Journal Subline

LNCS 4750

Transactions on Computational Science I

Marina L. Gavrilova · C. J. Kenneth Tan
Editors-in-Chief

Springer

Lecture Notes in Computer Science 4750

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Marina L. Gavrilova C.J. Kenneth Tan (Eds.)

Transactions on Computational Science I

Springer

Editors-in-Chief

Marina L. Gavrilova
University of Calgary
Department of Computer Science
2500 University Drive N.W.
Calgary, AB, T2N1N4, Canada
E-mail: marina@cpsc.ucalgary.ca

C.J. Kenneth Tan
OptimaNumerics Ltd.
Cathedral House
23-31 Waring Street
Belfast BT1 2DX, UK
E-mail: cjtan@optimanumerics.com

Library of Congress Control Number: 2008924861

CR Subject Classification (1998): F, D, C.2-3, G, E.1-2

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN	0302-9743 (Lecture Notes in Computer Science)
ISSN	1866-4733 (Transactions on Computational Science)
ISBN-10	3-540-79298-8 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-79298-7 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11530664 06/3180 5 4 3 2 1 0

LNCS Transactions on Computational Science

Computational science, an emerging and increasingly vital field, is now widely recognized as an integral part of scientific and technical investigations, affecting researchers and practitioners in areas ranging from aerospace and automotive research to biochemistry, electronics, geosciences, mathematics, and physics. Computer systems research and the exploitation of applied research naturally complement each other. The increased complexity of many challenges in computational science demands the use of supercomputing, parallel processing, sophisticated algorithms, and advanced system software and architecture. It is therefore invaluable to have input by systems research experts in applied computational science research.

Transactions on Computational Science focuses on original high-quality research in the realm of computational science in parallel and distributed environments, also encompassing the underlying theoretical foundations and the applications of large-scale computation. The journal offers practitioners and researchers the possibility to share computational techniques and solutions in this area, to identify new issues, and to shape future directions for research, and it enables industrial users to apply leading-edge, large-scale, high-performance computational methods.

In addition to addressing various research and application issues, the journal aims to present material that is validated – crucial to the application and advancement of the research conducted in academic and industrial settings. In this spirit, the journal focuses on publications that present results and computational techniques that are verifiable.

Scope

The scope of the journal includes, but is not limited to, the following computational methods and applications:

- Aeronautics and Aerospace
- Astrophysics
- Bioinformatics
- Climate and Weather Modeling
- Communication and Data Networks
- Compilers and Operating Systems
- Computer Graphics
- Computational Biology
- Computational Chemistry
- Computational Finance and Econometrics
- Computational Fluid Dynamics
- Computational Geometry

- Computational Number Theory
- Computational Physics
- Data Storage and Information Retrieval
- Data Mining and Data Warehousing
- Grid Computing
- Hardware/Software Co-design
- High-Energy Physics
- High-Performance Computing
- Numerical and Scientific Computing
- Parallel and Distributed Computing
- Reconfigurable Hardware
- Scientific Visualization
- Supercomputing
- System-on-Chip Design and Engineering

Preface

We would like to present, with great pleasure, the inaugural volume of a new scholarly journal, *Transactions on Computational Science*. This journal is part of the Springer series *Lecture Notes in Computer Science*, and is devoted to the gamut of computational science issues, from theoretical aspects to application-dependent studies and the validation of emerging technologies.

This new journal was envisioned and founded to represent the growing needs of computational science as an emerging and increasingly vital field, now widely recognized as an integral part of scientific and technical investigations. Its mission is to become a voice of the computational science community, addressing researchers and practitioners in areas ranging from aerospace to biochemistry, from electronics to geosciences, from mathematics to software architecture, presenting verifiable computational methods, findings, and solutions.

Transactions on Computational Science focuses on original high-quality research in the realm of computational science in parallel and distributed environments, encompassing facilitation of the theoretical foundations and the applications of large-scale computations to massive data processing. The Journal is intended as a forum for practitioners and researchers to share computational techniques and solutions in the area, to identify new issues and to shape future directions for research, while industrial users may apply techniques of leading-edge, large-scale, high-performance computational methods.

This inaugural volume is devoted to computer systems research and the application of such research, which naturally complement each other. In this spirit, the volume is divided into two parts, with the first devoted to core computational science issues faced by researchers and industries today, and the second focusing on the development of novel computational techniques that are versatile and verifiable in a wide range of applications.

Part 1 of this volume comprises five manuscripts, connected by a unifying theme: information systems design. Specifically, the presented articles can be categorized into the following groups:

- Data flow analysis
- Building fuzzy interference systems
- Multi-agent systems design
- Models for curve fitting
- Network map topology representation

These articles exemplify the analysis and exploration of complex computational models and data sets from various domains. They provide invaluable insights into the studied problems and offer convincing case studies and experimental analysis.

Part 2 is concerned with the specific computational science problems in the areas of data processing and their industrial applications. The four papers comprising this part present original research in the following areas:

- Missing value imputation techniques in data mining
- Normalization techniques for electrocardiogram data analysis
- A unified method for solving laminar forced convection problems
- A new versatile technique for solving non-linear stochastic differential equations

Each article provides an example of a concrete industrial application or a case study of the presented methodology to amplify the impact of the contribution.

Many scientists and institutions have contributed to the creation and the success of the computational science community. We are very thankful to everybody within that community who supported the idea of creating a new LNCS journal subline – the *Transactions on Computational Science*. We are certain that this very first issue will be followed by many others, reporting new developments in the computational science field. This issue would not have been possible without the great support of the Editorial Board members, and we would like to express our sincere thanks to all of them. We would also like to express our gratitude to the LNCS editorial staff of Springer, in particular Alfred Hofmann and Ursula Barth, who supported us at every stage of the project. Throughout preparation of this volume the Editors were supported by various research programs and funds, including NSERC funding.

It is our hope that this fine collection of articles will be a valuable resource for *Transactions on Computational Science* readers and will stimulate further research into the vibrant area of computational science.

March 2008

Marina L. Gavrilova
C.J. Kenneth Tan

LNCS Transactions on Computational Science – Editorial Board

Table of Contents

Part 1 - Information Systems Design

Part 2 - Data Processing and Industrial Applications

Formalization of Data Flow Computing and a Coinductive Approach to Verifying Flowware Synthesis

Phan Cong Vinh and Jonathan P. Bowen

London South Bank University
Centre for Applied Formal Methods, Institute for Computing Research
Faculty of BCIM, Borough Road, London SE1 0AA, UK
phanvc@ieee.org, jonathan.bowen@lsbu.ac.uk
http://www.cafm.lsbu.ac.uk/

Abstract. Reconfigurable computing refers to the notions of *configware* and *flowware*. Configware means structural programming, or programming in space to execute computation in space. Flowware means data-flow programming that schedules the data flow for output from or input to the configware architecture. In this paper, data flows of a synthesized computation are formalized. This means that data flow is specified as a behavioral stream function in stream calculus, which is used to underpin the semantics for Register Transfer Level (RTL) synthesis. A stream representation allows the use of coinductive principles in stream calculus. In particular, using the coinductive proof principle, we show that behavioral stream functions in the three-stage synthesis process (scheduling, register allocation and binding, allocation and binding of functional units) are always bisimilar regardless of changes in a scheduling, allocation or binding procedure. Our formalization makes pipelining possible, in which all functional units as well as registers of hardware resources are reused during different control steps (C-steps). Moreover, a coinductive approach to verifying flowware synthesis, which is independent of the heuristic during register allocating and binding step, is proposed as a practical technique.

Keywords: Dynamic reconfiguration, Reconfigurable computing, Dynamically Programmable Field Array (DPGA), Flowware, Configware, Configware engineering, Embedded systems, Formal methods.

1 Introduction

Reconfigurable computing, also sometimes known as configurable computing, is defined by DeHon and Wawrzynek as:

> "*computing via a post-fabrication and spatially programmed connection of processing elements*" [6].

DeHon and Wawrzynek's definition lays stress on two of the principal factors of reconfigurable computing:

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 1–36, 2008.

- Post-fabrication features of architecture adapting to the flexibility of applications due to changing data flows and operation conditions.
- The spatial paradigm of computation called *Configware* versus the temporal paradigm of traditional microprocessors called *Software*.

The engineering and technology to implement the reconfigurable computing concept have become realistic with the development of *Field-Programmable Gate Array* (FPGA) technology. In fact, Lysaght has said that

> "*the importance of dynamically reconfigurable logic will increase, as FPGAs become larger*" [24].

A specific type of FPGA that can be programmed on-the-fly during system operation is *dynamically reconfigurable* FPGA, also called a *Dynamically Programmable Field Array* (DPGA). This originated in the TRANSIT project at MIT in connection with their DPGA prototype [5]. The DPGA can be programmed in a partial or full reconfiguration. André DeHon states:

> "*Dynamically Programmable Gate Arrays (DPGAs) are programmable arrays which allow the strategic reuse of limited resources. In so doing, DPGAs promise greater capacity, and in some cases higher performance, than conventional programmable device architectures where all array resources are dedicated to a single function for an entire operational epoch*" [5].

These engineering aspects of reconfigurable computing refer to the notion of R. Hartenstein's *configware engineering* [9, 10, 11, 12, 13, 14, 17], including *configware* and *flowware*, in which both configware and flowware codes are produced by a compilation process from a high-level programming language source as inputs. Configware codes are for configuration before runtime, and flowware codes execute data flows at runtime. Configware means structural programming, or programming in space (see Figure 1). That is why configware executes computation in space. Configware is also sometimes called Soft IP (Intellectual Property) Cores [15]. Flowware means data-flow programming that schedules the data flow for output from or input to the DPGA-based architecture using one or several data counters.

Figure 1 illustrates the difference between the configware approach and the standard von Neumann-style processing. For von Neumann-based computing over time, only one operation is executed per clock tick (also called clock cycle) on a microprocessor and its intermediate results are saved in registers. For computing in space, a sequence of operations is mapped from time to space and executed per C-step (also called *control step*) in parallel on a DPGA; its intermediate results are stored in buffers for another sequence of operations in the next C-step.

Configware is only supported by the computing paradigm based on data flow and not the computing paradigm based on von Neumann. Both flowware and configware are two different complementary approaches that can be combined in programming for computing systems based on data flow in a manner such

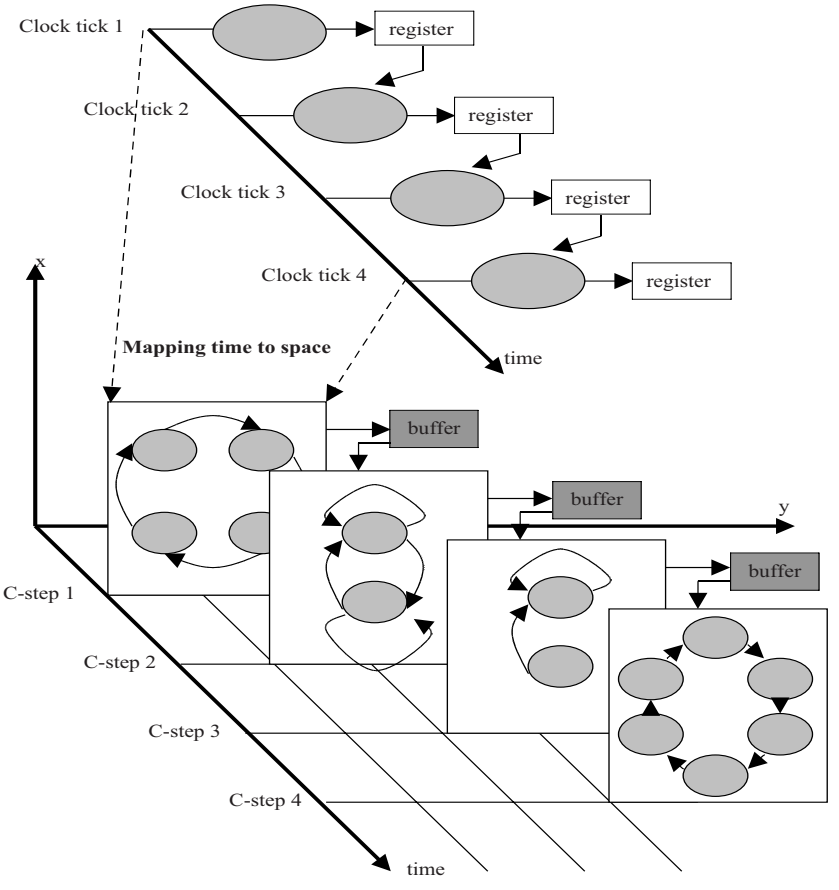
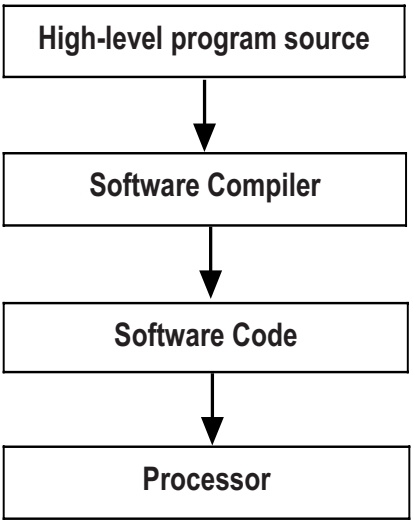


Fig. 1. Computation in space vs. time (based on [15])

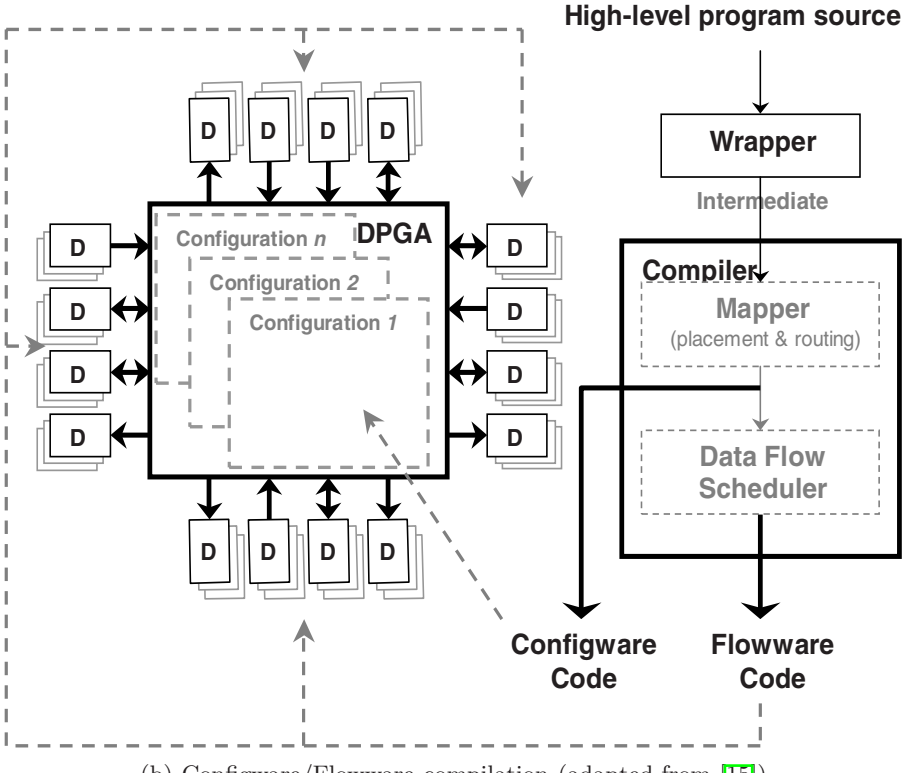
that the flowware determines which data item has to be connected with which DPGA port of which configuration at which time. Obviously, the working style is different from von Neumann computing systems based on programming the instruction streams from software. Instead of instruction streams with one or more program counters, data flows with data counters are programmed.

Figure 2(b) illustrates a Configware/Flowware Synthesis (CFS) process generating both configware and flowware from a high-level programming language source: configware for configuration before runtime and flowware to execute data flows at runtime. In software engineering, a software compiler generates only object code from a high-level programming language source; see Figure 2(a).

We view the CFS process as a derivation of the implementation by the application of transformations within a logical calculus. The end product is therefore not only the implementation but also the mathematical proof that the implementation satisfies the specification.



(a) Software compilation



(b) Configware/Flowware compilation (adapted from [15])

Fig. 2. Configware engineering vs. Software engineering

In conventional synthesis, hardware is represented by arbitrary data structures and there are no restrictions as to the transformations on these data structures. In CFS, hardware is represented by means of terms and formulae, and only correctness-preserving logical transformations are allowed. Restricting synthesis to only correctness-preserving logical transformations guarantees the correctness of the synthesis procedure in an implicit manner. In contrast to conventional synthesis, the result is not only a hardware implementation but also a proof of its correctness with respect to the specification.

Unfortunately, DPGA systems supporting reconfigurable computing have still not become widely commercialized because of high hardware costs associated with the required large configuration memory. Only a few such systems have been built for research. Hence to improve reconfiguration at lower hardware cost, Xilinx has developed several commercial DPGA families with functionality that is able to do fast partial reconfiguration from off-chip memory resources [32].

Dynamic reconfiguration is likely to become increasingly important in DPGA systems to allow greater flexibility. However, the added complexity can create additional opportunities for errors in such systems. Particularly in mobile, safety-critical or security-critical applications, it is important to use formal techniques to avoid the introduction of errors [4, 18, 33, 34, 35, 36, 37, 38, 39]. In such cases, it is hoped that a rigorous approach such as that presented in this paper will be applied.

We show that computing based on data flow of flowware in configware engineering [15] is considered as a behavioral function in stream calculus [27, 28, 29, 30], which supports the semantics for graphical networks at the Register Transfer Level (RTL) [2]; coinduction [21, 22, 25, 26] is used to compute such behavioral functions.

In particular, using the coinductive proof principle, we prove that behavioral functions (representing output data flows of a synthesized computation) in the three-stage synthesis process (including scheduling, register allocation and binding, allocation and binding of functional units) are always bisimilar [20, 31] regardless of internal changes in the three synthesis stages such as the changing of a proper scheduling, allocation or binding procedure.

The remainder of the paper is organized as follows. Section 2 briefs some related work and existing concepts of computing based on data flow and stream calculus. Section 3 formalizes computing based on data flow in stream calculus. A coinductive approach to verifying flowware synthesis is presented in section 4 and a short conclusion is given in section 5.

2 Preliminaries

In this section, some related work and existing concepts of computing based on data flow and stream calculus are presented for preliminary consideration.

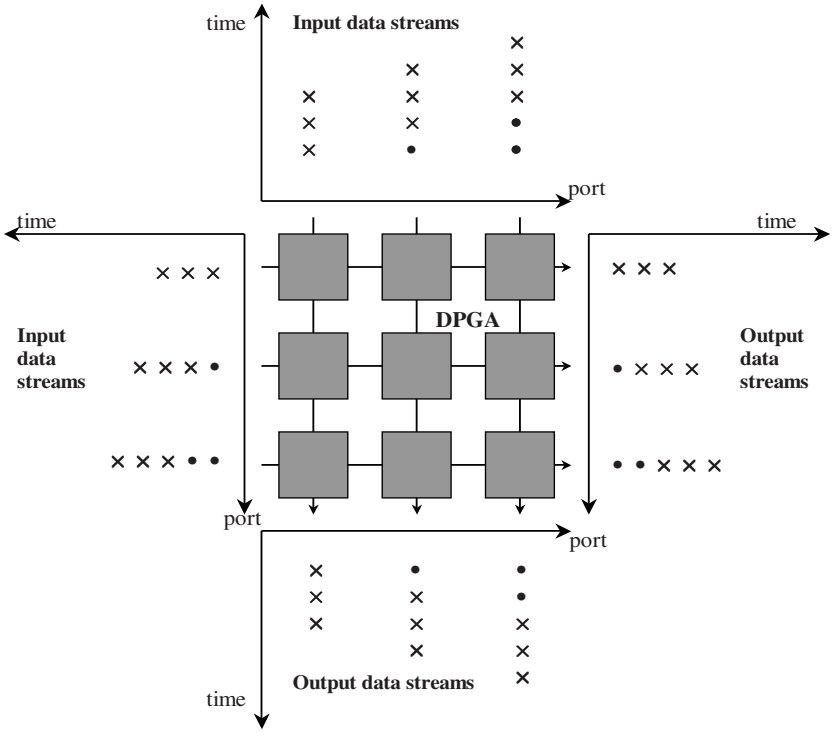


Fig. 3. Flowware (based on [15])

2.1 Computing Based on Data Flow

In computing based on data flow, data flows are programmed from flowware sources. Flowware determines which data item has to meet which DPGA port at which time [9,10,11,12,13,14,17] (see Figure 3).

We need to distinguish two different domains of programming in time: *instruction scheduling* and *data scheduling*. The programming code for von Neumann-like devices is an instruction schedule, compiled from software as in Figure 2(a). The programming code for resources like DPGAs is a data schedule that can be compiled from flowware code, determining which data item has to appear at which port at which time. Such data schedules manage the data flows. This is illustrated in Figure 3, showing a typical data flow notation introduced with DPGAs.

The DPGA-based architecture is vital for the flowware-based paradigm. This architecture allows the multiple data flows clocked into and out of such a pipe network, in a similar manner to the heart and blood stream. Its DPGAs do not have instruction sequencers. The mode of DPGA operation is transport-triggered by data items. If synchronization is done by handshake instead of clocking, a DPGA is sometimes called a *wavefront array* [15].

The traditional DPGA can be used only for applications with strictly regular data dependencies, because array synthesis methods use linear projections or algebraic methods resembling linear projections. Such synthesis methods yield only strictly uniform arrays with linear pipes. The Data Path Synthesis System (DPSS) [16], however, used simulated annealing instead (the mapper in Figure 2(b)), which removed the traditional application limitations, enabling the synthesis of DPGAs featuring any kind of non-uniform arrays with free form pipes like zigzag, spiral, completely irregular, and many others. This drastically improved flexibility is realized using runtime reconfigurable DPGAs.

The specification of data flows can be expressed by a flowware language. Data flows are created by executing flowware code on distributed memory arrays (denoted by the symbol D in Figure 2(b)) surrounding the DPGA to drive data flows from/to the DPGA.

Flowware may also be described by higher level flowware languages, which are similar to high-level software languages like C [1], for example SystemC [3,8]. These languages include control structures such as jumping and (nested) looping. The principal differences between software and flowware are that software makes reference to only a single program counter, but flowware refers to several data counters. As a result, flowware also includes parallel looping as an important control structure, which is not supported by software.

2.2 Basic Stream Calculus

During our formalization, the Rutten's stream calculus [27] and the following useful operators are used as *addition*, *convolution product*, *inverse*, *copy* and *register*, which are detailed in this subsection.

In stream calculus [27], a stream σ is defined by the following function:

$$\sigma : \mathbb{N} \longrightarrow \mathbb{R}$$

That is:

$$\sigma = (\sigma(0), \sigma(1), \sigma(2), \dots)$$

For most formal definitions of a stream, Rutten uses the following notation of stream derivative for a stream definition. For any $n \geqslant 0$,

Differential equation	Initial value
$(\sigma^{(n)})' = \sigma^{(n+1)}$	$\sigma(0)$

The initial value of σ is defined as its first element σ_0, and the stream derivative, denoted by σ', is defined by $(\sigma^{(n)})' = \sigma^{(n+1)}$, for $n \geqslant 0$. In other words, the initial value and derivative equal the head and tail of σ, respectively. The behavior of a stream σ consists of two aspects: it allows for the observation of its initial value $\sigma(0)$; and it can make an evolution to the new stream σ', consisting of the original stream from which the first element has been removed. The initial value of σ', which is $\sigma'(0) = \sigma(1)$, can in its turn be observed, but note that we had to move from σ to σ' first in order to do so. Now a behavioral differential equation

defines a stream by specifying its initial value together with a description of its derivative, which tells us how to continue.

Let $\mathbb{R}^\omega = \{\sigma \mid \sigma : \mathbb{N} \longrightarrow \mathbb{R}\}$ be a set of streams of real numbers. Stream calculus introduces a number of constants and the operations of addition, (convolution) product, and inverse of streams. These constants and operations make of $\mathbb{R}^\omega$ a calculus with many important properties. In particular, it will be possible to compute solutions of linear systems of equations.

The real numbers can be considered as streams in the following manner. For every $r \in \mathbb{R}$, a stream $[r] \in \mathbb{R}^\omega$ is defined by function:

$$[\,] : r \in \mathbb{R} \longrightarrow [r] \in \mathbb{R}^\omega$$

That is:

$$[r] = (r, 0, 0, 0, \dots)$$

or

Differential equation	Initial value
$[r]' = [0]$	$[r](0) = r$

Addition. Given $\sigma = (\sigma(0), \sigma(1), \sigma(2), \dots) \in \mathbb{R}^\omega$ and $\tau = (\tau(0), \tau(1), \tau(2), \dots) \in \mathbb{R}^\omega$. Addition of two streams σ and τ is defined by

$$\sigma + \tau = (\sigma(0) + \tau(0), \sigma(1) + \tau(1), \sigma(2) + \tau(2), \dots)$$

or

Differential equation	Initial value
$(\sigma + \tau)' = \sigma' + \tau'$	$(\sigma + \tau)(0) = \sigma(0) + \tau(0)$

Note that $\sigma + \tau$ denotes an addition of two streams, but $\sigma(i) + \tau(i)$ denotes an addition of two real numbers.

This definition of addition allows us to add real numbers r to a stream σ, yielding:

$$[r] + \sigma = (r, 0, 0, 0, \dots) + (\sigma(0), \sigma(1), \sigma(2), \dots) = (r + \sigma(0), \sigma(1), \sigma(2), \dots)$$

Usually, $r + \sigma$ is used to denote $[r] + \sigma$. The context will always make clear whether the notation r has to be interpreted as a real number r or as the stream $[r]$.

Property 2.1. *For all $r, s \in \mathbb{R}$ and $\sigma, \tau, \rho \in \mathbb{R}^\omega$,*

$$[r] + [s] = [r + s] \tag{2.1}$$
$$\sigma + 0 = \sigma \tag{2.2}$$
$$\sigma + \tau = \tau + \sigma \tag{2.3}$$
$$\sigma + (\tau + \rho) = (\sigma + \tau) + \rho \tag{2.4}$$

(Convolution) Product. The product of two streams σ and $\tau \in \mathbb{R}^\omega$ means a convolution product defined by:

$$\sigma \times \tau = (\sigma(0).\tau(0), (\sigma(0).\tau(1)) + (\sigma(1).\tau(0)), (\sigma(0).\tau(2)) + (\sigma(1).\tau(1)) + (\sigma(2).\tau(0)), \dots)$$

That is, for any $n \geqslant 0$,

$$(\sigma \times \tau)(n) = \sum_{i=0}^{n} \sigma(i).\tau(n-i)$$

or

Differential equation	Initial value
$(\sigma \times \tau)' = (\sigma' \times \tau) + (\sigma(0) \times \tau')$	$(\sigma \times \tau)(0) = \sigma(0) \times \tau(0)$

Note that

- $\sigma \times \tau$ or $\sigma\tau$ denotes a (convolution) product of two streams
- $\sigma(i).\tau(j)$ denotes a multiplication of two real numbers
- $\sigma^0 = 1$ and $\sigma^{n+1} = \sigma \times \sigma^n$
- For $k \in \mathbb{N}$, $k \times \sigma = \sum_{i=0}^{k} \sigma$

This definition of sum allows us to multiply real numbers r with a stream σ, yielding:

$$[r] \times \sigma = (r, 0, 0, 0, \dots) \times (\sigma(0), \sigma(1), \sigma(2), \dots) = (r.\sigma(0), r.\sigma(1), r.\sigma(2), \dots)$$

Normally, $r \times \sigma$ is used to denote $[r] \times \sigma$ and the following convention is used:

$$-\sigma = -1 \times \sigma = [-1] \times \sigma = (-\sigma(0), -\sigma(1), -\sigma(2), \dots)$$

Property 2.2. *For all $r, s \in \mathbb{R}$ and $\sigma, \tau, \rho \in \mathbb{R}^\omega$,*

$$[r] \times [s] = [r.s] \tag{2.5}$$
$$0 \times \sigma = 0 \tag{2.6}$$
$$1 \times \sigma = \sigma \tag{2.7}$$
$$\sigma \times \tau = \tau \times \sigma \tag{2.8}$$
$$\sigma \times (\tau + \rho) = (\sigma \times \tau) + (\sigma \times \rho) \tag{2.9}$$
$$\sigma \times (\tau * \rho) = (\sigma \times \tau) * (\sigma \times \rho) \tag{2.10}$$
$$\sigma \times (\tau \times \rho) = (\sigma \times \tau) \times \rho \tag{2.11}$$

For convenience when denoting the stream formation $\sigma = (r_0, r_1, \dots, r_n, 0, 0, \dots)$ where $n \geqslant 0$ and $r_0, \dots, r_n \in \mathbb{R}$, the following important constant is used:

$$X = (0, 1, 0, 0, \dots)$$

or

Differential equation	Initial value
$X' = [1]$	$X(0) = 0$

In fact, using X the following stream operations are elegantly and expressively represented:

$$r \times X = (0, r, 0, 0, \dots) \tag{2.12}$$

$$X \times \sigma = (0, \sigma(0), \sigma(1), 0, 0, \dots) \tag{2.13}$$

$$X^n = (0, \dots {}_{n-2\ times} \dots, 0, 1, 0, 0, \dots) \tag{2.14}$$

$$\sum_{i=0}^{n} r_i X^i = (r_0, r_1, \dots, r_n, 0, 0, \dots) \tag{2.15}$$

This latter stream formation is called a *polynomial stream*. An expression of streams only composed of the constant streams is called a *closed form expression*.

Inverse. Given a stream σ with $\sigma(0) \neq 0$, a stream called the inverse of σ is denoted by σ^{-1} (or $\frac{1}{\sigma}$) such that $\sigma \times \sigma^{-1} = 1$. Formally, for every $\sigma \in \mathbb{R}^\omega$ with $\sigma(0) \neq 0$, σ^{-1} is a unique stream satisfying the following differential equation.

Differential equation	Initial value
$(\sigma^{-1})' = -\sigma(0)^{-1} \times (\sigma' \times \sigma^{-1})$	$(\sigma^{-1})(0) = \sigma(0)^{-1}$

For presentational reasons, the following notations are used

$$\sigma^{-n} \equiv (\sigma^{-1})^n \qquad \frac{\sigma}{\tau} \equiv \sigma \times \tau^{-1}$$

with the convention that $(\sigma^{-1})'$ can be written as

$$(\frac{1}{\sigma})' = \frac{-\sigma'}{\sigma(0) \times \sigma}$$

Property 2.3. *For all* $\sigma, \tau \in \mathbb{R}^\omega$,

$$\sigma \times \sigma^{-1} = 1 \tag{2.16}$$

$$\sigma^{-1} \times \sigma = 1 \tag{2.17}$$

$$(\sigma^{-1})^{-1} = \sigma \tag{2.18}$$

$$(\sigma \times \tau)^{-1} = \sigma^{-1} \times \tau^{-1} \tag{2.19}$$

Copy. For all $\sigma, \tau, \rho \in \mathbb{R}^\omega$, at any moment $n \geqslant 0$, the copy operation inputs the value $\sigma(n)$ and outputs two identical copies $\tau(n)$ and $\rho(n)$. It is graphically represented as in Figure 4(a).

$$\sigma(n) = \tau(n) = \rho(n), \text{ for all } n \geqslant 0$$

or

$$\sigma = \tau = \rho$$

or

Differential equation	Initial value
$\sigma' = \tau' = \rho'$	$\sigma(0) = \tau(0) = \rho(0)$

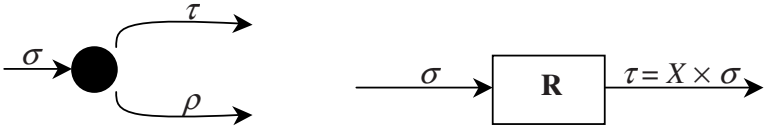
Register. Given $\sigma = (\sigma(0), \sigma(1), \sigma(2), \dots) \in \mathbb{R}^\omega$ and $\tau = (\tau(0), \tau(1), \tau(2), \dots) \in \mathbb{R}^\omega$, a register between streams σ and τ is defined by

$$\tau = X \times \sigma = (0, \sigma(0), \sigma(1), \sigma(2), \dots)$$

or

Differential equation	Initial value
$(\tau)' = \sigma$	$\tau(0) = 0$

The register operation is graphically represented as in Figure 4(b). It can be viewed as consisting of a one-place memory cell that initially contains the value 0. The operation starts its activity, at time 0, by outputting its value $\tau(0) = 0$, while it simultaneously inputs the value $\sigma(0)$ stored in the memory cell. At any time moment $n \geqslant 1$, the value $\tau(n) = \sigma(n-1)$ is output and the value $\sigma(n)$ is input and stored. For this reason, the register operation is sometimes also called a unit delay.



(a) Copy (b) Register

Fig. 4. Copy and Register operators (adapted from [28])

Rutten has proved the following statement, called the fundamental theorem of stream calculus in [27], which is used in all our calculations.

Theorem 2.4 (Fundamental Theorem of stream calculus). *For all streams* $\sigma \in \mathbb{R}^\omega$,

$$\sigma = \sigma(0) + (X \times \sigma')$$

3 Formalization of Computing Based on Data Flow in Stream Calculus

Representing data computing in stream calculus allows the formalization of the powerful concept of pipelining, in which all hardware resources, including functional units as well as registers, are reused during different C-steps (Control steps) of computing. Our formalization is described below at the *Register Transfer Level* (RTL) of reconfigurable computing in a pipelining style.

Computing based on data flow of flowware is considered as the function from $\mathbb{R}^\omega$ to $\mathbb{R}^\omega$ in stream calculus, which supports the semantics for graphical networks of RTL. Such networks can be viewed as implementations of computing based on data flow.

3.1 Some Definitions

For developing the formalization, we define two more stream operators named *multiplication* (*) and *assignment* (:=) as follows:

Multiplication. Given $\sigma = (\sigma(0), \sigma(1), \sigma(2), \dots) \in \mathbb{R}^\omega$ and $\tau = (\tau(0), \tau(1), \tau(2), \dots) \in \mathbb{R}^\omega$, multiplication of two streams σ and τ is defined by

$$\sigma * \tau = (\sigma(0).\tau(0), \sigma(1).\tau(1), \sigma(2).\tau(2), \dots)$$

or

Differential equation	Initial value
$(\sigma * \tau)' = \sigma' * \tau'$	$(\sigma * \tau)(0) = \sigma(0).\tau(0)$

Note that $\sigma * \tau$ denotes a multiplication of two streams, but $\sigma(i).\tau(i)$ is used for a multiplication of two real numbers.

This definition of multiplication allows the multiplication of a real number r with a stream σ, yielding:

$$[r] * \sigma = (r, 0, 0, 0, \dots) * (\sigma(0), \sigma(1), \sigma(2), \dots) = (r.\sigma(0), 0, 0, \dots)$$

Usually, $r*\sigma$ simply denotes $[r]*\sigma$. The context will always make it clear whether the notation r has to be interpreted as the real number r or as the stream $[r]$.

Property 3.1. *For all $r, s \in \mathbb{R}$ and $\sigma, \tau, \rho \in \mathbb{R}^\omega$,*

$$[r] * [s] = [r.s] \tag{3.1}$$
$$\sigma * 0 = 0 \tag{3.2}$$
$$\sigma * 1 = 1 \tag{3.3}$$
$$\sigma * \tau = \tau * \sigma \tag{3.4}$$
$$\sigma * (\tau * \rho) = (\sigma * \tau) * \rho \tag{3.5}$$
$$\sigma * (\tau + \rho) = (\sigma * \tau) + (\sigma * \rho) \tag{3.6}$$

Assignment. Given a stream $\sigma = (\sigma(0), \sigma(1), \sigma(2), \dots) \in \mathbb{R}^\omega$ and a stream variable $\tau = (\tau(0), \tau(1), \tau(2), \dots) \in \mathbb{R}^\omega$, assigning a stream σ to τ is defined by:

$$\tau := \sigma$$

That is, for any $i \geqslant 0$, it puts the value of $\sigma(i)$ into $\tau(i)$ (*i.e.*, $\tau(i) := \sigma(i)$) or

Differential equation	Initial value
$(\tau := \sigma)' = \tau' := \sigma'$	$(\tau := \sigma)(0) = \tau(0) := \sigma(0)$

Property 3.2. *For all stream variables $\tau \in \mathbb{R}^\omega$,*

$$\tau := 0 = 0 \tag{3.7}$$
$$\tau := 1 = 1 \tag{3.8}$$

3.2 Register Transfer Level (RTL) without Looping

We illustrate the various steps of flowware synthesis via an example that maps an input triple (a, b, c) onto the output pair (x, y) as defined by the following pseudo-procedural description:

```
Procedure comp1(
  inputs: a,b,c: real;
  outputs: x,y: real)
begin
  x := ((a*b)*(c+(b+c))+((a*b)-(b+c)));
  y := ((a*b)*(c+(b+c))*((a*b)-(b+c)));
end
```

The procedure uses the basic binary operations $\boxdot \in \{+, -, *, :=\}$

$$\boxdot \; : \; x \in \mathbb{R} \times y \in \mathbb{R} \longrightarrow x \boxdot y \in \mathbb{R}$$

The data flow diagram that corresponds to the procedural code is given in Figure 5, in which the intermediate results are named explicitly p, q, r, s and t.

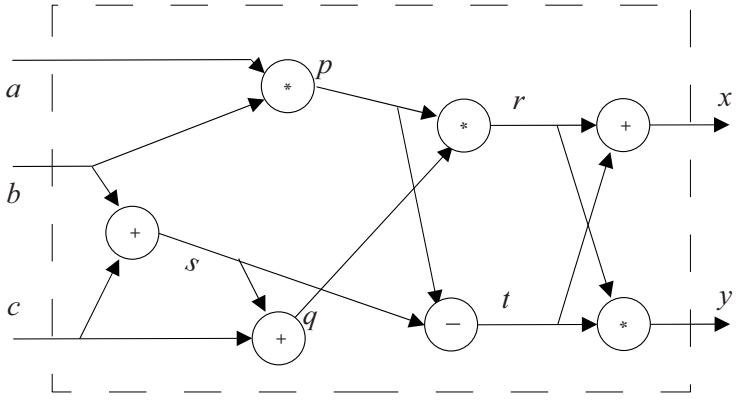


Fig. 5. Data flow diagram of `comp1`

The flowware synthesis process consists of the following steps: scheduling, register allocation and binding, allocation and binding of functional units [7].

Scheduling. Scheduling is a process of specifying the number of C-steps $k \geqslant 0$ required for computing the application and delivering each operation of the application to one particular C-step, numbered $0, 1, \ldots, k$. As in Figure 6, scheduling determines four C-steps $(0, 1, 2, 3)$ and assigns operations to the four C-steps as follows: one addition to C-step 0, one addition and one multiplication to C-step 1, one subtraction and one multiplication to C-step 2, and one addition and one

multiplication to C-step 3. Indeed, the schedule above is not unique and there may be another scheduling arrangement. As a result, two following major factors must be considered and traded off appropriately:

- How many C-steps are suitable?
- How many hardware requirements are needed for implementing the operations?

The above factors are important because the number of C-steps determines the speed of the implementation, whereas the hardware resources determine the size of the implementation. In addition, a scheduling process should consider the following two challenges:

- Constraints of hardware resources for the operations;
- Constraints of time for the execution.

In addressing each of these, suitable scheduling algorithms are required.

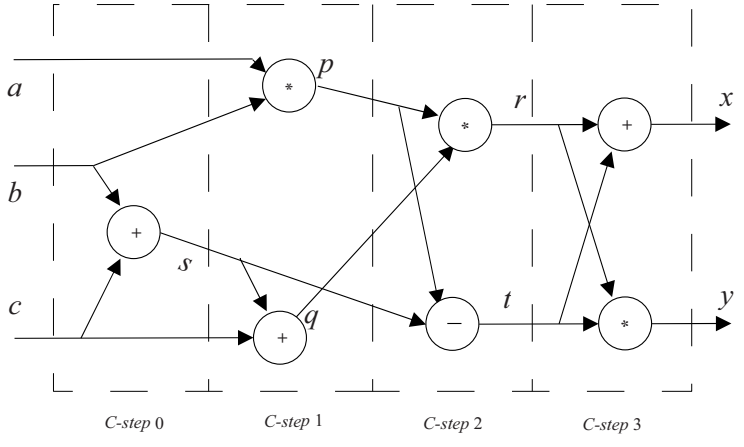


Fig. 6. Partitioning data flow of `comp1`

Register allocation and binding. This step of flowware synthesis performs two tasks:

- *Allocation:* specifying the number of registers needed for saving intermediate results between two C-steps.
- *Binding:* mapping the intermediate results to the registers for every C-step.

It is required that the number of registers must be equal to the maximum number of intermediate results between two C-steps. As in Figure 7, we need four registers numbered $1, 2, 3, 4$ for allocating and binding between C-steps 0 and 1; the registers 1, 2 and 3 are reused between C-steps 1 and 2; between C-steps 2 and 3, the registers 1 and 2 are reused again.

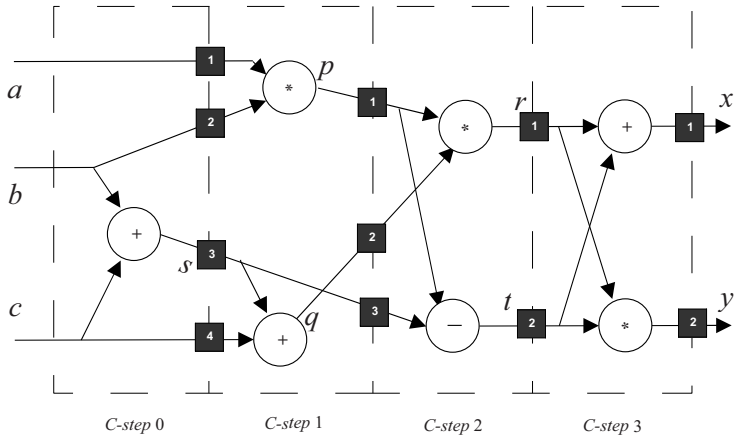


Fig. 7. Register allocation and binding of `comp1`

Register allocation and binding have an impact on utilization of hardware resources. In other words, the better that registers can be allocated and bound, the better that extra hardware resources are avoided.

Now, using stream calculus mentioned in subsection 2.2, we formalize data flow computing at the Register Transfer Level (RTL) of reconfigurable computing in a pipelining style. Formally, computing based on data flow is considered as the function from $\mathbb{R}^\omega$ to $\mathbb{R}^\omega$ in stream calculus.

For the data flows $b, c \in \mathbb{R}^\omega$, the function $s : \mathbb{R}^\omega \longrightarrow \mathbb{R}^\omega$ determines a computation based on b and c (see C-step 0 in Figure 7) to yield:

$$s = b + c \tag{3.9}$$

For the data flows $a, b, s \in \mathbb{R}^\omega$, two functions $p, q : \mathbb{R}^\omega \longrightarrow \mathbb{R}^\omega$ determine the computations based on a, b and s (see C-step 1 in Figure 7) to yield:

$$\begin{aligned} p &= X \times a * X \times b \\ &= X \times (a * b) \end{aligned} \tag{3.10}$$

$$\begin{aligned} q &= X \times s + X \times c \\ &= X \times (s + c) \end{aligned} \tag{3.11}$$

In a similar way, based on the data flows $p, q \in \mathbb{R}^\omega$, two functions $r, t : \mathbb{R}^\omega \longrightarrow \mathbb{R}^\omega$ are determined by (see C-step 2 in Figure 7):

$$\begin{aligned} r &= X \times p * X \times q \\ &= X \times (p * q) \end{aligned} \tag{3.12}$$

$$t = X \times p - X^2 \times s \tag{3.13}$$

Therefore, two output functions $x, y : \mathbb{R}^\omega \longrightarrow \mathbb{R}^\omega$ (see C-step 3 in Figure 7) yield:

$$\begin{aligned} x &= X \times t + X \times r \\ &= X \times (t + r) \\ &\text{by (3.12) and (3.13)} \\ &= X \times ((X \times p - X^2 \times s) + (X \times p * X \times q)) \\ &\text{by (3.10) and (3.11)} \\ &= X \times ((X^2 \times (a * b) - X^2 \times (b + c)) + \\ &\quad (X^2 \times (a * b) * X^2 \times (s + c))) \\ &\text{by (3.9)} \\ &= X \times ((X^2 \times (a * b) - X^2 \times (b + c)) + \\ &\quad (X^2 \times (a * b) * X^2 \times (b + c + c))) \\ &= X^3 \times (((a * b) - (b + c)) + ((a * b) * (b + c + c))) \\ &= X^3 \times (((a * b) - (b + c)) + ((a * b) * (b + 2c))) \end{aligned} \tag{3.14}$$

and

$$\begin{aligned} y &= X \times t * X \times r \\ &= X \times (t * r) \\ &\text{by (3.12) and (3.13)} \\ &= X \times ((X \times p - X^2 \times s) * (X \times p * X \times q)) \\ &\text{by (3.10) and (3.11)} \\ &= X \times ((X^2 \times (a * b) - X^2 \times (b + c)) * \\ &\quad (X^2 \times (a * b) * X^2 \times (s + c))) \\ &\text{by (3.9)} \\ &= X \times ((X^2 \times (a * b) - X^2 \times (b + c)) * \\ &\quad (X^2 \times (a * b) * X^2 \times (b + c + c))) \\ &= X^3 \times (((a * b) - (b + c)) * ((a * b) * (b + c + c))) \\ &= X^3 \times (((a * b) - (b + c)) * ((a * b) * (b + 2c))) \end{aligned} \tag{3.15}$$

Let $\sigma = a * b - (b + c) + a * b * (b + 2c)$ and $\tau = a * b - (b + c) * (a * b * (b + 2c))$, hence (3.14) and (3.15) yield output streams $x = X^3 \times \sigma$ and $y = X^3 \times \tau$. The following behavioral differential equations describes the two output streams x and y:

Differential equation	Initial value
$x^{(3)} = \sigma$	$x(0) = x(1) = x(2) = 0$
$y^{(3)} = \tau$	$y(0) = y(1) = y(2) = 0$

The visual stream representation in Figure 8 is also considered as a stream graph of RTL in Figure 7.

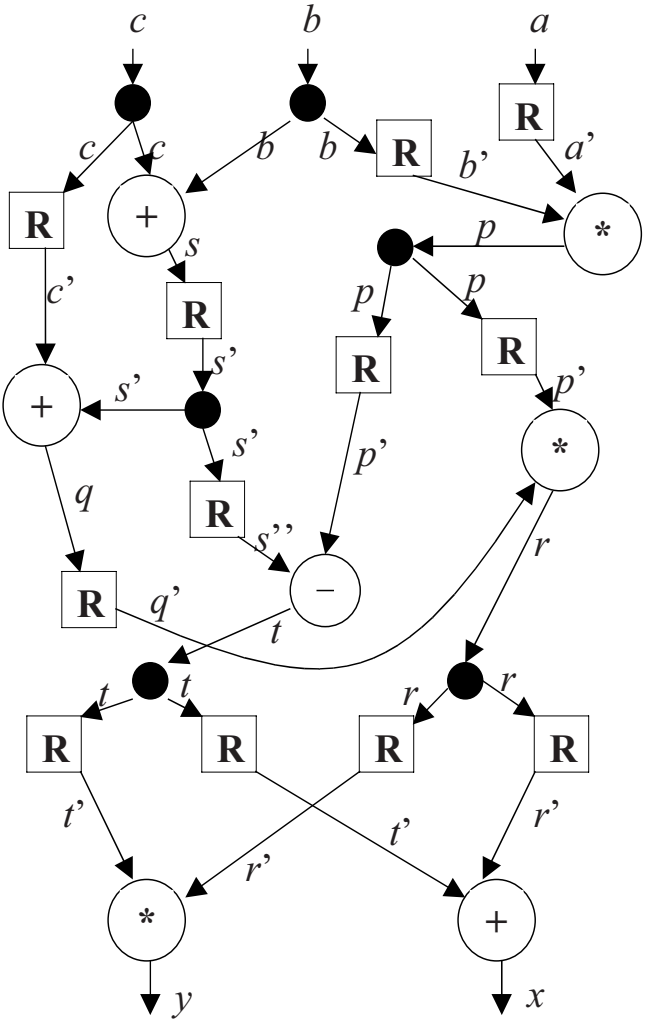


Fig. 8. Stream representation for RTL of `comp1`

Generally, if our computing is scheduled in n C-steps, for $n \geqslant 0$, then $x = X^n \times \sigma$ and $y = X^n \times \tau$ and

Differential equation	Initial value
$x^{(n)} = \sigma$	$x^{(i)}(0) = 0, 0 \leqslant i \leqslant n-1$
$y^{(n)} = \tau$	$y^{(i)}(0) = 0, 0 \leqslant i \leqslant n-1$

A bisimulation on $\mathbb{R}^\omega$ is a relation $B \subseteq \mathbb{R}^\omega \times \mathbb{R}^\omega$ such that, for all α and β in $\mathbb{R}^\omega$, if $\alpha\ B\ \beta$ then $\alpha(0) = \beta(0)$ and $\alpha'\ B\ \beta'$.

Proposition 3.3 (Bisimulation). *For $n \geqslant 0$, all output data flows $x = X^{(n)} \times \sigma$ (or $y = X^{(n)} \times \tau$) are bisimulation in pairs.*

Proof. To show this, we justify that there exists a bisimulation relation $B \subseteq \mathbb{R}^\omega \times \mathbb{R}^\omega$ with $\langle X^{(i)} \times \sigma, X^{(j)} \times \sigma \rangle \in B,\ 0 \leqslant i, j \leqslant n$. Such a relation B can be constructed in stages, by computing the respective derivatives of both $u = X^{(i)} \times \sigma$ and $v = X^{(j)} \times \sigma$ step by step. The pairs to be included in B are the following

$$\{\langle u_{i+k}, v_{j+k} \rangle \mid 0 \leqslant i, j \leqslant n, k \geqslant 0\}$$

Corollary 3.4 (Coinduction). *For all output data flows $X^{(n)} \times \sigma$ in $\mathbb{R}^\omega$ and $0 \leqslant i, j \leqslant n$, $X^{(i)} \times \sigma = X^{(j)} \times \sigma$.*

Proof. Consider two output data flows $u = X^{(i)} \times \sigma$ and $v = X^{(j)} \times \sigma$. From proposition 3.3, $B \subseteq \mathbb{R}^\omega \times \mathbb{R}^\omega$ is a bisimulation on $\mathbb{R}^\omega$ containing the pair $\langle u_i, v_j \rangle$. It follows by induction on k that $\langle u_{i+k}, v_{j+k} \rangle \in B$, for $k \geqslant 0$, because B is a bisimulation. This implies that $u^{(i+k)}(0) = v^{(j+k)}(0)$, this proving that $u = v$.

Function unit allocation and binding. A Functional Unit (FU) executes the following two tasks:

- *Allocation*: determining the operators for implementing the operations of each C-step;
- *Binding*: implementing the operations of the data flow graph.

The FUs are often built in a library describing the mapping between its functions and the relevant operations. The FUs can be implementations of single-purpose operations or multiple-purpose operations with control input signals for selecting the desired operation. As in Figure 9, three single-purpose FUs are required to implement three operations $+$, $-$ and $*$ in four C-steps. Specifically, in C-step 0 one FU is just for the implementation of the addition $(+)$. In C-step 1, we need two FUs where one FU is for implementing the multiplication $(*)$ and one other FU is reused for the addition $(+)$. In C-step 2, we also need two FUs where one FU is reused for implementing the multiplication $(*)$ and one other FU for the subtraction $(-)$. Finally, two FUs are reused for implementing the addition $(+)$ and multiplication $(*)$ in C-step 3.

An alternative is that we just need two FUs in total including one single-purpose FU for implementing the multiplication $(*)$ and one multiple-purpose FU for the addition $(+)$ and subtraction $(-)$. In this way, the single-purpose FU is reused in the last three C-steps and multiple-purpose FU with control input signals $(0/1)$ for selecting the desired operation $(-/+)$ is present in all four C-steps, as in Figure 10. If the control signal is switched to 0, then the subtraction $(-)$ is executed otherwise the addition $(+)$.

As a consequence, the better that FUs can be allocated and bound, the better that extra hardware resources are avoided. In other words, FU allocation and binding have an influence on how to use hardware resources in the effective way.

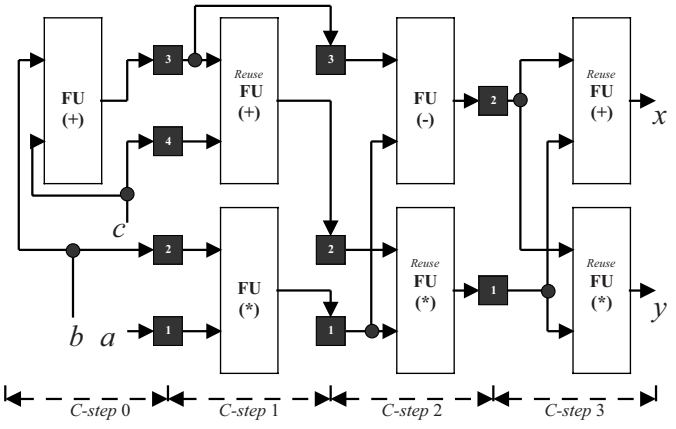


Fig. 9. FU allocation and binding of `comp1`: three single-purpose FUs

3.3 Register Transfer Level (RTL) Including Looping

Consider the synthesis of flowware via another example that calculates the elements of an array a:

$$a[0] := 1$$
$$a[1] := a[0]$$
$$a[2] := a[1] + a[0]$$

for $i \geqslant 3$, yielding

$$a[i] := a[i-1] + a[i-2] + a[i-3]$$

and maps the elements to output y as defined by the pseudo-procedural description as follows:

```
Procedure comp2(
  variables: a[]: real; i: num;
  outputs: y: real)
begin
    /* assigning the value 1 to the element a[0]
    of array a[] */
        a[0] := 1;
    /* outputting the value of a[0] */
        y := a[0];
    /* assigning the value of a[0] to the element a[1]
    of array a[] */
        a[1] := a[0];
    /* outputting the value of a[1] */
        y := a[1];
    /* assigning the value of expression a[1]+a[0]
```

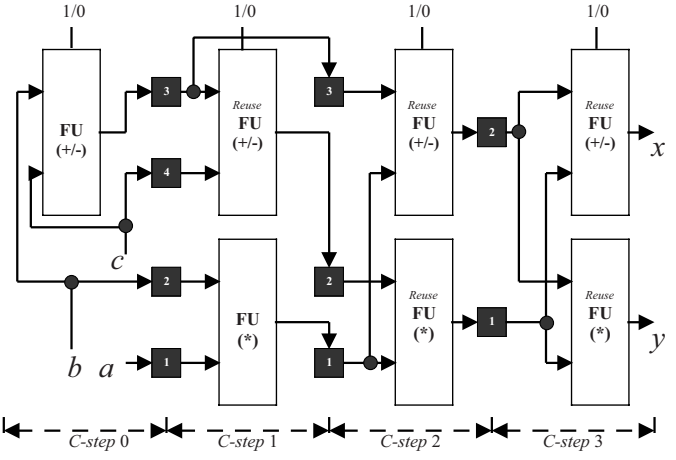


Fig. 10. FU allocation and binding of `comp1`: one single-purpose FU and one other multiple-purpose FU

```
    to the element a[2] of array a[] */
        a[2] := a[1]+a[0];
    /* outputting the value of a[2] */
        y := a[2];
    /* setting up the first value of looping index */
        i := 3;
  loop
    /* assigning the value of expression a[i-1]+a[i-2]+a[i-3]
    to the element a[i] of array a[] */
        a[i] := a[i-1]+a[i-2]+a[i-3];
    /* outputting the value of a[i] */
        y := a[i];
    /* increasing the looping index */
        i++;
  until unavailable
end
```

Note that each text in `/*...*/` of the above-mentioned procedure is used to explain the meaning of the next operation. As a result, the procedure `comp2` produces the following values to the elements of array $a[]$:

$$\begin{aligned} a[0] &= 1 \\ a[1] &= 1 \\ a[2] &= 1 + 1 (= 2) \\ a[3] &= 2 + 1 + 1 (= 4) \\ a[4] &= 4 + 2 + 1 (= 7) \\ \ldots &= \ldots \end{aligned}$$

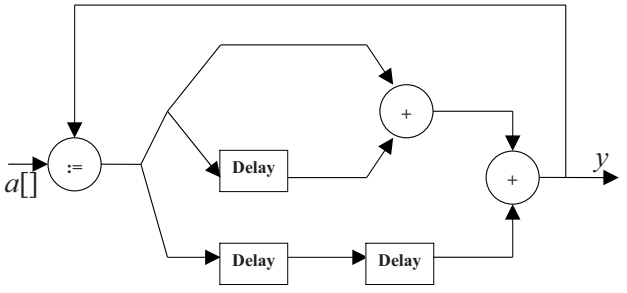


Fig. 11. Data flow diagram of `comp2`

The data flow diagram related to the procedural code `comp2` is given in Figure 11, which includes two types of the operations: addition (+) and unit delay, which is considered as consisting of a one-place memory cell containing the initial value 0. At the first moment, the unit delay starts its performance by outputting its initial value 0, while it simultaneously inputs the next value for saving in the one-place memory cell. At any future moment $n \geqslant 1$, the $(n-1)^{th}$ value in memory cell is output and the n^{th} new value is input and stored.

As mentioned in section 3.2, the RTL synthesis process of flowware goes through the following steps: scheduling, register allocation and binding, FU allocation and binding.

Scheduling. In general, the scheduling is not unique because it is decided according to the speed of implementation (i.e., the number of C-steps) and the size of implementation (i.e., the hardware resources) required. In the application `comp2`, we consider following two potential schedules of operations: single C-step as in Figure 12 and double C-step scheduling as in Figure 13.

