obtainable from an edge in (a) can also be obtained from some edge in (b).

can make the modification displayed in Figure 1 because the set of splits does not provide any information about the ordering of the reticulations along the path P. However, $\mathcal{T}$ may have trees that contain more than one of the depicted reticulation edges simultaneously, in which case a contraction of the path P to the two edges e, f may not be allowed, if the goal is to find a network N with $\mathcal{T} \subseteq \mathcal{T}(N)$, rather than with $\Delta \subseteq \Sigma(N)$.

4 The Decomposition Theorem for Galled Networks

Decomposition Theorems for reticulate networks aim at dividing Problem 1 into independent subproblems that can be identified by the pattern of incompatibilities in the input. Two very similar results of this type were presented at RECOMB 2005. In [16], a Decomposition Theorem is proven that can be phrased as follows:

> For any set of splits Δ, a reticulate network N exists that explains Δ so that each 2-connected component of N contains all and only the splits contained in one connected component of the incompatibility graph $IG(\Delta)$.

In [6], we prove the following:

> Let N be a reticulate network and let $\Delta = \Sigma(N)$ be the set of *all* splits that can be sampled from the network. Each 2-connected component of N contains all and only the splits contained in one connected component of $IG(\Delta)$.

In the former result, it remains unresolved whether a *minimal* network of this type exists. In the latter result, the existence of a minimal network N with the specified properties is guaranteed only for a dataset Δ that equals the *full* set of splits that can be sampled from the network N. Hence, the following claim is of interest (adapted from a conjecture formulated by Dan Gusfield at RECOMB 2005 [16]):

Conjecture 1. *Given an input set of splits Δ, a minimal reticulation network N exists that explains Δ so that for any two tree edges e and f of N we have the following: Any two splits $S \in \Delta \cap \Sigma(e)$ and $S' \in \Delta \cap \Sigma(f)$ are contained in the same connected component of $IG(\Delta)$, if and only if e and f are contained in a common unoriented cycle of N.*

Gusfield *et al.* [16] proved that this is true for galled trees. The following result implies that this conjecture is also true for a more general class of reticulate networks, namely galled networks.

Theorem 1 (Decomposition Theorem for Galled Networks). *Let Δ be an input set of splits and let N be a (non-degenerate) galled network that explains Δ. Assume that N is minimal in the sense that it uses a minimum number of reticulations, and, among all such networks, minimizes the number of splits that* *are sampled from edges that lie within reticulation cycles and then the total number of edges. For any two tree edges e and f in N we have that any two splits $S \in \Delta \cap \Sigma(e)$ and $S' \in \Delta \cap \Sigma(f)$ are contained in the same connected component of $IG(\Delta)$ if and only if e and f are contained in an unoriented cycle of N.*

Proof. "$\Rightarrow$": This direction is easy and was shown in [6].
"$\Leftarrow$": Let N be a minimal galled network that explains a given set of splits Δ on a taxon set X. Consider the subgraph G of N consisting of all nodes and edges that are connected to the root ρ of N by some path containing only tree edges. It follows from the assumed non-degenerancy of N that G is a rooted phylogenetic tree on a subset of taxa. Consider a reticulate node r that is attached to two nodes a and b by reticulation edges p and q. Because N is a galled network, either both a and b are contained in G, or neither is. By Lemma 1, we need to consider only the former case and can assume that r is a leaf node, labeled by a single taxon r.

Assume that a is an ancestor of b and let $P = (a, e_1, v_1, e_2, v_2, \ldots, v_{j-1}, e_j, b)$ be the path from a to b (see Figure 2(a)). Consider the set of splits $\Delta(e_1) := \Delta \cap \Sigma(e_1) \neq \emptyset$ associated with edge e_1. (Note that a given split $S \in \Delta$ may be obtainable from more than one edge in N. In this case, we remove S from the split sets of all but one edge.) None of the splits $\Sigma_a \in \Delta(e_1)$ separate r from o, because any such split can be "pushed up" in the network and out of the reticulation cycle associated with r (after placing a in the middle of e_1, if necessary), contradicting minimality. By the non-degenerancy property and the node-attachment lemma, every $S_a \in \Delta(e_1)$ separates some taxon x from o, where x is reached from v_1 by a path that is disjoint from P. Now, consider the set of splits $\Delta(e_j) := \Delta \cap \Sigma(e_j) \neq \emptyset$. Note that all of the splits in $\Delta(e_j)$ separate r from o because any split that does not do so can be "pushed down" in the network (after placing b in the middle of e_j, if necessary), contradicting minimality. Hence, all splits $S_b \in \Delta(e_j)$ separate both r and some other taxon y from o. The latter follows from the non-degenerancy property and the node-attachment lemma. Now, every split S_a also separates y from o by construction. In summary, every S_a separates x, y from o, r and every S_b separates x, o from y, r. Thus, all pairs of S_a and S_b are incompatible. Now, consider some edge e_i with $1 < i < j - 1$. We must have $\Delta(e_i) := \Delta \cap \Sigma(e_i) \neq \emptyset$, or otherwise we can contract e_i, contradicting the minimality of N. Consider $S \in \Delta(e_i)$. By non-degenerancy, a taxon z exists that can be reached from v_i by a path that is disjoint from P. If S separates r from o, then S separates y, z, r from x, o, and is thus incompatible with every S_a, or else S separates y, z from x, r, o, and is thus incompatible with every S_b, as S_b separates o, x, z from y, r.

Second, we now assume that neither is a an ancestor of b, nor vice versa (see Figure 2(b)). Let $P = (a, e_1, v_1, \ldots, e_j, b)$ be the path joining the two attachment nodes. By the same argument used above, we can assume that the edge e_1 gives rise only to splits S_a that separate o, y from taxa x, r, and we can assume that e_j gives rise only to incompatible splits S_b that separates o, x from y, r, where x and y are two taxa located below a and b, respectively. Thus, any split S

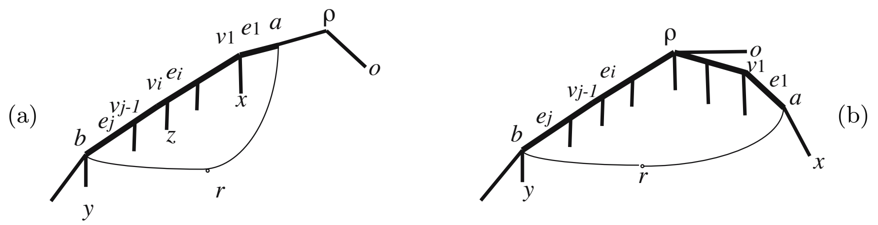


Fig. 2. (a) This shows the situation in the proof of Theorem 1 when a reticulation r attaches to two nodes a and b, where a is an ancestor of b. (b) The other possibility is that neither is a an ancestor of b, nor vice versa.

sampled from an edge in the interior of the path with be incompatible with all S_a or all S_b, depending on whether S separates r from o or not, and whether it lies between a and ρ or between ρ and b.

We have shown that the claim holds for any pair of edges contained in the same tree cycle. Now, if two edges e and f are contained in a common unoriented cycle, then a chain of tree cycles $C_1, C_2, \ldots, C_h$ must exist so that e is contained in C_1, f is contained in C_h, and any two adjacent tree cycles C_i, C_{i+1} $(i = 1, \ldots, h-1)$ share a mutual tree edge. Hence, the full claim holds. □

5 Computation of Galled Networks

In this section, we present an algorithm that takes a set of splits Δ on a given taxon set X and a parameter $K \geq 0$ as input and produces a minimal galled network N that explains Δ using at most K reticulations in any 2-connected component of N. As we will see, the problem is fixed-parameter-tractable, as the run-time of the algorithm is polynomial in the size of Δ and X, for any fixed value of K. In practice, the choice of K determines the amount of complexity that one is willing to endure to explain a part of the network, which will usually be quite small.

By Theorem 1, we can restrict our attention to the case that the incompatibility graph $IG(\Delta)$ has precisely one component and the resulting network N will consist of exactly one 2-connected component. To formulate our algorithm, we need a further definition. Let Δ be a set of splits on X and suppose that T is a tree on a subset of taxa $X' \subsetneq X$ with $\Sigma(T) \subset \Delta|_{X'}$. We say that a node $r \in X \setminus X'$ *bridges* an edge e of T, if for the split $S = \frac{A}{B}$ associated with e we have $\left\{\frac{A\cup\{r\}}{B}, \frac{A}{B\cup\{r\}}\right\} \subseteq \Delta|_{X'\cup\{r\}}$.

The main idea of the algorithm is to first select a subset of taxa $R \subset X$ such that the set of splits $\Delta|_{X'}$ induced by the remaining taxa $X' = X \setminus R$ is compatible and thus gives rise to a tree T. We then attempt to attach each taxon $r \in R$ to the tree T by a pair of reticulate edges in such a way that all bridged edges are encompassed.

Algorithm 1 (Construct Galled Network from Splits).
Input: Set of splits Δ on X
Output: Minimal galled network N that explains Δ, if one exists, or fail.

```
for $k = 1 \ldots K$ do
    for each possible choice of a subset $R \subset X$ of cardinality $k$ do
        if $\Sigma' := \Sigma|_{X \setminus R}$ is compatible then
            Build a rooted backbone tree $T$ that represents $\Sigma'$
            for each putative reticulation node $r \in R$ do
                Let $B$ be the set of all edges in $T$ that are bridged by $r$
                if $B$ is contained in a path in $T$ then
                    attach $r$ to the mid-points of the two end edges of
                    a shortest such path
                else we can't attach $R$, try next choice of $R$
            if all $r \in R$ were successfully attached and the resulting network $N$
                generates $\Delta$ then return $N$
return fail.
```

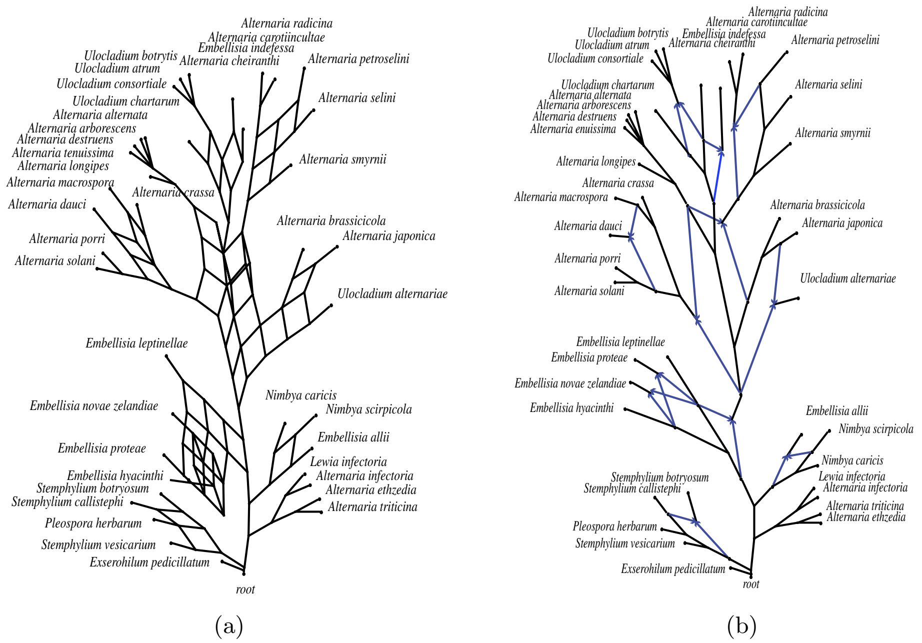
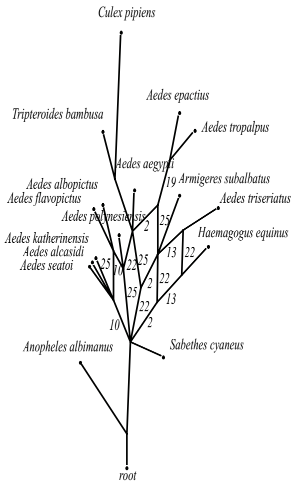


Fig. 3. (a) Split network representing all splits present in the three gene trees taken from [22]. (b) Galled network computed by our algorithm. A number of the reticulations overlap in this network and are thus loose galls, rather than galls. For example, the two taxa *Embellisia proteae* and *E. novae zelandiae* arise from two overlapping loose galls. Reticulation edges are shown as arrows pointing toward their reticulation node.

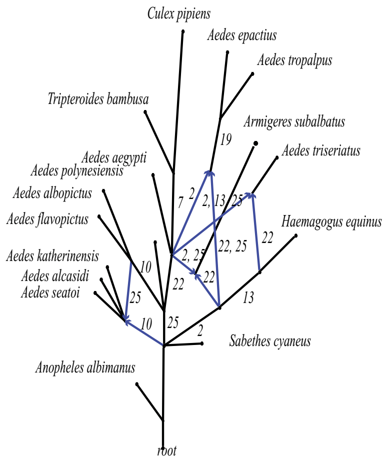
Lemma 3. *Given an input set of splits Δ and a parameter $K \in \mathbb{N}$, Algorithm [1] computes a minimal galled network that explains Δ using at most K reticulations per 2-connected component, if one exists, or returns* fail, *in polynomial time.*

Aedes albopictus	11110101010100010101010010
Aedes aegypti	11110101000100010101000010
Aedes seatoi	11110101010100010101010000
Aedes flavopictus	11110101010100010101010010
Aedes alcasidi	11110101010100010101010000
Aedes katherinensis	11110101010100010101010000
Aedes polynesiensis	11110101000100010101010010
Aedes triseriatus	10110101000110010101000000
Aedes atropalpus	10110101000100010111000010
Aedes epactius	10110101000100010111000010
Haemagogus equinus	10110101000110010101010000
Armigeres subalbatus	10110101000100010101000000
Culex pipiens	11110111000100011101001011
Tripteroides bambusa	11110111000100010101000010
Sabethes cyaneus	11110101001100010101010000
Anopheles albimanus	11011101100101110101110100

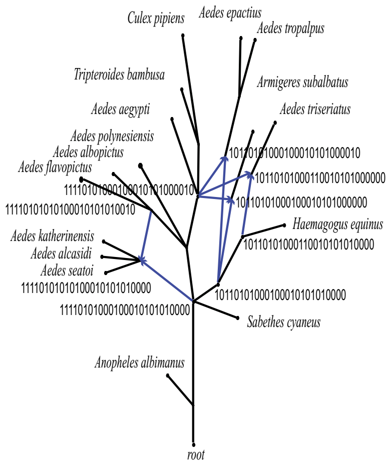
(a)



(b)



(c)



(d)

Fig. 4. (a) An alignment of binary sequences obtained from the restriction maps of the rDNA cistron (length ≈ 10 *kb*) of twelve species of mosquitoes using eight 6 *bp* recognition restriction enzymes [23]. (b) The corresponding split network [2]. Edges of incompatible splits are labeled by the corresponding mutation positions. (c) The recombination network computed using our algorithm. The edges involved in a possible recombination are labeled by the corresponding mutations in the alignment. (d) The same network, with nodes involved in the possible recombinations labeled by the corresponding inferred sequences.

Proof. The algorithm considers each connected component of $IG(\Delta)$ separately, by virtue of Theorem 1. For a fixed value of K, the number $\sum_{k=1}^{K}\binom{|X|}{k}$ of subsets of X of size $k \leq K$ is polynomial in the size $|X|$ of X. For each putative reticulation r, we compare all splits with the set of splits associated with the backbone tree T, which takes time that is polynomial in $|X|$ and $|\Delta|$. All possible galled networks with up to K reticulations per 2-connected component are generated in ascending order and the first (and thus minimal) one found that explains the input data is returned. $\square$

As discussed above (see Figure 1), because the algorithm is based on splits, it cannot resolve the ordering of multiple reticulations along a tree edge. We have developed an additional algorithm that does this sorting when the original input is a set of trees. However, due to space constraints this will be discussed elsewhere.

We have implemented Algorithm 1 as a plugin-in for the program SplitsTree4 [2] and it is available as of version 4.7. Our implementation first divides a given set of splits Δ into its incompatibility components $\Delta_1, \ldots, \Delta_t$, with induced taxon sets $X_1, \ldots, X_t$, respectively. For each component Δ_i, our software considers all minimal subsets R_i of X_i for which $\Delta_i|_{X_i \setminus R_i}$ is compatible, and computes the corresponding trees T_i. For each element r in R_i the algorithm marks those edges in the tree T_i that are bridged by r. If the set of marked edges is contained in a path in T_i, then we have found the backbone of the tree cycle for r; otherwise r cannot be attached and the algorithm moves to the next minimal set.

Finally, the algorithm constructs a split network for Δ using the algorithms described in [25] and modifies each 2-connected component for which it was able to find a solution, so as to display all reticulations explicitly. For additional analysis, our implementation can label nodes by the corresponding binary sequences and label edges by the corresponding mutations, when the input is binary sequences, and can highlight different input trees embedded in the network, when the input is a set of trees. Figures 3 and 4 show networks computed by our code.

From a practical point-of-view, an attractive feature of our implementation is that it processes each incompatibility component separately and produces the picture of a mixed network as output in which those incompatibility components that have an explanation in terms of a galled network are drawn as such, whereas the others that cannot be explained in this way are represented by a visualization of the contained splits in terms of a split network (not shown here).

6 Application to Real Data

We now demonstrate our approach using two different examples. The first example illustrates the computation of hybridization networks. The input data consists of three gene trees relating different fungal species, published in [22] and downloaded from TreeBASE [26]. They are based on the mitochondrial small subunit ribosomal DNA, a nuclear internal transcribed spacer and on a part of the glyceraldehyde-3-phosphate gene. We extracted the set of all splits $\varDelta$ and then ran our algorithm on this input. The dataset has seven non-trivial incompatibility components and it turns out that each can be explained in terms of a galled network. Our implementation required about 15 seconds to compute the galled network shown in Figure 3.

The second example illustrates the computation of recombination networks. In [23], a set of binary characters for 16 species of the mosquito subfamily *Culicinae* and an outgroup *Anopheles albimanus* is presented, obtained from the analysis of a restriction map of the rDNA cistron (see Figure 4(a)). The data consists of 26 sites, of which 18 are polymorphic. In the original publication, this dataset was analyzed using a number of different tree-reconstruction methods with inconclusive results. Using a previous algorithm for computing galled trees, we were only able to obtain a result for a partial dataset [17]. Our new algorithm can explain the complete dataset in terms of a galled network. In Figure 4(b), we display the split network (see [2] for details) representing all splits in the input. The recombination network computed using our new algorithm is shown in Figure 4(c–d). Here, the computation took less than 5 seconds.

7 Conclusion

In this paper, we introduce a new class of reticulate networks, *galled networks*, that are a natural generalization of galled trees and we provide an algorithm and implementation that computes these networks from a set of trees or from binary sequences. Like similar algorithms (e.g. [16, 5, 19, 6, 17]), our approach solves a purely combinatorial puzzle: given a set of input data $\varDelta$, find a minimal network N that explains *all* incompatibilities in the data by reticulation events. However, incompatible signals in real biological data may have many causes and the main challenge is to develop network construction methods that can tolerate such noise in the data. In a recent paper [27], we describe a filtering technique that aims at removing incompatible signals that cannot easily be explained by reticulation events. However, much remains to be done before we will have a widely applicable tool that robustly produces meaningful reticulation networks from noisy data.

References

1. Bandelt, H.J., Dress, A.W.M.: A canonical decomposition theory for metrics on a finite set. Advances in Mathematics **92** (1992) 47–105
2. Huson, D.H., Bryant, D.: Application of phylogenetic networks in evolutionary studies. Molecular Biology and Evolution **23** (2006) 254–267 Software available from `www.splitstree.org`.
3. Sang, T., Zhong, Y.: Testing hybrization hypotheses based on incongruent gene trees. System. Biol. **49** (2000) 422–424
4. Linder, C.R., Rieseberg, L.H.: Reconstructing patterns of reticulate evolution in plants. Am. J. Bot. **91** (2004) 1700–1708

5. Nakhleh, L., Warnow, T., Linder, C.R.: Reconstructing reticulate evolution in species - theory and practice. In: Proceedings of the Eighth International Conference on Research in Computational Molecular Biology (RECOMB). (2004) 337–346
6. Huson, D., Kloepper, T., Lockhart, P., Steel, M.: Reconstruction of reticulate networks from gene trees. In: Proceedings of the Ninth International Conference on Research in Computational Molecular Biology (RECOMB). Volume 3500 of LNCS., Springer Verlag (2005) 233–249
7. Bordewich, M., Semple, C.: Computing the minimum number of hybridisation events for a consistent evolutionary history. In press (2006)
8. Hallett, M., Largergren, J., Tofigh, A.: Simultaneous identification of duplications and lateral transfers. In: Proceedings of the Eight International Conference on Research in Computational Molecular Biology (RECOMB). (2004) 347–356
9. Hudson, R.R.: Properties of the neutral allele model with intergenic recombination. Theoretical Population Biology **23** (1983) 183–201
10. Hein, J.: Reconstructing evolution of sequences subject to recombination using parsimony. Math. Biosci. (1990) 185–200
11. Griffiths, R.C., Marjoram, P.: Ancestral inference from samples of DNA sequences with recombination. J. Computational Biology **3** (1996) 479–502
12. Gusfield, D., Eddhu, S., Langley, C.: Efficient reconstruction of phylogenetic networks with constrained recombination. In: Proceedings of the IEEE Computer Society Conference on Bioinformatics. (2003) 363
13. Song, Y., Hein, J.: Parsimonious reconstruction of sequence evolution and haplotype blocks: Finding the minimum number of recombination events. (2003) 287–302 Proceedings of the Workshop on Algorithms in Bioinformatics.
14. D. Gusfield, S.E., Langley, C.: The fine structure of galls in phylogenetic networks. INFORMS J. of Computing Special Issue on Computational Biology **16** (2004) 459–469
15. Song, Y., Hein, J.: On the minimum number of recombination events in the evolutionary history of DNA sequences. J. Math. Biol. **48** (2004) 160–186
16. Gusfield, D., Bansal, V.: A fundamental decomposition theory for phylogenetic networks and incompatible characters. In: Proceedings of the Ninth International Conference on Research in Computational Molecular Biology (RECOMB). (2005) 217–232
17. Huson, D., Kloepper, T.: Computing recombination networks from binary sequences. Bioinformatics **21** (2005) ii159–ii165 ECCB.
18. Song, Y., Hein, J.: Constructing minimal ancestral recombination graphs. J. Comp. Biol. **12** (2005) 147–169
19. Lyngsø, R.B., Song, Y.S., Hein, J.: Minimum recombination histories by branch and bound. In: WABI. (2005) 239–250
20. Baroni, M., Semple, C., Steel, M.A.: A framework for representing reticulate evolution. Annals of Combinatorics (In press)
21. Wang, L., Zhang, K., Zhang, L.: Perfect phylogenetic networks with recombination. Journal of Computational Biology **8** (2001) 69–78
22. Pryor, B.M., Bigelow, D.M.: Molecular characterization of Embellisia and Nimbya species and their relationship to Alternaria, Ulocladium and Stemphylium. Mycologia **95** (2003) 1141–1154
23. Kumar, A., Black, W., Rai, K.: An estimate of phylogenetic relationships among culicine mosquitoes using a restriction map of the rDNA cistron. Insect Molecular Biology **7** (1998) 367–373

24. Kimura, M.: The number of heterozygous nucleotide sites maintained in a finite population due to the steady flux of mutations. Genetics **61** (1969) 893903
25. Dress, A.W.M., Huson, D.H.: Constructing splits graphs. IEEE/ACM Transactions in Computational Biology and Bioinformatics **1** (2004) 109–115
26. Sanderson, M.J., Donoghue, M.J., Piel, W., Eriksson, T.: Treebase: a prototype database of phylogenetic analyses and an interactive tool for browsing the phylogeny of life. Amer. Jour. Bot. **81** (1994) 183
27. Huson, D., Steel, M., Whitfield, J.: Reducing distortion in phylogenetic networks. In Bücher, P., Moret, B., eds.: Algorithms in Bioinformatics. LNBI 4175 (2006) 150–161

Variational Upper Bounds for Probabilistic Phylogenetic Models

Ydo Wexler* and Dan Geiger

Dept. of Computer Science, Technion - Israel Institute of Technology, Haifa 32000, Israel
{ywex,dang}@cs.technion.ac.il

Abstract. Probabilistic phylogenetic models which relax the site independence evolution assumption often face the problem of infeasible likelihood computations, for example for the task of selecting suitable parameters for the model. We present a new approximation method, applicable for a wide range of probabilistic models, which guarantees to upper bound the true likelihood of data, and apply it to the problem of probabilistic phylogenetic models. The new method is complementary to known variational methods that lower bound the likelihood, and it uses similar methods to optimize the bounds from above and below. We applied our method to aligned DNA sequences of various lengths from human in the region of the CFTR gene and homologous from eight mammals, and found the upper bounds to be appreciably close to the true likelihood whenever it could be computed. When computing the exact likelihood was not feasible, we demonstrated the proximity of the upper and lower variational bounds, implying a tight approximation of the likelihood.

1 Introduction

Most organisms share a great deal of their genetic code with other forms of life. Phylogenetic tree models are used to associate the genetic makeup of different organisms according to their genetic variation. A node on phylogenetic trees corresponds to a piece of genetic code in a single organism, and the branches and the relative branch lengths measure the relative distance from each organisms' genes to the others. The greater the distance, the more the gene sequence has changed between one organism and the other.

The classical phylogenetic models of Neyman (1971) and Felsenstein (1981) make several assumptions regarding how evolution occurs in the trees, from which the most stringent assumption is that evolution takes place independently in different sites. Over the years more complex probabilistic phylogenetic models have been proposed, which relax the site independence evolution assumption. These complex models that are more biologically realistic, such as the one by Siepel and Haussler (2003), often face the problem of infeasible likelihood computations, for example for the task of selecting suitable parameters for the model. To overcome this problem Jojic et al. (2004) suggested to use variational approximations that lower bound the likelihood of data, and showed that such bounds tend to be close to the true likelihood.

In this paper, we develop tight upper bounds on the likelihood of a given data, that are close to lower bounds so that good estimates of the likelihood become available.

* Corresponding author.

Our new approximation method is applicable for a wide range of probabilistic models, including the discussed phylogenetic models. The method assumes a simple distribution Q which approximates the target distribution P of the model, and using Jensen's inequality it upper bounds the likelihood of data with a function of Q and P. The simplicity of Q yields a bound that can be computed efficiently.

Our method is complementary to known variational methods that lower bound the likelihood (e.g. Jordan et al., 1999), and can use an approximating distribution Q suggested by these methods to bound the likelihood also from above.

We applied our method to aligned DNA sequences of various lengths from human in the region of the CFTR gene and homologous from eight mammals, and found the upper bounds to be appreciably close to the true likelihood whenever it could be computed. When computing the exact likelihood was not feasible, we demonstrated the proximity of the upper and lower variational bounds, implying a tight approximation of the likelihood.

The rest of the paper is organized as follows: Section 2 briefly describes phylogenetic HMM models in terms of Bayesian networks or DAG models, and provides a quick overview regarding variational techniques that lower bound the likelihood of data. Section 3 develops our main contribution which are variational upper bounds for probabilistic models such as Bayesian networks. The experimental results are described in Section 4. Finally, we discuss the limitations of variational methods.

2 Preliminaries

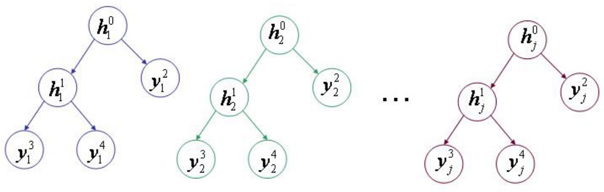
We provide background information regarding phylogenetic HMM trees, to which the variational upper bounds suggested herein are applied (Section 2.1), and outline known variational lower bounds of the likelihood of data, which turn out to be close to our upper bounds (Section 2.2).

2.1 Phylogenetic HMM Model

We consider the Phylogenetic HMM model described by Siepel and Haussler (2003). Since the model is given in terms of conditional probabilities, it is convenient to describe it as a DAG model, as done by Jojic et al. (2004). We repeat the description of the model from there with minor changes.

Given a domain of interest having a set of finite variables $\mathbf{s} = (s_1, \ldots, s_n)$ with a positive joint distribution $p(s)$, a DAG model for s is a pair (G, P) where G is a directed acyclic graph and P is a set of conditional probability distributions. A DAG model is also often called a Bayesian network (e.g. Pearl 1988, Jensen 2001). Each node s_i in G corresponds to a variable in s, and to a distribution $p(s_i|\mathbf{pa}(s_i))$, called a local probability distribution, where $\mathbf{pa}(s_i)$ are the parents of s_i in the graph. The joint distribution is given by $p(s) = \prod_{i=1}^{n} p(s_i|\mathbf{pa}(s_i))$. Consequently, the assumed independence relationships between random variables are represented through absence of edges in the model.

A DAG model structure that assumes that evolution takes place independently at each nucleotide site is illustrated in Figure 1a for a simple tree with five species. The



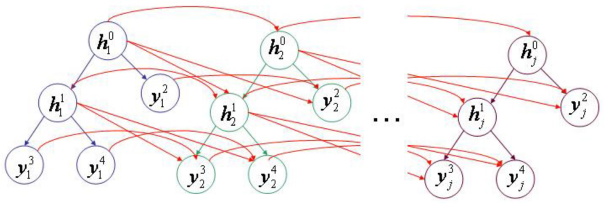


Fig. 1. Probabilistic phylogenetic trees expressed as DAG models. **(a)** The Neyman-Felsenstein tree model that assumes independent evolution in sites. **(b)** The dinucleotide phylogenetic HMM model suggested by Siepel and Haussler (2003).

unknown nucleotide in an ancestor species i at site j is denoted as h^i_j, and the observed nucleotide of an existing species i' at site j' is denoted as $y^{i'}_{j'}$. This is the usual model for which Felsenstein's algorithm for computing likelihood of data is readily applicable. The model of Siepel and Haussler (2003) does not assume that sites are independent, and therefore, edges that connect variables of adjacent sites are added (Figure 1b). This figure illustrates the phylogenetic HMM model of Siepel and Haussler (2003). In this model, a nucleotide of species i at site j depends on the nucleotide of that species at site $j-1$, and its ancestor's nucleotides at sites $j-1$ and j. This model is also called the dinucleotide HMM model, since the two nucleotides of species i and k at site j, where k is the ancestor species of i, are dependent only on the two nucleotides of that species at site $j-1$. Additional more complex models are discussed in Siepel and Haussler (2003).

The local probability distributions of this model are determined by a continuous-time Markov matrix Q of base substitution rates. The matrix Q is of size 16×16, and given evolutionary time t, which is the branch length in the tree, the conditional probabilities $p(s^i_j, s^i_{j-1} | s^k_j, s^k_{j-1})$ are obtained from Q, where k is the ancestor species of i. This distribution then determines the desired probabilities $p(s^i_j | s^i_{j-1}, s^k_j, s^k_{j-1})$. Let $P(t)$ be the matrix of substitution probabilities for branch length t. Then $P(t)$ is given by the solution to the differential equation $\frac{d}{dt}P(t) = P(t)Q$ with initial conditions $P(0) = I$, which is $P(t) = e^{Qt}$. With Q being diagonalizable as $Q = S\Lambda S^{-1}$, the matrix $P(t)$ can be computed as $P(t) = Se^{\Lambda t}S^{-1}$, where $e^{\Lambda t}$ is the diagonal matrix obtained by exponentiating each element on the main diagonal of Λt.

A standard criterion to choose between two DAG models is to prefer a model with higher log-likelihood of the data. However, for the phylogenetic HMM model described here, computing the log-likelihood of data is not feasible, and therefore approximations are needed. In the next section we review known approximations that give lower bounds.

2.2 Variational Lower Bounds

The problem of computing the likelihood, $P(Y=y)=\sum_h P(Y=y, H=h))$, in DAG models is NP-hard (Cooper, 1990; Dagum & Luby, 1993), and although there are many DAG models where exact algorithms are feasible, there are others in which the time and space complexity makes the use of such algorithms infeasible. In these cases fast yet accurate approximations are desired. Herein, we call the task of computing the likelihood by the term inference.

Variational techniques such as the ones suggested by Jordan et al. (1999) are a powerful tool for efficient approximate inference that offers guarantees in the form of lower bounds. In particular, let $P(X)$ be a joint distribution over a set of discrete variables X with the goal to compute the marginal probability $P(Y=y)$, where $Y \subseteq X$. Further assume that this exact computation is not feasible. The idea is to replace P with a distribution Q for which exact inference is feasible, and compute a lower bound for $P(Y=y)$ by using Jensen's inequality:

$$\log P(y) = \log \sum_h Q(h) \frac{P(y,h)}{Q(h)} \geq \sum_h Q(h) \log \frac{P(y,h)}{Q(h)} = -D(Q(H) \,\|\, P(Y{=}y, H))$$

where $H = X \setminus Y$ and $D(\cdot \| \cdot)$ denotes the KL divergence between two probability distributions.

To obtain tight lower bounds several variational algorithms were devised that try to find an approximating distribution Q which minimizes the KL divergence between Q and the target distribution P ([15,8,17,1,7]). Variational approaches such as the mean field, generalized mean field, and structured mean field differ only with respect to the family of approximating distributions that can be used. Such variational techniques were applied by Jojic et al. (2004) to find lower bounds for the phylogenetic HMM models. The lower bounds computed in the results section herein use a newer algorithm for finding tighter lower bounds suggested by Geiger et al. (2006).

3 Variational Upper Bounds

We denote distributions by $P(x)$ and $Q(x)$, where Q is not necessarily a normalized distribution. Let X be a set of variables and x be an instantiation of these variables. Let $P(x) = \prod_{i=1}^n \Psi_i(d_i)$ and $Q(x) = \prod_{i=1}^n \Phi_i(d_i)$ where d_i is the projection of the instantiation x to the variables in $D_i \subseteq X$, the subsets $\{D_i\}_{i=1}^n$ can overlap, and n is the number of sets D_i. Consider the marginal probability $P(Y=y) = \sum_h P(y,h) = \sum_h \prod_i \Psi_i(d_i)$ where $X = Y \cup H$. We assume throughout that $Q(x)$ is *tractable* in the sense that the marginal probability $Q(Y=y)$ is feasible to compute, while $P(Y=y)$ is not feasible to compute.

We now develop an upper bound for $P(Y=y)$ as summarized in Theorems 1 & 2.

According Jensen's inequality, if f is a concave function and $Z=\{z_1,\ldots,z_n\}$ is a set of real numbers then $f(\sum_{i=1}^n w_i z_i) \geq \sum_{i=1}^n w_i f(z_i)$, where each $w_i \geq 0$ and $\sum_{i=1}^n w_i = 1$. By using the concavity of the log function and Jensen's inequality for concave functions, we get the following upper bound:

$$P(Y=y) = \sum_h e^{\log \prod_i \Psi_i(d_i)} = \sum_h e^{\sum_i w_i(h) \log \Psi_i(d_i)^{(1/w_i(h))}} \tag{1}$$

$$\leq \sum_h e^{\log \sum_i w_i(h) \Psi_i(d_i)^{(1/w_i(h))}} = \sum_h \sum_i w_i(h) \Psi_i(d_i)^{(1/w_i(h))}$$

where $\sum_i w_i(h) = 1$ for every instantiation h. Note that this bound can be obtained also by using the weighted power means inequality[1]. Eq. 1 holds with equality regardless of the values of potentials Ψ if and only if

$$w_i(h) = \frac{\log \Psi_i(d_i)}{\log P(h,y)}. \tag{2}$$

Given a tractable distribution $Q(x) = \prod_{i=1}^n \Phi_i(d_i)$ we set $w_i(h) = \frac{\log \Phi_i(d_i)}{\log Q(h,y)}$, which approximates the optimal but intractable choice given by Eq. 2.

With these values for $w_i(h)$, and using the identity $x^{\frac{logy}{z}} = y^{\frac{logx}{z}}$, Eq. 1 can be written as:

$$P(Y=y) \leq \sum_h \sum_i \frac{\log \Phi_i(d_i)}{\sum_k \log \Phi_k(d_k)} \prod_m \Phi_m(d_m)^{\frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)}} \tag{3}$$

The upper bound in Eq. 3 holds with equality if Q equals P, because by replacing all occurrences of $\Phi_i(d_i)$ with $\Psi_i(d_i)$ we get

$$P(Y=y) \leq \sum_h \sum_i \frac{\log \Psi_i(d_i)}{\sum_k \log \Psi_k(d_k)} \prod_m \Psi_m(d_m) = \sum_h \prod_m \Psi_m(d_m) = P(Y=y)$$

Eq. 3 remains hard to compute until the sum over h is divided into smaller sums. To obtain a tractable bound we use the arithmetic-geometric means inequality, $\frac{1}{n}\sum_k \log \Phi_k(d_k) \geq \prod_k \log \Phi_k(d_k)^{1/n}$, where $\log \Phi_k(d_k) > 0$. To use this inequality we set all potentials $\Phi_i(d_i)$ to be greater than 1. The resulting tractable upper bound stemming from Eq. 3 is the following:

$$P(Y=y) \leq \frac{1}{n} \sum_h \sum_{i=1}^n \log \Phi_i(d_i) \prod_m \frac{\Phi_m(d_m)^{\frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)}}}{\log \Phi_m(d_m)^{1/n}} \tag{4}$$

Consequently, the following theorem holds.

[1] The weighted power mean $M_w^r(Z)$ of a series of real numbers $Z = \{z_1, \ldots, z_n\}$ is defined for every real $r \in \mathbb{R}$ as

$$M_w^r(z_1, \ldots, z_n) = \begin{cases} \left[\sum_{i=1}^n w_i z_i^r\right]^{1/r} & \text{if } r \neq 0 \\ \prod_{i=1}^n z_i^{w_i} & \text{if } r = 0 \end{cases}$$

where $w_1, \ldots, w_n$ are positive real numbers such that $\sum_{i=1}^n w_i = 1$. Note that $M_w^r(Z) \xrightarrow{r \to 0} M_w^0(Z)$.

The power mean inequality states that for two real numbers s, t, the relation $s < t$ implies $M_w^s < M_w^t$, and the upper bounds are obtained by setting $s = 0$, $t = 1$, and $z_i = \Psi_i(d_i)^{(1/w_i)}$.

Theorem 1 (upper bound). *Let H and Y be two disjoint sets of variables such that $H \cup Y = X$, and let $P(x)$ and $Q(x)$ be distributions that factor according to $P(x) = \prod_{i=1}^{n} \Psi_i(d_i)$ and $Q(x) = \prod_{i=1}^{n} \Phi_i(d_i)$ where d_i is the projection of the instantiation x to the variables in $D_i \subseteq X$. Then the following is an upper bound on $P(Y = y)$,*

$$P(Y = y) \leq \frac{1}{n} \sum_{i} \sum_{D_i} \log \Phi_i(d_i) \left[\sum_{h \setminus D_i} \prod_{m} \frac{\Phi_m(d_m)^{\frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)}}}{\log \Phi_m(d_m)^{1/n}} \right] \tag{5}$$

Proof: The proof is immediate from Eq. 4 where we replace the sums over i and h, and divide the sum over h such that first we sum over variables in D_i and then over the rest of the variables in H. □

Assuming that $M = \max_i\{|D_i|\}$ is at most a given constant, the time needed to compute the bound given in Eq. 5 is linear in the number of variables in the model and proportional to the time needed to compute $Q(y)$. Therefore, the tractability of this bound is a direct consequence of the assumption of tractable inference on distribution Q.

Since the maximal size M of the sets in the model can sometime be large enough to significantly slow computations of the upper bound, we develop a more efficient method to compute the upper bound that does not depend on M. To do so, we use the following lemma.

Lemma 1. *Given two sets of positive real numbers $X = \{x_1 \ldots, x_n\}$ and $Y = \{y_1 \ldots, y_n\}$ and a positive real number r, the following inequalities hold.*
If $0 < r \leq 1$, then

$$\sum_{i=1}^{n} \frac{x_i^r}{y_i} \leq \left(\sum_{i=1}^{n} \frac{x_i}{y_i} \right)^r \cdot \left(\sum_{i=1}^{n} y_i^{-1} \right)^{1-r} .$$

If $1 \leq r < 2$, then

$$\sum_{i=1}^{n} \frac{x_i^r}{y_i} \leq \left(\sum_{i=1}^{n} \frac{x_i}{y_i} \right)^{2-r} \cdot \left(\sum_{i=1}^{n} \frac{x_i^2}{y_i} \right)^{r-1} .$$

For $r = 1$ equalities hold.

Proof: We use the Euclidean case of Hölder's inequality, stating that for two sets of positive real numbers $X = \{x_1 \ldots, x_n\}$ and $Y = \{y_1 \ldots, y_n\}$, and for two real numbers $p, q \geq 1$ such that $\frac{1}{p} + \frac{1}{q} = 1$,

$$\sum_{i=1}^{n} x_i \cdot y_i \leq \left(\sum_{i=1}^{n} x_i^p \right)^{1/p} \cdot \left(\sum_{j=1}^{n} y_j^q \right)^{1/q} .$$

For $0 < r \leq 1$, we get using Hölder's inequality,

$$\sum_{i=1}^{n} \frac{x_i^r}{y_i} = \sum_{i=1}^{n} \left(\frac{x_i}{y_i} \right)^r \cdot y_i^{r-1} \leq \left(\sum_{i=1}^{n} \left(\frac{x_i}{y_i} \right)^{r \cdot p} \right)^{1/p} \cdot \left(\sum_{i=1}^{n} y_i^{(r-1) \cdot q} \right)^{1/q} .$$

Setting $p = \frac{1}{r}$ and $q = \frac{1}{1-r}$ we get

$$\sum_{i=1}^{n} \frac{x_i^r}{y_i} \le \left(\sum_{i=1}^{n} \frac{x_i}{y_i} \right)^{r} \cdot \left(\sum_{i=1}^{n} y_i^{-1} \right)^{1-r} .$$

Similarly, for $1 \le r < 2$, we get using Hölder's inequality,

$$\sum_{i=1}^{n} \frac{x_i^r}{y_i} = \sum_{i=1}^{n} \left(\frac{x_i}{y_i} \right)^{2-r} \cdot \left(\frac{x_i^2}{y_i} \right)^{r-1} \le \left(\sum_{i=1}^{n} \left(\frac{x_i}{y_i} \right)^{(2-r)\cdot p} \right)^{1/p} \cdot \left(\sum_{i=1}^{n} \left(\frac{x_i^2}{y_i} \right)^{(r-1)\cdot q} \right)^{1/q} .$$

Setting $p = \frac{1}{2-r}$ and $q = \frac{1}{r-1}$ we get

$$\sum_{i=1}^{n} \frac{x_i^r}{y_i} \le \left(\sum_{i=1}^{n} \frac{x_i}{y_i} \right)^{2-r} \cdot \left(\sum_{i=1}^{n} \frac{x_i^2}{y_i} \right)^{r-1} .$$

□

Theorem 2 (Efficient upper bound). *Let H and Y be two disjoint sets of variables such that $H \cup Y = X$, and let $P(x)$ and $Q(x)$ be distributions that factor according to $P(x) = \prod_{i=1}^{n} \Psi_i(d_i)$ and $Q(x) = \prod_{i=1}^{n} \Phi_i(d_i)$ where $\Psi_i > 1$, $\Phi_i > 1$ and $\frac{\log \Psi_i}{\log \Phi_i} < 2$ for every $i = 1, \ldots, n$, and where d_i is the projection of the instantiation x to the variables in $D_i \subseteq X$. In addition, let U_i denote the set of instantiations of D_i for which $\Phi_i(d_i) \le \Psi_i(d_i)$, and let L_i denote the rest of instantiations of D_i. Then the following is an upper bound on $P(Y = y)$,*

$$P(Y = y) \le \frac{1}{n} \sum_{i} \left[\sum_{d_i \in L_i} \log \Phi_i(d_i) \Lambda_{L_i} + \sum_{d_i \in U_i} \log \Phi_i(d_i) \Lambda_{U_i} \right] \tag{6}$$

where

$$\Lambda_{L_i} = \left(\sum_{h \setminus D_i} \prod_{m} \frac{\Phi_m(d_m)}{\log \Phi_m(d_m)^{1/n}} \right)^{\frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)}} \cdot \left(\sum_{h \setminus D_i} \prod_{m} \frac{1}{\log \Phi_m(d_m)^{1/n}} \right)^{1 - \frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)}}$$

and

$$\Lambda_{U_i} = \left(\sum_{h \setminus D_i} \prod_{m} \frac{\Phi_m(d_m)}{\log \Phi_m(d_m)^{1/n}} \right)^{2 - \frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)}} \cdot \left(\sum_{h \setminus D_i} \prod_{m} \frac{\Phi_m(d_m)^2}{\log \Phi_m(d_m)^{1/n}} \right)^{\frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)} - 1}$$

Proof: Lemma 1 implies that when $\Phi_i(d_i) \ge \Psi_i(d_i) > 1$, we can replace every bracketed term $\sum_{h \setminus D_i} \prod_m \left[\Phi_m(d_m)^{\frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)}} / \log \Phi_m(d_m)^{1/n} \right]$ in Eq. 5 with Λ_{L_i} and when $1 < \Phi_i(d_i) < \Psi_i(d_i)$, we can replace it with Λ_{U_i}, since $\frac{\log \Psi_i(d_i)}{\log \Phi_i(d_i)} < 2$. □

Computing each term, Λ_{U_i} or Λ_{L_i}, involves only two sums of products, where each sum factors according to distribution Q. These computations can be performed by using any algorithm such as *bucket elimination algorithm* or the *sum-product algorithm* described by Dechter (1999) and Kschischang, Frey & Loeliger (2001) . According to Eq. 6 only a linear number of calls to such procedures are needed to obtain the upper bound.

If each potential Ψ_i and Φ_i is multiplied by a large factor α, all the terms $\frac{\log \Psi_i}{\log \Phi_i}$ approach one as α grows. This reduces the accuracy gap when using Hölder's inequality in Eq. 6 with $r = \frac{\log \Psi_i}{\log \Phi_i}$. In addition, note that multiplying the potentials Φ_i by α also serves the tightness of the arithmetic-geometric inequality used to obtain Eq. 5, since for each pair of potentials Φ_j and Φ_k, the ratio $\frac{\log \Phi_j}{\log \Phi_k}$ approaches one as α grows. A large enough α guarantees that $\frac{\log \Psi_i}{\log \Phi_i} < 2$ for all sets D_i and thus the applicability of Theorem 2. In our experiments we use $\ln \alpha = 300$.

4 Approximations for Phylogenetic HMM Models

The dinucleotide phylogenetic HMM model of Siepel and Haussler (2003), described in Section 2.1, lead to improvements over previous models in several biological tasks such as gene finding. But, despite its enhanced power, it also requires evaluating an intractable likelihood for the purpose of finding optimal parameters for the model. Jojic et al. (2004) used variational techniques, similar to the ones described in Section 2.2 to lower bound the likelihood of data, and showed that when the exact likelihood can be computed (although with much effort), the approximations were tight.

We use the upper bounds suggested in Section 3 to compute the likelihood of phylogenetic trees with a small error, by bounding it tightly from above and below. First, we show the upper bounds are close to the true likelihood when this can be computed. Then, for larger phylogenetic trees, where computing the exact likelihood is infeasible, we show the proximity of the lower and upper bounds. To set a tractable approximating distribution Q, we use a parameter k which determines its topology: sets that contain variables from sites ck and $ck+1$, for $c = 1, 2, 3, \ldots$, are split into two disjoint subsets, D_{i1} and D_{i2}, where D_{i1} contains only variables in D_i from site ck and D_{i2} contains the rest of the variables in D_i. Their respective potentials $\Phi_i(d_i)$ therefore factor according to $\Phi_i(d_i) = \Phi_{i1}(d_{i1})\Phi_{i2}(d_{i2})$. In our experiments we used $k = 10$ when computing the exact likelihood was feasible and $k = 5$ when the likelihood computation was infeasible. The lower bounds were obtained by using a recent variational algorithm called VIP* (Geiger et al., 2006).

We repeat each upper bound computation twice, with the difference of the way potentials Φ_i are chosen. The first choice is what we call non-informative (NI), where each potential $\Phi_i(d_i) = \prod_{j=1}^{m_i} \Phi_{ij}(d_{ij})$ is a product of m_i sub-potentials of sets $D_{ij} \subseteq D_i$. A sub-potential $\Phi_{ij}(d_{ij})$ is set to be the $1/m_i$ power of the average value of $\Psi_i(d_i)$ of all instantiations d_i consistent with d_{ij}. More formally, $\Phi_{ij}(d_{ij})$ $= \left(\frac{1}{|C_{d_{ij}}|} \sum_{d_i \in C_{d_{ij}}} \Psi(d_i)\right)^{1/m_i}$ where $C_{d_{ij}}$ is the set of instantiations d_i consistent with d_{ij}.

The second choice of potentials, called variational-based (VB), is based on variational algorithms, such as VIP*, that optimize the approximating distribution Q in order to set tight lower bounds on the likelihood. If the topology of Q given for these algorithms follows the factorization suggested in Section 3 (i.e. every potential Ψ_i in P has its corresponding potential Φ_i in Q), the potentials found by these optimization algorithms to lower bound the likelihood can also serve to upper bound it using the method proposed herein.

We ran the tests on data used by Siepel and Haussler (2003) that contains sequences from human in the region of the CFTR gene and homologous from eight mammals: chimp, baboon, cow, pig, cat, dog, mouse and rat. The sequences are aligned, and we used portions of this alignment to obtain our results. The substitution probabilities in all models were computed from the dinucleotide substitution matrix obtained by Jojic et al. (2004), and the branch lengths in each tree were randomly chosen, normally distributed around predetermined means. The first tests used two data sets, similar to those used by Jojic et al. (2004), where each set consisted of three sequences. The sequences in set A were taken from the cow, mouse and human genomes and were of length 30Knc, and the sequences in set B were taken from the cow, pig and dog genomes and were of length 20Knc. Figure 2a and 2b plot the upper bounds versus the exact log-likelihoods of trees with different branch lengths. Lower bounds are also shown in the figure to demonstrate the tightness level of these bounds. The average differences for the trees in set A between the upper bounds and the exact likelihoods were 1% for the NI method

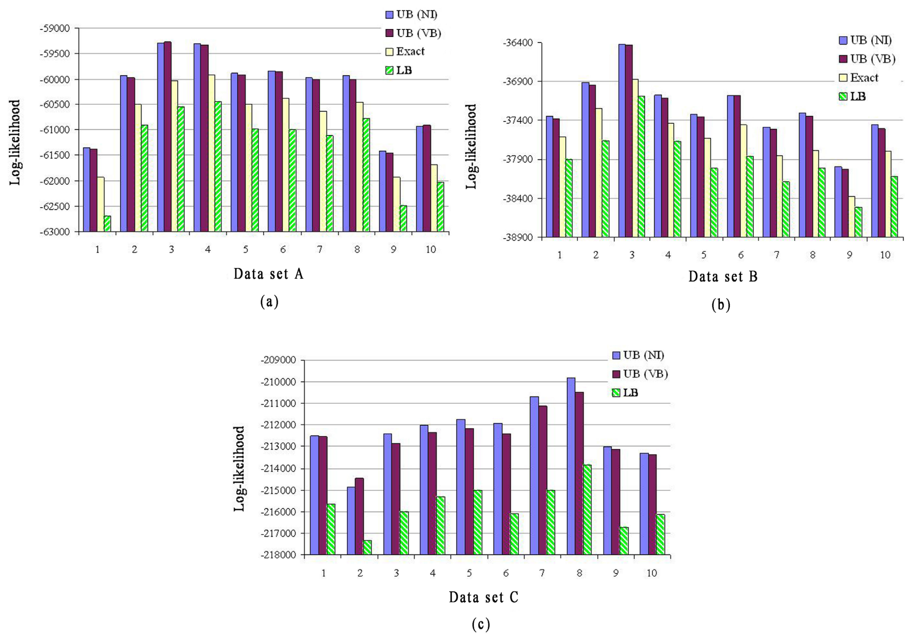
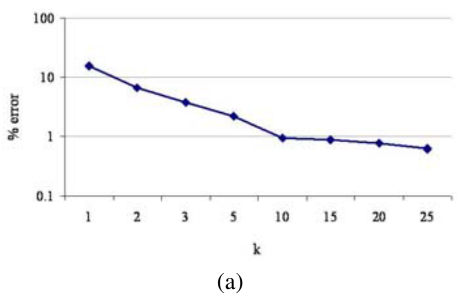


Fig. 2. Upper and lower bounds on the likelihood of data of phylogenetic HMM models for sets A, B and C with different branch lengths. **(a) & (b)** Bounds versus the exact likelihood for models of sets A and B. **(c)** Bounds for models of set C, for which computing the exact likelihood is infeasible.



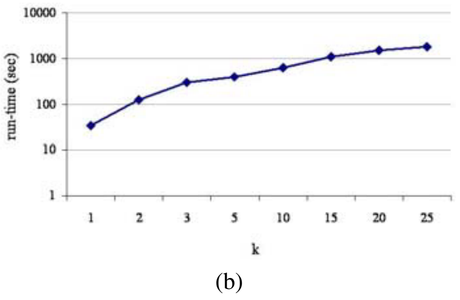


Fig. 3. Accuracy and run-time as a function of parameter k of decomposing the model. **(a)** Accuracy as a function of k. **(b)** Run-time as a function of k.

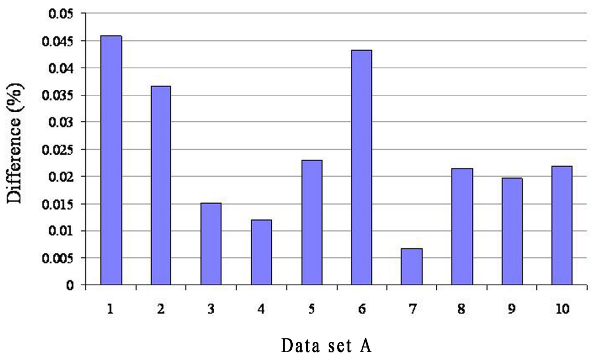


Fig. 4. The difference in accuracy between upper bounds computed via Eq. 5 and bounds computed via Eq. 6

and 0.95% for the VB method, and for trees in set B the average differences were 0.97% (NI) and 0.9% (VB).

The upper and lower bounds for an additional set of aligned sequences that contained sequences of length 30Knc from all nine organisms (Set C) are illustrated in Figure 2c. For this set it is infeasible to compute the exact likelihood, but the proximity of the upper and lower bounds allows us to predict the likelihood with a small error. The NI method yielded an average of 1.64% difference from the lower bounds and the VB method yielded an average of 1.52% from the lower bounds for the models in this set.

As shown in Figure 2, both choices of potentials (NI and VB) performed similarly, with a small advantage of the VB method over NI in most experiments. In other experiments we performed, we found that arbitrary choice of potentials often lead to significant decrease in the tightness of the bounds (up to 45%), and therefore an algorithm is desired to find potentials that lead to tight bounds.

The parameter k used for decomposing the tree model into parts of k sites is a trade-off between run-time and accuracy: the larger k is the more time consuming it is to compute the upper bounds, however, the bounds computed are also more accurate. The default value of k was set to 10 for trees in Set A. Figure 3 shows the results for these trees as a function of k in terms of accuracy and in terms of run-time.

Finally, we tested the difference in accuracy between upper bounds computed via Eq. 5 and those computed via Eq. 6. The expected run-time ratio between these two methods is the average probability table size in the model. Since no preprocessing such as summing over some variables was executed, the expected ratio was 81.25. As shown in Figure 4, the differences in accuracy of the upper bounds were negligible, less than 0.05% of their log value, when applied to phylogenetic trees in data set A. This implies that when the size of the probability tables is large, Eq. 6 is an attractive and efficient alternative to Eq. 5.

5 Discussion

Computing the likelihood of many probabilistic models is infeasible and calls for efficient approximations. Our results on phylogenetic models show that the suggested upper bounds are appreciably tight and together with other variational methods allow to compute the likelihood almost exactly in feasible time. We have also started using the upper bounds to approximate other probabilistic models and believe that they can be applied to a wide range of models and for various tasks. One additional task we explore is bounding the MAP assignment probability in order to set optimal parameters for models where finding the exact MAP assignment is infeasible. The goodness of the bounds heavily depends on the choice of an approximating distribution Q, and more work on choosing useful Q functions is desired, as indicated by Xing et al. (2004).

As with variational methods that offer lower bounds on the likelihood, if the dependence of variables under Q largely differs from their dependence under the target distribution P, these methods yield loose bounds. When exploring probabilistic models to genetic linkage analysis, as used by Fishlson and Geiger (2002), we found that the variational methods we used did not offer sufficiently good approximating distributions for these models, and therefore did not give tight enough bounds. Geiger et al. (2006) provided results of variational techniques on genetic linkage analysis problems and showed that although the lower bounds followed the shape of the likelihood function, the difference from the true log-likelihood reached 20%. The difficulty in finding good approximations to this model may lie in the level of determinism of the model: relaxing deterministic dependence relationships between variables reduced accuracy far more than when relaxing mild dependence relationships. When computing the upper bounds suggested herein for genetic linkage analysis, the results were within 10% from the true log-likelihood.

Ackowledgements

The research is supported by the Israel Science Foundation and the Israeli Science Ministry.

References

1. C. Bishop and J. Winn. Structured variational distributions in VIBES. In *Artificial Intelligence and Statistics*, Key West, Florida, USA, 2003. Society for Artificial Intelligence and Statistics.
2. G. Cooper. Probabilistic inference using belief networks is NP-hard. *Artificial Intelligence*, 42:393–405, 1990.
3. P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60(1):141–153, 1993.
4. R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1-2):41–85, 1999.
5. J. Felsenstein. Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, 17:368–376, 1981.
6. M. Fishelson and D. Geiger. Exact genetic linkage computations for general pedigrees. *Bioinformatics*, 18:S189–S198, 2002.
7. D. Geiger, C. Meek, and Y. Wexler. A variational inference procedure allowing internal structure for overlapping clusters and deterministic constraints. *Journal of Artificial Intelligence Research (JAIR)*, 27:1–23, 2006.
8. Z. Ghahramani and M. I. Jordan. Factorial hidden Markov models. *Machine Learning*, 29:245–273, 1997.
9. F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag New York, Inc., 2001.
10. V. Jojic, N. Jojic, C. Meek, D. Geiger, A. Siepel, D. Haussler, and D. Heckerman. Efficient approximations for learning phylogenetic HMM models from data. *Bioinformatics*, 20:161–168, 2004.
11. M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and S. L. K. *An introduction to variational methods for graphical models*. Learning Graphical Models. MIT Press, 1999.
12. F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, February 2001.
13. J. Neyman. Molecular studies of evolution: a source of novel statistical problems. In *S. S. Gupta and J. Yackel (eds), Statistical desicion theory and related topics*, pages 1–27. Academic Press, New York, 1971.
14. J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, San Mateo, CA, 1988.
15. L. K. Saul and M. I. Jordan. Exploiting tractable substructures in intractable networks. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 8, pages 486–492. The MIT Press, 1996.
16. A. Siepel and D. Haussler. Combining phylogenetic and hidden markov models in biosequence analysis. In *RECOMB '03: Proceedings of the seventh annual international conference on Research in computational molecular biology*, pages 277–286, New York, NY, USA, 2003. ACM Press.
17. W. Wiegerinck. Variational approximations between mean field theory and the junction tree algorithm. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 626–633, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.
18. E. P. Xing, M. I. Jordan, and S. Russell. Graph partition strategies for generalized mean field inference. In *AUAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 602–610, Arlington, Virginia, United States, 2004. AUAI Press.

Heuristics for the Gene-Duplication Problem: A $\Theta(n)$ Speed-Up for the Local Search[⋆]

Mukul S. Bansal[1], J. Gordon Burleigh[2], Oliver Eulenstein[1], and André Wehe[3]

[1] Department of Computer Science, Iowa State University, Ames, IA, USA
{bansal,oeulenst}@cs.iastate.edu
[2] National Evolutionary Synthesis Center Durham, NC, USA
jgb12@duke.edu
[3] Department of Electrical and Computer Engineering, Iowa State University, Ames, IA, USA
awehe@iastate.edu

Abstract. The gene-duplication problem is to infer a species supertree from a collection of gene trees that are confounded by complex histories of gene duplications. This problem is NP-hard and thus requires efficient and effective heuristics. Existing heuristics perform a stepwise search of the tree space, where each step is guided by an exact solution to an instance of a local search problem. We show how this local search problem can be solved efficiently by reusing previously computed information. This improves the running time of the current solution by a factor of n, where n is the number of species in the resulting supertree solution, and makes the gene-duplication problem more tractable for large-scale phylogenetic analyses. We verify the exceptional performance of our solution in a comparison study using sets of large randomly generated gene trees. Furthermore, we demonstrate the utility of our solution by incorporating large genomic data sets from GenBank into a supertree analysis of plants.

1 Introduction

The rapidly increasing amount of available genomic sequence data provides an abundance of potential information for phylogenetic analyses. Most phylogenetic analyses combine genes from presumably orthologous loci, or loci whose homology is the result of speciation. These analyses largely neglect the vast amounts of sequence data from gene families, in which complex evolutionary processes such as gene duplication and loss, recombination, and horizontal transfer generate gene trees that differ from species trees. One approach to utilize the data from gene families in phylogenetics is to reconcile their gene trees with species trees based on an optimality criterion, such as the gene-duplication model introduced by Goodman et al. [1]. This problem is a type of *supertree problem*, that is, assembling from a set of input gene trees a species supertree that contains all species found in at least one of the input trees. The decision version of the gene-duplication problem is NP-complete [2]. Existing heuristics aimed at solving the

[⋆] During this research, O.E. and M.S.B. were supported in part by NSF grant no. 0334832 and J.G.B. by NESCent NSF grant no. EF-0423641.

gene-duplication problem search the space of all possible supertrees guided by a series of exact solutions to instances of a local search problem [3]. The gene-duplication problem has shown much potential for building phylogenetic trees for snakes [4], vertebrates [5,6], *Drosophia* [7], and plants [8]. Yet, the run time performance of existing heuristics has limited the size of such studies. We improve on the best existing solution for the local search problem asymptotically by a factor of n, where n is the number of species from which sequences in the gene trees were sampled (that is the number of nodes in a resulting supertree). To show the applicability of our improved solution for the local search problem, we implemented it as part of standard heuristics for the gene-duplication problem. We demonstrate that the implementation of our method greatly improves the speed of standard heuristics for the gene-duplication problem and makes it possible to infer large supertrees that were previously difficult, if not impossible, to compute.

For convenience, the term "tree" refers to a rooted and full binary tree, and the terms "leaf-gene" and "leaf-species" refer to a gene or species that is represented by a leaf of a gene or species tree respectively throughout this work (unless otherwise stated).

Previous Results: The gene-duplication problem is based on the gene-duplication model from Goodman et al. In the following, we (i) describe the gene-duplication model, (ii) formulate the gene-duplication problem, and (iii) describe a heuristic approach of choice [3] to solve the gene-duplication problem.

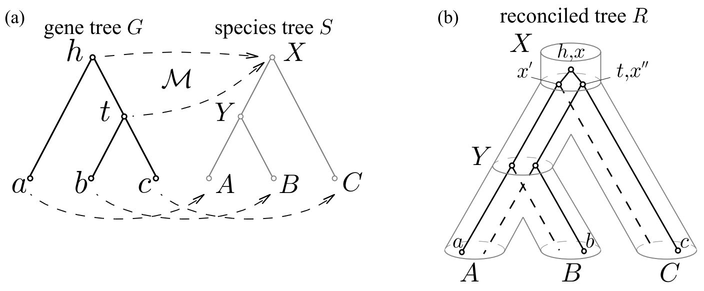


Fig. 1. (a) Gene tree G and species tree S are comparable, as the mapping from the leaf-genes to the leaf-species indicates. $\mathcal{M}$ is the lca-mapping from G to S. (b) R is the reconciled tree for G and S. In species X of R gene x duplicates into the genes x' and x''. The solid lines in R represent the embedding of G into R.

Gene-duplication model: The gene-duplication (GD) model [9,10,11,12,13,14,15,16] explains incompatibilities between a pair of "comparable" gene and species trees through gene duplications. A gene and a species tree are *comparable*, if a sample mapping, called *s-mapping*, exists that maps every leaf-gene to the leaf-species from which it was sampled. Fig. 1 depicts an example. Gene tree G is inferred from the leave-genes that were sampled from the leaf species of the species tree described by the s-mapping. However, both trees describe incompatible evolutionary histories. The GD model explains such incompatibilities by reconciling the gene tree with postulated gene duplications. For example, in Fig. 1 a reconciled gene tree R can be theoretically inferred from the species tree S by duplicating a gene x in species X into the copies x' and x'' and letting both copies speciate according to the topology of S. In this case, the gene tree can be embedded into the reconciled tree. Thus, the gene tree can be reconciled by the gene duplication x to explain the incompatibility. The gene duplications that are necessary under the GD model to reconcile the gene tree can be described by the *lca-mapping* $\mathcal{M}$, which is an extension of the given s-mapping. $\mathcal{M}$ maps every gene in the gene tree to the most recent species in the species tree that could have contained the gene. To make the definition precise, $\mathcal{M}$ maps each gene to the least common ancestor of the species from which the leaf-genes of the subtree rooted at the gene were sampled (given by the s-mapping). A gene in the gene tree is said to be a *gene duplication* if it has a child with the same lca-mapping. In Fig. 1 gene h and its child t map under the lca-mapping to the same species X. The *reconciliation cost* for a gene tree and a comparable species tree is measured in the number of gene duplications in the gene tree induced by the species tree. The *reconciliation cost* for a given set of gene trees and a species tree is the sum of the reconciliation cost for every gene tree in the set and the species tree. The lca-mapping is linear time computable on a PRAM [13] through a reduction from the least common ancestor problem [17,18]. Hence, the reconciliation cost for a set of gene trees and a species tree is computable in linear time.

Gene-duplication problem and heuristic: The *gene-duplication problem* is to find, for a given set of gene trees, a comparable species tree with the minimum reconciliation cost. The decision variant of this problem and some of its characterizations are NP-complete [2,19] while some parameterizations are fixed parameter tractable [20,21]. Therefore, in practice, heuristics are commonly used for the gene-duplication problem even if they are unable to guarantee an optimal solution. However, GeneTree [22], an implementation of a standard local search heuristic for the gene-duplication problem, demonstrated that the gene-duplication problem can be an effective approach for inferring species phylogenies. While the local search heuristic for the gene-duplication problem performs reasonably well in computing smaller sized instances, it does not allow the computation of larger species supertrees. In this heuristic, a *tree graph*, representing the tree space, is defined for the given set of gene trees and some fixed tree edit operation. The nodes in the tree graph are the species trees which are comparable with every given gene tree. An edge is drawn between two nodes exactly if the corresponding trees can be transformed into each other by the tree edit operation. The *reconciliation cost* of a node in the graph is the reconciliation cost of the species tree represented by that node and the given gene trees. Given a starting node in the tree graph, the heuristic's task is to find a maximal-length path of steepest descent in the reconciliation cost of its nodes and to return the last node on such a path. This path is found by solving the local search problem for every node along the path. The *local search problem* is to find a node with the minimum reconciliation cost in the neighborhood of a given node. The time complexity of the local search problem depends on the tree edit operation used. An edit operation of interest is the rooted subtree pruning and regrafting (rSPR) operation [23,24]. Given a tree S, an rSPR operation can be performed in three steps: (i) prune some subtree P from S, (ii) add a root edge to the remaining tree S, (iii) regraft P into an edge of the remaining tree S. The resulting tree graph is connected and every node has a degree of $\Theta(n^2)$, where n is the size of a species tree comparable to the given gene trees. Assuming, for convenience, similar gene tree and species tree sizes, the local search problem for the rSPR edit operation can be solved naively in $\Theta(n^3)$ time per gene tree. If there are k gene trees then this gives a total time bound of $O(kn^3)$. This is the best-known algorithm to solve the local search problem for rSPR operations. In practice, the cubic run time typically allows only the computation of smaller supertrees [3]. A common approach to overcome this limitation is to consider only an $O(n)$ cardinality subset of the rSPR neighborhood at each node by using the rooted nearest neighbor interchange (rNNI) edit operation. The local search problem for the rNNI edit operation can be solved in $O(kn^2)$ time. We solve the local search problem for the rSPR edit operations within the same $O(kn^2)$ time bound.

Contribution of the Manuscript: First we introduce an algorithm that, irrespective of the sizes of the gene trees, improves the run time of the current solution by $\Theta(n)$, where n is the size of any species tree resulting from the given gene trees. To support typical input gene trees, our algorithm also allows multiple leaf-genes from the same gene tree to map to a single leaf-species. This algorithm was implemented as part of a standard heuristic for the gene-duplication problem, and we compared the run times of our implementation and the program GeneTree, which can infer species trees using the same local search heuristic. Finally, we demonstrate the ability of our heuristic to utilize gene-family sequences to construct large subtrees of the Tree of Life.

Organization of the Manuscript: Section 2 introduces basic terminology and problem definitions. In Sect. 3 we formally introduce the local search problem for the rSPR tree edit operation and our approach for solving it. To solve this refined local search problem we study gene duplication properties when a tree is modified using rSPR operations in Sect. 4. In Sect. 5 we introduce our algorithm for the (refined) local search problem, show its correctness and analyze its run time. Experimental results are presented in Sect. 6 and concluding remarks appear in Sect. 7. In the interest of brevity we shall omit the formal proofs for the lemmas and theorems presented herein.

2 Basic Notation and Preliminaries

Recall that throughout this work the term *tree* refers to a rooted full binary tree, unless otherwise stated. Given a tree T, let $V(T)$ and $E(T)$ denote the node and edge sets of T respectively. $\mathsf{Root}(T)$ denotes the root node of T, and $\mathsf{Le}(T)$ the leaf set of T. Given a node $v \in V(T)$: (i) $\mathsf{Pa}_T(v)$ is the parent node of v, (ii) $\mathsf{Ch}_T(v)$ denotes the set of children of v, (iii) T_v denotes the complete subtree of T rooted at node v, and (iv) a node $u \in V(T_v)\backslash\{v\}$ is a *(proper) descendant* of v. Two nodes with the same parent are called *siblings* of each other. The *least* *common ancestor* of a set $L \subseteq \mathsf{Le}(T)$ in T is defined to be the node $v \in V(T)$ such that $L \subseteq \mathsf{Le}(T_v)$ and $L \nsubseteq \mathsf{Le}(T_u)$ for any descendant u of $v \in V(T)$.

A *species tree* and a *gene tree* are full binary trees that represent the evolutionary relationships between species and genes (of a gene family) respectively. In the following we define the gene-duplication problem and the terms necessary for its definition. Let G be a gene tree and S be a species tree.

Comparability: The trees G and S are *comparable* if $\mathsf{Le}(G) \subseteq \mathsf{Le}(S)$. A set of gene trees $\mathcal{G}$ and S are *comparable* if $\mathsf{Le}(S) = \bigcup_{G \in \mathcal{G}} \mathsf{Le}(G)$.

Gene duplication: The *(lca-)mapping* $\mathcal{M}_{G,S}\colon V(G) \to V(S)$ is defined for comparable trees G and S such that $\mathcal{M}_{G,S}(v)$ is the least common ancestor of $\mathsf{Le}(G_v)$ in S. A node $v \in V(G)$ is a *gene duplication* if there exists a child u of $v \in V(G)$ such that $\mathcal{M}_{G,S}(v) = \mathcal{M}_{G,S}(u)$.

Reconciliation cost: (i) The *reconciliation cost for G and S* is $\Delta(G, S) = |\{v\colon v \in V(G)$ and v is a gene duplication $\}|$. (ii) The *reconciliation cost for a set of gene trees $\mathcal{G}$ and S* is $\Delta(\mathcal{G}, S) = \sum_{G \in \mathcal{G}} \Delta(G, S)$. (iii) The *reconciliation cost for a set of gene trees $\mathcal{G}$* is $\Delta(\mathcal{G}) = \min_{S \in \mathcal{S}} \Delta(\mathcal{G}, S)$, where $\mathcal{S}$ is the set of all species trees that are comparable with $\mathcal{G}$.

The GENE-DUPLICATION *problem*
Instance: A set $\mathcal{G}$ of gene trees.
Find: A species tree S_{OPT} such that $\Delta(\mathcal{G}) = \Delta(\mathcal{G}, S_{OPT})$.

3 Refining the Local Search Problem

The gene-duplication problem is heuristically approached by repeatedly solving the local search problem for the rSPR edit operation. In this section we first give definitions for the rSPR operation and the local search problem that were motivated in the introduction. Then we observe that the local search problem can be solved by dividing it into problem instances of the restricted local search problem, which we will introduce here. Finally, we present our central idea for solving the restricted local search problem efficiently.

The rSPR operation: The rSPR operation for a tree S is defined as cutting any edge, say $\{u, v\}$, where $u = \mathsf{Pa}_S(v)$, and thereby pruning a subtree, S_v, and then regrafting the subtree by the same cut edge in one of the following ways:

1. *Regrafting S_v into an edge $e \in S \backslash S_v$*: Creating a new node u' which subdivides e and regrafting the subtree by the cut edge at node u'. Then, either suppressing the degree-two node u or, if u is the root of S, deleting u and the edge incident with u, making the other end-node of this edge the new root.
2. *Regrafting S_v above* $\mathsf{Root}(S)$: Creating a new root node u' and a new edge between u' and the original root. Then regrafting the subtree by the cut edge at node u' and suppressing the degree-two node u.

Note that the rSPR operation involves deleting a node in the original tree and creating a new one where the subtree is regrafted. Throughout this text we assume that the new node created is given the same label as the node removed. This forms a new tree whose leaf set is the same as the original tree.

Consider an rSPR operation on the tree S that prunes the subtree S_v. We define $\mathsf{rSPR}(S, v, u)$ to be the tree obtained by regrafing S_v in one of the following two ways: (i) if $u \neq \mathsf{Root}(S)$, then V_s is regrafted into the edge $(u, \mathsf{Pa}_S(u))$, and (ii) if $u \neq \mathsf{Root}(S)$, then S_v is regrafted above $\mathsf{Root}(S)$. The set of trees into which S can be transformed by regrafting only S_v is $\mathsf{rSPR}(S, v) = \bigcup_{w \in V(S) \setminus V(S_v)} \mathsf{rSPR}(S, v, w)$. The set of trees into which S can be transformed by one rSPR operation is $\mathsf{rSPR}(S) = \bigcup_{v \in V(S) \setminus \{\mathsf{Root}(S)\}} \mathsf{rSPR}(S, v)$.

The specific local search problem defined for rSPR operations is called the neighborhood search problem.

The NEIGHBORHOOD-SEARCH (NS) *problem*
Instance: A gene tree set $\mathcal{G}$, and a comparable species tree S.
Find: The reconciliation cost for every tree in $\mathsf{rSPR}(S)$.

Restricting the NS Problem: We will show that the NS problem can be solved without computing the reconciliation cost for every tree in the neighborhood of S separately. Therefore we divide the NS problem into subproblems, called restricted neighborhood search problems, that can be solved efficiently by reusing previously computed information.

The RESTRICTED-NEIGHBORHOOD-SEARCH (RNS) *problem*
Instance: A triple $(\mathcal{G}, S, P)$, where $\mathcal{G}$ is a set of gene trees, S a comparable species tree, and P is a subtree of S.
Find: The reconciliation cost for every tree in $\mathsf{rSPR}(S, \mathsf{Root}(P))$.

Observation 1. *The* NS *problem on S can be solved by solving the* RNS *problem for each subtree of S.*

Our Idea to solve the RNS Problem: To solve the RNS problem instance $(\mathcal{G}, S, P)$ we first determine the reconciliation cost $\Delta(\mathcal{G}, \Re)$ for a particular tree $\Re \in \mathsf{rSPR}(S, \mathsf{Root}(P))$. $\Re$ is the tree obtained after pruning and regrafting P to the root of S (see Fig. 2(a)). After this initial step the reconciliation cost $\Delta(\mathcal{G}, S')$ for each tree within $S' \in \mathsf{rSPR}(S, \mathsf{Root}(P))$ can be determined in amortized $O(|\mathcal{G}|)$ time by following a particular order. Beginning with $\Re$ the subtree P is stepwise "moved down" in the tree S using rSPR operations (see Fig. 2(a)). We define the *move-down* operation for the pruned subtree P of the tree S as the rSPR operation which produces a tree $\mathsf{rSPR}(S, \mathsf{Root}(P), w)$, where $w \in \mathsf{Ch}_S(v)$ for the sibling v of $\mathsf{Root}(P)$.

The set $movedowns_S(P)$ consists of all species trees that can be obtained by performing successive move-down operations starting from $\Re$ with a fixed pruned subtree P.

Observation 2. $\mathsf{rSPR}(S, \mathsf{Root}(P)) = movedowns_S(P) \bigcup \{\Re\}$.

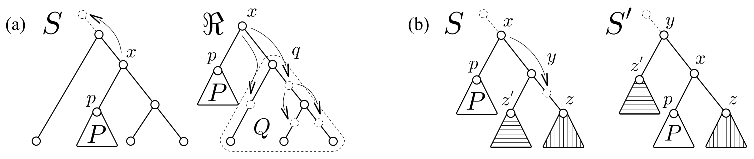


Fig. 2. (a): The tree $\Re$ is obtained from S after pruning and regrafting P to the root. Each tree in $\mathsf{rSPR}(S, \mathsf{Root}(P))$ can be obtained by starting from $\Re$ and successively performing move-down operations. (b): The subtree on the right, S', is obtained from S by moving x and P to the right subtree of y.

In Sect. 4 we show how the reconciliation cost is affected by move-down operations. These properties allow the design of an efficient algorithm for the RNS problem as shown in Sect. 5.

Naming Convention for this work: We establish the following notation throughout this work. The pruned subtree under study is denoted by P, its root node by p, and the parent of p by x. In the tree $\Re$, the sibling of p is q, and the subtree rooted at q is Q (see Fig. 2(a)). Note, $\mathsf{Root}(\Re)$ is always x.

The sibling of p is always denoted by y. Note, q and y refer to the same node in the tree $\Re$. A general situation is dipicted in Fig. 2(b). In general, g is used to refer to a node in a gene tree, and s to refer to a node in the species tree.

4 Structural Properties

Let G be a gene tree and S a species tree. In this section we study first the effect of move-down operations on the mapping $\mathcal{M}_{G,S}$ and the gene duplication status of genes in G. Finally, we describe the effect on the mapping after a sequence of move-down operations for a fixed pruned subtree.

A Single Move-down Operation: Consider a move-down operation that changes tree S into tree $S' = \mathsf{rSPR}(S, p, z)$, where $z \in \mathsf{Ch}_S(y)$. Fig. 2(b) shows an example for a move-down operation. Further, let $\mathcal{M}^{-1}_{G,S}(v)$ denote the set of nodes in G that map to node $v \in V(S)$ under the mapping $\mathcal{M}_{G,S}$. In the following we study the effects of the move-down operation on the mapping $\mathcal{M}_{G,S}$ and the gene duplication status.

Relating $\mathcal{M}_{G,S}$ and $\mathcal{M}_{G,S'}$:

Lemma 1. *$\mathcal{M}^{-1}_{G,S}(v) = \mathcal{M}^{-1}_{G,S'}(v)$, for all $v \in V(S) \backslash \{x, y\}$.*

Lemma 2. *$\mathcal{M}^{-1}_{G,S'}(x) \subseteq \mathcal{M}^{-1}_{G,S}(x)$ and $\mathcal{M}^{-1}_{G,S}(y) \subseteq \mathcal{M}^{-1}_{G,S'}(y)$.*

Effects on the gene duplication status: Based on the observations from Lemmas 1 and 2, the following three lemmas characterize the possible change in the gene duplication status of nodes in G.

Lemma 3. *The gene duplication status for any node in G that does not map to x under mapping $\mathcal{M}_{G,S}$ remains unchanged.*

Lemma 4. *If a node $g \in \mathcal{M}^{-1}_{G,S}(x)$ is not a gene duplication under mapping $\mathcal{M}_{G,S}$, then it becomes a gene duplication under mapping $\mathcal{M}_{G,S'}$ if and only if one of the children of g maps to node y in S.*

Lemma 5. *Let $g \in \mathcal{M}^{-1}_{G,S}(x)$ be a gene duplication under mapping $\mathcal{M}_{G,S}$ and z' be the sibling of z in S. Under the mapping $\mathcal{M}_{G,S'}$ the node g will lose its gene duplication status if and only if both of the following hold:*

1. *Under $\mathcal{M}_{G,S}$ one of the two children $b \in \mathsf{Ch}_G(g)$ maps to x and the other child maps to a node in the subtree $S_{z'}$.*
2. *Under $\mathcal{M}_{G,S'}$ node b maps to x.*

A Sequence of Move-down operations: We describe changes in the mapping $\mathcal{M}_{G,\Re}$ when move-down operations for a subtree P rooted at a child of $\mathsf{Root}(\Re)$ are successively performed to obtain a tree $S' \in \mathit{movedowns}_S(P)$.

The following proposition follows from Lemmas 1 and 2.

Proposition 1. *$\mathcal{M}_{G,\Re}(g)$ may only differ from $\mathcal{M}_{G,S'}(g)$, if $g \in \mathcal{M}^{-1}_{G,\Re}(x)$.*

Based on Proposition 1 we are left to characterize the differences between $\mathcal{M}_{G,\Re}(g)$ and $\mathcal{M}_{G,S'}(g)$ for all $g \in \mathcal{M}^{-1}_{G,\Re}(x)$. For this, we determine the nodes in G that can change in their mapping or that can be responsible for such a change.

The mapping of a node $g \in \mathcal{M}^{-1}_{G,\Re}(x)$ can change caused by a change in the mapping of its children. Such children would then be elements in $\mathcal{M}^{-1}_{G,\Re}(x)$ by Proposition 1. However, a change in the mapping can also be caused by other children, called supporting nodes, whose mapping does not change.

Definition 1 (Supporting nodes). *A node $g \in V(G)$ is a* supporting node*, if (i) $\mathcal{M}_{G,\Re}(g) \in V(Q)$ and (ii) $\mathsf{Pa}_G(g) \in M^{-1}_{G,\Re}(x)$.*

Definition 2 (Partial Gene Tree Γ). *The* partial gene tree *Γ is the subgraph of G induced by the set $\{g \in V(G) : g \in \mathcal{M}^{-1}_{G,\Re}(x)$ or g is a supporting node$\}$.*

Note, Γ is a binary tree (not necessarily full binary), and all its leaf nodes are supporting nodes.

The nodes in Γ induce a subtree in G and identify exactly the nodes whose mapping can change or that are responsible for such a change. The supporting nodes in Γ map to nodes in Q under mapping $\mathcal{M}_{G,\Re}$. Let this define an initial mapping from the leaves of Γ to the nodes in Q. This initial mapping can be extended to the (lca-)mapping $\mathcal{M}_{\Gamma,Q}$. All the internal nodes in Γ map to the root node of $\Re$, because they have at least one descendant in G that maps to a node in P under mapping $\mathcal{M}_{G,\Re}$. The mapping $\mathcal{M}_{\Gamma,Q}$ shows where those nodes would map under mapping $\mathcal{M}_{G,\Re}$ if all nodes in $\mathcal{M}^{-1}_{G,\Re}(s)$, for all $s \in V(P)$, are removed from G.

Based on the mapping $\mathcal{M}_{\Gamma,Q}$, we have the following lemma.

Lemma 6. *Let $s = \mathcal{M}_{\Gamma,Q}(\gamma)$ where γ is an internal node of Γ, and let $S' \in movedown_{\Re}(P)$. The location of node $\mathcal{M}_{G,S'}(\gamma)$ depends on the edge e in $E(Q)$ into which P is regrafted, as follows:*

1. *$\mathcal{M}_{G,S'}(\gamma) = x$, if e is on the path from q to node s in $V(Q)$.*
2. *$\mathcal{M}_{G,S'}(\gamma) = s$, if e is an edge in $E(Q_s)$.*
3. *$\mathcal{M}_{G,S'}(\gamma)$ is a node on the path from q to s, but not q or s, otherwise.*

5 Solving the RNS Problem

Based on the results obtained in the previous section we will first design an efficient algorithm, called ReconciliationCostTree (RCT), which solves the RNS problem for one input gene tree. Algorithm FastRNS then makes use of RCT to solve the RNS problem. We then show the correctness and analyze the run time of FastRNS.

Algorithm RCT(G, S, P)**:** The input for RCT is a gene tree G, a comparable species tree S, and subtree P to be pruned. The first step in the algorithm is to obtain the tree $\Re$ (see Fig. 2(a)). Recall that $x = \mathsf{Root}(\Re)$, $p = \mathsf{Root}(P)$, q denotes the sibling of p, and $Q = \Re_q$.

The output $\widetilde{Q}$ is a $W\colon V(\widetilde{Q}) \to \mathbb{N}_0$ node weighted version of tree Q, where $W(s) = \Delta(G, S')$ for $S' = rSPR(S, p, s)$.

Initialization: Create $\Re$ and initialize two counters $\mathsf{g}(s)$ and $l(s)$ with 0, for each node $s \in V(Q)$. Then, compute the mapping $\mathcal{M}_{G,\Re}$, the tree Γ, and the mapping $\mathcal{M}_{\Gamma,Q}$.

Computing the values for g and l: For each leaf $\gamma \in \mathsf{Le}(\Gamma)$ that has no sibling we do the following: If $\mathcal{M}_{\Gamma,Q}(\gamma) = \mathcal{M}_{\Gamma,Q}(\mathsf{Pa}_{\Gamma}(\gamma))$, then we increment $\mathsf{g}(\mathcal{M}_{\Gamma,Q}(\gamma))$ by 1. Similarly, for each leaf $\gamma \in \mathsf{Le}(\Gamma)$ that has a sibling, we do the following: Let $\alpha = \mathsf{Pa}_{\Gamma}(\gamma)$, σ be the sibling of γ, $a = \mathcal{M}_{\Gamma,Q}(\alpha)$, and $\mathsf{Ch}_Q(a) = \{b, c\}$. If σ maps into Q_b and γ into Q_c under mapping $\mathcal{M}_{\Gamma,Q}$, then increment the counter $l(b)$ by 1. Further, if σ maps into Q_c and γ into Q_b, then increment the counter $l(c)$ by 1.

The value $\mathsf{g}(s)$, represents the number of additional nodes from G that will become gene duplications when P is regrafted onto the edge $\{s, t\}$, $t \in \mathsf{Ch}_Q(s)$, from the edge $\{\mathsf{Pa}_Q(s), s\}$. The value $l(s)$ represents the number of nodes from G that will lose their gene duplication status when P is regrafted onto the edge $\{\mathsf{Pa}_Q(s), s\}$ from the edge $\{\mathsf{Pa}_Q(\mathsf{Pa}_Q(s)), \mathsf{Pa}_Q(s)\}$.

Computing $\widetilde{Q}$: The tree $\widetilde{Q}$ is initialized to be Q and its node weights are set to 0. Set $d = \Delta(G, \Re)$. For each node s in a preorder traversal on the tree $\widetilde{Q}$, we calculate the weight of that node as follows: If $s = \mathsf{Root}(Q)$ then $W(s) = d$. Otherwise, set $d = d + \mathsf{g}(\mathsf{Pa}_Q(s)) - l(s)$ and $W(s) = d$. Note, that the weight at the root node of $\widetilde{Q}$ represents the value $\Delta(G, \Re)$.

Algorithm FastRNS$(\mathcal{G}, S, P)$**:** Typically, the RNS problem needs to be solved for several input gene trees. In this case we execute RCT for each gene tree separately. We call this algorithm FastRNS. Note that the tree $\widetilde{Q}$ obtained for each gene tree is identical except for the weights on the nodes. The output of FastRNS is a tree Φ with topology identical to tree Q and the weight of each node equal to the sum of the weights at the corresponding node in each $\widetilde{Q}$ tree.

By Observation 1 the NS problem can be solved through solutions to the RNS problem. Each edge in the given species tree S, defines a subtree that can be pruned. To solve the NS problem we keep calling the algorithm described above for each of these subtrees that can be pruned in S. This produces a node weighted tree Φ for each pruned subtree, which solves the NS problem.

Correctness: Now we show the correctness of our algorithm for the NS problem. To do this it is sufficient to show that the RNS problem is correctly solved by FastRNS.

Lemma 7. RCT(G, S, P) *computes* $W(s) = \Delta(G, S')$ *for all* $S' \in \mathsf{rSPR}(S, p)$.

Proof (Sketch). A node in Γ is called *feasible* if it is the parent of a supporting node. The following statements hold.

- Values $\mathsf{g}(s)$ and $l(s)$ are only computed for $s = \mathcal{M}_{\Gamma,Q}(\gamma)$, if $\gamma \in V(\Gamma)$ is feasible . This is because all other internal nodes in Γ will maintain their gene duplication status as P is regrafted into edges in Q (see Lemma 1).
- Consider a feasible node $\gamma \in V(\Gamma)$ whose gene duplication status changes when subtree P is regrafted into an edge $\{s, \mathsf{Pa}_Q(s)\}$ in Q. If P is then regrafted into any edge in the subtree Q_s, the gene duplication status of γ is preserved.
- If a feasible node $\gamma \in V(\Gamma)$ has only one child and $\mathcal{M}_{\Gamma,Q}(\gamma) = s$, then it will gain gene duplication status if and only if P is regrafted into the subtree Q_s.
- If a feasible node $\gamma \in V(\Gamma)$ has two children, α and β, then one of them, say α, must be a non-supporting node. Suppose α maps to the subtree rooted at a child t of $\mathcal{M}_{\Gamma,Q}(s)$ in Q, and β maps to the subtree rooted at the sibling of t in Q. Then, if P is regrafted into Q_t, node γ loses its gene duplication status.

From the above statements and Lemmas 4, 5 and 6 it follows that the values for g and l, and hence the node weights of $\widetilde{Q}$, are computed correctly for each node.

Lemma 8 follows from Lemma 7 and the definition of reconciliation cost.

Lemma 8. *The weight of a node* s *in tree* Φ *is* $\Delta(\mathcal{G}, S')$ *where* $S' = rSPR(S, p, s)$.

Observation 3. *Each tree in* $rSPR(S, p)$ *can be obtained by starting at* $\Re$ *and regrafting* P *into an edge in the subtree* Q.

The node weights on the tree Φ thus provide all the information needed to solve the RNS problem.

Theorem 1. *The* RNS *problem is correctly solved by* FastRNS*.*

Time Complexity: The major component of our algorithm to solve the NS problem is FastRNS that solves the RNS problem. Therefore we first analyze the complexity of FastRNS. Note, to simplify our analysis we assume that all $G \in \mathcal{G}$ have approximately the same size. Even if this does not hold true, our algorithm shows the same improvement in complexity over the current solution.

The input for FastRNS is a set $\mathcal{G}$ of gene trees, a species tree S, and the pruned subtree P of S. Let $n = |\,\mathrm{Le}(S)|$, and $k = |\mathcal{G}|$. FastRNS calls RCT k times and then constructs the tree Φ.

Complexity of RCT(G, S, P): Let $m = |\,\mathrm{Le}(S)| + |\,\mathrm{Le}(G)|$. The overall time complexity of RCT(G, S, P) is bounded by $O(m)$. A step-by-step analysis of the complexity follows:

1. *Initialization in* $O(|V(S)| + |V(G)|)$: The initial tree $\Re$ and the counters g and l for each node in Q can be setup in $O(|V(Q)|)$ time. Computing the mapping $\mathcal{M}_{G,\Re}$ takes $O(|V(S)| + |V(G)|)$ time, and the tree Γ can then be constructed in $O(|V(\Gamma)|)$ time. The mapping $\mathcal{M}_{\Gamma,Q}$ can be computed in $O(|V(\Gamma)|+|V(S)|)$ time. Hence, the time for the initialization costs is bound by $O(|V(S)| + |V(G)|)$, which is $O(m)$.
2. *Computing* g *and* l *in* $O(|V(G)|) + O(|V(Q)|)$: The values for g and l can be computed by traversing through the tree Γ once. To update the values for l we have to check whether $\mathcal{M}_{\Gamma,Q}(\gamma) \in V(Q_a)$ and $\mathcal{M}_{\Gamma,Q}(\sigma) \in V(Q_b)$ or $\mathcal{M}_{\Gamma,Q}(\sigma) \in V(Q_a)$ and $\mathcal{M}_{\Gamma,Q}(\gamma) \in V(Q_b)$. This check can be done in $O(1)$ time as follows: Initially, we perform an inorder traversal of the tree Q and label the nodes with increasing integer values in the order in which they are traversed. This preprocessing step takes $O(|V(Q)|)$ time. Based on the resulting order we can check whether a given node is in $V(Q_a)$ or $V(Q_b)$ in $O(1)$. $O(|V(\Gamma)|)$ updates for g and l are necessary, and each update can be performed in $O(1)$ time. Hence, computing g and l can be done in time $O(|V(G)|) + O(|V(Q)|)$, which is $O(m)$.
3. *Computing* $\widetilde{Q}$ *in* $O(|V(G)|) + O(|V(Q)|)$ Computing W for each node in $\widetilde{Q}$ from the g and l values involves first computing the value $\Delta(G, \Re)$, then traversing the tree $\widetilde{Q}$ in preorder and spending $O(1)$ time at each node. The time complexity of this step is $O(|V(G)|) + O(|V(Q)|)$, which is $O(m)$.

Complexity of FastRNS$(\mathcal{G}, S, P)$: Computing the final tree $\widetilde{Q}$ involves traversing each of the $\widetilde{Q}$ trees produced in preorder. This step takes $O(n)$ time per tree and hence $O(kn)$ time overall. Thus, the time complexity of FastRNS is bounded by $O(km + kn)$, which is $O(km)$.

Complexity of the NS problem: The time complexity of our algorithm for the NS problem is thus $O(n) \times O(km) \equiv O(kmn)$ (based on Observation 1). The brute force algorithm to solve the NS problem requires $O(kmn^2)$ time. Our algorithm for the NS problem improves on this by a factor of n. Also observe that this speed up does not come at the expense of higher space complexity.

6 Experimental Results

In order to study the performance of our algorithm we implemented it as part of a standard local search heuristic for the GENE-DUPLICATION problem. This program is called FASTGENEDUP. We first analyzed the performance and scalability of FASTGENEDUP using simulated input data and then focused on an analysis of large empirical data sets.

Table 1. GeneTree vs. FASTGENEDUP

Taxa size	GeneTree	FASTGENEDUP	Taxa size	GeneTree	FASTGENEDUP
50	9m:23s	1s	400	–	9m:19s
100	3h:25m	6s	1000	–	3h:20m
200	108h:33m	58s	2000	–	38h:25m

Performance and Scalability: We first compared the run time performance of FASTGENEDUP against the program GeneTree [3]. GeneTree currently is the only publicly available program that can build species supertrees based on the same local search heuristic. We measured the run time of each program to compute its final species supertree for the same set of input gene trees and the same randomly generated starting species tree. The input gene trees for each run consisted of a set of 20 randomly generated gene trees, all with the same set of taxa. We conducted 6 such runs, each with a different number of taxa (50, 100, 200, 400, 1000, and 2000) in the input trees. All analyses were performed on a 3 Ghz Intel Pentium 4 CPU based PC with Windows XP operating system. FASTGENEDUP shows a vast improvement in run time and scalability compared to GeneTree(Table 1). We could not run GeneTree on input trees with more than 200 taxa. Also, the memory consumption of FASTGENEDUP was less than the memory consumption of GeneTree. Note that even though both FASTGENEDUP and GeneTree implement the same local search heuristic, they may produce different supertrees, which may also have different reconciliation costs. This happens because during a local search step, more than one neighboring node may have the smallest reconciliation cost. In this case the node to follow is chosen arbitrarily, and this may cause the programs to follow different paths in the search space. In practice we noticed little or no difference in the final reconciliation costs, though FASTGENEDUP inferred supertrees with smaller reconciliation cost more often than GeneTree.

Empirical Example: The abundance of protein sequence data from many taxa makes it possible to perform large-scale analyses of the gene-duplication problem that require fast heuristics. We demonstrated the feasibility of such phylogenomic analyses using FASTGENEDUP on plant gene trees. The gene trees were derived from the set of all plant (Viridiplantae) sequences in GenBank (`http://www.ncbi.nlm.nih.gov`) downloaded on April 13, 2006. In total, this included 390, 230 amino acid sequences. The amino acid sequences were clustered into sets of homologs, representing gene families, using the NCBI BLASTCLUST program [25], which performs single linkage clustering of the sequences based on pairwise BLAST scores. We used a 60% identity cutoff value for the single-linkage clustering and the BLASTCLUST default alignment length. We then identified a set of clusters containing at least 4 sequences from at least 3 taxa and containing only sequences from taxa that are found in 10 or more such clusters. We found $3,978$ clusters containing sequences from 624 taxa (or technically 624 GenBank taxon ids, most of which represent distinct taxa) that met this criterion. From this set of clusters, we made three data sets that were used to produce the input trees for gene duplication analysis. The first set, the small data set, consisted of the 94 clusters (or gene families) that each had sequences from at least 40 different taxa. This set contained a total of $18,402$ protein sequences. The second set, the medium data set, consisted of the 599 clusters that each had sequences from at least 10 different taxa and contained a total of $48,156$ sequences. Finally, the large data set consisted of all $3,978$ clusters and contained a total of $100,914$ sequences, over 25% of the available plant protein sequences. To our knowledge, the large data set contains by far the most sequences ever incorporated into a single phylogenetic analysis of plants.

The sequences from each of the chosen clusters were aligned using the default options in ClustalW [26]. To obtain the gene trees from our data set, we built neighbor-joining trees [27] using PAUP* [28]. Since the gene-duplication problem requires binary, rooted gene trees, zero length branches were randomly resolved, and the trees were rooted with midpoint rooting. We tested the performance of FastGeneDup using the local search heuristic starting from a random species tree. The analyses of the small and medium data sets were performed on a Macintosh power PC laptop computer with a 1.5 GHz G4 processor and Mac OS X 10.4 operating system.

The small data set took 3 h. 15 m. 12 s. and found a species tree with a score of $13,393$ gene duplications. The medium data set took 24 h. 55 m. 41 s. and found a species tree with a score of $36,080$ gene duplications. The analysis of the large data set was performed on a 3 GHz Intel Pentium 4 based PC with Windows XP. It took 62 h. 35 m. 29 s. and found a species tree with $75,621$ gene duplications. This purpose of this experiment was to demonstrate that large genomic data sets could be incorporated into phylogenetic analyses using FastGeneDup. Like other attempts to build large plant trees from genome-scale data sets [29], the resulting species trees contain some anomalous relationships as well as some expected relationships. The presence of anomalous relationships is not surprising since the supertree analyses consisted only of a single run of the simple heuristic starting from a random tree. Also, the input trees were built using a simple neighbor-joining, and their quality can be improved with more thorough phylogenetic methods. Finally, mid-point rooting assumes that the sequences are evolving according to a molecular clock, which is a questionable assumption for many gene families.

7 Outlook and Conclusion

Despite the inherent complexity of the gene-duplication problem, it has been an effective approach for incorporating data from gene families into a phylogenetic inference [4,5,6,7]. Yet, existing local search heuristics for the problem are slow and thus cannot utilize the vast quantities of newly available genomic sequence data. We introduced an algorithm that speeds up the stepwise search procedure of local search heuristics for the gene-duplication problem. Our algorithm eliminates redundant calculations in computing the reconciliation cost for all trees resulting from pruning a given subtree and regrafting it to all possible positions. We implemented our algorithm as part of standard local search heuristics, and the resulting program, FastGeneDup, greatly improves upon the performance of GeneTree, a previous implementation to solve the gene-duplication problem. Furthermore, FastGeneDup made it possible to compute a supertree with 624 leaves from 3,978 input gene trees, representing over 25% of all available plant protein sequences, in less than three days on a desktop computer. This speed up also allows searching a much larger portion of the solution space within the same time, and hence can be used to obtain better solutions.

References

1. Goodman, M., Czelusniak, J., Moore, G.W., Romero-Herrera, A.E., Matsuda, G.: Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. Systematic Zoology **28** (1979) 132–163
2. Ma, B., Li, M., Zhang, L.: On reconcstructing species trees from gene trees in term of duplications and losses. In: RECOMB. (1998) 182–191
3. Page, R.D.M.: GeneTree: comparing gene and species phylogenies using reconciled trees. Bioinformatics **14**(9) (1998) 819–820
4. Slowinski, J.B., Knight, A., Rooney, A.P.: Inferring species trees from gene trees: A phylogenetic analysis of the elapidae (serpentes) based on the amino acid sequences of venom proteins. Molecular Phylogenetics and Evolution **8**(3) (1997) 349–362
5. Page, R.D.M.: Extracting species trees from complex gene trees: reconciled trees and vertebrate phylogeny. Mol. Phylogenetics and Evolution **14** (2000) 89–106
6. Cotton, J., Page, R.D.M.: Vertebrate phylogenomics: reconciled trees and gene duplications. In: Pacific Symposium on Biocomputing. (2002) 536–547
7. Cotton, J., Page, R. In: Tangled tales from multiple markers: reconciling conflict between phylogenies to build molecular supertrees. Springer-Verlag (2004) 107–125
8. Sanderson, M.J., McMahon, M.M.: Inferring angiosperm phylogeny from EST data with widespread gene duplication. BMC Evolutionary Biology (In press)
9. Page, R.D.M.: Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. Systematic Biology **43**(1) (1994) 58–77
10. Guigó, R., Muchnik, I., Smith, T.F.: Reconstruction of ancient molecular phylogeny. Molecular Phylogenetics and Evolution **6**(2) (1996) 189–213
11. Mirkin, B., Muchnik, I., Smith, T.F.: A biology consistent model for comparing molecular phylogenies. Journal of Computational Biology **2**(4) (1995) 493–507
12. Eulenstein, O.: Predictions of gene-duplications and their phylogenetic development. PhD thesis, University of Bonn, Germany (1998) GMD Research Series No. 20 / 1998, ISSN: 1435-2699.

13. Zhang, L.: On a Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. Journal of Computational Biology **4**(2) (1997) 177–187
14. Chen, K., Durand, D., Farach-Colton, M.: Notung: a program for dating gene duplications and optimizing gene family trees. Journal of Computational Biology **7**(3/4) (2000) 429–447
15. Bonizzoni, P., Vedova, G.D., Dondi, R.: Reconciling gene trees to a species tree. In: Italian Conference on Algorithms and Complexity, Rome, Italy (2003)
16. Górecki, P., Tiuryn, J.: On the structure of reconciliations. In: Recomb Comparative Genomics Workshop 2004. Volume 3388. (2004)
17. Bender, M.A., Farach-Colton, M.: The LCA problem revisited. In: Latin American Theoretical INformatics. (2000) 88–94
18. Harel, D., Tarjan, R.E.: Fast algorithms for finding nearest common ancestors. SIAM Journal on Computing **13**(2) (1984) 338–355
19. Fellows, M., Hallett, M., Korostensky, C., Stege., U.: Analogs & duals of the mast problem for sequences & trees. In: European Symposium on Algorithms (ESA), LNCS 1461. (1998) 103–114
20. Stege, U.: Gene trees and species trees: The gene-duplication problem is fixed-parameter tractable. In: Proceedings of the 6th International Workshop on Algorithms and Data Structures, LNCS 1663, Vancouver, Canada (1999)
21. Hallett, M.T., Lagergren, J.: New algorithms for the duplication-loss model. In: RECOMB. (2000) 138–146
22. Page, R.D.M.: Genetree. (`http://taxonomy.zoology.gla.ac.uk/rod/genetree/-genetree.html`)
23. Allen, B.L., Steel, M.: Subtree transfer operations and their induced metrics on evolutionary trees. Annals of Combintorics **5** (2001) 1–13
24. Bordewich, M., Semple, C.: On the computational complexity of the rooted subtree prune and regraft distance. Annals of Combintorics **8** (2004) 409–423
25. Dondoshansky, I.: Blastclust version 6.1 (2002)
26. Thompson, J., Higgins, D., Gibson, T.: ClustalW: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific penalties and weight matrix choice. Nucleic Acids Research **22** (1994) 4673–4680
27. Saitou, N., Nei, N.: The neighbour-joining method: a new method for reconstructing phylogenetic trees. Journal of Mol. Biology and Evolution **4** (1987) 406–425
28. Swofford, D.L.: PAUP*: Phylogenetic analysis using parsimony (*and other methods), version 4.0b10 (2002)
29. Driskell, A., An, C., Burleigh, J., McMahon, M., O'Meara, B., Sanderson, M.: Prospects for building the tree of life from large sequence databases. Science **306** (2004) 1172–1174

Support Vector Training of Protein Alignment Models

Chun-Nam John Yu[1], Thorsten Joachims[1], Ron Elber[1], and Jaroslaw Pillardy[2]

[1] Dept. of Computer Science, Cornell University, Ithaca NY 14853, USA
{cnyu,tj,ron}@cs.cornell.edu
[2] Cornell Theory Center, Cornell University, Ithaca NY 14853, USA
jarekp@tc.cornell.edu

Abstract. Sequence to structure alignment is an important step in homology modeling of protein structures. Incorporation of features like secondary structure, solvent accessibility, or evolutionary information improve sequence to structure alignment accuracy, but conventional generative estimation techniques for alignment models impose independence assumptions that make these features difficult to include in a principled way. In this paper, we overcome this problem using a Support Vector Machine (SVM) method that provides a well-founded way of estimating complex alignment models with hundred-thousands of parameters. Furthermore, we show that the method can be trained using a variety of loss functions. In a rigorous empirical evaluation, the SVM algorithm outperforms the generative alignment method SSALN, a highly accurate generative alignment model that incorporates structural information. The alignment model learned by the SVM aligns 47% of the residues correctly and aligns over 70% of the residues within a shift of 4 positions.

Keywords: Machine learning, Pairwise sequence alignment, Protein structure prediction.

1 Introduction

Sequence to structure alignment is a crucial step in building accurate three-dimensional protein models in homology modeling. Most alignment methods are based on dynamic programming on a linear cost model. In the simplest cases, such as alignment with the BLOSUM matrices, the linear model specifies costs for substituting one amino acid with another and for inserting a gap. The choice of these costs greatly determines the quality of alignments. While it is well understood how to estimate the substitution costs for such simple models, sequence to structure alignment requires more complex cost models to take advantage of the structural information available. For these complex cost models, conventional generative estimation techniques like Hidden Markov Models (HMM) are difficult to use due to the independence assumptions they make, for example, in assuming that the amino acid sequence and the secondary structure labels of a protein are generated by independent processes.

This paper explores a Support Vector Machine (SVM) method for learning an application specific cost model from training data for sequence to structure alignment. The advantages of this SVM method over conventional generative techniques are threefold. First, unlike conventional generative estimation techniques, the SVM method does not require independence assumptions among features and therefore provides a well-founded way to learn cost models where each aligned position is described not only by its amino acid identity, but by a potentially high-dimensional feature vector. This feature vector may describe additional properties of the aligned position (e.g. predicted secondary structure) as well as properties of surrounding aligned positions (e.g. the previous amino acid is hydrophobic). This provides great flexibility in building expressive models. Second, the SVM method inherits the benefits of conventional classification SVMs, in particular its robustness to overfitting for high-dimensional and sparse data. Third, the SVM method allows optimizing for different loss functions. This allows accounting for uncertainty in the training data, as well as specifying which types of alignment errors are more costly than others.

The work reported in the following shows that the SVM algorithm can be used to learn highly accurate alignment models for sequence to structure alignments. It also provides the first large-scale implementation and empirical validation of this SVM alignment algorithm, extending the basic algorithm first proposed in [1,2] to include loss functions. We show that this SVM algorithm can effectively learn alignment models with hundreds of thousands of features, outperforming the accuracy of state-of-the-art generative estimation techniques [3]. Finally, we show that loss functions can be incorporated into the SVM training problem while maintaining polynomial runtime guarantees. We find that the use of application-dependent loss functions during training, in particular by only counting alignment errors if the shift from the correct residue is more than 4, is effective in modeling the uncertainty in the training data. The training and alignment program of this work is available for download at `http://svmlight.joachims.org`.

2 Related Work

Conventional estimation techniques for alignment models (see e.g. [4,5,6,7,8]) take the view of a generative probabilistic model. A generative alignment model (e.g. HMM) aims to model the process that generates the data as the joint probability distribution $P(S,T,Y)$, where $\mathbf{s} = (s^1, ..., s^{|\mathbf{s}|})$ and $\mathbf{t} = (t^1, ..., t^{|\mathbf{t}|})$ are two sequences and $\mathbf{y}$ is the alignment. We denote the length of a sequence with $|.|$. If $P(S,T,Y)$ (or a good estimate thereof) is known,

$$\operatorname{argmax}_{\mathbf{y}} P(S = \mathbf{s}, T = \mathbf{t}, Y = \mathbf{y}) \tag{1}$$

predicts an alignment $\mathbf{y}$ from two sequences $\mathbf{s}$ and $\mathbf{t}$. To make estimation of $P(S,T,Y)$ tractable, it is decomposed by making independence assumptions on the process that generates $\mathbf{s}$ and $\mathbf{t}$. While this leads to efficient and simple estimation problems, the independence assumptions restrict the interactions within the sequences $\mathbf{s}$ and $\mathbf{t}$ that we could model.

Machine learning research over the last decade has provided substantial evidence that discriminative learning (e.g. SVMs, MaxEnt classifiers) typically produces more accurate rules than generative learning (e.g. naïve Bayes classifiers, HMMs) (see e.g. [9,10,11]). This can be explained as follows. Since $P(Y|S,T)$ is already sufficient for making an optimal prediction

$$\mathrm{argmax}_{\mathbf{y}} P(Y=\mathbf{y}|S=\mathbf{s},T=\mathbf{t}), \tag{2}$$

modeling the joint distribution of the input sequences S and T is not necessary, and generative methods might be wasting effort in trying to do so. Discriminative learning applied to the alignment problem would directly estimate $P(Y|S,T)$ or a related discriminant function, thus focusing on the relevant part of the estimation problem.

Only few approaches to discriminative training of alignment models exist to date. While not motivated from this learning theoretical perspective, work on *inverse alignment* is closely related, since our SVM method can be viewed as solving an inverse alignment problem. Inverse alignment is the task of finding a cost model under which a given alignment algorithm outputs a desired alignment $\mathbf{y}$ for sequences $\mathbf{s}$ and $\mathbf{t}$. This problem was first formulated in [12]. They discuss inverse alignment in the context of parametric sequence alignment and identify geometric properties of the space of cost model. In more detail, the work in [13] analyzes the space of models and shows that some aspects of its complexity grow only polynomially.

The first concrete algorithm for inverse sequence alignment was proposed in [14]. While they prove that their algorithm finds a consistent cost model in polynomial time, their algorithm is limited to particular cost models with at most 3 parameters.

The work presented in this paper follows the Structural SVM algorithm first proposed in [1,2] for sequence alignment and later generalized to a wide class of multivariate prediction problems [15,10]. In this paper we present the first large-scale empirical evaluation of this type of algorithm for alignment. We also extend the algorithm to optimize particular loss functions, and show how the resulting optimization problems can be solved in polynomial time.

Related to our SVM approach are Conditional Random Fields (CRFs), which have recently been proposed for sequence alignment as well [16,17]. While CRFs share with SVMs the benefits of discriminative training, they do not allow the use of application-dependent loss functions.

Independent of the work on Structural SVMs in the machine learning community, recently an algorithm for inverse alignment was proposed in [18]. Their formulation of the problem is similar to a Structural SVM and the algorithm resembles the cutting-plane method used for training Structural SVMs. However, their approach is based on a linear programming formulation instead of the quadratic programming formulation used in SVMs. Furthermore, their approach to handling infeasibilities in the resulting optimization problem is different and it is unclear how it relates to an intuitively meaningful loss function. While they give empirical results, they are on a scale of 10 training examples and 212 features.

In the following, we will explore models trained over thousands of examples and hundred-thousands of features.

3 Sequence Alignment

We begin by introducing the class of alignment models considered in this paper. Since we focus on the problem of sequence to structure alignment, we will describe the model in these terms. However, the methods can obviously be used for other applications as well. Let $(\mathbf{s}, \mathbf{t})$ be a pair of target and template sequence that we wish to align. For an alignment $\mathbf{y}$ of $(\mathbf{s}, \mathbf{t})$, we write $\mathbf{y}$ as a sequence of alignment operations $(y^1, y^2, ..., y^{|\mathbf{y}|})$. Each y^k is an alignment operation of the form (i, j), where i, j are positions of characters in $\mathbf{s}$ and $\mathbf{t}$ respectively, or a special gap symbol '$-$'.

We consider alignment algorithms (e.g. [19]) that optimize a linear scoring function $D_{\mathbf{w}}(\mathbf{y}, \mathbf{s}, \mathbf{t}) = \mathbf{w} \cdot \Psi(\mathbf{y}, \mathbf{s}, \mathbf{t})$, where Ψ is a function that maps an alignment $\mathbf{y}$ of $\mathbf{s}$ and $\mathbf{t}$ to a feature vector, and $\mathbf{w}$ is a given cost vector. Note that $\mathbf{w}$ contains the parameters of the alignment model that we will learn. We require that $\Psi(\mathbf{y}, \mathbf{s}, \mathbf{t})$ be linear in the individual alignment operations y^k within $\mathbf{y}$, written in terms of equations,

$$\Psi(\mathbf{y}, \mathbf{s}, \mathbf{t}) = \sum_{k=1}^{|\mathbf{y}|} \phi(y^k, \mathbf{s}, \mathbf{t}), \tag{3}$$

where ϕ is a function that maps each individual alignment operation onto a feature vector. Note that this feature vector can be any function that depends on the operation y^k and the *full* target and template sequences, not just the current positions that are aligned by y^k. In general, alignment algorithms compute

$$\operatorname*{argmax}_{y \in \mathcal{Y}} [\mathbf{w} \cdot \Psi(\mathbf{y}, \mathbf{s}, \mathbf{t})] = \operatorname*{argmax}_{y \in \mathcal{Y}} \left[\sum_{k=1}^{|\mathbf{y}|} \mathbf{w} \cdot \phi(y^k, \mathbf{s}, \mathbf{t}) \right] \tag{4}$$

to determine the alignment, where $\mathcal{Y}$ is the set of all possible (local or global, as desired) alignments between $\mathbf{s}$ and $\mathbf{t}$. This is typically computed using dynamic programming (e.g. [19]). Note that our setting includes the common scenarios of alignment with substitution matrices such as BLOSUM as a special case, where the function $\phi(y^k, \mathbf{s}, \mathbf{t})$ return a sparse vector with exactly one '1' that corresponds to the particular substitution or gap score in $\mathbf{w}$. However, we will consider richer feature mappings ϕ that go beyond amino-acid identity and that include structural information of the template sequence.

4 Discriminative Training of Alignment Models

In the above section, the vector $\mathbf{w}$ parameterizes the scoring function D and has crucial influence on the quality of alignments between $\mathbf{s}$ and $\mathbf{t}$. In the following, we aim to learn $\mathbf{w}$ from a set of training examples

$$\mathcal{S} = ((\mathbf{s}_1, \mathbf{t}_1, \mathbf{y}_1), (\mathbf{s}_2, \mathbf{t}_2, \mathbf{y}_2), ..., (\mathbf{s}_n, \mathbf{t}_n, \mathbf{y}_n)) \tag{5}$$

of sequence pairs $(\mathbf{s}_i, \mathbf{t}_i)$ for which the (approximately) correct alignment $\mathbf{y}_i$ is known. This training set is assumed to be generated independently and identically distributed (i.i.d.) according to some unknown distribution $P(S, T, Y)$. Thinking of a sequence alignment algorithm as a function,

$$h_{\mathbf{w}}(\mathbf{s}, \mathbf{t}) = \text{argmax}_{\mathbf{y} \in \mathcal{Y}} \left[\mathbf{w} \cdot \Psi(\mathbf{y}, \mathbf{s}, \mathbf{t})\right] \tag{6}$$

maps a given sequence pair $(\mathbf{s}, \mathbf{t})$ to an alignment $\mathbf{y}$. Our goal is to find a parameter vector $\mathbf{w}$ so that the predicted alignment $h_{\mathbf{w}}(\mathbf{s}, \mathbf{t})$ matches the correct alignment on new test data as well as possible. In particular, we want to find a $\mathbf{w}$ that minimizes the expected loss (i.e. risk)

$$R_P(h_{\mathbf{w}}) = \int \Delta(\mathbf{y}, h_{\mathbf{w}}(\mathbf{s}, \mathbf{t})) \; dP(S, T, Y), \tag{7}$$

where $\Delta(\mathbf{y}, \mathbf{y}')$ is a user defined (non-negative) loss function that quantifies how "bad" it is to predict $\mathbf{y}'$ when $\mathbf{y}$ is the correct alignment. For example, one may choose $\Delta(\mathbf{y}, \mathbf{y}')$ to be 1 minus the Q-score (Q-score is the fraction of match operations from $\mathbf{y}$ that are also contained in $\mathbf{y}'$).

Following the principle of (Structural) Empirical Risk Minimization [20], finding a $\mathbf{w}$ that predicts well on new data can be achieved by minimizing the empirical loss (i.e. the training error) $R_{\mathcal{S}}(h_{\mathbf{w}}) = \sum_{i=1}^{n} \Delta(\mathbf{y}_i, h_{\mathbf{w}}(\mathbf{s}_i, \mathbf{t}_i))$ on the training set $\mathcal{S}$. This leads to the computational problem of finding the $\mathbf{w}$ which minimizes $R_{\mathcal{S}}(h_{\mathbf{w}})$ as follows.

5 Structural SVMs for Sequence Alignment

In the framework of structural SVMs [10], we formulate the problem of finding the parameters $\mathbf{w}$ that minimizes the empirical loss $R_{\mathcal{S}}(h_{\mathbf{w}})$ of the sequence alignment algorithm as the following optimization problem:

$$\min_{\mathbf{w}, \xi} \; \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i \tag{8}$$
$$s.t. \;\; \forall i \in \{1, .., n\} \, \forall \hat{\mathbf{y}} \in \mathcal{Y}_i \backslash \{\mathbf{y}_i\} : \mathbf{w} \cdot (\Psi(\mathbf{y}_i, \mathbf{s}_i, \mathbf{t}_i) - \Psi(\hat{\mathbf{y}}, \mathbf{s}_i, \mathbf{t}_i)) \geq \Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - \xi_i$$

The objective is the conventional regularized risk used in SVMs. The constraints state that the score $\mathbf{w} \cdot \Psi(\mathbf{y}_i, \mathbf{s}_i, \mathbf{t}_i)$ of the correct alignment $\mathbf{y}_i$ must be greater than the score $\mathbf{w} \cdot \Psi(\hat{\mathbf{y}}, \mathbf{s}_i, \mathbf{t}_i)$ of all alternative alignments $\hat{\mathbf{y}}$. Note that $\mathcal{Y}_i$ (i.e. the set of all possible alignments for example i) depends on whether the alignment is local or global. However, the optimization problem is well-formed in either case.

