KUOPION YLIOPISTON JULKAISUJA C. LUONNONTIETEET JA YMPÄRISTÖTIETEET 167
KUOPIO UNIVERSITY PUBLICATIONS C. NATURAL AND ENVIRONMENTAL SCIENCES 167

MIKKO T. KOLEHMAINEN

Data exploration with self-organizing maps in environmental informatics and bioinformatics

Thesis for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Computer Science and Engineering for public examination and debate in Auditorium AS1 (TUAS, Otaniementie 17) at Helsinki University of Technology (Espoo, Finland) on Friday 27th February 2004, at 12 noon.

Laboratory of Computer and Information Science
Department of Computer Science and Engineering
Helsinki University of Technology

KUOPION YLIOPISTO
KUOPIO 2004

Distributor: Kuopio University Library
P.O. Box 1627
FIN-70211 KUOPIO
FINLAND
Tel. +358 17 163 430
Fax +358 17 163 410
http://www.uku.fi/kirjasto/julkaisutoiminta/julkmyyn.html

Series editors: Professor Lauri Kärenlampi, Ph.D.
Department of Ecology and Environmental Science
University of Kuopio

Professor Jari Kaipio, Ph.D.
Department of Applied Physics
University of Kuopio

Author's address: Department of Environmetal Sciences
Research Unit of Environmental Informatics
P.O. Box 1627
FIN-70211 KUOPIO
FINLAND
Tel. +358 17 163 159
Fax +358 17 163 191

Supervisor: Professor Erkki Oja
Department of Computer Science and Engineering
Helsinki University of Technology

Reviewers: Docent Enso Ikonen
Department of Process and Environmental Engineering
University of Oulu

Dr. Erkki Pesonen
Department of Computer Science
University of Kuopio

Opponent: Professor Jussi Parkkinen
Department of Computer Science
University of Joensuu

ISBN 951-781-305-8
ISBN 951-27-0000-X (PDF)
ISSN 1235-0486

Kopijyvä
Kuopio 2004
Finland

Kolehmainen, Mikko T. Data exploration with self-organizing maps in environmental informatics and bioinformatics. Kuopio University Publications C. Natural and Environmental Sciences 167. 2004. 73 p.

ISBN 951-781-305-8
ISBN 951-27-0000-X (PDF)
ISSN 1235-0486

ABSTRACT

The aim of this thesis was to evaluate the usability of self-organizing maps and some other methods of computational intelligence in analysing and modelling problems of environmental informatics and bioinformatics. The concepts of environmental informatics, bioinformatics, computational intelligence and data mining are first defined. There follows an introduction to the data processing chain of knowledge discovery and the methods used in this thesis, namely linear regression, self-organizing maps (SOM), Sammon's mapping, U-matrix representation, fuzzy logic, c-means and fuzzy c-means clustering, multi-layer perceptron (MLP), and regularization and Bayesian techniques. The challenges posed by environmental processes and bioprocesses are then identified, including missing data problems, complex lagged dependencies among variables, non-linear chaotic dynamics, ill-defined inverse problems, and large search space in optimization tasks.

The works included in this thesis are then evaluated and discussed. The results show that the combination of SOM and Sammon's mapping has great potential in data exploration, and can be used to reveal important features of the measurement techniques (e.g. separability of compounds), reveal new information about already studied phenomena, speed up research work, act as a hypothesis generator for traditional research, and supply clear and intuitive visualization of the environmental phenomenon studied. The results of regression studies show, as expected, that the MLP network yields better estimates in predicting future values of airborne pollutant concentration of NO_2 compared with SOM based regression or the least squares approach using periodic components. Additionally, the use of local MLP models is shown to be slightly better for estimating future values of episodes compared with one MLP model only. However, it can be concluded in general that the architectural issues tested are not able to solve solely model performance problems.

Finally, recommendations for future work are laid out. Firstly, the data exploration solution should be enhanced with methods from signal processing to enable the handling of measurements with different time scale and lagged multivariate time-series. The main suggestion, however, is to create an integrated environment for testing different hybrid schemes of computational intelligence for better time-series forecasting in environmental informatics and bioinformatics.

Universal Decimal Classification: 504.064.2, 519.683, 681.3.02
National Library of Medicine Classification: WA 26.5, WA 754
INSPEC Thesaurus: environmental science computing; biology computing;
data analysis; data mining; knowledge acquisition; self-organising feature maps; neural nets

ACKNOWLEDGEMENTS

Preparing this thesis has been a process of learning computational methods, and about environmental issues such as air quality and bioinformatics in different forms. In my opinion, a cross-fertilization of disciplines of this kind is the most fruitful way of finding new innovations and should be encouraged especially at the university level. In the later stage of this endeavour, my job description has included teaching as the major responsibility. This has given me a special opportunity to learn the methods by teaching them to others, which I think is the most effective way of learning them.