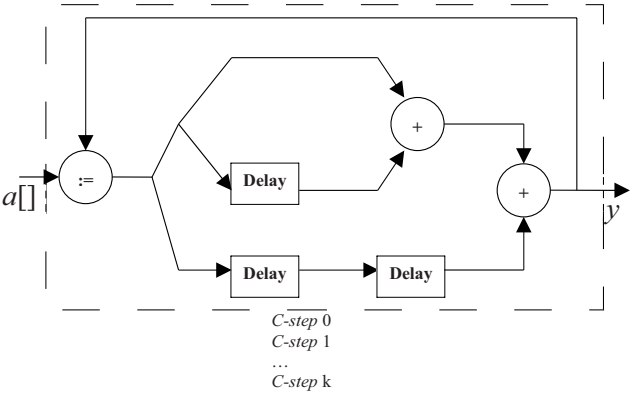


Fig. 12. Single C-step scheduling of `comp2`

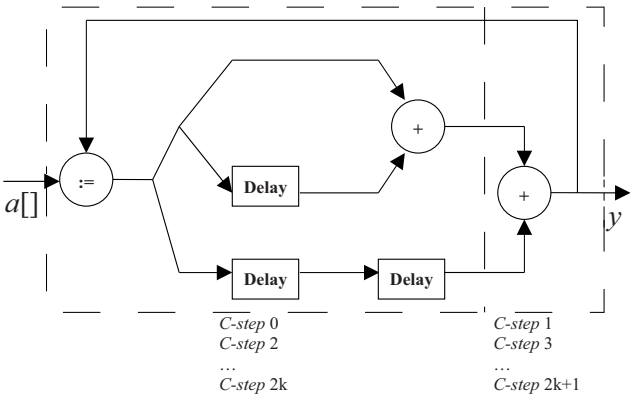


Fig. 13. Double C-step scheduling of `comp2`

Register allocation and binding. If the single C-step scheduling is used, it does need one extra register labeled 4 for allocation and binding at the end of each C-step; moreover three unit delays are semantically replaced by three registers named $1, 2, 3$ as in Figure 14. As a result, the following behavioral differential equation defines the behavior of `comp2`:

Differential equation	Initial values
$a^{(3)} = a^{(2)} + a^{(1)} + a$	$a(0) = 1$ $a^{(1)}(0) = a(0)$ $a^{(2)}(0) = a^{(1)}(0) + a(0)$

To solve this equation we use the fundamental theorem 2.4 on page 11 to obtain the following equations:

$$a = 1 + X \times a^{(1)} \tag{3.16}$$
$$a^{(1)} = 1 + X \times a^{(2)} \tag{3.17}$$
$$a^{(2)} = 2 + X \times a^{(3)} \tag{3.18}$$

By (3.18), then (3.17) and (3.16) are driven into

$$a^{(1)} = 1 + 2X + X^2 \times a^{(3)} \tag{3.19}$$
$$a = 1 + X + 2X^2 + X^3 \times a^{(3)} \tag{3.20}$$

By (3.18), (3.19) and (3.20), the following is yielded:

$$\begin{aligned} a^{(3)} &= a^{(2)} + a^{(1)} + a \\ &= (2 + X \times a^{(3)}) + (1 + 2X + X^2 \times a^{(3)}) + \\ &\quad (1 + X + 2X^2 + X^3 \times a^{(3)}) \end{aligned} \tag{3.21}$$

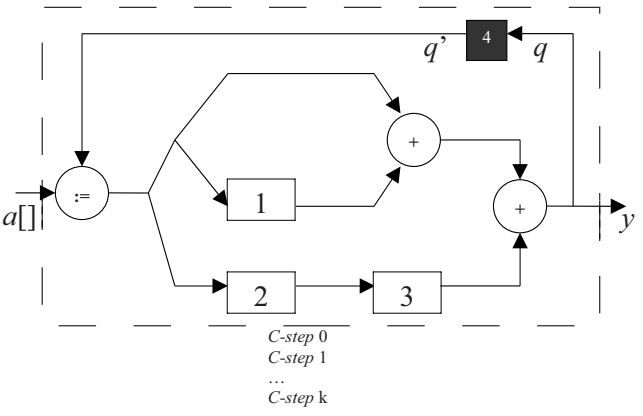


Fig. 14. Register allocation and binding for the single C-step RTL of `comp2`

Therefore, by transforming streams on (3.21)

$$a^{(3)} = \frac{4 + 3X + 2X^2}{1 - X - X^2 - X^3} \tag{3.22}$$

and, substituting (3.22) into (3.20)

$$a = 1 + X + 2X^2 + X^3\left(\frac{4 + 3X + 2X^2}{1 - X - X^2 - X^3}\right)$$

This yields the result in closed form

$$a = \frac{1}{1 - X - X^2 - X^3}$$

The visual stream representation in Figure 15 is also considered as an implementation of `comp2` and the following behavioral function defines the behavior of the output y:

Differential equation	Initial values
$y = a + X \times a + X^2 \times a$	$a(0) = 1$
	$a^{(1)}(0) = a(0)$
	$a^{(2)}(0) = a^{(1)}(0) + a(0)$

Substituting

$$a = \frac{1}{1 - X - X^2 - X^3}$$

into this differential equation yields the behavior of the output y in closed form:

$$y = \frac{1 + X + X^2}{1 - X - X^2 - X^3}$$

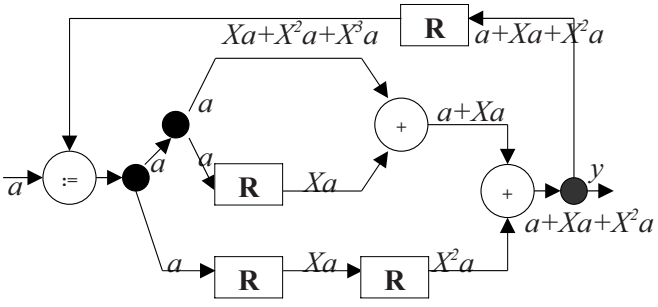


Fig. 15. Stream representation for the single C-step RTL of `comp2`

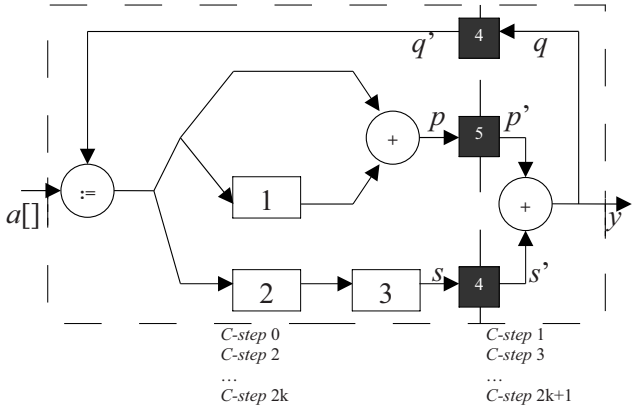


Fig. 16. Register allocation and binding for the double C-step RTL of `comp2`

If the double C-step scheduling is used, three unit delays are semantically replaced by three registers named $1, 2, 3$; moreover, two extra registers named 4 and 5 are allocated and bound at the end of each C-step labeled by an even number. Register 4 is reused at the end of each C-step labeled by an odd number as in Figure 16. As a result, the following behavioral differential equation defines the behavior of `comp2`:

Differential equation	Initial values
$a^{(3)} = X \times (a^{(2)} + a^{(1)} + a)$	$a(0) = 1$ $a^{(1)}(0) = a(0)$ $a^{(2)}(0) = a^{(1)}(0) + a(0)$

To solve this equation we apply the fundamental theorem 2.4 (see page 11) as in the single C-step case above. By (3.18), (3.19) and (3.20), the following is yielded:

$$\begin{aligned} a^{(3)} &= X \times (a^{(2)} + a^{(1)} + a) \\ &= X \times ((2 + X \times a^{(3)}) + (1 + 2X + X^2 \times a^{(3)}) + \\ &\quad (1 + X + 2X^2 + X^3 \times a^{(3)})) \end{aligned} \tag{3.23}$$

Therefore, by transforming streams on (3.23)

$$a^{(3)} = \frac{4X + 3X^2 + 2X^3}{1 - X^2 - X^3 - X^4} \tag{3.24}$$

and, substituting (3.24) into (3.20)

$$a = 1 + X + 2X^2 + X^3(\frac{4X + 3X^2 + 2X^3}{1 - X^2 - X^3 - X^4})$$

the result in closed form is:

$$a = \frac{1 + X + X^2 - 2X^3}{1 - X^2 - X^3 - X^4}$$

The visual stream representation in Figure 17 is also considered as an implementation of `comp2` and the following behavioral function defines the behavior of the output y.

Differential equation	Initial values
$y = X \times a + X^2 \times a + X^3 \times a$	$a(0) = 1$ $a^{(1)}(0) = a(0)$ $a^{(2)}(0) = a^{(1)}(0) + a(0)$

Replacing

$$a = \frac{1 + X + X^2 - 2X^3}{1 - X^2 - X^3 - X^4}$$

into this differential equation yields the behavior of the output y in closed form:

$$y = \frac{X(1 + X + X^2)(1 + X + X^2 - 2X^3)}{1 - X^2 - X^3 - X^4}$$

Proposition 3.5 (Bisimulation). *Two output data flows $a + X \times a + X^2 \times a$ and $X \times a + X^2 \times a + X^3 \times a$ are bisimilar each other.*

Proof. This follows from proposition 3.3 on page 18. In fact, there exists a bisimulation relation $B \subseteq \mathbb{R}^\omega \times \mathbb{R}^\omega$ with $\langle a + X \times a + X^2 \times a, X \times a + X^2 \times a + X^3 \times a \rangle \in B$. Such a relation B can be constructed in stages, by computing the respective derivatives of both $u = a + X \times a + X^2 \times a$ and $v = X \times a + X^2 \times a + X^3 \times a$ step by step. The pairs to be included in B are the following

$$\{\langle u_i, v_{i+1} \rangle \mid 0 \leqslant i \leqslant n\}$$

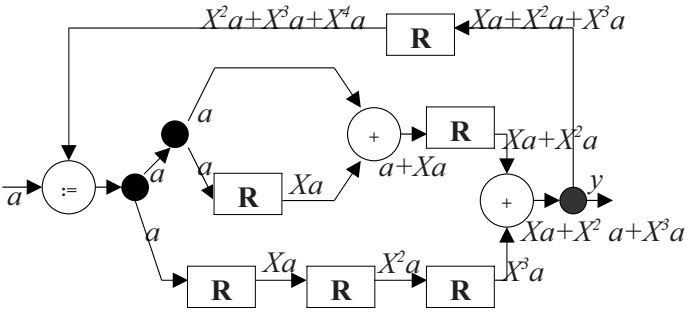


Fig. 17. Stream representation for the double C-step RTL of `comp2`

Corollary 3.6 (Coinduction). *If* $(a + X \times a + X^2 \times a) \sim (X \times a + X^2 \times a + X^3 \times a)$ *then* $(a + X \times a + X^2 \times a) = (X \times a + X^2 \times a + X^3 \times a)$

Proof. This is a result of corollary 3.4 on page 18. In fact, consider two output data flows $u = a + X \times a + X^2 \times a$ and $v = X \times a + X^2 \times a + X^3 \times a$. From proposition 3.5, $B \subseteq \mathbb{R}^\omega \times \mathbb{R}^\omega$ is a bisimulation on $\mathbb{R}^\omega$ containing the pair $\langle u_i, v_{i+1} \rangle$. It follows by induction on k that $\langle u_k, v_{k+1} \rangle \in B$, for $k \geqslant 0$, because B is a bisimulation. This implies that $u^{(k)}(0) = v^{(k+1)}(0)$, this proving that $u = v$.

FU allocation and binding. If single C-step scheduling is used, then two separate FUs are implementations of two additions (+) and one FU is implemented for the assignment (:=), with control input signals (0/1) to select suitable executions, as in Figure 18. If the control signal is switched to 0, then the input value of a goes through to the output, otherwise a is assigned by a second input value before being output.

If the double C-step scheduling is used, then we just need two FUs in total. In each C-step labeled by the even numbers, one FU is for the implementation

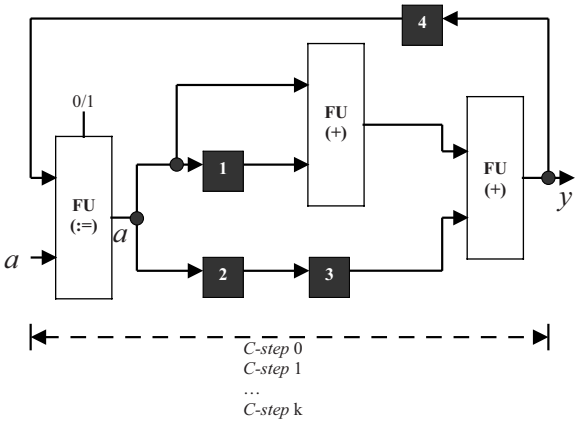


Fig. 18. FU allocation and binding for the single C-step RTL of `comp2`

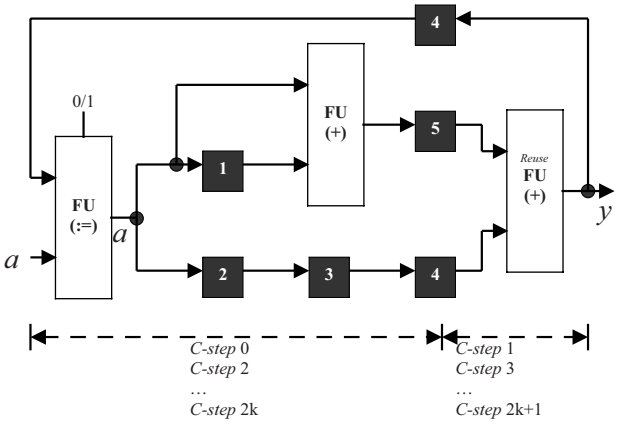


Fig. 19. FU allocation and binding for the double C-step RTL of `comp2`

of the addition (+) and one other is for implementing the assignment (:=). In each C-step labeled by the odd numbers, the FU for the addition is reused as in Figure 19.

4 A Coinductive Approach to Verifying Flowware Synthesis

4.1 Verifying Flowware Synthesis

Verified flowware synthesis is a verification technique that is aimed towards and integrated into synthesis steps. Beside the current post-synthesis verification techniques [23], we can also perform a verifying flowware synthesis to guarantee correct synthesis, as illustrated in Figure 20 for our coinductive approach.

The difference with post-synthesis verification is that the knowledge about *which* synthesis step has been performed is used to its advantage during verification. However, the knowledge about *how* the synthesis was performed is unknown. Thus our coinductive approach to verifying flowware synthesis is independent of the heuristic that is applied during scheduling, register allocating and binding or FU (Functional Unit) allocation and binding of a synthesis process.

In other words, our central goal of verifying flowware synthesis is to check that the synthesis does not violate global system properties or leads to inconsistent system configurations. Thus such an approach to verifying flowware synthesis avoids some disadvantages of postsynthesis verification such as being very time-consuming or even undecidable [37, 23, 40], and becomes a good choice to substitute for the current post-synthesis verification approaches.

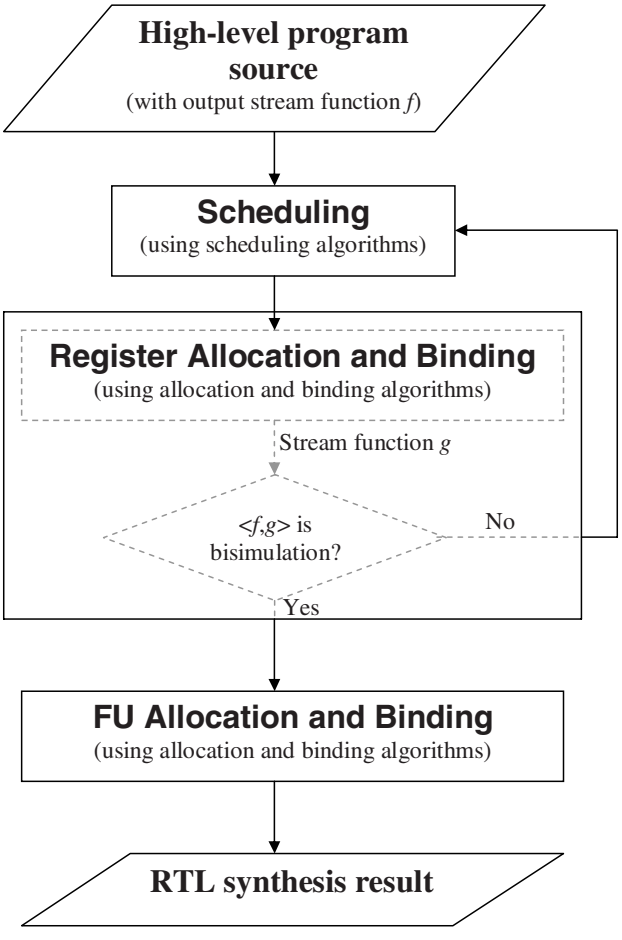


Fig. 20. Our coinductive approach to verifying flowware synthesis

4.2 Using Bisimulation Relation for Verifying Flowware Synthesis

Using a stream function and bisimulation relation, our verification technique is integrated into the register allocation and binding step of the three-step synthesis process. By this approach, we consider two following themes:

- *Bisimulation* (denoted by $\sim$) between the output stream function f specifying output from a high-level program source and the behavioral stream function g specifying output after allocation and binding the registers.
- *Integrating bisimulation* into the register allocation and binding step of the three-step synthesis process.

Bisimulation between f and g ($f \sim g$)

Let f and g be of polynomial streams represented by

$$\sum_{i=0}^{n} r_i X^i$$

(see stream function 2.15 on page 10)

Proposition 4.1 (Bisimulation). *If*

$$f = \sum_{i=0}^{n} f_i X^i \tag{4.1}$$

and

$$g = X^k \times f = \sum_{i=0}^{n} f_i X^{i+k} \tag{4.2}$$

then

$$f \sim g$$

Proof. This is a generalization of the propositions 3.3 on page 18 and 3.5 on page 25. A relation $\langle f, g\rangle \in B \subseteq \mathbb{R}^\omega \times \mathbb{R}^\omega$ can be constructed in stages, by computing the respective derivatives of both f and g step by step. The pairs to be included in B are the following

$$\{\langle f_i, g_{i+k}\rangle \mid 0 \leqslant i \leqslant n\}$$

For instance, consider the bisimulation relations in proposition 3.3 on page 18 and in proposition 3.5 on page 25; we have the following bisimulations:

f	$\sim$	g
σ		$X^{(n)} \times \sigma$, for $n \geqslant 0$
τ		$X^{(n)} \times \tau$, for $n \geqslant 0$
$a + X \times a + X^2 \times a$		$X \times a + X^2 \times a + X^3 \times a$

Corollary 4.2 (Coinduction). *If*

$$f = \sum_{i=0}^{n} f_i X^i \tag{4.3}$$

and

$$g = X^k \times f = \sum_{i=0}^{n} f_i X^{i+k} \tag{4.4}$$

are bisimilar then

$$f = g$$

Proof. Given two output data flows f in (4.3) and g in (4.4), from proposition 4.1, $B \subseteq \mathbb{R}^\omega \times \mathbb{R}^\omega$ is a bisimulation on $\mathbb{R}^\omega$ containing the pair $\langle f_i, g_{i+k} \rangle$. It follows by induction on m that $\langle f_m, g_{m+k} \rangle \in B$, for $m \geqslant 0$, because B is a bisimulation. This implies that $f^{(m)}(0) = g^{(m+k)}(0)$, and as a result this proving that $f = g$.

As an illustration, consider corollary 3.4 on page 18 and corollary 3.6 on page 26; they yield:

f	$=$	g
σ	Behavioral	$X^{(n)} \times \sigma$, for $n \geqslant 0$
τ	equivalence	$X^{(n)} \times \tau$, for $n \geqslant 0$
$a + X \times a + X^2 \times a$		$X \times a + X^2 \times a + X^3 \times a$

Specifying integration of ($f \sim g$) into a verified flowware synthesis

To see integration of a bisimulation relation into the flowware synthesis process as in Figure 20, we need to specify this algorithmic synthesis process in an algebraic manner. In other words, the flowware synthesis process needs to be represented as an algebraic expression and, in this way, it can be equivalently transformed by some algebraic laws [19] where each of three synthesis steps is considered as a *programmatic operand* of the algebraic expression, and the bisimulation between f and g as a *control operand* (i.e., Boolean operand). In fact, the notions of operand and operator are necessary to build up a so-called algebraic expression and this means that using suitable operators, the bisimulation relation $f \sim g$ (as a control operand) and three flowware synthesis steps (as three programmatic operands) can be integrated together for a verified flowware synthesis. Thus, an algebraic expression is seen as a formal semantics of the verified flowware synthesis.

As a reasonable convention, a programmatic operand refers to a program or command and control operand to a Boolean expression. Let S, R, F and $f \sim g$ be operands referring to programs of scheduling, register allocation and binding, FU allocation and binding, together with a bisimulation relation, respectively. A formal semantics of the verified flowware synthesis can be descriptively specified by the following semantic mapping:

Verified flowware synthesis(*scheduling, register, FU, bisimulation*)

semantic | *mapping*

Algebraic Expression($S, R, F, f \sim g$)

Besides the operands such as S, R, F or $f \sim g$, we need to define some suitable operators to able to set up an algebraic expression of flowware synthesis.

• **SKIP** *command:* This is a programmatic operand. Execution of this command terminates successfully, leaving everything unchanged. **SKIP** is denoted by **1**.

• *Operator of sequential composition* (denoted by ;): $(S; R)$ is a program that is executed by first executing S. If S does not terminate; neither does $(S; R)$. If and when S terminates, R is started; and then $(S; R)$ terminates when R does.

The sequential composition has some following algebraic laws:

– *Unit* **1:**

$$S; \mathbf{1} = \mathbf{1}; S = S \tag{4.5}$$

This means that preceding or following a program S by the command **1** does not change the effect of the program S.

– *Associativity:*

$$S; R; F = (S; R); F = S; (R; F) \tag{4.6}$$

This means that sequential composition is associative; to perform three programs in order, we can either perform the first program followed by the other two or the first two programs followed by the third.

• *Operator of nondeterministic choice* (denoted by ∪:) $(S \cup R)$ is a program that is executed by executing either S or R. The choice between them is arbitrary. Thus, we can delegate the decision to the synthesis process executing the program.

In the presence of nondeterminism, the following laws apply:

– *Symmetry*

$$S \cup R = R \cup S \tag{4.7}$$

Obviously, there is not any difference in what order a choice is offered.

– *Associativity*

$$S \cup (R \cup F) = (S \cup R) \cup F \tag{4.8}$$

A choice between three alternatives can be offered as first a choice between one alternative and the other two, followed by a choice between the other two, and it does not matter in which way the choices are grouped.

– *Idempotence*

$$S \cup S = S \tag{4.9}$$

A choice between one thing and itself offers no choice at all.

A choice between n different scheduling programs can be expressed by the indexed notation:

$$\bigcup_{i \leqslant n-1} S_i = S_0 \cup S_1 \cup \ldots \cup S_{n-1}$$

and thus, when a scheduling program S is considered, then this means that

$$S = \bigcup_{i \leqslant n-1} S_i$$

In a similar way, a choice between n allocation and binding programs relating to registers or FUs is represented by

$$R = \bigcup_{i \leqslant n-1} R_i = R_0 \cup R_1 \cup \ldots \cup R_{n-1}$$

and

$$F = \underset{i \leqslant n-1}{\cup} F_i = F_0 \cup F_1 \cup \ldots \cup F_{n-1}$$

• *Conditional operator* (denoted by $\lhd \rhd$:) $(F \lhd (f \sim g) \rhd S)$ is a program. It is executed by first evaluating $(f \sim g)$. If $(f \sim g)$ is true, then F is executed otherwise S is executed instead. A commonly used notation for a conditional is **if** $(f \sim g)$ **then** F **else** S.

Let a negation of $(f \sim g)$ be $(f \nsim g)$ (or $\neg(f \sim g)$). We have the following laws:

$$F \lhd (f \sim g) \rhd S = S \lhd (f \nsim g) \rhd F \tag{4.10}$$

$$(F \lhd (f \sim g) \rhd S); R = (F; R) \lhd (f \sim g) \rhd (S; R) \tag{4.11}$$

Law (4.10) offers a fairly familiar equivalence when replacing evaluation of $(f \sim g)$ by evaluation of its negation. Law (4.11) distributes leftwards through $(f \sim g)$, namely, evaluating $(f \sim g)$ is not affected by what happens afterwards.

• *Looping operator* (denoted by •:) $((f \nsim g) \bullet S)$ is a program. It is executed by first evaluating $(f \nsim g)$. If $(f \nsim g)$ is false, execution terminates successfully and nothing is changed, otherwise the process proceeds to execute $S; ((f \nsim g) \bullet S)$. A conventional notation for looping is **while** $(f \nsim g)$ **do** S. Thus, $((f \nsim g) \bullet S)$ can be also expressed by

$$(f \nsim g) \bullet S = (S; (f \nsim g) \bullet S) \lhd (f \nsim g) \rhd \mathbf{1} \tag{4.12}$$

Based on the above-mentioned operators, the verified flowware synthesis, in which the bisimulation $f \sim g$ is integrated (see Figure 20), is specified by the following expression:

$$(S; R); (F \lhd (f \sim g) \rhd (((S; R); (f \nsim g) \bullet (S; R)); F)) \tag{4.13}$$

It means that this is a program of the verified flowware synthesis. It is executed by first scheduling, allocating and binding registers, then evaluating $(f \sim g)$ (or $(f \nsim g)$). If $(f \sim g)$ is true (or $(f \nsim g)$ is false), allocating and binding FUs are executed, otherwise the process proceeds to execute repeatedly scheduling, allocating and binding registers, then evaluating $(f \sim g)$ (or $(f \nsim g)$). The verified flowware synthesis terminates when and if allocating and binding FUs terminate successfully.

By (4.5) under sequential composition, (4.13) is algebraically transformed to be the following expression

$$(S; R); ((\mathbf{1}; F) \lhd (f \sim g) \rhd (((S; R); (f \nsim g) \bullet (S; R)); F)) \tag{4.14}$$

(4.14) becomes the following expression by (4.11) under conditional operator

$$(S; R); (\mathbf{1} \lhd (f \sim g) \rhd ((S; R); (f \nsim g) \bullet (S; R))); F \tag{4.15}$$

By (4.10) under conditional operator, then (4.15) becomes the following expression

$$(S; R); (((S; R); (f \nsim g) \bullet (S; R)) \lhd (f \nsim g) \rhd \mathbf{1}); F \tag{4.16}$$

By (4.12) under looping operator, expression (4.16) is driven into

$$(S; R); ((f \nsim g) \bullet (S; R)); F \tag{4.17}$$

It can be seen that all expressions (4.13) – (4.17) have the same meaning but version (4.17) of the specification is apparently more expressive.

5 Conclusion

In this paper, using stream calculus, we have formalized the flowware of reconfigurable computing synthesis. In particular, RTL synthesis with and without looping is formalized as behavioral functions in stream calculus. This approach supports the semantics of RTL synthesis and is useful in applying coinduction to compare the behavioral functions. Our formalization allows pipelining, in which all hardware resources consisting of functional units as well as registers are reused during different control steps.

We have demonstrated that our coinductive approach to verifying flowware synthesis, which is independent of the heuristic during the register allocating and binding step, can be applicable in practice when using stream calculus and coinductive proof principles to formalize computing based on data flow of flowware. In our approach, it turns out that, in using a behavioral stream function and a bisimulation relation, the formal verification seems to be a promising alternative to the current commonly used post-synthesis verification approach. In addition, our intention is to develop a coinductively verifying configware/flowware synthesis approach that covers the full synthesis from the algorithmic level down to the logical level to help guarantee correct synthesis.

Acknowledgements. Thank you to the anonymous reviewers for their helpful comments. Jonathan Bowen is currently a Visiting Professor in the Centre for Research on Evolution Search & Testing (CREST) at King's College London.

References

1. Ast, A., Becker, J., Hartenstein, R., Kress, R., Reinig, H., Schmidt, K.: Data-Procedural Languages for FPL-based Machines. In: Hartenstein, R.W., Servít, M.Z. (eds.) FPL 1994. LNCS, vol. 849, pp. 183–195. Springer, Heidelberg (1994)
2. Bailey, B., Gajski, D.: RTL Semantics and Methodology. In: 14th International Symposium on Systems Synthesis (ISSS), Montreal, Canada, pp. 69–74 (2001)
3. Black, D.C., Donovan, J.: SystemC: From the Ground Up, 1st edn., p. 244. Springer, Heidelberg (2004)
4. Bowen, J.P., Hinchey, M.G.: Formal Methods. In: Tucker Jr., A.B. (ed.) Computer Science Handbook, ch.106, Section XI, Software Engineering, 2nd edn., pp. 106–1 – 106–125. Chapman & Hall / CRC, ACM, USA (2004)
5. DeHon, A.: DPGA Utilization and Application. In: 4th International Symposium on Field Programmable Gate Arrays (FPGA 1996), Monterey, CA, US, 11–13 February 1996, pp. 115–121. ACM Press, New York (1996)

6. DeHon, A., Wawrzynek, J.: Reconfigurable Computing: What, Why, and Implications for Design Automation. In: 36th Design Automation Conference (DAC), New Orleans, LA, USA, 21–25 June 1999, pp. 610–615. ACM Press, New York (1999)
7. Eisenbiegler, D., Kumar, R.: Formally Embedding Existing High Level Synthesis Algorithms. In: Camurati, P.E., Eveking, H. (eds.) CHARME 1995. LNCS, vol. 987, pp. 2–4. Springer, Heidelberg (1995)
8. Grötker, T., Liao, S., Martin, G., Swan, S.: System Design with SystemC, 1st edn., p. 240. Springer, Heidelberg (2002)
9. Hartenstein, R.: The Microprocessor is No Longer General Purpose: Why Future Reconfigurable Platforms Will Win. In: 2nd Annual International Conference on Innovative Systems in Silicon, Austin, TX, USA, 8–10 October 1997, pp. 2–12. IEEE, Los Alamitos (1997)
10. Hartenstein, R.: A Decade of Reconfigurable Computing: A Visionary Retrospective. In: Design, Automation, and Test in Europe Conference (DATE), Munich, Germany, 13–16 March 2001, pp. 642–649. IEEE, Los Alamitos (2001)
11. Hartenstein, R.: Coarse Grain Reconfigurable Architectures. In: Asia and South Pacific Design Automation Conference (ASP-DAC), Yokohama, Japan, 30 January – 2 February 2001, pp. 564–569. IEEE, Los Alamitos (2001)
12. Hartenstein, R.: Reconfigurable Computing: A New Business Model and its Impact on SoC Design. In: Euromicro Symposium on Digital Systems, Design (DSD), Warzaw, Poland, 4–6 September 2001, pp. 103–110. IEEE, Los Alamitos (2001)
13. Hartenstein, R.: Trends in Reconfigurable Logic and Reconfigurable Computing. In: 9th International Conference on Electronics, Circuits and Systems (ICECS), Dubrovnik, Croatia, 15–18 September 2002, vol. 2, pp. 801–808. IEEE, Los Alamitos (2002)
14. Hartenstein, R.: Are We Really Ready for the Breakthrough? [morphware]. In: International Parallel and Distributed Processing Symposium (IPDPS), Nice, France, 22–26 April 2003, IEEE, Los Alamitos (2003) (CD-ROM)
15. R. Hartenstein. Handbook of Nature-Inspired and Innovative Computing. Integrating Classical Models with Emerging Technologies, chapter Morphware and Configware. p. 780, Springer, Heidelberg (2005)
16. Hartenstein, R., Kress, R.: A Datapath Synthesis System for the Reconfigurable Datapath Architecture. In: Design Automation Conference of the ASP-DAC 1995/CHDL 1995/VLSI 1995, IFIP International Conference on Hardware Description Languages, IFIP International Conference on Very Large Scale Integration, Asian and South Pacific, Chiba, Japan, 29 August – 1 September 1995, pp. 479–484. IEEE, Los Alamitos (1995)
17. Herz, M., Hartenstein, R., Miranda, M., Brockmeyer, E., Catthoor, F.: Memory Addressing Organization for Stream-based Reconfigurable Computing. In: 9th International Conference on Electronics, Circuits and Systems (ICECS), Dubrovnik, Croatia, 15–18 September 2002, vol. 2, pp. 813–817. IEEE, Los Alamitos (2002)
18. Hinchey, M.G., Bowen, J.P.: Industrial-Strength Formal Methods in Practice. In: Formal Approaches to Computing and Information Technology (FACIT), Springer, London (1999)
19. Hoare, C.A.R., Hayes, I.J., Jifeng, H., Morgan, C.C., Roscoe, A.W., Sanders, J.W., Sorensen, I.H., Spivey, J.M., Sufrin, B.A.: Laws of Programming. Communications of the ACM 30(8), 672–686 (1987)
20. Hughes, J., Jacobs, B.: Simulations in Coalgebra. Theoretical Computer Science 327(1-2), 71–108 (2004)

21. Jacobs, B.: Exercises in Coalgebraic Specification. In: Blackhouse, R., Crole, R.L., Gibbons, J. (eds.) Algebraic and Coalgebraic Methods in the Mathematics of Program Construction. LNCS, vol. 2297, pp. 237–280. Springer, Heidelberg (2002)
22. Jacobs, B., Rutten, J.: A Tutorial on (Co)Algebras and (Co)Induction. Bulletin of EATCS 62, 222–259 (1997)
23. Kumar, R., Blumenröhr, C., Eisenbiegler, D., Schmid, D.: Formal Synthesis in Circuit Design – A Classification and Survey. In: Srivas, M., Camilleri, A. (eds.) FMCAD 1996. LNCS, vol. 1166, pp. 6–8. Springer, Heidelberg (1996)
24. Lysaght, P.: Aspects of Dynamically Reconfigurable Logic. In: Colloquium on Reconfigurable Systems, Glasgow, Scotland, UK, 10 March 1999, vol. 61, pp. 1/1–1/5. IEE (1999)
25. Rutten, J.J.M.M.: Automata and Coinduction (an Exercise in Coalgebra). In: Sangiorgi, D., de Simone, R. (eds.) CONCUR 1998. LNCS, vol. 1466, pp. 8–11. Springer, Heidelberg (1998)
26. Rutten, J.J.M.M.: Universal Coalgebra: A Theory of Systems. Theoretical Computer Science 249(1), 3–80 (2000)
27. Rutten, J.J.M.M.: Elements of Stream Calculus (An Extensive Exercise in Coinduction). Electronic Notes in Theoretical Computer Science, 45 (2001)
28. Rutten, J.J.M.M.: An Application of Stream Calculus to Signal Flow Graphs. In: de Boer, F.S., Bonsangue, M.M., Graf, S., de Roever, W.P. (eds.) FMCO 2003. LNCS, vol. 3188, pp. 4–7. Springer, Heidelberg (2004)
29. Rutten, J.J.M.M.: Algebra, Bitstreams, and Circuits. Technical Report SEN-R0502, CWI, Amsterdam, The Netherlands (2005)
30. Rutten, J.J.M.M.: Algebraic Specification and Coalgebraic Synthesis of Mealy Automata. In: Barbosa, L.S., Liu, Z. (eds.) 2nd International Workshop on Formal Aspects of Component Software (FACS), 24–25 October 2005, vol. 160, pp. 305–319. Elsevier Science Publishers Ltd, UNU/IIST, Macao (2006)
31. Sangiorgi, D.: Bisimulation: From the Origins to Today. In: 19th Annual Symposium on Logic in Computer Science, Turku, Finland, 13–17 July 2004, pp. 13–17. IEEE, Los Alamitos (2004)
32. Tessier, R., Burleson, W.: Reconfigurable Computing for Digital Signal Processing: A Survey. VLSI Signal Processing 28(1–2), 7–27 (2001)
33. Vinh, P.C.: Formal Aspects of Dynamic Reconfigurability in Reconfigurable Computing Systems. PhD thesis, London South Bank University, 103 Borough Road, London SE1 0AA, UK (May 4, 2006)
34. Vinh, P.C., Bowen, J.P.: Formalising Configuration Relocation Behaviours for Reconfigurable Computing. In: SFP Workshop, FDL 2002: Forum on Specification & Design Languages, Marseille, France, September 24–27, 2002 (2002) (CD-ROM)
35. Vinh, P.C., Bowen, J.P.: An Algorithmic Approach by Heuristics to Dynamical Reconfiguration of Logic Resources on Reconfigurable FPGAs. In: 12th International Symposium on Field Programmable Gate Arrays, Monterey, CA, USA, 22–24 February 2004, p. 254. ACM/SIGDA (2004)
36. Vinh, P.C., Bowen, J.P.: A Provable Algorithm for Reconfiguration in Embedded Reconfigurable Computing. In: Hinchey, M.G. (ed.) 29th Annual IEEE/NASA Software Engineering Workshop (SEW), Greenbelt, MD, USA, 6–7 April 2005, pp. 245–252. IEEE Computer Society Press, Los Alamitos (2005)
37. Vinh, P.C., Bowen, J.P.: Continuity Aspects of Embedded Reconfigurable Computing. Innovations in Systems and Software Engineering: A NASA journal 1(1), 41–53 (2005)

38. Vinh, P.C., Bowen, J.P.: POM Based Semantics of RTL and a Validation Method for the Synthesis Results in Dynamically Reconfigurable Computing Systems. In: Rozenblit, J., O'Neill, T., Peng, J. (eds.) 12th Annual International Conference and Workshop on the Engineering of Computer Based Systems (ECBS), Greenbelt, MD, USA, 4–5 April 2005, pp. 247–254. IEEE Computer Society Press, Los Alamitos (2005)
39. Vinh, P.C., Bowen, J.P.: A Formal Approach to Aspect-Oriented Modular Reconfigurable Computing. In: 1st IEEE & IFIP International Symposium on Theoretical Aspects of Software Engineering (TASE), Shanghai, China, 6–8 June 2007, pp. 369–378. IEEE Computer Society Press, Los Alamitos (2007)
40. Woodcock, J.C.P.: First Steps in the Verified Software Grand Challenge. IEEE Computer 39(10), 57–64 (2006)

Partners Selection in Multi-Agent Systems by Using Linear and Non-linear Approaches

Fenghui Ren and Minjie Zhang

School of Information Technology and Software Engineering
University of Wollongong, Australia
{fr510,minjie}@uow.edu.au

Abstract. Traditional negotiation approaches pay intensive attention to decision making models in order to reach the optimal agreements, while placing insufficient efforts on the problem of partner selection. In open and dynamic environments, when the number of potential partners is huge, it may be expensive or even impractical to perform complicated negotiations with all of its potential partners. In this paper, based on the proposed extended dual model, we propose both linear and non-linear approaches for partner selection in multi-agent systems. By employing these two approaches with the extended dual concern model, agents can adapt their individual behaviors for partners selection in negotiation. The proposed approaches have three merits, which are: (1) both agents' own benefits and their potential partners' benefits are considered during the partners selection process; (2) agents' preferences are employed by the proposed approaches which ensure the selection results to accord with agents' expectations; (3) the proposed approaches are sensitive to changes of the negotiation environment, so they can be adopted in open and dynamic negotiation environments. According to the case study in four scenarios, the selection results are reasonable and accord with agents' expectations.

1 Introduction

Traditional negotiation approaches in multi-agent systems (MASs), such as the game theory [1] [2] [3] [4] [5] and the argumentation-based negotiation [6] [7] [8] [9], emphasize the decision making models to determine the optimal coalition structure and the division of payoff, with a little devotion to the negotiation partners selection. In recent years, some researchers have recognized the importance of partners selection in negotiation and proposed several approaches for selecting suitable partners during the negotiation. In [3], a significant model is introduced by Faratin et al., which defines a range of strategies and can be employed by computational agents to generate initial offers, evaluate proposals and offer counter proposals. With such a model, in each cycle of the negotiation, a comprehensive analysis is applied to help agents find the optimal offers and most suitable partners. Kraus [10] further classifies negotiations into three categories, which are data allocation, resource allocation and task distribution, according to

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 37–60, 2008.

their application domains. In each of these categories, complicated and heuristic methods are introduced to help agents find the optimal negotiation agreements under different situations. However, as the rapid development of autonomous agents and the Internet techniques, most work environments of MASs become uncertain and dynamic [11] [12]. In such open and dynamic environments, when the number of potential partners is huge, to perform complicated negotiations with all of the potential partners may be expensive in terms of computational time and resources, or even impractical. Thus, an appropriate approach which can be employed by agents to choose partners with a high chance of reaching a good agreement in subsequent negotiation from a large scale of potential partners is required greatly. Such a selection mechanism is very important because of practicality and efficiency of MASs interactions.

Nevertheless, it is noticed that agents may perform various behaviors in negotiation by considering their motivations and aims, which makes the partners selection much more complicated and difficult to steer than expected. Therefore, it is necessary to discuss the kinds of agents' behaviors in negotiation before the partners selection mechanism is given. In general, agents may compete or cooperate with each others in order to gain their own goals or a common goal in MASs. The final agreements about how to compete or cooperate are achieved through negotiating. Therefore, negotiations can be classified into *competitive negotiation* and *cooperative negotiation* according to the behaviors of its participants. In a *competitive negotiation*, participators perform the roles as challengers, while in a *cooperative negotiation*, participators are cooperators. Thus, criterions on partner selection are also different in these two kinds of negotiations. For example, in a *cooperative negotiation*, since agents see others' gain as its own, it will select the agents which can increase global benefits as its parters. In a *competitive negotiation*, agents prefer choose a partner which can supply the highest benefit to itself than others. These two kinds of partners selection strategies are the most simple and direct approaches in extreme situations. However, in actually, researches [13] [14] found that it is not always beneficial for agents only to cooperate with others about global tasks in *cooperative negotiation*. Also in a *competitive negotiation*, agents might choose to commit to global tasks for other agents. What is more, in some circumstance, when agents' behaviors beyond these two extreme situations, agents cannot adopt these two extreme partners selection approaches simply, because agents need both appropriate competition and cooperation to maximum both the local and global utilities.

In order to address these issues mentioned above, an extended dual concern model for partners selection in negotiation is proposed in this paper. Furthermore, based on this model, both linear and non-linear partners selection approaches are proposed. The linear approach employs agents preference and balances the selection between these two extreme situations mentioned above, while the non-linear approach employs fuzzy logic mechanism to model and control partners selection process. The advantage of the linear approach is easy to be implemented and quicker results generating in a short time, while the advantage of the non-linear approach is generating more reasonable selection results and

more logical. In general, these two approaches have three common merits, which are: (1) both the agents' own benefit and their partners' benefits are considered; (2) by employing the extended dual concern model, agents' attitudes to their partners are captured in partners selection process; (3) both of them are sensitive to the change of the negotiation environment and can be employed an open and dynamic negotiation environment.

The remainder of this paper is organized as follows. In Section 2, the related works are presented and discussed. In Section 3, an extended dual concern model is proposed, the partners selection problem is formally described, and potential partners in general negotiation in MASs are further classified and analyzed. In Section 4, we propose a linear approach for the partners selection. In Section 5, a non-linear approach is proposed. In Section 6, several examples are demonstrated and evaluated. Finally, in Section 7, this paper is concluded and further work is outlined.

2 Related Work

Previous research [15] [16] [17] [18] [19] in the MASs literature has proposed models for partners selection, which take others' preferences into account when deliberating about agents' decisions. In this section, we just list and discuss few of them in order to highlight our motivation.

Brzostowski and Kowalczyk [20] [21] proposed a possibilistic case-based model in which the possibility of successful negotiation for each potential partners is predicted on the basis of its behaviors in previous offers. The qualitative expected utility for each potential partner is derived and the agents are ordered according to the values of these utilities. The order determines who is more and who is less desirable partner for negotiation. However, this possibilistic case-based model is based on the assumption that the more similar are situations the more possible that the outcomes are similar, which may restrict applications in open dynamical environments. That is because in an open dynamical environment, the complete factors set which may impact the selection results that cannot be captured nor monitored fully by the system. Thus, if the factors monitored by system are not significant for partners selection in a negotiation, even though some cases are very similar, it is still not confident enough to make a final decision. What is more, in an open dynamical environment, the importance of each selection factor may be changed too along with the changing of environment. Therefore, within different period, the most suitable partners may be different for the same factor set. This phenomena very likes the situation that the same person may have different perceptions on the same picture at different environments and spirits.

Munroe et al. [22] proposed a motivation-based mechanism to evaluate and select negotiation candidates. In this mechanism, the acceptable candidates were identified by use of motivation-based thresholds on objective scoring measures first. Then, all negotiation issues are classified and the significance of the negotiation issues were considered. According to this classification, each partner's expected performance are evaluated and compared. Thus, all potential partners

were judged and the most suitable partners for the agent can be selected. Also in this mechanism, as an agent's circumstance changes, the significance of each negotiation issue will also be changed, and this changing will be considered in the negotiation partners selection.

In [23], Brzostowski and Kowalczylk proposed a partners selection mechanism based on the Case-Based Reasoning (CBR) theory [24] [25] and Fuzzy Logic [26] [27]. According to the fact that similar problems usually have similar solutions, the authors employed CBR to retrieve relevant cases from the database and adapt them to fit new situations. Therefore, this mechanism is not to solve the new problem but to predict the outcome of a future negotiation and to match each new situation to a particular existing case. In this paper, the prediction process was finished by aggregating historical negotiations' records, and the case matching process was completed by employing the fuzzy logic. Three components, which were decision problems, possible acts and conceivable results, in the negotiation were employed to evaluate each new case and match the new case to an existing case.

However, to summarize previous researchers' works, some limitations are emerged. For instance, the case-based approach may not work well when the negotiation environment becomes open and dynamic. The reasons behind of this are the space limitations on the cases database and the faultiness of reasoning mechanisms. Also, the motivation-based approach may only work well for the *competitive negotiation* because the decision of the significance on each negotiated issue is only based on agents' own interesting, but does not consider negotiation partners' requirements. In order to improve the current partners selection approaches and to extend their application domains, both a linear and non-linear partners selection approaches are proposed in this paper based on the extended dual concern model. Compared with previous researches, the contributions of this research are to: (1) identify different potential cases of partners selection in general negotiation; (2) be sensitive to the changes of circumstance and suitable to work in open and dynamic negotiation environment; and (3) balance agents' own benefits and their partners' benefits. Thus, agents can have a global view over all perspective partners during negotiation and can select their preferable partners according to their own expectations.

3 Potential Partners Analysis in General Negotiations

3.1 The Extended Dual Concern Model

In [28], Zhang et al. proposed a dual concern model which is an outline about the degrees of concern of an agent for agents' own and partners' outcomes. However, this model just briefly presents the main trend of these degrees, not offering any calculation method about how to decide the values of these degrees and how to compare these degrees. To address these problems, we further extend this dual concern model in order to allow agent to make reasonable decisions on partners' behaviors during negotiation. The extended dual concern model is shown in Figure 1.

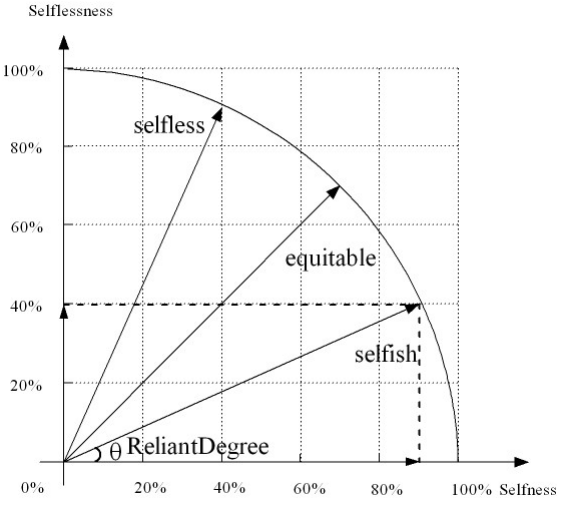


Fig. 1. The extended dual concern model

In Figure 1, x-axis indicates the percentage of the self-concern of an agent while y-axis is the percentage of other-concern from the agent. θ presents a *ReliantDegree* (i.e. reflection of the collaborate degree), where $\theta \in [0°, 90°]$. We use *selfness* to represent the percentage of self-concern of an agent, which can be calculated by $\cos(\theta)$, and *selflessness* to represent the percentage of other-concern, which can be evaluated by $\sin(\theta)$. A *ReliantDegree* can illustrate the level of collaboration between the agent and its potential partners. From the extended model, we can find that there are two extreme cases. (1) When the agent only emphases on its own outcome, its negotiation attitude is *selfish* ($\theta = 0°$); and (2) When the agent only care about its partners' incomes, its attitude is *selfless* ($\theta = 90°$).

From this model, it is clear that there are many other cases between *selfish* and *selfless* negotiation behaviors. In subsection 3.2, a formal problem description will be given. In subsection 3.3, potential cases in partner selection based on this extended dual concern model will be analyzed in detail.

3.2 Problem Description

Suppose that there are n potential partners for an agent ID_x in a MAS. If we use a four-tuple p_i^x to present the ith potential partner for agent ID_x, p_i^x can be formally defined by the following equation

$$p_i^x =< ID_i, GainRatio_i^x, ContributionRatio_i^x, ReliantDegree_i^x > \qquad (1)$$

where ID_i is the unique identification of the ith potential partner, and $GainRatio_i^x$, $ContributionRatio_i^x$ and $ReliantDegree_i^x$ are factors used to evaluate the potential partner ID_i to be selected in negotiation. These three factors are defined from Definitions 1 to 3, respectively.

Definition 1. *$GainRatio_i^x$ is the percentage of the benefit that agent ID_x obtains out of the global benefit upon the completion of the task, and can be calculated by Equation 2.*

$$GainRatio_i^x = \frac{S}{L} \times 100\% \tag{2}$$

where $GainRatio_i^x \in [0, 100\%]$, S denotes the benefit that agent ID_x gains by selecting agent ID_i as its partners, and L denotes the global benefit by completing the task.

Definition 2. *$ContributionRatio_i^x$ is the percentage of the benefit that agent ID_i obtains out of the global benefit upon the completion of the task, and can be calculated by Equation 3.*

$$ContributionRatio_i^x = \frac{I}{L} \times 100\% \tag{3}$$

where $ContributionRatio_i^x \in [0, 100\%]$, I denotes the benefit that agent ID_i gains by cooperating with agent ID_x, and L denotes the global benefit by completing the task.

Definition 3. *$ReliantDegree_i^x$ represents agent ID_x's attitude to the negotiation, and also indicates the dynamic behavior of the agent, such as selfness, selflessness or other cases. $ReliantDegree_i^x$ can be calculated by Equation 4.*

$$ReliantDegree_i^x = \arctan(\frac{Cr_x^i}{Cr_i^x}) \tag{4}$$

where $ReliantDegree \in [0°, 90°]$, Cr_x^i indicates how agent ID_x trusts agent ID_i, which can be defined as the trading success ratio from agent ID_x to ID_i or can be assigned by the system based on the performance record of agent ID_i, and Cr_x^i indicates how agent ID_i trusts agent ID_x, which can be defined in the similar way as Cr_x^i.

Then the collaborate degree between the agent ID_x and its potential partner ID_i is generated as Equation 5

$$CollaborateDegree_i^x = f(ID_x, p_i^x) \tag{5}$$

By calculating the collaborate degree between agent ID_x and all of its potential partners, the collaborate degree set ($CollaborateDegree^x$) is generated as follows:

$$CollaborateDegree^x = \{CollaborateDegree_i^x\}, i \in [1, n] \tag{6}$$

3.3 Potential Partners Analysis

Figure 2 demonstrates the relationship between *ContributionRatio* and *GainRatio*. From Definition 1 and Definition 2, we can see that $GainRatio + ContributionRatio \leq 100\%$, so that all curves in Figure 2 must be located under the curve of $GainRatio + ContributionRatio = 100\%$.

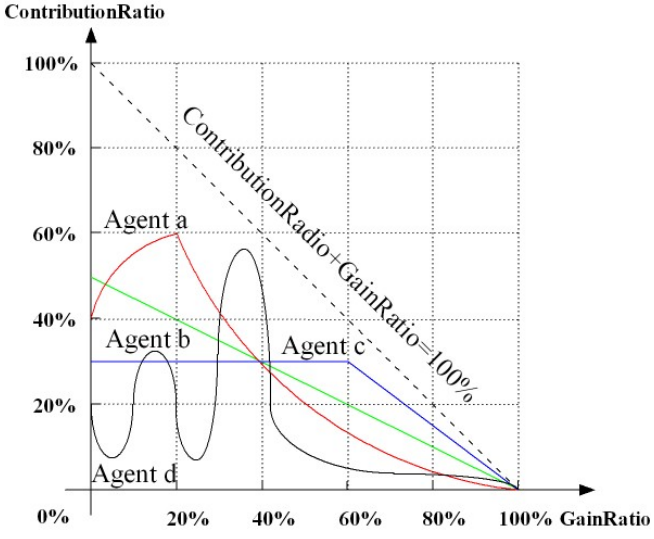


Fig. 2. Relationships between *GainRatio* and *contributionRatio*

In order to illustrate the relationship between parameters *GainRatio* and *ContributionRatio*, and to summary agents' possible negotiation behaviors, we specify some particular kinds of agents firstly. In general, as *GainRatio* increases, the behaviors of potential partners can be classified into the following four categories as shown in Figure 2.

1. As *GainRatio* increases, for the potential partner *a*, the *ContributionRatio* increases in the first range until its maximum value is achieved, then the *ContributionRatio* decreases to 0 continuously. For example, in stock markets, as the investors' investment increases, the whole market's benefit is increases. When the investors' investment overflows a threshold, the whole market's benefit will decrease because too much money is taken away by investors and the remainder is not enough for further development.
2. As *GainRatio* increases, for the potential partner *b*, the *ContributionRatio* decreases to 0 continuously. For example, in a biding model, as the buyer's utility increases, the seller's utility decreases.
3. As *GainRatio* increases, for the potential partner *c*, the *ContributionRatio* remains unchanged in the beginning and continuously decreases to 0 after a threshold. For example, in a forest, when the number of trees cut down by the woodchopper is smaller than a certain number, the local environment will not be impacted. While the number of trees cut down by the woodchopper is bigger than a certain number, the local environment will become more and more worse as the number increased.
4. As *GainRatio* increases, for the potential partner *d*, the *ContributionRatio* fluctuates and achieves 0 eventually. All agents whose behaviors do not include in the above situations will be classified in this case.

In general, a *selfless* agent will select the partner whose *ContributionRatio* is the maximum value in Figure 2 and ignore the factor *GainRatio*, while a *selfish* agent will select the partner based on *GainRatio* only. However, in most cases, agents behave between these two extreme situations. An agent needs to consider both own benefit and its partners' benefit. In order to balance these two extreme situations, an linear approach for partners selection in a more flexible way is proposed in the next Section.

4 Partners Selection by Using a Linear Approach

In last section, we introduced the extended dual concern model. We also defined three attributes to represent relationships between the agent and its potential partners. Finally, we analyze four possible situations on potential partners' behaviors and claim that the partners selection approach should balance all negotiation participators' benefits. In this section, based on the proposed extended dual concern model and the analysis results, we proposed a linear approach to select partners based on the relationship among negotiation participators.

In order to cover all potential cases in partners selection, we need consider not only both *GainRatio* and *ContributionRatio*, but also the preference of the agent on these two criterions. It is proposed that the agent's preference on *GainRatio* and *ContributionRatio* can be represented by a normalized weight. Let w_g stands for the weight on *GainRatio*, w_c stands for the weight on *ContributionRatio*, and $w_c + w_g = 1$. Then the *collaborationDegree* between agent ID_x and its potential partner ID_i is defined as follows:

$$CollaborateDegree_i^x = GainRatio_i^x \times w_g + ContributionRatio_i^x \times w_c \quad (7)$$

The *collaborationDegree* ($\in [0, 1]$) indicates the degree that the potential partner should be selected by the agent. The bigger the *collaborationDegree*, the more chance that the potential partner will be selected by the agent. In general, there are three extreme cases on different combinations of w_c and w_g as follows:

- When $w_g = 0$ and $w_c = 1$, *CollaborateDegree* is calculated only based on *ContributionRatio*, i. e. agent ID_x's attitude to negotiation is fully *selfless*.
- When $w_g = 1$ and $w_c = 0$, *CollaborateDegree* is calculated only based on *GainRatio*, i. e. agent ID_x's attitude to negotiation is fully *selfish*.
- When $w_g = w_c = 0.5$, *CollaborateDegree* is calculated based on both *GainRatio* and *ContributionRatio* equally, i. e. agent ID_x's attitude to negotiation is *equitable*.
- Besides above three cases, the restriction of $w_g + w_c = 1$ can also reflect agent ID_x's attitude on *GainRatio* and *ContributionRatio* in other cases.

The weights w_g and w_c can be calculated by employing the value of *ReliantDegree*, which are defined by Equation 8 and Equation 9, respectively.

$$w_g = \cos^2(ReliantDegree) \quad (8)$$

$$w_c = \sin^2(ReliantDegree) \tag{9}$$

These definitions about w_g and w_c can reflect the relationships between the factor of *ReliantDegree* and the agent' preference successfully. For example, as shown in Figure 1,

- when $ReliantDegree = 0°$, it is supposed that the agent ID_x should perform as selfness and only consider about its own benefits. According to Equations from 7 to 9, $w_g = \cos^2(0°) = 1$, $w_c = \sin^2(0°) = 0$, and $CollaborateDegree_i^x = GainRatio_i^x$, so the agent ID_x selects its partners by considering only the *GainRatio*, which accords with the agent's expected behavior;
- when $ReliantDegree = 45°$, the agent should consider both self-interest and partners' benefit equally. In this case, $w_g = \cos^2(45°) = 0.5$, $w_c = \sin^2(0°) = 0.5$, and $CollaborateDegree_i^x = (GainRatio_i^x + ContributionRatio_i^x)/2$, so the agent ID_x plays an equitable strategy on partners selection;
- when $ReliantDegree = 90°$, it is supposed that the agent ID_x should perform as a selfless agent and only considers about parters' benefits. In this situation, $w_g = \cos^2(90°) = 0$, $w_c = \sin^2(90°) = 1$, and $CollaborateDegree_i^x = ContributionRatio_i^x$, so the agent perform as selfless.

Finally, by combining the Equation 7 - Equation 9, the potential partners are evaluated by considering the factors of *GainRatio*, *ContributionRatio* and *ReliantDegree*. The *collaborationDegree* between the agent ID_x and its potential partner ID_i is:

$$\begin{aligned} CollaborateDegree_i^x = {} & GainRatio_i^x \times \cos^2(ReliantDegree_i^x) + \\ & ContributionRatio_i^x \times \sin^2(ReliantDegree_i^x) \end{aligned} \tag{10}$$

where $CollaborateDegree_i^x \in [0, 1]$.

Then by calculating each collaboration degree between the agent ID_x and its potential partners, the collaboration degrees set ($CollaborateDegree^x$) are generated, which is:

$$CollaborateDegree^x = \{CollaborateDegree_i^x\}, i \in [1, n] \tag{11}$$

Finally, any sorting algorithm can be employed to select favorable partners or exclude unsuitable partners from the collaboration degree set $CollaborateDegree^x$ for the agent ID_x.