Unlike the formulation in [2], our new formulation includes a loss function $\Delta(\mathbf{y}_i, \hat{\mathbf{y}})$ that scales the desired difference in score. Intuitively, the larger the loss of an incorrect alignment $\hat{\mathbf{y}}$, the further should the score be away from that of the correct alignment $\mathbf{y}_i$. This method for including a loss function is analogous to proposals for other structured prediction problems [21,10]. ξ_i is a slack variable shared among constraints from the same example, since in general the constraint system is not feasible. Following the proof in [10], it is easy to see that following result holds.

Theorem 1. *If* $(\mathbf{w}^*, \xi^*)$ *is the solution of the optimization problem in (8), the sum of the slack variables* ξ_i^* *is an upper bound on the training loss,* $R_{\mathcal{S}}(h_{\mathbf{w}}) \leq \sum_{i=1}^{n} \xi_i^*$.

This shows that our formulation minimizes training loss, while the SVM-style regularization with the norm of $\mathbf{w}$ in the objective provides protection against overfitting for high-dimensional $\mathbf{w}$. The parameter C allows the user to control the trade-off between training error and regularization.

5.1 Efficient Training Algorithm

While it is easy to see that the optimization problem in (8) is convex, it unfortunately has an exponential number of constraints. This results from the fact that there are exponentially many "wrong" alignments $\mathcal{Y}_i \backslash \{\mathbf{y}_i\}$ for each given pair of sequences $(\mathbf{s}_i, \mathbf{t}_i)$. Any attempt to solve this type of optimization problem using standard methods that require enumerating all constraints is obviously not tractable for sequences and training sets of interesting size.

Despite the exponential size, however, it has been shown that cutting-plane algorithms can be used to efficiently approximate the optimal solution of this type of optimization problem [10]. An adaptation of this algorithm to the problem of sequence alignment is given in Fig. 1. The algorithm iteratively constructs a subset of all constraints from (8) until this subset constrains the feasible region enough to ensure an ϵ-accurate solution. The desired precision ϵ is provided by the user. In particular, the algorithm starts with an empty set K of constraints. It then adds the most violated constraint among the exponentially many for each example. If no constraint exists that is violated by more than ϵ, the algorithm terminates. Otherwise, it solves the optimization problem over the current set K and repeats. Adapting the result from [10], it can be proved that only a polynomial number of constraints will be added before the algorithm converges.

Theorem 2. *For any* $\epsilon > 0$, $C > 0$, *and any training sample* $\mathcal{S} = ((\mathbf{s}_1, \mathbf{t}_1, \mathbf{y}_1), \ldots, (\mathbf{s}_n, \mathbf{t}_n, \mathbf{y}_n))$, *the algorithm in Fig. 1 converges after adding at most* $\max\left\{2n\bar{\Delta}/\epsilon, 8C\bar{\Delta}R^2/\epsilon^2\right\}$ *constraints to* K, *where* $R = \max_{i,\mathbf{y}} ||\Psi(\mathbf{y}, \mathbf{s}_i, \mathbf{t}_i)||$ *and* $\bar{\Delta}$ *is an upper bound on the loss function* $\Delta(\mathbf{y}_i, \hat{\mathbf{y}})$.

One crucial aspect of the algorithm, however, is the use of an oracle (often called a separation oracle in optimization theory) that can find the most violated constraint among the exponentially many in polynomial time. This is equivalent to the argmax problem in the algorithm. The following section shows that this argmax can be computed efficiently for a large class of loss functions.

```
Input: sequence pairs $(\mathbf{s}_1,\mathbf{t}_1),...,(\mathbf{s}_n,\mathbf{t}_n)$, correct alignments $\mathbf{y}_1,...,\mathbf{y}_n$, tolerated error
   $\epsilon \geq 0$.
$K = \emptyset$, $\mathbf{w} = 0$, $\xi = 0$
repeat
      – $K_{\mathrm{org}} = K$
      – for $i$ from 1 to $n$
            • $\hat{\mathbf{y}} = \mathrm{argmax}_{\hat{\mathbf{y}} \in \mathcal{Y}_i \setminus \{\mathbf{y}_i\}} [\Delta(\mathbf{y}_i, \hat{\mathbf{y}}) + \mathbf{w} \cdot (\Psi(\hat{\mathbf{y}}, \mathbf{s}, \mathbf{t}) - \Psi(\mathbf{y}_i, \mathbf{s}_i, \mathbf{t}_i))]$
            • if $\mathbf{w} \cdot (\Psi(\mathbf{y}_i, \mathbf{s}_i, \mathbf{t}_i) - \Psi(\hat{\mathbf{y}}, \mathbf{s}, \mathbf{t})) < \Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - \xi_i - \epsilon$
                  * $K = K \cup \{\mathbf{w} \cdot (\Psi(\mathbf{y}_i, \mathbf{s}_i, \mathbf{t}_i) - \Psi(\hat{\mathbf{y}}, \mathbf{s}, \mathbf{t})) \geq \Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - \xi_i - \epsilon\}$
                  * $(\mathbf{w}, \xi) = \mathrm{argmin}_{\mathbf{w},\xi} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i$ subject to $K$.
until $(K = K_{\mathrm{org}})$
Output: $\mathbf{w}$
```

Fig. 1. Cutting-plane algorithm for solving the SVM optimization problem

5.2 Loss Functions

We first introduce the Q-loss function, which we denote as $\Delta_{\mathrm{Q}}(\mathbf{y}_i, \hat{\mathbf{y}})$ for the loss between a correct alignment $\mathbf{y}_i$ and a predicted alignment $\hat{\mathbf{y}}$. The Q-loss measures the proportion of incorrect matches in a predicted alignment, which we want to minimize. Writing $\hat{\mathbf{y}}$ as a sequence of alignment operations $\hat{\mathbf{y}} = (\hat{y}^1, \hat{y}^2, ..., \hat{y}^{|\hat{\mathbf{y}}|})$, we can decompose the Q-loss $\Delta_{\mathrm{Q}}(\mathbf{y}_i, \hat{\mathbf{y}})$ into a sum of losses on individual alignment operations $\Delta_{\mathrm{Q}}(\mathbf{y}_i, \hat{\mathbf{y}}) = 1 - \sum_{k=1}^{|\hat{\mathbf{y}}|} \delta_{\mathrm{Q}}(\mathbf{y}_i, \hat{y}^k)$. The function $\delta_{\mathrm{Q}}(\mathbf{y}_i, \hat{y}^k)$ returns $1/M$ when $\hat{y}^k$ is a match contained in the correct alignment $\mathbf{y}_i$, and 0 otherwise. M is the number of matches in $\mathbf{y}_i$. With this decomposition, we can rewrite the computation of the most violated constraint in the cutting plane algorithm as:

$$\underset{\hat{\mathbf{y}} \in \mathcal{Y}_i \setminus \{\mathbf{y}_i\}}{\mathrm{argmax}} \left[\sum_{k=1}^{|\hat{\mathbf{y}}|} (\mathbf{w} \cdot \phi(\hat{y}^k, \mathbf{s}_i, \mathbf{t}_i)) - \delta_{\mathrm{Q}}(\mathbf{y}_i, \hat{y}^k) \right] + 1 - \mathbf{w} \cdot \Psi(\mathbf{y}_i, \mathbf{s}_i, \mathbf{t}_i). \quad (9)$$

Since the non-constant term in the argmax decomposes into a sum over individual alignment operations $\hat{y}^k$, we can apply dynamic programming similar to (4) with operation costs modified to $\mathbf{w} \cdot \phi(\hat{y}^k, \mathbf{s}_i, \mathbf{t}_i) - \delta_{\mathrm{Q}}(\mathbf{y}_i, \hat{y}^k)$ for each $\hat{y}^k$. Note that δ_{Q} can be computed efficiently using table lookup.

Another interesting loss function that we would like to consider is the Q_4-loss function, where we count a match as correct even if it is slightly shifted, in particular, shifted by not more than 4 positions. Suppose we have an alignment operation $\hat{y}^k = (i, j)$ in $\hat{\mathbf{y}}$. The shift of $\hat{y}^k$ is less than 4 if there is some match operation $y^l = (u, j)$ in $\mathbf{y}$ with $|i - u| \leq 4$. Similar to the Q-loss, we can decompose Q_4 linearly in its alignment operations as $\Delta_{\mathrm{Q}_4}(\mathbf{y}_i, \hat{\mathbf{y}}) = 1 - \sum_{k=1}^{|\hat{\mathbf{y}}|} \delta_{\mathrm{Q}_4}(\mathbf{y}_i, \hat{y}^k)$, where $\delta_{\mathrm{Q}_4}(\mathbf{y}_i, \hat{y}^k)$ is $1/M$ if $\hat{y}^k$ is a match operation contained in $\mathbf{y}$ with a shift of 4 or less, and 0 otherwise. The same dynamic programming algorithm applies to the computation of most violated constraint with Q_4-loss, since Q_4-loss decomposes in the same way as the Q-loss. Again, note that δ_{Q_4} can again be computed efficiently by table lookup of matched characters in the range $(i-4, j), ..., (i+4, j)$ within the correct alignment.

6 Experiments

The following experiments evaluate the SVM alignment algorithm on a sequence to structure alignment task. We evaluate whether the algorithm can effectively and efficiently learn complex alignment models with hundred-thousands of features, how optimizing to different loss functions might help, and how the algorithm compares to conventional methods.

In all our experiments, we train the algorithm on a training set, select any parameters and models based on a validation set, and then report performance on an independent test set. Beyond the choice of model $\phi(y^k, \mathbf{s}_i, \mathbf{t}_i)$ and loss function $\Delta(\mathbf{y}, \mathbf{y}')$, our method has only a single parameter to tune, namely the regularization parameter C. We train alignment models with C ranging from 1 to 2^{15}, in powers of 2, all to precision $\epsilon = 0.01$. We then pick the best model based on the performance on the validation set, and report its performance on the test set.

The training and validation sets are the same as those used in [3]. The data set contains 1379 target sequences, and each target sequence $\mathbf{s}$ has one or more template structures $\mathbf{t}$ associated with it. Structural alignments between target and template are generated using the CE program [22], and one example $(\mathbf{s}, \mathbf{t}, \mathbf{y})$ is generated whenever the structural alignment $\mathbf{y}$ between the structure of $\mathbf{s}$ and $\mathbf{t}$ has a CE Z-score of at least 4.5. The data set is randomly split into two sets, namely a training set with examples from 690 targets and a validation set with examples from 689 targets. The resulting training set contains 5119 examples (i.e. pairwise alignments) while the validation set contains 5169 examples.

The test set is based on a database of protein structures that is used by the modeling program LOOPP (`http://cbsuapps.tc.cornell.edu/loopp.aspx`). We select 4185 structures from the new PDB structures released between June 2005 and June 2006 via clustering. These structures serve as target sequences in our test set and none of them appear in the training or validation sets since they were developed earlier. Each of these 4185 structures is aligned against all other structures using the structural alignment program TM-align [23]. Pairs that score 0.5 or better are considered homologous and are added to the test set. The selected pairs are then aligned by the structural alignment program CE. Only alignments that have CE Z-score higher than 4.5 are included in the final test case, providing a total of 29764 alignments to consider.

As described in the following, we use structural annotations as features in our alignment models. The structural annotations of all the target sequences, i.e., the secondary structure and the relative exposed surface area, are predicted by the SABLE program [24]. There are 3 types of secondary structure predicted and the relative exposed surface areas are binned into 4 types. The structural annotations used in the template structures are computed by the program DSSP

[25]. The secondary structures are binned into 5 types while the exposed surface areas are binned into 6 types.

6.1 Can the SVM Algorithm Learn Complex Models Effectively?

In our first set of experiments we evaluate models $\phi(y^k, \mathbf{s}, \mathbf{t})$ of increasing complexity and number of features. Our focus is on exploring how far we can push the complexity of the model and still be able to train them efficiently and effectively.

We use $R_{\mathbf{s}}^i, S_{\mathbf{s}}^i, A_{\mathbf{s}}^i$ to denote the residue, predicted secondary structure, and predicted exposed surface area at the ith position of the target sequence, and $R_{\mathbf{t}}^j, S_{\mathbf{t}}^j, A_{\mathbf{t}}^j$ to deonte the residue, actual secondary structure, and actural exposed surface area at the jth position of the template structure. We also use $\mathbf{R}, \mathbf{S}, \mathbf{A}$ to denote the set of possible values for residue, secondary structure, and exposed surface area.

Substitution Cost Models. For the substitution costs, we consider the following six models for $\phi(y^k, \mathbf{s}, \mathbf{t})$. Since the length of alignments of different examples varies greatly, we normalize each ϕ by dividing with $|\mathbf{s}| + |\mathbf{t}|$.

Simple: In this alignment model we only consider the subsitution cost of single features. Let $y^k = (i, j)$ be a match operation. We define $\phi(y^k, \mathbf{s}, \mathbf{t})$ to be

$$\begin{aligned}
&\phi_{Simple}(y^k, \mathbf{s}, \mathbf{t}) \\
&= \sum_{r_1, r_2 \in \mathbf{R}} \mathbb{I}[R_{\mathbf{s}}^i = r_1, R_{\mathbf{t}}^j = r_2] + \sum_{r_1 \in \mathbf{R}, s_2 \in \mathbf{S}} \mathbb{I}[R_{\mathbf{s}}^i = r_1, S_{\mathbf{t}}^j = s_2] + \sum_{r_1 \in \mathbf{R}, a_2 \in \mathbf{A}} \mathbb{I}[R_{\mathbf{s}}^i = r_1, A_{\mathbf{t}}^j = a_2] \\
&+ \sum_{s_1 \in \mathbf{S}, r_2 \in \mathbf{R}} \mathbb{I}[S_{\mathbf{s}}^i = s_1, R_{\mathbf{t}}^j = r_2] + \sum_{s_1, s_2 \in \mathbf{S}} \mathbb{I}[S_{\mathbf{s}}^i = s_1, S_{\mathbf{t}}^j = s_2] + \sum_{s_1 \in \mathbf{S}, a_2 \in \mathbf{A}} \mathbb{I}[S_{\mathbf{s}}^i = s_1, A_{\mathbf{t}}^j = a_2] \\
&+ \sum_{a_1 \in \mathbf{A}, r_2 \in \mathbf{R}} \mathbb{I}[A_{\mathbf{s}}^i = a_1, R_{\mathbf{t}}^j = r_2] + \sum_{a_1 \in \mathbf{A}, s_2 \in \mathbf{S}} \mathbb{I}[A_{\mathbf{s}}^i = a_1, S_{\mathbf{t}}^j = s_2] + \sum_{a_1, a_2 \in \mathbf{A}} \mathbb{I}[A_{\mathbf{s}}^i = a_1, A_{\mathbf{t}}^j = a_2],
\end{aligned}$$

where $\mathbb{I}[\rho]$ is a function that returns a vector with '1' in the position designated to ρ if the boolean expression ρ is true, and returns '0' otherwise and in all other positions. For example, $\mathbb{I}[R_{\mathbf{s}}^3 = \text{'A'}, S_{\mathbf{t}}^7 = \text{'}\alpha\text{'}]$ returns '1' in the particular dimension corresponding to $\mathbb{I}[R_{\mathbf{s}}^i = \text{'A'}, S_{\mathbf{t}}^j = \text{'}\alpha\text{'}]$, if $y^k = (3, 7)$ aligns the residue alanine 'A' in $\mathbf{s}$ with an alpha helix 'α' in $\mathbf{t}$. Otherwise, it returns '0' in this dimension. For all other dimensions it always returns '0'. Note that each such dimension corresponds to a particular position in cost vector $\mathbf{w}$. Note also that each feature vector $\phi_{Simple}(y^k, \mathbf{s}, \mathbf{t})$ has exactly 9 '1's corresponding to the 9 terms in the sum, and is zero elsewhere.

Anova2: In this more complex feature vector we take the interactions between pairs of structural annotations at the same position in the sequence into account. We define $\phi(y^k, \mathbf{s}, \mathbf{t})$ to be

$$\begin{aligned}\phi_{Anova2}(y^k, \mathbf{s}, \mathbf{t}) = &\sum_{r_1,r_2\in\mathbf{R},s_1,s_2\in\mathbf{S}} \mathbb{I}[R^i_\mathbf{s}=r_1, S^i_\mathbf{s}=s_1, R^j_\mathbf{t}=r_2, S^j_\mathbf{t}=s_2]\\
&+\sum_{\substack{r_1\in\mathbf{R},a_2\in\mathbf{A},\\ s_1,s_2\in\mathbf{S}}} \mathbb{I}[R^i_\mathbf{s}=r_1, S^i_\mathbf{s}=s_1, S^j_\mathbf{t}=s_2, A^j_\mathbf{t}=a_2] + \sum_{\substack{r_1,r_2\in\mathbf{R},s_1\in\mathbf{S},\\ a_2\in\mathbf{A}}} \mathbb{I}[R^i_\mathbf{s}=r_1, S^i_\mathbf{s}=s_1, A^j_\mathbf{t}=a_2, R^j_\mathbf{t}=r_2]\\
&+\sum_{\substack{r_2\in\mathbf{R},a_1\in\mathbf{A},\\ s_1,s_2\in\mathbf{S}}} \mathbb{I}[S^i_\mathbf{s}=s_1, A^i_\mathbf{s}=a_1, R^j_\mathbf{t}=r_2, S^j_\mathbf{t}=s_2] + \sum_{\substack{s_1,s_2\in\mathbf{S},\\ a_1,a_2\in\mathbf{A}}} \mathbb{I}[S^i_\mathbf{s}=s_1, A^i_\mathbf{s}=a_1, S^j_\mathbf{t}=s_2, A^j_\mathbf{t}=a_2]\\
&+\sum_{\substack{r_2\in\mathbf{R},s_1\in\mathbf{S},\\ a_1,a_2\in\mathbf{A}}} \mathbb{I}[S^i_\mathbf{s}=s_1, A^i_\mathbf{s}=a_1, A^j_\mathbf{t}=a_2, R^j_\mathbf{t}=r_2] + \sum_{\substack{r_1,r_2\in\mathbf{R},s_2\in\mathbf{S},\\ a_2\in\mathbf{A}}} \mathbb{I}[A^i_\mathbf{s}=a_1, R^i_\mathbf{s}=r_1, R^j_\mathbf{t}=r_2, S^j_\mathbf{t}=s_2]\\
&+\sum_{\substack{r_1\in\mathbf{R},s_2\in\mathbf{S},\\ a_1,a_2\in\mathbf{A}}} \mathbb{I}[A^i_\mathbf{s}=a_1, R^i_\mathbf{s}=r_1, S^j_\mathbf{t}=s_2, A^j_\mathbf{t}=a_2] + \sum_{\substack{r_1,r_2\in\mathbf{R},\\ a_1,a_2\in\mathbf{A}}} \mathbb{I}[A^i_\mathbf{s}=a_1, R^i_\mathbf{s}=r_1, A^j_\mathbf{t}=a_2, R^j_\mathbf{t}=r_2].\end{aligned}$$

For example, the term $\mathbb{I}[R^i_\mathbf{s} = r_1, S^i_\mathbf{s} = s_1, R^j_\mathbf{t} = r_2, S^j_\mathbf{t} = s_2]$ returns '1' in the appropriate position, if $y^k = (i, j)$ aligns residue of type r_1 in secondary structure s_1 in the target with residue of type r_2 in secondary structure s_2 in the template. These features capture pairwise interaction of structural annotations within the same sequence.

Tensor: In this even more complex alignment model we consider the interaction of all three structural annotations. Note that there is only one non-zero feature in this feature vector.

$$\phi_{Tensor}(y^k, \mathbf{s}, \mathbf{t}) = \sum_{r_1,r_2\in\mathbf{R},s_1,s_2\in\mathbf{S},a_1,a_2\in\mathbf{A}} \mathbb{I}[R^i_\mathbf{s} = r_1, S^i_\mathbf{s} = s_1, A^i_\mathbf{s} = a_1, R^j_\mathbf{t} = r_2, S^j_\mathbf{t} = s_2, A^j_\mathbf{t} = a_2]$$

Simple+Anova2: This alignment model is the union of the features in the *Simple* and the *Anova2* alignment models, i.e. $\phi_{Simple}(y^k, \mathbf{s}, \mathbf{t}) + \phi_{Anova2}(y^k, \mathbf{s}, \mathbf{t})$.

Simple+Anova2+Tensor: This alignment model is the union of all features in the first three alignment models, i.e. $\phi_{Simple}(y^k, \mathbf{s}, \mathbf{t}) + \phi_{Anova2}(y^k, \mathbf{s}, \mathbf{t}) + \phi_{Tensor}(y^k, \mathbf{s}, \mathbf{t})$.

Window: On top of the *Simple+Anova2+Tensor* feature vector, we add several terms involving the substitution score of a sliding window of features centered around positions i and j.

$$\begin{aligned}\phi_{Window}(y^k, \mathbf{s}, \mathbf{t}) = &\phi_{Simple}(y^k, \mathbf{s}, \mathbf{t}) + \phi_{Anova2}(y^k, \mathbf{s}, \mathbf{t}) + \phi_{Tensor}(y^k, \mathbf{s}, \mathbf{t})\\
&+\sum_{r_1,r_2,r_3\in\mathbf{R},r_4,r_5,r_6\in\mathbf{R}} \mathbb{I}[R^{i-1}_\mathbf{s}=r_1, R^i_\mathbf{s}=r_2, R^{i+1}_\mathbf{s}=r_3, R^{j-1}_\mathbf{t}=r_4, R^j_\mathbf{t}=r_5, R^{j+1}_\mathbf{t}=r_6]\\
&+\sum_{s_1,\ldots,s_5\in\mathbf{S},s_6,\ldots,s_{10}\in\mathbf{S}} \mathbb{I}[S^{i-2}_\mathbf{s}=s_1, \ldots, S^{i+2}_\mathbf{s}=s_5, S^{j-2}_\mathbf{t}=s_6, \ldots, S^{j+2}_\mathbf{t}=s_{10}]\\
&+\sum_{a_1,\ldots,a_7\in\mathbf{A},a_8,\ldots,a_{14}\in\mathbf{A}} \mathbb{I}[A^{i-3}_\mathbf{s}=a_1, \ldots, A^{i+3}_\mathbf{s}=a_7, A^{j-3}_\mathbf{t}=a_8, \ldots, A^{j+3}_\mathbf{t}=a_{14}]\end{aligned}$$

The first sliding window term counts the occurence of substituting a triplet of residues (r_1, r_2, r_3) in the target with another triplet (r_4, r_5, r_6) in the template. The other two terms counts the occurence of substitution of two windows of secondary structures of length 5, and the occurence of substitution of two windows of surface area type of length 7 respectively. To reduce dimensionality of these features, we bin the residues into 7 groups ({A,G,P,S,T}, {C}, {D,E,N,Q}, {F,W,Y}, {H,K,R}, {I,L,M,V}, {X}, where X stands for missing value and ends of sequences), and the surface area into 2 values, exposed or buried.

Gap Cost Model. All alignment models above share the following gap model. Consider the cost of opening a gap between position i and $i+1$ in the target sequence $\mathbf{s}$ against position j in the template structure $\mathbf{t}$, as depicted by the following diagram

$$\begin{array}{lcccccc} \text{Target} & & \mathbf{s}^i & - & - & \cdots & - & \mathbf{s}^{i+1} \\ \text{Template} & \cdots & \mathbf{t}^j & \mathbf{t}^{j+1} & \cdots & \mathbf{t}^{j+k} & \cdots & \end{array}$$

We allow the cost of opening a gap to depend on the structural type at position j in the template structure. It also depends on the structural type of the target sequence immediately before the gap at position i as well as the structural type immediately after the gap at position $i+1$. Suppose y^k is a gap operation that opens a gap between position i and $i+1$ in the target against position j in the template sequence. The feature vector for this gap operation is:

$$\begin{aligned} \phi_{Gap}(y^k, \mathbf{s}, \mathbf{t}) = & \sum_{r_1 \in R} \mathbb{G}[R^j_{\mathbf{t}} = r_1] + \sum_{s_1 \in S, a_1 \in A} \mathbb{G}[S^j_{\mathbf{t}} = s_1, A^j_{\mathbf{t}} = a_1] \\ & + \sum_{s1, s2 \in S, a_1, a_2 \in A} \mathbb{G}[S^i_{\mathbf{s}} = s_1, A^i_{\mathbf{s}} = a_1, S^{i+1}_{\mathbf{s}} = s_2, A^{i+1}_{\mathbf{s}} = a_2] \end{aligned}$$

$\mathbb{G}$ is analogous to $\mathbb{I}$, but we use a different symbol to indicate that it maps to a different set of dimensions. The first two terms create features for the residue types and joint features of secondary structure with exposed surface area at $\mathbf{t}^j$. The term $\mathbb{G}[S^i_{\mathbf{s}} = s_1, A^i_{\mathbf{s}} = a_1, S^{i+1}_{\mathbf{s}} = s_2, A^{i+1}_{\mathbf{s}} = a_2]$ considers the structure before and after the gap. For example, $\mathbb{G}[S^i_{\mathbf{s}} = \text{`}\alpha\text{'}, A^i_{\mathbf{s}} = \text{`0'}, S^{i+1}_{\mathbf{s}} = \text{`}\alpha\text{'}, A^{i+1}_{\mathbf{s}} = \text{`1'}]$ maps to the dimension for the cost of opening a gap between a position in an alpha-helix of surface type 0 with a consecutive position in the alpha-helix with surface type 1.

The case of opening a gap in the template involves exactly the same costs, with the role of target and template reversed.

Results. Table 1 shows the Q-scores of the different alignment models trained with the SVM algorithm using Q-loss. As decribed above, we report the results for the value of C that optimizes performance on the validation set. The table also shows the number of features in each model. Note that the training and the validation set are composed of more difficult cases than the test set, which explains the generally higher Q-scores on the test set. All performance differences on the test set are statistically significant according to the paired Wilcoxon test,

Table 1. Q-score of the SVM algorithm for different alignment models

	# Features	Training	Validation	Test
Simple	1020	26.83	27.79	39.89
Anova2	49634	42.25	35.58	44.98
Tensor	203280	52.36	34.79	42.81
Simple+Anova2	50654	42.29	35.34	44.74
Simple+Anova2+Tensor	253934	47.80	35.79	44.39
Window	447016	51.26	38.09	46.30

Table 2. Comparing training for Q-score with training for Q_4-score by test set performance

Anova2	test Q	test Q_4
train Q	44.98	67.20
train Q_4	45.65	69.45

Window	test Q	test Q_4
train Q	46.30	68.33
train Q_4	47.65	70.71

except for the three closely related alignment models *Anova2*, *Simple+Anova2*, and *Simple+Anova2+Tensor*.

Table 1 shows that the *Simple* alignment model is too simple to fit the training data, indicated by the low Q-score on the training set. This alignment model perform considerably worse than the other alignment models. The more expressive *Anova2* model leads to substantial improvement in Q-score over *Simple* on both the valiation and test sets, showing that considering pairwise interaction between structural annotations is meaningful. The *Tensor* alignment model does worse than *Anova2*. There are signs of overfitting in the relatively high Q-score on the training set. However, the performance on the validation and test sets are respectable nonetheless. Adding the substitution costs in the alignment models together, as in *Simple+Anova2* and *Simple+Anova2+Tensor*, does not give us any gain in accuracy. Their performance on the validation and test sets are very close to *Anova2*. Only when we incorporate structural information in the local neighbourhood, as in the alignment model *Window*, do we see another jump in the Q-score on the test set. The Q-score of 46.30 in the *Window* alignment model is substantially better than the Q-score of 39.89 of the *Simple* alignment model that we started with. To provide a baseline, the Q-score of BLAST is 23.88 on the test set.

6.2 Is Training to Different Loss Functions Beneficial?

The SVM method allows the use of different loss functions during training. The Q-loss used in the previous subsection is rather stringent and does not necessarily summarize the quality of an alignment well. For example, if all the aligned positions are shifted by just 1, the Q-loss will jump from 0 to 1, which is roughly the same Q-loss as that of a completely random alignment. Furthermore, the Q-loss does not account for the approximate nature of the training alignments, since

Table 3. Comparing training for Q-score with training for Q_4-score by test set performance

	Q on test	Q_4 on test
SVM (Window, Q_4)	47.65	70.71
SSALN	47.06	67.30
BLAST	23.88	28.44
TM-align	69.99	85.32

there is typically no single exact alignment in sequence to structure alignment that is clearly correct.

Instead of Q-loss, we now consider the Q_4-loss function. Q_4-loss counts a residue as correctly aligned if the shift from its position in the reference alignment is no more than 4. The Q_4-loss function captures our intuition that small shifts in alignment could be tolerated, and such alignments should be differentiated from alignments that are completely wrong. We repeat our experiments on two alignment models from the last section, *Anova2* and *Window*, but this time we train them with Q_4 as the loss function. The results on the test set are shown in Table 2. For each table entry, we select C on the validation set with respect to the performance measure that is reported.

Table 2 shows that the models trained on Q_4 show better Q_4 performance on the test set. More surprisingly, the models trained on Q_4 also show (statistically significantly) better Q-score on the test set. This gives evidence that Q_4 can indeed effectively account for the inaccuracy of the training alignments, instead of trying to model the noise. However, in situations where the alignments have higher sequence similarity or we are more confident of the alignments, the use of Q-loss or reducing the allowable shift of of 4 in Q_4 to lower values could be beneficial. The flexibility of the SVM regarding the selection of loss function would cater either of these situations.

6.3 How Does the Accuracy of SVM Models Compare to Conventional Methods?

As selected by validation performance, the best alignment model is *Window* trained on Q_4. Table 3 shows the test set performance of various other methods in comparison. SSALN [3] is one of the best current alignment algorithm trained using generative methods, and it outperforms alignment and threading algorithms like CLUSTALW, GenTHREADER, and FUGUE on a variety of benchmarks. It incorporates structural information in its substitution matrices, and contains a hand-tuned gap model. SSALN was trained on exactly the same training set and same set of structural annotations as our SVM model, so a direct comparison is particularly meaningful. The SVM model substantially outperforms SSALN with respect to Q_4-score, and is slightly better than SSALN on Q-score. The performance of BLAST is included to provide a baseline. The performance of the structural alignment program TM-align [23] is reported here to show its agreement with the CE alignments, and demonstrates the rather high inherent noise in the data.

7 Discussions and Conclusions

This paper explore an SVM method for learning complex alignment models for sequence to structure alignment. We show that the algorithm can learn high-dimensional models that include many features beyond residue identity while effectively controlling overfitting. Unlike generative methods, it does not require independence assumptions between features. The SVM method provides great modeling flexibility to biologists, allowing the estimation of models that include all available information without having to worrying about statistical dependencies between features. Furthermore, we show that one can incorporate different loss functions during training, which provides the flexibility to specify the costs of different alignment errors. The empirical results show that the SVM algorithm outperforms one of the best current generative models, and is practical to train on large datasets.

Acknowledgments

We thank Jian Qiu for his detailed answers on our many questions on SSALN, Tamara Galor-Neah for her help with the LOOPP program, and Phil Zigoris for his work on an early prototype of the SVM software. This work is supported by NIH Grants IS10RR020889 and GM67823, and by the NSF Award IIS-0412894.

References

1. Joachims, T.: Learning to align sequences: A maximum-margin approach. http://www.joachims.org (August 2003)
2. Joachims, T., Galor, T., Elber, R.: Learning to align sequences: A maximum-margin approach. In et al., B.L., ed.: New Algorithms for Macromolecular Simulation. Volume 49 of LNCS. Springer (2005) 57–68
3. Qiu, J., Elber, R.: SSALN: an alignment algorithm using structure-dependent substitution matrices and gap penalties learned from structurally aligned protein pairs. Proteins **62** (2006) 881–91
4. Bucher, P., Hofmann, K.: A sequence similarity search algorithm based on a probabilistic interpretation of an alignment scoring system. In: International Conference on Intelligent Systems for Molecular Biology (ISMB). (1996)
5. Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: Biological Sequence Analysis. Cambridge University Press (1998)
6. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks. Proceedings of the National Academy of Sciences **89** (1992) 10915–10919
7. Dayhoff, M.O., Schwartz, R.M., Orcutt, B.C.: A model of evolutionary change in proteins. Atlas of Protein Sequence and Structure **5** (1978) 345–352
8. Ristad, S.E., Yianilos, P.N.: Learning string edit distance. IEEE Transactions on Pattern Recognition and Machine Intelligence **Vol. 20(5)** (1998) 522–532

9. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Proceedings of the European Conference on Machine Learning, Berlin, Springer (1998) 137 – 142
10. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research (JMLR) **6** (September 2005) 1453 – 1484
11. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning (ICML). (2001)
12. Gusfield, D., Stelling, P.: Parametric and inverse-parametric sequence alignment with XPARAL. Methods in Enzymology **266** (1996) 481–494
13. Pachter, L., Sturmfelds, B.: Parametric inference for biological sequence analysis. In: Proceedings of the National Academy of Sciences. Volume 101. (2004) 16138–16143
14. Sun, F., Fernandez-Baca, D., Yu, W.: Inverse parametric sequence alignment. In: International Computing and Combinatorics Conference (COCOON). (2002)
15. Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In: International Conference on Machine Learning (ICML). (2004)
16. Do, C.B., Gross, S.S., Batzoglou, S.: CONTRAlign: Discriminative training for protein sequence alignment. In: International Conference in Research on Computational Molecular Biology (RECOMB). (2006)
17. McCallum, A., Bellare, K., Pereira, F.: A conditional random field for discriminatively-trained finite-state string edit distance. In: Conference on Uncertainty in Artificial Intelligence. (2005)
18. Kececioglu, J.D., Kim, E.: Simple and fast inverse alignment. In Apostolico, A., Guerra, C., Istrail, S., Pevzner, P.A., Waterman, M.S., eds.: Annual International Conference on Research in Computational Molecular Biology (RECOMB). Volume 3909 of Lecture Notes in Computer Science., Springer (2006) 441–455
19. Smith, T., Waterman, M.: Identification of common molecular subsequences. Journal of Molecular Biology **147** (1981) 195–197
20. Vapnik, V.: Statistical Learning Theory. Wiley, Chichester, GB (1998)
21. Taskar, B., Guestrin, C., Koller, D.: Maximum-margin markov networks. In: Neural Information Processing Systems (NIPS). (2003)
22. Shindyalov, I.N., Bourne, P.E.: Protein structure alignment by incremental combinatorial extension(CE) of the optimal path. Protein Eng **11** (1998) 739–747
23. Zhang, Y., Skolnick, J.: TM-align: A protein structure alignment algorithm based on TM-score. Nucleic Acids Research **33** (2005) 2302–2309
24. Adamczak, R., Porollo, A., Meller, J.: Accurate prediction of solvent accessibility using neural networks-based regression. Proteins **56** (2004) 753–67
25. Kabsch, W., Sander, C.: Dictionary of protein secondary structure: pattern recognition of hydrogen bond and geometrical features. Biopolymers **22** (1983) 2577–2637

Tools for Simulating and Analyzing RNA Folding Kinetics*

Xinyu Tang, Shawna Thomas, Lydia Tapia, and Nancy M. Amato

Parasol Lab, Dept. of Comp. Sci., Texas A&M University, College Station, TX 77843

Abstract. It has recently been found that some RNA functions are determined by the actual *folding kinetics* and not just the RNA's nucleotide sequence or its native structure. We present new computational tools for simulating and analyzing RNA folding kinetic metrics such as population kinetics, folding rates, and the folding of particular subsequences. Our method first builds an approximate representation (called a map) of the RNA's folding energy landscape, and then uses specialized analysis techniques to extract folding kinetics from the map. We provide a new sampling strategy called Probabilistic Boltzmann Sampling (PBS) that enables us to approximate the folding landscape with much smaller maps, typically by several orders of magnitude. We also describe a new analysis technique, Map-based Monte Carlo (MMC) simulation, to stochastically extract folding pathways from the map. We demonstrate that our technique can be applied to large RNA (e.g., 200+ nucleotides), where representing the full landscape is infeasible, and that our tools provide results comparable to other simulation methods that work on complete energy landscapes. We present results showing that our approach computes the same relative functional rates as seen in experiments for the relative plasmid replication rates of ColE1 RNAII and its mutants, and for the relative gene expression rates of MS2 phage RNA and its mutants.

1 Introduction

Ribonucleic acid (RNA) performs diverse and important functions such as synthesizing proteins, catalyzing reactions, splicing introns, and regulating cellular activities [25,14,2]. It was once believed that an RNA's functions are primarily determined by its nucleotide sequence and native state. However, it has recently been found that some RNA functions are determined by the folding process itself. For example, RNA folding kinetics may regulate the plasmid copy number, e.g., accelerating the refolding speed of RNA II can increase the E. coli ColE1 plasmid copy number [9]. Similarly, the velocity of RNA folding can also regulate gene expression at the translational level. It has also been shown that the

* Supported in part by NSF Grants EIA-0103742, ACR-0081510, ACR-0113971, CCR-0113974, ACI-0326350, by the DOE, and by HP. Thomas supported in part by a Dept. of Education GAANN Fellowship, a NSF Graduate Research Fellowship, and a P.E.O. Scholarship. Tapia supported in part by a NIH Molecular Biophysics Training Grant (T32GM065088) and a Dept. of Education GAANN Fellowship.

mRNA folding kinetics regulate the expression of phage MS2 maturation protein [8,14,10]. The mRNA acts as a regulator only when a particular subsequence is open. Since it is closed in the native state, this can only happen before folding finishes. The longer the RNA stays in a open metastable state, the higher the gene expression rate. Thus, it is imperative to have a computational method that can study both the global properties of RNA folding and more detailed features related to kinetics-based functions.

Our work provides computational tools to approximate the folding energy landscape and extract global properties and detailed features of the folding process. The key advantage of our approach over other computational techniques is that it is fast and efficient while bridging the gap between high-level folding events and low-level folding details. Our method builds an approximate representation (called a map) of the RNA's folding energy landscape, and then uses specialized analysis techniques to extract folding kinetics from the map. We give a new sampling strategy called *Probabilistic Boltzmann Sampling (PBS)* that approximates the folding landscape with much smaller maps, enabling us to handle RNA with hundreds of nucleotides. We present a new analysis technique, *Map-based Monte Carlo (MMC)* simulation, to stochastically extract folding pathways from the map. These tools allow us to study population kinetics, folding rates, and the folding of particular subsequences we could not study before.

We validate our methods against other computational methods (Monte Carlo Simulation) and experimental data. We demonstrate that our maps efficiently capture major features of much larger energy landscapes by comparing kinetics metrics extracted from them with those computed using a complete energy landscape. We also show that our method scales well to large RNA with hundreds of nucleotides. Finally, we present two case studies. First, we compare simulated folding rates for ColE1 RNAII and its mutants against experimental rates and show that we compute the same relative folding order as seen in experiment. Second, we predict the gene expression rates of wild-type MS2 phage RNA and three other of its mutants and and again we show that we predict the same relative functional rates as seen in experiment.

2 Preliminaries

An RNA molecule is a sequence of nucleotide bases. There are four types of bases: adenine (A), cytosine (C), guanine (G), and uracil (U). The complementary Watson-Crick bases, C-G and A-U, form stable, hydrogen bonds (*base pairs*) when they form a contact. The wobble pair, G-U, constitutes another strong base pair. These are the three most commonly considered base pairings [29] and are what we consider in our model.

RNA Structure. *Tertiary structure* is a 3D spatial RNA conformation of a set of base pairs. *Secondary structure* is a planar representation of an RNA conformation. Although there are several slightly different accepted definitions [3,11], secondary structure is usually considered to be a planar subset of the base pair contacts present. Non-planar contacts, often called *pseudo knots*, are not allowed in secondary structure. We adopt the definition in [11] that eliminates other types of contacts that are not physically favored: (1) contacts must be separated by at least 3 other bases, (2) each base cannot be involved in more than 1 contact, and (3) contacts must be planar. Tertiary structure gives the most complete representation of RNA structure. However, secondary structure is commonly used [29,11] because in many cases it provides sufficient information to study many aspects of folding while dramatically reducing the size of the conformation space to explore. One justification for this simplification is that research has shown the RNA folding process is hierarchical, i.e., secondary structure forms before tertiary structure [25]. In this work, we focus on the first stage: secondary structure formation.

Energy Calculations. To model the RNA folding energy landscape, we must be able to calculate the energy of any conformation. We use a common energy function called the Turner or nearest neighbor rules [29]. This method involves determining the types of loops that exist in the molecule and looking up their free energy in a table of experimentally determined values. Intuitively, adjacent contacts typically form stable subunits called stacks or stems that have low energy. Much work has been done to improve the accuracy of these rules.

3 Related Work

Computational research on RNA folding falls into two main categories: structure prediction and folding kinetics. Structure prediction attempts to compute the native state given only the nucleotide sequence. Folding kinetics, on the other hand, is concerned with the folding process itself and not just the end result.

Structure Prediction. Structure prediction is commonly solved with dynamic programming. Nussinov introduced a dynamic programming solution to find the conformation with the maximum number of base pairs [18]. Zuker and Stiegler [29] formulated an algorithm to address the minimum energy problem. Today, Zuker's MFOLD algorithm is widely used for structure prediction. McCaskill's algorithm [15] uses dynamic programming to calculate the partition function, i.e., the the sum of Boltzmann factors over all possible secondary structures, while Chen [3] uses matrices to approximate the partition function over all possible conformations. Eddy and Dirks et. al. [20,6] include pseudo-knots in their structure prediction algorithms. Partly due to the inaccuracy of the energy model, the prediction of pseudo-knot structures is typically less accurate.

Folding Kinetics. Several approaches have been used to study RNA kinetics. For example, [7,10,28] used Monte Carlo algorithms to find folding pathways while Gultyaev and Shapiro et. al. [9,21] used genetic algorithms. Isambert [28] extended the Monte Carlo method to consider pseudo-knots.

Some methods involve computations on the folding landscape. Dill [3] used matrices to compute the partition function over all possible structures and approximate the complete folding landscape. Ding and Lawrence [5] extended

McCaskill's algorithm to generate statistical samplings of RNA structures based on the partition function. Wuchty [27] modified Zuker's algorithm to generate all secondary structures within some given energy range of the native structure. Flamm and Wolfinger [7,26] extended this algorithm to find local minima within some energy threshold of the native state and connect them via energy barriers. The resulting energy barrier tree represents the energy landscape. To calculate the energy barrier, they used a flooding algorithm that is exponential in the size of RNA. Thus, it is impractical for large RNA. Some statistical mechanical methods are also used to study the RNA folding kinetics. For example, the master equation is used to compute the population kinetics of the folding landscape. It uses a matrix of differential equations to represent the transition probabilities between conformations. Once solved, the dominate modes of the solution describe the general folding kinetics [19,12,3].

RNA Folding with PRMs. Our approach is based on the *probabilistic roadmap* (PRM) technique for motion planning [13]. Motion planning determines valid paths to move objects from one conformation to another. PRMs build graphs (roadmaps) that approximate the topology of the feasible planning space by first sampling valid conformations (nodes) and connecting them with feasible transitions (edges). Connections are only attempted to a conformation's nearest neighbors, as determined by some distance metric. In previous work, we used PRMs to approximate the folding energy landscape and studied protein folding [1,22,24] and RNA folding [23]. We obtained promising results that validated against experimental data and were even able to observe subtle folding differences between structurally similar proteins [24]. We were also able to validate the population kinetics of several small RNA against experiment [23].

4 Computational Methods

Our method first constructs a roadmap to approximate the energy landscape. Then we use our map-based tools to analyze the energy landscape. In our previous work, we presented two successful roadmap construction techniques: base-pair enumeration (BPE) and stack-pair enumeration (SPE). While the results were promising, they were limited to small RNA (less than 40 nucleotides). In this work, we develop a *Probabilistic Boltzmann Sampling (PBS)* method to build smaller (up to 10 orders of magnitude smaller than BPE) maps, and thus enables us to study much larger RNA. We provide several map-based analysis tools including a Map-based Master Equation (MME) and Map-based Monte Carlo (MMC) simulation to extract folding kinetics. MME, introduced in our previous work, can extract global properties such as folding rates and transition states, while MMC, introduced here, can extract microscopic features of the folding process, e.g., subsequence formation order.

4.1 Using Roadmaps to Describe Energy Landscapes

The goal of roadmap construction is to approximate the energy landscape and capture its important features. The quality of this approximation highly depends on the quality of the sampling and connection methods. We will describe each sampling and connection method in more detail below.

Roadmap Node Sampling. Our method is general and thus can use conformations generated by any technique. In our previous work, we used three methods for generating RNA conformations: complete base-pair enumeration (for small RNA), stack-pair enumeration, and maximal-contact sampling. While stack-pair enumeration approximated the energy landscape well, it is limited to small RNA where enumeration is feasible (e.g., 40 nucleotides or less). Here we provide a new sampling method for larger RNA.

Probabilistic Boltzmann Sampling (PBS). Wuchty [27] proposes a dynamic programming algorithm to enumerate suboptimal (low energy) conformations within a given energy threshold. However, as the size of the RNA or the threshold increases, the number of generated nodes increases exponentially. Thus, it is difficult for this method to generate high energy nodes. In our method, we use these suboptimal conformations as "seeds" and augment the sampling with additional random conformations. Then, we use a probabilistic filter to retain a subset of the conformations based on their Boltzmann distribution factors. For a given conformation i with free energy E_i, the probability P_i to keep it is:

$$P_i = \begin{cases} e^{\frac{-(E_i - E_0)}{kT}} & \text{if } (E_i - E_0) > 0 \\ 1 & \text{if } (E_i - E_0) \leq 0 \end{cases} \tag{1}$$

E_0 is a reference energy threshold that we can use to control the number of samples kept. In this way, we may generate more conformations probabilistically with the Boltzmann distribution which prefers low energy conformations but will allow some high energy conformations. Our results indicate that this sampling method captures the important features of the energy landscape well.

Roadmap Node Connection. Once we have a set of samples, we connect them using a so-called local planner to form an approximate map of the energy landscape. It is impractical (and generally not necessary) to attempt all possible connections. Instead, we attempt to connect a node with the k closest neighboring conformations according to some distance metric, where k is a user-specified constant.

To connect a given pair of conformations, we need to compute a transition path (i.e., intermediate conformations) between them and approximate the Boltzmann transition probability which is stored as an edge weight in the roadmap. Note that these two goals are not always the same. For example, when conformations are close to each other, one single (most energetic) transition path may dominate the transition probability. However, when conformations are far apart, there might be many possible transition paths where none dominate.

In our previous work, we presented a simple greedy algorithm that generates a single transition path and computes the transition probability from that path. It works well when conformations are close to each other. However, as the size of RNA increases and thus the feasible sampling density decreases, this method fails. Here we present methods designed to compute transition probabilities and to generate transition pathways that do not have these problems.

Computing the Transition Probability. When an edge (q_i, q_j) is added to the roadmap, it is assigned a weight W_{ij} that reflects the Boltzmann transition probability between its two end points q_i and q_j. First, we find the stable subunits (stems) that are different between q_i and q_j. We calculate the nucleation cost for each stem (which is the energy barrier to form each stem) and find the maximum cost. This maximum cost is an energy barrier E_b the folding process must go over to form all the stems. We use E_b to estimate the transition probability between q_i and q_j. This strategy is widely used in Monte Carlo simulations [10,28] and genetic algorithms for folding pathways [9,21].

We calculate the Boltzmann transition probability K_{ij} (or transition rate) of moving from q_i to q_j using Metropolis rules [4]:

$$K_{ij} = \begin{cases} e^{\frac{-\Delta E}{kT}} & \text{if } \Delta E > 0 \\ 1 & \text{if } \Delta E \leq 0 \end{cases} \tag{2}$$

where $\Delta E = max(E_b, E_j) - E_i$, k is the Boltzmann constant, and T is the temperature. Note that the same energy barrier E_b is also used to estimate the transition probability K_{ji}, so the calculation satisfies the detailed balance:

$$\frac{K_{ij}}{K_{ji}} = e^{\frac{-(E_j - E_i)}{kT}} \tag{3}$$

Thus, the edge weight W_{ij} is:

$$W_{ij} = -log(K_{ij)}) = \frac{-\Delta E}{kT}. \tag{4}$$

(Negative logs are used since $0 \leq K_{ij} \leq 1$.) By assigning the weights in this manner, we can easily extract the most energetically feasible path in our roadmap using simple graph search algorithms.

Generating Transition Pathways. First, we find the stems between the start and goal configurations and calculate their nucleation costs. Then we generate a transition pathway connecting the start and the goal configuration by probabilistically opening/closing the stems based on their nucleation cost.

4.2 Map-Based Analysis Tools

In this section, we describe several different map-based analysis tools to study folding kinetics including Map-based Master Equation (MME) and Map-based Monte Carlo (MMC) simulation. As implied by their names, MME and MMC are variants of the master equation and Monte Carlo simulations that work on our maps. Basically, the master equation calculates global properties of the folding process while Monte Carlo simulations provide details on individual folding pathways. However, they can both produce population kinetics, one directly and the other indirectly. Given an ensemble of Monte Carlo simulation pathways, we can can compute the population kinetics of a particular conformation by summing up its population in each pathway for every time step. This approach is less accurate and will take more time and space than using the master equation directly.

However, it does not have the same numerical limitations as the master equation and can handle much larger RNA. In our experimental results (Section 5.1), we empirically compare the population kinetics by the master equation, standard Monte Carlo simulation (implementation of Vienna Package), and MMC.

Map-based Monte Carlo Simulation. The folding process is stochastic rather than deterministic [12]. Transitioning from one conformation to another is probabilistically biased by the transition probabilities. The Monte Carlo method [17,12] simulates this random walk in the real (or complete) energy landscape. Kinfold is a well-known implementation of Monte Carlo simulation in the ViennaRNA Package [7]. These simulations can be computationally intensive since at each step they must calculate the local energy landscape to choose the next step.

In previous work, we simply extracted the most energetically feasible path in the roadmap to study the folding process. However, this does not mirror the stochastic folding process. Instead in this work, we apply Monte Carlo simulation directly to our roadmaps, which are approximations of the energy landscape where edge weights reflect Boltzmann transition probabilities. Similar to Monte Carlo simulation, our method starts from a random node in the roadmap and iteratively chooses a next node based on the transition probabilities. Because the edge weight W_{ij} encodes the transition probabilityK_{ij} between two endpoints i and j (see equation 4), we can calculate K_{ij} as $K_0 e^{-W_{ij}}$ where K_0 is a constant adjusted according to experimental results.

To generate the transitional conformations between two nodes, we use the method described in Section 4.1. Results presented here are generated using a fast variant of the standard Monte Carlo method [17].

Population Kinetics and Map-based Master Equation. Population kinetics give the time evolution of the population of different conformations and provide information such as folding rate, equilibrium distribution, and transition states, which can be correlated to experimental results. For completeness, we sketch the Map-based Master Equation (MME) method we introduced in [23] to analyze the population kinetics. Master equation formalism has been developed for folding kinetics in a number of earlier studies [12,3] The stochastic folding process is represented as a set of transitions among all n conformations (states). The time evolution of the population of each state, $P_i(t)$, can be described by:

$$dP_i(t)/dt = \sum_{i \neq j}^{n} (K_{ji} P_j(t) - K_{ij} P_i(t)) \tag{5}$$

where K_{ij} denotes the transition rate (probability) from state i to state j. The change in population $P_i(t)$ is the difference between transitions *to* and *from* state i. We compute transition rates from the roadmap's edge weights: $K_{ij} = K_0 e^{-W_{ij}}$. K_0 is a constant adjusted according to experimental results.

If we use an n-dimensional column vector $\mathbf{p}(\mathrm{t}) = (P_1(t), P_2(t), \ldots, P_n(t))'$ to denote the population of all n conformational states, then we can construct an $n \times n$ matrix M to represent the transitions, where

$$\begin{cases} M_{ij} = K_{ji} & i \neq j \\ M_{ii} = -\sum_{i \neq j} K_{ij} & i \neq j \end{cases} \tag{6}$$

The master equation can be represented in matrix form:

$$d\mathbf{p}(t)/dt = M\mathbf{p}(t). \tag{7}$$

The solution to the master equation is:

$$P_i(t) = \sum_k \sum_j N_{ik} e^{\lambda_k t} N_{kj}^{-1} P_j(0) \tag{8}$$

where N is the matrix of eigenvectors N_i for the matrix M in equation 6 and λ_i is its corresponding eigenvalue. $P_j(0)$ is the initial population of conformation j.

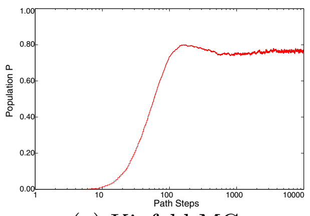
From equation 8, we see that the eigenvalue spectrum is composed of n modes. If sorted by magnitude in ascending order, the eigenvalues include $\lambda_0 = 0$ and several small magnitude eigenvalues. Since all the eigenvalues are negative, the population kinetics will stabilize over time. The population distribution $\mathbf{p}(t)$ will converge to the equilibrium Boltzmann distribution, and no mode other than the mode with the zero eigenvalue will contribute to the equilibrium. Thus the eigenmode with eigenvalue $\lambda_0 = 0$ corresponds to the stable distribution, and its eigenvector corresponds to the equilibrium Boltzmann distribution.

By a similar argument, large magnitude eigenvalues correspond to fast folding modes, i.e., those which fold in a burst. Their contribution to the population will die away quickly. Conversely, small magnitude eigenvalues have a large influence on the global folding process, and thus determine the global folding rates.

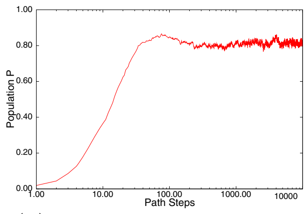
5 Results and Discussion

Here we present our simulation results and validate our methods against both another computational method (Monte Carlo Simulation) and experimental data. The computational validations show that our small roadmaps can capture the major features of much larger complete energy landscapes efficiently. The roadmaps scale well with RNA length, which enables us to study larger RNA with hundreds of nucleotides. The experimental validation shows that our methods correctly computed the kinetics-based functions of two different RNA and their mutants by studying two different properties of the folding kinetics.

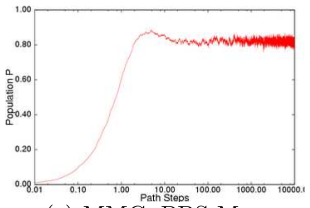
In Section 5.1, we compare the population kinetics using our roadmaps against other computational methods working on complete energy landscapes. We first quantitatively compare the population kinetics computed from different maps and show we can capture the major features of larger complete folding landscapes using much smaller roadmaps. Then, we empirically compare the scalability of our methods on different RNA. We present population kinetics using three different analysis methods: Map-based Master Equation (MME), Monte Carlo (MC) simulation, and Map-based Monte Carlo (MMC) simulation. The results show that the solutions of different methods are comparable to each other.



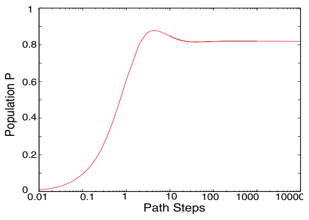
(a) Kinfold MC



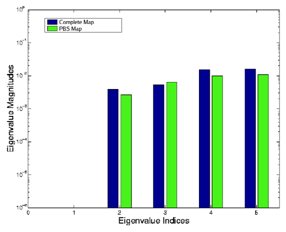
(b) MMC, Complete Map



(c) MMC, PBS Map



(d) ME, PBS Map



(e)

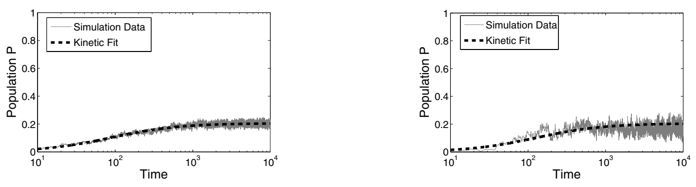
Fig. 1. The population kinetics of the native state of 1k2g: (a) Kinfold Monte Carlo simulation, (b) Our MMC simulation on a fully enumerated roadmap (12,137 conformations), (c) Our MMC simulation on a PBS roadmap (42 conformations), and (d) Master equation solution on the PBS roadmap (42 conformations). All analysis techniques produce similar population kinetics curves and similar equilibrium distribution. (e) Comparison of the eigenvalues of 1k2g by the master equation on a fully enumerated roadmap (12,137 conformations) and new PBS roadmap (42 conformations). Both eigenvalues are similar between the different roadmaps.

They also indicate that our roadmaps scale well for large RNA. In Section 5.2, we present two case studies to demonstrate how we can use our method to study kinetics-based functions. Our method correctly predicts (1) the relative plasmid replication rates of ColE1 RNAII and its mutants, and (2) the relative gene expression rates of MS2 phage RNA and its mutants.

5.1 Computational Validations

We demonstrate with two different RNA that the different analysis methods (ME, MC, MMC) produce comparable results and can be used interchangeably. This is important since some methods like the master equation do not scale as well as others like Map-based Monte Carlo simulation with RNA size.

Comparison with other Simulation methods. Here we present the results of 1k2g (CAGACUUCGGUCGCAGAGAUGG), a 22 nucleotide RNA. Figure 1 compares the population kinetics of the native state using (a) standard Monte Carlo simulation (implemented by Kinfold [7]), (b) Map-based Monte Carlo simulation on a fully enumerated roadmap (12,137 conformations), (c) Map-based Monte Carlo simulation on a roadmap with our new PBS sampling method (42 conformations), and (d) the master equation on a PBS roadmap (42 conformations). The fully enumerated roadmap is the most accurate model. However, it is



(a) Kinfold MC (b) MMC, PBS Map

Fig. 2. Comparison of population kinetics of a metastable state for Leptomonas Collosoma Spliced Leader RNA using (a) Kinfold Monte Carlo simulation and (b) our MMC simulation on a PBS roadmap with 5453 conformations. We are able to capture the same kinetics while only sampling a tiny fraction of the entire conformation space.

not feasible to enumerate RNA with more than 40 nucleotides. The statistical-sampling roadmaps yield much smaller subsets of the entire conformation space that effectively approximate the energy landscape. Note that numerical limitations in computing the eigenvalues and eigenvectors limit the master equation to small roadmaps (e.g., up to 10,000 conformations).

All population kinetics curves have similar features (see Figure 1). In each figure, the population first increases quickly, then it gradually decreases and eventually stabilizes to the equilibrium distribution. Note that the equilibrium (final) distributions are very close to each other at 80%, even though the PBS roadmap (c) and (d) contains less than 0.4% of all possible conformations. Thus, these roadmaps capture the main features of the energy landscape. This data indicates that these analysis methods are interchangeable.

Figure 1(e) compares the four smallest eigenvalues of the fully enumerated (base-pair) roadmap and the statistical-sampling roadmap computed by the master equation. All the eigenvalues, i.e., folding rates, are similar. This indicates that our extremely sparse roadmaps not only capture the major features of the equilibrium distribution, but also capture the major features of the kinetics.

Scalability of the Approximated Roadmaps. Here we compare our simulation results on a larger 56 nucleotide RNA. Leptomonas Collosoma Spliced Leader RNA is known to have many metastable structures [5]. This RNA has approximately $2.0 * 10^{14}$ conformations, so it is not feasible to enumerate even the stack-pair conformations, let alone the entire conformation space. Thus, we are only able to compare kinetics from the Kinfold Monte Carlo simulation and our Map-based Monte Carlo simulation using PBS roadmaps. For each simulation technique, we compute 1000 different folding pathways. We combine these pathways to calculate the population kinetics of a particular conformation.

Figure 2 shows that although we only use 5033 conformations in the roadmap, our Map-based Monte Carlo simulation results in (b) have qualitatively similar features with the Kinfold Monte Carlo simulation in (a). To qualitatively compare the two simulations, we fit a two state kinetic curve to both plots. First, kinetic parameters were derived for the Kinfold plot (hashed lines in Figure 2 (a)). Then, these parameters were used for the curve shown in Figure 2 (b).

For the fits shown, the only kinetic parameter that was modified was the rate at which the traces changed from unfolded to folded. These simulated rates changed from 89 for Kinfold to 130 for MMC. The similarity in these plots is striking because the MMC simulation approximates the entire conformation space ($2.0 * 10^{14}$ conformations) with only a tiny subset ($5.0 * 10^3$). In contrast, such kinetic features are very different from other RNA, such as the population kinetics of 1k2g shown in Figure 1. Again, this gives strong evidence that our sparse roadmap captures the main features of the energy landscape. Another benefit of our MMC simulation is that it requires fewer iterations to stabilize (an order of magnitude fewer) and uses less space (1G versus 8G for Kinfold).

5.2 Experimental Validation: Kinetics Related Functions

Many RNA can perform a variety of functions such as regulating the gene expression rate or plasmid replication rate. It has been found that some functions are not only determined by their native states but also by metastable states formed during the folding process, where the functional units are active [8,14,10,16]. Thus these functions are based on the RNA's folding *kinetics*. These functions are studied experimentally by comparing the kinetics and functional rates of different mutants that share the same thermodynamic stability and native structure. Below we give two case studies that show how we can also study these kinetics-based functions and compare to experimental data.

ColE1 RNAII: Predict Plasmid Replication Rates. ColE1 RNAII regulates the replication of E. coli ColE1 plasmids through its folding kinetics [9,14]. The slower it folds, the higher the plasmid replication rate. A specific mutant, MM7, differs from the wild-type (WT) by a single nucleotide out of the 200 nucleotide sequence. This mutation causes it to fold slower while maintaining the same thermodynamics of the native state. Thus, the overall plasmid replication rate increases in the presence of MM7 over the WT.

We can study this difference computationally by computing the folding rates of both WT and MM7 using the master equation and comparing their eigenvalues. A similar study is performed in [9]. However, they solve the master equation on a much more simplified energy landscape using a specific subsequence (130 of 200 nucleotides) and 9 stems hand-picked from 30 conformations. In contrast, we simulate the kinetics of the entire sequence using around 4000 conformations.

Figure 3 shows the eigenvalues calculated using the master equation. Note that the smallest non-zero eigenvalues correspond to the folding rate. All eigenvalues of WT are larger than MM7 indicating that WT folds faster than MM7. Thus, our method correctly estimated the functional level of the new mutant.

MS2 phage RNA: Predict Protein Expression Rate. MS2 phage RNA (135 nucleotides) regulates the expression rate of phage MS2 maturation protein [8,14] at the translational level. It works as a regulator only when a specific subsequence (the SD sequence) is open (i.e., does not form base-pair contacts). Since this SD sequence is closed in the native state, this RNA can only perform this function before the folding process finishes. Thus, its function is based on

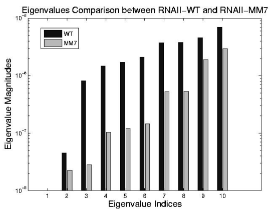


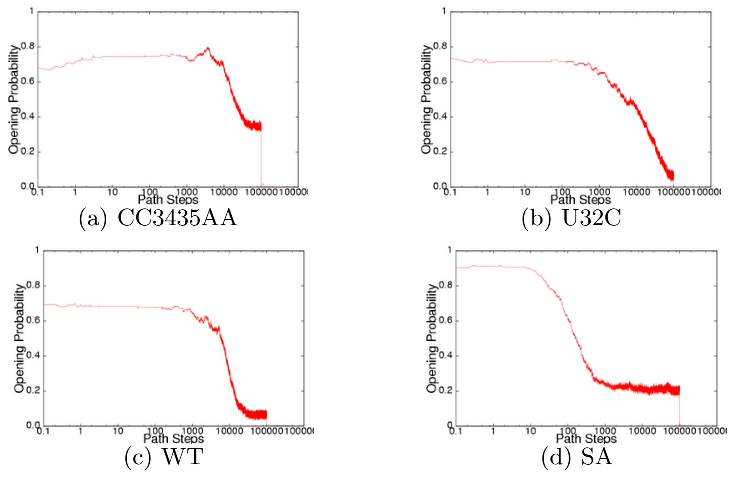
Fig. 3. Comparison of the 10 smallest non-zero eigenvalues (i.e., the folding rates) for WT and MM7 of ColE1 RNAII as computed by the master equation. The overall folding rate of WT is faster than MM7 matching experimental data.

its folding *kinetics* and not the final native structure. Three mutants have been studied that have similar thermodynamic properties as the wild-type (WT) but different kinetics and therefore different gene expression rates. Experimental results indicate that mutant CC3435AA has the highest gene expression rate, WT and mutant U32C are similar, and mutant SA has the lowest rate [8,14].

Intuitively, the functional rate (e.g., gene expression rate in this case) is correlated with the opening of the SD sequence. If the SD sequence is opened longer, or has higher opening probability (i.e., having more nucleotides on the SD sequence open), then the mutant should have higher functional rate. We use our simulation method to study this opening probability during the folding process. In our study, we first simulate the folding process for each mutant by generating 1000 folding pathways for each mutant using Map-based Monte Carlo simulation. Then we analyze the pathways for each mutant and calculate the opening probability of the SD sequence. We calculate the opening probability as the percentage of open nucleotides in the SD sequence. In [10], Higgs performed a similar study using a stem-based Monte-Carlo simulation. However, in that work, they simulated the folding process only when the RNA sequence is growing. Their results may depend on the selection of growth rate. If the growth rate was too high or too low, the results may or may not be able to compare to experiment. Our simulation results, on the other hand, do not require this growth rate parameter and thus can be used to quantitatively predict the functional level of a new mutant in a more reliable way.

Figure 4 shows the time evolution of the SD opening probability for the WT and the three mutants. Note that CC3435AA has the longest duration at a relatively high level of opening probability while SA has the shortest duration. This correlates with experimental data. The opening probability of U32C decreases earlier but finishes later than WT, so it is not clear which one has a larger total opening probability during folding, again matching experimental findings.

The gene expression rate is determined from two factors: (1) how high the opening probability is at any given time and (2) how long the RNA stays in the high opening probability state. To compare each RNA quantitatively, we compute the integration of the opening probability (Figure 4) over the whole folding process. Note that the RNA regulates gene expression only when the SD opening probability is "high enough". We used thresholds ranging from 0.2 to 0.6



(a) CC3435AA (b) U32C (c) WT (d) SA

Fig. 4. Comparison of the SD opening probability during the folding process

to estimate the gene expression rate. Thresholds higher than 0.6 will yield zero opening probability on WT and most mutants and thus cannot be correlated to experimental results. Similarly, thresholds lower than 0.2 are not considered since mutant SA could be active in the equilibrium condition, contradicting experimental results. Table 1 shows the results for the WT and for each mutant. For most thresholds, mutant CC3435AA has the highest rate and mutant SA has the lowest rate, the same relative functional rate as seen in experiment. In addition WT and mutant U32C have similar levels (particularly between 0.4-0.6), again correlating with experimental results. Aside from simply validating our method against experiment, we can also use our method to suggest that the SD sequence may only be active for gene regulation when more than 40% of its nucleotides are open.

Table 1. Comparison of expression rates between WT and three mutants of MS2. It shows that we can predict similar relative functional rates as seen in experiments.

Mutant	Experimental Expression Rate (order of magnitude)	Our Estimation				
		$t = 0.2$	$t = 0.3$	$t = 0.4$	$t = 0.5$	$t = 0.6$
SA	0.1	0.1	0.04	0.03	0.03	0.08
WT	1	1.0	1.0	1.0	1.0	1.0
U32C	1	2.1	1.8	1.4	0.8	1.2
CC3435AA	5	7.2	8.4	3.8	3.5	9.8

6 Conclusion

We have proposed new sampling techniques and a new analysis tool called Map-based Monte Carlo (MMC) simulation that can be used to study kinetics-based functions for RNA such as population kinetics, folding rates, and the folding of particular subsequences. These new tools enable us to study larger RNA than before – increasing from RNA with tens of nucleotides (e.g., 40) to those with hundreds of nucleotides (e.g., 200+).

We validated our method against known experimental data and analyzed two case studies in detail. For the first, we showed that our method identified the same relative folding rates as those noted in experiment for ColE1 RNAII and its mutant. In the second case study, we showed that our approach predicted the same relative gene expression rates of wild-type MS2 phage RNA and three of its mutants. We believe that our method will be a valuable tool for discovering such relationships for other RNA that have not been characterized experimentally. Although we only study secondary structure now, in the future, we plan to include pseudo knots and tertiary structures using an appropriate energy model.

Acknowledgments

We thank Dr. David Giedroc for many insightful discussions.

References

1. N. M. Amato, K. A. Dill, and G. Song. Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures. *J. Comput. Biol.*, 10(3-4):239–256, 2003. Special issue of Int. Conf. Comput. Molecular Biology (RECOMB) 2002.
2. D. Bartel. MicroRNAs: genomics, biogenesis, mechanism, and function. *Cell*, 116:281–297, 2004.
3. S.-J. Chen and K. A. Dill. RNA folding energy landscapes. *Proc. Natl. Acad. Sci. USA*, 97:646–651, 2000.
4. K. A. Dill and H. S. Chan. From Levinthal to pathways to funnels: The new view of protein folding kinetics. *Nat. Struct. Biol.*, 4:10–19, 1997.
5. Y. Ding and C. E. Lawrence. A statistical sampling algorithm for RNA secondary structure prediction. *Nucleic Acids Research*, 31:7280–7301, 2003.
6. R. Dirks and N. Pierce. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *Journal of Computational Chemistry*, 24:1664–1677, 2003.
7. C. Flamm. *Kinetic Folding of RNA*. PhD thesis, University of Vienna, Austria, August 1998.
8. H. Groeneveld, K. Thimon, and J. van Duin. Translational control of matruation-protein synthesis is phage MS2: a role of the kinetics of RNA folding? *RNA*, 1:79–88, 1995.
9. A. Gultyaev, F. V. Batenburg, and C. Pleij. The computer simulation of RNA folding pathways using a genetic algorithm. *J. Mol. Biol.*, 250:37–51, 1995.
10. P. G. Higgs. RNA secondary structure: physical and computational aspects. *Quarterly Reviews of Biophysics*, 33:199–253, 2000.
11. I. L. Hofacker. RNA secondary structures: A tractable model of biopolymer folding. *J. Theor. Biol.*, 212:35–46, 1998.

12. N. G. V. Kampen. *Stochastic Processes in Physics and Chemistry.* North-Holland, New York, 1992.
13. L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.
14. P. Klaff, D. Riesner, and G. Steger. RNA structure and the regulation of gene expression. *Plant Mol. Biol.*, 32:89–106, 1996.
15. J. S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29:1105–1119, 1990.
16. J. H. Nagel and C. W. Pleij. Self-induced structural switches in RNA. *Biochimie*, 84:913–923, 2002.
17. M. E. J. Newman and G. T. Barkenma. *Monte Carlo Methods in Statistical Physics.* Clarendon Press, Oxford, 1999.
18. R. Nussinov, G. Piecznik, J. R. Griggs, and D. J. Kleitman. Algorithms for loop matching. *SIAM J. Appl. Math.*, 35:68–82, 1972.
19. S. B. Ozkan, K. A. Dill, and I. Bahar. Computing the transition state population in simple protein models. *Biopolymers*, 68:35–46, 2003.
20. E. Rivas and S. Eddy. A dynamic programming algorithm for rna structure prediction including pseduoknots. *JMB*, 285:2053–2068, 2000.
21. B. A. Shapiro, D. Bengali, W. Kasprzak, and J. C. Wu. RNA folding pathway functional intermediates: Their prediction and analysis. *J. Mol. Biol.*, 312:27–44, 2001.
22. G. Song, S. Thomas, K. Dill, J. Scholtz, and N. Amato. A path planning-based study of protein folding with a case study of hairpin formation in protein G and L. In *Proc. Pacific Symposium of Biocomputing (PSB)*, pages 240–251, 2003.
23. X. Tang, B. Kirkpatrick, S. Thomas, G. Song, and N. M. Amato. Using motion planning to study RNA folding kinetics. *J. Comput. Biol.*, 12(6):862–881, 2005. Special issue of Int. Conf. Comput. Molecular Biology (RECOMB) 2004.
24. S. Thomas, X. Tang, L. Tapia, and N. M. Amato. Simulating protein motions with rigidity analysis. In *Proc. Int. Conf. Comput. Molecular Biology (RECOMB)*, pages 394–409, 2006.
25. I. Tinoco and C. Bustamante. How RNA folds. *J. Mol. Biol.*, 293:271–281, 1999.
26. M. Wolfinger. The energy landscape of RNA folding. Master's thesis, University of Vienna, Austria, March 2001.
27. S. Wuchty. Suboptimal secondary structures of RNA. Master's thesis, University of Vienna, Austria, March 1998.
28. A. Xayaphoummine, T. Bucher, F. Thalmann, and H. Isambert. Prediction and statistics of pseudoknots in RNA structures using exactly clustered stochastic simulations. *Proc. Natl. Acad. Sci. USA*, 100:15310–15315, 2003.
29. M. Zuker, D. H. Mathews, and D. H. Turner. Algorithms and thermodynamics for RNA secondary structure prediction: A practical guide. In J. Barciszewski and B. F. C. Clark, editors, *RNA Biochemistry and Biotechnology*, NATO ASI Series. Kluwer Academic Publishers, 1999.

Multiple Sequence Alignment Based on Profile Alignment of Intermediate Sequences

Yue Lu[1] and Sing-Hoi Sze[1,2]

[1] Department of Biochemistry & Biophysics
[2] Department of Computer Science,
Texas A&M University, College Station, TX 77843, USA

Abstract. Despite considerable efforts, it remains difficult to obtain accurate multiple sequence alignments. By using additional hits from database search of the input sequences, a few strategies have been proposed to significantly improve alignment accuracy, including the construction of profiles from the hits while performing profile alignment, the inclusion of high scoring hits into the input sequences, the use of intermediate sequence search to link distant homologs, and the use of secondary structure information. We develop an algorithm that integrates these strategies to further improve alignment accuracy by modifying the pair-HMM approach in ProbCons to incorporate profiles of intermediate sequences from database search and utilize secondary structure predictions as in SPEM. We test our algorithm on a few sets of benchmark multiple alignments, including BAliBASE, HOMSTRAD, PREFAB and SABmark, and show that it significantly outperforms MAFFT and ProbCons, which are among the best multiple alignment algorithms that do not utilize additional information, and SPEM, which is among the best multiple alignment algorithms that utilize additional hits from database search. The improvement in accuracy over SPEM can be as much as 5 to 10% when aligning divergent sequences. A software program that implements this approach (ISPAlign) is at http://faculty.cs.tamu.edu/shsze/ispalign.

1 Introduction

Although many algorithms have been proposed for multiple sequence alignment (Thompson et al. 1994; Morgenstern et al. 1996; Stoye 1998; Notredame et al. 2000; Lee et al. 2002; Edgar 2004; Van Walle et al. 2004; Do et al. 2005; Katoh et al. 2005; Lassmann and Sonnhammer 2005; Pei and Grishin 2006; Roshan and Livesay 2006; Yamada et al. 2006), it remains difficult to obtain accurate alignments. Common techniques to improve alignment accuracy include performing iterative refinements after the initial alignment is constructed (Gotoh 1996; Edgar 2004; Do et al. 2005; Roshan and Livesay 2006; Yamada et al. 2006), using consistency-based pairwise alignments in progressive approaches (Notredame et al. 2000; Do et al. 2005; Pei and Grishin 2006; Roshan and Livesay 2006), and incorporating structural alignments (O'Sullivan et al. 2004; Van Walle et al. 2004). A few other strategies combine alignments from existing algorithms to obtain an improved alignment (Bucka-Lassen et al. 1999; Wallace et al. 2006).

With the rapidly increasing number of sequences in biological databases, it has been observed that the use of additional sequences from database search can significantly improve alignment accuracy. Among the most successful approaches that use this strategy are profile alignment algorithms that use database search to find related sequences for each input sequence, construct a profile from the hits, and then align the profiles instead of the sequences, including algorithms that start from two sequences (Marti-Renom et al. 2004), and algorithms that start from multiple sequences (Simossis et al. 2005; Zhou and Zhou 2005). Alternatively, Heger et al. (2004) identified clusters of residues to form columns of a multiple alignment by linking distant homologs through the hits.

We observe that instead of constructing a profile for each input sequence from the hits, which only compares each hit to the input sequence that generates it, it may be more accurate to perform a more extensive multiple alignment of the hits together with the input sequences, which allows comparisons among all the sequences involved. The usefulness of such a strategy has been demonstrated during the construction of the PREFAB database (Edgar 2004), in which the incorporation of additional hits from database search into the input sequences significantly improves accuracy as opposed to aligning the input sequences alone. One drawback of this approach is that the inclusion of hits that are not intermediate between the input sequences can introduce noise, since these hits do not contribute to defining a better alignment between them. We will show that a careful definition of intermediate sequences from database search in addition to the computation of profiles for these sequences will significantly improve alignment accuracy.

By defining an intermediate sequence as a common hit from database search that links two input sequences, an intermediate sequence search technique has been used successfully to establish distant homologs (Park et al. 1997; Gerstein 1998). The strategy was later generalized to multiple intermediate sequence search (Salamov et al. 1999; Li et al. 2000), in which chains of intermediate sequences found through iterative database search are used to link very distant homologs. Bolten et al. (2001) used such transitive homologies to cluster protein sequences for structure predictions. Heger et al. (2004) used a graph-theoretic approach to link intermediate sequences through transitive homologies to detect short active site motifs, while Margelevičius and Venclovas (2005) used the intermediate sequence search strategy to distinguish between reliable and unreliable regions in alignments. Instead of defining intermediate sequences as common hits, we will develop a more relaxed definition to maximize the amount of information that can be extracted from the hits.

Since the number of hits that are also intermediate sequences can be very large, it is not practical to simply add them to the input sequences and perform a multiple alignment on the combined sequence set. Motivated by the fact that similar sequences are likely to contain redundant information, our algorithm uses a greedy strategy to choose a small subset of intermediate sequences that are far away from each other, which, together with the original sequences form a combined set of input sequences. Instead of aligning these sequences directly, we construct a profile for each sequence in the combined set by incorporating information from other intermediate sequences and aligning the profiles by modifying the pair-HMM approach (Durbin et al. 1998) in ProbCons (Do et al. 2005). This is in contrast with the strategy used in Simossis et al. (2005) and Zhou and Zhou (2005) which constructs a profile from the hits of an input sequence. We will show that our strategy of constructing profiles from intermediate sequences instead of from the hits helps to prevent the introduction of excessive noise when aligning closely related sequences. To further improve alignment accuracy, we obtain a secondary structure prediction for each sequence in the combined set and incorporate these predictions into the pair-HMM alignment. While this strategy of using secondary structure predictions is similar to the one employed in Zhou and Zhou (2005), it is different from the technique used in Pei and Grishin (2006) which employs secondary structure information during HMM training without explicitly using secondary structure predictions in alignments.

We compare the performance of our algorithm to MAFFT (Katoh et al. 2005) and ProbCons (Do et al. 2005), which are among the best multiple alignment algorithms that do not utilize additional information, and SPEM (Zhou and Zhou 2005), which is among the best multiple alignment algorithms that utilize additional hits from database search, on benchmark multiple alignments from BAliBASE (Thompson et al. 2005), HOMSTRAD (Mizuguchi et al. 1998), PREFAB (Edgar 2004), and SABmark (Van Walle et al. 2004). We will show that our algorithm outperforms MAFFT, ProbCons and SPEM in almost all situations, with very significant improvements when aligning divergent sequences. Before presenting the algorithm in detail, we first describe the general strategies employed in each stage in the next few sections.

2 Finding Intermediate Sequences

Although most intermediate sequence search strategies define an intermediate sequence either as a common hit from database search that links two input sequences (Park et al. 1997; Gerstein 1998), or as hits that form a chain linking two input sequences (Salamov et al. 1999; Li et al. 2000), such a requirement is very stringent since it may not be possible to link very divergent sequences together even if the database search is performed iteratively. We consider the following relaxed definition of an intermediate sequence which only requires that it is intermediate between the two input sequences.

Definition 1. Given two sequences s_1 and s_2, and a distance score $d(s_1, s_2)$ between them, a sequence r is intermediate between s_1 and s_2 if $d(r, s_1) < d(s_1, s_2)$ and $d(r, s_2) < d(s_1, s_2)$.

The problem of finding intermediate sequences between multiple input sequences is defined as follows.

Definition 2. Given n input sequences $s_1, \ldots, s_n$, and m hits $r_1, \ldots, r_m$ from database search of these sequences, find all hits r_k that are intermediate between some pair of input sequences s_i and s_j.

Similar to previous approaches, our goal is to find an appropriate subset of sequences that contain useful information between the input sequences $s_1, \ldots, s_n$. We do not require that these intermediate sequences have a phylogenetic interpretation or have an appropriate evolutionary relationship to the input sequences. Also, since any hit that is intermediate between some pair of input sequences is potentially useful, it is included in the definition. Note that there is no need to compute pairwise distances between the potentially very large number of hits. The number of pairwise distance score computations that are needed to identify the intermediate sequences from among the hits is $O(mn + n^2)$, while the number of score comparisons is $O(mn^2)$.

3 Choosing Intermediate Sequences

The next problem of choosing a small subset of intermediate sequences to add to the input sequences is defined as follows. Our goal is to identify a combined set of sequences that are as divergent as possible.

Definition 3. Given n input sequences $s_1, \ldots, s_n$, m intermediate sequences $r_1, \ldots, r_m$, add k intermediate sequences from among $r_1, \ldots, r_m$, denoted by $s_{n+1}, \ldots, s_{n+k}$, so that the minimum distance between sequences in the combined set $s_1, \ldots, s_{n+k}$ is the largest possible when distances between the input sequences $s_1, \ldots, s_n$ are ignored.

Figure 1 shows a greedy algorithm that iteratively adds an intermediate sequence s_{n+j} that is farthest away from the current sequence set $s_1, \ldots, s_{n+j-1}$, in which the minimum distance between s_{n+j} and $s_1, \ldots, s_{n+j-1}$ is the largest possible. Although the greedy strategy does not guarantee optimum divergence of the sequences $s_1, \ldots, s_{n+k}$, they should be reasonably far away from each other. The total number of pairwise distance score computations needed is $O(m(n + k))$, and there is no need to compute distances between all pairs of the potentially very large number of intermediate sequences.

```
Input: $n$ input sequences $s_1, \ldots, s_n$, $m$ intermediate sequences $r_1, \ldots, r_m$,
          distance score $d(r, s)$ between two sequences $r$ and $s$.
Output: $k$ intermediate sequences $s_{n+1}, \ldots, s_{n+k}$ added to $s_1, \ldots, s_n$.

$R \leftarrow \{r_1, \ldots, r_m\}$;
for each $r_i$ in $R$ do { $d_i \leftarrow \min_{1 \le j \le n} d(r_i, s_j)$; }
for $j \leftarrow 1$ to $k$ do {
    $s_{n+j} \leftarrow r_i$ with the maximum $d_i$; remove $r_i$ from $R$;
    for each $r_i$ in $R$ do { $d_i \leftarrow \min(d_i, d(r_i, s_{n+j}))$; } }
```

Fig. 1. Greedy algorithm to choose a small subset of intermediate sequences to add to the input sequences

4 Constructing Sequence Profiles

Instead of aligning the sequences $s_1, \ldots, s_{n+k}$ directly, a profile is constructed for each of these sequences as follows: for each intermediate sequence r_i from among $r_1, \ldots, r_m$, assign it to the s_j from among $s_1, \ldots, s_{n+k}$ that is most similar to r_i. For each sequence s_j with assigned sequences $r_{i_1}, \ldots, r_{i_t}$, we combine all the pairwise alignments between s_j and each r_{i_p} into a star alignment with s_j as the center (Gusfield 1993). For each column in the star alignment that contains a residue of s_j, the relative frequency of each residue within the column is then used to construct a profile as a probability distribution of residues (gap characters are ignored). Here the choice of scoring functions for the profile is not very important since Edgar and Sjölander (2004) showed that most scoring functions do not have significant performance differences. One caution is that we need to make sure that the number of very closely related sequences assigned to each s_j is not excessively large to avoid over-contribution of these sequences to the profile. This can be achieved by removing sequences from the original set of intermediate sequences so that none of the remaining sequences are very similar to each other before choosing the subset of intermediate sequences. In difference from the approach in Simossis et al. (2005) and Zhou and Zhou (2005), hits that are not intermediate sequences are not used to avoid noise from these hits.

5 Alignment Via Modified Pair-HMM

We modify the pair-HMM approach in Durbin et al. (1998) to incorporate profiles and secondary structure predictions. The original model consists of three states: M emits an aligned pair of residues (x, y) with probability $e(x, y)$, X emits a residue x in the first sequence that is aligned to a gap with probability $e(x)$, while Y emits a residue y in the second sequence that is aligned to a gap with probability $e(y)$ (Fig. 2). In addition to the original residue, each position is now associated with a probability distribution of residues. Let $p_1(x, i)$ be the probability of finding the residue x at position i in the first sequence and let $p_2(y, j)$ be the probability of finding the residue y at position j in the second sequence. We modify the model to incorporate profiles as follows: define the emission probability of state M as $e'(i, j) = \sum_x \sum_y p_1(x, i)p_2(y, j)e(x, y)$ if the emission is at position i in the first sequence and at position j in the second sequence, the emission probability of state X as $e'(i) = \sum_x p_1(x, i)e(x)$ if the emission is at position i in the first sequence, and the emission probability of state Y as $e'(j) = \sum_y p_2(y, j)e(y)$ if the emission is at position j in the second sequence. These changes replace the original emission probabilities of the single residues by the average emission probabilities over a distribution of residues so that in the degenerate case when the profiles represent simple sequences, the effect is the same as before.

We incorporate secondary structure predictions into the pair-HMM model as follows: in state M, we introduce an additional parameter α and subdivide the emission probability $e'(i, j)$ into two cases to obtain a modified state $M(\alpha)$ with

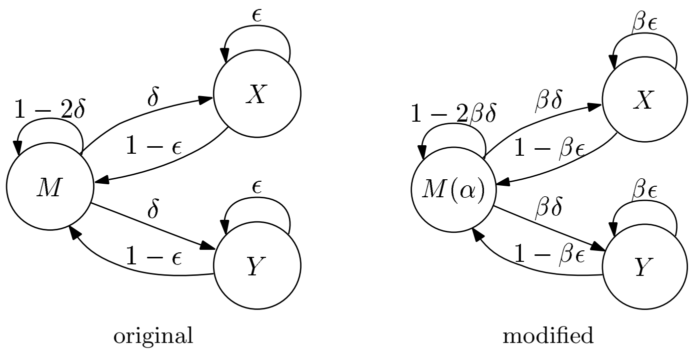


Fig. 2. The original and the modified pair-HMM models. In the original model, state M emits an aligned pair of residues, states X and Y emit a residue in the first and the second sequences respectively that is aligned to a gap, δ is the gap opening probability, and ϵ is the gap extension probability (Durbin et al. 1998). In the modified model, the state $M(\alpha)$ is obtained from M with emission probability $e(x,y)$ by defining the emission probability to be $\alpha e(x,y)$ if the paired residues (x,y) have the same secondary structure type and $(1-\alpha)e(x,y)$ otherwise. The factor β is applied to δ and ϵ to compensate for the change. To incorporate profiles, the residue emission probabilities are replaced by the average emission probabilities over a distribution of residues.

emission probability $\alpha e'(i,j)$ if the original paired residues (x,y) at position i in the first sequence and at position j in the second sequence have the same secondary structure type, and with emission probability $(1-\alpha)e'(i,j)$ otherwise. Since this decrease in emission probability will tend to allow more gaps than before in the ideal case in which every aligned residue pair has the same secondary structure type, we apply the factor β to the gap opening and extension probabilities to compensate for it while keeping the ratio between the two probabilities unchanged to preserve the affine gap model (Fig. 2). This modified pair-HMM can then be utilized within a progressive alignment strategy to obtain a multiple alignment (Do et al. 2005).

6 Detailed Algorithm

We now describe a procedure and the associated parameters that give very good results for our algorithm. Note that this is only among one of the many possible ways to implement the algorithm.

Following SPEM (Zhou and Zhou 2005), for each input sequence, we use PSI-BLAST (Altschul et al. 1997) to perform database search on a filtered version of the non-redundant protein database (NR) that excludes low complexity regions, transmembrane regions and likely coiled-coil regions (Jones 1999), and retain hits that have less than 98% identity to the input sequence and have e-value less than 0.001. One advantage of using PSI-BLAST is that it performs iterative database search automatically to look for distant homologs. Instead of keeping the entire sequence of a hit, only the regions within a PSI-BLAST local alignment are retained to avoid the introduction of noise from unrelated regions. Note that if there are more than one PSI-BLAST local alignment that satisfy the above condition within a hit, they are considered to be separate hits.

We then extract intermediate sequences from among these hits according to Definitions 1 and 2. To obtain an accurate distance score $d(s_1, s_2)$ between two sequence s_1 and s_2, we use SSEARCH (Smith and Waterman 1981) to obtain an optimal alignment between s_1 and s_2 and define $d(s_1, s_2)$ as the e-value of the alignment. Note that the use of e-values here does not pose any problems since no addition operations are performed.

To avoid over-contribution of very similar intermediate sequences in the later profile construction step, we use CD-HIT (Li et al. 2002) to remove some of the closely related sequences so that the identity between the remaining intermediate sequences is less than 85%. We then use Definition 3 and the algorithm in Fig. 1 with $k = 5$ to add at most five intermediate sequences to the input sequences to obtain $s_1, \ldots, s_{n+k}$. We choose $k = 5$ so that the final multiple alignment step will not become much slower than simply aligning the original input sequences. The identity of a pairwise alignment from SSEARCH is used to obtain an accurate distance score $d(s_1, s_2)$ between two sequences s_1 and s_2 by defining $d(s_1, s_2)$ as $1 -$ identity (note that this distance is different from what we use above). Note that CD-HIT cannot be used for this purpose since it initially uses counts of short tuples to estimate pairwise similarity, which is inaccurate when the identity level between the sequences $s_1, \ldots, s_{n+k}$ is low.

We then construct profiles according to the algorithm in Section 4 in which an intermediate sequence r_i is assigned to the sequence from among $s_1, \ldots, s_{n+k}$ that has the best SSEARCH alignment to r_i. To obtain a secondary structure prediction for each of the sequences $s_1, \ldots, s_{n+k}$, we follow SPEM (Zhou and Zhou 2005) and use PSIPRED (Jones 1999) to assign one of the three possible types (helix, strand or coil) to each residue.

With the profiles and secondary structure predictions, we modify ProbCons (Do et al. 2005) by changing its pair-HMM model according to Section 5. The parameters in Fig. 2 are as follows: the original residue emission probabilities and the transition probabilities δ and ϵ are from ProbCons. The parameter α that modifies the emission probabilities is 0.65, while the parameter β that modifies the transition probabilities is 0.75. These two parameters are determined by testing a few combinations and choosing one that gives satisfactory performance in PREFAB (Edgar 2004). We use the default setting in ProbCons that utilizes two sets of gap states with the same modifying parameter β for both sets. There is no change in the later progressive alignment or the iterative refinement steps and the alignment on the original input sequences is returned.

7 Performance on Benchmark Sets

We test our algorithm (ISPAlign) on benchmark multiple alignments from BAliBASE 3.0 (Thompson et al. 2005), HOMSTRAD (Mizuguchi et al. 1998), PREFAB 4.0 (Edgar 2004), and SABmark 1.65 (Van Walle et al. 2004). We

Table 1. Average SPS and CS scores (in %) on the full length sequence set in BAliBASE 3.0. Reference 1 is further subdivided into two subsets: 1V1 (< 25% identity), and 1V2 (20–40% identity). The number in braces denotes the number of alignments in each subset. Within each subset, the best accuracy value is in bold. The values in parentheses denote the p-values, with — indicating insignificant differences. Since most of the subsets are very small, p-values are computed only for reference 1 and the entire set. Twenty-two cases are omitted due to unavailability of results from SPEM.

	SPS				CS			
	MAFFT	ProbCons	SPEM	ISPAlign	MAFFT	ProbCons	SPEM	ISPAlign
1V1 {38}	64.8	64.5	73.1	**76.0**	44.6	40.4	51.6	**56.9**
1V2 {42}	92.8	93.4	92.1	**93.5**	83.9	85.6	82.6	**85.8**
1 (V1–V2) {80}	79.5	79.7	83.1	**85.2**	65.2	64.2	67.9	**72.1**
(vs MAFFT)			(4e–5)	(5e–8)			(0.01)	(2e–7)
(vs ProbCons)			(7e–4)	(2e–6)			(0.01)	(2e–5)
(vs SPEM)				(0.002)				(9e–5)
2 {37}	91.8	89.7	88.0	**91.9**	46.0	40.8	47.1	**53.8**
3 {29}	81.4	78.8	82.8	**83.5**	56.8	54.3	51.4	**59.9**
4 {36}	89.2	86.8	87.5	**90.3**	**67.9**	60.9	55.4	63.3
5 {14}	88.2	87.5	87.0	**90.3**	57.6	59.4	55.9	**63.9**
All (1–5) {196}	84.5	83.3	85.0	**87.5**	60.3	57.3	58.3	**64.6**
(vs MAFFT)			(0.005)	(2e–11)			(—)	(2e–10)
(vs ProbCons)			(5e–4)	(2e–13)			(—)	(4e–10)
(vs SPEM)				(3e–7)				(5e–11)

compare our performance to MAFFT 5.8 (using the most accurate linsi strategy, Katoh et al. 2005), ProbCons 1.10 (Do et al. 2005) and SPEM (Zhou and Zhou 2005).

For BAliBASE, two score measures are used to perform accuracy assessment of each multiple alignment on the original input sequences: the sum-of-pairs score (SPS) evaluates the percentage of residue pairs that an algorithm can align correctly in the reference alignment, while the column score (CS) evaluates the percentage of entire columns that an algorithm can align correctly (Thompson et al. 1999). For PREFAB, evaluations are made on the original pairs of input sequences using the Q score defined in Edgar (2004), which has the same meaning as the SPS score. For BAliBASE and PREFAB, evaluations are made only on the core regions that are assigned to the reference alignments. While we test MAFFT and ProbCons both on the original pairs in PREFAB and on the full set of sequences that includes random hits from database search, we test SPEM and ISPAlign only on the original pairs since these algorithms utilize hits from database search automatically. For SABmark, reference sequences are specified in pairs and evaluations are based on the f_D and the f_M scores in Van Walle et al. (2004), in which f_D has the same meaning as SPS and f_M evaluates the percentage of correctly aligned residue pairs in the test alignment. We define the f_D score and the f_M score for each alignment as the average f_D score and the average f_M score respectively over all these pairs. For each test set, we use the Wilcoxon matched-pairs signed-ranks test (Wilcoxon 1947) over large enough subsets with 0.05 as the p-value cutoff for significance.

Table 1 shows performance comparisons on the full length sequence set in BAliBASE 3.0. For both reference 1 and the entire set, ISPAlign improved over MAFFT, ProbCons and SPEM very significantly, with the biggest improvements

Table 2. Average SPS and CS scores (in %) on HOMSTRAD. Each subset includes all alignments with average pairwise identity within the specified range, with * indicating worse performance in *p*-value. Since ProbCons consistently performs better than MAFFT, comparisons are made only between ProbCons, SPEM and ISPAlign. Only the *p*-values for the CS scores are shown.

	SPS			CS			SPEM	ISPAlign	ISPAlign
	ProbCons	SPEM	ISPAlign	ProbCons	SPEM	ISPAlign	(vs ProbCons)	(vs ProbCons)	(vs SPEM)
0–20% {156}	49.7	67.2	**68.5**	43.1	61.0	**62.7**	(4e–23)	(5e–24)	(4e–5)
20–40% {459}	80.5	85.6	**86.8**	74.7	80.4	**81.9**	(2e–29)	(2e–53)	(7e–7)
40–70% {348}	94.8	94.9	**95.5**	92.2	92.3	**93.2**	(0.03)	(2e–9)	(0.003)
70–100% {69}	**99.1**	98.5	99.0	**99.1**	98.4	98.9	(0.007*)	(—)	(—)
All {1032}	81.9	86.8	**87.8**	77.4	82.7	**84.0**	(2e–46)	(8e–87)	(1e–12)

Table 3. Average Q scores (in %) on PREFAB 4.0. Each subset includes all structure pairs with identity within the specified range, with * indicating worse performance in *p*-value. Comparisons are made between MAFFT and ProbCons using two sequences (MAFFT^2, ProbCons^2) and using all (at most 50) sequences (MAFFT^{50}, ProbCons^{50}), SP^2 (which is a specialized version of SPEM for two sequences), and ISPAlign^2 (ISPAlign starting from two sequences). Since MAFFT^{50} has the best accuracy among MAFFT and ProbCons, *p*-value comparisons are made only against MAFFT^{50}.

	MAFFT^2	ProbCons^2	MAFFT^{50}	ProbCons^{50}	SP^2	ISPAlign^2	SP^2 (vs MAFFT^{50})	ISPAlign^2 (vs MAFFT^{50})	ISPAlign^2 (vs SP^2)
0–20% {887}	36.2	38.9	56.7	55.6	64.6	**64.8**	(3e–36)	(5e–46)	(0.03)
20–40% {588}	81.0	82.8	87.1	87.2	89.7	**90.1**	(2e–16)	(6e–28)	(0.01)
40–70% {112}	96.2	96.4	96.0	95.4	95.3	**97.6**	(0.02*)	(—)	(—)
70–100% {95}	97.9	97.8	**98.0**	97.3	97.2	**98.0**	(6e–4*)	(—)	(0.005)
All {1682}	59.4	61.4	72.3	71.7	77.3	**77.7**	(1e–46)	(7e–69)	(2e–4)

in the 1V1 subset when identity is very low (improvement in the CS score was over 5%). SPEM improved over MAFFT and ProbCons very significantly for the SPS score. For the CS score, SPEM significantly improved over MAFFT and ProbCons for reference 1, but the overall improvement was not significant for the entire set.

Table 2 shows performance comparisons on HOMSTRAD. Except for 70 to 100% identity, all the *p*-values of ISPAlign over SPEM, ISPAlign over ProbCons, and SPEM over ProbCons were highly significant. For 70 to 100% identity, SPEM performed significantly worse than ProbCons, while the differences between ISPAlign and ProbCons or SPEM were not significant. In general, as identity increases, less improvements were observed for both SPEM and ISPAlign.

Table 3 shows performance comparisons on PREFAB 4.0 using two versions of MAFFT and ProbCons: MAFFT^2 and ProbCons^2 use the original input pair, while MAFFT^{50} and ProbCons^{50} use the full sequence set that includes random hits from database search and has at most 50 sequences. For 0 to 20% identity and 20 to 40% identity, the improvements of SPEM or ISPAlign over MAFFT^{50} were highly significant, while the improvements of ISPAlign over SPEM were significant but not as much. For 40 to 70% identity, SPEM performed significantly worse than MAFFT^{50}, while the differences between ISPAlign and MAFFT^{50}

Table 4. Average f_D and f_M scores (in %) on the Twilight and Superfamily subsets of SABmark 1.65. Four cases are omitted in the Twilight subset and three cases are omitted in the Superfamily subset since no reference alignments of sufficiently good quality are available. None of these subsets include false positive sequences. Since ProbCons consistently performs better than MAFFT, comparisons are made only between ProbCons, SPEM and ISPAlign.

	f_D			f_M		
	ProbCons	SPEM	ISPAlign	ProbCons	SPEM	ISPAlign
Twilight {205}	29.3	44.2	**46.1**	21.0	30.8	**32.0**
(vs ProbCons)		(2e–26)	(6e–29)		(1e–27)	(3e-29)
(vs SPEM)			(0.01)			(0.005)
Superfamily {422}	57.1	68.3	**69.0**	43.6	50.9	**51.6**
(vs ProbCons)		(4e–49)	(1e–51)		(1e–48)	(1e–51)
(vs SPEM)			(0.02)			(7e–4)

Table 5. Average CS scores (in %) on HOMSTRAD and average Q scores (in %) on PREFAB 4.0 using a few methods that are of increasing levels of complexity. Method 1 constructs a profile from the hits of each input sequence and performs profile alignment using the modified HMM model that incorporates profiles but not secondary structure predictions. Method 2 removes the hits that are not intermediate sequences before performing profile alignment. Method 3 adds intermediate sequences to the input sequences, constructs profiles based on the intermediate sequences and performs profile alignment on the combined sequence set. Method 4 is the full ISPAlign algorithm that also utilizes secondary structure predictions. For PREFAB, ProbCons uses the original input pair while all the methods start from this input pair. The p-value comparisons are made against the previous method to the left, with * indicating worse performance.

	HOMSTRAD CS					PREFAB Q				
	ProbCons	Method1	Method2	Method3	Method4	ProbCons	Method1	Method2	Method3	Method4
0–20%	43.1	59.1	59.2	59.4	**62.7**	38.9	58.2	58.6	61.3	**64.8**
(vs previous)		(3e–22)	(—)	(0.04)	(6e–8)		(2e–103)	(—)	(6e–12)	(7e–29)
20–40%	74.7	79.1	79.6	81.4	**81.9**	82.8	88.7	89.0	89.7	**90.1**
(vs previous)		(2e–24)	(0.003)	(7e–14)	(0.005)		(9e–45)	(—)	(2e–4)	(0.004)
40–70%	92.2	92.1	92.5	93.1	**93.2**	96.4	94.4	96.6	**97.8**	97.6
(vs previous)		(—)	(8e–4)	(0.001)	(—)		(—)	(0.002)	(—)	(0.008*)
70–100%	99.1	98.2	99.1	**99.2**	98.9	97.8	97.0	96.9	**98.1**	98.0
(vs previous)		(6e–4*)	(1e–4)	(—)	(0.003*)		(0.04*)	(0.02)	(—)	(—)
All	77.4	81.7	82.2	83.2	**84.0**	61.4	73.5	73.9	75.7	**77.7**
(vs previous)		(5e–38)	(1e–6)	(1e–14)	(1e–6)		(7e–146)	(—)	(2e–15)	(4e–28)

or SPEM were not significant. For 70 to 100% identity, ISPAlign performed significantly better than SPEM but did not improve over MAFFT^{50}, while SPEM performed significantly worse than MAFFT^{50}. For the entire set, all the p-values of ISPAlign over MAFFT^{50}, ISPAlign over MAFFT^{50} and SPEM over MAFFT^{50} were highly significant.

Table 4 shows performance comparisons on the Twilight and Superfamily subsets of SABmark 1.65. While the improvements of SPEM or ISPAlign over ProbCons for both subsets were highly significant, the improvements of ISPAlign over SPEM were significant but not as much.

In all the subsets that we have assessed, ISPAlign always performs at least as well as ProbCons and SPEM and is much better in many cases, especially when the input sequences are divergent in which the improvements are always significant and in many cases highly significant. Also, the improvements in the CS scores are sometimes more significant than the improvements in the SPS scores. In general, the contribution from utilizing additional sequences from database search decreases as the input sequences become more closely related. When the input sequences become very similar, while SPEM has significant accuracy decreases in many cases, ISPAlign still always performs at least as well. Since not many intermediate sequences are added to the input sequences before performing the profile alignment step, ISPAlign is efficient enough to perform an individual multiple alignment of moderate size in a reasonable time. In most cases, ISPAlign is only slightly slower than SPEM, with at most about a two times slowdown in some cases.

To evaluate contributions from various components of the algorithm to the alignment accuracy under different identity levels, we compare the performance of a few methods that are of increasing levels of complexity on HOMSTRAD and PREFAB 4.0 (Table 5). When the identity is low, the biggest improvements were from the use of profiles, while significant improvements were obtained from the addition of intermediate sequences to the input sequences and from the use of secondary structure predictions. When the identity is high, improvements were mainly from the removal of hits that are not intermediate sequences.

8 Discussion

While we have described a procedure for ISPAlign that gives very good performance, there are still many opportunities to further improve its accuracy. Instead of adding a fixed number of intermediate sequences to the input sequences, it may be better to add more sequences as the number of input sequences increases. Alternatively, intermediate sequences can be added until all the minimum distances between each of the remaining intermediate sequences and the current set of sequences fall below a threshold. Also, instead of modifying the parameters used by ProbCons by applying the factors α and β, it may be better to re-train the pair-HMM using a set of confirmed secondary structures. This can be done in a framework suggested by Do et al. (2006). It is also possible to use other multiple alignment algorithms to perform the profile alignment step as long as profiles and secondary structure predictions can be incorporated, which can lead to further improvements as better multiple alignment algorithms become available. It may also be beneficial to utilize three-dimensional structures when they are available.

Acknowledgments. This work was supported by NSF grants CCR-0311590 and DBI-0421815.

References

Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. **25** (1997) 3389–3402

Bolten, E., Schliep, A., Schneckener, S., Schomburg, D., Schrader, R.: Clustering protein sequences — structure prediction by transitive homology. Bioinformatics **17** (2001) 935–941

Bucka-Lassen, K., Caprani, O., Hein, J.: Combining many multiple alignments in one improved alignment. Bioinformatics **15** (1999) 122–130

Do, C.B., Gross, S.S., Batzoglou, S.: CONTRAlign: discriminative training for protein sequence alignment. Lect. Notes Bioinformatics **3909** (2006) 160–174

Do, C.B., Mahabhashyam, M.S., Brudno, M., Batzoglou, S.: ProbCons: probabilistic consistency-based multiple sequence alignment. Genome Res. **15** (2005) 330–340

Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: Biological sequence analysis. Cambridge University Press (1998)

Edgar, R.C.: MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res. **32** (2004) 1792–1797

Edgar, R.C., Sjölander, K.: A comparison of scoring functions for protein sequence profile alignment. Bioinformatics **20** (2004) 1301–1308

Gerstein, M.: Measurement of the effectiveness of transitive sequence comparison, through a third 'intermediate' sequence. Bioinformatics **14** (1998) 707–714

Gotoh, O.: Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. J. Mol. Biol. **264** (1996) 823–838

Gusfield, D.: Efficient methods for multiple sequence alignment with guaranteed error bounds. Bull. Math. Biol. **55** (1993) 141–154

Heger, A., Lappe, M., Holm, L.: Accurate detection of very sparse sequence motifs. J. Comp. Biol. **11** (2004) 843–857

Jones, D.T.: Protein secondary structure prediction based on position-specific scoring matrices. J. Mol. Biol. **292** (1999) 195–202

Katoh, K., Kuma, K., Toh, H., Miyata, T.: MAFFT version 5: improvement in accuracy of multiple sequence alignment. Nucleic Acids Res. **33** (2005) 511–518

Lassmann, T., Sonnhammer, E.L.L.: Kalign — an accurate and fast multiple sequence alignment algorithm. BMC Bioinformatics **6** (2005) 298

Lee, C., Grasso, C., Sharlow, M.F.: Multiple sequence alignment using partial order graphs. Bioinformatics **18** (2002) 452–464

Li, W., Jaroszewski, L., Godzik, A.: Tolerating some redundancy significantly speeds up clustering of large protein databases. Bioinformatics **18** (2002) 77–82

Li, W., Pio, F., Pawlowski, K., Godzik, A.: Saturated BLAST: an automated multiple intermediate sequence search used to detect distant homology. Bioinformatics **16** (2000) 1105–1110

Margelevičius, M., Venclovas, Č.: PSI-BLAST-ISS: an intermediate sequence search tool for estimation of the position-specific alignment reliability. BMC Bioinformatics **6** (2005) 185

Marti-Renom, M.A., Madhusudhan, M.S., Sali, A.: Alignment of protein sequences by their profiles. Protein Sci. **13** (2004) 1071–1087

Mizuguchi, K., Deane, C.M., Blundell, T.L., Overington, J.P.: HOMSTRAD: a database of protein structure alignments for homologous families. Protein Sci. **7** (1998) 2469–2471

Morgenstern, B., Dress, A., Werner, T.: Multiple DNA and protein sequence alignment based on segment-to-segment comparison. Proc. Natl. Acad. Sci. USA **93** (1996) 12098–12103

Notredame, C., Higgins, D.G., Heringa, J.: T-Coffee: a novel method for fast and accurate multiple sequence alignment. J. Mol. Biol. **302** (2000) 205–217

O'Sullivan, O., Suhre, K., Abergel, C., Higgins, D.G., Notredame, C.: 3DCoffee: combining protein sequences and structures within multiple sequence alignments. J. Mol. Biol. **340** (2004) 385–395

Park, J., Teichmann, S.A., Hubbard, T., Chothia, C.: Intermediate sequences increase the detection of homology between sequences. J. Mol. Biol. **273** (1997) 349–354

Pei, J., Grishin, N.V.: MUMMALS: multiple sequence alignment improved by using hidden Markov models with local structural information. Nucleic Acids Res. **34** (2006) 4364–4374

Roshan, U., Livesay, D.R.: Probalign: multiple sequence alignment using partition function posterior probabilities. Bioinformatics **22** (2006) 2715–2721

Salamov, A.A., Suwa, M., Orengo, C.A., Swindells, M.B.: Combining sensitive database searches with multiple intermediates to detect distant homologues. Protein Eng. **12** (1999) 95–100

Simossis, V.A., Kleinjung, J., Heringa, J.: Homology-extended sequence alignment. Nucleic Acids Res. **33** (2005) 816–824

Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. J. Mol. Biol. **147** (1981) 195–197

Stoye, J.: Multiple sequence alignment with the divide-and-conquer method. Gene **211** (1998) GC45–56

Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice. Nucleic Acids Res. **22** (1994) 4673–4680

Thompson, J.D., Koehl, P., Ripp, R., Poch, O.: BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark. Proteins **61** (2005) 127–136

Thompson, J.D., Plewniak, F., Poch, O.: A comprehensive comparison of multiple sequence alignment programs. Nucleic Acids Res. **27** (1999) 2682–2690

Van Walle, I., Lasters, I., Wyns, L.: Align-m — a new algorithm for multiple alignment of highly divergent sequences. Bioinformatics **20** (2004) 1428–1435

Wallace, I.M., O'Sullivan, O., Higgins, D.G., Notredame, C.: M-Coffee: combining multiple sequence alignment methods with T-Coffee. Nucleic Acids Res. **34** (2006) 1692–1699

Wilcoxon, F.: Probability tables for individual comparisons by ranking methods. Biometrics **3** (1947) 119–122

Yamada, S., Gotoh, O., Yamana, H.: Improvement in accuracy of multiple sequence alignment using novel group-to-group sequence alignment algorithm with piecewise linear gap cost. BMC Bioinformatics **7** (2006) 524

Zhou, H., Zhou, Y.: SPEM: improving multiple sequence alignment with sequence profiles and predicted secondary structures. Bioinformatics **21** (2005) 3615–3621

Connectedness Profiles in Protein Networks for the Analysis of Gene Expression Data

Joël Pradines[1], Vlado Dančík[1,2], Alan Ruttenberg[1], and Victor Farutin[1,3]

[1] Computational Sciences, Millennium Pharmaceuticals Inc., 40 Landsdowne Street, Cambridge, MA 02139, USA
[2] Mathematical Institute, Slovak Academy of Sciences, Grešákova 6, 040 01 Košice, Slovakia
[3] Pfizer Global R&D, Research Technology Center, 620 Memorial Drive, Cambridge, MA 02139, USA

Abstract. Knowledge about protein function is often encoded in the form of large and sparse undirected graphs where vertices are proteins and edges represent their functional relationships. One elementary task in the computational utilization of these networks is that of quantifying the density of edges, referred to as connectedness, inside a prescribed protein set. For instance, many functional modules can be identified because of their high connectedness. Since individual proteins can have very different numbers of interactions, a connectedness measure should be well-normalized for vertex degree. Namely, its distribution across random sets of vertices should not be affected when these sets are biased for hubs. We show that such degree-robustness can be achieved via an analytical framework based on a model of random graph with given expected degrees. We also introduce the concept of connectedness profile, which characterizes the relation between adjacency in a graph and a prescribed order of its vertices. A straightforward application to gene expression data and protein networks is the identification of tissue-specific functional modules or cellular processes perturbed in an experiment. The strength of the mapping between gene-expression score and interaction in the network is measured by the area of the connectedness profile. Deriving the distribution of this area under the random graph enables us to define degree-robust statistics that can be computed in $O(M)$, M being the network size. These statistics can identify groups of microarray experiments that are pathway-coherent, and more generally, vertex attributes that relate to adjacency in a graph.

1 Introduction

Much knowledge about the interactome is now available [1,2]. Whether it is an artifact of sampling or not [3], large graphs representing protein interactions always display a marked heterogeneity of vertex degrees: a small number of hubs and a majority of proteins having few interactions. Therefore, to avoid bias towards or against hubs, computational work with protein networks often requires a control that preserves vertex degrees. A commonly used control is the random graph with fixed degree sequence: edges are randomly paired and rewired [4,5,6]. However, such numerical approach is time expensive and thus limits the range of applications [7,8]. There is no easy analytical alternative in the case of the random graph with fixed degree sequence, because it models edges as dependent random variables [8,9,10,11,12]. Rendering edges independent yields a model known as the Random Graph with Given Expected Degrees (RGGED) [12,13]. Some of the analytical treatment becomes then much easier [14].

In this paper we show that the RGGED is a powerful framework to design statistics that are not degree-biased and for which there exist good analytical approximations. This allows us to perform degree-robust yet fast computational work with protein networks. We focus on one quantity referred to as connectedness: the density of interactions inside a prescribed protein set. For instance, many protein functional modules can be identified because of their high connectedness [15].

The connectedness of a protein set L is quantified by the probability $P_L(x)$ of observing x or more edges inside L in the RGGED. The p-value P_L can be rapidly computed via an approximation based on the Poisson distribution. We show that P_L is robust with respect to vertex degrees in the sense that its distribution across random sets of $|L|$ proteins is not very sensitive to the degrees represented in L. Also, we demonstrate that P_L has a rather intuitive meaning: it is a reasonable approximation of the p-value for higher connectedness when selecting a random set of $|L|$ proteins.

We then introduce the more general concept of a connectedness profile, which characterizes the relation between adjacency in a network and a given order of its vertices. An application to gene expression data and protein networks is the identification of tissue-related functional modules or cellular processes perturbed in an experiment. The strength of the mapping between gene expression score and interaction in the network can be quantified by the area of the connectedness profile. Studying the distribution of this area under the RGGED yields degree-robust statistics that can be computed in $O(M)$, where M is the network size. These statistics are useful to identify groups of microarray experiments that are pathway-coherent, and more generally, vertex attributes that relate to adjacency in a network. The profile area is similar in spirit to the mixing coefficient previously introduced by Newman [16]. The difference is that our statistics were designed via a random graph model in order to be degree-robust. In addition, not just its area but other features of a connectedness profile provide useful information.

In the next sections, as mathematical results are derived, we provide examples of application to the analysis of gene expression data with a network representing known functional relationships between proteins. The microarray data was downloaded from the Gene Expression Omnibus resource of NCBI [17]. The protein network represents protein interactions stored in the Human Protein Reference Database (HPRD) [2] and adjacency of human enzymes in the metabolic pathways of the LIGAND database [18,19]. The full graph has 7,302 vertices, 36,344 edges, median degree 4 and maximal degree 165.

2 Connectedness Profile

We start with the problem of quantifying the connectedness of a given protein set and then introduce the concept of connectedness profile.

2.1 Quantifying Connectedness

The protein network is modeled with an undirected graph $G(V, E)$, where the vertex set V represents proteins and the edge set E their functional relationships. We use the following notations: $N = |V|$, $M = |E|$, k_i is the degree of vertex v_i and $\langle k \rangle$ the average vertex degree. Consider a prescribed order of the vertices: $L_N = (v_1, \ldots, v_N)$. For instance, v_1 is the most differentially expressed gene in an experiment, v_2 the second most and so on. We wish to test for a relation between L_N and E: are the first elements of L_N closely related in the graph? We call L_i the restriction of L_N to its first i elements. If the number of edges x_i of the subgraph induced by L_i is large, then L_i is well connected. To define what a large x_i means we need a control. An obvious one is a random set of i vertices, i.e. a permutation of the elements of L_N. The set of all permutations defines a random variable $\mathbf{x}_i$ and large values of x_i correspond to small values of $\Pr(\mathbf{x}_i \geq x_i)$. However, a small p-value based on $\mathbf{x}_i$ might only reflect that L_i contains vertices of large degrees. We illustrate this with a numerical experiment, where connectedness is perturbed by adding the network hub to random protein sets.

We generated 10^5 random sequences L_N of all proteins and, for each sublist L_i $(i < N)$, estimated the threshold value T_i of x_i for which $\Pr(\mathbf{x}_i \geq T_i) \leq 0.05$. One protein of each list L_i was then replaced with the network hub ($k = 165$) and the frequency F of induced sizes x_i greater than T_i estimated. If x_i were a degree-robust measure, the values of F should stay close to 0.05. Instead, the black circles of Fig. 1(a) show that F tends to be much higher. This is due to the wide range of degrees in the network combined with the small proportion of hubs.

A better connectedness measure should take into account the degrees of the vertices in L_i. Given $k_1, \ldots, k_i$ we can compute the largest possible value x_i^m of x_i with an algorithm similar to that designed to test whether an integer sequence is graphical or not [14,20,21]. We can then quantify connectedness with x_i/x_i^m. Yet, the white circles of Fig. 1(a) show that x_i/x_i^m is still very degree sensitive. Indeed, the relative increase of x_i^m is small compared to that of x_i upon the addition of a hub in L_i. Such sensitivity to degrees is problematic. The goal of testing a list for its connectedness is to detect functional proximity, not the presence of hubs.