This study has been financially supported by Tekes (National Technology Agency of Finland), APPETISE EU project (IST-99-11764), the Maj and Thor Nessling Foundation, the Yrjö Jahnsson Foundation, the Research Foundation for Information Technology and the Savo Foundation for Advanced Technology. I am deeply grateful to all these sources of funding for making my work possible.

This kind of multidisciplinary approach naturally leads to a situation where the network of people connected to research work becomes large. This a benefit in many respects since a large network offers opportunities for co-operation (i.e. dividing the dull parts of the work), funding and especially learning from people in different disciplines. By far most thanks for supporting this work go to Professor Juhani Ruuskanen (University of Kuopio), who has in practice arranged the environment for me to chase this goal. I would also like to thank Professor Erkki Oja (Helsinki University of Technology) who has supervised my work and guided throughout the process from the first weak ideas to understanding what this is all about. As already said, this kind of research is not possible without co-workers and I would like to thank them: Dr. Olavi Raatikainen, Hannu Martikainen, Teri Hiltunen, Harri Niska, Päivi Rönkkö, Heikki Junninen, Anna Ruuskanen, Eelis Rissanen, Toni Patama and Dr. Kari Tuppurainen. Two of the works were done in cooperation with other research groups, which gives me an opportunity to also thank Professor Eero Castren, Dr. Garry Wong and Petri Törönen, who introduced me to the interesting field of bioinformatics. I also thank Veli-Pekka Valkonen, Professor Jukka Salonen and Dr. Hanna-Maaria Lakka, who were my co-authors in the study of applying data exploration to epidemiological data. At the review stage the valuable comments of Dr. Enso Ikonen and Dr. Erkki Pesonen helped to improve the structure of the thesis as well as several details in it. I would like to thank the research group on Engineering and Computational Intelligence at the University of Jyväskylä, Finland, for allowing me to use their Neural Data Analysis (NDA) package (http://erin.mit.jyu.fi) and Visipoint Oy for the use of their Visual Data software (http://www.visipoint.fi) for the interactive analysis. Finally, I would like to thank my wife and children for putting up with a situation where the priorities of family, work, renovating a house and doing research were not always clear.

Kuopio, January 2004

Mikko Kolehmainen

LIST OF PUBLICATIONS

The thesis consists of an introduction together with the following original publications:

1. Kolehmainen M., Martikainen H., Hiltunen T., and Ruuskanen J., Forecasting air quality parameters using hybrid neural network modelling, *Environmental Monitoring and Assessment*, 2000, 65, 277-286.

2. Kolehmainen M., Martikainen H., and Ruuskanen J., Neural networks and periodic components used in air quality forecasting, *Atmospheric Environment*, 2001, 35, 815-825.

3. Kolehmainen M., Rissanen E., Raatikainen O., and Ruuskanen J., Monitoring odorous sulfur emissions using self-organizing maps for handling ion mobility spectrometry data, *Journal of Air and Waste Management*, 2001b, 51, 966-971.

4. Kolehmainen M., Rönkkö P., and Raatikainen O., Monitoring of yeast fermentation by ion mobility spectrometry measurement and data visualisation with self-organizing maps, *Analytica Chimica Acta*, 2003, 484, 93-100.

5. Niska H., Hiltunen T., Kolehmainen M., and Ruuskanen J., Hybrid models for forecasting air pollution episodes, *International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA'03)*, University Technical Institute of Roanne, 80-84, France, April 23-25, 2003b, Springer-Verlag/Wien.

6. Törönen P., Kolehmainen M., Wong G., and Castren E., Analysis of gene expression data using self-organizing maps, *Federation of European Biochemical Societies (FEBS) Letters*, 1999, 451, 142-146.

7. Valkonen V-P., Kolehmainen M., Lakka H-M., and Salonen J., Insulin resistance syndrome revisited: application of self-organizing maps, *International Journal of Epidemiology*, 2002, 31, 864-871.

THE AUTHOR'S CONTRIBUTION

The publications which compose the backbone of this thesis are the result of several research projects carried out at the University of Kuopio during the years 1997-2002. The author's contribution varies from case to case, as explained below.

The articles 1 and 2 about regression and hybrid modelling of air quality (Kolehmainen et al., 2000; Kolehmainen et al., 2001) were produced by the author, and the role of other researchers (H. Martikainen and T. Hiltunen) was to help with numerical calculations, whereas Professor Ruuskanen was in a supervising role. In studies 3 and 4, which used the IMS measurement data (Kolehmainen et al., 2001b; Kolehmainen et al., 2003) , the author designed the laboratory experiments and also participated in the practical work with P. Rönkkö and E. Rissanen. The data handling and preparing of the manuscripts were the author's sole responsibility and the other co-authors (Professor Ruuskanen and Dr. Raatikainen) mainly participated in collaborating in the manuscript writing phase.