In this section, we proposed a linear approach for partners selection in multi-agent negotiation. By employing the factor of $ReliantDegree_i^x$ and Equations 8 and 9, the relationships between the agent and its partners are captured and used in the partners selection process. The advantages of this linear approach are that (1) it is easy to be implemented and (2) it solves problems directly where relationships of agents can be represented by linear functions. However, in some cases, the relationships between agents and their partners cannot be represented by a linear function. For example, when the negotiation issues are discrete such as the cars' colors or manufactories, the linear approach hardly can represent

the problem efficiently and is not competent for new situation anymore. In order to address this issue, we will introduce another non-linear partners selection approach in next section, which employs the fuzzy logic mechanism to capture the agent's preference.

5 Partners Selection by Using a Non-linear Approach

In the last section, we proposed a linear approach to select partners for agents. However, in some cases, human's behavior cannot be modeled by a linear function simply. So in order to mimic real world situation and generate more reasonable selection results, we propose a non-linear approach for partners selection in this section. We employ the fuzzy logic [26, 27] mechanism in the proposed non-linear approach. The structure of this section is organized as follows. In subsection 5.1, the principle of this non-linear approach is introduced and a basic framework of the approach is proposed. From section 5.2 to 5.4, the methods of fuzzification, approximate reasoning, and defuzzification are introduced in detail, respectively.

5.1 Framework of a Fuzzy-Based Approach

Based on the analysis in Section 3, we propose a non-linear approach that agents can select their suitable partners dynamically by considering individual benefits, global benefits and agents' behaviors.

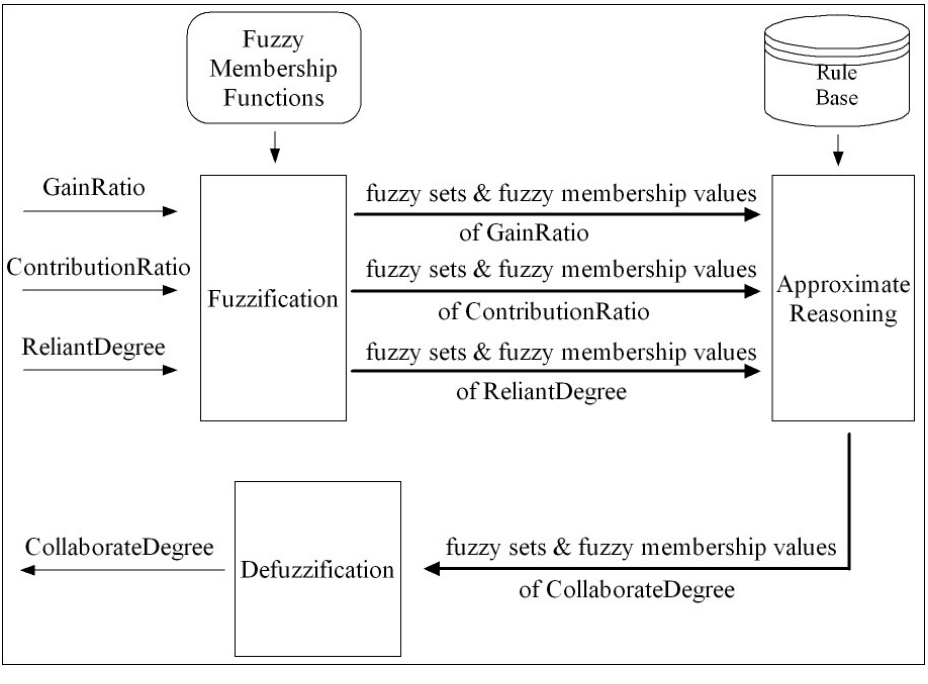


Fig. 3. The framework of the non-linear partners selection approach

The framework of the non-linear approach for selecting most suitable partners in MASs is graphically illustrated in Figure 3. There are five units in this approach, which are: (1) a library of fuzzy functions, (2) a fuzzy rule base, (3) a fuzzification module, (4) an approximate reasoning module, and (5) a defuzzification module.

The input parameters of the framework are *GainRatio*, *ContributionRatio* and *ReliantRatio* which have defined in Section 3. The output of this framework provides suggestions to agent for partners selection by considering these three factors.

5.2 Fuzzification

Fuzzy Membership Functions for Input Parameters: For these three input parameters (*GainRatio*, *ContributionRatio* and *ReliantDegree*), the linguistic states for them are defined as follows.

– ***GainRatio.*** For the input parameter *GainRatio*, five linguistic states are selected and expressed by appropriate fuzzy sets which are {*VerySmall*, *Small*, *Medium*, *Large* and *VeryLarge*}.
Figure 4 depicts these fuzzy sets as applied to parameter *GainRatio*. The triangle membership function [29] is adopted here to define fuzzy memberships. Fuzzy membership functions for fuzzy sets {*VerySmall*, *Small*, *Medium*, *Large*, *VeryLarge*} are defined from Equation 12 to 16, respectively.

$$F_{VerySmall}(x) = \begin{cases} \frac{20-x}{20} & 0 \le x \le 20 \\ 0 & x > 20 \end{cases} \tag{12}$$

$$F_{Small}(x) = \begin{cases} 0 & x \le 10 \\ \frac{x-10}{20} & 10 < x \le 30 \\ \frac{50-x}{20} & 30 < x \le 50 \\ 0 & x > 20 \end{cases} \tag{13}$$

$$F_{Medium}(x) = \begin{cases} 0 & x \le 30 \\ \frac{x-30}{20} & 30 < x \le 50 \\ \frac{70-x}{20} & 50 < x \le 70 \\ 0 & x > 70 \end{cases} \tag{14}$$

$$F_{Large}(x) = \begin{cases} 0 & x \le 50 \\ \frac{x-50}{20} & 50 < x \le 70 \\ \frac{90-x}{20} & 70 < x \le 90 \\ 0 & x > 90 \end{cases} \tag{15}$$

$$F_{VeryLarge}(x) = \begin{cases} 0 & x \le 80 \\ \frac{x-80}{20} & x > 80 \end{cases} \tag{16}$$

where $x \in [0, 100]$

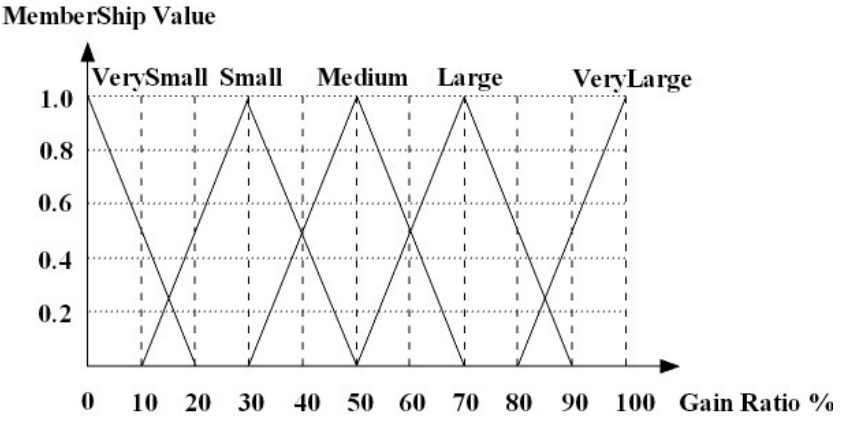


Fig. 4. Fuzzy quantization of range [0, 100] for *GainRatio*

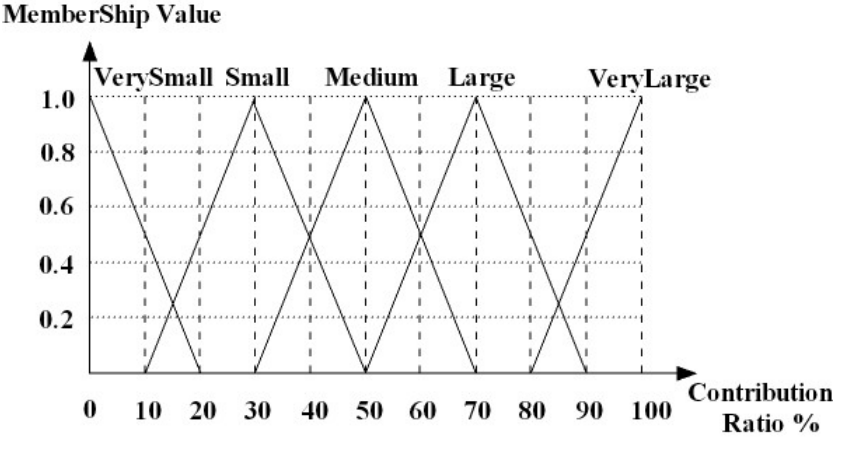


Fig. 5. Fuzzy quantization of range [0, 100] for *ContributionRatio*

- ***ContributionRatio.*** For the parameter *ContributionRatio*, both the fuzzy sets and membership functions are same as *GainRatio*'s (Equation 12-16). Figure 5 depicts the fuzzy sets as applied to parameter *ContributionRatio*.
- ***ReliantDegree.*** For the parameter *ReliantDegree*, five linguistic states are selected and expressed by appropriate fuzzy sets, which are {*Complete Self-Driven*, *Self-Driven*, *Equitable*, *External-Driven*, *Complete External-Driven*}. Figure 6 depicts these fuzzy sets as applied to parameter *ReliantDegree*. Fuzzy membership functions for fuzzy sets {*Complete Self-Driven*, *Self-Driven*, *Equitable*, *External-Driven*, *Complete External-Driven*} are defined from Equation 17 to 21, respectively.

$$F_{CompleteSelfDriven}(x) = \begin{cases} \frac{22.5-x}{22.5} & 0 \leq x \leq 22.5 \\ 0 & x > 22.5 \end{cases} \tag{17}$$

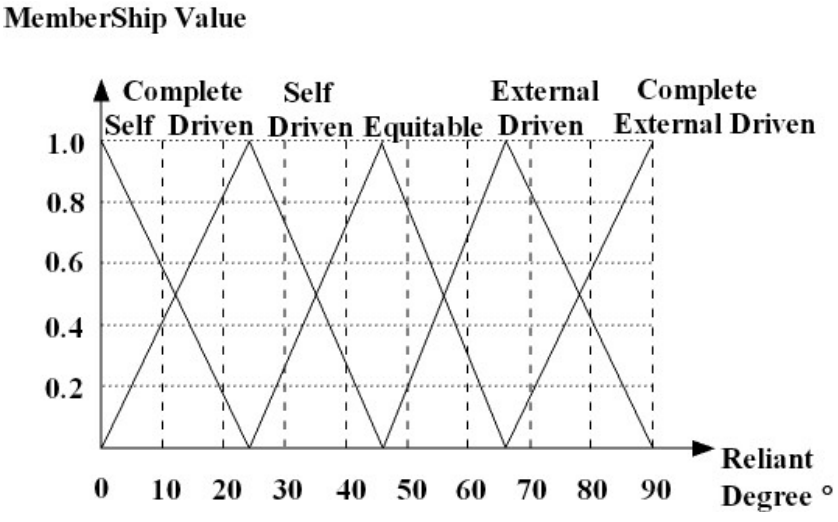


Fig. 6. Fuzzy quantization of range $[0°, 90°]$ for *ReliantDegree*

$$F_{SelfDriven}(x) = \begin{cases} \frac{x}{22.5} & 0 < x \leq 22.5 \\ \frac{45-x}{22.5} & 22.5 < x \leq 45 \\ 0 & x > 45 \end{cases} \tag{18}$$

$$F_{Equitable}(x) = \begin{cases} 0 & x \leq 22.5 \\ \frac{x-22.5}{22.5} & 22.5 < x \leq 45 \\ \frac{67.5-x}{22.5} & 45 < x \leq 67.5 \\ 0 & x > 67.5 \end{cases} \tag{19}$$

$$F_{ExternalDriven}(x) = \begin{cases} 0 & x \leq 45 \\ \frac{x-45}{22.5} & 45 < x \leq 67.5 \\ \frac{90-x}{22.5} & 67.5 < x \leq 90 \end{cases} \tag{20}$$

$$F_{CompleteExternalDriven}(x) = \begin{cases} 0 & x \leq 67.5 \\ \frac{x-67.5}{22.5} & x > 67.5 \end{cases} \tag{21}$$

where $x \in [0°, 90°]$.

Fuzzy Membership Functions for Output Parameters: For the output parameter *CollaborateDegree*, five linguistic states are selected and expressed by corresponding fuzzy sets {*Averse*, *Reluctant*, *Indifferent*, *Acceptable*, and *Anticipant*}. Figure 7 depicts these fuzzy sets as applied to parameter *CollaborateDegree*. The

fuzzy membership functions for parameter *CollaborateDegree* are defined through Equation 22-26.

$$F_{Averse}(x) = \begin{cases} \frac{20-x}{20} & 0 \le x \le 20 \\ 0 & x > 20 \end{cases} \tag{22}$$

$$F_{Reluctant}(x) = \begin{cases} 0 & x \le 10 \\ \frac{x-10}{20} & 10 < x \le 30 \\ \frac{50-x}{20} & 30 < x \le 50 \\ 0 & x > 20 \end{cases} \tag{23}$$

$$F_{Indifferent}(x) = \begin{cases} 0 & x \le 30 \\ \frac{x-30}{20} & 30 < x \le 50 \\ \frac{70-x}{20} & 50 < x \le 70 \\ 0 & x > 70 \end{cases} \tag{24}$$

$$F_{Acceptable}(x) = \begin{cases} 0 & x \le 50 \\ \frac{x-50}{20} & 50 < x \le 70 \\ \frac{90-x}{20} & 70 < x \le 90 \\ 0 & x > 90 \end{cases} \tag{25}$$

$$F_{Anticipant}(x) = \begin{cases} 0 & x \le 80 \\ \frac{x-80}{20} & x > 80 \end{cases} \tag{26}$$

where $x \in [0, 100]$

5.3 Approximate Reasoning

The approximate reasoning is hired to calculate output membership values, which can further be used to compute corresponding output values. The approximate reasoning is based on the use of rules in the rule base.

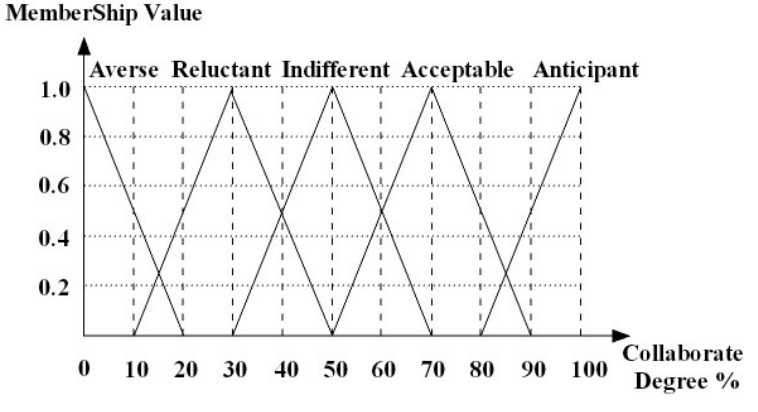


Fig. 7. Fuzzy quantization of range [0, 100] for *CollaborateDegree*

Rule Base: A rule base is a matrix of combinations of each of the input linguistic parameters. The rule bases in this approach are displayed in Table 1-5.

Table 1. Fuzzy rule base (*ReliantDegree=Complete Self-Driven*)

GainRetio	*ContributionRetio*	*CollaborateDegree*
VerySmall	Any	*Averse*
Small	Any	*Reluctant*
Medium	Any	*Indifferent*
Large	Any	*Acceptable*
VeryLarge	Any	*Anticipant*

Table 2. Fuzzy rule base (*ReliantDegree=Self-Driven*)

GainRetio	*ContributionRetio*	*CollaborateDegree*
VerySmall	*VeryLarge*	*Reluctant*
	Others	*Averse*
Small	*VeryLarge*	*Indifferent*
	Others	*Reluctant*
Medium	*VeryLarge*	*Acceptable*
	Large	*Acceptable*
	Others	*Indifferent*
Large	*VerySmall*	*Indifferent*
	Others	*Acceptable*
VeryLarge	*VerySmall*	*Acceptable*
	Others	*Anticipant*

The Determination of Output Membership Values: Each entry of the rule base is a rule, which is defined by *AND*ing three linguistic input parameters to produce an individual output, in the form of:

$$\begin{aligned}\textbf{IF}((F(GainRatio)=\alpha)&\textbf{AND}(F(ContributionRatio)=\beta)\\ &\textbf{AND}(F(ReliantDegree)=\gamma))\\ &\textbf{THEN}F(collaborateDegree)=\delta\end{aligned} \tag{27}$$

where, $\alpha \in \{VerySmall,\ Small,\ Medium,\ Large,\ VeryLarge\}$, $\beta \in \{VerySmall,\ Small,\ Medium,\ Large,\ VeryLarge\}$, $\gamma \in \{Complete\ Self\text{-}Driven,\ Self\text{-}Driven,\ Equitable,\ External\text{-}Driven,\ Complete\ External\text{-}Driven\}$, $\delta \in \{Averse,\ Reluctant,\ Indifferent,\ Acceptable,\ Anticipant\}$, and *F(collaborateDegree)* denotes a fuzzy set into which the parameter *collaborateDegree* is mapped.

An output membership value $\mu_\delta(\nu)$ can be calculated by Equation 28.

$$\begin{aligned}\mu_\delta(\nu) = MIN(&\mu_\alpha(GainRatio), \mu_\beta(ContributionRatio),\\ &\mu_\gamma(ReliantDegree))\end{aligned} \tag{28}$$

Table 3. Fuzzy rule base (*ReliantDegree=Equitable*)

GainRetio	*ContributionRetio*	*CollaborateDegree*
VerySmall	*VeryLarge*	*Indifferent*
	Large	*Reluctant*
	Medium	*Averse*
	Small	*Averse*
	VerySmall	*Averse*
Small	*VeryLarge*	*Acceptable*
	Large	*Indifferent*
	Medium	*Reluctant*
	Small	*Averse*
	VerySmall	*Averse*
Medium	*VeryLarge*	*Anticipant*
	Large	*Acceptable*
	Medium	*Indifferent*
	Small	*Reluctant*
	VerySmall	*Averse*
Large	*VerySmall*	*Anticipant*
	Large	*Anticipant*
	Medium	*Acceptable*
	Small	*Indifferent*
	VerySmall	*Reluctant*
VeryLarge	*VerySmall*	*Anticipant*
	Large	*Anticipant*
	Medium	*Anticipant*
	Small	*Acceptable*
	VerySmall	*Indifferent*

Table 4. Fuzzy rule base (*ReliantDegree=External-Driven*)

ContributionRetio	*GainRetio*	*CollaborateDegree*
VerySmall	*VeryLarge*	*Reluctant*
	Others	*Averse*
Small	*VeryLarge*	*Indifferent*
	Others	*Reluctant*
Medium	*VeryLarge*	*Acceptable*
	Large	*Acceptable*
	Others	*Indifferent*
Large	*VerySmall*	*Indifferent*
	Others	*Acceptable*
VeryLarge	*VerySmall*	*Acceptable*
	Others	*Anticipant*

Table 5. Fuzzy rule base (*ReliantDegree*=*Complete External-Driven*)

ContributionRetio	*GainRetio*	*CollaborateDegree*
VerySmall	Any	*Averse*
Small	Any	*Reluctant*
Medium	Any	*Indifferent*
Large	Any	*Acceptable*
VeryLarge	Any	*Anticipant*

5.4 Defuzzification

There are many defuzzification approaches. The centroid defuzzification method [29] is one approach to defuzzify the output membership values.

$$DV = \frac{\sum_{i=1}^{k}(\nu_i \times \mu(\nu_i))}{\sum_{i=1}^{k}\mu(\nu_i)} \tag{29}$$

where $\mu(\nu_i)$ is the ith output membership value, ν_i is its corresponding output value, and k is the number of fuzzy rules which are activated.

DV is the final output value of *collaborateDegree* in a particular case. *DV* can be used to evaluate the relationship between the agent and its potential partners, and can also be used as an important factor for selecting or adopting a most suitable partner for an agent in a particular case.

5.5 Summary

In this section, we proposed a non-linear approach for the partners selection. By comparing with the linear approach introduced in Section 4, the non-linear approach is more logical and accurate. However, it needs more complex process to achieve the selection results. In next section, we will give some examples to test these two proposed approaches under different situations.

6 Case Study

In this session, four scenarios are demonstrated. In each example, agent g is going to select the most suitable partner from three potential partners (agent g_a, g_b and g_c). These examples will illustrate how the proposed approaches select the most suitable partner for the agent g.

6.1 Scenario 1

In scenario 1, all of three potential partners share a common *ReliantDegree*, which is 0°. The agent g is a *Complete Self-Driven* agent so that agent g_a should be selected as the most suitable partner because it can contribute the highest *GainRatio* to agent g among three potential partners. All input parameters for the three potential partners are shown in Table 8.

Table 6. Input parameters for scenario 1

Partner	*GainRatio*	*ContributionRatio*	*ReliantDegree*
g_a	80%	20%	0°
g_b	50%	50%	0°
g_c	20%	80%	0°

In Table 7, the selection results by using the proposed linear function are presented. Since $w_g = cos^2(0°) = 1$ and $w_c = sin^2(0°) = 0$, the *GainRatio* is used to make the final decision. Therefore, agent g_a is selected by the agent g as the most preferable partner, which is same as the estimation.

Table 7. Output for scenario 1 by using the linear function

Partner	$w_g \times$ *GainRatio*	$w_c \times$ *ContributionRatio*	*CollaborateDegree*
g_a	0.8	0	0.8
g_b	0.5	0	0.5
g_c	0.2	0	0.2

The results for scenario 1 by using the proposed non-linear function are shown in Table 8. According to the selection results generated by the proposed approach, agent g_a is the most suitable partner which is same as the selection result by the linear approach.

Table 8. Output for example 1 by using the non-linear function

Partner	*ReliantDegree*	*GainRatio*	*ContributionRatio*	*CollaborateDegree*	Defuzzification
g_a	*Complete Self-Driven*=1	*Large*=0.5	*Small*=0.5	*Acceptable*=0.5	70%
g_b	*Complete Self-Driven*=1	*Medium*=1	*Medium*=1	*Indifferent*=1	50%
g_c	*Self-Driven*=1	*Small*=0.5	*Large*=0.5	*Reluctant*=0.5	30%

6.2 Scenario 2

In scenario 2, all of three potential partners share a common *ReliantDegree*, which is 90°. Both the *GainRatio* and *ContributionRatio* are same as scenario 1. The agent is a *Complete External-Driven* agent so that agent g_c should be selected as the most suitable partner because it has the largest *ContributionRatio*. All input parameters for the three potential partners are shown in Table 9.

Since *ReliantDegree* is 90°, the $w_g = 0$ and $w_c = 1$ according to Equations 8 and 9. The agent g selects partner based on *ContributionRatio* only. The selection results by using the linear function are displayed in Table. According

Table 9. Input parameters for scenario 2

Partner	*GainRatio*	*ContributionRatio*	*ReliantDegree*
g_a	80%	20%	90°
g_b	50%	50%	90°
g_c	20%	80%	90°

to these results, the agent g_c should be selected as the most suitable partner for the agent g. This selection result is exactly same as the output by using the non-linear function as shown in Table.

Table 10. Output for scenario 2 by using the linear function

Partner	$w_g \times$ *GainRatio*	$w_c \times$ *ContributionRatio*	*CollaborateDegree*
g_a	0	0.2	0.2
g_b	0	0.5	0.5
g_c	0	0.8	0.8

Table 11. Output for scenario 2 by using the non-linear function

Partner	*ReliantDegree*	*ContributionRatio*	*GainRatio*	*CollaborateDegree*	Defuzzification
g_a	*Complete External-Driven*=1	*Small*=0.5	*Large*=0.5	*Reluctant*=0.5	30%
g_b	*External-Driven*=1	*Medium*=1	*Medium*=1	*Indifferent*=1	50%
g_c	*Self-Driven*=1	*Large*=0.5	*Small*=0.5	*Acceptable*=0.5	70%

6.3 Scenario 3

In scenario 3, *ReliantDegree* = 45° and all others parameter are same as scenario 1. In this case, the agent g is an *Equitable* agent so that the estimation partner is any one of the potential partners by considering both *GainRatio* and *ContributionRatio* equally. All input parameters for the three potential partners are shown in Table 12.

Table 12. Input parameters for example 3

Partner	*GainRatio*	*ContributionRatio*	*ReliantDegree*
g_a	80%	20%	45°
g_b	50%	50%	45°
g_c	20%	80%	45°

According to the proposed linear function, since *ReliantDegree* = 45°, then $w_g = w_c = 0.5$. The *CollaborateDegree* for all potential partners are exactly same (0.5) as shown in Table 13. So either one could be selected as the most

Table 13. Output for scenario 3 by using the linear function

Partner	$w_g \times GainRatio$	$w_c \times ContributionRatio$	$CollaborateDegree$
g_a	0.4	0.1	0.5
g_b	0.25	0.25	0.5
g_c	0.1	0.4	0.5

Table 14. Output for scenario 3 by using the non-linear function

Partner	*ReliantDegree*	*GainRatio*	*ContributionRatio*	*CollaborateDegree*	Defuzzification
g_a	*Equitable*=1	*Large*=0.5	*Small*=0.5	*Indifferent*=0.5	50%
g_b	*Equitable*=1	*Medium*=1	*Medium*=1	*Indifferent*=1	50%
g_c	*Equitable*=1	*Small*=0.5	*Large*=0.5	*Indifferent*=0.5	50%

preferable partner for the agent g. The non-linear approach's selection results are displayed in Table 14. It also suggests that any potential partner could be the most suitable partner in this case.

6.4 Scenario 4

In scenario 4, all of three potential partners share common *GainRatio* and *ContributionRatio*, but with different *ReliantDegree*. The agent g has different attitudes to its potential partners. For potential partner g_a, agent g performs as a *Complete Self-Driven* agent so that only the *GainRatio* (80%) will be used to select the most suitable partner. For potential partner g_b, agent g performs as a *Equitable* agent so that both *GainRatio* (80%) and *ContributionRatio* (20%) will be used to evaluate whether g_b could be chosen as a suitable partner. Therefore, the final benefit by considering both *GainRatio* and *ContributionRatio* for g_b should be between 20% and 80%. For potential partner g_c, agent g performs as a *Complete External-Driven* agent so that only the benefit of *ContributionRatio* (20%) will be used for the selection of g_c as a partner. By comparing the three cases, agent g_a should be selected as the most suitable partner because agent g would gain the largest benefit(80%) when collaborating with agent g_a. All input parameters for the three potential partners are shown in Table 15.

By employing the proposed linear approach, partner g_a is selected as the most suitable partner shown in Table 16. Because these three partners can offer same *GainRatio* and *ContributionRatio*, the relationships between the agent g and each one of them are crucial for the partner selection in this case. As the

Table 15. Input parameters for scenario 4

Agent	*GainRatio*	*ContributionRatio*	*ReliantDegree*
g_a	80%	20%	0°
g_b	80%	20%	45°
g_c	80%	20%	90°

Table 16. Output for scenario 4 by using the linear function

Partner	$w_g \times GainRatio$	$w_c \times ContributionRatio$	$CollaborateDegree$
g_a	0.8	0	0.8
g_b	0.4	0.1	0.5
g_c	0	0.2	0.2

Table 17. Output for scenario 4 by using the non-linear function

Agent	*ReliantDegree*	*GainRatio*	*ContributionRatio*	*CollaborateDegree*	Defuzzification
g_a	*Self-Driven*=1	*Large*=0.5	*Small*=0.5	*Acceptable*=0.5	70%
g_b	*Equitable*=1	*Large*=0.5	*Small*=0.5	*Indifferent*=0.5	50%
g_c	*External-Driven*=1	*Small*=0.5	*Large*=0.5	*Reluctant*=0.5	30%

GainRatio is bigger than *ContributionRatio*, and the agent g's attitude on the partner g_a is selfness, so g_a is the most preferable partner. In Table 17, the selection results by employing the non-linear function are presented, which is same as the linear approach.

Therefore, from these four different scenarios, it can be seen that by considering the factors of *GainRatio*, *ContributionRatio* and *ReliantDegree* between the agent and its potential partners, both of our proposed partner selection mechanisms can be employed by agents to generate reasonable judgement on their potential partners and to select the most suitable partner for the agent. Also, it is noticed that both of the proposed approach are easy to be administrated by agents in an dynamic negotiation environment to filter partners according to agents' expectations. However, the proposed linear and non-linear approaches have their own individual merits. The linear approach is suitable for the negotiation environment where the situation is not complex and the requirement on accuracy is not very severe, but asking for a quicker responds on any change of the situation. While the non-linear approach can be employed in a more complex negotiation, where agents' behaviors cannot be represented by linear function simply. The non-linear function is sensitive to the change of situation and can generate more reasonable and accurate selection results by employing the predefined fuzzy linguistic languages and membership functions. Therefore, the purpose of proposing two partners selection functions is to satisfy all kinds of requirements in different negotiation environments.

7 Conclusion and Further Work

In this paper, we identified four potential cases of relationships between an agent and its potential partners. Both a linear and non-linear partner selection approaches were proposed. For the linear approach, the *ReliantDegree* is employed to calculate the normalized weights for both *GainRatio* and *ContributionRatio*. Agents attitudes on their potential partners are represented and controlled by

these two normalized weights. For the non-linear approach, a framework is proposed which consists a fuzzification module, a fuzzy rule base, an approximate reasoning module, a defuzzification module, and a library of fuzzy membership functions. All of the fuzzy membership functions for corresponding fuzzy sets have been carefully defined and rules of fuzzy logic operations during the procedure of approximate reasoning have also been defined.

The further works on this research will focus on how to extend the current approaches for the partners selection under the circumstances of (1) when considering the factors of punishment, compensation and successful ratio; and (2) when considering the issue of multi-attribute negotiation.

References

1. Rubinstein, Z., Corkill, D.: Mixed-Initiative Management of Dynamic Business Processes. In: Proceedings of the 2003 IEEE International Workshop on Soft Computing in Industrial Applications, Binghamton, New York, June 2003, pp. 39–44 (2003)
2. Li, C., Sycara, K., Giampapa, J.: Dynamic Outside Options in Alternating-Offers Negotiations. In: HICSS, IEEE Computer Society Press, Los Alamitos (2005)
3. Faratin, P., Sierra, C., Jennings, N.: Negotiation Decision Functions for Autonomous Agents. Journal of Robotics and Autonomous Systems 24(3-4), 159–182 (1998)
4. Lai, G., Li, C., Sycara, K.: A General Model for Pareto Optimall Multi-Attribute Negotiations. In: Second International Workshop on Rational, Robust, and Secure Negotiations in Multi-Agent Systems (RRS 2006), Future University, Hakodate, Japan, May 2006, pp. 55–76 (2006)
5. Sierra, C., Faratin, P., Jennings, N.: A Service-Oriented Negotiation Model between Autonomous Agents. In: Boman, M., Van de Velde, W. (eds.) MAAMAW 1997. LNCS, vol. 1237, pp. 17–35. Springer, Heidelberg (1997)
6. Rahwan, I., Ramchurn, S., Jennings, N., McBurney, P., Parsons, S., Sonenberg, L.: Argumentation-Based Negotiation. The Knowledge Engineering Review 18(4), 343–375 (2004)
7. Parsons, S., Sierra, C., Jennings, N.: Agents that Reason and Negotiate by Arguing. Journal of Logic and Computation 8(3), 261–292 (1998)
8. Amgoud, L., Dimopoulos, Y., Moraitis, P.: A Unified and General Framework for Argumentation-based Negotiation. In: Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, Hawai'i, IFAAMAS, May 2007, pp. 963–970 (2007)
9. nón, S.O., Plaza, E.: Learning and Joint Deliberation through Argumentation in Multi-Agent Systems. In: Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, hawai'i, IFAAMAS May, 2007, pp. 971–978 (2007)
10. Kraus, S.: Strategic Negotiation in Multiagent Environments. MIT Press, Cambridge, MA (2001)
11. Ren, F., Sim, K., Zhang, M.: Market-Driven Agents with Uncertain and Dynamic Outside Options. In: Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, Hawai'i (2007)

12. Sim, K., Choi, C.: Agents That React to Changing Market Situations. In: IEEE Transaction on Systems, Man and Cybernetics, Part B: Cybernetics, April 2003, vol. 33, pp. 188–201 (2003)
13. Zhang, X., Lesser, V., Wagner, T.: Integrative Negotiation In Complex Organizational Agent Systems. In: Gini, M., Ishida, T., Castelfranchi, C., Johnson, W.L. (eds.) Proceedings of the First International Joint Conference on Autonomous Agents & MultiAgent Systems (AAMAS 2002), pp. 503–504 (July 2002)
14. Jung, H., Tambe, M., Kulkarni, S.: Argumentation as Distributed Constraint Satisfaction: Applications and Results. In: Proceedings of the Fifth International Conference on Autonomous Agents, Montreal, Canada, pp. 324–331. ACM Press, New York (2001)
15. Dondio, P., Barrett, S.: Presumptive Selection of Trust Evidence. In: Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, hawai'i, IFAAMAS, May 2007, pp. 1071–1078 (2007)
16. Fullam, K., Barber, K.: Dynamically Learning Sources of Trust Information: Experience vs. Reputation. In: Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, hawai'i, IFAAMAS, May 2007, pp. 1055–1062 (2007)
17. Reece, S., Rogers, A., Roberts, S., Jennings, N.: Rumours and Reputation: Evaluating Multi-Dimensional Trust within A Decentralised Reputation System. In: Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, hawai'i, IFAAMAS, May 2007, pp. 1063–1070 (2007)
18. Munroe, S., Luck, M.: Balancing Conflict and Cost in the Selection of Negotiation Opponents. In: First International Workshop on Rational, Robust, and Secure Negotiations in Multi-Agent Systems (RRS 2005), Amsterdam, Netherlands, pp. 39–53 (July 2005)
19. Fatima, S., Wooldridge, M., Jennings, N.: The Influence of Information on Negotiation Equilibrium. In: Padget, J.A., Shehory, O., Parkes, D.C., Sadeh, N.M., Walsh, W.E. (eds.) AMEC 2002. LNCS (LNAI), vol. 2531, Springer, Heidelberg (2002)
20. Brzostowski, J., Kowalczyk, R.: On Possibilistic Case-Based Reasoning for Selecting Partners for Multi-Attribute Agent Negotiation. In: 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), University Utrecht, The Netherlands, pp. 273–279. ACM, New York (2005)
21. Brzostowski, J., Kowalczyk, R.: On Possibilistic Case-Based Reasoning for Selecting Partners in Multi-agent Negotiation. In: Webb, G.I., Yu, X. (eds.) AI 2004. LNCS (LNAI), vol. 3339, Springer, Heidelberg (2004)
22. Munroe, S., Luck, M., d'Inverno, M.: Motivation-Based Selection of Negotiation Partners. In: 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), pp. 1520–1521. IEEE Computer Society, Los Alamitos (2004)
23. Brzostowski, J., Kowalczyk, R.: Negotiation Partners Selection Mechanism based on Context-Dependent Similarity Relations. In: Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, hawai'i, IFAAMAS, May 2007, pp. 1037–1039 (2007)
24. Leake, D. (ed.): Case-Based Reasoning: Experiences, Lessons, and Future Directions. AAAI Press/MIT Press, Menlo Park, CA (1996)
25. Gilboa, I., Schmeidler, D.: Case-Based Decision Theory. The Quarterly Journal of Economics 110(3), 605–639 (1995)
26. Yan, J., Ryan, M., Power, J. (eds.): Using Fuzzy Logic. Prentice Hall, Upper Saddle River, NJ (1994)

27. Klir, G., Yuan, B. (eds.): Fuzzy Sets and Fuzzy Logic Theory and Applications. Prentice Hall, Upper Saddle River, NJ (1995)
28. Zhang, X., Lesser, V., Wagner, T.: Integrative Negotiation among Agents Situated in Organizations. IEEE Transactions on Systems, Man, and Cybernetics, Part C 36(1), 19–30 (2006)
29. Eberhart, R., Simpson, P., Dobbin, R. (eds.): Computational Intelligence PC Tools. AP Professional Press, Orlando, USA (1996)

Topology Representing Network Map – A New Tool for Visualization of High-Dimensional Data

Agnes Vathy-Fogarassy[1], Attila Kiss[2], and Janos Abonyi[3]

[1] University of Pannonia, Department of Mathematics and Computing, P.O. Box 158,Veszprem, H-8201 Hungary
vathya@almos.uni-pannon.hu
[2] Eötvös Lóránd University, Department of Information Systems, P.O. Box 120, H-1518 Budapest, Hungary
kiss@ullman.inf.elte.hu
[3] University of Pannonia, Department of Process Engineering, P.O. Box 158, Veszprem, H-8201 Hungary
abonyij@fmt.uni-pannon.hu
http://www.fmt.uni-pannon.hu/softcomp

Abstract. In practical data mining problems high-dimensional data has to be analyzed. In most of these cases it is very informative to map and visualize the hidden structure of complex data set in a low-dimensional space. The aim of this paper is to propose a new mapping algorithm based both on the topology and the metric of the data.

The utilized Topology Representing Network (TRN) combines neural gas vector quantization and competitive Hebbian learning rule in such a way that the hidden data structure is approximated by a compact graph representation. TRN is able to define a low-dimensional manifold in the high-dimensional feature space. In case the existence of a manifold, multidimensional scaling and/or Sammon mapping of the graph distances can be used to form the map of the TRN (TRNMap).

The systematic analysis of the algorithms that can be used for data visualization and the numerical examples presented in this paper demonstrate that the resulting map gives a good representation of the topology and the metric of complex data sets, and the component plane representation of TRNMap is useful to explore the hidden relations among the features.

Keywords: Data visualization, Dimensionality reduction, Topology Representing Networks, Isomap, CDA.

1 Introduction

1.1 Classical Dimension Reduction Algorithms

Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis that employs a variety of techniques (mostly graphical) to maximize insight into

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 61–84, 2008.

a data set, uncover underlying structure, extract important variables, detect outliers and anomalies, and test underlying assumptions. The seminal work in EDA is written by Tukey [30]. Most EDA techniques are graphical in nature with a few quantitative techniques [21]. The role of EDA is to open-mindedly explore the data. The main defect of EDA is arising from the limit of the visualization. Data can be thought as points in a high-dimensional vector space, with each dimension corresponding to an attribute of the observed object. Because humans simply can not see high-dimensional data, it is necessary to reduce the dimensionality of the data. The reduction of dimensionality of the feature space is also important because of the curse of dimensionality. In a nutshell, same number of examples fill more of the available space when the dimensionality is low, and its consequence is that exponential growth with dimensionality in the number of examples is required to accurately estimate a function. Hence, dimensionality reduction is an important task of (visual) data mining.

The goal of dimensionality reduction is to map a set of observations from a high-dimensional space (D) into a low-dimensional space (d, $d \ll D$) preserving as much of the intrinsic structure of the data as possible. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ be a set of the observed data, where $\mathbf{x}_i$ denotes the i-th observation ($\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,D}]^T$). Each data object is characterized by D dimensions, so $x_{i,j}$ yields the j-th ($j = 1, 2, \ldots, D$) attribute of the i-th ($i = 1, 2, \ldots, N$) data object. Dimensionality reduction techniques transform data set $\mathbf{X}$ into a new data set $\mathbf{Y}$ with dimensionality d ($\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$, $\mathbf{y}_i = [y_{i,1}, y_{i,2}, \ldots, y_{i,d}]^T$). In the reduced space many data analysis tasks (e.g. classification, clustering, image recognition) can be carried out faster than in the original data space.

Three types of dimensionality reduction methods can be distinguished: (i) metric methods try to preserve the distances of the data defined by a metric, (ii) non-metric methods try to preserve the global ordering relations of the data, (iii) other methods that differ from the previously introduced two groups.

The most widely applied dimensionality reduction method is the Principal Component Analysis (PCA) [10,12], a member of the last class. PCA represents the data as linear combinations of a small number of basis vectors. The method finds the projection that stores the largest variance possible in the original data and rotates the set of the objects such that the maximum variability becomes visible. Accordingly, PCA tries to preserve the variance of the input data elements. Geometrically, PCA transforms the data into a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate, the second greatest variance on the second coordinate, and so on. The main steps of the PCA are the follows: (1) subtract the mean from each of the data dimensions (2) calculate the covariance matrix (3) calculate the eigenvectors and the eigenvalues of the covariance matrix (4) choose the main significant eigenvectors (5) derive the new data set from the significant eigenvectors and from the original data matrix. Independent Component Analysis (ICA) [4] is similar to PCA, except that it tries to find components that are independent.

In the following let us have a closer look at the mostly used metric dimensionality reduction methods. Multidimensional scaling (MDS) [3] refers to a group of unsupervised data visualization techniques. Given a set of data in a high-dimensional feature space, MDS maps them into a low-dimensional (generally 2-dimensional) data space in a way that objects that are very similar to each other in the original space are placed near each other on the map, and objects that are very different from each other are placed far away from each other. There are two types of MDS: (i) metric MDS and (ii) non-metric MDS. While metric MDS discovers the underlying structure of data set by preserving similarity information (pair wise distances) among the data objects, non-metric MDS attempts to preserve the rank order among the dissimilarities.

Sammon mapping (SM) [27] is a metric, nonlinear dimensionality reduction method which maps the set of points in a high-dimensional vector space onto a 2-dimensional output space. It tries to optimize the cost function that describes how well the pairwise distances in a data set are preserved. The Sammon stress function (distortion of the Sammon projection) can be written as:

$$E_{SM} = \frac{1}{\sum\limits_{i<j}^{N} d^*_{i,j}} \sum_{i<j}^{N} \frac{(d^*_{i,j} - d_{i,j})^2}{d^*_{i,j}}, \tag{1}$$

where $d^*_{i,j}$ denotes the distance between the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, and $d_{i,j}$ respectively for $\mathbf{y}_i$ and $\mathbf{y}_j$.

1.2 Topology Based Visualization Techniques - Geodesic Distance Measures

Dimensionality reduction methods in many cases are confronted with low-dimensional structures hidden in the high-dimensional space. In these cases the Euclidean distance, which is the length of the straight line between two points, is not suitable to compute distances among the data points. The problem of the Euclidean distance is that this straight line can go through shortcuts outside the data cloud. The geodesic distance between two points of manifold is computed in such a way that it always goes along the manifold. Using this distance it can be given more essential description about the data distances. Obviously, in many cases only a few points of the manifold are known. In these cases the geodesic distances can be approximated by computing the graph distances [2]. To compute graph distances a graph should be built on the data. There are two basic variations to construct the adjacency graph: (i) ϵ-neighboring and (ii) k-neighboring. In the case of the first approach objects $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected by an edge, if $\|\mathbf{x}_i - \mathbf{x}_j\|^2 < \epsilon$, where the norm is the Euclidean norm. In the second case objects $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected by an edge if $\mathbf{x}_i$ is among the k-nearest neighbors of $\mathbf{x}_j$ or $\mathbf{x}_j$ is among the k-nearest neighbors of $\mathbf{x}_i$. The edges of the graph are weighted with their Euclidean lengths, so the graph distance is obtained as the shortest path for each pair of points: given a set of objects or representatives ($\mathbf{X}$), the graph distance between two objects $\mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}$ is

the length of the shortest paths of $\mathbf{X}$ joining $\mathbf{x}_i$ and $\mathbf{x}_j$. Figure 1 illustrates a 2-dimensional manifold embedded in a 3-dimensional vector space. It can be seen that in this case the the similarity measure based on the Euclidean distance not properly characterizes the distance of the selected two points.

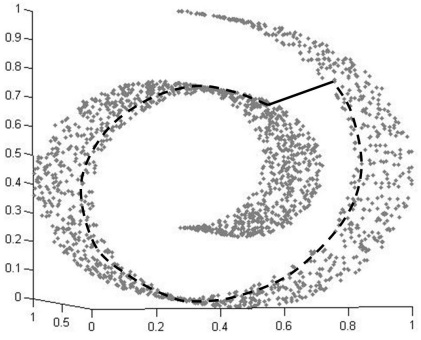


Fig. 1. The Euclidean (solid line) and the graph distance (broken line) of two objects of a manifold

The Isomap algorithm proposed by Tenenbaum et al. in 2000 [29] is based on such a distance measure. Isomap deals with finite data sets of points in $\mathbb{R}^D$ which are assumed to lie on a smooth submanifold M^d ($d \ll D$). The aim of this method is to preserve the intrinsic geometry of the data set and visualize the data in a low-dimensional feature space. For this purpose Isomap calculates the geodesic distances between all data points and then projects them into a low-dimensional vector space using the method of MDS. Isomap consists of three major steps. (1) Construct the neighborhood graph of the data by using the k-neighboring or ϵ-neighboring approach. (2) Compute the geodesic distances between every pair of objects. (3) Construct a d-dimensional embedding of the data points. For the low-dimensional (generally $d = 2$) visualization Isomap utilizes MDS. In this case the multidimensional scaling is not based on the Euclidean distance, but it utilizes the previously computed geodesic distances. Isomap uses a non-Euclidean metric for mapping, therefore a nonlinear projection is obtained as a result. However, if the first step of the Isomap algorithm is applied to a multi-class data set, several disconnected subgraphs can be formed, thus the MDS can not be performed on the whole data set. Wu and Chan [32] give an extension of the Isomap solving this problem. However, real-word data are often noisy. Applying Isomap to noisy data shows also some limitations.

1.3 Application of Vector Quantization in Visualization Techniques

The previously presented metric algorithms are sensitive to noise and outliers and are not applicable to large data sets due to the numerical complexity of the algorithms and the need for the calculation of pairwise (in some cases geodesic)

distances among each data. To handle these difficulties it is suggested to execute a vector quantization (VQ) step to define representatives for the data and thereby to reduce the computational time and handle noise and outliers.

The Self-Organizing Map (SOM) [13] is one of the most popular neural network models. The main goal of SOM is to transform a high-dimensional pattern into a low-dimensional discrete map in a topologically ordered grid. The 2-dimensional grids may be arranged in a rectangular or a hexagonal structure. Actually, the SOM is a neural network, where each neuron is represented by a D-dimensional weight vector. Creating and utilizing the SOM means two basic methods: (1) training process and (2) mapping process. During the training process the weight vectors have to be initialized first (e.g. randomly or sample initialization). After that given a random sample from the training data set, the best matching unit (BMU) in the SOM is located. BMU is the closest neuron to the selected input pattern based on the Euclidean distance. The weights of BMU and neurons closest to it in the SOM grid are then updated towards the sample vector in the input space. The coverage of the change decreases with time. BMU and its neighboring neurons in the SOM grid are updated towards the sample object based on the following formula:

$$\mathbf{w}_i\left(t+1\right)=\mathbf{w}_i\left(t\right)+h_{c,i}\left(t\right)\left[\mathbf{x}\left(t\right)-\mathbf{w}_i\left(t\right)\right], \tag{2}$$

where t denotes time, the $\mathbf{w}_i$ denotes the neurons in the grid, the $\mathbf{x}(t)$ is the the random sample objects at time t and $h_{c,i}\left(t\right)$ yields the neighborhood function about the winner unit c at time t.

There is a variety of different kinds of visualization techniques available for the SOM. (e.g. U-matrix [31], component planes). Component planes are visualized by taking from each weight vector the value of the component (attribute) and depicting this as a color or as a height on the grid. SOM can handle the missing values, but when the map structure does not match the input data structure, SOM is not able to preserve the topology of the input data structure.

There are many algorithms that combines the different visualization techniques with the topology representation possibilities. The method Isotop [16] can be seen as a variation of SOM with a data-driven topology grid. Firstly, Isotop performs a vector quantization step in order to reduce the number of the data. Secondly, Isotop builds a graph structure based on the k-neighboring or ϵ-neighboring. Finally, Isotop performs a non-metric dimensionality reduction. This process uses the graph distances defined by the previously calculated neighborhood connections.

Curviliear Component Analysis (CCA) [5] algorithm was proposed as an improvement of the Kohonen Self-Organizing Maps, in that the output lattice has no fixed structure predetermined. CCA algorithm performs two tasks separately: (1) a vector quantization and (2) a nonlinear projection of these quantized vectors (centroids) onto a d-dimensional output space. After mapping of the centroids the full data set is mapped on the nonlinear discrete subspace defined by

the mapped centroids. CCA maps the data by minimizing the topology error function defined as:

$$E_{CCA} = \frac{1}{2} \sum_{i<j}^{N} \left(d^*_{i,j} - d_{i,j} \right)^2 F\left(d_{i,j} \right), \tag{3}$$

where $d^*_{i,j}$ denotes the distance between the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, and $d_{i,j}$ respectively for $\mathbf{y}_i$ and $\mathbf{y}_j$. F is a decreasing function of $d_{i,j}$ ($F : \mathbb{R}^+ \rightarrow [0,1]$). The function F allows the local topology to be favored with respect to the global topology.

Curvilinear Distances Analysis (CDA) [15] is the nonlinear variant of CCA. While CCA is based on the Euclidean distance, CDA utilizes the curvilinear (graph) distance.

Although most of the algorithms utilize the previously presented two approaches (ϵ-neighboring and k-neighboring) for the construction of the representative graph of the data set, there are other possibilities to disclose the topology of the data, too. Topology representing networks refers to a group of methods that generate a compact topology preserving maps for different data sets. The topology representative methods combine the neural gas (NG) [19] vector quantization method and the competitive Hebbian learning rule [9]. For a given data distribution firstly a cloud will be created by running the neural gas algorithm and then the topology is generated by the competitive Hebbian learning rule. The methods generate their topology map as a result.

There are many methods published in the literature proposing to capture the topology of the given data set. Martinetz and Shulten [20] showed how the simple competitive Hebbian rule forms topology representing networks. The algorithm of Topology Representing Network (TRN) firstly selects some random points (weights) in the input space. The number of weights is a predefined parameter. The algorithm then iteratively selects an object from the input data set randomly and moves all weights closer to this pattern. After this step, the two weights closest to the randomly selected input pattern will be connected. Finally, edges exceeding a predefined age are removed. This iteration process is continued until a termination criterion is satisfied. The algorithm results in the Topology Representing Network.

Dynamic Topology Representing Networks (DTRN) were introduced by Si at al. [28]. In this method the topology graph incrementally grows by adding and removing edges and vertices. The algorithm starts with only one node. The algorithm examines a vigilance test in each iteration. If the nearest (winner) node to the randomly selected input pattern fails this test, a new node is created and this new node is connected to the winner. If the winner passes the vigilance test, the winner and its adjacent neighbors are moved closer to the selected input pattern. In this second case, if the winner and the second closest weights are not connected, the algorithm creates an edge between them. Similarly to the TRN algorithm DTRN also removes those connections whose age achieves a predefined threshold. The most important input parameter of DTRN algorithm

is the vigilance threshold. This vigilance threshold gradually decreases from an initial value to a final value.

Weighted Incremental Neural Network (WINN) [22] produces a weighted connected net. This net consists of weighted nodes connected by weighted edges. This algorithm starts with two randomly selected nodes from the input pattern. In each iteration the algorithm selects a pattern from the input data set, and the nearest node to this object and its direct topological neighbors move towards the selected object. If the nearest node and the other $n-1$ nearest nodes are not connected, the algorithm establishes a connection between them. The ages and the weight-variables of edges, the error-variables and the weights of nodes are updated step by step. This method inserts a new node in the net when the number of the generated input pattern is a multiple of a predefined λ parameter. Similarly to the previous algorithms WINN also removes the 'old' connections.

In the literature there are a few dimensionality reduction methods that utilizes the combination of the neural gas and the competitive Hebbian learning rule. The OVI-NG algorithm [7] is an enhancement over CCA. It utilizes the Euclidean distance and performs a non-metric dimensionality reduction method that preserves the neighborhood relationships between the representative elements defined by the NG algorithm and the input data elements. The GNLP-NG algorithm [8] is an extension of the OVI-NG as it uses graph distances instead of the Euclidean ones.

1.4 Motivation

All these methods seem a good choice for dimensionality reduction, but each of them has some disadvantages. When the data set contains nonlinear structure, PCA, ICA and classical MDS (based on Euclidean distance) are not applicable. Isomap cannot model multi-class problems and it is not efficient on large and noisy data sets. The main disadvantage of Isotop, OVI-NG and GNLP-NG methods are that they use a non-metric mapping method and thereby only the rank ordering of the representatives is preserved during the mapping process. The SOM has a predefined and limited structure of the neurons. Although CCA and CDA are a more complicated techniques, they need to be well parameterized [17].

In this paper a new robust method for data exploration is proposed. This method aims to fulfill the following three criterions: (i) give a low-dimensional representation of the data, (ii) preservation of the metric of the input data space, and (iii) preservation of the intrinsic data structure (topology). The proposed method as vector quantization and topology exploration utilizes the Topology Representing Network algorithm, and it also exploits the main advantages of dimensionality reduction methods. As a result it gives compact low-dimensional topology preserving feature maps to explore the hidden structure of data.

2 Topology Representation of High-Dimensional Data

The previously presented systematic overview showed that visualization algorithms can be defined as combinations of vector quantization, distance calculation,

and mapping algorithms. As vector quantization the well-known k-means and neural gas algorithms can be used. The distances can be calculated based on either Euclidean norm or graph distance. The graph distance can be calculated based on the graphs arising from the ϵ-neighboring, k-neighboring or the Topology Representing Network. In the following this algorithm is presented.

The combination of the neural gas algorithm and the competitive Hebbian rule gives a powerful tool for the topology representation. In contrast to the k-neighboring and to the ϵ-neighboring it also considers the distribution of the data elements. Although, the TRN, DTRN and WINN algorithms are quite similar, the TRN algorithm gives the most robust representation of the data.

Given a set of the data ($\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, $\mathbf{x}_i \in \mathbb{R}^D$, $i = 1, \ldots, N$) and a set of neural units ($\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n\}$, $\mathbf{w}_i \in \mathbb{R}^D$, $i = 1, \ldots, n$). The TRN algorithm distributes the pointers $\mathbf{w}_i$ between the data objects by the the neural gas algorithm, and forms connections between them by applying competitive Hebbian rule. The run of the algorithm results in a topology map of the data. The topology map means a graph $G = (\mathbf{W}, \mathbf{C})$, where $\mathbf{W}$ denotes the nodes (neural units, centres, representatives) quantified by the neural gas algorithm and $\mathbf{C}$ yields the set of edges between them.

The TRN Algorithm

1. Initialize the units $\mathbf{w}_i$ randomly. Set all connection strengths $c_{i,j}$ to zero. Set $t = 0$.
2. Select an input pattern $\mathbf{x}$ with equal probability for each $\mathbf{x}$. Increase the iteration step $t = t + 1$.
3. For each unit $\mathbf{w}_i$ determine the number k_i of units $\mathbf{w}_j$ with

$$\|\mathbf{x} - \mathbf{w}_j\| < \|\mathbf{x} - \mathbf{w}_i\| \tag{4}$$

   by determining the sequence $(i_0, i_1, \ldots, i_{n-1})$ with

$$\|\mathbf{x} - \mathbf{w}_{i_0}\| < \|\mathbf{x} - \mathbf{w}_{i_1}\| < \ldots < \|\mathbf{x} - \mathbf{w}_{i_{n-1}}\|. \tag{5}$$

4. Update the units $\mathbf{w}_i$ according to the neural gas algorithm by setting

$$\mathbf{w}_i^{new} = \mathbf{w}_i^{old} + \varepsilon \cdot e^{-k_i/\lambda} \left(\mathbf{x} - \mathbf{w}_i^{old}\right), \quad i = 1, \ldots, n \tag{6}$$

5. If a connection between the first and the second closest unit to $\mathbf{x}$ does not exist already ($c_{i_0,i_1} = 0$), create a connection between them by setting $c_{i_0,i_1} = 1$ and set the age of this connection to zero by $t_{i_0,i_1} = 0$. If this connection already exists ($c_{i_0,i_1} = 1$), set $t_{i_0,i_1} = 0$, that is, refresh connection of the units i_0 - i_1.
6. Increment the age of all connections of $\mathbf{w}_{i_0}$ by setting $t_{i_0,j} = t_{i_0,j} + 1$ for all $\mathbf{w}_j$ with $c_{i_0,j} = 1$.
7. Remove those connections of unit $\mathbf{w}_{i_0}$ the age of which exceeds the parameter T by setting $c_{i_0,j} = 0$ for all $\mathbf{w}_j$ with $c_{i_0,j} = 1$ and $t_{i_0,j} > T$.
8. If $t < t_{max}$ go back to step 2.

The TRN algorithm combines the neural gas algorithm with the competitive Hebbian rule. Steps 1-5 (without setting the connection strengths $c_{i,j}$ to zero) and step 8 describe the neural gas algorithm. How the connections of the objects are formed is based on the Hebbian rule.

The algorithm has many parameters. The number of the iterations (t_{max}) is determined by the user. The parameter λ, the step size ε and the lifetime T is dependent on the number of the iteration. This time dependence can be expressed by the following general form:

$$g(t) = g_i \left(\frac{g_f}{g_i} \right)^{t/t_{max}} \tag{7}$$

The paper [20] gives good suggestions to tune these parameters.

3 Visualization of the TRN

This section proposes an algorithm for the visualization of Topology Representing Networks. The section includes the presentation of the applied dimensionality reduction method and the proposed algorithm.

The main goal of the proposed algorithm is to preserve both the topology and the metric of the input data set. For this purpose it combines the main benefits of the topology representing networks with a metric dimensionality reduction method. Although, the dimensionality reduction methods offer several low-dimensional visualization techniques for the presentation, we suggest to use a metric dimensionality reduction method to preserve the the metric of the data set. To achieve this purpose can be used for example the metric MDS or the Sammon mapping or its variations. In this paper the metric MDS is preferred. However, distance-preservation methods fully works only for linear manifolds. To avoid this disadvantage we suggest to compute the dissimilarity of the data based on a graph distances instead of computing a distance as the Euclidean distance. To compute the graph distances the set of the data is replaced by the graph resulted of the TRN algorithm. The edges of the graph are labeled with their Euclidean length and Dijkstra's algorithm [6] is run on the graph, in order to compute the shortest path for each pair of points.