A much better robustness to degrees is obtained for a connectedness measure that we refer to as P_L [14]. The squares of Fig. 1(a) show that the distribution of P_L across protein sets is barely affected when the network hub is added to each set. $P_L(x)$ is the probability of observing x or more edges in L under a null model of the network known as the Random Graph with Given Expected

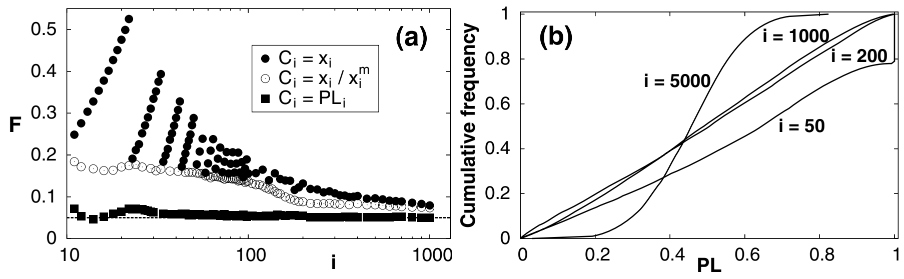


Fig. 1. (a) Sensitivity of different connectedness measures to vertex degree. A degree-robust measure C_i should yield values of F close to 0.05. (b) Distributions of P_{L_i} across random sets of i proteins.

Degrees (RGGED) [12,13]. In the RGGED all possible edges $v_i v_j$ are modeled with independent Bernoulli random variables of parameters

$$p_{ij} = \min\left(k_i k_j / 2M, 1\right) \text{ with } 2M = N \langle k \rangle . \tag{1}$$

The RGGED tends to preserve vertex degrees in a sparse network. We will consider two conditions of sparseness: strong: $k_i^2 \ll N \langle k \rangle, \forall i$ and weak: $\langle k \rangle / N \ll 1$. The strong condition implies the weak one and $p_{ij} \ll 1$. In the RGGED the size X of the subgraph induced by L is the sum of independent Bernoulli variables and P_L can be easily computed [14]:

$$P_L = \Pr(X \geq x) \simeq \alpha^{-1} e^{-\lambda} \sum_{y=x}^{m} \lambda^y / y!,\ \alpha = e^{-\lambda} \sum_{y=0}^{m} \lambda^y / y!,\ \lambda = \sum_{i<j|L} p_{ij}, \tag{2}$$

with $m = |L| (|L| - 1) / 2$ and $\alpha \simeq 1$. Equation (2) is a good approximation for strongly-sparse graphs [14], because error bounds scale as p_{ij}^2 and p_{ij} [22,23].

In addition to being degree-robust, P_L has also an interesting meaning with respect to the permutation control. Figure 1(b) displays a few distributions of P_{L_i} across random protein orders L_N. The important point is that for intermediate list sizes ($100 < i < 1000$) the distributions are close to uniform. Therefore, for such i values, P_{L_i} is a good approximation of the p-value for higher connectedness when selecting a random set of i proteins. In summary, P_L is a connectedness measure that has a simple interpretation, is degree-robust and can be rapidly computed.

2.2 P_L Profile

Rather than limiting ourselves to one list L_i, we can characterize the whole ordering L_N with the sequence $(\log(P_{L_1}), \ldots, \log(P_{L_N}))$ that we call a P_L profile. The computation of all the required x_i and λ_i values, as well as the variance σ_i^2 of X_i, is performed as follows.

(a) $x_1 = 0$, $\lambda_1 = 0$, $\sigma_1^2 = 0$, $k_m = k_1$, $\Sigma k = k_1$, $\Sigma k^2 = k_1^2$.
(b) $x_{i+1} = x_i + x_{a_{i+1}}$, with $x_{a_{i+1}} = |L_i \cap \nu(v_{i+1})|$, $\nu(v) = \{u \in V | uv \in E\}$.
(c) $p_m = k_m k_{i+1}/2M$. If $p_m > 1$ do (d) otherwise do (e)
  (d) $\lambda_{a_{i+1}} = \sum_{j=1}^{i} p_{i+1,j}$, $\sigma_{a_{i+1}}^2 = \sum_{j=1}^{i} p_{i+1,j}(1 - p_{i+1,j})$.
  (e) $\lambda_{a_{i+1}} = k_{i+1} \Sigma k / 2M$, $\sigma_{a_{i+1}}^2 = \lambda_{a_{i+1}} - k_{i+1}^2 \Sigma k^2 / 4M^2$.
(f) $\lambda_{i+1} = \lambda_i + \lambda_{a_{i+1}}$, $\sigma_{i+1}^2 = \sigma_i^2 + \sigma_{a_{i+1}}^2$, $\Sigma k \mathrel{+}= k_{i+1}$, $\Sigma k^2 \mathrel{+}= k_{i+1}^2$.
(g) if $k_{i+1} > k_m$, $k_m = k_{i+1}$.

All the steps are $O(1)$, except for (b) which is $O(k_i)$ and (d) which is $O(i)$. For the network we use here, (d) is never executed because there is no pair of vertices such that $k_i k_j > 2M$. Consequently, the computation is $O(M)$. More generally, the number of vertices requiring step (d) is small compared to N for large protein networks because they are weakly sparse: $\xi = \langle k \rangle / N \ll 1$. Let H stand for the set of vertices such that $k^2 > N \langle k \rangle$ and $\overline{H}$ its complement in V. The average degree over H is $\langle k \rangle_H = \sqrt{N \langle k \rangle} + \delta$, with $\delta > 0$. Let $f_H = |H|/N$ denote the fraction of hubs. We have

$$\langle k \rangle = f_H \langle k \rangle_H + (1 - f_H) \langle k \rangle_{\overline{H}} \Rightarrow f_H = \frac{\xi - \langle k \rangle_{\overline{H}}/N}{\sqrt{\xi} + \delta/N - \langle k \rangle_{\overline{H}}/N} \Rightarrow f_H < \sqrt{\xi}.$$

For large protein networks $\xi \approx 10^{-3}$ so that $f_H < 0.04$, which implies that step (d) is rarely executed and the computation of the P_L profile is fast.

2.3 Applications to Gene Expression Data

We consider the comparison of two groups A and B of microarray experiments. Each gene g is scored for its differential expression between A and B with

$$D(g, A, B) = \left(\langle \ln(I_g) \rangle_A - \langle \ln(I_g) \rangle_B\right) / \left(sd(\ln(I_g))_A + sd(\ln(I_g))_B\right), \quad (3)$$

where I is the intensity, $\langle \rangle_A$ denotes the average over group A and $sd()$ the standard deviation. Figure 2(a) presents two P_L profiles obtained when ordering genes by the score D comparing expression in one tissue (group A, two samples) to expression in other tissues (group B, 2×78 samples) [24]. We can approximate the pathways highly expressed in adipocytes with the subgraph induced by the first 393 genes, as they correspond to a clear local minimum of P_L. We broke this subgraph into densely connected parts called communities [25,26] and likely to represent functional modules. Often, this is done by looking for a partition into c vertex groups that maximizes the global modularity score

$$Q = \sum_{ij} (a_{ij} - p_{ij}) \sum_{r=1}^{c} \delta(g_i, r)\, \delta(g_j, r) = \sum_{r=1}^{c} x_r - \lambda_r, \quad (4)$$

where $a_{ij} = 1$ if $v_i v_j \in E$ and 0 otherwise, and δ is the Kronecker function. Here, instead of using only the difference between x and its expected value λ, we defined Q as the average $|\log(P_L)|$ across the c candidate communities. We obtained five main modules, two of which are displayed in Fig. 2. They correspond to

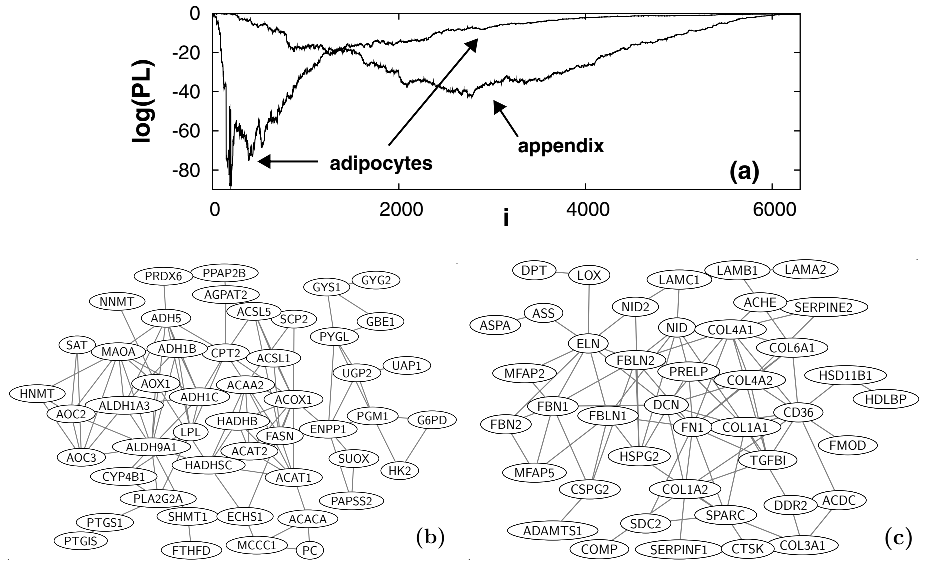


Fig. 2. (a) Two P_L profiles obtained when genes are ordered for their higher expression level in one tissue versus 78 others. (b-c) Parts of the network induced by the genes highly expressed in adipocytes.

lipid-metabolism pathways and components and modifiers of the extra-cellular matrix. These modules are relevant to adipose tissue. So, by combining the P_L profile and a community finding algorithm we can identify tissue related functional modules or cellular processes perturbed in an experiment.

However, such approach becomes questionable when the optimal i giving the best P_{L_i} is large, e.g. 43% of the vertices with the appendix samples. Partitioning such a large graph into communities is unlikely to be much driven by the expression scores. One can then adopt other strategies that identify portions of the graph rich in differentially expressed genes [27,28]. This said, the P_L profile gives us valuable information: differential expression can map to a large number of pathways, because functional neighbors tend to be co-expressed [29,30].

To illustrate this we generated 10^4 random partitions of 173 tumor samples [31] in two groups of sizes 86 and 87. Each partition was used to order genes by descending values of D (3), compute a P_L profile and measure some of its characteristics. The results, as well as those obtained with 10^4 random L_N sequences, are summarized in Table 1. The optimal i values corresponding to the minimal P_{L_i} values are large compared to the permutation control. The best values of P_L are smaller with the D-based order than they are with random order. A better measure than $\min P_L$ to characterize a profile is the average value of $\log(P_L)$, because its distributions for random and expression-based orders overlap little. The results of Table 1 show that the P_L profile can detect co-expression of

Table 1. Mean (standard deviation) of P_L profile characteristics for gene expression based and random protein orders

protein order	optimal i	$\min_i (\log P_{L_i})$	$\langle \log P_{L_i} \rangle_i$
based on gene expression	940 (771)	-7.1 (6.2)	-1.9 (1.3)
random	672 (834)	-1.3 (0.6)	-0.4 (0.1)

pathway neighbors. Now, to state about a given partition of the tumor samples, we might want to compare its P_L profile to those obtained with random partitions. Building such control distribution is not a problem because P_L profiles are fast to compute. As mentioned above, a good statistic to compare P_L profiles is the sum of the $\log(P_L)$ values. Each P_L is appropriately normalized for degrees, but adequate normalization of the sum requires further mathematical treatment.

3 Area of a Profile

In the previous section we indicated that P_L is a degree-robust measure which reasonably approximates the p-value for higher connectedness under the permutation control. In this section we show that the same holds for the area of a connectedness profile.

3.1 Distribution of Connectedness Profile Area Under the RGGED

Instead of working with $\log(P_{L_i})$ we consider a centered and reduced version of the size X_i of the subgraph induced by the protein set L_i:

$$Z_i = \frac{X_i - \lambda_i}{\sigma_i} \text{ for } i \geq 2 \text{ and } Z_1 = 0, \text{ with } \sigma_i^2 = \sum_{r<s\leq i} p_{rs}(1 - p_{rs}) \tag{5}$$

and p_{rs} being defined by (1). When i is large, the distribution of Z_i is close to normal. Convergence can be roughly estimated by approximating X_i with a Poisson variable:

$$\sigma_i^2 \simeq \lambda_i \simeq \frac{i(i-1)}{2} \frac{\langle k \rangle^2}{2M} \simeq \frac{i^2 \langle k \rangle}{2N}. \tag{6}$$

If our criterion for normality is $\lambda_i \geq 5$, then we must have $i \geq 86$ for the network used in this paper. For large i there is a monotonic mapping between Z_i and $\log(P_{L_i})$, so that the P_L profile area is well represented by the sum of the Z_i values. We define the Z-profile area up to index $n \leq N$ by

$$A_n = \sum_{i=1}^{n} Z_i = \sum_{i=2}^{n} \frac{X_i}{\sigma_i} - \gamma_n, \quad \gamma_n = \sum_{i=2}^{n} \frac{\lambda_i}{\sigma_i}. \tag{7}$$

By definition of Z_i, the expected value of A_n is 0. We next turn X_i into a sum of independent attachment variables X_{a_j}, each X_{a_j} being the number of edges between v_j and v_l for $l < j$. The variance S_n^2 of A_n is then easily obtained:

$$A_n = \sum_{i=2}^{n} \beta_i X_{a_i} - \gamma_n;\; S_n^2 = \sum_{i=2}^{n} \beta_i^2 \sigma_{a_i}^2,\; \beta_i = \sum_{j=i}^{n} \frac{1}{\sigma_j},\; \sigma_{a_i}^2 = \sum_{j<i} p_{ij}\left(1 - p_{ij}\right). \quad (8)$$

We argue that the asymptotic distribution of A_n is Gaussian. This is not trivial because the variables X_{a_i} have different distributions. Also, since the Z_i variables are dependent, their normality does not necessarily imply that of A_n. We decompose X_{a_i} into a sum of Bernoulli variables B_{ij}, one for each possible edge:

$$A_n = \sum_{i=2}^{n} \beta_i \sum_{j=1}^{i-1} B_{ij} - \gamma_n. \quad (9)$$

Sufficient conditions for convergence of the distribution of A_n to normality are given by a theorem due to Lindeberg [32]:

Lindeberg's theorem. *The central limit theorem applies when for any $\epsilon > 0$ the truncated variables U_{ij} defined by*

$$U_{ij} = \beta_i \left|B_{ij} - p_{ij}\right| \text{ if } \beta_i \left|B_{ij} - p_{ij}\right| \le \epsilon S_n \text{ and } U_{ij} = 0 \text{ otherwise} \quad (10)$$

are such that when $n \to \infty$

$$S_n \to \infty \text{ and } \frac{1}{S_n^2} \sum_{(ij)} E\left(U_{ij}^2\right) \to 1. \quad (11)$$

Note that the rigorous formulation of (10) is with $\left|\beta_i B_{ij} - E\left(\beta_i B_{ij}\right)\right|$. We have simplified given that $\beta_i > 0$. If for any $\epsilon > 0$, S_n increases fast enough with n so that $U_{ij} = \beta_i \left|B_{ij} - p_{ij}\right|$, then (11) is satisfied. We now show that this is the case when there is no degree correlation in L_N. Approximating σ_i^2 with (6) and $\sigma_{a_i}^2$ with $i \langle k \rangle / N$ gives

$$\beta_i \simeq \rho \sum_{j=i}^{n} \frac{1}{j},\; \rho = \sqrt{2N / \langle k \rangle}, \quad S_n^2 \simeq 2 \sum_{i=2}^{n} i \left(\sum_{j=i}^{n} \frac{1}{j}\right)^2. \quad (12)$$

We then notice that $S_n^2 \ge 4\left(h_n - 1\right)^2$, where h_n is the harmonic series. Given that h_n diverges, the left part of (11) is satisfied. To find a lower bound of S_n^2 we use

$$i \sum_{j=i}^{n} \frac{1}{j} \ge i \sum_{j=i}^{n} \frac{1}{n} \ge f(i) \text{ with } f(i) = i\left(1 - \frac{i}{n}\right).$$

Studying f yields a lower bound of S_n^2 and there exists a trivial upper bound for U_{ij}.

$$S_n^2 \ge 2 \sum_{n/4}^{3n/4} \frac{1}{i} f^2(i) \ge 2 \sum_{n/4}^{3n/4} \frac{4}{3n} \frac{9n^2}{16^2} \ge \frac{3n^2}{64}; \quad U_{ij} \le \beta_i \le \rho\left(1 + \ln(n)\right). \quad (13)$$

Let $\epsilon > 0$ be given. Since $x/\ln(x) \to \infty$ as $x \to \infty$, there exists n such that $n \geq 8\rho(1+\ln(n))/\epsilon\sqrt{3}$ and $\epsilon S_n \geq \beta_i |B_{ij} - p_{ij}|$ so that the variables U_{ij} and $\beta_i |B_{ij} - p_{ij}|$ are identical and (11) is satisfied. Since the Lindeberg theorem applies to $A_n + \gamma_n$, the asymptotic distribution of A_n is normal.

In our proof we considered that σ_i^2 scaled as i^2 and $\sigma_{a_i}^2$ as i, based on the assumption of no correlation between rank and degree in L_N. The values of S_n predicted by such approximations (12) are displayed by the line of Fig. 3(a). Squares corresponds to the exact S_n values (8) averaged over 10^3 random vertex sequences. The fit is excellent. However, ordering vertices by ascending degrees yields smaller S_n values (circles), and ordering by descending degrees gives larger values (triangles). Therefore, we numerically checked convergence to normality for different sequences of degrees. Figure 3(b) shows that the L_1 distance d_1 between the normal distribution and the distribution of A_n/S_n (sampled with 10^3 realizations of the RGGED) decreases as n increases. The speed of convergence depends of the sequence of degrees. Yet, in all cases, the Gaussian approximation is reasonable when $n > 5,000$. The protein network we use in this paper has $N = 7,302$ vertices. Therefore, whatever the sequence of degrees is, A_N has normal distribution under the RGGED. Also, note that the computation of A_N and S_N can be performed in $O(M)$ via an algorithm similar that of Sect. 2.2.

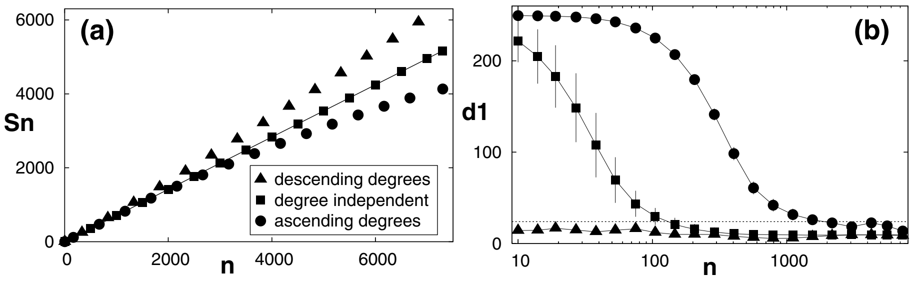


Fig. 3. (a) Standard deviation S_n of the Z profile area as a function of the profile length n. (b) Convergence of A_n/S_n to a normal variable under the RGGED: values of the L_1 distance to normality, averaged over 10^2 protein orders. Values above the horizontal line have probability less than 0.01 with a normal variable.

The normality of A_N under the RGGED is a useful property. It allows us to compare connectedness profiles by their values of A_N/S_N, this quantity being appropriately normalized for the sequence of degrees. We indicate in the next section that this property carries over to the permutation control.

3.2 Distribution Under the Permutation Control

Figure 4(a) shows the distribution of A_N/S_N across 10^4 random L_N sequences. It is Gaussian, its mean is slightly shifted from zero (+0.07) and its standard deviation is smaller (0.41) than that expected in the RGGED. The difference of means is due to λ_N being smaller than M: by half the sum of p_{ii} over all vertices. When we compute λ_N, we do not consider self edges $(v_i v_i)$ but the formula $p_{ij} = k_i k_j / 2M$ does. The difference of variances seen in panel (a) of Fig. 4 is explained by panel (b). In the RGGED the parameter σ_i increases linearly with i. In the permutation control the standard deviation sd_i of $\mathbf{x}_i$ is a non-monotonic function of i, because the total number of edges is constrained to M. Using the values of sd_i instead of σ_i in equation (8) gives the value $S_N = 0.41$ observed with the permutation control. As shown by Fig. 4(a), such difference of variances implies that the p-value for A_N/S_N under the RGGED is a conservative estimation of the p-value under the permutation control.

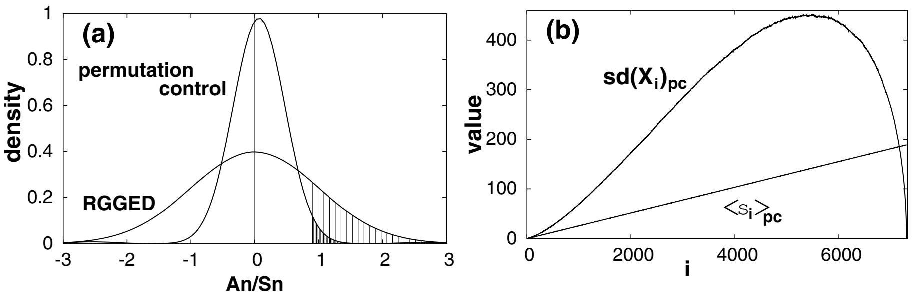


Fig. 4. (a) Distributions of the Z-profile area under the RGGED and the permutation control. (b) Standard deviation of x_i under the permutation control and average σ_i.

We now show that this property is preserved when the permutation control is no longer independent of vertex degree. Table 2 gives the mean and the standard deviation of A_N/S_N for different types of random protein orders. All values were estimated with 10^4 L_N sequences. When the protein order is correlated to degrees, the mean of A_N/S_N significantly increases. Rather than indicating a problem in the way A_N/S_N is normalized for degrees, the shift of the distribution points out a property of the network known as positive degree mixing [16]: vertices with similar degrees tend to be adjacent. To show that degree mixing accounts for the observed shift, we rewired the network with the so-called switching algorithm [4]. This destroys mixing while preserving degrees. Table 2 shows that both means and standard deviations of A_N/S_N go back towards zero. Therefore, it is reasonable to state that the p-value for A_N/S_N under the RGGED is a conservative approximation for the p-value under the permutation control, whatever the sequence of degrees is. Degree mixing has no meaningful impact on the analysis of experimental data, as long as the knowledge of the degrees is not used to order the proteins.

3.3 Application to Gene Expression Data

As an example, we consider two groups of tumors: 173 multiple myeloma samples that are associated with bone lesions and 36 that are not [31]. The values of the profile areas obtained for different expression scores are given in Table 3. The

Table 2. Mean (standard deviation) of A_N/S_N for different types of random protein orders

order	degree independent	ascending degrees	descending degrees
original network	0.07 (0.41)	42.30 (0.42)	11.96 (0.01)
rewired network	0.07 (0.41)	0.48 (0.25)	0.06 (0.005)

best mapping to the network is obtained when ranking genes for their higher expression in the samples associated with bone lesions ($A_N/S_N = 9.15$). This corresponds to a p-value of 10^{-20} in the RGGED, and thus to an even smaller p-value under the permutation control. A more informative p-value can be estimated by comparing the profile area to those obtained with random groups. Table 3 shows that only 0.2% of 10^4 random partitions yield a mapping to the network better than that obtained with the original groups. Therefore, we can be confident that genes having higher expression in the tumors associated with bone lesions are significantly close in the protein network. The mapping to the network less significant for genes having lower expression in samples with bone lesions ($p = 0.015$) and an intermediate p-value is obtained when ordering genes by their fold change ($p = 0.005$). Also, genes which are not differentially expressed between the two groups (ascending values of $|D|$) are not as well organized inside the network ($p = 0.02$). The pathways they map to are less likely to be related to the presence or absence of bone lesions.

Table 3. Differential expression associated with bone-lesion inducing tumors is pathway coherent

	Bone lesions / no lesions				Random groups				
Expression score	D		$\lvert D\rvert$		D		$\lvert D\rvert$		Expression score
Order	desc.	asc.	desc.	asc.	desc.	asc.	desc.	asc.	Order
A_N/S_N value	9.15	6.8	2.27	1.27	2.37	2.37	0.45	0.30	Mean
p-value	0.002	0.015	0.005	0.021	1.49	1.49	0.53	0.45	S.D.

4 Area of a Weighted Profile

As mentioned above, connectedness profiles can be compared by the degree-robust quantity A_N/S_N. For instance, in Fig. 3(a) the profile for the appendix samples has a larger area (11.8) than the profile obtained with the adipocytes (8.0). Yet, one might find the latter more interesting because the best P_L value corresponds to fewer proteins. Such difference can be quantified by making a weighted profile $\mathsf{w}_i Z_i$, where large indices i are deemphasized, and using the sum Aw_N of the weighted variables. Since the standard deviation σ_i of X_i scales as i, a convenient weighting scheme is

$$Aw_N = \sum_{i=1}^{N} \mathsf{w}_i Z_i, \quad \text{with} \quad \mathsf{w}_i = i^{-\theta} \quad \text{and} \quad 0 < \theta < 1. \tag{14}$$

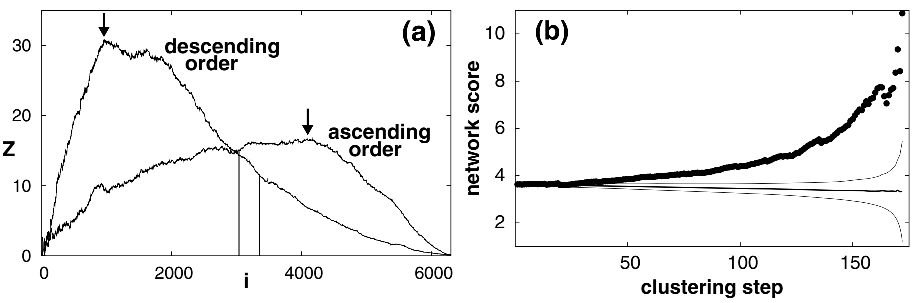


Fig. 5. (a) Connectedness profiles based on differential expression D in promyelocytic leukemia compared to 78 other tissues. Vertical lines indicate the points at which $D = 0$, and arrows the best values of connectedness Z. (b) Evolution of a network-mapping score based on Aw_N/Sw_N during the clustering of tumor samples. A high score indicates a partition into pathway-coherent groups. Lines show the average values ($\pm$ one standard deviation) over 10^3 runs of random clustering.

All the results obtained for the non-weighted profile carry over to the weighted one. Weighted profiles can be compared by their values of Aw_N/Sw_N, where Sw_N is the standard deviation of Aw_N in the RGGED. For instance, with weight parameter $\theta = 0.5$, Aw_N is larger (13.3) for the adipocyte profile of Fig. 3(a) than it is for the appendix profile (11.9).

Weighting a connectedness profile can also avoid paradoxical results in the case of extreme profiles such as those displayed in Fig. 5(a). Two profiles were obtained by ordering genes for their higher or lower expression in samples of promyelocytic leukemia compared to 78 other tissues [24]. Ordering genes by descending values of the expression score D (3) gives a Z profile reaching its maximum before D becomes negative (vertical line). On the contrary, the reverse order yields a maximal Z value only after D changes sign. So, up-regulated genes largely contribute to the profile area A_N intended to characterize down-regulated genes. Such problem can be avoided by using Aw_N instead of A_N.

Comparing the profile area of a protein order to that of the reverse order is required for the last example of application we consider. Clustering of microarray experiments is a widely used technique to uncover groups of samples based on gene expression. One important question is whether these groups are associated with known pathways. Here, we clustered 173 multiple myeloma samples [31] via a greedy agglomerative algorithm. The distance between samples is based on the Pearson correlation coefficient and the clustering output is a binary tree. Samples were grouped together because some genes have higher or lower expression in these samples versus the others. So, each cluster defines a differential expression score D and we can estimate how well it maps to the protein network with the largest value $\max A$ of Aw_N/Sw_N ($\theta = 0.5$), when genes are ordered by descending or ascending D values. Each clustering step gives a partition of the samples and an average score $\langle \max A \rangle$. Figure 5(b) shows that $\langle \max A \rangle$ increases during the clustering. For comparison, $\langle \max A \rangle$ tends to decrease in average across 10^3 runs of random clustering. Therefore, the clusters obtained with the greedy algorithm might be associated with certain functional modules. These groups of samples do not significantly overlap with the two categories defined by the presence or absence of bone lesions. So, the clustering gave other pathway-coherent partitions of the tumor samples.

5 Conclusion

In this paper we have presented degree-robust statistics of connectedness that are useful for working with protein networks. Robustness to degrees is important, since individual proteins can have very different numbers of interactions. The computation of our statistics is fast, in contrast to the commonly utilized approach of rewiring edges to control for degrees. Hence, this work should open new possibilities in the computational biology of protein networks.

Even though our connectedness measures were defined via a rather abstract model of random graph, they have simple interpretations. The probability P_L is a good approximation of the p-value for higher connectedness when selecting $|L|$ proteins at random. The p-value of A_N/S_N in the RGGED approximates the p-value for a stronger mapping between gene expression score and interaction in the network when randomly ordering genes.

We have presented a few examples of application to the analysis of gene expression data. Clearly, there are more potential use cases and they can be based on other gene attributes than expression scores. For instance, one could order proteins for the match of their sequence to a given amino-acid motif. An optimization algorithm based on Aw_N/Sw_N could then identify motifs that relate to protein interaction. Finally, the degree-robust yet fast identification of vertex attributes that relate to adjacency in a graph could prove useful in general.

Acknowledgments

We thank Rich Ferrante, Stan Letovsky and Noah Pradines for their comments.

References

1. Bader, G., Betel, D., Hogue, C.: Bind: the biomolecular interaction network database. Nucleic Acids Res. **31** (2003) 248–250
2. Peri, S., Navarro, J., Amanchy, R., Kristiansen, T., Jonnalagadda, C., Surendranath, V., Niranjan, V., Muthusamy, B., Gandhi, T., Gronborg, M., Ibarrola, N., Deshpande, N., Shanker, K., Shivashankar, H., Rashmi, B., Ramya, M., Zhao, Z., Chandrika, K., Padma, N., Harsha, H., Yatish, A., Kavitha, M., Menezes, M., Choudhury, D., Suresh, S., Ghosh, N., Saravana, R., Chandran, S., Krishna, S., Joy, M., Anand, S., Madavan, V., Joseph, A., Wong, G., Schiemann, W., Constantinescu, S., Huang, L., Khosravi-Far, R., Steen, H., Tewari, M., Ghaffari, S., Blobe, G., Dang, C., Garcia, J., Pevsner, J., Jensen, O., Roepstorff, P., Deshpande, K., Chinnaiyan, A., Hamosh, A., Chakravarti, A., Pandey, A.: Development of human protein reference database as an initial platform for approaching systems biology in humans. Genome Res. **13** (2003) 2363–2371

3. Han, J., Dupuy, D., Bertin, N., Cusick, M., Vidal, M.: Effect of sampling on topology predictions of protein-protein interaction networks. Nat. Biotechnol. **23** (2005) 839–844
4. Maslov, S., Sneppen, K.: Specificity and stability in topology of protein networks. Science **296** (2002) 910–913
5. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science **298** (2002) 824–827
6. Sharan, R., Ideker, T., Kelley, B., Shamir, R., RM, K.: Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. J. Comp. Biol. **12**(6) (2005) 835–846
7. Koyutürk, M., Grama, A., Szpankowski, W.: Assessing significance of connectivity and conservation in protein interaction networks. In: RECOMB-06. LNBI (2006) 45–59
8. Itzkovitz, S., Milo, R., Kashtan, N., Ziv, G., Alon, U.: Subgraphs in random networks. Phys. Rev. E **68** (2003) 026127
9. Bender, E., Canfield, E.: The asymptotic number of labelled graphs with given degree sequences. J. Combin. Theory (A) **24** (1978) 296–307
10. Molloy, M., Reed, B.: The size of the giant component of a random graph with a given degree sequence. Comb. Prob. Comp. **7** (1998) 295–305
11. Newman, M., Strogatz, S., Watts, D.: Random graphs with arbitrary degree distributions and their applications. Phys. Rev. E **64** (2001) 026118
12. Park, J., Newman, M.: The statistical mechanics of networks. Phys. Rev. E **70** (2004) 066117
13. Chung, F., Lu, L.: The average distance in random graphs with given expected degrees. Proc. Natl. Acad. Sci. USA **99** (2002) 15879–15882
14. Pradines, J., Farutin, V., Rowley, S., Dančík, V.: Analyzing protein lists with large networks: edge-count probabilities in random graphs with given expected degrees. J. Comp. Biol. **12**(2) (2005) 113–128
15. Farutin, V., Robison, K., Lightcap, E., Dancik, V., Ruttenberg, A., Letovsky, S., Pradines, J.: Edge-count probabilities for the identification of local protein communities and their organization. Proteins **62**(3) (2006) 800–818
16. Newman, M.: Mixing patterns in networks. Phys. Rev. E **67** (2003) 026126
17. Barrett, T., Suzek, T., Troup, D., Wilhite, S., Ngau, W., Ledoux, P., Rudnev, D., Lash, A., Fujibuchi, W., Edgar, R.: Ncbi geo: mining millions of expression profiles–database and tools. Nucleic Acids Res. **33** (2005) D562–D566
18. Goto, S., Okuno, Y., Hattori, M., Nishioka, T., Kanehisa, M.: Ligand: database of chemical compounds and reactions in biological pathways. Nucleic Acids Res. **30**(1) (2002) 402–404
19. Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K., Itoh, M., Kawashima, S., Katayama, T., Araki, M., Hirakawa, M.: From genomics to chemical genomics: new developments in kegg. Nucleic Acids Res. **34** (2006) D354–D357
20. Hakimi, S.: On realizability of a set of integers as degrees of the vertices of a linear graph. J. Soc. Ind. Appl. Math. **10** (1962) 496–506
21. Soffer, S., Vazquez, A.: Clustering coefficient without degree correlations biases. Phys. Rev. E **71**(5 Pt 2) (2005) 057101
22. Le Cam, L.: An approximation theorem for the poisson binomial distribution. Pacif. J. Math. **10** (1960) 1181–1197
23. Kerstan, J.: Verallgemeinerung eines satzes von prochorow und le cam. Z Washrscheinlichkeitstheorie und Verw. Gebiete **2** (1964) 173–179

24. Su, A., Wiltshire, T., Batalov, S., Lapp, H., Ching, K., Block, D., Zhang, J., Soden, R., Hayakawa, M., Kreiman, G., Cooke, M., Walker, J., JB, H.: A gene atlas of the mouse and human protein-encoding transcriptomes. Proc. Natl. Acad. Sci. USA **101**(16) (2004) 6062–6067
25. Newman, M.: Fast algorithm for detecting community structure in networks. Phys. Rev. E **69** (2004) 066133
26. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Phys. Rev. E **70** (2004) 066111
27. Ideker, T., Ozier, O., Schwikowski, B., Siegel, A.: Discovering regulatory and signalling circuits in molecular interaction networks. Bioinformatics **18**(Suppl. 1) (2002) S233–S240
28. Pradines, J., Rudolph-Owen, L., Hunter, J., Leroy, P., Cary, M., Coopersmith, R., Dancik, V., Eltsefon, Y., Farutin, V., Leroy, C., Rees, J., Rose, D., Rowley, S., Ruttenberg, A., Wieghardt, P., Sander, C., Reich, C.: Detection of activity centers in cellular pathways using transcript profiling. J. Biopharm. Stat. **14** (2004) 1–21
29. Grigoriev, A.: A relationship between gene expression and protein interactions on the proteome scale: analysis of the bacteriophage t7 and the yeast saccharomyces cerevisiae. Nucleic Acids Res. **29**(17) (2001) 3513–3519
30. Jansen, R., Greenbaum, D., Gerstein, M.: Relating whole-genome expression data with protein-protein interactions. Genome Res. **12**(1) (2002) 37–46
31. Tian, E., Zhan, F., Walker, R., Rasmussen, E., Ma, Y., Barlogie, B., Shaughnessy, J.: The role of the wnt-signaling antagonist dkk1 in the development of osteolytic lesions in multiple myeloma. N. Engl. J. Med. **349**(26) (2003) 2483–2494
32. Feller, W.: XI. In: An introduction to probability theory and its applications. Volume I. John Wiley & Sons, New York (1970) 254–255

Multivariate Segmentation in the Analysis of Transcription Tiling Array Data

Antonio Piccolboni

Affymetrix, Inc., 6650 Vallejo St. Suite 100, Emeryville, CA 94608
antonio_piccolboni@affymetrix.com

Abstract. Tiling DNA microarrays extend current microarray technology by probing the non-repeat portion of a genome at regular intervals in an unbiased fashion. A fundamental problem in the analysis of these data is the detection of genomic regions that are differentially transcribed across multiple conditions. We propose a linear time algorithm based on segmentation techniques and linear modeling that can work at a user-selected false discovery rate. It also attains a four-fold sensitivity gain over the only competing algorithm when applied to a whole genome transcription data set spanning the embryonic development of *Drosophila melanogaster*.

1 Introduction

The ever increasing density of DNA microarrays, or simply arrays, has made possible the development of tiling arrays. Most commercially available arrays are designed using existing genome annotations, such as Refseq for expression or dbSNP for genotyping; specifically, a small set of probes is dedicated to measuring expression level of each gene or to detect a particular allele at one locus. Instead, tiling arrays interrogate a genome, or a region thereof, in an unbiased fashion by selecting probes at regular intervals, subject only to constraints such as synthesizability, good hybridization behavior and sequence specificity, which often imply the exclusion of repeat regions. Therefore, in combination with a variety of sample preparation techniques, tiling arrays are general purpose discovery tools [30] that have been successfully used to further our knowledge of the transcriptome [20,14,17,19], protein-DNA interactions [22], DNA modifications [9] and DNA replication [25]. Furthermore, large datasets based on tiling arrays are freely available to the scientific community [1,2] and more are being generated, for instance, in the framework of a large collaborative project known as ENCODE [35].

We assume that the input data consist of normalized intensity values for each position at which a probe is centered, for each of one or more conditions. The present paper won't leverage the availability of control probes known as mismatch probes as in [18] or an additional control experiment as in [13], but will rely on a small number of replicate experiments. The biological interpretation of the intensity depends on the specific experiment being conducted and can be related, for instance, to the level of gene expression or DNA enrichment. In the following we focus on the expression case, but some of the ideas developed could be useful for other applications, such as protein-DNA interactions and DNA modifications. The goal is to identify genomic regions that are differentially expressed under different experimental conditions and to provide some statistically meaningful estimate of the detection error. We chose to focus on this problem for three reasons: first, differential expression (DE) is a first indicator of biological function; second, the data sets enabling this kind of analysis are increasingly available; and third, we wish to improve on the only other available algorithm [18].

Applications of tiling arrays for the detection of sites of DNA-protein interaction (CHIP^2) or the characterization of karyotypic modifications (array-CGH) have received increasing attention in the statistical and computational literature [38,23,27,26,28], whereas transcription has received comparatively less [13]. In particular, to our knowledge, only one work [18] deals with the detection of differentially expressed regions. In that paper, a non parametric test (Kruskal-Wallis) is applied to each probe, returning a p-value that the (ranks of the) data could have been observed under the null hypothesis of no condition effects. These p-values are thresholded and then runs of probes exceeding the threshold are simplified by filling in small gaps and filtering out short runs. A *false discovery rate* (FDR) [8] is provided using a permutation technique.

Taking a wider look at the tiling array analysis literature we can try to classify different methods according to two important characteristics that can be described as *local modeling* and *data aggregation*. The first is concerned with the relation between observed intensity and underlying variables such as expression level or probe affinity and the noise model. While occasionally non-parametric methods have been used, thus avoiding strong assumptions about the underlying model, more often simple linear models have proved useful. One trend is towards the use of aggregate information from the whole data set to estimate parameters of interest for each local model, sometimes using Bayesian techniques [26] and including sequence information [27], with the aim of making such estimates less noisy and lessen or even eliminate the need for replicate experiments. The second characteristic defines how probes are grouped to apply local modeling in the absence of predefined probe sets, and how local models are aggregated to produce the final output, in the form of a collection of regions displaying differential expression or sites of DNA enrichment. Four main approaches can be identified:

Sliding window. The local statistics are a function of all the data in a fixed-size window starting at a given probe position, for instance taking the pseudo-median of the signal or calculating the t-statistics for each probe and averaging them; this procedure is repeated for every window. Further processing is necessary to define output regions. These methods struggle to define the optimal window size: a narrow setting allows the detection of smaller features in the data, whereas a wide one increases the sensitivity for larger ones and helps reducing false positives which arise from isolated, aberrant signal. In the case of transcription, where the feature size is related to exon size and is extremely variable and where transcripts of interest, like small RNAs, may hybridize with only one probe due to their length, this issue seems particularly crucial. [14,23,26,28,16,38].

HMM. The local models are assumed independent conditional to the state of a suitable HMM, which could be as simple as a first order two state (expressed vs. non-expressed) HMM or have a high number of states and order; the regions are defined by self transitions in the expressed state or transitions within a subset of the states [26,37,31]. Adapting a finite state model to a phenomenon that is inherently continuous is difficult: the intensity observed for two consecutive probes within the same expressed exon is highly correlated, even conditional to the knowledge that both are hybridizing with some target, because the observed intensity is roughly proportional to the abundance of the target.

Long runs. The local statistics are thresholded and long, mostly uninterrupted runs are used to define the output regions [10,18]. This is used often in combination with the sliding window technique. Thresholding early in the process gives up useful quantitative information, with a potential loss in power.

Segmentation models. The local models are combined into a larger model that explicitly incorporates regions of expression and non expression; the regions of expression are obtained by simply estimating the model parameters [13,34]. The most important difference with HMMs is that the aforementioned independence assumption is no longer necessary; therefore, the model can more accurately capture the correlation between signals obtained from neighboring probes hybridizing to the same target and more specifically the dependance on the abundance of that target. This is the general approach followed in this paper.

Some methods let (force) the user to specify some relatively obscure parameters, whereas others try to estimate them from the data; occasionally, training data have also been incorporated [31]. Our goals were the following:

- improve the sensitivity and specificity w.r.t. available methods;
- let the user control the sensitivity/specificity trade off without having to set too many parameters whose correct value or even interpretation is obscure;
- provide an efficient algorithm that can handle the available large data sets.

In Sect. 2 we introduce our model and derive a method to detect regions of DE. In Sect. 3 we describe an efficient algorithm that implements it. In Sect. 4 we devise a method to estimate the error of this algorithm. In Sect. 5 we report the results of running the algorithm on real and simulated data. Finally, in Sect. 6 we point to directions for future research stemming from this work.

2 The Method

Let's denote with $i \in [I]$ a condition, with $j \in [J]$ a probe interrogating a given position and with $k \in [K]$ a replicate experiment. Let y_{ijk} be the set of observed normalized log intensities. The log transformation is often recommended since the dominant noise component is multiplicative [21]. What we want to compute is a set of non overlapping regions of DE $R = \{(b_h, e_h)\}$ with $b_h, e_h \in [J]$ and $b_h < e_h \forall h \in [H]$, and $e_h < b_{h+1} \forall h \in [H-1]$. H is the total number of regions and is part of the output. That is, the output is a set of disjoint intervals on the integer numbers, where the boundaries correspond to probe positions. In analogy with [14] we will refer to such intervals as *diffrags*.

We will first focus on modeling a run of consecutive probes within one diffrag. We make the assumption that they are hybridizing with the same target or multiple targets in the case of transcript isoforms. Let the log of the total concentration of these targets be denoted with x_{ih}. It is well known that the observed intensity is dependent on the concentration of the target but also on the propensity of each probe to hybridize, or *affinity*. The affinity is modeled by a single parameter a_j. With these definitions, a simple model is the following [21]:

$$y_{ijk} = x_{ih} + a_j + \varepsilon_{ijk}, \quad \varepsilon \sim N(0, \sigma_h). \tag{1}$$

In the case of no DE, $x_{ih} = x_h$ and, without loss of generality, $x_h = 0$ (the common abundance can be absorbed into the corresponding a_j's). Let L_h be the log-likelihood function according to Eq. 1 and L_h^0 be the corresponding function under the hypothesis of no DE. Under the assumption of independence between non-overlapping regions, the log-likelihood for all the regions in R is as follows:

$$\mathcal{L}_J = \sum_{h \in R} L_h + \sum_{h \in \bar{R}} L_h^0, \tag{2}$$

where $\bar{R}$ is the set of regions complementary to R or $\{(e_h + 1, b_{h+1} - 1)\}_h \cup \{(0, b_0)\}$. In all experiments reported in this paper we set the minimum size of each region to be two probes. An equivalent expression, up to an additive constant, that will be useful in the following, is:

$$\sum_{h \in R} (L_h - L_h^0). \tag{3}$$

Following a maximum likelihood approach, one could think that finding $\max_{R,\boldsymbol{x},\boldsymbol{a}} L$ is the correct way of fitting this model, but in this case we are selecting among models with different complexity — measured, for instance, by the number of free parameters — and therefore the maximum likelihood criterion alone is not appropriate. It is straightforward to verify that $\mathcal{L}_J$ in Eq. 3 is trivially maximized by defining as many regions as there are probes. A number of criteria have been proposed to control the complexity of the optimal solution, for instance in applications to genomic data [29] and to array-CGH data [33]. The different criteria amount to subtracting a *penalty term* from the likelihood function that is larger for more complex solutions. Here we follow a suggestion of Broman et al. [12], according to which introducing a penalty term is "approximately equivalent" to thresholding the LOD score. Each term in the Eq. 3 sum is equivalent to fitting a two-way ANOVA model to each region and computing the appropriate F-statistics for the hypothesis of equal target concentration in each condition. Therefore it seems reasonable to penalize it with a quantile of the F-distribution with degrees of freedom determined by the number of conditions and replicates and the size of region h, as in the following equation:

$$\mathcal{L}_J = \sum_{h \in R} \left(L_h - L_h^0 - (I-1)\mathrm{qf}(p, d_1, d_{2h}) \right), \tag{4}$$

where $\mathrm{qf}(\cdot,\cdot,\cdot)$ is the quantile function for the F-distribution, p is a probability and $d_1 = I - 1$ and $d_{2h} = (I-1)(e_h - b_h)K$ are degrees of freedom. One consequence of using this penalty term is that no region h for which the LOD score is smaller than the penalty can be part of an optimal solution. Therefore specifying a probability p implies that applying the F-test for the hypothesis of unequal means to all regions in an optimal solution would result in rejecting the null hypothesis (equal means) with p-value p or better. The converse is not true as two overlapping regions can not be simultaneously part of a solution. Since this is a very simplified model, we make no claim that these p-values are accurate and we won't rely on them to estimate error rates. In the next section we will assume p is supplied by the user but we will return to this issue in Sect. 4.

Any penalty function that can be expressed as a sum of terms that are only dependent on each DE region is compatible with the general structure of this algorithm. Of two popular criteria, AIC and BIC, only the former can be rewritten this way but, since it isn't parametrized, it doesn't offer the flexibility of the one adopted in the present work. An alternative to applying a heuristic penalty term is to define a prior on all possible models and pick the model with the largest posterior distribution given the data. While determining priors that are both realistic and analytically tractable is a challenge, we believe it is an approach worth further consideration.

3 The Algorithm

In this section we discuss how to efficiently maximize $\mathcal{L}_J$, as defined in Eq. 4, with respect to the set of regions R, the expression parameters $\boldsymbol{x}$ and the affinity parameters $\boldsymbol{a}$. Once the regions of DE are known, the maximum likelihood estimates for the remaining parameters are as follows:

$$\hat{a}_j = \frac{1}{IK} \sum_{ik} y_{ijk} \tag{5}$$

$$\hat{x}_{ih} = \frac{1}{(b_h - e_h)K} \sum_{j \in (b_h, e_h), k} (y_{ijk} - a_j), \tag{6}$$

subject to the constraints $\sum_i x_{ih} = 0$. As to optimizing over the set of regions, exhaustive search is clearly not feasible, as the set of all subsets of $[J]$ is exponential in J. Even introducing an upper bound W on the region size, a reasonable assumption we will rely on in the following, the search space size is still $\Omega(W^{\frac{J}{W}})$. An optimal solution can be found efficiently via dynamic programming. The general idea can be traced back to [7], where the problem was to approximate a sequence with a set number of line segments. Here the sequence takes values in $\mathcal{R}^{IK}$ and the approximation is evaluated according to a simplified model of hybridization. We need to reformulate the objective in Eq. 4 as a recurrence relation as follows:

$$\mathcal{L}_0 = 0 \tag{7}$$

$$\mathcal{L}_t = \max\left(\max_{w \leq W}\left(\mathcal{L}_{t-w} + L_h - L_h^0 - \mathrm{qf}(p, d_1, d_{2h})\right), \mathcal{L}_{t-1}\right), \tag{8}$$

where $h = (t-w, t)$. The algorithm computes $\mathcal{L}_t$ for $0 < t \leq J$ in ascending order and stores the previous W values and records the argmax information for both the inner and outer max operations. This requires JW steps.

We next show how to evaluate $L_h - L_h^0$ in constant time w.r.t. the size of region h to reduce the complexity of the algorithm to $O(JWIK)$. By simple algebraic manipulations $L_h - L_h^0$ can be shown to be equal to

$$\frac{\sum_i x_{ih}^2}{\hat{\sigma}_h^2},$$

where

$$\hat{\sigma}_h^2 = \frac{1}{I(e_h - b_h)} \sum_{i,j \in (b_h, e_h)} (y_{ijk} - \bar{y}_{ij})^2$$

and

$$\bar{y}_{ij} = \frac{1}{K} \sum_k y_{ijk}.$$

The direct evaluation of these expressions requires W steps, bringing the overall complexity of the algorithm to $\Theta(JW^2IK)$. Since there are tiling arrays where the density of interrogation is as small as 5 base pairs, in organisms such as mouse or *homo sapiens* that means that one exon can contain several tens of probes and that makes a quadratic run time in W less than ideal. A better solution is to precompute, for each i, the cumulative sums w.r.t. j of

$$X_{ij} = \sum_k x_{ijk},\ j \in [J]$$

and

$$S_j = \frac{\sum_{ik}(y_{ijk} - \bar{y}_{ih\cdot})^2}{IK}.$$

Let us denote them with $\mathcal{X}_{ij}$ and $\mathcal{S}_j$, resp. Then

$$x_{ih} = \frac{\mathcal{X}_{ie_h} - \mathcal{X}_{i,b_h-1}}{e_h - b_h}$$

and

$$\hat{\sigma}_h^2 = \frac{\mathcal{S}_{e_h} - \mathcal{S}_{b_h-1}}{(e_h - b_h)}.$$

Computing the cumulative sums takes linear time in J and is independent from W. Once those are available, evaluating $L_h - L_h^0$ takes only $O(IK)$. This can be seen as a repeated application of the cumulative sums technique which is well known and used also in the tiling array context in the single sample, homoscedastic case [13]. One final speed up is achieved by storing previously computed quantiles of the F-distribution, since one needs to evaluate only W of them, as W affects the second degree of freedom. Amortized over tens of millions of likelihood evaluations, this cost is negligible.

4 Error Estimation

In Sect. 2 we delayed the problem of selecting the parameter p. As stated in Sect. 1 our goal was to provide a way for the user to control the sensitivity/specificity trade-off without having to set parameters that have a difficult interpretation or for which an appropriate value is not obvious. We selected a commonly used error measure, the False Discovery Rate (FDR, [8]), defined as the expected ratio of false positives to positives. In this application, a positive is a probe falling within a diffrag. The user, in light of application-specific considerations, will elect to obtain a more comprehensive list of differentially expressed regions with a higher FDR or a shorter one at a lower FDR. We will first show how to estimate the FDR for a given p and then how to calculate the value of p yielding the desired FDR.

In analogy with the widely used SAM algorithm [36] for the detection of differentially expressed genes from expression array data, we used a permutation technique to estimate the FDR. The algorithm is as follows. We first run the algorithm on the original data. For each diffrag, we subtract from the data the condition effects, that is we compute $\tilde{y}_{ijk} = y_{ijk} - x_{ih}$ for all $j \in (b_h, e_h)$ and $h \in R$, $\tilde{y}_{ijk} = y_{ijk}$ otherwise. Then the condition labels are permuted, that is $\tilde{y}^P_{if(j,k)g(j,k)} = \tilde{y}_{ijk}$ where $f(j,k) = P_{jK+k} \div K$, $g(j,k) = P_{jK+k} \bmod K$ and P is a permutation of $[JK]$ selected uniformly at random. The algorithm is run on the permuted data and all positives are considered false positives. A new permutation is selected and a new false positive estimate is obtained, and this is repeated a number of times — 20 in the experiments reported below. We will report also the estimate standard deviation, but establishing a confidence interval for the FDR would be ideal.

We are left with the problem of selecting p so as to match a target FDR. Lacking an analytical way of linking the two and leveraging the speed of this algorithm and its implementation, we simply performed binary search on $\log p$. While there is a monotonic relationship between p and the number of diffrags, this is not guaranteed between p and the number of positive probes, the unit chosen to measure error rates, since the length of the detected regions can increase when p is decreasing. In practice, though, both relations appear to be monotonic and binary search always terminates within 1% of the target.

An implementation of this algorithm is available under the terms of the GNU General Public License, version 2 [3].

5 Experimental Results

Evaluation of data analysis methods for tiling array data is *per se* a challenge. Few control data sets are available, and none relevant to the detection of differentially transcribed regions to the best of our knowledge. There isn't a consensus on what model to use to generate simulated data, and the variety of organisms, tissues, conditions and assays that can underlie the input to this kind of analysis is daunting. Giving up any pretense of exhaustivity, we combined different simulations and the comparison with a competing method [18] on a real data set to evaluate the algorithm's performance. In particular, we wanted to show that this algorithm has high specificity and sensitivity which do not depend on strict compliance with the model and that the error estimates provided are reasonably accurate. This doesn't replace the need for benchmarks like the ones available for standard expression arrays [11] or, more recently, CHIP^2 type applications of tiling arrays [4].

Since we will later consider some results based on the organism *Drosophila melanogaster*, we took exons from the initial part of chromosome X corresponding to the first 10^5 probes for that chromosome in the tiling array design used in [18]. We simulated affinities, expression levels and error terms randomly and independently from the normal distribution for each exon (if two exons overlap, it was done independently for the overlapping and non overlapping parts). For consistency with the real data set considered later, we set the number of conditions to 12 and the number of replicates to 3. In one simulation the expression levels have standard deviation equal to the simulated error terms, in the other it is 10 times larger. In intronic and intergenic regions the simulation is similar but with expression levels clamped to some fixed value — this is irrelevant to the algorithm. We also run the algorithm with two values of p, bypassing the normal procedure of selecting a FDR first and letting the algorithm pick p. All the rates in this section are based on probe counts and a probe is considered positive if and only if it falls within a region detected as differentially expressed.

Table 1 shows that the algorithm has high sensitivity and specificity even on the more noisy dataset and using less stringent parameter settings only seems to degrade specificity. The FDR estimate is off by slightly more than 1% in the worst case.

A most important simplifying assumption of our model is that it assumes expression levels associated with different exons to be independent, but, for exons

Table 1. Performance on ideal simulation. From the left: standard deviation of DE levels, p, estimated FDR, standard deviation of FDR in permutation, the same divided by the square root of the number of permutations, real FDR, difference between estimated and real FDR and true positive rate.

DE std	p	Est.FDR	FDR std	FDR std/$\sqrt{n}$	Real FDR	Δ FDR	true positive
10	10^{-3}	0.088	0.006	0.0014	0.082	-0.0065	0.992
10	10^{-4}	0.016	0.002	0.0005	0.022	0.0055	0.992
1	10^{-3}	0.096	0.006	0.0014	0.083	-0.0133	0.982
1	10^{-4}	0.018	0.002	0.0004	0.024	0.0061	0.980

Table 2. Performance on simulation with dependencies. See Table 1 caption for detailed explanations.

DE std	p	Est.FDR	FDR std	FDR std/$\sqrt{n}$	Real FDR	Δ FDR	true positive
10	10^{-3}	0.092	0.006	0.0014	0.075	-0.0167	0.992
10	10^{-4}	0.017	0.002	0.0004	0.020	0.003	0.992
1	10^{-3}	0.096	0.008	0.0019	0.081	-0.015	0.982
1	10^{-4}	0.019	0.003	0.0007	0.024	0.0053	0.981

Table 3. Performance on simulation with log-normal noise. See Table 1 caption for detailed explanations.

DE std	p	Est.FDR	FDR std	FDR std/$\sqrt{n}$	Real FDR	Δ FDR	true positive
10	10^{-3}	0.162	0.020	0.0046	0.098	-0.0640	0.957
10	10^{-4}	0.071	0.017	0.0039	0.052	-0.0189	0.956
1	10^{-3}	0.103	0.006	0.0013	0.099	-0.004	0.904
1	10^{-4}	0.029	0.004	0.0008	0.033	0.0042	0.899

belonging to the same transcript, this is clearly not the case. On the contrary, such exons have highly correlated expression levels, even if splice variation makes such correlation less then perfect. To study its effects on the performance of our algorithm we modified our simulation so that exons associated with the same set of transcripts would have the same expression levels, and again performed four runs with two parameter settings and two noise levels.

Overall (see Table 2), there are only modest changes in performance, in the true positive rate and in the real and estimated FDR. Visual inspection of the results suggests that in this simulation short introns in between identically expressed exons are likely to be erroneously detected, because detecting one large region including the intron and the two flanking exons is preferable according to the algorithm than calling two smaller regions. This observation points to the modeling of this kind of dependency as a promising research direction, not only with the goal of improved performance but also to infer some structural information implicit in these data [24,18].

For a third group of simulations we returned to the independent model, but sampled the residuals from a log-normal distribution with log-scale mean and standard deviation of -0.75 and 1 resp., which implies a standard deviation of roughly 1. This was to shed light on the resilience of the algorithm to deviations from the parametric assumptions underlying it. In Table 3 we observe only a slight degradation in performance both in the sensitivity/specificity trade-off and the ability to accurately estimate the FDR.

Let's turn our attention to a real data set related to the early development of *Drosophila melanogaster* [18]. Total RNA greater than 200 nucleotides was collected at 2 hour time intervals for a total of 24 hours. Microarray hybridizations were performed for three replicates using the same biological sample; therefore, no sampling of biological variability is available. The tiling microarrays used in

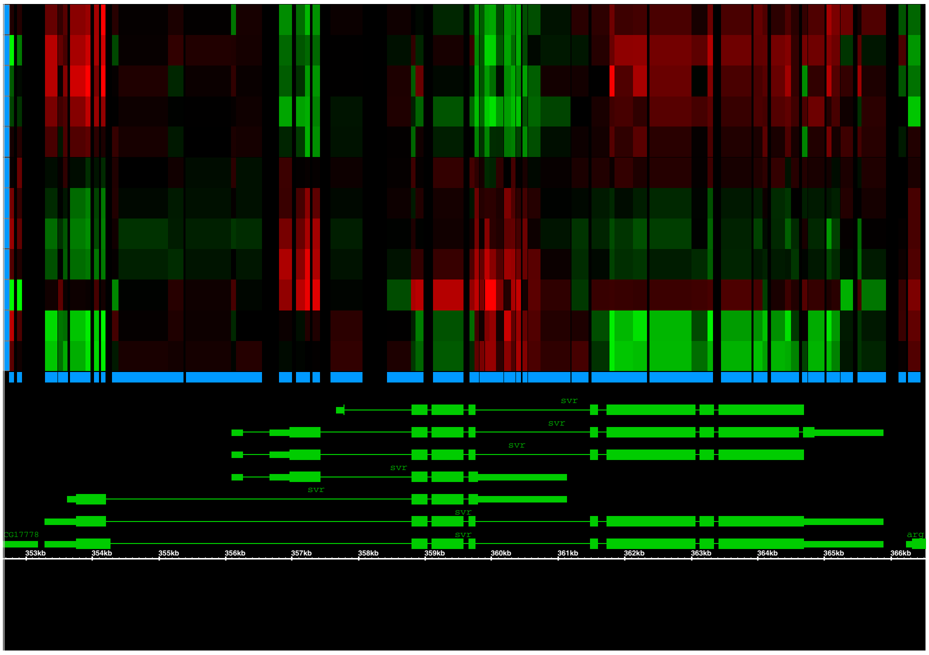


Fig. 1. A screenshot from the Integrated Genome Browser [15] showing the genomic region of gene *svr*. From top to bottom: a heat map of DE levels, for each diffrag, across 12 consecutive time points of the embryonic development of *Drosophila*; a track (blue) representing detected diffrags; and a track (green) representing annotated genes [6]. The heat-map is organized in increment of 2 hours, with the earliest developmental stages at the bottom, and over- or under-expression, relative to the average signal, for each diffrag, over the whole time course, are represented in green and red resp.

that experiment interrogate, in an unbiased fashion, the non-repeat portion of the *Drosophila* genome every 35 bp on average, for a total of slightly more than 3×10^6 interrogation positions. The data can be obtained on line [5] and were quantile normalized [11].

A typical run of the algorithm, implemented in C++, for fixed p, $W = 20$ and 20 permutations takes less than 35 minutes of CPU time and slightly more than 3GB of RAM on an single 2.2 GHz Opteron processor. The core dynamic programming algorithm runs in about 100 seconds for each permutation after reading the input.

In [18] 27.6% of the non-repeat portion of the genome is reported as transcribed at some point during development, either differentially or constitutively, at a false positive rate of 5%, equivalent to an FDR of 18%, on each time point analyzed independently. At the 3.19% FDR, 13.8% of all interrogation positions – roughly equivalent to 13.8% of non-repeat sequence – is found to be differentially expressed. At the closely matched FDR of 3.21%, our algorithm reports 50.3% of differentially expressed interrogation positions, representing almost a four-fold improvement in positive rate, and, if the FDR estimates are accurate, in sensitivity. This is not too surprising since the algorithm used in [18] uses a non-parametric test and doesn't pool nearby probes together, which, together with a small number of replicates, results in low power. On the other hand, that approach sidesteps most concerns about modeling of the data, which we tried to address with simulations. Even if our algorithm can not detect transcripts that are expressed but not differentially expressed, we are still able to detect as transcribed during development almost twice as much sequence at a much more stringent FDR.

Diffrags and their expression patterns can be conveniently visualized in the Integrated Genome Browser [15] as shown in Fig. 1. Several isoforms of the silver gene (*svr*) are clearly differentially expressed across embryonic development. It appears that a long isoform is up-regulated for the first two time points whereas a short form is up-regulated towards the end of embryonic development. Introns are also detected albeit with weaker DE.

We now turn to some preliminary analysis of the results. A more thorough analysis will be the subject of a follow up paper. Following [18] we broke down diffrags by their relative position to annotated genes; that is, whether they overlap exons, introns or neither (*intergenic*). Diffrags that cannot be unequivocally assigned to one category are split according to their overlap and the summary statistics are reported in base pairs (bp). Of 5.78×10^7 differentially expressed bps, 38% are classified as exonic and 38% as intronic, with the remaining 25% being intergenic. While the comparison with the work of Manak et al.[18] is difficult because of different goals and error rates, it is interesting to observe that the gain in sensitivity is confined to intronic (three-fold) and intergenic (7-fold) regions, with only a modest gain in exonic regions. A possible explanation is that intronic and intergenic expression occurs at lower levels [14].

We also looked at the global patterns of DE, measured as log-scale difference w.r.t. the mean, for each diffrag, over the 12 time points and summarized with the same number of box-plots in Fig. 2.

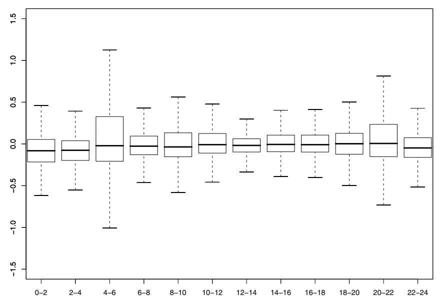


Fig. 2. Log-scale DE pattern distributions

The main observation that can be drawn is that at 4–6h and 20–22h there is a greater variability of expression. To the best of our knowledge, this has not been reported previously. Those two time points coincide with very important biological transitions: the onset of detectable zygotic transcription and the beginning of larval-specific transcription at the end of embryogenesis. An equivalent plot where each diffrag is weighted with its length (not shown) is very similar.

We then performed a Principal Component Analysis on the patterns of DE. The first 4 components are plotted in Fig. 3. The first component represents an early/late expression dimension.

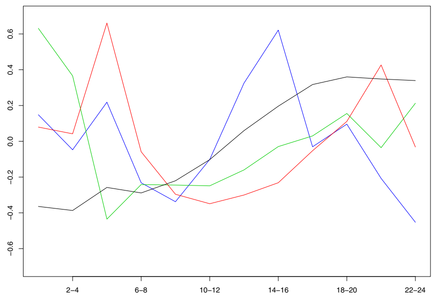


Fig. 3. First 4 principal components (colored in black, red, green and blue in order of decreasing variance) of the DE patterns

More interesting is the second, with two peaks at 4-6 and 20-22 hours, again the two time points we singled out before as being of particular biological significance. This points to a global change and reorganization of expression patterns as the organisms undergoes important biological transitions and deserves further study. The third principal component seems to decline as maternal transcripts are degraded in the first 4 hours and then constantly increase.

6 Discussion and Further Work

We developed an algorithm for the detection of regions of DE from tiling array data. The algorithm is based on a simplified but reasonable model of transcription and hybridization and generalizes classic segmentation techniques. It achieves a marked improvement in sensitivity and specificity over the only published alternative and can work at a target FDR, which appears to be accurate in simulations. Moreover, it is very efficient and can handle the largest available datasets. A preliminary analysis of the results of running the algorithm on a real dataset points to interesting biological observations worth further investigation.

While finding the extent of DE has been the main focus of this work, estimating the number and length of the diffrags is also an important goal. More accurately detecting diffrag boundaries is a first step towards characterizing the structure of novel exons and transcripts detected with tiling arrays. In [18] a clustering algorithm has been successfully employed to discover several transcript isoforms that had eluded previous annotation efforts. Dedicated algorithms that model splice variation and richer datasets that more deeply sample the diversity of the transcriptome promise to be a powerful addition to more established genome annotation methods [32].

Acknowledgments

I wish to thank Richard Bourgon for many insightful discussions, Gregg Helt and Ed Erwin for help with the IGB browser and the Transcriptome Group at Affymetrix for making the data available, in particular John Manak. Simon

Cawley and Earl Hubbell also provided useful feedback. Chuck Sugnet, Harley Gorrell, John Manak, Jim Veitch and Richard Bourgon reviewed early versions of this manuscript suggesting many improvements, as did on anonymous reviewer.

References

1. http://www.affymetrix.com/transcriptome.
2. http://www.ncbi.nlm.nih.gov/geo.
3. http://www.affymetrix.com/Auth/support/developer/downloads/Tools/seg-limo.zip.
4. http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE5053.
5. http://transcriptome.affymetrix.com/publication/drosophila_development/.
6. http://genome.ucsc.edu.
7. R. Bellman. On the approximation of curves by line segments using dynamic programming. *Communications of the ACM*, 4(6):284, 1961.
8. Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B. Methodological*, 57(1):289–300, 1995.
9. B.E. Bernstein, M. Kamal, K. Lindblad-Toh, S. Bekiranov, D.K. Bailey, D.J. Huebert, S. McMahon, E.K. Karlsson, E.J. Kulbokas, T.R. Gingeras, et al. Genomic Maps and Comparative Analysis of Histone Modifications in Human and Mouse. *Cell*, 120(2):169–181, 2005.
10. M. Bieda, X. Xu, M.A. Singer, R. Green, and P.J. Farnham. Unbiased location analysis of E 2 F 1-binding sites suggests a widespread role for E 2 F 1 in the human genome. *Genome Research*, 16(5):595, 2006.
11. B.M. Bolstad, R.A Irizarry, M. Åstrand, and T.P. Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, 2003.
12. K.W. Broman and T.P. Speed. A model selection approach for the identification of quantitative trait loci in experimental crosses. *Journal of the Royal Statistical Society, Series B*, 64(4):641–656, 2002.
13. Lior David, Wolfgang Huber, Marina Granovskaia, Joern Toedling, Curtis J Palm, Lee Bofkin, Ted Jones, Ronald W Davis, and Lars M Steinmetz. A high-resolution map of transcription in the yeast genome. *Proc Natl Acad Sci U S A*, 103(14):5320–5, 2006.
14. D. Kampa et al. Novel RNAs identified from an in-depth analysis of the transcriptome of human chromosomes 21 and 22. *Genome Research*, 14(3):331–342, 2004.
15. G. Helt et al. http://www.affymetrix.com/support/developer/tools/download igb.affx.
16. J. Castle et al. Optimization of oligonucleotide arrays and RNA amplification protocols for analysis of transcript structure and alternative splicing. *Genome Biology*, 4:R66, 2003.
17. J. Cheng et al. Transcriptional maps of 10 human chromosomes at 5-nucleotide resolution. *Science*, 307, 2005.
18. J.R. Manak et al. Biological function of unannotated transcription during the early development of Drosophila melanogaster. *Nature Genetics*, 38:1151–1158, 2006.
19. P. Bertone et al. Global identificaion of human transcribed sequences with genome tiling arrays. *Science*, 306:2242–2246, 2004.

20. P. Kapranov et al. Large-scale transcriptional activity in chromosomes 21 and 22. *Science*, 296:916–919, 2002.
21. R.A. Irizarry et al. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264, 2002.
22. S. Cawley et al. Unbiased mapping of transcription factor binding sites along human chromosomes 21 and 22 points to widespread regulation of non-coding rnas. *Cell*, 116(4):499–511, 2004.
23. S. Keles et al. Multiple testing methods for chip-chip high density oligonucleotide array data. Technical Report 147, U.C. Berkeley Division of Biostatistics, June 2004.
24. B.J. Frey, N. Mohammad, Q.D. Morris, W. Zhang, M.D. Robinson, S. Mnaimneh, R. Chang, Q. Pan, E. Sat, J. Rossant, et al. Genome-wide analysis of mouse transcripts using exon microarrays and factor graphs. *Nat Genet*, 37(9):991–6, 2005.
25. Y. Jeon, S. Bekiranov, N. Karnani, P. Kapranov, S. Ghosh, D. MacAlpine, C. Lee, D.S. Hwang, T.R. Gingeras, and A. Dutta. Temporal profile of replication of human chromosomes. *Proceedings of the National Academy of Sciences*, 102(18):6419–6424, 2005.
26. Hongkai Ji and Wing Hung Wong. TileMap: create chromosomal map of tiling array hybridizations. *Bioinformatics*, 21(18):3629–36, 2005.
27. W. Evan Johnson, Wei Li, Clifford A. Meyer, Raphael Gottardo, Jason S. Carroll, Myles Brown, and X Shirley Liu. Model-based analysis of tiling-arrays for ChIP-chip. *Proc Natl Acad Sci U S A*, 103(33):12457–62, 2006.
28. W. Li, C.A. Meyer, and X.S. Liu. A hidden Markov model for analyzing ChIP-chip experiments on genome tiling arrays and its application to p53 binding sequences. *Bioinformatics*, 21(1):i274–i282, 2005.
29. Wentian Li. Dna segmentation as a model selection process. In *RECOMB*, pages 204–210, 2001.
30. T.C. Mockler and J.R. Ecker. Applications of DNA tiling arrays for whole-genome analysis. *Genomics*, 85:1–15, 2005.
31. K. Munch, P.P. Gardner, P. Arctander, and A. Krogh. A hidden Markov model approach for determining expression from genomic tiling micro arrays. *BMC Bioinformatics*, 7(1):239, 2006.
32. Brian Oliver. Tiling dna microarrays for fly genome cartography. *Nature Genetics*, 38:1101–1102, October 2006.
33. F. Picard, S. Robin, M. Lavielle, C. Vaisse, and J.J. Daudin. A statistical approach for array CGH data analysis. *BMC Bioinformatics*, 6(1):27–27, 2005.
34. A. Piccolboni and N. Xu. An HSMM-based algorithm for espression detection in tiling DNA microarray data. In *Genome Informatics*, page 117, Cold Spring Harbor, New York, October 2005. Cold Spring Harbor Laboratory.
35. ENCODE project consortium. The ENCODE (ENCyclopedia of DNA elements) project. *Science*, 306:636–640, 2004.
36. J.D. Storey and R. Tibshirani. SAM thresholding and false discovery rates for detecting differential gene expression in DNA microarrays. *The Analysis of Gene Expression Data: Methods and Software*, 2003.
37. T. Toyoda and K. Shinozaki. Tiling array-driven elucidation of transcriptional structures based on maximum-likelihood and Markov models. *The Plant Journal*, 43(4):611, 2005.
38. H. Willenbrock, J. Fridlyand, and O. Journals. A comparison study: applying segmentation to array CGH data for downstream analyses. *Bioinformatics*, 21(22):4084–4091, 2005.

A Bayesian Model That Links Microarray mRNA Measurements to Mass Spectrometry Protein Measurements

Anitha Kannan[1], Andrew Emili[2], and Brendan J. Frey[3]

[1] Microsoft Research, Cambridge, UK
http://www.research.microsoft.com/∼ankannan
ankannan@microsoft.com
[2] Banting & Best Department of Medical Research, University of Toronto, Canada
http://emililab.med.utoronto.ca
andrew.emili@utoronto.ca
[3] Electrical & Computer Engineering, University of Toronto, Canada
http://www.psi.toronto.edu
frey@psi.toronto.edu

Abstract. An important problem in biology is to understand correspondences between mRNA microarray levels and mass spectrometry peptide counts. Recently, a compendium of mRNA expression levels and protein abundances were released for the entire genome of the laboratory mouse, *Mus musculus*. The availability of these two data sets facilitate using machine learning methods to automatically infer plausible correspondences between the gene products. Knowing these correspondences can be helpful either for predicting protein abundances from microarray data or as an independent source of information that can be used for learning richer models such as regulatory networks. We propose a probabilistic model that relates protein abundances to mRNA expression levels. Using cross-mapped data from the above-mentioned studies, we learn the model and then score the genes for their strength of relationship by performing probabilistic inference in the learned model. While we gave a simplified outline of our technique in a publication aimed at biologists (Cell 2006), in this paper, we give a complete description of the Bayesian model and the computational technique used to perform inference. In addition, we demonstrate that the Bayesian technique achieves mappings with higher statistical significance, compared to standard linear regression and a maximum likelihood version of the proposed model.

1 Introduction

Proteins are macromolecules essential to the structure and function of all living cells. The biological process in which cells produce proteins from DNA involves an intermediate step where the DNA is transcribed into messenger RNA (mRNA), before being translated into a protein. An important problem in biology is to understand correspondences between levels of mRNA transcripts and abundances of proteins that are produced. However, the biological processes underlying translational regulation are quite complex, so inferring correspondences between these two gene products is non-trivial. Addressing this correspondence problem would facilitate better understanding of cell functionality (c.f. [1,2]). If we know that there is a direct relationship between the two gene products, we can determine protein abundance level at the genome level using simpler and more cost-effective microarray-based mRNA expression measurements. Alternatively, if we can ascertain no relationship between them, they can be treated as complementary independent sources of information that can be used in learning richer models, such as for predicting interaction networks.

In this paper, we seek to infer relationships between protein abundance and mRNA level using noisy high throughput expression profiles of mRNA obtained using microarrays and expression profiles of protein obtained using mass spectrometry. We define a probabilistic model that relates cross-mapped products from these data sets. After learning the parameters of the model using the cross-mapped data, we score the strength of the relationship between protein abundance level and mRNA expression level on a gene by gene basis. In addition, we perform permutation testing and assign a p-value to each gene, thereby obtaining a confidence measure of the significance of the inferred relationship. In [1], we provide a biologist's overview of some parts of the model described in this paper. Here, we provide thorough description of the computational method used to analyze the data. Further, we compare our method to linear regression (which has previously been proposed for this problem) and maximum likelihood estimation in our model, and show that the Bayesian approach recovers a larger number of statistically-significant relationships. The relationships thus detected provide a resource for potential new biological discoveries [1].

There have been a number of previous approaches to inferring correlations between mRNA and protein levels [3,4,5]. Almost all previous methods perform correlation analysis on a global scale, and report positive but weak association between transcript and translational levels. These methods suffer from three main problems. First, they usually summarize global relationships between the measured levels of the two gene products. However, it is an accepted fact that the processes involved in translating mRNA into protein product are quite complicated and vary between genes. Therefore, a more relevant goal is to infer correlation on a gene-by-gene basis [6]. Second, measurements obtained from existing technologies are prone to be quite noisy, but most previously proposed methods either do not explicitly account for noise or assume a strictly Gaussian form of noise. One way to account for non-Gaussian noise is to include additional hidden variables, such as unknown abundances of bio-molecules, but most previously-proposed methods do not incorporate such hidden variables. Methods reported in [3,6] use robust correlation approach and were able to report stronger correlations, thereby conveying that we can better relate the two gene products by properly accounting for non-Gaussian noise. However, one problem with the approach suggested in [6] is that it uses a concordance test based on a presence or absence call, making it inappropriate when quantitative expression

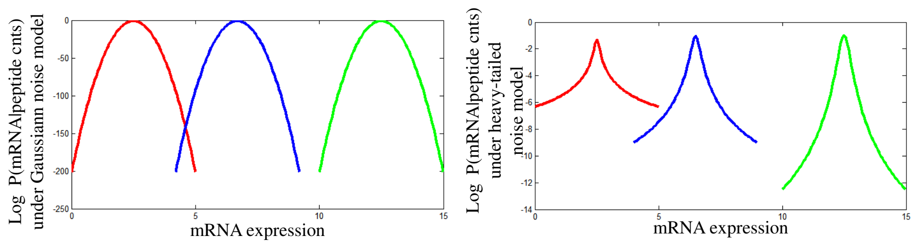


Fig. 1. A schematic figure illustrating differences between our approach and a Gaussian-noise-model approach. In (a), we show the probability of mRNA expression given different values for the peptide count, under a linear model with Gaussian noise of fixed variance. This noise model does not capture the fact that the variance of a sum of counts depends on the number of counts. Further, it does not account for outlying mRNA expression measurements, caused by spurious noise. In (b) we show the kind of probability model used in this paper – the variance decreases for larger values of the peptide count, and the noise model is heavy-tailed and thus the model is less sensitive to outliers.

values are to be analyzed. Following this line of thinking, we formulate a proper probability model that accounts for relevant noise processes, as shown schematically in Fig. 1. Third, previously, models that take into account multiple sources of variability that govern the relationships between the two gene products could not be directly learned from data mainly because of limited availability of data. Therefore, earlier approaches suffered from inability to be invariant to known sources of variation.

In this paper, we overcome these three problems in a principled fashion, allowing us to better understand relationships between mRNA and protein levels. We propose a method that analyzes the available data on a gene-by-gene basis. To obtain an understanding at the gene level, we introduce a probabilistic model that uses a Bernoulli switch variable, which when inferred, either explains microarray expression levels of mRNA as a noisy linear function of the hidden parameter of a Poisson distribution over observed peptide counts, or as being independent of peptide counts and accounted for by a background model learned using only microarray measurements. The probabilistic framework can account for both biological and experimental sources of uncertainty. In conjunction with our biology collaborators, we recently published comprehensive dataset of mRNA expression [8] and protein expression [1] across the entire genome of the laboratory mouse, *Mus musculus*. We make use of these large-scale data sets to learn the model. The probabilistic framework enables us to incorporate other possible kinds of data in a principled way.

The reminder of the paper is organized as follows: Sec. 2 describes the reliably cross-mapped dataset we used for our analysis. Sec. 3 describes the probability model we propose for inferring relationship between the two datsets, and the method used for inference and learning in this model. In Sec. 4, we provide the results of our analysis, and compare them with other existing standard techniques. We draw conclusions in Sec. 5 and outline potential directions for future work.

2 Data and Its Representation

In this paper, we make use of the compendium of protein abundances reported in [1]. This molecular compendium provides the protein content of 4,768 proteins in four major organelle compartments (cytosol, membranes or microsomes, mitochondria and nuclei) in six organs (brain, heart, kidney, liver, lung and placenta) of the laboratory mouse, *Mus musculus*. Protein abundance is measured using a comprehensive comparative proteomic profiling procedure based on gel-free multidimensional protein identification technology (MudPIT). We performed 7 MudPIT experiments and summed their spectra to obtain a discrete measure of protein abundance known as peptide count. It is shown that peptide counts produced by this procedure are positively correlated with actual protein abundance and thus can plausibly be used as a quantitative measure of protein abundance [7].

Recently, two genome-scale surveys of mRNA transcripts levels in mouse tissues were published [8,9]. While [8] uses high-density inkjet synthesized long-oligonucleotide microarrays, [9] uses custom short-oligonuleotide Affymetrix gene chips to study gene expression. We performed a three-way cross-mapping between these three data sets, and found 1,914 detected gene products to be in common across all three platforms. We used this cross-mapped set of genes to perform our analysis of inferring relationships between mRNA transcript levels from [8] and protein abundance from [1].

3 A Probability Model of mRNA Expression and Protein Abundance

Figure 2 shows a Bayesian network for inferring relationships between mRNA expression levels and protein abundance levels on a gene-by-gene basis. We consider a set of G genes that are indexed by g. Let $\mathbf{m}$ and $\mathbf{y}$ be the measurements of its two gene products, mRNA expression level and protein abundance level. Both $\mathbf{m}$ and $\mathbf{y}$ are T-dimensional vectors corresponding to measurements of T tissues. We index the tissues by i so that i^{th} element, y_i, of the vector $\mathbf{y}$ is the peptide count corresponding to the i^{th} tissue.

As described in Sec. 2, the peptide count for a particular tissue is the sum of counts across multiple MudPIT experiments. When multiple MudPIT experiments record the presence of a particular protein, the final peptide count corresponding to the protein will be large. Therefore, for each tissue, we model the effect of multiple MudPIT experiments on its observed peptide counts using a latent variable x_i. We can view this variable as the true rate at which peptides are presented in all MudPIT experiments. We represent this unknown rate for

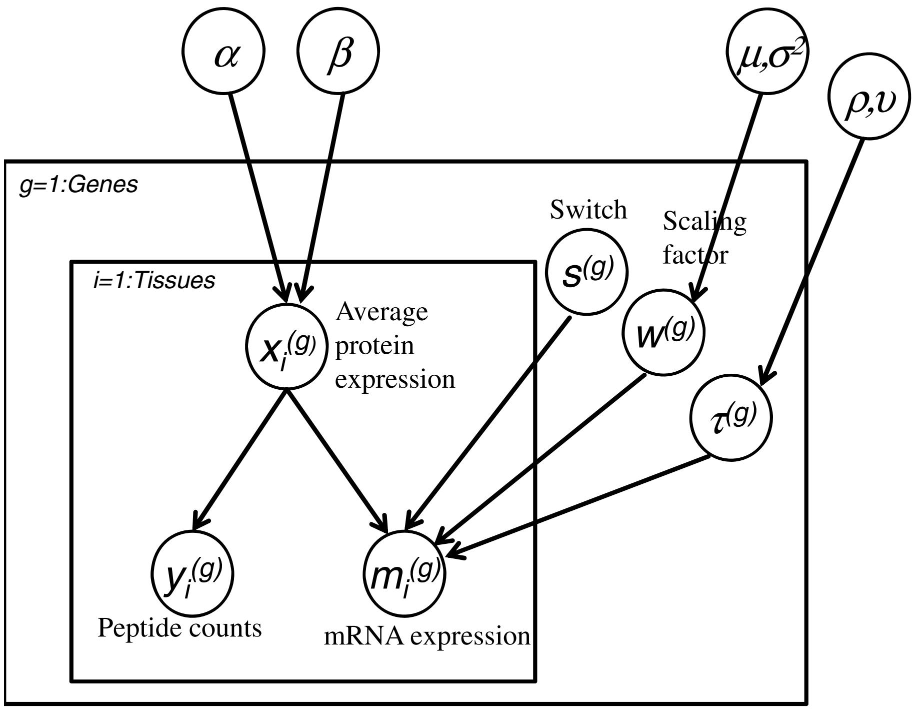


Fig. 2. A probability model for inferring the relationship between peptide counts measuring protein expression and microarray mRNA expression levels. This Bayesian network makes use of plate notation, where the sub-model within a rectangle, including edges entering the rectangle, is replicated; The nodes outside the rectangle and connected to the nodes in the rectangle are shared across all replications. For instance, the graph in the innermost rectangle corresponds to a single gene g and is replicated T times to match with T tissues. All the T tissues for a single gene g shares the same s, w and τ variable. As each gene has independent set of these variables, we use another sub model indexed by g. The variables that are shared across multiple genes are outside the outermost rectangular enclosing.

all tissues by $\mathbf{x}$. With this, the distribution of the peptide counts for the protein given the rate is modeled using an independent Poisson distribution for each tissue, with rate parameter x_i:

$$p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{T} p(y_i|x_i) = \prod_{i=1}^{T} \frac{e^{-x_i} x_i^{y_i}}{y_i!}. \tag{1}$$

We model the rate parameter $\mathbf{x}$ using a Gamma distribution,

$$p(\mathbf{x}) = \prod_{i=1}^{T} p(x_i) = \prod_{i=1}^{T} \frac{\beta^{\alpha}}{\Gamma(\alpha)} x_i^{\alpha-1} e^{-x_i \beta}. \tag{2}$$

The Gamma distribution is suitable because it is the conjugate prior to the Poission distribution, so we can analytically compute the posterior distribution $p(\mathbf{x}|\mathbf{y})$ over the rate variables given the peptide counts. This is also a Gamma distribution given by

$$p(\mathbf{x}|\mathbf{y}) = \prod_{i=1}^{T} p(x_i|y_i) = \prod_{i=1}^{T} \frac{(\beta+1)^{\alpha+y_i}}{\Gamma(\alpha+y_i)} x_i^{(\alpha+y_i)-1} e^{-x_i(\beta+1)}. \qquad (3)$$

This posterior distribution is concentrated on a small range of $\mathbf{x}$ when $\mathbf{y}$ is large; A large value for $\mathbf{y}$ indicates that the corresponding protein level is measured reliably by multiple MudPIT experiments. When the observed peptide count is small, it reflects uncertainty in the expression of a particular protein and therefore the prior distribution plays a more dominating role, making the posterior distribution more spread out.

We represent the expression profile of the mRNA abundance corresponding to a particular gene by a vector $\mathbf{m}$, so that the expression for the i^{th} tissue is m_i. We know that there are many genes for which mRNA expression and protein abundance levels do not agree due to either biological factors such as post-translational modifications or experimental factors such as noisy measurements and changes in conditions between measurements. We model the decision about whether or not the data can be mapped using a binary switch variable, s, with prior distribution denoted by $P(s)$. We typically fix $P(s = 1) = .95$, but this is only a prior on the decision and hence will play only a weak role and not force linear relationships where there aren't any. For a particular gene, if the switch is in the 'off' state (s=0), it indicates that the mRNA expression levels and the hidden peptide rates for that gene do not agree. In this case, the microarray measurements are assumed to be independent of the proteomics measurements, and the microarray measurements are accounted for by using a background model $p_o(\mathbf{m})$ learned using kernel density estimation (c.f. [10]). Kernel density estimation captures the underlying space of microarray expression profile by placing common variance Gaussian kernels on each profile, where variance is computed using leave-one-out cross validation. For our model, we use the entire available microarray data for each tissue to learn their corresponding independent background model.

If the switch variable is in the 'on' state (s=1), the microarray measurement for each tissue is explicitly modeled as a noisy linearly weighted function of the average peptide counts, given by $m_i = wx_i +$ noise. We assume the noise in the microarray measurement is Gaussian with mean 0 and variance τ. While we assume Gaussian noise here, the model on the whole is multi-layer and hierarchical with many more random variables with different distributions including Poisson, Gamma, and discrete. This means that the resulting model is far from being Gaussian and therefore does not have the shortcomings of linear regression and correlation-based methods, which effectively assume Gaussian noise.

The scalar weight w is shared across all tissues for a particular gene and is interpreted as the scaling factor required to match x_i with m_i for all tissues, up to some noise level. As w models gene-specific effects, it accounts for technological effects such as microarray probe sensitivity and microarray data normalization, to name a few. Also, w can capture biological effects such as translational efficiency; When the translational efficiency is higher for one gene compared to another, it will have a smaller value for w. The distribution of $\mathbf{m}$ conditioned on $s, \mathbf{x}$, w and τ is given by

$$p(\mathbf{m}|\mathbf{x}, \tau, s, w) = \begin{cases} \prod_i p_o(m_i) & \text{if } s = 0 \\ \prod_i \frac{1}{\sqrt{2\pi\tau}} \exp(-(m_i - wx_i)^2/2\tau) & \text{if } s = 1 \end{cases} \tag{4}$$

As described in Sec. 2, we have, for each gene, measurements from 6 tissues. This means that we need to infer the scaling factor w for each gene using only 6 measurements. Using only 6 measurements to determine w is likely to lead to overfitting, making the inferred relationship statistically insignificant. Instead, we take a Bayesian approach that enables us to integrate over all possible values of w. For this, we model the scaling factor w as a random variable with a Gaussian prior distribution so that the effect of w can be averaged out over its entire range of values. Similarly, we also treat the inverse variance τ^{-1} as a random variable with a Gamma distribution as the prior. The parameters of both these prior distributions are shared across all genes, and are learned using the entire data set of expression profiles.

Since the joint model is a Bayesian network, we can write the joint distribution over the variables modeled by it as the product of all the conditional distributions. With $\theta = \{w, \tau\}$,

$$p(\mathbf{x}, \mathbf{y}, \mathbf{m}, \theta, s) = p(\mathbf{x})p(\mathbf{y}|\mathbf{x})p(\mathbf{m}|\mathbf{x}, s, \theta)p(\theta)P(s). \tag{5}$$

The objective of the model is to account for relationship between microarray mRNA expression data and protein peptide count data. One approach to this is to learn the model so as to maximize $p(\mathbf{m}, \mathbf{y})$. However, if we trained the model to maximize the joint probability, $p(\mathbf{m}, \mathbf{y})$, the model will try to account for variability in protein and mRNA expression due to factors such as gene function and tissue-specificity. In fact, our intention for the model is not to explain the biological variability in gene expression but to infer the mapping between the two sources of data. Therefore, we learn the model $p(\mathbf{m}|\mathbf{y})$, which is conditioned on the observed peptide counts and thus need not explain gene- and tissue-specific variations, unless they pertain to the predictability of the mRNA abundance from the protein abundance. While one approach will be to model $p(\mathbf{y}|\mathbf{m})$, we choose to model $p(\mathbf{m}|\mathbf{y})$ because inference and learning is more straight-forward. In summary, given a data set of cross-mapped proteomics and microarray data, our goals are to

- Learn the parameters of the model that maximizes the probability $p(\mathbf{m}|\mathbf{y})$ of observing the mRNA expression profile given the peptide counts for the entire set of genes, and
- Infer, for each gene, the probability $P(s = 1|\mathbf{m}, \mathbf{y})$ that a linear relationship does exist between measurements of these gene products.

3.1 Learning the Model

Given the mRNA expression levels and the protein abundance levels for a set of T tissues corresponding to G genes, the goal is to learn the parameters of the model that maximizes the conditional distribution $\prod_{g=1}^{G} p(\mathbf{m}^{(g)}|\mathbf{y}^{(g)})$. Using (5), we can write the joint distribution over the observations by integrating over all the hidden variables, $s, \mathbf{x}, \theta$:

$$p(\{\mathbf{m}^{(g)}, \mathbf{y}^{(g)}\}) = \int_{\theta} \prod_{g=1}^{G} \sum_{s} P(s^{(g)}) \int_{\mathbf{x}} p(\mathbf{x}^{(g)}) p(\mathbf{y}^{(g)}|\mathbf{x}^{(g)}) p(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)}, \theta) p(\theta) \tag{6}$$

As described above, we do not want our model to account for biological variability in gene expression. Therefore, we can approximate the marginal distribution over the peptide counts $p(\mathbf{y}^{(g)})$ by its empirical distribution. To achieve this, we replace $p(\mathbf{x}^{(g)})p(\mathbf{y}^{(g)}|\mathbf{x}^{(g)})$ with $p(\mathbf{x}^{(g)}|\mathbf{y}^{(g)})p(\mathbf{y}^{(g)})$ as follows

$$p(\{\mathbf{m}^{(g)}, \mathbf{y}^{(g)}\}) = \int_{\theta} \prod_{g=1}^{G} \sum_{s} P(s^{(g)}) \int_{\mathbf{x}} p(\mathbf{x}^{(g)}|\mathbf{y}^{(g)}) p(\mathbf{y}^{(g)}) p(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)}, \theta) p(\theta). \tag{7}$$

This expression gives us the desired conditional probability:

$$p(\{\mathbf{m}^{(g)}\}|\{\mathbf{y}^{(g)}\}) \approx \int_{\theta} \prod_{g=1}^{G} \int_{\mathbf{x}} \sum_{s} P(s^{(g)}) p(\mathbf{x}^{(g)}|\mathbf{y}^{(g)}) p(\theta) p(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)}, \theta), \tag{8}$$

where $p(\mathbf{x}^{(g)}|\mathbf{y}^{(g)})$ can be analytically computed as shown before.

The integration over θ cannot be computed analytically, and hence we cannot optimize the above quantity exactly. But, for each gene, we can lower bound it using an approximate posterior distribution $p(w, \tau^{-1}|\mathbf{y}^{(g)}, \mathbf{m}^{(g)}) \approx q^{(g)}(\theta) = q^{(g)}(w)q^{(g)}(\tau^{-1})$. Here, we use a factorized distribution because accounting for the joint distribution is computationally more difficult. For mathematical and computational convenience, we choose $q^{(g)}(w)$ as a Gaussian distribution and $q^{(g)}(\tau^{-1})$ as a Gamma distribution.

$$\begin{aligned} \log p(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)}) &\geq \log q(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)}) \\ &= \int_{\theta} q^{(g)}(\theta) \log \frac{p(\theta)p(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)}, \theta)}{q^{(g)}(\theta)} \end{aligned} \tag{9}$$

We optimize the bound by alternating between finding the posterior distributions and updating the model parameters. This guarantees that the bound becomes tighter with each iteration and becomes equal when the approximate posterior distribution is same as the true posterior distribution [11]. After computing $q(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)})$, it can be substituted into (8) to give

$$p(\{\mathbf{m}^{(g)}\}|\{\mathbf{y}^{(g)}\}) \geq \prod_{g=1}^{G} \sum_{s} P(s^{(g)}) \int_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}^{(g)}) \int_{\theta} q^{(g)}(\theta) \log \frac{p(\theta)p(\mathbf{m}^{(g)}|\mathbf{x}, s, \theta)}{q^{(g)}(\theta)}$$
$$\approx \prod_{g=1}^{G} \sum_{s} P(s^{(g)}) \int_{\mathbf{x}} p(\mathbf{x}^{(g)}|\mathbf{y}^{(g)}) q(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)}). \tag{10}$$

Computing this integral over $\mathbf{x}$ is hard because it involves taking an expectation of $q(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)})$ with respect to $p(\mathbf{x}|\mathbf{y})$. An approach to compute this expectation is to sample from $p(\mathbf{x}|\mathbf{y})$ and average $q(\mathbf{m}^{(g)}|\mathbf{x}^{(g)}, s^{(g)})$ using the sample [12]. We resort to this approach as it is easy to sample from $p(\mathbf{x}|\mathbf{y})$, as given by (3). We draw N samples $\mathbf{x}^{(1)}, \cdots \mathbf{x}^{(N)}$ from $p(\mathbf{x}|\mathbf{y})$ and use them to approximate the true expectation using the sample average,

$$p(\mathbf{m}|\mathbf{x}) \approx \sum_{s} P(s) \sum_{n=1}^{N} q(\mathbf{m}|\mathbf{x}^{(n)}, s). \tag{11}$$

At this juncture, one may wonder why not collect samples from θ as opposed to using variational inference to integrate over θ. The reason we choose to do it this way is that it is hard to sample from the distribution governing θ, and would require Markov-chain monte-carlo methods. In contrast, sampling from $p(\mathbf{x}|\mathbf{y})$ is exact as it is a known distribution that is easy to sample from. Our goal of learn a model that maximizes $p(\{\mathbf{m}^{(g)}\}|\{\mathbf{y}^{(g)}\})$ lends itself to a simple inference algorithm.

3.2 Inferring Strength of Relationships

For a gene under consideration, the strength of the relationship between its pair of mRNA expression level and protein abundance is given by the probability, $P(s|\mathbf{m}, \mathbf{y})$. We can compute this quantity by applying Bayes rule:

$$P(s|\mathbf{m}, \mathbf{y}) = \frac{\int_{\mathbf{x}} p(\mathbf{m}|s, \mathbf{x}) p(\mathbf{x}|\mathbf{y}) P(s)}{\sum_s \int_{\mathbf{x}} p(\mathbf{m}|s, \mathbf{x}) p(\mathbf{x}|\mathbf{y}) P(s)} \approx \frac{\frac{1}{N} \sum_{n=1}^{N} P(s) q(\mathbf{m}|s, \mathbf{x}^{(n)})}{\sum_s \frac{1}{N} \sum_{n=1}^{N} P(s) q(\mathbf{m}|s, \mathbf{x}^{(n)})}. \tag{12}$$

For this computation, we evaluated the integral using sampling; We used N samples $\mathbf{x}^{(1)}, \cdots \mathbf{x}^{(n)}$ from $p(\mathbf{x}|\mathbf{y})$, in combination with (11). To compute this probability, we first fix a value for s. When $s = 1$, for each tissue, we obtain 1000 samples from $p(x_i|y_i)$. We then compute $p(\mathbf{m}|s, \mathbf{x}^{(n)})$ for each of the samples and then average the probabilities. When $s = 0$, we do not need to sample from $p(x_i|y_i)$ as we use the background model for explaining the microarray measurements. We can normalize the two to obtain the desired distribution, $P(s = 1|\mathbf{m}, \mathbf{y})$.

4 Results

We learned the model described above with the data from Sec. 2. Then, for each of the $1,914$ genes, we computed the probability of a linear relationship between the mRNA expression profile and the corresponding peptide counts as given by (12). We used a permutation test to examine whether our probability calculation is well-calibrated. Since the model parameters are shared across all genes, we used 720 permutations where each permutation involved independently permuting the peptide counts of the tissues within the gene, while making sure that the permutation did not result in the observed data (this is possible because many peptide counts are 0). After each set of permutations, we re-learned the model and scored the genes. Then, we computed the p-value based on how many times we observed a particular probabilistic score by chance.

A plausible alternative approach to our proposed Bayesian way of integrating over parameters θ, is to replace the parameters with point estimates, known as maximum likelihood (ML) estimation. We learned the model by performing ML estimation on w (using an EM-type algorithm) and scored the genes based on p-value obtained using permutation testing as described above. We also studied another much simpler approach, which is to fit a linear regressor (LR) for each gene individually, assuming a fixed variance. We used permutation testing in this case as well to obtain a p-value for each gene. After obtaining p-values using these three methods, we chose to look at the 568 genes that had p-values less than .05 for all the three methods. From Fig. 3(a), it is clear that for a large number of genes, the p-values achieved by the Bayesian method is much lower than the p-value achieved by ML or LR, with ML doing slightly better than LR. Further, for a given p-value threshold, the number of genes satisfying the threshold is larger for the Bayesian method than for either the ML or LR methods. The advantage of using the Bayesian approach is that when under uncertainty (here, we need to estimate w from only 6 measurements), it integrates over all possible ranges of values for the parameters, as opposed to ML and LR, where only a single value for the parameters is used. This is another advantage of our approach which allows us to obtain relationships that are much more reliable as it can better reason under uncertainty, taking into account all the modeled sources of variability. Fig. 3(b)-(c) shows each of these 568 genes as a point in a scatter plot comparing the Bayesian p-values to the ML or LR p-values.

We also analyzed scenarios when one of the methods infers a much stronger relationship (or lack thereof) than the other methods. For this, we performed three experiments. We selected all genes that had p-values less than .005 under our proposed Bayesian method, and had p-values greater than .05 under the other two methods. We found 158 such genes, 10 of which are shown in Fig. 4a. We see that these genes have expression values that are large and would require paying a huge penalty under LR and ML to match the mRNA expression with the average peptide count. The Bayesian method, in contrast, can average over all values of w, appropriately weighted, and thus is less sensitive to the large deviations. In Fig. 4b, we show 10 of the 44 genes where ML performs better than the other methods under the same threshold criterion described above. For

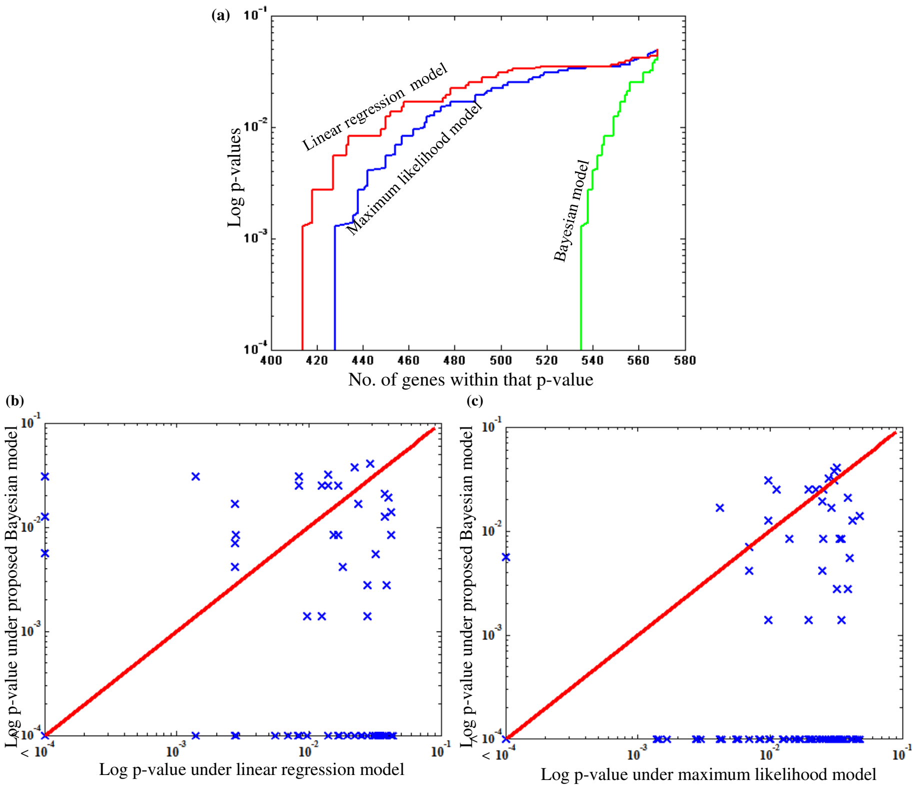
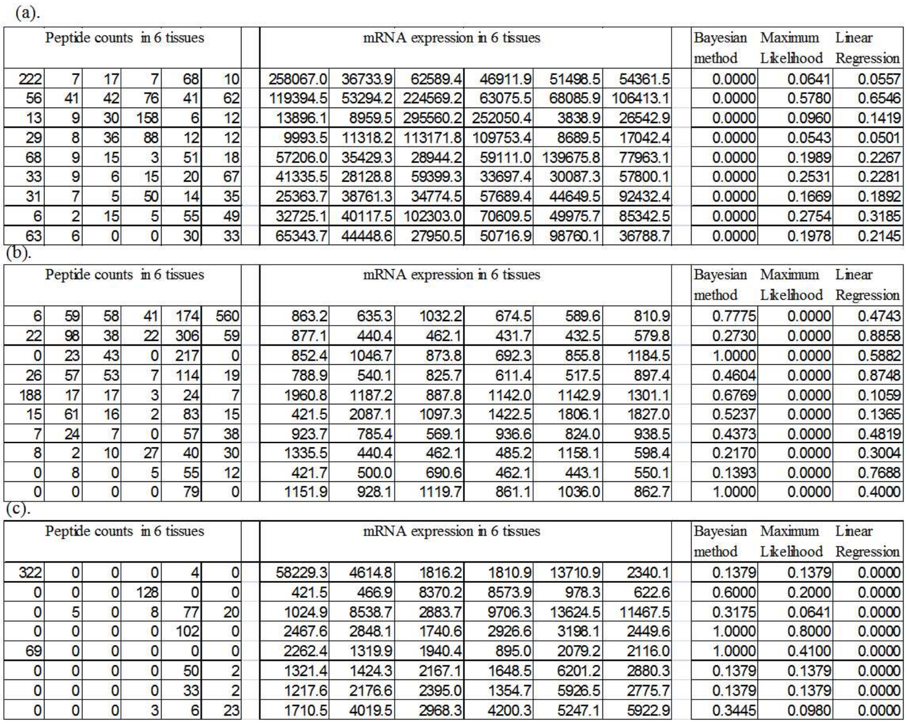


Fig. 3. Comparison between our Bayesian method, maximum likelihood (ML) estimation in the same model, and linear regression (LR). We consider only 568 genes that have p-values (computed from a permutation test) less than .05 in all three methods. (a) shows that for a large number of genes, the Bayesian mapping achieves much lower p-values than the ML or LR methods and for a given p-value threshold detects a larger number of cases with p-values below this threshold. (b) shows the scatter plot of the p-values obtained by the LR and Bayesian method while (c) shows the scatter plot of the p-values obtained by the ML and Bayesian method. In (b) and (c), markers below the 45˙ red line correspond to genes whose Bayesian p-value is lower (indicating higher statistical significance for the Bayesian method) than the ML or LR methods respectively, and vice versa.

these genes, the expression profiles of both gene products are relatively flat and small, indicating quite weak signal values. Fig. 4c shows 10 of the 61 examples in which the p-value under LR is smaller than the other two methods. In this case, the peptide counts are quite tissue-specific and the mRNA expressions are similar, thereby allowing LR to fit the data exactly.

We can also use this model to partition the data into three groups of biological interest: inliers, borderline cases and outliers. The outliers correspond to the set of genes that have $P(s = 1|\mathbf{m}, \mathbf{y}) \leq .33$ and have p-values that within .05.



(a).

Peptide counts in 6 tissues						mRNA expression in 6 tissues						Bayesian method	Maximum Likelihood	Linear Regression
222	7	17	7	68	10	258067.0	36733.9	62589.4	46911.9	51498.5	54361.5	0.0000	0.0641	0.0557
56	41	42	76	41	62	119394.5	53294.2	224569.2	63075.5	68085.9	106413.1	0.0000	0.5780	0.6546
13	9	30	158	6	12	13896.1	8959.5	295560.2	252050.4	3838.9	26542.9	0.0000	0.0960	0.1419
29	8	36	88	12	12	9993.5	11318.2	113171.8	109753.4	8689.5	17042.4	0.0000	0.0543	0.0501
68	9	15	3	51	18	57206.0	35429.3	28944.2	59111.0	139675.8	77963.1	0.0000	0.1989	0.2267
33	9	6	15	20	67	41335.5	28128.8	59399.3	33697.4	30087.3	57800.1	0.0000	0.2531	0.2281
31	7	5	50	14	35	25363.7	38761.3	34774.5	57689.4	44649.5	92432.4	0.0000	0.1669	0.1892
6	2	15	5	55	49	32725.1	40117.5	102303.0	70609.5	49975.7	85342.5	0.0000	0.2754	0.3185
63	6	0	0	30	33	65343.7	44448.6	27950.5	50716.9	98760.1	36788.7	0.0000	0.1978	0.2145

(b).

Peptide counts in 6 tissues						mRNA expression in 6 tissues						Bayesian method	Maximum Likelihood	Linear Regression
6	59	58	41	174	560	863.2	635.3	1032.2	674.5	589.6	810.9	0.7775	0.0000	0.4743
22	98	38	22	306	59	877.1	440.4	462.1	431.7	432.5	579.8	0.2730	0.0000	0.8858
0	23	43	0	217	0	852.4	1046.7	873.8	692.3	855.8	1184.5	1.0000	0.0000	0.5882
26	57	53	7	114	19	788.9	540.1	825.7	611.4	517.5	897.4	0.4604	0.0000	0.8748
188	17	17	3	24	7	1960.8	1187.2	887.8	1142.0	1142.9	1301.1	0.6769	0.0000	0.1059
15	61	16	2	83	15	421.5	2087.1	1097.3	1422.5	1806.1	1827.0	0.5237	0.0000	0.1365
7	24	7	0	57	38	923.7	785.4	569.1	936.6	824.0	938.5	0.4373	0.0000	0.4819
8	2	10	27	40	30	1335.5	440.4	462.1	485.2	1158.1	598.4	0.2170	0.0000	0.3004
0	8	0	5	55	12	421.7	500.0	690.6	462.1	443.1	550.1	0.1393	0.0000	0.7688
0	0	0	0	79	0	1151.9	928.1	1119.7	861.1	1036.0	862.7	1.0000	0.0000	0.4000

(c).

Peptide counts in 6 tissues						mRNA expression in 6 tissues						Bayesian method	Maximum Likelihood	Linear Regression
322	0	0	0	4	0	58229.3	4614.8	1816.2	1810.9	13710.9	2340.1	0.1379	0.1379	0.0000
0	0	0	128	0	0	421.5	466.9	8370.2	8573.9	978.3	622.6	0.6000	0.2000	0.0000
0	5	0	8	77	20	1024.9	8538.7	2883.7	9706.3	13624.5	11467.5	0.3175	0.0641	0.0000
0	0	0	0	102	0	2467.6	2848.1	1740.6	2926.6	3198.1	2449.6	1.0000	0.8000	0.0000
69	0	0	0	0	0	2262.4	1319.9	1940.4	895.0	2079.2	2116.0	1.0000	0.4100	0.0000
0	0	0	0	50	2	1321.4	1424.3	2167.1	1648.5	6201.2	2880.3	0.1379	0.1379	0.0000
0	0	0	0	33	2	1217.6	2176.6	2395.0	1354.7	5926.5	2775.7	0.1379	0.1379	0.0000
0	0	0	3	6	23	1710.5	4019.5	2968.3	4200.3	5247.1	5922.9	0.3445	0.0980	0.0000

Fig. 4. Comparison of the genes products when (a) Proposed Bayesian method (b) Maximum likelihood (c) Linear regression estimates a significant relationship than the other two methods. See accompanying text for details.

These correspond to genes where there is significant disagreement between the measurements of the two gene products. We found that 503 pairs of gene products were in this class. Several of these were blood-borne factors that showed highest mRNA probe signal intensity in liver (where they are primarily synthesized), whereas the corresponding proteins are preferentially detected in the lung and placenta (which are rich in blood vessels). We were able to uncover 409 genes in which the measurements of the two gene products were significantly correlated ($P(s = 1|\mathbf{m}, \mathbf{y}) \geq .66$ and had p-values that were within .05). We call these genes inliers. We group all genes that do not belong to either inliers or outliers as borderline. Figure 5 depicts a breakdown of the genes into the three categories.

We further performed analysis to find if these categorizations have relationships to biological functions. We found that inlier genes were significantly enriched (p-value $< 10^{-3}$) in gene ontology (GO) function annotations such as cell adhesion and central nervous system. The outlier genes were enriched for GO function annotations including embryogenesis and transport, while borderline genes were enriched with function annotations such as mitochondrion, and functional and skeletal developmental anomalies.

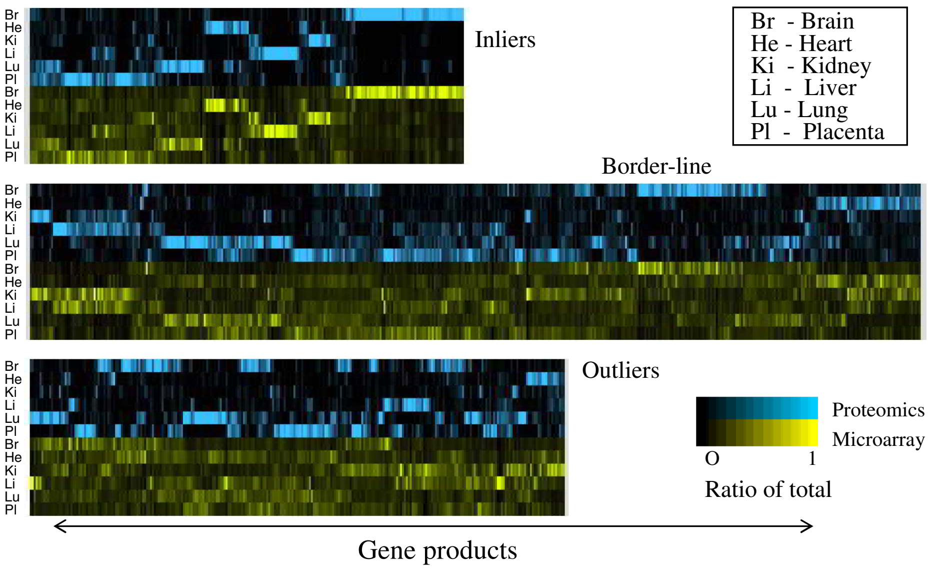


Fig. 5. Clustering of combined peptide and microarray gene profiles into three categories: inliers, borderline and outliers. We can see that for inliers, the genes are typically tissue specific and have similar expression patterns between the two data sets. For outliers, there is significant disagreement in expression profiles.

5 Conclusion

We introduced a probabilistic model for inferring relationships between mRNA expression levels and protein abundance measured as peptide counts. Our model enables probabilistic scoring of the strength of the relationship between the gene products on a gene-by-gene basis. In addition, the same model can be used to test the significance of the relationship. Our model provides a principled framework for including various hidden variables and sources of uncertainty, both biological and experimental, that can affect the measured protein abundance and mRNA expression levels. Importantly, we showed that a Bayesian treatment of our model yields a larger number of statistically significant predictions, in comparison to a maximum-likelihood treatment of our model and linear regression (correlation). We studied experimental variability in the measurement of protein and mRNA levels and we were able to partition a set of genes into inliers, outliers and borderline, based on whether their gene products significantly agree or disagree or unknown. We can further augment the model to incorporate various other information such as protein functions, protein-protein interactions and temporal measurements. As an example, for learning regulatory networks, we can augment our proposed model for inferring correlation between mRNA of a gene and all its protein products, or infer RNA interference by finding relationships between several mRNAs and a single protein.

Acknowledgments

We thank Clement Chung, Brian Cox, Thomas Kislinger, Quaid Morris and Timothy Hughes for helpful discussions. This research was supported by grants from the Natural Sciences and Engineering Research Council of Canada, Genome Canada/Ontario Genomics Institute, and the Canadian Institutes for Health Research. B.J. Frey is a Fellow of the Canadian Institute for Advanced Research.

References

1. Kislinger, T., Cox, B., Kannan, A. *et al.*, *Global survey of organ and organelle protein expression in mouse: combined proteomic and transcriptomic profiling* **Cell**, Apr 7;125(1):173-86(2006).
2. Greenbaum, D., Colangelo, C., Williams, K., Gerstein, M. *Comparing protein abundance and mRNA expression levels on a genomic scale.* **Genome Biol.**, 4(9):117(2003)
3. Gygi, SP., Rochon, Y., Franza, BR., Aebersold, R. *Correlation between protein and mRNA abundance in yeast* **Mol Cell Biology**, Mar;19(3):1720-30,(1999).
4. Griffin, TJ., Gygi, SP., Ideker, T., Rist, B., Eng, J., Hood, L., Aebersold, R., *Complementary profiling of gene expression at the transcriptome and proteome levels in Saccharomyces cerevisiae* **Mol Cell Proteomics**, Apr;1(4):323-33.,(2002).
5. Lian, Z., Kluger, Y., Greenbaum, DS., Tuck, D., Gerstein, M., Berliner, N., Weissman, SM. and Newburger, PE. *Genomic and proteomic analysis of the myeloid differentiation program: global analysis of gene expression during induced differentiation in the MPRO cell line* **Blood**, Nov 1;100(9):3209-20, (2002).
6. Mootha, VK., Bunkenborg, J., Olsen, JV., Hjerrild, M., Wisniewski, JR., Stahl, E., Bolouri, MS., Ray, HN., Sihag, S., Kamal, M., Patterson, N., Lander, ES., Mann, M. *Integrated analysis of protein composition, tissue diversity, and gene regulation in mouse mitochondria.* **Cell**, Nov 26;115(5):629-40, (2003).
7. Liu, H., Sadygov, RG., Yates, JR. *A Model for Random Sampling and Estimation of Relative Protein Abundance in Shotgun Proteomics* **Anal. Chem.**, 76 (14), 4193 -4201(2004).
8. Zhang, W., Morris, Q., *et al.*, *The functional landscape of mouse gene expression.* **Journal of Biology**, 3(5):21(2004).
9. Su, A., Wiltshire, T., Batalov, S. *et al.*, *A gene atlas of the mouse and human protein-encoding transcriptomes* **PNAS**, 101(16): 60626067 (2004)
10. Duda, RO. and Hart, PE. *Pattern Classification and Scene Analysis* **Wiley-Interscience**, (2000).
11. Jordan, MI., Ghahramani, Z., Jaakkola, TS. and Saul, LK, *An Introduction to Variational Methods for Graphical Models* **Machine Learning**, 37-2, (1999).
12. R.M. Neal, *Probabilistic Inference Using Markov Chain Monte Carlo Methods* **Technical Report, University of Toronto**, CRG-TR-93-1, (1993).

Rearrangements in Genomes with Centromeres Part I: Translocations*

Michal Ozery-Flato and Ron Shamir

School of Computer Science, Tel-Aviv University, Tel Aviv 69978, Israel
{ozery,rshamir}@post.tau.ac.il

Abstract. A *centromere* is a special region in the chromosome that plays a vital role during cell division. Every new chromosome created by a genome rearrangement event must have a centromere in order to survive. This constraint has been ignored in the computational modeling and analysis of genome rearrangements to date. Unlike genes, the different centromeres are indistinguishable, they have no orientation, and only their location is known. A prevalent rearrangement event in the evolution of multi-chromosomal species is translocation, i.e., the exchange of tails between two chromosomes. A translocation may create a chromosome with no centromere in it. In this paper we study for the first time centromeres-aware genome rearrangements. We present a polynomial time algorithm for computing a shortest sequence of translocations transforming one genome into the other, where all of the intermediate chromosomes must contain centromeres. We view this as a first step towards analysis of more general genome rearrangement models that take centromeres into consideration.

1 Introduction

Genomes of related species tend to have similar genes that are, however, ordered differently. The distinct orderings of the genes are the result of genome rearrangements. Inferring the sequence of genome rearrangements that took place during the course of evolution is an important question in comparative genomics. The genomes of higher organisms, such as plants and animals, are partitioned into continuous units called *chromosomes*. Every chromosome contains a special region called *a centromere*, which plays a vital role during cell division. An *acentric* chromosome, i.e. one that lacks a centromere, is likely to be lost during subsequent cell divisions [9]. Thus a rearrangement scenario that preserves a centromere in each chromosome is more biologically realistic than a one that does not. The computational studies on genome rearrangements to date have ignored the existence and role of centromeres. Hence, the rearrangement scenarios for multi-chromosomal genomes produced by current algorithms may include genomes with non-viable chromosomes. In this study we begin to address the centromeres in the computational analysis of genome rearrangements.