The classical MDS algorithm is an algebraic method that rests on the fact that the matrix $\mathbf{Y}$ containing the output coordinates can be derived by eigenvalue decomposition from the scalar product matrix $\mathbf{B} = \mathbf{Y}\mathbf{Y}^T$. The detailed metric MDS algorithm is the following:

The MDS Algorithm [24]

1. Let the searched coordinates of n points in a d-dimensional Euclidean space be given by $\mathbf{y}_i \quad (i = 1, \ldots, n)$, where $\mathbf{y}_i = (y_{i,1}, \ldots, y_{i,d})^T$. Matrix $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_n]^T$ is the $n \times d$ coordinates matrix. The Euclidean distances $\{d^*_{i,j} = (\mathbf{y}_i - \mathbf{y}_j)^T (\mathbf{y}_i - \mathbf{y}_j)\}$ are known. The inner product of matrix $\mathbf{Y}$ is denoted

$\mathbf{B} = \mathbf{Y}\mathbf{Y}^T$. Find matrix $\mathbf{B}$ from the known distances $\{d^*_{i,j}\}$ using Young-Householder process [33]:

(a) Define matrix $\mathbf{A} = [a_{i,j}]$, where $a_{i,j} = -\frac{1}{2}d^{*2}_{i,j}$,

(b) Deduce matrix $\mathbf{B}$ from $\mathbf{B} = \mathbf{HAH}$, where $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{l}\mathbf{l}^T$ is the centering matrix, and $\mathbf{l}$ is an $n \times 1$ column vector of n ones

2. Recover the coordinates matrix $\mathbf{Y}$ from $\mathbf{B}$ using the spectral decomposition of $\mathbf{B}$:

(a) The inner product matrix $\mathbf{B}$ is expressed as $\mathbf{B} = \mathbf{Y}\mathbf{Y}^T$. The rank of $\mathbf{B}$ is $r(\mathbf{B}) = r\left(\mathbf{Y}\mathbf{Y}^T\right) = r(\mathbf{Y}) = d$. $\mathbf{B}$ is symmetric, positive semi-definite and of rank d, and hence has d non-negative eigenvalues and $n - d$ zero eigenvalues.

(b) Matrix $\mathbf{B}$ is now written in terms of its spectral decomposition, $\mathbf{B} = \mathbf{V}\Lambda\mathbf{V}^T$, where $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$ the diagonal matrix of eigenvalues λ_i of $\mathbf{B}$, and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ the matrix of corresponding eigenvectors, normalized such that $\mathbf{v}_i^T\mathbf{v}_i = \mathbf{l}$,

(c) Because of the $n - d$ zero eigenvalues, $\mathbf{B}$ can now be rewritten as $\mathbf{B} = \mathbf{V}_1\Lambda_1\mathbf{V}_1^T$, where $\Lambda_1 = diag(\lambda_1, \lambda_2, \ldots, \lambda_d)$ and $\mathbf{V}_1 = [\mathbf{v}_1, \ldots, \mathbf{v}_d]$,

(d) Finally the coordinates matrix is given by $\mathbf{Y} = \mathbf{V}_1\Lambda_1^{\frac{1}{2}}$, where $\Lambda_1^{\frac{1}{2}} = diag\left(\lambda_1^{\frac{1}{2}}, \ldots, \lambda_d^{\frac{1}{2}}\right)$.

The proposed data analysis tool is an unsupervised mapping algorithm that gives a compact representation of the data set as result. It results in a new visualization map, called Topology Representing Network Map (TRNMap). TRNMap is a self-organizing model with no predefined structure which provides an expressive presentation of high-dimensional data in low-dimensional data space. The dimensionality of input space is not restricted. Although we can get arbitrary dimensional output map as a result, in this paper the 2-dimensional output map is recommended. The method for constructing the TRNMap (TRNMap algorithm) is based on the graph distances, therefore it is able to handle the set of data lying on a low-dimensional manifold that is nonlinearly embedded in a higher-dimensional input space.

Given a set of data $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, $\mathbf{x}_i \in \mathbb{R}^D$. The main goal of the algorithm is to give a compact, perspicuous representation of the objects. For this purpose the set of $\mathbf{X}$ is mapped into a lower-dimensionality output space, $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\}$,$n \leq N$, ($\mathbf{y}_i \in \mathbb{R}^d$, $d \ll D$, in this paper $d = 2$).

To avoid the influence of the range of the attributes a normalization procedure is suggested as a preparing step (Step 0). After the normalization the algorithm creates the topology representing network of the input data set (Step 1). Although, any kind of topology representing networks is adaptable in the proposed algorithm, in this paper we suggest to use of the Topology Representing Network

proposed by Martinetz and Shulten [20]. The number of the nodes (representatives) of the TRN is determined by the user. By the use of the TRN, this step ensures the exploration of the correct structure of the data set, and includes a vector quantization, as well. If the resulting graph is unconnected, the algorithm connects the subgraphs by linking the closest elements (Step 2). Then the pairwise graph distances are calculated between every pair of representatives (Step 3). In the following, the original topology representing network is mapped into a 2-dimensional graph (Step 4). In this paper it is carried out by the metric multidimensional scaling, where the similarities of the data points are provided by the previously calculated graph distances. The use of graph distances during the mapping process ensures that the algorithm is able to reveal low-dimensional manifolds embedded in a high-dimensional vector space. For the expressive visualization component planes are also created by the D-dimensional representatives (Step 5).

The Topology Representing Network Map Algorithm

0. Normalize the input data set $\mathbf{X}$.
1. Create the topology representing network of $\mathbf{X}$ by the use of the TRN algorithm. Yield $M^{(D)} = (W, C)$ the resulting graph, let $\mathbf{w}_i \in W$ the representatives (nodes) of $M^{(D)}$. If exists an edge between the representatives $\mathbf{w}_i$ and $\mathbf{w}_j$ ($\mathbf{w}_i, \mathbf{w}_j \in W$, $i \neq j$), $c_{i,j} = 1$, otherwise $c_{i,j} = 0$.
2. If $M^{(D)}$ is not connected, connect the subgraphs in the following way:

   While there are unconnected subgraphs ($m_i^{(D)} \subset M^{(D)}$, $i = 1, 2, \ldots$):
   (a) Choose a subgraph $m_i^{(D)}$.
   (b) Let the terminal node $\mathbf{t}_1 \in m_i^{(D)}$ and its closest neighbor $\mathbf{t}_2 \notin m_i^{(D)}$ from:
   $\|\mathbf{t}_1 - \mathbf{t}_2\| = min\|\mathbf{w}_j - \mathbf{w}_k\|, \quad \mathbf{t}_1, \mathbf{w}_j \in m_i^{(D)}, \quad \mathbf{t}_2, \mathbf{w}_k \notin m_i^{(D)}$
   (c) Set $c_{\mathbf{t}_1,\mathbf{t}_2}$=1.

   Yield $M^{*(D)}$ the modified $M^{(D)}$.

3. Calculate the graph distances between all $\mathbf{w}_i, \mathbf{w}_j \in M^{*(D)}$.
4. Map the graph $M^{(D)}$ into a 2-dimensional vector space by metric MDS based on the graph distances of $M^{*(D)}$.
5. Create component planes for the resulting Topology Representing Network Map based on the values of $\mathbf{w}_i \in M^{(D)}$.

The parameters of the TRNMap algorithm are the same as those of the Topology Representing Networks algorithm. The number of the nodes of the output graph (n) is determined by the user. The bigger the n the more detailed the output map will be. We suggest the choice $n = 0.2N$. If the number of the input data elements is high, it can result in numerous nodes. In these cases it is practical to decrease the number of the representatives and iteratively run the algorithm to capture the structure more perspicuously. Values of the other parameters of TRN (λ, the step size ε, and the threshold value of edge's ages T) can be the same as proposed by Martinetz & Schulten [20].

Besides the visualization of the data structure, the nodes of TRNMap also visualizes the high-dimensional information by the use use of the component plane representation. A component plane displays the value of one component of each node. If the input data set has D attributes, the Topology Representing Network Map component plane includes D different maps according to the D components. The structure of these maps is the same as the TRNMap map, but the nodes are represented in greyscale. White color means the smallest value, black color corresponds to the greatest value of the attribute. By viewing several component maps at the same time it is also easy to see simple correlations between attributes. Because nodes of TRNMap can be seen as possible cluster prototypes, TRNMap can provide the basis for an effective clustering method.

4 Application Examples

Three examples are now used to demonstrate the applicability of the proposed algorithm. The first example is based on the Swiss roll data set, widely used in the related literature (e.g. [17,29]). The other two examples are based on real clustering problems coming from the UCI Repository of Machine Learning Databases (http://www.ics.uci.edu). The second example deals with the visualization and clustering of the well-known wine data set. The third example shows the result of the mapping process on the Wisconsin breast cancer database.

To compare TRNMap with its alternatives several combinations of vector quantization, distance calculation, and mapping algorithms have been studied. As vector quantization the well-known k-means and neural gas algorithms were used. The distances were calculated based on either Euclidean norm (Eu) or graph distance. Although the graph distance can be calculated based on the graphs arising from the ϵ-neighboring, k-neighboring or the Topology Representing Network, only the two last methods (knn with k=3 and TRN) have been applied, since the ϵ-neighboring method is very sensitive to the data density. As dimensionality reduction method the MDS and the Sammon mapping have been applied. The Sammon mapping was applied without initialization (Sammon) and with initialization based on the MDS (Sammon_mds), where the result of the MDS algorithm serves the initial projection of the data.

Three quantitative measures are used to evaluate the different mapping methods on comparable grounds. The firs measure evaluates the Sammon stress function (11). Since mappings often utilize MDS as dimensionality reduction method, the second quantitative measure evaluates the MDS stress defined as follows:

$$\frac{1}{\sum\limits_{i<j}^{N} d_{i,j}^{*2}} \sum_{i<j}^{N} \left(d_{i,j}^{*} - d_{i,j}\right)^2, \tag{8}$$

where $d_{i,j}^{*}$ denotes the distances between the data points or representatives to be projected, and $d_{i,j}$ respectively for the mapped objects. The third measure is based on the residual variance defined as:

$$1 - R^2(\mathbf{D}_X^{*}, \mathbf{D}_Y), \tag{9}$$

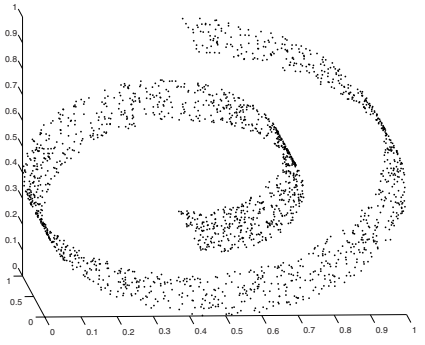
where $\mathbf{D}_Y$ denotes the matrix of Euclidean distances in the low-dimensional output space ($\mathbf{D}_Y = [d_{i,j}]$), and $\mathbf{D}_X^*$, $\mathbf{D}_X^* = [d_{i,j}^*]$ is the best estimation of the distances of the data to be projected. $\mathbf{R}$ is the standard linear correlation coefficient, taken over all entries of $\mathbf{D}_Y$ and $\mathbf{D}_X^*$. When the examined methods utilize a geodesic or a graph distances to calculate the pairwise dissimilarities of the objects in the high-dimensional space the value of the dissimilarities of these objects ($d_{i,j}^*$) are also calculated based on this principle. The values of $d_{i,j}$ denote the Euclidean distances of the objects in the low-dimensional output space in all cases.

4.1 The Swiss Roll Data Set

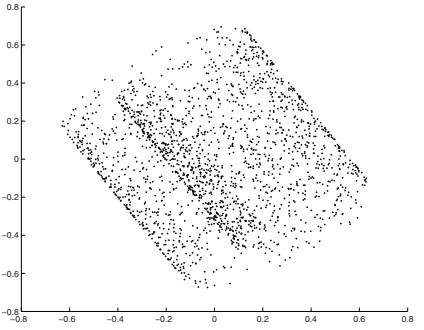
The Swiss roll data set (Fig. 2(a)) is a 3-dimensional data set with a 2-dimensional nonlinear embedded manifold. In this example the Swiss roll data set contains 2000 data points. Linear mapping algorithms, such Principal Component Analysis do not come to the proper results (Fig. 2(b)), because of the 2-dimensional nonlinear embedding. As can be seen in Fig. 2(c) the nonlinear CCA is also not able to uncover the real structure of the data. Figure 2(d) shows that Isomap finds the structure of the 2-dimensional embedded manifold. Figures 2(e) and (f) show the result of the CDA algorithms (the number of the representatives is 200 and the neighboring graph was calculated based on the k-neighboring approach with k=3). The results of the CDA algorithm were calculated with two different vector quantization methods. Figure 2(e) shows the result of the CDA based on the k-means VQ, and Fig. 2(f) shows the result obtained by the use of the neural gas vector quantization. It can be seen that the CDA algorithm can not uncover the structure in all cases.

In the following let us have a closer look at the result of the proposed Topology Representing Network Map algorithm. For more compact representation the number of the output nodes in this case was chosen to be $n = 200$. The parameters of the TRN algorithm were tuned according to the rules presented in [20]: $\lambda(t) = \lambda_i(\lambda_f/\lambda_i)^{t/t_{max}}$ and $\varepsilon(t) = \varepsilon_i(\varepsilon_f/\varepsilon_i)^{t/t_{max}}$, where $\lambda_i = 0.2n$, $\lambda_f = 0.01$, $\varepsilon_i = 0.3$, $\varepsilon_f = 0.05$ and $t_{max} = 200n$. Unlike to the suggested formula ($T(t) = T_i(T_f/T_i)^{t/t_{max}}$), the threshold of the maximum age of the edges was always kept on $T_i = T_f = 0.1n$ during the whole running time.

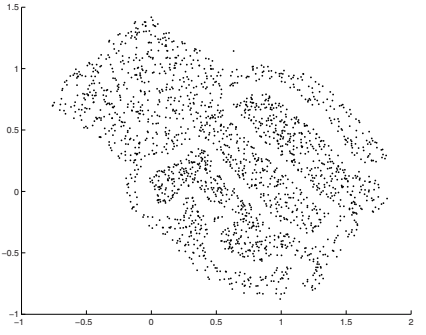
The Topology Representing Network of the Swiss roll data is shown in Fig. 3(a) (Step 2 - Step 4). Figure 3(b) shows the 2-dimensional TRNMap of the Swiss roll data set. It can be seen that the Topology Representing Network Map algorithm is able to uncover the embedded 2-dimensional manifold. Both TRNMap and Isomap algorithms are able to disclose the structure of the manifold, but the TRNMap algorithm gives more compact representation. Moreover, when the parameter k (or ϵ) is not properly selected, Isomap fails to uncover the real structure of the data. If these parameters are too small the resulting graphs are unconnected, so multidimensional scaling is not executable on the whole data set. Contrary, when these parameters are too big, Isomap cannot uncover the real data structure.



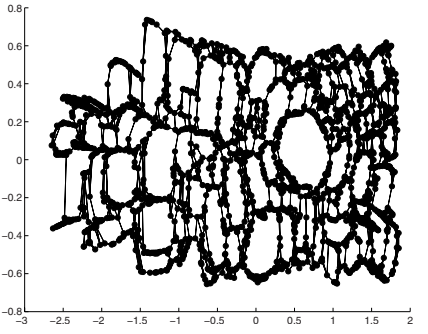
(a) The 3-dimensional Swiss roll data set



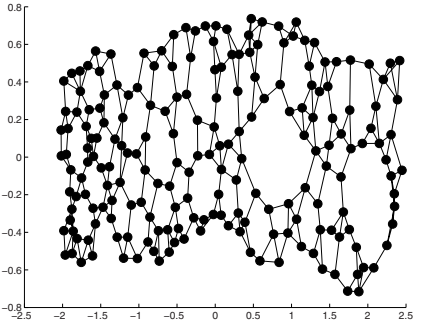
(b) 2-dimensional PCA projection of the Swiss roll data set



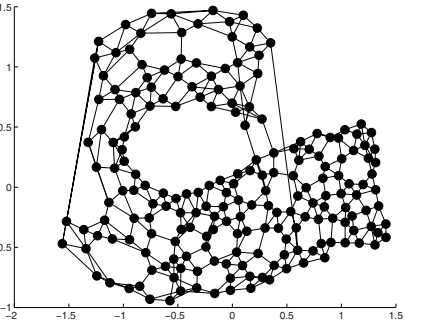
(c) 2-dimensional CCA projection of the Swiss roll data set



(d) 2-dimensional Isomap projection of the Swiss roll data set

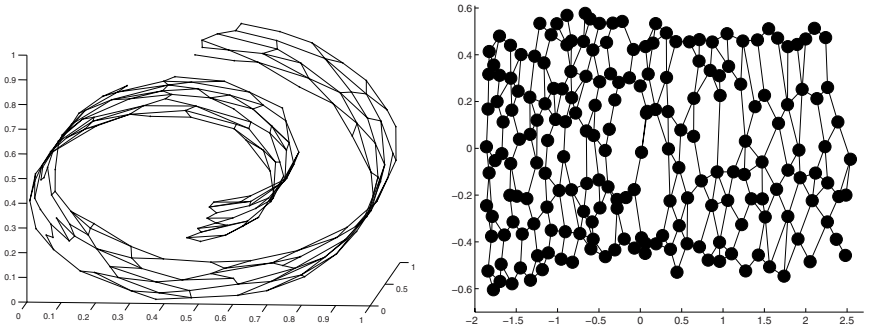


(e) 2-dimensional CDA projection of the Swiss roll data set with k-means VQ



(f) 2-dimensional CDA projection of the Swiss roll data set with NG VQ

Fig. 2. The Swiss roll data set and its PCA, CCA, CDA and Isomap projection



(a) The TRN of the Swiss roll data set *(b) The TRNMap of the Swiss roll data set*

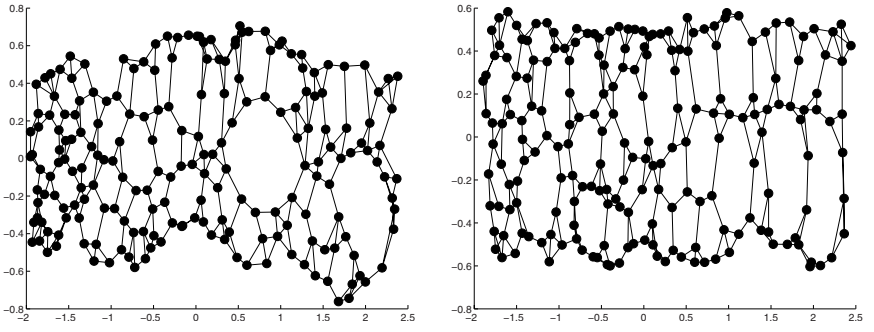
Fig. 3. The TRN and the TRNMap of the Swiss roll data set

Table 1. The values of the Sammon stress, MDS stress and residual variance of different algorithms on the Swiss roll data set

Algorithm	Sammon stress	MDS stress	residual variance
kmeans-Eu-MDS	0,05088	0,20743	0,22891
kmeans-Eu-Sammon	0,050837	0,2132	0,242
kmeans-Eu-Sammon_mds	0,049968	0,20931	0,23268
kmeans-knn-MDS	0,0021229	0,00090904	0,0032618
kmeans-knn-Sammon	0,0077129	0,0044013	0,015745
kmeans-knn-Sammon_mds	0,0019803	0,00097396	0,003475
NG-Eu-MDS	0,058255	0,049412	0,26781
NG-Eu-Sammon	0,057581	0,05104	0,27613
NG-Eu-Sammon_mds	0,057163	0,050242	0,27169
NG-knn-MDS	0,0020823	0,00085883	0,003066
NG-knn-Sammon	0,0039808	0,0024227	0,009162
NG-knn-Sammon_mds	0,0039215	0,0023683	0,0089164
TRNMap1 (TRN+MDS)	0,0014528	0,00062888	0,0022373
TRNMap2 (TRN+Sammon)	0,010487	0,0049348	0,015857
TRNMap3 (TRN+Sammon_mds)	0,0013435	0,00067616	0,0023464

For the comparison the TRNMap was also calculated with all three mapping possibilities. The errors of all projections can be seen in Table 1.

Table 1 shows that the mappings based on the Euclidean distance are not able to uncover the structure of the data because of the nonlinear embedded manifold. On the other hand, it can be seen that the initialization of the Sammon mapping with the result of the MDS improves the mapping quality of the mapping. When the distances are calculated based on a graph, the MDS results in better mapping quality than the Sammon mappings. The best mapping results are given by



(a) The kmeans-knn-MDS mapping of the Swiss roll data set

(b) The NG-knn-MDS mapping of the Swiss roll data set

Fig. 4. Different mappings of the Swiss roll data set

the kmeans-knn-MDS, NG-knn-MDS and TRNMap1 algorithms (see Fig. 4 and Fig. 3(b)). Comparing all methods it can be seen that the TRNMap1 outperform all other methods.

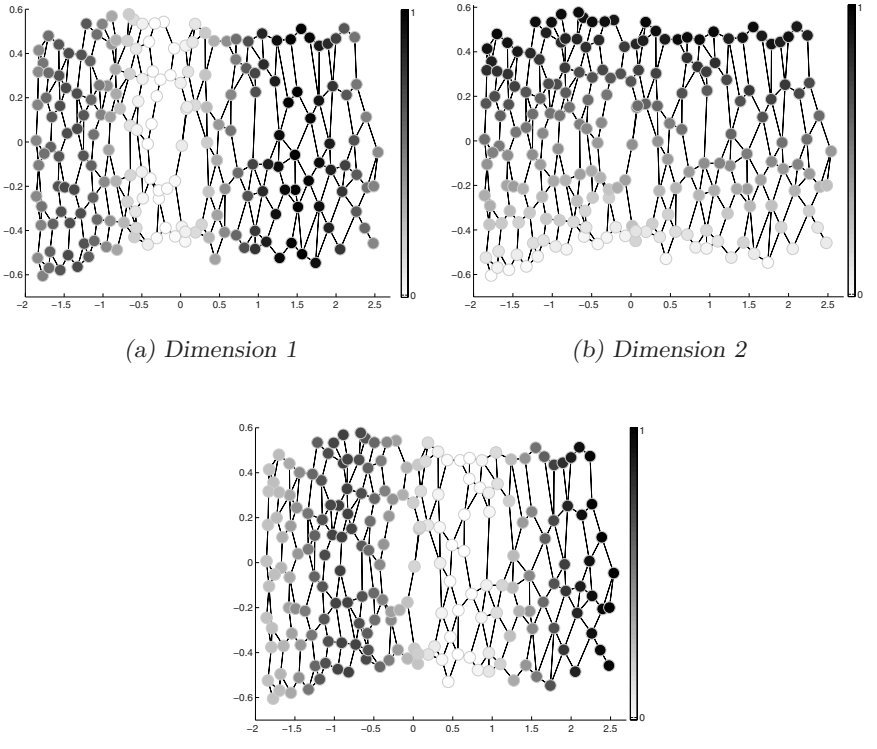
From the methods presented in the Sect. 1.3 the method CDA is the most closely related algorithm to the TRNMap based on the following three reasons: (i) both methods utilizes vector quantization, (ii) both methods utilize graph distances to compute the dissimilarities of the representatives and (iii) both methods use metric dimensionality reduction methods. To compare the results of these methods see Fig. 2(e), Fig. 2(f), Fig. 3(b) and Table 2. Based on the previously made observation the CDA algorithm was initialized with the result of the MDS algorithm. Both the graphical representations and the error measures point out that TRNMap outperforms the CDA algorithm.

Table 2. The values of the Sammon stress, MDS stress and residual variance of the CDA and the TRNMap algorithms on the Swiss roll data set

Algorithm	Sammon stress	MDS stress	residual variance
kmeans-CDA	0,0028834	0,0020072	0,0049501
NG-CDA	0,079221	0,05667	0,20326
TRNMap1 (TRN+MDS)	0,0014528	0,00062888	0,0022373

The visualization of the Topology Representing Network Map also includes the construction of the component planes (Step 5). The Topology Representing Network Map ordered component planes are shown in Fig. 5. The largest value of the attributes of the representatives corresponds to the black and the smallest value to the white dots surrounded by a grey circle. Figure 5(a) shows that alongside the manifold the value of the first attribute (first dimension) initially grows a little, then it decreases to the smallest value, after that it grows to the

highest value, and finally it decreases a little. The value of the second attribute is invariable alongside the manifold, but across the manifold it changes uniformly. The third component starts from a middle value and grows a little, then it falls to the smallest value, and finally increases to the highest value. The degree of the changes in all three cases is uniform.



(a) Dimension 1 *(b) Dimension 2*

(c) Dimension 3

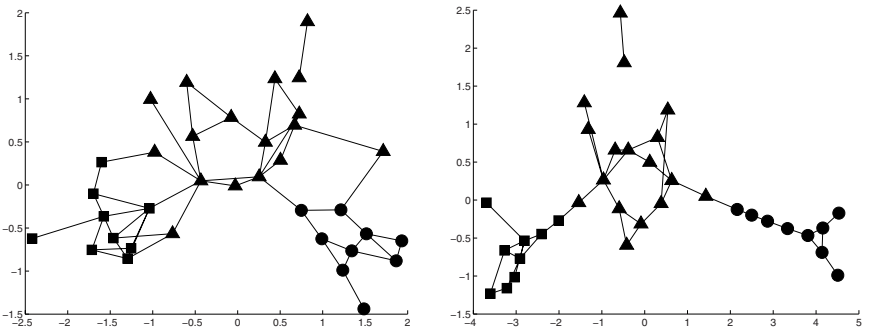
Fig. 5. Component planes of the Topology Representing Network Map of the Swiss roll data set

4.2 Visualization of Real Data Sets

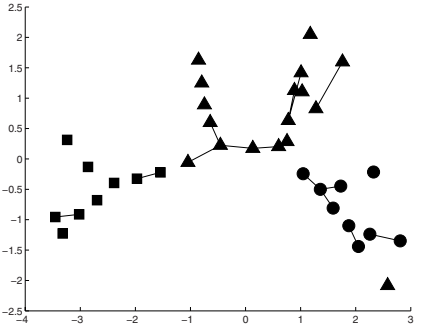
In this subsection two real problems are considered. In all two cases λ, ε and t_{max} parameters were set to the values presented in Section 4.1. The tuning of the parameter age of the edges is shown in the following. On the Topology Representing Network Map and on its component maps the class labels are also presented in all two cases. The representatives are labeled based on the principle of the majority vote: (1) each data point is assigned to the closest representative; (2) the representatives are labeled with the class label occurring in the highest degree between its data elements.

Table 3. The values of the Sammon stress, MDS stress and residual variance of the CDA and the TRNMap algorithms on the wine data set

Algorithm	Sammon stress	MDS stress	residual variance
kmeans-CDA	0,05035	0,029662	0,084369
NG-CDA	0,016299	0,0083896	0,031398
TRNMap (TRN+MDS)	0,0079656	0,0033589	0,011283



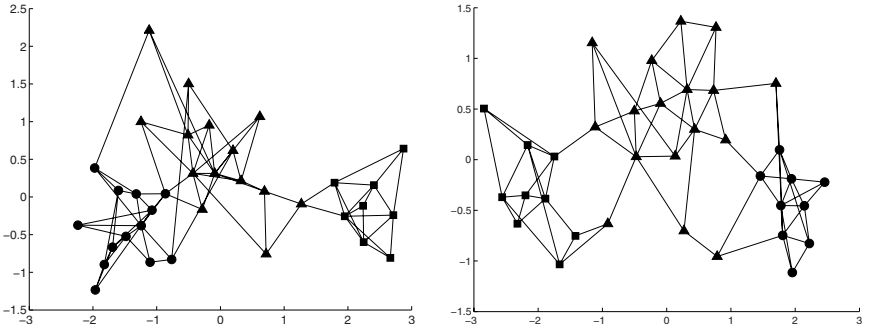
(a) $T_f = 0.3n$ *(b)* $T_f = 0.15n$



(c) $T_f = 0.05n$

Fig. 6. Topology Representing Network Maps of the wine data set with 35 nodes and with different values of T_f

The wine database consists of the chemical analysis of 178 wines from three different cultivars in the same Italian region. Each wine is characterized by 13 continuous attributes, and there are three classes distinguished. The number of the representatives was chosen to be $n = 0.2N$. It means 35 nodes in this case. The data seems relatively diverse because the maps resulted by the multiple runs are more different. The effect of the change of the parameter T_f is shown in Fig. 6



(a) The kmeans-CDA representation of the wine data set with MDS initialization

(b) The NG-CDA representation of the wine data set with MDS initialization

Fig. 7. Different representations of the wine data set based on the NG VQ

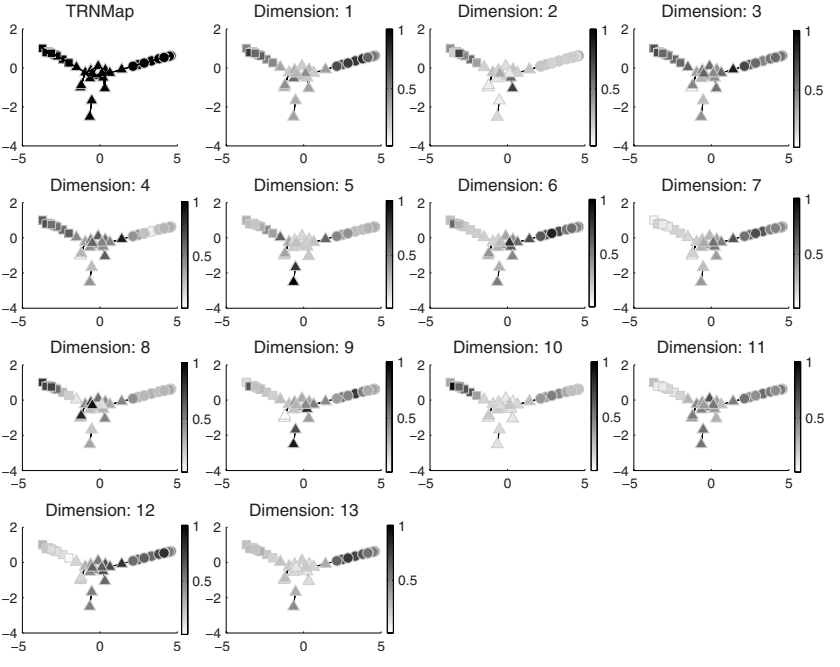
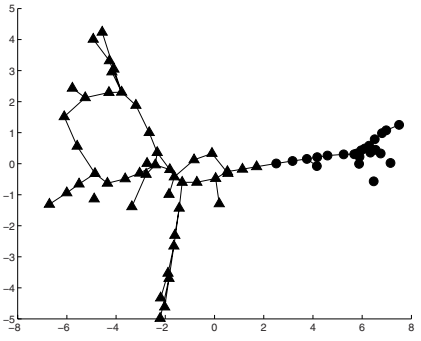


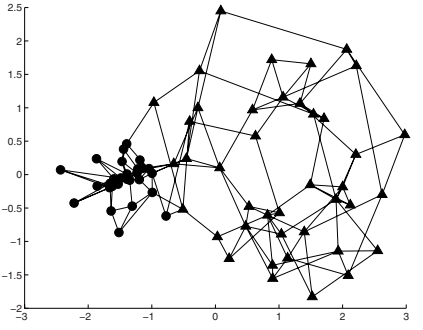
Fig. 8. Complete Topology Representing Network Map representation of the wine data set

(other parameters were chosen the same as in the previous example). It can be seen that the deletion of edges produces smoother graphs.

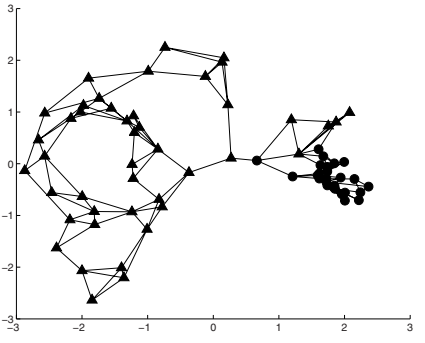
For the comparison the CDA algorithm has been also applied on this data set. The best result of the CDA algorithm applying the k-neighboring approach was



(a) TRNMap of Wisconsin data set



(b) kmeans-knn-MDS projection of Wisconsin data set



(c) NG-knn-MDS projection of Wisconsin data set

Fig. 9. Different visualizations of the Wisconsin breast cancer data set

given with $k = 3$ (Fig. 7(a) and Fig. 7(b)). The class labels in these figures are the same as in Fig. 6. The CDA mapping was initialized with the result of the MDS algorithm, and the error measures for the TRNMap were calculated for the parameter setting $T_f = 0.15n$. Table 3 shows the numerical evaluation of the algorithms. It can be seen that the TRNMap indicate the best mapping quality. The component planes of TRNMap with $T_f = 0.15n$ are shown in Fig. 8.

The **Wisconsin breast cancer data base** is a well known diagnostic data set for breast cancer compiled by Dr William H. Wolberg, University of Wisconsin Hospitals [18]. This data set contains 9 attributes and the class labels for the 683 instances (16 records with missing values were deleted) of which 444 are benign and 239 are malignant. The number of the nodes in this case was reduced to $n = 70$. The results of the several runs seem to have drawn a fairly wide partition and a compact partition. The resulting Topology Representing Network Maps with 70 nodes are shown in Fig. 9(a). For the comparison we have also

Table 4. The values of the Sammon stress, MDS stress and residual variance of the different mapping algorithms on the Wisconsin breast cancer data set

Algorithm	Sammon stress	MDS stress	residual variance
kmeans-knn-MDS	0,027987	0,019241	0,074187
NG-knn-MDS	0,017252	0,010299	0,039967
TRNMap (TRN+MDS)	0,011065	0,0059399	0,021162

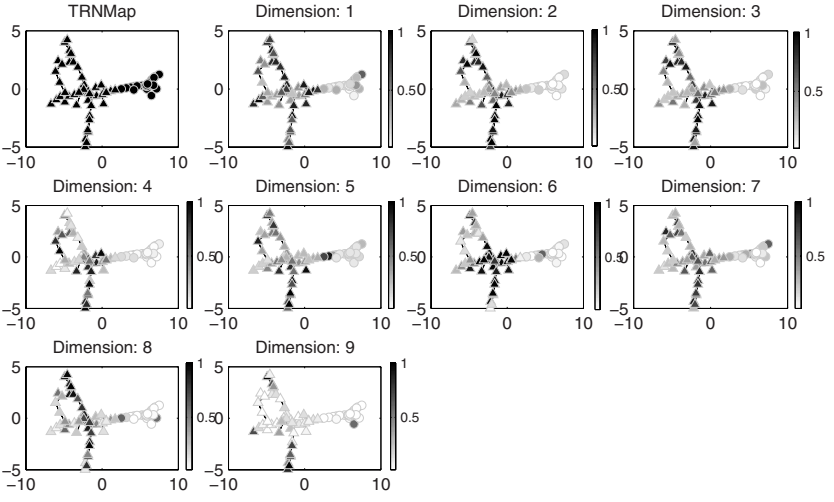


Fig. 10. Complete Topology Representing Network Map representation of the Wisconsin breast cancer data set

run the kmeans-knn-MDS (Fig. 9(b)) and the NG-knn-MDS (Fig. 9(c)) algorithms on this data set. The number of the representatives in these cases was also chosen to $n = 70$ and the neighborhood graph was established based on the k-neighboring approach with parameter setting $k = 3$. In these figures the representatives of the benign class are labeled with circle markers and the malignant class is yielded with triangle markers. Table 4 shows the numerical evaluation of the mappings. The efficiency of the proposed TRNMap algorithm in this case is also confirmed by the error values. The complete TRNMap visualization of the Wisconsin breast cancer data set is shown in Fig. 10.

5 Conclusion

This paper introduced a new tool for the visualization of the hidden structure of high-dimensional data sets. It has been shown (1) if low-dimensional manifolds exist in the high-dimensional feature space of the data set algorithms based on geodesic (graph) distance should be preferred over classical Eucledian distance

based methods. (2) Among the wide range of possible approaches graphs obtained by Topology Representing Networks are the best suitable to approximate this low-dimensional manifold. (3) Multidimensional Scaling (MDS) is an effective method to form a low-dimensional map of the TRN based on the calculated graph distances. (4) Sammon projection is sensitive to the initial map of the graph, so it should be used only to fine tune the mapping obtained by MDS.

The resulted Topology Representing Network Map can be seen as an improvement of Isomap in the following viewpoints: (5) TRNMap is able to map such data sets also that are located far away from each other (multi-class problems); (6) by the applied vector quantization method the computational cost of the mapping is significantly reduced; (7) the resulting representatives provide excellent facilities for further data exploratory analysis, such as clustering or classification; (8) TRNMap is less sensitive noise and outliers in the data. Comparing the TRNMap and the CDA algorithms, it can be seen that TRNMap utilizes more efficient calculation of the graph distances than CDA.

Synthetic and real life examples have showed that Topology Representing Network Map utilizes advantages of several dimensionality reduction methods so that it is able to give a compact representation of low-dimensional manifolds nonlinearly embedded in the high-dimensional feature spaces.

Acknowledgement. The authors acknowledge the financial support of the Cooperative Research Centre (VIKKK, project 2004-I), the Hungarian Research Found (OTKA 49534), the Bolyai Janos fellowship of the Hungarian Academy of Science, and the Öveges fellowship.

References

1. Bauer, H., Villmann, T.: Growing a hypercubical output space in a self-organizing feature map. IEEE Transactions on Neural Networks 8(2), 218–226 (1997)
2. Bernstein, M., de Silva, V., Langford, J.C., Tenenbaum, J.B.: Graph approximations to geodesics on embedded manifolds. Technical report, Stanford University, Stanford (2000)
3. Borg, I., Groenen, P.: Modern Multidimensional Scaling: Theory and Applications. Springer, New York (1997)
4. Comon, P.: Independent component analysis: A new concept? Signal Processing 36(3), 287–317 (1994)
5. Demartines, P., Herault, J.: Curvilinear Component Analysis: A Self-Organizing Neural Network for Nonlinear Mapping of Data Sets. IEEE Trans. Neural Networks 8(1), 148–154 (1997)
6. Dijkstra, E.W.: A note on two problems in connection with graphs. Numerical Mathematics 1, 269–271 (1959)
7. Estévez, P.A., Figueroa, C.J.: Online data visualization using the neural gas network. Neural Networks 19, 923–934 (2006)
8. Estéivez, P.A., Chong, A.M., Held, C.M., Perez, C.A.: Nonlinear Projection Using Geodesic Distances and the Neural Gas Network. In: Kollias, S., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006. LNCS, vol. 4131, pp. 464–473. Springer, Heidelberg (2006)

9. Hebb, D.O.: The organization of behavior. John Wiley and Son, New York (1949)
10. Hoteling, H.: Analysis of a complex of statistical variables into principal components. Journal of Education Psychology 24, 417–441 (1933)
11. Jain, A., Zongker, D.: Feature selection: evaluation, application, and small sample performance. IEEE Trans. on Pattern Analysis and Machine Intelligence 192, 153–158 (1997)
12. Jolliffe, I.T.: Principal Component Analysis. Springer, New York (1996)
13. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (1995)
14. Kruskal, J., Wish, M.: Multidimensional Scaling. SAGE publications, Beverly Hills (1978)
15. Lee, J.A., Lendasse, A., Donckers, N., Verleysen, M.: A Robust Nonlinear Projection Method. In: Verleysen, M. (ed.) Proceedings of ESANN 2000, 8th European Symposium on Artificial Neural Networks, D-Facto public., Bruges (Belgium), pp. 13–20 (2000)
16. Lee, J.A., Verleysen, M.: Nonlinear Projection with the Isotop Method. In: Dorronsoro, J.R. (ed.) ICANN 2002. LNCS, vol. 2415, pp. 933–938. Springer, Heidelberg (2002)
17. Lee, J.A., Lendasse, A., Verleysen, M.: Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis. Neurocomputing 57, 49–76 (2004)
18. Mangasarian, O.L., Wolberg, W.H.: Cancer diagnosis via linear programming. Society for Industrial and Applied Mathematics News 23(5), 1–18 (1990)
19. Martinetz, T.M., Schulten, K.J.: A neural-gas network learns topologies. In: Kohonen, T., Mäkisara, K., Simula, O., Kangas, J. (eds.) Artificial Neural Networks, pp. 397–402. North-Holland, Amsterdam (1991)
20. Martinetz, T.M., Shulten, K.J.: Topology representing networks. Neural Networks 7(3), 507–522 (1994)
21. Militk, J., Meloun, M.: Some graphical aids for univariate exploratory data analysis. Analytica Chimica Acta 277(2), 215–221 (1993)
22. Muhammed, H.H.: Unsupervised Fuzzy Clustering Using Weighted Incremental Neural Networks. International Journal of Neural Systems (IJNS), 14(6), 355–371 (2004)
23. Narendra, P., Fukunaga, K.: A branch and bound algorithm for feature subset selection. IEEE Trans. on Computers C-269, 917–922 (1977)
24. Naud, A.: Neural and statistical methods for the visualization of multidimensional data. Thesis, Technical Science Katedra Metod Komputerowych Uniwersytet Mikoaja Kopernika w Toruniu (2001)
25. Pudil, P., Novoviov, J., Kittler, J.: Floating search methods in feature selection. Pattern Recognition Letters 15(1), 119–125 (1994)
26. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science 290, 2323–2326 (2000)
27. Sammon, J.W.: A Non-Linear Mapping for Data Structure Analysis, IEEE Trans. on Computers C18(5), 401–409 (1969)
28. Si, J., Lin, S., Vuong, M.-A.: Dynamic topology representing networks. Neural Networks 13, 617–627 (2000)
29. Tenenbaum, J.B., Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290, 2319–2323 (2000)
30. Tukey, J.: Exploratory Data Analysis. Addison-Wesley, Reading (1977)
31. Ultsch, A.: Self-organization neural networks for visualization and classification. In: Opitz, O., Lausen, B., Klar, R. (eds.) Information and classification, pp. 307–313. Springer, Heidelberg (1993)

32. Wu, Y., Chan, K.L.: An Extended Isomap Algorithm for Learning Multi-Class Manifold. In: Proceeding of IEEE International Conference on Machine Learning and Cybernetics (ICMLC 2004), Shanghai, China, vol. 6, pp. 3429–3433 (2004)
33. Young, G., Householder, A.S.: Discussion of a set of points in terms of their mutual distances. Psychometrika 3(1), 19–22 (1938)

Curve Fitting by Fractal Interpolation

Polychronis Manousopoulos, Vassileios Drakopoulos, and Theoharis Theoharis

Department of Informatics and Telecommunications,
University of Athens, Panepistimioupolis, 157 84, Athens, Greece
{polyman,vasilios,theotheo}@di.uoa.gr

Abstract. Fractal interpolation provides an efficient way to describe data that have an irregular or self-similar structure. Fractal interpolation literature focuses mainly on functions, i.e. on data points linearly ordered with respect to their abscissa. In practice, however, it is often useful to model curves as well as functions using fractal intepolation techniques. After reviewing existing methods for curve fitting using fractal interpolation, we introduce a new method that provides a more economical representation of curves than the existing ones. Comparative results show that the proposed method provides smaller errors or better compression ratios.

Keywords: fractal interpolation, curve fitting, iterated function systems.

1 Introduction

Fractal interpolation has been developed as an alternative interpolation technique suitable for capturing data with inherent *fractal structure*, i.e. details at different scales or some degree of self-similarity. In contrast to traditional interpolation, which is built on elementary functions such as polynomials, fractal interpolation is based on the theory of iterated function systems producing interpolants that are convenient for fitting physical or experimental data.

Fractal interpolation literature focuses on functions, i.e. the data points are linearly ordered with respect to their abscissa and the interpolant is a function of (usually) non-integral dimension. This is often sufficient, e.g. when interpolating time series data. In practice, however, there are many cases where the data are suitable for fractal interpolation but define a curve rather than a function, e.g. when modelling coastlines or plants. So, it is useful to extend fractal interpolation to include curves as well as functions, an issue not fully addressed so far. Methods based on generalizations to higher dimensions are introduced in [1], [2] and [3]. The use of index coordinates is suggested in [4]. Non-affine fractal interpolation is employed in [5]. Various combinations of IFS models and free form curves are proposed in [6] and [7]. A method of data fitting by means of fractal interpolation functions is proposed in [8]. An interpolation method for multifractal structures is presented in [9].

In this paper we review existing approaches in this area and introduce a new method for curve fitting by fractal interpolation. Our motivation is to create a method that is more accurate and economical than the existing ones, thus

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 85–103, 2008.

being more suitable for practical applications such as shape representation. All methods are compared in practical applications showing the advantages of the proposed FCF method in terms of either accuracy or compression ratio. The paper is structured as follows. In Sect. 2 we present the necessary background on fractal interpolation functions. Section 3 contains existing appoaches to curve fitting by fractal interpolation, while Sect. 4 introduces the new method. Section 5 contains the application of our method in various practical cases and comparisons against previous approaches. Finally, Sect. 6 presents our conclusions and indicates areas of future work.

2 Fractal Interpolation Functions

Fractal interpolation functions as defined in [10] and [11] are based on the theory of *iterated function systems*. An iterated function system (IFS), denoted by $\{X; w_n, n = 1, 2, \ldots, N\}$, consists of a complete metric space (X, ρ), e.g. $(\mathbb{R}^n, ||\cdot||)$ or a subset, and a finite set of continuous mappings $w_n: X \to X$, $n = 1, 2, \ldots, N$. If w_n are contractions with respective *contractivity factors* s_n, $n = 1, 2, \ldots, N$, the IFS is termed *hyperbolic*. The transformation $W: \mathcal{H}(X) \to \mathcal{H}(X)$ with $W(B) = \cup_{n=1}^{N} w_n(B)$, where $\mathcal{H}(X)$ denotes the metric space of nonempty compact subsets of X with respect to the Hausdorff metric, has a unique fixed point $A_\infty = W(A_\infty) = \lim_{n\to\infty} W^n(B)$ for every $B \in \mathcal{H}(X)$, which is called the *attractor* of the IFS.

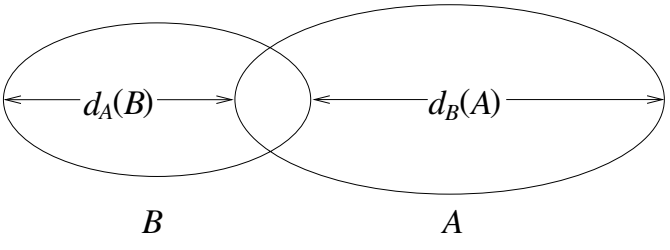


Fig. 1. The difference between $d_A(B)$ and $d_B(A)$

The *Hausdorff distance* between the points A and B of $\mathcal{H}(X)$ is given by

$$h(A, B) = \max\{d_A(B), d_B(A)\},$$

where $d_B(A) = \max\{d(x, B) : x \in A\}$ and $d_A(B) = \max\{d(x, A) : x \in B\}$ (Fig. 1). The function $d_A(B)$ sometimes is called the *directed Hausdorff distance* from A to B. Because of the sensitivity of the Hausdorff metric to noise or isolated points that stems from its 'worst-case' nature, some Hausdorff-like metrics have been proposed such as the *Modified Hausdorff Distance (MHD)* (see [12]). Specifically,

$$h_{MHD}(A,B) = \max\{d_{MHD}(A,B), d_{MHD}(B,A)\}$$

where $d_{MHD}(A,B) = (1/N_a)\Sigma_{a\in A} d(a,B)$, N_a denotes the number of points in A and $d(a,B)$ is the usual point to set distance.

2.1 Fractal Interpolation Functions in the Plane

Let us represent the given set of *data points* as $\{(u_m, v_m) \in \mathbb{R}^2 : m = 0, 1, \ldots, M\}$. In general, the interpolation is applied to a subset of them, the *interpolation points*, represented as $\{(x_i, y_i) \in \mathbb{R}^2 : i = 0, 1, \ldots, N\}$. Both sets are linearly ordered with respect to their abscissa, i.e. $u_0 < u_1 < \cdots < u_M$ and $u_0 = x_0 < x_1 < \cdots < x_N = u_M$. The interpolation points partition the set of data points into *interpolation intervals* and may be chosen equidistantly or not. The greater the number of interpolation points the better the fit of the data, but more interpolation points result in a smaller compression ratio since more information is required to describe the interpolation function.

Let $\{\mathbb{R}^2; w_n, n = 1, 2, \ldots, N\}$ be an IFS with affine transformations

$$w_n \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_n & 0 \\ c_n & s_n \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_n \\ e_n \end{bmatrix}$$

constrained to satisfy

$$w_n \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} \quad \text{and} \quad w_n \begin{bmatrix} x_N \\ y_N \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix}$$

for every $n = 1, 2, \ldots, N$. Solving the above equations results in

$$\begin{aligned} a_n &= \frac{x_n - x_{n-1}}{x_N - x_0} \\ d_n &= \frac{x_N x_{n-1} - x_0 x_n}{x_N - x_0} \\ c_n &= \frac{y_n - y_{n-1}}{x_N - x_0} - s_n \frac{y_N - y_0}{x_N - x_0} \\ e_n &= \frac{x_N y_{n-1} - x_0 y_n}{x_N - x_0} - s_n \frac{x_N y_0 - x_0 y_N}{x_N - x_0}, \end{aligned}$$

i.e. the real numbers a_n, d_n, c_n, e_n are completely determined by the interpolation points, while the s_n are *free* parameters of the transformations satisfying $|s_n| < 1$, in order to guarantee that the IFS is hyperbolic with respect to an appropriate metric. The transformations w_n are *shear transformations*: line segments parallel to the y-axis are mapped to line segments parallel to the y-axis contracted by the factor $|s_n|$. For this reason, the s_n are called *vertical scaling* (or *contractivity*) *factors*.

It is well known (see for example [11]) that the attractor $G = \bigcup_{n=1}^{N} w_n(G)$ of the aforementioned IFS is the graph of a continuous function $f : [x_0, x_N] \to \mathbb{R}$ that passes through the interpolation points. This function is called *fractal interpolation function (FIF)* corresponding to these points. It is a *self-affine* function

since each affine transformation w_n maps the entire (graph of the) function to its *section*, i.e. function values between the interpolation points (x_{n-1}, y_{n-1}) and (x_n, y_n) for all $n=1, 2, \ldots, N$. For example, let $\{(0,0), (0.4, 0.5), (0.7, 0.2), (1,0)\}$ be a given set of data points. Figure 2 shows the graph of an affine FIF with $s_1 = 0.5$, $s_2 = -0.2$ and $s_3 = 0.4$.

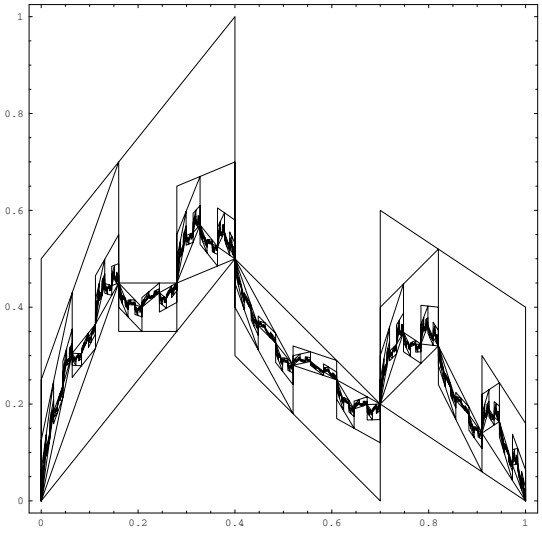


Fig. 2. The construction of an affine FIF starting from the unit square

The graph of a FIF is bounded by the rectangle $[x_0, x_N] \times [a, b]$ if the vertical scaling factors s_n satisfy $s_n^{min} \leq s_n \leq s_n^{max}$ and $|s_n| < 1$, where

$$s_n^{min} = \max\left\{\frac{a-y_{n-1}}{b-y_0}, \frac{a-y_n}{b-y_N}, \frac{b-y_{n-1}}{a-y_0}, \frac{b-y_n}{a-y_N}\right\}$$

$$s_n^{max} = \min\left\{\frac{b-y_n}{b-y_N}, \frac{b-y_{n-1}}{b-y_0}, \frac{a-y_n}{a-y_N}, \frac{a-y_{n-1}}{a-y_0}\right\}$$

for every $n = 1, \ldots, N$ (see [5], [13]).

Although the FIF passes by definition through the interpolation points, this is not necessarily the case for the remaining data points $\{(u_m, v_m)\} \setminus \{(x_i, y_i)\}$. The accuracy of fit can be measured as the squared error between the ordinates of the original and the reconstructed points

$$\varepsilon = \sum_{m=0}^{M} (v_m - G(u_m))^2 \tag{1}$$

or, alternatively, as the *Hausdorff distance* between the two sets

$$\varepsilon = h(\{(u_m, v_m)\}, G).$$

The vertical scaling factors of a FIF are usually chosen so as to minimize such an error measure. For example, in [14] and [15] the minimization of (1) is achieved by algebraic or geometric methods. Moreover in [15], a greedy algorithm for finding some proper (but not necessarilly globally optimal) interpolation points is presented.

An extension of the FIFs are the so-called *piecewise self-affine FIFs* (see [15]), which are essentially an application of the *recurrent IFSs* (see [11]). Their motivation is the fact that data often present self-affinity in subintervals and not in their whole length. This is modelled by the introduction of the *address points*, represented as $\{(\tilde{x}_{n,1}, \tilde{y}_{n,1}), (\tilde{x}_{n,2}, \tilde{y}_{n,2}) \in \mathbb{R}^2 \colon n = 1, 2, \ldots, N\}$, that define the intervals of self-affinity. The affine transformations $w_n, n = 1, 2, \ldots, N$ are then constrained to satisfy

$$w_n \begin{bmatrix} \tilde{x}_{n,1} \\ \tilde{y}_{n,1} \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} \quad \text{and} \quad w_n \begin{bmatrix} \tilde{x}_{n,2} \\ \tilde{y}_{n,2} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix}.$$

Piecewise self-affine FIFs are more flexible than affine FIFs, but require the additional cost of determining the address points. Moreover, a greedy algorithm for locating both some proper (but not necessarilly globally optimal) interpolation and address points is presented in [15].

2.2 Generalized Fractal Interpolation Functions

The FIF model described in the previous section can be extended to higher dimensions, producing functions that interpolate points in $\mathbb{R}^k$. Let $\{p_m \in \mathbb{R}^k \colon m = 0, 1, \ldots, M\}$ be the set of data points and $\{q_i \in \mathbb{R}^k \colon i = 0, 1, \ldots, N\}$ the set of interpolation points. Both sets are again assumed to be linearly ordered with respect to their abscissa. Let $\{\mathbb{R}^k; w_n, n = 1, 2, \ldots, N\}$ be an IFS with affine transformations

$$w_n \begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^k \end{bmatrix} = \underbrace{\begin{bmatrix} a_n & 0 & \cdots & 0 \\ c_n^1 & s_n^{1,1} & \cdots & s_n^{1,k-1} \\ \vdots & \vdots & \ddots & \vdots \\ c_n^{k-1} & s_n^{k-1,1} & \cdots & s_n^{k-1,k-1} \end{bmatrix}}_{k \times k} \underbrace{\begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^k \end{bmatrix}}_{k \times 1} + \begin{bmatrix} d_n^1 \\ d_n^2 \\ \vdots \\ d_n^k \end{bmatrix}$$

constrained to satisfy

$$w_n(q_0) = q_{n-1} \quad \text{and} \quad w_n(q_N) = q_n$$

for every $n = 1, 2, \ldots, N$. The real numbers a_n, c_n^i, d_n^j for every $n = 1, \ldots, N$, $i = 1, 2, \ldots, k-1$ and $j = 1, 2, \ldots, k$ are completely determined by the interpolation points by solving the above equations, while the $s_n^{i,j}$, $i, j = 1, 2, \ldots, k-1$ are free parameters of the transformations chosen such that the contractivity factor s_n of the matrix (called *contractivity matrix*)

$$\begin{bmatrix} s_n^{1,1} & \cdots & s_n^{1,k-1} \\ \vdots & \ddots & \vdots \\ s_n^{k-1,1} & \cdots & s_n^{k-1,k-1} \end{bmatrix}$$

has modulus less than unity, in order to guarantee that the IFS is hyperbolic with respect to an appropriate metric. The exact values of $s_n^{i,j}$ can be determined by minimizing an error measure as in the planar case (see e.g. [2]) .

The attractor $G = \bigcup_{n=1}^{N} w_n(G)$ of the IFS is the graph of a continuous function $f\colon [q_0^1, q_N^1] \to \mathbb{R}^{k-1}$ that interpolates the points q_i, $i = 0, 1, \ldots, N$ (see [11]). It is a self-affine FIF in $\mathbb{R}^k$; however, its orthogonal projections to $\mathbb{R}^2$ are not necessarilly self-affine. The accuracy of fit of a Generalized FIF can be defined similarly to the planar case.

For example, in $\mathbb{R}^3$ we have the IFS $\{\mathbb{R}^3; w_n, n = 1, 2, \ldots, N\}$ with affine transformations

$$w_n \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_n & 0 & 0 \\ c_n^1 & s_n^{1,1} & s_n^{1,2} \\ c_n^2 & s_n^{2,1} & s_n^{2,2} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} d_n^1 \\ d_n^2 \\ d_n^3 \end{bmatrix}$$

constrained to satisfy

$$w_n \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \\ z_{n-1} \end{bmatrix} \quad \text{and} \quad w_n \begin{bmatrix} x_N \\ y_N \\ z_N \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix}$$

for $n = 1, 2, \ldots, N$. Solving the above equations results in

$$\begin{aligned}
a_n &= \frac{x_n - x_{n-1}}{x_N - x_0} \\
d_n^1 &= \frac{x_N x_{n-1} - x_0 x_n}{x_N - x_0} \\
c_n^1 &= \frac{y_n - y_{n-1}}{x_N - x_0} - s_n^{1,1} \frac{y_N - y_0}{x_N - x_0} - s_n^{1,2} \frac{z_N - z_0}{x_N - x_0} \\
c_n^2 &= \frac{z_n - z_{n-1}}{x_N - x_0} - s_n^{2,1} \frac{z_N - z_0}{x_N - x_0} - s_n^{2,2} \frac{z_N - z_0}{x_N - x_0} \\
d_n^2 &= \frac{x_N y_{n-1} - x_0 y_n}{x_N - x_0} - s_n^{1,1} \frac{x_N y_0 - x_0 y_N}{x_N - x_0} - s_n^{1,2} \frac{x_N z_0 - x_0 z_N}{x_N - x_0} \\
d_n^3 &= \frac{x_N z_{n-1} - x_0 z_n}{x_N - x_0} - s_n^{2,1} \frac{x_N y_0 - x_0 y_N}{x_N - x_0} - s_n^{2,2} \frac{x_N z_0 - x_0 z_N}{x_N - x_0}
\end{aligned}$$

i.e. the real numbers $a_n, d_n^1, c_n^1, c_n^2, d_n^2, d_n^3$ are completely determined by the interpolation points, while $s_n^{1,1}, s_n^{1,2}, s_n^{2,1}, s_n^{2,2}$ are *free* parameters of the transformations chosen such that the contractivity factor of the matrix

$$\begin{bmatrix} s_n^{1,1} & s_n^{1,2} \\ s_n^{2,1} & s_n^{2,2} \end{bmatrix}$$

has modulus less than unity.

For example, let $\{(0,2,1), (1,4,3), (2,8,5), (3,6,2), (4,5,6), (5,2,4), (6,3,7), (7,4,4), (8,2,3), (9,1,2)\}$ be a given set of data points in $\mathbb{R}^3$. Figure 3 shows the graph of a Generalized FIF with $s_n^{1,1} = s_n^{1,2} = s_n^{2,1} = s_n^{2,2} = 0.1$. The

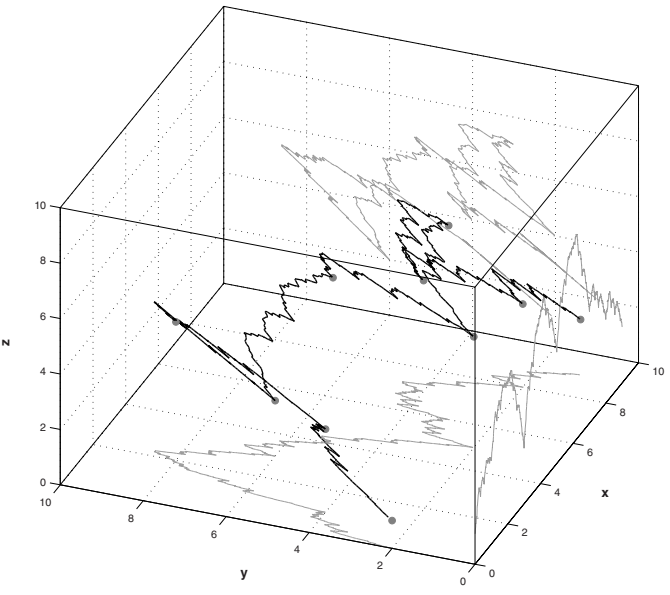


Fig. 3. A Generalized FIF in $\mathbb{R}^3$. The projections of the Generalized FIF on the xy, xz and yz planes are depicted in gray.

concept of piecewise self-affine FIFs in $\mathbb{R}^2$ can be similarly extended to higher dimensions.

Hidden Variable Fractal Interpolation Functions. The Generalized FIFs can also be used for interpolating points in $\mathbb{R}^2$. The idea is to extend the data to a higher-dimensional space, interpolate them by a Generalized FIF and project it back to $\mathbb{R}^2$ to obtain a function that interpolates the original data. Specifically, we apply to the interpolation points the mapping $T: \mathbb{R}^2 \to \mathbb{R}^3$ with $(x_i, y_i) \mapsto (x_i, y_i, H_i)$, $i = 0, 1, \ldots, N$, where the H_i are freely chosen. The new set of points $(x_i, y_i, H_i), i = 0, 1, \ldots, N$ is the generalized set of data corresponding to the original points and is interpolated by creating an IFS in $\mathbb{R}^3$ as described in the previous section. The attractor $G' = \bigcup_{n=1}^{N} w_n(G')$ of the IFS is the graph of a continuous function $f': [x_0, x_N] \to \mathbb{R}^2$ that interpolates the points (x_i, y_i, H_i), $i = 0, 1, \ldots, N$. The orthogonal projection of the attractor to $\mathbb{R}^2$, defined by $P_H: G' \to G$ with $(x, y, H) \mapsto (x, y)$, is the graph of a continuous function $f: [x_0, x_N] \to \mathbb{R}$ that interpolates the points (x_i, y_i), $i = 0, 1, \ldots, N$. The extra coordinate H_i is called *hidden variable* and can be used to adjust the shape of the resulting interpolation function f which is thus called *hidden variable fractal interpolation function (HVFIF)*. Note that although the attactor G' is self-affine, this is not necessarily the case for its projection G.

The hidden variable FIFs can be extended by introducing more than one hidden variables, having thus more free parameters in order to adust the shape of the resulting interpolation function.

3 Existing Applications of FIFs to Curve Fitting in the Plane

When the interpolation points define a curve rather than a function, i.e. they are not linearly ordered with respect to their abscissa, the direct use of a fractal interpolation function is not possible. In order to construct an IFS whose attractor interpolates the given points, and is therefore a curve, we can transform or extend the original points such that the application of a FIF is possible. This is then transformed or projected back to the plane to obtain a curve that interpolates the original points.

3.1 Curves as Projections of Generalized FIFs

One possibility is to extend the idea of hidden-variable FIFs ([1], [2], [3]). We transform the original set of points to a higher-dimensional set that defines a function, then create the respective FIF and project it back to $\mathbb{R}^2$ to obtain a curve that interpolates the original points.

Let $\{(x_i, y_i) \in \mathbb{R}^2 \colon i = 0, 1, \ldots, N\}$ be the set of interpolation points. These points do not define a function but a curve on the xy-plane, i.e. it is not necessarily $x_i < x_j$ for $i < j$. We apply the transformation $T \colon \mathbb{R}^2 \to \mathbb{R}^3$ with $(x_i, y_i) \mapsto (t_i, x_i, y_i)$, $i = 0, 1, \ldots, N$, where the introduced index coordinates t_i satisfy $t_0 < t_1 < \cdots < t_N$; usually we set $t_i = i$. The new set of points $(t_i, x_i, y_i), i = 0, 1, \ldots, N$ is the *generalized set* corresponding to the original points and defines a function. We create a FIF that interpolates the generalized set as described in Sect. 2.2. The attractor $G' = \bigcup_{n=1}^{N} w_n(G')$ of the IFS is the graph of a continuous function $f' \colon [t_0, t_N] \to \mathbb{R}^2$ that interpolates the generalized set of points (t_i, x_i, y_i), $i = 0, 1, \ldots, N$. The projection of the attractor to $\mathbb{R}^2$, defined by $P_t \colon G' \to G$ with $(t_i, x_i, y_i) \mapsto (x_i, y_i)$, is the graph of a continuous curve $f \colon [x_0, x_N] \to \mathbb{R}$ that interpolates the points (x_i, y_i), $i = 0, 1, \ldots, N$ and is thus called *fractal interpolation curve (FIC)* [1]. Note that although the attactor G' is self-affine, its projection G is not necessarily self-affine.

The FIC defined above is open, assuming that the first and last interpolation points are different. In order to construct a closed fractal interpolation curve, we append to the original points an additional one that is the same as the first, i.e. we add $(x_{N+1}, y_{N+1}) = (x_0, y_0)$. The curve is afterwards constructed in the same way. A FIC of this kind is depicted in Fig. 4(a), which is constructed on a simple, manually selected set of 10 interpolation points. Specifically, the data points $\{(3, 1), (2, 2), (1, 4), (0, 3), (-1, 3), (-2, 1), (-1, -1), (0, -2), (2, -1), (3.5, -0.5)\}$ have been used and the contractivity factors s_n have been set to 0.1.

3.2 Curves by Coordinate Separation

A similar way to construct a FIC involves the introduction of index coordinates without generalization to a higher-dimensional space ([4]). Specifically, we split the original set of points into two new sets by introducing an index for each coordinate. Then a fractal interpolation function is constructed for each new set, and these are finally combined in a single curve that interpolates the original points.

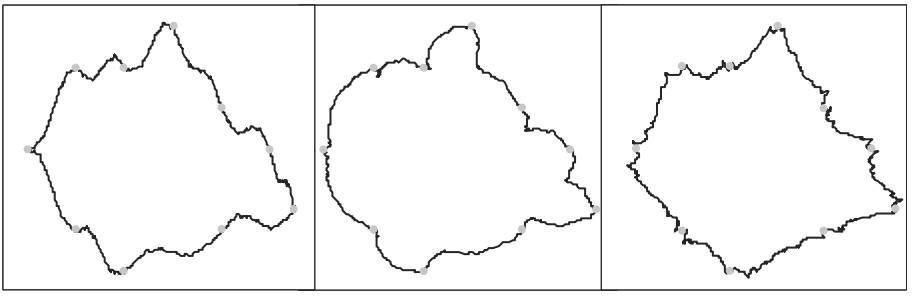


Fig. 4. (a) A FIC constructed by projecting a Generalized FIF. (b) A FIC constructed by coordinate separation. (c) A polar FIF. All three interpolation curves have been constructed from the same ten interpolation points (depicted in grey) using predefined vertical scaling factors.

As previously, let $\{(x_i, y_i) \in \mathbb{R}^2 \colon i = 0, 1, \ldots, N\}$ be the set of interpolation points. We apply the transformations $T_1 \colon \mathbb{R}^2 \to \mathbb{R}^2$ with $(x_i, y_i) \mapsto (t_i, x_i)$, $i = 0, 1, \ldots, N$ and $T_2 \colon \mathbb{R}^2 \to \mathbb{R}^2$ with $(x_i, y_i) \mapsto (t_i, y_i)$, $i = 0, 1, \ldots, N$, where the introduced index coordinates t_i satisfy $t_0 < t_1 < \cdots < t_N$; usually we set $t_i = i$.

Then, we create a fractal interpolation function for each of the two sets in the way described in Sect. 2.1. Let $G^x = (t_i^x, x_i)$ and $G^y = (t_i^y, y_i)$ be the attractors of the respective IFS. We can merge G^x and G^y in order to obtain $G = (x_i, y_i)$ which is the graph of a continuous curve $f \colon [x_0, x_N] \to \mathbb{R}$ that interpolates the points (x_i, y_i), $i = 0, 1, \ldots, N$ and is thus called fractal interpolation curve (FIC). Note that although the attactors G^x and G^y are self-affine, this is not necessarilly the case for G.

We can construct a closed curve, as previously, by appending to the original points an additional one that is the same as the first. A FIC of this kind is depicted in Fig. 4(b), where the same interpolation points and contractivity factors as in Fig. 4(a) have been used.

3.3 Curves as Polar Fractal Interpolation Functions

If the data points that define the curve are ordered by angle in their polar form, we can interpolate them using a class of non-affine FIFs, the polar FIFs (see [5]). Specifically, let $\{(x_i, y_i) \in \mathbb{R}^2 \setminus \{(0, 0)\};\ i = 0, 1, \ldots, N-1\}$ be the set of interpolation points and $(r_i, \theta_i) \in (0, \infty] \times [0, 2\pi)$, $i = 0, 1, \ldots, N-1$ be their representation in polar coordinates obtained by the transformations $x = r \cos\theta$ and $y = r \sin\theta$. We assume that at least one point (x_i, y_i) exists in each quadrant and that it is $0 = \theta_0 < \theta_1 < \cdots < \theta_{N-1} < \theta_N = 2\pi$, i.e. the points define a function in the polar plane.

We create a FIF for the points $(r_i, \theta_i), i = 0, 1, \ldots, N$, where $(r_N, \theta_N) = (r_0, \theta_0)$ and we transform it back to the $xy-$plane to obtain a closed curve

that interpolates the points (x_i, y_i). This FIF is called *polar fractal interpolation function*. Note that this curve is not self-affine since we have used a non-affine (polar) transformation. A polar FIF is depicted in Fig. 4(c), where the same interpolation points and contractivity factors as in Fig. 4(a) have been used.

4 Fractal Interpolation Curves in the Plane

We introduce a new method for creating fractal interpolation curves (FICs) in the plane without using index coordinates or generalizing to a higher-dimensional space. Our motivation is to create a more compact representation of curves using fewer parameters, as will be analyzed in the next section. We apply a reversible transformation to the data points in order to define a function in the plane. Then a FIF is constructed as usual and its attractor is transformed back to the original coordinates in order to obtain a curve that interpolates the original points.

Let us represent the given set of data points as $\{(u_m, v_m) \in \mathbb{R}^2 : m = 0, 1, \ldots, M\}$ and the set of interpolation points as $\{(u_{\mathbb{J}(i)}, v_{\mathbb{J}(i)}) \in \mathbb{R}^2 : i = 0, 1, \ldots, N\}$, where the labelling function $\mathbb{J}\colon \{0, 1, \ldots, N\} \to \{0, 1, \ldots, M\}$ defines the indices of the interpolation points. We apply the transformation $T_1(u_m, v_m) = (u'_m, v'_m)$, $m = 0, 1, \ldots, M$, where

$$u'_m = u_0 + \sum_{j=1}^{m} (|u_j - u_{j-1}| + \varepsilon) = u'_{m-1} + (|u_m - u_{m-1}| + \varepsilon)$$
$$v'_m = v_m,$$

and $\varepsilon > 0$ is an arbitrary constant necessary when all points in an interpolation interval have equal u-coordinates, i.e. $u_m = u_{m-1}$ for every $m = \mathbb{J}(i)+1, \ldots, \mathbb{J}(i+1)$ and some $i \in \{0, 1, \ldots, N\}$. Otherwise, we set $\varepsilon = 0$. The resulting points (u'_m, v'_m), $i = 0, 1, \ldots, M$ are linearly ordered with respect to their abscissa, i.e. $u'_m < u'_n$ for every $m < n$. This transformation is essentially arraying the data points so as to preserve their horizontal distances. This is shown in the example depicted in Fig. 5, where the same interpolation points as in Fig. 4 have been used. Note that this transformation preserves the distances between consecutive points, i.e. $d((u_m, v_m), (u_{m-1}, v_{m-1})) = d((u'_m, v'_m), (u'_{m-1}, v'_{m-1}))$ for all $m = 1, \ldots, M$.

The next step is to create an IFS whose attractor is the graph of a function that interpolates the points $(u'_{\mathbb{J}(i)}, v'_{\mathbb{J}(i)})$, $i = 0, 1, \ldots, N$. This is achieved by using a $2D$ affine IFS (Sect. 2.1) and the result is its attractor G'.

The final step is to apply a second transformation to G' in order to obtain the graph G of a curve that interpolates the initial points $\{(u_m, v_m)\colon m = 0, 1, \ldots, M\}$. Let $(u', v') \in G'$ be a point of the attractor. We apply the transformation $T_2\colon G' \to G$ with $(u', v') \mapsto (u, v)$, where

$$u = u_{m-1} + (u_m - u_{m-1}) \left(\frac{u' - u'_{m-1}}{u'_m - u'_{m-1}} \right), \quad u' \in [u'_{m-1}, u'_m]$$
$$v = v'.$$

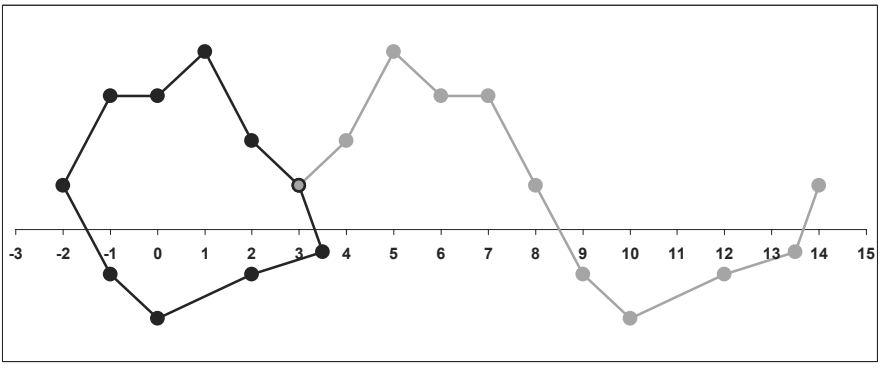


Fig. 5. The interpolation points (black) and their transformation (grey) for the FIC construction

Note that the overlapping at the endpoints of successive intervals $[u'_{i-1}, u'_i]$ in the above formula is not ambiguous, since the resulting u is the same in both cases. The transformation T_2 can be efficiently computed, if the points of the attractor are first sorted by u' and then the attractor and transformed data points are sweeped in parallel in order to calculate the appropriate (u, v).

The FIC defined above is open, assuming that the first and last points are different. To construct a closed FIC, we append an additional interpolation point that is the same as the first, i.e. we add $(u_{\mathbb{J}(N+1)}, v_{\mathbb{J}(N+1)}) = (u_{\mathbb{J}(0)}, v_{\mathbb{J}(0)})$. The curve is afterwards constructed in the same way. A fractal interpolation curve constructed by this method is depicted in Fig. 6, where the same interpolation points and contractivity factors as in Fig. 4(a) have been used. We note that the resulting curve is more similar to the one generated by the projection of Generalized FIF methods.

We call the proposed method *fractal curve fitting (FCF)*. The advantage of the FCF method is that it offers a more compact representation using fewer parameters. As will be explained in the next section, for each interpolation interval it

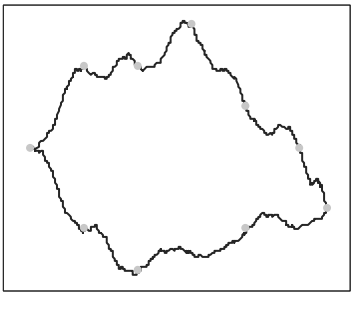


Fig. 6. A fractal interpolation curve constructed by the proposed FCF method

requires five parameters while the methods of Sect. 3.1 and 3.2 require ten. Moreover, it can be readily extended to represent curves in $\mathbb{R}^3$. The third coordinate has the same treatment as v, i.e. it remains unchanged by the transformations.

5 Results

In Fig. 7-9 the coastlines of the Greek islands Kimolos (A), Skyros (B) and Lemnos (C) consisting of 3897, 4663 and 7185 points, respectively, are presented. The coastlines have been extracted from digital aerial photographs of the islands using typical edge detection techniques. These data sets define closed curves and are suitable for fractal interpolation. A traditional interpolation method, using e.g. polynomials, would require very dense interpolation points in order to capture all the fine details of the coastlines.

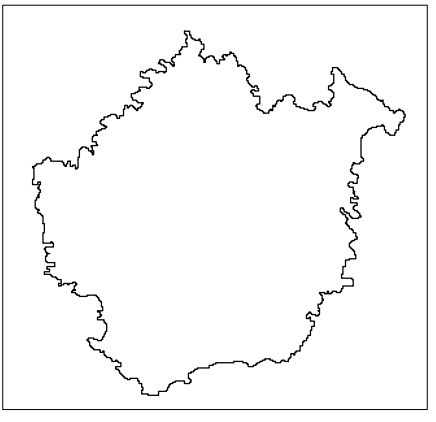


Fig. 7. A data set consisting of 3897 points and representing coastline A (Kimolos)

Tables 1-3 show the Hausdorff distance h and the Modified Hausdorff distance h_{MHD} between the original and reconstructed data for the three aforementioned coastlines, along with the total number of required transformation parameters p. We compare the projection of Generalized FIF method (Sec. 3.1), the coordinate separation method (Sec. 3.2) and the proposed FCF method (Sec. 4)[1]. The contractivity matrix for the projection of generalized FIF method is calculated with the algorithm of [3], while the vertical scaling factors for the other two methods are calculated with the analytic algorithm of [15]. The two algorithms are similar[2], both minimizing the sum of squared distances between original and reconstructed point coordinates using derivatives. Thus the results reflect

[1] We have not compared the polar FIFs since the data are not ordered by angle in their polar form.

[2] The first is essentially an extention of the second.

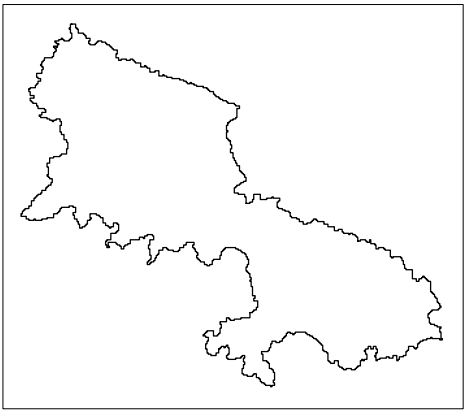


Fig. 8. A data set consisting of 4663 points and representing coastline B (Skyros)

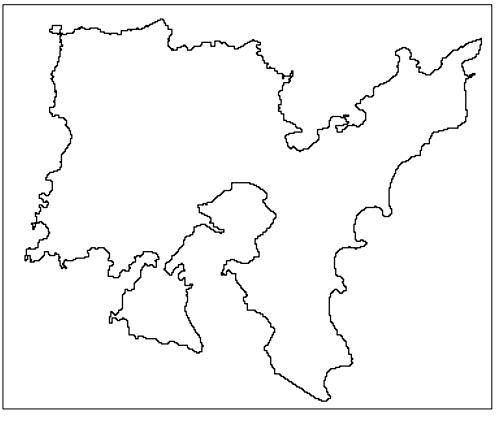


Fig. 9. A data set consisting of 7185 points and representing coastline C (Lemnos)

the differences between the curve construction methods and not between the algorithms for calculating the scaling factors. The interpolation intervals have been chosen with a fixed increment L of 10 to 100, i.e. by taking every 10th to 100th point as interpolation point. As expected for all methods, the smaller the interpolation intervals/compression ratio, the smaller the distance between the original and reconstructed data. We also notice that in a few cases the increase in the length of the interpolation interval decreases the Hausdorff distance. This is rational, since the Hausdorff distance is sensitive to isolated, poorly approximated points. In these cases, the Modified Hausdorff distance provides a better overall comparison.

In terms of Hausdorff and Modified Hausdorff distances, the proposed FCF method significantly outperforms the coordinate separation method in almost

Table 1. The Hausdorff and Modified Hausdorff distance between original/ reconstructed data and the number of required parameters for various interpolation interval lengths (coastline A)

L	Method								
	Proj. of Gen. FIF			Coord. separation			Proposed FCF		
	h	h_{MHD}	p	h	h_{MHD}	p	h	h_{MHD}	p
10	1.559	0.365	3890	2.974	0.519	3890	1.731	0.347	1945
20	2.651	0.486	1940	5.220	1.000	1940	3.157	0.506	970
30	2.976	0.683	1290	8.517	1.552	1290	5.047	0.704	645
40	4.503	0.874	970	11.493	1.939	970	5.545	0.855	485
50	4.987	1.107	770	13.865	2.449	770	6.262	1.113	385
60	10.161	1.330	640	13.930	2.982	640	9.234	1.278	320
70	10.430	1.592	550	16.396	3.378	550	9.225	1.524	275
80	9.967	1.829	480	19.057	3.644	480	14.469	1.766	240
90	10.162	2.085	430	17.758	3.828	430	11.369	1.811	215
100	14.481	2.238	380	18.363	3.928	380	15.559	2.135	190

Table 2. The Hausdorff and Modified Hausdorff distance between original/ reconstructed data and the number of required parameters for various interpolation interval lengths (coastline B)

L	Method								
	Proj. of Gen. FIF			Coord. separation			Proposed FCF		
	h	h_{MHD}	p	h	h_{MHD}	p	h	h_{MHD}	p
10	1.712	0.370	4660	2.105	0.443	4660	1.861	0.339	2330
20	2.202	0.498	2330	4.017	0.755	2330	3.374	0.484	1165
30	3.017	0.664	1550	5.759	1.105	1550	4.364	0.680	775
40	3.788	0.858	1160	6.433	1.362	1160	5.274	0.849	580
50	4.926	1.033	930	9.327	1.701	930	6.211	1.049	465
60	7.596	1.263	770	10.846	1.939	770	7.885	1.278	385
70	7.946	1.460	660	11.908	2.404	660	8.650	1.501	330
80	7.721	1.624	580	11.411	2.356	580	10.538	1.520	290
90	11.632	1.875	510	15.132	3.099	510	12.063	1.907	255
100	11.571	2.007	460	13.173	3.245	460	13.874	2.126	230

all cases and performs equally well to the projection of generalized FIF method. Moreover, the proposed method uses five parameters for each pair of consecutive interpolation points (one affine transformation of five parameters (Eq. 2.1)), while the other two methods require ten parameters. Specifically, the projection of generalized FIF method uses one affine transformation of ten parameters (Eq. 2.2), while the coordinate separation method uses two affine transformations of five parameters each (Eq. 2.1). This implies that for a specific compression ratio we can use the proposed FCF method with twice the number of interpolation points than the other two and thus obtain better results. Comparing the previous results from this point of view it is evident that, for a specific compression

Table 3. The Hausdorff and Modified Hausdorff distance between original/ reconstructed data and the number of required parameters for various interpolation interval lengths (coastline C)

L	Method								
	Proj. of Gen. FIF			**Coord. separation**			**Proposed FCF**		
	h	h_{MHD}	p	h	h_{MHD}	p	h	h_{MHD}	p
10	2.403	0.372	7180	2.836	0.476	7180	2.397	0.353	3590
20	2.666	0.481	3590	5.375	0.742	3590	2.859	0.482	1795
30	3.433	0.627	2390	7.182	1.096	2390	4.564	0.645	1195
40	4.187	0.776	1790	10.063	1.451	1790	6.246	0.844	895
50	5.300	0.966	1430	11.532	1.812	1430	6.648	1.012	715
60	6.342	1.132	1190	13.486	2.326	1190	7.695	1.163	595
70	7.588	1.285	1020	16.582	2.640	1020	9.971	1.451	510
80	8.608	1.482	890	20.037	3.006	890	10.110	1.544	445
90	9.560	1.584	790	21.066	3.462	790	10.813	1.617	395
100	9.845	1.830	710	24.233	3.973	710	12.081	1.865	355

ratio, the proposed method clearly outperforms both others achieving smaller error or, conversely, for a specific error level it achieves better compression ratio. Moreover, it has the advantage that for each pair of consecutive interpolation points only one free parameter (vertical scaling factor) has to be determined, while two and four such parameters are required for the coordinate separation and projection of generalized FIF methods respectively. This is useful, for example, when using these free parameters to describe a family of curves, thus offering a more compact representation.

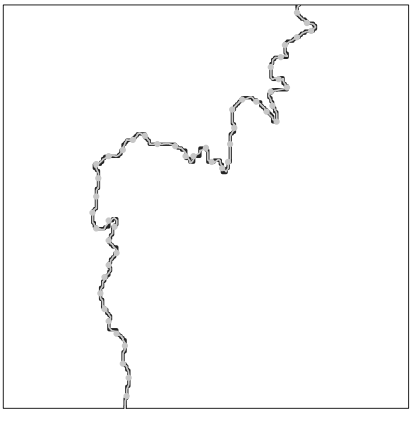


Fig. 10. The data points (black), interpolation points (grey circles) and FIC (grey) for coastline A using the proposed FCF method with interpolation intervals of length 10 ($r = 1{:}\,4.01$)

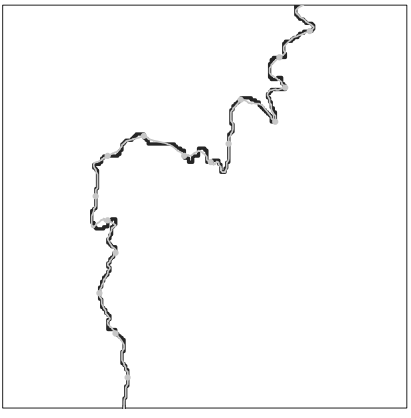


Fig. 11. The data points (black), interpolation points (grey circles) and FIC (grey) for coastline A using the proposed FCF method with interpolation intervals of length 30 ($r = 1{:}\,12.08$)

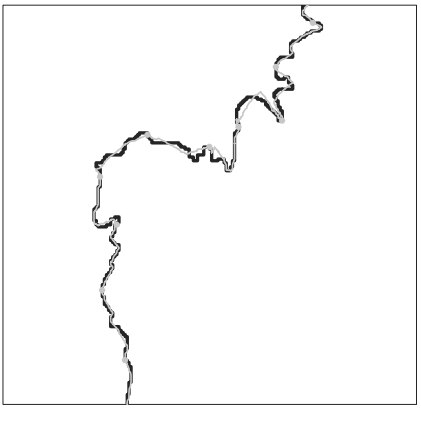


Fig. 12. The data points (black), interpolation points (grey circles) and FIC (grey) for coastline A using the proposed FCF method with interpolation intervals of length 50 ($r = 1{:}\,20.24$)

In Fig. 10–12 parts of the reconstructed FICs for coastline A using the proposed FCF method with interpolation intervals of length 10, 30 and 50 respectively are presented. As shown in the figures, the reconstructed FICs provide an accurate representation of the coastline even with sparse interpolation points, and therefore high compression ratios are achieved. Specifically, the compression ratios for these examples are $1{:}\,4.01$, $1{:}\,12.08$ and $1{:}\,20.24$ respectively[3].

[3] The compression ratio is calculated as $r = 5N/(2M)$, where N is the number of affine transformations and M is the number of data points. Note that the denominator is multiplied by 2 since the data points have two coordinates.

Fig. 13. (a) Coastline D (Amorgos) consisting of 4410 points. (b) The reconstructed curve using FCF method with $r = 1:5.01$.

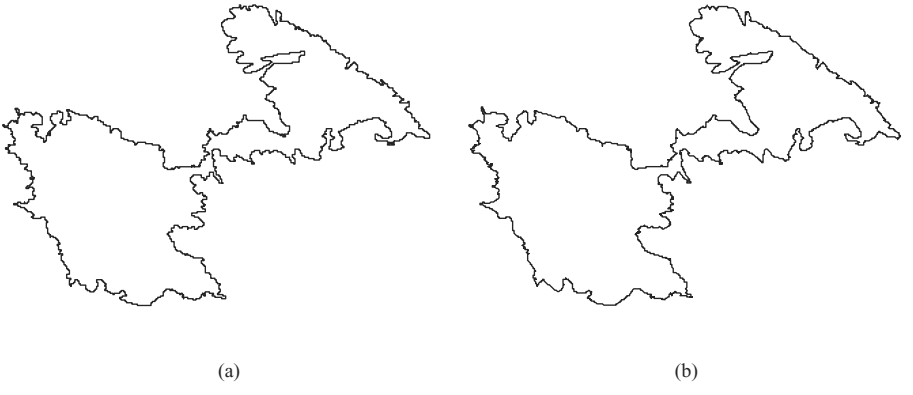
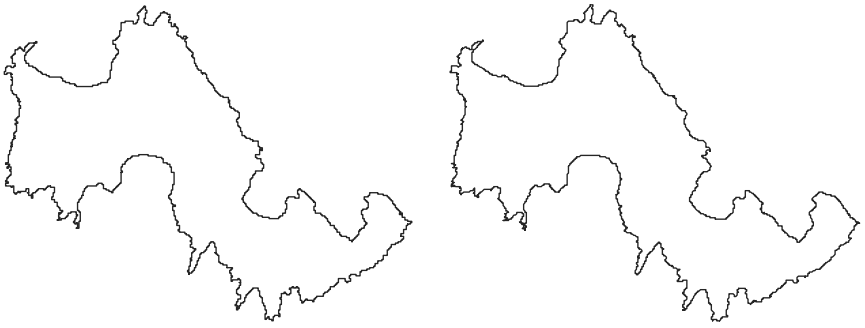


Fig. 14. (a) Coastline E (Astypalaia) consisting of 7510 points. (b) The reconstructed curve using FCF method with $r = 1:6.68$.

In the left part of Fig. 13–15 three more coastlines are presented (Amorgos (D), Astypalaia (E), Tilos (F)), consisting of 4410, 7510 and 6172 points, respectively. In the right part of the figures, the reconstructed curves using FCF method are presented, achieving compression ratios of 1: 5.01, 1: 6.68 and 1: 8.02, respectively. As shown in the figures, the reconstructed FICs accurately represent the coastlines, while requiring considerably less data.

In the previous examples we have used interpolation intervals of fixed length. It is possible to define intervals of variable length, e.g. using the iterative algorithm of [15]. In this case, we could achieve even better results by exploiting more efficiently the possible self-affinity of the data.



(a) (b)

Fig. 15. (a) Coastline F (Tilos) consisting of 6172 points. (b) The reconstructed curve using FCF method with $r = 1{:}\,8.02$.

6 Conclusions

An accurate and economical new method for curve fitting using fractal interpolation has been introduced. Results show that, for a specific compression ratio, the proposed method clearly outperforms existing ones. Moreover, it has the advantage of offering a more economical representation using fewer bound and free parameters. Further work will focus on using piecewise self-affine FIFs ([15]) for interpolating the transformed data of the proposed method. This approach is expected to be better for curves that present self-affinity in subintervals and not at their whole length. Moreover, it will be useful to define bounds for the contractivity factors such that the resulting curve is not self-intersecting.

Acknowledgements. The authors would like to acknowledge financial support from the 03ED375 research project, implemented within the framework of the Reinforcement Programme of Human Research Manpower (PENED) and co-financed by National and Community Funds (25% from the Greek Ministry of Development-General Secretariat of Research and Technology and 75% from E.U.-European Social Fund).

References

1. Cochran, W.O., Hart, J.C., Flynn, P.J.: On approximating rough curves with fractal functions. Proc. Graphics Interface. 1, 65–72 (1998)
2. Mazel, D.S.: Representation of discrete sequences with three-dimensional iterated function systems. IEEE Trans. Signal Processing 42, 3269–3271 (1994)
3. Mazel, D.S., Hayes, M.H.: Hidden-variable fractal interpolation of discrete sequences. Proc. Int. Conf. ASSP. 1, 3393–3396 (1991)

4. Uemura, S., Haseyama, M., Kitajima, H.: Efficient contour shape description by using fractal interpolation functions. IEEE Proc. ICIP. 1, 485–488 (2002)
5. Dalla, L., Drakopoulos, V.: On the parameter identification problem in the plane and the polar fractal interpolation functions. J. Approx. Theory 101, 290–303 (1999)
6. Guérin, E., Tosan, E., Baskurt, A.: Fractal coding of shapes based on a projected IFS model. In: ICIP, vol. 2, pp. 203–206. IEEE Computer Society Press, Los Alamitos (2000)
7. Guérin, E., Tosan, E., Baskurt, A.: A fractal approximation of curves. Fractals 9, 95–103 (2001)
8. Navascués, M.A., Sebastián, M.V.: Fitting curves by fractal interpolation: An application to the quantification of cognitive brain processes. In: Novak, M.M. (ed.) Thinking in patterns: Fractals and related phenomena in nature, pp. 143–154. World Scientific, Singapore (2004)
9. Cader, A., Krupski, M.: New interpolation method with fractal curves. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 1071–1081. Springer, Heidelberg (2006)
10. Barnsley, M.F.: Fractal functions and interpolation. Constr. Approx. 2, 303–329 (1986)
11. Barnsley, M.F.: Fractals everywhere, 2nd edn. Academic Press, San Diego (1993)
12. Zhao, C., Shi, W., Deng, Y.: A new Hausdorff distance for image matching. Pattern Recognition Lett. 26, 581–586 (2005)
13. Ruan, H.J., Sha, Z., Su, W.Y.: Counterexamples in parameter identification problem of the fractal interpolation functions. J. Approx. Theory 122, 121–128 (2003)
14. Marvasti, M., Strahle, W.: Fractal geometry analysis of turbulent data. Signal Processing 41, 191–201 (1995)
15. Mazel, D.S., Hayes, M.H.: Using iterated function systems to model discrete sequences. IEEE Trans. Signal Processing 40, 1724–1734 (1992)

Building Fuzzy Inference Systems with a New Interval Type-2 Fuzzy Logic Toolbox

Juan R. Castro[1], Oscar Castillo[2], Patricia Melin[2], and Antonio Rodríguez-Díaz[1]

[1] Baja California Autonomous University, UABC. Tijuana, Mexico
jrcastror@uabc.mx, ardiaz@uabc.mx
[2] Department of Computer Science, Tijuana Institute of Technology, Tijuana, Mexico
ocastillo@tectijuana.mx, pmelin@tectijuana.mx

Abstract. This paper presents the development and design of a graphical user interface and a command line programming Toolbox for construction, edition and simulation of Interval Type-2 Fuzzy Inference Systems. The Interval Type-2 Fuzzy Logic System (IT2FLS) Toolbox, is an environment for interval type-2 fuzzy logic inference system development. Tools that cover the different phases of the fuzzy system design process, from the initial description phase, to the final implementation phase, constitute the Toolbox. The Toolbox's best qualities are the capacity to develop complex systems and the flexibility that allows the user to extend the availability of functions for working with the use of type-2 fuzzy operators, linguistic variables, interval type-2 membership functions, defuzzification methods and the evaluation of Interval Type-2 Fuzzy Inference Systems.

Keywords: Interval Type-2 Fuzzy Inference Systems, Interval Type-2 Fuzzy Logic Toolbox, Interval Type-2 Membership Functions, Footprint of Uncertainty.

1 Introduction

Over the past decade, fuzzy systems have displaced conventional technologies in different scientific and system engineering applications, especially in pattern recognition and control systems. The same fuzzy technology, in approximation reasoning form, is resurging also in the information technology, where it is now giving support to decision-making and expert systems with powerful reasoning capacity and a limited quantity of rules. The fuzzy sets were presented by L.A. Zadeh in 1965 [1-3] to process or manipulate data and information affected by unprobabilistic uncertainty/imprecision. These were designed to mathematically represent the vagueness and uncertainty of linguistic problems; thereby obtaining formal tools to work with intrinsic imprecision in different type of problems; it is considered a generalization of the classic set theory. Intelligent Systems based on fuzzy logic are fundamental tools for nonlinear complex system modeling. Fuzzy sets and fuzzy logic are the base for fuzzy systems, where the objective has been to model how the brain manipulates inexact information. Type-2 fuzzy sets are used for

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 104–114, 2008.

modeling uncertainty and imprecision in a better way. These type-2 fuzzy sets were originally presented by Zadeh in 1975 and are essentially "fuzzy fuzzy" sets where the fuzzy degree of membership is a type-1 fuzzy set [4,6]. The new concepts were introduced by Mendel and Liang [8,9] allowing the characterization of a type-2 fuzzy set with a inferior membership function and an superior membership function; these two functions can be represented each one by a type-1 fuzzy set membership function. The interval between these two functions represents the footprint of uncertainty (FOU), which is used to characterize a type-2 fuzzy set. The uncertainty is the imperfection of knowledge about the natural process or natural state. The statistical uncertainty is the randomness or error that comes from different sources as we use it in a statistical methodology. Type-2 fuzzy sets have been applied to a wide variety of problems by Castillo and Melin [13].

2 Interval Type-2 Fuzzy Set Theory

A type-2 fuzzy set [6,7] expresses the non-deterministic truth degree with imprecision and uncertainty for an element that belongs to a set. A type-2 fuzzy set denoted by $\tilde{\tilde{A}}$, is characterized by a type-2 membership function $\mu_{\tilde{\tilde{A}}}(x,u)$, where x∈X, u ∈ $J_x^u \subseteq [0,1]$ and $0 \le \mu_{\tilde{\tilde{A}}}(x,u) \le 1$ is defined in equation (1).

$$\tilde{\tilde{A}} = \left\{(x, \mu_{\tilde{\tilde{A}}}(x)) \mid x \in X\right\} = \left\{ \int_{x \in X} \left[\int_{u \in J_x^u \subseteq [0,1]} f_x(u)/u \right] / x \right\} \tag{1}$$

An example of a type-2 membership function constructed in the IT2FLS Toolbox [15] was composed by a Pi primary and a Gbell secondary type-1 membership functions, these are depicted in Figure 1.

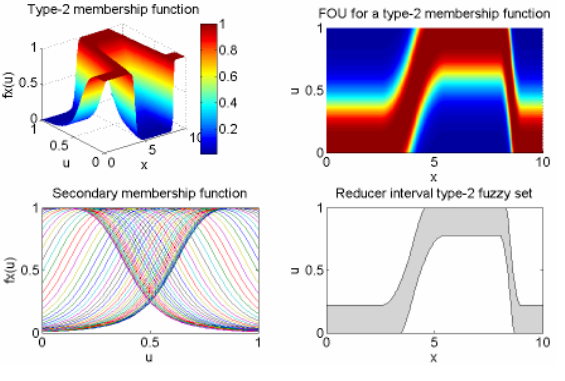


Fig. 1. FOU for Type-2 Membership Functions

If $f_x(u)=1, \forall u \in [\underline{J}_x^u, \overline{J}_x^u] \subseteq [0,1]$, the type-2 membership function $\mu_{\tilde{\tilde{A}}}(x,u)$ is expressed by one inferior type-1 membership function, $\underline{J}_x^u \equiv \underline{\mu}_A(x)$ and one superior type-1 membership function, $\overline{J}_x^u \equiv \overline{\mu}_A(x)$ (Fig. 2), then it is called an interval type-2 fuzzy set [8] denoted by equation (2).

$$\tilde{\tilde{A}} = \left\{ \int_{x \in X} \left[\int_{u \in [\underline{\mu}_A(x), \overline{\mu}_A(x)] \subseteq [0,1]} 1/u \right] / x \right\} \tag{2}$$

If $\tilde{\tilde{A}}$ is a type-2 fuzzy singleton, the membership function is defined by equation (3).

$$\mu_{\tilde{\tilde{A}}}(x) = \begin{Bmatrix} 1/1 & si\ x = x' \\ 1/0 & si\ x \neq x' \end{Bmatrix} \tag{3}$$

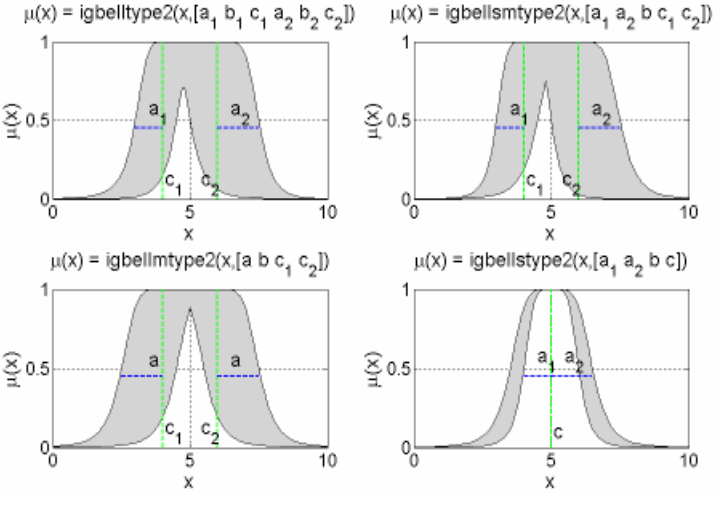


Fig. 2. FOU for Gbell Primary Interval Type-2 Membership Functions

Interval Type-2 Fuzzy Inference System

The human knowledge is expressed as a set of fuzzy rules. The fuzzy rules are basically of the form IF <Antecedent> THEN <Consequent> and express a fuzzy relationship or proposition. In fuzzy logic the reasoning is imprecise, it is approximated, which means that we can infer from one rule a conclusion even if the

antecedent doesn't comply completely. We can count on two basic inference methods between rules and inference laws, Generalized Modus Ponens (GMP) [5,6,8,11] and Generalized Modus Tollens (GMT) that represent the extensions or generalizations of classic reasoning. The GMP inference method is known as direct reasoning and is resumed as:

Rule 1: *IF x_1 is A_{11} and x_2 is A_{21} THEN y_1 is C_1*
Rule 2: *IF x_1 is A_{12} and x_2 is A_{22} THEN y_1 is C_2*
Fact: *x_1 is B1 and x2 is B2*

Conclusion: *y_1 is C*

Where A_{11}, A_{12}, A_{21}, A_{22},C_1,C_2, B_1, *and* B_2 are interval type-2 fuzzy sets. This relationship is expressed as:

$$C_1' = [B_1 \circ (A_{11} \rightarrow C_1)] \sqcap [B_2 \circ (A_{21} \rightarrow C_1)]$$
$$C_2' = [B_1 \circ (A_{21} \rightarrow C_2)] \sqcap [B_2 \circ (A_{22} \rightarrow C_2)]$$

$C = C_1' \sqcup C_2'$, where $\sqcap$ =meet and $\sqcup$=join [10,11,13]. Figure 3 shows an example of non-singleton interval type-2 fuzzy logic system with Mamdani reasoning [9], two inputs x_1 and x_2 and output y_1. An Interval type-2 Fuzzy Inference System is a rule base system that uses Interval type-2 fuzzy logic, instead of Boolean logic utilized in data analysis [4,9,11,12]. A rule based Fuzzy Logic System (FLS) contains four components: Rules, fuzzifier, inference engine, and output processor that are inter-connected, as shown in Figure 4.

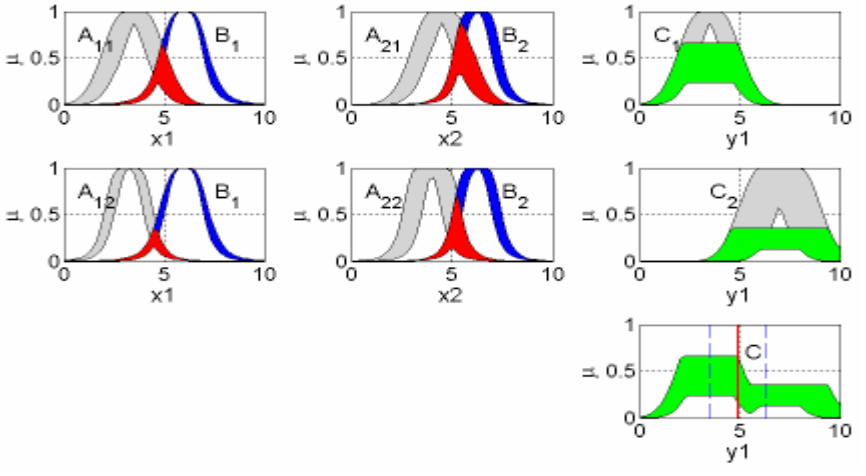


Fig. 3. Interval Type-2 Fuzzy Reasoning

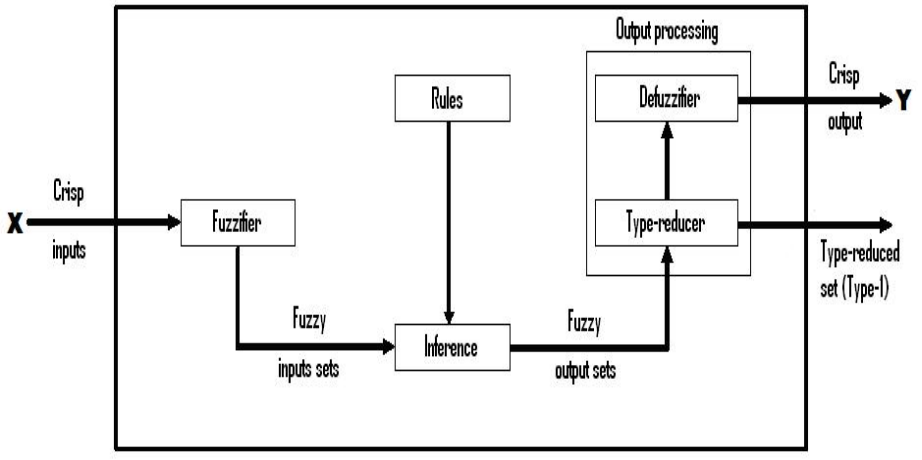


Fig. 4. Type-2 fuzzy logic System

3 Interval Type-2 Fuzzy Logic System Design

The Mamdani and Takagi-Sugeno-Kang Interval Type-2 Fuzzy Inference Models [9] and the design of Interval Type-2 membership functions and operators are implemented in the IT2FLS (Interval Type-2 Fuzzy Logic Systems) Toolbox which was build on top of the Matlab® commercial Fuzzy Logic Toolbox. The IT2FLS Toolbox [15] contains the functions to create Mamdani and TSK Interval Type-2 Fuzzy Inference Systems (newfistype2.m), functions to add input-output variables and their ranges (addvartype2.m), it has functions to add 22 types of Interval Type-2 Membership functions for input-outputs (addmftype2.m), functions to add the rule matrix (addruletype2.m), it can evaluate the Interval Type-2 Fuzzy Inference Systems (evalifistype2.m), evaluate Interval Type-2 Membership functions (evalimftype2.m), it can generate the initial parameters of the Interval Type-2 Membership functions (igenparamtype2.m), it can plot the Interval Type-2 Membership functions with the input-output variables (plotimftype2.m), it can generate the solution surface of the Fuzzy Inference System (gensurftype2.m), it plots the Interval type-2 membership functions (plot2dtype2.m, plot2dctype2.m), a folder to evaluate the derivatives of the Interval type-2 Membership Functions (dit2mf) and a folder with different and generalized Type-2 Fuzzy operators (it2op, t2op).

The implementation of the IT2FLS GUI is analogous to the GUI used for Type-1 FLS in the Matlab® Fuzzy Logic Toolbox, thus allowing the experienced user to adapt easily to the use of IT2FLS GUI [15]. Figures 5 and 6 show the main viewport of the Interval Type-2 Fuzzy Inference Systems Structure Editor called IT2FIS (Interval Type-2 Fuzzy Inference Systems).

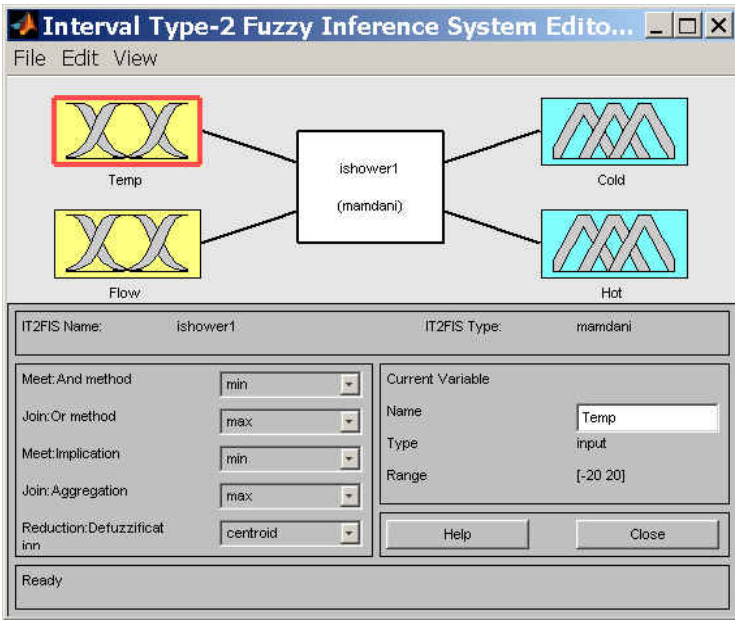


Fig. 5. IT2FIS Editor

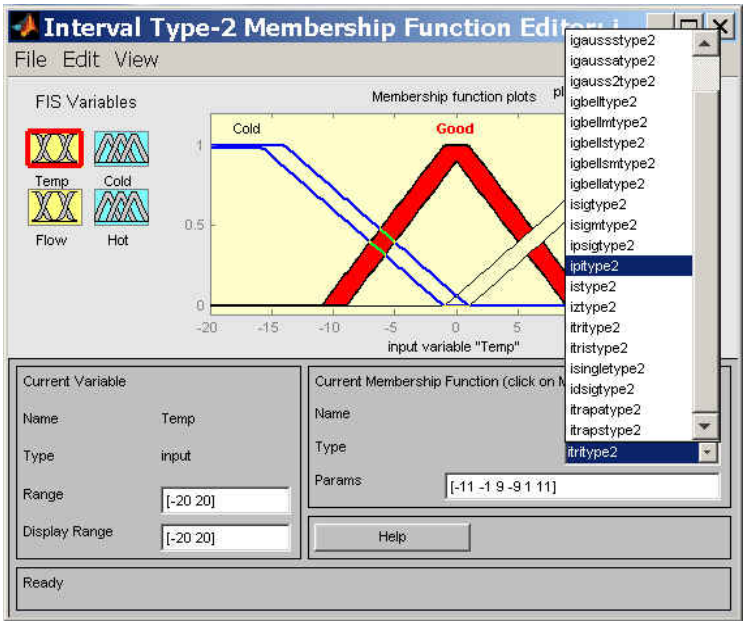


Fig. 6. Interval Type-2 MF's Editor

4 Simulation Results

We present results of a comparative analysis of the Mackey-Glass chaotic time-series forecasting study using an intelligent cooperative architecture of hybrid methods (neuro-genetic, fuzzy-genetic and neuro-fuzzy), with neural networks, type-1 fuzzy inference systems (Mamdani, Takagi-Sugeno-Kang), genetic algorithms (GA) and an interval type-2 fuzzy logic model, for the implicit knowledge acquisition in a time series behavioral data history [14]. Also we present a shower simulation and a truck backer-upper simulation with interval type-2 fuzzy logic systems using the IT2FLS Toolbox.

Mackey-Glass Chaotic Time-Series

To identify the model we make an exploratory series analysis with 5 delays, $L^5x(t)$, 6 periods and 500 training data values to forecast 500 output values. The IT2FLS (Takagi-Sugeno-Kang) system works with 4 inputs, 4 interval type-2 membership functions (igbellmtype2) for each input, 4 rules (Fig. 7) and one output with 4 interval linear functions, it is evaluated with no normalized values. The forecasted root mean square error (RMSE) is 0.0235. Table 1 shows the RMSE differences of six forecasting methods, where CANFIS (CoActive Neuro-Fuzzy Inference Systems) and IT2FLS-TSK (Takagi-Sugeno-Kang) evaluate the best Mackey-Glass series forecasts respectively. The advantage of using the interval type-2 fuzzy logic forecasting method is that it obtains better results, even when data contains high levels of noise.

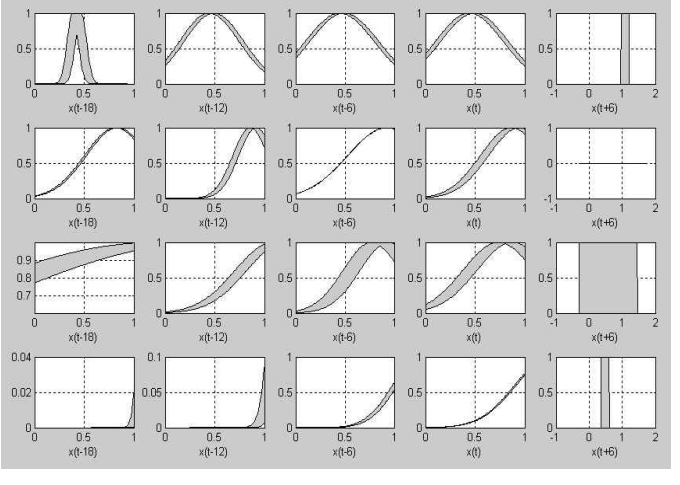


Fig. 7. IT2FLS (TSK) Rules

Table 1. Forecasting of Time Series

METHODS	MACKEY-GLASS			
	RMSE	Trn/Chk	Epoch	Cpu(s)*
FFNN(Feed-Forward Neural Networks)**†	0.0595	500/500	200	13.36
CANFIS	0.0016	500/500	50	7.34
FFNN-GA†	0.0236	500/500	150	98.23
FLS(TKS)-GA†	0.0647	500/500	200	112.01
FLS(Mamdani)-GA†	0.0693	500/500	200	123.21
IT2FLS	0.0235	500/500	6	20.47

* POWER BOOK G4 1.5 Ghz / 512 MB RAM.
** Architecture: 4-13-1. † 30 samples average.

Shower Control Simulation

In this experiment we evaluate the system response to compare the type-1 and type-2 controls with the ISE (Integral of Square Error), IAE (Integral of Absolute value of the Error) and ITAE (Integral of the Time multiplied by the Absolute value of the Error) functionality criteria. The best results were in type-2 controls, as shown in Table 2. In figure 8 we show an interval type-2 fuzzy control scheme and in figures 9 the control results.

Table 2. Control functionality criteria comparison

TYPE FLS	ISE	IAE	ITAE	VARIABLE
Type-1	277.3	76.21	3934	Temperature
Type-2	243.8	64.88	3344	Temperature
Type-1	0.6735	3.5153	172.35	Flow
Type-2	0.6427	3.3039	162.63	Flow

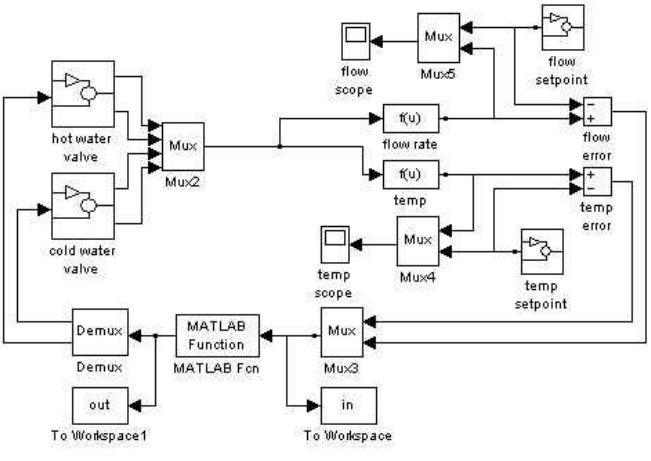


Fig. 8. Simulink interval type-2 fuzzy control scheme

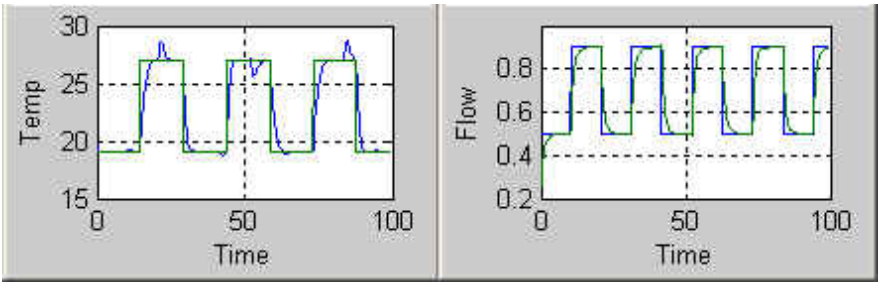


Fig. 9. Temperature and flow interval type-2 fuzzy control

Truck Backer-Upper Control Simulation

In this case study we use a SNR=28 dB signal-to-noise-ratio to generate uncertainty in the plant output variables. We compare the type-1 and type-2 controls using the mean functionality criteria for each trajectory, obtaining the following results: ISE=2.2053, IAE=2.9759 y ITAE=6.2091 for type-1 and ISE=2.0386, IAE=2.8301 y ITAE=5.7254 for type-2. The type-2 controller was better. In figure 10 have the interval type-2 fuzzy control' scheme and in figure 11 the control results of the car trajectories.