Since sequencing a centromere is almost impossible due to the repeated sequences it contains, the only information we have on a centromere is its location in the genome. Therefore, in the model we define, centromeres appear as anonymous and orientationless elements. We say that a genome is *legal* if each of its chromosomes contains a

* This study was supported in part by the Israeli Science Foundation (grant 309/02).

single centromere. A *legal* rearrangement operation results in a legal genome (Fig. 1). The *legal rearrangement sorting problem* is defined as follows: given two legal genomes A and B, find a shortest sequence of legal rearrangement operations that transforms A into B. The length of this sequence is the *legal distance* between A and B.

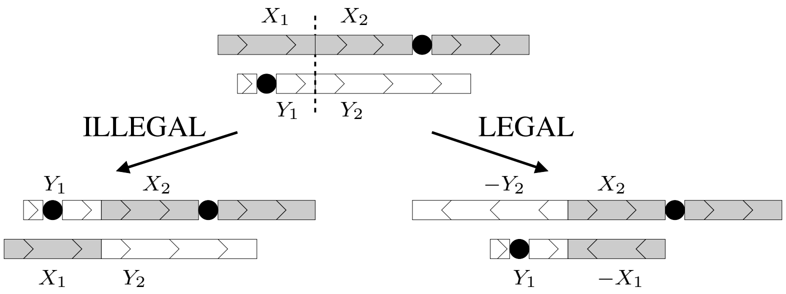


Fig. 1. An example of legal and illegal translocations for a certain cut of two chromosomes. The black circles denote the location of the centromeres. The broken line indicates the positions where the two chromosomes were cut.

A *reciprocal translocation* is a rearrangement in which two chromosomes exchange non-empty ends. A reciprocal translocation results in an illegal genome if exactly one of the exchanged ends contains a centromere. In this paper we focus on the problem of legal sorting by reciprocal translocations, abbreviated hereafter LSRT. This problem is a refinement of the "sorting by reciprocal translocations" problem (hereafter SRT), which ignores centromeres. SRT was studied in [3,2,5,6] and is solvable in polynomial time. Clearly a solution to SRT may not be a solution to LSRT since 50% of the possible reciprocal translocations are illegal (Fig. 1). Indeed, in many cases more rearrangements are needed in order to legally sort a genome.

In this study we present a polynomial time algorithm for LSRT. The basic idea is to transform LSRT into SRT, by replacing pairs of centromeres in the two genomes by new unique oriented elements. Our algorithm is based on finding a mapping between the centromeres of the two given genomes such that the solution to the resulting SRT instance is minimum. We show that an optimal mapping can be found in polynomial time. To the best of our knowledge, this is the first rearrangement algorithm that considers centromeres. While a model that permits only reciprocal translocations is admittedly quite remote from the biological reality, we hope that the principles and structure revealed here will be instrumental for analyzing more realistic models in the future. One additional advantage of centromere-aware models is that they restrict drastically the allowed sequences of operations, and therefore are less likely to suffer from high multiplicity of optimal sequences.

The paper is organized as follows. Section 2 gives the necessary preliminaries. In Section 3 we model LSRT and present some elementary properties of it. Section 4 describes an exponential algorithm for LSRT, which searches for an optimal mapping between the centromeres of A and B, .i. e., one that leads to a minimum SRT solution. In Section 5 we take a first step towards a polynomial time algorithm for LSRT by proving a bound that is at most two translocations away from the legal translocation distance. In Section 6 we present a theorem leading to a polynomial time algorithm for computing the legal translocation distance and solving LSRT. For lack of space most proofs are omitted and the final polynomiality result is only sketched.

2 Preliminaries

This section provides the needed background for SRT. The definitions follow previous literature on translocations [3,2,5,6]. In the model we consider, a *genome* is a set of chromosomes. A *chromosome* is a sequence of genes. A *gene* is identified by a positive integer. All genes in the genome are distinct. When it appears in a genome, a gene is assigned a sign of plus or minus. The following is an example of a genome with two chromosomes and six genes: $\{(1, -5), (-4, -3, -2, 6)\}$.

The *reverse* of a sequence of genes $I = (x_1, \ldots, x_l)$ is $-I = (-x_l, \ldots, -x_1)$. Two chromosomes, X and Y, are called *identical* if either $X = Y$ or $X = -Y$. Therefore, *flipping* chromosome X into $-X$ does not affect the chromosome it represents.

Let $X = (X_1, X_2)$ and $Y = (Y_1, Y_2)$ be two chromosomes, where X_1, X_2, Y_1, Y_2 are sequences of genes. A *translocation* cuts X into X_1 and X_2 and Y into Y_1 and Y_2 and exchanges segments between the chromosomes. It is called *reciprocal* if X_1,X_2, Y_1 and Y_2 are all non-empty. There are two types of translocations on X and Y. A *prefix-suffix* translocation switches X_1 with Y_2:

$$(\underline{X_1}, X_2), (Y_1, \underline{Y_2}) \Rightarrow (-Y_2, X_2), (Y_1, -X_1).$$

A *prefix-prefix* translocation switches X_1 with Y_1:

$$(\underline{X_1}, X_2), (\underline{Y_1}, Y_2) \Rightarrow (Y_1, X_2), (X_1, Y_2).$$

Note that we can mimic one type of translocation by a flip of one of the chromosomes followed by a translocation of the other type.

For a chromosome $X = (x_1, \ldots, x_k)$ define *Tails*$(X) = \{x_1, -x_k\}$. Note that flipping X does not change *Tails*(X). For a genome A define *Tails*$(A) = \bigcup_{X \in A}$ *Tails*(X). For example: *Tails*$(\{(1, -3, -2, 4, -7, 8), (6, 5)\}) = \{1, -8, 6, -5\}$. Two genomes A_1 and A_2 are *co-tailed* if *Tails*(A_1) = *Tails*(A_2). In particular, two co-tailed genomes have the same number of chromosomes. Note that if A_2 was obtained from A_1 by performing a reciprocal translocation then *Tails*(A_2) = *Tails*(A_1). Therefore, SRT is solvable only for genomes that are co-tailed. For the rest of this paper the word "translocation" refers to a reciprocal translocation and we assume that the given genomes, A and B, are co-tailed. Denote the set of tails of A and B by *Tails*.

2.1 The Cycle Graph

Let $n$and N be the number of genes and chromosomes in A (equivalently, B) respectively. We shall always assume that both A and B consist of the genes $\{1, \ldots, n\}$. The *cycle graph* of A and B, denoted $G(A, B)$, is defined as follows. The set of vertices is $\bigcup_{i=1}^{n} \{i^0, i^1\}$. The vertices i^0 and i^1 are called the two *ends* of gene i (think of them as ends of a small arrow directed from i^0 to i^1). For every two genes, i and j, where

j immediately follows i in some chromosome of A (respectively, B) add a black (respectively, grey) edge $(i, j) \equiv (out(i), in(j))$, where $out(i) = i^1$ if i has a positive sign in A (respectively, B) and otherwise $out(i) = i^0$, and $in(j) = j^0$ if j has a positive sign in A (respectively, B) and otherwise $in(j) = j^1$. An example is given in Fig. 2(a). There are $n - N$ black edges and $n - N$ grey edges in $G(A, B)$. A grey edge (i, j) is *external* if the genes i and j belong to different chromosomes of A, otherwise it is *internal*. A cycle is *external* if it contains an external edge, otherwise it is *internal*.

Every vertex in $G(A, B)$ has degree 2 or 0, where vertices of degree 0 (isolated vertices) belong to *Tails*. Therefore, $G(A, B)$ is uniquely decomposed into cycles with alternating grey and black edges. An *adjacency* is a cycle with two edges. A *breakpoint* is a black edge that is not part of an adjacency.

2.2 The Overlap Graph with Chromosomes

A *signed permutation* $\pi = (\pi_1, \ldots, \pi_n)$ is a permutation on the integers $\{1, \ldots, n\}$, where a sign of plus or minus is assigned to each number. If A is a genome with the set of genes $\{1, \ldots, n\}$ then any concatenation π_A of the chromosomes of A is a signed permutation of size n.

Place the vertices of $G(A, B)$ along a straight line according to their order in π_A. Now, every grey edge and every chromosome is associated with an interval of vertices in $G(A, B)$. Two intervals *overlap* if their intersection is not empty but none contains the other. The *overlap graph with chromosomes* of A and B w.r.t. π_A, denoted *OVCH*(A, B, π_A), is defined as follows. The set of nodes is the set of grey edges and chromosomes in $G(A, B)$. Two nodes are connected if their corresponding intervals overlap. An example is given in Fig. 2(b). This graph is an extension of the overlap graph of a signed permutation defined in [4]. Let *OV*(A, B, π_A) be the subgraph of *OVCH*(A, B, π_A) induced by the set of nodes that correspond to grey edges (i.e. excluding the chromosomes' nodes). We shall use the word "component" for a connected component of *OV*(A, B, π_A).

In order to prevent confusion, we will refer to nodes that correspond to chromosomes as "chromosomes" and reserve the word "vertex" for nodes that correspond to grey edges. A vertex is external (resp. internal) if it corresponds to an external (resp. internal) grey edge. Obviously a vertex is external iff it is connected to a chromosome. A component is *external* if it contains an external vertex, otherwise it is *internal*. A component is *trivial* if it is composed of one (internal) vertex. A trivial component corresponds to an adjacency. Note that the internal/external state of a vertex in *OVCH*(A, B, π_A) does not depend on π_A. Therefore, the set of internal components in *OVCH*(A, B, π_A) is independent of π_A. The *span* of a component M is the minimal interval of genes $I(M) = [i, j] \subset \pi_A$ that contains the interval of every vertex in M. Clearly, $I(M)$ is independent of π_A iff M is internal. The following lemma follows from A and B being co-tailed and [4, Corollary 2.2]:

Lemma 1. *Every internal component corresponds to the set of grey edges of a union of cycles in* $G(A, B)$.

The set of internal components can be computed in linear time using an algorithm in [1].

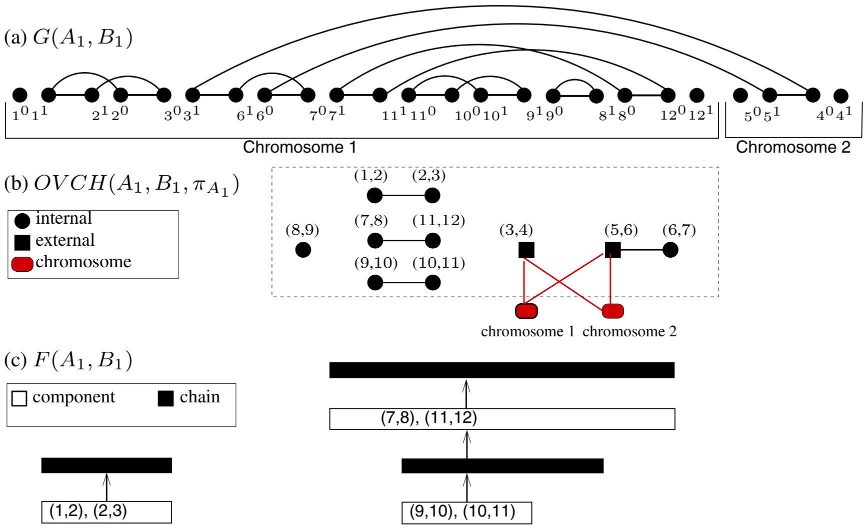


Fig. 2. Auxiliary graphs for $A_1 = \{(1, -2, 3, -6, 7, -11, 10, -9, -8, 12), (5, 4)\}$, $B_1 = \{(1, \ldots, 4), (5, \ldots, 12)\}$ ($\pi_{A_1} = (1, -2, 3, -6, 7, -11, 10, -9, -8, 12, 5, 4)$). (a) The cycle graph. Black edges are horizontal, grey edges are curved. (b) The overlap graph with chromosomes. The graph induced by the vertices within the dashed rectangle is *OV*(A_1, B_1, π_{A_1}). (c) The forest of internal components.

2.3 The Forest of Internal Components

$(M_1, \ldots, M_t)$ is a *chain* of components if $I(M_j)$ and $I(M_{j+1})$ overlap in exactly one gene for $j = 1, .., t-1$. The *forest of internal components* [2], denoted $F(A, B)$, is defined as follows. The vertices of $F(A, B)$ are *(i)* the non-trivial internal components and *(ii)* every maximal chain of internal components that contains at least one non-trivial component. Let M and C be two vertices in $F(A, B)$ where M corresponds to a component and C to a chain. $M \to C$ is an edge of $F(A, B)$ if $M \in C$. $C \to M$ is an edge of $F(A, B)$ if $I(C) \subset I(M)$ and $I(M)$ is minimal. See Fig. 2(c) for an example. We will refer to a component that is a leaf in $F(A, B)$ as simply a *leaf*.

2.4 The Reciprocal Translocation Distance

The *reciprocal translocation distance* between A and B is the length of a shortest sequence of reciprocal translocations that transforms A into B. Let $c(A, B)$ denote the number of cycles in $G(A, B)$. Let $|F(A, B)|$ and $l(A, B)$ denote the number of trees and leaves in $F(A, B)$ respectively. Obviously $|F(A, B)| \le l(A, B)$. Define

$$\delta(A, B) \equiv \delta(F(A, B)) = \begin{cases} 2 & \text{if } |F(A, B)| = 1 \text{ and } l(A, B) \text{ is even} \\ 1 & \text{if } l(A, B) \text{ is odd} \\ 0 & \text{otherwise } (|F(A, B)| \ne 1 \text{ and } l(A, B) \text{ is even}) \end{cases}$$

Theorem 1. *[2,3] The reciprocal translocation distance between A and B is $n - N - c(A, B) + l(A, B) + \delta(A, B)$*

Let Δc denote the change in the number of cycles after performing a translocation on A. Then $\Delta c \in \{-1, 0, 1\}$ [3]. A translocation is *proper* if $\Delta c = 1$, *improper* if $\Delta c = 0$ and *bad* if $\Delta c = -1$.

Corollary 1. *Every translocation in a shortest sequence of translocations transforming A into B is either proper or bad.*

Proof. An improper translocation cannot decrease the translocation distance since it does not affect any parameter in its formula. □

3 Incorporating Centromeres into a Genome

We extend the model described above by adding the requirement that every genome is legal (i.e. every chromosome contains exactly one centromere). We denote the location of a centromere in a chromosome by the element $\bullet$. The element $\bullet$ is unsigned and thus does not change under chromosome flips. The following is an example of a legal genome: $\{(1, 2, 3, \bullet, 4), (\bullet, 5, 6)\}$. The set of tails is defined for regular elements, thus *Tails*$(\bullet, 5, 6) = \{5, -6\}$. We assume that a cut of a chromosome does not split a centromere. Clearly, for every cut of two chromosomes one translocation is legal while the other is not (see Fig. 1).

3.1 A New Precondition

We present here a simple condition for the solvability of LSRT. If this condition is not satisfied then A cannot be transformed into B by legal translocations. For chromosome $X = (x_1, \ldots, x_i, \bullet, x_{i+1}, \ldots, x_k)$ define *Elements*$(X) = \{x_1, \ldots, x_i, -x_{i+1}, \ldots, -x_k\}$. Note that *Elements*$(X) =$ *Elements*$(-X)$. For genome A we define *Elements*$(A) = \bigcup_{X \in A}$ *Elements*(X). For example:

$$\textit{Elements}(\{(1, 2, \bullet, 3, 4), (\bullet, 5, 6)\}) = \{1, 2, -3, -4, -5, -6\}.$$

Observation 1. *Let A and B be two legal genomes. If A can be transformed into B by a sequence of legal translocations then Elements$(A) =$ Elements(B).*

We will see later that this condition is also sufficient. Thus, for the rest of this paper we assume that the input to LSRT is co-tailed genomes A and B satisfying *Elements*$(A) =$ *Elements*$(B) =$ *Elements*. The cycle graph of A and B, $G(A, B)$, ignores the $\bullet$ elements.

3.2 On the Gap Between the Legal Distance and the "Old" Distance

Let $d(A, B)$ denote the legal translocation distance between A and B. Let $d_{\text{old}}(A, B)$ denote the translocation distance between A and B when the $\bullet$ elements are ignored. Obviously $d(A, B) \geq d_{\text{old}}(A, B)$. Consider the genomes A_2 and B_2 in Fig. 3. It can be easily verified that $d_{\text{old}}(A_2, B_2) = 3$ and $d(A_2, B_2) = 4$. This example is easily extendable to two genomes A_{2k} and B_{2k}, with $2k$ chromosomes each, such that $d_{\text{old}}(A_{2k}, B_{2k}) = 3k$ and $d(A_{2k}, B_{2k}) = 4k$.

3.3 Telocentric Chromosomes

A chromosome is *telocentric* if its centromere is located at one of its endpoints. For example the chromosome $(\bullet, 5, 6)$ is telocentric.

Lemma 2. *Let A and B be co-tailed genomes satisfying Elements(A)=Elements(B). Then A and B have the same number of telocentric chromosomes. Moreover, the set of genes adjacent to the centromeres in the telocentric chromosomes is the same.*

Let η denote the number of non-telocentric chromosomes in A and B. We shall show later how mapping between centromeres in non-telocentric chromosomes in A and B can help us to solve LSRT.

3.4 Pericentric and Paracentric Edges

A grey (respectively, black) edge in $G(A, B)$ is said to be *pericentric* if the two genes it connects flank a centromere in genome B (respectively, A). Otherwise it is called *paracentric*. See Fig. 3(a). For a gene i we define:

$$\textit{cent}(i^0) = \begin{cases} -1 & \text{if } i \text{ has a positive sign in } \textit{Elements}, \\ 1 & \text{otherwise.} \end{cases} \qquad \textit{cent}(i^1) = -\textit{cent}(i^0)$$

In other words, the sign of the end closer to the centromere (in both A and B) is positive, and the sign of the remote end is negative. The legality precondition (Section 3.1) implies the following key property:

Lemma 3. *Let (u, v) be an edge in $G(A, B)$. If (u, v) is pericentric then cent(u) = cent$(v) = 1$. Otherwise cent(u)cent$(v) = -1$.*

3.5 Peri-cycles

Let C be a cycle in $G(A, B)$. The *peri-cycle* of C, C^P, is defined as follows. The vertices of C^P are the pericentric edges in C. A vertex in C^P is colored grey (respectively, black) if the corresponding edge in C is grey (respectively, black). A path between two consecutive pericentric edges in C is translated to an edge between the two corresponding vertices in C^P. See Fig. 3. Note that if C contains no pericentric edges then its peri-cycle is a null cycle (i.e. a cycle with no vertices).

Lemma 4. *Every peri-cycle has an even length and its node colors alternate along the cycle.*

Proof. Let C be a cycle that contains a black pericentric edge (u_1, v_1). Suppose $u_1, v_1, \ldots, u_k, v_k$ is a path between two consecutive black pericentric edges in C. In other words, (u_k, v_k) is a black pericentric edge (possibly $u_1 = u_k$ and $v_1 = v_k$) and there are no other black pericentric edges in this path. Then according to Lemma 3 *cent*(v_1) = *cent*$(u_k) = 1$. There is an odd number of edges in the path between v_1 and u_k and thus there must be an odd number of pericentric edges between v_1 and u_k (Lemma 3). It follows that there must exist at least one grey pericentric edge between any two consecutive black pericentric edges. The same argument for a pair of consecutive grey pericentric edges implies that between two such edges there must be at least one black pericentric edge. □

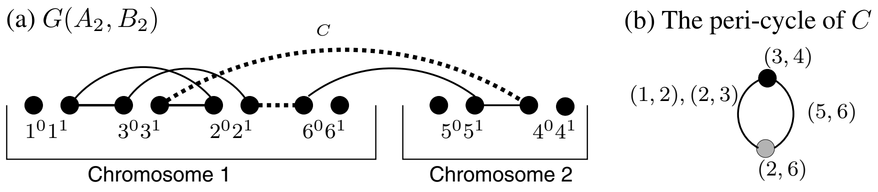


Fig. 3. Pericentric edges and peri-cycles. $A_2 = \{(1,3,2,\bullet,6),(\bullet,5,4)\}$, $B_2 = \{(1,2,3,\bullet,4),(\bullet,5,6)\}$. (a) The cycle graph $G(A_2, B_2)$. Pericentric edges are denoted by dotted lines. (b) The peri-cycle of the single cycle in $G(A_2, B_2)$. The labels of the edges denote the set of grey edges in the corresponding paths.

It follows that every vertex / edge in a peri-cycle has an *opposite* vertex / edge. Removing two opposite vertices / edges from a peri-cycle results in two paths of equal length. We define the *degree* of a cycle as the number of grey (equivalently, black) vertices in its peri-cycle. For example, the single cycle in Fig. 3 is of degree 1.

4 Mapping the Centromeres

This section demonstrates how mapping between the centromeres of A and B can be used to solve LSRT. We shall first see that trying all possible mappings and then solving the resulting SRT gives an exact exponential algorithm for LSRT. Later we shall show how to get an optimal mapping in polynomial time. Let *CEN* $= \{n+1, \ldots, n+\eta\}$. For a genome A let $\dot{\mathbb{A}}$ be the set of all possible genomes obtained by the replacement of each $\bullet$ element in the non-telocentric chromosomes by a distinct element from *CEN*. Each $i \in$ *CEN* can be added with either positive or negative sign. Thus $|\dot{\mathbb{A}}| = \eta! 2^\eta$. For example, if $A_1 = \{(1,2,\bullet,3,4),(\bullet,5,6)\}$ then $\dot{\mathbb{A}}_1$ consists of the genomes $\{(1,2,7,3,4),(\bullet,5,6)\}$ and $\{(1,2,-7,3,4),(\bullet,5,6)\}$. Note that every $\dot{A} \in \dot{\mathbb{A}}$ satisfies *Tails*$(\dot{A}) =$ *Tails*. For each $i \in$ *CEN* we define *cent*$(i^0) =$ *cent*$(i^1) = -1$. A pair $\dot{A} \in \dot{\mathbb{A}}$ and $\dot{B} \in \dot{\mathbb{B}}$ defines a mapping between the centromeres in non-telocentric chromosomes of A and B.

Observation 2. *Let $\dot{A} \in \dot{\mathbb{A}}$ and $\dot{B} \in \dot{\mathbb{B}}$. Then every edge (u,v) in $G(\dot{A},\dot{B})$ is paracentric and satisfies cent(u)cent$(v) = -1$.*

The notion of legality is easily generalized to partially mapped genomes: a genome is *legal* if each of its chromosomes contains either a single $\bullet$ element or a single, distinct element from *CEN* (but not both). Since A and $\dot{A} \in \dot{\mathbb{A}}$ differ only in their centromeres, there is a trivial bijection between the set of translocations on $\dot{A}$ and the set of translocations on A. This bijection also preserves legality: a legal translocation on $\dot{A}$ is bijected to a legal translocation on A.

Lemma 5. *Let $\dot{A} \in \dot{\mathbb{A}}$ and $\dot{B} \in \dot{\mathbb{B}}$. Then every proper translocation on $\dot{A}$ is legal and $d(\dot{A},\dot{B}) = d_{old}(\dot{A},\dot{B})$.*

Proof. Let $k = d_{\text{old}}(\dot{A},\dot{B})$. If $k = 0$ then $\dot{A} = \dot{B}$ and hence $d(\dot{A},\dot{B}) = 0$. Suppose $k > 0$. Let ρ be a translocation on $\dot{A}$ satisfying $d_{\text{old}}(\dot{A} \cdot \rho, \dot{B}) = k - 1$. According to Corollary [1], ρ is either proper or bad. Suppose ρ is bad. Then there is another bad translocation ρ' that cuts the exact positions as ρ, thus satisfying $d_{\text{old}}(\dot{A} \cdot \rho', \dot{B}) = k-1$, and either ρ or ρ' is legal. Suppose ρ is proper. We shall prove that each of the new chromosomes contains a centromere and hence ρ is legal. Let X be a new chromosome resulting from the translocation ρ and let (u, v) be the new black edge in it. Since ρ is proper, $G(\dot{A} \cdot \rho, \dot{B})$ contains a path between u and v where all the edges existed in $G(\dot{A}, \dot{B})$. This path contains an odd number of edges. Following Observation [2] for $G(\dot{A}, \dot{B})$, *cent*(u)*cent*$(v) = -1$. X is composed of two old segments, X_u and X_v, that contain u and v respectively. If *cent*$(u) = -1$ then X_u contains an element from *CEN*, otherwise X_v contains one. In either case X contains an element from *CEN*. □

Theorem 2. *Let* $\dot{A} \in \dot{\mathbb{A}}$. *Then* $d(A, B) = \min\{d_{old}(\dot{A}, \dot{B}) | \dot{B} \in \dot{\mathbb{B}}\}$.

Proof. By Lemma [5], $d(\dot{A}, \dot{B}) = d_{\text{old}}(\dot{A}, \dot{B})$ for every $\dot{A} \in \dot{\mathbb{A}}$ and $\dot{B} \in \dot{\mathbb{B}}$. Obviously a legal sorting of $\dot{A}$ into any $\dot{B} \in \dot{\mathbb{B}}$ induces a legal sorting sequence of the same length, of A to B. Thus, $\min\{d_{\text{old}}(\dot{A}, \dot{B}) | \dot{B} \in \dot{\mathbb{B}}\} \geq d(A, B)$. On the other hand, every sequence of legal translocations that sorts A into B induces a legal sorting of $\dot{A}$ into some $\dot{B} \in \dot{\mathbb{B}}$, thus $\min\{d_{\text{old}}(\dot{A}, \dot{B}) | \dot{B} \in \dot{\mathbb{B}}\} \leq d(A, B)$. □

A pair of genomes, $\dot{A} \in \dot{\mathbb{A}}$ and $\dot{B} \in \dot{\mathbb{B}}$, define an *optimal* mapping between the centromeres of A and B if $d(A, B) = d_{\text{old}}(\dot{A}, \dot{B})$. Theorem [2] and Lemma [5] imply the following algorithm for LSRT:

Algorithm *Legal_Sorting*

```
1. Choose $\dot{A} \in \dot{\mathbb{A}}$ arbitrarily.
2. Compute $\dot{B} = arg \min\{d_{\text{old}}(\dot{A}, \ddot{B}) | \ddot{B} \in \dot{\mathbb{B}}\}$.
3. Solve SRT on $\dot{A}$ and $\dot{B}$ - making sure that every bad translocation in the
   sorting sequence is legal.
```

It can be shown, by a minor modification of the algorithm in [5], that solving SRT with the additional condition that every bad translocation is legal can be done in $O(n^{3/2}\sqrt{\log(n)})$. Step 2 can be performed by enumerating all possible mappings and computing the SRT distance for each. This implies:

Lemma 6. *LSRT can be solved in* $O(\eta! 2^{\eta} n + n^{3/2}\sqrt{\log(n)})$.

Our goal in the rest of this paper is to improve this result by speeding up Step 2, i.e., finding efficiently an optimal mapping between the centromeres of A and B.

5 Cent-Mappings

Our general strategy will be to iteratively map between two centromeres in A and B and replace them with a regular element until all centromeres in non-telocentric chromosomes are mapped. The resulting instance can be solved using SRT, but the increase in the number of elements may have also increased the solution value. The main effort henceforth will be to guarantee that the overall increase is minimal. For this we need to study in detail the effect of each mapping step on the the cycle graph $G(A, B)$. Our analysis uses the SRT distance formula (Theorem 1). We shall ignore for now the parameter δ, and focus on the change in the simplified formula $n-c+l$ (N is not changed by mapping operations).

A mapping between two centromeres affects their corresponding black and grey pericentric edges. Let (i, i') and (j, j') be pericentric black and grey edges in $G(A, B)$ respectively. Suppose $cen \in CEN$ is added between i and i' in $\dot{A}$ and between j and j' in $\dot{B}$. In this case (i, i') and (j, j') in $G(A, B)$ are replaced by the four (paracentric) edges (i, cen), (cen, i'), (j, cen) and (cen, j') in $G(\dot{A}, \dot{B})$. (The first two edges are black, the latter are grey.) We refer to the addition of $cen \in CEN$ between (i, i') and (j, j') as a *cent-mapping* since it maps between two centromeres. Note that for each pair of centromeres in A and B there are two possible cent-mappings (corresponding to the relative signs of the added elements). Given $\dot{A} \in \dot{\mathbb{A}}$, every $\dot{B} \in \dot{\mathbb{B}}$ defines η disjoint cent-mappings and vice versa. Obviously every cent-mapping increases the number of genes by one ($\Delta n = +1$).

Lemma 7. *Every cent-mapping satisfies* $\Delta c \in \{-1, 0, 1\}$.

In the rest of the paper we will analyze the effect of a cent-mapping using peri-cycles. A peri-cycle can be viewed as a compact representation of a cycle focused on pericentric edges, which are the only edges affected by cent-mappings. A cent-mapping is called *proper*, *improper*, *bad* if $\Delta c = 1, 0, -1$ respectively. See Fig. 4 for illustrations of the three types of cent-mappings. We say that a cent-mapping *operates* on a cycle C if C contains at least one of the mapped pericentric edges. A proper / improper cent-mapping always operates on one cycle in $G(A, B)$; A bad cent-mapping always operates on two different cycles in $G(A, B)$.

Observation 3. *Every proper cent-mapping satisfies* $\Delta l \in \{0, 1\}$. *An improper cent-mapping satisfies* $\Delta l = 0$. *A bad cent-mapping satisfies* $\Delta l \in \{0, -1, -2\}$.

It follows that a proper cent-mapping satisfies $\Delta(n-c+l) = 0$ iff $\Delta l = 0$; An improper cent-mapping satisfies $\Delta(n-c+l) = 1$; A bad cent-mapping satisfies $\Delta(n-c+l) = 0$ iff $\Delta l = -2$. A proper cent-mapping is *safe* if it satisfies $\Delta l = 0$. In the following sections we present two classes of cycles, "annoying" and "evil" for which any set of proper cent-mappings that eliminates all their pericentric edges is unsafe.

5.1 Annoying Cycles

In this section we focus on cycles in leaves. The degree of every cycle in a leaf is at most 1 (otherwise it must be external). Moreover, a leaf can contain at most one cycle of degree 1 (for the same reason). A cycle is called *annoying* if: *(i)* it is contained in a leaf, *(ii)* its degree is 1, and *(iii)* a proper cent-mapping on its two pericentric edges satisfies $\Delta l = 1$ (i.e. one leaf is split into two leaves). See Fig. 5(a). Thus a proper cent-mapping on an annoying cycle satisfies $\Delta(n - c + l) = 1$. On the other hand, any bad cent-mapping on a cycle contained in the span of a leaf (annoying or not) results in the elimination of that leaf. Thus, a cent-mapping on two cycles in leaves satisfies

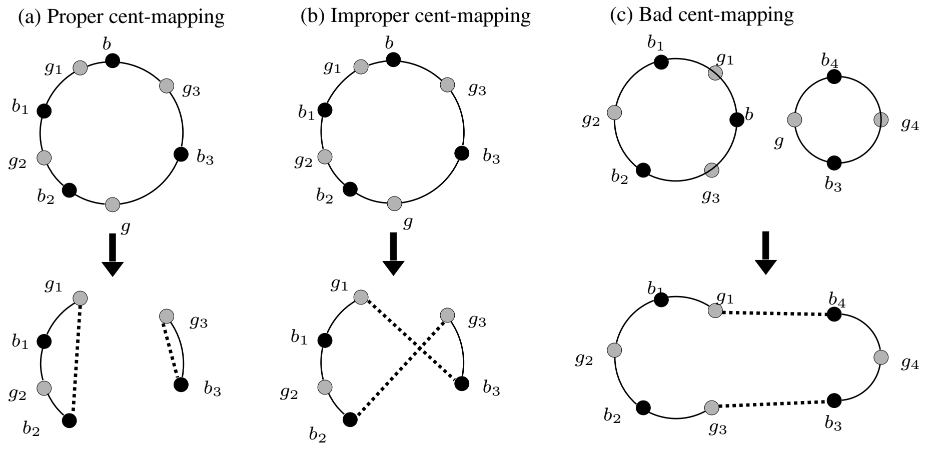


Fig. 4. The effect of a cent-mapping on peri-cycles. Each of the cycles is a peri-cycle with black and grey nodes corresponding to centromeres (pericentric edges) in A and B, respectively. In all cases a cent-mapping on b and g in the top peri-cycles is performed, and the bottom peri-cycles are the result. Dotted lines denote new edges. (a) and (b) show the two alternative cent-mappings of a pair of pericentric edges in the same cycle. In case (c) each of the two alternatives generate a single cycle.

$\Delta(n - c + l) = 1 + 1 - 2 = 0$. Let $\mathbb{C}_{\text{ann}}$ denote the set of annoying cycles and let *ann* $= |\mathbb{C}_{\text{ann}}|$. Let $\mathbb{C}_{\text{nona}}$ be the set of non-annoying cycles of degree 1 that are contained in the span of a leaf. See Fig. 5(b). Let *nona* $= |\mathbb{C}_{\text{nona}}|$.

5.2 Evil Cycles

In this section we focus on cycles that are not in leaves. Let C be a cycle of degree 1 that is not in a leaf and let C^P be its peri-cycle. Let (b, g) be an edge in C^P. Denote by $V(b, g)$ the set of grey edges in the corresponding path between b and g in C. The edge (b, g) is *bad* if after a proper cent-mapping on b and g the edges in $V(b, g)$ belong to a leaf, otherwise it is *good*. For example, in Fig. 3 the edge (b, g) where $V(b, g) = \{(1, 2), (2, 3)\}$ is bad.

Lemma 8. *The "badness" of edge (b, g) in a peri-cycle is unchanged by other cent-mappings not involving b and g.*

Lemma 9. *Let C be a cycle satisfying: (i) deg$(C) > 0$, and (ii) C contains a new grey edge, g_{new}, that was created by a cent-mapping. Let (b, g) be an edge in the peri-cycle of C such that $V(b, g)$ contains g_{new}. Then (b, g) is good.*

A path in a peri-cycle is *bad* if all the edges in it are bad. For a path P, let *len*(P) denote the number of vertices in P. A cycle C is called *evil* if its peri-cycle contains a bad path P such that *len*$(P) >$ *deg*(C). For example, the single cycle in Fig. 3 is evil since it contains a bad edge, which is a bad path of length 2, and its degree is 1. An example of an evil cycle with only bad edges in its peri-cycle is presented in Fig. 5. Let $\mathbb{C}_{\text{evil}}$ denote the set of all evil cycles that are not in leaves. Define *evil* $= |\mathbb{C}_{\text{evil}}|$.

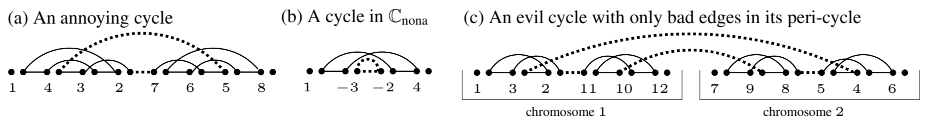

(a) An annoying cycle
1 4 3 2 7 6 5 8
(b) A cycle in $\mathbb{C}_{\text{nona}}$
1 −3 −2 4
(c) An evil cycle with only bad edges in its peri-cycle
1 3 2 11 10 12
chromosome 1
7 9 8 5 4 6
chromosome 2

Fig. 5. Examples of cycles in $\mathbb{C}_{\text{ann}}$, $\mathbb{C}_{\text{nona}}$ and $\mathbb{C}_{\text{evil}}$. In all the figures the target genome B is a fragmented identity permutation (i.e., every grey edge is of the form $(i, i+1)$), pericentric edges are denoted by dotted lines.

Lemma 10. *Let C be a cycle that does not belong to a leaf. There is a set of safe proper cent-mappings of all the pericentric edges in C iff C is not evil.*

Proof. Let C^P be the peri-cycle of C and let $k = deg(C)$. Suppose C is evil. Then PC contains a bad path P with $k+1$ vertices. There are $2k$ vertices in C^P, thus any proper matching of all the pericentric edges in C must match two vertices from P. It follows that there must be a proper cent-mapping on the two ends of an edge in P. Hence, by definition this cent-mapping is unsafe.

Suppose C is not evil. If $k = 1$ then the two edges in C^P are good and the proper cent-mapping of the two pericentric edges in C is safe. Suppose $k > 1$. Let $C^P = P_1, P_2$ where P_1 is a longest bad path in C^P. Let u be the first vertex in P_1 and let v be the last vertex in P_2. Then (u, v) is a good edge in C^P. Let C_1 and C_2 be the two cycles created by the proper cent-mapping on u and v, where C_1 contains $V(u, v)$. Obviously this proper cent-mapping is safe, $deg(C_1) = 0$ and $deg(C_2) = k-1$. It suffices to prove that C_2 is not evil. Let C_2^P be the peri-cycle of C_2. Then $C_2^P = P_1'P_2'$ where $len(P_1') = len(P_1) - 1$, $len(P_2') = len(P_2) - 1$, and P_1' and P_2' are connected by good edges (Lemma 9). Let p be the length of the longest bad path in C_2^P. Then *(i)* $p \leq len(P_1) \leq k$ (since P_1 is a longest bad path in C), *(ii)* $p \leq \max(len(P_1'), len(P_2')) = len(P_2')$, and *(iii)* $len(P_1) + len(P_2) = 2k$. It follows that $p \leq k-1 = deg(C_2)$. Thus by definition C_2 is not evil. □

Corollary 2. *Every cent-mapping satisfies $\Delta(n - c + l + evil) \geq 0$.*

We partition $\mathbb{C}_{\text{evil}}$ into three classes:

- $\mathbb{C}^1_{\text{evil}}$: Cycles of even degree and only bad edges in their peri-cycle.
- $\mathbb{C}^2_{\text{evil}}$: Cycles of odd degree and only bad edges in their peri-cycle.
- $\mathbb{C}^3_{\text{evil}}$: Cycles with at least one good edge in their peri-cycle.

If $C \in \mathbb{C}_{\text{evil}}$ is of degree 1 then $C \in \mathbb{C}^3_{\text{evil}}$ (since otherwise it would be in a leaf). Every new evil cycle (i.e. an evil cycle created by a cent-mapping) contains a good edge (Lemma 9) and hence belong to $\mathbb{C}^3_{\text{evil}}$. Let $C \in \mathbb{C}^3_{\text{evil}}$ and let (b, g) be an edge opposite to a good edge in the peri-cycle of C. A proper cent-mapping on b and g satisfies $\Delta l = 1$, $\Delta evil = -1$ and hence $\Delta(n-c+l+evil) = 0$. Such a cent-mapping can be viewed as a *replacement* of an evil cycle with a leaf. On the other hand, every proper cent-mapping on a cycle in $\mathbb{C}^1_{\text{evil}} \cup \mathbb{C}^2_{\text{evil}}$ satisfies $\Delta(n - c + l + evil) = \Delta(l + evil) = 1$.

Lemma 11. *Let $C \in \mathbb{C}_{evil}$. There exists an improper cent-mapping on C for which $\Delta evil = -1$ iff $C \notin \mathbb{C}^1_{evil}$.*

In other words: for every cycle in $\mathbb{C}^2_{\text{evil}} \cup \mathbb{C}^3_{\text{evil}}$ there exists an improper cent-mapping satisfying $\Delta(n - c + l + \textit{evil}) = 0$; Every improper cent-mapping on a cycle in $\mathbb{C}^1_{\text{evil}}$ satisfies $\Delta(n - c + l + \textit{evil}) = 1$. It follows that a cent-mapping on $C \in \mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}$ satisfies $\Delta(n - c + l + \textit{evil}) = 0$ only if it is bad.

Lemma 12. *Let $C_1, C_2 \in \mathbb{C}_{\textit{evil}} \cup \mathbb{C}_{\textit{ann}}$, where $\textit{deg}(C_1) \leq \textit{deg}(C_2)$. If $\textit{deg}(C_1) = \textit{deg}(C_2)$ then every bad cent-mapping on C_1 and C_2 satisfies $\Delta(l + \textit{evil}) = -2$. If $\textit{deg}(C_1) < \textit{deg}(C_2)$ there exists a bad cent-mapping on C_1 and C_2 satisfying $\Delta(l + \textit{evil}) = -2$ iff $C_2 \in \mathbb{C}^3_{\textit{evil}}$.*

5.3 Sorting by $d + 2$ Legal Translocations in Polynomial Time

In this section we present upper and lower bounds for the legal translocation distance. These bounds provide an intuition for the rather complicated formula for the legal translocation distance presented in the next section. The proof of the upper bound implies an approximation algorithm that sorts A into B using at most $d(A, B) + 2$ legal translocations. For a set of cycles $\mathbb{C}$ let $\textit{subset}(\mathbb{C}, i) = \{C \in \mathbb{C} : \textit{deg}(C) = i\}$. For example, $\textit{subset}(\mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}, 1) = \mathbb{C}_{\text{ann}}$. Define

$$\textit{DEG} = \{i : |\textit{subset}(\mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}, i)| \text{ is odd}\} \qquad \textit{CYC} = \mathbb{C}^3_{\text{evil}} \cup \mathbb{C}_{\text{nona}}$$

For example, if the degrees of the cycles in $\mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}$ are $\{1, 2, 2, 2, 4, 4, 6, 8\}$ then $\textit{DEG} = \{1, 2, 6, 8\}$. The *bad cent-mappings* graph, *BCM*, is defined as follows. It is a bipartite graph whose two parts are *DEG* and *CYC*. Vertices $i \in \textit{DEG}$ and $C \in \textit{CYC}$ are connected by an edge if $\textit{deg}(C) \geq i$. See Fig. 6 for an example. Thus an edge (i, C) represents a bad cent-mapping operating on C and $C' \in \textit{subset}(\mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}, i)$ for which $\Delta(n - c + l + \textit{evil}) = 0$ and $\Delta|\textit{DEG}| = -1$.

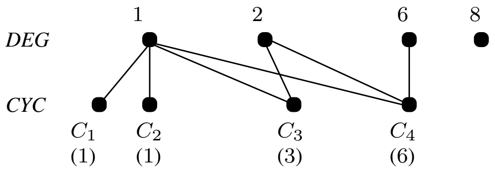


Fig. 6. An example for a bad cent-mappings (*BCM*) graph. $\textit{DEG} = \{1, 2, 6, 8\}$, $\textit{CYC} = \{C_1, C_2, C_3, C_4\}$. The degree of each cycle in *CYC* appears in brackets below the the cycle.

A *matching*, M, is a subgraph of *BCM* where every vertex is adjacent to exactly one edge. The size of a matching M, denoted $|M|$, is the number of edges in it. Finding a maximal matching in *BCM* is an easy task that can be completed in linear time. Define $\textit{fbad} = |\textit{DEG}| - |M|$, where M is a maximal matching. For a matching M let F_M be the forest of internal components after performing a bad cent-mapping on every $C \in \mathbb{C}_{\text{ann}} \cup M$. In other words, F_M is obtained from F by the deletion of every component containing in its span a cycle from either $\mathbb{C}_{\text{ann}}$ or $\mathbb{C}_{\text{nona}} \cap M$. Below we describe Algorithm *Get_Mapping_1* for finding a mapping between the centromeres of A and B.

```
Algorithm Get_Mapping_1

1. Let $M$ be any maximal matching in BCM
2. Perform a bad cent-mapping on every $C_1, C_2 \in \mathbb{C}^1_{\text{evil}}$, where $deg(C_1) = deg(C_2)$.
3. Perform a bad cent-mapping on every pair of cycles in $\mathbb{C}_{\text{ann}}$.
/* Now $|\mathbb{C}^1_{evil} \cup \mathbb{C}_{ann}| = |DEG|$ */
4. For every $(i, C) \in M$ perform a bad cent-mapping on $C$ and $C' \in subset(\mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}, i)$, such that $\Delta(l + evil) = -2$ (Lemma 12).
5. While $|DEG| \geq 3$: For $i = 1, 2, 3$ let $C_i \in \mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}$, and $\deg(C_1)$ is minimal. Perform a bad cent-mapping on $C_2$ and $C_3$ and let $C_4$ be the new evil cycle. Perform a bad cent-mapping on $C_1$ and $C_4$ such that $\Delta(l + evil) = -2$ (Lemma 12).
6. If $|DEG| = 2$: Perform a bad cent-mapping on $C, C' \in \mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}$.
   If $|DEG| = 1$: Perform an improper cent-mapping on $C \in \mathbb{C}^1_{\text{evil}} \cup \mathbb{C}_{\text{ann}}$.
/* Now $|\mathbb{C}^1_{evil}| = ann = 0$ */
7. Perform an improper cent-mapping on every $C \in \mathbb{C}_{\text{evil}}$ such that $\Delta evil = -1$ (Lemma 11).
/* Now $evil = 0$ */
8. Perform safe proper cent-mappings on every cycle of degree at least 1 (Lemma 10).
9. Perform a proper cent-mapping on every $C \in \mathbb{C}_{\text{nona}}$.
```

Theorem 3. *Let $d = d(A, B)$ and let $f = n - N - c + l + evil + \lceil fbad/3 \rceil$. Then $d \in [f, f + 2]$. In particular, Algorithm Get_Mapping_1 produces $\dot{A} \in \dot{\mathbb{A}}$ and $\dot{B} \in \dot{\mathbb{B}}$ for which $d(\dot{A}, \dot{B}) \leq d + 2$.*

6 The Legal Translocation Distance

Define $mbad = fbad \mod 3$. For a matching M define $fgood(M) = |\mathbb{C}^3_{\text{evil}} \setminus M|$. Define $\delta' \in \{0, 1, 2\}$ as follows. $\delta' = 2$ iff the following conditions are satisfied: *(i)* $\mathbb{C}^2_{\text{evil}} = \mathbb{C}^3_{\text{evil}} = DEG = \emptyset$, *(ii)* $|F_\emptyset| = 1$, *(iii)* l and *ann* are even, and *(iv)* If $ann > 0$ then $nona = 0$. Let $\delta' = 0$ iff at least one of the following is satisfied:

(γ1) There exists a maximal matching M satisfying l_M is even and $|F_M| \neq 1$.
(γ2) $mbad = 1, DEG = \{1\}$ and either *(i)* $|F| > 1$, or *(ii)* $l_\emptyset$ is odd and $|\mathbb{C}^2_{\text{evil}}| > 0$
(γ3) $mbad = 1$ and $DEG \neq \{1\}$.
(γ4) $mbad = 2$ and there exists a maximal matching M for which either *(i)* l_M is odd or *(ii)* $fgood(M) \geq 1$.
(γ5) There exists a maximal matching M for which $fgood(M) \geq 2$
(γ6) There exists a maximal matching M for which $fgood(M) \geq 1$, l_M is odd and $C \in \mathbb{C}^3_{\text{evil}} \setminus M$ can be replaced by a leaf such that $|F_M| > 1$.

Theorem 4. *The legal translocation distance between A and B is $d(A, B) = n - N - c(A, B) + l(A, B) + evil(A, B) + \lceil fbad(A, B)/3 \rceil + \delta'(A, B)$.*

The proof of Theorem 4 is by a case analysis of the change in each the parameters $n - c$, l, *evil*, *fbad* and δ' for each cent-mapping and hence is quite involved. It leads to a polynomial time algorithm for finding an optimal mapping between the centromeres of A and B. This algorithm can be viewed as an extension of Algorithm *Get_Mapping_1* that includes a constant number of additional operations that consider δ'.

Theorem 5. *LSRT can be solved in $O(\eta n + n^{3/2}\sqrt{\log(n)})$ time.*

Proof. Finding an optimal mapping between the centromeres of A and B can be done in $O(\eta n)$ in the following manner. The set of peri-cycles can be computed in $O(n)$. For every edge in a peri-cycle we compute its "badness" in $O(n)$ by simply performing the corresponding proper cent-mapping. Computing the badness of all the edges thus takes $O(\eta n)$. Computing $\mathbb{C}^1_{\text{evil}}$, $\mathbb{C}^2_{\text{evil}}$, $\mathbb{C}^3_{\text{evil}}$, $\mathbb{C}_{\text{ann}}$ and $\mathbb{C}_{\text{nona}}$ and *DEG* requires a simple traversal of all the edges in every peri-cycle. Hence it can be done in $O(\eta)$. Overall the algorithm performs $O(\eta)$ operations where each can be implemented in $O(n)$ time. □

7 Summary and Future Work

Computational studies in genome rearrangements have overlooked centromeres to date. In this study we presented a new model for genomes that accounts for centromeres. Using this model we defined the problem of legal sorting by reciprocal translocations (LSRT) and proved that it can be solved in polynomial time. Unfortunately, the legal translocation distance formula appears to be quite complex and it is an interesting open problem whether it or its proof can be simplified.

A solvable LSRT instance requires the two input genomes to be co-tailed and with the same set of elements (see Section 3.1). This requirement is a rather strong and unrealistic. Allowing for reversals, non-reciprocal translocations, fissions and fusions will cancel these restrictions. Under a centromere-aware model, fissions and fusions are legal if they are centric [7,8]. In future work we intend to study an extension of LSRT that allows for reversals, (centric) fusions and fissions. We expect an exact algorithm for this extended problem to bring us nearer to realistic rearrangement scenarios than can be done today.

References

1. D.A. Bader, B. M.E. Moret, and M. Yan. A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. *Journal of Computational Biology*, 8(5):483–491, 2001.
2. A. Bergeron, J. Mixtacki, and J. Stoye. On sorting by translocations. *Journal of Computational Biology*, 13(2):567–578, 2006.
3. S. Hannenhalli. Polynomial algorithm for computing translocation distance between genomes. *Discrete Applied Mathematics*, 71:137–151, 1996.
4. H. Kaplan, R. Shamir, and R. E. Tarjan. Faster and simpler algorithm for sorting signed permutations by reversals. *SIAM Journal of Computing*, 29(3):880–892, 2000.
5. M. Ozery-Flato and R. Shamir. An $O(n^{3/2}\sqrt{\log(n)})$ algorithm for sorting by reciprocal translocations. In *Proceedings of the 17th Annual Symposium on Combinatorial Pattern Matching (CPM 2006)*, volume 4009 of *LNCS*, pages 258–269. Springer, 2006.
6. M. Ozery-Flato and R. Shamir. Sorting by translocations via reversals theory. In *Proceedings of the 4th RECOMB Satellite Workshop on Comparative Genomics*, volume 4205 of *LNCS*, pages 87–98. Springer, 2006.
7. J. Perry, H.R. Slater, and K.H. Andy Choo. Centric fission–simple and complex mechanisms. *Chromosome Research*, 12(6):627–640, 2004.
8. J.B. Searle. Speciation, chromosomes, and genomes. *Genome Research*, 8(1):1–3, 1998.
9. B.A. Sullivan, M.D. Blower, and G.H. Karpen. Determining centromere identity: Cyclical stories and forking paths. *Nature Reviews Genetics*, 2(8):584–596, 2001.

Identification of Deletion Polymorphisms from Haplotypes

Erik Corona[1], Benjamin Raphael[2], and Eleazar Eskin[3]

[1] Dept. of Computer Science and Engineering, University of California,
San Diego, La Jolla, CA 92092
ecorona@ucsd.edu

[2] Dept. of Computer Science & Center for Computational Molecular Biology,
Brown University, Providence, RI 02912
braphael@brown.edu

[3] Dept. of Computer Science, Dept. of Human Genetics, University of California,
Los Angeles, CA 90095
eeskin@cs.ucla.edu

Abstract. Numerous efforts are underway to catalog genetic variation in human populations. While the majority of studies of genetic variation have focused on single base pair differences between individuals, i.e. single nucleotide polymorphisms (SNPs), several recent studies have demonstrated that larger scale structural variation including copy number polymorphisms and inversion polymorphisms are also common. However, direct techniques for detection and validation of structural variants are generally much more expensive than detection and validation of SNPs. For some types of structural variation, in particular deletions, the polymorphism produces a distinct signature in the SNP data. In this paper, we describe a new probabilistic method for detecting deletion polymorphisms from SNP data. The key idea in our method is that we estimate the frequency of the haplotypes in a region of the genome both with and without the possibility of a deletion in the region and apply a generalized likelihood ratio test to assess the significance of a deletion. Application of our method to the HapMap Phase I data revealed 319 candidate deletions, 142 of these overlap with variants identified in earlier studies, while 177 are novel. Using Phase II HapMap data we predict 6730 deletions.

1 Introduction

Since the completion of the reference human genome sequence, efforts have been undertaken to identify genetic variants in the human population. The most comprehensive of these efforts is the International HapMap Project which aims to produce a genome-wide catalog of variation in a population of 270 individuals [1]. Both HapMap and the majority of studies of genetic variation have focused on single base pair differences between individuals, or single nucleotide polymorphisms (SNPs), due to recent technological advances which have significantly reduced the cost of measuring SNPs. However, a number of recent studies have demonstrated that variation at the single nucleotide level is only part of the picture, and that other larger-scale variants are also

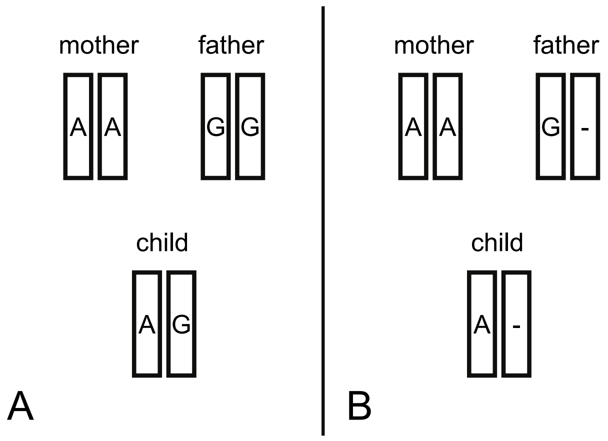


Fig. 1. A. A parent child trio where both parents are homozygous and the child is heterozygous at a particular SNP. In this case, the genotypes are consistent with Mendelian inheritance. **B.** A parent child trio where the father has a hemizygous deletion and will be incorrect genotyped as homozygous. In this case, the deletion is transmitted and the child is also incorrectly genotyped as homozygous resulting in a Mendelian error.

common including copy number polymorphisms, i.e. gains and losses of specific chromosomal regions as well as inversion polymorphisms [11]. The relative contribution of such structural polymorphisms to human genetic variation is still unclear.

Direct techniques for detection and validation of structural variants are generally much more expensive than detection and validation of SNPs. However, for some types of structural variation such as deletion polymorphisms, the structural variation leaves a distinct signature in the SNP data. Specifically, a genotyping assay will incorrectly identify a heterozygous deletion as a homozygous genotype of the undeleted allele, while a homozygous deletion with result in a missing SNP. If we observe the genotypes of both parents and a child, then deletions will typically be observed as an inconsistencies in the rules of Mendelian inheritance (Figure 1). This inconsistency occurs when one parent transmits a deletion to the child and the other parent does not. The child's measured genotype will then be homozygous in the undeleted allele. If the parent transmitting this deletion does not have this allele, then a Mendelian error results.

Two recent papers [2], [8] cleverly exploited this idea and introduced techniques for detecting deletion polymorphisms from the HapMap data by looking for Mendelian errors in trios. Conrad et al. [2] search for groups of Mendelian errors that are physically close on the genome. Candidate deletions are filtered using a series of heuristics. McCarroll et al. [8] create a binary vector for each SNP that indicates the presence or absence of a Mendelian error for each parent-child pair in the population. They then cluster these vectors across different SNPs and look for clusters that contain SNPs physically close on the genome. These methods report 375 and 315 deletion polymorphisms respectively, but only 225 of these are common to both studies suggesting that the sensitivity and/or specificity of the methods are relatively poor [11].

The difficulty with detecting deletions from SNP data, is that there is no one-to-one relationship between Mendelian errors and deletion polymorphisms. First, most observed Mendelian errors are due to experimental errors in genotyping assays rather than the presence of deletions. Indeed, these errors are routinely filtered out of the official HapMap releases. Second, not every deletion produces Mendelian errors at the deleted SNPs: whether or not there is an error depends on the undeleted alleles. For these reasons, using Mendelian errors as a signal for deletions requires sensitive methods to distinguish deletions from experimental errors.

In this paper, we describe a new probabilistic method for detecting deletion polymorphisms from SNP data. The key idea in our method is that we estimate the frequency of the haplotypes in a region of the genome both with and without the possibility of a deletion in the region and apply a generalized likelihood ratio test to assess the significance of a deletion. We argue that the use of haplotypes is a more precise way to examine patterns of Mendelian errors than ad hoc clustering used in earlier studies. Our method has several additional advantages over the earlier methods, neither of which take into account the haplotype structure in the region. First, our method provides a p-value confidence level for each prediction and estimates the frequency of the deletion in a region. Second, the method explicitly combines information from multiple trios, achieving higher specificity than [2] who predict deletions using information only from individual trios. Finally, our method incorporates a flexible model for genotyping errors, which can be adapted to the case when neighboring errors are correlated. This is an important advantage over earlier methods that assume errors are independent because we can directly apply our method to the Phase II HapMap data. This data includes approximately 4 million SNPs, but due to the long range PCR technology[5] used to collect the data by Perlegen Sciences, errors in neighboring SNPs are inherently correlated.

Application of our method to the HapMap Phase I data with 1,109,711 SNPs using a very conservative threshold revealed 319 candidate deletions, 142 of these overlap with variants identified in earlier studies, while 177 are novel. The advantage of our method is by using less conservative thresholds, we can potentially discover many more deletions. We applied our approach to the HapMap Phase II data with 3.8 million SNPs and discover 6730 candidate deletions.

While this paper was under review, a comprehensive map of copy number variation in the HapMap samples was published[9]. A direct comparison of our predictions to this data is complicated for several reasons. The Redon et al. study uses two different technologies to discover copy number variation. These technologies differ in their predictions of the genome coverage of copy number polymorphisms in the HapMap by a factor of two. While the predictions were carefully calibrated to minimize the false positive rate, the false negative rates of the two technologies are unclear. For these reasons, a comprehensive benchmarking of our methods will require a reanalysis of the raw data collected in the Redon et al. study and is beyond the scope of what was possible under the time constraints of the revision cycle of this paper. A compete comparison will appear in the journal version.

2 Methods

2.1 Probabilistic Model for Deletion Polymorphisms

Our method for detecting deletion polymorphisms relies on a generative probabilistic model for trios of mother-father-child genotypes. For a window of L SNPs, we are given a set of haplotypes $h_D, h_1, \ldots, h_N$ with frequencies $p_D, p_1, \ldots, p_N$, where $p_D + \sum_{j=1}^{N} p_j = 1$. Here, h_D represents the deletion haplotype with corresponding frequency p_D. Each haplotype, $h_j \in \{0,1\}^L$ is a string of length L consisting of 0's and 1's, which represent the two possible alleles of a SNP. Let h_j^i denotes the ith SNP of haplotype h_j. We denote the set of haplotype frequencies with the vector $P = (p_1, p_2, \ldots, p_N, p_D)x$.

Each mother-father-child trio is characterized by four-tuple(F_T, F_U, M_T, M_U) of haplotypes, where F_T and F_U are the haplotypes of the father, and M_T and M_U are the haplotypes of the mother. Here the subscript T denotes the haplotype that is transmitted from parent to child, and U denotes the untransmitted haplotype, so that the haplotypes of the child are F_T and M_T.

Given a pair of haplotypes h_j and h_k, with $h_j \neq h_D$ and $h_k \neq h_D$, we define a genotype $G(h_j, h_k) \in \{0,1,2\}^L$ as the string of length L where

$$G(h_j, h_k)^i = \begin{cases} h_j^i, & h_j^i = h_k^i, \\ 2, & h_j^i \neq h_k^i, \end{cases} \tag{1}$$

for $i = 1, \ldots, L$. Thus, 2 represents a heterozygous genotype. We also define $G(h_j, h_D) = G(h_D, h_j) = h_j$. We use the character 3 to indicate a missing SNP with no measured genotype. Consequently, $G(h_D, h_D)$ is defined to be the string of L 3's. We define the genotypes $F = G(F_T, F_U)$, $M = G(M_T, M_U)$ and $C = G(F_T, M_T)$ for the father, mother, and child, respectively.

The experimental procedure used to measure genotypes is subject to errors that either mutate an individual SNPs or fail to assign a value to a SNP. We assume that errors are independent and identically distributed for each SNP, which is a reasonable assumption for most genotyping platforms. (See below for exceptions.) Thus, the error process is given by a set of probabilities $\{e_{i \to j}\}_{i,j=0}^{3}$ where $e_{i \to j}$ indicates the probability that a SNP with the value i is measured as the value j. Then the likelihood of observing genotype $\hat{G}$ when the true genotype is G is given by $L(\hat{G}, G) = \prod_{i=1}^{L} e_{G^i \to \hat{G}^i}$.

Given the observed genotypes in a father-mother-child trio, we are uncertain of the underlying haplotypes due to the possibility of genotype errors as well as the inherent ambiguity of genotypes. However, if we know the underlying haplotype frequencies P, then we can explicitly model this uncertainty by summing over all possible choices of haplotypes to compute the likelihood of the observed trio of father-mother-child genotypes $(\hat{F}, \hat{M}, \hat{C})$, giving the formula:

$$L(\hat{F}, \hat{M}, \hat{C}|P) = \sum_{F_T} \sum_{F_U} \sum_{M_T} \sum_{M_U} p_{F_T} p_{F_U} p_{M_T} p_{M_U} L(\hat{F}, F) L(\hat{M}, M) L(\hat{C}, C) \tag{2}$$

For a window W of L SNPs, the likelihood of a data set consisting of K trios $(\hat{F}_1, \hat{M}_1, \hat{C}_1) \ldots (\hat{F}_K, \hat{M}_K, \hat{C}_K)$ is then the product of the likelihoods of each trio:

$$L(W|P) = \prod_{k=1}^{K} L(\hat{F}_k, \hat{M}_k, \hat{C}_k). \tag{3}$$

In our model, the possibility of a deletion within a window is determined by the value of p_D: if $p_D > 0$ then a deletion is present in the window, while if $p_D = 0$, then no deletion is present in the window. In order to determine whether a deletion occurs in a region, we use a generalized likelihood ratio test to distinguish the null hypothesis $p_D = 0$ versus the alternative $p_D > 0$. For the window W we define the score $S(W)$ by the log likelihood ratio

$$S(W) = 2 \log \frac{\max_P L(W|P)}{\max_{P, p_D = 0} L(W|P)} \tag{4}$$

which intuitively measures the increase in likelihood of the data when deletions can be used to explain the observed genotypes.

$S(W)$ is asymptotically equal to the chi-square distribution with one degree of freedom since (4) is a generalized likelihood ratio test []. Thus, we obtain a p-value $\rho(W)$ indicating the statistical significance of a deletion in W from the chi-square distribution.

2.2 Efficient Scoring of Candidate Deletions

For each window, (4) can be computed by maximizing likelihood over the haplotype frequencies using the Expectation Maximization (EM) algorithm [3]. Unfortunately, direct computation of (4) is impractical because the number of terms in the sum in (2) is $(N+1)^4$ for a single trio, and we must perform this computation during each iteration of the EM algorithm. Since the HapMap contains genotypes for close to 4 million SNPs, this computation becomes intractable for a genome-wide analysis.

In order to make the computation feasible for the HapMap data, we take advantage of recent progress in haplotype phasing algorithms. A recent benchmarking study[7] has shown that several of the state-of-the-art phasing algorithms perform extremely well on trio data. Thus, we make the assumption that these algorithms will accurately predict the haplotype frequencies $p_1^*, \ldots, p_N^*$ assuming there is no deletion ($p_D = 0$) and use one of these algorithms, HAP[4], to predict the haplotype frequencies. Since deletions are typically rare in the population, we also assume that in the presence of a deletion, $p_D > 0$, the remaining haplotypes are correspondingly scaled as $p_i = (1 - p_D)p_i^*$ for $i = 1, \ldots, N$. We use the notation $p_D \circ P = ((1 - p_D)p_1^*, \ldots, (1 - p_D)p_N^*, p_D)$ to denote the vector of scaled probabilities.

Under these assumptions, we can efficiently compute the likelihood of the observed data for any value of p_D using the likelihood in the case that $p_D = 0$. We denote the likelihood in the special case where $p_D = 0$ as $L_0 = L(\hat{F}, \hat{M}, \hat{C}|P, p_D = 0)$. To compute the likelihood for any other value of p_D, we decompose the sum in the likelihood (2) into two parts: one contains all of the terms without any deletions, and the other part contains terms with at least one deletion. Using the notation $|D|$ to denote the number of deletions in the set of haplotypes F_T, F_U, M_T, M_U, the likelihood is then

$$\begin{aligned}
L(\hat{F}, \hat{M}, \hat{C}|P) &= \sum_{F_T, F_U, M_T, M_U, |D|=0} p_{F_T} p_{F_U} p_{M_T} p_{M_U} L(\hat{F}, F) L(\hat{M}, M) L(\hat{C}, C) && (5)\\
&+ \sum_{F_T, F_U, M_T, M_U, |D|>0} p_{F_T} p_{F_U} p_{M_T} p_{M_U} L(\hat{F}, F) L(\hat{M}, M) L(\hat{C}, C) && (6)\\
&= (1 - p_D)^4 L_0 && (7)\\
&+ \sum_{F_T, F_U, M_T, M_U, |D|>0} p_{F_T} p_{F_U} p_{M_T} p_{M_U} L(\hat{F}, F) L(\hat{M}, M) L(\hat{C}, C). && (8)
\end{aligned}$$

Using this equation to compute the likelihood significantly improves the running time since we only need to compute L_0 once and then for any value of p_D we can compute the likelihood by only considering the $4 * (N + 1)^3$ terms in the sum (8).

We then use EM [3] to compute the maximum likelihood estimate $\hat{p_D}$. The likelihood function (3) can be viewed as a weighted sum of multinomials which motivates the following update. Given an estimate β of $\hat{p_D}$, we iteratively compute a new estimate β' as follows. For each observed trio $(\hat{F}_k, \hat{M}_k, \hat{C}_k)$ we compute

$$\beta'_k = \frac{\sum_{F_T, F_U, M_T, M_U, |D|>0} |D| p_{F_T} p_{F_U} p_{M_T} p_{M_U} L(\hat{F}_k, F_k) L(\hat{M}_k, M_k) L(\hat{C}_k, C_k)}{4L(\hat{F}_k, \hat{M}_k, \hat{C}_k|\beta \circ P)}. \quad (9)$$

The new estimate β' is the average of the estimates over all trios $\beta' = \frac{1}{K} \sum_{k=1}^{K} \beta'_k$. Since EM is a local optimization method, for each window, we initialize p_D to multiple starting points and repeatedly apply the update until convergence.

3 Results

3.1 Data Preparation and Parameter Selection

We downloaded SNPs for thirty CEPH parent-child trios, a Caucasian population from Utah (CEU), from Build 16c.1 of the "redundant-unfiltered" version of the Phase I HapMap data. We filtered the data using the same criteria as described in [2] obtaining a total of 1,109,711 SNPs.[1] The SNP data was phased using HAP [4].

Our model depends on two sets of parameters: the vector P of haplotype frequencies and the error probabilities $e_{i \to j}$. We determine the haplotype frequencies p_i, $i = 1, \ldots, n$, directly from the phased haplotype data according to

$$p_i = \frac{\text{occurrences of } h_i \text{ in all parents}}{2(\text{number of parents})}. \quad (10)$$

Note that the haplotypes of the child are transmitted from the parents, and thus are not considering when estimating haplotype frequencies.

To determine the parameters of the error model, we estimate the probability of a genotyping error by examining specific cases where we can predict either the presence

[1] Conrad et al. [2] report 1,108,950 SNPs after filtering the same data. The 761 extra SNPs in our study is likely due to minor implementation differences. These extra SNPs represent only 0.068% of the total, and thus are unlikely to impact a comparison of the two methods.

or absence of an error. We examined every SNP in which the father and mother are homozygous and the offspring has no missing data. In these cases, the genotypes of the parents define the child's genotype. If the observed child genotype does not match the expected genotype, then we observe an error. The genotype error frequency was set to be the total number of observed errors divided by the total number of SNPs examined. We found a total of 23,641 errors in 18,093,695 SNP assays, yielding a genotyping error rate of 1.31E-3 over all chromosomes. Considering chromosomes separately, we find the maximum and minimum estimated genotyping error rates were 3.82E-3 (chromosome 20) and 1.71E-4 (chromosome 15), respectively. This chromosomal variation is not surprising since the genotyping centers responsible for different chromosomes used different criteria for filtering SNPs prior to releasing the data. We set the error probabilities $\{e_{i\to j}\}_{i,j=0}^{2}$ separately for each chromosome according to the empirical values. A similar approach was taken for errors involving missing data. We set $e_{0\to 3} = e_{1\to 3} = e_{2\to 3}$ for each chromosome equal to the number of missing SNPs on that chromosome divided by the total number of genotyped SNPs. We noticed a large disparity between the rate of missing SNPs among individuals. On chromosome 1, the highest reported missing SNP rate was 0.0316 (on individual NA12761) and the lowest was 2.14E-4 (on NA12248), a difference of over 100 times. This seems to indicate great disparity in the quality of data for each individual. Due to this extreme difference in the quality of data for each individual, we also tested our model using the empirical frequency of missing SNPs for each individual. The results were similar to using the empirical frequency of missing SNPs over all individuals.

3.2 Selecting and Filtering Windows

To improve the efficiency of our approach we examine only windows for which our method has a reasonable chance of detecting a deletion. Specifically, we examine only windows containing $1 \leq L \leq 40$ SNPs, at most twenty unique haplotypes for the 30 trios (out of the 120 possible haplotypes), and at least a single Mendelian error compatible with a deletion. We compute a p-value for each window and merge all windows that exceed a specified p-value threshold to obtain predictions over the whole genome. The merging produces a set of non-overlapping windows and an associated p-value for each window giving the probability that this window contains a deletion.

3.3 Predictions and Comparison with Earlier Studies

We compare our method to the method presented in Conrad et al.[2]. Briefly, that method categorizes each SNP into one of seven categories for each trio, labeled A,B,C, D,E, or F. A and B being a Mendelian error consistent with a deletion present in the mother and father, respectively. C represents a Mendelian error that is incompatible with a deletion in either parent. D represents a configuration with child, mother, and father being homozygous or having missing data. E and F represent configurations that include missing data with a possibility of a deletion in the mother and father, respectively. G represents a configuration incompatible with a deletion. The method considers a window from any trio to be a potential deletion if all SNPs in the windows are in states A, D, or E for the mother or B, D, F for the father. Another condition is that there are at least two SNPs in the A category (in an A, D, or E run of consecutive SNPs in the case

Table 1. The number deletions predicted by our method for different p-value thresholds that overlap with a deletion found by [2] or [8] or are novel

P-Value Cutoff	Novel Deletions	Previously Detected Deletions	Conrad	McCarroll
0.05	1527	334	254/1861	156/1861
0.01	782	280	214/1062	134/1062
0.001	350	192	152/542	106/542
1E-4	177	142	111/319	81/319
1E-5	64	98	80/162	62/162
1E-7	9	59	56/68	45/68
1E-9	1	39	36/40	33/40

of the mother having a deletion) or two SNPs in the B category (in a B, D, or F run of consecutive SNPs in the case of the father having a deletion). The maximum range of the deletion is bounded by SNPs in categories compatible with the deletion while the minimum range of the deletion is bounded by the first and last SNP in the A (resp. B) category in the case of the mother (resp. father) having a deletion.

Our proposed method has several advantages over the method in Conrad et al.[2]. First, our probabilistic model takes advantage of information from the complete set of genotypes from all trios in the population while Conrad et al. make a prediction for one trio at a time. Second, since our method uses a generalized likelihood test, we obtain a p-value for our predictions. Third, we use information from the haplotype structure in the region to help predict the deletions. Finally, we estimate the parameters of the model (the rates of genotype errors and missing data) directly from the data.

Table 1 shows the number of deletions detected by our method and the number reported in the Conrad et al.[2] and also in the McCarroll et al[8] studies that used the same data for deletion identification.

For the 216 deletions reported in the CEU population by Conrad et al. [2], our method discovers 319 deletions with a $p < 10^{-4}$. Similarly we discover 81 of the total 304 deletions reported in [8], with a $p < 10^{-4}$.

Since many of the Conrad and McCarroll deletions are non-overlapping is is likely that many more deletions exist in the data which have not been predicted by either study [11]. However, without a complete set of experimentally verified deletions in the HapMap individuals, it is unclear if our novel predictions are true deletions are false positives. In order to evaluate our results, we performed comparisons of our predictions to the combined predictions of several existing studies of deletion polymorphisms, in order to examine the sensitivity and specificity of our method. We formed the following sets of deletions.

1. **Set 1:** Deletion polymorphisms reported in at least one of the following earlier studies. Deletions identified by experimental techniques: namely array-CGH [10,6] or fosmid end sequencing [12], and that contain at least one SNP in the reported deletion. Deletions inferred from SNPs by the methods of Conrad et al. and McCarroll et al. [8,2]. (606 total).
2. **Set 2:** Deletions reported in at least one of the experimental studies only. (173 total).
3. **Set 3:** Deletions reported in at least two of the experimental studies. (18 total).

In the Set 3, we say that a deletion is reported twice if there is any overlap in the deleted intervals. For each set of assumed correct deletions, we define a set of intervals over the genome as follows. We define an interval for each deletion in the set and label that interval as a positive example. Starting at the ends of the deletion, we partition the remaining portion of the genome into 50kb intervals. Each interval immediately adjoining a deletion is labeled as a gap and ignored for the purposes of the evaluation. The remaining intervals are considered not to be deletions and labeled as negative examples.

Figure 2 shows the specificity and sensitivity of our method and the method of [2] when each of these three sets of deletions is the "true" set. It is apparent that our method consistently outperforms the method of [2]. For example, on Set 3, the method of Conrad et al. report only 3 true positives and 608 false positives, achieving extremely poor performance. Even with Set 1, which includes all predictions of Conrad as "true positives", the true positive and false positive fractions are 0.256 and 0.743, respectively.

Conrad et al. [2] performed an experimental validation of 97 detected deletions and validated 80 of them. These 80 validated deletions correspond to 41 non-overlapping regions in the genome. Based on these experimental results, they estimated that their method has a false positive rate of approximately 5% to 14%.

It is very difficult for us to directly compute our false positive rate because we can not distinguish between false positives and novel deletions. However, by comparing our findings to all known deletions (Set 1), we estimate an upper bound on our false positive rate of 2.5%, 13.2%, and 53.9% for p-value thresholds of 1E-9, 1E-7 and 1E-4, respectively. However, this evaluation methodology is overly conservative: since many of the experimental studies were performed on a different set of individuals, it is likely that some of the experimentally predicted deletion polymorphisms are simply not present in the HapMap data. Furthermore, since the experimental studies are performed over a small set of individuals, it is likely that the HapMap contains many more deletion polymorphisms. Most likely, many of the predicted deletions counted as false positives in Figure 2 are in fact novel deletions. We note that even with these extremely conservative upper bounds on the false positive rates, we still discover novel candidate deletions.

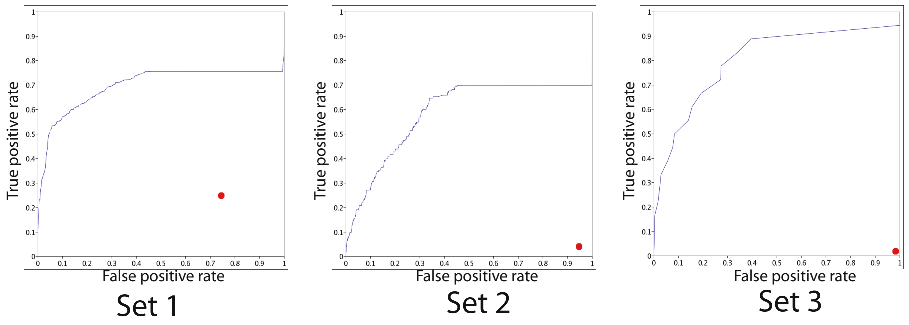


Fig. 2. Receiver operator curve (ROC) curve showing the sensitivity and specificity of our method of deletion detection on three different sets of deletion polymorphisms. By comparison, the method of Conrad (the point on the figure) has significantly lower performance.

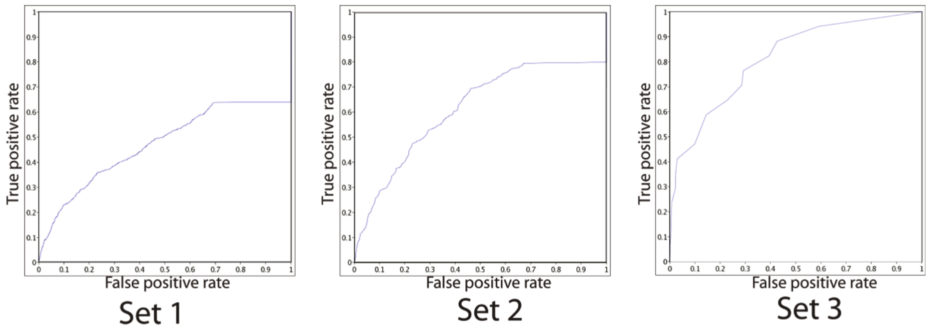


Fig. 3. Receiver operator curve (ROC) for the Aug 12, 2006 release of the Phase II HapMap data

Table 2. Number of Deletions predicted by our method on the HapMap Phase II data

P-value Cutoff	Deletions
`P < 1E-5`	6730
`P < 1E-7`	4227
`P < 1E-10`	2545
`P < 1E-15`	1334
`P < 1E-20`	777
`P < 1E-30`	323
`P < 1E-50`	72
`P < 1E-80`	17

3.4 Phase II Data

We applied our method to the latest Phase II HapMap data as of Aug 12, 2006 containing 3,806,476 SNPS, subjected to the same filtering process as above, and phased with HAP. Phase II presents some unique challenges due to the long range PCR technology [5] used to collect the data by Perlegen Sciences. This technology causes errors in neighboring SNPs that are inherently correlated. Since our method takes into account the haplotype structure, our method is less sensitive to these correlated errors than previously proposed methods. Figure 3 shows the specificity and sensitivity of our method applied to the Phase II data. Sets two and three produce very similar ROC curves in Phase I and Phase II as is apparent in Figures 2 and 3. Table 2 shows the number of predicted deletions in the Phase II data. Due to the larger number of SNPs, there are many more deletions detected with significant p-values.

4 Discussion

We describe a new probabilistic method for detecting deletion polymorphisms from SNP and phased haplotype data of parent-child trios. Unlike previous heuristic techniques for the same problem, our method is grounded in a generative model which allows us to assign a p-value for each prediction. As expected, when benchmarked against the limited set of experimentally validated predictions, our method out performs previously proposed methods.

However, due to the lack of a complete experimentally verified set of deletion polymorphisms, the question of how many deletion polymorphisms can be detected from SNPs remains open and will likely not be resolved until these methods are compared against larger collections of experimentally validated deletions. Methods such as ours for predicting deletion polymorphisms may lead to predictions for novel deletion polymorphisms which may then be further experimentally verified and extend the set of known deletion polymorphisms. By verifying all predicted regions and measuring the relative number of true deletions versus false predictions, we may obtain a better estimate of the false positive rate of this method. However, without knowing the complete set of deletion polymorphisms, it will be very difficult to measure the false negative rate.

Haplotype phasing methods must be updated to account for structural polymorphisms. Presently, these algorithms will either ignore Mendelian errors, or treat them as missing data and infer incorrect haplotypes. We have begun to update the HAP phasing algorithm [4] using the method described here to account for the presence of deletion polymorphisms.

References

1. D. Altshuler, L. D. Brooks, A. Chakravarti, F. S. Collins, M. J. Daly, and P. Donnelly. A haplotype map of the human genome. *Nature.*, 437:1299–1320, 2005.
2. D. F. Conrad, T. D. Andrews, N. P. Carter, M. E. Hurles, and J. K. Pritchard. A high-resolution survey of deletion polymorphism in the human genome. *Nat Genet*, 38:75–81, 2006.
3. A. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal Royal Stat. Soc., Series B*, 39(1):1–38, 1977.
4. E. Halperin and E. Eskin. Haplotype reconstruction from genotype data using imperfect phylogeny. *Bioinformatics (Oxford, England)*, 20:1842–9, 2004.
5. D. A. Hinds, L. L. Stuve, G. B. Nilsen, E. Halperin, E. Eskin, D. G. Ballinger, K. A. Frazer, and D. R. Cox. Whole-genome patterns of common dna variation in three human populations. *Science.*, 307:1072–9, 2005.
6. A. J. Iafrate, L. Feuk, M. N. Rivera, M. L. Listewnik, P. K. Donahoe, Y. Qi, S. W. Scherer, and C. Lee. Detection of large-scale variation in the human genome. *Nature genetics.*, 36:949–51, 2004.
7. J. Marchini, D. Cutler, N. Patterson, M. Stephens, E. Eskin, E. Halperin, S. Lin, Z. S. Qin, H. M. Munro, G. R. Abecasis, and P. Donnelly. A comparison of phasing algorithms for trios and unrelated individuals. *American journal of human genetics.*, 78:437–50, 2006.
8. S. A. McCarroll, T. N. Hadnott, G. H. Perry, P. C. Sabeti, M. C. Zody, J. C. Barrett, S. Dallaire, S. B. Gabriel, C. Lee, M. J. Daly, D. M. Altshuler, and I. H. Consortium. Common deletion polymorphisms in the human genome. *Nat Genet*, 38:86–92, 2006.
9. R. Redon, S. Ishikawa, K. R. Fitch, L. Feuk, G. H. Perry, T. D. Andrews, H. Fiegler, M. H. Shapero, A. R. Carson, W. Chen, E. K. Cho, S. Dallaire, J. L. Freeman, J. R. González, M. Gratacòs, J. Huang, D. Kalaitzopoulos, D. Komura, J. R. MacDonald, C. R. Marshall, R. Mei, L. Montgomery, K. Nishimura, K. Okamura, F. Shen, M. J. Somerville, J. Tchinda, A. Valsesia, C. Woodwark, F. Yang, J. Zhang, T. Zerjal, J. Zhang, L. Armengol, D. F. Conrad, X. Estivill, C. Tyler-Smith, N. P. Carter, H. Aburatani, C. Lee, K. W. Jones, S. W. Scherer, and M. E. Hurles. Global variation in copy number in the human genome. *Nature*, 444(7118):444–454, 2006.

10. J. Sebat, B. Lakshmi, J. Troge, J. Alexander, J. Young, P. Lundin, S. Mnér, H. Massa, M. Walker, M. Chi, N. Navin, R. Lucito, J. Healy, J. Hicks, K. Ye, A. Reiner, T. C. Gilliam, B. Trask, N. Patterson, A. Zetterberg, and M. Wigler. Large-scale copy number polymorphism in the human genome. *Science*, 305:525–528, 2004.
11. A. J. Sharp, Z. Cheng, and E. E. Eichler. Structural variation of the human genome. *Annu Rev Genomics Hum Genet*, 2006.
12. E. Tuzun, A. J. Sharp, J. A. Bailey, R. Kaul, V. A. Morrison, L. M. Pertz, E. Haugen, H. Hayden, D. Albertson, D. Pinkel, M. V. Olson, and E. E. Eichler. Fine-scale structural variation of the human genome. *Nat Genet*, 37:727–732, 2005.

Free Energy Estimates of All-Atom Protein Structures Using Generalized Belief Propagation

Hetunandan Kamisetty, Eric P. Xing, and Christopher J. Langmead*

Computer Science Department, Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA 15217
{hetu,epxing,cjl}@cs.cmu.edu

Abstract. We present a technique for approximating the free energy of protein structures using Generalized Belief Propagation (GBP). The accuracy and utility of these estimates are then demonstrated in two different application domains. First, we show that the entropy component of our free energy estimates can be useful in distinguishing native protein structures from *decoys* — structures with similar internal energy to that of the native structure, but otherwise incorrect. Our method is able to correctly identify the native fold from among a set of decoys with 87.5% accuracy over a total of 48 different immunoglobin folds. The remaining 12.5% of native structures are ranked among the top 4 of all structures. Second, we show that our estimates of $\Delta\Delta G$ upon mutation upon mutation for three different data sets have linear correlations between 0.63-0.70 with experimental values and statistically significant p-values. Together, these results suggests that GBP is an effective means for computing free energy in all-atom models of protein structures. GBP is also efficient, taking a few minutes to run on a typical sized protein, further suggesting that GBP may be an attractive alternative to more costly molecular dynamic simulations for some tasks.

Keywords: Protein Structure, Decoy Detection, Free Energy, Probabilistic Graphical Models.

1 Introduction

This paper describes a technique for modeling protein structures as complex probability distributions over a set of torsion angles, represented by a set of rotamers. Specifically, we model protein structures using probabilistic graphical models. Our representation is complete in that it models every atom in the protein. A probabilistic representation confers several advantages including that it provides a framework for predicting changes in *free energy* in response to internal or external changes. For example, structural changes due to changes in temperature, pH, ligand binding, and mutation, can all be cast as inference problems over the model. Recent advances in inference algorithms for graphical models, such as Generalized Belief Propagation (GBP), can then be used to

* Corresponding author.

efficiently solve these problems. This is significant because GBP is a rigorous approximation to the free-energy of the system [36]. We will show that these free energy estimates are accurate enough to perform non-trivial tasks within structural biology. In particular, we use GBP to a) identify native immunoglobin structures from amongst a set of decoys with 87.5% accuracy, and b) compute changes in free energy after mutation that have a linear correlation of upto 0.70 to laboratory measurements.

The Free energy is defined as $G = E - TS$, where E is the enthalpy of the system, T is the absolute temperature, and S is the entropy of the system. There are numerous energy functions (i.e., E) from which to choose. These functions often model inter- and intra molecular interactions (e.g., van der Waals, electrostatic, solvent, etc.). Unfortunately, entropy estimates can be difficult to compute because they involve sums over an exponential number of states. For this reason, the entropy term is often ignored altogether, under the assumption that it does not contribute significantly to the free energy. This is equivalent to modeling the system at 0 Kelvin. Not surprisingly, this simplification can sometimes limit the accuracy, and thus the utility of the energy calculations. For example, it has been conjectured [3,30] that energy functions comprising sums of pairwise interactions cannot distinguish a protein's native structure from decoy structures within about 1 Å RMSD. If true, one likely explanation is that entropy contributions become significant when structures are similar. Our findings are consistent with this hypothesis. In particular, we find that the native structure is usually the one with the highest entropy. This is in agreement with the findings of others who have have demonstrated the practical benefits of including entropy in energy calculations (e.g., [16]).

Numerous investigators have observed and attempted to address the limitations of pairwise energy functions. Multi-body statistical potentials are a common alternative (e.g., [7,28]). Such potentials do not model the physics directly, but instead use statistics mined from the Protein Data Bank [2] under the assumption that these statistics encode both the entropy and the internal energy. Carter and co-workers [7], for example, have developed a 4-body statistical potential that predicts $\Delta\Delta G$s upon mutations with significant accuracy. There are, however, those that doubt the ultimate utility of statistical potentials (e.g., [29]).

We note that the contributions of this paper do not lie in the suggestion that a protein's structure be treated as a probability distribution — clearly this is the very essence of statistical physics. Rather, our contribution lies in the demonstration that an inference-based approach to free energy calculations is sufficiently accurate to perform non-trivial tasks. Additionally, our technique is efficient and runs in minutes on typical-sized proteins, suggesting it is well-suited for large-scale proteomic studies.

2 A Markov Random Field Model for Protein Structure

In what follows, random variables are represented using upper case variables, sets of random variables appear in bold face while lower case variables represent specific values that the random variables can take. Thus, the random variables representing the position of all the backbone atoms is denoted by $\mathbf{X_b}$, those representing all the side chain atoms, by $\mathbf{X_s}$, X_s^i is the random variable representing the side chain conformation of the i^{th} residue and x_b^i represents a particular value that the backbone of the i^{th} residue takes.

Let $\mathbf{X} = \{\mathbf{X_b}, \mathbf{X_s}\}$ be the random variables representing the entire protein structure. $\mathbf{X_b}$ can be represented by a set of 3-d coordinates of the backbone atoms, or equivalently, by a sequence of bond lengths and dihedral angles. Thus, X_b is typically a continuous random variable. Each X_s^i, is usually represented by a set of dihedral angles[1]. While this too is a continuous random variable, due to steric clashes not all dihedral angles are energetically favorable, allowing a discretization of this state space into a set of discrete favorable conformations called *rotamers*.

The probability of a particular conformation $\mathbf{x}$ can be written as

$$p(\mathbf{X} = \mathbf{x}|\Theta) = p(\mathbf{X_b} = \mathbf{x_b})p(\mathbf{X_s} = \mathbf{x_s}|\mathbf{X_b}, \Theta)$$

or more compactly,

$$p(\mathbf{X}|\Theta) = p(\mathbf{X_b})p(\mathbf{X_s}|\mathbf{X_b}, \boldsymbol{\Theta})$$

where Θ represents any parameters used to describe this model, including sequence information, temperature etc. Frequently the backbone is assumed to be rigid with a known conformation. Therefore $\mathbf{X_b} = \mathbf{x_b}$ for some particular $\mathbf{x_b}$. The term of interest then becomes, $p(\mathbf{X_s}|\mathbf{X_b} = \mathbf{x_b}, \boldsymbol{\Theta})$.

This can be further simplified. Specifically, it is possible to list out conditional independencies that the above probability distribution must satisfy. Consider the random variables X_s^i, X_s^j representing the side chain conformations of residues i, j. Due to the underlying physics, if the residues are not close to each other, their direct influence on each other is negligible. Also, if the residues that directly influence these residues are in specific conformations, X_s^i, X_s^j become conditionally independent of each other. Similar independencies can be listed between side chain variables and backbone variables. These conditional independencies can be compactly encoded using an undirected probabilistic graphical model, also called a Markov Random Field(MRF).

For example, consider a particular backbone conformation $\mathbf{x_b}$ of Lysozyme (pdb id: 2lyz) shown in Fig. 1(a) with a few residues highlighted. Fig. 1(b) shows that part of the Markov Random Field that is induced by the highlighted set of residues. Two variables share an edge if they are closer than a threshold distance. Edges can thus be present between backbone atoms, between backbone and side chain atoms and between side chain atoms. This MRF thus represents the probability distribution of the side chain atoms of a protein with a given backbone.

[1] This is a slight abuse of notation, since it is actually the differences $X_b^i - X_b^{i-1}$ and $X_s^i - X_b^i$ that are represented using bond lengths and angles.

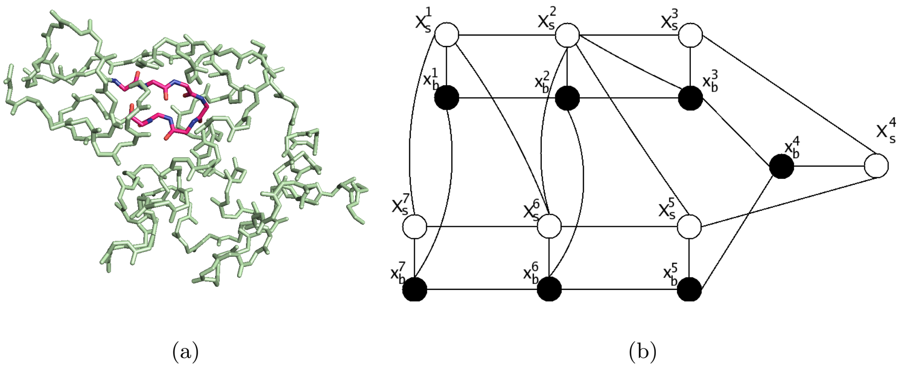


Fig. 1. (a) Structure of the backbone atoms of lysozyme (pdb id: 2lyz) with a few residues highlighted (b) Part of the random field induced by the highligted residues: X_s^i's are the hidden variables representing the rotameric state, the visible variables are the backbone atoms in conformations x_b^i

In general, an MRF encodes the following conditional independencies for each vertex X_i and for any set of vertices $\mathbf{X}'$ not containing X_i.

$$p(X_i|\mathbf{X}', Neighbors(X_i)) = p(X_i|Neighbors(X_i))$$

That is, a random variable X_i is conditionally independent of every other set of nodes in the graph, given its immediate neighbors in the graph.

Given this representation, the probability of a particular side chain conformation $\mathbf{x_s}$ given the backbone conformation $\mathbf{x_b}$ can be expressed as

$$p(\mathbf{X_s} = \mathbf{x_s}|\mathbf{X_b} = \mathbf{x_b}) = \frac{\mathbf{1}}{\mathbf{Z}} \prod_{\mathbf{c} \in \mathbf{C}(\mathbf{G})} \psi_{\mathbf{c}}(\mathbf{x_s^c}, \mathbf{x_b^c})$$

where $C(G)$ is the set of all cliques in G, ψ is a potential defined over the variables, and Z is the so called partition function.

To completely characterize the MRF, it is necessary to define the potential function ψ. A common simplifying assumption is that of a pair-wise potential. We use the Boltzmann Distribution to define a pairwise potential function in the following manner:

$$\psi(X_s^{i_p}, X_s^{j_q}) = exp(-E(x_s^{i_p}, x_s^{j_q})/k_B T)$$

where E_{i_p,j_q} is the energy of interaction between rotamer state p of residue X_s^i and rotamer state q of residue X_s^j and k_B is the Boltzmann constant. Similarly, we can define the potential function between a side chain random variable X_s^i and a backbone random variable X_b^j which is in an observed state x_b^j

$$\psi(X_s^{i_p}, X_b^j) = exp(-E(x_s^{i_p}, x_b^j)/k_B T)$$

Finally, we define the potential function between two backbone random variables to have the trivial value of 1, since both are observed, i.e. $\psi(X_b^i, X_b^j) = 1$.

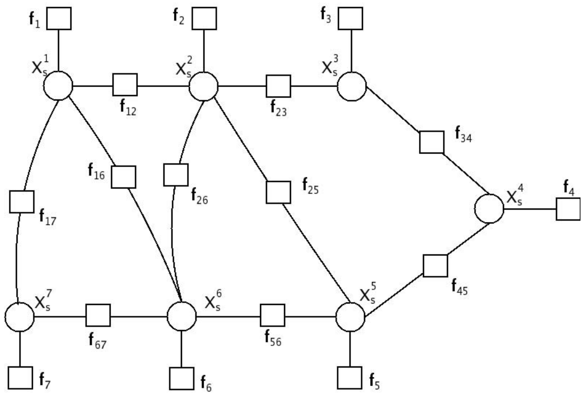


Fig. 2. Factor graph representation for the graph shown in Fig. 1(b). The observed variables corresponding to the backbone atoms can be replaced by a factor at each side chain variable.

This undirected graphical model, characterized by the variables $\mathbf{X}$, the edges between the variables and the potential ψ can also be represented more conveniently, as a bipartite graph $(\mathbf{X}, F)$, called a factor graph. If we restrict ourselves to pairwise potentials, as we have done already by our form of potential function, the equivalent factor graph for the MRF of Fig. 1(b) is shown in Fig. 2. Each edge between side chain variables has been replaced by edges to a factor representing the interaction between these variables. Also, it can be shown that the observed backbone variables can be dropped from the factor graph by replacing their interactions with each side chain variable by a factor. The probability of a particular conformation can then be expressed using the factor notation, as

$$p(\mathbf{x_s}) = \frac{1}{Z} \prod_{f_a \in F} f_a(\mathbf{x_s^a})$$

where $\mathbf{X_s^a}$ is the set of variables connected to factor f_a in the factor graph.

3 Approximating Free Energy

A corollary of the second law of thermodynamics is that a physical system seeks to minimize its free energy. Thus, the most accurate entropy estimates are obtained when the system has the least free energy. Under the assumption of constant temperature, the free energy of a system is given by

$$G = E - TS$$

where E is the enthalpy of the system, T the temperature and S, the entropy. If we associate a belief $b(\mathbf{x})$ with state $\mathbf{x}$, this can be rewritten as

$$G = \sum_{\mathbf{x}} b(\mathbf{x}) E(\mathbf{x}) + T \sum_{\mathbf{x}} b(\mathbf{x}) ln(b(\mathbf{x}))$$

where the first term and second terms on the right are the enthalpic and entropic contributions respectively and the summation is over all possible $\mathbf{x}$. Intuitively, the enthalpic term corresponds to the energy of the system. However, the second law of thermodynamics dictates that not all energy can be used to do work. The free energy is the energy left to be used to do work after deducting the energy that is "lost" which is the entropic deduction mentioned above.

There has been a considerable amount of work by physicists at developing approximations to estimate these terms [4,11,22,23]. The popular methods are based on approximating the free energy using a *region based* free energy. Intuitively, the idea is to break up the factor graph into a set of regions R, each containing multiple factors $\mathbf{f_R}$ and variables $\mathbf{X_R}$, compute the free energy over the region using estimates of the marginal probability over $\mathbf{X_R}$, and then approximate the total free energy by the sum of the free energies over these regions. Since the regions could overlap, contributions of nodes – factors or variables – which appear in multiple regions have to be subtracted out, so that each node is counted exactly once. This can be done by associating weights w_R to the contribution of every node in region R, in such a way that the sum of weights of the regions that the node appears in, sums to one.

This region graph formalism is fairly general and one can create approximations of varying degrees. For example, the Bethe approximation[4] is a region graph with each region containining atmost one factor, while the Kikuchi approximation is a region graph where the regions are created using the so-called *cluster variational approach* that allows regions to contain more than one factor, and is therefore a better approximation[11,36].

While the Kikuchi approximation has been extensively studied, until recently, there was a dearth of algorithms that could compute such region graph based approximations efficiently. See [24] for a recent survey of previously used methods and their performance relative to GBP. In fact, even computing exact marginals for the purpose of computing these approximations is NP-Hard, if the graph, like the MRF described above, has cycles. The Junction Tree algorithm for exact inference has a running time that is exponential in the tree width of the graph, which can be prohibitively expensive in large graphs. However, recent advances within the Machine Learning community on approximate algorithms for inference now allow efficient computation of these approximations [2][34,36].

3.1 Generalized Belief Propagation

Generalized Belief Propagation(GBP) is a message passing based algorithm that approximates the true marginals. As the name suggests, it is a generalization of the famous Belief Propagation(BP) algorithm, due to Pearl, and differs from the latter in the size of its regions that estimate the Free Energy. While BP attempts to find a fixed point of the Bethe approximation to the free energy mentioned above, GBP computes fixed points of the more general region based free energy.

There are many variants of GBP; we focus on the so called *Two-Way* [36] algorithm since it naturally extends BP. The algorithm can be viewed as running

[2] The free energy is often referred to as the "Energy Functional" in this literature.

BP on the region graph, with one crucial difference in the messages – since the same node can appear in multiple regions, its contribution to each region must be weighed in such a way as to ensure it is counted only once. This is done, by first defining the "pseudo" messages for a region R with parents $\mathcal{P}(R)$ and children $\mathcal{C}(R)$

$$n^0_{R \to P}(\mathbf{x_r}) = \tilde{f}_R(\mathbf{x_R}) \prod_{P' \in \mathcal{P}(R) \setminus P} m_{P' \to R}(\mathbf{x_r}) \prod_{C \in \mathcal{C}(R)} n_{C \to R}(\mathbf{x_C})$$

$$m^0_{R \to C}(\mathbf{x_C}) = \sum_{x_R \setminus x_C} \tilde{f}_R(x_R) \prod_{P \in \mathcal{P}(R)} m_{P \to R}(\mathbf{x_R}) \prod_{C' \in \mathcal{C}(R) \setminus C} n_{C' \to R}(\mathbf{x_{C'}}),$$

where $\tilde{f}_R(\mathbf{x_R}) = (\prod_{a \in A_r} f_a(\mathbf{x_a}))^{w_R}$ and then compensating for overcounting by defining the actual messages as

$$n_{R \to P}(\mathbf{x_r}) = (n^0_{R \to P}(\mathbf{x_r}))^{\beta_R} (m^0_{R \to C}(\mathbf{x_C}))^{\beta_R - 1}$$

$$m_{P \to R}(\mathbf{x_r}) = (n^0_{R \to P}(\mathbf{x_r}))^{\beta_R - 1} (m^0_{R \to C}(\mathbf{x_C}))^{\beta_R}$$

where w_R is the weight given to region R, p_R the number of parents of region R, and $\beta_R = p_R/(2p_R + w_R - 1)$. The beliefs at R, are then given by

$$b_R(x_R) = \tilde{f}_R(\mathbf{x_R}) \prod_{C \in \mathcal{C}(R)} n_{C \to R}(\mathbf{x_C}) \prod_{P \in \mathcal{P}(R)} m_{P \to R}(\mathbf{x_P})$$

Note that if $\beta_R = 1$, this algorithm becomes equivalent to running BP directly on the region graph.

The algorithm is typically started with randomly initialized messages and run until the beliefs converge. If it does converge, GBP is guaranteed to find a fixed point of the region based free energy. While convergence isn't guaranteed, in practice, it has been found to converge successfully in many cases, even when BP doesn't [33,35].

3.2 Related Work

Probabilistic graphical models have been used to address a number of problems in structural biology, primarily in the area of secondary structure prediction (e.g., [8]). Applications of graphical models to tertiary structure are generally limited to applications of Hidden Markov Models (HMMs) (e.g., [10]). HMMs make severe independence assumptions to allow for efficient learning and inference, the result of which is that long-range interactions cannot be modeled. Long-range interactions are, of course, found in all protein structures. Our method models these long range interactions. Graphical models have also been used in the area of fold recognition/threading [17]. An important difference between threading and our work is that we model every atom in the structure, while threading is generally performed over reduced representations.

We focussed on the problem of computing entropy using marginal probabilities for the unobserved variables, $\mathbf{X_s}$. This however isn't the only interesting inference problem. If our task was to find the *single* most likely structure, the problem reduces to Side Chain Placement. Indeed, one of the recent approaches to this problem of placing side chains [32] can be viewed as a variant of the Junction Tree algorithm for computing the most likely estimate.

It must be noted that our model is essentially similar to that of [33]. While they use it in a study to evaluate inference algorithms and perform Side Chain Placement, our task is to use it to obtain entropy and free energy estimates.

Recent work [20] has shown that most message passing algorithms can be viewed as minimizing the divergence between the actual probability distribution and a family of distributions suitably parametrized. The different algorithms differ in their choice of the divergence measure and their parametrization of the family of distributions. The pioneering work of [14,15] computes estimates using a sampling scheme which can be computationally expensive, while [12] attempts to solve the same problem using a mean-field approach. Mean field methods minimize the Kullback-Leibler Divergence while Generalized Belief Propagation (and BP) minimize an "inclusive" divergence. While the former is more accurate at capturing the zeros of the actual distribution, the latter performs better at predicting marginals. As we have shown in this section, marginal probabilities allow us to compute estimates of the entropy and free energy of the distribution. Thus, Generalized Belief Propagation is more suitable for the problem at hand.

4 Implementation and Results

We implemented the *Two-way* GBP algorithm described earlier, to compute region graph estimates of free energy and entropy. We parsed the pdb files using the pdb parser in the Molecular Biology Toolkit [21]. We then created the factor graph by computing interatomic distances and creating a factor between residues if the C_α distance between them was lesser than a threshold value. This threshold is largely dictated by the sensitivity of the energy function. For the energy terms we used, we found a threshold of 8.0 Å to be adequate. In the few datasets that we tested, our results were not affected by small changes in this threshold. We used the backbone dependent library provided by [6] and a linear approximation to the repulsive van der Waals force used by [6,33]. Each rotamer in the library also had an associated apriori probability which we incorporated into the factor as a prior. We set the temperature of the system to be 300K, which corresponds to normal room temperature.

We used a region graph construction which created two levels of regions. The top level contained "big" regions – regions with more than one variable – while the lower level contained regions representing single variables. Since we expect the interaction between residues closest in sequence to be very strong, we placed all factors and nodes between residues within two sequence positions of each other in one region. Each of the rest of the factors, representing edges between residues connected in space, formed "big" regions with two nodes in them. Thus, in the example shown in Fig. 2, $(X_s^1, X_S^2, X_S^3, f_1, f_2, f_3, f_{12}, f_{23})$, $(X_s^2, X_S^3, X_S^4, f_2, f_3, f_4, f_{23}, f_{34})$ and (X_s^1, X_s^7, f_{17}) would be examples of big regions which appear in the top level, while (X_s^1) would be an example of a small region in the lower level. Finally, we add edges from "big" regions to all small regions that contain a strict subset of the "big" region's nodes. In our example, the region encompassing X_s^1, X_s^2, X_s^3 would thus be connected to the small regions corresponding to each of X_s^1,X_s^2, and X_s^3.

Since the region graph formalism is very flexible, other equally valid alternatives for creating the graph exist. The best choice of regions will largely depend on the application at hand and the computational constraints. Our choice of regions reflects a balance between accuracy and running time by focussing on residues which are expected to be closely coupled together and placing them in bigger regions. [1] studies this class of region graphs in more detail.

We initialized the GBP messages to random starting points and ran until beliefs converged or a maximum number of iterations was reached. It must be noted that we did not have any problems with convergence: the beliefs converged in all cases.

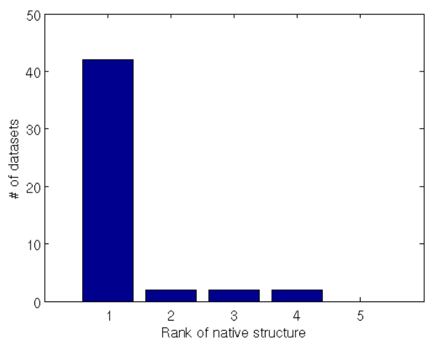
We ran our program on datasets obtained from the "Decoys R Us" database[26]. We used the immunoglobin datasets from the "multiple decoy sets". Each such dataset consisted of multiple decoy structures along with the native structure of a protein. We selected immunoglobin because it had a large number of decoys close to the native structure and has been used extensively to test methods for decoy detection[28].

Under our assumption of a rigid backbone, our estimates of entropy of different structures will be comparable only when the other sources of entropy are largely similar. Thus, our estimates will be most relevant only when the structures have largely similar backbones. To ensure that we didn't have backbones very different from the native structure among our decoys, we removed all decoys with a C_α RMSD greater than 2.0 Å to the native structure, from each dataset. We then removed any dataset that ended up with less than 5 decoys so that we didn't end up with too few decoys in a dataset. We also removed three datasets which had missing backbone atoms. At the end of this pruning, there were 48 datasets left with an average of around 35 decoys per data set.

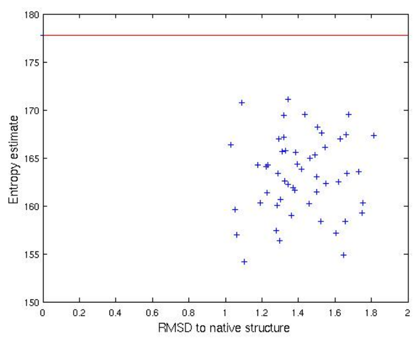
Fig. 3 shows our results on the immunoglobin dataset. When we ranked the structures in the decreasing order of their entropy, the native structure ended up at the top in 42 of the 48 datasets (87.5%). In no dataset was the native structure ranked higher than 4. Fig. 3(b) shows the scatter plot of the entropy estimates for a dataset where the native structure(3hfm) has the highest entropy.

To study the structures further, we ran PROCHECK[13] – a program for structure validation that runs a suite of structural tests. PROCHECK reported a very high number of main chain bond angles (nearly 13 angles on an average) as "off graph" – bond angles so far away from the mean that they don't show up on the output plots of PROCHECK – for the four native structures which have a rank three or four.

For example, a total of 27 angles were determined to be "off graph" for 1igc. In contrast, there were an average of around 2 such angles, among the rest of the structures. However, not all datasets in which the native structure had bad main



(a)



(b)

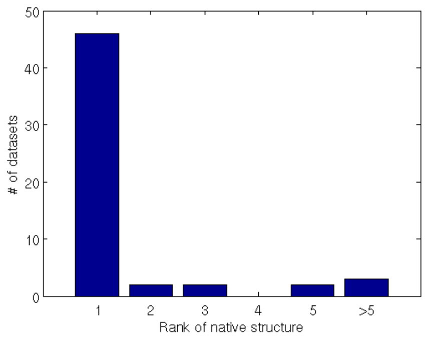
Fig. 3. (a) Histogram shows the distribution of the rank of the native structure, when ranked in decreasing order of entropy for the culled immunoglobin decoy dataset. Over this dataset, the native structure has the highest entropy 87.5% of the time(b) Entropy estimates for 3hfm and its decoys, with the value of the entropy along the Y-axis and the rmsd to native structure along the X-axis. The horizontal line indicates the value of the entropy of the native structure; all other structures have a lower entropy in this dataset.

chain bond angles had a decoy as the best rank. 1jel, for example, had 21 main chain bond angles "off graph" and yet had the best rank among its dataset. This is not unexpected, since the rank of the native structure is not only determined by its quality, but also by the quality of the decoys. Thus, our results seem to be affected, but not solely determined, by unusual main chain conformations.

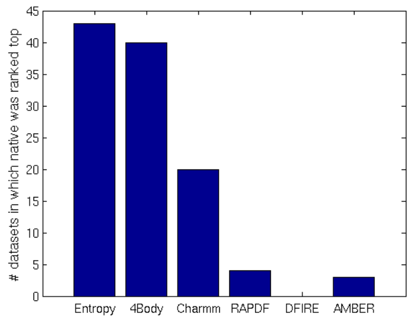
Since the structures have very similar backbones, we expect that the entropic contributions from the backbone atoms and our entropy estimates to be most meaningful in relative *order* and *magnitude*. However, in order to test the efficacy of these estimates in decoy detection, we repeated our experiments on the entire immunoglobin dataset. Our hope is that while the magnitudes of the entropy estimates might not be meaningful, the relative order of the native structure will still be useful.

Fig. 4(a) shows the results of our experiments on the entire immunoglobin dataset. As can be seen, despite the addition of the dissimilar backbones, the ranking of the native structure isn't affected much – in 84% of the datasets, the native structure has the highest entropy. We then compare our results to the following different energy functions as reported in [28]: a four body statistical potential("4body") developed in [28], the coulombic part of the CHARMM19 forcefield [5], "RAPDF" [27], "DFIRE" [37] and the sum of vanderwal and coulombic terms of the AMBER force field [31]. These energy functions are described in detail in [28].

It must be noted that "4body" has a distance parameter; the numbers reported are the best results obtained across different values of this parameter.



(a)



(b)

Fig. 4. (a) Histogram showing the distribution of the rank of the native structure. (b) Comparison of Results using various energy functions as reported in [28], along with rankings based on our Entropy estimates. These results are on the 51 Immunoglobin datasets for which data was available, including decoys with RMSD greater than 2.0 Å. Overall, the entropy estimates outperform all energy functions.

In contrast, the temperature T, which is the only tunable parameter in our approach, was set to room temperature. Yet, our entropy estimates calculated using a simple linear potential function, marginally outperforms "4body" and significantly outperforms all the other pairwise energy terms on this dataset.

Thus these results show that our entropy estimates are very successful in detecting the native structure from a set of decoys. However, they do not provide any evidence about the relative magnitude of these estimates. To test this, we perform a different experiment. We compare experimentally determined values of difference in the free energy, between the native structures of Barnase, T4 Lysozyme and Staphylococcal Nuclease (pdb ids: 1BNI, 1L63 and 1STN respectively) and their multiple single point mutants selected from the ProTherm database[19], with corresponding estimates obtained using GBP. Only mutations in buried positions were considered in order to minimize the effects of the solvent. All the $\Delta\Delta G$ experiments in a single dataset were conducted at the same pH value.

Since these mutants have different sequences, the free energy of the denatured state has to be estimated along with that of the crystal structure, in order to estimate $\Delta\Delta G$ values. We estimate the free energy of the denatured state by computing the free energy of the system before inference. Fig. 5 shows our results on the three datasets. The correlation coefficient between our estimates of $\Delta\Delta G$ and the experimentally determined values varied from 0.63 to 0.70 with p values between $1.5*10^{-5}$ to 0.0063. This compares favorably with the estimates – correlations between 0.7 and 0.94 – obtained using the four body potential of [7] over all their (much smaller) datasets. This gives evidence that our estimates predict the relative magnitude of $\Delta\Delta G$ with reasonable accuracy.

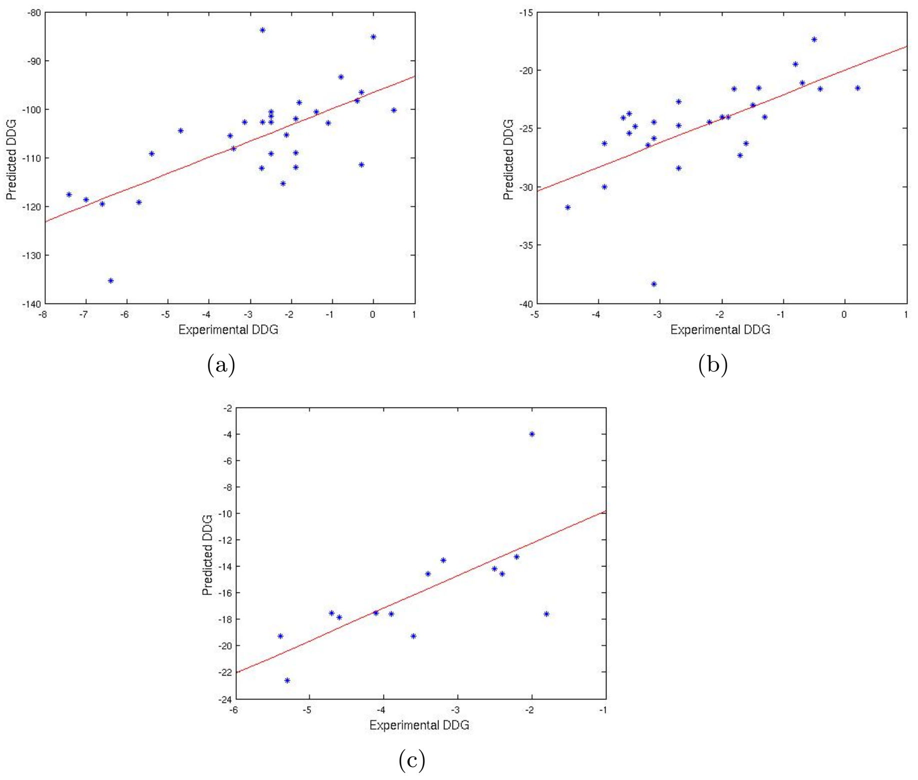


Fig. 5. Plots showing variation of experimental $\Delta\Delta G$ (on the X-axis) with computed estimates of $\Delta\Delta G$, along with a least squares fit for (a) thirty one mutants of barnase (pdb id: 1BNI), R=0.70, p=$1.5*10^{-5}$ (b) twenty eight mutants of T4 Lysozyme(pdb id:1L63), R=0.63, p=$3.0*10^{-4}$ and (c) fourteen mutants of staphylococcal nuclease (pdb id:1STN), R=0.69, p=0.0063

5 Conclusion

We have shown that free energy calculations for all-atom models of protein structures can be computed efficiently using Generalized Belief Propagation. Moreover, these estimates are sufficiently accurate to perform non-trivial tasks. We first demonstrated that it is possible to identify native immunoglobin structure from a set of decoys, with high accuracy, by comparing the computed entropies. We then demonstrated that our $\Delta\Delta G$ predictions for a set of mutations achieved high linear correlations with experimentally measured quantities. This suggests that our predictions are not only in the right relative order, but also have approximately the right relative magnitudes.

Our results have implications for a number of problem domains. First, we believe that our method could be used in the contexts of protein structure prediction and comparative modeling. Our decoy-detection results suggest that our method could be used in conjunction with protein structure prediction programs that produce multiple putative folds, like ROSETTA [25]. The accuracy of existing homology modeling methods is acknowledged to be an important issue in structural biology (e.g., [18,9]). We are presently extending our technique to allow backbone flexibility. This would facilitate refining of homology models towards a lower free-energy configuration, and potentially higher accuracy. Second, we note that one of the advantages of a graphical model is that it is easily extended. For example, we could enhance our edge potentials to incorporate experimental measurements from X-ray crystallography, Nuclear Magnetic Resonance, or Cryogenic Electron microscopy. These enhancements could be very beneficial in the context of structure determination experiments where the data are sparse or low-resolution. Third, we can also extend our model to include ligands by adding nodes to our graph. This, plus a combination of a backbone flexibility and a somewhat more sophisticated energy term may lead to more accurate $\Delta\Delta G$ calculations which, in turn, may be useful in the context of ligand binding and docking studies. Finally, while our experiments assumed a known protein sequence, it is possible to simultaneously perform inference over the sequence and structure, leading to new techniques for performing protein design. We are actively pursuing these goals as part of ongoing research into the application of graphical models to protein structures.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. We also thank K. Arun and Jianhua Shao of the Langmead lab and all members of the CBML group for their suggestions. This research was supported by a U.S. Department of Energy Career Award (DE-FG02-05ER25696), and a Pittsburgh Life-Sciences Greenhouse Young Pioneer Award to C.J.L. E.P.X. is supported by the Pennsylvania Department of Health's Health Research Program under Grant No. 2001NF-Cancer Health Research Grant ME-01-739 and by a NSF CAREER Award under Grant No. DBI-0546594.

References

1. S.M. Aji and R.J. McEliece. The generalized distributive law and free energy minimization. *Proceedings of the 39th Allerton Conference on Communication, Control and Computing*, pages 459–467, 2003.
2. H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, and P.E. Bourne. The Protein Data Bank. *Nucl. Acids Res.*, 28:235–242, 2000.
3. M.R. Betancourt and D. Thirumalai. Pair potentials for protein folding: choice of reference states and sensitivity of predicted native states to variations in the interaction schemes. *Protein Science*, 8:361–369, 1999.
4. H.A. Bethe. Statistical theory of superlattices. *Proc. Roy. Soc. London A*, 150:552–575, 1935.