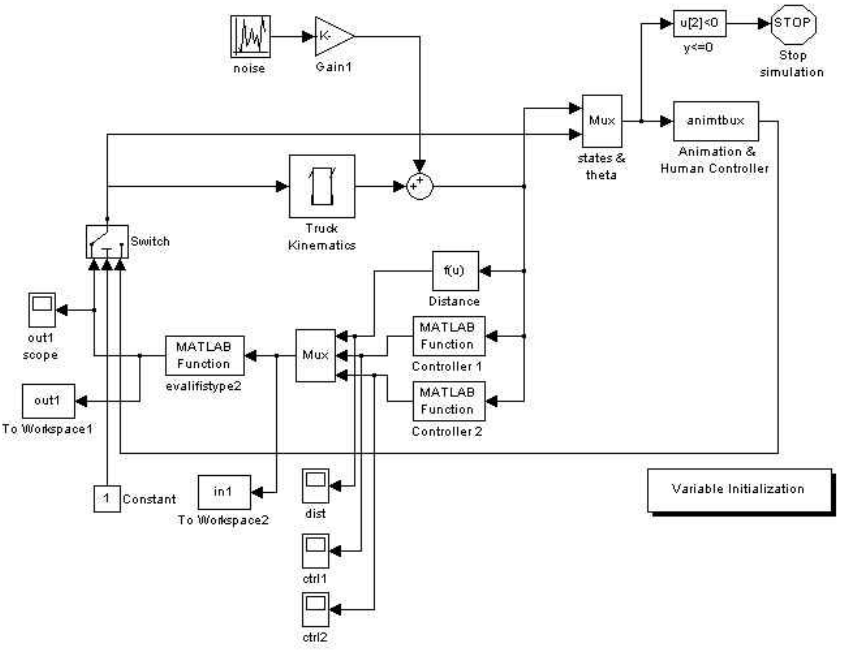


Fig. 10. Simulink interval type-2 fuzzy control scheme

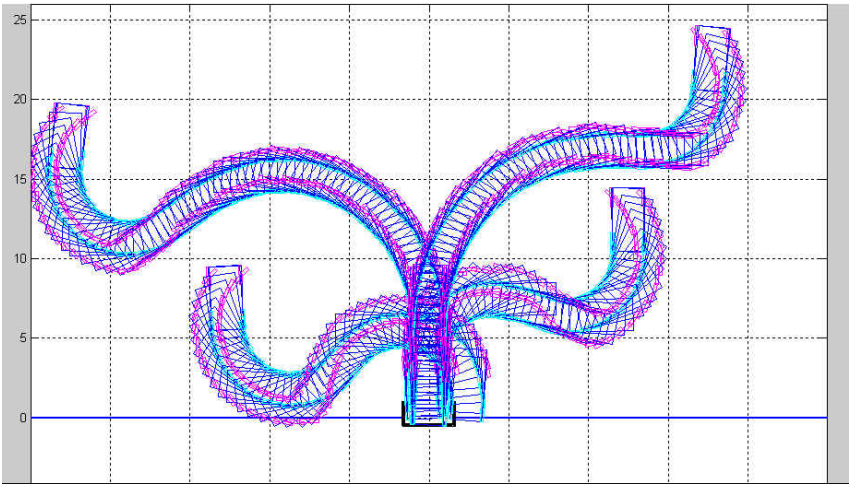


Fig. 11. Trajectory Interval type-2 fuzzy control

5 Conclusions

The time series results show that intelligent hybrid methods and interval type-2 fuzzy models can be derived as a generalization of the autoregressive non-linear models in the context of time series. This derivation allows a practical specification for a general class of prognosis and identification time series models, where a set of input-output variables are part of the dynamics of the time series knowledge base. This helps the application of the methodology to a series of diverse dynamics, with a very low number of causal variables to explain the behavior. The results in the interval type-2 fuzzy control cases of the shower and truck backer upper have similar results to the type-1 fuzzy control with moderate footprints of uncertainty. To better characterize the interval type-2 fuzzy models we need to generate more case studies with better knowledge bases for the proposed problems, therefore classify the interval type-2 fuzzy model application strengths. The design and implementation of the IT2FLS Toolbox is potentially important for research in the interval type-2 fuzzy logic area, thus solving complex problems on the different applied areas. Our future work includes improving the IT2FLS Toolbox with a better graphics user interface (GUI), integrating a learning technique Toolbox to optimize the knowledge base parameters of the interval type-2 fuzzy inference system, and the design of interval type-2 fuzzy neural network hybrid models.

References

1. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)
2. Zadeh, L.A.: Outline of a new approach to the analysis of complex systems and decision processes. IEEE Transactions on Systems, Man, and Cybernetics 3(1), 28–44 (1973)
3. Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning, Parts 1, 2, and 3. Information Sciences 9, 43–80, 8, 199–249, 8, 301–357 (1975)

4. Zadeh, L.A.: Fuzzy Logic. Computer 1(4), 83–93 (1988)
5. Zadeh, L.A.: Knowledge representation in fuzzy logic. IEEE Transactions on Knowledge and Data Engineering 1, 89–100 (1989)
6. Karnik, N.N., Mendel, J.M.: An Introduction to Type-2 Fuzzy Logic Systems. Univ. of Southern Calif, Los Angeles (1998b)
7. Zadeh, L.: Fuzzy logic = computing with words. IEEE Transactions on Fuzzy Systems 2, 103–111 (1996)
8. Liang, Q., Mendel, J.: Interval type-2 fuzzy logic systems: Theory and design. IEEE Transactions Fuzzy Systems 8, 535–550 (2000)
9. Mendel, J.: Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions. Prentice-Hall, NJ (2001)
10. Mizumoto, M., Tanaka, K.: Some Properties of Fuzzy Sets of Type-2. Information and Control 31, 312–340 (1976)
11. Yager, R.R.: On the Implication Operator in Fuzzy logic. Information Sciences 31, 141–164 (1983)
12. Yager, R.: On a general class of fuzzy connectives. Fuzzy Sets and Systems 4, 235–242 (1980)
13. Castillo, Melin: A New Method for Adaptive control of Non-Linear Plants using Type-2 Fuzzy Logic and Neural Networks. International Journal General Systems 33(2), 289–304
14. Castro, J.R.: Hybrid Intelligent Architecture Development for Time Series Forecasting. Masters Degree Thesis. Tijuana Institute of Technology (December 2005)
15. Castro, J.R.: Interval Type-2 Fuzzy Logic Toolbox. In: Proceedings of International Seminar on Computational Intelligence, pp.100–108. Tijuana Institute of Technology (October 2006)

Comparative Analysis of Electrocardiogram Data by Means of Temporal Locality Approach with Additional Normalization

Victor F. Dailyudenko

Institute of Informatics Problems NAS of Belarus,
Surganov St. 6, 220012, Minsk, Belarus
selforg@newman.bas-net.by

Abstract. Evolution of cardiac activity is investigated by means of nonlinear dynamics, namely the method of temporal localization on the attractor reconstructed from a digitized electrocardiogram signal. Convergence for the function of topological instability at changing dimensionality is proven both theoretically and numerically, independently from personal features of subjects in a latter case. This provides an opportunity to estimate the complexity (expressed through the number of freedom degrees) of cardiac dynamics. On the other hand, this instability function normalized by its average displays a different kind of behavior that somewhat differs for various persons and reflects their individual features. The essential reduction of computation time and necessary statistics are also attained by means of the developed algorithm.

Keywords: Nonlinear dynamics, topological analysis, phase space, embedding, ECG time series.

1 Introduction

Most recent investigations of physiologic processes such as heart rate, blood pressure, or nerve activity have shown that biomedical signals vary in a complex and irregular way, even during stable external conditions [1-3]. Considerable attention has been recently devoted to unifying various aspects of cardiac physiology using nonlinear dynamics, particularly through methods of topological analysis and fractal geometry [1, 4, 5]. However, accurate and quick diagnosis of the cardiovascular state of a patient on a base of an electrocardiogram (ECG) analysis is sometimes rather complicated problem in clinical practice, especially computer ECG interpretation and processing [6-8]. For improvement of methods of computer ECG processing, various approaches are used, including both nonlinear dynamics methods [4, 9, 10-11] and spectral-correlative ones [10]. In particular, in the present paper the dynamics of cardiac activity is investigated by means of exploring topological properties of the signal attractor in a phase space with dimension m.

It is worth noting that the computational complexity of fractal-topological algorithms usually being applied for cardiovascular dynamics investigation makes

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 115–127, 2008.

these algorithms rather cumbersome from standpoint of their computer time and the quantity of required experimental data N. Therefore, the total time of both observation and diagnosis process becomes rather long that may result in difficulties in clinical practice. So, in this paper we investigate application of the topological method based on temporal locality approach to investigation of phase trajectories. In comparison with the most conventional methods of fractal-topological analysis based on spatial localization on a chaotic system (CS) attractor (such as the Grassberger - Procaccia algorithm (GPA) [12-14], "nearest neighbor" method [15, 16], or the box-count algorithm [1, 17]), the developed method allows us to obtain reduction of N and computation time, as well as is insensible to growing m on these characteristics.

2 The Algorithm for Exploration of Topological Instability

In this section, we develop the algorithm for analysing the features of a CS leading to the multidimensional attractor in its time evolution (it occurs similarly to the Ruelle - Takens model of turbulence) [12, 18]. The efficiency of consideration of phase trajectories for the proposed algorithm results from reduction of required computer resources and experimental data. This is attained by elaborating a topological method based on temporal localization in relation to points of an attractor. The most conventional methods of fractal-topological analysis imply just the spatial localization, i.e. investigation of distribution of points on the attractor basin based on estimating the quantity of hits into the m-dimensional cell with size l [1, 5, 14]. Those have rather large computational complexity that increases for multidimensional cases. For instance, the computational complexity increases exponentially for the box-count algorithm [17] and almost linearly for the GPA [13, 14] with growth of m at expense of growing a number of computational operations. In contrast to these methods, we show that temporal localization provides more convenient realization of topological analysis with essential reduction of required experimental data and computation time and makes both these characteristics practically independent on dimensionality within some restricted range of changing m.

Let us consider a CS whose behavior is described by a system of d nonlinear differential equations with d kinetic variables. The method of delayed coordinates (affirmed mathematically by Takens [19]) for reconstruction of phase trajectories forming an attractor R_T^m (using one variable only $\eta(t)$) is given by [5, 15-16, 19]

$$\boldsymbol{x}_i^{(m)} = (\eta_i, \eta_{i+p}, ..., \eta_{i+(m-1)p}), \tag{1}$$

where $\eta(i\Delta t) = \eta_i$, $i = 1, 2, ..., N$ is a time series (TS) of a kinetic variable measured from the CS with a fixed time interval Δt, $\tau = p\Delta t$ is the delay time, p is an integer. The points $\boldsymbol{x}_i^{(m)} \subset R^m$, R^m is an Euclidean phase space with a dimension m, $i = 1, 2, ..., L^{(p,m)}$, the common quantity of the attractor points is given by $L^{(p,m)} = N - p(m-1)$. In accordance with (1), phase trajectories

forming the attractor R_T^m can be represented as a superposition of p rarefied sequences $\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_p$ shifted by one sample with respect to each other, those are defined as $\mathrm{X}_s = \{\boldsymbol{x}_{s+p(k-1)}^{(m)}\}_{k=1}^{L_s^{(p,m)}}$. These sequences are formed by p rarified TS $\Psi_1, \Psi_2, \ldots, \Psi_p$ obtained from the initial one, where $\Psi_s = \{\eta_{s+p(k-1)}\}_{k=1}^{N_s^{(p)}}$, $L_s^{(p,m)} = N_s^{(p)} - m + 1$.

Since some samples of TS are repeated within separate X_s sequences at direct representation of R_T^m by (1), we construct a compact representation $A^{(m)} = X_0^{(m)} \cup \Re^{(m)} \cup D^{(m)}$ of R_T^m excluding such redundancy using temporal locality properties. In such a case, the following notations are being introduced: (i) a set of initial points $X_0^{(m)} = \{\boldsymbol{x}_1^{(m)}, \boldsymbol{x}_2^{(m)}, \ldots, \boldsymbol{x}_p^{(m)}\}$, (ii) a set $\Re^{(m)}$ of distances between points with nearest indices within all X_s, namely $\Re^{(m)} = \{\Re_s^{(m)}\}_{s=1}^p$, where $\Re_s^{(m)} = \{r_{(s+p(k-1))\uparrow}^{(m)}\}_{k=1}^{L_s^{(p,m)}-1}$, and (iii) a set of orientation sequences $D^{(m)} = \{D_s^{(m)}\}_{s=1}^p$, where $D_s^{(m)} = \{d_{(s+p(k-1))}^{(m)}\}_{k=1}^{L_s^{(p,m)}-1}$. Distances $r_{k\uparrow}^{(m)}$ are the ones between points $\boldsymbol{x}_k^{(m)}$ and $\boldsymbol{x}_{k+p}^{(m)}$, these points are separated by the minimal temporal interval τ in a sequence of points within some X_s. Those are given by

$$r_{k\uparrow}^{(m)} = \left[\sum_{j=1}^{m} \Delta\eta_{k+(j-1)p}^2\right]^{\frac{1}{2}}, \tag{2}$$

where $\Delta\eta_k = \eta_{k+p} - \eta_k$. Terms of the orientation sequence allowing elimination of ambiguity at successive determination of phase trajectories are defined as follows

$$d_k^{(m)} = sgn(\Delta\eta_{k+(m-1)p}) \tag{3}$$

where it is supposed that $sgn(0) = 1$. From (1) - (3) one can show the uniqueness of such representation, i.e. that the map $G_T : X^{(m)} \to A^{(m)}$ and the inverse one $G_T^{-1} : A^{(m)} \to X^{(m)}$ exist and are determined by only one way, where $X^{(m)} = \{\boldsymbol{x}_k^{(m)}\}_{k=1}^{L^{(p,m)}}$.

But the set $A^{(m)}$ is not invariant to the linear transformations of TS when the topological structure of R_T^m does not change (at uniform shift of R_T^m points and uniform stretching (narrowing) of distances on R_T^m). So, for characterization of the topological structure we derive from $A^{(m)}$ the measure $S^{(m)} = B^{(m)} \cup D^{(m)}$, where $B^{(m)} = \{B_s^{(m)}\}_{s=1}^p$, and $B_s^{(m)} = \{\beta_{(s+p(k-1))}^{(m)}\}_{k=1}^{L_s^{(p,m)}-2}$ represents a sequence of relative distances

$$\beta_k^{(m)} = \frac{r_{(k+p)\uparrow}^{(m)}}{r_{k\uparrow}^{(m)}}. \tag{4}$$

The set $S^{(m)}$ offered as a characteristic of the topological structure does not change at such transformations of R_T^m, which are characterized by (i) uniform change of distances on R_T^m or (ii) uniform shift of its points. On the other hand, under determination of additional conditions of absence of such transformations,

this set is sufficient for the complete determination ofR_T^m. It can be shown similarly [20].

On a basis of $S^{(m)}$ consideration, let us investigate the dynamics of the topological structure of R_T^m when increasing m. It reflects the features of temporal behavior of a CS under investigation and again allows obtaining characteristics to be needed for optimal reconstruction of phase trajectories. A proper choice of a minimal embedding dimension m_0 allows both computational complexity and the necessary amount of experimental data to be reduced for respective algorithms of the fractal-topological analysis [15, 21]. Besides, m_0 characterizes a number of CS freedom degrees and defines minimal quantity of differential equations, required for modeling the CS under investigation [12]. Takens obtained the following estimate for dimensionality: $m \geq 2d + 1$ [19], while in a series of subsequent works it was shown (mostly by means of numerical simulations) [15, 16] that $m < 2d + 1$ can be good enough for reconstruction of phase trajectories.

At $R^m \to R^{m+1}$, the shift of every sample of the orientation sequence towards a point with a nearest index within a certain X_s takes place, i.e.

$$d_l^{(m+1)} = d_{l+p}^{(m)}, \tag{5}$$

where $l = 1, 2, ..., L^{(p,\, m+1)} - p$. It follows from the expression (3), determining the orientation sequence. From (5) one can see that $D^{(m)}$ reflects only linear changes of $S^{(m)}$. So, for estimating nonlinear changes in R_T^m structure at $R^m \to R^{m+1}$, explore changes in $B^{(m)}$, therefore introduce a sequence of ratios

$$\gamma_j^{(m\uparrow)} = \frac{\beta_j^{(m)}}{\beta_j^{(m+1)}}, \tag{6}$$

where $j = 1, 2, ..., L^{(p,m+1)} - 2p$. In a case of completely uniform topological dynamics (that is called the ideal topological stabilization (ITS), similarly to [20]), the terms $\beta_k^{(m)}$ have no changes at $R^m \to R^{m+1}$ and $\gamma_j^{(m\uparrow)} \equiv 1$ on such condition.

Rewrite (6) in the form

$$\gamma_j^{(m\uparrow)} = \sqrt{\frac{1 + \mu_j^{(m)}}{1 + \mu_{j+p}^{(m)}}}, \tag{7}$$

where the relative partition sequence (RPS) $\{\mu_j^{(m)}\}$ is constructed by means of segmentation of difference-quadratic TS $\Delta\eta_j^2$ and is defined as follows:

$$\mu_j^{(m)} = \frac{\Delta\eta_{j+mp}^2}{\tilde{\sigma}_j^{(m)}}, \tag{8}$$

where partition sums are $\tilde{\sigma}_j^{(m)} = \sum_{i=0}^{m-1} \Delta\eta_{j+ip}^2$ and $j = 1, 2, ..., L^{(p,m+1)} - p$. Using $\{\mu_j^{(m)}\}$ for characterization of topological dynamics permits higher accuracy of

m_0 determination as it is shown by means of numerical simulations. Analogously [20], introduce the following measure of topological instability:

$$Z_{\mu}^{\Sigma}(m) = \sigma(\mu_j^{(m)}), \tag{9}$$

where $\sigma(\mu_j^{(m)})$ is the mean square variance (MSV):
$\sigma(\mu_j^{(m)}) = \sqrt{\langle\left(\mu_j^{(m)} - \langle\mu_j^{(m)}\rangle\right)^2\rangle}$, the averaging is made over R_T^m points. In a case of ITS, supposing $\gamma_j^{(m\uparrow)} \equiv 1$, one can obtain from (7) that $\mu_j^{(m)} \equiv \mu^{(m)}$, where $\mu^{(m)}$ is constant $\forall j$. On the other hand, any local change in $\{\mu_j^{(m)}\}$ sequence entails corresponding change in $\{\gamma_j^{(m\uparrow)}\}$. The measure $Z_{\mu}^{\Sigma}(m)$ declines with enlarging m (for $\mu_j^{(m)} \to 0$ at $m \to \infty$), and near the point $m = m_0$ the dependence $Z_{\mu}^{\Sigma}(m)$ is expected to display a sharp decrease because of topological stability appearance, this fact is shown by means of computer experiments and is used for m_0 determination. But because of monotonic decrease of $\mu_j^{(m)}$, even for $m > m_0$, we cannot estimate asymptotic deviation $Z_{\mu}^{\Sigma}(m)$ from ITS without considering inevitable reduction of $\mu_j^{(m)}$. So, we introduce the following normalization and obtain the normalized instability function:

$$\breve{Z}_{\mu}^{\Sigma}(m) = \frac{Z_{\mu}^{\Sigma}(m)}{\left\langle\mu_j^{(m)}\right\rangle}. \tag{10}$$

Taking into account that $Z_{\mu}^{\Sigma}(m) = \sqrt{\left\langle(\mu_j^{(m)})^2\right\rangle - \left\langle\mu_j^{(m)}\right\rangle^2}$, one can rewrite (10) in the form

$$\breve{Z}_{\mu}^{\Sigma}(m) = \left(\frac{\left\langle(\mu_j^{(m)})^2\right\rangle}{\left\langle\mu_j^{(m)}\right\rangle^2} - 1\right)^{\frac{1}{2}}. \tag{11}$$

As it was recently shown, rarefying on attractor points is reasonable for numerical simulation of fractal-topological analysis [16]. Otherwise, using points that are too close together in time leads to essential underestimates of the dimension, i.e. to aggravating accuracy of the topological analysis. So, we also implement temporal rarifying of phase trajectories for creating a subset of points with reduced correlative relations in the embedding space. It is attained by the approach that is realized in the most convenient way, namely we use only one X_s for numerical experiments.

For this reason, in the present work the numerical simulations are made using the sequence $\Psi_p = \{\eta_p,\ \eta_{2\cdot p},\ \ldots, \eta_{N_p^{(p)}\cdot p}\}$, and rarifying is determined as p=2 in this work, where the length of rarefied TS $N_p^{(p)} \approx \left[\frac{N}{p}\right]_{int}$, $[.]_{int}$ is an integer part of a number. Denoting its components for brevity as $\Psi_p = \{\xi_1, \xi_2,\ \ldots, \xi_{N_p^{(p)}}\}$, we

obtain that terms of reduced RPS $\{\hat{\mu}_j^{(m)}\}$ constructed by means of segmentation of $\{\Delta\xi_{j+i}^2\}$ TS are defined analogously (8) as follows

$$\hat{\mu}_j^{(m)} = \frac{\Delta\xi_{j+m}^2}{\hat{\sigma}_j^{(m)}}, \tag{12}$$

where $\hat{\sigma}_j^{(m)} = \sum_{i=0}^{m-1} \Delta\xi_{j+i}^2$, $\Delta\xi_j = \xi_{j+1} - \xi_j$. In turn, the topological instability function for rarified TS is defined similarly to (9)

$$Z_\mu(m) = \sigma(\hat{\mu}_j^{(m)}), \tag{13}$$

while the normalized instability function reflecting relative variance on $\{\hat{\mu}_j^{(m)}\}$ is calculated analogously (11) as follows:

$$\breve{Z}_\mu(m) = \left(\frac{\left\langle (\hat{\mu}_j^{(m)})^2 \right\rangle}{\left\langle \hat{\mu}_j^{(m)} \right\rangle^2} - 1 \right)^{\frac{1}{2}}. \tag{14}$$

As follows from (14), $\breve{Z}_\mu(m) \equiv 0$ in a case of ITS, when uniform change of distances takes place at enlarging dimensionality. At the same time, normalization in a form of (10) reflects statistical indeterminacy at enlarging $m$because both numerator and denominator in (10) tend to zero at $m \to \infty$. Nevertheless, in this paper we prove numerically that this dependence provides useful information in a case of ECG signal processing.

3 Numerical Simulations with ECG Signal

In this work the digitized ECG TS ς_i containing N=2500 points is used for calculation of topological curves, time interval of discretization $\Delta t = 0.002$ s. The obtained TS are those of adult healthy subjects, four samples containing first two thousand of discrete levels ς_i of measured TS are shown in Fig. 1. We explore two groups of databases: the first group is obtained from subjects in a state of rest (most typical examples are shown in Fig 1, (a) and (b), pulse rate for ECG pertaining to this group is 58-70), and the second group, where ECG TS are measured from subjects which undergo very intensive physical exercises (see Fig 1, (c) and (d), pulse rate for such group is 90-100 typically and even 110-114 for some cases). For investigation of cardiovascular dynamics at relatively long time ranges (for providing a high reliability), we take more than seven QRS-complexes, though the convergence of the locally topological approach can be attained using only three ones [20]. For decreasing linear autocorrelation effect and reduction of influence of low-frequency periodical component, we use difference TS $\eta_i = \varsigma_i - \varsigma_{i+1}$ instead of "raw" digitized ECG signal ς_i (as well as in [5]). The phase trajectories for three-dimensional attractors reconstructed

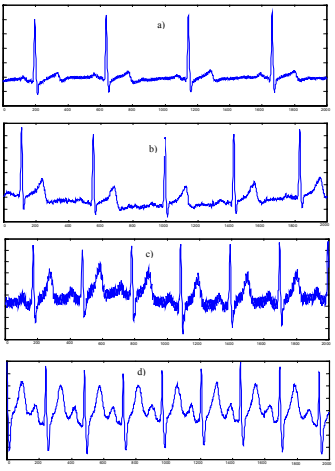


Fig. 1. The initial ECG signals: a) and b) reflect slow cardiac processes in a state of rest; c) and d) are obtained under physical exercises

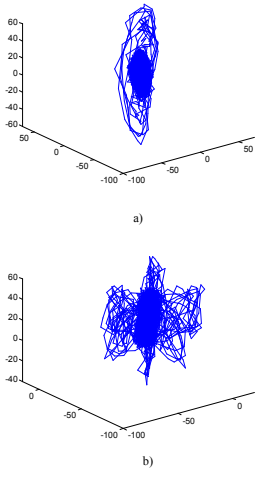


Fig. 2. Temporal evolution of cardiac activity represented through the phase trajectories reconstructed from the difference ECG TS: a) $p = 1$; b) $p = 20$

from η_i by (1) at $p = 1$ and $p = 20$ respectively are shown in Fig. 2, a) and b), the latter case reveals the effect of partial decorrelation (similarly [5]).

For investigation of relationships between features of cardiac activity and topological dependencies, we calculated topological curves using ECG taken out of twenty separate databases, half of those is obtained from subjects in a state of rest, while half is measured under physical exercises (as it was mentioned above). The ECG TS forming these databases were obtained from healthy adult

subjects at different time within about three months. At calculation of topological dependencies, the following additional normalization is used for scale unification:

$$Y(m) = \frac{Z_\mu(m)}{Z_\mu(1)}; \tag{15}$$

$$\breve{Y}(m) = \frac{\breve{Z}_\mu(m)}{\breve{Z}_\mu(1)}. \tag{16}$$

The functions $Z_\mu(m)$, $\breve{Z}_\mu(m)$ are calculated by (13), (14) using expressions (12), averaging in (13) - (14) being implemented as follows $\left\langle \hat{\mu}_j^{(m)} \right\rangle = \frac{1}{\Delta N} \sum_{j=i_0}^{N_\mu} \hat{\mu}_j^{(m)}$, where $\Delta N = N_\mu - i_0 + 1$, i_0 is chosen so that to avoid the initial points of ECG TS where transitive processes may occur, in this paper $i_0 = 5$. In turn, N_μ is determined according to above couched considerations (see (1)), i.e.

$$N_\mu = N_p^{(p)} - m - 2. \tag{17}$$

We calculated more than thirty dependencies (15), (16) altogether for both above mentioned groups, and most typical results are shown in Figs. 3-4. These dependencies display sufficient convergence for $Y(m)$ at $m \geq m_0$, i.e. m_0 is just the value of the dimension that provides preservation of topological structure of phase trajectories at enlarging dimensionality beyond m_0 and thus can be really considered as the minimal embedding dimension of the attractor. The convergence of $Y(m)$ is shown to be of the same character, almost independently on individual features of subjects, and one can conclude that $m_0 = 5$ is sufficient for optimal embedding of the attractor into Takens' phase space for ECG derived at the state of rest (under physical activity this value increases to 7, as one can conclude from Fig. 4). These values are in a good coincidence with those obtained in [5] using GPA. On the other hand, the dependence $\breve{Y}(m)$ is approximately linear at $m \geq m_0$, but it differs for various persons with respect to average level of convergence and slant angle. Evidently, it can provide some additional information concerning individual features of subjects that seems to be useful for the sake of early diagnosis.

One can conclude from Figs 3-4 that any essential change of ECG form (appearance and increase of lateral peaks, as well as pulse rate deviation) leads to related change of topological curves (for instance, increase of cardiac activity at physical exercises entails the growth of m_0, as mentioned above). Again, at processing of "slow" ECG we noticed that the growth of S-peak leads to lowering the main peaks for both $Y(m)$ and $\breve{Y}(m)$. Such typical effect is shown in Fig. 3, a) and b) (see the line with circle markers). For "fast" ECG, the increase of slant angle for $\breve{Y}(m)$ is evident (at the region of convergence for $Y(m)$, see Fig.4) in comparison with that for "slow" ones. This allows us to suppose that the extent of cardiac activity has influence on this characteristic, especially it is explicit

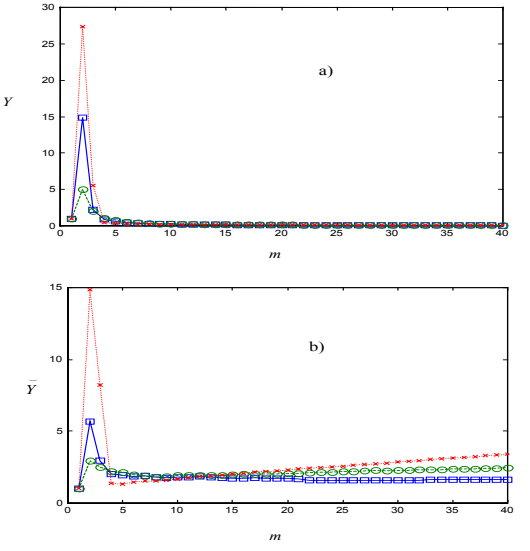


Fig. 3. The topological dependencies calculated using ECG TS obtained from slow cardiac processes in a state of rest; lines with square, circle, and x-markers correspond to ECG of different subjects taken out of separate databases, namely x-markers line is obtained from ECG shown in Fig.1 (a), circles markers line - from (b)

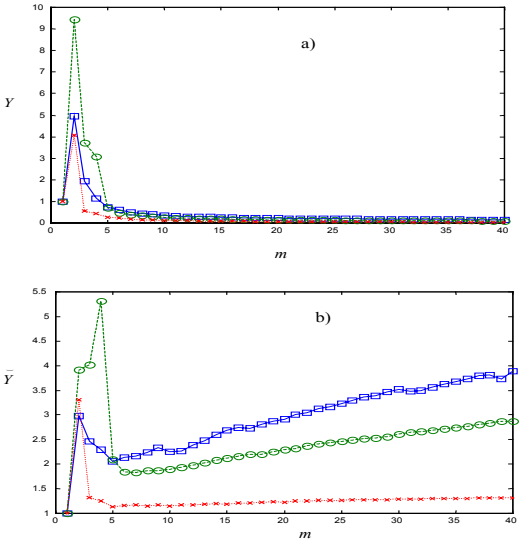


Fig. 4. The topological dependencies calculated using ECG TS obtained under physical exercises (with enlarged puls rate); lines with square, circle, and x-markers correspond to different subjects, namely x-markers line is obtained from ECG shown in Fig.1 (c), square markers line - from (d)

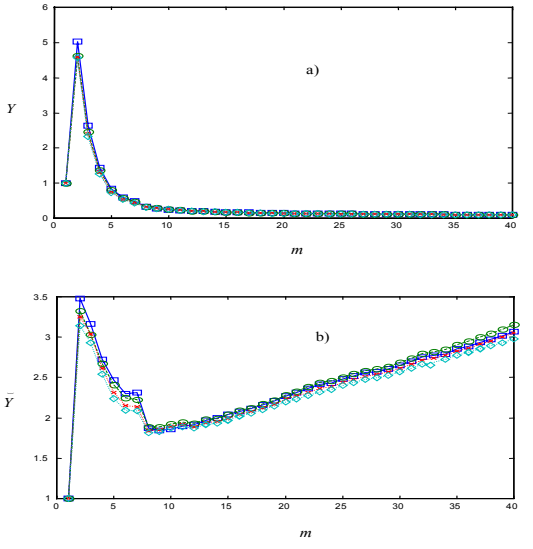


Fig. 5. The topological dependencies calculated at changing length of ECG TS measured from the same subject: lines with square markers correspond to $N = 2000$; lines with circle markers - $N = 2500$; lines with x-markers - $N = 3000$; and lines with diamonds - $N = 3500$

in a case of large loading (see ECG in Fig. 1, d) and corresponding curves in Fig. 4, b)). The mentioned circumstances display relationship between some features of cardiac activity and topological dependencies.

Calculation of $\breve{Y}(m)$ has also allowed us to detect one "abnormal" case with monotonic decline of $\breve{Y}(m)$ at $m > m_0$. This case is shown in Fig. 3, the line with square markers. The subject with such ECG has no any cardiac pathology, nevertheless he delivered some complaints of weakness and refused from any physical exercises on that day.

We also calculated the curves for different length N of TS (see Fig. 5). The obtained curves show that changing N from 2000 to 3500 does not result in essential change of the form of topological dependencies, and $N = 2500$ used in this work is enough for obtaining reliable results.

4 Evaluation of Obtained Results and Discussions

The obtained results of embedding determination on $Y(m)$ are in a good coincidence with results obtained in [5] by GPA, where slopes of the plots ln $C(l)$ versus ln l appear to be same for $m \geq 6$. It is worth noting that in GPA method $N = 16000$ [5], while in our experiments N=2500 (moreover, the length of really used rarefied TS $N_p^{(p)} = 1250$) that means a significant reduction of required experimental data.

For the algorithm developed in this paper, required quantity of experimental data N is not only less than for spatially-localized methods [5, 21], but is constant $\forall \quad m \in [1,\ m_{\max}]$. At the same time, when using the GPA N increases exponentially with growth of m (see [13] and references therein).

Computation time of the proposed algorithm t_{com} does not depend on m within a certain segment $[1,\ m_{\max}]$ due to the following recurrent relationship with respect to $\hat{\sigma}_j^{(m)}$ calculation derived from (12)

$$\hat{\sigma}_j^{(m+1)} = \hat{\sigma}_j^{(m)} + \Delta\xi_{j+m}^2 \tag{18}$$

In the developed algorithm, the reduction of t_{com} is achieved due to minimal quantity of computation operations required for topological dynamics analysis. It takes place because the proposed analytical model of temporal localization implies the use of minimal quantity of neighboring points, and no additional operations are required for detection of those. Again, functional relationships for estimating the topological nonuniformity (including averaging) are also very convenient for computer calculations. Additional reduction of t_{com} results from shortening the used TS as is mentioned above.

We have calculated the topological dependencies using a computer with relatively low clock rate of the processor (about 150 MHz). Nevertheless, computation time for calculation of one dependence $Y(m)$ or $\breve{Y}(m)$ is only t_{com}=36 s. For comparison, we calculated the dependence of correlation integral $C(l)$ by GPA, in such a case t_{com}^{GPA}=27 min (it is worth noting that for reflection of topological dynamics approximately ten such dependencies $C(l)$ at various m must be calculated). The improved scheme of calculation and comparative sorting of distances on R_T^m based on locality properties of (1) (this scheme is developed in [22]) is applied at the calculations with GPA.

Thus, the convergence of the proposed algorithm of temporal localization for the ECG TS attractor is shown, that can be used for purposes of diagnosis because the m_0 value allows an estimate of the freedom degrees of ECG signal and complexity degree of myocardium dynamics.

5 Conclusions

In this paper, we implement the estimation of characteristics of ECG signal attractor for several different databases by means of the topological method based on temporal localization. The computer experiments showed good convergence for the asymptotic measure of topological instability independently on personal features of subjects (except a factor of exercises, when slight shift of convergence region is possible), in spite of presence of noise. This allows estimation of freedom degree of cardiac dynamics. On the other hand, that instability function normalized by average displays different kind of behavior that differs for various persons with respect to the average level of convergence and a slant angle. It provides some additional information concerning individual features of subjects that seems to be useful for the sake of early diagnosis. From topological standpoint,

one can conclude that such different behavior of $\breve{Z}_\mu(m)$ describes deviation of the structure of ECG attractor from that of a fractal manifold obtained from a model TS, because convergence of $\breve{Z}_\mu(m)$ is quite uniform for application of the algorithm to an attractor reconstructed from chaotic TS obtained from delay differential equations (as it is shown in [20]), such type of TS represents just the model ones reflecting the multidimensional behavior of a CS.

These results provide information that reflects temporal evolution of cardiac activity. Numerical simulations proved good accuracy of determination of topological characteristics. These results are obtained with essential reduction of required experimental data (about by an order, taking into account the rarefying stage). The reduction of computation time at the expense of minimization of computational operations is also attained.

In conclusion, the proposed algorithm being pertained to those based on nonlinear dynamics methods provides information that can not be obtained with traditional spectral-correlative approaches (including, first of all, spectral analysis, autocorrelation function calculation, and linear regression methods). As it has been shown in this paper, consideration of topological dynamics of the attractor embedding can be reduced to nonlinear analysis of ECG TS. At the same time, the above couched algorithm has some similar features with respect to common statistical methods as it deals with averaging of strings derived from digital information (measured in experiments or obtained from proper differential equations describing nonlinear phenomena observed in a CS under investigation).

References

1. Nonnenmacher, T.F., Losa, G.A., Wribel, E.R.: Fractals in Biology and Medicine. Birkhäuser, Basel (1994)
2. Peng, C.-K., et al.: Quantification of scaling exponents and crossover phenomena in nonstationary heartbeat time series. Chaos 5, 82–87 (1995)
3. Gonzalez, H., et al.: Resetting and annihilation reentrant waves in a ring cardiac tissue: theory and experiment. Progr. Theor. Phys. Suppl. 139, 83–89 (2000)
4. Wang, J., Ning, X., Chen, Y.: Multifractal analysis of electronic cardiogram taken from healthy and unhealthy adult subjects. Physica A 323, 561–568 (2003)
5. Bezerianos, A., Bountis, T., Papaioannou, G., Polydoropoulos, P.: Nonlinear time series analysis of electrocardiograms. Chaos 5, 95–101 (1995)
6. Guglin, M.E., Thatai, D.: Common errors in computer electrocardiogram interpretation. Int J Cardiol 106, 232–237 (2006)
7. Iliev, I., Krasteva, V., Tabakov, S.: Real-time detection of pathological cardiac events in the electrocardiogram. Physiol. Meas. 28, 259–276 (2007)
8. Shi, Z., Zhanga, C.: Semi-blind source extraction for fetal electrocardiogram extraction by combining non-Gaussianity and time-correlation. Neurocomputing 70, 1574–1581 (2007)
9. Perc, M.: Nonlinear time series analysis of the human electrocardiogram. Eur. J. Phys. 26, 757–768 (2005)
10. Jagric, T., et al.: Irregularity test for very short electrocardiogram (ECG) signals as a method for predicting a successful defibrillation in patients with ventricular fibrillation. Transl. Res.: J. Lab. and Clin. Medicine 149, 145–151 (2007)

11. Gutierrez, R.M., Sandoval, L.A.: A method to separate stochastic and deterministic information from electrocardiograms. Phys. Scr. T118, 132–135 (2005)
12. Schuster, H.G.: Deterministic Chaos, 2nd edn. Wiley-VCH, Weinheim (1987)
13. Abarbanel, H.D.I.: The analysis of observed chaotic data in physical systems. Rev. Mod. Phys. 65, 1331–1395 (1993)
14. Grassberger, P., Procaccia, I.: Characterization of strange attractors. Phys. Rev. Lett. 50, 346–349 (1983)
15. Liebert, W., Pawelzik, K., Schuster, H.G.: Optimal embedding of chaotic attractor from topological consideration. Europhys. Lett. 14, 521–526 (1991)
16. Albano, A.M., Passamante, A., Farrell, M.E.: Using higher-order correlations to define an embedding window. Physica D 54, 85–97 (1991)
17. Bershadskii, A.: Multifractal Bernoulli fluctuations in disordered mesoscopic systems. J. Phys. A: Math. Gen. 31, 707–711 (1998)
18. Guckenheimer, J., Holmes, P.: Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields. Springer, Berlin (1983)
19. Takens, F.: Detecting strange attractors in turbulence. In: Dynamical Systems and Turbulence. Lecture Notes in Math., vol. 898, pp. 366–381. Springer, Berlin (1981)
20. Dailyudenko, V.F.: Nonlinear time series processing by means of ideal topological stabilization analysis and scaling properties investigation. In: Priddy, K.L., Keler, P.E., Fogel, D.B., Bezdek, J.C. (eds.) Applications and Science of Computational Intelligence II. SPIE's Proc., vol. 3722, pp. 108–119 (1999)
21. Casdagli, M.: Nonlinear prediction of chaotic time series. Physica 35D, 335–356 (1989)
22. Dailyudenko, V.F.: Reduced fractal analysis of the multidimensional attractor reconstructed from chaotic time series. In: Kumar, V., Gavrilova, M.L., Tan, C.J.K., L'Ecuyer, P. (eds.) ICCSA 2003. LNCS, vol. 2667, pp. 921–926. Springer, Heidelberg (2003)

Missing Value Imputation Based on Data Clustering*

Shichao Zhang[1], Jilian Zhang [2], Xiaofeng Zhu[1], Yongsong Qin[1], and Chengqi Zhang[3]

[1] Department of Computer Science, Guangxi Normal University, Guilin, China
[2] School of Information Systems, Singapore Management University, Singapore
[3] Faculty of Information Technology, University of Technology Sydney
P.O. Box 123, Broadway, NSW2007, Australia
zhangsc@mailbox.gxnu.edu.cn, zhangjilian@yeah.net,
xfzhu_dm@163.com, ysqin@mailbox.gxnu.edu.cn,
chengqi@it.uts.edu.au

Abstract. We propose an efficient nonparametric missing value imputation method based on clustering, called CMI (Clustering-based Missing value Imputation), for dealing with missing values in target attributes. In our approach, we impute the missing values of an instance *A* with plausible values that are generated from the data in the instances which do not contain missing values and are most similar to the instance *A* using a kernel-based method. Specifically, we first divide the dataset (including the instances with missing values) into clusters. Next, missing values of an instance *A* are patched up with the plausible values generated from *A*'s cluster. Extensive experiments show the effectiveness of the proposed method in missing value imputation task.

1 Introduction

Missing values imputation is an actual yet challenging issue confronted in machine learning and data mining [1, 2]. Missing values may generate bias and affect the quality of the supervised learning process or the performance of classification algorithms [3, 4]. However, most learning algorithms are not well adapted to some application domains due to the difficulty with missing values (for example, Web applications) as most existed algorithms are designed under the assumption that there are no missing values in datasets. That implies that a reliable method for dealing with those missing values is necessary. Generally, dealing with missing values means to find an approach that can fill them and maintain (or approximate as closely as possible) the original distribution of the data. For example, in a database, if the known values for an attribute *A* are: 2 in 60% of cases, 6 in 20% of cases and 10 in 10% of

* This work is partially supported by Australian Large ARC grants (DP0559536 and DP0667060), China NSF major research Program (60496327), China NSF grant for Distinguished Young Scholars (60625204), China NSF grants (60463003, 10661003), an Overseas Outstanding Talent Research Program of Chinese Academy of Sciences (06S3011S01), an Overseas-Returning High-level Talent Research Program of China Ministry of Personnel, China NSF grant for Distinguished Young Scholars (60625204), a Guangxi NSF grant, and an Innovation Project of Guangxi Graduate Education (2006106020812M35).

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 128–138, 2008.

cases, it is reasonable to expect that missing values of *A* will be filled with 2 (if *A* is discrete) or 3.4 (if *A* is continuous) (see [5]).

Missing values may appear either in conditional attributes or in class attribute (target attribute). There are many approaches to deal with missing values described in [6], for instance: (a) Ignore objects containing missing values; (b) Fill the missing value manually; (c) Substitute the missing values by a global constant or the mean of the objects; (d) Get the most probable value to fill in the missing values. The first approach usually lost too much useful information, whereas the second one is time-consuming and expensive in cost, so it is infeasible in many applications. The third approach assumes that all missing values are with the same value, probably leading to considerable distortions in data distribution. However, Han et al. 2000, Zhang et al. 2005 in [2, 6] think: 'The method of imputation, however, is a popular strategy. In comparison to other methods, it uses as more information as possible from the observed data to predict missing values.

Traditional missing value imputation techniques can be roughly classified into parametric imputation (e.g., the linear regression) and non-parametric imputation (e.g., non-parametric kernel-based regression method [20, 21, 22], Nearest Neighbor method [4, 6] (referred to as NN)). The parametric regression imputation is superior if a dataset can be adequately modeled parametrically, or if users can correctly specify the parametric forms for the dataset. For instance, the linear regression methods usually can treat well the continuous target attribute, which is a linear combination of the conditional attributes. However, when we don't know the actual relation between the conditional attributes and the target attribute, the performance of the linear regression for imputing missing values is very poor. In real application, if the model is misspecified (in fact, it is usually impossible for us to know the distribution of the real dataset), the estimations of parametric method may be highly biased and the optimal control factor settings may be miscalculated.

Non-parametric imputation algorithm, which can provide superior fit by capturing structure in the dataset (note that a misspecified parametric model cannot), offers a nice alternative if users have no idea on the actual distribution of a dataset. For example, the NN method is regarded as one of non-parametric techniques used to compensate for missing values in sample surveys [7]. And it has been successfully used in, for instance, U.S. Census Bureau and Canadian Census Bureau. What's more, using a non-parametric algorithm is beneficial when the form of relationship between the conditional attributes and the target attribute is not known a-priori [8].

While nonparametric imputation method is of low-efficiency, the popular NN method faces two issues: (1) each instance with missing values requires the calculation of the distances from it to all other instances in a dataset; and (2) there are only a few random chances for selecting the nearest neighbor. This paper addresses the above issues by proposing a clustering-based non-parametric regression method for dealing with the problem of missing value in target attribute (named Clustering-based Missing value Imputation, denoted as CMI). In our approach, we fill up the missing values with plausible values that are generated by using a kernel-based method. Specifically, we first divide the dataset (including instances with missing values) into clusters. Then each instance with missing-values is assigned to a cluster most similar to it. Finally, missing values of an instance *A* are patched up with the plausible values generated from *A*'s cluster.

The rest of the paper is organized as follows. In section 2, we give related work on missing values imputation. Section 3 presents our method in detail. Extensive experiments are given in Section 4. Conclusions and future work are presented in Section 5.

2 Related Work

In recent years, many researchers focused on the topic of imputing missing values. Chen and Chen [9] presented an estimating null value method, where a fuzzy similarity matrix is used to represent fuzzy relations, and the method is used to deal with one missing value in an attribute. Chen and Huang [10] constructed a genetic algorithm to impute in relational database systems. The machine learning methods also include auto associative neural network, decision tree imputation, and so on. All of these are pre-replacing methods. Embedded methods include case-wise deletion, lazy decision tree, dynamic path generation and some popular methods such as C4.5 and CART. But, these methods are not a completely satisfactory way to handle missing value problems. First, these methods only are designed to deal with the discrete values and the continuous ones are discretized before imputing the missing value, which may lose the true characteristic during the converting process from the continuous value to discretized one. Secondly, these methods usually studied the problem of missing covariates (conditional attributes).

Among missing value imputation methods that we consider in this work, there are also many existing statistical methods. Statistics-based methods include linear regression, replacement under same standard deviation, and mean-mode method. But these methods are not completely satisfactory ways to handle missing value problems. Magnani [11] has reviewed the main missing data techniques (MDTs), and revealed that statistical methods have been mainly developed to manage survey data and proved to be very effective in many situations. However, the main problem of these techniques is the need of strong model assumptions. Other missing data imputation methods include a new family of reconstruction problems for multiple images from minimal data [12], a method for handling inapplicable and unknown missing data [13], different substitution methods for replacement of missing data values [14], robust Bayesian estimator [15], and nonparametric kernel classification rules derived from incomplete (missing) data [16]. Same as the methods in machine learning, the statistical methods, which handle continuous missing values with missing in class label are very efficient, are not good at handling discrete value with missing in conditional attributes.

3 Our Algorithm

3.1 Clustering Process Strategy

The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering. In this paper, we use a clustering technique, such as, K-Means [17] to group the instances of the whole dataset (denoted as *S*). We separate the whole database *S* into clusters each of which contains similar instances. When *S*

has more than one discrete attribute, we then use the simple matching method to compute the similarities of these discrete attributes and use the Euclidean distance to process the continuous attributes. Then the distance between instance and cluster center is a mixed one, which is a combination distances of the discrete and continuous attributes based on [6].

Our motivation in this paper is based on the assumption [6] that the instance with missing values is more likely to have the similar target attribute value as the instance that is closest to it based on the distance's principle, such as, the Euclidean distance. So we adopt the clustering method on the whole dataset in order to separate the instances into clusters based on the differences of their distances. Then the non-parametric method is utilized to deal with missing values for each cluster. Note that K=1 is a special case of K-Means method, it is the situation without clustering, that is to say, it is only a simple kernel-based imputation method while the number of clusters is 1 in our CMI algorithm. Our goal in this paper is to show the effectiveness of our method than the kernel function without clustering. Given K=1 and K>1, we can compare the performance of this non-parametric method with and without clustering the dataset. We adopt the well-known K-Means as clustering algorithm mainly for its simplicity and efficiency. As an alternative, one can choose a more powerful clustering technique for this task, for example, the G-means algorithm [19] that can determine the parameter K automatically for the clustering task.

3.2 Kernel Function Imputation Strategy

Kernel function imputation is an effective method to deal with missing values, for its computationally efficient, robust and stable [20]. In the statistical area, kernel function completion is also known as kernel nonparametric regression imputation. For instance, Zhang [20] uses the kernel method to impute missing values. In this paper, a kernel function nonparametric random imputation is proposed to make inference for the mean, variance and the distribution function (DF) of the data.

Let X be an $n \times d$-dimensional vector and let Y be a variable influenced by X, we denote X, Y as factor attributes (FA) (or conditional attributes) and target attribute (TA) respectively. We assume that X has no missing values, while only Y has. To simplify the discussion, the dataset is denoted as $(X_{i1}, X_{i2}, ..., X_{id}, Y_i, \delta_i), i = 1, ..., n$, where δ_i is an indictor function, i.e., $\delta_i = 0$ if Y_i is missing and $\delta_i = 1$ if Y_i is not missing. In a real world database, we suppose that X and Y satisfy:

$$Y_i = m(X_{i1}, X_{i2}, ..., X_{id}) + \varepsilon_i, i = 1, ..., n. \tag{1}$$

Where $m(X_{i1}, X_{i2}, ..., X_{id})$ is an unknown function, ε_i is a random error with mean 0 and variance σ^2. In other words, we assume that Y has relation with X, but we have not any idea about it. In the case of the unknown function $m(.)$ is a linear function, Wang and Rao [21, 22] show that the deterministic imputation method performance well in making inference for the mean of Y, Zhang [20] shows that one must use random imputation method in make inference for distribution functions of Y when the unknown function $m(.)$ is a arbitrary function because in many complex practical situations, the unknown function $m(.)$ is not a linear function.

In Eq.1, suppose Y_i is missing, and the value of $m(X_i)$=$m(X_{i1}, X_{i2},\ldots, X_{id})$ is computed by using kernel methods as follows:

$$\hat{m}(X_{i1}, X_{i2},..., X_{id}) = \frac{\sum_{j=1}^{n} \delta_j Y_j \prod_{s=1}^{d} K(\frac{X_{is} - X_{js}}{h})}{\sum_{j=1}^{n} \delta_j \prod_{s=1}^{d} K(\frac{X_{is} - X_{js}}{h}) + n^{-2}}, i = 1,...,n, \quad (2)$$

Where $\hat{m}(X)$ is the kernel estimate of the unknown function $m(X)$ and n^{-2} is introduced in order to avoid the case that $\sum_{j=1}^{n} \delta_j \prod_{s=1}^{d} K(\frac{X_{is} - X_{js}}{h})$ vanishes, and h refers to bandwidth with $h=Cn^{-1/5}$ (we will discuss the choosing of h later in this paper). The method of using $\hat{m}(X_i)$ as imputed value of Y_i is called *kernel imputation*.

In Eq.2, $K(.)$ is a kernel function. There are many commonly used forms of kernel functions, such as the Gaussian kernel:

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)$$

and the uniform kernel is presented as follows:

$$K(x) = \begin{cases} 1/2, & |x| \le 1, \\ 0, & |x|>1. \end{cases}$$

There are not any differences for selecting the kinds of the kernel function if the optimal bandwidth can be received during the process of learning. In this paper, we adopt the widely used Gaussian kernel function.

3.3 The Strategy for Evaluating Unknown Parameters of Imputed Data

We are interesting in make inferences for the parameters of the target attribute Y such as $\mu = E(Y)$, $\sigma^2 = D(Y)$ and $\theta = F(y_0)$, i.e. the mean, the variance and the distribution function of Y, where y_0 is a fixed point, $y_0 \in R$. Based on the complete data after imputation, above parameters can be estimated as follows.

The mean of Y is given by:

$$\hat{Y} = \frac{1}{n} \sum_{i=1}^{n} \{ \delta_i Y_i + (1 - \delta_i) Y_i^* \} \quad (3)$$

Where $Y_i^* = \hat{m}(X_i)$ if Y is completed by the kernel deterministic imputation method.

The variance of Y is given by:

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} [(\delta_i Y_i + (1 - \delta_i Y_i) Y_i^*) - \hat{Y}]^2 \quad (4)$$

In real applications, it is very difficult to work out the exact form of the distribution function of Y. So, we use the empirical form of the distribution function of Y replacing the values of the distribution function of Y:

$$\hat{F}(y_0) = \frac{1}{n}\sum_{i=1}^{n} I(\delta_i Y_i + (1-\delta_i)Y_i^* \le y_0) \tag{5}$$

where $I(x)$ is the indicator function, Y_i^* is the imputed data processed by the kernel imputation methods.

3.4 Clustering-Based Missing Value Imputation Algorithm

This section presents our CMI method for missing data completion. By using the clustering techniques on the factor attributes (i.e., X), we divide the whole dataset into clusters. After clustering, we then utilize the kernel method to fill the missing-valued instance for each cluster. Note that in this paper, the kernel method is utilized to deal with the situation that Y is continuous. As for the situation of Y is discrete, we can use the nearest neighbor method (NN) to complete the missing values. Based on the above discussions, the CMI algorithm is presented as follows.

```
Procedure: CMI
Input: Missing-valued dataset S, k;
Output: Complete dataset S';
1.  (C1, C2, ...Ck) ← k-means(S, k);
2.  FOR each cluster Ci
3.    FOR each missing-valued instance Ik in cluster Ci
4.       use Eq. (2) to comput m̂(Xi), R;
5.    FOR each missing-valued instance Ik in cluster Ci
6.         use m̂(Xi) to fill missing-value in Ik;
7.    S' ← ∪_{i=1}^{k} Ci ;
```

3.5 The Choosing of c and Complexity Analysis

Kernel methods can be decomposed into two parts: one for the calculation of the kernel and another for bandwidth choice. Silverman [23] stated that one important factor in reducing the computer time is the choice of a kernel that can be calculated very quickly. Having chosen a kernel that is efficient to compute, one must then choose the bandwidth. Silverman [23] turns out that the choice of bandwidth is much more important than the choice o f kernel function. Small value of bandwidth h makes the estimate look 'wiggly' and shows spurious features, whereas too big values of h will lead to an estimate that is too smooth, in the sense, that it is too biased and may not reveal structural features. There is no generally accepted method for choosing the bandwidths. Methods currently available include 'subjective choice' and automatic methods such as the "plug-in", 'cross-validation' (CV), and 'penalizing function' approaches. In this paper we use the method of cross-validation to minimize the

approximate mean integrated square error (AMISE) of $\hat{m}(x_i)$. For a given sample of data, the CV function is defined as:

$$CV = \sum_{i=1}^{n} (y_i - \hat{m}(x_{-i}, c))^2 \tag{6}$$

where $\hat{m}(x_{-i}, c)$ denotes the leave-one-out estimator evaluated for a particular value of c.

That is, the value of the missing attribute of instance i is predicted by all of the instances except instance i itself in the same class. Thus, for every missing value prediction, nearly all of the instances are selected as compared instances.

The time complexity of the kernel method is $O(n^2)$, where n is the number of instances of the dataset. After clustering, assume that the dataset is divided into k clusters, where $n_i (i = 1, 2, ..., k)$ is the size of cluster i. Because our CMI algorithm performs the kernel method independently on each cluster for missing value filling, so the complexity of our clustering-based kernel imputation method is $O(n_j^2)$, where n_j is the biggest number, i.e., cluster j is the largest one of all the clusters. Generally speaking, n_j is smaller than n when $k>1$, so we have $O(n_j{}^2) < O(n^2)$. That is, the time complexity of our method is better than the method in [20] without clustering.

4 Experimental Studies

In order to evaluate the effectiveness of our approach, we have conducted extensive experiments on datasets from the UCI machine learning repository [18]. We evaluate our algorithm on the dataset *abalone*, which contains 8 continuous attributes, one class attribute and 4177 instances in total. The other dataset is *housing* dataset, which contains 13 continuous attributes (including "class" attribute "MEDV"), one binary-valued attribute and 506 instances in total. For ease of comparison, we use random missing mechanism to generate missing values with missing rates at 5%, 20% and 40%. In the previous discussions of our strategy for handling missing values, we know that the situation of K=1 (i.e., only one cluster) is the special case, which is equal to the situation of processing the whole dataset without clustering and also similar to the kernel-based imputation method without clustering in [20].

In this paper we only report results on the mean and distribution function of Y. We use the AE (*average error*) to measure performance in making inference on the former two parameters:

$$AE = \frac{1}{k} \sum_{i=1}^{k} (|\hat{V}_i - V_i| / V_i) \tag{7}$$

Where $\hat{V}_i$ is the estimated parameter (variance or empirical distribution function) value, computed from the imputed target attribute, and V_i is the parameter value of the original target attribute and k is the number of clusters.

In this paper, for the *abalone* dataset, the last attribute *Rings* is set to target attribute, others are set to factor attributes. The experimental results on *abalone* are presented in Figure 1, from (1) to (6). For the *housing* dataset, the attribute *MEDV* is set to target attribute, the results are presented in Figure 2, from (7) to (12). In these figures, 'Mean substitution' means the method of imputing missing values with mean, our method is regard as 'CMI'. In particular, it is the method in [20] while k=1 in our method, i.e., it is the method for missing values imputation without cluster.

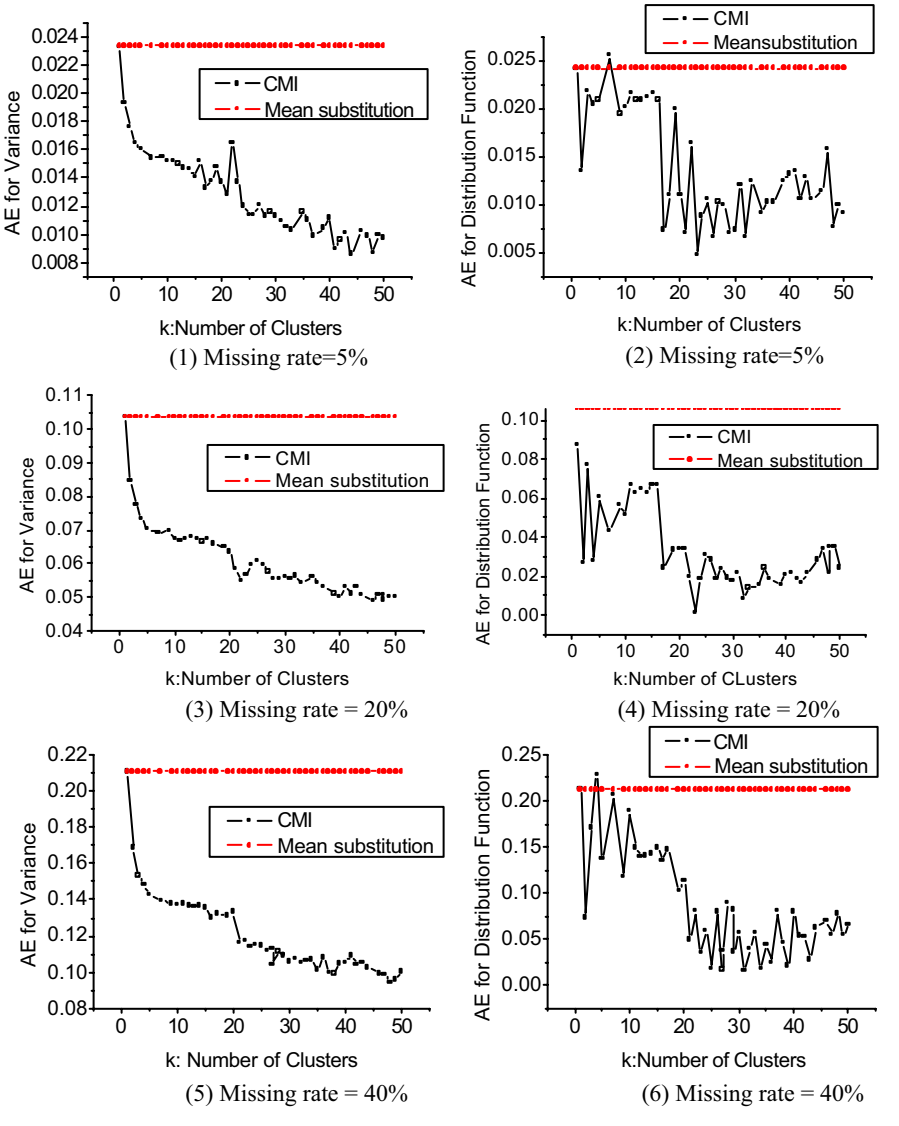
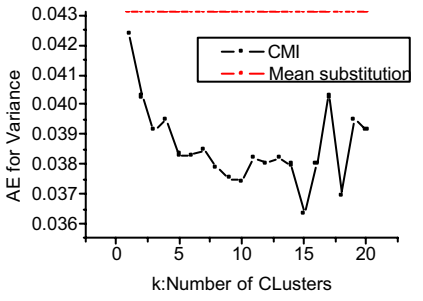
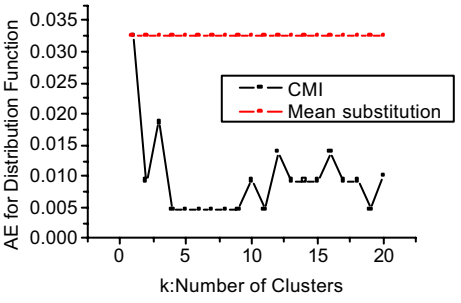


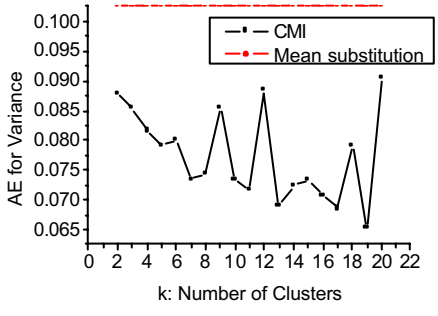
Fig. 1. CMI vs Mean substitution under different missing rates on dataset *abalone* for variance and distribution function



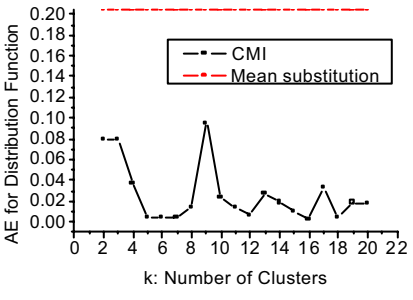
(7) Missing rate=5%



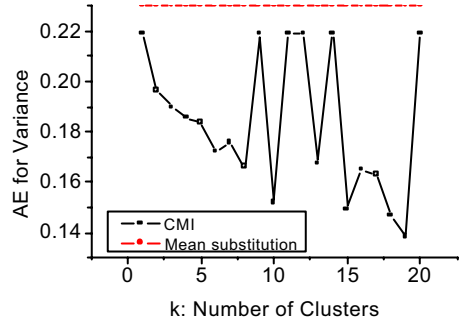
(8) Missing rate=5%



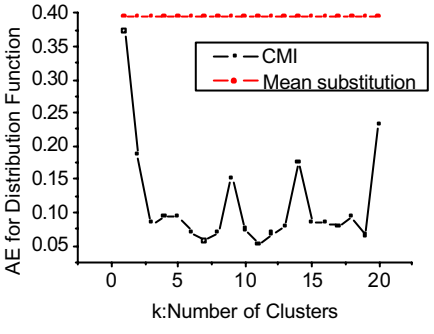
(9) Missing rate = 20%



(10) Missing rate = 20%



(11) Missing rate = 40%



(12) Missing rate = 40%

Fig. 2. CMI vs Mean substitution under different missing rates on dataset *housing* for variance and distribution function

From above figures, the method 'Mean substitution' is worst, the kernel method without clustering (k=1 in our CMI algorithm) outperform the former, and we can see that the clustering-based kernel method performs better most of the time than the kernel method that without clustering (i.e., the situation of k=1 when using k-means) in terms of variance and distribution function. All of results of our method are better than the results of the method 'Mean substitution'. What's more, with increase of

cluster number *k*, the average errors (AE) for variance and distribution function are decreasing. That implies that it is reasonable for us to impute missing values with cluster-based kernel methods. Yet the value of AE will increase when the number of clusters is big enough, this trend will be observed in the above figures. That is to say, the more clusters the worse performance the results of imputation are. That is because there will be less instances for imputing missing values in one cluster while the number of clusters become bigger. In our experiments, for the *Abalone* dataset (in figure 1, (1) to (6)), the best *K*, that is the number of clusters for K-means algorithm, ranges from 25 to 35; while for the *Housing* dataset (in figure 2, (7) to (12)), the best *K* ranges from 4 to 7. Note that for the large dataset, such as, the dataset *abalone* in Fig.1, the AE increases gradually while in small dataset (for instance, the dataset *Housing* in Fig.2) it increases rapidly. Because the number of instances in each cluster will change slightly when the dataset is large and there are more observed information for imputing missing values in one cluster. That makes the values of AE for variance and distribution function relatively stable compared with the previous imputation results.

These results are consistent with the results obtained by using the G-means algorithm in [19]. This means that user can use the G-means algorithm to work out the number of clusters, i.e. *K*, for the dataset at first, and then utilizes our CMI algorithm based on the *K*, in order to deal with the missing value problems on each of the cluster. As a consequence, user will be easily to choose an appropriate *K* for clustering in advance, without degrading the system performances for missing value imputation.

5 Conclusions and Future Work

In this paper, we propose a clustering-based non-parametric kernel-based imputation method, called CMI, for dealing with missing values, which is presented in target attribute in data preprocessing. Extensive experimental results have demonstrated the effectiveness of CMI method in making inference for variance and the distribution function after clustering. In practice, datasets usually present missing values in conditional attributes and class attributes, which makes the problem of missing value imputation more sophisticated. In our future work, we will deal with this problem.

References

1. Zhang, S.C., et al.: Information Enhancement for Data Mining. IEEE Intelligent Systems, 2004 19(2), 12–13 (2004)
2. Zhang, S.C., et al.: Missing is useful: Missing values in cost-sensitive decision trees. IEEE Transactions on Knowledge and Data Engineering 17(12), 1689–1693 (2005)
3. Qin, Y.S., et al.: Semi-parametric Optimization for Missing Data Imputation. Applied Intelligence 27(1), 79–88 (2007)
4. Zhang, C.Q., et al.: An Imputation Method for Missing Values. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 1080–1087. Springer, Heidelberg (2007)

5. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
6. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2006)
7. Chen, J., Shao, J.: Jackknife variance estimation for nearest-neighbor imputation. J. Amer. Statist. Assoc. 96, 260–269 (2001)
8. Lall, U., Sharma, A.: A nearest-neighbor bootstrap for resampling hydrologic time series. Water Resource. Res. 32, 679–693 (1996)
9. Chen, S.M., Chen, H.H.: Estimating null values in the distributed relational databases environments. Cybernetics and Systems: An International Journal 31, 851–871 (2000)
10. Chen, S.M., Huang, C.M.: Generating weighted fuzzy rules from relational database systems for estimating null values using genetic algorithms. IEEE Transactions on Fuzzy Systems 11, 495–506 (2003)
11. Magnani, M.: Techniques for dealing with missing data in knowledge discovery tasks (2004) (Version of June 2004), http://magnanim.web.cs.unibo.it/data/pdf/missingdata.pdf
12. Kahl, F., et al.: Minimal Projective Reconstruction Including Missing Data. IEEE Trans. Pattern Anal. Mach. Intell. 23(4), 418–424 (2001)
13. Gessert, G.: Handling Missing Data by Using Stored Truth Values. SIGMOD Record, 2001 20(3), 30–42 (1991)
14. Pesonen, E., et al.: Treatment of missing data values in a neural network based decision support system for acute abdominal pain. Artificial Intelligence in Medicine 13(3), 139–146 (1998)
15. Ramoni, M., Sebastiani, P.: Robust Learning with Missing Data. Machine Learning 45(2), 147–170 (2001)
16. Pawlak, M.: Kernel classification rules from missing data. IEEE Transactions on Information Theory 39(3), 979–988 (1993)
17. Forgy, E.: Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. Biometrics 21, 768 (1965)
18. Blake, C.L., Merz, C.J.: UCI Repository of machine learning databases (1998)
19. Hamerly, H., Elkan, C.: Learning the k in k-means. In: Proc. of the 17th intl. Conf. of Neural Information Processing System (2003)
20. Zhang, S.C., et al.: Optimized Parameters for Missing Data Imputation. In: Yang, Q., Webb, G. (eds.) PRICAI 2006. LNCS (LNAI), vol. 4099, pp. 1010–1016. Springer, Heidelberg (2006)
21. Wang, Q., Rao, J.: Empirical likelihood-based inference in linear models with missing data. Scand. J. Statist. 29, 563–576 (2002a)
22. Wang, Q., Rao, J.N.K.: Empirical likelihood-based inference under imputation for missing response data. Ann. Statist. 30, 896–924 (2002b)
23. Silverman, B.: Density Estimation for Statistics and Data Analysis. Chapman and Hall, New York (1986)
24. Friedman, J., et al.: Lazy Decision Trees. In: Proceedings of the 13th National Conference on Artificial Intelligence, pp. 717–724 (1996)
25. John, S., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge (2004)
26. Lakshminarayan, K., et al.: Imputation of Missing Data Using Machine Learning Techniques. KDD 1996, 140–145 (1996)

Laminar Forced Convection in Circular Duct for Power-Law Fluid

Tudor Boaca[1] and Ioana Boaca[2]

[1] Petroleum-Gas University of Ploiesti, Romania
tboaca@yahoo.com
[2] University of Bucharest, Romania
ioana.boaca@yahoo.com

Abstract. This paper considers the problem of viscous dissipation in power-law fluid flow through a tube of circular cross section. The solution to the problem is obtained by a series expansion about the complete eigenfunctions system of a Sturm-Liouville problem. The eigenfunctions and eigenvalues of this Sturm-Liouville problem are obtained by Galerkin's method. The Graetz problem is also considered. Numerical examples are given for a viscous fluid with unit Brinkman number.

Keywords: dissipation, power-law, eigenfunction, Graetz problem, Galerkin.

1 Introduction

The problem of laminar forced convection in circular tubes is important in many practical applications. As an example we mention the transport of petroleum products, the polymer processing, certain technological process from the chemical industry and food industry.

Even in the case of laminar flow, the determination of the exact solution of the problem is very difficult. In many situations we consider the easier problem of determining the temperature in the case when the temperature doesn't depend on time (the Graetz-Nusselt problem). In this case the approximate solution of the problem is searched by using the simplified hypothesis:

-the physical properties of the fluid don't depend on the temperature;
-the heat transfer by conduction in the axial direction is negligible compared to both the convection in the axial direction and the conduction in the radial direction.

In the literature of the laminar forced convection in circular tubes, the effect of viscous dissipation is almost always neglected, Indeed, this effect is usually considered to be relevant in two cases: the flow in capillary tubes and the flow of very viscous fluids. Anyway, in the case of power-law fluids the neglection of viscous dissipation doesn't simplify the problem.

The problem of laminar forced convection in circular tubes has constituted the object of many researches. Various approximate methods have been proposed for the determination of the solution: the separation of variables method, the Laplace

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 139–158, 2008.

transform, Galerkin's method, the finite difference method, the finite integral transform method, the Bessel series and power series expansions of Kummer function [20], [6], [18], [10], [8], [12], [13], [14], [9].

The Graetz problem [8] describes the temperature field in fully-developed laminar flow in a circular tube where the wall temperature profile is a step function [17]. The first approximate solution of this problem was obtained by the separation of variable method and by the power series method [8], [17]. As well as the power series method other methods were also proposed for the determination of the eigenvalues of the Graetz problem [11], [20], [4], [9].

The temperature distribution of molten polymer flows inside a semi-infinite circular straight tube with viscous dissipation is related to the problem of viscous dissipation of power-law fluids [2], [15]. The equation which describes the phenomena is of parabolic and nonlinear type. A mathematical analysis of this equation and a finite element analysis are given in [21], [22].

In this article we present a unified approach to some problems concerning the laminar forced convection in circular tubes for power-law fluids: the viscous dissipation and the Graetz problem with Dirichlet and Neumann boundary conditions. By using the separation of variables method the solutions of these problems are obtained under the form of Fourier series by the eigenfunction complete system of some Sturm-Liouville problems. We use Galerkin's method to determine the eigenfunctions and the eigenvalues of these Sturm-Liouville problems. Thus, we use Bessel's functions of first kind as functions of coordinates.

Now we will consider the laminar flow of power-law fluid through a tube of circular cross section. At the entrance of tube the temperature of fluid is T_0. The flow is slow thus we can neglect the heat transfer by conduction in flow direction. At the same time we will consider that the fluid density ρ, specific heat C_p and the heat transfer coefficient k are constant. The flow is related to a polar spatial coordinate system, the Ox axis is along the tube axis, the radial coordinate will be considered to be *r* and R is the radius of the tube. For the fluid velocity in the cross section we will consider the expression

$$v = v_m \cdot \frac{3\nu+1}{\nu+1} \cdot \left[1-\left(\frac{r}{R}\right)^N\right] \tag{1}$$

where v_m is the mean fluid velocity, $N=(\nu+1)/\nu$ where ν is a rheological constant of the fluid. For Newtonian fluids $\nu=1$, for Bingham expanded fluid $\nu<1$ and for Bingham pseudo plastic fluid $\nu>1$.

Given those conditions the energy equation is [20]:

$$\rho C_p \mathrm{v_m} \frac{3\nu+1}{\nu+1}\left[1-\left(\frac{r}{R}\right)^N\right]\frac{\partial T}{\partial x} = k\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right)+K\left|\frac{\partial v}{\partial r}\right|^{\nu+1} \tag{2}$$

where *K* is a rheological constant of the fluid.

The aim of this article is to establish an approximate solution of equation (2), which verifies certain initial and boundary conditions.

The plan of the article is: we formulate the mathematical problem in section two, section three contains the algorithm for the determination of eigenvalues and eigenfunctions (for the Sturm-Liouville problem obtained by method of separation of variables) with Galerkin's method, section four contains the approximate solution of the problems, section five contains the case of Newtonian fluid and in the last section we present some numerical results. Conclusions and future works are presented at the end of the article.

2 The Mathematical Problem

We consider the equation (2) and some initial and boundary conditions.

2.1 Viscous Dissipation with Dirichlet Boundary Condition

In this case we study the viscous dissipation in the laminar flow of a power-law fluid through a tube of circular cross section. At the entrance of the tube the fluid temperature is T_0 (condition (3)) and the wall of the tube has the same temperature (condition (5)).

We associate to equation (2) the initial condition

$$x = 0\,,\, T = T_0 \tag{3}$$

and the boundary conditions

$$r = 0\,,\, \frac{\partial T}{\partial r} = 0\,,\, (x > 0) \tag{4}$$

$$y = R\,,\, T = T_0\,,\, (x > 0) \tag{5}$$

Condition (4) specifies that at the axis of the tube the temperature has a maximum point. It is suitable to rewrite the equation (2) and the initial and boundary conditions (3), (4), (5) in dimensionless form. In this case we have to determine the solution of the equation (2) with the conditions (3), (4) and (5). With the transformation group

$$\theta = \frac{T\text{-}T_0}{T_0}\,,\, \eta = \frac{r}{R}\,,\, \psi = \frac{(\nu+1)k}{(3\nu+1)\rho C_p R^2 v_m}x \tag{6}$$

the equation (2) and the boundary conditions (3), (4), (5) become:

$$\left(1-\eta^N\right)\frac{\partial\theta}{\partial\psi} = \frac{1}{\eta}\frac{\partial}{\partial\eta}\left(\eta\frac{\partial\theta}{\eta}\right) + N_{Br}\eta^N \tag{7}$$

$$\psi = 0\,,\, \theta = 0 \tag{8}$$

$$\eta = 0\,,\ \frac{\partial \theta}{\partial \eta} = 0\,,\ (\psi > 0) \tag{9}$$

$$\eta = 1\,,\ \theta = 0\,,\ (\psi > 0) \tag{10}$$

In equation (7) the coefficient N_{Br} is the Brinkman number [20], [24].

It is easy to demonstrate that a particular solution of equation (7) which verifies the conditions (9) and (10) is:

$$\theta_1 = \frac{N_{Br}}{(N+2)^2}\left(1-\eta^{N+2}\right) \tag{11}$$

The change of function

$$\theta = u + \theta_1 \tag{12}$$

leads to the equation

$$\left(1-\eta^N\right)\frac{\partial u}{\partial \psi} = \frac{1}{r}\frac{\partial}{\partial \eta}\left(r\frac{\partial \theta}{\partial \eta}\right) \tag{13}$$

unknown function u satisfies the conditions (9) and (10). The initial condition (8) is replaced by:

$$\psi = 0\,,\ \ u = -\theta_1 \tag{14}$$

Thus the problem which has to be solved consist of the equation (13), the initial condition (14) and the boundary conditions (9) and (10).

2.2 Viscous Dissipation with Neumann Boundary Condition

In this case we study the viscous dissipation in the laminar flow of a power-law fluid through a tube of circular cross section. At the entrance of the tube the fluid temperature is T_0 and the heat flux vanish at the wall. So the condition (5) is replaced by

$$y = R\,,\ \frac{\partial T}{\partial r} = 0\,,\ (x > 0) \tag{15}$$

With the same transformation group (6) this condition becomes

$$\eta = 1\,,\ \frac{\partial \theta}{\partial \eta} = 0\,,\ (\psi > 0) \tag{16}$$

The problem which has to be solved in this case consists of the equation (7) and the conditions (8), (9) and (16). It is easy to demonstrate that a particular solution of equation (7) which verifies the conditions (9) and (16) is:

$$\theta_2 = \frac{N_{Br}}{2N}\left(4\psi + \eta^2 - \frac{2}{N+2}\eta^{N+2}\right) \tag{17}$$

The change of function

$$\theta = u + \theta_2 \tag{18}$$

leads to the equation (13) and the unknown function u will satisfy the conditions (9) and (16); the initial condition (8) is replaced by:

$$\psi = 0\,,\ u = -\theta_2 \tag{19}$$

2.3 The Graetz Problem with Dirichlet Boundary Condition

In this case the viscous dissipation is neglected. At the entrance of the tube the fluid temperature is T_0 and the temperature of the wall is $T_1 \neq T_0$. The energy equation is

$$\rho C_p \mathrm{v_m} \frac{3\nu+1}{\nu+1}\left[1-\left(\frac{r}{R}\right)^N\right]\frac{\partial T}{\partial x} = k\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) \tag{20}$$

and the unknown function T satisfies the conditions (3), (4) and

$$y = R\,,\ T = T_1\,,\ (x > 0) \tag{21}$$

with $T_1 \neq T_0$.

With the transformation group

$$\theta = \frac{T\text{-}T_0}{T_1 - T_0}\,,\ \eta = \frac{r}{R}\,,\ \psi = \frac{(\nu+1)k}{(3\nu+1)\rho C_p R^2 v_m}x \tag{22}$$

the equation (20) and the conditions (3), (4) and (21) become:

$$\left(1-\eta^N\right)\frac{\partial\theta}{\partial\psi} = \frac{1}{\eta}\frac{\partial}{\partial\eta}\left(\eta\frac{\partial\theta}{\eta}\right) \tag{23}$$

$$\psi = 0\,,\ \theta = 0 \tag{24}$$

$$\eta = 0\,,\ \frac{\partial\theta}{\partial\eta} = 0\,,\ (\psi > 0) \tag{25}$$

$$\eta = 1\,,\ \theta = 1\,,\ (\psi > 0)\,. \tag{26}$$

In this case a particular solution of the equation (23) which verifies the conditions (25) and (26) is

$$\theta_3 = 1 \tag{27}$$

The change of function

$$\theta = u + \theta_3 \tag{28}$$

leads to the problem

$$\left(1-\eta^N\right)\frac{\partial\theta}{\partial\psi}=\frac{1}{\eta}\frac{\partial}{\partial\eta}\left(\eta\frac{\partial\theta}{\eta}\right) \tag{29}$$

$$\psi=0\,,\,\theta=-\theta_3 \tag{30}$$

$$\eta=0\,,\,\frac{\partial\theta}{\partial\eta}=0\,,\,(\psi>0) \tag{31}$$

$$\eta=1\,,\,\theta=0\,,\,(\psi>0) \tag{32}$$

2.4 The Graetz Problem with Neumann Boundary Condition

In this case the viscous dissipation is neglected. At the entrance of the tube the fluid temperature is T_0 and the heat flux is constant at the wall. Thus the problem which has to be solved consists of equation (20) and the conditions (3), (4) and

$$r=R\,,\,k\frac{\partial T}{\partial r}=q\,,\,(x>0) \tag{33}$$

where $q\neq 0$ is the constant wall heat flux.

With the transformation group

$$\theta=\frac{k}{q\cdot R}\cdot\left(T-T_0\right),\eta=\frac{r}{R}\,,\,\psi=\frac{(\nu+1)k}{\left(3\nu+1\right)\rho C_p R^2 v_m}x \tag{34}$$

the equation (20) becomes the equation (23), the conditions (3), (4) become (24), (25) and the condition (33) becomes:

$$\eta=R\,,\,\frac{\partial\theta}{\partial\eta}=1\,,\,(\psi>0)\,. \tag{35}$$

In this case a particular solution of the equation (23) which verifies the conditions (25) and (33) is

$$\theta_4=\frac{N+2}{N}\cdot\left[2\cdot\psi-\frac{2}{\left(N+2\right)^2}\cdot\eta^{N+2}+\frac{1}{2}\cdot\eta^2\right] \tag{36}$$

The change of function

$$\theta=u+\theta_4 \tag{37}$$

leads to the problem

$$\left(1-\eta^N\right)\frac{\partial\theta}{\partial\psi}=\frac{1}{\eta}\frac{\partial}{\partial\eta}\left(\eta\frac{\partial\theta}{\eta}\right) \tag{38}$$

$$\psi = 0\,,\ \theta = -\theta_4 \tag{39}$$

$$\eta = 0\,,\ \frac{\partial\theta}{\partial\eta} = 0\,,\ (\psi > 0) \tag{40}$$

$$\eta = 1\,,\ \frac{\partial\theta}{\partial\eta} = 0\,,\ (\psi > 0) \tag{41}$$

The four problems above are thus reduced to the determination of the solution of the mixed problem of parabolic type:

$$\left(1-\eta^N\right)\frac{\partial\theta}{\partial\psi} = \frac{1}{\eta}\frac{\partial}{\partial\eta}\left(\eta\frac{\partial\theta}{\eta}\right) \tag{42}$$

$$\psi = 0\,,\ \theta = -\theta_i \quad,\quad i = \overline{1,4} \tag{43}$$

$$\eta = 0\,,\ \frac{\partial\theta}{\partial\eta} = 0\,,\ (\psi > 0) \tag{44}$$

$$\eta = 1\,,\ \theta = 0\,,\ (\psi > 0) \tag{45}$$

or

$$\eta = 1\,,\ \frac{\partial\theta}{\partial\eta} = 0\,,\ (\psi > 0) \tag{46}$$

The type of equation (42) and boundary conditions (44) and (45), (46) allow us to apply the method of separation of variables in order to determine the function u. By this method the function u is obtained under the form:

$$u(\psi,\eta) = \sum_{n=1}^{\infty} c_n \Phi_n(\eta)\exp\left(-\lambda_n^2\psi\right) \tag{47}$$

where Φ_n and λ_n are the eigenvalues and the eigenfunctions of Sturm-Liouville problem:

$$\frac{\mathrm{d}}{d\eta}\left(\eta\frac{\mathrm{d}\Phi}{\mathrm{d}\eta}\right) + \lambda^2\eta\left(1-\eta^N\right)\Phi = 0 \tag{48}$$

$$\eta = 0\,,\ \frac{\mathrm{d}\Phi}{\mathrm{d}\eta} = 0\,;\ \eta = 1\,,\ \frac{d\Phi}{d\eta} = 0 \tag{49}$$

or of Sturm-Liouville problem:

$$\frac{\mathrm{d}}{d\eta}\left(\eta\frac{\mathrm{d}\Phi}{\mathrm{d}\eta}\right) + \lambda^2\eta\left(1-\eta^N\right)\Phi = 0 \tag{50}$$

$$\eta = 0\,,\ \frac{\mathrm{d}\Phi}{\mathrm{d}\eta} = 0\,;\ \eta = 1\,,\ \Phi = 0 \tag{51}$$

For the statements above we draw the conclusion that solving the problem 2.1, 2.2, 2.3 and 2.4 is reduced to solving the Sturm-Liouville problem (48), (49) and (50), (51). In order to determine the eigenfunctions and the eigenvalues from the formula (47) we will use Galerkin's method in the next paragraph.

3 The Application of Galerkin's Method

For the determination of eigenfunctions and eigenvalues of Sturm-Liouville problem (48), (49) we will apply Galerkin's method. Galerkin's method is one of the most used methods for the determination of the eigenvalues and the eigenfunctions of the Sturm-Liouville problem. The key of this method consist in determining the projections of the eigenfunctions on certain finite dimensional linear subspace. Next these linear subspace will be generated by the Bessel functions of first kind.

In order to apply Galerkin's method we consider the bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ defined on $H^1(0,1)\times H^1(0,1)$:

$$\begin{aligned} a(u,v) &= -\int_0^1 \frac{\mathrm{d}}{\mathrm{d}\eta}\left(\eta\frac{\mathrm{d}u}{\mathrm{d}\eta}\right)\cdot v\cdot \mathrm{d}\eta = \int_0^1 \eta\frac{\mathrm{d}u}{\mathrm{d}\eta}\frac{\mathrm{d}v}{\mathrm{d}\eta}\mathrm{d}\eta, \\ \mathrm{b}(u,v) &= \int_0^1 \eta\cdot\left(1-\eta^N\right)\cdot u\cdot v\cdot \mathrm{d}\eta\,. \end{aligned} \tag{52}$$

We look for the eigenpair (λ,Φ) which satisfies

$$\begin{aligned} &\Phi\in H^1(0,1),\ \Phi\neq 0 \\ &a(\Phi,v) = \lambda^2\cdot b(\Phi,v),\ (\forall)\, v\in H^1(0,1). \end{aligned} \tag{53}$$

For the problem (50), (51) the bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ are defined on $\tilde{H}(0,1)\times\tilde{H}(0,1)$ where

$$\tilde{H}(0,1) = \left\{ u\in H^1 \;\middle|\; \frac{\mathrm{d}u}{\mathrm{d}\eta}(0) = 0\,,\ u(1) = 0 \right\}. \tag{54}$$

In order to solve (50), (51) we look for the eigenpair (λ,Φ) which satisfies

$$\begin{aligned} &\Phi\in \tilde{H}(0,1),\ \Phi\neq 0 \\ &a(\Phi,v) = \lambda^2\cdot b(\Phi,v),\ (\forall)\, v\in \tilde{H}(0,1). \end{aligned} \tag{55}$$

(53) and (55) are called a variational formulation of (48), (49) and (50), (51) respectively [5].

We look for the solution of (53) and (55) under the approximate form

$$\Phi(\eta)=\sum_{k=1}^{n} a_k \varphi_k(\eta) \tag{56}$$

where $n\in \mathbf{N}^*$ is the approach level of function Φ and $(\varphi_k)_{k\in \mathbf{N}^*}$ is a complete system of functions in H^1 and $\tilde{H}$ respectively. For the problem (49) the functions φ_k verify the conditions

$$\frac{d\varphi_k}{d\eta}(0)=0\,,\ \frac{d\varphi_k}{d\eta}(1)=0\,,\ k\in \mathbf{N}^* \tag{57}$$

The unknown coefficients a_k, $k=\overline{1,n}$ are determined if we give the conditions

$$a(\Phi_n,\varphi_j)=\lambda^2\cdot b(\Phi_n,\varphi_j),\ j=\overline{1,n}\,. \tag{58}$$

By applying these conditions we obtain the linear algebraic system in unknown a_k, $k=\overline{1,n}$

$$\sum_{k=1}^{n}\left(\alpha_{kj}+\lambda^2\beta_{kj}\right)a_k=0\,,\ j=\overline{1,n} \tag{59}$$

where

$$\alpha_{kj}=-a(\varphi_k,\varphi_j),\ j,k=\overline{1,n}\,, \tag{60}$$

$$\beta_{kj}=b(\varphi_k,\varphi_j),\ j,k=\overline{1,n}\,. \tag{61}$$

Obviously the system (59) has the trivial solution. This solution can't be used because it leads us to the trivial eigenfunction (see formula (56)) which is in contradiction with the condition (55). This condition implies that the system (59) has also nontrivial solutions. Thus, we obtain the equation

$$\Delta_n\equiv\left|A+\lambda^2 B\right|=0\,, \tag{62}$$

where *A* and *B* are the matrix $A=(\alpha_{kj})_{k,j=\overline{1,n}}$, $B=(\beta_{kj})_{k,j=\overline{1,n}}$.

The solutions of the equations (62) represent the approximate values, for the *n* approach level, for the eigenvalues λ_1^2, $\lambda_2^2,\cdots,\lambda_n^2$.

The solutions of equation (62) are difficult to be obtained under this form. Consequently, through elementary transformations of determinant Δ_n this equation takes the form:

$$\left|C-\lambda^2 I_n\right|=0\,, \tag{63}$$

where I_n is the identity matrix of *n* order.

In the following we will use the complete system of functions $(\varphi_k)_{k\in\mathbf{N}^*}$

$$\varphi_k(\eta)=J_0(\mu_k\cdot\eta) \tag{64}$$

where J_0 is the Bessel function of the first kind and zero order and $\mu_k, k\in N^*$ are the roots of equation:

$$J_1(\mu)=0 \tag{65}$$

for the problem (53) and the equation

$$J_0(\mu)=0 \tag{66}$$

for the problem (55).

The integrals which appear in formula (52) are calculated with a quadrature formula. The eigenvalues of the Sturm-Liouville problem obtained through this method are presented in the next section.

The eigenfunctions of the problems (53), (55) are the analytical form

$$\Phi_i(\eta)=\sum_{j=1}^{n}c_{ij}\cdot J_0(\mu_k\cdot\eta)\ ,\ i=\overline{1,n}\,, \tag{67}$$

where $(c_{i1},c_{i2},\cdots,c_{in}), i=\overline{1,n}$ are the eigenvectors of matrix $A+\lambda^2 B$.

4 The Approximate Solution of the Problems

The unknown function *u*, for the *n* level of approximation of Galerkin's method, is obtained from (15) and (56):

$$u(\psi,\eta)=\sum_{k=1}^{n}\left(\sum_{i=1}^{n}c_i c_{ik}\mathrm{e}^{-\lambda_i^2\psi}\right)J_0(\mu_k\eta). \tag{68}$$

The coefficients c_i , $i=\overline{1,n}$ from (68) are determined by using the condition (45) and by considering that the solutions $\Phi_i, i=\overline{1,n}$ of the Sturm-Liouville problem are orthogonal with weight $\eta(1-\eta^N)$ on $[0,1]$. Because the functions $\Phi_i, i=\overline{1,n}$ are not obtained exactly, we prefer to use the orthogonality with weight η of Bessel functions on $[0,1]$.

Thus, for the *n* level of approximation, the constants c_i , $i=\overline{1,n}$ are determined by the resolution of the linear algebraic system:

$$\sum_{i=1}^{n}c_{ik}c_i=-\frac{N_{Br}}{2N}\frac{\int_0^1\eta\cdot\theta_j\cdot J_0(\mu_k\eta)\mathrm{d}\eta}{\int_0^1\eta J_0^2(\mu_k\eta)\mathrm{d}\eta},\quad k=\overline{1,n}\,. \tag{69}$$

The final solution of the problem is obtained now by using the relations (12), (28) (37) and (68):

$$\theta(\psi,\eta)=\theta_j+\sum_{k=1}^{n}\left(\sum_{i=1}^{n}c_i c_{ik}\,\mathrm{e}^{-\lambda_i^2\psi}\right)J_0(\mu_k\eta). \tag{70}$$

5 The Case of Newtonian Fluid

In the case of Newtonian fluid ($\nu=1$) the eigenfunctions can be obtained exactly by using the confluent hypergeometric function as follows. In the equation

$$\frac{d}{d\eta}\left(\eta\frac{\mathrm{d}\Phi}{\mathrm{d}\eta}\right)+\lambda^2\eta\left(1-\eta^2\right)\Phi=0 \tag{71}$$

we make the change of variables

$$\begin{gathered}\tau=\lambda\eta^2\\ \Phi(\eta)=e^{-\frac{\lambda\eta^2}{2}}\varphi\left(\lambda\eta^2\right).\end{gathered} \tag{72}$$

Thus the equation (71) becomes:

$$\tau\frac{d^2\varphi}{d\tau^2}+(1-\tau)\frac{d\varphi}{d\tau}+\left(\frac{\lambda}{4}-\frac{1}{2}\right)\varphi=0. \tag{73}$$

This is the confluent hypergeometric equation [1].

The bounded solution of this equation is $\varphi(\tau)={}_1F_1\left(\frac{1}{2}-\frac{\lambda}{4},1,\tau\right)$ where ${}_1F_1$ is the Kummer's function [1]. Thus the solution of the equation (71) is:

$$\Phi(\eta)=e^{-\frac{\lambda\eta^2}{2}}\,{}_1F_1\left(\frac{1}{2}-\frac{\lambda}{4},1,\lambda\eta^2\right). \tag{74}$$

From the Dirichlet boundary condition

$$\Phi(1)=0 \tag{75}$$

we obtain the transcendent equation

$${}_1F_1\left(\frac{1}{2}-\frac{\lambda}{4},1,\lambda\right)=0, \tag{76}$$

equation from which the eigenvalues of Sturm-Liouville problem are determined.

For the eigenfunctions of Sturm-Liouville problem we have the integral representation

$$\Phi(\eta)=\frac{e^{\frac{\lambda\eta^2}{2}}}{\Gamma\left(\frac{\lambda}{4}+\frac{1}{2}\right)}\cdot\int_0^{\infty}e^{-t}\cdot t^{\frac{\lambda}{4}-\frac{1}{2}}\cdot J_0\left(2\cdot\eta\cdot\sqrt{\lambda\cdot t}\right)\mathrm{d}t \tag{77}$$

obtained from the integral representation of Kummer's function [7]. In (77) Γ is the Euler's gamma function.

From (77) is easily obtained the inequality:

$$\left|\Phi(\eta)\right|\le\exp\left(\frac{\lambda\eta^2}{2}\right) \tag{78}$$

The determination of the coefficients from the variational method:

a) Dirichlet boundary condition. Taking into account that

$$\int_0^1\eta J_0(\mu_k\eta)J_0(\mu_j\eta)\mathrm{d}\eta=\begin{cases}0, & k\neq j\\ \frac{1}{2}J_1^2(\mu_k\eta) & , \quad k=j\end{cases}, \tag{79}$$

the coefficients α_{kj} , β_{kj} are calculated with the formulas:

$$\alpha_{kj}=\begin{cases}0, & k\neq j\\ -\frac{1}{2}J_1^2(\mu_k\eta) & , \quad k=j\end{cases}, \tag{80}$$

$$\beta_{kj}=\begin{cases}-\int_0^1\eta^3J_0(\mu_k\eta)J_0(\mu_j\eta)\mathrm{d}\eta & , \quad k\neq j\\ \frac{1}{2}J_1^2(\mu_k\eta)-\int_0^1\eta^3J_0(\mu_k\eta)J_0(\mu_j\eta)\mathrm{d}\eta & , \quad k=j\end{cases}. \tag{81}$$

b) Mixed boundary condition $\Phi(1)+\beta\cdot\Phi'(1)=0$. In this case we take

$$\varphi_k(\eta)=J_0(\mu_k\eta) \tag{82}$$

where μ_k are the roots of the equation

$$J_0(\mu)+\beta\cdot\mu\cdot J_1(\mu)=0\,. \tag{83}$$

Because we have [19]

$$\int_0^1 \eta J_0(\mu_k\eta)J_0(\mu_j\eta)\mathrm{d}\eta=\begin{cases}0 & , \quad k\neq j\\ \dfrac{\beta^2+\mu_k^2}{2\cdot\mu_k^2}J_0^2(\mu_k) & , \quad k=j\end{cases}, \tag{84}$$

and [16]

$$\int_0^1 \eta^3 J_0(\mu_k\eta)J_0(\mu_j\eta)\mathrm{d}\eta=\begin{cases}\dfrac{2\left(\beta^2+\mu_k^2+\mu_j^2\right)}{\left(\mu_k^2-\mu_j^2\right)^2}J_0(\mu_k)J_0(\mu_j) & , \quad k\neq j\\ \dfrac{\left(\mu_k^2+\beta^2\right)^2+\beta^2\left(\mu_j^2-3\right)}{6\cdot\mu_k^4}J_0^2(\mu_k) & , \quad k=j\end{cases} \tag{85}$$

we obtain the formulas:

$$\alpha_{kj}=\begin{cases}0 & , \quad k\neq j\\ -\dfrac{\beta^2+\mu_k^2}{2}J_0^2(\mu_k) & , \quad k=j\end{cases} \tag{86}$$

$$\beta_{kj}=\begin{cases}-2\cdot\dfrac{\beta^2+\mu_k^2+\mu_j^2}{\left(\mu_k^2-\mu_j^2\right)^2}J_0(\mu_k)J_0(\mu_j) & , \quad k\neq j \quad , \quad k,j=\overline{1,n}\\ \left[\dfrac{\beta^2+\mu_k^2}{2\mu_k^2}-\dfrac{\left(\beta^2+\mu_k^2\right)^2+\beta^2\left(\mu_k^2-3\right)}{6\cdot\mu_k^4}\right]\cdot J_0^2(\mu_k), k=j, k,j=\overline{1,n}\end{cases} \tag{87}$$

c) Neumann boundary condition. In this case the formulas are determined from (81) and (82) where we consider $\beta=0$:

$$\alpha_{1j}=\alpha_{j1} \quad , \quad j=\overline{1,n},$$
$$\alpha_{kj}=\begin{cases}0 & , \quad k\neq j \quad , \quad k,j=\overline{2,n}\\ -\mu_k^2J_0^2(\mu_k)/2 & , \quad k=j \quad , \quad k,j=\overline{2,n}\end{cases} \tag{88}$$
$$\beta_{11}=\frac{1}{4}, \ \beta_{1j}=\beta_{j1}=-\frac{2}{\mu_j^2}J_0(\mu_j) \quad , \quad j=\overline{2,n}$$

$$\beta_{kj}=\begin{cases}-2\cdot\dfrac{\mu_k^2+\mu_j^2}{\left(\mu_k^2-\mu_j^2\right)^2}J_0\left(\mu_k^2\right)J_0\left(\mu_j^2\right) & , \quad k\neq j, \quad k,j=\overline{2,n}\\ J_0^2(\mu_k)/3 & , \quad k=j, \quad k,j=\overline{2,n}\end{cases} \tag{89}$$

6 Numerical Results

As an example we will consider a fluid with unit Brinkman number. The coefficients given by (60), (61) are obtained by a numerical quadrature procedure [23].The eigenvalues of the Sturm-Liouville problem (48), (49) are obtained by solving the equation (63) with the use of the procedures BALANC, ELMHES, HQR [23]. The eigenfunctions are obtained by solving the system (59) by using the Gauss method [23]. The coefficients c from the formula (47) were obtained by solving the linear algebraic system (69) with Gauss's method [23]. The eigenvalues of the Sturm-Liouville problem (48), (49) are presented in table 1.

The variation of dimensionless temperature θ given by (70) is presented in figures 1-4 for viscous dissipation with Neumann boundary condition and in figures 5-7 for viscous dissipation with Dirichlet boundary condition. The reduced radial distance η is presented in the abscisse axis and the dimensionless temperature θ is presented in the axis of ordinates. The variation of the dimensionless temperature θ is presented for some values of the dimensionless axial variable ψ .

The figure 1-4 tell us that the dimensionless temperature θ increases along the tube axis (for some value of radial variable) and with the radial variable (for some value of the axial variable). The temperature increase with ν for certain value of Ψ and η. The figure 5-7 tell us that in the case of Dirichlet boundary condition, the dimensionless temperature increases along the tube axis and converge asymptotically to an expression which doesn't depend on the axial distance.

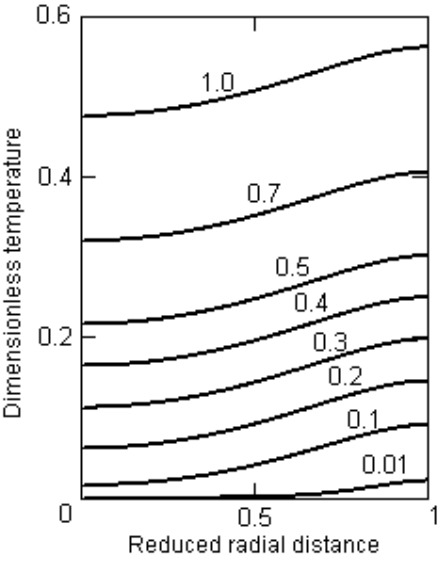


Fig. 1. Dimensionless temperature profile for Neumann boundary condition, $\nu = 0.35$

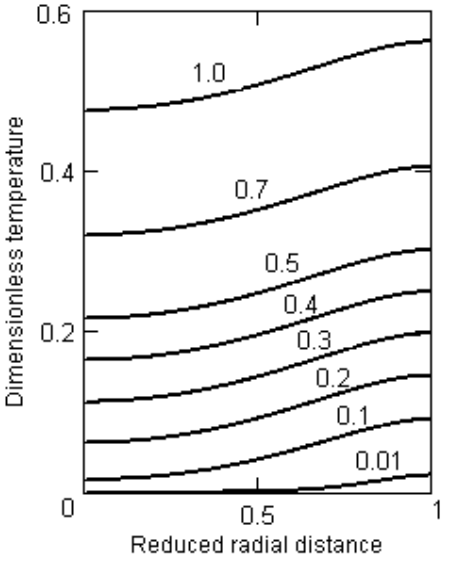


Fig. 2. Dimensionless temperature profile for Neumann boundary condition, $\nu = 0.5$

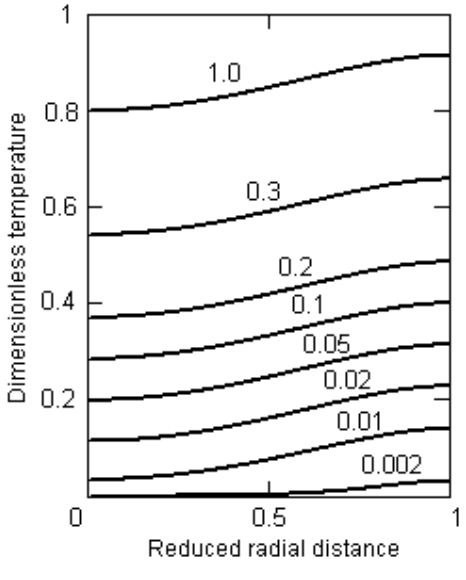


Fig. 3. Dimensionless temperature profile for Neumann boundary condition, $\nu = 0.75$

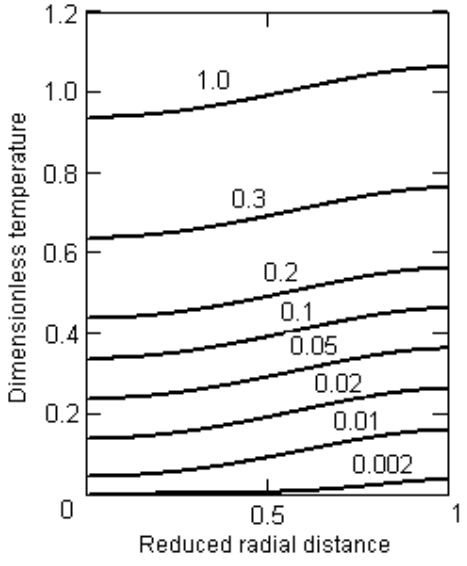


Fig. 4. Dimensionless temperature profile for Neumann boundary condition, $\nu = 1$

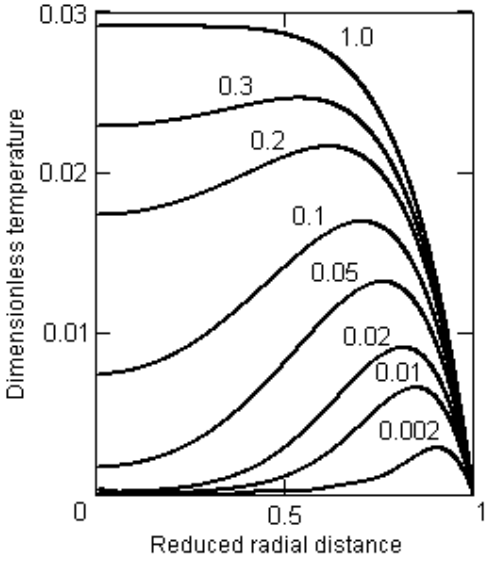


Fig. 5. Dimensionless temperature profile for Dirichlet boundary condition, $\nu = 0.35$

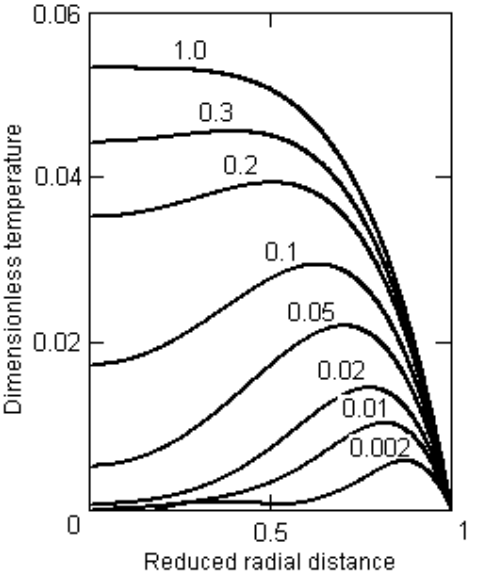


Fig. 6. Dimensionless temperature profile for Dirichlet boundary condition, $\nu = 0.75$

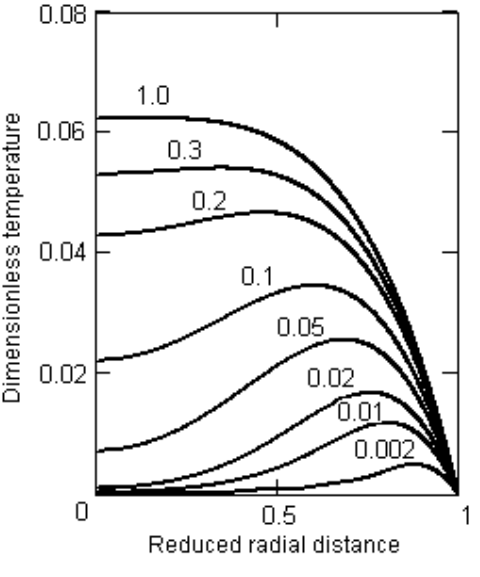


Fig. 7. Dimensionless temperature profile for Dirichlet boundary condition, $\nu = 1$

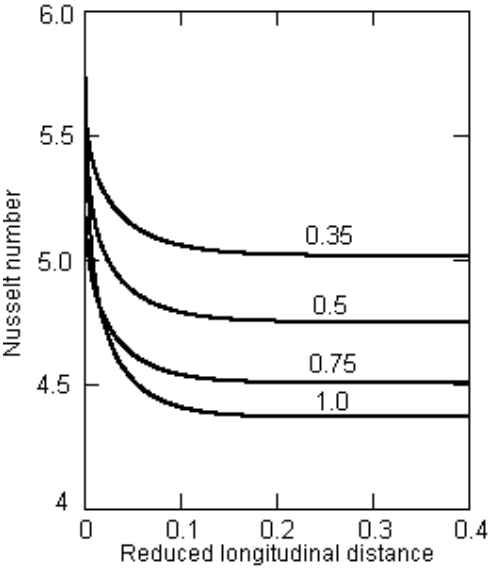


Fig. 8. Variation of Nusselt number for Neumann boundary condition

Table 1. Eigenvalue of Sturm-Liouville problem (Neumann boundary condition)

ν							
0,5	0,6	0,7	0,8	0,9	1,0	1,1	1,2
λ_n^2							
0	0	0	0	0	0	0	0
22.427	23.247	23.966	24.602	25.169	25.679	26.141	26.560
73.102	75.800	78.175	80.283	82.168	83.863	85.395	86.788
151.783	157.384	162.323	166.712	170.637	174.170	177.366	180.271
258.414	267.946	276.356	283.832	290.522	296.544	301.993	306.947
392.974	407.462	420.251	431.622	441.799	450.961	459.253	466.792
555.452	575.924	593.997	610.070	624.458	637.411	649.135	659.795
745.840	773.322	797.588	819.170	838.490	855.885	871.630	885.948
964.136	999.656	1031.02	1058.91	1083.89	1106.38	1126.73	1145.24
1210.33	1254.91	1294.29	1329.31	1360.66	1388.89	1414.45	1437.69

An important similarity criterion in the study of convective heat transfer is the Nusselt number. In the study of heat transfer in the thermal-entry regions of hidrodynamically developed viscous flows, the computation of the local Nusselt number as a function of axial distance is of both practical and theoretical importance, deriving from numerous applications of both Newtonian and non-Newtonian fluids. This number is calculated with the formula [20]:

$$\mathrm{Nu} = -\frac{2 \cdot \left.\dfrac{\partial \theta}{\partial \eta}\right|_{\eta=1}}{<\theta> - \theta_w} \tag{90}$$

where

$$<\theta> = \frac{\displaystyle\int_0^1 \eta\left(1-\eta^N\right)\theta(\eta)\,\mathrm{d}\eta}{\displaystyle\int_0^1 \eta\left(1-\eta^N\right)\mathrm{d}\eta} \tag{91}$$

is the bulk temperature and θ_w is the wall temperature. In figure 8 we present the variation of Nusselt number in function of dimensionless longitudinal variable ψ and some values of parameter ν for Graetz problem and Neumann boundary condition. We find the value 48/11 for the Nusselt number for Newtonian fluids and great values of the distance in the direction of the flow.

7 Conclusions

In this article we present a unified method for solving some problems concerning the laminar forced convection in circular tubes for power-law fluids. The problems are reduced to some Sturm-Liouville problems. Galerkin's method is used for solving these Sturm-Liouville problems. The Bessel functions of first kind are used in this method. We present some numerical examples for a fluid with unit Brinkman number. The results were obtained for ten level of approximation from Galerkin's method. We observe that the proposed algorithm has a good stability and a good precision.

The study of the algorithm convergence and the errors estimation will be the subject of a future article.

References

1. Abramowitz, M., Stegun, I.A.: Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables, National Bureau of Standard, Applied Mathematics Series (1972)
2. Agur, E.E., Vlachopoulos, J.: Heat transfer to molten polymer flow in tubes. J. Appl. Polm. Sci. 26, 765–773 (1981)

3. Barletta, A.: Fully developed laminar forced convection in circular ducts for power-law wall with viscous dissipation. Int. J. Heat Mass Transfer 40(1), 15–26 (1997)
4. Barron, R.F., Wang, X., Warrington, R.O., Ameel, T.: Evaluation of the eigenvalues for the Graetz problem in slip-flow. Int. Comm. Heat Mass Transfer 23(4), 563–574 (1996)
5. Ciarlet, P.C., Lions, J.L.: Handbook of numerical analysis, vol. II. Elsevier, Amsterdam (1991)
6. Gottifredi, J.C., Quiroga, O.D., Floree, A.F.: Heat Transfer to Newtonian and non-Newtonian Fluids Flowing in a Laminar Regime. Int. J. Heat Mass Transfer 26, 1215–1220 (1983)
7. Gradstein, S., Rijik, I.M.: Tables of Integrals, Sums, Serials, and Products, Nauka, Moscow (in Russian) (1971)
8. Graetz, L.: Uber die Wärmeleitungsfähigheit von Flűssigkeiten. Ann. Phys. Chem. 25, 337–357 (1885)
9. Housiadas, C., Larrodé, F.E., Drossinos, Y.: Numerical evaluation of the Graetz series. Int. J. Heat Mass Transfer 42, 3013–3017 (1999)
10. Johnston, P.R.: A Solution Method for the Graetz Problem for non-Newtonian Fluids with Dirichlet and Neumann Boundary Conditions. Math. Comput. Model 19, 1–19 (1994)
11. Őzisik, M.N., Sadeghipour, M.S.: Analytic solution for the eigenvalues and coefficients of the Graetz problem with third kind boundary condition. Int. J. Heat Mass Transfer 25(5), 736–739 (1982)
12. Papoutsakis, E., Ramkrishna, D., Lim, H.C.: The Extended Graetz Problem with Prescribed Wall Flux. AIChE J. 26(779), 779–787 (1980)
13. Papoutsakis, E., Ramkrishna, D., Lim, H.C.: The Extended Graetz Problem with Dirichlet wall Boundary Conditions. Appl. Sci. Res. 36(13) (1980)
14. Papoutsakis, E.: Nusselt numbers Hear Entrance of Heat-Exchange Section in Flow Systems. AIChE J. 27(4), 687–689 (1981)
15. Rice, R.G., Duong, D.D.: Applied Mathematics and Modeling for Chemical Engineers. John Wiley &Sons, New York (1995)
16. Sankarasubramanian, R., Gill, W.N.: Unsteady Convective Diffusion with Interphase mass Transfer. Proc. Roy. Soc. Lond. A 333(115) (1973)
17. Shah, R.K., London, A.L.: Laminar Flow Forced Convection in Ducts. Academic Press, New York (1978)
18. Shih, Y.P., Tsou, J.D.: Extended Leveque Solutions for Heat Transfer to Power-law Fluids in Laminar Flow in a Pipe. Chem. Eng. J. 15, 55–62 (1978)
19. Sneddon, I.N.: Fourier transform, McGraw-Hill Book Company, Inc., New York, Toronto, London (1951)
20. Valkó, P.P.: Solution of the Graetz–Brinkman Problem with the Laplace Transform Galerkin Method. Int. J. Heat Mass Transfer 48, 1874–1882 (2005)
21. Wei, D., Zhang, Z.: Decay estimates of heat transfer to molten polymer flow in pipes with viscous dissipation. Electronic Journal of Differential Equations 39, 1–14 (2001)
22. Wei, D., Luo, H.: Finite element solutions of heat transfer in molten polymer flow in tubes with viscous dissipation. Int. J. Heat Mass Transfer 46, 3097–3108 (2003)
23. William, H.P.: Numerical Recipes in Pascal. Cambridge University Press, Cambridge (1989)
24. Ybarra, R.M., Eckert, R.E.: Viscous Heat Generation in Slit Flow. AiChE Journal 26(5), 751–762 (1980)

The Homotopy Wiener-Hermite Expansion and Perturbation Technique (WHEP)

Magdy A. El-Tawil

Cairo University, Faculty of Engineering, Engineering Mathematics Department, Giza, Egypt
magdyeltawil@yahoo.com

Abstract. The Wiener-Hermite expansion linked with perturbation technique (WHEP) was used to solve perturbed non-linear stochastic differential equations. In this article, the homotopy perturbation method is used instead of the conventional perturbation methods which generalizes the WHEP technique such that it can be applied on non-linear stochastic differential equations without the necessity of the presence of the small parameter. The technique is called homotopy WHEP and is demonstrated through many non-linear problems.

Keywords: Non-linear stochastic differential equations, Wiener-Hermite expansion, WHEP technique, Homotopy perturbation.

1 Introduction

Stochastic non-linear differential equations are mathematical models for many scientific research problems in a variety of applied science and engineering fields. Searching for exact (if possible) or approximate solutions is an active and important research area. Many scientists develop numerous methodologies and algorithms for solving such problems.

Since Meecham and his co-workers [1] developed a theory of turbulence involving a truncated Wiener-Hermite expansion (WHE) of the velocity field, many authors studied problems concerning turbulence [2-7]. A number of general applications in fluid mechanics was also studied in [8,9,10]. Scattering problems attracted the WHE applications through many authors [11-15]. The non-linear oscillators were considered as an opened area for the applications of WHE as can be found in [16-22]. There are many applications in boundary value problems [23,24] and generally in different mathematical studies [25,28].

The application of the WHE aims at finding a truncated series solution to the solution process of differential equations. The truncated series are composed of two major parts; the first is the Gaussian part which consists of the first two terms, while the rest of the series constitute the non-Gaussian part. In non-linear cases, there exists always difficulties of solving the resultant set of deterministic integro-differential equations obtained from the applications of a set of comprehensive averages on the stochastic integro-differential equation resulted after the

M.L. Gavrilova and C.J.K. Tan (Eds.): Trans. on Comput. Sci. I, LNCS 4750, pp. 159–180, 2008.

direct application of WHE. Many authors introduced different methods to face these obstacles. Among them, the WHEP technique was introduced in [21] using the perturbation method to solve non-linear problems.

In this paper, we generalize the WHEP algorithm using the homotopy perturbation. Describing such a general algorithm enlarges its applicability and benefits among users of non-linear models since through which approximate solutions can be got, mainly the average and some useful statistical moments, that describe the stochastic applied problem efficiently according to the required order of approximation.

The WHEP technique is described in section 2 and is applied to a non-linear problem in subsection 2.1. The homotopy perturbation is briefly summarized in section 3 with an example in 3.1 and is extended to non-linear systems in section 4 with an illustrative application in subsection 4.1. Section 5 has the full description of the proposed homotopy WHEP technique. An illustrative case-study is introduced in subsection 5.1 with giving an example in 5.1.1 and its modification in 5.1.2.