5. B.R. Brooks, B.D. Bruccoleri, D.J. Olafson, S. States, S. Swaminathan, and M. Karplus. Charmm: A program for macromolecular energy minimization and dynamics calculations. *Journal of Comp. Chem.*, 4:187–217, 1983.
6. A. Canutescu, A.A. Shelenkov, and R.L. Dunbrack Jr. A graph theory algorithm for protein side-chain prediction. *Protein Science*, 12:2001–2014, 2003.
7. C.W. Carter Jr, B.C. LeFebvre, S.A. Cammer, A. Tropsha, and M.H. Edgell. Four-body potentials reveal protein-specific correlations to stability changes caused by hydrophobic core mutations. *Journal of Mol. Bio*, 311:625–38, 2001.
8. W. Chu, Z. Ghahramani, and D. Wild. A graphical model for protein secondary structure prediction. *Proc. 21st Ann. Intl. Conf. on Machine Learning (ICML) Banff, Canada)*, 2004.
9. Protein Structure Initiative. Report on the nigms workshop on high accuracy comparative modeling.
http://archive.nigms.nih.gov/psi/reports/comparative_modeling.html, 2003.
10. K. Karplus, R. Karchin, J. Draper, J. Casper, Y. (Mandel-Gutfreund), M. Diekhans, and R. Hughey. Combining local-structure, fold-recognition, and new-fold methods for protein structure prediction. *Proteins*, 53:491–6, 2003.
11. R. Kikuchi. A theory of cooperative phenomena. *Phys. Rev*, 81:988–1003, 1951.
12. P. Koehl and M. Delarue. Application of a self-consistent mean field theory to predict protein side-chains conformation and estimate their conformational entropy. *Journal of Mol. Bio.*, pages 249–275, 1994.
13. R.A. Laskowski, M.W. MacArthur, D.S. Moss, and J.M. Thornton. PROCHECK: a program to check the stereochemical quality of protein structures. *J. Appl. Cryst.*, 26:283–291, 1993.
14. C. Lee. Predicting protein mutant energetics by self-consistent ensemble optimization. *Journal of Mol. Bio.*, 236:918–939, 1994.
15. C. Lee and M. Levitt. Accurate prediction of the stability and activity effects of site-directed mutagenesis on a protein core. *Nature*, 352:448–451, 1991.
16. R. Lilien, B. Stevens, A. Anderson, and B.R. Donald. A Novel Ensemble-Based Scoring and Search Algorithm for Protein Redesign, and its Application to Modify the Substrate Specificity of the Gramicidin Synthetase A Phenylalanine Adenylation Enzyme. *J. Comp Biol*, 12(6-7):740–761, 2005.
17. Y. Liu, J. Carbonell, P. Weigele, and V. Gopalakrishna. Segmentation conditional random fields (SCRFs): A new approach for protein fold recognition. *Proc. of the 9th Ann. Intl. Conf. on Comput. Biol. (RECOMB) Boston, MA, May 14-18)*, pages 408–422, 2005.
18. M.A. Marti-Renom, A. Stuart, A. Fiser, R. Sanchez, F. Melo, and A. Sali. Comparative Protien Structure Modeling of Genes and Genomes. *Ann Rev Biophys Biomol Struct*, 29:291–325, 2000.
19. Kumar MD, Bava KA, Gromiha MM, Parabakaran P, Kitajima K, Uedaira H, and Sarai A. ProTherm and ProNIT: thermodynamic databases for proteins and protein-nucleic acid interactions. *Nuleic Acids Res.*, 34:D204-6, Database issue, 2006.
20. T. Minka. Divergence measures and message passing. *Microsoft Technical Report*, 2005.
21. J.L. Moreland, A. Gramada, O.V. Buzko, Qing Zhang, and P.E. Bourne. The molecular biology toolkit (MBT): A modular platform for developing molecular visualization applications. *BMC Bioinformatics*, 6, 2005.
22. T. Morita. Cluster variation method for non-uniform ising and heisenberg models and spin-pair correlation function. *Prog. Theor. Phys.*, 85:243 – 255, 1991.

23. T. Morita, T.M. Suzuki, K. Wada, and M. Kaburagi. Foundations and applications of cluster variation method and path probability method. *Prog. Theor. Phys. Supplement*, 115, 1994.
24. A. Pelizzola. Cluster variation method in statistical physics and probabilistic graphical models. *J. phys. A : math. gen*, 38:R309–R339, 2005.
25. C.A. Rohl, C.E. Strauss, K.M. Misura, and D. Baker. Protein structure prediction using Rosetta. *Methods Enzymol*, 383:66–93, 2004.
26. R. Samudrala. Decoys 'R' Us. http://dd.compbio.washington.edu/.
27. R. Samudrala and J. Moult. An all-atom distance-dependent conditional probability discriminatory function for protein structure prediction. *J. Mol. Biol.*, 275:895916, 1998.
28. C.M. Summa, M. Levitt, and W.F. Degrado. An atomic environment potential for use in protein structure prediction. *Journal of Mol. Bio.*, 352:986–1001, 2005.
29. P.D. Thomas and K.A. Dill. Statistical potentials extracted from protein structures: How accurate are they? *Journal of Mol. Bio.*, 257:457–469, 1994.
30. R. Tobi, D. Elber. Distance-dependent, pair potential for protein folding: Results from linear optimization. *Proteins: Structure, Function and Genetics*, 41:40–46, 2000.
31. S.J. Weiner, P.A. Kollman, D.A. Case, U.C. Singh, C. Ghio, and G. et al Alagona. A new force field for molecular mechanical simulation of nucleic acids and proteins. *Journal of Am. Chem. Soc.*, 106:765–794, 1984.
32. J. Xu. Rapid protein side-chain packing via tree decomposition. *Research in Computational Molecular Biology, Lecture Notes in Computer Science*, 3500:423–439, 2005.
33. C. Yanover and Y. Weiss. Approximate inference and protein folding. *Proceedings of NIPS 2002*, pages 84–86, 2002.
34. J.S. Yedidia, W.T. Freeman, and Y. Weiss. Generalized Belief Propagation. *Advances in Neural Information Processing Systems (NIPS)*, 13:689–695, 2000.
35. J.S. Yedidia, W.T. Freeman, and Y. Weiss. Characterizing belief propagation and its generalizations. http://www.merl.com/reports/TR2002-35/, 2002.
36. J.S. Yedidia, W.T. Freeman, and Weiss Y. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51:2282–2312, 2005.
37. H. Zhou and Y. Zhou. Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. *Protein Sci*, 11:27142726, 2002.

Minimizing and Learning Energy Functions for Side-Chain Prediction

Chen Yanover[1], Ora Schueler-Furman[2], and Yair Weiss[1]

[1] School of Computer Science and Engineering,
The Hebrew University of Jerusalem, 91904 Jerusalem, Israel
{cheny,yweiss}@cs.huji.ac.il
[2] Department of Molecular Genetics and Biotechnology, Hadassah Medical School
The Hebrew University of Jerusalem, 91120 Jerusalem, Israel
oraf@ekmd.huji.ac.il

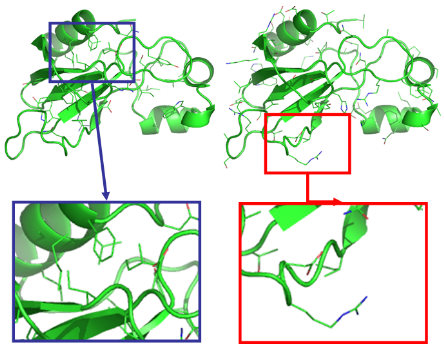
Abstract. Side-chain prediction is an important subproblem of the general protein folding problem. Despite much progress in side-chain prediction, performance is far from satisfactory. As an example, the ROSETTA program that uses simulated annealing to select the minimum energy conformations, correctly predicts the first two side-chain angles for approximately 72% of the buried residues in a standard data set. Is further improvement more likely to come from better search methods, or from better energy functions? Given that exact minimization of the energy is NP hard, it is difficult to get a systematic answer to this question.

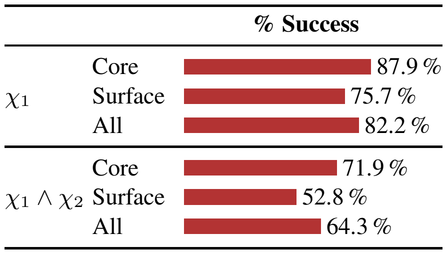
In this paper, we present a novel search method and a novel method for learning energy functions from training data that are both based on Tree Reweighted Belief Propagation (TRBP). We find that TRBP can find the *global* optimum of the ROSETTA energy function in a few minutes of computation for approximately 85% of the proteins in a standard benchmark set. TRBP can also effectively bound the partition function which enables using the Conditional Random Fields (CRF) framework for learning.

Interestingly, finding the global minimum does not significantly improve side-chain prediction for an energy function based on ROSETTA's default energy terms (less than 0.1%), while learning new weights gives a significant boost from 72% to 78%. Using a recently modified ROSETTA energy function with a softer Lennard-Jones repulsive term, the global optimum does improve prediction accuracy from 77% to 78%. Here again, learning new weights improves side-chain modeling even further to 80%. Finally, the highest accuracy (82.6%) is obtained using an extended rotamer library and CRF learned weights. Our results suggest that combining machine learning with approximate inference can improve the state-of-the-art in side-chain prediction.

1 Introduction

Proteins are chains of *residues*, each containing one of 20 possible *amino acids*. All amino acids are connected together by a common backbone structure, onto which amino-specific *side-chains* are attached. The 3-dimensional structure of a protein can thus be fully defined by the dihedral angles that specify the backbone conformation on the one hand (ϕ, ψ and ω angles), and the side-chain conformations on the other hand (up to 4 dihedral angles, denoted χ_1 to χ_4).


(a)



(b)

Fig. 1. (a) Buried and exposed residues of Barnase (PDB code 1brn). The challenge in side-chain prediction is to locate the native side-chain conformation (sticks), starting from the protein backbone (depicted as a cartoon), and its amino acid sequence. Blowups for specific regions are shown for buried residues (left) and exposed residues (right). Note that due to packing, core residues are significantly more constrained than their exposed counterparts. **(b)** Success rate of the state-of-the-art ROSETTA package using default parameters. It can be seen that even for the core, the fraction of residues for which either the χ_1 or χ_2 angles are incorrectly modeled is about 30%. Is improvement more likely to come from better search methods or from better energy functions?

The problem of predicting the residue side-chain conformations given a backbone structure is considered of central importance in protein-folding and molecular design and has been tackled extensively using a wide variety of methods (for a recent review, see [1]). The typical way to predict side-chain configurations is to define an energy function and a discrete set of possible side-chain conformations, and then search for the minimal energy configuration.

Despite much progress, the performance of side-chain prediction is far from satisfactory. To illustrate the state-of-the-art, Figure 1b shows the results of the ROSETTA package [2] on a standard benchmark set. The prediction success is typically reported separately for core residues and surface residues, since core residues are much more tightly constrained (see Figure 1a). ROSETTA uses an elaborate energy function for side-chain modeling that contains 8 energy terms. Simulated annealing is used to search for the minimal energy configuration. As can be seen, the success rate for the first two angles is around 72% for core residues and 53% for surface residues. Thus even for the better constrained residues, the prediction is wrong for almost one third of the residues.

One can think of two different approaches to improve this performance: (1) using a better optimization algorithm to find a lower energy conformation; and (2) changing the energy function. Deciding between these two approaches is currently difficult because simulated annealing and many of the other minimizers used in side-chain prediction are only guaranteed to find *local* minima of the energy function. We therefore do not know if a better optimizer would find a better solution.

Obviously a method that can find the *global* optimum of the energy function could shed light on this question. Unfortunately, it has been shown that for energy functions typically used in side-chain prediction, finding the global optimum is NP complete [3]. While this makes it extremely unlikely that we will be able to find the global optimum for *all* proteins in polynomial time, it leaves open the option for finding the global optimum for *some* proteins. Indeed, methods such as dead-end-elimination (DEE) [4,5,6] and linear programming relaxations [7] have been shown to find the global optimum for simple energy functions in side-chain prediction. However, as reported in [7], these techniques do not work well for more complicated energy functions and to the best of our knowledge, no one has successfully found the global optimum for the elaborate ROSETTA energy function.

In this paper, we present a novel search method and a novel method for learning energy functions from training data that are both based on Tree Reweighted Belief Propagation (TRBP). We find that TRBP can find the *global* optimum of the ROSETTA energy function in a few minutes of computation for approximately 85% of the proteins in a standard benchmark set. TRBP can also effectively bound the partition function which enables using the Conditional Random Fields (CRF) framework for learning of better energy functions.

Interestingly, finding the global minimum does not significantly improve side chain prediction for an energy function based on ROSETTA's default energy terms (less than 0.1%), while learning new weights gives a significant boost from 72% to 78%. A recent modification of the ROSETTA energy function is aimed at optimal side-chain modeling and uses a softer van der Waals term [8]. This energy function yields significantly better results than ROSETTA's default parameters (77% with simulated annealing). In this case, the global optimum improves prediction accuracy by 1.2%. Learning new weights again improves side-chain modeling, to 80%. Finally, not unexpectedly, the use of extended rotamer libraries improves modeling: combined with CRF learned weights it yields the highest accuracy (82.6%). Our results suggest that combining machine learning with approximate inference can improve the state-of-the-art in side-chain prediction.

2 Side-Chain Prediction

The input to the side-chain prediction task, which we will denote by y, is a list of amino-acids that make up the protein as well as the three-dimensional shape of the backbone. The output, which we will denote by x, is up to 4 dihedral angles, denoted χ_1 to χ_4, for each amino acid. In principle, the output is a continuous valued vector whose length is 4 times the number of amino acids in the protein. However, the common practice is to discretize the output space into a small number of possible angles. These discrete angles (usually up to 3 possibilities per angle) define a discrete set of possible side-chain configurations called *rotamers* [9]. Side-chain prediction thus becomes a discrete optimization problem:

$$x^* = \arg\min_{x \in \mathcal{R}} E(x, y) \tag{1}$$

where $\mathcal{R}$ is the discrete set of rotamer configurations, and the energy function $E(x, y)$ is, typically, defined in terms of pairwise interactions among nearby residues and interactions between a residue and the backbone. Approaches to side-chain prediction differ in their choices of energy functions and search methods.

Search Methods. Although the minimization problem for side-chain prediction has been shown to be NP hard [3], recent years have shown significant progress in search methods. Simulated annealing with Monte Carlo sampling used in Rosetta is a fast and efficient method to locate energy minima, but is not guaranteed to find the *global* minimum energy conformation.

The dead end elimination (DEE) algorithm is an exhaustive search algorithm that tries to reduce the search space as much as possible. It is based on a simple condition that identifies rotamers that cannot be members of the global minimum energy conformation [4,5,6]. In cases where enough rotamers can be eliminated, the *global minimum energy conformation* can be found by an exhaustive search of the remaining rotamers.

Kingsford *et al.* [7] used the method of *Linear Programming (LP) Relaxation* to locate the global optimum. They rewrote equation (1) as an integer program and then relaxed the integer constraints to obtain a linear program. They found that for an energy function similar to SCWRL [1], the LP solution was almost always integral, meaning that the LP relaxation found the *global minimum*. However, once they added a second energy term, the percentage of problems for which LP found an integer solution dropped dramatically. They also discussed using a commercial Integer-Programming package (CPLEX) and found it could work on the two-term energy functions that LP could not solve.

Energy Functions. Many of the early energy functions were primarily based on the repulsive part of the van der Waals energy term. The successful SCWRL program [1,9] approximates the repulsive portion of the 12-6 Lennard-Jones potential with a piecewise linear function. SCWRL also takes into account the prior probabilities of rotamers in a training set.

ROSETTA's energy function that is used for side-chain prediction also includes a repulsive term and prior probabilities of rotamers, but combines these with six other terms to obtain an atomic level, physically realistic energy function. Specifically it contains the following energy terms [10]:

1. The attractive portion of a 12-6 Lennard-Jones potential (herein denoted by *atr*).
2. The repulsive portion of a 12-6 Lennard-Jones potential (*rep*). This term is dampened in order to compensate for the use of a fixed backbone and rotamer set.
3. A solvation term, calculated using the model of Lazaridis and Karplus [11] (*sol*).
4. Rotamer energy: Backbone dependent internal free energies of rotamers, estimated from PDB statistics performed by Dunbrack and Karplus [9] (*dun*).
5. A hydrogen-bonding potential, dependent both on distance and angles [12]. For historical reasons, this term was divided into: **(a)** Side-chain to side-chain interactions (*hbond* 1); **(b)** Side-chain to backbone interactions (*hbond* 2); and **(c)** Backbone to backbone interactions (constant for the task of side-chain prediction).
6. A pair term that primarily reflects the electrostatic attraction and repulsion (*pair*). It describes the tendency of polar amino acid residues to contact each other, based on a statistical analysis of PDB structures of seeing two amino acids close together in space (after accounting for the intrinsic probabilities of these amino acids to be in that environment).
7. An internal term that reflects clashes within a side-chain conformation (*intra*).

The energy function, $E(x, y)$ is defined as a weighted sum of the eight terms. Denoting by λ_i the weight of the ith term, the energy is:

$$E(x, y; \lambda) = \sum_{i=1}^{8} \lambda_i E_i(x, y) \tag{2}$$

2.1 Learning Energy Functions

Most current energy functions are based on a combination of parameters that describe different aspects of a protein structure, some from physical chemistry (such as *atr* and *rep*), others from analyses of given protein structures (such as *dun*).

What is the relative importance of the different terms in the energy function? The supervised learning problem of setting the relative contribution, i.e. the *weights*, of the energy terms can be formulated as follows: given a set of training proteins $\{x_t, y_t\}_{t=1}^{T}$, where x_t is the side-chain configuration in the crystal structure of protein t and y_t denotes its backbone structure, seek parameters λ that maximize the prediction success rate. Kuhlman and Baker [2] used a conjugate gradient-based optimization method to optimize the weights of these energy terms by decreasing the energy of the native state relative to a small number of decoy configurations.

Conditional Random Fields [13] provide a principled way of learning energy functions from labeled data [14, 15, 16]. Defining the probability of the native side-chain configuration (for a given backbone structure) as:

$$\Pr(x_t|y_t; \lambda) = \frac{1}{Z_t(\lambda)} e^{-E(x_t, y_t; \lambda)} \tag{3}$$

with:

$$Z_t(\lambda) = \sum_{x \in \mathcal{R}} e^{-E(x, y_t; \lambda)} \tag{4}$$

CRFs seek to maximize the product of the probabilities $\Pr(x_t|y_t; \lambda)$ over all training proteins $\{x_t, y_t\}_{t=1}^{T}$. The term "Conditional Random Fields" comes from the fact that we are maximizing the conditional likelihood – we are not maximizing the joint probability of side-chain and backbone, but rather the conditional probability of a side-chain configuration given the backbone. CRFs have several attractive properties for learning energy functions: the conditional log likelihood is a convex function of the parameters λ and the gradient of the log likelihood is simply:

$$\frac{\partial \ln \Pr(x_t|y_z; \lambda)}{\partial \lambda_i} = -E_i(x_t) + < E_i >_\lambda \tag{5}$$

Hidden Conditional Random Fields (HCRFs) [17, 18] extend conditional random fields to settings where some of the variables are hidden. This is simply done by marginalizing out the hidden variables. In practice, a Viterbi approximation in which the marginalization is replaced with maximization, is often used [15]. This leads to maximizing:

$$\Pr(x_t|y_t; \lambda) \approx \max_h \frac{1}{Z_t(\lambda)} e^{-E(x_t, y_t, h; \lambda)} \tag{6}$$

where h are the hidden variables.

Applying the CRF framework to side-chain prediction raises a tremendous computational challenge. Note that calculating Z_t (equation (4)) requires summing over all possible side-chain configurations for a given protein. For the vast majority of proteins this summation is intractable. Similarly, calculating the gradient in equation (5) is based on taking expectations which requires a weighted sum over all possible side-chain configuration for a given protein. Finally, equation (6) requires maximizing over all possible configurations for the hidden variables. Similar computational problems arise with other supervised learning methods for learning energy functions [19, 15].

3 Tree Reweighted Belief Propagation

To summarize the results of the previous section, side-chain prediction raises major computational difficulties, either in finding the global minimum of the energy or calculating the partition function with respect to an energy function. In this work, we use tree-reweighted belief propagation (TRBP) to address both problems.

Tree-reweighted belief propagation (TRBP) is a variant of belief propagation introduced by Wainwright and colleagues [20]. We start by briefly reviewing ordinary max-product belief propagation (see e.g. [21,22]). The algorithm receives as input a graph G and the potentials Ψ_{ij}, Ψ_i. In energy minimization settings, the potentials are inversely related to the energy: $\Psi_{ij}(x_i, x_j) = e^{-E(x_i,x_j)}, \Psi_i(x_i) = e^{-E(x_i)}$. In the side-chain prediction setting the nodes of the graphs correspond to residues, and there are edges between any two residues that interact [23].

At each iteration, a node i sends a message $m_{ij}(x_j)$ to its neighbor in the graph j. The messages are updated as follows:

$$m_{ij}(x_j) \leftarrow \alpha_{ij} \max_{x_i} \Psi_{ij}(x_i, x_j)\Psi_i(x_i) \prod_{k \in N_i \setminus j} m_{ki}(x_i) \tag{7}$$

where $N_i \setminus j$ refers to all neighbors of node i except j. The constant α_{ij} is a normalization constant typically chosen so that the messages sum to one (the normalization has no influence on the final beliefs). After the messages have converged, each node can form an estimate of its local "belief" defined as:

$$b_i(x_i) \propto \prod_{j \in N_i} m_{ji}(x_i)\Psi_i(x_i) \tag{8}$$

It is easy to show that when the graph is singly-connected, choosing an assignment that maximizes the local belief will give the minimal energy configuration [22]. In fact, when the graph is a chain, equation (7) is simply a distributed computation of dynamic programming. When the graph has cycles, ordinary belief propagation (BP) is no longer guaranteed to converge, nor is there a guarantee that it can be used to find the minimal energy configuration.

In tree-reweighted BP (TRBP), the algorithm receives an additional set of input *edge appearance probabilities*, ρ_{ij}. These edge appearance probabilities are essentially free parameters of the algorithm and are derived from a distribution over spanning trees of the graph G. They represent the probability of an edge (i, j) to appear in a spanning tree under the chosen distribution. As in standard belief propagation, at each iteration a node i sends a message $m_{ij}(x_j)$ to its neighbor in the graph j. The messages are updated as follows:

$$m_{ij}(x_j) \leftarrow \alpha_{ij} \max_{x_i} \Psi_{ij}^{1/\rho_{ij}}(x_i, x_j) \Psi_i(x_i) \frac{\prod_{k \in N_i \backslash j} m_{ki}^{\rho_{ki}}(x_i)}{m_{ji}^{1-\rho_{ji}}(x_i)} \tag{9}$$

Note that for $\rho_{ij} = 1$ the algorithm reduces to standard belief propagation.

After one has found a fixed-point of these message update equations, the singleton and pairwise beliefs are defined as:

$$b_i(x_i) \propto \Psi_i(x_i) \prod_{j \in N_i} m_{ji}^{\rho_{ji}}(x_i)$$

$$b_{ij}(x_i, x_j) \propto \Psi_i(x_i) \Psi_j(x_j) \Psi_{ij}^{1/\rho_{ij}}(x_i, x_j) \cdot \frac{\prod_{k \in N_i \backslash j} m_{ki}^{\rho_{ki}}(x_i)}{m_{ji}^{1-\rho_{ji}}(x_i)} \frac{\prod_{k \in N_j \backslash i} m_{kj}^{\rho_{kj}}(x_j)}{m_{ij}^{1-\rho_{ij}}(x_j)}$$

The theoretical properties of TRBP are a subject of ongoing research [20,24,25,26]. We briefly summarize some relevant properties:

- If the TRBP beliefs contain no ties, that is for every i the maximum of $b_i(x_i)$ is attained at a unique value, then the assignment that locally maximizes the beliefs is the global minimum of the energy function.
- If the TRBP beliefs contain ties, running an additional algorithm on a problem defined only on nodes that have ties, gives an easily verified condition for the solution to be a global optimum (see [26] for details).
- Using the sum-product version of TRBP (in which the maximization in equation (9) is replaced with summation) it is possible to calculate a rigorous upper bound Z_{TRBP} on the partition function.

$$-\log Z_{TRBP} = < E >_b - \left(\sum_{ij} \rho_{ij} H(b_{ij}) + \sum_i c_i H(b_i) \right) \tag{10}$$

where $c_i = 1 - \sum_j \rho_{ij}$ and $< E >_b$ is the average energy with respect to the TRBP beliefs, and $H(b_{ij}), H(b_i)$ are the entropies of the beliefs.

We used these properties of TRBP for minimizing and learning energy functions for side-chain prediction. For minimizing energy functions, we used the max-product version of TRBP followed by post-processing as described in [26]. For learning energy functions, we replaced the partition function $Z(\lambda)$ in equations (3),(6) with the TRBP bound $Z_{TRBP}(\lambda)$. This enables us to maximize a lower bound on the probability:

$$\Pr(x_t | y_t; \lambda) = \frac{1}{Z(\lambda)} e^{-E(x_t, y_t; \lambda)} \geq \frac{1}{Z_{TRBP(\lambda)}} e^{-E(x_t, y_t; \lambda)} \tag{11}$$

We used the implementation of TRBP publically available at `www.cs.huji.ac.il/~talyam/inference.html`. The same package was also used to solve the LP relaxation, as discussed in [27].

4 Results

In the first part of this study, we evaluate whether location of the global minimum energy conformation improves side-chain modeling accuracy. We then proceed to improving the energy function by optimizing the weights of the different parameters in the energy function to maximize the probability of native side-chain conformations. We show that this improves side-chain prediction accuracy more than finding the minimum energy conformation. The next step evaluates those approaches on an additional energy function with a softer repulsive term, and finally we investigate the use of extended rotamer libraries.

Data set and Evaluation. A data set of 276 single chain proteins, up to 700 amino acids long (all in all 64,397 positions) was used for this study (taken from the *RosettaDesign* webserver [28]). We randomly selected 20% of these proteins (55 proteins, 11,067 positions) as a training set and used the remaining 80% (221 proteins, 53,330 positions) as a test set.

We define the success rate of an energy function as the percentage of side-chain angles that are predicted correctly, i.e. when the predicted angles are in the same bin as those of native side-chain conformation in the crystal (e.g. gauche+, gauche−, or trans). As widely accepted, we report the success rates for the first angle (χ_1) and the first two angles (χ_1 and χ_2) on all test set proteins. We also calculated the success rate separately for core residues, defined as residues with more than 19 interacting neighbors, and surface residues (up to 19 interacting neighbors), where residues are termed neighbors if the distance between their C_β atoms is less than $10Å$.

4.1 Location of Global Minimum Energy Configuration

Our first set of experiments was designed to measure the importance of locating the global minimum energy conformation of the energy functions currently used in side-chain prediction. We first asked which methods can find the global optimum in reasonable time? Consistent with Kingsford *et al.*'s report, the LP relaxation works well for the simple SCWRL energy function (over 90% in a database of 370 proteins) but rarely does so for the ROSETTA function (less than 5%). In other words, the LP solution is almost never integer for the ROSETTA energy functions. In contrast, the TRBP method finds the global optimum in over 80% of the proteins in our database for ROSETTA, while the commercial Integer Programming package (CPLEX) can find the minimum for all the examples in our database (although its run time is generally much larger than that of TRBP). Also consistent with the report in [7], DEE [5] never found the global optimum for these problems, indicating that not enough rotamers could be eliminated.

How much then does location of the global minimum energy conformation improve performance? Our results indicate that the improvement obtained from locating the global minimum energy (compared to simulated annealing) is negligible: side-chain modeling accuracy for the first two χ angles of core, surface and all residues are essentially unchanged (Figure 2).

Given a method that can find the global minimum energy, a better comparison of the usefulness of different energy functions for side-chain modeling can be performed:

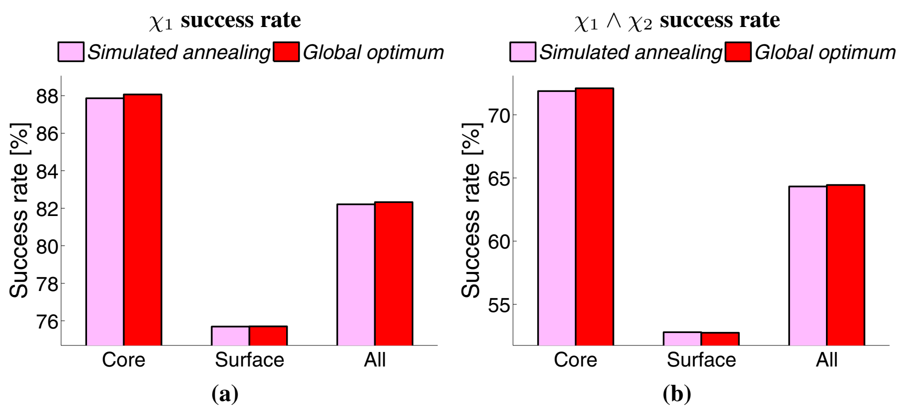


Fig. 2. The location of the global minimum energy conformation does not improve side-chain modeling for the ROSETTA original (default) energy function. The percentages of correctly predicted χ_1 side-chain angles **(a)**, and both χ_1 and χ_2 angles **(b)** are indicated for the whole set of side-chains, as well as for the buried and exposed subsets separately.

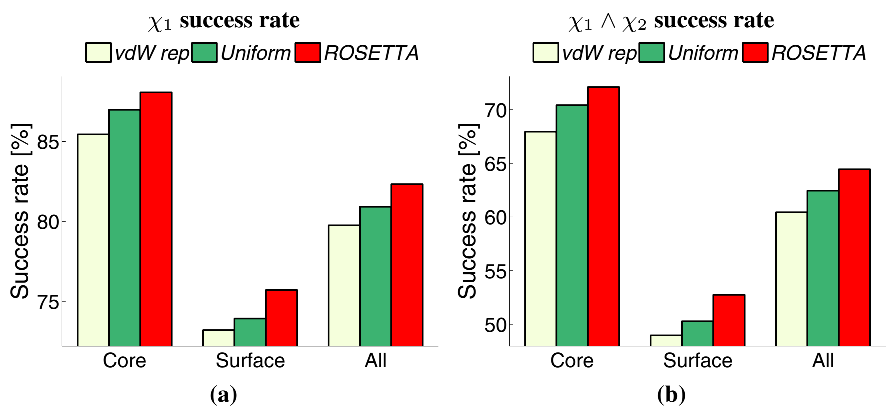
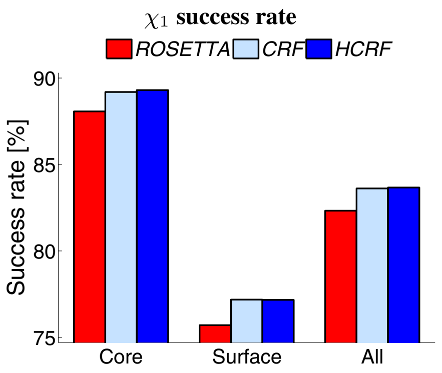


Fig. 3. Comparison of different energy functions with global minimization. Side-chain prediction success rate for energy functions that use either ROSETTA's repulsive van der Waals and rotamer energy terms (*vdw rep*) only, the full ROSETTA energy function with uniform weights, or ROSETTA's default weights.

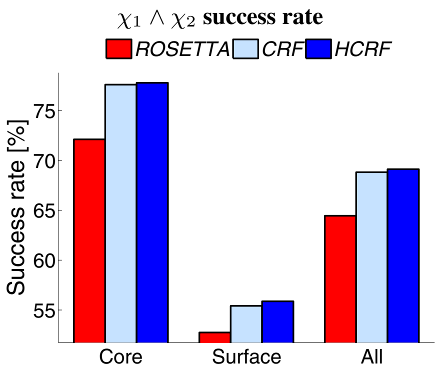
For which energy function do the global energy minima coincide best with near-native models? Figure 3 compares different energy functions defined by different weightings of ROSETTA's eight energy terms – using only the repulsive van der Waals (rep) and rotamer energy (dun) terms (which simulates the setup of SCWRL [1]), using a uniform weighting on all eight terms, and using ROSETTA's default weights. It can be seen that the van der Waals and rotamer terms on its own give the worst performance, followed by a uniform weighting of ROSETTA's eight terms and the best performance is given by ROSETTA's weights. These results are consistent with previously reported conclusions. Note however that in the present study effects due to correlation between the energy function and the search algorithm are excluded since only proven global energy minima are considered. For all three cases, we can therefore conclusively attribute the improvement in performance to a more accurate energy function.

4.2 Learning

Our second set of experiments deals with the effect of reweighting the energy terms in ROSETTA. We compared the default ROSETTA weights to those obtained using supervised learning by two learning methods: (1) the standard CRF framework – when all angles are considered observed; and (2) the Hidden CRF framework – when angles χ_3, χ_4 are considered hidden. Note that our database includes ground truth for all angles based on crystallography, but we hypothesized that due to the large variability in the angles far from the backbone, ignoring the crystallographic "ground truth" might enable better performance on the first two angles. As mentioned earlier, we used a small subset of the proteins as a training set, and report here results for the *test set*—proteins that were not seen by the learning algorithm.



(a)



(b)

Fig. 4. The success rates, obtained using CRF and HCRF learned weights compared to ROSETTA's weights. Learning gives a significant improvement in performance.

Figure 4 shows χ_1 and χ_2 success rates on the test set using ROSETTA's original weights and the weights learned by the CRF and the HCRF algorithms. Both learning algorithms improve over ROSETTA's weightings.

Note that the improvement obtained by reweighting the terms (either using CRFs or using Hidden CRFs) is far larger than that obtained by using a better minimizer. Whereas going from simulated annealing to global minimization yields less than 0.1% improvement for the first two angles in core residues, reweighting the energy terms increases performance by almost 6%.

Figure 5a shows the weights learned by CRF and HCRF compared to ROSETTA's weights. While the change in most weights is mild, the repulsive van der Waals weight almost vanishes. Note however that complete exclusion of the repulsive term from ROSETTA's default energy function significantly decreases the success rates. The reason for the significant reduction of the van der Waals repulsive term is its sensitivity to discretization. Native structures are well-packed, therefore modeling with near-to-native, discrete conformations (that is, using rotamers) can easily lead to clashes. Consequently, when optimizing an energy function that distinguishes near-native conformations from wrong conformations, the repulsive term will be down-weighted. While an energy function with low repulsive weight might be useful for selecting correct side-chain conformations from a discrete set of possible combinations, procedures that involve continuous minimization will be impeded by the missing term that contributes significantly in guiding the structure towards the correct conformation.

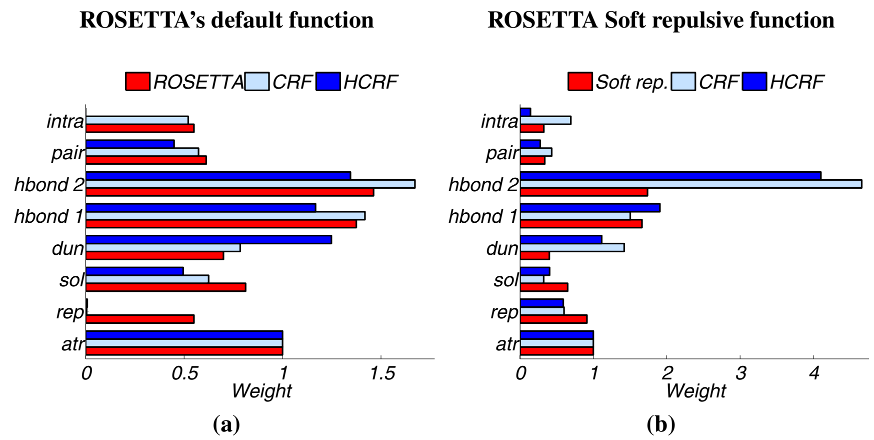


Fig. 5. ROSETTA's weights (a) and ROSETTA soft repulsive weights (b) compared to CRF and HCRF learned weights. Weights are normalized so that attractive weights equal 1.

When we analyzed the performance for different amino acids, we found that the greatest improvements were obtained on aromatic amino acids – Phenylalanine (F), Tyrosine (Y), Tryptophan (W) and Histidine (H). These bulky aromatic rings tend to clash if no extra rotamers are included in the rotamer library [29]. Since the repulsive contribution to the energy function is significantly reduced as a consequence of the low repulsive weight (Figure 5a), the selection of near-native conformations that clash with the surrounding environment – but still create favorable contacts that contribute to other terms in the energy function – is improved.

5 Results with "Soft Repulsion"

The fact that better performance can be obtained by decreasing the weight of the repulsive term has been observed previously in ROSETTA (e.g. [30]). In order to overcome this unnaturally small contribution of the repulsive part of the Lennard-Jones potential, a "dampened" version has been developed (the "-soft_rep" option, referred to as *DampRep* in [8]). In this function, the repulsive energy increases less dramatically when two atoms are brought together, and therefore, clashes are penalized less in the course of discrete optimization. This energy function was shown to allow improved side-chain modeling in ROSETTA [8]. In order to evaluate the importance of the search strategy and the energy function optimization, we conducted additional experiments based on this energy function.

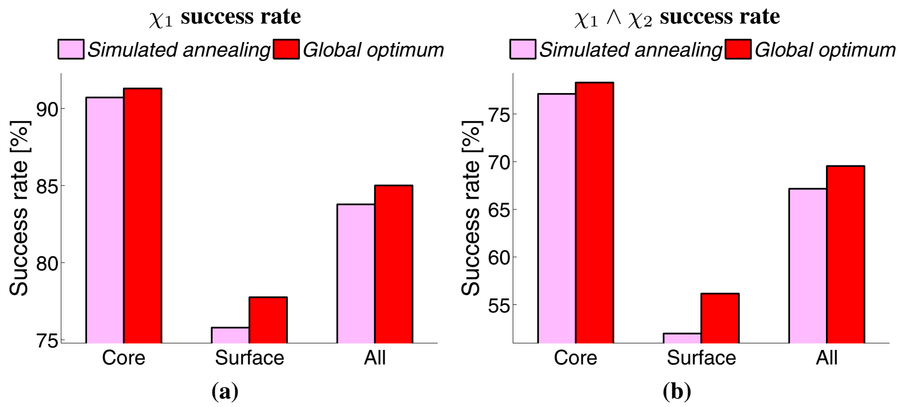


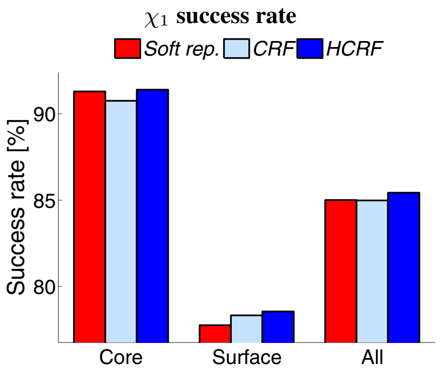
Fig. 6. The location of the global minimum energy conformation improves side-chain modeling for the ROSETTA soft repulsive energy function. Legend as in Figure 2.

We again found that TRBP can obtain the global optimum in a few minutes of computation for the majority of the proteins in our database (approximately 80%) while the LP relaxation and DEE could not. Figure 6 again shows that using the global optimizer leads to only a small improvement in prediction accuracy (approximately 1% improvement for the first two χ angles of core residues). Consistent with our earlier experiments, the gain from using a different energy function is larger than that using better minimizers – note that using simulated annealing with the "soft repulsion" energy gives better results than global optimization of the default ROSETTA function.

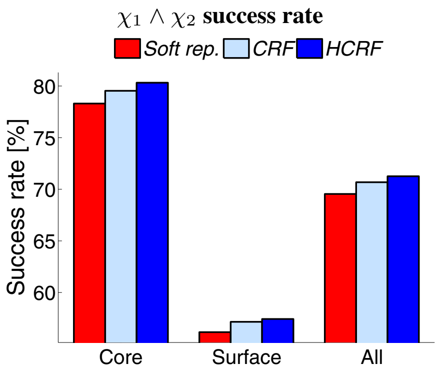
Figure 7 shows the results of applying reweighting to ROSETTA with soft repulsion. Even though this energy function had been optimized for side-chain modeling, supervised learning is able to find better reweighting of the energy terms. In particular, the new weights allow an improvement of correct modeling of $\chi 1$ and $\chi 2$ angles from 78% to 80%. Note that our test set contains approximately 32,000 residues for which both χ_1 and χ_2 are defined, so that a 2% improvement corresponds to approximately 640 residues and is highly significant. For this data set the HCRF learning criterion performed slightly better than the CRF criterion.

Figure 5b confirms that indeed, in the soft repulsive function the Lennard-Jones repulsive term is of comparable size to the Lennard-Jones attractive term. Interestingly, the contribution of hydrogen bonds is significantly increased.

Using a large rotamer library. A bottleneck in further improvement of side-chain modeling is the rotamer library from which side-chain conformations are selected. Side-chains that are not adequately represented in the library, cannot be correctly modeled. Therefore, in addition to energy functions and search methods, another direction of possible improvement is to modify the discrete set of rotamers that define the search space



(a)



(b)

Fig. 7. The success rates, obtained using CRF and HCRF learned weights compared to ROSETTA's soft repulsive weights

[31, 32]. We therefore repeated our experiments using extra rotamers to core residues, for which accurate modeling is especially important to guarantee tight packing.

The much larger number of rotamers makes minimization much slower; TRBP can still obtain the global optimum for 70%-90% of the proteins in our data set, whereas the commercial Integer Programming package (CPLEX) fails to find the optimum for many proteins due to memory limitations (even after pruning the search space using DEE).

Indeed, extended sampling improves the performance of ROSETTA's soft repulsive energy function by more than 1% even when simulated annealing is used (81.4% for $\chi_1 \wedge \chi_2$ in core positions). Using the global minimum energy configurations when available (and the configurations obtained by simulated annealing otherwise) only slightly improved accuracy (less than 0.25%). In this case, using CRF learned weights leads to only a small improvement (0.35%, to 81.9%). The highest accuracy (82.6%) is obtained with weights learned using a local HCRF variant, in which we maximize the sum of the marginal log likelihoods of the native rotamers (and treat all other positions as hidden). For speed reasons we used ordinary BP in this variant.

6 Discussion

Side-chain prediction is an important subtask of the protein folding problem and has multiple applications in linking protein structure and function. Traditionally, it has been approached by formulating energy functions over a discrete set of angles and using discrete optimization algorithms to find the minimal energy configuration. Despite much progress in search methods and energy functions, performance is far from satisfactory and it has been difficult to systematically determine whether the energy functions or the search methods are to blame. In this paper, we have shown that using tree-reweighted belief propagation (TRBP) it is possible to find the global minimum for many side-chain prediction problems in a few minutes. TRBP can also be used to bound the partition function and this is useful for learning new weights in the CRF framework. Using these computational tools we have shown that (1) global optimization tends to yield a smaller improvement in performance than adapting the energy function, and that (2) supervised learning can be used to automatically reweight the energy terms to obtain relatively large improvement in side-chain modeling. By combining our learned weights with global optimization we obtain significantly better performance on test data compared to the ROSETTA package, widely considered the state-of-the-art.

The present study suggests that supervised learning can also be used to devise novel energy terms, in addition to reweighting the existing ones. In addition, we plan to learn task-specific weights in a more general setting; for example, by focusing on interface modeling in protein-protein interactions (e.g. *docking*). We believe that the tools of approximate inference and machine learning will have great benefit in many applications of structural biology.

Acknowledgments. We are thankful for insightful discussions with Brian Kuhlman and Jim Havranek. C. Y. is supported by Yeshaya Horowitz Association through the Center for Complexity Science. Support from the Israeli Science Foundation is gratefully acknowledged.

References

1. Canutescu, A.A., Shelenkov, A.A., Dunbrack, Jr., R.L.: A graph-theory algorithm for rapid protein side-chain prediction. Protein Sci **12**(9) (2003) 2001–2014
2. Kuhlman, B., Baker, D.: Native protein sequences are close to optimal for their structures. PNAS **97**(19) (2000) 10383–10388
3. Fraenkel, A.S.: Protein folding, spin glass and computational complexity. In: Proceedings of the 3rd DIMACS Workshop on DNA Based Computers, held at the University of Pennsylvania, June 23 – 25, 1997. (1997) 175–191
4. Desmet, J., Maeyer, M.D., Hazes, B., Lasters, I.: The dead-end elmination theorem and its use in protein side-chain positioning. Nature **356** (1992) 539–542
5. Goldstein, R.F.: Efficient rotamer elimination applied to protein side-chains and related spin glasses. Biophys. J. **66**(5) (1994) 1335–1340
6. Pierce, N.A., Spriet, J.A., Desmet, J., Mayo, S.L.: Conformational splitting: A more powerful criterion for dead-end elimination. J. of Computational Chemistry **21**(11) (2000) 999–1009
7. Kingsford, C.L., Chazelle, B., Singh, M.: Solving and analyzing side-chain positioning problems using linear and integer programming. Bioinformatics **21**(7) (2005) 1028–1039
8. Dantas, G., Corrent, C., Reichow, S.L., Havranek, J.J., Eletr, Z.M., Isern, N.G., Kuhlman, B., Varani, G.Merritt, E.A., Baker, D.: High-resolution structural and thermodynamic analysis of extreme stabilization of human procarboxypeptidase by computational protein design'. Journal of Molecular Biology **In Press** (2007)
9. Dunbrack, Jr., R.L., Karplus, M.: Back-bone dependent rotamer library for proteins: Application to side-chain predicrtion. J. Mol. Biol **230**(2) (1993) 543–574
10. Rohl, C.A., Strauss, C.E.M., Chivian, D., Baker, D.: Modeling structurally variable regions in homologous proteins with Rosetta. Proteins: Structure, Function, and Bioinformatics **55**(3) (2004) 656–677
11. Lazaridis, T., Karplus, M.: Effective energy function for proteins in solution. Proteins: Structure, Function, and Genetics **35**(2) (1999) 133–152
12. Kortemme, T., Morozov, A.V., Baker, D.: An orientation-dependent hydrogen bonding potential improves prediction of specificity and structure for proteins and protein-protein complexes. Journal of Molecular Biology **326**(4) (2003) 1239–1259

13. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML, 2001. (2001) 282–289
14. Lafferty, J., Zhu, X., Liu, Y.: Kernel conditional random fields: Representation and clique selection. In: ICML. (2004)
15. LeCun, Y., Huang, F.: Loss functions for discriminative training of energy-based models. In: Proc. of the 10-th International Workshop on Artificial Intelligence and Statistics (AIStats'05). (2005)
16. Vishwanathan, S., Schraudolph, N., Schmidt, M., Murphy, K.: Accelerated training of conditional random fields with stochastic meta-descent. In: ICML. (2006)
17. Gunawardana, A., Mahajan, M., Acero, A., Platt, J.C.: Hidden conditional random fields for phone classification. In: INTERSPEECH. (2005)
18. Quattoni, A., Collins, M., Darrell, T.: Conditional random fields for object recognition. In Thrun, S., Saul, L., Schölkopf, B., eds.: Advances in Neural Information Processing Systems 17. MIT Press, Cambridge, MA (2005)
19. Taskar, B., Guestrin, C., Koller, D.: Max-margin markov networks. In Thrun, S., Saul, L., Schölkopf, B., eds.: Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, MA (2004)
20. Wainwright, M.J., Jaakkola, T., Willsky, A.S.: MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. IEEE Transactions on Information Theory **51**(11) (2005) 3697–3717
21. Yedidia, J.S., Freeman, W.T., Weiss, Y.: Understanding belief propagation and its generalizations. In: IJCAI (distinguished lecture track). (2001)
22. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann (1988)
23. Yanover, C., Weiss, Y.: Approximate inference and protein folding. Advances in Neural Information Processing Systems (2002)
24. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. In: Proceedings AI Stats. (2005)
25. Kolmogorov, V., Wainwright, M.: On the optimality of tree-reweighted max-product message passing. In: Uncertainty in Artificial Intelligence (UAI). (2005)
26. Meltzer, T., Yanover, C., Weiss, Y.: Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In: Proceedings International Conference on Computer Vision (ICCV). (2005)
27. Yanover, C., Meltzer, T., Weiss, Y.: Linear programming relaxations and belief propagation – an empirical study. Journal of Machine Learning Research **7** (2006) 1887–1907
28. Liu, Y., Kuhlman, B.: Rosettadesign server for protein design. NAR **34** (2006) W235–238
29. Wang, C., Schueler-Furman, O., Baker, D.: Improved side-chain modeling for protein-protein docking. Protein Sci **14**(5) (2005) 1328–1339
30. Gray, J.J., Moughon, S., Wang, C., Schueler-Furman, O., Kuhlman, B., Rohl, C.A., Baker, D.: Protein-protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations. Journal of Molecular Biology **331**(1) (2003) 281–299
31. Leaver-Fay, A., Kuhlman, B., Snoeyink, J.: An adaptive dynamic programming algorithm for the side chain placement problem. Pacific Symposium on Biocomputing **10** (2005) 16–27
32. Peterson, R.W., Dutton, P.L., Wand, A.J.: Improved side-chain prediction accuracy using an ab initio potential energy function and a very large rotamer library. Protein Sci **13**(3) (2004) 735–751

Protein Conformational Flexibility Analysis with Noisy Data

Anshul Nigham[1] and David Hsu[2]

[1] Singapore–MIT Alliance, Singapore 117576, Singapore
anshulni@comp.nus.edu.sg
[2] National University of Singapore, Singapore 117543, Singapore
dyhsu@comp.nus.edu.sg

Abstract. Protein conformational changes play a critical role in biological functions such as ligand-protein and protein-protein interactions. Due to the noise in structural data, determining salient conformational changes reliably and efficiently is a challenging problem. This paper presents an efficient algorithm for analyzing protein conformational changes, using noisy data. It applies a statistical flexibility test to all contiguous fragments of a protein and combines the information from these tests to compute a consensus flexibility measure for each residue of the protein. We tested the algorithm, using data from the Protein Data Bank and the Macromolecular Movements Database. The results show that our algorithm can reliably detect different types of salient conformational changes, including well-known examples such as hinge and shear, as well as the flap motion of HIV-1 protease. The software implementing our algorithm is available at http://motion.comp.nus.edu.sg/projects/proflexana/proflexana.html.

1 Introduction

Protein structural changes, called conformational changes, play a critical role in vital biological functions such as immune protection, enzymatic catalysis, and cellular locomotion [7]. An example is the "flap" motion of HIV-1 protease, a major inhibitory drug target for AIDS therapy. Conformational changes are a direct consequence of protein structural flexibility and provide insight into the essential link between structure and function.

The structures of an increasing number of proteins have been determined in multiple conformations. In the long term, one may hope to reconstruct, computationally, protein motions from multiple experimentally-determined structures. The motions can then be classified and archived, in order to better understand protein structures and their relationships with protein functions [3]. More immediately, analyses of multiple conformations can help in identifying salient conformational changes, such as hinge or loop motions, as well as in locating active sites in ligand-protein binding [22].

With these goals in mind, our work focuses on analyzing protein conformational changes, an important problem that has received much attention over the years (see Section 2). Specifically, our problem is to identify the flexible and rigid regions of a single protein, given its structure, *i.e.*, the 3D coordinates, in two

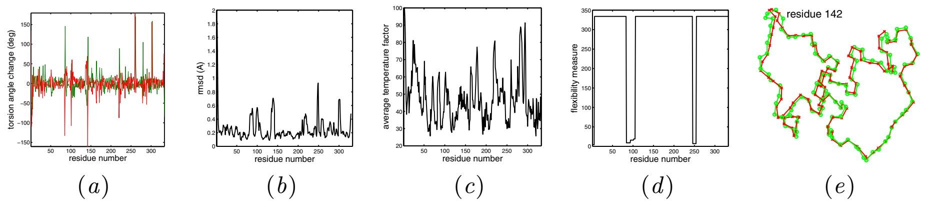


Fig. 1. Various methods for detecting flexibility in the N-lobe of lactoferrin. (a) Torsion angle differences. (b) The minimum RMSD for 5-residue fragments centered at each residue. (c) Average temperature factors from X-ray crystallography data. (d) Our new algorithm. For (a)–(c), large absolute values indicate flexible regions. For (d), small values indicate flexible regions. (e) Superimposition of the two conformations (in red and green, respectively) for the 40-residue fragment centered around residue 142.

different conformations. An example of such flexible regions is a *hinge*, a consecutive sequence of flexible residues that cause rotational motion between two rigid domains of a protein. This analysis can also be easily extended to more than two conformations through pairwise comparison, if a protein has a relatively small number of distinct conformations that are biologically relevant.

Our problem may appear easily solved by comparing backbone torsion angles, ϕ and ψ. Unfortunately, experimental data obtained through X-ray crystallography or NMR methods are *noisy*. A rigid domain may appear flexible due to noise in the data. Consider Fig. 1a, in which the peaks of the curves indicate large differences in torsion angles ϕ and ψ between two conformations of the N-lobe of lactoferrin. The N-lobe of lactoferrin is known to undergo inter-domain motion hinged around residues 90 and 250 [1]. However, the curves appear quite noisy and show many peaks in regions where there are no genuine conformational changes. For example, although there is a sharp peak at residue 142, superimposing the two conformations for the 40-residue protein fragment centered around residue 142 shows no significant conformational change (Fig. 1e). Other common methods for detecting conformation changes, such as the root-mean-square distance (RMSD) and the temperature factor, are also susceptible to noise to various degrees (see Fig. 1b–c).

Key to our problem is to distinguish genuine conformational change from noise. Our algorithm addresses this difficulty at two levels. At the low level, we have developed a reliable statistical test for determining the flexibility of a protein fragment, with noisy data. At the high level, we apply this test to all fragments of a protein and combine information from both short and long fragments to compute a consensus flexibility measure for each residue of the protein. As a result, the algorithm highlights the genuine conformational changes by suppressing the spurious ones due to noise. See Fig. 1d for an example, in which our new algorithm unambiguously detects the two main conformations changes in the N-lobe of lactoferrin, despite the noise in the data. Our algorithm takes $O(n^2)$ time for a protein with n backbone atoms. In our tests, it ran at interactive speed even for large proteins with thousands of atoms.

In the following, after a brief review of previous work (Section 2), we first describe our algorithm for protein flexibility analysis under noise (Section 3). We then give details on efficient implementation of the algorithm and provide a running time analysis (Section 4). Using data from the Protein Data Bank (PDB), we tested our algorithm on proteins that exhibit different types of conformational changes. The results show that our algorithm can reliably detect salient conformational changes (Section 5). We then highlight the main features of the algorithm and address some remaining issues (Section 6). Finally we summarize the results and point out possible future improvements (Section 7).

2 Related Work

Many approaches have been proposed to study protein conformational flexibility. At one extreme, some methods use none or a single experimentally determined protein conformation [11,14]. In particular, it has been suggested that temperature factors obtained from X-ray crystallography may be correlated with protein flexibility [25]. However, temperature factors reflect mainly the thermal motion and disorder of atoms, and are not reliable for detecting salient conformational changes (see, *e.g.*, Fig. 1*c*). At the other extreme, one may exploit the huge number of different conformations generated by molecular dynamics simulation and infer coordinated motion involving many residues of a protein [23].

Most methods, however, compare two or a small number of experimentally determined conformations, because, despite the rapid growth of protein structural data, the number of known conformations for any particular protein usually remains small. These methods differ in the similarity metric used for comparing protein conformations. They also differ in how they search for flexible and rigid regions of a protein. Below, we briefly review some of them.

Backbone torsion angles are used in several methods to determine the similarity of protein conformations [12,13,16]. As mentioned earlier, torsion angles are highly sensitive to noise: small changes in atom coordinates may cause drastic changes in torsion angles. These methods are useful, only if the noise level is extremely low. A better similarity metric makes use of the pairwise distance matrix, in which every entry is the distance between two atoms of a protein [10,18]. The most commonly used similarity metric is probably the minimum RMSD between the backbone atoms or the C_α atoms of a protein. Other related metrics have been suggested as well [2]. As shown in Fig. 1*b*, RMSD is less sensitive to noise than torsion angles, but not immune to it.

To search for flexible or rigid regions of a protein, the sieve-fit method chooses a rigid core of atoms to align two protein conformations and iteratively improves the alignment until a user-selected threshold is reached [4,15,26]. The results are sensitive to the initial choice of the rigid core. A different method uses a heuristic measure of protein flexibility, called the deformation index [10], to locate hinge regions. The fit-all method of Gerstein and Chothia [6] systematically computes the RMSD of all contiguous fragments of a protein between two conformations. It treats the resulting RMSD values as a function of two variables and uses optimization methods to search for the function's inflection points, which indicate hinge regions. However, the search may get stuck locally if not restricted to a suitable domain, which must be chosen manually.

To our knowledge, few methods systematically take into consideration noise in the data when comparing protein structures.

The problem addressed here is related to that of protein structure alignment, which tries to find structural similarities in arbitrary proteins [20,27]. Our problem is more constrained. We focus on the same protein in different conformations and require no alignment. This simplifies the problem and allows us to develop a more efficient and robust algorithm. However, an important issue common to both problems is to compare the structural similarity of protein fragments. The statistical test that we have developed is thus useful in both problems.

3 Methods

To detect protein conformational flexibility accurately and reliably under noise, we check the flexibility of all contiguous fragments of a protein between two conformations. We then extract a set of "minimal flexible fragments" and use them to compute a flexibility measure for each residue of the protein (Section 3.1). An important element of our algorithm is a statistical test for determining the flexibility of a protein fragment based on the similarity of its structures in two conformations (Section 3.2). The details are described below.

3.1 An All-Fragment Analysis of Protein Flexibility

Our algorithm aims to identify flexible and rigid regions of a protein. A flexible protein fragment changes its shape between two conformations, while a rigid fragment remains the same. To distinguish flexible and rigid fragments, we need a measure of similarity between two conformations of a protein fragment. We have chosen the minimum RMSD, a commonly used similarity metric. Intuitively, the minimum RMSD tries to superimpose two conformations of a protein fragment as well as possible, using translations and rotations.

Ideally, the minimum RMSD is 0 if the fragment is rigid and increases as the fragment becomes more flexible. With noisy data, RMSD is unlikely to be 0, even if two conformations are the same. To decide whether a fragment is flexible or not, we use a statistical test to set a threshold for the RMSD. The exact threshold values depend on the amount of noise in the data, the required confidence level, and the length of the fragment tested. In particular, the threshold values are higher for shorter fragments (see Fig. 3). It is thus more difficult to detect small conformational changes in shorter fragments. We defer the detailed discussion until Section 3.2.

Given a suitable threshold, we can compute the minimum RMSD for a fragment of a protein and test its flexibility. However, we still must choose which fragments of the protein to test. If we test short fragments and they turn out to be flexible, we can localize the flexible residues better. On the other hand, short fragments may fail to reveal small conformational changes, which could be masked as noise. We must then rely on longer fragments. To identify all flexible residues accurately and reliably, we examine *all* contiguous fragments and derive a *consistent* interpretation of the information from them. The main advantage of this approach is that it collates information from both long and short fragments and is thus more robust against noise.

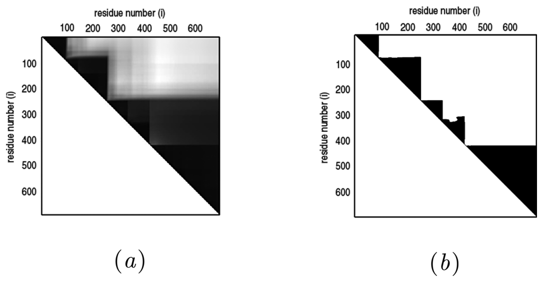


Fig. 2. (a) The RMSD matrix $\hat{\mathbf{R}}$ for lactoferrin. Darker colours indicate smaller RMSD values. (b) The corresponding matrix $\mathbf{T}$. Black indicates 0 (rigid). White indicates 1 (flexible).

RMSD matrices. We represent a protein as a sequence of backbone atoms. Let $F(i,j)$ denote the protein fragment between backbone atoms i and j. The *length* of $F(i,j)$ is the number of backbone atoms contained in it.

We start by computing the minimum RMSD of every contiguous fragment of a protein and storing the results in an upper triangular matrix $\hat{\mathbf{R}}$. For $i < j$, the entry $\hat{R}(i,j)$ of $\hat{\mathbf{R}}$ is the minimum RMSD of the fragment $F(i,j)$ between the two given conformations. For example, Fig. 2*a* shows the pseudo-colored minimum RMSD matrix for lactoferrin, with darker colors indicating smaller RMSD values. The dark triangular regions along the matrix's main diagonal correspond to relatively rigid protein fragments.

Next, we apply our statistical test (see Section 3.2) to threshold each entry of $\hat{\mathbf{R}}$ and compute a new matrix $\mathbf{T}$, which tentatively classifies every contiguous protein fragment as flexible or rigid. If $\hat{R}(i,j)$ is greater than the threshold, the entry $T(i,j)$ of $\mathbf{T}$ is 1, and the fragment $F(i,j)$ is considered flexible. Otherwise, $T(i,j)$ is 0, and $F(i,j)$ is considered rigid. See Fig. 2*b* for an example.

Minimal flexible fragments. The matrix $\mathbf{T}$ contains a wealth of information on the flexibility of a protein, but requires careful interpretation. Suppose that $T(i,j) = 1$, which indicates that the fragment $F(i,j)$ is flexible. Shall we then consider every residue within $F(i,j)$ flexible? The answer is no. Possibly, $F(i,j)$ contains two sub-fragments, one flexible and one rigid. It is thus inaccurate to declare the whole fragment flexible. Also, what shall we do if we have two overlapping fragments, both of which are classified as flexible according to $\mathbf{T}$?

To give a consistent interpretation of the information in $\mathbf{T}$, we introduce the notion of *minimal flexible fragment* (MFF). An MFF is a flexible fragment that contains no proper sub-fragment that is also flexible. In other words, all proper sub-fragments of an MFF are rigid. Two remarks can be made about an MFF $F(i,j)$. First, $F(i,j)$ is flexible, based on the evidence from data. Second, there is no further evidence to attribute the flexibility to any sub-fragment of $F(i,j)$. Therefore, an MFF identifies a flexible region of a protein as accurately as possible, given all the evidence in $\mathbf{T}$.

The definition of MFF implies that a fragment $F(i,j)$ is an MFF if and only if $T(i,j) = 1$ and $T(i',j') = 0$ for $i \le i' < j' \le j$ in the upper triangular part of **T**. This leads to an efficient dynamic programming algorithm for computing the set $\mathcal{L}$ of all MFFs (see Section 4).

The flexibility measure. The length of an MFF $F \in \mathcal{L}$ is correlated with the magnitude of conformational change. As mentioned earlier, the threshold for declaring a fragment flexible is higher for shorter fragments. Thus, small MFF length indicates that conformational changes are large, as they are detectable even in a short fragment. Such conformational changes are often observed in hinge regions, which cause large rotational motion between two rigid domains of a protein and create open and closed conformations. In comparison, large MFF length indicates that statistically significant conformational changes are only detectable in long fragments, which implies that the conformational changes are relatively small. This type of conformational changes include intra-domain motions such as "induced fit", which involves gradual, directed displacements around the binding site of a protein in order to accommodate ligand binding.

The above discussion suggests that the length of an MFF is a good indicator of conformational flexibility, and we use it assign a flexibility measure $f(i)$ to each shortest fragment $F(i, i+1)$, for $1 \leq i < n$. For a given i, let $\mathcal{L}'$ be the subset of $\mathcal{L}$ such that every fragment in $\mathcal{L}'$ contains $F(i, i+1)$ as a sub-fragment. The flexibility measure $f(i)$ for $F(i, i+1)$ is the length of the shortest fragment in $\mathcal{L}'$. Smaller f values indicate higher flexibility. If $\mathcal{L}'$ is empty, we set $f = n+1$ by convention to indicate that $F(i, i+1)$ is rigid.

In practice, we almost always use the standard kinematic model of protein motion. It assumes that bond lengths and bond angles remain fixed during conformational change. In addition, we are usually more interested in determining conformation change at the level of residues rather than atoms. Due to these restrictions, we only need to consider the fragments $F(3i, 3j)$ for $1 \leq i, j \leq n/3$ and assign the flexibility measure to $F(3i, 3i+3)$. In this case, i corresponds to the residue number, and the flexibility measure is assigned on a per residue basis. Our algorithm also applies, with little change, if only the C_α atoms, instead of all the backbone atoms, are used.

Interpretation of the results. The final output of our algorithm is a flexibility measure $f(i)$ for each residue i (see Fig. 5 for examples). For example, $f(50) = 15$ means that to detect conformational change due to residue 50, we require a fragment of at least 15 atoms. Since the length of an MFF is correlated with the magnitude of conformational change, $f(i)$ gives an indication of conformational flexibility at residue i. As another example, if $f_{90} = n+1$, where n is the length of a protein, then no MFF contains residue 90. Any flexible fragment that contains residue 90 must have sub-fragments that are also flexible. Hence the flexibility cannot be reliably attributed to residue 90. It is thus considered rigid.

Instead of giving a binary classification of each residue as either flexible or rigid, our flexibility measure provides a richer description by indicating the degree of flexibility. Based on our experiments, the conformational changes reported in the literature usually have values less than 30 in our measure.

3.2 A Statistical Test for Protein Flexibility

To test a fragment F for flexibility, we make the null hypothesis that F is rigid. We then compare the minimum RMSD $\hat{R}$ of F between two given conformations with a threshold r. If $\hat{R} > r$, we reject the hypothesis and consider F flexible; otherwise, we consider F rigid. The key issue here is to choose a suitable threshold r that is robust against the noise in the data. These thresholds are used to convert the minimum RMSD matrix $\hat{\mathbf{R}}$ to the matrix $\mathbf{T}$, as described in the previous section.

The noise model. Our flexibility test uses the minimum RMSD as the similarity metric. Let (x_i, y_i, z_i) and (x'_i, y'_i, z'_i) for $1 \le i \le n$ be the backbone atom coordinates of respectively two conformations q and q' of a protein fragment F. The RMSD is given by $R = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left((x_i - x'_i)^2 + (y_i - y'_i)^2 + (z_i - z'_i)^2\right)}$. The similarity between q and q' is defined as the minimum RMSD $\hat{R}$, which minimizes R over all possible translations and rotations of the two conformations.

Let us now analyze the effect of noise on the distribution of RMSD values. We assume that the noise at each coordinate of each atom of a protein is independently and identically distributed (i.i.d.) according to the normal distribution with zero mean and a given variance. Although this is a simple model, it allows us perform principled statistical analysis and has led to good results in our work (see Section 5) and in related previous work [26].

If the fragment F is rigid, then q and q' actually represent the same conformation. We can apply suitable translation and rotation to the coordinates (x'_i, y'_i, z'_i) so that the resulting new coordinates (x''_i, y''_i, z''_i) are the same as (x, y, z), except for the noise. More precisely, let σ^2 and σ'^2 be the variances of the coordinate noise for q and q', respectively. We have

$$(x_i - x''_i) \sim N(0, \sigma^2 + \sigma'^2), \tag{1}$$

where N denotes a normal random variable, because x_i and x''_i both follow the normal distribution and the sum of normal random variables is again a normal random variable. Thus,

$$\frac{x_i - x''_i}{\sqrt{\sigma^2 + \sigma'^2}} \sim N(0, 1), \tag{2}$$

i.e., a standard normal random variable with mean 0 and variance 1. The same holds for $(y_i - y''_i)/\sqrt{\sigma^2 + \sigma'^2}$ and $(z_i - z''_i)/\sqrt{\sigma^2 + \sigma'^2}$.

Now, let R be the RMSD between (x_i, y_i, z_i) and (x''_i, y''_i, z''_i) for $1 \le i \le n$. Consider

$$S = \frac{nR^2}{\sigma^2 + \sigma'^2} = \sum_{i=1}^{n}\left(\left(\frac{x_i - x''_i}{\sqrt{\sigma^2 + \sigma'^2}}\right)^2 + \left(\frac{y_i - y''_i}{\sqrt{\sigma^2 + \sigma'^2}}\right)^2 + \left(\frac{z_i - z''_i}{\sqrt{\sigma^2 + \sigma'^2}}\right)^2 \right) \tag{3}$$

According to (2), each term in the above sum is a squared standard normal random variable. By definition, S is then a Chi-square (χ^2) random variable with $3n$ degrees of freedom, and R^2 is a scaled χ^2 random variable.

The threshold for the minimum RMSD. To choose the threshold r, we need to bound the probability $\Pr(\hat{R} > r)$. Since $\hat{R} \le R$, we have $\Pr(\hat{R} > r) \le \Pr(R > r)$. We thus calculate $\Pr(R > r)$:

$$\Pr(R > r) = \Pr(R^2 > r^2) = \Pr(\frac{nR^2}{\sigma^2 + \sigma'^2} > \frac{nr^2}{\sigma^2 + \sigma'^2}) = 1 - F_{\chi^2}(\frac{nr^2}{\sigma^2 + \sigma'^2}) \quad (4)$$

which follows from (3) and F_{χ^2} denotes the cumulative distribution function of a χ^2 random variable. Given a desired bound p on $\Pr(\hat{R} > r)$, we can calculate the threshold r from (4), which shows that r depends on the noise level in the data (σ and σ'), the p-value, and the length of the protein fragment (n). In particular, r increases with decreasing n (see Fig. 3).

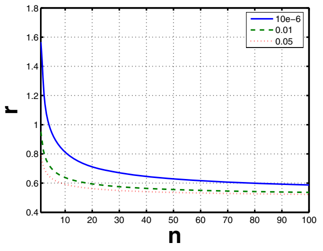


Fig. 3. The change of threshold r as a function of fragment length n, with p-value set to 1×10^{-6}. Each curve corresponds to a different noise level.

The threshold r implies that if F is rigid, then $\hat{R} > r$ with probability at most p. So, the p-value represents the confidence level of our statistical flexibility test. For example, suppose that $p = 0.01$. If $\hat{R} > r$, then F is rigid with probability at most $p = 0.01$, or equivalently, F is flexible with probability at least $1 - p = 0.99$.

Choosing the p-value requires some additional thought. Suppose that the probability of incorrectly assigning *any* fragment of a protein to be flexible should be at most γ. The obvious choice of $p = \gamma$ is incorrect, because we encounter the *multiple testing* problem. For a protein with n backbone atoms, our algorithm applies the statistical test once for each contiguous fragment of the protein, resulting in $n(n-1)/2$ tests in total. Let E denote the event that any of the tests gives the incorrect result and E' denote the event that a particular test gives the incorrect result. Then,

$$\Pr(E) \leq \frac{n(n-1)}{2}\Pr(E') \leq \frac{n(n-1)}{2}p.$$

Since we want $\Pr(E) \leq \gamma$, we must choose $p \leq 2\gamma/(n(n-1))$. As an example, for $\gamma = 0.05$ and $n = 300$, p should be smaller than 1.1×10^{-6}.

Although the thresholds calculated this way may appear overly conservative, it is justified due to the presence of noise in the data. Also, they are only used to generate intermediate results stored in $\mathbf{T}$. These results are further synthesized to generate the final output. Tests of our algorithm on PDB data show that it is reliable and does not miss salient conformational changes (Section 5).

4 Computational Efficiency

We now show that our algorithm runs efficiently in $O(n^2)$ time, where n is the number of backbone atoms in a protein. Our algorithm consists of four main steps: (i) computing the minimum RMSD matrix $\hat{\mathbf{R}}$, (ii) converting $\hat{\mathbf{R}}$ into the matrix $\mathbf{T}$, (iii) extracting the set $\mathcal{L}$ of minimal flexible fragments, and (iv) computing the flexibility measure $f(i)$ for each residue i.

Computing the minimum RMSD matrix. To compute $\hat{R}$ between two given conformations q and q' of a fragment, we apply the eigenvalue algorithm of

Horn [9]. It computes the covariances between the coordinates of the two conformations and then use them to build a matrix whose largest eigenvalue gives the minimum RMSD. This algorithm takes $O(m)$ time for a fragment of length m.

Since there are $O(n^2)$ contiguous fragments for a protein with n backbone atoms and computing the $\hat{R}$ for each takes $O(n)$ time, we can trivially compute the minimum RMSD matrix $\hat{\mathbf{R}}$ in $O(n^3)$ time. However, by constructing the covariances incrementally, we can reduce the total running time to $O(n^2)$, in other words, constant time per fragment on the average, which is asymptotically optimal. The details can be found in [17]. A similar algorithm was reported recently in [21].

Computing the matrix T. Given the RMSD value $\hat{R}$ for a protein fragment, we apply our flexibility test by computing the threshold value r and comparing it to $\hat{R}$. The threshold value r can be computed in $O(1)$ time. Hence, each entry in the matrix $\mathbf{T}$ can be computed in $O(1)$ time from the corresponding entry in $\hat{\mathbf{R}}$. Since $\mathbf{T}$ has $O(n^2)$ entries, computing it requires $O(n^2)$ time.

Extracting minimal flexible fragments. Dynamic programming is used to extract the set $\mathcal{L}$ of MFFs from $\mathbf{T}$, To do so, we construct another binary matrix $\mathbf{T}'$ based on $\mathbf{T}$ and go through $\mathbf{T}'$ diagonal by diagonal. We start with the first off-diagonal of T' and set $T'(i, i+1) = T(i, i+1)$ for $1 \leq i < n$. We then move to the next off-diagonal and iterate. If $T(i, j) = 1$, $T'(i+1, j) = 0$, and $T'(i, j-1) = 0$, then the corresponding fragment $F(i, j)$ is an MFF by definition. We add it to $\mathcal{L}$ and set $T'(i, j) = 1$ to indicate that $F(i, j)$ contains a flexible sub-fragment, in this case, itself. If $T(i, j) = 0$, $T'(i+1, j) = 0$, and $T'(i, j-1) = 0$, then $F(i, j)$ is rigid, and all its sub-segments are rigid. We set $T'(i, j) = 0$. Otherwise, we set $T'(i, j) = 1$, because $F(i, j)$ must contain a proper sub-segment that is flexible. Since $\mathbf{T}'$ contains $O(n^2)$ entries, the algorithm completes in $O(n^2)$ time.

Furthermore, the set $\mathcal{L}$ contains at most n fragments. To see this, consider any two fragments $F(i, j)$ and $F(i', j')$ in $\mathcal{L}$. We must have $i \neq i'$. Otherwise, one fragment would be a sub-fragment of the other. This is impossible, as both are MFFs. Since every fragment in $\mathcal{L}$ must have a distinct i value and $1 \leq i \leq n$, $\mathcal{L}$ contains at most n fragments.

Computing the flexibility measure. Since $\mathcal{L}$ has length at most n, it takes $O(n)$ time to compute the flexibility measure $f(i)$ for each fragment $F(i, i+1)$ and $O(n^2)$ time to compute f for all such fragments in a protein.

Since each of the four steps can be performed in $O(n^2)$ time or better, we have shown that for a protein with n backbone atoms, it takes $O(n^2)$ time to compute the flexibility measure f for all fragments $F(i, i+1), 1 \leq i < n$.

5 Results

We tested our algorithm on both synthetic data, in which case we know the ground truth, and experimental data from the PDB.

5.1 Synthetic Data

We took the PDB data for the TBSV coat protein (PDB code 2tbv, residues 102–387) and artificially changed a single torsion angle at residue 245 by 50°. We then added noise with mean 0 and standard deviation 0.2 to the atom coordinates. After creating the new structure, we compared it with the original structure using our algorithm. The results (Fig. 4) show that residues 240–249 are flexible and the rest are rigid. The f-values for residues 240–249 are 24. We then varied the same torsion angle by a smaller amount, 10°, and repeated the test. This time, the algorithm reported a much larger flexible region, residues 212–260, and the f-values for these residues range from 100 to 139. The larger f-values in the second test clearly indicate that the conformational change is smaller than that in the first test, and the residues involved are thus less flexible.

The single torsion angle that was changed is not identified because given the noise in the coordinates, there is not sufficient statistical evidence in the data that allows this. If the conformational change is small, we must examine a large fragment in order to differentiate genuine conformational change from noise, with confidence. Thus, the smaller the conformational change, the less precisely we can identify the region of flexibility.

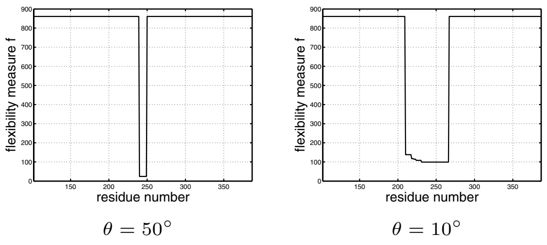


Fig. 4. Results for synthesized conformational change. A single torsion angle in residue 245 of the TBSV coat protein is changed by an angle of θ, and Gaussian noise is added to all atom coordinates.

5.2 Protein Structures

Using PDB data, we tested our algorithm on proteins exhibiting a wide range of conformational changes. Our data set (see Table 1) consists of all the proteins used in [22] to test similar algorithms. It also includes two additional proteins: adenosylcobinamide kinase, which undergoes shear motion, and HIV-1 protease, which undergoes a gradual, induced-fit type of motion. We performed tests on other proteins as well, but cannot report all the results here for lack of space; the readers are encouraged to use our software, which is freely available, to test other proteins of interest.

Tomato Bushy Stunt Virus (TBSV) coat protein. The results on this viral coat protein (Fig. 5*a*) show small f-values for residues 267–276 and very large f-values for the rest of the protein, which suggests a small region of high conformational flexibility with the rest of the protein being rigid. This closely matches the experimental evidence for the conformational change in this protein, which is reported to exhibit rigid-body closure about a single hinge at residues 266–272 [8].

Adenosylcobinamide kinase. Conformational change in adenosylcobinamide kinase is limited to a small fragment and involves the shearing of a helix (chain B, residues 233–249) effected by the residues at both ends of the helix. [24]. This is clearly shown in a morph of two conformations available from the Macromolecular Movements Database [3]. Our results (Fig. 5*b*) agree well with this

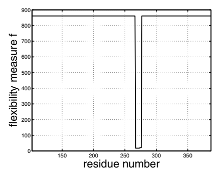
Table 1. Test proteins

Protein	Num. Res.	PDB code	σ	Motion
TBSV coat protein (residues 102–387)	286	2tbv, A	0.2	inter-domain,
		2tbv, C	0.2	hinge
adenosylcobinamide kinase	180	1cbu, B	0.2	intra-domain,
		1c9k, B	0.2	shear
lactoferrin	691	1lfg	0.2	inter-domain,
		1lfh	0.2	hinge
HIV-1 protease	99	3hvp	0.1	induced-fit
		4hvp	0.1	
lactate dehydrogenase	329	1ldm	0.1	intra- and
		6ldh	0.1	inter-domain
aspartate trans-carbamoylase	310	5at1, A	0.1	intra- and
		8atc, A	0.1	inter-domain
control	310	1rab, A	0.1	none
		1rac, A	0.1	

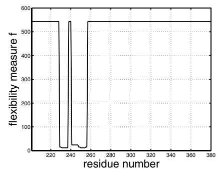
interpretation. Residues near the ends of the helix (residues 229–236 and 241–256) are identified as the flexible regions. The middle of the helix is not much affected by the shearing. The rest of the protein is rigid.

Lactoferrin. Lactoferrin is responsible for the reversible binding and transport of ferric iron. It contains multiple hinges and is folded into two similar lobes, the N-lobe (residues 1–333) and the C-lobe (residues 345–691), each of which binds with a cation. The domain closure is effected by local changes in two β-strands centered around residues 90 and 250 in the N-lobe [1]. Our results (Fig. 5*c*) correctly identify the two flexible β-strands, residues 90–95 and residues 249–252, which separate the N-lobe into three regions. See also Fig. 1 for improvement of our algorithm over some common existing ones. The C-lobe, which also binds with a cation, may also exhibit conformational flexibility. However, according to earlier work [5], "(The C-lobe) . . . shows no appreciable conformational change... The absence of changes in the C-lobe is not completely understood, but could arise from crystal-packing effects." Our plot of the flexibility measure shows a definitive flexible region (residues 415–425) between two rigid regions, which indicates the presence of the suspected conformational change. Movements of the N-lobe relative to the C-lobe is also detected through a flexible region between residue 321 and 362.

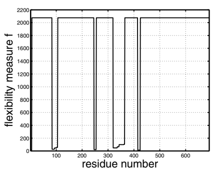
HIV-1 protease. HIV-1 protease plays a critical role in the maturation of HIV-1 virus and is a major inhibitory drug target. Its conformational flexibility affects the effectiveness of various inhibitors [19]. We applied our algorithm to two of the many known conformations of HIV-1 protease. The results (Fig. 5*d*) show that most of the residues have moderately low f-values. Thus, almost the entire protein is flexible to some degree. This reflects that the ligand binding process in HIV-1 protease fits the induced-fit model, in which many small movements of the receptor occur during the binding process. The results also show three regions of high conformational flexibility (residues 14–16, 38–40, and 50–53), and they are consistent with results reported in the literature [11].



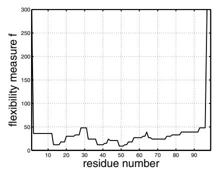
(a) TBSV coat protein



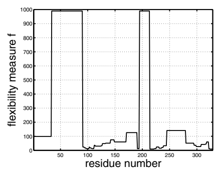
(b) adenosylcobinamide kinase



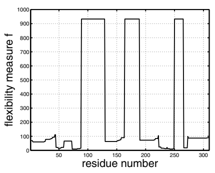
(c) lactoferrin



(d) HIV-1 protease



(e) lactate dehydrogenase



(f) aspartate transcarbamoylase

Fig. 5. Computed protein flexibility measure f, based on PDB data

Lactate dehydrogenase (LDH). The binding of LDH with the cofactor of nicotinamide adenine dinucleotide (NAD) induces major conformational changes as well as several smaller intra-domain changes. Our results (Fig. 5e) indicate regions of maximum flexibility in residues 91–114, 191–196, 214–235 and 322–329, and larger regions of moderate flexibility in residues 1–30, 115–190 and 235–320. These results agree well with the conformational changes suggested by Gerstein and Chothia [6]. The differences occur in only two regions. In [6], residues 1–8 are designated as static, and residues 191–196 have no designation.

Aspartate transcarbamoylase. Aspartate transcarbamoylase, from *E. coli.*, exhibits a complex combination of inter-domain and intra-domain conformational changes. The enzyme is found in two states often referred to as the tense (T) and the relaxed (R) states. Our results show high flexibility in three regions (residues 45–55, 75–90, and 230–246), which correspond to regions of intra-domain conformational changes found in earlier work [22]. Our results also show conformational flexibility in residues 130–155 and residues 260–270, located near the boundaries of known domains. They correspond to inter-domain conformational changes. Several other regions of moderate flexibility within the domains are also detected.

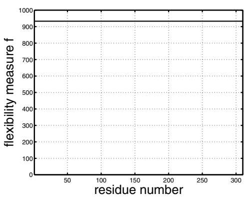


Fig. 6. Control experiment

Control. As a control experiment, we applied our method to two independently-determined structures of aspartate transcarbamoylase in the T state (PDB codes 1rab and 1rac). As expected, no flexibility is detected even at $\sigma = 0.1$ (Fig. 6).

The above results show that our algorithm works well for both inter-domain and intra-domain motions, including well-known examples such as hinge and shear. These results also show that despite its simplicity, the Gaussian model of coordinate noise is adequate, as an approximation, for accurate detection of conformational flexibility.

6 Discussion

An important feature of our algorithm is the assignment of a continuous per-residue flexibility measure, which allows it to handle sharp conformational changes as well as smaller, more gradual ones. Our algorithm does not presuppose the existence of a particular type of conformational change and, as a result, is able to identify a wide range of conformational changes. This is clearly illustrated in the HIV-1 protease example, in which the induced-fit motion is correctly identified. Some alignment algorithms (*e.g.*, [20,27]) which presuppose the existence of hinges separating rigid domains detect only a single hinge in this protein. This is clearly an incomplete picture of the conformational change in HIV-1 protease.

Our algorithm gives more accurate results than a number of commonly used approaches, as shown earlier in Fig. 1. We believe this is primarily due to the principled treatment of noisy data through the all-fragment analysis at the high level and the statistical flexibility test at the low level.

One issue that affects the accuracy of our algorithm is the setting for σ^2, the variance of the noise in the input protein structure coordinates. Sometimes, σ is readily available from multiple structure determination experiments. Other times, we can get a rough estimate from standard parameters in crystallographic data, such as temperature factors, but getting an accurate estimate may be difficult, as the relationship between these parameters and σ is complex and not easy to establish quantitatively. In such cases, we have found out from our experiments with PDB data that σ values between 0.1 and 0.2 Å work well.

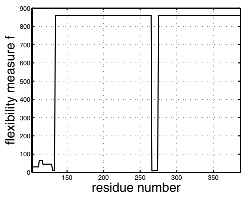


Fig. 7. Applying our algorithm to TBSV with $\sigma = 0.1$

Let us now consider what happens if we over- or under-estimate σ. Essentially, σ controls the sensitivity of our algorithm. For larger σ values, the sensitivity of our algorithm decreases. It detects only more significant conformational changes and may miss some subtle ones, which are masked as noise. For smaller σ values, the sensitivity of our algorithm increases. It is more likely to detect subtle conformational changes, but may also generate some false positives due to noise. For example, we set $\sigma = 0.1$ and re-ran our algorithm on the TBSV coat protein. The result (Fig. 7) is consistent with that for $\sigma = 0.2$ (Fig. 5*a*). The single hinge is detected in both cases. However, the result for $\sigma = 0.1$ shows an additional flexible region at one end of the protein (residues 102–133). This is likely due to increased noise in the structural data at the ends of a protein, rather than genuine conformational change.

Thus, when it is difficult to get an accurate estimate of σ, we can run the algorithm multiple times. We start with a relatively large σ value (say, 0.2) and gradually reduce σ. The conformational changes detected at high σ values are more reliable. With reduced σ, additional, more subtle conformational changes can be detected, but some false positives may also occur.

7 Conclusion

We have developed an efficient algorithm for analyzing conformational changes of a protein. It applies a statistical flexibility test to all contiguous fragments of a protein and combines the information to compute a consensus flexibility measure for each residue of the protein.

We tested the algorithm with PDB data. The results show that our algorithm reliably detects a broad range of protein conformational changes, including both inter-domain and intra-domain ones. Furthermore, this algorithm is fully automated. The user only needs to provide an estimate of the level of noise in the input protein structural data and the required confidence level of the results. In contrast to some earlier algorithms, the algorithm does not require the user to know the type of motion (*e.g.*, hinge or shear) in advance. Neither does it ask the user to select an arbitrary threshold for determining flexible protein fragments. Instead, our algorithm chooses such thresholds automatically based on principled statistical analysis. Our algorithm is efficient. It takes $O(n^2)$ for a protein with n backbone atoms and runs at interactive speed on a desktop PC even for large proteins with thousands of atoms.

Currently, our statistical test assumes that the coordinate noise in each atom is i.i.d., and the basis for identifying genuine conformational change is the magnitude of displacements in atom positions. An interesting extension is to explore the correlation among displacements. This may help improve our algorithm's accuracy in detecting coordinated motion involving many atoms.

While our work focuses on finding the flexible regions of a single protein in different conformations, the principle used by our statistical test for noise analysis applies to many other structural comparison problems. An example is to compare a set of different proteins in order to identify a common domain. When the effect of noise is significant, the statistical test may improve the performance of many algorithms for such problems.

Acknowledgements. We thank Jean-Claude Latombe of Stanford, Tomás Lozano-Pérez of MIT, and Lisa Tucker-Kellogg for insightful discussions, and Jack Snoeyink of UNC at Chapel Hill for supporting the initial phase of the work. This research was funded in part by NUS research grant R252-000-145-112.

References

1. B.F. Anderson, H.M. Baker, G.E. Norris, S.V. Rumball, and E.N. Baker. Apolactoferrin structure demonstrates ligand-induced conformational change in transferrins. *Nature*, 344:784–787, 1990.
2. L.P. Chew, D. Huttenlocher, K. Kedem, and J. Kleinberg. Fast detection of common geometric substructure in proteins. In *Proc. ACM Int. Conf. on Computational Biology (RECOMB)*, pages 104–113, 1999.
3. N. Echols, D. Milburn, and M. Gerstein. MolMovDB: Analysis and visualization of conformational change and structural flexibility. *Nucleic Acids Res.*, 31(1):478–482, 2003. http://molmovdb.mbb.yale.edu/molmovdb/.
4. M. Gerstein and R.B. Altman. Average core structures and variability measures for protein families: Application to the immunoglobins. *J. Mol. Biol.*, 251:161–175, 1995.

5. M. Gerstein, B.F. Anderson, G.E. Norris, E.N. Baker, A.M. Lesk, and C. Chothia. Domain closure in lactoferrin. *J. Mol. Biol.*, 234:357–372, 1993.
6. M. Gerstein and C. Chothia. Analysis of protein loop closure: Two types of hinges produce one motion in lactate dehydrogenase. *J. Mol. Biol.*, 220(1):133–149, 1991.
7. M. Gerstein, R. Jansen, T. Johnson, J. Tsai, and W. Krebs. Motions in a database framework: from structure to sequence. In M.F. Thorpe and P.M. Duxbury, editors, *Rigidity Theory and Applications*, pages 401–442. Kluwer Academic/Plenum Publishers, 1999.
8. P. Hopper, S.C. Harrison, and R.T. Sauer. Structure of tomato bushy stunt virus. V. Coat protein sequence determinations and its structural implications. *J. Mol. Biol.*, 177(4):701–713, 1984.
9. B.K.P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society A*, 4(4):629–642, 1987.
10. E.S. Huang, E.P. Rock, and S. Subbiah. Automatic and accurate method for analysis of proteins that undergo hinge-mediated domain and loop movements. *Curr Biol.*, 3(11):740–748, 1993.
11. D.J. Jacobs, A.J. Rader, L.A. Kuhn, and M.F. Thorpe. Protein flexibility predications using graph theory. *Proteins: Structure, Function, and Genetics*, 44(2):150–165, 2001.
12. D. Joseph, G. A. Petsko, and M. Karplus. Anatomy of a conformational change: Hinged "lid" motion of the triose phosphate isomerase loop. *Science*, 249:1425–1428, 1990.
13. A.P. Korn and D.R. Rose. Torsion angle differences as a means of pinpointing local polypeptide chain trajectory changes for identical proteins in different conformational states. *Protein Engineering*, 7:961–967, 1994.
14. W.G. Krebs, V. Alexandrov, C.A. Wilson, N. Echols, H. Yu, and M. Gerstein. Normal mode analysis of macromolecular motions in a database framework: Developing mode concentration as a useful classifying statistic. *Proteins: Structure, Function, and Genetics*, 48(4):682–695, 2002.
15. A.M. Lesk. *Protein Architecture: A Practical Approach.* Oxford University Press, 1991.
16. M. Levine, D. Stuart, and J. Williams. A method for systematic comparison of the three-dimensional structures of proteins and some results. *Acta Crystallography*, A40:600–610, 1984.
17. A. Nigham and D. Hsu. Protein conformational flexibility analysis with noisy data. Technical Report TRD1/07, National University of Singapore, School of Computing, Jan 2007.
18. A. Nishikawa, T. Ooi, Y. Isogai, and N. Saito. Tertiary structure of proteins. *J Phys. Soc. Jpn.*, 32:1333–1337, 1972.
19. A.L. Perryman, J. Lin, and A. McCammon. Hiv-1 protease molecular dynamics of a wild-type and of the v82f/i84v mutant: Possible contributions to drug resistance and a potential new target site for drugs. *Protein Science*, 13:1108–1123, 2004.
20. M. Shatsky, H.J. Wolfson, and R. Nussinov. Flexible protein alignment and hinge detection. *Proteins: Structure, Function and Genetics*, 48:242–256, 2002.
21. T. Shibuya. Geometric suffix tree: A new index structure for protein 3-d structures. In *Combinatorial Pattern Matching*, LNCS 4009, pages 84–93, 2006.
22. S. Subbiah. *Protein Motions.* Chapman & Hall, 1996.
23. M. Teodoro, G.N. Jr. Phillips, and L.E. Kavraki. A dimensionality reduction approach to modeling protein flexibility. In *Proc. ACM Int. Conf. on Computational Biology (RECOMB)*, pages 299–308, 2002.

24. T.B. Thompson, M.G. Thomas, J.C. Escalante-Semerena, and I. Rayment. Three-dimensional structure of adenosylcobinamide kinase/adenosylcobinamide phosphate guanylyltransferase from salmonella typhimurium determined to 2.3 a resolution,. *Biochemistry*, 37(21):7686–7695, 1998.
25. M. Vihinen, E. Torkkila, and P. Riikonen. Accuracy of protein flexibility predictions. *Proteins*, 19(2):141–149, 1994.
26. W. Wriggers and K. Schulten. Protein domain movements: Detection of rigid domains and visualization of hinges in comparisons of atomic coordinates. *Proteins: Structure, Function, and Genetics*, 29:1–14, 1997.
27. Y. Ye and A. Godzik. Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics*, 19 Suppl 2:ii246–ii255, 2003.

Deterministic Pharmacophore Detection Via Multiple Flexible Alignment of Drug-Like Molecules

Yuval Inbar[1], Dina Schneidman-Duhovny[1], Oranit Dror[1], Ruth Nussinov[2,3], and Haim J. Wolfson[1]

[1] School of Computer Science, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel
Fax: +972-3-640 6476
inbaryuv@tau.ac.il

[2] Sackler Inst. of Molecular Medicine, Sackler Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel

[3] Basic Research Program, SAIC-Frederick, Inc. Center for Cancer Research Nanobiology Program NCI-Frederick, Frederick, MD 21702, USA

Abstract. We present a novel highly efficient method for the detection of a pharmacophore from a set of ligands/drugs that interact with a target receptor. A pharmacophore is a spatial arrangement of physico-chemical features in a ligand that is responsible for the interaction with a specific receptor. In the absence of a known 3D receptor structure, a pharmacophore can be identified from a multiple structural alignment of the ligand molecules. The key advantages of the presented algorithm are: (a) its ability to multiply align flexible ligands in a deterministic manner, (b) its ability to focus on subsets of the input ligands, which may share a large common substructure, resulting in the detection of both outlier molecules and alternative binding modes, and (c) its computational efficiency, which allows to detect pharmacophores shared by a large number of molecules on a standard PC. The algorithm was extensively tested on a dataset of almost 80 ligands acting on 12 different receptors. The results, which were achieved using a standard default parameter set, were consistent with reference pharmacophores that were derived from the bound ligand-receptor complexes. The pharmacophores detected by the algorithm are expected to be a key component in the discovery of new leads by screening large drug-like molecule databases.

Supplementary Material:
http://bioinfo3d.cs.tau.ac.il/pharma/supp.html

Keywords: Computer-Aided Drug Design (CADD), Rational Drug Discovery, 3D Molecular Similarity, 3D Molecular Superposition.

1 Introduction

A pharmacophore is the three-dimensional (3D) arrangement of features that is essential for a ligand molecule in order to interact with a receptor in a specific binding mode. An identified pharmacophore can serve as an important model in rational drug design, since it can aid in the discovery of new lead compounds that can bind to a target receptor. Many computational methods for pharmacophore identification have been developed [1,2]. The methods are classified into *direct* and *indirect* methods. Direct methods use both ligand and receptor structural information. However, often the 3D structure of the receptor is unknown. In such cases, only indirect methods, which derive a pharmacophore only from a set of ligands that have been experimentally observed to interact with the receptor, are applicable. Generally, given a set of active ligands, the indirect methods search for the largest or highest scoring 3D pattern of features responsible for binding that is shared by all or most of the input ligands. If we represent the ligands by the 3D positions of the features that they possess, then a simpler variant of the problem is the *largest common point set* (LCP) problem in Computational Geometry, which is known to be NP-hard even when the input is only three 3D point sets [3,4]. The pharmacophore identification problem is further complicated by the fact that drug-like molecules are flexible, namely they possess many internal degrees of freedom, due mainly to rotatable bonds. As a result, they may have many possible conformations. The specific ligand conformations that bind in the active site of the receptor are unknown. Thus, all the feasible conformations of each input ligand have to be considered.

Due to the hardness of the problem, no indirect method finds the optimal solution in polynomial-time. The various existing approaches mainly differ in: (i) the chosen feature descriptors and structure representation, (ii) their technique for addressing the ligand flexibility, and (iii) the pattern identification algorithm [1]. The different feature descriptors mainly depend on the desired level of resolution. At the highest level, a feature is defined as the 3D position of an atom associated with the atom type [5,6,7]. At the next (coarser) level, atoms are grouped into topological features like phenyl ring and carbonyl group [8]. Finally, at the lowest level of resolution, spatially adjacent atoms are grouped into physico-chemical functional features that are important for ligand-receptor binding, such as aromaticity, charge, hydrogen bonding and hydrophobicity [9,10,11,12]. The ligands as well as the searched pharmacophore pattern are then described by the features that they possess, and their structures are represented mainly as 3D point sets [7], distance matrices [13,14], graphs [15,14], or trees [16]. Most indirect methods treat the conformational search as a separate initial stage. A discrete set of conformations is generated with the goal of sampling the whole conformational space of each ligand [17,11,10,9,7,5,18,19]. The main drawback of this approach is that the number of conformations required to cover the whole conformational space might be extremely large, especially for highly flexible compounds. An alternative approach is to combine the conformational search within the pattern identification process. The main advantage of this approach is that the search space is not limited to a precomputed discrete number of conformations. However, to date the methods that adopt this approach are based on a random search [8,20,6,21]. Furthermore, even for the simplified problem of superimposing only a pair of (and not multiple) ligands, deterministic algorithms that do incorporate the conformational search within their superposition process are rare. Two such methods are FlexS [22] and fFLASH [23]. The most common techniques for identifying pharmacophore patterns are clique-detection [17,5,24], exhaustive search [9,8] and genetic algorithms [5,20,6,21].

Here, we present a new indirect method for pharmacophore detection, which is, to the best of our knowledge, the first deterministic algorithm that performs this task through multiple flexible alignment. The main novelty of the method lies in its explicit consideration of ligand flexibility in the pattern identification stage. The algorithm is very efficient, as demonstrated in the Results section. Another key advantage of the method is its ability to find candidate pharmacophores shared by non-predefined subsets of the input ligands. This makes the method tolerant to outliers and to several binding modes. The performance of the method has been successfully evaluated on a benchmark dataset taken mainly from the FlexS dataset [25]. This dataset consists of almost 80 ligands that are classified into 12 cases according to the protein receptor they bind to. A web interface of this pharmacophore detection application is available at `http://bioinfo3d.cs.tau.ac.il/PharmaGist/`.

2 Method

Problem definition and hardness. Given a set of ligands, the goal is to find candidate pharmacophores, namely the largest (or highest scoring) 3D patterns of features responsible for binding that are shared by a significant number of input ligands. If we consider the ligands as rigid bodies represented by the 3D positions of their features, then a simpler optimization task is to search for the maximal cardinality set of features that is shared by all ligands. This task is equivalent to the *largest common point set* (LCP) problem in Computational Geometry, which is NP-hard even for the case of only three 3D point sets [3,4]. The pharmacophore detection task is even more complicated, since drug-like ligands are flexible and thus can adopt many conformations. As shown in the supplementary material, even the simplest case of finding the largest common set of features shared by a pair of molecules, one rigid and one flexible, is NP-Hard.

There are two other related requirements that are expected from a robust method for pharmacophore detection. The first requirement is motivated by the fact that, due to alternative binding modes, the same set of ligands may share several pharmacophores. Thus, the aim is to detect not only the largest (or highest scoring) candidate pharmacophore, but also other candidates, as long as their score is larger than a predefined threshold. Additionally, in order to overcome outlier ligands and to be able to deal with several binding sites of the target receptor, it is important to find candidate pharmacophores shared by only some of the ligands. This requirement complicates the problem since the number of ligand subsets is exponential in the number of input ligands. Furthermore, since there is a trade-off between the number of ligands and the number of features in their common 3D pattern, the exact definition of the requirement is quite vague mathematically. One possible approach, which we have adopted, is to find for any possible number of r input ligands, candidate pharmacophores shared by exactly r input ligands for which the score is greater than a predefined threshold.

Our approach. The input is a set of ligands, each given by the 3D coordinates of its atoms' centers and the covalent bonds between them. To avoid explicit conformational search, we assume that one ligand, the *pivot*, is given in its active conformation and thus considered as rigid. In contrast, the other (*target*) ligands are treated as capable of exhibiting torsional flexibility about their rotational bonds. Informally, the goal is to find the highest scoring 3D pattern of pivot features that can be aligned to most of the target ligands. We approach this task by searching for conformations of the target ligands and their superpositions on the pivot such that the score of the superimposed common features is maximized. Note that the pivot ligand may be selected by the user. However, the default assumption is that the identity of the pivot ligand is unknown. Thus, the method iteratively selects each one of the input ligands to serve as a pivot.

Formally, we define a *feature* of a molecule to be a set of atoms with a physico-chemical property important for ligand-receptor binding (aromaticity, charge, hydrogen bonding or hydrophobicity). Let $S(f^p, f^t)$ be a given scoring function for measuring the similarity between a pair of features, f^p of the pivot and f^t of a target ligand. We associate each feature with its center of mass and say that a pair of features, f^p on the pivot and f^t on the target ligand t, are *potentially matched* if their Euclidean distance is below a predefined threshold ϵ and their score, $S(f^p, f^t)$, is positive. Two equal-sized sets of l features, one of the pivot and one of a target ligand, $F^p = \{f_i^p\}_{1\leq i\leq l}$ and $F^t = \{f_i^t\}_{1\leq i\leq l}$, are said to be *flexibly matched* if there are a feasible conformation of the target ligand and a 3D pose (position and orientation) for it, such that the corresponding features, f_i^p and f_i^t (for any $1 \leq i \leq l$) are potentially matched. A set of features of the pivot, $F^p = \{f_i^p\}_{1\leq i\leq l}$ is said to be *m-matched* if there are m sets of target features, F^t ($1 \leq t \leq m$), each belonging to a different ligand, such that F^p and F^t are flexibly matched. The score of an m-matched set of pivot features is the center-star score of all matched feature pairs with the m target molecules, $\sum_{t=1}^{m}\sum_{i=1}^{l} S(f_i^p, f_i^t)$. For two sets of l features each, one of the pivot and one of a target ligand t, $F^p = \{f_i^p\}_{1\leq i\leq l}$ and $F^t = \{f_i^t\}_{1\leq i\leq l}$, we define $S(F^p, F^t)$ to be the sum of the similarity scores of the corresponding features, that is $\sum_{i=1}^{l} S(f_i^p, f_i^t)$.

Given a pivot, a set of M target ligands, and a distance error $\epsilon \geq 0$, **the goal** is to find for any m ($1 \leq m \leq M$) the highest scoring sets of pivot features that are m-matched. The pivot can be selected as the ligand with highest affinity to the receptor or the one with the smallest number of degrees of freedom. By default an iteration over all input ligands is performed. In this (default) scenario, the goal is generalized to selecting the best pivot as well.

Method Outline. The method consists of four stages: Ligand Representation, Pairwise Alignment, Multiple Alignment and Pharmacophore Clustering. In the first stage, each ligand is partitioned into rigid groups connected by rotatable bonds and is assigned a set of physico-chemical features. In the second stage, pairwise flexible alignments between the pivot and each target ligand are computed. In the third stage, we combine pairwise alignments into multiple alignments between the pivot and at least two target ligands. In the fourth stage, all candidate pharmacophores are clustered to produce a non-redundant set of solutions. While the second and the third stages are invoked for each possible pivot separately, the clustering stage is invoked only once to cluster solutions generated by all pivot iterations.

2.1 Ligand Representation

A ligand is represented by an *atom graph*. The vertices of the graph are the ligand atoms and the edges are the covalent bonds between them. The rotatable bonds of the ligands are identified and each ligand is divided into rigid groups. A bond is considered *rotatable* if it is not: (i) double, (ii) a ring bond, (iii) a bond connecting a single (leaf) atom, or (iv) a peptide bond. A *rigid group* of a ligand is defined as a set of atoms between rotatable bonds (including their atoms). To determine the rigid groups of a ligand, the connected components of a graph identical to the ligand atom graph but without the rotatable bonds are detected by DFS. Then, a rigid group is specified as the set of atoms of such a connected component and the atoms of the rotatable bonds to which it is connected in the atom graph. This definition ensures at least three atoms in a rigid group (in the extreme cases a rigid group consists of a leaf atom connected to a rotatable bond or two adjacent rotatable bonds). It also ensures that adjacent rigid groups are not disjoint, but share the atoms of the rotatable bond between them. Note that including both atoms of a rotatable bond in a rigid group does not violate the group rigidity, since when rotating the bond, its atoms remain in the same position relative to the other atoms in the group. The decomposition of a ligand into rigid parts is represented by a directed tree called *rigid group tree*. The vertices of the tree are the ligand rigid groups and the edges connect adjacent rigid groups. By DFS, the vertices (rigid groups) are topologically sorted and the edges are directed so that the out-degree of each vertex is at most one.

Finally, we compute for each rigid group the features that it possesses. A *feature* is a set of atoms with a physico-chemical property that is important for binding, namely it is one of the following types: (i-ii) a hydrogen-bond acceptor/donor atom; (iii-iv) an anion/cation atom; (v) a set of atoms of an aromatic ring (detected by a variant of BFS applied on the ligand atom graph); and (iv) a pair of adjacent hydrophobic atoms.

Applying this stage to a single ligand takes linear time in the size of the ligand, that is $O(n)$ if n is the maximal number of atoms in an input ligand. This is due to the fact that we apply several variants of DFS and BFS on the ligand atom graph.

2.2 Pairwise Alignment

The input is the pivot and a single target ligand. The pivot is considered as rigid, while the target is treated as flexible. The goal is to simultaneously find feasible conformations of the target ligand and their superpositions on the pivot such that the score of their aligned features is maximized. The algorithm consists of two stages, *Rigid Group Alignment* and *Rigid Group Assembly into a Flexible Alignment*. In the first stage, a set of transformations is generated for each rigid group of the target ligand. Each transformation superimposes a target rigid group on the pivot and yields a new candidate *pose* for the rigid group. In the second stage, we combine candidate poses of the target rigid groups. The result is a set of feasible conformations of the target ligand superimposed on the pivot such that the score of the aligned features is maximized.

Rigid Group Alignment. The goal is to generate candidate transformations for superimposing the rigid groups of the target ligand onto the pivot. For this purpose, we apply a hybrid technique of Pose-Clustering [26] and Geometric Hashing [27]. In the pre-processing stage, for each rigid group of the target ligand, we extract all non-collinear triplets of atoms and store them in a 3D hash table. The hash key of a triplet is the triple of side lengths of the triangle that it forms and is invariant to 3D translation and rotation. In the recognition stage, we extract non-collinear triplets of atoms from each rigid group of the pivot and use them to query the hash table. The result of each query with a pivot triplet is a list of all almost-congruent triplets from the target rigid groups.

Each pair of almost-congruent triplets, one of a pivot rigid group and one of a target rigid group, uniquely defines a transformation that superimposes the target triplet onto the pivot triplet with minimal RMSD [28]. Different pairs of almost-congruent triplets can lead to nearly identical transformations. Thus, for each target rigid group, we cluster similar transformations and join their matched triplets of atoms into one list. The clustering method is similar to the one used in [29] and is based on the RMSD distance between the images of the transformations on the atoms of the target rigid group.

Finally, for each cluster we compute a representative transformation with minimum RMSD between the matched atoms in the joined list in linear time in the size of the list [28]. These transformations when applied on the target rigid group represent new *poses* for it on the pivot. For each new pose we compute a *feature match list*. This is a list of pairs of matched features, one of the pivot and one of the target rigid group. Two such features can be matched if: (i) they have an identical physico-chemical property[1], and (ii) the distance between their centers of mass, on the pivot and in the new pose of the target rigid group, is less than a predefined threshold ϵ. Each feature match list is associated with a *feature score* ($S(F^p, F^t)$ as defined above). Given a new pose of a target rigid group, the feature match list with the maximized score is found by an exact algorithm for finding maximal matching in a bipartite graph [30]. The vertices of the graph are the features of both the pivot and the target rigid group. The edges connect potential pairs of matched features and are efficiently constructed by using a 3D look-up grid as follows. The pivot features are stored in the grid by

[1] In principle the method can match features of different types and maximize their overall score.

their midpoints. The midpoints of the features on the target rigid group pose are then used to query the grid and find ε-coincident pivot features. Since different features of a molecule cannot be too close in space, the number of pivot features that can be matched to a target feature is bounded by a small constant and the number of edges is thus linear in the number of vertices.

Let n be the maximal number of atoms in a ligand. In the worst case, both ligands are rigid and the number of atom triplets that are constructed for each one of them is $O(n^3)$. The maximal number of transformations for superimposing the target ligand on the pivot is thus $O(n^6)$ and clustering them takes $O(n^{12} \log n)$ time [29]. The representative transformation of each cluster is computed in $O(n)$ time [28] and its feature match list is computed in $O(n\sqrt{n})$ time [30]. The overall theoretical time complexity is thus $O(n^{12} \log n)$. In practice, since the atoms of a molecule are not random 3D points and cannot penetrate each other, the number of atoms that are close in space is bounded. Thus, by considering only triplets of spatially close atoms, we get a linear number of atom triplets per ligand, $O(n^2)$ poses, and an overall complexity of $O(n^4 \log n)$.

Rigid Group Assembly into a Flexible Alignment. The input is a set of candidate poses for each rigid group of the target ligand. We define a *feasible flexible alignment* of the whole target ligand on the pivot as an assembly of input poses that fulfills the following criteria: (i) there is exactly one pose for each rigid group; (ii) poses of two adjacent rigid groups are *consistent*, namely they agree on the location of their two shared atoms and cause no steric clashes between non-shared atoms; (iii) there are no steric clashes between poses of atoms on non-adjacent rigid groups. The goal is to find K feasible flexible alignments of the target ligand on the pivot with the highest feature scores of the associated feature match lists. This problem is NP-Hard (supplementary material). Below, we first present a graph-theory algorithm that solves the problem in polynomial time under the relaxation that: (*) steric clashes cannot occur between poses of non-adjacent rigid groups (condition iii). Then, we explain how we use this algorithm to find the K highest-scoring feasible flexible alignments that fulfill all the three conditions.

Assembly Graph Construction. We construct a weighted N-partite DAG, called *assembly graph*, where N is the number of rigid groups in the target ligand (Fig. 1). Each partition is associated with all the candidate poses of a specific rigid group. A vertex represents one pose for the respective rigid group. A pair of vertices in two different partitions are connected by an edge if they represent consistent poses of adjacent rigid groups. The rationale behind this construction is that any feasible flexible alignment of the target ligand on the pivot is represented by a tree in the graph and its feature score is the sum of the weights of all its vertices and edges. The weight of a vertex is the feature score of the represented rigid group pose. Adjacent rigid groups share the two atoms of the connecting rotatable bond and thus may have common features. To avoid double scoring of these features, we assign to each edge a non-positive weight computed as follows. We merge the feature match lists of the two connected rigid group poses and define the weight of the edge as the feature score of the resulting list minus the sum of the scores of the two separate feature match lists. The partitions of the assembly graph are ordered according to the order of the rigid groups in the rigid group tree of the target ligand (see Section 2.1). The edges of the assembly graph are directed according to this order, from the vertex in the partition with the lower index (*source partition*) to the vertex in the partition with the higher index (*target partition*). The directionality of the edges ensures that all the out-edges that start from a specific source partition always end at the same target partition. This ordering is exploited in the next stage of the algorithm.

Search for the K-Best Assembly Trees. Our aim is to find the K-best (highest-scoring) trees in the assembly graph that include at most one vertex from each partition. Such an *assembly tree* represents a feasible flexible alignment of the target ligand with the pivot (under relaxation *), since at most one pose is selected for each rigid group and the edges in the tree connect consistent poses of adjacent rigid groups. The K-best assembly trees are computed by dynamic programming in a bottom-up manner from the leaves to the root. In each step the K-best assembly trees rooted at vertices in a particular partition are computed. Specifically, the K-best assembly trees rooted at vertex v in the current partition are computed based on the K-best assembly trees rooted at vertices in source partitions that have out-edges to v. The order of the partitions and the directionality of the edges ensures that the K-best assembly trees of all vertices in source partitions with out-edges to v have already been computed in previous steps.

An assembly tree rooted at vertex v is computed by combining assembly subtrees rooted at vertices in source partitions and the corresponding in-edges of v such that at most one subtree is selected from each source partition. This guarantees that in the resulting tree only one pose is selected for each rigid group. Each vertex v holds a sorted list of the K-best assembly trees rooted at v. For a vertex with no in-edges, the only possible assembly tree consists of the vertex itself. For a vertex v with D in-edges, the K-best trees are computed in two stages: *Source Partition Merge* and *Tree Union*. In the first, Source Partition Merge, stage we select for each source partition of v

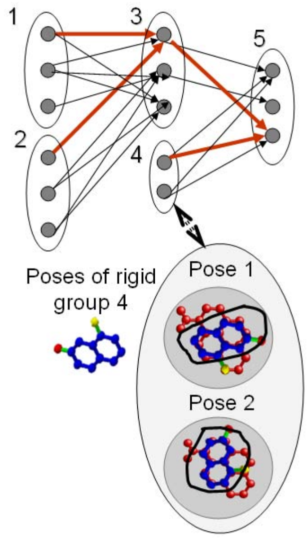


Fig. 1. Assembly Graph. An N-partite DAG. The vertices of a partition are circled and each represents a different pose of the same rigid group. Edges exist between consistent poses of adjacent rigid groups. An example for an assembly tree is depicted in red.

all the vertices that have out-edges to v and merge their sorted list of K-best trees into one sorted list with the K-highest scoring trees. The score of each tree in the merged list is the sum of the score of the original tree and the score of the connecting edge to v. In the second, Tree Union, stage the input is the merged list of K-best assembly trees for each source partition of v. Any combination of subtrees, each taken from a different input list, defines an assembly tree of v. The K-best assembly trees of v are computed by selecting the K-highest scoring combinations of assembly subtrees. The score of such tree is the sum of the scores of the combined subtrees and the score of v. There are K^D such possible combinations. However, since we are interested only in the K highest scoring ones, there is no need to enumerate all of them. This problem can be generally stated as: given D sorted arrays of numbers, find the K D-tuples of array indices with the highest sum of numbers. This problem can be efficiently solved [31]. Once the lists of K-best assembly trees for all the vertices are computed, we merge the lists of all vertices into one list and retain the K highest scoring trees.

The construction of the assembly graph ensures that any two connected vertices represent consistent poses of two adjacent rigid groups in the target molecule. However, a tree in the assembly graph might present inconsistency (steric clashes) between atoms on non-adjacent rigid groups. Thus, not all the K-best trees found by the algorithm represent feasible flexible alignments. Due to the geometry of drug-like molecules, the number of trees with inconsistent poses is usually much smaller than that of the consistent trees. Therefore, we iteratively increase the number of searched assembly trees until K feasible flexible alignments with a positive score are found (if exist).

An assembly tree does not necessarily contain poses for all the rigid groups of the target ligand. To obtain a feasible conformation for the whole target ligand, for each missing rigid group in the assembly tree, we set its pose according to the transformation of its predecessor rigid group (in DFS order) in the rigid group tree of the target ligand. Finally, for each new conformation of the target ligand, we update the list of matched features with the pivot by applying a maximum matching algorithm as described in the Rigid Group Alignment Section.

The algorithm works separately on each vertex. It first performs the merging procedure (possibly multiple times) and then it performs the union procedure. Let D^v be the number of in-edges of a vertex v. $D^v = \sum_i D_i^v$, where D_i^v is the number of poses of rigid group i that have out-edges to v. The merging complexity on the in-edges from rigid group i is $O(K \log D_i^v)$[31]. The union procedure complexity is $O(KD'_v \log D'_v + K \log K)$, where D'_v is the number of rigid groups that have poses with out-edges to v. The overall complexity for a single vertex v is $O(KD_v \log D_v + K \log K)$, resulting in an $O(K|E| \log D + |V|K \log K)$ time complexity for all vertices, where D is the maximum degree of a vertex in V. According to Rigid Group Alignment Section, $|V|$, which is the number of poses, is $O(n^2)$. Thus, in the worst case, D is $O(n^2)$ and $|E|$ is $O(n^4)$. This leads to an overall time complexity of $O(Kn^4 \log n + n^2 K \log K)$.

2.3 Multiple Matching

The input is the pivot, M target molecules and K pairwise alignments between each target molecule and the pivot. Each pairwise alignment matches a set of pivot features F_p to a set of features F_i of a target molecule with score $S(F_p, F_i)$. A selection of m $(2 \leq m \leq M)$ target molecules and exactly one pairwise alignment for each one of them defines a *multiple alignment.* The *consensus (matched) pivot feature set* F_p^c of a multiple alignment is defined as the intersection of the pivot feature sets matched by all its pairwise alignments. The *score* of a multiple alignment is the sum of the fractions of the scores of its pairwise alignments with respect to the consensus pivot feature set, that is $\sum_{i=1}^{m} S(F_p^c, F_i^c)$, where F_i^c is the feature set matched to F_p^c in the i^{th} target molecule. The goal is to find for each m $(2 \leq m \leq M)$ the highest scoring multiple alignments consisting of exactly m target molecules. The consensus pivot feature sets of these multiple alignments are pharmacophore candidates. This problem is NP-Hard even for K=1 (proved by a reduction from the maximum k-intersection problem [32]).

There is an exponential number of $O((K+1)^M)$ combinations to construct a multiple alignment from K pairwise alignments for each of the M target molecules. An enumeration over all these possible combinations is impractical. Moreover, we are interested in a method that will be scalable in the number of input molecules. Therefore, a more applicable approach is to enumerate the possible subsets of pivot features that can be matched by multiple alignments. If n is the number of pivot features, then there are $O(2^n)$ such subsets. An enumeration over all these subsets is practical since the number of atoms, and thus the number of features, in a typical drug-like molecule is small. We have adopted this approach and enumerate only *relevant pivot feature sets.* These are subsets of pivot features that are matched by at least two input pairwise alignments.

A relevant pivot feature set can be matched by several multiple alignments depending on the selected target molecules and the selected pairwise alignment for each one of them. Given a relevant pivot feature set F_p, the method represents all the multiple alignments for which F_p is part of their consensus pivot feature set in a data structure called *combinatorial bucket* (CB). Specifically, the CB of F_p holds for each target molecule all the pairwise alignments for which the set of matched pivot features includes F_p. Selecting at most one pairwise alignment for each target molecule in the CB defines a multiple alignment for which F_p is part of their consensus pivot feature set.

The method computes the relevant pivot feature sets and their CBs incrementally. First, all the relevant sets of size one and their CBs are created. In each of the following steps, all the relevant sets of size i are computed from relevant sets of size $i-1$. Specifically a relevant set of size i can be computed as the union of two relevant sets of size $i-1$. However, not every union of two sets of size $i-1$ leads to a set of size i. Additionally, different pairs of sets of size $i-1$ may lead to multiple copies of the same set of size i. Thus, a naive enumeration over all the pairs of sets of size $i-1$ is inefficient. Instead, an efficient enumeration based on the following observation is applied. Let $\{f_{k_1}, ..., f_{k_i}\}$ be a relevant set with i pivot features sorted by their indices. We can uniquely build this set as the union of two relevant pivot feature sets of size $i-1$, one without the first feature $\{f_{k_2},...,f_{k_i}\}$ and the other without the last feature $\{f_{k_1},...,f_{k_{i-1}}\}$. Thus, in step i the method enumerates over all the relevant pivot feature sets of size $i-1$. For each such set $F'_p = \{f_{k_1},...,f_{k_{i-1}}\}$, the method looks for another relevant pivot feature set F''_p of size $i-1$ without the first feature f_{k_1} and with an additional feature f_{k_i} ($k_i > k_{i-1}$), that is $F''_p = \{f_{k_2},...,f_{k_{i-1}},f_{k_i}\}$. To find all the possible sets F''_p for a given F'_p, the method enumerates over all the pivot features indexed from $k_{i-1}+1$ to n and checks if there is a relevant pivot feature set F''_p as required. The CBs of F'_p and F''_p are then combined by intersecting their sets of pairwise alignments for each target molecule, meaning that the resulting CB of the union pivot feature set contains a pairwise alignment if and only if it is present in both original CBs.

The many union and intersection operations of sets are efficiently performed using a bitwise representation of both the pivot feature subsets and the CBs. Assuming that the number of features in a drug-like molecule is smaller than the number of bits in a word (a standard of 64 bits), a pivot feature subset is represented by a single integer word. This allows us to keep all the pivot feature subsets in a hash table with a bitwise representation key. A molecule possesses $O(n)$ features, where n is its number of atoms. Thus, the complexity of the stage is $O(n|S|MK)$, where S is the set of all relevant pivot subsets. In the worst case, $S = O(2^n)$ and the overall complexity is $O(n2^nMK)$.

2.4 Pharmacophore Clustering

This stage recieves as an input the candidate pharmacophores from all pivot iterations. Different pivot iterations may lead to similar pharmacophores (similar spatial arrangements of the same feature types). Moreover, in case of intra molecular symmetry, the same pharmacophore may be detected in different regions of the pivot molecule. Therefore, a clustering stage is required to produce a set of non redundant candidate pharmacophores. We keep for each cluster a feature key and a representative pharmacophore. The feature key is simply the feature content (e.g. 2 aromatic, 3 acceptors and 1 anion). The representative solution is the highest scoring member. The clustering procedure enumerates the candidate pharmacophores in a descending score order. First, a feature key to the current solution is generated. Next, all clusters with this feature key are retrieved (using a hash table). If there is a cluster that its representative has almost the same spatial arrangement of features (according to types) as the current solution, the solution is added to the cluster. Otherwise, a new cluster is created with the current solution as its representative.

Overall Time Complexity. For M target ligands and a single pivot the pairwise and multiple alignment stages together take $O(M(Kn^4 \log n + n^2K \log K) + n2^nMK)$ time. Since we try all the input ligands as pivots, the overall time complexity is $O(M^2Kn(n^3 \log n + n \log K + 2^n)$. This analysis is based on the assumption that the degree of each vertex in the assembly graph is $O(|V|)$. In practice, the degree is much lower. Let D be the maximum degree of a vertex in the assembly graph, then the overall time complexity is $O(M^2 n^4 \log n + nKM^2(2^n + nD \log D + M^2 n \log K))$.

3 Results

Evaluation Procedure. We evaluated the method on a diverse dataset consisting of 74 drug-like ligands divided into 12 test cases. Each test case included several (crystal structure) complexes of the same protein receptor with different ligands. The ligands were separated from their complexes and their structures were minimized. We then applied our method on the minimized ligand structures of each test case (enumerating over all the possible pivots). The resulting candidate pharmacophores were compared to the superposition of the ligand structures in their bound modes. This *reference superposition* of the ligands was computed by superimposing the structures of the receptor from the different complexes. The rationale behind this evaluation approach is that a pharmacophore of a receptor is the 3D pattern of features shared by the active conformations of its ligands.

The evaluation procedure consists of two stages: (i) preparation of *reference pharmacophores* from the reference superposition and (ii) comparison of the candidate pharmacophores produced by our method to the reference pharmacophores. The *reference pharmacophores* are computed from the reference superposition as follows. We iteratively select each ligand in the superposition to serve as a pivot. Based on the extracted superposition, we compute the maximal set of matched pivot features for each of the remaining M target ligands by applying a maximal bipartite matching algorithm [30], where two features can be matched if they have the same type and their distance is below ϵ. This results in M pairwise alignments (one for each target ligand) that are given as an input to the multiple alignment algorithm. Since this algorithm performs exhaustive enumeration over all subsets of matched pivot features, it will produce all the possible reference pharmacophores that can be extracted from the reference superposition, including pharmacophores based on subsets of ligands. In the second stage, a candidate pharmacophore matched by m input ligands is compared to all reference pharmacophores of exactly m ligands. If there are less than three spatially distinct common features, we compare the candidate pharmacophore to the reference pharmacophores of $m - 1$ ligands and so on, in attempt to find a significant set of common features. In each comparison we apply Geometric Hashing[27] to produce a transformation that superimposes the candidate pharmacophore on reference pharmacophore, such that the type of matched features is the same and their distance is below ϵ after superposition. Each feature of the candidate pharmacophore is considered a *hit* if after applying the transformation, there are at least two features from different reference ligands with the same feature type in the distance below ϵ. We count the number of hits for the given candidate pharmacophore and compare to the total number of features of both the pharmacophore itself and the reference pharmacophore (see table in Fig. 2).

Test Cases: Our benchmark is mainly based on the FlexS dataset [25]. This dataset consists of 77 X-ray complexes that are classified into 14 test cases

# lig	hits/found	RMSD	# ref.	maxK
ACE 3 runtime 0:01				
3	7/7	0.00	7	22
G-phosphorylase 4 runtime 0:03				
4	9/9	0.36	9	5
3	13/13	0.76	13	5
Carboxyptd-A 5 runtime 0:04				
5	6/6	0.27	6	12
3	10/10	0.48	10	3
Thrombin 3 runtime 0:04				
3	6/6	0.31	4	12
Immunoglobulin 5 runtime 1:46				
5	13/13	0.76	14	20
4	15/15	0.73	15	20
3	17/18	0.64	18	20
Endothiapepsin 5 runtime 0:37				
5	3/5	0.31	6	594
4	8/8	0.38	10	116
3	9/13	0.62	13	33
Rhinovirus 8 runtime 0:17				
8(4)	6/7	0.63	0	277
5(4)	8/10	0.65	0	44
4	5/5	0.46	10	92
Trypsin 7 runtime 0:01				
3(0)	0/3	N/A	0	3

# lig	hits/found	RMSD	# ref.	maxK
Streptavidin 5 runtime 0:03				
5	10/10	0.18	8	9
3	10/11	0.14	10	4
HIV-protease 10 runtime 3:27				
10	3/4	0.28	4	1842
9	4/4	0.77	5	1910
8	4/4	0.37	6	20
7	5/5	0.50	6	308
6	5/6	0.56	7	829
5	5/6	0.55	8	13
4	6/8	0.77	9	308
3	16/18	0.63	12	50
Elastase 7 runtime 0:17				
6(0)	0/3	N/A	0	68
5(3)	6/6	0.69	0	1
3	6/11	0.55	6	1
Thermolysin 12 runtime 0:29				
9(7)	5/5	0.15	0	16
8(7)	5/5	0.25	3	56
7	6/6	0.40	6	264
6	5/6	0.31	7	44
5	6/7	0.49	8	286
4	12/12	0.59	12	8
3	12/13	0.59	14	8

Fig. 2. The Benchmark Results. The data presented for each case consists of name of the receptor, the total no. of input ligands, runtime (mm:ss - on a standard PC), and the details of the top-scoring candidate pharmacophore for every no. of input ligands: (i) the no. of ligands that match it (# lig). If the no. of ligands in the reference pharmacophore it was compared to is different, it is specified in brackets; (ii) the no. of feature hits out of the total pharmacophore features found by the algorithm; (iii) the feature RMSD between the common features of the reference pharmacophore and the top scoring candidate; (iv) the maximal no. of features (# ref) in the reference for the same no. of ligands; (v) maxK, which is the maximal rank of the pairwise alignments that was used in multiple alignment.

according to their receptor. We have considered all cases, except for three cases that are unsuitable for testing multiple alignments since each includes only two complexes. In addition, we have considered another test case of three ACE inhibitors, which was used to evaluate MTree [16]. Overall, our benchmark dataset consists of 12 different test cases, each with 3 to 12 ligands. The ligands vary from small molecules with only several heavy atoms to peptide that have dozens of rotatable bonds (see Table S-1 in the supplementary material).

The method was applied on the ligands of each test case using the same parameter set. No specific ligand was used as a pivot. Therefore, the algorithm iteratively selected each ligand to serve as a pivot. Then, the solutions (from all pivot iterations) were clustered, resulting in a non-redundant set of candidate pharmacophores. The score for matching two features of different types was set to 0, while the score for matching two features of the same type was 1, except for the score of two matched hydrophobic features, which was 0.3. The value of K given to the Flexible Pairwise Alignment algorithm was 1500. The value of the maximal distance error ϵ between two matched features was set to 1 Å. In order to account for inaccuracies in the crystal structures, the value of ϵ in the evaluation procedure was set to 1.4-2.0 Å.

All results are on the web site and a summary is in the table of Fig. 2. In the table we present a comparison of the top scoring candidate for every number of input ligands to the reference pharmacophores. A candidate pharmacophore is compared to a reference pharmacophore by: (i) the RMSD between the superimposed common features and (ii) the number of feature hits out of the total number of features in the candidate pharmacophore. In addition, we give the maximal number of features in the reference pharmacophore for the same number of molecules as in the evaluated candidate. Below, we summarize the performance of our algorithm on the dataset, followed by two detailed examples. In the supplementary material one can find a detailed discussion of all the test cases including interesting issues like alternative binding modes when presented.

As detailed in the table, in all test cases, we found relevant candidate pharmacophores with an RMSD below 1Å between their matched features. In some cases our candidate pharmacophores were matched by more input ligands (Rhinovirus, Elastase and Thermolysin) or more pharmacophoric features (Thrombin, Streptavidin, Elastase - 3 ligands pharmacophore, HIV-protease - 3 ligands pharmacophore) compared to the reference pharmacophores. In several cases (Immunoglobulin, Endothiapepsin, HIV-protease, Thermolysin) one or two features were missed compared to the reference pharmacophore. Additionally, for the top scoring candidate pharmacophore of each test case, we have computed the maximal rank of the pairwise alignments that take part in the respective multiple alignment ($maxK$ in the table of Fig. 2). This parameter helps us to evaluate the number of pairwise alignments (K) required by the multiple alignment algorithm in order to find the correct solution. In all cases, except for HIV-protease, the value of $maxK$ was less than 1000. The running times of the test cases are between one second for the three ACE inhibitors and three and a half minutes for ten HIV-protease inhibitors.

The **Rhinovirus** case is interesting since the ligands can bind to Rhinovirus in two alternative modes (Fig. 3a-b). The top-scoring candidate pharmacophore for all the eight ligands has seven features, six of them are hits (Fig. 3c). Due to reversed orientation of half of the ligands in the crystal structures, the candidate pharmacophore was compared to the corresponding reference pharmacophore of the four ligands. For five ligands, the top-scoring candidate pharmacophore is larger with ten features (eight hits) (Fig. 3d). For four ligands the largest reference pharmacophore consists of 10 features, while we find only five features in our top scoring candidate for four molecules. This is due to the fact that the 10 feature pharmacophore was already found for five molecules.

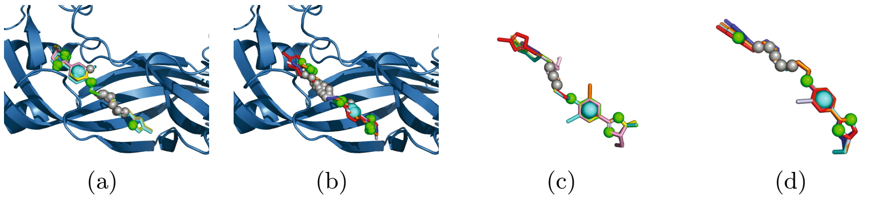


Fig. 3. (a-b) The two binding modes of Rhinovirus ligands can be distinguished by the relative location of their aromatic ring (a cyan sphere) in the protein binding site (blue). **(c)** The top-scoring candidate pharmacophore of all eight ligands contains seven features. **(d)** The top-scoring candidate pharmacophore for five ligands contains ten features. In all figures, green, cyan and gray spheres represent HB-acceptor, aromatic and hydrophobic features, respectively.

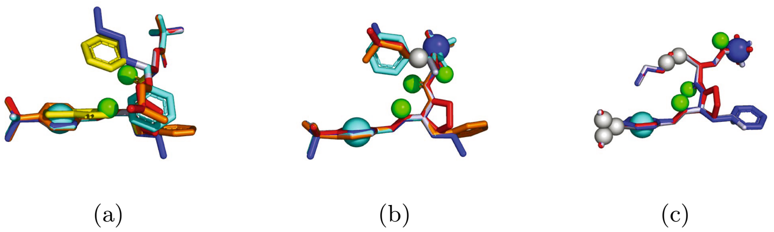


Fig. 4. The Elastase inhibitors case. (a-c) The top-scoring candidate pharmacophore for six, five and three input ligands consist of 3, 6 and 11 features, respectively. Green, cyan and blue spheres represent HB-acceptor, aromatic and positively charged features, respectively.

The Elastase binding site contains four specificity pockets. The tripeptidic structure of **Elastase** inhibitors allows them to occupy several of these pockets. Therefore, a significant reference pharmacophore was detected only for three out of the seven input ligands. Our top-scoring candidate pharmacophore is matched by six out of the seven ligands and consists of an aromatic ring and two HB-acceptors, which are important for binding to one of the specificity pockets (Fig. 4a). The top-scoring candidate pharmacophores found for five ligands is more significant and includes all the features of the reference pharmacophore of three ligands (Fig. 4b). In addition, our candidate pharmacophore for three ligands consists of eleven features, six of which are hits (Fig. 4c).

4 Conclusions and Future Work

We have described a fully automated indirect method for pharmacophore elucidation. Given a collection of (acyclic) drug-like flexible ligands that are known to interact with a specific receptor, the goal is to find the highest scoring 3D patterns of physico-chemical features that are shared by as many as possible input ligands. This problem is NP-hard with respect to the number of ligands, the number of features and the number of degrees of freedom. The method provides a heuristic solution. Its time complexity is exponential in the number of features, but polynomial in the number of ligands and the number of degrees of freedom. In practice, since the number of atoms, and, thus, the number of features in most drug-like ligands is small, the running time of the method are immensely satisfying, taking seconds to minutes on a standard PC.

The performance of the method has been successfully evaluated on a benchmark dataset consisting of 74 drug-like ligands divided into 12 test cases. The results show the ability of the method to deal with different types of drug-like ligands including peptides with more than 30 rotatable bonds. The ligand flexibility is fully taken into account in a deterministic manner, an attribute that, to the best our of knowledge, is unique to our pharmacophore detection method. The results also demonstrate that the method is capable of detecting candidate pharmacophores that are shared by non-predefined subsets of input ligands. This makes the method tolerant to the presence of outlier ligands and may aid in distinguishing between pharmacophores of different binding modes. In all test cases no prior knowledge on the receptor was assumed and the default parameters were used. However, in 'real-life' drug-design practice, some data on the receptor binding site or on the affinity of the ligands may be available and can be easily taken into account by setting the parameters.

In future work we intend to improve the method so it will be able to find pharmacophore patterns that are present on *non-adjacent* rigid groups of the ligands. In addition, we would like to generalize the definition of the searched candidate pharmacophores to deal with cases where a ligand is active despite lacking an important feature for binding that other ligands possess. The presented pharmacophore detection method will be a key component in the selection of new leads for drug design by virtual screening of small ligand databases.

Acknowledgments. We want to thank D. Fishlovitch, A. Oron and H. Senderowitz for useful advice. The research of O. Dror and Y. Inbar has been supported by the Eshkol Fellowship funded by the Israeli Ministry of Science. The research of H.J. Wolfson has been supported in part by the Israel Science Foundation (grant no. 281/05) and by the Hermann Minkowski-Minerva Center for Geometry at TAU. The research of H.J. Wolfson and R. Nussinov has been supported by the NIAID, NIH (grant No. 1UC1AI067231), and by the Binational US-Israel Science Foundation (BSF). This project has been funded in whole or in part with Federal funds from the National Cancer Institute, National Institutes of Health, under contract number N01-CO-12400. The content of this publication does not necessarily reflect the view of the policies of the Department of Health and Human Services, nor does mention of trade names, commercial products, or organization imply endorsement by the U.S. Government. This research was supported [in part] by the Intramural Research Program of the NIH, National Cancer Institute, Center for Cancer Research.

References

1. O. Dror, A. Shulman-Peleg, R. Nussinov, and H.J. Wolfson. Predicting molecular interactions in silico: I. an updated guide to pharmacophore identification and its applications to drug design. *Frontiers in Medicinal Chemistry*, 3:551–584, 2006.
2. O.F. Güner, editor. *Pharmacophore Perception, Development, and Use in Drug Design.* International University Line, La Jolla, CA, USA, 2000.
3. T. Akutsu and M.M. Halldorsson. On the approximation of largest common subtrees and largest common point sets. *Theoretical Computer Science*, 233:33–50, 2000.
4. M. Shatsky, A. Shulman-Peleg, R. Nussinov, and H.J. Wolfson. The multiple common point set problem and its application to molecule binding pattern detection. *J. Comp. Biol.*, 13:407–42, 2006.
5. J.D. Holliday and P. Willet. Using a genetic algorithm to identify common structural features in sets of ligands. *J. of Molecular Graphics and Modelling*, 15:203–253, 1997.
6. S. Handschuh, M. Wagener, and J. Gasteiger. The search for the spatial and electronic requirements of a drug. *J. Mol. Model.*, 6:358–378, 2000.
7. P.W. Finn, L.E. Kavraki, J.-C. Latombe, R. Motwani, C. Shelton, S. Venkatasubramanian, and A. Yao. RAPID: Randomized pharmocophore indentification for drug design. *Computational Geometry: Theory and Applications*, 10:263–272, 1998.
8. X. Chen, A. Rusinko III, A. Tropsha, and S.S. Young. Automated pharmacophore identification for large chemical data sets. *J. Chem. Inf. Comput. Sci.*, 39:887–896, 1999.
9. O.F. Güner, O. Clement, and Y. Kurogi. Pharmacophore modeling and three dimensional database searching for drug design using Catalyst: Recent advances. *Current Medicinal Chemistry*, 11:2991–3005, 2004.
10. O.A. Clement and A.T. Mehl. *Pharmacophore Perception, Development, and Use in Drug Design*, chapter HipHop: Pharmacophores Based on Multiple Common-Feature Alignments, pages 69–84. International University Line, La Jolla, CA, USA, 2000.
11. D. Barnum, J. Greene, A. Smellie, and P. Sprague. Identification of common functional configurations among molecules. *J. of Chemical Information and Computer Sciences*, 36:563–571, 1996.
12. H. Li, J. Sutter, and R. Hoffmann. *Pharmacophore Perception, Development, and Use in Drug Design*, chapter HypGen: An automated system for generating 3D predictive pharmacophore models, pages 171–189. International University Line, La Jolla, CA, USA, 2000.
13. C.W. Crandell and D.H. Smith. Computer-assisted examination of compounds for common three-dimensional substructures. *J. of Chemical Information and Computer Sciences*, 23:186–197, 1983.
14. A.T. Brint and P. Willett. Algorithms for the identification of three-dimensional maximal common substructures. *J. of Chemical Information and Computer Sciences*, 27:152–158, 1987.
15. Y. Takahashi, Y. Satoh, H. Suzuki, and S. Sasaki. Recognition of largest common structural fragment among a variety of chemical structures. *Analytical Sciences*, 3:23–28, 1987.
16. G. Hessler, M. Zimmermann, H. Matter, A. Evers, T. Naumann, Thomas Lengauer, and M. Rarey. Multiple-ligand-based virtual screening: Methods and applications of the MTree approach. *J. Med. Chem.*, 48:6575–6584, 2005.

17. Y. Martin, M. Bures, E. Dahaner, J. DeLazzer, I. Lico, and P. Pavlik. A fast new approach to pharmacophore mapping and its application to dopaminergic and benzodiazepine agonists. *J. Comput. Aided Mol. Des.*, 7:83–102, 1993.
18. N.J. Richmond, C.A. Abrams, P.R. Wolohan, E. Abrahamian, P. Willett, and R.D. Clark. GALAHAD: 1. Pharmacophore identification by hypermolecular alignment of ligands in 3D. *J. Comput. Aided Mol. Des.*, 20:567–87, 2006.
19. S.L. Dixon, A.M. Smondyrev, E.H. Knoll, S.N. Rao, D.E. Shaw, and R.A. Friesner. PHASE: a new engine for pharmacophore perception, 3D QSAR model development, and 3D database screening: 1. Methodology and preliminary results. *J. Comput. Aided Mol. Des.*, 20:647–71, 2006.
20. G. Jones, P. Willett, and R.C. Glen. A genetic algorithm for flexible molecular overlay and pharmacophore elucidation. *J. Comput. Aided Mol. Des.*, 9:532–49, 1995.
21. S.J. Cottrell, V.J. Gillet, R. Taylor, and D.J. Wilton. Generation of multiple pharmacophore hypotheses using multiobjective optimisation techniques. *J. Comput. Aided Mol. Des.*, 18:665–682, 2004.
22. C. Lemmen and T. Lengauer. Time-efficient flexible superposition of medium-sized molecules. *J. Comput. Aided Mol. Des.*, 11:357–368, 1997.
23. A. Krämer, H.W. Horn, and J.E. Rice. Fast 3D molecular superposition and similarity search in databases of flexible molecules. *J. Comput. Aided Mol. Des.*, 17:13–38, 2003.
24. D. Baum. Multiple semi-flexible 3D superposition of drug-sized molecules. In *Computational Life Sciences: First International Symposium, CompLife 2005*, volume 3695 of *LNCS*, pages 198–207, Konstanz, Germany, 2005. Springer.
25. C. Lemmen, T. Lengauer, and G. Klebe. FlexS: A method for fast flexible ligand superposition. *J. Med. Chem.*, 41:4502–4520, 1998.
26. G. Stockman. Object recognition and localization via Pose Clustering. *J. of Computer Vision, Graphics, and Image Processing*, 40:361–387, 1987.
27. Y. Lamdan and H.J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proceedings of the IEEE Int. Conf. on Computer Vision*, pages 238–249, Tampa, Florida, USA, December 1988. IEEE Computer Society Press.
28. W. Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Cryst.*, A 34:827–828, 1978.
29. M. Rarey, S. Wefing, and T. Lengauer. Placement of medium-sized molecular fragment into active sites of protein. *J. Comput. Aided Mol. Des.*, 10:41–54, 1996.
30. K. Mehlhorn. *The LEDA Platform of Combinatorial and Geometric Computing.* Cambridge University Press, United Kingdom, 1999.
31. L. Huang and D. Chiang. Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technologies (IWPT)*, pages 53–64, Vancouver, October 2005.
32. A.V. Staal. Privacy: A Machine Learning View. *IEEE Transactions on knowledge and data engineering*, 16:939–948, 2004.

Design of Compact, Universal DNA Microarrays for Protein Binding Microarray Experiments

Anthony A. Philippakis[1,3,4,*], Aaron M. Qureshi[1,5,*], Michael F. Berger[1,4], and Martha L. Bulyk[1,2,3,4,**]

[1] Division of Genetics, Department of Medicine and
[2] Department of Pathology, Brigham and Women's Hospital and Harvard Medical School, Boston, MA 02115
[3] Harvard/MIT Division of Health Sciences and Technology (HST), Cambridge, MA 02138
[4] Harvard University Graduate Biophysics Program, Cambridge, MA 02138
mlbulyk@receptor.med.harvard.edu
[5] Department of Mathematics, University of Maryland, College Park, MD 20742

Abstract. Our group has recently developed a compact, universal protein binding microarray (PBM) that can be used to determine the binding preferences of transcription factors (TFs) [1]. This design represents all possible sequence variants of a given length *k* (i.e., all *k*-mers) on a single array, allowing a complete characterization of the binding specificities of a given TF. Here, we present the mathematical foundations of this design based on de Bruijn sequences generated by linear feedback shift registers. We show that these sequences represent the maximum number of variants for any given set of array dimensions (i.e., number of spots and spot lengths), while also exhibiting desirable pseudo-randomness properties. Moreover, de Bruijn sequences can be selected that represent gapped sequence patterns, further increasing the coverage of the array. This design yields a powerful experimental platform that allows the binding preferences of TFs to be determined with unprecedented resolution.

Keywords: de Bruijn sequences, linear feedback shift registers, protein binding microarrays, motif, transcription factor.

1 Introduction

Detailed knowledge of the DNA binding specificities of TFs is crucial for both genomic studies attempting to map TFs to their target genes [2], as well as biophysical investigations of protein-DNA interactions [3]. Despite the importance of this data type, the binding preferences of the vast majority of TFs remain unknown, largely due to a historical lack of suitable experimental technologies. While chromatin immunoprecipitation (ChIP) experiments [4] (and, more recently, ChIP-chip experiments [5]) give specific examples of sequences bound by a TF *in vivo*, they do not provide an exhaustive characterization of the sequences that a TF can and

[*] These authors contributed equally to this work.
[**] Corresponding author.

(just as importantly) cannot bind. Similarly, approaches such as *in vitro* selection [6] typically identify only a limited number of high-affinity binding sites, making a direct quantification of relative binding preferences difficult.

To address this challenge, our group has developed the protein binding microarray (PBM) technology for high-throughput characterization of the *in vitro* binding specificities of protein-DNA interactions [1,7,8]. Briefly, a DNA-binding protein of interest is expressed with an epitope tag, then purified and applied to a double-stranded DNA microarray. The washed, protein-bound microarray is labeled with a fluorophore-conjugated anti-GST antibody. By scanning the array, quantitative information is generated regarding the preferences of the TF for each of the sequences on the array. Prior work by our group and others has demonstrated that this is an effective technology that allows rapid and high-quality determination of the DNA binding specificities of TFs [1,7-10].

A limitation of previous PBM studies, however, has been the lack of a universal array that can be used for the majority of TFs, regardless of their structural class or genome of origin. Earlier studies have utilized either microarrays containing a limited number of binding site variants chosen for the TF under consideration [7,9], or large genomic fragments obtained from the same genome as the TF (specifically, *S. cerevisiae*) [8]. The former approach has the twofold disadvantage of requiring a new microarray for each additional TF assayed and also requiring some *a priori* knowledge of the DNA binding specificities of the TF; the latter approach suffers from the limitation that longer sequences can contain several binding sites for a given TF, making it difficult to acquire quantitative information on protein-DNA interactions. Thus, a single microarray is desired that represents all possible binding sites of a given width k (i.e., all k-mers), in order to provide a complete survey of all candidate binding sites.

Our group has recently developed such a universal array [1]. The key to our design is two-fold. First, we have selected our double-stranded DNA probes to have a length (L) significantly longer than the motif widths (k) that we intend to inspect, so that each spot contains L-k+1 potential binding sites of width k. For a microarray composed of N spots, this increases the total number of k-mers represented from N (in the naïve construction where there is one k-mer per spot, as has been previously utilized [10]) to $N(L\text{-}k+1)$. Second, we have designed these spots to completely cover all k-mer sequence variants, so that a maximal number of distinct k-mers are represented. Consider the circular sequence shown in **Fig. 1A** that contains all 16 2-mer variants exactly once. Such sequences containing all 4^k overlapping k-mers one time are named de Bruijn sequences [11,12] of order k, and the spots of our universal array are obtained by computationally segmenting appropriately chosen de Bruijn sequences, leaving an overlap between adjacent sequences in order to not omit any k-mers. With this design, we are able to represent a maximal number of sequence variants in a minimum amount of sequence.

The implementation of this design, along with generated data for five TFs, has been presented in the work of Berger *et al* [1]. Here, we give an exposition of the underlying combinatorial and algebraic theory utilized in designing the array. Specifically, we provide a mathematical treatment of 1) the motivation for and utilization of linear feedback shift registers (LFSRs) to generate de Bruijn sequences; 2) theoretical developments made by our group in order to design de Bruijn sequences that not only contain contiguous k-mers, but also k-mers with biologically relevant

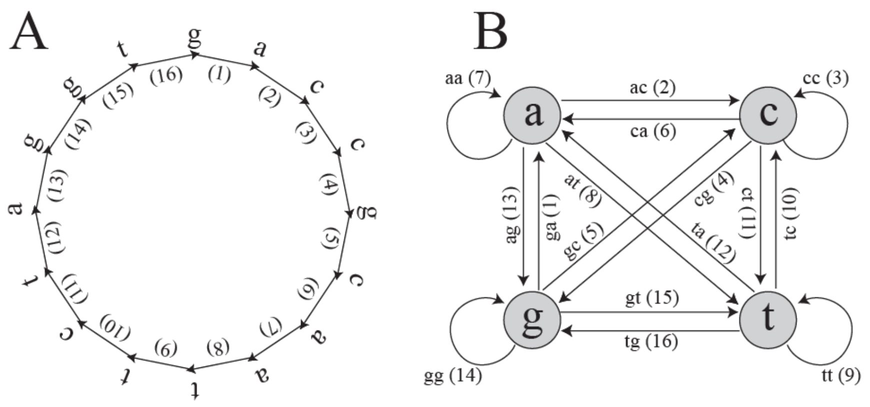


Fig. 1. De Bruijn sequence of order 2 (A) and its associated de Bruijn graph (B)

gaps; 3) methods for selecting de Bruijn sequences that are optimized for determining TF binding site motifs that are wider than the order of the utilized de Bruijn sequence; 4) the utilization of complementary, independently generated de Bruijn sequences for use in replicate PBM experiments. Finally, we note that de Bruijn sequences have previously been utilized in cryptography [13,14], random number generation [13,14] and the design of tags for DNA microarrays [15]. Recently, another group has independently suggested the use of de Bruijn sequences for use in PBM experiments, although that work did not consider the coverage of gapped *k*-mers and did not utilize LFSRs [16]. We hope that this work will be useful to individuals either seeking to design sequences for PBM experiments or analyzing data generated by a PBM utilizing de Bruijn sequences. Additionally, we hope that the mathematical methods developed for this application will lead to other, un-anticipated biological applications.

2 Results

2.1 LFSRs and the Generation of de Bruijn Sequences

For any alphabet Σ of size $|\Sigma|$ and any word length *k*, there exist sequences $S = \left(s_1 s_2 ... s_{|\Sigma|^k+k-1}\right)$ that are circular (i.e., $s_{|\Sigma|^k+1} ... s_{|\Sigma|^k+k-1} = s_1..s_{k-1}$) and of length $|\Sigma|^k$ containing all *k*-mers exactly once when words are considered in a stacked fashion. Such sequences are known as de Bruijn sequences of order *k*, and their existence can be confirmed by considering the directed graph whose vertices are all *k*-1-mers and whose edges are all *k*-mers, where two vertices are connected by an edge if the last *k*-2 letters of the first vertex are identical to the first *k*-2 letters of the second. **Fig. 1B** gives an example of such a graph (often referred to as a "de Bruijn graph" [12]), and we note that graphs of this form have previously been applied to the analysis of repetitive DNA [17] and sequence alignment [18]. Observe that a de Bruijn sequence is equivalent to a walk on this directed graph that traverses every edge (i.e., an Eulerian tour [12]). Since the number of edges going into each vertex is equal to the number of edges that exit it, Euler's theorem guarantees that such paths exist [12]. Indeed, for a given choice of $|\Sigma|$ and k, the number of paths is $\frac{\left(|\Sigma|!\right)^{|\Sigma|^{k-1}}}{|\Sigma|^{k}}$ [12]; for example, for $|\Sigma|=4$ (i.e., the DNA alphabet) and $k=9$, the number of de Bruijn sequences is greater than $10^{90,000}$.

De Bruijn sequences contain a maximum density of sequence variants, as they contain all distinct k-mers within a sequence of minimum length. Moreover, for any $m>k$ the $|\Sigma|^k$ sequences of length m represented in the de Bruijn sequence will all be distinct; thus, although not all m-mers are represented on an order k de Bruijn sequence, as many distinct m-mers as possible are represented within the given sequence length. Similarly, for all $m' < k$, each m'-mer is represented exactly $|\Sigma|^{k-m'}$ times, insuring that the sampling of m'-mers is uniform.

Clearly, the regularity and variability of de Bruijn sequences makes them a promising tool for designing a universal PBM. An especially facile method of generating such sequences when $|\Sigma| = p^n$, for p a prime and n any integer, is through the use of *linear feedback shift registers* (LFSRs) [13,14]. As background, recall that a Galois field GF(p^n) is a set containing p^n elements that is closed over the multiplication and addition $\{\times, +\}$ operations (one can show that such operations can be suitably defined if and only if the field contains a prime power of elements [19]). For example, **Fig. 2** gives multiplication and addition tables over GF(4) = $\{0, 1, \alpha, \alpha+1\}$

+	0	1	α	α+1
0	0	1	α	α+1
1	1	0	α+1	α
α	α	α+1	0	1
α+1	α+1	α	1	0

x	0	1	α	α+1
0	0	0	0	0
1	0	1	α	α+1
α	0	α	α+1	1
α+1	0	α +1	1	α

Fig. 2. Addition and multiplication tables over GF(4)

In order to construct a de Bruijn sequence of order k over the alphabet Σ, take an arbitrary embedding of the alphabet into GF($|\Sigma|$), and consider the following recursive linear equation for generating the i'th element of a sequence $S = \left(s_1 s_2 \ldots s_{\Sigma^k+k-1}\right)$ from the previous k elements, where arithmetic is performed over GF($|\Sigma|$):

$$s_i = \theta_{k-1}s_{i-1} + \theta_{k-2}s_{i-2} + \ldots + \theta_0 s_{i-k}\,. \qquad (1)$$

If the coefficients $\theta_i \in$GF($|\Sigma|$) are chosen so that the polynomial $\sum_{i=0}^{k-1}\theta_i x^i$ is primitive [19], one can demonstrate [14] that the sequence S generated by this recursive equation has periodicity $|\Sigma|^k$-1 and contains every k-mer in GF($|\Sigma|$) except the sequence of k consecutive 0's (this word can be easily included to make S a true de Bruijn sequence by inserting an additional 0 into any of the subsequences of k-1 0*'s* appearing in S). Moreover, S will exhibit the following three properties characteristic of pseudo-random sequences [13,14]:

1) Balance: The number of occurrences in S of each letter differs by no more than 1.
2) Low autocorrelation: There is low correlation between pairs of letters separated by a distance j, for any j.
3) Proportional runs: The number of j consecutive occurrences of the same letter ω is n^{k-j} if $\omega \neq 0$ and n^{k-j}-1 if ω=0.

Thus, de Bruijn sequences generated by LFSRs resemble random sequence, an advantageous property as it guarantees that any trends observed in the data are not an artifact of the method of sequence generation. Moreover, unlike random sequence, LFSRs represent a maximal number of sequence variants (a truly random sequence of equivalent length would represent only $1\text{-}e^{-1} \approx 63\%$ of k-mers on average [1]). Since the DNA alphabet contains a prime power of elements ($4=2^2$), LFSRs are available for use in generating de Bruijn sequences. Indeed, there are (at least) two approaches for using LFSRs to generate de Bruijn sequences over the DNA alphabet. In the first and more natural approach, one takes an arbitrary embedding of $\{a, c, g, t\}$ into GF(4) = $\{0, 1, \alpha, \alpha+1\}$ where $\alpha^2=\alpha+1$, and one then picks a primitive polynomial of degree k over GF(4) to use as a LFSR generating a sequence of length 4^k-1. This is schematized in **Fig. 2A**, using the embedding $\{a\leftrightarrow 0, c\leftrightarrow 1, g\leftrightarrow \alpha, t\leftrightarrow \alpha+1\}$ (again, under this embedding the generated sequences do not contain the sequence of k consecutive a's). Alternatively, one can pick a polynomial of degree $2k$ over GF(2)={0,1} and use it to generate a sequence of length 2^{2k}-1 over the 0-1 alphabet. Here, one associates each element of the DNA alphabet with a pair of elements in GF(2), and one must traverse this sequence over GF(2) twice, considering both reading frames. This is schematized in **Fig. 2B**, where we have used the embedding $\{a\leftrightarrow 00, c\leftrightarrow 10, g\leftrightarrow 01, t\leftrightarrow 11\}$. Henceforth, we shall refer to the embeddings $\{a\leftrightarrow 0, c\leftrightarrow 1, g\leftrightarrow \alpha, t\leftrightarrow \alpha+1\}$ and $\{a\leftrightarrow 00, c\leftrightarrow 10, g\leftrightarrow 01, t\leftrightarrow 11\}$ of the DNA alphabet into GF(4) and GF(2), respectively, as the *standard embeddings* (note that both methods of utilizing LFSRs to generate de Bruijn sequences can be generalized to arbitrary number fields with a prime power of elements). In the next section, we show that de Bruijn sequences generated by primitive polynomials over GF(2) and GF(4) actually behave differently with respect to the coverage of gapped k-mers.

Our basic design, then, is to utilize LFSRs to generate de Bruijn sequences of order k, where k is as large as possible for a given set of array dimensions and spot lengths. The generated de Bruijn sequence is then computationally segmented into shorter sequences of length l corresponding to spots on the array, leaving an overlap of k-1 letters between adjacent spots so as not to omit any k-mers. For example, consider an array composed of spots of length 30; then all 9-mers could be represented using fewer than 12,000 spots, well within the range of current array dimensions. Such an array would also contain nearly 1/4 of all 10-mers, 1/16 of all 11-mers, etc., and thus could be expected to yield substantial information about TFs having motif widths greater than 9.

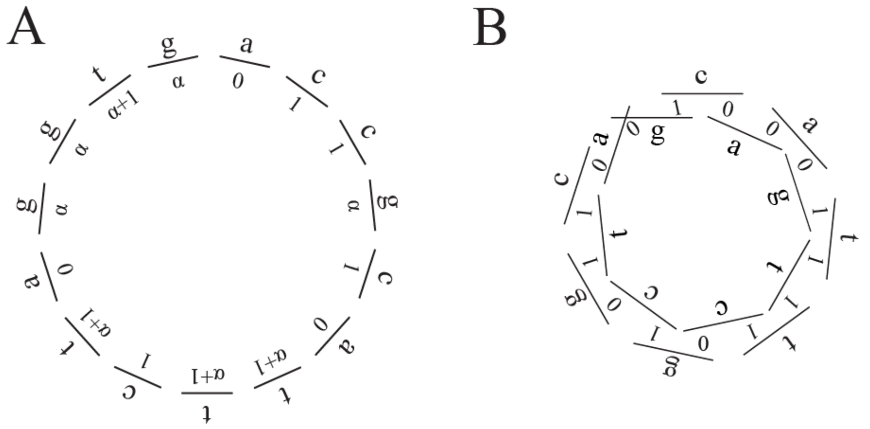


Fig. 3. Generation of de Bruijn sequences over (**A**) 4-letter and (**B**) 2-letter alphabets

2.2 LFSRs and the Coverage of Spaced Seeds

In Berger *et al* [1], we performed a survey of known TFBS motifs in order to determine what value of k is required in order to design an array suitable for most TFs. There, we observed that a majority of motifs contained <=9 informative positions. We also observed, however, that for nearly 25% of motifs, the pattern of informative positions was not contiguous and contained one or more gaps (i.e., positions with $\leq$0.3 bits of information when using the standard position-weight-matrix representation [20]). While de Bruijn sequences of order k will, by definition, contain all contiguous k-mers, they do not necessarily contain all gapped k-mers. Therefore, we inspected the representation of gapped k-mers in de Bruijn sequences.

We define a *seed* to be the set of all possible words over the DNA alphabet with a given (possibly gapped) pattern of positions, and we represent the seed with a 0-1 string where 1's are placed at the informative positions. For example, the seed "11" corresponds to the set {*aa, ac, ..., tg,tt*} that contains all contiguous 2-mers, and the seed "1001" corresponds to the set {*a(2)a, a(2)c, ..., t(2)g, t(2)t}* where the numbers in parentheses denote the gap size. Words with gaps will be said to be *elements of spaced seeds*, and those without gaps will be said to be *elements of contiguous seeds*. We shall use the variable ζ to represent an arbitrary seed and, for a seed ζ containing k informative positions, we say the *order* of ζ is k. Finally, a given LFSR L is said to *cover* a seed ζ if all its elements except the string composed of all a's (*e.g.*, *aa* and *a(2)a* for the order 2 seeds 11 and 1001, respectively) appear in the sequence generated by L (the reasons for ignoring the elements composed of only the letter a will be explained shortly). Similarly, we shall refer to the *coverage* of ζ by L with the variable $\chi(L, \zeta)$, which takes the value of 1 if the seed ζ is covered by the LFSR and 0 otherwise (again ignoring the element composed of only the letter a).

For a given sequence S over {a, c, g, t}, let $\mathcal{A}_{k,S}$ denote the set of all (potentially gapped) subsequences of k a's that occur in S; for example, in the sequence shown in **Fig. 2A** $\mathcal{A}_{2,S}$ = {$a(4)a$, $a(9)a$}, and in **Fig. 2B** $\mathcal{A}_{2,S}$ = {$a(1)a$, $a(5)a$, $a(6)a$, $a(7)a$, $a(8)a$, $a(12)a$}. For ζ a seed of order k and S a sequence generated by a LFSR over GF(q), where q equals either 2 or 4, one can demonstrate the following facts concerning the coverage of spaced seeds by LFSRs.

Proposition 1. a) Under the standard embeddings of the DNA alphabet, ζ is covered by S if and only if $\mathcal{A}_{k,S} \cap \zeta = \emptyset$. **b)** There exists a $j \in \mathbb{N}$ such that every element of ζ not in $\mathcal{A}_{k,S}$ occurs either 0 times or exactly q^j times in S. Also, the element of $\mathcal{A}_{k,S}$ in ζ occurs q^j-1 times.

Proof: Consider the case where q=4. Because our sequence $S = (s_1 s_2 s_3 \ldots)$ is generated by a LFSR, we know that for any i

$$s_i = \theta_{k-1} s_{i-1} + \theta_{k-2} s_{i-2} + \ldots + \theta_0 s_{i-k} \,. \tag{2}$$

Given values of i and m where $m \geq k$, let $\left(s_{i,m}\right)$ denote the subsequence in S of m letters beginning at the letter s_i; also, let this same notation denote the vector of dimension m over GF(q) $\left(s_{i,m}\right) = (s_{i+m-1}, s_{i+m-2}, \ldots, s_i)$. Consider the matrix

$$A = \begin{bmatrix} \theta_{k-1} & \theta_{k-2} & \theta_{k-3} & \cdots & \theta_1 & \theta_0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}$$

It is clear that for any i, $\left(s_{i+1,k}\right) = A\left(s_{i,k}\right)$ and, by induction, for any $j \geq 0$ $\left(s_{i+j,k}\right) = A^j \left(s_{i,k}\right)$. Also, observe that for any $m \geq k$, $\left(s_{i,m}\right)$ can be constructed from $\left(s_{i,k}\right)$ by applying the $m \times k$ matrix

$$\tilde{A}(m) = \begin{bmatrix} \left(A^{m-k}\right)_{1,1} & \left(A^{m-k}\right)_{1,2} & \cdots & \left(A^{m-k}\right)_{1,k-1} & \left(A^{m-k}\right)_{1,k} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \left(A^2\right)_{1,1} & \left(A^2\right)_{1,2} & \cdots & \left(A^2\right)_{1,k-1} & \left(A^2\right)_{1,k} \\ \theta_{k-1} & \theta_{k-2} & \cdots & \theta_1 & \theta_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

as $\left(s_{i,m}\right) = \tilde{A}(m)\left(s_{i,k}\right)$ (note that in $\tilde{A}(m)$, the entries $\left(A^n\right)_{i,j}$ refer to the matrix element in row i and column j in the n'th power of A). Consider a seed ζ having width m and containing k informative positions. Let $\tilde{A}(m,k)$ be the $k \times k$ sub-matrix of $\tilde{A}(m)$ when restricting to only those rows corresponding to the informative positions of ζ.

Consider the set $\{\tilde{A}(m,k)(s_{i,k}) | 1 \le i < 4^k - 1\}$ (i.e., the set of elements of ζ that occur in S). $\tilde{A}(m,k)$ is either invertible or it is not. If $\tilde{A}(m,k)$ is invertible, then its image is all 4^k elements of ζ, and every element of the seed occurs in the LFSR with the exception of the sequence that contains a 0 (equivalently, an *"a"* under the standard embedding) at every informative position, as the kernel of $\tilde{A}(m,k)$ is trivial. Thus, ζ will be covered if the sequence with *a*'s at the informative positions of the spaced seed (which is an element of $\mathcal{A}_{k,S}$) does not appear in *S*. Similarly, this argument is reversible and so the converse holds; thus, **Prop. 1a** is proven. If $\tilde{A}(m,k)$ is not invertible, then its kernel will contain 4^j elements for $j \in \mathbb{N}$, its image will contain 4^{k-j} elements, and each of these elements will be the image of 4^j vectors $(s_{i,k})$. Since every contiguous *k*-mer $(s_{i,k})$ except the sequence of *k* consecutive *a*'s occurs in *S*, **Prop. 1b** holds and the proof is completed for the case q=4. The proof for q=2 is nearly identical. Now, however, our matrix *A* will have dimension $2k \times 2k$. Note that here, the kernel for the matrix analogous to $\tilde{A}(m,k)$ will contain 2^j elements for some *j*. □

Thus, the spaced seeds that are missed correspond exactly to gapped patterns of *a*'s occurring within the LFSR and, for any spaced seed, the fraction of elements that are represented will be approximately either 2^{-j} when using a polynomial over GF(2), or 4^{-j} when using a polynomial over GF(4). We inspected the coverage of seeds most likely to correspond to known motifs for LFSRs over GF(2) and GF(4), in order to see if some polynomials empirically provided better coverage than others. Here, it is known that the number of primitive polynomials of degree *k* over a field with *q* elements is given by the formula $\frac{\phi(q^k - 1)}{k}$ where ϕ is Euler's totient function [21] that returns the number of integers relatively prime to the input value [14,19]; also it is easily seen that the number of spaced seeds of width up to *m* and order *k* is given by the formula $\sum_{i=k}^{m} \binom{i-2}{k-2}$.

Because we could not see how to determine $\mathcal{A}_{k,S}$ (and thus the set of covered seeds) for a given LFSR other than by explicit computation, we focused our analysis on the 7776 polynomials over GF(2) of order 18 and the 15,552 polynomials over GF(4) of order 9. For each of the de Bruijn sequences generated by these polynomials, we inspected whether each of the 44 seeds of widths $9 \le m \le 11$ and order k=9 was covered. For a given LFSR *L*, let $C(L) = \frac{1}{44}\sum_{\zeta} \chi(L,\zeta)$ (i.e., the average coverage), where the summation is over all of the spaced seeds ζ with widths between 9 and 11. We were pleased to observe that, by a judicious choice of LFSR, it is possible to completely cover over ~86% (38/44) of these seeds when considering polynomials over GF(4) and ~82% (36/44) of these polynomials over GF(2). Also, the mean coverage of polynomials over GF(4) was ~74±12%, significantly higher than average coverage of ~44±12% for polynomials over GF(2).

2.3 Sampling *k*-Mers Larger Than the Order of the de Bruijn Sequence

In this section, we demonstrate that the representation of spaced seeds is connected to the uniform sampling of words longer than the order of the shift register. As stated previously, the fraction of *m*-mers represented in an order *k* de Bruijn sequence is 4^{k-m} (where $m \geq k$). In this section, we demonstrate that if the sequence covers all spaced seeds of width $\leq m$ and order *k*, then the sampled *m*-mers are well-spaced and regularly sampled (this will be made precise momentarily), facilitating interpolation to *m*-mers not represented on the array. Thus, a suitable selection of de Bruijn sequence to cover spaced seeds is valuable to determining TFBSs whose width is greater than the order of the generating de Bruijn sequence.

Let *d* be the Hamming metric on words of length *k* over the DNA alphabet [21] (*i.e.*, the metric that counts the number of mismatches between pairs of words). For a de Bruijn sequence of order *k*, let *m* be an integer such that $m > k$. We say that the sampling of *m*-mers is *m,k-spaced* if for each word *w* of width *m* occurring in the de Bruijn sequence, there does not exist a distinct word *w'* in the de Bruijn sequence such that $d(w, w') \leq m - k$. Also, we say that the sampling of *m*-mers is *m,k-equidistant* if 1) for any choice of *k*-1 positions in *w* there exists a *w'* occurring in the de Bruijn sequence that agrees with *w* at these *k*-1 positions and such that $d(w, w') = m - k + 1$, and 2) the number of words *w'* appearing in the de Bruijn sequence such that $d(w, w') = m - k + 1$ is constant over the choice of *w*.

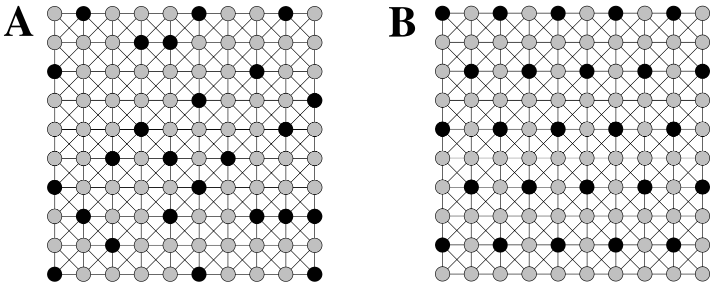


Fig. 4. Cartoon depicting all *m*-mers (grey vertices) and *m*-mers sampled by an order *k*<*m* de Bruijn sequence (black vertices). Vertices are connected by an edge if they are 1 mismatch away. (**A**) de Bruijn sequence that samples *m*-mers randomly. (**B**) de Bruijn sequence that samples *m*-mers regularly.

Intuitively, *m*-mers are regularly sampled if they are *m,k*-spaced and *m,k*-equidistant. This is cartooned by the graphs in **Fig. 4**, where nodes represent the 4^m possible *m*-mers, and the black nodes represent the 4^k *m*-mers that are represented within a given de Bruijn sequence of order *k*. In this graph, two *m*-mers are adjacent in the graph if they differ at only one position. A randomly chosen de Bruijn sequence will sample a random collection of *m*-mers (**Fig. 4A**), yet an auspiciously chosen de Bruijn sequence (i.e., *m,k*-spaced and *m,k*-equidistant) will regularly sample *m*-mers (**Fig. 4B**).

One can then prove the following two propositions regarding *m,k*-spacing and *m,k*-equidistance. We note that they apply to general de Bruijn sequences, and not only those de Bruijn sequences generated by an LFSR.

Proposition 2. The sampling of *m*-mers is *m,k*-spaced in an order *k* de Bruijn sequence if and only if all spaced seeds of width $m' \le m$ and order *k* are covered.

Proof: Assume that the de Bruijn sequence covers all spaced seeds of width *m'* and order *k*. Assume (for contradiction) that there are words *w* and *w'* such that $d(w, w') \le m - k$; then *w* and *w'* will agree at at least *k* letters. Consider the spaced seed that contains 1's at the positions where *w* and *w'* agree, and let *m'* be the distance between the first and last 1 in this spaced seed (note that *m'* may be less than *m* if *w* and *w'* do not agree at the first or last positions). Then *w* and *w'* will map to the same element of this spaced seed, and so the seed cannot be covered by the pigeonhole principle, giving a contradiction. Conversely, assume that a given de Bruijn sequence is *m,k*-spaced. Let ζ be a spaced seed of width $m' \le m$ and order *k*. Every element of ζ that appears in the de Bruijn sequence must occur only once. To see this, assume (for contradiction) that there is some element of ζ that occurs more than once. Then there are *m*-mers *w* and *w'* appearing in the de Bruijn sequence that agree at the *k* informative positions of the spaced seed. Then $\tilde{d}(w, w') \le m - k$, in violation of our assumption that the sampling of *m*-mers is *m,k*-spaced. Thus, ζ must be covered since the number of its elements that occur in the de Bruijn sequence is 4^k, all of which are distinct. □

Proposition 3. If the sampling of *m*-mers is *m',k*-spaced for all $k \le m' \le m$, it is *m,k*-equidistant.

Proof: For any $m' \le m$, we know that all spaced seeds of width *m'* and order *k* are covered, by **Prop. 2**. Let *w* be an *m*-mer and pick any *k*-1 informative positions in *w*. Since all spaced seeds of width $m' \le m$ and order *k* are covered, there will be exactly three distinct *m*-mers *w'* such that $w' \ne w$ and that occur in the de Bruijn sequence and agree with *w* at these *k*-1 informative positions (call this set *W*). The elements of *W* will all be at a distance of $\tilde{d}(w, w') = m - k + 1$ (so condition 1 is satisfied). Also, take a different choice of *k*-1 informative positions in *w*, and consider the set of three words *W'* agreeing with *w* at these *k*-1 informative positions. *W* and *W'* must be disjoint, since if there is a word in common between them, then it would agree with *w* at at least *k* informative positions, and then the de Bruijn sequence could not cover all spaced seeds of width $m' \le m$ and order *k*. This implies that every element has a constant number of *m*-mers at a distance of *m*-*k*+1, and so condition 2 holds. □

Finally, for the special case of *m*=*k+1*, one can state the following proposition giving analytic conditions relating *m,k*-spacing and *m,k*-equidistance to the choice of polynomial used for the LFSR:

Proposition 4. a) A de Bruijn sequence of order *k* generated by a LFSR over GF(4) is *k+1,k-spaced* and *k+1,k*-equidistant if and only if none of the coefficients θ_i of the generating polynomial $\sum_{i=0}^{k-1}\theta_i x^i$ are equal to 0. **b)** A de Bruijn sequence of order k generated by a LFSR over GF(2) is *k+1,k-spaced* and *k+1,k*-equidistant if and only if it does not contain three consecutive coefficients (θ_i, θ_{i+1}, θ_{i+2}) for even i such that $\theta_i\theta_{i+2}+\theta_{i+1}^2=0$.

Proof: Assume the case where the de Bruijn sequence is generated by the polynomial $\sum_{i=0}^{k-1}\theta_i x^i$ over GF(4). By **Props. 2** and **3** it is sufficient to prove that all seeds of width k+1 and order k are covered if and only if all θ_i are non-zero. Coverage of order k width k+1 seeds is equivalent to asserting that for any k+1-mer ($s_1,\ldots s_{k+1}$) and any $1\le i\le k$, there exists a value $\tilde{s}_i$ such that $\sum_{j=0}^{i-1}\theta_j s_j+\theta_i\tilde{s}_i+\sum_{j=i+1}^{k-1}\theta_j s_j=s_k$. Clearly, this can occur if and only if for all i $\theta_i\neq 0$. The proof for polynomials over GF(2) is nearly identical, but involves solving two such equations simultaneously. □

2.4 Complementary de Bruijn Sequences and Replicate Experiments

An additional advantage of our design is that, for any given value of k and desired set of represented gapped k-mers, if one acceptable de Bruijn sequence exists, there will in general be several that could be used (this is easily seen by, for example, permuting the letters or taking the reversal of the de Bruijn sequence). In Berger *et al* [1], we exploited this fact by doing replicate experiments on *distinct* de Bruijn sequences, both of which were *11,10-spaced* and *11,10-equidistant* (i.e., they covered all 10-mers containing a single gapped position). There, we observed that performing replicate experiments on distinct de Bruijn sequences, rather than the same de Bruijn sequence, improved our ability to correctly quantify the binding preferences of the well-characterized TF Zif268. We anticipate that this approach of performing replicate experiments on distinct de Bruijn sequences will be a valuable means for improving PBM experiments. In this section, we inspect some formal aspects of this experimental strategy.

The following proposition implies that all pairs of order k de Bruijn sequences generated by LFSRs will share a constant number of *k+1*-mers.

Proposition 5. Let S and S' be two de Bruijn sequences of order k, both generated by an LFSR over either GF(2) or GF(4). Then exactly 4^{k-1} k+1-mers will be commonly represented on both S and S'.

Proof: Assume that S and S' are generated by the polynomials $\sum_{i=0}^{k-1}\theta_i x^i$ and $\sum_{i=0}^{k-1}\theta'_i x^i$, respectively, over GF(4). Then S and S' will share a k+1-mer ($s_{i+k+1},\ldots s_i$) if and only if $\langle\Theta,\tilde{S}\rangle=\langle\Theta',\tilde{S}\rangle\Leftrightarrow\langle\Theta-\Theta',\tilde{S}\rangle=0$, where $\Theta=(\theta_{k-1},\ldots,\theta_0)$, $\Theta'=(\theta'_{k-1},\ldots,\theta'_0)$ and $\tilde{S}=(s_{i+k},\ldots,s_i)$. Since the null space of a linear form must always be of dimension k-1, there will be exactly 4^{k-1} values that satisfy this equation. The proof for

GF(2) is nearly identical, but involves finding the null space for two linear forms simultaneously. □

Thus, it is not in general possible to optimize the coverage of words longer than the order of the de Bruijn sequences in performing replicate experiments, as the number of k+1-mers represented on at least one of the two de Bruijn sequences will always be $2*4^k-4^{k-1}$. Note that **Prop. 5** also answers a natural question regarding the selection of the optimal order (k) to use for a given set of array dimensions (either on a single array or multiple arrays). It is not immediately clear whether it is better to have 4 different de Bruijn sequences of order 4^{k-1} or one de Bruijn sequence of order 4^k, as each requires an equal number of spots of the same length. **Prop. 5** implies that a de Bruijn sequence of order 4^k is preferable, as de Bruijn sequences of order 4^{k-1} will have overlap with respect to the k-mers that they represent.

Finally, we note that, although it does not seem that complementary order k primitive polynomials can be utilized in order to maximize the coverage of m-mers, $m>k$, we have observed that suitable sets of complementary polynomials can be selected for the coverage of gapped k-mers. Here, we have found by empirical inspection that if one polynomial misses a given spaced seed, then another polynomial can be identified that covers it. Thus, this parameter can be optimized.

2.5 Open Questions

We see (at least) three broad areas in which further mathematical/algorithmic efforts could lead to improvements in array design. First, assuming the use of LFSRs for generating de Bruijn sequences, there is need for an improved mathematical theory relating the coverage of spaced seeds to the generating polynomial. In this work, we have presented an explicit formula for determining whether a given polynomial represents all k-mers with a single gapped position (i.e., $k+1,k$-spaced and $k+1, k$-equidistant de Bruijn sequences), but we have not yet been able to extend this theory to k-mers with multiple gaps.

Second, only a small fraction of de Bruijn sequences correspond to sequences generated by an LFSR, and the utility of such non-LFSR-generated de Bruijn sequences remains largely unexplored. In current applications we have utilized LFSRs as they provably satisfy pseudo-randomness properties that are advantageous, since they guarantee that there are no confounding correlations in the experimental data that are an artifact of the methods utilized to generate the de Bruijn sequences. Additionally, LFSRs allow for the complete coverage of certain gapped k-mer patterns, which we have observed to be useful for determination of the binding specificities of TFs. However, it may well be the case that there are additional families of de Bruijn sequences that cover even more gapped k-mers while still resembling random sequence. Similarly, there may be additional desirable properties of de Bruin sequences that we have not yet considered, and for which LFSRs might not be optimal.

Finally, when considering protein-DNA interactions, it is often a reasonable assumption to identify k-mers and their reverse complements, as this symmetry is present in double-stranded DNA. In the case of PBM experiments in particular, this is a reasonable assumption to make if the DNAs are randomly fixed to the slide (although it is debatable whether or not this is appropriate in the case of end-attached DNA, as was the case for the arrays utilized in Berger *et al* [1]). The work presented here could be extended to generate de Bruijn sequences modulo reverse complements (i.e., sequences where only the word or its reverse complement is present, but not both) as was done in a related work [16]. If such sequences could be generated that had the desired pseudo-randomness properties as well as coverage of gapped *k*-mers, then their utilization might be advantageous.

3 Concluding Remarks

We have presented the combinatorial design of a protein binding microarray. Importantly, this design has been optimized in several key aspects: 1) All *k*-mers are represented in a minimum amount of sequence, permitting a maximum number of binding site variants to be represented in a cost-efficient manner. This allows the binding specificities of TFs to be assayed with word-by-word resolution. 2) The unbiased nature of the construction provides a design that can be used for TFs from any species and/or structural class, making it a universal platform. 3) Our design is flexible, allowing ever greater binding site coverage as array technology improves and feature density increases. For example, all 11-mers can already be represented with Agilent arrays [1], and all 12-mers with NimbleGen technology [22]; moreover, this number is expected to grow quickly. Similarly, our design allows replicate experiments to be performed with distinct de Bruijn sequences, resulting in reduced experimental noise and greater coverage of sequence space. 4) We have utilized de Bruijn sequences generated by LFSRs which provably resemble random sequence. This guarantees that any statistical trends observed in data generated by a PBM experiment are not an artifact of how the sequences were constructed. 5) Our design not only maximizes the coverage of contiguous *k*-mers, but also gapped *k*-mers. This simultaneously allows an interrogation of the binding specificities of TFs with gapped motifs and also improves the ability of the design to approximate the binding specificities of TFs whose width is greater than the order of the de Bruijn sequence.

Indeed, our group has already implemented this design and applied it to determine the binding specificities of five TFs from different organisms and structural classes with an unprecedented level of resolution [1]. There, we demonstrated that this platform could be used to assay the binding preferences of individual binding site variants, allowing us to identify at least one case of positional interdependence in a binding site motif. Similarly, we were able to approximate a binding site motif that was 12 bases in length using a de Bruijn sequence of order 10, attesting to the advantages of a careful and thorough coverage of gapped *k*-mers (point 5 above). Our group is now using this technology to determine the binding specificities of many TFs from a range of organisms, providing a much needed data type for the biological community. Thus, this microarray design provides a powerful, general and robust platform, and its implementation provides an experimental tool that will allow effective determination of TF binding site specificities both now and in the future.

Acknowledgments. We thank Savina Jaeger for critical reading of the manuscript. This work was funded in part by grant R01 HG003985 from NIH/NHGRI to M.L.B. M.F.B. was supported in part by a National Science Foundation Graduate Research Fellowship. A.A.P. was supported in part by a National Defense Science and Engineering Graduate Fellowship, a National Science Foundation Graduate Research Fellowship, and an Athinoula Martinos Fellowship.

References

1. Berger M.F., Philippakis A.A., Qureshi A.M., He F.S., Estep P.W., 3rd, Bulyk M.L.: Compact, universal DNA microarrays to comprehensively determine transcription-factor binding site specificities. Nat Biotechnol 24 (2006) 1429-1435
2. Bulyk M.L.: Computational prediction of transcription-factor binding site locations. Genome Biol 5 (2003) 201
3. Benos P.V., Lapedes A.S., Stormo G.D.: Is there a code for protein-DNA recognition? Probab(ilistical)ly. Bioessays 24 (2002) 466-475
4. Das P.M., Ramachandran K., vanWert J., Singal R.: Chromatin immunoprecipitation assay. Biotechniques 37 (2004) 961-969
5. Wyrick J.J., Young R.A.: Deciphering gene expression regulatory networks. Curr Opin Genet Dev 12 (2002) 130-136
6. Oliphant A.R., Brandl C.J., Struhl K.: Defining the sequence specificity of DNA-binding proteins by selecting binding sites from random-sequence oligonucleotides: analysis of yeast GCN4 protein. Mol. Cell. Biol. 9 (1989) 2944-2949
7. Bulyk M.L., Huang X., Choo Y., Church G.M.: Exploring the DNA-binding specificities of zinc fingers with DNA microarrays. Proc Natl Acad Sci U S A 98 (2001) 7158-7163
8. Mukherjee S., Berger M.F., Jona G. *et al.*: Rapid analysis of the DNA-binding specificities of transcription factors with DNA microarrays. Nat Genet 36 (2004) 1331-1339
9. Linnell J., Mott R., Field S., Kwiatkowski D.P., Ragoussis J., Udalova I.A.: Quantitative high-throughput analysis of transcription factor binding specificities. Nucleic Acids Res 32 (2004) e44
10. Warren C.L., Kratochvil N.C., Hauschild K.E. *et al.*: Defining the sequence-recognition profile of DNA-binding molecules. Proc Natl Acad Sci U S A 103 (2006) 867-872
11. De Bruijn N.G.: A Combinatorial Problem. Proc. Kon. Ned. Akad. v.Wetensch. 49 (1946) 758-764
12. Gross J.L., Yellen J.: Handbook of Graph Theory. CRC Press, New York (2004)
13. Joyner D., Kreminski R., Turisco J.: Applied Abstract Algebra. The Johns Hopkins University Press, Baltimore, MD (2004)
14. Golomb S.: Shift Register Sequences. Aegean Park Press, Laguna Hills, CA (1967)
15. Ben-Dor A., Karp R., Schwikowski B., Yakhini Z.: Universal DNA tag systems: a combinatorial design scheme. J Comput Biol 7 (2000) 503-519
16. Mintseris J., Eisen M.B.: Design of a combinatorial DNA microarray for protein-DNA interaction studies. BMC Bioinformatics 7 (2006) 429
17. Pevzner P.A., Tang H., Tesler G.: De novo repeat classification and fragment assembly. Genome Res 14 (2004) 1786-1796
18. Zhang Y., Waterman M.S.: An Eulerian path approach to local multiple alignment for DNA sequences. Proc Natl Acad Sci U S A 102 (2005) 1285-1290
19. Stewart I.: Galois Theory. Chapman & Hall, London, UK (1989)
20. Stormo G.D.: DNA binding sites: representation and discovery. Bioinformatics 16 (2000) 16-23
21. Terras A.: Fourier Analysis on Finite Groups and Applications. Cambridge University Press, Cambridge, UK (1999)
22. Singh-Gasson S., Green R.D., Yue Y. *et al.*: Maskless fabrication of light-directed oligonucleotide microarrays using a digital micromirror array. Nat Biotechnol 17 (1999) 974-978

Improved Ranking Functions for Protein and Modification-Site Identifications

Marshall Bern and David Goldberg

Palo Alto Research Center, 3333 Coyote Hill Rd., Palo Alto, CA 94304, USA
bern@parc.com, goldberg@parc.com

Abstract. There are a number of computational tools for assigning identifications to peptide tandem mass spectra, but many fewer tools for the crucial next step of integrating spectral identifications into higher-level identifications, such as proteins or modification sites. Here we describe a new program called ComByne for scoring and ranking higher-level identifications. We compare ComByne to existing algorithms on several complex biological samples, including a sample of mouse blood plasma spiked with known concentrations of human proteins. A Web interface to our software is at `http://bio.parc.xerox.com`.

1 Introduction

Proteomics offers a direct view of biological systems at the molecular level and can provide information, such as protein modification states and subcellular localization, that is not readily available from genomic sequence or RNA expression data. Moreover, because proteins throughout the body are commonly released into the bloodstream, plasma proteomics holds great promise for the discovery of new biomarkers for early detection of diseases such as cancer [7]. Currently biomarker discovery is severely limited by analytical sensitivity. Human blood plasma is thought to contain over 10,000 different proteins, varying in abundance over about 8 orders of magnitude [1], but state-of-the-art instrumentation now detects only hundreds of proteins at a time [2,16,26,27].

High-throughput, "shotgun", proteomics analyzes a biological sample with a multi-stage pipeline: (1) digestion of proteins to peptides, (2) chromatographic separation, (3) tandem mass spectrometry (MS/MS), (4) identification of peptide spectra, and (5) integration of spectral identifications into higher-level identifications. Of the two computational stages, (4) and (5), the former has been studied much more intensively. There are more than a dozen well-known programs for peptide identification, including database search programs, such as SEQUEST [13], Mascot [24], Sonar, OMSSA [15], and X!Tandem [8]; *de novo* sequencing programs such as PEAKS [20] and PepNovo [14]; and hybrid tools such as GutenTag [29], InsPecT [30], and our own new tool, ByOnic [5].

By contrast, stage (5) is underserved, with only a few programs, notably ProFound [33], DTASelect [28], Qscore [21], and ProteinProphet [22]. ProFound is designed for single-MS mass fingerprinting, a less sensitive method than MS/MS. DTASelect and Qscore are designed for use with SEQUEST. DTASelect collates the peptide identifications for each protein, and then applies user-defined criteria (such as thresholds for SEQUEST's XCorr and Delta scores) to decide which identifications to accept. Qscore is more automated, giving a protein score that reflects the number and quality of peptide matches.

Although ProteinProphet currently supports mainly SEQUEST and Mascot, it can in principle be used with any peptide identification program, so long as the peptide scores are first mapped to peptide probabilities. A companion program, PeptideProphet [17], can be used to compute such a mapping; PeptideProphet uses the well-known expectation maximization (EM) algorithm to fit a mixture of two Gaussians (one for true identifications and one for false) to the observed score distribution. ProteinProphet then reuses the EM algorithm to make protein identifications. It computes protein probabilities assuming that distinct peptide matches are independent, and then recomputes peptide probabilities based on the protein probabilities, iterating the peptide/protein loop until convergence.

Despite the availability of ProteinProphet, many—if not most—proteomics laboratories still use very little automation for stage (5). The Human Proteome Organization (HUPO) Plasma Proteome Project recently undertook a collaborative study of the same human serum and plasma samples by 18 different laboratories [23,27]. Some 13 of the 18 laboratories used only simple thresholds for protein identifications, such as requiring two Mascot peptide scores of at least 20 or one Mascot score of at least 30. One laboratory used ProteinProphet and another used DTASelect. (See Supplementary Table 1 in [27].) With such a diversity of tools and standards, it is not surprising that the HUPO study found a problem with reproducibility: less than 50% agreement between the proteins found by two different laboratories using exactly the same set of spectra.

In this paper, we describe a new program called ComByne for integrating spectral identifications into higher-level identifications. We demonstrate ComByne using spectral identifications from our own database-search tool, ByOnic, but the algorithm is more generally applicable. ComByne takes a *p*-value approach, modeling the probability that the peptide identifications would arise by chance alone. We back up ComByne's *p*-values with the use of reverse sequences [6], that is, deliberate false positives included for an empirical measurement of the false discovery rate (FDR). ComByne uses more information than previous tools, optionally incorporating chromatographic retention times and score deltas (percent difference between the top and the second-best identifications). The most novel aspect of ComByne, however, is its use of "corroboration". A simple example of corroboration is the following: a semitryptic peptide identification of YTYGKPVPGH becomes more believable if another spectrum in the data set matches the tryptic peptide YTYGKPVPGHK. Similarly, an identification of ASLGS[-18]LEGEAEAEASSPK becomes more believable if another spectrum matches ASLGS[+80]LEGEAEAEASSPK, because phosphorylated serine has a common neutral loss of 98 Daltons. We determined the contributions such corroborations should make to overall protein scores using a simple combination of empirical and theoretical analysis. We demonstrate the effectiveness of ComByne on several data sets, including a large phosphoprotein data set.

2 Algorithms

The input to ComByne is a set of identifications (peptide, protein, and position within the protein). There is typically only one identification per spectrum, but an *ambiguous peptide* occurs in more than one protein, or is indistinguishable from a peptide in another protein, for example, AELVYS and AEIVYS. Each identification also includes a score and a delta—the percent difference in score between the best and the second-best peptides. For the purpose of computing deltas, we do not consider a deamidated peptide to be distinct, so ILNGGTLLGLK cannot be second-best to ILN[+1]GGTLLGLK.

Scores to p-values. The first step of ComByne converts the scores and deltas to p-values. Like all the other database search tools, ByOnic does not achieve perfect separation of true and false peptide identifications. Figure 1 shows quite substantial overlap of the distributions for true and false identifications. (Here we defined true identifications to be matches to the top 50 plasma proteins, and false identifications to be matches to reverse protein sequences.) The overlap is such that the overall score distribution is not noticeably bimodal, and neither distribution has Gaussian tails, so PeptideProphet would not find an accurate model of the distributions.

For ComByne, we model only the false score distribution. The false distribution varies less than the true distribution from sample to sample, and hence we use a single fixed reference distribution, and then transform this distribution to account for the effective database size and length of peptide. To build the reference distribution, we used an empirical histogram up to ByOnic score 300, and a fitted exponential distribution for scores above 300. A ByOnic score of 300

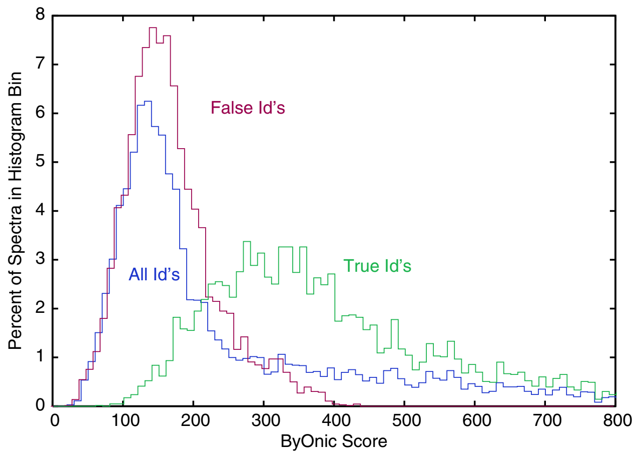


Fig. 1. The figure shows a histogram of ByOnic scores for the top-scoring identifications for about 10,000 ion-trap spectra of mouse blood plasma. Notice that the curve for all identifications is not noticeably bimodal, and that the curves for true and false identifications have considerable overlap.

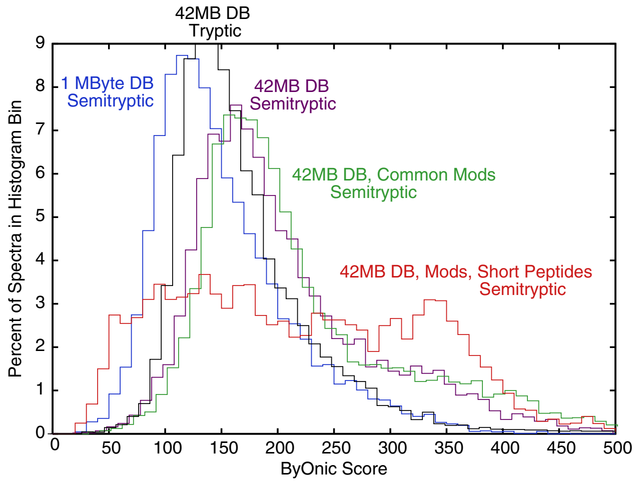


Fig. 2. The distribution of ByOnic noise scores (top matches to reverse protein sequences) varies with the size of the database, cleavage specificity, and the number of modifications enabled. The length of the peptide has a large effect on the distribution; here we define a short peptide to be one with at most 9 amino acid residues. ComByne uses smooth curves fit to empirical distributions to map ByOnic's scores to p-values.

corresponds approximately to a Mascot score of 30, and typically gives a false positive rate of less than 10%. An exponential distribution makes theoretical sense if we think of the score as arising stochastically: each database peptide of the right mass picks a score at random from some unknown distribution f, and the maximum of these picks is assigned to the spectrum. For this experiment, as the number of picks goes to infinity, many choices of f (including Gaussian) give a Gumbel or type I extreme value distribution, which has an exponential right-hand tail.

As seen in Figure 2, the false distribution varies with the size of the database, and whether we restrict attention to peptides with specific cleavage. There are about 10 times fewer fully tryptic peptides (both cleavages after R or K, or at a protein terminus) than semitryptic peptides (only one such cleavage). The length of the peptide also has a major effect on the false score distribution, because a database of 42 million amino acid residues contains a significant fraction of all sequences of lengths 5 and 6, and even for lengths 8 and 9 a high score is often achieved by a false database peptide that shares a prefix or suffix with the true peptide. ComByne translates its reference distribution to account for effective database size, and it stretches the distribution (by $12/n$, where $n < 12$ is the peptide length) to account for the dependence upon length. These simple (perhaps simplistic) transformations gave good agreement to empirical distributions such as those shown in Figure 2. Thus ComByne maps each score to a p-value, the probability that a randomly picked false identification would give such a score. With a 42-MByte database, a ByOnic score of 300 for an unmodified tryptic peptide of length at least 12 gives a p-value of 0.11; for a tryptic peptide with a modification it gives 0.15; for a semitryptic peptide it gives 0.20; and for an 8-residue tryptic peptide it gives 0.28.

ComByne optionally incorporates deltas into the p-values,[1] using a single, empirically trained, approximately exponential, distribution that is then translated to account for database size. A delta of 0.10 (that is, a 10% drop from the first- to second-best peptide) gives a p-value of 0.17 on a 42-MByte database; a delta of 0.15 gives a p-value of 0.082; and a delta of 0.20 gives a p-value of 0.042. We model scores and deltas as independent and thus the overall p-value of a long tryptic peptide scoring 300 with delta 0.10, on a 42-MByte database, is $0.11 \times 0.17 \approx 0.019$, meaning that only one false identification in about 50 would have numbers so good. We trim the delta p-value to the maximum of its original value and the score p-value; this is an empirical improvement.

Initial Protein Scores. ComByne computes an initial score for each protein simply by summing the logarithms of the p-values of all of the protein's peptide identifications (with no discount for duplicates). This crude approximation assumes that peptide identifications are independent. ComByne sorts the proteins by score, and then assigns each ambiguous peptide to only its highest-ranking protein. The assignments to lower-ranking proteins can be removed entirely or marked as "homologs" for later visual inspection.

At this point, ComByne optionally incorporates chromatographic retention times into its peptide p-values. Other groups [25,32] have built software to make use of retention time, typically using it as a filter to discard dubious peptide identifications. ComByne uses a model developed by Krokhin *et al.* [18] with coefficients trained by Ted Jones (PPD Inc., Menlo Park); this model computes a quantity called "retention index" as a weighted sum of contributions for each amino acid residue, along with peptide-length and N- and C-terminus corrections. ComByne fits the observed retention times for well-identified tryptic peptides from well-identified proteins (total log p-value < -20.0) to the computed retention indices using robust regression (see for example [4]) with a quadratic correction curve. It then computes the standard deviation σ of residuals for the well-identified peptides; a σ of 3.0 minutes is typical for a 90-minute run. Log p-values for peptide identifications are now adjusted by adding $\log_2\left((|r(P)| + 1.0)/(2\sigma)\right)$, where $r(P)$ is the retention time residual for peptide P. This (admittedly *ad hoc*) adjustment can either increase or decrease the log p-value: peptides eluting within about 2σ of their predicted times receive mild bonuses, whereas ones with larger elution errors receive mild penalties. (Adding 1.0 to $|r(p)|$ prevents overly large bonuses.) If the overall log p-value becomes positive, the peptide identification is discarded. We also tried a more principled p-value adjustment using an empirical model of the retention time residuals of false identifications, but the results were less satisfactory.

[1] There is some controversy in the literature over whether to use deltas, which depend on the entire database, rather than just the top-scoring identification. To remove this dependence, one possibility is to measure deltas by scoring variations of the top-scoring peptide.

Final Protein Scores. Final protein scores take into account correlations among peptide identifications. The most obvious correlation between peptide identifications is duplication. In almost any proteomic data set, abundant peptides are found in multiple spectra. As pointed out by Nesvizhskii *et al.* [22], duplication is not a sign of correctness, because spectra of the same peptide will generally give the same false match if the true match is not in the database. Indeed, we found almost the same percent of duplication (24% versus 22%) when we searched a set of spectra against forward and reverse proteins. Hence, ComByne, like ProteinProphet, discounts duplicates (even those with different parent charge states) and uses only the best p-value from a set of duplicate identifications.

Some peptide identifications, however, corroborate other identifications, so that observing them together increases the confidence of correctness. For example, the identification AHFSVMGDILSSAIR corroborates AHFSVM[+16]GDILSSAIR. We write $\text{Prob}[P] \cdot \text{Prob}[P' \mid P]$ for the probability of observing both a peptide P and a modified version P'. For the probability of observing P by chance, we use the p-value as described above. For the probability of observing P', conditioned on the observation of P, we use the better (that is, the smaller) of two p-values: the chance of observing P' as an independent observation (that is, the p-value determined by ByOnic's score and delta), and the empirically trained chance of observing such a corroboration, regardless of scores. The rationale here is that the observation of P' is an unlikely event, regardless of score, and if the score is good enough, then P' is "independent enough" to be counted in the same way as any other peptide from that protein. On three training sets of 10,000 spectra each, run against a 42-MByte database of reverse proteins, ByOnic matched a total of 8468 spectra to modified peptides, and only 10 of the 8468 were corroborated by unmodified peptides. So the p-value for corroboration in this case should be about $10/8468 = 0.0012$. To first approximation, the chance of corroboration scales linearly with the number of peptide identifications divided by the size of the database; ComByne makes this adjustment. An identification to a modified peptide can receive only one "corroboration bonus" (having its p-value replaced by a better one). Duplication takes precedence over corroboration, so that an observation of P and two observations of P' is no better than an observation of P and a single observation of P'. We do not give corroboration bonuses to deamidated peptides, with parent masses differing from the unmodified parent mass by 2 Daltons or less, as these identifications are not sufficiently independent, but neither do we discount deamidated peptides as duplicates.

Similarly, we defined and trained corroboration bonuses for the following situations: (1) a semitryptic peptide sharing its starting or ending position with a tryptic peptide (with a trained p-value of 0.0014 for a data set of 10,000 spectra and a 42-MByte database), (2) a nontryptic peptide completely contained within a tryptic peptide (p-value of 0.0012), (3) a tryptic peptide sharing its starting or ending position with a longer tryptic peptide (p-value of 0.0019), and (4) a phosphorylation and a loss of phosphate, for example S[+80] and S[-18], at the same site (p-value of 0.0006).

Finally, to account for the various sizes of proteins and remove the "large protein bias", we compute the expected sum of log p-values each protein in the database would receive from the set of spectra if all identifications were false, and start each protein with a compensating positive score (a handicap), so that if the protein receives its expected share of identifications, without any corroborations, its final score will be 0. Thus the final score for a protein can be interpreted as a log E-value, the overall p-value of the protein's peptide identifications, normalized for the number of spectra in the data set.

Modification Sites. In some studies, the "bottom line" is not protein identifications, but other information, such as modification sites within the proteins. ComByne has the capability to score phosphorylation sites by summing the p-values of all peptide identifications that include a phosphorylation modification at the site. Phosphorylation modifications include both phosphorylation and beta elimination, that is, S[+80], T[+80], Y[+80], S[-18], and T[-18]. (Phosphorylated tyrosine rarely undergoes the neutral loss.) For simplicity, we count [+80] and [-18] equally, even though the former is stronger and more direct evidence than the latter. As above, duplicate peptides are discounted, and retention times and corroborations are incorporated into peptide p-values. Because a modification site is a single location in the database, no handicap is necessary to correct for the varying lengths of proteins. It is tempting to make use of the log E-values of proteins in the modification-site ranking, but doing so spoils the estimation of the modification site false discovery rate (FDR) by preferentially removing reverse proteins.

3 Experiments

We tested ComByne on three trypsin-digested samples.

1. 1200 LC-ESI-QTOF spectra of human blood plasma from PPD Inc. (Menlo Park, CA). The sample was depleted of 6 abundant proteins using a multiple affinity removal system, and cysteines were carboxymethylated (+58).
2. Approximately 40,000 LC-ESI ion-trap (Thermo LTQ) spectra from PPD of mouse blood plasma (again depleted and carboxymethylated) spiked with low concentrations of 13 human proteins. The human proteins were produced recombinantly, and hence are contaminated only with low concentrations of *E.coli* proteins.
3. Approximately 240,000 LC/LC (MudPIT) ESI ion-trap (Thermo LTQ) spectra of human cell lysate, enriched for phosphopeptides, from the Yates laboratory at The Scripps Research Institute. Of these spectra, about 10,000 are MS^3 spectra, triggered by an $\mathrm{MS}^2/\mathrm{MS}^3$ scanning mode that looks for a strong peak in the MS^2 spectrum at $M-98$, $M-49$, or $M-(98/3)$, where M is the parent ion m/z, and then gates off ions of this mass for another round of fragmentation and mass measurement. Such a peak is characteristic of a phosphopeptide, which readily loses phosphoric acid (98 Daltons).

We first benchmarked ComByne versus a simple spectrum counting algorithm, with both protein ranking algorithms using peptide identifications computed by

ByOnic [5]. **Spectral Count** collates the matches for each protein. It sorts the proteins by the number of matches, breaking ties by total score, and then assigns each ambiguous peptide to its highest-ranking protein. Finally it sorts the proteins by the number of unique matches, again breaking ties by total score. Duplicate peptides do not contribute to the number of unique matches, but do contribute to the total score.

We then benchmarked the combination of ByOnic and ComByne versus Mascot and Spectral Count, Mascot and ProteinProphet, and X!Tandem and E-Value Product. **E-Value Product** computes the product of the E-values of peptide matches to each protein (including only matches with E-value at most 1.0, as recommended in the X!Tandem documentation), assigns each ambiguous peptide to its highest-ranking protein, and then recomputes the products of E-values. Only the best (minimum E-value) from a set of duplicate peptides contributes to the product. On X!Tandem's search results, E-Value Product consistently outperformed Spectral Count.

We evaluate ranked lists by FDR, empirically measured using reverse protein sequences. In this approach, ByOnic, Mascot, and X!Tandem search a database containing both the forward (N to C) and the reverse (C to N) sequence for each protein. If the list of the top 100 proteins includes 5 reverse proteins, then the list is also likely to include about 5 unknown false positives (forward proteins). Of course, this presumes that the ranking algorithm does not contain any unwitting biases against reverse proteins, e.g., implicit use of trigram frequencies.[2]

3.1 Human Blood Plasma

Table 1 gives the results for data set (1), the 1200 LC-ESI-QTOF spectra. We searched tryptic, semitryptic, and even nontryptic peptides, with the following common modifications enabled: oxidized M, H, W, deamidated N and Q, pyro-glu from N-terminal E and Q, and hydroxylated P. Human blood plasma has been studied quite intensively [1,2,27], using very sensitive LC/LC (multidimensional chromatography) assays, so expert inspection can be used to judge the plausibility of protein identifications. All of ComByne's identifications down to rank 60 are plausible, but ranks 61 and 65 are implausible. The implausible identifications start at the same point as the reverse sequences, and at an unimpressive log E-value. ComByne is not quite perfect on this human plausibility test, as it gives two known plasma proteins [27] lower ranks: Fibulin at rank 85 and Protocadherin at rank 87. ComByne, however, is much better than Spectral Count, which ranks many more known plasma proteins (including Ig kappa, Complement C8, Ficolin, Complement C1r, Serum amyloid A-4, Afamin, Fibulin, and Protocadherin) below the top reverse proteins.

On this data set, ComByne's advantage has nothing to do with its more advanced features, such as retention time or corroborations. We did not have retention times—or even elution order—for these spectra, and corroborations

[2] To guard against this possibility, we compared reverse sequences with decoys created to match the bigram and trigram frequencies of forward sequences. The tests gave essentially identical results.

Table 1. This table shows ComByne's proteins with ranks 40 to 65. With the exception of 59 (the enzyme used for digestion), proteins 40–60 are all abundant plasma proteins. Spec Count gives the rank of the same protein using the Spectral Count algorithm for protein ranking; the number in parentheses is the number of reverse proteins with higher ranks. For example, the Spectral Count algorithm gives Ig kappa rank 55, below 3 reverse proteins. A simple summarizing statistic is the rank of the highest reverse protein: 60 for ComByne and 50 for Spectral Count.

Rank	logE	Description	# Spec	# Uniq	Spec Count	
40	-12.34	Clusterin precursor	3	3	40	(0)
41	-11.94	Ig lambda chain C regions	2	2	43	(0)
42	-10.54	Ig kappa chain C region	2	1	55	(3)
43	-10.44	Ig gamma-3 chain C region	2	2	44	(0)
44	-10.18	Transthyretin (Prealbumin)	2	2	48	(0)
45	-9.80	Apolipoprotein C-I	3	3	39	(0)
46	-9.58	Alpha-2-antiplasmin	2	2	47	(0)
47	-8.53	Complement component C8 alpha	1	1	58	(5)
48	-8.31	Apolipoprotein C-II	2	2	46	(0)
49	-8.11	Ficolin-3	1	1	60	(5)
50	-8.01	Complement C1r subcomponent	1	1	62	(5)
51	-7.98	N-acetylmuramoyl-L-alanine amidase	1	1	59	(5)
52	-8.01	Complement factor I	1	1	61	(5)
53	-7.05	Plasma retinol-binding protein	2	2	45	(0)
54	-4.21	Complement C5 precursor	3	3	41	(0)
55	-3.94	Serum amyloid P-component	2	2	49	(0)
56	-3.11	Apolipoprotein M	1	1	63	(5)
57	-2.95	Serum amyloid A-4 protein	1	1	67	(7)
58	-2.69	Gelsolin precursor	2	2	51	(0)
59	-2.55	Trypsin I precursor	1	1	126	(31)
60	-2.54	Afamin precursor	1	1	76	(11)
61	-2.44	Tripartite motif protein	1	1	118	(27)
62	-1.50	Reverse Tubby-like protein 4	1	1	68	(7)
63	-1.48	Arginase-1 (Liver-type arginase)	1	1	65	(6)
64	-1.17	Reverse HLA class II	1	1	66	(6)
65	-1.06	Headcase protein homolog	1	1	71	(9)

played essentially no role in determining ranks 40–90, because these proteins are mostly "one-hit wonders" with single peptide identifications. ComByne's advantage derives almost entirely from its protein-length handicap and its careful mapping of ByOnic scores to p-values.

3.2 Spiked Mouse Plasma

For this experiment, the sample consisted of mouse blood plasma spiked with 13 soluble human proteins at concentrations of 1 microgram per milliliter. Many of the spiked proteins (leptin, prolactin, tumor necrosis factors 9, 11b, 13b, and α, and fibroblast growth factors 4, 5, 16, 17, and 19) are of clinical interest, and one spiked protein, human Apolipoprotein A-I, presents a "homolog problem" due to the abundant presence of mouse Apolipoprotein A-I. Thus this sample provides a realistic test bed for improving the sensitivity of analytical techniques, both chemical and computational, for the purpose of biomarker discovery. Again we searched all peptides, enabling oxidized M, H, W, deamidated N and Q, pyro-glu from N-terminal E and Q, and hydroxylated P.

Table 2 shows the number of forward proteins and the number of spiked proteins ranked above the fifth-best reverse protein for Spectral Count and three

Table 2. A comparison of four algorithms on the spiked mouse plasma. ComByne1 includes neither retention times nor corroborations, ComByne2 includes only retention times, and ComByne3 includes both. The plasma sample was divided into three replicates, each of which was run through the LC-ESI-MS/MS pipeline. The fourth row shows the spectra from all three runs combined. For each algorithm, we show the number of forward proteins and the number of spiked proteins that ranked above the fifth-best reverse protein. Thus on the first replicate, ComByne1 found 122 proteins and 9 spikes at 3.9% (= 5/127) false discovery rate, and Spectral Count found 93 and 4 at 5.1% FDR (= 5/98).

	ComByne1		ComByne2		ComByne3		Spectral Count	
	Proteins	Spikes	Proteins	Spikes	Proteins	Spikes	Proteins	Spikes
Rep 1	122	9	129	10	137	10	93	4
Rep 2	115	6	115	6	116	6	99	5
Rep 3	120	8	124	8	128	8	101	6
Pooled	140	9	143	9	144	9	96	4

versions of ComByne. We obtain similar results with other performance metrics. In this experiment, each of ComByne's advanced features plays a small role. The data set includes retention times, and as shown in the table, when we incorporate these times into the protein ranking, the performance improves.[3] Corroborations help only slightly, because the proteins of ranks 100–150, the "gray zone" of dubious identifications, are mainly one- to three-hit wonders with few chances for corroborations. Corroborations, however, are fairly common for abundant proteins. Some 17% of all identifications are modified peptides and 21% of all identifications are semitryptic peptides, and in each of these (overlapping) groups about one-third of the identifications are corroborated. Overall about 12% of all identifications are corroborated.

We also compared the combination of ByOnic and ComByne with three other combinations: (1) Mascot and Spectral Count, (2) Mascot and ProteinProphet, and (3) X!Tandem and E-Value Product. Because Mascot is relatively slow, we searched fully tryptic peptides only, with the same modifications as above, and for a fairer comparison, ComByne did not use retention time in computing protein ranks. Figure 3 gives the results in a graphical form, showing the number of reverse proteins as a function of the length of the list of proteins. The gray zone starts at the first reverse, and the "black zone" containing almost exclusively false identifications starts when the slope of the step function approaches 0.5, meaning that half of all identifications are reverse proteins. For closer comparison with X!Tandem, which uses a two-pass approach for peptide identification [9], we also ran ByOnic in two-pass mode, searching first for unmodified peptides, and then running the full search on a small protein database compiled in the first pass. As shown in Figure 3, two-pass search can improve protein sensitivity as well as running time. Finally, Figure 4 shows a more detailed comparison with

[3] The use of retention time for the pooled set of spectra is not entirely justified, because the spectra are from three separate chromatographic runs, but the runs are similar enough, and ComByne's algorithm, which sets the expected bonuses and penalties based on the spread of well-identified peptides, is robust enough that the use of retention time still helps.

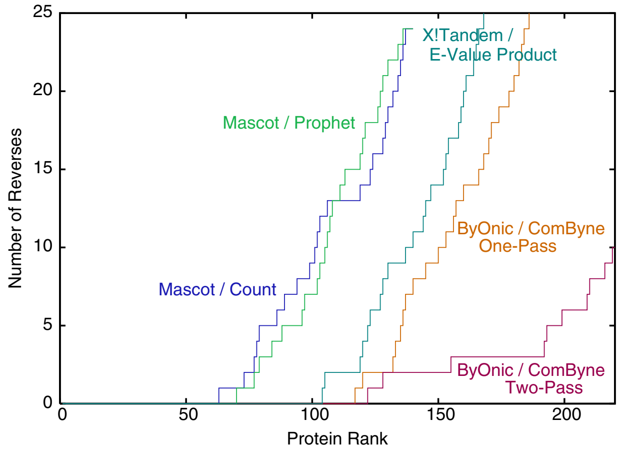


Fig. 3. A plot of the number of reverse proteins as a function of the length of the list of proteins, for a fully tryptic 8-modification search. Thus for Mascot with Spectral Count, the first reverse occurs at rank 63, the second at 73, and so forth.

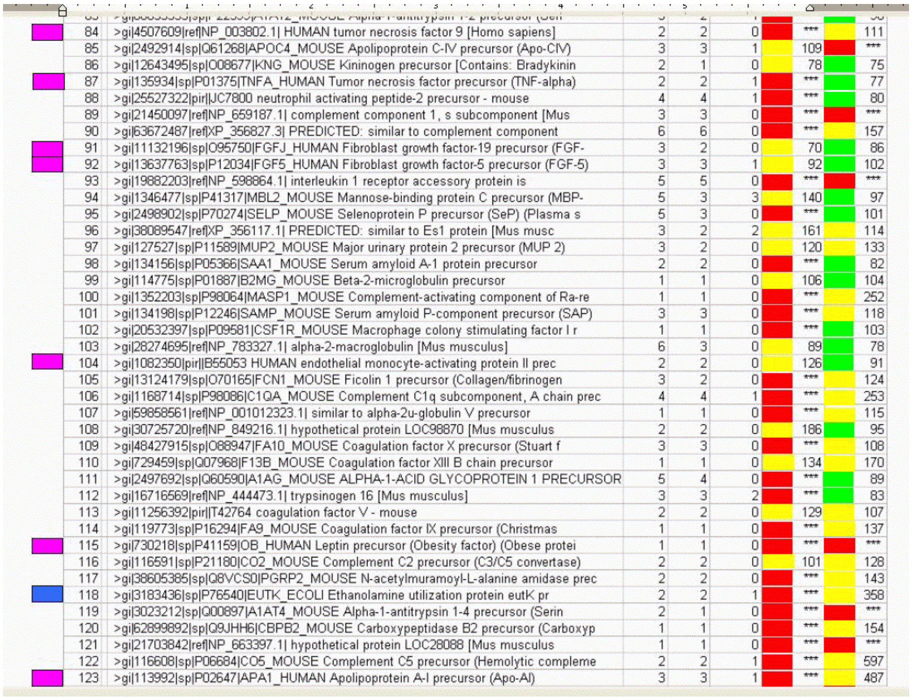
ByOnic/ComByne at its most sensitive (first reverse at rank 138), with ByOnic using a two-pass search and ComByne incorporating retention times.

3.3 Phosphorylation Sites

For data set (3), the "bottom line" is not protein identifications, but rather phosphorylation sites. Phosphoproteins tend to have a number of phosphorylation sites, often rather tightly clustered [6], and a single peptide identification can be the sole evidence for more than one site. To our knowledge, no other peptides-to-proteins integration tool provides the capability to score modification sites, so no exact comparison was possible.

We ran ComByne on all 200,000+ spectral identifications (ByOnic, all peptides, but only phospho modifications) at once without using retention time. These spectra are from 12 strong cation exchange salt fractions, and each chromatographic run includes interleaved MS/MS and MS^3 spectra, with MS^3 run on an MS/MS peak at the parent m/z minus 98, 49, or 32.7, characteristic of a neutral loss of phosphoric acid.[4] The highest ranked phosphorylation site from a reverse protein has rank 830, and there are only 4 reverse proteins among the top 2000 sites. In some cases, the existence of phosphorylation sites is clear, but the exact localization [6] remains ambiguous, for example, in the tryptic peptide AESSNSSSSDDSSEEEEEKLK from nucleolar phosphoprotein p130, aa's 349–369, we find strong evidence (large deltas) for phosphoserine at positions 3, 6, 7, 12, and 13, weaker evidence for positions 4 and 8, and no evidence for position 9.

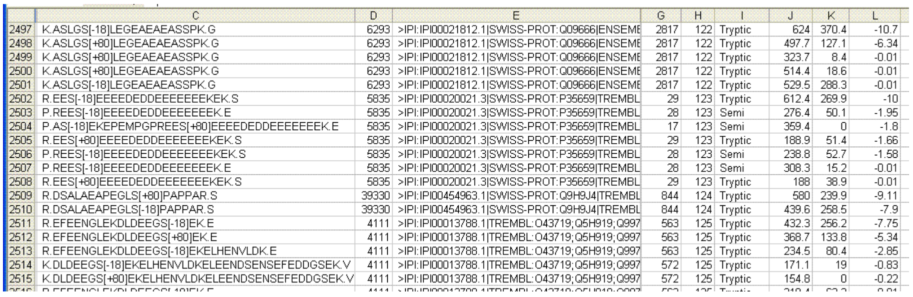
[4] We analyzed the MS^3 spectra as independent fragmentation spectra, but because their quality was often low, a better approach would analyze them along with their MS/MS progenitors by scoring each candidate against both spectra.



🟪	84	>gi|4507609|ref|NP_003802.1| HUMAN tumor necrosis factor 9 [Homo sapiens]	2	2	0	🟥	***	🟨	111
	85	>gi|2492914|sp|Q61268|APOC4_MOUSE Apolipoprotein C-IV precursor (Apo-CIV)	3	3	1	🟨	109	🟥	***
	86	>gi|12643495|sp|O08677|KNG_MOUSE Kininogen precursor [Contains: Bradykinin	2	1	0	🟨	78	🟩	75
🟪	87	>gi|135934|sp|P01375|TNFA_HUMAN Tumor necrosis factor precursor (TNF-alpha)	2	2	1	🟥	***	🟩	77
	88	>gi|25527322|pir||JC7800 neutrophil activating peptide-2 precursor - mouse	4	4	1	🟥	***	🟩	80
	89	>gi|21450097|ref|NP_659187.1| complement component 1, s subcomponent [Mus	3	3	0	🟥	***	🟥	***
	90	>gi|63672487|ref|XP_356827.3| PREDICTED: similar to complement component	6	6	0	🟥	***	🟨	157
🟪	91	>gi|11132196|sp|O95750|FGFJ_HUMAN Fibroblast growth factor-19 precursor (FGF-	3	2	0	🟨	70	🟩	86
🟪	92	>gi|13637763|sp|P12034|FGF5_HUMAN Fibroblast growth factor-5 precursor (FGF-5)	3	3	1	🟨	92	🟩	102
	93	>gi|19882203|ref|NP_598864.1| interleukin 1 receptor accessory protein is	5	5	0	🟥	***	🟥	***
	94	>gi|1346477|sp|P41317|MBL2_MOUSE Mannose-binding protein C precursor (MBP-	5	3	3	🟨	140	🟩	97
	95	>gi|2498902|sp|P70274|SELP_MOUSE Selenoprotein P precursor (SeP) (Plasma s	5	3	0	🟥	***	🟩	101
	96	>gi|38089547|ref|XP_356117.1| PREDICTED: similar to Es1 protein [Mus musc	3	2	2	🟨	161	🟨	114
	97	>gi|127527|sp|P11589|MUP2_MOUSE Major urinary protein 2 precursor (MUP 2)	3	2	0	🟨	120	🟨	133
	98	>gi|134156|sp|P05366|SAA1_MOUSE Serum amyloid A-1 protein precursor	2	2	0	🟥	***	🟩	82
	99	>gi|114775|sp|P01887|B2MG_MOUSE Beta-2-microglobulin precursor	1	1	0	🟨	106	🟩	104
	100	>gi|1352203|sp|P98064|MASP1_MOUSE Complement-activating component of Ra-re	1	1	0	🟥	***	🟨	252
	101	>gi|134198|sp|P12246|SAMP_MOUSE Serum amyloid P-component precursor (SAP)	3	3	0	🟥	***	🟨	118
	102	>gi|20532397|sp|P09581|CSF1R_MOUSE Macrophage colony stimulating factor I r	1	1	0	🟥	***	🟩	103
	103	>gi|28274695|ref|NP_783327.1| alpha-2-macroglobulin [Mus musculus]	6	3	0	🟨	89	🟩	78
🟪	104	>gi|1082350|pir||B55053 HUMAN endothelial monocyte-activating protein II prec	2	2	0	🟨	126	🟩	91
	105	>gi|13124179|sp|O70165|FCN1_MOUSE Ficolin 1 precursor (Collagen/fibrinogen	3	2	0	🟥	***	🟨	124
	106	>gi|1168714|sp|P98086|C1QA_MOUSE Complement C1q subcomponent, A chain prec	4	4	1	🟥	***	🟨	253
	107	>gi|59858561|ref|NP_001012323.1| similar to alpha-2u-globulin V precursor	1	1	0	🟥	***	🟨	115
	108	>gi|30725720|ref|NP_849216.1| hypothetical protein LOC98870 [Mus musculus	2	2	0	🟨	186	🟩	95
	109	>gi|48427915|sp|O88947|FA10_MOUSE Coagulation factor X precursor (Stuart f	3	3	0	🟥	***	🟨	108
	110	>gi|729459|sp|Q07968|F13B_MOUSE Coagulation factor XIII B chain precursor	1	1	0	🟨	134	🟨	170
	111	>gi|2497692|sp|Q60590|A1AG_MOUSE ALPHA-1-ACID GLYCOPROTEIN 1 PRECURSOR	5	4	0	🟥	***	🟩	89
	112	>gi|16716569|ref|NP_444473.1| trypsinogen 16 [Mus musculus]	3	3	2	🟥	***	🟩	83
	113	>gi|11256392|pir||T42764 coagulation factor V - mouse	2	2	0	🟨	129	🟨	107
	114	>gi|119773|sp|P16294|FA9_MOUSE Coagulation factor IX precursor (Christmas	1	1	0	🟥	***	🟨	137
🟪	115	>gi|730218|sp|P41159|OB_HUMAN Leptin precursor (Obesity factor) (Obese protei	1	1	0	🟥	***	🟥	***
	116	>gi|116591|sp|P21180|CO2_MOUSE Complement C2 precursor (C3/C5 convertase)	2	2	0	🟨	101	🟨	128
	117	>gi|38605385|sp|Q8VCS0|PGRP2_MOUSE N-acetylmuramoyl-L-alanine amidase prec	2	2	0	🟥	***	🟨	143
🟦	118	>gi|3183436|sp|P76540|EUTK_ECOLI Ethanolamine utilization protein eutK pr	2	2	1	🟥	***	🟨	358
	119	>gi|3023212|sp|Q00897|A1AT4_MOUSE Alpha-1-antitrypsin 1-4 precursor (Serin	2	1	0	🟥	***	🟥	***
	120	>gi|62899892|sp|Q9JHH6|CBPB2_MOUSE Carboxypeptidase B2 precursor (Carboxyp	1	1	0	🟥	***	🟨	154
	121	>gi|21703842|ref|NP_663397.1| hypothetical protein LOC28088 [Mus musculus	1	1	0	🟥	***	🟥	***
	122	>gi|116608|sp|P06684|CO5_MOUSE Complement C5 precursor (Hemolytic compleme	2	2	1	🟥	***	🟨	597
🟪	123	>gi|113992|sp|P02647|APA1_HUMAN Apolipoprotein A-I precursor (Apo-AI)	3	3	1	🟥	***	🟨	487

Fig. 4. This screenshot shows the proteins of ranks 84–123 for two-pass ByOnic and ComByne. A purple block indicates a spiked protein, known to be in the sample, and a blue block indicates an *E.coli* protein, a likely contaminant. The first three columns after the protein name give the numbers of peptides, unique peptides, and modified peptides. The columns on the right give the ranks for Mascot and Spectral Count and for X!Tandem and E-Value Product. A green block indicates that the protein ranked above the first reverse; a yellow block indicates that the protein was ranked below the first reverse; and a red block indicates that the protein was not detected. The column of red and yellow shows the relatively poor performance of Mascot/Count.

This data set was also analyzed by Lu *et al.* [19] using SEQUEST and DTASelect. DTASelect ranks sites at the spectrum level (as in [6]); thus each site is ranked according to only its best spectrum. SEQUEST/DTASelect gave 804, 875, and 1015 phosphorylation sites at 1%, 2%, and 5% FDR (matches to reverse proteins), whereas ByOnic/ComByne give 2000 sites at 0.2% FDR. We believe that these numbers (especially our 0.2%) are underestimates, because in this type of search, other modifications are more likely to match phosphorylations than they are to match wholly incorrect peptides. Lu *et al.* [19] developed a binary classifier called DeBunker for distinguishing true and false phosphopeptide identifications. DeBunker assigns a probability of correctness greater than 0.9 to about 1100 of ComByne's top 2000 sites, confirming that ByOnic/ComByne is indeed finding more sites than SEQUEST/DTASelect. We have also validated many of ComByne's phosphorylation sites using two databases, Phosphosite (`www.phosphosite.org`) and Phospho.ELM [12] (`phospho.elm.eu.org`).



	C	D	E	G	H	I	J	K	L
2497	K.ASLGS[-18]LEGEAEAEASSPK.G	6293	>IPI:IPI00021812.1|SWISS-PROT:Q09666|ENSEM	2817	122	Tryptic	624	370.4	-10.7
2498	K.ASLGS[+80]LEGEAEAEASSPK.G	6293	>IPI:IPI00021812.1|SWISS-PROT:Q09666|ENSEM	2817	122	Tryptic	497.7	127.1	-6.34
2499	K.ASLGS[+80]LEGEAEAEASSPK.G	6293	>IPI:IPI00021812.1|SWISS-PROT:Q09666|ENSEM	2817	122	Tryptic	323.7	8.4	-0.01
2500	K.ASLGS[+80]LEGEAEAEASSPK.G	6293	>IPI:IPI00021812.1|SWISS-PROT:Q09666|ENSEM	2817	122	Tryptic	514.4	18.6	-0.01
2501	K.ASLGS[-18]LEGEAEAEASSPK.G	6293	>IPI:IPI00021812.1|SWISS-PROT:Q09666|ENSEM	2817	122	Tryptic	529.5	288.3	-0.01
2502	R.EES[-18]EEEEDEDDEEEEEEEKEK.S	5835	>IPI:IPI00020021.3|SWISS-PROT:P35659|TREMBL	29	123	Tryptic	612.4	269.9	-10
2503	P.REES[-18]EEEEDEDDEEEEEEEK.E	5835	>IPI:IPI00020021.3|SWISS-PROT:P35659|TREMBL	28	123	Semi	276.4	50.1	-1.95
2504	P.AS[-18]EKEPEMPGPREES[+80]EEEEDEDDEEEEEEEK.E	5835	>IPI:IPI00020021.3|SWISS-PROT:P35659|TREMBL	17	123	Semi	359.4	0	-1.8
2505	R.EES[+80]EEEEDEDDEEEEEEEKEK.S	5835	>IPI:IPI00020021.3|SWISS-PROT:P35659|TREMBL	29	123	Tryptic	188.9	51.4	-1.66
2506	P.REES[-18]EEEEDEDDEEEEEEEKEK.S	5835	>IPI:IPI00020021.3|SWISS-PROT:P35659|TREMBL	28	123	Semi	238.8	52.7	-1.58
2507	P.REES[-18]EEEEDEDDEEEEEEEK.E	5835	>IPI:IPI00020021.3|SWISS-PROT:P35659|TREMBL	28	123	Semi	308.3	15.2	-0.01
2508	R.EES[+80]EEEEDEDDEEEEEEEKEK.S	5835	>IPI:IPI00020021.3|SWISS-PROT:P35659|TREMBL	29	123	Tryptic	188	38.9	-0.01
2509	R.DSALAEAPEGLS[+80]PAPPAR.S	39330	>IPI:IPI00454963.1|SWISS-PROT:Q9H9J4|TREMBL	844	124	Tryptic	580	239.9	-9.11
2510	R.DSALAEAPEGLS[-18]PAPPAR.S	39330	>IPI:IPI00454963.1|SWISS-PROT:Q9H9J4|TREMBL	844	124	Tryptic	439.6	258.5	-7.9
2511	R.EFEENGLEKDLDEEGS[-18]EK.E	4111	>IPI:IPI00013788.1|TREMBL:O43719;Q5H919;Q997	563	125	Tryptic	432.3	256.2	-7.75
2512	R.EFEENGLEKDLDEEGS[+80]EK.E	4111	>IPI:IPI00013788.1|TREMBL:O43719;Q5H919;Q997	563	125	Tryptic	368.7	133.8	-5.34
2513	R.EFEENGLEKDLDEEGS[-18]EKELHENVLDK.E	4111	>IPI:IPI00013788.1|TREMBL:O43719;Q5H919;Q997	563	125	Tryptic	234.5	80.4	-2.85
2514	K.DLDEEGS[-18]EKELHENVLDKELEENDSENSEFEDDGSEK.V	4111	>IPI:IPI00013788.1|TREMBL:O43719;Q5H919;Q997	572	125	Tryptic	171.1	19	-0.83
2515	K.DLDEEGS[+80]EKELHENVLDKELEENDSENSEFEDDGSEK.V	4111	>IPI:IPI00013788.1|TREMBL:O43719;Q5H919;Q997	572	125	Tryptic	154.8	0	-0.22

Fig. 5. ComByne outputs a list of phosphorylation sites, ranked by p-value, and a matched list of peptide id's supporting the site id's. This screenshot shows the latter list in Microsoft Excel. Column C gives the phosphopeptide, D the protein number in the database, E the accession number and name, G the position of the peptide within the protein, H ComByne's rank (these peptide id's support sites 122–125), J and K ByOnic's score and delta (as a score drop, not a percent), and L ComByne's log p-value (with −0.01 indicating a duplicate peptide). All four of the sites shown in this screenshot (from Desmoyokin, USP42, DEK, and TAT-SF1) appear in either Phosphosite or Phospho.ELM, curated databases of phosphorylation sites.

4 Discussion

We have explored the utility of various ideas in improving the integration of spectral identifications into higher-level, more biologically meaningful, proteomic identifications. As is often the case, the most straightforward ideas proved to be the most effective. For example, more careful mapping of database-search scores to p-values turned out to be more important than corroboration bonuses. In our case, the scores were the output of a new program called ByOnic, but we expect that the same observation holds true for SEQUEST and Mascot scores.

Corroboration bonuses are somewhat related to a new peptide identification approach, called "spectral networks analysis" [3], in which pairwise similarities—with mass shifts—between unidentified spectra are used to help make peptide identifications in difficult (nontryptic, modified, or mutated) data sets. The pairwise similarities are discovered by "blind" searching, that is, without knowledge that certain shifts such as +16 and, in the case of phosphopeptides, −98, are more likely than other integers. Corroboration is a way to use anticipated pairwise similarities, found by conventional database-search tools, to bolster the confidence of low-scoring identifications that might otherwise be discounted or discarded.

Acknowledgements

We thank Chris Becker, Hua Lin, Tom Shaler, and Ted Jones (PPD, Inc.) and John R. Yates, III, Bingwen Lu, and Cristian Ruse (The Scripps Research Institute) for valuable data and discussions.

References

1. J.N. Adkins, S.M. Varnum, K.J. Auberry, R.J. Moore, N.H. Angell, R.D. Smith, D.L. Springer, J.G. Pounds. Toward a human blood serum proteome: analysis by multidimensional separation coupled with mass spectrometry. *Molecular and Cellular Proteomics* 1 (2002), 947–955.
2. N.L. Anderson, M. Polanski, R. Pieper, T. Gatlin, R.S. Tirumalai, T.P. Conrads, T.D. Veenstra, J.N. Adkins, J.G. Pounds, R. Fagan, and A. Lobley. The human plasma proteome. *Molecular and Cellular Proteomics* 3 (2004), 311–326.
3. N. Bandeira, D. Tsur, A. Frank, and P. Pevzner. A new approach to protein identification. *RECOMB 2006*, LNCS 3909, Springer, 2006, 363–378.
4. M. Bern and D. Goldberg. EigenMS: De novo analysis of peptide tandem mass spectra by spectral graph partitioning. *J. Comp. Biology* 13 (2006), 364–378.
5. M. Bern, Y. Cai, and D. Goldberg. Lookup peaks: a hybrid of de novo sequencing and database search for protein identification by tandem mass spectrometry. *Anal. Chem.* 79 (2007).
6. S.A. Beausoleil, J. Villén, S.A. Gerber, J. Rush, and S.P. Gygi. A probability-based approach for high-throughput protein phosphorylation analysis and site localization. *Nature Biotechnology* 24 (2006).
7. K.R. Coombes, J.S. Morris, J. Hu, S.R. Edmonson, and K.A. Baggerly. Serum proteomics profiling—a young technology begins to mature. *Nature Biotechnology* 23 (2005), 291–292.
8. R. Craig and R.C. Beavis. TANDEM: matching proteins with mass spectra. *Bioinformatics* 20 (2004), 1466–1467.
9. R. Craig and R.C. Beavis. A method for reducing the time required to match protein sequences with tandem mass spectra. *Rapid Commun. Mass Spectrometry* 17 (2003), 2310–2316.
10. D.M. Creasy and J.S. Cottrell. Error tolerant searching of uninterpreted tandem mass spectrometry data. *Proteomics* 2 (2002), 1426–1434.
11. D.M. Creasy and J.S. Cottrell. Unimod: Protein modifications for mass spectrometry. *Proteomics* 4 (2004), 1534–1536.
12. F. Diella, S. Cameron, C. Gemünd, R. Linging, A. Via, B. Kuster, T. Sicheritz-Pontén, N. Blom, and T.J. Gibson. Phospho.ELM: A database of experimentally verified phosphorylation sites in eukaryotic proteins. *BMC Bioinf.* 2004, 5:79.
13. J.K. Eng, A.L. McCormack, and J.R. Yates, III. An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *J. Am. Soc. Mass Spectrom.* 5 (1994), 976–989.
14. A. Frank and P. Pevzner. PepNovo: De novo peptide sequencing via probabilistic network modeling. *Anal. Chem.* 77 (2005), 964–973.
15. L.Y. Geer, S.P. Markey, J.A. Kowalak, L. Wagner, M. Xu, D.M. Maynard, X. Yang, W. Shi, and S.H. Bryant. Open mass spectrometry search algorithm. *J. Proteome Research* 3 (2004), 958–964.
16. E.A. Kapp, *et al.* An evaluation, comparison, and accurate benchmarking of several publicly available MS/MS search algorithms: sensitivity and specificity analysis. *Proteomics* 5 (2005), 3226–3245.
17. A. Keller, A.I. Nesvizhskii, E. Kolker, and R. Aebersold. Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search. *Anal. Chem.* 74 (2002), 5383–5392.