2 WHEP Technique

The WHE method utilizes the Wiener-Hermite (WH) polynomials which are the elements of a complete set of statistically orthogonal random functions [29].The Wiener-Hermite polynomial $H^{(i)}(t_1, t_2, ..., t_i)$ satisfies the following recurrence relation:

$$H^{(i)}(t_1, t_2, ..., t_i) = H^{(i-1)}(t_1, t_2, ..., t_{i-1})H^{(1)}(t_i)$$

$$- \sum_{m=1}^{i-1} H^{(i-2)}(t_{i_1}, t_{i_2}, ..., t_{i_{i-2}}) \cdot \delta(t_{i-m} - t_i), i \geq 2 \tag{1}$$

where

$$\begin{aligned}
H^{(0)} &= 1, \\
H^{(1)}(t) &= n(t), \\
H^{(2)}(t_1, t_2) &= H^{(1)}(t_1).H^{(1)}(t_2) - \delta(t_1 - t_2), \\
H^{(3)}(t_1, t_2, t_3) &= H^{(2)}(t_1, t_2).H^{(1)}(t_3) - H^{(1)}(t_1).\delta(t_2 - t_3) \\
&\quad -H^{(1)}(t_2).\delta(t_1 - t_3), \\
H^{(4)}(t_1, t_2, t_3, t_4) &= H^{(3)}(t_1, t_2, t_3).H^{(1)}(t_4) - H^{(2)}(t_1, t_2).\delta(t_3 - t_4) \\
&\quad -H^{(2)}(t_1, t_3).\delta(t_2 - t_4) - H^{(2)}(t_2, t_3).\delta(t_1 - t_4),
\end{aligned} \tag{2}$$

in which $n(t)$ is the white noise with the following statistical properties

$$En(t) = 0, En(t_1).n(t_2) = \delta(t_1 - t_2) \tag{3}$$

where $\delta(-)$ is the Dirac delta function and E denotes the ensemble average operator. The Wiener-Hermite set is a statistically orthogonal set, i.e.

$$E \cdot H^{(i)} \cdot H^{(j)} = 0 \; \forall \; i \neq j. \tag{4}$$

The average of almost all H functions vanishes, particularly,

$$E \cdot H^{(i)} = 0 \text{ for } i \geq 1. \tag{5}$$

Due to the completeness of the Wiener-Hermite set, any random function $G(t;\omega)$ can be expanded as

$$\begin{aligned} G(t;\omega) = G^{(0)}(t) &+ \int_{-\infty}^{\infty} G^{(1)}(t;t_1)H^{(1)}(t_1)dt_1 \\ &+ \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2 \\ &+ \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3 + ... \end{aligned} \tag{6}$$

where the first two terms are the Gaussian part of $G(t;\omega)$. The rest of the terms in the expansion represent the non-Gaussian part of $G(t;\omega)$. The average of $G(t;\omega)$ is

$$\mu_G = EG(t;\omega) = G^{(0)}(t) \tag{7}$$

The covariance of $G(t;\omega)$ is

$$\begin{aligned} Cov(G(t;\omega), G(\tau;\omega)) &= E(G(t;\omega) - \mu_G(t))(G(t;\omega) - \mu_G(\tau)) \\ &= \int_{-\infty}^{\infty} G^{(1)}(t;t_1)G^{(1)}(\tau,t_1)dt_1 \\ &\quad +2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G^{(2)}(t;t_1,t_2)G^{(2)}(\tau,t_1,t_2)dt_1dt_2 \\ &\quad +2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G^{(3)}(t;t_1,t_2,t_3)[G^{(3)}(\tau,t_1,t_2,t_3) \end{aligned}$$

$$+G^{(3)}(\tau,t_1,t_3,t_2) + G^{(3)}(\tau,t_2,t_3,t_1)dt_1dt_2dt_3 + ... \tag{8}$$

The variance of $G(t;\omega)$ is

$$\begin{aligned} VarG(t;\omega) &= E(G(t;\omega)-\mu_G(t))^2 \\ &= \int_{-\infty}^{\infty}[G^{(1)}(t;t_1)]^2dt_1 + 2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}[G^{(2)}(t;t_1,t_2)]^2dt_1dt_2 \\ &\quad +2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}[G^{(3)}(t;t_1,t_2,t_3)]^2dt_1dt_2dt_3 \\ &\quad +2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}[G^{(3)}(t;t_1,t_2,t_3)G^{(3)}(t,t_1,t_3,t_2)]dt_1dt_2dt_3 \\ &\quad +2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}[G^{(3)}(t;t_1,t_2,t_3)G^{(3)}(t,t_2,t_3,t_1)dt_1dt_2dt_3+... \end{aligned} \tag{9}$$

The WHE method can be elementary used in solving stochastic differential equations by expanding the solution process as well as the stochastic input processes via the WHE. The resultant equation is more complex than the original one due to being a stochastic integro-differential equation. Taking a set of ensemble averages together with using the statistical properties of the WH polynomials, a set of deterministic integro-differential equations are obtained in the deterministic kernels $G^{(i)}(t;\omega), i = 0,1,2,....$ To obtain approximate solutions for these deterministic kernels, one can use perturbation theory in the case of having a perturbed system depending on a small parameter, say, t. Expanding the kernels as a power series of t, another set of simpler iterative equations in the kernel series components are obtained. This is the main algorithm of the WHEP technique. The technique was successfully applied to several nonlinear stochastic equations, see [19,21,22,24]. The following is an illustrative example.

2.1 Illustrative Example

Let us consider the nonlinear stochastic partial differential equation:

$$\frac{\partial u(x,t;\omega)}{\partial t} = \frac{\partial^2 u}{\partial x^2} - \varepsilon\cdot u^2 + \sigma\cdot n(x); (x,t)\in(0,L)\times(0,\infty), \tag{10}$$

with $u(0,t)=0, u(L,t)=0$ and $u(x,0)=\phi(x)$, where $n(x)$ is the white noise process and ω is a random outcome of a triple probability space (Ω,B,P) where Ω is a sample space, B is a σ-algebra associated with Ω and P is a probability measure.

The existence of the solution as a power series in ε is proved in [24]. Searching for the Gaussian part of the solution process, $u(x,t;\omega)$ can be expanded as:

$$u(x,t;\omega) = u^{(0)}(x,t) + \int_{-\infty}^{\infty} u^{(1)}(x,t;x_1)H^{(1)}(x_1)dx_1, \tag{11}$$

Substituting in the original equation (10) and taking the necessary averages, we get the following two deterministic equations:

$$i)\ \frac{\partial u^{(0)}(x,t)}{\partial t} = \frac{\partial^2 u^{(0)}}{\partial x^2} - \varepsilon[u^{(0)}]^2 - \varepsilon \int_{-\infty}^{\infty} [u^{(1)}(x,t;x_1)]^2 dx_1,$$

$$u^{(0)}(0,t) = 0, u^{(0)}(L,t) = 0 \text{ and } u^{(0)}(x,0) = \phi(x), \tag{12}$$

$$ii)\ \frac{\partial u^{(1)}(x,t;x_1)}{\partial t} = \frac{\partial^2 u^{(1)}}{\partial x^2} - 2\varepsilon u^{(0)} u^{(1)} + \sigma\delta(x - x_1),$$

$$u^{(1)}(0,t;x_1) = 0, u^{(1)}(L,t;x_1) = 0 \text{ and } u^{(1)}(x,0;x_1) = 0, \tag{13}$$

Applying the perturbation technique, the deterministic kernels can be represented in first order approximation as:

$$u^{(0)} = u_0^{(0)} + \varepsilon u_1^{(0)}, \tag{14}$$

$$u^{(1)} = u_0^{(1)} + \varepsilon u_1^{(1)}, \tag{15}$$

Substituting in the previous set of equations (12) and (13), we get the following equations:

$$\frac{\partial u_0^{(0)}(x,t)}{\partial t} = \frac{\partial^2 u_0^{(0)}}{\partial x^2}$$

$$u_0^{(0)}(0,t) = 0, u_0^{(0)}(L,t) = 0 \text{ and } u_0^{(0)}(x,0) = \phi(x), \tag{16}$$

$$\frac{\partial u_1^{(0)}(x,t)}{\partial t} = \frac{\partial^2 u_1^{(0)}}{\partial x^2} - [u_0^{(0)}]^2 - \int_{-\infty}^{\infty} [u_0^{(1)}(x,t;x_1)]^2 dx_1,$$

$$u_1^{(0)}(0,t) = 0, u_1^{(0)}(L,t) = 0 \text{ and } u_1^{(0)}(x,0) = 0, \tag{17}$$

$$\frac{\partial u_0^{(1)}(x,t;x_1)}{\partial t} = \frac{\partial^2 u_0^{(1)}}{\partial x^2} + \sigma\delta(x - x_1)$$

$$u_0^{(1)}(0,t;x_1) = 0, u_0^{(1)}(L,t;x_1) = 0 \text{ and } u_0^{(1)}(x,0;x_1) = 0, \tag{18}$$

$$\frac{\partial u_1^{(1)}(x,t;x_1)}{\partial t} = \frac{\partial^2 u_1^{(1)}}{\partial x^2} - 2u_0^{(0)} u_0^{(1)}$$

$$u_1^{(1)}(0,t;x_1) = 0, u_1^{(1)}(L,t;x_1) = 0 \text{ and } u_1^{(1)}(x,0;x_1) = 0. \tag{19}$$

The solution is to evaluate $u_0^{(0)}$ and $u_0^{(1)}$ first using the separation of variables and the eigenfunction expansion respectively and then computing the other two kernels independently using the eigenfunction expansion. The final results are:

$$E \cdot u(x,t) = u_0^{(0)}(x,t) + \varepsilon u_1^{(0)}(x,t), \tag{20}$$

$$Var\ u(x,t) = \int_{-\infty}^{\infty} [u_0^{(1)}(x,t;x_1)]^2 dx_1 + 2\varepsilon^2 \int_{-\infty}^{\infty} [u_0^{(1)} u_1^{(1)} dx_1. \tag{21}$$

The WHEP technique can be applied on linear or non-linear perturbed systems described by ordinary or partial differential equations. The solution can be modified in the sense that additional parts of the Wiener-Hermite expansion can always be taken into considerations and the required order of approximations can always be made. It can even be implemented through a package if it is coded in some sort of symbolic languages.

3 The Homotopy Perturbation Method (HPM)

In this technique [30-33], a parameter $p \in [0,1]$ is embedded in a homotopy function $v(r,p) : \phi \times [0,1] \to \Re$ which satisfies

$$H(v,p) = (1-p)[L(v) - L(u_0)] + p[A(v) - f(r)] = 0 \tag{22}$$

where u_0 is an initial approximation to the solution of the equation

$$A(u) - f(r) = 0, r \in \phi \tag{23}$$

with boundary conditions

$$B(u, \frac{\partial u}{\partial n}) = 0, r \in \Gamma \tag{24}$$

in which A is a nonlinear differential operator which can be decompose into a linear operator L and a non-linear operator N, B is a boundary operator, $f(r)$ is a known analytic function and Γ is the boundary of ϕ. The homotopy introduces a continuously deformed solution for the case of $p = 0$, $L(v) - L(u_0) = 0$, to the case of $p = 1$, $A(v) - f(r) = 0$, which is the original equation (23). This is the basic idea of the homotopy method which is to deform continuously a simple problem (and easy to solve) into the difficult problem under study [34].

The basic assumption of the HPM method is that the solution of the original equation (23) can be expanded as a power series in p as:

$$v = v_0 + pv_1 + p^2 v_2 + p^3 v_3 + ... \tag{25}$$

Now, setting $p = 1$, the approximate solution of equation (23) is obtained as:

$$u = \lim_{p \to 1} v = v_0 + v_1 + v_2 + v_3 + ... \tag{26}$$

The rate of convergence of the method depends greatly on the initial approximation u_0 which is considered as the main disadvantage of HPM.

It has to be noted that HPM is a special case of homotopy analysis method (HAM) propounded by Liao in 1992 [35]. The HAM was systematically described

in Liao's book in 2003 [36] and was applied by many authors in [37-40]. The disadvantages of HPM can be overcome by using the HAM method that possesses auxiliary parameters and functions which can control the convergence of the obtained series solution.

The idea of the imbedded parameter can be utilized to solve non-linear problems by imbedding this parameter to the problem and then forcing it to be unity in the obtained approximate solution if converge can be assured. A simple technique which enables the extension of the applicability of the perturbation methods from small value applications to general ones.

3.1 Illustrative Example

Consider the mixed non-linear stochastic problem:

$$\ddot{x}\ (t;\omega) + w^2 x + \varepsilon_1 x^2 + \varepsilon_2 x^3 = F(t;\omega), t \in [0,T] \tag{27}$$

under stochastic excitation $F(t;\omega)$ with deterministic initial conditions

$$x(0) = x_0, \dot{x}\ (0) = \dot{x}_0\ .$$

In this case, the following data (w.r.t. homotopy perturbation) can be extracted:

$$A(x) = \ddot{x} + w^2 x + \varepsilon_1 x^2 + \varepsilon_2 x^3, L(x) = \ddot{x} + w^2 x,$$
$$N(x) = \varepsilon_1 x^2 + \varepsilon_2 x^3, f(r) = F(t;\omega).$$

The homotopy function takes the following form:

$$H(v,p) = (1-p)[L(v) - L(u_0)] + p[A(v) - f(r)] = 0.$$

or equivalently,

$$L(v) - L(u_0) + p[L(u_0) + \varepsilon_1 v^2 + \varepsilon_2 v^3 - F(t;\omega) = 0. \tag{28}$$

Letting $v = v_0 + pv_1 + p^2 v_2 + p^3 v_3 + ...$, substituting in equation (28) and equating the equal powers of p in both sides of the equation, one can get the following results:

i) $L(v_0) = L(y_0)$, in which one may consider the following simple solution:

$$v_0 = y_0 \quad y_0(0) = \dot{x}_0, \dot{y}_0\ (0) = \dot{x}_0\ .$$

ii) $L(v_1) = F(t;\omega) - L(v_0) - \varepsilon_1 v_0^2 + \varepsilon_2 v_0^3, \quad v_1(0) = 0, \dot{v}_1\ (0) = 0.$
iii) $L(v_2) = -2\varepsilon_1 v_0 v_1 - 3\varepsilon_2 v_0^2 v_1, \quad v_2(0) = 0, \dot{v}_2\ (0) = 0.$
iv) $L(v_3) = -\varepsilon_1(v_1^2 + 2v_0 v_2 - 3\varepsilon_2(v_0^2 v_2 + v_0 v_1^2), \quad v_3(0) = 0, \dot{v}_3\ (0) = 0.$
v) $L(v_4) = -2\varepsilon_1(v_0 v_3 + v_1 v_2) - \varepsilon_2(6v_0 v_1 v_2 + 3v_0^2 v_3 + v_1^3), \quad v_4(0) = 0,$
$\dot{v}_4\ (0) = 0.$

The approximate solution is

$$x(t;\omega) = \lim_{p\to 1} v = v_0 + v_1 + v_2 + v_3 + ...$$

which can be considered to any approximation order.

One can notice that the algorithm of the solution is straight forward and that many flexibilities can be made. For example, we have many choices in guessing the initial approximation together with its initial conditions. For zero initial conditions, we can choose $v_0 = 0$ which leads to:

$$\begin{aligned} x(t;\omega) \cong x_4 &= v_0 + v_1 + v_2 + v_3 + v_4 \\ &= \frac{1}{w}\int_0^t F(s;\omega)\sin w(t-s)ds - \frac{\varepsilon_1}{w}\int_0^t v_1^2(s;\omega)\sin w(t-s)ds \\ &= \frac{\varepsilon_2}{w}\int_0^t v_1^3(s;\omega)\sin w(t-s)ds \end{aligned} \tag{29}$$

that is a bad approximation as a fourth order one. In spite of this fact, the approximations can be simply modified, especially when using symbolic language manipulator , or the initial guess can be changed to get better results.

We have to notice that ε_1 and ε_2 are just deterministic scales and can take any values. Also, we can use the approximate solution formula to compute any required approximate statistical solution moments, for example the average and the covariance.

4 HPM Applied to Simultaneous Systems

The homotopy perturbation technique can be applied to a system of non-linear differential equations. Let us have the following coupled system

$$\begin{aligned} L_1(x) + N_1(x,y) &= F_1(x,y) \\ L_2(y) + N_2(x,y) &= F_2(x,y) \end{aligned} \tag{30}$$

where L_1 and L_2 are linear differential operators and N_1 and N_2 are non-linear operators. The homotopy functions can be constructed as follows:

$$H_1 = L_1(v) - L_1(\varphi_0) + p[L_1(\varphi_0) + N_1(v,u) - F_1(v,u)] = 0$$

$$H_2 = L_2(u) - L_2(\phi_0) + p[L_2(\phi_0) + N_2(v,u) - F_2(v,u)] = 0$$

Letting

$$v = v_0 + pv_1 + p^2v_2 + p^3v_3 + ... \tag{31}$$

and

$$u = u_0 + pu_1 + p^2u_2 + p^3u_3 + ... \tag{32}$$

and then substituting in the original equations (30), enables getting iterative equations in the unknowns v_i and u_i. The solutions are got as

$$\begin{aligned} x(t) &= \lim_{p\to 1} v = v_0 + v_1 + v_2 + v_3 + ... \\ y(t) &= \lim_{p\to 1} u = u_0 + u_1 + u_2 + u_3 + ... \end{aligned} \tag{33}$$

4.1 Illustrative Example

Let us solve

$$\begin{aligned} \ddot{x} + w^2 x + \varepsilon_1 y^2 &= F_1(x, y) \\ \ddot{y} + w^2 y + \varepsilon_2 x^2 &= F_2(x, y) \end{aligned} \tag{34}$$

The homotopy functions take the following form:

$$L_1(v) - L_1(\varphi_0) + p[L_1(\varphi_0) + N_1(v, u) - F_1(v, u)] = 0$$

$$L_2(u) - L_2(\phi_0) + p[L_2(\phi_0) + N_2(v, u) - F_2(v, u)] = 0 \tag{35}$$

where

$$\begin{aligned} L_1(x) &= \ddot{x} + w^2 x, \\ L_2(y) &= \ddot{y} + w^2 y, \\ N_1(x, y) &= \varepsilon_1 y^2, \\ N_2(x, y) &= \varepsilon_2 x^2, \end{aligned} \tag{36}$$

and φ_0 and ϕ_0 are arbitrary functions as before. At $p = 0$, we get $v = \varphi_0$ and $u = \phi_0$ satisfying any initial conditions in the original equations. At $p = 1$, we get the solutions of the original equations (34). Making the basic assumptions, substituting in the homotopy functions and equating the equal powers of p in both sides of the equations, we get the following iterative equations:

i) $L_1(v_0) - L_1(\varphi_0) = 0 \Rightarrow v_0 = \varphi_0, v_0(0) = x_0, \dot{v}_0\ (0) = \dot{x}_0,$
$\quad L_2(u_0) - L_1(\phi_0) = 0 \Rightarrow u_0 = \phi_0, u_0(0) = y_0, \dot{u}_0\ (0) = \dot{y}_0\ .$
ii) $L_1(v_1) + L_1(\varphi_0) + \varepsilon_1 u_0^2 = 0, v_1(0) = 0, \dot{v}_1\ (0) = 0,$
$\quad L_2(u_1) + L_2(\phi_0) + \varepsilon_2 v_0^2 = 0, u_1(0) = 0, \dot{u}_1\ (0) = 0.$
iii) $L_1(v_2) + 2\varepsilon_1 u_0 u_1 = 0, v_2(0) = 0, \dot{v}_2\ (0) = 0,$
$\quad L_2(u_2) + 2\varepsilon_2 v_0 v_1 = 0, u_2(0) = 0, \dot{u}_2\ (0) = 0.$
iv) $L_1(v_3) + \varepsilon_1(2u_0 u_2 + u_1^2) = 0, v_3(0) = 0, \dot{v}_3\ (0) = 0,$
$\quad L_2(u_3) + \varepsilon_2(2v_0 v_2 + v_1^2) = 0, u_3(0) = 0, \dot{u}_3\ (0) = 0.$

and so on.

Solving this set of equations iteratively, one can get the approximate solutions to any approximation order.

We can generalize the previous technique to n equations in the form

$$L_k(x_k) + N_k(x_1, x_2, ..., x_n) = F_k(x_1, x_2, ..., x_n), k = 1, 2, 3, ..., n \tag{37}$$

and construct the homotopy functions as

$$H_k = L_k(v_k) - L_k(\varphi_0^{(k)}) + p[L_k(\varphi_0^{(k)}) + N_k(v_1, v_2, ..., v_n)$$

$$-F_k(v_1, v_2, ..., x_n)] = 0, k = 1, 2, ..., n \tag{38}$$

The basic assumptions are

$$v_k = v_k^{(0)} + pv_k^{(1)} + p^2 v_k^{(2)} + p^3 v_k^{(3)} + ..., k = 1, 2, ..., n \tag{39}$$

and still

$$x_k = \lim_{p \to 1} v_k = v_k^{(0)} + v_k^{(1)} + v_k^{(2)} + v_k^{(3)} + ..., k = 1, 2, ..., n \tag{40}$$

where

$$v_k^{(0)} = x_k(0), \dot{v}_k^{(0)} = \dot{x}_k\ (0) \forall k = 1, 2, ..., n; \quad v_k^{(j)} = 0, \dot{v}_k^{(j)} = 0 \forall j = 1, 2, ..., n.$$

5 The Homotopy WHEP Technique

One of the major disadvantages of the WHEP technique is solving only perturbed problems, i.e. the problem should contain a small parameter. This disadvantage can be faced by using the HPM instead of the conventional perturbation methods. Let us call the new link between Wiener-Hermite expansion and homotopy perturbation technique, the homotopy WHEP.

The algorithm of homotopy WHEP is charted in Fig. 1. Following the flowchart in Fig. 1, the WH expansion is applied on the stochastic non-linear (or linear) differential equation where a complex stochastic integro-differential equation is obtained. We can call this step as the "horizontal expansion" where it is truncated up to the required order of approximation. Taking the necessary set of averages of this equation, we get a set of deterministic integro-differential equations in the deterministic kernels of the WH expansion of the solution process using the statistical properties of these polynomials. Applying HPM to the previous system of equations and constructing the necessary homotopy functions, one can get an iterative set of equations in the deterministic terms of the basic assumptions. This step can be called "vertical expansion" where an order of correction to each order of approximation can be specified. The approximate solutions of these deterministic terms constitute the approximate solution process to the required order of approximation and correction.

Consequently, the required statistical moments of the solution process are computed according to their statistical definitions and according to the required

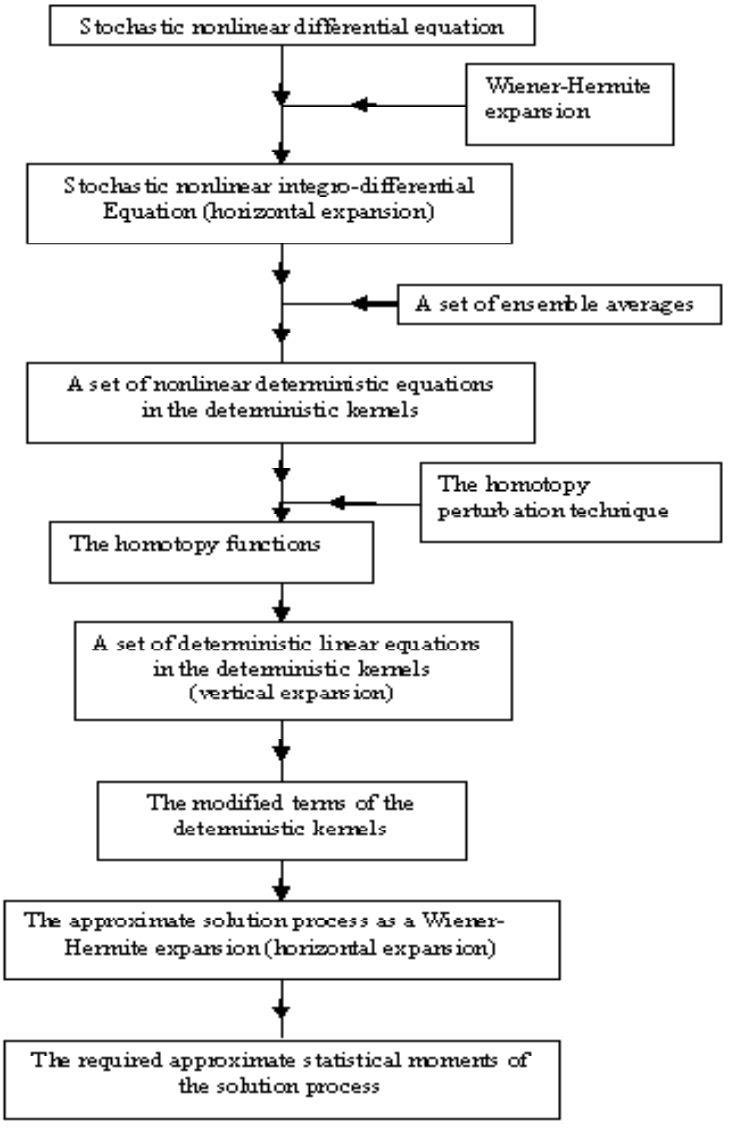


Fig. 1. The homotopy WHEP algorithm

order of approximation and correction. This algorithm provides a systematic way for solving stochastic differential equations when convergence is assured.

5.1 Illustrative Case-Study: The Quadratic Nonlinearity

Let us solve

$$\ddot{x} + 2w\xi\,\dot{x} + w^2 x + \varepsilon w^2 x^2 = G(t;\omega) \tag{41}$$

with the deterministic initial conditions $x(0) = x_0, \dot{x}\,(0) = \dot{x}_0$. Let

$$G(t;\omega) = G^{(0)}(t) + \int_{-\infty}^{\infty} G^{(1)}(t;t_1)H^{(1)}(t_1)dt_1 + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2 + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3 + \ldots \quad (42)$$

where $G^{(3)}(-)$ are known kernels (since $G(t;\omega)$ is a known function) and

$$x(t;\omega) = x^{(0)}(t) + \int_{-\infty}^{\infty} x^{(1)}(t;t_1)H^{(1)}(t_1)dt_1 + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2 + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3 + \ldots \quad (43)$$

where $x^{(k)}(-)$ are unknown deterministic kernels.

Applying equations (42) and (43) in equation (41), we get

$$Lx^{(0)}(t) + \int_{-\infty}^{\infty} Lx^{(1)}(t;t_1)H^{(1)}(t_1)dt_1 + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} Lx^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2 + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} Lx^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3 + \ldots$$

$$+\varepsilon w^2\Big[\Big[x^{(0)}(t)\Big]^2 + \left[\int_{-\infty}^{\infty} x^{(1)}(t;t_1)H^{(1)}(t_1)dt_1\right]^2 + \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2\right]^2 + \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3\right]^2 + \ldots$$

$$+2x^{(0)}(t)\int_{-\infty}^{\infty} x^{(1)}(t;t_1)H^{(1)}(t_1)dt_1 + 2x^{(0)}(t)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2$$

$$+2x^{(0)}(t)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}x^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3+...$$

$$+2\int_{-\infty}^{\infty}x^{(1)}(t;t_1)H^{(1)}(t_1)dt_1\cdot\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}x^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2$$

$$+2\int_{-\infty}^{\infty}x^{(1)}(t;t_1)H^{(1)}(t_1)dt_1\cdot\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}x^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3+...$$

$$+2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}x^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2$$

$$\cdot\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}x^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3+...]$$

$$=G^{(0)}(t)+\int_{-\infty}^{\infty}G^{(1)}(t;t_1)H^{(1)}(t_1)dt_1+\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}G^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2$$

$$+\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}G^{(3)}(t;t_1,t_2,t_3)H^{(3)}(t_1,t_2,t_3)dt_1dt_2dt_3+... \tag{44}$$

which is a stochastic integro-differential equation in the deterministic kernels $x^{(k)}(-)$. The linear operator L is defined as

$$L(x)=\ddot{x}+2w\xi\,\dot{x}+w^2x \tag{45}$$

Performing the direct average of equation (44) and using the statistical properties of the Wiener-Hermite polynomials [35], equation (2) and others, we get

$$Lx^{(0)}(t)+\varepsilon w^2[\left[x^{(0)}(t)\right]^2+\int_{-\infty}^{\infty}\left[x^{(1)}(t;t_1)\right]^2dt_1+2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[x^{(2)}(t;t_1,t_2)\right]^2dt_1dt_2$$

$$+2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}x^{(3)}(t;t_1,t_2,t_3)x^{(3)}(t;t_2,t_3,t_1)dt_1dt_2dt_3$$

$$+2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}x^{(3)}(t;t_1,t_2,t_3)x^{(3)}(t_1,t_3,t_2)dt_1dt_2dt_3$$

$$+\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[x^{(3)}(t;t_1,t_2,t_3)\right]^2dt_1dt_2dt_3]=G^{(0)}(t) \tag{46}$$

Multiplying equation (44) by $H^{(1)}(t_7)$, taking the average with using the statistical properties of Wiener-Hermite polynomials [35] and letting $t_7 \to t_1$, we get

$$Lx^{(1)}(t,t_1) + \varepsilon w^2[2x^{(0)}(t)x^{(1)}(t,t_1) + 4\int_{-\infty}^{\infty} x^{(1)}(t;t_1)x^{(2)}(t;t_1,t_2)dt_1dt_2$$

$$+4\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(2)}(t;t_2,t_3)[2x^{(3)}(t;t_1,t_2,t_3)$$

$$+x^{(3)}(t;t_2,t_3,t_1)dt_2dt_3] = G^{(1)}(t,t_1) \tag{47}$$

Multiplying equation (44) by $H^{(2)}(t_7,t_8)$, taking the average with using the statistical properties of Wiener-Hermite polynomials [35] and letting $t_7 \to t_1$, $t_8 \to t_2$, we get

$$Lx^{(2)}(t,t_1,t_2) + \varepsilon w^2[2x^{(0)}(t)x^{(2)}(t,t_1,t_2) + x^{(1)}(t,t_1)x^{(1)}(t,t_2)$$
$$+4\int_{-\infty}^{\infty} x^{(2)}(t;t_1,t_3)x^{(2)}(t;t_2,t_3)dt_3 + 2\int_{-\infty}^{\infty} x^{(1)}(t;t_3)x^{(3)}(t;t_1,t_3,t_2)dt_3$$
$$+2\int_{-\infty}^{\infty} x^{(1)}(t;t_3)x^{(3)}(t;t_2,t_3,t_1)dt_3 + 2\int_{-\infty}^{\infty} x^{(1)}(t;t_3)x^{(3)}(t;t_1,t_2,t_3)dt_3$$

$$+2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_2,t_5,t_6)x^{(3)}(t;t_1,t_6,t_5)dt_5dt_6$$

$$+4\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_2,t_5,t_6)x^{(3)}(t;t_1,t_5,t_6)dt_5dt_6$$

$$+3\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_2,t_5,t_6)x^{(3)}(t;t_5,t_6,t_1)dt_5dt_6$$

$$+2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_1,t_5,t_6)x^{(3)}(t;t_2,t_6,t_5)dt_5dt_6$$

$$+3\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_1,t_5,t_6)x^{(3)}(t;t_5,t_6,t_2)dt_5dt_6$$

$$+\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_1,t_6,t_5)x^{(3)}(t;t_5,t_6,t_2)dt_5dt_6$$

$$+2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_5,t_6,t_2)x^{(3)}(t;t_5,t_6,t_1)dt_5dt_6$$
$$+\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(3)}(t;t_2,t_6,t_5)x^{(3)}(t;t_5,t_6,t_1)dt_5dt_6]$$

$$= G^{(2)}(t,t_1,t_2) \tag{48}$$

Multiplying equation (44) by $H^{(3)}(t_7,t_8,t_9)$, taking the average with using the statistical properties of Wiener-Hermite polynomials [41] and letting $t_7 \to t_1$, $t_8 \to t_2, t_9 \to t_3$, we get

$$Lx^{(3)}(t,t_1,t_2,t_3) + Lx^{(3)}(t,t_1,t_3,t_2) + Lx^{(3)}(t,t_2,t_3,t_1)$$
$$+\varepsilon w^2[2x^{(0)}(t)[x^{(3)}(t,t_1,t_2,t_3) + x^{(3)}(t,t_1,t_3,t_2) + x^{(3)}(t,t_2,t_3,t_1)]$$
$$+4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_1)x^{(3)}(t;t_4,t_3,t_2)dt_4 + 4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_1)x^{(3)}(t;t_4,t_2,t_3)dt_4$$
$$+4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_2)x^{(3)}(t;t_4,t_3,t_1)dt_4 + 4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_2)x^{(3)}(t;t_4,t_1,t_3)dt_4$$
$$+4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_3)x^{(3)}(t;t_4,t_1,t_2)dt_4 + 4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_3)x^{(3)}(t;t_4,t_2,t_1)dt_4$$
$$+4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_1)x^{(3)}(t;t_3,t_2,t_4)dt_4 + 4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_2)x^{(3)}(t;t_3,t_1,t_4)dt_4$$
$$+4\int_{-\infty}^{\infty} x^{(2)}(t;t_4,t_3)x^{(3)}(t;t_1,t_2,t_4)dt_4]$$

$$= G^{(3)}(t,t_1,t_2,t_3) + G^{(3)}(t,t_1,t_3,t_2) + G^{(3)}(t,t_2,t_3,t_1) \tag{49}$$

5.1.1 Illustrative Example-1

Let us take the simple case of evaluating the only Gaussian part of the solution process of the previous case study, mainly

$$x(t;\omega) = x^{(0)}(t) + \int_{-\infty}^{\infty} x^{(1)}(t;t_1)H^{(1)}(t_1)dt_1 \tag{50}$$

In this case, the governing equations are

$$Lx^{(0)}(t) + \varepsilon w^2[\left[x^{(0)}(t)\right]^2 + \int_{-\infty}^{\infty}\left[x^{(1)}(t;t_1)\right]^2 dt_1] = G^{(0)}(t) \tag{51}$$

$$Lx^{(1)}(t,t_1) + \varepsilon w^2 2x^{(0)}(t)x^{(1)}(t,t_1) = G^{(1)}(t,t_1) \tag{52}$$

The ensemble average is

$$\mu_x(t) = x^{(0)}(t) \tag{53}$$

and the variance is

$$\sigma_x^2(t) = \int_{-\infty}^{\infty} \left[x^{(1)}(t;t_1)\right]^2 dt_1 \tag{54}$$

Now, using the homotopy perturbation in solving the governing equations (51) and (52) respectively and using the results of section 4., we get the following homotopy functions

$$\begin{aligned} &Lx_0^{(0)}(t) + pLx_1^{(0)}(t) + p^2Lx_2^{(0)}(t) + p^3Lx_3^{(0)}(t) + ... - L(\varphi_0) \\ &+p[L(\varphi_0) + \varepsilon w^2[x_0^{(0)}(t) + px_1^{(0)}(t) + p^2x_2^{(0)}(t) + p^3x_3^{(0)}(t) + ...]^2] \end{aligned}$$

$$+p[\varepsilon w^2 \int_{-\infty}^{\infty} [x_0^{(1)}(t;t_1) + px_1^{(1)}(t;t_1) + p^2x_2^{(1)}(t;t_1) + ...]^2 dt_1 - G^{(0)}(t)] = 0 \tag{55}$$

$$\begin{aligned} &Lx_0^{(1)}(t,t_1) + pLx_1^{(1)}(t,t_1) + p^2Lx_2^{(1)}(t,t_1) + p^3Lx_3^{(1)}(t,t_1) + ... - L(\phi_0) \\ &+p[L(\varphi_0) + 2\varepsilon w^2[x_0^{(0)}(t) + px_1^{(0)}(t) + p^2x_2^{(0)}(t) + p^3x_3^{(0)}(t) + ...] \end{aligned}$$

$$[x_0^{(1)}(t,t_1) + px_1^{(1)}(t,t_1) + p^2x_2^{(1)}(t,t_1) + p^3x_3^{(1)}(t,t_1) + ...] - pG^{(1)}(t,t_1)] = 0 \tag{56}$$

Equating the equal powers of p in both sides of equations (55) and (56), we get the following pairs as iterative equations:

i) $Lx_0^{(0)}(t) - L(\varphi_0) = 0 \Rightarrow x_0^{(0)}(t) = \varphi_0, x_0^{(0)}(0) = \varphi_0(0) = x_0,$
$\dot{x}_0^{(0)}(0) = \dot{\varphi}_0(0) = \dot{x}_0,$
$Lx_0^{(1)}(t,t_1) - L(\phi_0) = 0 \Rightarrow x_0^{(1)}(t,t_1) = \phi_0, x_0^{(1)}(0,t_1) = \phi_0(0,t_1) = 0,$
$\dot{x}_0^{(1)}(0,t_1) = \dot{\phi}_0(0,t_1) = 0$

ii) $Lx_1^{(0)}(t) + L(\varphi_0) + \varepsilon w^2 \left[x_0^{(0)}(t)\right]^2 + \varepsilon w^2 \int_{-\infty}^{\infty} \left[x_0^{(1)}(t;t_1)\right]^2 dt_1 = G^{(0)}(t),$
$x_1^{(0)}(0) = \dot{x}_0^{(0)}(0) = 0,$
$Lx_1^{(1)}(t,t_1) + L\varphi_0(t,t_1) + 2\varepsilon w^2 x_0^{(0)}(t)x_0^{(1)}(t,t_1) = G^{(1)}(t,t_1),$
$x_1^{(1)}(0,t_1) = \dot{x}_1^{(1)}(0,t_1) = 0$

iii) $Lx_2^{(0)}(t) + 2\varepsilon w^2 x_0^{(0)}(t)x_1^{(0)}(t) + 2\varepsilon w^2 \int_{-\infty}^{\infty} \left[x_0^{(1)}(t;t_1)x_1^{(1)}(t;t_1)\right] dt_1 = 0,$
$x_2^{(0)}(0) = \dot{x}_2^{(0)}(0) = 0,$
$Lx_2^{(1)}(t,t_1) + 2\varepsilon w^2[x_0^{(0)}(t)x_1^{(1)}(t,t_1) + x_1^{(0)}(t)x_0^{(1)}(t,t_1)] = 0,$
$x_2^{(1)}(0,t_1) = \dot{x}_2^{(1)}(0,t_1) = 0,$

$$\begin{aligned}
&\text{iv) } Lx_3^{(0)}(t) + \varepsilon w^2[2x_0^{(0)}(t)x_2^{(0)}(t) + \left[x_1^{(0)}(t)\right]^2] \\
&\quad +\varepsilon w^2 \int_{-\infty}^{\infty} \left[2x_0^{(1)}(t;t_1)x_2^{(1)}(t;t_1) + \left[x_1^{(1)}(t;t_1)\right]^2\right] dt_1 = 0, \\
&\qquad x_3^{(0)}(0) = \dot{x}_3^{(0)}(0) = 0, \\
&Lx_3^{(1)}(t,t_1) + 2\varepsilon w^2[x_0^{(0)}(t)x_2^{(1)}(t,t_1) + x_1^{(0)}(t)x_1^{(1)}(t,t_1) + x_2^{(0)}(t)x_0^{(1)}(t,t_1)] = 0, \\
&\qquad x_3^{(1)}(0,t_1) = \dot{x}_3^{(1)}(0,t_1) = 0.
\end{aligned}$$

and so on, to any required order of correction. We can call this procedure as the vertical expansion.

It has to be noticed that all previous equations are deterministic linear ones in the general form $\ddot{x} + 2w\xi\,\dot{x} + w^2 x = F(t)$ with deterministic initial conditions $x(0) = x_0, \dot{x}(0) = \dot{x}_0$which has the general solution

$$x(t) = x_0\phi_1(t) + \dot{x}_0\,\phi_2 + \int_0^t h(t-s)F(s)ds = 0, \tag{57}$$

in which we have

$$\begin{aligned}
h(t) &= \frac{1}{w\sqrt{1-\xi^2}} e^{-w\xi_1} \sin w\sqrt{1-\xi^2}t, \\
\phi_1(t) &= \frac{\xi + \sqrt{\xi^2-1}}{2\sqrt{\xi^2-1}} e^{mt} + \frac{-\xi + \sqrt{\xi^2-1}}{2\sqrt{\xi^2-1}} e^{qt} \\
\phi_2(t) &= \frac{1}{2w\sqrt{\xi^2-1}}[e^{mt} - e^{qt}],
\end{aligned}$$

where $m = -w\xi + w\sqrt{\xi^2-1}, q = -w\xi - w\sqrt{\xi^2-1}$.

Now, we can compute the average and variance of the solution process using equations (53) and (54) respectively. The solution process is assumed to be Gaussian, i.e. $x(t) - N(\mu_x(t), \sigma_x^2(t))$. This is the first approximation of the solution process. Now, we can go forward to modify the obtained results via the enhancement of the non-Gaussian part of the solution process, mainly computing $x^{(2)}(t,t_1,t_2)$ and other required deterministic kernels of the Wiener-Hermite expansion of the solution process, we can call this procedure as the horizontal expansion.

5.1.2 Illustrative Example-2

Let us take the first term in the non-Gaussian part of the solution process of the previous case study, mainly

$$\begin{aligned}
x(t;\omega) = x^{(0)}(t) &+ \int_{-\infty}^{\infty} x^{(1)}(t;t_1)H^{(1)}(t_1)dt_1 \\
&+ \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{(2)}(t;t_1,t_2)H^{(2)}(t_1,t_2)dt_1dt_2
\end{aligned} \tag{58}$$

In this case, the governing equations are

$$Lx^{(0)}(t) + \varepsilon w^2[\left[x^{(0)}(t)\right]^2 + \int_{-\infty}^{\infty} [x^{(1)}(t;t_1)]^2 dt_1]$$
$$+2\varepsilon w^2 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} [x^{(2)}(t;t_1,t_2)]^2 dt_1 dt_2 = G^{(0)}(t) \tag{59}$$

$$Lx^{(1)}(t,t_1) + 2\varepsilon w^2 x^{(0)}(t)x^{(1)}(t,t_1)$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty} x^{(1)}(t;t_2)x^{(2)}(t;t_1,t_2)dt_2 = G^{(1)}(t,t_1) \tag{60}$$

$$Lx^{(2)}(t,t_1,t_2) + \varepsilon w^2[2x^{(0)}(t)x^{(2)}(t,t_1,t_2) + x^{(1)}(t,t_1)x^{(1)}(t,t_2)$$
$$+4 \int_{-\infty}^{\infty} x^{(2)}(t;t_1,t_3)x^{(2)}(t;t_2,t_3)dt_3] = G^{(2)}(t,t_1,t_2) \tag{61}$$

The ensemble average is still

$$\mu_x(t) = x^{(0)}(t) \tag{62}$$

while the variance is

$$\sigma_x^2(t) = \int_{-\infty}^{\infty} [x^{(1)}(t;t_1)]^2 dt_1 + 2 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} [x^{(2)}(t;t_1,t_2)]^2 dt_1 dt_2 \tag{63}$$

Following the same algorithm achieved in the previous example, the following computational algorithm is obtained:

i) $Lx_0^{(0)}(t) - L(\varphi_0) = 0 \Rightarrow x_0^{(0)}(t) = \varphi_0, x_0^{(0)}(0) = \varphi_0(0) = x_0,$
$\dot{x}_0^{(0)}(0) = \dot{\varphi}_0(0) = \dot{x}_0,$
$Lx_0^{(1)}(t,t_1) - L(\phi_0) = 0 \Rightarrow x_0^{(1)}(t,t_1) = \phi_0, x_0^{(1)}(0,t_1) = \phi_0(0,t_1) = 0,$
$\dot{x}_0^{(1)}(0,t_1) = \dot{\phi}_0(0,t_1) = 0$
$Lx_0^{(2)}(t,t_1,t_2) - L(\gamma_0) = 0 \Rightarrow x_0^{(2)}(t,t_1,t_2) = \gamma_0,$
$x_0^{(2)}(0,t_1,t_2) = \gamma_0(0,t_1,t_2) = 0, \dot{x}_0^{(2)}(0,t_1,t_2) = \dot{\gamma}_0(0,t_1,t_2) = 0.$
ii) $Lx_1^{(0)}(t) + L(\varphi_0) + \varepsilon w^2\left[x_0^{(0)}(t)\right]^2 + \varepsilon w^2 \int_{-\infty}^{\infty} \left[x_0^{(1)}(t;t_1)\right]^2 dt_1$
$+2\varepsilon w^2 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} [x_0^{(2)}(t;t_1,t_2)]^2 dt_1 dt_2 = G^{(0)}(t), x_1^{(0)}(0) = \dot{x}_1^{(0)}(0) = 0,$

$$Lx_1^{(1)}(t,t_1)+L\varphi_0(t,t_1)+2\varepsilon w^2 x_0^{(0)}(t)x_0^{(1)}(t,t_1)+4\varepsilon w^2 \int_{-\infty}^{\infty} x_0^{(1)}(t;t_2)x_0^{(2)}(t;t_1,t_2)dt_1dt_2$$
$$= G^{(1)}(t,t_1), x_1^{(1)}(0,t_1) = \dot{x}_1^{(1)}(0,t_1) = 0.$$

$$Lx_1^{(2)}(t,t_1,t_2)+L\gamma_0(t,t_1,t_2)+\varepsilon w^2 x_0^{(1)}(t,t_1)x_0^{(1)}(t,t_2)+2\varepsilon w^2 x_0^{(0)}(t)x_0^{(2)}(t;t_1,t_2)$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty} x_0^{(2)}(t;t_1,t_3)x_0^{(2)}(t;t_2,t_3)dt_3 = G^{(2)}(t,t_1,t_2),$$
$$x_1^{(2)}(0,t_1,t_2) = \dot{x}_1^{(2)}(0,t_1,t_2) = 0.$$

iii) $$Lx_2^{(0)}(t) + 2\varepsilon w^2 x_0^{(0)}(t)x_1^{(0)}(t) + 2\varepsilon w^2 \int_{-\infty}^{\infty} \left[x_0^{(1)}(t;t_1)x_1^{(1)}(t;t_1)\right] dt_1$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x_0^{(2)}(t;t_1,t_2)x_1^{(2)}(t;t_1,t_2)dt_1dt_2 = 0, x_2^{(0)}(0) = \dot{x}_2^{(0)}(0) = 0,$$

$$Lx_2^{(1)}(t,t_1) + 2\varepsilon w^2[x_0^{(0)}(t)x_1^{(1)}(t,t_1) + x_1^{(0)}(t)x_0^{(1)}(t,t_1)]$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty} [x_0^{(1)}(t;t_2)x_1^{(2)}(t;t_1,t_2) + x_1^{(1)}(t;t_2)x_0^{(2)}(t;t_1,t_2)]dt_2 = 0,$$
$$x_2^{(1)}(0,t_1) = \dot{x}_2^{(1)}(0,t_1) = 0.$$

$$Lx_2^{(2)}(t,t_1,t_2) + \varepsilon w^2[x_0^{(1)}(t,t_1)x_1^{(1)}(t,t_2) + x_1^{(1)}(t,t_1)x_0^{(1)}(t,t_2)]$$
$$+2\varepsilon w^2[x_0^{(0)}(t)x_1^{(2)}(t,t_1,t_2) + x_1^{(0)}(t)x_0^{(2)}(t,t_1,t_2)]$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty} x_0^{(2)}(t;t_1,t_3)x_1^{(2)}(t;t_2,t_3)dt_3$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty} x_1^{(2)}(t;t_1,t_3)x_0^{(2)}(t;t_2,t_3)dt_3 = 0, x_2^{(2)}(0,t_1,t_2)$$
$$= \dot{x}_2^{(2)}(0,t_1,t_2) = 0.$$

iv) $$Lx_3^{(0)}(t) + \varepsilon w^2[2x_0^{(0)}(t)x_2^{(0)}(t)\left[x_1^{(0)}(t)\right]^2]$$
$$+\varepsilon w^2 \int_{-\infty}^{\infty} \left[2x_0^{(1)}(t;t_1)x_2^{(1)}(t;t_1) + \left[x_1^{(1)}(t;t_1)\right]^2\right] dt_1$$
$$+2\varepsilon w^2 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} [2x_0^{(2)}(t;t_1,t_2)x_2^{(2)}(t;t_1,t_2) + [x_1^{(2)}(t;t_1,t_2)]^2 dt_1dt_2 = 0,$$
$$x_3^{(0)}(0) = \dot{x}_3^{(0)}(0) = 0,$$

$$Lx_3^{(1)}(t,t_1) + 2\varepsilon w^2[x_0^{(0)}(t)x_2^{(1)}(t,t_1) + x_1^{(0)}(t)x_1^{(1)}(t,t_1) + x_2^{(0)}(t)x_0^{(1)}(t,t_1)]$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty} [x_0^{(1)}(t;t_2)x_2^{(2)}(t;t_1,t_2) + x_1^{(1)}(t;t_2)x_1^{(2)}(t;t_1,t_2)$$
$$+x_2^{(1)}(t;t_2)x_0^{(2)}(t;t_1,t_2)dt_2 = 0, x_3^{(1)}(0,t_1) = \dot{x}_3^{(1)}(0,t_1) = 0.$$

$$Lx_3^{(2)}(t,t_1,t_2)+\varepsilon w^2[x_0^{(1)}(t,t_1)x_2^{(1)}(t,t_1)+x_1^{(1)}(t,t_1)x_1^{(1)}(t,t_2)]+x_2^{(1)}(t,t_1)x_0^{(1)}(t,t_2)]$$
$$+2\varepsilon w^2[x_0^{(0)}(t)x_2^{(2)}(t,t_1,t_2) + x_1^{(0)}(t)x_1^{(2)}(t,t_1,t_2) + x_2^{(0)}(t)x_0^{(2)}(t,t_1,t_2)]$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty} [x_0^{(2)}(t;t_1,t_3)x_2^{(2)}(t;t_2,t_3) + x_1^{(2)}(t;t_1,t_3)x_1^{(2)}(t;t_2,t_3)dt_3$$
$$+4\varepsilon w^2 \int_{-\infty}^{\infty} x_2^{(2)}(t;t_1,t_3)x_0^{(2)}(t;t_2,t_3)dt_3 = 0,$$
$$x_3^{(2)}(0,t_1,t_2) = \dot{x}_3^{(2)}(0,t_1,t_2) = 0$$

and so on, to any required order of correction.

6 Discussion and Conclusions

In previous works, the WHEP technique proved to be successful in introducing an approximate solution to some perturbed stochastic differential equations. In this paper, the use of homotopy perturbation instead of the conventional perturbation theory enlarges the use of WHEP which is now called homotopy WHEP. The approximate solution is obtained through systematic procedures and can be improved through two main ways, horizontally through the addition of more terms in Wiener-Hermite expansion to obtain a modified solution, and vertically through improving this modified solution via the homotopy perturbation technique. The technique transforms the stochastic non-linear equation into a set of deterministic linear equations which can be easily solved iteratively to any order of approximation through horizontal or vertical ways. The huge developments in symbolic computations will facilitate the good use of such algorithms.

The use of the homotopy WHEP technique is better than the use of perturbation method alone since the latter produces a set of stochastic linear equations that is more complex than the original one. The WH expansion is known to be convergent but the application of HPM may lead to divergence. This disadvantage is overcome by using homotopy analysis method (HAM) which can be studied in future works.

References

[1] Crow, S., Canavan, G.: Relationship Between a Wiener-Hermite Expansion and an Energy Cascade. J. of Fluid Mechanics. 41(2), 387–403 (1970)
[2] Saffman, P.: Application of Wiener-Hermite Expansion to the Diffusion of a Passive Scalar in a Homogeneous Turbulent Flow. Physics of Fluids 12(9), 1786–1798 (1969)
[3] Kahan, W., Siegel, A.: Cameron-Martin-Wiener Method in Turbulence and in Burger's Model: General Formulae and Application to Late Decay. J. of Fluid Mechanics. 41(3), 593–618 (1970)
[4] Wang, J., Shu, S.: Wiener-Hermite Expansion and the Inertial Subrange of a Homogeneous Isotropic Turbulence. Physics of Fluids 17(6) (June 1974)
[5] Hogge, H., Meecham, W.: Wiener-Hermite Expansion Applied to Decaying Isotropic Turbulence Using a Renormalized Time-Dependent Base. J. of Fluid of Mechanics. 85(2), 325–347 (1978)
[6] Doi, Masaaki, Imamura, Tsutomu: Exact Gaussian Solution for Two-Dimensional Incompressible Inviscid Turbulent Flow. J. of the Physical Society of Japan 46(4), 1358–1359 (1979)
[7] Kambe, Ryouichi, Doi, Masaaki, Imamura, Tsutomu: Turbulent Fows Near Flat Plates. J. of the Physical Society of Japan 49(2), 763–778 (1980)
[8] Chorin, Alexandre, J.: Gaussian Fields and Random Flow. J. of Fuid of Mechanics 63(1), 21–32 (1974)
[9] Kayanuma, Yosuka: Stochastic Theory for Non-adiabatic Crossing With Fluctuating Off-Diagonal Coupling. J. of the Physical Society of Japan 54(5), 2037–2046 (1985)

[10] Joelson, M., Ramamonjiarisoa, A.: Random Fields of Water Surface Waves Using Wiener-Hermite Functional Series Expansions. J. of Fluid of Mechanics 496, 313–334 (2003)

[11] Eftimiu, Cornel: First-Order Wiener-Hermite Expansion in the Electromagnetic Scattering by Conducting Rough Surfaces. Radio Science 23(5), 769–779 (1988)

[12] Gaol, Nakayama, J.: Scattering of a TM Plane Wave From Periodic Random Surfaces. Waves Random Media 9(11), 53–67 (1999)

[13] Tamura, Y., Nakayama, J.: Enhanced Scattering From a Thin Film With One-Dimensional Disorder. Waves in Random and Complex Media 15(2), 269–295 (2005)

[14] Tamura, Y., Nakayama, J.: TE Plane Wave Reflection and Transmission From One-Dimensional Random Slab. IEICE Transactions on Electronics E88-C(4), 713–720 (2005)

[15] Skaropoulos, N., Chrissoulidis, D.: Rigorous Application of the Stochastic Functional Method to Plane Wave Scattering From a Random Cylindrical Surface. J. of mathematical physics 40(1), 156–168 (1999)

[16] Jahedi, A., Ahmadi, G.: Application of Wiener-Hermite Expansion to Non-Stationary Random Vibration of a Duffing Oscillator. J. of Applied Mechanics. Transactions ASME 50(2), 436–442 (1983)

[17] Orabi, Ismail, I., Ahmadi, A., Goodarz: Functional Series Expansion Method for Response Analysis of Nonlinear Systems Subjected to Ransom Excitations. Int. J. of Nonlinear Mechanics 22(6), 451–465 (1987)

[18] Orabi, Ismail, I.: Response of the Duffing Oscillator to a Non-Gaussian Random Excitation. J. of Applied Mechanics. Transaction of ASME 55(3), 740–743 (1988)

[19] Gawad, E.A., El-Tawil, M., Nassar, M.A.: Nonlinear Oscillatory Systems With Random Excitation. Modeling, Simulation and Control -B 23(1), 55–63 (1989)

[20] Orabi,, Ismail, I., Ahmadi, A., Goodarz,: New Approach for Response Analysis of Nonlinear Systems under Random Excitation. American Society of Mechanical Engineers. Design Engineering Division E 37, 147–151 (1991) (publication)

[21] Gawad, E., El-Tawil, M.: General Stochastic Oscillatory Systems. Applied Mathematical Modeling 17(6), 329–335 (1993)

[22] El-Tawil, M., Mahmoud, G.: The Solvability of Parametrically Forced Oscillators Using WHEP Technique. Mechanics and Mechanical Engineering 3(2), 181–188 (1999)

[23] Tamura, Y., Nakayama, J.: A Formula on the Hermite Expansion and its Application to a Random Boundary Value Problem. IEICE Transactions on Eectronics E86-C(8), 1743–1748 (2003)

[24] El-Tawil M.: The application of WHEP technique on Stochastic partial Differential Equations. Int. J. of Differential Equations and Applications 7(3), 325-337 (2003).

[25] Kayanuma, Y., Noba, K.: Wiener-Hermite Expansion Formalism for the Stochastic Model of a driven Quantum System. Chemical physics 268(1–3), 177–188 (2001)

[26] Kenny, O., Nelson, D.: Time-Frequency Methods for Enhancing Speech. In: Proceedings of SPIE- the International Society for Optical Engineering, vol. 3162, pp. 48–57 (1997)

[27] Isobe, Etsu, Sato, Shunsuke: Wiener-Hermite Expansion of a Process Generated by an Ito Stochastic Differential Equations. J. of Applied Probability 20(4), 754–765 (1983)

[28] Rubinstein, R., Choudhari, M.: Uncertainty Quantification for Systems With Random Initial Conditions Using Wiener-Hermite Expansions. Studies in Applied Mathematics 114(2), 167–188 (2005)
[29] Imamura, T., Meecham, W., Siegel, A.: Symbolic Calculus of the Wiener Process and Wiener-Hermite Functionals. J. Math. Phys. 6(5), 695–706 (1983)
[30] He, J.H.: Homotopy Perturbation Technique. Comput. Methods Appl. Mech. Engrg. 178, 257–292 (1999)
[31] He, J.H.: A Coupling Method of a Homotopy Technique and a Perturbation Technique for Nonlinear Problems. Int. J. of Nonlinear Mechanics. 35, 37–43 (2000)
[32] He, J.H.: Homotopy Perturbation Method: a New Nonlinear Analytical Technique. Applied Math. and Computations 135, 73–79 (2003)
[33] He, J.H.: The Homotopy Perturbation Method for Nonlinear Oscillators With Discontinuities. Applied Math. and Computations 151, 287–292 (2004)
[34] He, J.H.: Some Asymptotic Methods for Strongly Nonlinear Equations. Intern J. of Modern Physics B 20(10), 1141–1199 (2006)
[35] Liao, S.J.: On the Proposed Homotopy Analysis Techniques for Nonlinear Problems and its Applications, Ph.D. dissertation. Shanghai JiaoTong University (1992)
[36] Liao, S.J.: Beyond Perturbation:Introduction to the Homotopy Analysis Method. Chapman & Hall/CRC press, Boca Raton (2003)
[37] Liao, S.J.: An Approximate Solution Technique Which Does not Depend Upon Small Parameters: A Special Example. Intern. J. of Nonlinear Mechanics 30, 371–380 (1995)
[38] Hayat, T., Khan, M., Asghar, S.: Homotopy Analysis of MHD Flows of an Oldroyd 8-Constant fluid. Acta Mechanica 168, 213–232 (2004)
[39] Asghar, S., Mudassar, M., Ayub, M.: Rotating Flow of a Third Fluid by Homotopy Analysis Method. Appl. Math. and Computation 165, 213–221 (2005)
[40] Zhu, S.P.: A Closed Form Analytical Solution for the Valuation of Convertible Bonds with Constant Dividend Yield. ANZIAM J. 47(4), 477–494 (2006)
[41] El-Tawil, M.A.: Solution Processes of a Class of Stochastic Differential Equations, Ph.D. Dissertation, Engineering Mathematics Dept., Faculty of Engineering. Cairo University (1989)

Author Index