18. O.V. Krokhin, R. Craig, V. Spicer, W. Ens, K.G. Standing, R.C. Beavis, and J.A. Wilkins. An improved model for prediction of retention times of tryptic peptides in ion pair reversed-phase HPLC. *Mol. Cell. Proteomics* 3.9 (2004), 908–919.
19. B. Lu, C. Ruse, T. Xu, S.K. Park, and J. Yates, III. Automatic validation of phosphopeptide identifications from tandem mass spectra. *Anal. Chem.* 79 (2007).
20. B. Ma, K. Zhang, C. Hendrie, C. Liang, M. Li, A. Doherty-Kirby, and G. Lajoie. PEAKS: powerful software for peptide de novo sequencing by tandem mass spectrometry. *Rapid Comm. in Mass Spectrometry* 17 (2003), 2337–2342.
21. R.E. Moore, M.K. Young, and T.D. Lee. Qscore: an algorithm for evaluating SEQUEST database search results. *J. Am. Soc. Mass Spec.* 13 (2002), 378–386.
22. A.I. Nesvizhskii, A. Keller, E. Kolker, and R. Aebersold. A statistical model for identifying proteins by tandem mass spectrometry. *Anal. Chem.* 75 (2003), 4646–4658.
23. G.S. Omenn, *et al.* Overview of the HUPO plasma proteome project: results from the pilot phase with 35 collaborating laboratories and multiple analytical groups, generating a core dataset of 3020 proteins and a publicly-available database. *Proteomics* 5 (2005), 3226–3245.
24. D.N. Perkins, D.J.C. Pappin, D.M. Creasy, and J.S. Cottrell. Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis* 20 (1999), 3551-3567.
25. K.A. Resing, *et al.* Improving reproducibility and sensitivity in identifying human proteins by shotgun proteomics. *Anal. Chem.* 76 (2004), 3556–3568.
26. N. Rifai, M.A. Gillette, and S.A. Carr. Protein biomarker discovery and validation: the long and uncertain path to clinical utility. *Nature Biotechnology* 24 (2006), 971–983.
27. D.J. States, G.S. Omenn, T.W. Blackwell, D. Fermin, J. Eng, D.W. Speicher, and S.M. Hanash. Challenges in deriving high-confidence protein identifications from data gathered by a HUPO plasma proteome collaborative study. *Nature Biotechnology* 24 (2006), 333–338.
28. D.L. Tabb, W.H. McDonald, and J.R. Yates, III. DTASelect and Contrast: Tools for assembling and comparing protein identifications from shotgun proteomics. *J. Proteome Research* 1 (2002), 21–26.
29. D.L. Tabb, A. Saraf, and J.R. Yates, III. GutenTag: High-throughput sequence tagging via an empirically derived fragmentation model. *Anal. Chem.* 75 (2003), 6415–6421.
30. S. Tanner, H. Shu, A. Frank, L-C. Wang, E. Zandi, M. Mumby, P.A. Pevzner, and V. Bafna. InsPecT: Identification of posttranslationally modified peptides from tandem mass spectra. *Anal. Chem.* 77 (2005), 4626–4639.
31. J.R. Yates, III, J. Eng, A. McCormack, and D. Schietz. A method to correlate tandem mass spectra of modified peptides to amino acid sequences in a protein database. *Anal. Chem.* 67 (1995), 1426–1436.
32. C.-Y. Yen, S. Russell, A.M. Mendoza, K. Meyer-Arendt, S. Sun, K.J. Cios, N.G. Ahn, and K.A. Resing. Improving sensitivity in shotgun proteomics using a peptide-centric database with reduced complexity: protease cleavage and SCX elution rules from data mining of MS/MS spectra. *Anal. Chem.* 78 (2006), 1071–1084.
33. W. Zhang and B.T. Chait. ProFound: an expert system for protein identification using mass spectrometric peptide mapping information. *Anal. Chem.* 72 (2000), 2482–2489.

Peptide Retention Time Prediction Yields Improved Tandem Mass Spectrum Identification for Diverse Chromatography Conditions

Aaron A. Klammer, Xianhua Yi,
Michael J. MacCoss, and William Stafford Noble

Genome Sciences Department
University of Washington
1705 NE Pacific Street
Seattle, WA 98195-5065
{aklammer,xhyi,maccoss,noble}@gs.washington.edu

Abstract. Most tandem mass spectrum identification algorithms use information only from the final spectrum, ignoring precursor information such as peptide retention time (RT). Efforts to exploit peptide RT for peptide identification can be frustrated by its variability across liquid chromatography analyses. We show that peptide RT can be reliably predicted by training a support vector regressor on a *single* chromatography run. This dynamically trained model outperforms a published statically trained model of peptide RT across diverse chromatography conditions. In addition, the model can be used to filter peptide identifications that produce large discrepancies between observed and predicted RT. After filtering, estimated true positive peptide identifications increase by as much as 50% at a false discovery rate of 3%, with the largest increase for non-specific cleavage with elastase.

Keywords: Mass spectrometry, proteomics, peptide identification, retention time, chromatography, machine learning, support vector regression.

1 Introduction

Full understanding of the cell requires accurate measurement and characterization of its main biochemical actors, proteins. While much can be learned from the study of individual proteins, *in vivo* a protein invariably acts in concert with other biomolecules. These interactions differ according to cell type, the state of the cell, and its response to external stimuli. Several technologies have the potential to provide a comprehensive view of many or all of an the cell's proteins. One such technology is shotgun proteomics using liquid chromatography and tandem mass spectrometry (LC-MS/MS)[1] (McCormack et al., 1997; Yates, III, 1998).

[1] Abbreviations used in this manuscript include retention time (RT), liquid chromatography (LC), mass spectometry (MS), tandem mass spectrometry (MS/MS), support vector regressor (SVR), artificial neural network (ANN) and false discovery rate (FDR).

In a typical liquid chromatography (LC)-MS/MS experiment (Figure 1A), proteins are digested to peptides, and the peptides are separated by LC on a reverse phase column in order of increasing hydrophobicity. The peptides elute into the mass spectrometer, where tandem mass spectrometry (MS/MS) measures the mass-to-charge ratio of the intact and fragmented peptides, yielding a tandem mass spectrum. One LC-MS/MS experiment yields tens of thousands of MS/MS spectra. The identity of the peptides that produced the spectra, and thus the identity of the original proteins, can be automatically deduced by a database search algorithm such as SEQUEST (Eng et al., 1994).

As with any high-throughput technology, shotgun proteomics practioners must constantly battle false positive identifications (Cargile et al., 2004; Qian et al., 2005). The need to reduce false positives has spurred a proliferation of methods for increasing peptide identification confidence. However, most of these methods use information exclusively from the MS and MS/MS stages of analysis, ignoring information from the LC stage, such as retention time (RT). RT is the amount of time that a peptide is retained on the LC column (Figure 1B, top). It has the advantage of being almost entirely independent of the information contained in the MS/MS scan, and can therefore be used to increase peptide identification confidence.

The goal of this paper is to incorporate RT into the peptide identification process to increase peptide identification confidence. Previous efforts along these lines have been hindered by RT variability, even on identical columns or multiple runs of the same sample (Palmblad et al., 2004). Most such methods train a single RT predictor using a limited subset of highly-reproducible chromatography conditions (Krokhin et al., 2004), or perform a normalization that attempts to eliminate variability (Petritis et al., 2006; Strittmatter et al., 2004). In practice, however, researchers use a large number of diverse chromatographic conditions, making a static RT predictor less useful. In this work, we demonstrate how to dynamically train a support vector regressor (SVR) to predict RT for peptides in a given chromatographic analysis, using only data generated during the current run using composition related features (Figures 2 and 1B, bottom).

This approach makes the method portable to new chromatography conditions or sample preparation protocols, adapting to differences in column length, digestion condition, peptide chemistry and MudPIT salt step. Our RT predictions are better correlated with observed RT than those produced by a static predictor trained on different data. Furthermore, by eliminating peptide identifications with an observed retention time that deviates greatly from predicted retention time, our method increases the number of true positive peptide identifications over a range of false discovery rates. For one data set digested with a non-standard enzyme (elastase), we demonstrate an increase of approximately 50% in true positives at a false discovery rate (FDR) of 3%. This result compares favorably with (Strittmatter et al., 2004) (a true positive increase of 15% at 3% FDR, from Table 2), but with much less training data. Thus, the results presented here have implications both for traditional shotgun proteomics research using trypsin, as well as possibly enabling new strategies using non-standard enzymes.

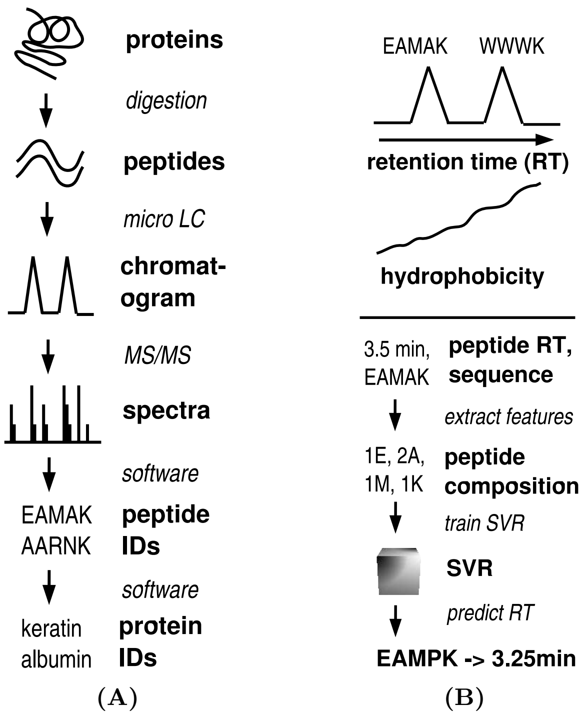


Fig. 1. Experimental overview. (A) In mass spectrometry, proteins are digested to peptides, which are then separated using liquid chromatography and analyzed in a tandem mass spectrometer. The experimental procedure yields a chromatogram, MS and MS/MS spectra, and (ideally) peptide and protein identifications. (B, top) For reverse phase chromatography, each peak in a chromatogram corresponds to a peptide retained on the column for an amount of time that depends on its hydrophobicity. (B, bottom) We train a support vector regressor with composition-related features to predict RT for unknown peptides from the *same* chromatography run.

1.1 Related Work

Understanding and predicting peptide RT has a long history. For reverse phase chromatography, peptide RT is roughly proportional to peptide hydrophobicity (Frenz et al., 1990). Many models assume that peptide RT is a linear function of peptide amino acid composition (Meek, 1980; Browne et al., 1982; Guo et al., 1987; Hearn et al., 1988; Bihan et al., 2004). More recent models augment the compositional approach with parameters for peptide length or mass (Mant et al., 1989), or terms for residue context (Mant and Hodges, 2006) or positional effects such as the identity of the N-term residue (Krokhin et al., 2004). Still more sophisticated models include parameters for structural features or measured chemical properties (Bączek et al., 2005; Petritis et al., 2006).

The most accurate and sophisticated peptide RT predictor is that of Petritis et al. (2006), first presented in simpler form in Petritis et al. (2003),

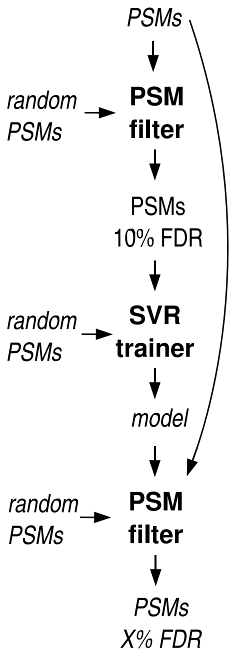


Fig. 2. Overview of data flow for peptide identification improvement. For each LC-MS/MS experiment, we start with a collection of peptide-spectrum matches (PSMs, top). We use PSMs to a random proteome to filter the original PSMs, producing high-confidence PSMs at a 10% false discovery rate. These filtered PSMs, along with a second set of random PSMs, are used to train a support vector regressor and to select a threshold for filtering PSMs based on their retention time, yielding a trained model. The model and a third set of random PSMs are used to select the final set of real PSMs at a desired FDR. Not shown is the five-fold cross-validation used to validate this method.

which uses an artificial neural network (ANN) to predict a normalized form of RT. The large amount of data required to train the ANN (for Petritis et al. (2006) 345,000 nonredundant peptides) makes retraining for new chromatography conditions impractical. Although one could in theory transform predicted RT values for different conditions, it is not clear how to handle changes in peptide elution order. A more recent, but less complicated, ANN has since been published (Shinoda et al., 2006).

Recently, a handful of RT predictors have found practical application in enhancing confidence of peptide identifications. Palmblad et al. (2002) predict RT for each peptide using least-squares regression to determine amino acid weights, and then use a χ^2 test to rank candidate peptides based on deviation from expected mass and predicted RT. As the authors admit, their RT prediction is poor compared to other competing efforts, and improvement in protein identification is modest.

Kawakami et al. (2005) use the sum of residue retention coefficients to predict RT for peptides and phosphopeptides, but they make no clear distinction between training and test data. When predictions are made on data not used in training, the correlation between predicted and observed RT deteriorates, a sign of overfitting.

By far, Strittmatter et al. (2004) is the most successful example of using RT prediction to increase peptide identification confidence. They use the ANN of Petritis et al. (2003) to exclude all SEQUEST peptide identifications with predicted normalized RT that deviates more than 10% from observed normalized RT. The result is roughly a 50% decrease in estimated false positives, although true positives also decrease, frustrating a straightforward interpretation.

The methods presented in this paper yield a reduction in false positives as great as that in Strittmatter et al. (2004), but require reduced training data, produced from a single LC-MS/MS run.

2 Methods

2.1 Data Sets

We analyze eight separate data sets (Table 1), chosen to represent a diverse set of chromatography conditions. Exact sample preparation protocols are given in the supplement (`noble.gs.washington.edu/proj/rt`); here we give only brief descriptions to highlight major differences. All data sets are from the yeast *Saccharomyces Cerevisiae*. The first three data sets are taken from the middle and end of a 12-hour, 6-step, 2-phase strong cation exchange and reverse phase multi-dimensional protein identification technology (MudPIT) analysis (Washburn et al., 2001) of a tryptic digest of the soluble *S. Cerevisiae* proteome. The MudPIT was performed with C18 beads, while all subsequent analyses are with C12 beads. The number after the C refers to the length of the carbon chain on the beads to which the peptides bind. Different length chains interact with the peptides differently. The next three data sets are reverse phase analyses of a tryptic digest of the soluble yeast proteome, each with a different length column of 20cm, 40cm and 60cm. A fourth identically prepared yeast sample was analyzed with the ion-pairing agent trifluoroacetic acid (TFA). The two final data sets are from yeast samples digested with the non-specific enzymes chymotrypsin or elastase. Chymotrypsin cleaves after aromatic residues F, W, and Y, and elastase cleaves after small hydrophobic residues A, L, I and V. Summary statistics for the data sets, and for the training and testing data sets extracted from them, are shown in Table 1.

2.2 Training and Testing Set Extraction

A high-confidence set of training and testing data is extracted from each of the eight data sets. The spectra are first searched against both the real and shuffled versions of the *S. cerevisiae* proteome with SEQUEST (Eng et al., 1994) and then identifications are filtered using the following criteria: charge state of +2, peptide sequence ending in K or R (except for the chymotrypsin and elastin data sets), and allowing any number of missed tryptic cleavages.

We use the number of matches in the search against a shuffled proteome as an estimate of false positive matches in a search against the real proteome. Both searches use identical search criteria. High-confidence spectra identifications are selected by setting an Xcorr threshold so that the number of matches to the shuffled proteome above this threshold is 10% of the number of matches to the real proteome; this is equivalent to a 10% FDR. If the number of real matches at a 10% FDR is less than 200, then the top 200 spectra are used; this is because regression performance deteriorated with less than 200 spectra. When multiple spectra matched a single peptide according to these criteria, the spectrum with the highest Xcorr is selected, to avoid bias in the regression towards common peptides. The resulting set of peptides and retention times is split to form a 3:1 ratio between the training and testing data sets (Table 1) for each chromatography run, which are then used to train and test the SVR. No peptides are allowed to occur in both the training and testing data sets.

2.3 Support Vector Regression

As with other forms of regression, an SVR learns a function that relates a dependent variable (in this case, RT) to a set of independent variables. An SVR builds a regressor out of a subset of the training examples, known as support vectors. Training examples that are within a tolerance value ϵ of the model prediction are ignored (Vapnik, 1995). To generate the independent variables, each peptide from the training and test sets is represented as a 63-element vector comprised of the following: 20 elements to represent the total number of each amino acid residue in the peptide; 40 binary elements to represent the identity of the extreme N-terminal (N-term) and penultimate C-terminal (C-term) residues, respectively; and three additional elements to represent the identity of the last C-term residue (either K or R), and the peptide length and mass. For the non-specific enzymes, the ultimate C-terminal residue is used instead of the penultimate, and the K or R term is set to zero.

An SVR is trained on each high-quality training set and tested by measuring the R value between predicted and observed RT on a held-out test set. R value is a statistical measure of the correlation between two data sets. The R value for two data sets x and y of length n is given by $r = Cov(x, y)/\sigma_x\sigma_y$, where $Cov(x, y) = n\sum xy - \sum x \sum y$, the covariance of data sets x and y, and $\sigma_x = \sqrt{n\sum x^2 - (\sum x)^2}$, the standard deviation of dataset x. It is important to note that a separate SVR is trained for each data set in Table 1.

The SVR is trained and tested twice using two kinds of kernels: a linear kernel, because it allows ready interpretion of the weight it assigns to each feature (Section 3.2); and a Gaussian kernel (also known as a radial-basis function kernel), because it allows maximum flexibility in the functions that it can successfully regress.

Hyperparameters for each kernel are chosen by three-fold cross-validation on the training set. For both kernels, the SVR is trained with an ϵ insensitive-loss hyperparameter of 0.1; other values of ϵ did not yield radically different results. Another hyperparameter used in the regression is the soft-margin penalty C, which can be thought of as a bound on the weight that can be given to each training example. C was initially allowed to range over ten orders of magnitude from 10^{-3} to 10^{7}. For the final cross-validation, to decrease processing time, C is constrained to be 10^{-1}, 10^{0}, or 10^{1} for the linear kernel, and 10^{5}, 10^{6} or 10^{7} for the Gaussian kernel. The Gaussian kernel has an additional hyperparameter σ, which corresponds to the width of the Gaussians used; it is set to 10^{-6}, 10^{-7} and 10^{-8}. R values are reported after hyperparameter selection on the appropriate held-out test set (Table 1).

The SVR is implemented using the publicly available software package PyML (`pyml.sourceforge.net`). Source code for producing the results presented here can be found at `http://noble.gs.washington.edu/proj/rt`.

Table 1. Eight data sets used to train and test the support vector regressor. Each column lists the total number of +2 spectra associated with peptides that satisfy that data set's trypticity requirements (Total), the number of high-confidence spectra selected at a 10% FDR (Confident), and the subsets of the high-confidence spectra used to train and test the performance of the regressor.

Data set	Total	Confident	Train	Test
Y-20CM	6929	2073	1554	519
Y-40CM	7220	2409	1806	603
Y-60CM	7459	2774	2080	694
Y-TFA	11977	3179	2384	795
Y-CHYMO	2191	200	150	50
Y-ELAST	4377	200	150	50
Y-MUDPIT-1	2227	280	210	70
Y-MUDPIT-2	3035	485	363	122

3 Results

3.1 Support Vector Regression

We first evaluate our dynamically trained regressor by comparing it to a published, fixed-parameter regressor from Krokhin et al. (2004). We measure performance by comparing correlation (measured by R value) between observed and predicted RT for our SVR with the correlation between observed and predicted relative hydrophobicity from the fixed-parameter regression. One of the kernels (either Gaussian or linear kernel) outperforms the fixed parameter regression across all data sets (Table 2 and Figure 3). Furthermore, the performance of the fixed and learned regressors are qualitatively the same: data sets that had relatively poor correlation for one method had similarly poor correlation for the other. In general, the regression performs best on data sets with a large number of high-confidence identifications (Table 1).

Table 2. R values for a fixed regression compared to a learned regression using the Gaussian or linear kernels. Correlation for eight data sets for fixed parameters described in Krokhin et al. (2004) (Fixed) and parameters learned for each dataset with a Gaussian or linear kernel. The Fixed values differ in the first and third columns because they are evaluated on slightly different randomly selected subsets of the high-confidence PSMs.

Data set	Fixed	Gaussian	Fixed	Linear
20CM	0.881	0.908	0.877	0.892
40CM	0.892	0.897	0.894	0.891
60CM	0.914	0.926	0.889	0.892
CHYMO	0.871	0.865	0.761	0.792
ELAST	0.823	0.850	0.843	0.856
TFA	0.818	0.842	0.882	0.905
MUDPIT-1	0.743	0.783	0.797	0.850
MUDPIT-2	0.806	0.803	0.791	0.828

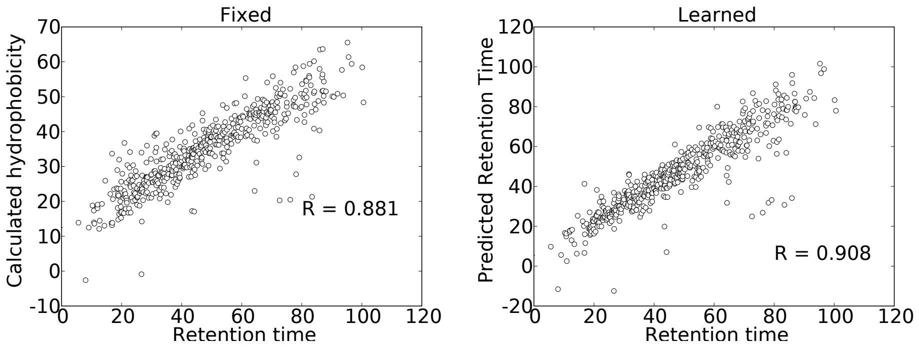


Fig. 3. Example of retention time prediction. Predictions of hydrophobicity, a proxy for retention time (RT), made by a fixed parameter linear regression from Krokhin et al. (2004) (left) are less accurate than RT predictions by a support vector regression that is trained and tested on subsets of data from the same chromatography run (right).

3.2 Residue Weights

An advantage of using a linear kernel for the SVR is that it allows calculation of the weights for each feature, using the following formula:

$$\hat{w} = \sum_i \alpha_i \hat{x}_i \tag{1}$$

where $\hat{w}$ is the feature weight vector, $\hat{x}_i$ is the ith training example (in this case, the 63-element vector representing a peptide), and α_i is the weight associated with the ith training example by the SVR. Weights correspond to the feature's relative contribution to retention time. After performing the regression on each data set, we calculate the weights given to each residue for peptide composition, shown in

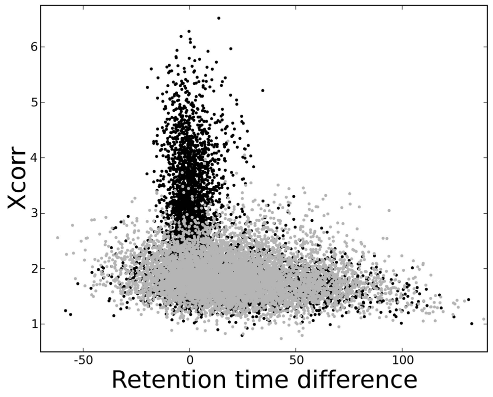


Fig. 4. Predicted retention time difference for real and random peptide-spectrum matches. Shown are the Xcorr values and difference between observed RT and RT predicted by the Gaussian kernel for matches to the real yeast proteome (black) and the shuffled yeast proteome (gray) for the 20CM data set.

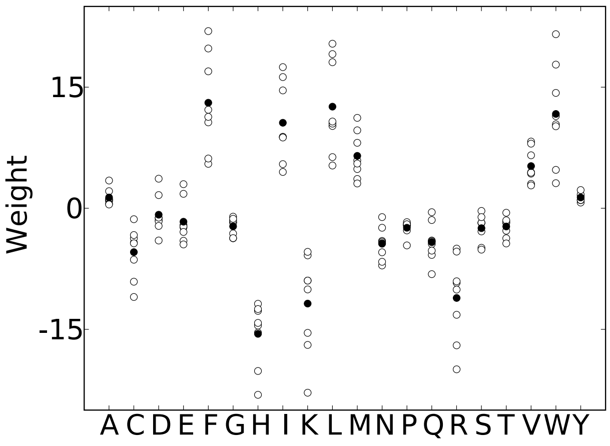


Fig. 5. Contributions to retention time (RT). Shown are the support vector regression weights for the linear kernel for the 20 features corresponding to peptide amino acid composition; higher values indicate a positive contribution to RT. White circles indicate the individual weights for each of the eight data sets; black circles indicate the means for all data sets.

Figure 5. We observe several expected trends: hydrophobic residues such F and W have higher weights, and hydrophilic residues such as K and R show lower weights. While the SVR weights are largely consistent across chromatography conditions, there are some notable differences, such as the relative weight of K and R. Weights for different length columns (20CM, 40CM, 60CM) are qualitatively similar, but differ in magnitude. The largest weights are associated with the non-specific cleavages, while the smallest are associated with the MudPIT analysis (supplement).

3.3 Improved Peptide Identification

In addition to measuring the R value of predicted RT on the test set, each trained SVR is also tested for its ability to eliminate false positive peptide identifications from its respective chromatography run. We assess confidence of peptide identifications by searching the spectra from each data set against a shuffled version of the appropriate organism's proteome sequence database; any hits to this database above a particular Xcorr threshold are considered an estimate of the number of false positives FP against the *real* database. Then, if P is the number of positive hits to the real database, FDR can be calculated using: $FDR = FP/P$. To reduce FDR, we eliminate identifications with observed RT that deviate from the predicted RT by a constant amount of time, and then measure whether this filtering step improves the number of true positives over a range of FDR thresholds compared to identifications without filtering. An example of the deviation of predicted and observed retention time for matches to the real and random proteomes is shown in Figure 4.

The retention time threshold used to filter identifications is identified in the following manner, as outlined in schematic in Figure 2. In addition to the PSMs from the real yeast proteome, we use PSMs from three shuffled proteomes. The first shuffled proteome is used to select identifications at 10% FDR, as described in Section 2.2. The second shuffled proteome is used to calculate the true positives across a range of FDR values between 0.5% and 10% (in 0.5% increments) for a range of retention time thresholds between 0 and 240 minutes (in 10 minute increments). The retention time threshold that produces the highest number of true positives across the largest number of FDR values is selected as the optimal maximum RT deviation threshold. We then determine the performance of that threshold by calculating true positives across the same range of FDR values using the third shuffled proteome. We repeat this procedure five times, and report an average of the true positives obtained on each of the five iterations. This is compared to an average of true positive performance without any retention time filtration across the same five iterations. The multiple iterations are made necessary by the high variance associated with false positive estimates from shuffled proteomes (Huttlin et al., 2006).

The results, shown in Figure 6, show a consistent decrease in false positive peptide identifications across all the data sets and most FDR thresholds. The dynamically trained SVR effectively adapts to variation in column length (Figure 6, top), digestion conditions (Figure 6, middle) and MudPIT salt step (Figure 6, bottom). The improvement in peptide identification is largest with the non-specific digest elastase. Increases in true positives tend to be largest in the 2% to 3% FDR range, and the Gaussian kernel outperforms the linear kernel in most cases, except for the 60CM column. At a 3% FDR, the largest relative increase in true positive peptide identifications is 52% for the Gaussian kernel on the ELAST data set, from 509 to 772 identifications; the smallest increase is 15% for the 60CM data set, from 1967 to 2270 identifications.

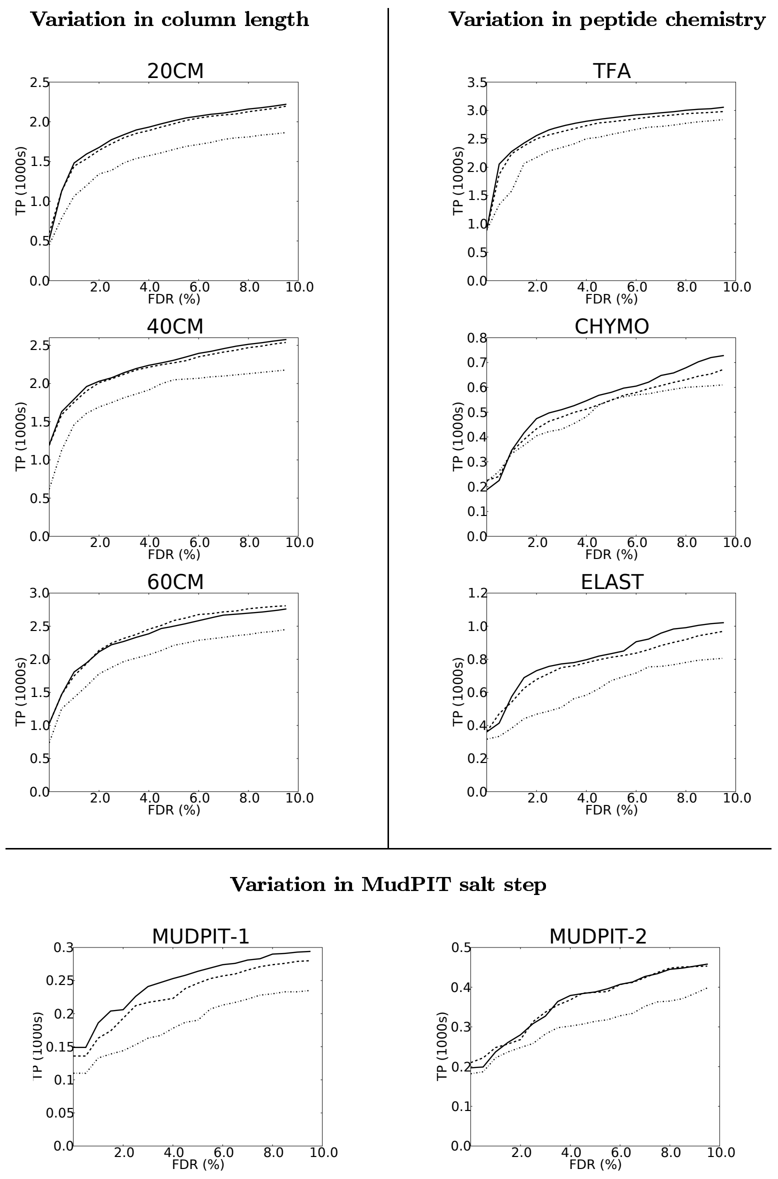


Fig. 6. Improved peptide identification over varying conditions. The dynamically trained SVR is able to cope with chromatographic differences due to variations in column length (left), peptide chemistry (right) and MudPIT salt step (bottom). Spectra from diverse chromatography conditions are searched against the appropriate proteome to yield positive IDs and a shuffled proteome to yield an estimate of false positive IDs. Shown are plots of false discovery rate vs. true positives. The solid curve (Gaussian) and heavy dotted curve (Linear) are for the test data set after filtering with the best classifier found on the training data using the Gaussian and linear kernels, respectively, while the light dotted curve (Unfiltered) is without any filtering.

4 Discussion

We have demonstrated that a dynamically trained support vector regressor is capable of learning to predict peptide RT from a single LC-MS/MS run across a variety of chromatographic conditions, adapting to variation in column length, digestion conditions, peptide chemistry, and MudPIT salt step. Furthermore, using the SVR to filter peptide identifications results in an increase in true positive identifications across almost all false discovery rates and data sets. Of special interest is the improvement in identifications for samples with non-specific enzyme cleavage, a form of analysis typically plagued by false positive identifications.

It is important to note that filtering identifications in this manner is not possible with other methods of predicting RT (such as calculating relative hydrophobicity, as in (Krokhin et al., 2004)), since these methods only predict relative, not absolute retention time. This is highlighted by the difference in scales between the learned and fixed retention time regression in Figure 3. Converting relative to absolute retention time would require methods similar to those outlined here.

Our SVR method does not come without limitations. In particular, data sets of low complexity would probably not produce a diverse enough set of peptides to allow for accurate regression. In addition, poor quality data sets, with few identifications (less than 100 above the 10% FDR), will also fail to yield good regressions. Analysis of such data sets could benefit from improved selection of high-confidence identifications, or from an approach that combines data from the poor quality data set with data from higher quality data sets.

Acknowledgments. The authors would like to thank the NSF for funding through grant T32 HG00035 and the NIH for funding through grant P41 RR11823, as well as anonymous reviewers for providing several useful suggested improvements.

Bibliography

Bączek, T., P. Wiczling, M. Marszałł, Y. V. Heyden, and R. Kaliszan (2005). Prediction of peptide retention at different HPLC conditions from multiple linear regression models. *Journal of Proteome Research 4*, 555–563.

Bihan, T. L., M. D. Robinson, I. I. Stewart, and D. J. Figeys (2004). Definition and characterization of a "trypsinosome" from specific peptide characteristics by nano-HPLC-MS/MS and *in silico* analysis of complex protein mixtures. *Journal of Proteome Research 3*, 1138–1148.

Browne, C. A., H. P. J. Bennett, and S. Solomon (1982). The isolation of peptides by high-performance liquid chromatography using predicted elution positions. *Analytical Biochemistry 124*, 201–208.

Cargile, B. J., J. L. Bundy, T. W. Freeman, and J. J. L. Stephenson (2004). Potential for false positive identifications from large databases through tandem mass spectrometry. *Journal of Proteome Research 3*, 1082–1085.

Eng, J. K., A. L. McCormack, and J. R. Yates, III (1994). An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *Journal of the American Society for Mass Spectrometry 5*, 976–989.

Frenz, J., W. S. Hancock, W. J. Henzel, and C. Horva'th (1990). *HPLC of Biological Macromolecules: Methods and Applications*. Marcel Dekker.

Guo, D., C. T. Mant, A. K. Taneja, J. M. Parker, and R. S. J. Hodges (1987). Effects of ion-pairing reagents on the prediction of peptide retention in reversed-phase high-performance liquid chromatography. *Journal of Chromatography 386*, 205–222.

Hearn, M. T., M. I. Aguilar, C. T. Mant, and R. S. Hodges (1988). High-performance liquid chromatography of amino acids, peptides and proteins. LXXXV. evaluation of the use of hydrophobicity coefficients for the prediction of peptide elution profiles. *Journal of Chromatography 438*, 197–210.

Huttlin, E. L., A. D. Hegeman, A. C. Harms, and M. R. Sussman (2006). Prediction of error associated with false-positive rate determination for peptide identification in large-scale proteomics experiments using a combined reverse and forward peptide sequence database strategy. *Journal of Proteome Research*.

Kawakami, T., K. Tateishi, Y. Yamano, T. Ishikawa, K. Kuroki, and T. Nishimura (2005). Protein identification from product ion spectra of peptides validated by correlation between measured and predicted elution times in liquid chromatography/mass spectrometry. *Proteomics 5*, 856–64.

Krokhin, O. V., R. Craig, V. Spicer, W. Ens, K. G. Standing, R. C. Beavis, and J. A. Wilkins (2004). An improved model for prediction of retention times of tryptic peptides in ion pair reversed-phase hplc. *Molecular & Cellular Proteomics 3*, 908–919.

Mant, C. T. and R. S. Hodges (2006). Context-dependent effects on the hydrophilicity/hydrophobicity of side-chains during reversed-phase high-performance liquid chromatography: Implications for prediction of peptide retention behaviour. *Journal of Chromatography A 1125*, 211–219.

Mant, C. T., N. E. Zhou, and R. S. Hodges (1989). Correlation of protein retention times in reversed-phase chromatography with polypeptide chain length and hydrophobicity. *Journal of Chromatography A 476*, 363–375.

McCormack, A. L., D. M. Schieltz, B. Goode, S. Yang, G. Barnes, D. Drubin, and J. R. Yates, III (1997). Direct analysis and identification of proteins in mixtures by LC-MS/MS and database searching at the low-femtomole level. *Analytical Chemistry 69*(4), 767–776.

Meek, J. L. (1980). Prediction of peptide retention times in high-pressure liquid chromatographic on the basis of amino acid composition. *Proceedings of the National Academy of Sciences of the United States of America 77*, 1632–1636.

Palmblad, M., M. Ramstrom, G. B. Bailey, S. L. McCutchen-Maloney, J. Bergquist, and L. C. Zeller (2004). Protein identification by liquid chromatography-mass spectrometry using retention time prediction. *Journal of Chromatography, B 803*, 131–135.

Palmblad, M., M. Ramstrom, K. E. Markides, P. Hakansson, and J. Bergquist (2002). Prediction of chromatographic retention and protein identification in liquid chromatography/mass spectrometry. *Analytical Chemistry 74*, 5826–5830.

Petritis, K., L. Kangas, B. Yan, M. E. Monroe, E. F. Strittmatter, W. J. Qian, J. N. Adkins, R. J. Moore, Y. Xu, M. S. Lipton, D. G. C. 2nd, and R. D. Smith (2006). Improved peptide elution time prediction for reversed-phase liquid chromatography-MS by incorporating peptide sequence information. *Analytical Chemistry 78*(14), 5026–5039.

Petritis, K., L. J. Kangas, P. L. Ferguson, G. A. Anderson, L. Pasa-Tolic, M. S. Lipton, K. J. Auberry, E. F. Strittmatter, Y. Shen, R. Zhao, and R. D. Smith (2003). Use of artificial neural networks for the accurate prediction of peptide liquid chromatography elution times in proteome analyses. *Analytical Chemistry 75*(5), 1039–1048.

Qian, W. J., T. Liu, M. E. Monroe, E. F. Strittmatter, J. M. Jacobs, L. J. Kangas, K. Petritis, D. G. C. II, and R. D. Smith (2005). Probability-based evaluation of peptide and protein identifications from tandem mass spectrometry and SEQUEST analysis: The human proteome. *Journal of Proteome Research 4*(1), 53–62.

Shinoda, K., M. Sugimoto, N. Yachie, N. Sugiyama, T. Masuda, M. Robert, T. Soga, and M. Tomita (2006). Prediction of liquid chromatographic retention times of peptides generated by protease digestion of the escherichia coli proteome using artificial neural networks. *Journal of Proteome Research 5*, 3312–3317.

Strittmatter, E. F., L. J. Kangas, K. Petritis, H. M. Mottaz, G. A. Anderson, Y. Shen, J. M. Jacobs, D. G. C. 2nd, and R. D. Smith (2004). Application of peptide LC retention time information in a discriminant function for peptide identification by tandem mass spectrometry. *Journal of Proteome Research 3*(4), 760–769.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. New York: Springer.

Washburn, M. P., D. Wolters, and J. R. Yates, III (2001). Large-scale analysis of the yeast proteome by multidimensional protein identification technology. *Nature Biotechnology 19*, 242–247.

Yates, III, J. R. (1998). Mass spectrometry and the age of the proteome. *Analytical Chemistry 33*, 1–19.

A Fast and Accurate Algorithm for the Quantification of Peptides from Mass Spectrometry Data

Ole Schulz-Trieglaff[1,2], Rene Hussong[3], Clemens Gröpl[2], Andreas Hildebrandt[3], and Knut Reinert[2]

[1] Max Planck Research School, Berlin, Germany
trieglaf@inf.fu-berlin.de
[2] Department of Computer Science and Mathematics, Free University Berlin
[3] Center for Bioinformatics, Saarland University

Abstract. Liquid chromatography combined with mass spectrometry (LC-MS) has become the prevalent technology in high-throughput proteomics research. One of the aims of this discipline is to obtain accurate quantitative information about all proteins and peptides in a biological sample. Due to size and complexity of the data generated in these experiments, this problem remains a challenging task requiring sophisticated and efficient computational tools.

We propose an algorithm that can quantify even low abundance peptides from LC-MS data. Our approach is flexible and can be applied to preprocessed and raw instrument data. It is based on a combination of the sweep line paradigm with a novel wavelet function tailored to detect isotopic patterns. We evaluate our technique on several data sets of varying complexity and show that we are able to rapidly quantify peptides with high accuracy in a sound algorithmic framework.

1 Introduction

Quantitative proteomics is increasingly developing into one of the cornerstones of fundamental research in the life sciences and of clinical studies [17,18,21]. In a typical experimental setting, the protein sample is subjected to a proteolytic digestion yielding a mixture of peptides which is inserted into a chromatographic column for a first separation. The peptides elute at different retention times due to their interaction with the stationary phase of the column. The protein digest is thus separated according to a physical property, like the peptide's hydrophobicity in the case of reversed-phase (RP) liquid chromatography. The fractions of the analyte are transferred into a mass spectrometer where they are ionized and separated by their mass/charge ratio.

The resulting data consists of a sequence of MS spectra (*scans*). Each scan gives a snapshot of the peptides eluting from the column during a fixed time interval. It consists of ion counts or *intensities* measured by the mass spectrometer within a certain interval of mass/charge ratios. The scans are acquired at (more or less equally spaced) periodic time intervals, and the collection of scans constitutes what we will call an *LC-MS map*.

Problem statement. Many applications in proteomics, such as mass-spectrometry based diagnostics, rely on an accurate, and – given the size of the data – fast quantification of proteins or peptides contained in a biological sample. Indeed, the quantification problem lies at the very base of a whole proteomics pipeline, where all subsequent steps depend on the quality of the data generated in the beginning. In this work, we report on a fast and reliable approach to peptide quantification tailored for very large LC-MS maps.

To accurately estimate the abundance of peptides in a biological sample, we need to collect all data points that are caused by a charge variant of a peptide, that is by all ions with identical sequence and charge. We will refer to this problem as the *peptide quantification problem.* The detection of peptidic features in LC-MS spectra can be considerably improved by exploiting prior knowledge about the data produced during the experimental process: data points in an LC-MS map belonging to the same peptide are locally highly correlated in the mass over charge dimension as well as in the retention time domain. The atoms contained in a peptide occur in different isotopic variants, and the distribution of the naturally occurring isotopes gives rise to a characteristic *isotopic pattern* of adjacent peaks in the mass spectrum. Similarly, each peptide elutes over a certain interval of time from the column and can be observed in several consecutive scans. The *elution profile* ideally follows a normal distribution around its centroid, but fronting and tailing effects are frequently observed in practice [3].

In addition to these local relationships, non-local correlations can also be observed: different charge states of ionized peptides show up at distant m/z values, and different peptides originating from the same protein have distant RT (and m/z) values. For the remainder of this work, however, we will restrict ourselves to local correlation effects.

Previous work. Several computational approaches to peptide quantification have been developed recently. Some of them are embedded into a software framework comprising other processing steps such as identification of proteins or alignment of LC-MS maps as well. A recent review of these software tools is given in [22], while [16] gives a more general overview of the computational problems in proteomics data analysis.

An algorithm for peptide quantification needs to address the following problems: in a first step, prominent data points (or *seeds*) in the LC-MS map need to be found. These seeds are data points that are very likely to be in the region (in m/z and retention time dimension) of data that can be attributed to a peptide charge variant. Second, we want to *extend* these seeding points to *regions* of interest in the LC-MS map. Due to posttranslational modifications, isotopic variants and different charge states of the same peptide, it is not feasible to restrict the search to single points in the spectrum. Rather, we always need to consider clusters of data points centered around the seeds. In the literature, several approaches have been proposed to identify such regions based on the intensity of the data points [1,6,15,14,27] or using image segmentation techniques [12,26].

The image-based approach usually starts by *resampling* the data to obtain a gray-scale image from the LC-MS map [12,26]. But the dimensions of LC-MS separation have quite different characteristics and require different handling. and a resampling diminishes the resolution of the data and will almost certainly lead to a loss of information.

If the set of candidate regions is chosen based on the intensity of single data points or local maxima in the LC-MS map, it is likely to contain many false positives, i.e. groups of data points caused by noise or contaminants of the sample and not by peptides. In addition, these approaches are hampered by the low signal-to-noise ratio of peaks at the beginning and end of the peptide elution period. Sophisticated methods are required to estimate the background noise in the spectra and to exclude outlier data points from the isotopic pattern before a peptide abundance can be estimated [14,27]. In some cases, information from tandem spectra and database search is taken into consideration to increase the confidence in detected seeds [5,14]. But due to the high error rates of current peptide identification algorithms [8], this approach has its own disadvantages.

After the extension one can employ an additional *refinement step* during which a theoretical peptide model is adjusted to the selected data points [1,6,12,15]. The quality of this fit is taken as a measure of confidence that this region is indeed caused by a peptide. Regions with poor correspondence to the theoretical model are discarded. By summing the intensities of all data points in the regions identified, we can obtain an estimate of the peptide that can be used for a relative quantification. This sequence of finding seeds, extension and refinement is a general concept, and the majority of algorithms follow these steps e.g. [1,6,12,15].

A typical proteomic sample consists of several thousand peptides and clinical studies typically consist of hundreds of samples. It is therefore desirable to quantify all peptides in a sample as quickly as possible. The refinement step in particular takes considerable time and sometimes even requires a manual validation of the peptide candidates. An efficient algorithm that is suitable for real-word applications should thus aim for a low number of seeds. However, if seeds or regions in the map are chosen based on their intensity alone, we will obtain a large number of seeds, many of which will simply be caused by chemical noise or contaminants of the sample.

Our contribution. In this work, we propose a new seeding stage based on the sweep line paradigm [2], that allows to efficiently apply sophisticated isotopic pattern detection on large data sets. To our best knowledge, the presented algorithm is the first technique that fully exploits the two-dimensional information contained in LC-MS maps with an efficiency that scales to real-world applications. In addition, the presented method does neither rely on lossy signal pre-processing steps (baseline subtraction, noise reduction) nor on potentially disturbing resampling of the often unequally spaced data sets. Noise and baseline removal are implicitly and reversibly included through the use of a novel *isotope wavelet*, and all steps of the algorithm have been specifically designed to work even on unequally spaced spectra.

We show that our algorithm results in the selection of fewer unnecessary seeds while achieving the same accuracy as the significantly slower high-precision approach presented in [6]. On the other hand, the presented approach does not only reduce the number of false positives that have to be filtered out in the later stages of the algorithm, but also has the potential to detect true isotopic patterns that would most probably be ignored in any merely intensity-based seeding approach: as we will demonstrate, the isotope wavelet transform often identifies even hardly noticeable patterns that nearly vanish in the noise. This is particularly important at the tails of the elution profile of a given peptide where its signal intensity is low. In addition, the isotope wavelet allows for a rapid, yet very accurate classification of any isotopic pattern candidate into one of several possible charge states. Using this information, the subsequent model fitting stage of our algorithm becomes (a) significantly faster since fewer charge states have to be tested and (b) simpler and safer since a sensible initial solution for the fitting process is already provided.

The presented algorithm has been developed using OpenMS [9], a software library for shotgun proteomics, and is available to the community under the GNU Lesser General Public License.

2 Methods

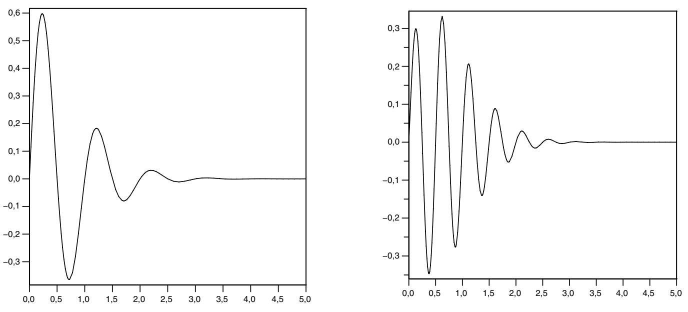
We follow the proposed sequence of seeding, region finding (extension) and refinement stage. But in contrast to previous approaches we employ a model-driven technique to detect isotopic patterns in multiple scans which will be introduced in the next two sections. Inspired by the sweep line paradigm, we collect supporting information from neighboring scans to increase our confidence in the detection. We conservatively extend this initial guess for the pattern region in the extension phase. Finally, in our refinement step, we fit a theoretical model to this region. This model is two-dimensional and consists of an average isotopic distribution of a peptide with a given mass (for the mass/charge domain) and an exponentially-modified Gaussian (for the time domain). The quality of this fit reflects our confidence in this peptide candidate and regions of low quality are discarded.

Modeling isotope distributions of peptides. The chemical elements of peptides naturally occur in different isotopic variants. The mass differences between light and heavy isotopes can be approximated by multiples of $\delta_{\mathrm{av}} \sim 1.00235$ Da [7]; for our data of intermediate resolution even $\delta_{\mathrm{av}} = 1$ can be used. Thus, we can compute the theoretical spectrum of a peptide from its empirical formula. There exist several algorithms for this task [10,24,28]. Here we use a straight-forward algorithm, the only noteworthy detail being that e.g. the isotopic distributions of $C, C_2, C_4, C_8, \ldots$ are computed by 'squaring'. The abundances of heavy isotopes are approximated using an average amino acid, sometimes called 'averagine' which represents the amino acid composition observed in large protein databases. The averagine [25] has the molecular formula $C_{4.9384}H_{7.7583}N_{1.3577}O_{1.4773}S_{0.0417}$ and a total mass of 111.1254 Da. Fractional numbers of atoms are rounded to the next integer. We used protein sequences in a recent Swiss-Prot release to estimate the averagine composition and therefore our formula differs slightly from previous works [25].

Wavelets to detect regions of interest. Detecting isotopic patterns in a typical mass spectrum is complicated by the large degree of disturbing influences such as a signal baseline as well as electrical and chemical noise. Many efforts have been made to develop techniques for the automated or manually assisted reduction of these parasitics, but all of those ultimately lead to a certain distortion of the 'real' signal. In addition, it is often unclear if all significant noise contributions have been successfully filtered out or if some remainder has still leaked into the signal. An alternative route to identify features of interest *without explicit* removal of disturbing influences relies on signal theoretic analysis of the mass spectrometric scan, typically based on wavelets. The wavelet transform naturally generalizes the well-known Fourier transform in that a signal is *locally* split into components of different frequencies. This spectral decomposition can be very useful for feature detection algorithms, since, in reality, baseline, electrical noise, and the 'real' signal usually live on different frequency ranges. By computing a wavelet transformed version of the signal that corresponds to the 'correct' frequency range, disturbing components are automatically suppressed, greatly simplifying the analysis.

Wavelet techniques involve an additional degree of freedom in the choice of the analytical form of the so-called 'mother wavelet' the transform is based on. Intuitively, we replace the mass signal by a measure of how well it locally correlates with the shape of the wavelet. In [11], we have demonstrated that the established Marr wavelet is well suited for the detection of individual peaks and thus is a sensible foundation for a high-accuracy peak picking application. In particular, if the scale of the wavelet transform (corresponding to the frequency range considered) is chosen carefully, the Marr wavelet transform of a given peak is to a large degree independent of neighboring peaks, even if they strongly overlap. But while this property allows for astonishing accuracy in the separation of overlapping peaks, it is clearly counterproductive for the detection of isotopic patterns, where we explicitly want to base our analysis on the behaviour of the spectrum in the neighbourhood of a given peak. Therefore, we designed tailored 'isotope wavelets', which are based on the mass distributions in typical isotopic patterns as described above.

There exist other applications of the wavelet transform to mass spectrometry data [4,23]. For example, [23] use wavelets to denoise the spectra prior to database searching for the inference of the amino acid sequence. [4] employ wavelet analysis to detect single peaks in low resolution spectra but not clearly resolved isotopic pattern for quantification as we do. However, our isotopic mother wavelet is clearly novel and has not been used before. Furthermore, we are not aware of any other work in which wavelet analysis is used for the quantification task. make Since the shape of the isotopic patterns depends on the mass as well as on the charge of the considered peptide, several wavelet functions must be used. While the wavelet can adapt automatically to the considered mass during



(a) m/z ≈ 500 Th, z=1 (b) m/z ≈ 1000 Th, z=2

Fig. 1. Isotopic wavelets

the computation of the transform, each charge state requires its 'own' mother wavelet. Therefore, if we assume peptides to be at most 4-fold charged, e.g., we have to compute four transformed versions of each scan. The actual wavelet design process is technically cumbersome and out of scope of this work. Instead, we show two exemplary isotope wavelets in Fig. 1.

With the isotope wavelet, regions of interest can now be detected by first searching for local maxima in the transform. A 'real' isotopic pattern will lead to a chirp-like signal in the wavelet transform, since each of its mass peaks will lead to a resonance with the wavelet, and we make extensive use of this special shape to improve the specificity of our approach. Shape and regularity of the wavelet transform of a candidate pattern are used to derive a score, denoting how well the candidate fits the current wavelet type. Repeating the transform with an isotope wavelet of each charge state yields a set of charge dependent scores, leading to a powerful and robust charge prediction method.

Due to the design of the wavelet, we only need to compute one scale of the wavelet transform, i.e. compute the correlation integral of the isotope wavelet with the mass signal. While at first glance this seems to require $\mathcal{O}(n^2)$ operations, the real runtime is actually much smaller: the wavelet has finite and typically small support that is independent of the length of the mass signal, so that the transform can be performed in linear time.

Region extension using the sweep line paradigm. The sweep line algorithm is a general paradigm from the field of computational geometry that has, e.g., been used to detect intersections of line segments in an efficient manner [2]. The algorithm can be illustrated by an imaginary line sliding over the segments. It keeps track of segments it encounters using a dynamic data structure. The beginning or end of a line segment triggers an update of the datastructure and a check for intersections is performed if the algorithm meets the endpoint of a segment.

We follow the general sweep line paradigm, but compared to the line segment intersection algorithm we are not searching for intersecting lines but adjacent and possibly overlapping isotopic patterns. Hence, we sweep across the LC-MS map scan by scan and use our isotope wavelet to detect the starting positions of isotopic patterns in each spectrum. That is, we apply the transform to each scan and sweep across the time domain. A significant signal in the wavelet transform triggers an event and we check if we detected a pattern in the previous scan at the same mass with a small tolerance. The predicted monoisotopic masses of each pattern in each scan are stored in a tree-based data structure.

This approach allows to quickly discard potential isotopic peaks that are not supported by isotopic peaks in adjacent scans and to significantly reduce the number of candidate regions for the next refinement step. Furthermore, our wavelet function gives us an initial guess for the charge state of the peptide, which further reduces the number of peptide models that need to be tested. Since we scan through the LC-MS map in a linear manner, we can work efficiently on secondary memory data structures storing the peak data, allowing to apply our algorithm to very large data sets.

Refinement stage by model fitting. As stated above, we fit a two-dimensional model to each potential peptide signal identified in the two previous steps of the algorithm. The part of the model which is applied to the $\frac{m}{z}$ domain relies on the average isotopic distribution for the given mass region. In addition, we model the imprecision of the mass analyzer by a normal distribution with variance σ^2. The resulting model for the isotopic distribution of a peptide is

$$\phi(m) = \frac{A}{\sqrt{2\pi\sigma^2}} \sum_{i=0}^{i_{\max}} a_i(m_0) e^{-(m-m_0-i\delta_{\text{av}})^2/(2\sigma^2)} ,$$

where m_0 = monoisotopic mass, $a_i(m_0)$ = relative abundance of i-th isotopic peak of a peptide with monoisotopic mass m_0, $i_{\max}$ = last isotopic peak considered, and A = area under curve.

The elution profile of the peptide is modeled by an exponentially modified Gaussian (EMG). For computational efficiency, we use a simplified version [3]. Its density function is given by

$$\text{emg}(x) = \frac{hw}{s}\sqrt{2\pi}\frac{\exp\left(\frac{w^2}{2s^2} - \frac{x-z}{s}\right)}{1 + \exp\left(-\frac{2.4055}{\sqrt{2}}\left[\frac{x-z}{w} - \frac{w}{s}\right]\right)}$$

where parameter z controls the centroid, w the width, and h the height of the elution profile. The parameter s represents the skewness of the distribution. An earlier version of our algorithm modeled the elution profiles by a normal distribution. Using the EMG makes our model much more robust in the presence of heading or tailing effects. Since the partial derivatives of the EMG can be computed analytically, we use the Levenberg-Marquardt algorithm [13,19] to minimize the least squared distance of the model to the selected region of the LC-MS map. The quality of the model fit is measured using the squared correlation between data and model. If the correlation is too low, we discard the corresponding peptide region. The monoisotopic mass is estimated from the theoretical isotope distribution and the coordinate in retention time is taken as the centroid of the fitted EMG.

Greedy Separation of Overlapping Isotopic Patterns. In high-resolution spectra of complex samples, overlapping isotopic patterns might pose a severe problem for an accurate quantification. Here, we propose a greedy approach to this problem. If the wavelet transform detects overlapping isotopic patterns, these are independently assembled during the sweepline stage and passed to the model fit. Peaks that have a good correlation with the theoretical model are removed from the spectrum. If the wavelet indicates that there might be another isotopic pattern in the same region, a new model is fitted using the remaining peaks in this area until no more isotopic pattern remain.

This approach is straightforward and similar to other methods already published e.g. in [7]. More sophisticated approaches are imaginable and easy to integrate into our framework. However, we found that his greedy approach works well in practice.

3 Results

We evaluate our approach on two different data sets. The first one was obtained from a peptide standard mix consisting of nine peptides. Here, we systematically introduced noise into the data set to evaluate the ability of our algorithm to correctly detect and quantify peptides in noisy signals.

The second data set consists of human blood serum samples from a myoglobin quantification study [20]. We use this data to show that we are able to perform very precise quantification of peptides in complex samples. Furthermore, we demonstrate that we are considerably faster than an approach which is of high accuracy but slow since it selects the seeding regions in the LC-MS map based on their intensity only [6] and therefore performs many time-consuming refinement steps using a theoretical peptide model.

Stability analysis. As a first step, we show that our model-driven quantification approach (*SweepWavelet*) is able to produce reliable results in the presence of noise. We systematically introduce noise in an LC-MS map of standard peptides. We are aware that performance evaluation on simulated datasets has its caveats. However, by doing so we can measure performance on data with specific characteristics.

The data set chosen – an artificial mix of 9 peptides that is described in detail in [11] – is of very high quality with unusually low noise level. We thus consider manual annotation of the unperturbed spectrum as the gold standard against which we test our technique. In order to relate the noise level to the intensity of the isotopic pattern of interest, we add uniformly distributed noise with zero mean and an amplitude of $10\%, 25\%, 50\%$, and 75% of the intensity of the monoisotopic peak, respectively. If this resulted in negative values, these were replaced by zero. While this uniformly distributed noise does not provide

Table 1. Feature detection on a spectrum with varying levels of uniform noise. The percentages denote the amplitude of the noise in terms of the intensity of the monoisotopic peak of the pattern. The '#scans' - row gives the number of scans (retention times) in which the isotopic pattern was found as compared to the number found by manual annotation in the unperturbed spectrum. The 'charge' - row indicates whether the charge state was correctly assigned.

	Oxytocine, 1007.5 Th, charge 1					Substance P, 674.5 Th, charge 2				
	0%	10%	25%	50%	75%	0%	10%	25%	50%	75%
#scans	11/11	11/11	10/11	10/11	0/11	16/20	16/20	13/20	12/20	13/20
charge	✓	✓	✓	✓	n/a	✓	✓	✓	✓	-

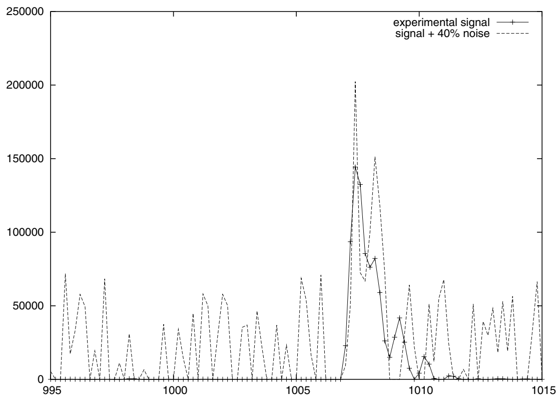


Fig. 2. Experimental signal with artificial noise

a realistic model of all noise effects in a real spectrum, this experiment should still give us with an idea of the stability of our method if applied to noisy data.

The combination of the isotopic wavelet for seeding with the sweep line approach leads to a robust feature detection (see Table 1). Even for extremely low signal-to-noise ratios, the pattern is usually detected in a sufficient number of scans to allow for accurate seeding, and correct charge prediction. The results are exemplarily discussed for two of the peptides, with charges of 1 and 2, respectively. Performance on the other peptides is very similar. An example where the drastic amount of noise leads to severe distortion of the signal without hurting our seeding and charge prediction can be found in Fig. 2.

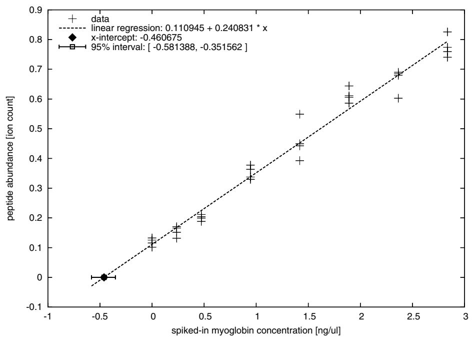
It is of course not clear whether our approach to introduce noise into the data comes close to real noise in mass spectra. Other possibilities would be to introduce additional isotope distributions to simulate chemical noise or to model instrument noise by adding single, poisson-distributed peaks. We decided to distort existing isotope peaks since we wanted to test the ability of our algorithm to detect deformed isotopic patterns. Adding further, artificial patterns would merely increase the running time of our algorithm and not give any information about its performance in the presence of noise. Single peaks caused by instrument noise would be simply filtered out during the wavelet transform as long as they don't resemble isotopic pattern by chance.

Accuracy and speed of quantification. We now apply our algorithm to a realistic task, the quantification of myoglobin from human blood serum. Myoglobin is a protein of low-molecular weight which appears quickly in blood after tissue injuries and is considered as an important biomarker for myocardial necrosis. A fast but accurate quantification of myoglobin in human blood samples is therefore important. Sample preparation and details of the absolute quantification process have already been described elsewhere [20]. In short, the myoglobin was separated from the highly abundant serum proteins by anion-exchange chromatography. The myoglobin fraction was trypsinized and the resulting peptides were analyzed by reversed-phase liquid chromatography coupled to an ion-trap mass spectrometer. To perform an absolute quantification, known amounts of human myoglobin were added to aliquots of the sample. Absolute quantification was then performed by determining the x-intercept of a linear regression using the ratio of the eleventh tryptic myoglobin peptide and an internal standard consisting of the tenth tryptic peptide of horse myoglobin. The LC-MS maps were recorded using a quadrupole ion trap mass spectrometer (Bruker Daltonics, Germany) coupled to a reversed-phase HPLC column.

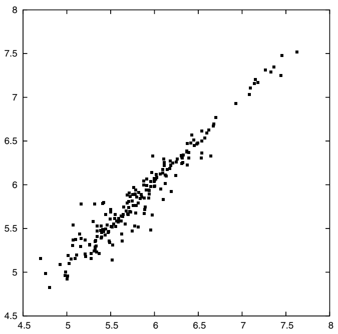
We compare our approach to an algorithm [6] which was developed for this myoglobin quantification study. This method (from now on referred to as *CompLife05*) carefully collects regions of data points with high ion count and fits a theoretical isotopic model to the data. This refinement step is similar to ours but uses a simpler model. Regions having a sufficiently good correlation with the theoretical model are considered as true peptides and quantification is performed by summing the ion counts of all raw data points in the chosen region. Outlier points are excluded before quantification if their predicted intensity under the model is below a given threshold. We consider this algorithm as representative for the common approach that detects potential peptides based on the intensity of single, but high, peaks in the LC-MS map, collects a cluster of raw data points and then refines this selection by fitting a theoretical peptide model to the data.

Table 2 compares our algorithm to *CompLife05* and to a manual quantification by a human expert. The manual quantification was performed using the Bruker Data Analysis software and Microsoft Excel. Peak areas were estimated from extracted ion chromatograms smoothed by a Gaussian filter. The measurements were performed on two independently acquired data sets. For the computational quantification, we used the OpenMS tools [9] to perform an alignment of the LC-MS maps and to match corresponding peptides across the LC-MS maps. No smoothing or other preprocessing steps were performed.

Manual and both automated measurements estimated the true concentration of myoglobin with high precision. The regression results of *SweepWavelet* are given in Fig. 3(a). Fig. 3(b) shows a good reproducibility of the peptide abundances in the replicate measurements of the myoglobin study. Note that we do not claim to perform a significantly better quantification than [6]. Marginal differences like the ones presented above might be caused by favorable parameter



(a) Regression results using *SweepWavelet*



(b) Reproducibility of peptide abundancies

Fig. 3. Accuracy and reproducibilty of quantification Figure (a) shows the additive measurement as it was computed using our sweep line based algorithm. The regression was performed using the ratio of the eleventh tryptic myoglobin peptide and an internal standard consisting of the tenth tryptic peptide of horse myoglobin. (b) gives the log-transformed peptide abundances estimated from two replicate myoglobin samples.

Table 2. Results of absolute myoglobin quantification in human plasma. *SweepWavelet* refers to our algorithm, *CompLife05* is the approach with intensity-based seeding [6]. Column *Manual* gives the results obtained by a human expert.

	SweepWavelet	CompLife05	Manual
Myoglobin data set 1	**True concentration** [ng/μl] *0.463*		
Computed concentration [ng/μl]	0.460	0.474	0.382
95% confidence interval [ng/μl]	[0.351;0.581]	[0.408;0.545]	[0.315;0.454]
Relative deviation from true value [%]	−0.65	+2.46	−17.42
Myoglobin data set 2	**True concentration** [ng/μl] *0.456*		
Computed concentration [ng/μl]	0.432	0.502	0.420
95% confidence interval [ng/μl]	[0.309;0.572]	[0.381;0.640]	[0.305;0.535]
Relative deviation from true value [%]	−5.55	+10.10	−7.89

settings. But we do claim that we are able to perform a quantification of equal accuracy at a much higher speed compared to a high-accuracy algorithm that was developed and tailored for the myoglobin quantification task. Our approach is therefore more suitable for large-scale studies and high-throughput experiments.

The run time of our algorithm on a set of myoglobin maps was measured on a 3.2 GHz Intel Xeon CPU with 3 GB memory running Debian Linux (Table 3). We used the same parameter settings as for the myoglobin quantification described above. Peptide models up to charge 4 were fitted, i.e. we discarded the charge prediction of the wavelet transform to obtain a fairer comparison. Algorithm *SweepWavelet* discarded all isotopic pattern that occurred in less than three consecutive scans and *CompLife05* considered all signals up to an ion count of 4000 as potential seeds. Thus very week peptide signals were discarded. Each data set consisted of about 1830 scans measured in full scan mode ($\frac{m}{z}$ range 500 - 1500 Th).

The new sweep line algorithm is faster on all data sets while *CompLife05* detects about 6 times more seeds. Since the subsequent refinement step takes considerable time, *CompLife05* is significantly slower. Nevertheless this refinement step helps to discard a large number of seeds in both algorithms. Note that the number of peptides found by *CompLife05* is always higher than the number of peptides found by *SweepWavelet*. This has two reasons: the refinement step in both algorithms is imperfect and a high number of seeds will necessarily result in a higher number of false positives. Manual inspection of the results confirmed this. Second, our wavelet apparently fails to detect poorly resolved regions that show no isotopic pattern. But since mass spectrometers are evolving rapidly, we anticipate that high-resolution instruments will become standard very soon and this disadvantage will diminish.

Note that the seeds in Table 3 correspond to putative peptide signals identified either by a combination of isotopic wavelet and sweepline algorithm or based on their ion count by algorithm *CompLife05*. Column *peptides* gives the number of signals that were classified as peptide charge variants for each algorithm. *SweepWavelet* detects 200 peptides on average whereas *CompLife05* finds 500. A theoretical digest of Human Myoglobin yields only 19 peptides. The fact that both algorithms claim to find a much larger number of peptides than one would expect can be explained by several facts. The Myoglobin was extracted from human plasma. Some other peptides or contaminants will inevitably remain in the sample even after depletion and filtering. Some peptides occur in different charge states and will be independently reported by each algorithm. Finally, some signals will be false positives.

In this particular application, the high number of putative peptides was not a problem since we performed the quantification using only two Myoglobin peptides of known mass. We align the maps and filtered for the masses and expected retention times of these two peptides. In more complex applications, such as a difference detection in complex samples [16], normalisation and statistical testing for differential expression are likely to eliminate these false positive signals.

Increasing the correlation threshold in the least-squares fitting stage of both algorithms might decrease the number of false positives but also the probability of missing important signals in large-scale applications. Note that this would not

Table 3. Running time, number of seeds and number of peptides after refinement on three exemplary data sets of the myoglobin study

	SweepWavelet			CompLife05		
Data set	Time [min]	# Seeds	# Peptides	Time [min]	# Seeds	# Peptides
Myoglobin 01	4.11	511	261	21.15	2652	521
Myoglobin 02	4.35	561	301	25.26	3537	557
Myoglobin 03	4.17	538	297	20.41	2549	492

influence the running time since it is mainly determined by the number of seeds on which the model fitting is performed.

4 Conclusions and Outlook

In this work, we have presented a novel algorithm for the peptide quantification problem. It combines the sweep line paradigm and a tailored wavelet function to scan for isotopic patterns in mass spectra. We have shown that this approach is able to accurately detect monoisotopic masses and charge states of peptides even in the presence of noise and that we can perform quantifications in complex data sets with high accuracy (less than 0.65% and 5.55% relative error) in an efficient manner.

Basing the feature detection process on an integral transform has a number of important advantages from a signal theoretic point of view, but might also be seen as a possible shortcoming of our algorithm: if the resolution of the data falls below a certain critical threshold, the approach is no longer practical. In our experience, however, our technique works well on real-world data. In addition, the resolution of available mass spectrometric data is ultimately going to increase, while for poorly resolved data, slower techniques like the one presented in [6] can typically be applied since the resulting maps are considerably smaller than the ones considered here.

We focused on a label-free setting in which detected peptides have to be mapped across data sets using additional computational tools. However, our approach is flexible and can also be applied to experiments in which isotope or mass tag labeling of peptides is applied. In this scenario, we have to search for pairs of peptides in the same map – a computational problem which can be solved by a range query for each detected peptide.

The application of our method presented here is the quantification of peptides from LC-MS samples. But one can easily imagine other likewise important applications such as the accurate generation of mass and time tags and peptide mass mapping.

To summarize, instead of a tiresome and error-prone manual inspection, efficient algorithms as the one presented here allow high-throughput studies and will emerge as a useful computational tool in quantitative proteomics.

Acknowledgements

We would like to thank Prof. Christian Huber (Saarland University, Saarbrücken) for providing the peptide standard mix data and the myoglobin data sets. Bettina Mayr (Saarland University, Saarbrücken) performed the manual myoglobin quantification. We are also indebted to Jens Joachim and Marcel Grunert who helped to implement an earlier version of the peptide quantification algorithm. Ole Schulz-Trieglaff was supported by the Max Planck Research School for Computational Biology and Scientific Computing Berlin. Rene Hussong and Andreas

Hildebrandt were supported by DFG grants BIZ4:1-4. Clemens Gröpl and Knut Reinert acknowledge funding by the Berlin Center for Genome Based Bioinformatics (BCB). We also thank the anonymous reviewers who helped to improve this document.

References

1. M. Bellew, M. Coram, M. Fitzgibbon, M. Igra, T. Randolph, P. Wang, D. May, J. Eng, R. Fang, C.-W. Lin, J. Chen, D. Goodlett, J. Whiteaker, A. Paulovich, and M. McIntosh. A suite of algorithms for the comprehensive analysis of complex protein mixtures using high-resolution LC-MS. *Bioinformatics*, 22(15):1902–1909, 2006.
2. J.L. Bentley and T. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Trans. Comput*, C28:643–647, 1979.
3. Valerio B. Di Marco and G. Giorgio Bombi. Mathematical functions for the representation of chromatographic peaks. *Journal of Chromatography A*, 931:1–30, 2001.
4. Pan Du, Warren A. Kibbe, and Simon M. Lin. Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching. *Bioinformatics*, 22:2059 – 2065, 2006.
5. B. Fischer, J. Grossmann, V. Roth, and W. Gruissem. Semi-supervised LC/MS alignment for differential proteomics. *Bioinformatics*, 22:e132–e140, 2006.
6. C. Gröpl, E. Lange, K. Reinert, O. Kohlbacher, M. Sturm, C.G. Huber, B. Mayr, and C. Klein. Algorithms for the automated absolute quantification of diagnostic markersin complex proteomics samples. In Michael Berthold, editor, *Procceedings of CompLife 2005*, Lecture Notes in Bioinformatics, pages 151–163. Springer, Heidelberg, 2005.
7. David M. Horn, Roman A. Zubarev, and Fred W. McLafferty. Automated reduction and interpretation of high resolution electrospray mass spectra of large molecules. *Journal of the American Society for Mass Spectrometry*, 11(4):320–332, April 2000. Seminal paper on quantification of peptides of high-resolution spectra.
8. A. Keller, A. Nesvizhskii, E. Kolker, and R. Aebersold. Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search. *Anal. Chem.*, 74:5383–5392, 2002.
9. O. Kohlbacher, K. Reinert, C. Gröpl, E. Lange, N. Pfeifer, O. Schulz-Trieglaff, and M. Sturm. TOPP - The OpenMS proteomics pipeline. In *Proceedings of the 5th European Conference on Computational Biology (accepted)*, 2006.
10. H. Kubinyi. Calculation of isotope distributions in mass spectrometry. a trivial solution for a non-trivial problem. *Anal. Chim. Acta*, 247:107–119, 1991.
11. E. Lange, C. Gröpl, K. Reinert, O. Kohlbacher, and A. Hildebrandt. High accuracy peak-picking of proteomics data using wavelet techniques. In *Proceedings of the Pacific Symposium on Biocomputing (PSB) 2006*, pages 243–254, 2006.
12. K.C. Leptos, D.A. Sarracino, J.D. Jaffe, B. Krastins, and G.M. Church. MapQuant: Open-Source software for large-scale protein quantification. *Proteomics*, 6(6):1770–1782, 2006.
13. K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.
14. H. Li, X.-J.and Zhang, J.R. Ranish, and R. Aebersold. Automated statistical analysis of protein abundance ratios from data generated by stable isotope dilution and tandem mass spectrometry. *Anal. Chem.*, 75:6648–6657, 2003.

15. X.-J. Li, E.C. Yi, C.J. Kemp, H. Zhang, and R. Aebersold. A software suite for the generation and comparison of peptide arrays from sets of data collected by liquid chromatography-mass spectrometry. *Mol. Cell Proteomics*, 4(9):1328–1340, 2005.
16. J. Listgarten and A. Emili. Statistical and computational methods for comparative proteomic profiling using liquid chromatography-tandem mass spectrometry. *Mol Cell Proteomics*, 4(4):419–434, 2005.
17. M.J. MacCoss and D.E. Matthews. Quantitative MS for proteomics: Teaching a new dog old tricks. *Anal. Chem.*, 77(15):294A–302A., 2005.
18. M. Mann and R. Aebersold. Mass spectrometry-based proteomics. *Nature 422*, 422:198 – 207, 2003.
19. D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 11:431–441, 1963.
20. B.M. Mayr, O. Kohlbacher, K. Reinert, M. Sturm, C. Gröpl, E. Lange, C. Klein, and C. Huber. Absolute myoglobin quantitation in serum by combining two-dimensional liquid chromatography-electrospray ionization mass spectrometry and novel data analysis algorithms. *J. Proteome Res.*, 5:414–421, 2006.
21. S.-E. Ong and M. Mann. Mass spectrometry-based proteomics turns quantitative. *Nature Chem. Biology*, 1(5):252–262, 2005.
22. P.M. Palagi, P. Hernandez, D. Walther, and R.D. Appel. Proteome informatics I: Bioinformatics tools for processing experimental data. *Proteomics*, ISSN 1615-9861, http://dx.doi.org/10.1002/pmic.200600273 (epub ahead of print), 2006.
23. Tomas Rejtar, Hsuan shen Chen, Victor Andreev, Eugene Moskovets, and Barry L. Karger. Increased identification of peptides by enhanced data processing of high-resolution maldi tof/tof mass spectra prior to database searching. *Analytical Chemistry*, 76:6017 –6028, 2004.
24. A.L. Rockwood, S.L. Van Orden, and R.D. Smith. Rapid calculation of isotope distributions. *Analytical Chemistry*, 67:2699?2704, 1995.
25. M.W. Senko, S.C. Beu, and F.W. McLafferty. Determination of monoisotopic masses and ion populations for large biomolecules from resolved isotopic distributions. *Journal of the American Society for Mass Spectrometry*, 6(4):229–233, 1995.
26. C.A. Smith, E.J. Want, G. O'Maille, R. Abagyan, and G. Siuzdak. XCMS: processing mass spectrometry data for metabolite profiling using nonlinear peak alignment, matching, and identification. *Analytical Chemistry*, 78(3):779–787, 2006.
27. G. Wang, W.W. Wu, T. Pisitkun, J.D. Hoffert, M.A. Knepper, and R.-F. Shen. Automated quantification tool for high-throughput proteomics using stable isotope labeling and LC-MS. *Analytical Chemistry*, 78(16):5752–5761, 2006.
28. J.A. Yergey. A general approach to calculating isotopic distributions for mass spectrometry. *Int. J. Mass Spectrom. Ion Phys.*, 52:337?349, 1987.

Association Mapping of Complex Diseases with Ancestral Recombination Graphs: Models and Efficient Algorithms

Yufeng Wu

Department of Computer Science
University of California, Davis
Davis, CA 95616, U.S.A.
wuyu@cs.ucdavis.edu

Abstract. Association, or LD (linkage disequilibrium), mapping is an intensely-studied approach to gene mapping (genome-wide or in candidate regions) that is widely hoped to be able to efficiently locate genes influencing both complex and Mendelian traits. The logic underlying association mapping implies that the best possible mapping results would be obtained if the genealogical history of the sampled individuals were explicitly known. Such a history would be in the form of an "ancestral recombination graph (ARG)". But despite the conceptual importance of genealogical histories to association mapping, few practical association mapping methods have explicitly used derived genealogical aspects of ARGs. Two notable exceptions are [35] and [23].

In this paper we develop an association mapping method that explicitly constructs and samples minARGs (ARGs that minimize the number of recombinations). We develop an ARG sampling method that provably samples minARGs *uniformly* at random, and that is practical for moderate sized datasets. We also develop a different, faster, ARG sampling method that still samples from a well-defined subspace of ARGs, and that is practical for larger sized datasets. We present novel efficient algorithms on extensions of the "phenotype likelihood" problem, a key step in the method in [35]. We also prove that computing the phenotype likelihood for a different natural extension of the penetrance model in [35] is NP-hard, answering a question unresolved in that paper. Finally, we put all of these results into practice, and examine how well the implemented methods perform, compared to the results in [35]. The empirical results show great speed ups, and definite but sometimes small, improvements in mapping accuracy. Speed is particularly important in doing genome-wide scans for causative mutations.

1 Introduction

One type of genetic disease, or more generally any phenotype (observable trait), is caused by a single mutation at a single locus, and that mutation has "high penetrance", meaning that the probability of the trait given the mutation is very large. Sometimes this type of disease is called "Mendelian". In contrast, "complex traits" can originate independently at many different loci; or a combination of mutations is required to create a phenotype; or different combinations of mutations can create the trait; or the penetrance of the mutations may be low. Understandably, although many Mendelian traits have been mapped quite successfully, mapping the genetic origin of complex traits remains a very challenging problem.

Association, or LD (linkage disequilibrium), mapping is a current, intensely-studied approach to gene mapping (genome-wide or in candidate regions) that is widely hoped to be able to efficiently locate genes influencing both complex and Mendelian traits. Indeed, one of the major motivations behind the international HapMap project [16,17] is to provide SNP (single nucleotide polymorphism) data from several populations, at a density of about one SNP per one to five Kb, to facilitate association mapping (in humans). The association mapping approach uses (sparse) data obtained from a number of *unrelated* individuals in a population, looking for sites, or small regions, whose states strongly discriminate between those individuals (called "cases") with the trait of interest, and those without it (called "controls"). Association mapping relies on the assumption that the cases (or a significant fraction of them) share a genealogical history that is distinct from the history of the controls, and that over time, meiotic recombination has shortened the shared region(s) containing the causative mutation(s). It follows from these assumptions that SNP sites near a causative mutation will have states (alleles) that more highly correlate with the trait of interest than do sites that are far from a causative mutation, and this is the general basis for association mapping. The following papers provide good overviews and discussions of association mapping [28,5].

The logic behind association mapping implies that the best possible mapping results would be obtained if the true genealogical history was explicitly known for the cases and controls. Such a history would be in the form of an "ancestral recombination graph (ARG)" [7,26,13], also called a "phylogenetic network" in [9,8]. The true ARG would explicitly show all the ancestral relations, the mutations and the recombinations that lead to the extant SNP sequences of the sampled individuals, starting from some ancestral SNP sequence. Quoting from a recent paper (also see [23]):

> Unless we have the actual disease variants in our marker set, the best information that we could possibly get about association is to know the full coalescent genealogy of our sample at that position. If we knew this, the marker genotypes would provide no extra information; ... [35].

It is important to note that the concept of ARG in this paper is synonymous with phylogenetic network, which is about a network showing ancestral relations among samples. This use of the term "ARG" is similar to the usage of this term in [23], and is *different* from a stochastic process called coalescent-with-recombination. Even with this restriction, an ARG is still very informative about the genealogical history of the samples.

Despite the conceptual centrality of the genealogical history, few association mapping methods have tried to explicitly deduce or exploit the underlying genealogical histories to map complex traits, particularly with recombination. Initial work on association mapping with ARGs by a full coalescent likelihood approach suggests that this is indeed a very challenging problem [19], and most existing genealogy-based mapping approaches [33,22,24,27] make some approximations (i.e. not using a full genealogical network) in modeling recombination.

Recently, however, a few papers have developed association mapping methods that explicitly try to exploit recombination or some aspects of the "underlying ARG space", that is, the set of ARGs that can generate a given input set of SNP sequences. Two papers, the first published in 2005 by Zollner and Pritchard [35], and the second by Minichiello and Durbin [23], are the most highly developed examples of these efforts.

The method of Zollner and Pritchard [35] explicitly uses some inferred information on recombination for genealogy inference, although it does not generate full ARGs. The method uses a rigorous stochastic framework and disease model to map certain kinds of complex traits. The basic strategy in [35] is to generate, and average over, some information from many samples of the ARG space for the given data. In particular it generates (independent) subtrees embedded in the ARGs, at different loci. Each such tree describes how the SNP sequences, restricted to an interval of SNP sites, could have evolved. The quality of a subtree is assessed using a rigorous statistical model (detailed later). A locus where many generated subtrees have significant scores is then deduced as being near a site that is causative for the trait.

The Zollner and Pritchard paper is an important advance because it defines a formal disease model, and it uses a rigorous likelihood approach to evaluate the significance of the mapping results. Moreover, it considers complex phenotypes showing "allelic heterogeneity", where the genetic basis for the trait can be a mutation at one of several different sites, but the sites are located close to each other. This is the case for example for BRCA1, or for mutations causative for the ability to metabolize lactose in adults. However, the implemented method (based on Markov Chain Monte Carlo) is very slow in practice, does not guarantee proper mixing, and does not use the full ARG model. Moreover, the disease model is somewhat limited and it would be desirable to extend it in several natural directions.

More recently, Minichiello and Durbin [23] developed an association mapping method that is similar to that in [35] at a high-level, but quite different in detail. In [23], full "plausible" ARGs are explicitly generated by using heuristics that allow rapid computation. There is no precise definition of what a "plausible" ARG is, although the algorithm tries to locally reduce the number of recombinations used, and can be viewed as producing an approximation to a "minARG" [29], i.e., an ARG that globally minimizes the number of recombinations used to generate the SNP sequences (in a model detailed below). There is no characterization of the sampling bias that is caused when ARGs are created in this way.

Our paper is centrally motivated by the paper of Zollner and Pritchard [35]. Our paper addresses computational, and some statistical, challenges from [35], with results concerning all the essential steps of the method in [35]. Essentially, we show that the statistical approach in [35] can be sped up and made practical when full minARGs (or near-minimum ARGs) are sampled. We adopt the disease model introduced in [35] and present new results in evaluating the "phenotype likelihood", used to assess the significance of a "marginal tree" (defined below). However, some parts of our method are more similar to the method in [23], and so some of our results also relate to that paper.

2 Definitions and Background

A *single nucleotide polymorphism (SNP)* is a single nucleotide site where exactly two (of four) different nucleotides occur in a large percentage of the population. A *genotype* comes from a pair of *haplotypes.* As in [35] we assume that haplotypes can be determined from sampled genotypes, and to simplify the exposition, we just say that haplotypes are sampled in the population. The set of haplotypes sampled from a population is denoted by M, where M has n haplotypes (rows) and m sites (columns). We assume at most one mutation in any sampled SNP site in the evolution of the haplotypes, which is supported by the standard "infinite sites model" in population genetics [13,14]. This is particularly justified in the context of association mapping where the time-scale of interest is short enough that two mutations at any single site are unlikely. In addition to mutation, haplotypes evolve by (meiotic) recombination. Recombination takes two equal length sequences (haplotypes) and produces a third sequence of the same length consisting of some *prefix* of one sequence, denoted P, followed by a *suffix*, of the other sequence, denoted S. The changeover point is called the "crossover point" or "breakpoint".

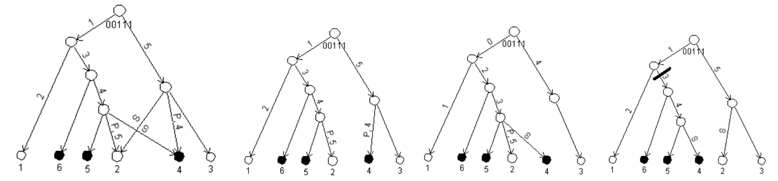
The evolutionary history of a set of haplotypes M, that evolve by mutations and recombinations is displayed on a rooted, directed acyclic graph called an "Ancestral Recombination Graph (ARG)" [7] (also in [26,13]), or a "Phylogenetic Network" in [9,8]. An example of an ARG is shown in Figure 1(b); A formal definition of an ARG is given in [9,8]. An ARG that derives a set of sequences M and *minimizes* the number of recombinations assuming at most one mutation per site is called a "minARG" [29]. The problem of finding a minARG for M, or even determining the number of recombinations in it, is NP-hard [34,3], but there are methods constructing minARGs for moderate-size haplotype data [29,21]. Other methods construct minARGs with some structural constraints [8,9,10]. We can also efficiently compute close upper bounds [11,12,31] and lower bounds [15,25,30,1,2,31] on the number of recombinations in a minARG.

2.1 The Marginal Trees of an ARG

Let N denote an ARG for M. The following crucial observation is central in the methods in [35] and [23] and in our method. For any site x, the full evolutionary

1	1 1 1 1 1
2	1 0 0 0 0
3	0 0 1 1 0
4	0 0 1 0 1
5	1 0 0 0 1
6	1 0 0 1 1

(a) SNP data.



(b) An ARG deriving the data.

(c) Marginal tree for positions between sites 1-3

(d) Marginal tree for positions near site 4

(e) Marginal tree to the right of site 4 fits the phenotypes well

Fig. 1. An example of the general association mapping method using ARGs. Figure 1(a) shows the input haplotypes M, where rows 1-3 are controls and rows 4-6 are cases. Figure 1(b) shows an ARG for the haplotypes in Figure 1(a), with 00111 as root sequence. A site mutation changes the state from that at the root to the opposite state. Leaves are labeled with row numbers in the input matrix. Figures 1(c) is the marginal tree embedded in the ARG at positions between site 1 and 3. Figure 1(d) is the marginal tree near site 4. Figure 1(e) is the marginal tree to the right of site 4. Note this tree shows a cut of an edge that clearly separate cases and controls.

history of the states of site x in the sequences M, is completely represented by a *subtree* T_x of N, which can be extracted from N by removing, at each recombination node v in N, one of the two directed edges entering v. In particular, suppose b is the breakpoint for the recombination at v; then remove the edge into v from the node labeled with the sequence P (providing the prefix for the recombination), if site x is to the right of b; otherwise remove the edge into v from the node labeled S. The resulting subtree, T_x, is a rooted directed tree that details how each of the sequences in M obtained their polymorphism value at x. An example is given in Figure 1. Tree T_x is called a *marginal tree*. The next critical point is that if no recombination in N has occurred at a breakpoint between sites x and y, then the marginal trees T_x and T_y are identical. Hence, there is a *single* well-defined marginal tree for each interval between successive breakpoints in N (along the linear ordering of the polymorphic sites), and also for the two intervals before the first, and after the last, breakpoints.

3 The High-Level Strategy

Our high-level strategy involves sampling ARGs from the ARG space; then constructing the marginal trees for each sampled ARG; and then assessing the statistical significance of the marginal trees given the observed cases and controls, and the disease model. To get the intuition behind this assessment, suppose the disease is Mendelian. In that case a significant marginal tree is expected to have an edge, such that there is a "larger than random" number of cases in the leaves of the subtree below the edge, and few controls. A locus contained in marginal trees that have high significance is then deduced as being near a site that is causative for the trait. This basic idea is adopted in [23]. The general method, in the Mendelian case, is illustrated in Figure 1, where the third marginal tree corresponding to sites to the right of site 4, contains an edge that perfectly separates the cases from the controls. Therefore, the tree to the right of site 4 *fits* the observed phenotypes well. If no further sampling were done, we might then conclude that the genomic region near site 5 is the most likely to contain the causative mutation for the trait. Now, for complex diseases, the assessment of the significance of a marginal tree is more involved. Then we need to compute the probability, given the disease model, that the observed cases and controls would have been derived on that marginal tree. If the disease model specifies low penetrance or multiple causative mutations, then we would not expect a perfect separation of the cases and controls as in Figure 1.

The above high-level strategy is in the same spirit as some parts of the methods in [35] and [23], but our method differs from, extends and sometimes outperforms those methods in several important ways. Most importantly, in contrast to both earlier methods, our methods explicitly compute minARGs, or ARGs that (empirically) use a number of recombinations close to the global minimum, rather than other ARGs from the ARG space. There are several reasons we want to generate minARGs (or near minimum ARGs) rather than other ARGs from the ARG space. First, it is currently believed that in the human genome (and perhaps others) there are many regions (haplotype blocks) where the recombination rate is low, but not zero. In those regions, we expect that minARGs reflect the true genealogical history better than an ARG with many extra recombinations. Also, the method in [23] has implicit (but not rigorous) rules for reducing the number of recombinations used, and the use of minARGs better formalizes that implicit effect.

Phenotype likelihood is defined as, given a marginal tree T_x at position x with leaf labels, the probability $Pr(\Phi|X = x, T_x)$ of the observed phenotypes Φ of the leaves being generated on T_x according to the following disease model assuming disease mutations occur near x [35]. Here Φ is the collection of case and control status for each sample sequence. The basic *disease model* that we adopt in this paper is the one introduced by Zollner and Pritchard [35].

– The disease loci are not sampled, i.e. are not in the sampled SNP sites.
– Phenotypes are determined by mutations at disease loci and the disease penetrance. There may be multiple independent disease mutations, but these mutations occur relatively close together, so that they may all occur on a single marginal tree.
– There are two alleles M_0 (wild-type) and M_1 (mutant) at a disease locus. Mutations M_0 to M_1 occur at edges of the marginal tree according to Poisson process with a rate of $\nu/2$. There is *no* mutation from M_1 to M_0. Furthermore, mutations on different edges of a marginal tree occur independently.
– Multiple mutations on the same haplotype have the same effect on phenotypes as a single mutation.

The concept of *penetrance* is important to this paper. Zollner and Pritchard uses the *haploid penetrance* model to specify the effect of alleles M_0, M_1 on phenotypes: $P_{\phi,m}$ is the probability of a haplotype exhibiting phenotype ϕ for wild-type haplotype ($m = 0$) or mutant haplotype ($m = 1$). Here, $\phi \in \{A, C\}$

(i.e. cAse or Control). Since $P_{\phi,m}$ is for a single sequence (haplotype), we call it haploid penetrance. In practice, penetrances are often not known. Zollner and Pritchard used a numerical integration approach, averaging over a grid points of penetrance (e.g. an evenly spaced 20 by 20 grid with range [0.0, 1.0], where each point represents a possible penetrance $P_{A,0}$ and $P_{A,1}$). Thus in the following, we assume penetrance is known when we compute phenotype likelihood.

Suppose we are given a (rooted) tree T_x. Let M be a binary vector with one bit for each edge, indicating whether the corresponding edge has at least one disease mutation or not. Then,

$$Pr(\Phi|X = x, T_x) = \sum_{M} P_{A,0}^{n_{A,0}} P_{A,1}^{n_{A,1}} P_{C,0}^{n_{C,0}} P_{C,1}^{n_{C,1}} Pr(M|x, T_x)$$

where $n_{\phi,m}$ = number of sequences in the sample showing phenotype ϕ who have mutation state m, and $Pr(M|x, T_x)$ is the probability of the edge mutations specified by M in T_x, which can be easily computed. Zollner and Pritchard use the *Peeling algorithm* [6] to efficiently compute phenotype likelihood with haploid penetrance. Understanding this is important for our new results on phenotype likelihood in Section 5. Refer to [35] for more details.

4 Sampling Ancestral Recombination Graphs

4.1 Uniform Sampling of minARGs

Now we present two methods for sampling ARGs given a set of sequences M. We first present a method to count the number of minARGs. With a simple modification, the method can be turned into a uniform sampler of minARGs.

We begin by reviewing the *self-derivability* problem originally studied in [31]. In the following, we assume that M does not contain two identical sequences. Often an ARG may derive sequences that are not present in input data M. We call these sequences *Steiner* sequences. The self-derivability problem is to decide whether there is an ARG N deriving M (assuming at most one mutation per site) which only contains the input sequences in M. That is, N contains no Steiner sequences. We call such an ARG, if it exists, a *self-derived* ARG. We first describe algorithms for data with self-derived ARGs, and then extend to more general case. Lemma 1 is a simple extension of a result in [31] and we omit the proof here.

Lemma 1. *A self-derived ARG is also a minARG.*

Here, we give an algorithm (Algorithm 1) for counting the number of self-derived ARGs for M. The algorithm runs in $O(2^n + n^3 m)$ time. Two ARGs are different if they derive a different set of sequences. Moreover, the nodes (i.e. sequences) in an ARG are derived in a particular linear time order. That is, for any two nodes, we know which corresponding sequence is derived earlier. We consider two ARGs to be different if the derivation order of the sequences in them are different, even if they are topologically identical and derive the same set of sequences. This total time-order property is quite convenient to avoid over-counting (as explained later). Also, the time-ordering suggests ways to determine edge lengths for the edges in an ARG, and this will be useful for the phenotype likelihood computation, discussed later. Moreover, true ARGs are total-ordered since genealogical events are time-ordered.

For each subset $S \subseteq Rows(M)$, we define $N[S]$ as the *number* of the minARGs deriving sequences in S (and deriving no other sequences). Therefore, $N[Rows(M)]$ is equal to the total number of self-derived ARGs that derive haplotype matrix M. For a subset of rows S and a single row $r \notin S$, we denote $D(S, r)$ as the total number of ways of deriving r by sequences in S through an unused mutation or a recombination. By "unused mutation" we mean a mutation at a site where all sequences in S have the same states (i.e. either all 0 or all 1) and different from that of r. Note that if r is derived by an unused mutation, this is only one way of deriving r and no two sequences in S can recombine to derive r. Similarly, when r can be derived through recombination of two sequences in S, r can not be derived through an unused mutation from a sequence in S. Thus, there are the following mutually exclusive situations:

1. If a sequence in S can derive r by an unused mutation, then $D(S, r) = 1$.
2. If two sequences in S can derive r by a recombination, then $D(S, r) \geq 1$.
3. Otherwise, $D(S, r) = 0$.

Algorithm 1

```
1. For each row $r \in M$, set $N[\{r\}] \leftarrow 1$.
2. Set $sz \leftarrow 1$.
3. while $sz < n$
   3.1 $sz \leftarrow sz + 1$
   3.2 For each subset of rows $S \subseteq Rows(M)$ and $|S| = sz$, initialize $N[S] \leftarrow 0$.
       3.2.1 For all $r \in S$, such that (a) $N[S - \{r\}] \geq 1$, and (b) $D(S - \{r\}, r) \geq 1$,
             then $N[S] \leftarrow N[S] + N[S - \{r\}] \times D(S - \{r\}, r)$.
```

The correctness of Algorithm 1 is easy to establish from the total time-ordered property. The key observation is that since the ARG is fully time ordered, each way of choosing r generates a different ARG. Intuitively, picking r means we choose to derive r *immediately* after the sequences in $S - \{r\}$ in the ARG. Two ARGs generated by picking r_1 and r_2 respectively at the same stage are different because the time order of r_1, r_2 is different.

Now we show that Algorithm 1 can be converted to a uniform sampler of minARGs for data M when M is self-derivable. This is presented in the following.

Algorithm 2

```
1. First count the total time-ordered minARGs using Algorithm 1.
2. Initialize $S$, the set of current underived rows, to $Rows(M)$. Do:
   2.1 Choose a sequence $r$ from $S$ as the last sequence in $S$ to derive, with
       probability $\frac{N(S-\{r\}) \times D(S-\{r\},r)}{N(S)}$.
```

2.2 Choose uniformly at random (and remember) a derivation way (i.e. either through a mutation or a recombination) for r from total $D(S - \{r\}, r)$ possible ways of derivation.
2.3 Let $S \leftarrow S - \{r\}$. Go to Step 3 if $|S| = 1$ (i.e. the only remaining sequence is the root sequence). Otherwise, continue on step 2.1.

3. Construct an ARG (starting from the root) according to the (reverse) order for each sequence r and the chosen way of deriving r.

It is easy to show that the above algorithm is indeed a uniform sampler of self-derived minARGs. Here is the intuitive idea behind this method. We construct a uniformly sampled time-ordered minARG backwards in time. That is, we decide how to derive the last sequence first. For a given data, we choose a sequence r as the last sequence to derive in the minARG with probability equal to the ratio of the number of minARGs with r as the last sequence to the total number of minARGs. This ensures that the last sequence is picked uniformly. We pick the rest of sequences uniformly backwards in time. Thus, the generated ARG is sampled uniformly. We remark that the (exponential) set-up time for the uniform sampling is the dominant portion of the running time. Note however that once the counting is performed, sampling of an ARG takes $O(n^2)$ time.

A remaining issue is that the above uniform sampling method only works for a special type of data M, namely the data where M is self-derivable. The method can be extended to handle general data for moderate-sized datasets as follows. When the number of needed Steiner sequences and the number of candidate Steiner sequences are small, we can simply add Steiner sequences to M in order to make the expanded dataset self-derivable. This and several other ideas that we have implemented make uniform sampling of minARGs practical in a range of data we report in Section 6. Recall also that association mapping is often done on candidate regions or on windows in a genome, and these also fall in the range of practicality for minARG generation. The efficiency of our minARG sampling method depends largely on the number of haplotypes (and number of Steiner sequences needed) of M. Our experience indicates that minARGs can often be found (within practical amount of time) for data with up to 30 haplotypes when the data is self-derivable, and up to 20 haplotypes when the number of sites is small and one or two Steiner sequences are needed.

Remarks. The above uniform sampling can be extended to weighted sampling, where larger weights are assigned to more likely operations. Weighted sampling may improve the mapping accuracy. We omit the details here.

4.2 ARG Sampling for Larger Data

The performance of the uniform sampling method degrades with the increase of the number of haplotypes in M or the number of needed Steiner sequences. To sample ARGs for larger data, we need heuristic sampling methods.

The efficient ARG sampling method presented here samples a special type of ARGs, where sequences are derived by a derivation pathway. Constructing a single ARG by a derivation pathway has been previously used in [31], and implicit application of the pathway is also used in [20] for estimating recombination rate. In this paper, we develop an ARG sampling method based on minimum pathway. Minimum pathway is a way of deriving a new sequence from a set of derived sequences using the fewest recombinations. Thus, we also explicitly reduce the number of recombinations here. We also sample uniformly derivation paths from minimum pathways. We call the ARGs constructed from pathways as pathway ARGs, and this sampling method *pathway sampling.*

The detailed description of the pathway ARG sampling method together with a uniform sampling method of a derivation path from the minimum pathway for a fixed derivation order is deferred to the full version of this paper.

5 Phenotype Likelihood

5.1 More on Phenotype Likelihood

An alternative to the the phenotype likelihood (denoted as PL and described in Section 3) in [35] is that, instead of summing over all possible subsets of mutated edges, we seek *one* subset of edges in the marginal tree where disease mutations occur, which *maximizes* the probability of the observed phenotypes caused by the chosen mutations [4]. Due to the lack of a better name, we name the maximized probability as maximum phenotype likelihood (MPL). More precisely, MPL is equal to $MAX_M(Pr(\Phi|M, X = x, T_x) * Pr(M|X = x, T_x))$, where M is a vector indicating on which tree edges mutations occur. It is easy to demonstrate an efficient procedure (details omitted) for computing maximum likelihood with haploid penetrance by dynamic programming (a variant of the peeling algorithm used in [35]). Initial experiments show, however, mixed results of using MPL as the scoring scheme. Thus, in this paper, we adopt the original PL scheme as the scoring scheme.

5.2 Expected Phenotype Likelihood

An important computational problem for a statistical method is assessing statistical significance of the results. For the phenotype likelihood problem, a natural question to ask is whether the given phenotypes are indeed caused by disease mutations (i.e. the alternative model) or just some random noise (i.e. the null model). Imagine that we randomly permute phenotypes of leaves in a given tree (i.e. without changing the number of cases). A commonly used scheme to assess statistical significance is to compute a P-value, which is the adopted method in [35]. In practice, computing the P-value is often through permutation tests. Permutation tests may not give accurate result and are time-consuming, and may be the bottleneck of whole genome scan for trait mapping [35].

Besides the P-value, other statistics may provide hints on statistical significance, including, for example, the expected phenotype likelihood. It might be possible to develop another scheme for assessing significance involving the expected likelihood. Unlike the permutation tests used in [35], our method computing the expected phenotype likelihood is an exact method, running in polynomial-time and fully deterministic. In the following, we show that the expected value of phenotype likelihood with haploid penetrance can be efficiently computed. With some small modifications, we can also compute variance of phenotype likelihood with a polynomial-time algorithm (details omitted). For simplicity, we assume the marginal tree T is binary. Note that if the given T is not binary, we can easily transform T to a binary tree T' without changing the phenotype likelihood [4].

We define $L_r(s, m, k)$ to be expected (randomized) phenotype likelihood for the subtree under node (i.e. sequence) K_s where node K_s has disease mutation state m and the subtree contains exactly k case haplotypes. Recall that m is either 0 (K_s is a wild-type) or 1 (K_s is a mutant). The base case when K_s is a leaf is easy. We have, $L_r(s, 0, 0) = 1.0 - P_{A,0}$, $L_r(s, 1, 0) = 1.0 - P_{A,1}$, $L_r(s, 0, 1) = P_{A,0}$, and $L_r(s, 1, 1) = P_{A,1}$.

Now consider an internal node K_s with k case haplotypes under K_s. Denote the number of leaves (both cases and controls) under K_s as $k_{t,s}$. Denote the two children of K_s as K_{s_l} and K_{s_r}. Denote μ_l is the probability of at least one mutation occurs at edge from K_s to K_{s_l}. Similarly, μ_r is defined for K_{s_r}. There are up to $O(k)$ different way of splitting the k cases into two subtrees under K_{s_l} and K_{s_r}. Suppose in one way of splitting, we have k_1 cases in subtree under K_{s_l} and $k - k_1$ cases in subtree under K_{s_r}. The probability of such split is equal to the probability of a randomly chosen k_{t,s_l} balls from total $k_{t,s}$ balls (with k black balls) such that k_1 black balls are chosen. This is equal to:

$$P_s(s, k, k_1) = \frac{\binom{k}{k_1}\binom{k_{t,s}-k}{k_{t,s_l}-k_1}}{\binom{k_{t,s}}{k_{t,s_l}}}$$

Therefore, we have the following recursions (whose proof is omitted) :

$$L_r(s, 1, k) = \sum_{k_1} P_s(s, k, k_1) L_r(s_l, 1, k_1) L_r(s_r, 1, k - k_1)$$

When $d = 0$, we need to consider the probability of edge mutation as well.

$$\begin{aligned} L_r(s, 0, k) = \sum_{k_1} P_s(s, k, k_1) * ((1 - \mu_l) L_r(s_l, 0, k_1) + \mu_l L_r(s_l, 1, k_1)) * \\ ((1 - \mu_r) L_r(s_r, 0, k - k_1) + \mu_r L_r(s_r, 1, k - k_1)) \end{aligned}$$

Note that the expected phenotype likelihood is precisely $L_r(r, 0, n_c)$, where K_r is the root of the tree and n_c is the number of cases. The above recursion can be easily implemented in a dynamic programming algorithm with $O(n^3)$ running time.

5.3 Diploid Penetrance

Zollner and Pritchard used haploid penetrance in phenotype likelihood computation. Since we are mostly interested in diploid samples, diploid penetrance, rather than haploid penetrance, seems more natural. A diploid sample contains *two* haplotypes, and its phenotype is decided by the *joint* mutation status of the two haplotypes. In diploid penetrance, we have $P_{\phi,00}$, which is the probability of a diploid sample exhibiting phenotype ϕ if *both* of its haplotypes are wild-type haplotypes. Similarly, $P_{\phi,01}$ (resp. $P_{\phi,11}$) is the probability of a diploid sample exhibiting phenotype ϕ if exactly one (resp. none) of its haplotypes is wild-type haplotype. An important question stated but unanswered in Zollner and Pritchard [35] is how to efficiently compute phenotype (denoted $Pr_d(\Phi|X = x, T_x)$) likelihood using diploid penetrance model.

The main difference between $Pr_d(\Phi|X = x, T_x)$ and $Pr(\Phi|X = x, T_x)$ is that the diploid likelihood considers a diploid individual as a single entity rather than two haplotypes as in haploid likelihood. Similar to the haploid penetrance case, we can define the maximum likelihood problem with diploid penetrance. For easy reference, we name the problem of computing $Pr_d(\Phi|X = x, T_x)$ Diploid-Phenotype-Likelihood or DPL. We name the problem regarding maximum likelihood as Max-Diploid-Phenotype-Likelihood or MDPL. Theorem 1 says that diploid likelihood problems are NP-hard, whose proof is deferred to the full version of this paper.

Theorem 1. *The MDPL and DPL problems are both NP-hard.*

6 Experimental Results

The described methods are implemented using C++ in an association mapping program, which we name as Trait Mapping tool with ARG (TMARG). For testing statistical significance of phenotypes, TMARG uses the PL scheme with haploid penetrance as the scoring scheme. Different from LATAG (program developed in [35]), we take maximum of PL values over penetrance grid points, rather than taking average. Initial experiences show that taking maximum slightly improves mapping accuracy. However, taking maximum also appears to give less repeatable mapping results.

TMARG takes a matrix of haplotypes or phase-known genotypes and their phenotypes (i.e. case/control status) as input, and provides point estimate for the complex trait loci. TMARG supports both uniform sampling of minARGs in a sliding window, and pathway ARG sampling for the *entire* data. TMARG currently only allows data consisting of binary SNPs. For unphased or noisy data, we suggest to first preprocess the data using haplotype inference programs, such as PHASE [32]. To evaluate the effectiveness of our program, we test with both real biological data and simulated data. We compare TMARG with both LATAG and MARGARITA (the program developed in [23]). When running MARGARITA, we sample 50 ARGs and perform 10000 permutations for each data.

The first data is the Cystic Fibrosis (CF) data [18], which has been analyzed by many association mapping methods. It contains 23 binary markers over 1.8 Mb. There are 94 disease haplotypes and 92 control haplotypes. The most common mutation is located 885 Kb from the left end of the region. We use program

PHASE 2.1 [32] to impute missing data. The uniform sampling scheme gives the point estimate at 1096 Kb, while the pathway sampling scheme gives 915 Kb. For each method, we perform 50 independent runs, while in each run we sample 50 genealogies. The reported results are the consensus point estimates over the 50 runs. The LATAG's point estimate is at 867 Kb, while MARGARITA's point estimate is at 870 Kb.

The second data contains 50 simulated datasets used in Zollner and Pritchard [35], which we call ZPS data. These data were intended for testing effectiveness of gene mapping methods regarding complex traits. Each ZPS data contains 30 diploid cases and 30 diploid controls (with known phases). Each data typically contains between 45 to 65 binary markers. The generation of these data essentially follows from the disease model in Section 5. Typically 10-25 disease mutations (including many redundant) are generated for each data. One reason that these data may not be easy for mapping is that only part (10 to 33 among 60) of case haplotypes do actually carry disease mutations while some (0 to 9 among 60) control haplotypes also carry disease mutations. Table 1 lists our mapping result using TMARG, comparing to LATAG/MARGARITA.

Our simulation results show that TMARG is comparable with LATAG and MARGARITA for CF data and slightly outperforms the other two programs in accuracy for the ZPS data. Note that we only tested MARGARITA with one settings and its mapping result may change when using different settings (e.g. more permutations per data). We note that both uniform sampling and pathway sampling methods are comparable to MARGARITA in speed and are much faster than LATAG for the data we tested (when same number of samples are generated). For example, for the CF data, LATAG takes 8 hours for each run, while our sampling methods take a few minutes. As seen in Table 1, pathway sampling method appears to be slightly more accurate than the uniform sampling method and sampling more ARGs per data may slightly improve mapping accuracy. On the other hand, uniform sampling appears to produce more repeatable results than the pathway sampling. We remark that more simulation tests on both real biological and simulated data are needed to further validate our proposed methods and compare our methods with LATAG/MARGARITA.

Table 1. Mapping for simulated data in Zollner and Pritchard [35]. We test both **U**niform and **P**athway sampling schemes. We measure the accuracy by the average point estimate error, standard error and percentage of data with point estimate within 0.1 cM distance from the true trait loci for the 50 datasets. The units of all the point estimates are cM. The results of TMARG are consensus from 10 independent runs, where each run sample 50 or 5000 genealogies.

	U	P	P	LATAG	MARGARITA
Sample Num	50	50	5000	50	50
Ave. Err.	0.184	0.180	**0.166**	0.19	0.229
Std. Err.	0.215	0.210	0.197	0.23	0.255
% < 0.1 cM	50%	50%	56%	54%	44%

Acknowledgments. I am very grateful to my advisor, Dan Gusfield, for many invaluable discussions and for reading the manuscript carefully and making many good suggestions. I would like to thank Dan Brown for very fruitful and stimulating discussions on the phenotype likelihood problems. I thank Chuck Langley for making several important comments on the manuscript. I also thank Yun S. Song and Katie Pollard for helpful discussions. Work supported by grants CCF-0515278 and IIS-0513910 from National Science Foundation. Simulations are performed on a Linux cluster supported by NSF grant CNS-0224469.

References

1. V. Bafna and V. Bansal: The number of recombination events in a sample history: conflict graph and lower bounds. IEEE/ACM Trans. on Computational Biology and Bioinformatics, v.1, 78-90, 2004.
2. V. Bafna and V. Bansal, Inference about Recombination from Haplotype Data: Lower Bounds and Recombination Hotspots. J. of Comp. Bio., v.13, p.501-521, 2006.
3. M. Bordewich and C. Semple: On the computational complexity of the rooted subtree prune and regraft distance. Annals of Combinatorics, v.8, p.409-423, 2004.
4. D. Brown: Private communications.
5. A. G. Clark: Finding genes underlying risk of complex disease by linkage disequilibrium mapping. Current Opinion in Genetics and Development, 13, p296-302, 2003.
6. J. Felsenstein, Evolutionary trees from DNA sequences: a maximum likelihood approach, J. Mol. Evol., v.17, p.368-376, 1981.
7. R. C. Griffiths and P. Marjoram: Ancestral inference from samples of DNA sequences with recombination. J. of Comp. Bio., v.3, p479-502, 1996.
8. D. Gusfield: Optimal, efficient reconstruction of Root-Unknown phylogenetic networks with constrained and structured recombination. JCSS, 70, 381-398, 2005.
9. D. Gusfield, S. Eddhu and C. Langley: Optimal, efficient reconstruction of phylogenetic networks with constrained recombination. J. Bioinformatics and Computational Biology, v. 2, 173-213, 2004.
10. D. Gusfield, S. Eddhu and C. Langley: The fine structure of galls in phylogenetic networks. INFORMS J. on Computing, v. 16, p.459-469, 2004.
11. J. Hein: Reconstructing evolution of sequences subject to recombination using parsimony. Math. Biosci, v. 98, p.185-200, 1990.
12. J. Hein, A heuristic method to reconstruct the history of sequences subject to recombination. J. Mol. Evol., v.36, p.396-405, 1993.
13. J. Hein, M. Schierup and C. Wiuf: Gene Genealogies, Variation and Evolution: A primer in coalescent theory. Oxford University Press, 2005.
14. D. Hinds, L. Stuve, G. Nilsen, E. Halperin, E. Eskin, D. Gallinger, K. Frazer and D. Cox: Whole-Genome Patterns of Common DNA variation in three human populations. Science, v. 307, p.1072-1079, 2005.
15. R. Hudson and N. Kaplan: Statistical properties of the number of recombination events in the history of a sample of DNA sequences. Genetics, v.111, p.147-164, 1985.
16. International HapMap Consortium: The HapMap project, Nature, 426, p789-796, 2003.

17. International HapMap Consortium: A haplotype map of the human genome. Nature, 437, p1299-1320, 2005.
18. B. Kerem, J. M. Rommens, J. A. Buchanan, D. Markiewicz, T. K. Cox, A. Chakravarti, M. Buchwald and L. C. Tsui: Identification of the cystic fibrosis gene: genetic analysis. Science, 245, p1073-1080, 1989.
19. F. Larribe, S. Lessard and N. J. Schork: Gene mapping via Ancestral Recombination Graph. Theor. Popul. Biol., v. 62, p. 215-229, 2002.
20. N. Li and M. Stephens: Modeling Linkage Disequilibrium, and identifying recombination hotspots using SNP data. Genetics, 165, p2213-2233, 2003.
21. R. Lyngso, Y. S. Song and J. Hein: Minimum Recombination Histories by Branch and Bound. Proceedings of Workshop on Algorithm of Bioinformatics (WABI) 2005, v.3692, p.239-250.
22. M. S. McPeek and A. Strahs: Assessment of linkage disequilibrium by the decay of haplotype sharing, with application to fine-scale genetic mapping. Am. J. Hum. Genet., 65, p858-875, 1999.
23. M. Minichiello and R. Durbin: Mapping trait loci using inferred ancestral recombination graphs. Am. J. Hum. Genet., v.79, p.910-922, 2006.
24. A. P. Morris, J. C. Whittaker and D. J. Balding: Fine-scale mapping of disease loci via shattered coalescent modeling of genealogies. Am. J. Hum. Genet., 70, p686-707, 2002.
25. S. R. Myers and R. C. Griffiths: Bounds on the minimum number of recombination events in a sample history. Genetics, v. 163, p.375-394, 2003.
26. M. Norborg and S. Tavare: Linkage disequilibrium: what history has to tell us. Trends in Genetics, 18, 83-90, 2002.
27. B. Rannala and J. P. Reeve: High-resolution multipoint linkage-disequilibrium mapping in the context of a human genome sequence. Am. J. Hum. Genet., 69, p159-178, 2001.
28. N. Risch and K. Merikangas: The Future of Genetic Studies of Complex Human Diseases. Science, 275, p1516-1517, 1996.
29. Y. Song and J. Hein: Parsimonious reconstruction of sequence evolution and haplotype blocks: Finding the minimum number of recombination events. Proc. of 2003 Workshop on Algorithms in Bioinformatics (WABI), 2003.
30. Y. Song and J. Hein: On the Minimum Number of Recombination Events in the Evolutionary History of DNA Sequences. J. of Math. Biology, v.48, p.160-186, 2003.
31. Y. S. Song, Y. Wu and D. Gusfield: Efficient computation of close lower and upper bounds on the minimum number of needed recombinations in the evolution of biological sequences. Bioinformatics, v. 421, p.i413-i422, Proceedings of ISMB 2005.
32. M. Stephens, N. Smith and P. Donnelly: A new statistical method for haplotype reconstruction from population data. Am. J. Hum. Genet., v. 68, p.978-989, 2001.
33. A. R. Templeton, E. Boerwinkle and C. F. Sing: A cladistic analysis of phenotypic associations with haplotypes inferred from restriction endonuclease mapping. I. Basic theory and an analysis of alcohol dehydrogenase activity in Drosophila. Genetics, 117, p343-351, 1987.
34. L. Wang, K. Zhang and L. Zhang: Perfect Phylogenetic Networks with Recombination. J. of Comp. Bio., v.8, p.69-78, 2001.
35. S. Zollner and J.K. Pritchard: Coalescent-Based Association Mapping and Fine Mapping of Complex Trait Loci. Genetics, 169, p. 1071-1092, 2005.

An Efficient and Accurate Graph-Based Approach to Detect Population Substructure

Srinath Sridhar[1], Satish Rao[2], and Eran Halperin[3]

[1] Computer Science Dept, Carnegie Mellon University
srinath@cs.cmu.edu
[2] Computer Science Dept, University of California, Berkeley
satishr@eecs.berkeley.edu
[3] International Computer Science Institute (ICSI), Berkeley
heran@icsi.berkeley.edu

Abstract. Currently, large-scale projects are underway to perform whole genome disease association studies. Such studies involve the genotyping of hundreds of thousands of SNP markers. One of the main obstacles in performing such studies is that the underlying population substructure could artificially inflate the p-values, thereby generating a lot of false positives. Although existing tools cope well with very distinct sub-populations, closely related population groups remain a major cause of concern.

In this work, we present a graph based approach to detect population substructure.Our method is based on a distance measure between individuals. We show analytically that when the allele frequency differences between the two populations are large enough (in the l_2-norm sense), our algorithm is guaranteed to find the correct classification of individuals to sub-populations.

We demonstrate the empirical performance of our algorithms on simulated and real data and compare it against existing methods, namely the widely used software method `STRUCTURE` and the recent method `EIGENSTRAT`. Our new technique is highly efficient (in particular it is hundreds of times faster than `STRUCTURE`), and overall it is more accurate than the two other methods in classifying individuals into sub-populations. We demonstrate empirically that unlike the other two methods, the accuracy of our algorithm consistently increases with the number of SNPs genotyped. Finally, we demonstrate that the efficiency of our method can be used to assess the significance of the resulting clusters. Surprisingly, we find that the different methods find population sub-structure in each of the homogeneous populations of the HapMap project. We use our significance score to demonstrate that these substructures are probably due to over-fitting.

1 Introduction

Studying the etiology of common complex disease such as cancer, or Parkinson's disease, is an important task in the search for better treatments and diagnosis tools for these diseases. A common practice towards this task is to perform an association study, in which the genetic variation of a set of cases (individuals carrying the disease) and a set of controls (background population) is compared, and large discrepancies between the two populations indicate an association of a specific locus with the studied phenotype.

There are different forms of genetic variations that can be studied in the context of association tests, the most common one is single nucleotide polymorphisms (SNPs), which are nucleotides in the genome that are found to be varying among different individuals. In general, these SNPs are bi-allelic, that is only two alleles are found in the population. SNPs are commonly used in association studies, as the SNP variation is believed to capture most of the human genetic variation [4,5,6], and furthermore, recent technology (e.g. Affymetrix or Illumina) allows the genotyping of hundreds of thousands of SNPs per individual for a couple of hundred dollars. Thus, whole genome association studies, in which hundreds of thousands of SNPs are genotyped for thousands of individuals is becoming a common practice.

The validity of the results of an association study heavily depends on the statistical analysis performed. One of the main growing concerns is that population substructure may raise spurious discoveries. In association studies, the discrepancies in the SNP-allele frequencies between the cases and the controls are believed to imply an association of the SNP with the disease, but if the cases and controls were collected from two very different populations, this discrepancy may be explained by the difference between the two populations, and hence the SNP is not necessarily associated with the disease. Even subtle differences in the population structures of the cases and the controls may result in spurious associations. In particular, this problem is becoming more acute when large scale association studies are performed(see for e.g., [8,15]).

There are many computer programs that try to cope with this problem, most notably the widely used software `STRUCTURE` [13] and a recently developed method `EIGENSTRAT` [14]. `STRUCTURE` uses a Markov Chain Monte Carlo (MCMC) approach to find population substructure of a given population using DNA variation data. `EIGENSTRAT` is based on principal component analysis (PCA). Mathematically, this problem can be seen as a clustering problem, in which the different clusters correspond to different populations. Such clustering problems have been studied under a variety of different theoretical frameworks that share close similarities. For instance the max-cut of a graph shares properties with the eigenvectors of the corresponding (adjacency or Laplacian) matrix and therefore Spectral methods or PCA, `STRUCTURE` and finding max-cuts of graphs share close mathematical relationships [1,2,3,12].

`STRUCTURE` has been used extensively in genetic studies (cited more than 700 times), and it has been shown to find population substructure quite accurately in many examples. Even though `STRUCTURE` performs very well in terms of accuracy, it is quite inefficient, and it may take weeks to run over one whole genome dataset. Furthermore, even though `STRUCTURE` outputs a likelihood score which assists in interpreting the results, it is not clear whether this likelihood score can be used to determine whether there is actually a significant presence of population substructure. Finally, as the MCMC is inherently a heuristic approach, it is hard to know which parameters to set for the algorithm; in particular, as we show in this paper, there is no uniform set of parameters that performs well for all the data-sets.

In order to cope with these problems, we introduce a new graph based method for clustering populations. We concentrate in this paper on the clustering of two populations, although the method can be easily extended to multiple populations. Our technique is based on a simple paradigm. We define a distance between every pair of individuals, and we then search for a maximum cut in the graph induced by these distances. From that cut, we perform a local search that maximizes the likelihood of the data, similar to the criterion used in `STRUCTURE`. The main advantage of our method is that it is extremely efficient, and at the same time very accurate. Furthermore, since eventually the algorithm optimizes the same score function as that of `STRUCTURE`, it can be viewed as a fast method that finds a local optimum for this criterion.

It is important to note that the efficiency of our method allows us to measure the significance of the population substructure by running our algorithm on thousands of permutations of the data. For instance, we find that both our method and `STRUCTURE` find a population substructure in the YRI population, genotyped by the HapMap project [10]. On the other hand, after the permutation test, we observe that the p-value is 0.75, indicating that this partition is probably just an artifact. Since `STRUCTURE` is too slow to perform such a test, our method gives a rigorous alternative to the significance estimators of `STRUCTURE`.

We measured the performance of our method and compared it to `STRUCTURE` on the HapMap populations, as well as on simulated data. We find that our method is at least as accurate as `STRUCTURE`, and an order of magnitude more efficient. Furthermore, we find that the accuracy of `STRUCTURE` degrades when many SNPs are used (thousands), while the accuracy of our method consistently improves when the number of SNPs increases. We have also compared our method to `EIGENSTRAT`, a recent program that corrects for population stratification using the eigenvalues of the genotype covariance matrix [14]. In [14] they suggest a method based on principle component analysis, that assigns each individual a vector representing its ancestral composition. Although their method is not specifically designed for clustering populations, we have adapted their method in a natural way and compared it to the method developed in this paper. We found that `EIGENSTRAT` is quite efficient, but it appears not to perform very well on many of our datasets. We believe that this is due to the fact that the principal component analysis fails when the sub-populations structures are not independent.

Technically, our method is based on a distance defined between pairs of individuals. There are many possible distance measures, and the resulting algorithm is very sensitive to the choice of the distance measure. Surprisingly, one of the most natural measures, i.e., the Hamming distance, performs quite poorly. We therefore use as a starting point the mother-father distance defined in [3]. This measure satisfies the property that the expected distance between two individuals drawn from the same sub-population is zero while the distance between individuals from two different sub-populations is positive. Furthermore, in [3] it is shown that the max-cut induces the correct partitioning asymptotically, at least when the sub-population sizes are equal. Our final distance uses a more complicated procedure which takes into account the genotypes of the whole population in order to determine the distance between a pair of individuals. We show empirically that this procedure is advantageous and that the resulting distance better represents the population structure. This distance measure may be of independent interest, as it may be used in other population based applications.

2 Problem Formulation

We consider the setting in which a set of n individuals are genotyped over m SNPs. The problem of population stratification focuses on the assignment of each of the individuals to a population cluster. In practice, an individual could belong to more than one cluster, (for instance when the individual's ancestors come from two or more different populations). In this paper, however we concentrate on the simpler case, in which each individual is assumed to belong to exactly one population. Furthermore, we assume that the number of populations K is known. We will observe later that this assumption is not too restrictive, as one can test for the validity of the solution. Our goal is to cluster the set of individuals into K clusters, based on their genotype information.

In order to define the problem mathematically, we first introduce a random generative model for the individuals' genotypes. Each genotype is represented by a vector $g \in \{0, 1, 2\}^m$, where g_j represents the minor allele in SNP j, that is, $g_j = 1$ for heterozygous, and it is 0 and 2 for the homozygous major or minor alleles respectively. A population is characterized by the minor allele frequency in each of the SNPs. Thus, a population i is defined by an m-dimensional vector $\boldsymbol{p^i} = (p^i_1, \ldots, p^i_m)$, where p^i_j represents the minor allele frequency of population i in position j. The random generative model assumes that all individuals are sampled independently, and that for each individual g, the different SNP values are sampled independently, where g_j is sampled from the distribution $\{(p^i_j)^2, 2p^i_j(1-p^i_j), (1-p^i_j)^2\}$ (e.g., the probability that $g_j = 1$ is $2p^i_j(1-p^i_j)$). This model has been used by previous approaches, and in particular by `STRUCTURE`. The assumption that the different SNPs are independent can be justified if the SNPs are physically distant from each other (and thus, they are in *linkage equilibrium*). We define the distance between two sub-populations i, i' as:

$$d(i, i') = \sqrt{\sum_j (p^i_j - p^{i'}_j)^2}$$

Formally, we assume that we get as an input an $n \times m$ genotype matrix A, where the rows $R(A)$ denote diploid individuals and the columns $C(A)$ represent SNP sites. Each entry in A is in $\{0, 1, 2\}$. We search for a classification $\theta : R(A) \to \{1, \ldots, K\}$, that assigns every individual to a particular sub-population. Let $\hat{\theta}$ be the correct classification. Our objective is to minimize the number of errors made by the algorithm, that is, we would like to minimize $|\{r \in R(A) \mid \theta(r) \neq \hat{\theta}(r)\}|$.

2.1 The Graph Based Approach

It is convenient to think of the above problem as a clustering problem in a graph. In this case, we construct a complete graph $G = (V, E)$, where vertex set V corresponds to the set of individuals, and edge set E is the set of all pairs of individuals. We assign a distance for each edge, which will intuitively represent the genomic distance between the two individuals. Then, the main idea of the algorithm is to find a max-K-cut in the resulting graph. This makes sense since G captures the fact that the genomic distance between two individuals from the same sub-population is small, while the distance between two individuals from different sub-populations may be large. Clearly, the resulting algorithm is sensitive to the choice of the distance measure.

The most natural distance measure is the Hamming distance, which counts the number of differences between the two vectors. However, we observe that in practice the Hamming distance does not provide very good results[1]. We therefore follow [3], and start from the so called *Mother-Father distance (MF).* The MF-distance satisfies the property that the expected distance between two individuals from the same population group is 0 and the expected distance between two individuals of different populations is positive. Actually, it is not hard to see that the MF-distance is the only pair-wise distance measure that satisfies this property (up to a constant factor).

Formally, for any two individuals r_1, r_2, we define $\delta_j(r_1, r_2)$, the MF-distance at SNP j as follows. We set $\delta_j(r_1, r_2) = -1$ if $r_{1j} = r_{2j} = 1$, $\delta_j(r_1, r_2) = 2$ if $r_{1j} = 0, r_{2j} = 2$ or $r_{1j} = 2, r_{2j} = 0$, and 0 otherwise. We then define the MF-distance $\delta(r_1, r_2)$ to be the sum of the MF-distances over all SNPs. That is, $\delta(r_1, r_2) = \sum_j \delta_j(r_1, r_2)$.

We can now compute the expected distance between two individuals r_1, r_2 from populations i and i' $E[\delta(r_1, r_2)]$:

$$\begin{aligned}
&= \sum_j E[\delta_j(r_1, r_2)] \\
&= \sum_j 2(p_j^i)^2(1 - p_j^{i'})^2 + 2(p_j^{i'})^2(1 - p_j^i)^2 - 2p_j^i(1 - p_j^i)(2p_j^{i'}(1 - p_j^{i'})) \\
&= 2\sum_j (p_j^i(1 - p_j^{i'}) - p_j^{i'}(1 - p_j^i))^2 = 2\sum_j (p_j^i - p_j^{i'})^2 = 2d(i, i')^2
\end{aligned}$$

Consequently, if $i \neq i'$, then the expected MF-distance between two individuals of different populations is positive. On the other hand, if $i = i'$, then $d(i, i') = 0$, and the expected MF-distance between two individuals of the same population is zero. It is further shown in [3], that if the distance between two different sub-populations $d(i, i') >> \sqrt{1.5}(m \log n)^{0.25}$, then with high probability, all the pair-wise distances within a sub-population are at most $d(i, i')^2$, while pair-wise distances across the two sub-populations are at least $d(i, i')^2$. In that case, the

[1] For example, when $p_j^1 = 2/3, p_j^2 = 1$, the expected distance within population 1 is larger than the expected distance across.

max-K-cut algorithm may be reduced to a connected component algorithm. Furthermore, it can be shown that even with much smaller separation of the two populations, the max-cut on the graph with MF-distances produces the correct cut [3].

2.2 Triplets-Based Distance

Even though the MF-distance has some very nice properties, our empirical studies (see Appendix A) show that the max-cut solution obtained from this distance is sometimes biased towards an unbalanced partition. Intuitively, although the expected value of the MF-distance is monotone with the distance between the populations, unbalanced cuts may be chosen by the algorithm by pure chance. It is therefore essential to find a distance measure that has smaller variance than the MF-distance.

We build on top of MF-distances to obtain a more sensitive distance measure, which we call the triplet distance. The main idea of the triplet measure is to utilize information from all genotypes to determine the distance between a pair of individuals.

We will now formally define the triplet distance for a pair of individuals r_1 and r_2. The triplet distance depends on two parameters a, b that will be fixed later. For every third individual in the population, r, we consider the unordered set $\{r_1, r_2, r\}$, which we refer to as a *triplet*. For each such triplet, we define two indicator variables X_r and Y_r such that $X_r = 1$ if $\delta(r_1, r_2) \geq \max(\delta(r_1, r), \delta(r_2, r))$, and $X_r = 0$ otherwise. Similarly, $Y_r = 1$ if $\delta(r_1, r_2) \leq \min(\delta(r_1, r), \delta(r_2, r))$, and it is zero otherwise. We define the triplet-based distance as $d_{a,b}(r_1, r_2) = \sum_r (aX_r + bY_r)$. In other words, to compute the triplet distance of r_1 and r_2, we consider every third individual r and if $\delta(r_1, r_2)$ is the largest among the three MF-distances, then we add a and if it is the smallest, then we add b.

We now find the expected triplet distance $d_{a,b}(r_1, r_2)$ for a pair of individuals r_1, r_2. We will implicitly assume that all MF-distances are different (this is true if the number of SNPs is sufficiently large). For a triplet (r_1, r_2, r), we consider the following two cases. First, assume that all three individuals are from the same population. Then by symmetry, $\Pr(X_r = 1) = \Pr(Y_r = 1) = \frac{1}{3}$. Otherwise, if r_1, r_2 are from population i, and r is from another sub-population i', we will bound the probability $\Pr[\delta(r_1, r_2) \geq \delta(r_1, r)]$. Intuitively, this probability should be small if the distance $d(i, i')$ is large enough. Formally, we know that

$$E[\delta(r_1, r_2) - \delta(r_1, r)] = -2d(i, i')^2.$$

Furthermore, $\delta(r_1, r_2) - \delta(r_1, r)$ is the sum of m random variables that lie in the interval $[-2, 2]$. This is because, if $\delta(r_1, r_2) = -1$ then $\delta(r_1, r) \neq 2$ and vice-versa. Therefore, we could use the following tail bound, known as Hoeffding bound[9]:

Theorem 1. *Let $X_1, \ldots, X_n$ be n independent random variables, and let a, b be such that for every i, $a \leq X_i \leq b$. Denote $X = X_1 + \ldots + X_n$. Then,*

$$\Pr(X - E[X] > \alpha) \leq \exp\left(\frac{-2\alpha^2}{n(b-a)^2}\right).$$

Thus, using the Hoeffding bound, we get $\Pr[\delta(r_1, r_2) - \delta(r_1, r) > 0]$

$$= \Pr[\delta(r_1, r_2) - \delta(r_1, r) + 2d(i, i')^2 > 2d(i, i')^2] \leq \exp\left(-\frac{d(i, i')^4}{2m}\right)$$

If $d(i, i') = (6tm \log n)^{0.25}$ for $t > 1$, we get by the union bound that with very high probability all triplets satisfy the property that edge distance within a sub-population is lesser than any edge distance across two populations. The probability that this event does not happen is smaller than $\frac{1}{n^{3t-2}}$. We now use these observations to compute the expected triplet distances. Assume that r_1, r_2 are from sub-population i, and that P_i is the frequency (prior) of this sub-population in the entire population. Then,

$$\begin{aligned} E[d_{a,b}(r_1, r_2)] &= E[a \sum_r X_r + b \sum_r Y_r] \\ &\leq n\left(a\frac{P_i}{3} + b(1 - \frac{2P_i}{3})\right) + \frac{|a| + |b|}{n^{3t-2}} \\ &\approx n\left(a\frac{P_i}{3} + b(1 - \frac{2P_i}{3})\right) \end{aligned}$$

Similarly, for r_1, r_2 from different sub-populations i, i', it is easy to see that

$$E[d_{a,b}(r_1, r_2)] \geq \tfrac{an}{2} - \tfrac{|a|+|b|}{n^{3t-2}} \approx \tfrac{an}{2}$$

If we know the frequency of the sub-populations in the entire population, then we can take $P = \max_i P_i$. For instance, if we set $a = (2/P) - 2, b = -1$ we get that the expected distance between individuals from two different populations is positive, while the expected distance between individuals of the same population is non-positive. For a balanced cut, selecting $a = 4, b = -1$, gives positive expected distance between individuals of different populations and zero otherwise. In practice, even though we do not know the correct value of P, we try different values of P to determine a, b, each giving different partitions. We then pick the partition with the largest likelihood score, where the likelihood score is similar to the one used for `STRUCTURE`, as we now describe.

Recall that A is the input genotype matrix with $R(A)$ being the genotypes of the n input individuals, $\theta : R(A) \mapsto \{1, \ldots, K\}$ is the classification of individuals to sub-populations and $\boldsymbol{p^i}$ is an m-dimensional vector of the MAF of sub-population i. Given θ, the maximum likelihood estimate of $\boldsymbol{p_i}$ is obtained by simply counting the allele frequencies in each of the sub-populations defined by the partition. The posterior probability is given by

$$\Pr[\theta, \boldsymbol{p^i} | A] \propto \Pr[\theta] \Pr[\boldsymbol{p^i}] \Pr[A | \theta, \boldsymbol{p^i}]$$

We set the priors for θ and $\boldsymbol{p^i}$ to be fixed and uniform, and thus maximizing the posterior is equivalent to maximizing the likelihood $\mathcal{L}(A | \theta, \boldsymbol{p^i}) = \Pr[A | \theta, \boldsymbol{p^i}]$.

Algorithm (`GRAPH-TRIPLETS`*).* We can now describe the whole algorithm. The algorithm begins by computing the MF-distance for each pair of individuals. Then, for every pair of individuals r_1, r_2, we compute $X(r_1, r_2) = \sum_r X_r$, and $Y(r_1, r_2) = \sum_r Y_r$. The algorithm then proceeds in iterations. In each iteration we pick a value for P, and we search for a partition that maximizes the likelihood score, based on the prior information that one of the sub-populations is of size P. We take values of P ranging from 0.5 through 0.9 in 0.1 increments. Each such value determines the values of a and b. The triplet distances are then computed for each pair of individuals, by setting $d_{a,b}(r_1, r_2) = ((2/P) - 2)X(r_1, r_2) - Y(r_1, r_2)$. These distances induce a complete graph $G = (V, E)$, where the vertices represent individuals and the edges are weighted by the triplet distances. We are then interested in finding the maximum K-cut.

Unfortunately, finding the max-K-cut of the graph is an NP-hard problem even when $K = 2$ [7]. We therefore use the Kernighan-Lin heuristic [9], which is a hill-climbing method to find the optimal cut. The algorithm for the case when $K = 2$ is presented in Figure 1. The algorithm randomly partitions the vertices $V(G)$ into two disjoint sets V_1 and V_2. The algorithm then proceeds in *rounds* each of which involves performing $|V(G)|$ *iterations*. At each iteration we move a vertex u from one side of the cut to the other. The vertex u is chosen so that the resulting cut is maximized. Unlike standard local search techniques, the algorithm swaps u even if this results in the reduction of the cut-size. Once a vertex u is swapped, it cannot be swapped again until the next round. At the end of a round all vertices have been swapped, and the best partition in that round is chosen for the next round. We repeat until the cuts in the beginning and the end of a round are identical, and thus no improvement can be achieved. In Figure 1, set $V_x(V_{x'})$ denotes the vertex set of V_1, V_2 that currently contains x (does not contain x) and $\text{cut}(V_1, V_2)$ denotes the cost of the cut.

Using Kernighan-Lin, for each setting of a, b we find a max-K-cut and we select the one that maximizes $\mathcal{L}(A|\theta, \boldsymbol{p^i})$. Finally, we perform a greedy local search by moving a vertex from one side to another, if it improves the likelihood. This final step, typically improves the accuracy by a little in practice.

3 Results

To evaluate the performance of our method, we compared `GRAPH-TRIPLETS` to two state of the art methods that deal with population stratification, namely `STRUCTURE` and `EIGENSTRAT`, which are described below.

`STRUCTURE` [13] is a well established package that uses Markov Chain Monte Carlo method (MCMC) to heuristically maximize the posterior probability. Structure can be seeded with a number of different parameters. We used $K = 2$, to denote that the program should look for two sub-populations. By default, the program assumes that the allele frequencies of the two populations are independent. For closely related sub-populations, however, the software allows for a mode in which the frequencies are assumed to be correlated. We ran the program on both modes, with the parameter turned off and on. The default number of *MCMC* and

```
kernighanLin(graph $G$)

1. randomly partition $V(G)$ into $V_1, V_2$
2. $\alpha \leftarrow 1$
3. while $\alpha = 1$ do    (* rounds *)
   (a) $\alpha \leftarrow 0, \chi \leftarrow V_1 \cup V_2$
   (b) while $|\chi| > 0$ do    (* iterations *)
        i. $u \leftarrow \text{argmax}_{x \in \chi} \text{cut}(V_x \setminus \{x\}, V_{x'} \cup \{x\})$
       ii. $\chi \leftarrow \chi \setminus \{u\}$
      iii. if $u \in V_1$ then $V_1 \leftarrow V_1 \setminus \{u\}, V_2 \leftarrow V_2 \cup \{u\}$
       iv. else $V_1 \leftarrow V_1 \cup \{u\}, V_2 \leftarrow V_2 \setminus \{u\}$
        v. if $\text{cut}(V_1, V_2) > \text{cut}(V_1^*, V_2^*)$ then $V_1^* \leftarrow V_1, V_2^* \leftarrow V_2$,
           $\alpha \leftarrow 1$
   (c) $V_1 \leftarrow V_1^*, V_2 \leftarrow V_2^*$
```

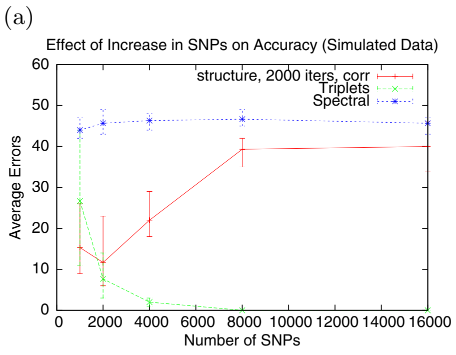
Fig. 1. Kernighan-Lin heuristic to find max-cut of graph G

Burnin iterations is 2000 each. We varied this number to analyze the trade-off between run-time and accuracy. We used the default values for the rest of the parameters.

We note that STRUCTURE is a software that does much more than just clustering individuals. Among other things, it can cope with admixed populations, and it can incorporate linkage disequilibrium into its model. We have not compared our method to these modes of STRUCTURE, as it is beyond the scope of this paper, and our algorithm is not optimized for such tasks at this point.

EIGENSTRAT [14] is a relatively new software tool, which corrects population sub-structure by the spectral properties of the covariance genotype matrix. In a nutshell, EIGENSTRAT takes A an $m \times n$ input genotype matrix, where rows are SNPs and columns are individuals and normalizes each entry of A by subtracting the row mean (minor allele frequency of the SNP) and dividing by the row's standard deviation. It then takes the largest eigenvectors of the covariance $n \times n$ matrix Ψ, and uses those to correct for population sub-structure. Even though EIGENSTRAT is not explicitly described as a genetic clustering method, we adapt their algorithm in a natural way, resulting in a clustering algorithm in which the clusters are determined by using the sign of the entries of the highest eigenvector of Ψ. We implemented this clustering algorithm in MatLab and compared it to our method. We refer to our implementation as `Spectral` in the results presented.

Datasets. For the evaluation, we used datasets from two different sources. First, we used simulated data generated using the following model. Each sub-population i is represented by an m-dimensional vector of allele frequencies $\boldsymbol{p^i}$, and an individual of the population is sampled by randomly and independently picking allele counts according to the allele frequency distributions of the sub-population. For simulations, we assumed that all SNPs within a sub-population had the same allele frequency, i.e., for any $i, p^i_j = p^i_{j'}$.



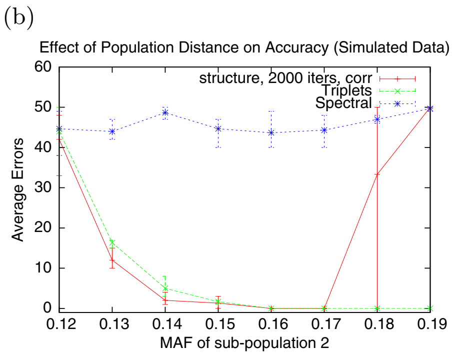


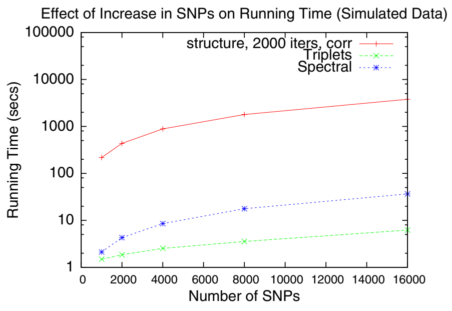
Fig. 2. Comparison of accuracy on simulated data. `GRAPH-TRIPLETS` is consistent in its accuracy and converges to the correct partition with increase in SNPs or increase in distance. On (a) MAF of sub-populations 1 and 2 were 0.1 and 0.13 respectively. On (b) the number of SNPs was fixed to 1000 and MAF of sub-population 1 was fixed at 0.1. We used an average of three randomly generated data sets to obtain every point. Error bars indicate highest and lowest values obtained.

We have also used the publicly available data from the International HapMap consortium [10]. This data-set consists of four population groups: Utah residents with ancestry from northern and western Europe(CEU), Yoruba in Ibadan, Nigeria (YRI), Han Chinese in Beijing, China (CHB) and Japanese in Tokyo, Japan (JPT) with 90, 90, 45 and 44 individuals respectively. The Central Europeans and Yoruba Africans consisted of thirty trios each, and therefore in order to avoid these dependencies, we used the 60 parents from each of the two populations, ignoring the 30 children. To obtain a test set where the SNPs are independent, we sampled m SNPs uniformly at random from chromosome 10. We evaluated the programs on each of the six pairs of populations, with different numbers of SNPs, ranging from 1000 to 8000.

Evaluation Measures. There are many possible ways to evaluate the performance of the algorithms. We chose to let each of the program separate the genotypes of two populations (say Africans and Chinese in the HapMap data) into two clusters, and the error rate of such an experiment would be the number of individuals misclassified (for instance, the number of Africans classified as Chinese). We have also compared the running-time of the methods. In summary, our experiments show that the graph-based method is significantly faster while being at least as accurate as existing methods.

Simulated Data. On simulated data, we studied closely related populations. We fixed the minor allele frequency of one population to be 0.1 for each of its SNPs, while for the other population the minor allele frequency varied from 0.12 to 0.19. On all the data presented, `STRUCTURE` running on default parameters fails to find two sub-populations, i.e., it returns solution with all individuals in one cluster. We therefore only report `STRUCTURE` running for 2000 iterations with the correlation

(a)



(b)

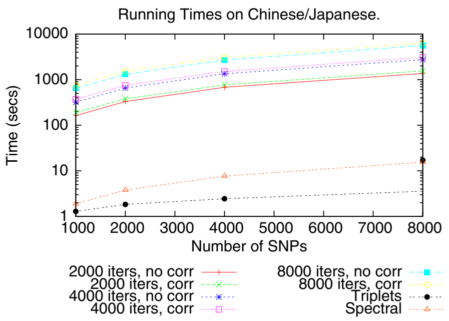
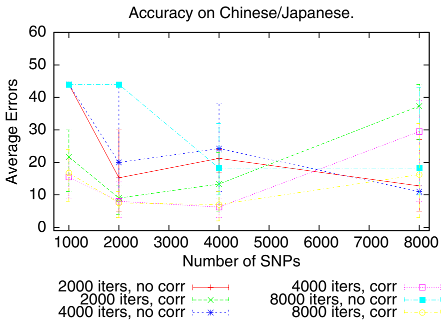


Fig. 3. Comparison of run-times on simulated and real data. GRAPH-TRIPLETS is hundreds of times faster than STRUCTURE.

(a)



(b)

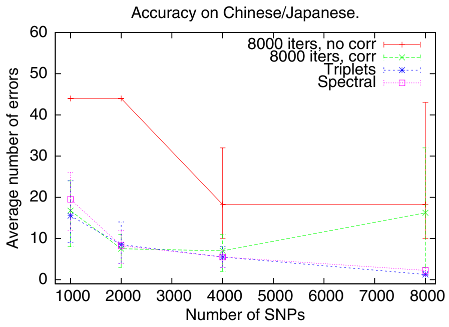


Fig. 4. Comparison of accuracy on HapMap Chinese/Japanese. GRAPH-TRIPLETS is accurate and converges to the correct partition with increase in SNPs. It is unclear what parameters of STRUCTURE to use. We used the average of four randomly drawn data sets to obtain every point. Error bars indicate the highest and lowest values obtained.

mode turned on. Figure 3(a) shows that GRAPH-TRIPLETS is an order of magnitude faster than STRUCTURE. Figure 2 shows that it makes substantially less errors overall than STRUCTURE and the spectral clustering which is based on EIGENSTRAT. It is conceivable that STRUCTURE would find better solutions if it uses even more time, although this seems rather prohibitive. We note that the performance of STRUCTURE seems not to be monotonic with the number of SNPs used.

HapMap. For the HapMap datasets, we considered all six pairs of populations. For most pairs, all methods worked perfectly (no errors were made) with as few as 200 SNPs. The only hard instance was the Chinese-Japanese pair. For this pair, none of the methods could give a perfect clustering prediction even when 8000 SNPs were used. As can be seen from Figure 4, the spectral method and

GRAPH-TRIPLETS seem to produce lesser errors than the classification returned by STRUCTURE. Furthermore, we used the two different modes of STRUCTURE (with or without correlations), and the results were inconclusive regarding which one works better on this dataset, as when small number of SNPs were used (1000, 2000 or 4000), the correlation mode seemed to perform better. When more SNPs were used (8000 SNPs, 2000 or 4000 iters), the 'no correlation' mode of STRUCTURE was better.

As before, Figure 3(b) demonstrates that GRAPH-TRIPLETS is much more efficient than STRUCTURE. Specifically, the run time for GRAPH-TRIPLETS $\approx$ 300 times faster for 1000 SNPs and $\approx$ 1000 times faster for 8000 SNPs.

4 Significance of Clusters

An obvious and important question to consider is whether the clusters obtained from the methods are significant. In practice, we could run STRUCTUREor GRAPH-TRIPLETS with K, the number of sub-populations set to 2. When the software returns a solution, with individuals divided into two populations, there is no guarantee on whether the input set of taxa actually even contains two sub-populations. To test the significance of the clusters, one could perform the statistical tests described by Pritchard et al. [13], which as the authors themselves point out either uses dubious assumptions or does not work for large number of SNPs in practice. Alternatively, we could examine the change in the likelihood function or use information based evaluation such as minimum description length to decide on whether there is truly two sub-populations or not.

A simple and direct approach is to permute the alleles in each site independently such that the input no longer has sub-structure. We can then re-run the algorithm on the new input. The p-value is simply the fraction of times, the permuted input had solution larger (or more likely) than that of the original input.

However, such tests can only be performed if the algorithm to find clusters is very efficient. Here, we simply considered a randomly drawn data set with 8000 SNPs from each of the HapMap populations. We ran structure (default parameters) and triplets with $K = 2$. We report the number of errors made (size of the smaller set) along with the p-value which can be efficiently computed using the triplets method. We believe that the computation of this p-value is a great benefit in practice that our new technique can offer. The results for 1000 permutations is presented in Table 1.

Table 1. Triplets can be used to compute p-values directly

	Errors		
	structure	Triplets	Triplets p-value
Central Europeans, CEU	21	2	0.01
Yoruba Africans, YRI	26	20	0.75
Chinese, CHB	17	16	0.267
Japanese, JPT	18	12	0.929

5 Conclusion

The problem of population stratification is an increasing concern in the context of disease association studies. In particular, its influence on whole genome association studies (for e.g. [8,15]) are severe. Even though existing methods for clustering individuals based on their SNPs provide relatively accurate predictions, there is no rigorous theory that ensures the convergence of these methods to the correct solution. Furthermore, there is no study that compares these methods on a variety of datasets, both real and simulated. Our paper has been motivated by these two concerns.

In this paper, we suggest a graph based method for detecting population stratification. The distance measure used in our method builds on the Mother-Father distance that was suggested in [3], where it has been rigorously and analytically shown that if the sample size is large enough, the measure will represent the correct distance between individuals, and therefore our algorithms will converge to the true population clusters. We believe that this theoretical foundation for our algorithm is an important advantage that proves itself in practice. In particular, we show that our algorithm is consistently at least as accurate as `STRUCTURE` and `EIGENSTRAT`.

One of the questions we raised in this paper is the validity of the population substructure found by the clustering algorithm. We have demonstrated that the different methods will tend to cluster the individuals in two clusters, even if in reality there is only one population. In order to assess the significance of these partitions, we suggest a permutation test, which seems to give the correct significance scores on the HapMap populations. This permutation test can only be carried out if the clustering methods are highly efficient. As our method runs in seconds over thousands of SNPs and hundreds of individuals, this was feasible.

We note that our paper focuses on the clustering of two populations while `STRUCTURE` and `EIGENSTRAT` can do much more than that. In particular, they can deal with admixed populations, correlated populations, and linkage disequilibrium. We hope that a combination of the existing methods such as `STRUCTURE` and `EIGENSTRAT`, together with our graph based approach may lead to improved tools for these cases as well.

Acknowledgments

EH and SS were supported by NSF grant IIS-0513599. SS was also partially supported by NSF grant IIS-0612099.

References

1. R. Boppana. Eigenvalues and Graph Bisection: An Average Case Analysis. In proc *IEEE Symposium on Foundations of Computer Science*, 1987.
2. W. Buntine and A. Jakulin. Applying Discrete PCA in Data Analysis. In proc *Uncertainty in AI*, 2004.

3. K. Chaudhuri, E. Halperin, S. Rao and S. Zhou. Separating Populations with Small Data. To appear in proc *Symposium on Discrete Algorithms (SODA)* 2007.
4. D. F. Conrad, T. D. Andrews, N. P. Carter, M. E. Hurles and J. K. Pritchard. A high-resolution survey of deletion polymorphism in the human genome. *Nature Genetics*, 38, 2006.
5. M. J. Daly, J. D. Rioux, S. F. Schaffner, and T.J. Hudson. High-resolution haplotype structure in the human genome. *Nature Genetics*, 29, 2001.
6. S. Gabriel, S. Schaffner, H. Nguyen, J. Moore, J. Roy, B. Blumenstiel, J. Higgens, M. DeFelice, A. Lochner, M. Faggart, S.N. Liu-Cordero, C. Rotimi, A. Adeyemo, R. Cooper, R. Ward, E.S. Lander, M.J. Daly, and D. Altschuler. The Structure of Haplotype Blocks in the Human Genome. *Science*, 296, 2002.
7. M. R. Gary and D. S. Johnson. Computers and Intractability. *Freeman*, 1979.
8. J. N. Hirschhorn and M. J. Daly. Genome-wide Association Studies for Common Diseases and Complex Traits. *Nature Review Genetics*, 6, 2005.
9. W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association(JSTOR)*, 58, 1963.
10. The International HapMap Consortium. A Haplotype Map of the Human Genome. *Nature*, 437, 2005.
11. B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 1970.
12. F. McSherry. Spectral Partitioning of Random Graphs. *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2001.
13. J. K. Pritchard, Matthew Stephens and Peter Donnelly. Inference of Population Structure Using Multilocus Genotype Data. *Genetics*, 155, 2000.
14. A. L. Price, N. J. Patterson, R. M. Plenge, M. E. Weinblatt, N. A. Shadick and D. Reich. Principal components analysis corrects for stratification in genome-wide association studies. *Nature Genetics*, 38, 2006.
15. D. C. Thomas et al. Recent Developments in Genome wide Association Scans: a Workshop Summary and Review. *American Journal of Human Genetics*, 77, 2005.

A Appendix: Empirical Comparison of MF with Triplets

To motivate our choice of triplet based distance instead of MF-distance, we show empirically that the triplet distance gives a better separation of the two populations into two clusters. In order to do so, we randomly generate two subpopulations according to the following model. We generate the two populations, such that one population has minor allele p_1 for all the SNPs, and the other population has minor allele p_2 for all the SNPs. Furthermore, each of these subpopulations has the same number of individuals. We measure the cut distance of the correct cut using the MF-distance and the triplet distance. Let d_{mf} and d_t be the correct cut weight for mother-father and triplets. We then find $10,000$ random balanced cuts of the graph and measure the cut weights. We then measure the fraction of times the random cut cost was larger than d_{mf} or d_t respectively. We call this measure the *balanced p-value*. We also measured the max-cut for $10,000$ randomly generated unbalanced cuts, where 0.25-fraction of the vertices were on one side. We call this measure the *unbalanced p-value*.

The results of the various simulation tests are shown in Figure 5. Figure 5(a), shows that the balanced p-values are similar for triplets based distance and for

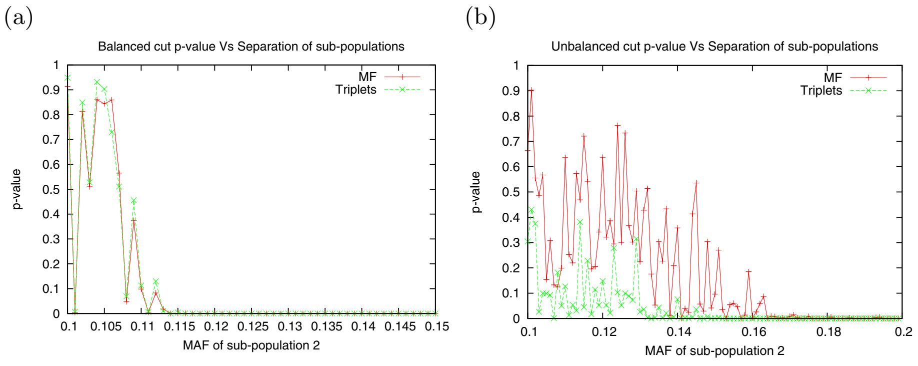


Fig. 5. Comparison of MF and Triplet distance measures. Figure (a) shows that when only balanced cuts are considered, the distance measures provide comparable p-values. Figure (b) shows that more often unbalanced cuts in the MF distance contains cost larger than that of the correct cut. In both figures MAF of sub-population 1 was fixed to 0.1. For Figure (a) we used 100 individuals in each population and for Figure (b) we used 10 individuals in each population. We used parameters $a = 2, b = -1$ for the triplets.

the MF-distance, and that the balanced p-value of both methods quickly go down to zero. More importantly, Figure 5(b) shows that triplets clearly has a lower p-value in the case of unbalanced cuts. While MF-distance contains several unbalanced cuts of high weight, on the triplets, the correct cut has cost typically higher than all other cuts. This provides an evidence that the triplet distance is advantageous.

RB-Finder: An Improved Distance-Based Sliding Window Method to Detect Recombination Breakpoints

Wah-Heng Lee[1,2] and Wing-Kin Sung[1,2]

[1] Genome Institute of Singapore
[2] National University of Singapore
leewhc@gis.a-star.edu.sg, ksung@comp.nus.edu.sg

Abstract. Recombination detection is important before inferring phylogenetic relationships. This will eventually lead to a better understanding of pathogen evolution, more accurate genotyping and advancements in vaccine development. In this paper, we introduce RB-Finder, a fast and accurate distance-based window method to detect recombination in a multiple sequence alignment. Our method introduces a more informative distance measure and a novel weighting strategy to reduce the window size sensitivity problem and hence improve the accuracy of breakpoint detection. Furthermore, our method is faster than existing phylogeny-based methods since we do not need to construct and compare complex phylogenetic trees. When compared with the current best method Pruned-PDM, we are about a few hundred times more efficient. Experimental evaluation of RB-Finder using synthetic and biological datasets showed that our method is more accurate than existing phylogeny-based methods. We also show how our method has potential use in other related applications such as genotyping.

Keywords: Recombination detection, sliding window, phylogeny, genotyping.

1 Introduction

Recombination is the exchange of genetic material between two genomes. It violates the assumption in molecular phylogenetics that there is only one evolutionary history in a sequence data set. This exchange of DNA subsequences may cause a change in phylogenetic relationships in the affected region. Conflicting phylogenetic information from different regions of the alignment will eventually lead to the reconstruction of some lower quality phylogenetic tree [1]. Thus, it is important to detect recombination before inferring phylogenetic relationships.

A breakpoint is defined as the location where a recombination event occurs in a sequence. Generally, the detection of breakpoints depends on the strength of a recombination event, which is affected by factors such as the mutation rate and the time at which the recombination event took place [2]. The unpredictable conditions at which recombination events occur make the task of finding breakpoints difficult. For recombination events which make little changes to the sequence, the detection of breakpoints may even be impossible [3]. This underlines the importance of developing sensitive and accurate methods for detecting recombination.

There are many approaches to detect recombination within alignments. See Posada [1] for a comparative study on the performances of 14 different recombination detection methods. We focus on a particular class of methods which uses a sliding window to detect recombination. Sliding window approaches are preferred over those that use a global reference tree because they can localize breakpoints more accurately and thus detect weak recombination events in the presence of strong recombination events [4]. Given a length-N alignment of a set of sequences, window-based methods make use of a sliding window to detect recombination. A length-w 'sliding window' is defined as a window enclosing the alignments from positions i to i+w. Suppose b is a breakpoint, we would expect the immediate left and right neighboring windows [b-w…b-1], [b…b+w] of b to enclose alignments that are significantly different. Hence, to find the exact positions of the breakpoints (if any), the sliding window approach performs an exhaustive search by sliding a length-w window across the alignment and for every possible position i ($w \leq i \leq N - w$), it compares the alignments enclosed by neighboring windows [i-w…i-1] and [i…i+w]. Typically, recombination is detected by comparing and noting significant differences in distance measures (distance-based) or in phylogenetic trees' topologies (phylogeny-based) computed for the alignments in [i-w…i-1] and [i…i+w].

Distance-based window methods such as PhyPro [2] and DSS [5] are fast and fairly accurate. PhyPro computes, for every sliding window, a test statistic known as the minimum distance vector correlation using only non-conserved sites of an alignment. Then, they estimated the p-value for the null hypothesis of no recombination by permuting the alignment 1000 times and counting the number of times the original minimum distance vector correlation was smaller than the minimum distance vector correlation for each permutation. In DSS, a length-500 window is slid along an alignment of DNA sequences. For each window, two distance matrices (one for each half of the window) are calculated according to some Markov model of nucleotide substitution. A sum-of-squares statistic is then computed for each distance matrix. Since each distance matrix encodes the phylogenetic relationships in its corresponding window, the presence of recombination will result in a big difference in the two sum-of-squares statistics for the two matrices. Unfortunately, distance-based methods tend to suffer from information loss when estimating recombination. Phylogenetic information is lost when only pair-wise distance comparisons are made using conventional distance metrics that measures only the global homology between two sequences.

On the other hand, phylogeny-based window methods such as PDM [6], Pruned-PDM [7] and RECOMP [8] try to generate the most likely phylogenetic trees for the alignments enclosed in neighboring windows and compare them to estimate recombination more accurately. The PDM and Pruned-PDM method focused on estimating the topology changes based on a likelihood score. To reduce the uncertainty of tree estimation from short sequence alignments enclosed by the sliding window, they used a distribution of trees instead of a single tree as reference. Although accurate, their methods are very slow due to the need for Markov chain Monte Carlo simulations. RECOMP was then developed to provide a faster means of detecting recombination. Using a sliding window, a set of trees is generated for each window based on a maximum parsimony heuristic. Recombination is determined by comparing four different measures such as the Robinson-Foulds distance of sets of trees in adjacent windows. An accuracy comparable to Pruned-PDM is claimed but the interpretation of the four measures as an indication of recombination is sometimes not straight-forward and may even be ambiguous at times.

Regardless of distance-based or phylogeny-based methods, a sliding window approach has a major concern: the selection of window length. In previous works, the window length is usually arbitrary chosen within the range of 200-500. However, window length affects the sensitivity and accuracy of window-based methods to detect recombination. Recent works have shown that their results are most accurate when the given window length is approximately the recombinant subsequence length [5, 7]. If the length of the recombinant is not known in advance, an algorithm using different window lengths may produce vastly different analysis results on the same dataset. Furthermore, there may be problems in detecting recombinant regions shorter than the given window length due to the noise caused by the original sequence on either side of the recombinant subsequence included in the window (Fig. 1(c)(d)).

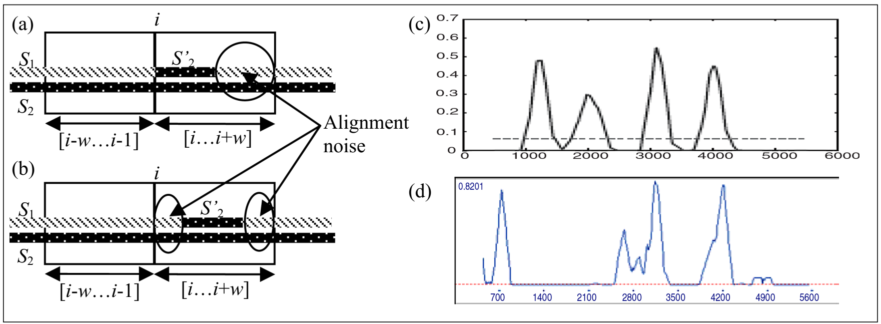


Fig. 1. Window length sensitivity problem when window length *w* is longer than the recombinant subsequence (S'_2) length. (*a*) and (*b*) show the introduction of alignment noise into the computation of any distance measure or phylogenetic tree of the alignment in [*i*…*i*+*w*]. (*c*) shows the result of running Pruned-PDM on a synthetic dataset SD3 with breakpoints at site 1000, 2000, 3000 and 4000 using a length-500 window [7]. (*d*) shows the result of running Pruned-PDM on the same dataset using a length-600 window. Here, the breakpoints are inaccurately detected at sites 700, 2600, 3200 and 4200.

This paper presents RB-Finder, a distance-based window method for finding recombination breakpoints. By avoiding the computationally expensive and complicated comparisons of phylogenetic trees of phylogeny-based methods, our algorithm is faster and thus more scalable to analyze big datasets. Moreover, we minimize information loss experienced by conventional distance-based methods by introducing a new distance metric that takes into consideration important details such as the distribution of mismatches, number of consecutive matches and the locations of common subsequences in an alignment. To improve the accuracy when using non-optimal window lengths, we propose using a weighting strategy that assigns different weights to positions enclosed in a window with respect to a putative breakpoint. The idea is to lessen the contribution of less important regions of a window when computing a distance measure for the alignment. Subsequently, we applied our weighting strategy to our new distance metric and describe a fast, simple and intuitive algorithm to detect recombination. Experiment results using simulated and real datasets show that the accuracy of RB-Finder is better than that of most existing methods. In addition, we present an application of RB-Finder in genotyping by analyzing a set of 13 HIV recombinant sequences. In our analysis, we detected almost all reported breakpoints of the 13 sequences and made several novel findings regarding their genotypes. Specifically, we found irregularities in the genotyping of 6 sequences which may trigger new considerations when assigning genotypes. Refer to http://www.comp.nus.edu.sg/~bioinfo/RBFinder/Supp.htm for supplementary.

2 Method

There are two sources of inaccuracy when using distance-based window methods to detect recombination. Firstly, the use of conventional distance metrics, such as hamming-distance and edit-distance that measure overall homology results in phylogenetic information loss. Secondly, recombination detection is too sensitive on the choice of window length. To improve the information content of conventional distance measures and to reduce the impact of different window lengths on recombination detection, we employed three techniques on which our proposed recombination detection algorithm RB-Finder is based on:

1. Instead of using base-by-base comparison, the similarity of an alignment is computed using the number of shared (k,m)-mers, that is, length-k alignments with at most m mismatches. This measure takes into account the different mutation rates along the alignment by varying its mismatch threshold m automatically. This avoids the effect of random point mutations which causes inaccuracy in distance measures using base-by-base comparison.
2. Given a window instance, we use a weighting strategy that assigns heavier weights to positions nearer the putative breakpoint and lighter weights to positions further away from the putative breakpoint. This reduces the effect of alignment noise as seen in Fig. 1(a)(b) when computing the similarity score.
3. Use contiguous chains of (k,m)-mers to form Contigs. Contigs have even distribution of mismatches and thus, are better estimations of long common subsequences in an alignment.

We describe the above techniques in detail in Sections 2.1, 2.2 and 2.3. Using the three techniques, we describe a recombination detection score to indicate the presence of breakpoints and formally present the algorithm RB-Finder in Section 2.4.

2.1 Using (k,m)-Mers as the Basic Unit of Similarity Measurement

Conventional distance measures such as hamming-distance and edit-distance that perform base-by-base comparisons are susceptible to noise caused by random mutations. Furthermore, they compute only overall sequence homology and omit important details about the distribution of mismatches and the distribution of contiguous matches that may provide further indication of recombination. A possible solution is to use shared (k,m)-mers as the basic unit of similarity measurement in place of base-by-base comparisons.

Definition 1. Let A be a length-n alignment of two sequences S_1, S_2. Let $A[x\ldots x+k\text{-}1]$ be a length-k sub-alignment from position x to position $(x + k\text{-}1)$ of A. $A[x\ldots x+k\text{-}1]$ is a shared (k,m)-mer iff $A[x\ldots x+k\text{-}1]$ has less than m mismatches. This is shown in Fig. 2.

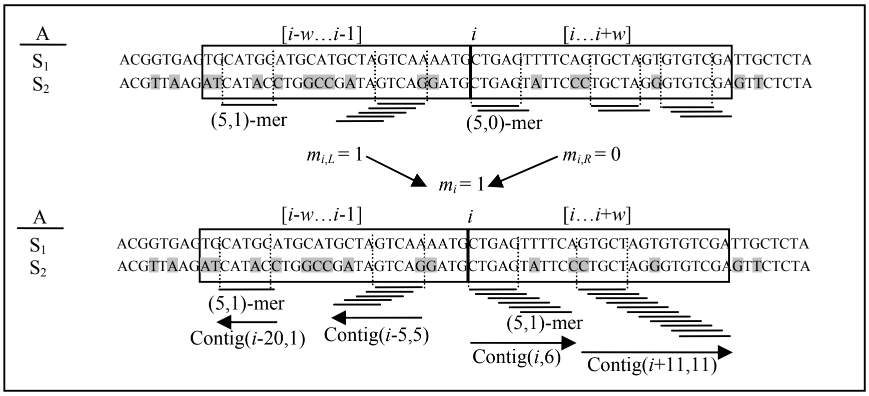


Fig. 2. A diagram showing how shared (k, m)-mers are defined in each window of a putative breakpoint i of an alignment A of two sequences S_1, S_2. *Top*: Shared (k, m)-mers for $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i+w]$ using original mismatch thresholds $m_{i,L} = 1$ and $m_{i,R} = 0$ respectively. *Bottom*: After normalization ($m_i = 1$), the number of shared (5,1)-mers in each window reflects more accurately the relative sequence homology. Hence, $kmd_{i,L} = 6$ and $kmd_{i,R} = 16$. Note that consecutive shared (k, m)-mers form a Contig which we will elaborate in Section 2.2.

Essentially, by counting the number of shared (k,m)-mers, we can identify homologous regions of two sequences with an underlying rate of random mutation. Here, the selection of values for k and m is vital as k determines the specificity of homologous regions found and m estimates the underlying random mutation rate. Note that k should be small but at least of length $(\log_2 n + 1)$ to ensure specificity. Clearly, the selection of a global value for m is not feasible because the underlying rate of random mutation varies across the alignment. Hence, a localized value of m must be chosen for each sub-alignment enclosed by a window instance. We describe a heuristic to automatically determine m for a window instance: Given a length-w window $[i\ldots i+w]$ with parameters k and m, let K_m and K'_m be the number of shared (k,m)-*mer*s and the number of non-shared (k,m)-mers respectively (note that $K_m + K'_m = w - k + 1$). Starting with $m = 0$, we iteratively increase m until $K_m > K'_m$. Denote $m_i = \min\{m \mid K_m > K'_m\}$. At this point of time, a majority of (k,m)-mers in the window are shared (k,m)-mers having less than m_i random mutations. This becomes a reasonable estimate of the underlying random mutation rate of the alignment enclosed by $[i\ldots i+w]$. Hence, we use m_i as the mismatch threshold in $[i\ldots i+w]$.

To detect recombination in an alignment of two sequences S_1 and S_2, we compute and compare a similarity score based on the shared (k,m)-mers of the two neighboring windows $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i+w]$ of a putative breakpoint i. Let $m_{i,L}$ and $m_{i,R}$ be the mismatch thresholds of [*i*-*w*…*i*-1] and [*i*…*i*+*w*] respectively. Clearly, we cannot compare any similarity score of [*i*-*w*…*i*-1] and [*i*…*i*+*w*] when $m_{i,L} \neq m_{i,R}$. Thus, we need to normalize $m_{i,L}$ and $m_{i,R}$ so that shared (*k,m*)-mers in [*i*-*w*…*i*-1] and [*i*…*i*+*w*] are defined based on a single mismatch threshold under the assumption of a common random mutation rate. Note that if *i* is a breakpoint or has high mutation rates, then the sequence homologies in the alignment enclosed by one window would be quite different from that enclosed in the other window (ie $m_{i,L} \neq m_{i,R}$). We exploit this observation and use the normalized mismatch threshold at *i*, $m_i = \max(m_{i,L}, m_{i,R})$ to define shared (*k,m*)-mers in both [*i*-*w*…*i*-1] and [*i*…*i*+*w*] (Fig. 2). In this way, the window with the lower mismatch threshold will have many more shared (*k,m*)-mers than the other window with the higher mismatch threshold. Thus, any irregularities at *i* such as recombination and high mutation rates would be shown as a huge discrepancy in the number of shared (*k,m*)-mers in [*i*-*w*…*i*-1] and [*i*…*i*+*w*]. Formally, we denote the number of shared (*k,m*)-mers between S_1 and S_2 in each neighboring window of a putative breakpoint *i* as the km-distance (*kmd*):

$$\text{For } [i\text{-}w...i\text{-}1],\ kmd_{i,L}(S_1, S_2) = |\, P_{i,L} \,| \ . \tag{1}$$

$$\text{For } [i...i\text{+}w],\ kmd_{i,R}(S_1, S_2) = |\, P_{i,R} \,| \ . \tag{2}$$

where $P_{i,L} = \{j \mid i\text{-}w \leq j \leq i\text{-}1\text{-k}$ and $A[j\ldots j\text{+}k\text{-}1]$ is a shared (*k*, *m*)-mer$\}$ and $P_{i,L} = \{j \mid i \leq j \leq i\text{+}w\text{-}k$ and $A[j\ldots j\text{+}k\text{-}1]$ is a shared (*k*, *m*)-mer$\}$. Recombination is then inferred by some metric computed based on the magnitude of difference between $kmd_{i,L}$ and $kmd_{i,R}$. This is further elaborated in Section 2.4.

2.2 Contigs as a Better Estimation of Long Common Subsequences

The previous section championed the use of shared (*k,m*)-mers over base-by-base comparisons when measuring homology between two sequences. However, they are too short to truly represent the degree of homology between two sequences. A better indication of homology between two sequences would be the number and length distributions of long common subsequences. Now, the question is how do we obtain long common subsequences of two sequences enclosed by a window with only short shared (*k,m*)-mers? Note that a length-*L* common subsequence of S_1 and S_2 is a tiling of $(L - k + 1)$ consecutive shared (*k,m*)-mers when $k \leq L$. In this paper, we define a common subsequence of S_1, S_2 as a chain of consecutive shared (*k,m*)-mers, which is known as a Contig.

Definition 2. A Contig is a length-*L* common subsequence of two sequences S_1 and S_2 formed by a chain of consecutive shared (*k,m*)-mers shared by S_1 and S_2. It has two parameters, namely the starting position *p* and the member size *s*. Here, *p* refers to the position of the Contig nearest to a putative breakpoint *i* and $s = L - k + 1$, the number of consecutive shared (*k,m*)-mers chained to form the Contig. Thus a Contig can be written as Contig(*p*, *s*). (See Fig. 2)

It is easy to see that any length-*L*' sub-Contig of a length-*L* Contig where $k \leq L' \leq L$ is guaranteed to have less than $(m/k * L')$ mismatches. In addition, Contigs have an even distribution of mismatches. On the contrary, long common subsequences may have concentrations of mismatches in localized regions, despite passing the overall mismatch threshold. This creates a dilemma of whether to split a long common subsequence into shorter ones at regions where there are many mismatches. Our definition of Contigs avoids this problem and thus more reflective of the localized similarity between two sequences.

This leads to the assessment of a position i being a true breakpoint by two criteria:

1. A Kolmogorov-Smirnov (KS) test on the Contig length distributions in $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$ at 99% confidence interval. This is because if i is a breakpoint, the Contig length distributions in $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$ will be significantly different due to the distinct difference in sequence homology of the alignment enclosed by each window.
2. A high score for the metric computed based on the magnitude of difference between the similarity scores in $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$.

Note that in the previous section, the km-distances, $kmd_{i,L}$ and $kmd_{i,R}$, were used as the similarity scores for the alignment enclosed by $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$. In the next section, we shall describe a weighting strategy to improve the km-distance to incorporate the concept of Contigs. Similarly, we elaborate on the metric to detect recombination in Section 2.4.

2.3 Breakpoint Specific Positional Weighted Distance Measure

Depicted in Fig. 1(a)(b), the alignment noise affect the detection of breakpoint. We solve the issue by assigning weights to all positions enclosed by $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$ with respect to a putative breakpoint i. More specifically, we assign heavier weights to positions in $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$ near to the putative breakpoint i while lighter weights to positions in $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$ that are further away from the putative breakpoint. We justify our proposed weighting strategy to solve the window length sensitivity problem based on Fig. 1(a)(b): In Fig. 1(a) where i is a true breakpoint, positions that are furthest away from i are most likely to contribute to alignment noise if the window length is too large. Thus, by assigning these positions the lightest weights when computing the km-distance for each window, alignment noise is reduced. In Fig. 1(b) where i is not a true breakpoint, the alignment near i in $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$ will not experience a sudden significant change. Since positions near i are assigned heavier weights, the km-distance in $[i\text{-}w\ldots i\text{-}1]$ and $[i\ldots i\text{+}w]$ will most likely not have a sudden significant change too.

Clearly, a good implementation of our weighting strategy requires a suitable function or a suitable family of functions that assigns a weight to a position in neighboring windows $[i\text{-}w\ldots i\text{-}1]$, $[i\ldots i\text{+}w]$ based on its absolute distance from a putative breakpoint i. Let S_1 and S_2 be two aligned length-n sequences with a putative breakpoint at position i with neighboring windows $[i\text{-}w\ldots i\text{-}1]$, $[i\ldots i\text{+}w]$. Let x be the relative distance of a position j in $[i\text{-}w\ldots i\text{-}1]$, $[i\ldots i\text{+}w]$ from the breakpoint i, that is $x = j\text{-}i$, and $-w \le x \le w$. Next, we define a positive weight function $F_i : X \rightarrow \Re^+$ where $X = \{x \mid -w \le x \le w\}$. $F_i(x)$ satisfies the properties that (i) $F_i(x) = F_i(-x)$ and (ii) $F_i(x)$ is decreasing when $|x|$ increases. For simplicity, we set $F_i(0) = 1$ and $F_i(w) = F_i(-w) = 0$. A family of functions satisfies the above properties is as follows:

$$F_i(x) = \frac{w^k - |x|^k}{w^k} \tag{3}$$

Note that k control the decreasing rate of $F_i(x)$. In our application, we need moderate decreasing rate and hence we set k=2 by default. In this case, $F_i(x)$ is in fact a reverse parabola shape.

Next, we describe how this weighting strategy incorporates the notion of Contigs in the km-distance. The idea is to assign a weight to each shared (k, m)-mer based on which Contig it belongs to. In this way, shared (k, m)-mers belonging to the more informative Contigs (closer to the putative breakpoint) are assigned heavier weights than those belonging to Contigs that are more prone to alignment noise (further away from the putative breakpoint). More specifically, our weighting strategy assigns each Contig(p, s) and its member shared (k, m)-mers a weight based on its starting position, $F_i(|i\text{-}p|)$. Since Contigs do not overlap in a window, the weight assigned to each Contig and its member shared (k, m)-mers is unique. Consequently, given Contig(p_1, s_1) and Contig(p_2, s_2) with assigned weights $F_i(|i\text{-}\ p_1|)$ and $F_i(|i\text{-}\ p_2|)$ respectively, $F_i(|i\text{-}\ p_1|) > F_i(|i\text{-}\ p_2|)$ iff $|i\text{-}\ p_1| < |i\text{-}\ p_2|$.

Thus, given a putative breakpoint i and the two neighboring windows [i-w…i-1] and [i…i+w], we compute the improved Breakpoint specific positional Weighted Contig-Alignment (*BWCA*) score for the alignment of S_1 and S_2 in each neighboring window of a putative breakpoint i:

$$\text{For } [i\text{-}w\ldots i\text{-}1],\quad BWCA_{i,L}(S_1,S_2) = \sum_{p_j\in P_{i,L};s_j\in S_{i,L}} F_i(i-p_j)\times s_j \tag{4}$$

$$\text{For } [i\ldots i\text{+}w],\quad BWCA_{i,R}(S_1,S_2) = \sum_{p_j\in P_{i,R};s_j\in S_{i,R}} F_i(p_j-i)\times s_j \tag{5}$$

where $C_{i,L}$ = {Contig(p_j, s_j) | $i\text{-}w \le p_j \le i$ -1} and $C_{i,R}$ = {Contig(p_j, s_j) | $i \le p_j \le i + w$} are the sets of Contigs in [i-w…i-1] and [i…i+w] respectively; $S_{i,L}$ = {s_j | Contig(p_j, s_j) $\in$ $C_{i,L}$} and $S_{i,R}$ = {s_j | Contig(p_j, s_j) $\in$ $C_{i,R}$} are the corresponding set of member sizes of the Contigs in $C_{i,L}$ and $C_{i,R}$; $P_{i,L}$ = {p_j | Contig(p_j, s_j) $\in$ $C_{i,L}$} and $P_{i,R}$ = {p_j | Contig(p_j, s_j) $\in$ $C_{i,R}$} are the corresponding set of starting positions of the Contigs in $C_{i,L}$ and $C_{i,R}$.

Similarly, we elaborate in Section 2.4 how recombination is inferred by some metric computed based on the magnitude of difference between $BWCA_{i,L}$ and $BWCA_{i,R}$.

2.4 The RB-Finder Algorithm to Detect Recombination

We have presented three techniques to address the main criticisms (accuracy and efficiency) of distance-based window methods to detect recombination. Next, we empirically investigate the effectiveness of our *km*-distance and *BWCA* scores to detect recombination as opposed to using a conventional distance measure such as the Kronecker delta function. Our simulations show that in real datasets where recombination events are more complex and harder to detect, our highly sensitive *BWCA* score stands a better chance of detecting the breakpoints than other distance measures (refer to supplementary for simulation details).

We make use of the *BWCA* score and propose the RB-Finder algorithm to detect recombination in a multiple sequence alignment. Given a length-*n* alignment of *M* sequences, the idea is to move a length-*w* sliding window along the alignment and, for each position *i*, computes a Recombination Detection Score (RDS_i) based on the highly sensitive *BWCA* score and two key observations to differentiate recombination and high mutation rates. At the real breakpoint *i*, two concurrent observations are prevalent: (1) there exists two sequences S_α, S_β in the alignment *M* such that the *BWCA* score increase significantly and suddenly across *i*, ie $BWCA_{i,L}(S_\alpha, S_\beta) - BWCA_{i,R}(S_\alpha, S_\beta) << 0$ and (2) there exists yet another sequence S_γ ($S_\gamma \neq S_\beta$) such that the *BWCA* score decreases significantly and suddenly across *i*, ie $BWCA_{i,L}(S_\alpha, S_\gamma) - BWCA_{i,R}(S_\alpha, S_\gamma) >> 0$. The first observation is made when there is a transfer of genetic sequence from S_β to S_α at *i* resulting in a sudden increase in homology between S_α and S_β. The second observation is that after the recombination event at *i*, S_α is no longer as homologous to some sequence S_γ as compared to prior the recombination event. From a phylogeny point of view, the two observations are in effect looking for the most divergent branch between the phylogenetic tree in [*i*-*w*…*i*-1] and the phylogenetic tree in [*i*…*i*+*w*]. This is shown in Fig. 3.

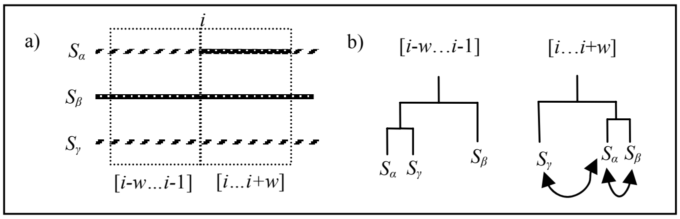


Fig. 3. (a) shows that S_α is a recombinant of S_β at position *i* to (*i*+*w*). (b) shows the phylogenetic trees in [*i*-*w*…*i*-1] and [*i*…*i*+*w*] respectively. In [*i*…*i*+*w*], S_α is phylogenetically closer to S_β, than in [*i*-*w*…*i*-1], resulting in $BWCA_{i,L}(S_\alpha, S_\beta)$ - $BWCA_{i,R}(S_\alpha, S_\beta) << 0$. Concurrently, S_α is phylogenetically further away from S_γ, resulting in $BWCA_{i,L}(S_\alpha, S_\gamma)$ - $BWCA_{i,R}(S_\alpha, S_\gamma) >> 0$. S_α is the most divergent branch between the phylogenetic tree in [*i*-*w*…*i*-1] and [*i*…*i*+*w*] and thus detected to most likely contain recombination.

At every putative breakpoint *i*, we examine each of the *M* sequences for the two observations. Specifically, for each sequence $S_\alpha \in M$ at *i*, we find Contigs with each of the other sequences S_β ($S_\beta \in M$ and $S_\alpha \neq S_\beta$) in neighboring windows [*i*-*w*…*i*-1] and [*i*…*i*+*w*]. The KS-test is then used to filter sequences whose distributions of contig lengths with S_α in both windows are not significantly different at 99% confidence interval. We then compute the *BWCA* scores of S_α with sequences that pass the KS-test in neighboring windows [*i*-*w*…*i*-1] and [*i*…*i*+*w*]. We obtain the most significant increase in *BWCA* score across [*i*-*w*…*i*-1] and [*i*…*i*+*w*] for S_α, that is, $BWCA_{i,incr}(S_\alpha) = \max\{BWCA_{i,R}(S_\alpha, S_\beta) - BWCA_{i,L}(S_\alpha, S_\beta) \mid S_\beta \in M \text{ and } S_\alpha \neq S_\beta\}$. Similarly, we also obtain the most significant decrease in *BWCA* score across [*i*-*w*…*i*-1] and [*i*…*i*+*w*] for S_α, $BWCA_{i,decr}(S_\alpha) = \max\{BWCA_{i,L}(S_\alpha, S_\gamma) - BWCA_{i,R}(S_\alpha, S_\gamma) \mid S_\gamma \in M \text{ and } S_\alpha \neq S_\gamma \neq S_\beta\}$. Subsequently, we compute $RDS_i(S_\alpha)$, the recombination detection score for S_α accordingly as shown in Table 1:

Table 1. The formula to compute the *RDS* for a sequence based on the 2 observations

Scenario	Observations Present	$RDS_i(S_\alpha) =$	Reason
1	(1) and (2)	$BWCA_{i,incr}(S_\alpha) * BWCA_{i,decr}(S_\alpha)$	Recombination
2	(1)	0	Homologous regions not caused by recombination, most probably conserved regions
3	(2)	$- BWCA_{i,decr}(S_\alpha)$	High rates of mutation
4	None	0	No significant change in homology

Note that scenario 1 produces a distinctly high $RDS_i(S_\alpha)$ that indicates S_α has a recombination event at position *i*. On the other hand, scenario 3 produces a negative $RDS_i(S_\alpha)$ to clearly indicate high mutation rates. The other scenarios are deemed uninteresting in recombination detection and assigned $RDS_i(S_\alpha) = 0$.

Finally, we select the highest *RDS* among the *M* sequences to representing the *RDS* for breakpoint *i*:

$$RDS_i = \max(RDS_i\,(S_\alpha)) \qquad (6)$$

It is easy to see that if $RDS_i(S_\alpha) < 0$ for all $S_\alpha \in M$, then $RDS_i < 0$. This would indicate that the region around *i* suffers from high mutation rates. Conversely, a high RDS_i would mean that *i* is most likely a true recombination breakpoint. We present the pseudo-code for the RB-Finder algorithm below in Fig. 4.

3 Evaluation of the RB-Finder Algorithm

Evaluation of our recombination detection algorithm is carried out by applying our algorithm to three synthetic and one biological datasets used in two previous papers [7, 8]. The three synthetic datasets (SD1, SD2 and SD3) each contains a 5500-bp alignment of eight sequences (S_1, S_2, …., S_8) whose evolution was simulated with the Kimura model [10]. For SD1 and SD2, two recombination events were simulated: an ancient event affecting the region between sites 1000 and 1500, and a recent event affecting the region between sites 2500 and 3000. To test whether the detection method can successfully differentiate between recombination and rate variation, a mutational hotspot between sites 4000 and 4500 was introduced. The average branch length of the underlying phylogenetic trees for SD1 and SD2 are 0.1 and 0.01 respectively. The third synthetic dataset SD3 contains an ancient event affecting the region between sites 1000 and 2000, and a recent event between sites 3000 and 4000. The branch lengths of the tree were drawn from a uniform distribution on the interval [0.003, 0.005]. SD3 was deliberately created by Husmeier *et al.* [7] to thwart previous algorithms cited in their paper. The biological dataset used in our experiment is a length-3049 gap-removed ClustalW [11] alignment of 10 Hepatitis B virus sequences. It consists of two recombinant strains (D0329 and X68292) and eight non-recombinant strains (V00866, M57663, D00330, M54923, X01587, D00630, M32138 and L27106).

```
RB-Finder{
   Given an length-$N$ alignment of $M$ sequences,length of $(k,m)$-mer $k$,
   and sliding window length $w$,
   for each position $i$ from 1 to $(N - w)${
        Declare neighboring windows [$i$-$w$…$i$-1] and [$i$…$i$+$w$] ;
        for each sequence $S_\alpha$ in $M$ {
                Determine $m$ to define a shared $(k,m)$-mer;
                for each sequence $S_{\alpha'}$ in $M$ {
                        Obtain set of Contigs in [$i$-$w$…$i$-1], $C_{i,L}$
                        Obtain set of Contigs in [$i$…$i$+$w$], $C_{i,R}$
                        If(KS($P_{i,L}$, $P_{i,R}$) <= 0.01) {
                                Compute $BWCA_{i,L}$ ($S_\alpha$ , $S_{\alpha'}$);
                                Compute $BWCA_{i,R}$ ($S_\alpha$ , $S_{\alpha'}$);
                        }
                }
                Compute $BWCA_{i,incr}$($S_\alpha$);
                Compute $BWCA_{i,decr}$($S_\alpha$);
                Compute $RDS_i$ ($S_\alpha$ );
        }
        Compute $RDS_i$ ;
   }
}
```

Fig. 4. The pseudo-code for RB-Finder algorithm

We ran our algorithm using (k,m)-mers with k=20 while m is automatically determined depending on the point mutation rate, and two window lengths 500 and 600. Note that the optimal window length to detect recombination in three of the four datasets (SD1, SD2 and Hep B) is 500 since all the recombination events that happened in these three datasets are of span 500 nucleotides. Thus, a window length of 600 would generate alignment noise and decrease accuracy of recombination detection in the three datasets. We shall see from the following results that the effects of alignment noise on recombination detection using our algorithm were minimal. We also compare our results with that from Pruned-PDM since it has the highest accuracy.

We only show our results for SD3 (refer to supplementary for the analysis results for SD1, SD2 and HepB dataset) since it is a difficult dataset to analyze because there are only subtle differences in the alignment. Despite the very low rate of evolution in SD3, our algorithm detected recombination breakpoints at sites 1000, 2000, 3000 and 4000. In addition, our algorithm also detected a mutational hotspot around site 5000. The results for SD3 using window lengths 500 and 600 are shown in Fig. 5. This time, Pruned-PDM not only produced inaccurate breakpoints, but also failed to detect the breakpoint at position 2000.

The results of the four datasets are consistent with the DSS, Pruned-PDM and RECOMP methods. Through the use of our proposed weighting strategy and two key recombination-identifying observations, our algorithm is able to compute recombination breakpoints of a given alignment in a matter of minutes. Compared to Pruned-PDM which takes hours to analyze the same dataset, our algorithm is very much faster and achieves similarly accurate results. Unlike RECOMP which gives the user a choice of four graphs to decipher the recombination breakpoints, our algorithm generates

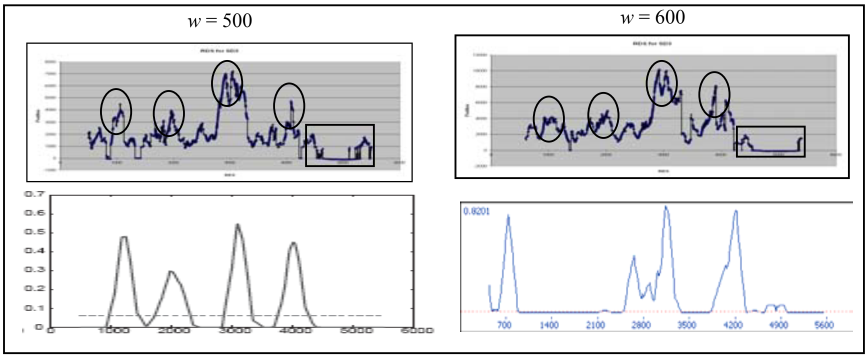


Fig. 5. Top: Results of our algorithm on SD3 with window lengths of 500 and 600. The circles highlight the recombination breakpoints at position 1000, 2000, 3000 and 4000 respectively. The rectangles highlight the high mutation regions which we detected. Bottom: Results of Pruned-PDM on SD3 with window lengths of 500 and 600. When w = 600, the breakpoint at position 2000 was undetected. Other breakpoints are also inaccurate.

only a single graph and thus prevents ambiguity of deciding which graph best represents the correct recombination breakpoints. An example of this ambiguity is shown in their results for the SD3 dataset [8]. In their analysis, three of four graphs wrongly indicated that there is a recombination breakpoint at position 5000. Hence, it is difficult for the user to correctly infer that the recombination breakpoint detected at position 5000 by the three graphs is incorrect based on the remaining graph. In our analysis, we correctly identify the region around position 5000 as a mutational hotspot. This example also illustrates another advantage that our algorithm has over previous methods. Previous algorithms cannot differentiate normal regions from mutation hotspots. A useful feature of our algorithm is that it produces a $RDS < 0$ when the region has a high mutation rate. This immediately provides more biological information about the sequences to aid experimental studies.

3.1 Analysis of Circulating Recombinant Forms of HIV-1

Acquired Immune Deficiency Syndrome (AIDS) is a worldwide epidemic caused by a virus known as human immunodeficiency virus (HIV). Efforts to develop HIV vaccines and medicine have been thwarted with difficulties due to the fast mutation and recombination rates of HIV [12]. There are two types of HIV, namely HIV-1 and HIV-2. HIV-1, which is responsible for most human infections, consists of three major groups: M, N and O. The most common group M of HIV-1 is further characterized into 9 subtypes (A, B, C, D, F, H, J and K). The ease at which HIV-1 subtypes recombine has also resulted in numerous circulating recombinant forms (CRFs) of HIV-1 [13]. These CRFs pose more difficulties to finding a cure. Thus, the analysis of existing CRFs and the identification of new CRFs will be vital to the efforts against HIV.

The HIV database [14] contains a list of all existing CRFs to date. Each CRF is classified according to its recombinant subtypes. Eg CRF02_AG is a recombinant of subtype A and subtype G. In addition, a graphical representation of each CRF in terms of where the recombination occurs at gene level is also available [15]. From the database, we downloaded the sequences of 13 CRFs and 35 reference sequences of the 9 subtypes. Experiments are performed in the following manner:

1. For each CRF, download its sequence and several reference sequences of its parent subtypes. Align with ClustalW with default parameters.
2. Run RB-Finder on each alignment. Gaps are not removed because they appear in almost all regions of the alignments. Instead, we modified our program so that a length-w sliding window is allowed to have $0.1w$ gaps. This is to accommodate insertions and deletions in our computation of recombination breakpoints. To prevent inaccuracies caused by incompletely assemblies, sequences with more than $0.1w$ gaps in a particular window will not be considered.
3. Identify breakpoints of each CRF and compare with the graphical representation of the corresponding CRF. In addition, identify the reference sequences of its parent subtypes that contributed to the recombination.

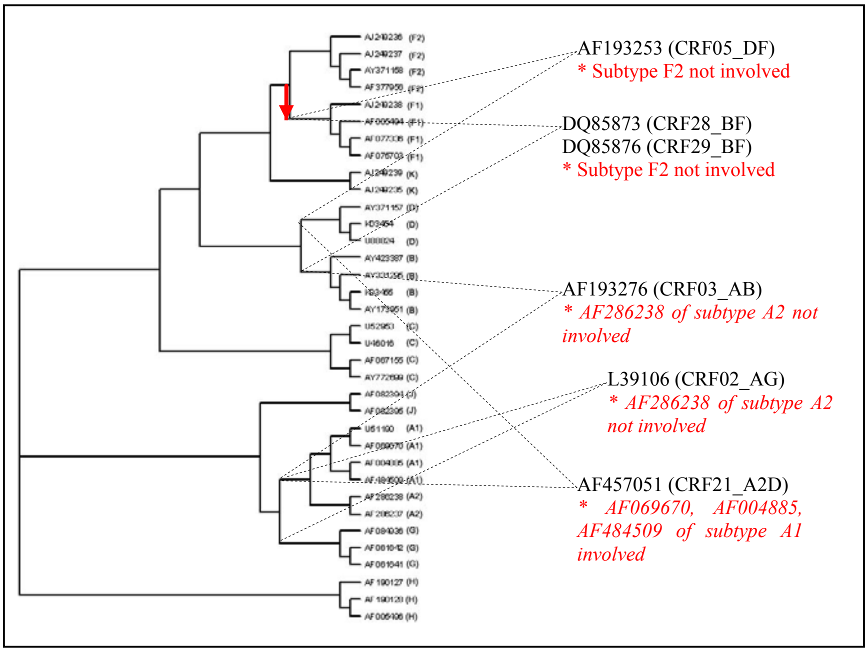


Fig. 6. The proposed putative phylogenetic network of 35 reference sequences of the 9 HIV type M subtypes and the 6 CRFs which had irregularities with their subtyping. In the diagram, irregularities are presented in red, italic font under their respective CRFs. In addition, the red arrow indicates our proposed change in genotyping of CRF05_DF, CRF28_BF and CRF29_BF not to include the subtype F2. The other 7 CRFs which had no irregularities with their subtyping are not shown.

RB-Finder finds almost all breakpoints indicated by the literature, except that when the recombination region is too short, RB-Finder reports only one breakpoint for that region instead of two breakpoints. Furthermore, using RB-Finder to identify reference sequences of parent subtypes that contributed to recombination events in the CRFs yielded some interesting findings. Firstly, RB-Finder is able to determine that none of the reference sequences belonging to subtype F2 contributed to recombination events in CRF28_BF and CRF29_BF. Subsequently, we found that although some CRFs are labeled as a recombinant of two subtypes, not all reference sequences of the parent subtypes are involved in the recombination events (Eg CRF02_AG, CRF03_AB and CRF05_DF). On the other hand, we also found that some reference sequences not in the reported parent subtype of a CRF may be involved in some of its recombination events (Eg CRF21_A2D may be a recombinant of subtype A1 as well). This suggests that further analysis may be needed to more accurately classify CRFs. Based on the results from RB-Finder, we propose a putative phylogenetic network of the 35 reference sequences belonging to the 9 subtypes and the 6 CRFs which had irregularities with their subtyping in Fig. 6. (Refer to supplementary for full summary).

4 Conclusion

This paper introduced a breakpoint specific positional weighted distance measure to detect recombination in DNA sequence alignment. Until our paper, there is no distance measure defined to compare the similarity of an alignment of two or more sequences with respect to putative breakpoints. Furthermore, the graphical representation of the results from our algorithm shows clearly mutational hotspots which previous algorithms could not identify. In terms of detection accuracy, our results are at least as good as the best current methods such as pruned PDM. Furthermore, our algorithm only takes a few minutes to compute the results for each of the four datasets described. In terms of result presentation, our algorithm produces only a single graph which clearly shows recombination breakpoints and mutational hotspots. Compared to RECOMP which produces four graphs, our presentation is less ambiguous. Our future work involves exploring the possibility of using shared (*k,m*)-mers, Contigs and our weighting strategy to estimate a localized optimal window length to detect regional recombination of an alignment. This would avoid the selection of a window length altogether and will no doubt improve the sensitivity and accuracy of window-based methods to detect recombination. An implementation of the algorithm in C or Java and all data used in this paper are available upon request.

References

1. Posada, D.: Evaluation of methods for detecting recombination from data sequences: Empirical data. Molecular Biology and Evolution 19 (2002) 1198-1212
2. Weiller, G.F.: Phylogenetic profiles: a graphical method for detecting genetic recombination in homologous sequences. Molecular Biology and Evolution **15** (1998) 326-335
3. Myers, S.R. and Griffiths, R.C.: Bounds on the minimum number of recombination events in a sample history. Genetics **163** (2003) 375-394
4. Schierup, M.H. and Hein, J.: Consequences of recombination on traditional phylogenetic analysis. Genetics **156** (2000) 879-891

5. McGuire, G. and Wright, F.: TOPAL 2.0: improved detection of mosaic sequences within multiple alignments. Bioinformatics **16** (2000) 130-134
6. Husmeier, D. and Wright, F.: Probabilistic divergence measures for detecting interspecies recombination. Bioinformatics **17** (2001) 123-131
7. Husmeier, D., Wright, F. and Milne, I.: Detecting interspecific recombination with a pruned probabilistic divergence measure. Bioinformatics **21** (2005) 1797-1806
8. Ruths, D. and Nakhleh L.: RECOMP: A parsimony-based method for detecting recombination. Proceedings of The Fourth Asia-Pacific Bioinformatics Conference (2006) 59-68
9. Press, W.H., Flannery, B.P., Teukolsky, S.A. and Vetterling, W.T.: Kolmogorov-Smirnov Test. Numerical Recipes in FORTRAN: The Art of Scientific Computing, 2nd ed. (1992) 617-620
10. Kimura, M.: A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. Journal of Molecular Evolution **16** (1980) 111-120
11. Thompson, J.D., Higgins, D.G. and Gibson, T.J.: CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequencing weighting, position specific gap penalties and weight matrix choice. Nucleic Acids Res. **22** (1994) 4673-4680
12. Sturmer, M., Doerr, W. and Preiser, W.: Variety of interpretation systems for human immunodeficiency virus type 1 genotyping: Confirmatory information or additional confusion? Current Drug Targets Infectious Disorder **3** (2003) 373-382
13. Leitner, T., Korber, B., Daniels, M., Calef, C. and Foley, B.: HIV-1 Subtype and Circulating Recombinant Form (CRF) Reference Sequences. http://www.hiv.lanl.gov/content/hiv-db/REVIEWS/RefSeqs2005/RefSeqs05.html (2005)
14. The HIV Sequence Database. http://www.hiv.lanl.gov/content/hiv-db/mainpage.html
15. The Circulating Recombinant Forms.
http://hiv-web.lanl.gov/content/hiv-db/CRFs/ CRFs.html

Comparative Analysis of Spatial Patterns of Gene Expression in *Drosophila melanogaster* Imaginal Discs

Cyrus L. Harmon[1], Parvez Ahammad[2], Ann Hammonds[1], Richard Weiszmann[3], Susan E. Celniker[3], S. Shankar Sastry[2], and Gerald M. Rubin[1]

[1] Department of Molecular and Cell Biology, University of California, Berkeley, Berkeley, CA 94720
[2] Department of Electrical Engineering and Computer Science, University of California, Berkeley, Berkeley, CA 94720
[3] Department of Genome Sciences, Lawrence Berkeley National Laboratory, Berkeley, CA 94720

Abstract. Determining the precise spatial extent of expression of genes across different tissues, along with knowledge of the biochemical function of the genes is critical for understanding the roles of various genes in the development of metazoan organisms. To address this problem, we have developed high-throughput methods for generating images of gene expression in *Drosophila melanogaster* imaginal discs and for the automated analysis of these images. Our method automatically learns tissue shapes from a small number of manually segmented training examples and automatically aligns, extracts and scores new images, which are analyzed to generate gene expression maps for each gene. We have developed a reverse lookup procedure that enables us to identify genes that have spatial expression patterns most similar to a given gene of interest. Our methods enable us to cluster both the genes and the pixels that of the maps, thereby identifying sets of genes that have similar patterns, and regions of the tissues of interest that have similar gene expression profiles across a large number of genes.

Primary keyphrases: Genomic imaging, Gene expression analysis, Clustering.

Secondary keyphrases: Microarray data analysis, Imaginal discs.

1 Introduction

We have developed a high-throughput pipeline for generating images of imaginal discs, stained with labeled probes that hybridize to individual genes, and an automated system for analyzing a database of such images and for building an atlas of gene expression patterns. Using these tools, we construct consensus shape models of individual imaginal disc types which are used to automatically extract imaginal disc shapes from new images and to align these features to consensus shape models. In parallel, we extract the stain pattern for each disc which is aligned to and overlaid on the consensus shape model, thereby yielding a representation of the spatial extent of expression of a given gene in the context of the consensus shape. These patterns are averaged to produce consensus gene expression representations for each gene. The consensus representations are assembled in a database and clustered to identify both genes and regions of the discs with similar patterns of gene expression. New images can be used to search the database to identify known patterns similar to a query pattern, which can be automatically extracted from a new image. The data-flow of our approach is shown in Figure 1.

Previous work has identified the vast majority of the genes in *Drosophila* [1, 2], shown how a large numbers of genes are dynamically expressed across the entire organism throughout the life cycle [3], and shown how genes are expressed across space and time in embryos [4]. DNA-microarrays have enabled parallel, high-throughput analysis of gene expression from many different tissues under different conditions or at different time points [5, 6]. However, these techniques do not provide spatial information about where these genes are expressed within these tissues.

Image processing techniques have been applied to the problem of quantitative gene expression analysis by analyzing the hybridization of fluorescent probes to specific spots on DNA microarrays [6, 7], and, recently, fully-automated approaches to this problem have been developed [8].

Large-scale studies of patterns of gene expression in *Drosophila* have been performed using DNA microarrays both on whole organisms [9, 3] and individual tissues such as imaginal discs [10, 11]. Klebes et al. compared differential gene expression in different imaginal discs and between imaginal discs and non-disc tissue. Butler et al. manually dissected imaginal discs and were able to identify transcripts that were enriched in specific compartments of the wing discs [10]. However, these studies yield little information about the precise spatial patterns of gene expression.

2 Pipeline Overview

Our pipeline, shown in Figure 1, begins with the acquisition and purification of the biological material, followed by steps to select appropriate genes for inclusion in the project. The next phase of the pipeline entails generation of labeled probes for specific genes, hybridization of these probes to appropriate pools of imaginal discs, and imaging of the resulting stained discs. The data analysis portion of the pipeline consists of two main sections, a learning process in which a small number of manually segmented examples are used to learn the shapes of the imaginal discs and an automated analysis phase in which the learned shapes are used to automatically isolate instances of imaginal discs in the complete set of images. These aligned images are then scored for stain, and used to produce consensus maps for each gene. The consensus maps are then used for a variety of clustering and analysis procedures.

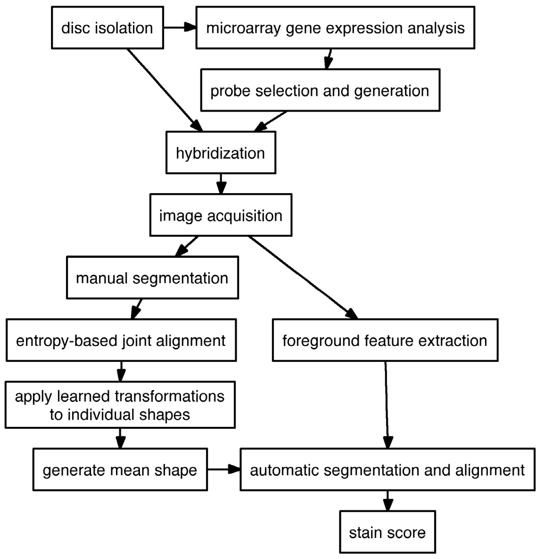

disc isolation
microarray gene expression analysis
probe selection and generation
hybridization
image acquisition
manual segmentation
entropy-based joint alignment
foreground feature extraction
apply learned transformations to individual shapes
generate mean shape
automatic segmentation and alignment
stain score

Fig. 1. Overview of the high-throughput process for determining spatial patterns of expression of genes in *Drosophila* imaginal discs

For the high-throughput pipeline, we needed a sufficient source of imaginal discs such that we would have a number of each type of imaginal disc for each probe. Moreover, it was desirable to be able to work in 96-well plates, following the protocol of the Berkeley Drosophila Genome Project (BDGP) *Drosophila* embryo gene expression pattern project [4]. We adapted the imaginal disc mass-isolation protocol from Eugene et al. [12] for use in a 96-well format. We collected eggs from Canton S stocks and grew larvae until the wandering third instar larval stage, at which point the larvae were harvested and the discs isolated. The mass-isolation procedure typically yielded 500,000 discs, including wing, leg, eye-antennal, haltere and genital discs, but not the smaller disc types, as these were not recovered in our purification process. Approximately 100,000 discs were used per 96-well plate, yielding on the order of 1000 discs per probe.

2.1 Gene Selection and Microarray Gene Expression Analysis

While it would be desirable to examine the spatial pattern of every gene, this is prohibitively expensive and time-consuming. For the current project, we had the resources to produce probes for only a few hundred genes. Many genes have no detectable spatial expression pattern, or, on the other hand, are ubiquitously expressed and we would like to avoid making probes for these genes. Therefore, we have developed a protocol to examine the over 13,000 genes in the *Drosophila* genome using gene expression microarrays to analyze the levels of expression across multiple tissues. The central idea is that by examining the quantitative level of gene expression across multiple tissues, including *Drosophila* embryos at multiple time points, distinct classes of imaginal discs, and adults, we could identify genes that were differentially expressed in imaginal discs and therefore more likely to have a non-trivial spatial pattern of gene expression than genes that were expressed at a constant level across all tissues. In particular, we identified genes that were expressed at a higher level in certain imaginal disc types, relative to the other disc types, or that were more highly expressed in all imaginal discs relative to the embryo and adult samples.

2.2 Disc Annotation Database

In order to facilitate the process of automatic segmentation, the operator records meta-data about the images, as they are being captured, in a database [4]. The annotations include the orientation of the peripodial epithelium and the handedness of the disc, from which we deduce whether or not the disc needs to be reflected about the vertical axis with respect to the canonical orientation.

All imaginal discs except the genital discs occur in pairs, one on each side of the larval body, yielding a left disc and a right disc of each type. When the discs are placed on a microscope slide for imaging, the can be found in on of two orientations, with the peripodial epithelium, which exists on only one side of the disc, either on the top or the bottom. The combination of the handedness of the disc and the orientation of the peripodial epithelium gives us 4 possibilities for the combined state and orientation of the disc. We make a simplifying assumption and assume that the left and right discs are mirror images of each other and that the stain pattern of a gene in a right disc will be equivalent to the mirror image of the stain pattern of that gene from the left disc of the same type. This assumption gives us two handedness/orientation combinations that are considered to be in the canonical orientation, and two combinations that are mirror images of these. The shapes corresponding to the two mirror image handedness/orientation combinations are automatically reflected about the vertical axis by the congealing pipeline after manual segmentation, to bring the images into the canonical orientation.

2.3 Shape Representation and Learning

The results of our manual segmentation process are a set of binary image masks that represent shapes from individual images. From these shapes we learn a shape representation in the form of a canonical binary image in which pixels are either 1 for foreground or 0 for background, or a canonical grayscale image in which pixels values are between 0 and 1 and where the value represents the probability that a given pixel is on in an element of that particular shape. This simple shape model for each imaginal disc type serves as the reference map upon which comparisons regarding patterns of gene expression will be made. While there are many possible approaches to this task, including parameterizing the curve that defines the borders of the shapes, we use a simple non-parametric method called Congealing [13, 14], that iteratively learns a set of affine transformations that minimize the overall pixelwise entropy of a stack of images to learn a canonical shape model for each disc type [15].

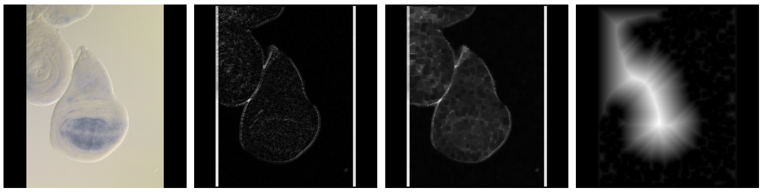


Fig. 2. (a) Raw RGB image file of a wing imaginal disc. (b) Discrete approximation of the second derivative of a wing imaginal disc image, produced by convolution with a Laplacian filter kernel. (c) The previous image after a round of morphological closing by dilation and erosion. (d) Resulting image after applying the 3-4 distance transform.

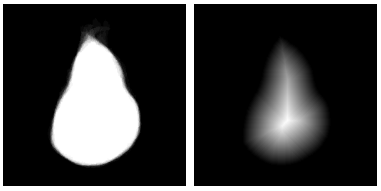


Fig. 3. (a) Canonical wing disc shape model. (b) 3-4 distance transform of the wing disc shape model.

3 Parallel Alignment and Automatic Feature Extraction

Our approach to identifying imaginal discs in images is based on extraction of simple foreground features and subsequent alignment of the foreground features to a shape model by optimizing the parameters of an affine transformation between the foreground features and the shape model. This approach was initially proposed by Barrow et al. [16] for use with manually extracted point features which were aligned to a traced boundary. In order to avoid becoming trapped in local minima during the optimization process, a process known as "Chamfering" is used to generate a map of the distance of foreground features to the nearest edge. Borgefors called this map of the distance to the foreground features a "Distance Transform" and proposed a hierarchical method for performing this style of alignment in a coarse-to-fine manner [17].

3.1 Foreground Feature Extraction

Foreground biological material has substantially more variability than the background. While the measured intensity values of the background change across the image, these values generally change smoothly. The biological material present in the image, on the other hand, contains substantial pixel intensity variability due to changes in the scattering and absorption of light by the material. Therefore, we use the absolute value of a discrete approximation of the second derivative of the image, produced by discrete convolution with a laplacian filter kernel [18], to detect the foreground biological material. An example image is shown in Figure 2(b). After applying the Laplacian operation, the image is then morphologically closed by first dilating the image, then eroding the image using a square structuring element. The resulting image is shown in Figure 2(c).

The Distance Transform is a transformation of an image in which edge (or foreground) pixels are assigned a value of zero and non-edge (or non-foreground) pixels are assigned a value proportional to the distance to the nearest edge pixel [17, 19]. The true Euclidean distance is somewhat computationally expensive. Fortunately, Borgefors provides an efficient approach, known as the 3-4 Distance Transform, for approximating the true Euclidean distance using a two-pass, forward and backward, algorithm that analyzes the backward and forward 8-neighbors, respectively, of each pixel and computes an approximation of the distance of each pixel to the nearest edge pixel. A visual inspection of this approach and the L_1-based approximation provided by Soille [19] shows that the 3-4 Distance Transform yields substantially better results.

Just as we apply the distance transform to the target image, we apply the distance transform to the model to produce a smoothed-out representation of the model, suitable for template matching. The results of the distance transformed model can be seen in Figure 3.

3.2 Distance Metrics

There are multiple instances in our pipeline where it is necessary to compare one template with another template, and assign a score based on a distance measure that measures how dissimilar they are to each other. Depending on the scenario, the template can be a real-valued vector or a real-valued or binary two-dimensional image patch. Common distance choices include the L_1 and L_2 distance metrics. However the L_1 and L_2 distances are not invariant to scalings and shifts in image intensities. Therefore, we used the Normalized Cross-Correlation distance[20] to compare images. To compare two image patches, X and Y, where $X, Y \in \mathbf{R}^{\mathbf{MxN}}$, we use the following formula:

$$\mathrm{NCC(X,Y)} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(X_{i,j}-\bar{X})(Y_{i,j}-\bar{Y})}{\sqrt{\sum_{i=1}^{M}\sum_{j=1}^{N}(X_{i,j}-\bar{X})^2\sum_{i=1}^{M}\sum_{j=1}^{N}(Y_{i,j}-\bar{Y})^2}} \tag{1}$$

where $\bar{X}$ denotes the mean value of the template X. The value returned by NCC always ranges between -1 and +1, irrespective of the size of the template. When NCC=+1, the two templates match perfectly. If the templates X and Y take only binary values $0, 1$, we define the set of pixels on in both templates as $A = \sum_{i=1}^{M}\sum_{j=1}^{N}\mathbf{1}(\mathbf{X_{i,j}=1, Y_{i,j}=1})$, the sets of pixels on in either template as $B = \sum_{i=1}^{M}\sum_{j=1}^{N}\mathbf{1}(\mathbf{X_{i,j}=1, Y_{i,j}=0})$, and $C = \sum_{i=1}^{M}\sum_{j=1}^{N}\mathbf{1}(\mathbf{X_{i,j}=0, Y_{i,j}=1})$, and the set of pixels off in both templates as $D = \sum_{i=1}^{M}\sum_{j=1}^{N}\mathbf{1}(\mathbf{X_{i,j}=0, Y_{i,j}=0})$ where $\mathbf{1(p)}$ is an indicator function that takes value 1 if the condition p is satisfied. The Jaccard metric can then be defined in terms of A, B, C, D.

$$\mathrm{Jaccard(X,Y)} = \frac{A}{A+B+C} \tag{2}$$

The NCC measure is equivalent to the Jaccard metric when $D >> (A + B + C)$. While the Jaccard metric explicitly avoids computing distance between the features that are absent in both X and Y templates – and is thus desirable, it doesn't extend well to real-valued templates.

3.3 Coordinate Descent from Multiple Starting Configurations

Given the distance-transformed model, an image corresponding to the identified foreground features, and an appropriate distance metric, we seek an affine transformation that, when applied to the target image, minimizes the distance between the (affine) transformed target features and the model. We use a reasonable set of starting configurations, such as the identity affine transform, and 90 degree rotations of the identity transform, and perform coordinate descent on the parameters of the affine transformation. This procedure quickly yields an acceptable match between the target and the imaginal disc model over 85% of the time. The procedure can fail to find a good match when the image is cluttered or contains multiple discs. Poorly performing instances are manually removed from the atlas prior to further processing.

While the correct alignment between two images generally yields a local minimum, this is not the only minimum to be found in the space described by the parameters of the affine transformation. In congealing, we avoid settling on an incorrect local minimum which may be found by reducing the scale of both images towards zero or infinity, essentially matching two all black or all white images, by rescaling the affine transformations in each iteration of the algorithm. In the case of the distance transforms of the foreground features and the model, we occasionally find that a better score is found via an incorrect alignment of two images that have undergone substantial transformations to achieve this alignment. To address this problem, we constrain the affine transformations such that the log-space parameters for x- and y-scaling are never allowed to go below -0.35 or above 0.35, constraining the minimum size of the figure to be 70% of the original size and the maximum to be 142% of the original size. Similarly, the log scale x- and y-shear parameters are never allowed to go below -0.2 or above 0.2. Finally, as an additional constraint on the transformations, we never allow the absolute value of the difference between the x-scale and y-scale log-space parameters to go over 0.2. This drives the alignment procedure to local minima that are more likely to represent the true biologically relevant alignment, even if the procedure would otherwise a find a lower-scoring alignment by using a radical deformation of the shapes.

3.4 Stain Scoring

Imaginal discs are stained with digoxigenin labeled mRNA or DNA probes. The labeled discs were then treated with anti-digoxigenin coupled to alkaline phosphatase, an enzyme that catalyzes a reaction that generates a blue dye from a colorless substrate. The presence of labeled probe is indicated by a blue color, resulting from the absorption of non-blue photons by the dye. We have developed a quantitative stain-scoring heuristic that uses the difference between the

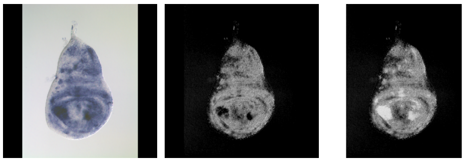


Fig. 4. (a) An image of *Drosophila melanogaster* wing imaginal discs with areas of heavy stain. (b) The result from the stain-scoring algorithm using Equation 3 producing clipped areas in the center where no stain is reported. (c) The improved stain-scoring result obtained by using Equation 4.

pixel intensity values of the blue channel and the average of the red and green channels. We compute the stain intensity for a given pixel as shown below.

$$s_{ij} = b_{ij} - \frac{r_{ij} + g_{ij}}{2} \tag{3}$$

As shown in Figure 4, this approach yields satisfactory results except in areas of very high staining, at which point the pixel values become very dark gray with no discriminating values between the blue channel and red and green channel average. Therefore, we set a minimum value for the blue channel and treat values in the red and green channels beneath this threshold as evidence of staining and use the difference between the value in the average of the red and green channels and the minimum as the value for the stain intensity in these areas.

$$s_{ij} = \max\left(b_{\min}, b_{ij}\right) - \frac{r_{ij} + g_{ij}}{2} \tag{4}$$

Ideally, we would be able to use the measured stain intensity as a direct measure of the abundance of a particular transcript at a given location. Unfortunately, this is not the case as there is substantial variability for the effectiveness of individual probes. While the current method does not yield directly comparable measures of abundance that are valid for comparisons between different genes, the measures of abundance for a single gene, over the spatial extent of the disc should be more directly comparable. In order to maximize the measured differences in expression of a single gene across the entire disc, one can normalize the recorded values, such that the highest measured stain value is assigned to a predetermined value, and other values scaled, linearly, such that the highest measured value is assigned to the highest value on the new scale, and the zero values on the original scale are assigned to zero values on the new scale. By performing this kind of normalization, we expand the dynamic range of the measured pixel values for given genes.

In Figure 5, we show images of imaginal discs that have been automatically aligned, extracted from background. We have aligned over 800 images from over 130 different genes across the four main imaginal disc types.

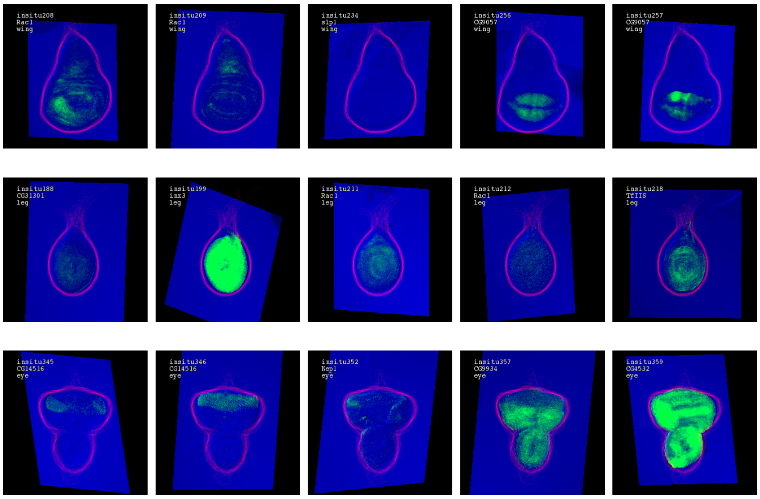


Fig. 5. Images of *Drosophila melanogaster* wing (first row), leg (second row), and eye/antenna (third row) imaginal discs, automatically segmented and aligned to the model. The original grayscale image is shown in the blue channel, the outline of the disc model is shown in red and the scored stain is shown in green.

4 Gene Expression Maps and Map Clustering

We would like to construct maps of gene expression that incorporate multiple samples, when possible. Two simple approaches are to use the mean and the median of the available maps. For each gene and disc-type pair, if there is only one image, we treat this image as the map. If there is more than one image, we construct maps where the value of each pixel is the median of the values from each aligned, stain-scored image at the given pixel, which are shown in Figure 6. In addition to the median, we construct maps of the pixelwise standard deviation of the stain scored image, yielding a measure of the variability at for a given pixel across the samples (data not shown).

4.1 Reverse Lookup to Find Similar Expression Patterns

Having built a data set of images and maps, one can interrogate new images to automatically extract a disc of a given type, score the image for stain and compare the stain pattern to either the individual stain patterns in the database or, perhaps more importantly, the median maps for each gene, thereby allowing one to discover to which genes the new pattern is most similar. In Figure 7 we see a new image which is then automatically segmented, extracted and scored for stain. The resulting stain pattern is then searched against the median maps using a simple L_2 distance metric and the 5 resulting maps most similar to this pattern are shown.

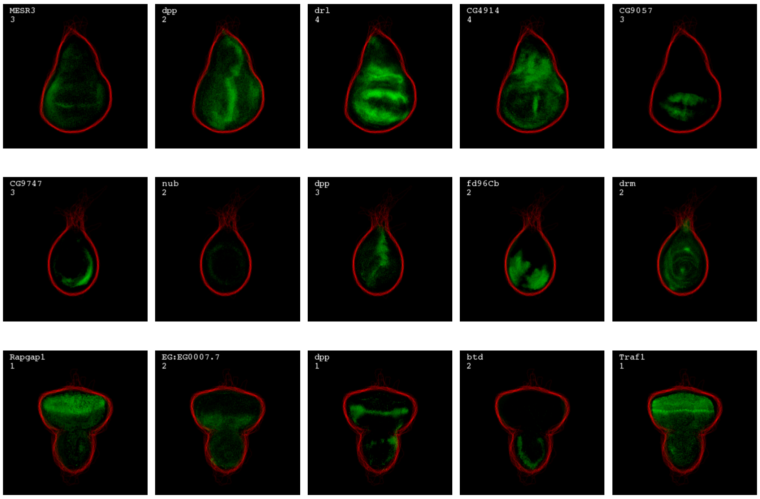


Fig. 6. Gene expression maps of imaginal discs in *Drosophila melanogaster*. The maps are made by taking the median expression value at each pixel from the stack of images for each gene. (a) Maps of MESR3, dpp, drl, CG4914, and CG9057 in the wing. (b) Maps of CG9747, nub, dpp, fd96Cb, and drm in the leg. (c) Maps of Rapgap1, EG:EG0007.7, dpp, btd, and Traf1 in the eye/antenna disc.

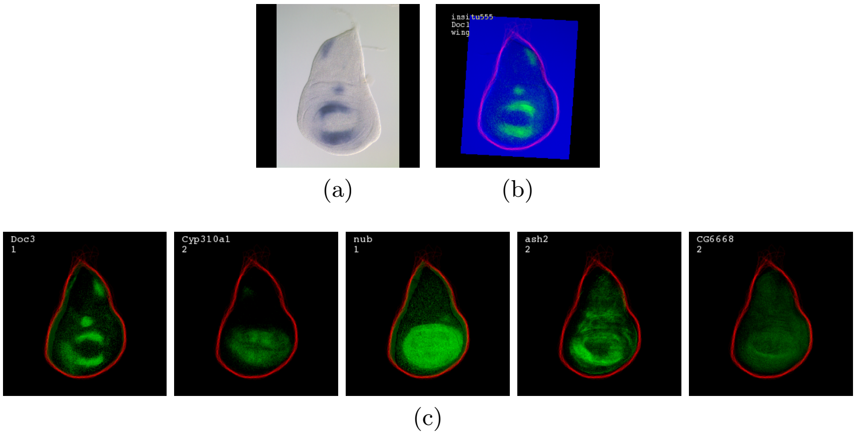


Fig. 7. Reverse lookup procedure to search the database of average gene expression maps for patterns most similar to the Doc1 gene. (a) Image of a wing imaginal disc stained for Doc1 gene expression. (b) Automatically aligned and extracted stain pattern for Doc1. (c) Average gene expression maps of Doc3, Doc2, nub, Cyp310a1 and Pepck.

4.2 Comparative Maps of Multiple Genes

By registering the images to a common global template for each shape, we are able to make comparisons between expression patterns determined in different

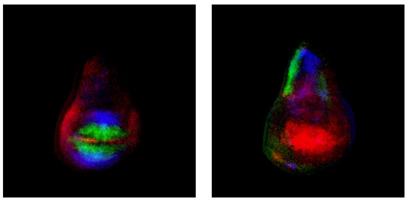


Fig. 8. Comparative gene expression maps. The expression patterns of multiple genes in Wing imaginal discs are shown overlaid on a common reference map. (a) MESR3 (red), CG9057 (green) and Sp558 (blue). (b) BG:DS00180-3 (red), drm (green) and Timp (blue).

experiments and we are able to create overlay maps that contain a representation of the spatial patterns of more than one gene. While one can use biological techniques to image multiple genes simultaneously [21, 22, 23], one must know which genes to look for . Using our methods, we can computationally construct virtual comparative maps of arbitrary sets of genes. Example overlay maps are shown in Figure 8.

In addition to the reverse lookup and image overlay capabilities, we can look for common features between different genes by clustering them together using any number of standard clustering techniques. We have chosen to use simple clustering techniques to perform two classes of clustering, a clustering of the genes and a clustering of vectors of gene expression for all of the genes that can be computed for each pixel.

4.3 Gene Clustering

The k-means clustering algorithm clusters n points in a d-dimensional space around k cluster centers [24]. We denote the data points x_n and establish a variable z_n which contains the value of the current cluster in which point x_n resides. The assignment to clusters stored in z is computed by the following equation:

$$z_n = \underset{j}{\operatorname{argmin}} ||x_n - \mu_j|| \tag{5}$$

where $||.||$ is an appropriate distance metric. The new cluster means μ are computed by:

$$\mu_i = \frac{\sum_n \delta(z_n, n) x_n}{\sum_n \delta(z_n, n)} \tag{6}$$

where δ is the Kronecker delta function. Thus, for a given a set of gene maps, x, with x_n indicating the nth gene map, and μ_j representing the jth cluster, we run the k-means clustering as in Equations 5 and 6. In our experiments, we found that using normalized cross correlation (NCC) produced better results than using L_2 distance metric. Some representative clusters are shown in Figure 9.

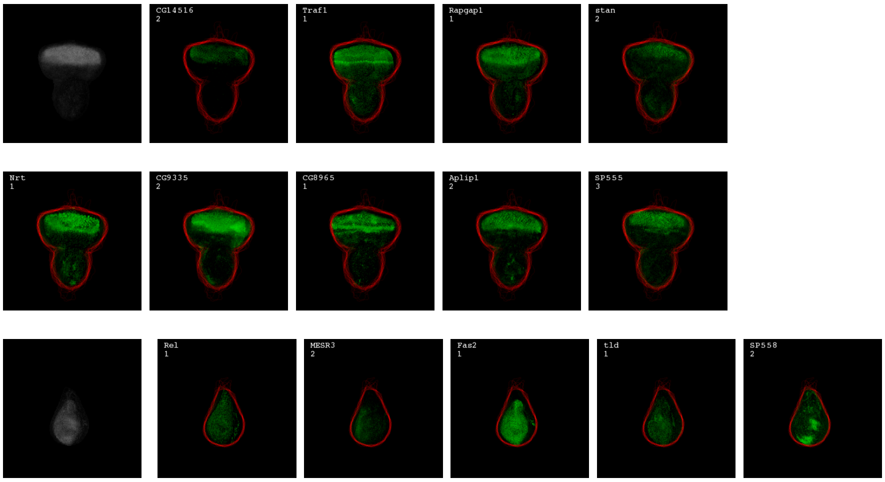


Fig. 9. Representative clusters from gene clustering. The first image in each set contains a cluster center from an eye/antenna disc cluster (top and middle rows) and a haltere disc cluster (bottom row). Other images are cluster members for the given cluster centers: CG14516, Trafl, Rapgap1, stan, Nrt, CG9335, CG8965, Aplip1, and SP555 for the eye/antenna cluster; Rel, MESR3, Fas2, tld, and SP558 for the haltere cluster.

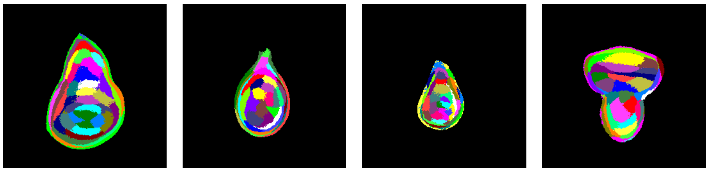


Fig. 10. Wing, leg, haltere and eye/antenna pixel clusters with 32 clusters (color-coded) in each image. This figure shows which pixels in the imaginal disc are similar to each other based on the gene expression profiles across all the genes measured in that particular disc.

4.4 Pixel Clustering

Given a set of gene maps, we want to cluster the pixels based on a vector of gene expression across all of the genes at each pixel. Given the set of all pixels P and $p \in P$, we want to cluster the x^p vectors. We refer to the cluster centers as π_j. To cluster the pixels, instead of the genes, we use the k-means clustering equations as follows:

$$z_n = \underset{j \in \{1,\ldots,P\}}{\operatorname{argmin}} ||x^p - \pi_j|| \tag{7}$$

where $||.||$ is an appropriate distance metric for comparing the vectors, in our experiments we used the L_2 distance. The new cluster means π are computed by:

$$\pi_i = \frac{\sum_p \delta(z_p, p) x^p}{\sum_p \delta(z_p, p)} \tag{8}$$

The results of pixel clustering of wing, leg, haltere and eye/antenna imaginal discs with 32 clusters are shown in Figure 10.

5 Discussion

In this paper, we have presented a method of generating a large number of spatial patterns of gene expression in *Drosophila melanogaster* imaginal discs, and for using generative models of shapes learned from all of the data, rather than using a single exemplar, as a global model of the shape of interest, to which a set of patterns can be aligned. We have presented methods for automatically detecting an imaginal disc in an image and for aligning this image to the learned shape model, for automatically scoring these images for stain, and for registering these stain patterns to the global model. Our representation of stain patterns as a quantitative measure of gene expression, aligned to a global model, enables us to efficiently cluster both the patterns of the genes themselves, and the regions of the tissues, as represented by the pixels in the global model. Finally, we have developed a reverse-lookup procedure, that enables us to take a new image, stained for a gene of interest, and to search our database of patterns to find genes with similar spatial patterns of gene expression.

Using our methods, we have determined the patterns of over 130 genes in some or all of the four largest and most well-characterized imaginal disc types, the wing, leg, haltere and eye/antenna discs. Our parallel automated alignment and feature extraction method works adequately over 85% of the time, but may fail when presented with images containing multiple imaginal discs or substantial amounts of biological clutter.

Our method of pixel clustering represents a novel way of viewing spatial gene expression data. By clustering the pixels, we are able to identify regions of the discs that are, on the basis of their gene expression profiles of on the order of 100 genes, more similar to the other pixels in that region than to the other regions. The pixel clusters identified are reminiscent of the zones of development identified in the classical imaginal disc fate maps in the literature. We look forward to exploring the relationships between these clusters and genetically-identified regions of the discs for which the eventual fates of the cells have been determined.

Future work may include extending the procedure to be able to identify multiple discs in a single image, to be more robust to biological noise such as the presence of trachea or other larval material in the image, and to use hierarchical clustering methods for the gene and pixel clustering.

Acknowledgments

We thank Amy Beaton, Ben Berman and Michael Eisen for helpful discussions and Pavel Tomancak for development of the BDGP in situ image tracking database. We thank David Forsyth and Erik Learned-Miller for helpful discussions regarding Congealing.

References

[1] M. Adams, *et al.*, *Science* **287**, 2185 (2000).
[2] S. Misra, *et al.*, *Genome Biol* **3**, 1 (2002).
[3] M. N. Arbeitman, *et al.*, *Science* **297**, 2270 (2002).
[4] P. Tomancak, *et al.*, *Genome Biology* **3**, 1 (2002).
[5] R. Lipshutz, S. Fodor, T. Gingeras, D. Lockhart, *Nat Genet* **21**, 20 (1999).
[6] M. B. Eisen, P. O. Brown, *Methods Enzymol* **303**, 179 (1999).
[7] G. Kamberova, S. Shah, *DNA Array Image Analysis – Nuts and Bolts* (DNA Press LLC, 2002).
[8] A. N. Jain, *et al.*, *Genome Res* **12**, 325 (2002).
[9] K. P. White, S. A. Rifkin, P. Hurban, D. S. Hogness, *Science* **286**, 2179 (1999).
[10] A. Klebes, B. Biehs, F. Cifuentes, T. Kornberg, *Genome Biol* **3**, RESEARCH0038 (2002).
[11] M. J. Butler, *et al.*, *Development* **130**, 659 (2003).
[12] O. Eugene, A. Yund, J. W. Fristrom, *Tissue Culture Association Manual* **5**, 1055 (1979).
[13] E. Miller, N. Matsakis, P. Viola, *IEEE Conference on Computer Vision and Pattern Recognition* (2000), pp. 464–471.
[14] E. Miller, Learning from One Example in Machine Vision by Sharing Probability Densities, Ph.D. thesis, Massachusetts Institute of Technology (2002).
[15] P. Ahammad, C. L. Harmon, A. Hammonds, S. S. Sastry, G. M. Rubin, *CVPR (2)* (IEEE Computer Society, Washington, DC, USA, 2005), pp. 755–760.
[16] H. G. Barrow, J. M. Tenenbaum, R. C. Boles, H. C. Wolf, *IJCAI* (1977), pp. 659–663.
[17] G. Borgefors, *IEEE Transactions on Pattern Analysis and Machine Intelligience* **10**, 849 (1988). Published.
[18] R. C. Gonzalez, R. E. Woods, *Digital Image Processing* (Prentice Hall, 2002), second edn.
[19] P. Soille, *Morphological Image Analysis* (Springer-Verlag, Berlin, Heidelberg, New York, 1999).
[20] J. Lewis, Fast normalized cross-correlation (1995).
[21] J. W. O'Neill, E. Bier, *Biotechniques* **17**, 870, 874 (1994).
[22] G. Hauptmann, *Methods* **23**, 359 (2001).
[23] D. Kosman, *et al.*, *Science* **305**, 846 (2004).
[24] R. O. Duda, P. E. Hart, D. G. Stork, *Pattern Classification* (John Wiley and Sons, 2001), second edn.

A Appendix

A.1 Foreground Feature Extraction

Algorithm A.1: EXTRACT-FOREGROUND-FEATURES(*img*)

```
$laplacian \leftarrow$ LAPLACIAN($img$)
$absLaplacian \leftarrow$ ABS($laplacian$)
$dilated \leftarrow$ DILATE($absLaplacian, SquareStructuringElement$)
$eroded \leftarrow$ ERODE($dilated, SquareStructuringElement$)
$foregroundFeatures \leftarrow$ DISTANCE-TRANSFORM($eroded$)
return ($foregroundFeatures$)
```

A.2 Aligning Target Image to Model

Algorithm A.2: ALIGN-TARGET-TO-MODEL(*target*, *model*)

```
$feat \leftarrow$ EXTRACTFOREGROUNDFEATURES($target$)
for each $\mathbf{x} \in initialConfigurationList$
    for $iteration \leftarrow 0$ to $maxIterations$
        $origScore \leftarrow$ NCC($feat, model$)
        for $x \in \mathbf{x}$
            $x_- \leftarrow x - stepSize$
            $stepDownImage \leftarrow$ AFFINETRANSFORMIMAGE($feat, x_-, \mathbf{x}$)
            $stepDownScore \leftarrow$ NCC($stepDownImage, model$)
            if $stepDownScore > origScore$
              then $x \leftarrow x_-$
            $x_+ \leftarrow x + stepSize$
            $stepUpImage \leftarrow$ AFFINETRANSFORMIMAGE($feat, x_+, \mathbf{x}$)
            $stepUpScore \leftarrow$ NCC($stepUpImage, model$)
            if $stepUpScore >$ MAX($origScore, stepDownScore$)
              then $x \leftarrow x_+$
            $\mathbf{x} \leftarrow$ UPDATE($x, \mathbf{x}$)
$alignedTarget \leftarrow$ AFFINETRANSFORMIMAGE($feat, \mathbf{x}$)
return ($alignedTarget$)
```

Author Index