The article 6 on gene expression data handling (Törönen et al., 1999) was mainly prepared by Petri Törönen, and the role of the author was largest in the initial phase of this project, contributing to the design of the study as well as development of the computational methodology and software introduced. In the similar study (7) about epidemiological data (Valkonen et al., 2002), the roles of Veli-Pekka Valkonen and the author were equal and we carried out most of the work in design, data handling and manuscript preparation. The role of Dr. Lakka was to help with epidemiological interpretation, and the work was supervised by Professor Salonen.

In the conference proceedings article 5 (Niska et al., 2003b), the role of the author was that of designing and guiding, and the actual work was in practise mainly carried out by H. Niska and T. Hiltunen, with Professor Ruuskanen as supervisor. It should also be noted that article 5 was a direct continuation of article 1 (Kolehmainen et al., 2000), reworking the case in a more systematic manner.

GLOSSARY

Dissolved oxygen	Concentration of oxygen dissolved in the medium.
Evolutionary Computation	Techniques mimicking nature's evolutionary principles to drive its search towards an optimal solution (Deb, 2001).
Exploratory Data Analysis	Data analysis based on visualizing and exploring data without a clear idea of what to look for.
Feature selection	An algorithm that selects the most useful variables to be used in the feature vector.
Fermentation	In broad terms, use of micro-organisms to carry out enzyme-catalyzed transformations of organic matter (Ward, 1989).
Functional state model	A modeling concept that divides a process into upper level of process states and lower level of local models.
Fuzzy logic	A form of logic that handles uncertainty by using intermediate values between true and false.
Genetic Algorithm	An evolutionary algorithm which generates each individual from some encoded form known as a "chromosome" or "gene".
Hidden layer	A layer in a neural network that is not directly connected to the outside world.
Ion Mobility Spectrometry	A collection of principles, practice and instrumentation used for characterizing chemical substances using their ion mobilities in gas phase (Eiceman and Karpas, 1994).
IMCELL™	An advanced form of traditional IMS technique; a proprietary technology of Environics Oy (Mikkeli, Finland).
Knowledge Discovery	Process of extracting information from data.
Input layer	A buffer in a neural network that distributes the input signals to hidden layers.
Linear regression	A technique of fitting an equation to data points with the parameters to be determined being in a linear position.
Metabolism	Chemical processes occurring in cells: includes anabolic (synthetizing) and catabolic (degrading) processes.
MGD-1	A commercial gas detector (Environics Oy) based on the IMCELL™ technique.
Multi-Layer Perceptron	A neural network model consisting of input, hidden and output layers, with forward connections between them.
Neurocomputing	A way of computing that is able to learn from examples: it imitates neuron models found in the brain.

Output layer	The layer in a neural network that outputs the computed result of the network.
Phase	A perceptile period of time when a process or culture expresses certain typical properties such as growth of biomass.
Radial Basis Function Network	Neural networks based on Cover's theorem about linear separability in higher dimensions.
Sammon's mapping	A gradient search method, which aims at representing points in p-dimensional space in 2 dimensions, conserving as much as possible of the original information.
Self-Organizing Map	Type of neurocomputing based on an unsupervised learning algorithm and a special type of map-like neural network.
Stationary process	A process in which the statistics of any subset of measured data accurately describe the statistics of the entire data.
Support Vector Machines	Neural networks, based on the idea of constructing a hy perplane as a decision surface so that the margin of separa tion between positive and negative examples is maximized.

LIST OF SYMBOLS AND ABBREVIATIONS

DO	Dissolved Oxygen
EC	Evolutionary Computation
EDA	Exploratory Data Analysis
GA	Genetic Algorithm
IMS	Ion Mobility Spectrometry
KDD	Knowledge Discovery in Databases
MLP	Multi-Layer Perceptron
RBFN	Radial Basis Function Network
SOM	Self-Organizing Map
SVM	Support Vector Machines

x, y, z	random variables
a, b, c, d	constants
e	error residual
i, j, k, l, m, q	general indexes
t	time
$\hat{z}$	estimate of variable z
$\mathbf{X}$	data matrix
$\mathbf{x}_k$	kth input vector
$\mathbf{w}(t)$	weight vector of the neural network with t as a counter for iterations
p	number of dimensions (variables)
n	number of input vectors (data lines)
d_{xx}	distance according to metrics xx
E	stress or error function to be minimized
$\alpha(t)$	learning rate factor as a function of time
$\sigma(t)$	width of the kernel as a function of time
c_{BMU}	index for the Best Matching Unit (neuron)
h_{cm}	neighbourhood function according to location vectors r_c and r_m
$\mathbf{r}_c$	location vector for the BMU neuron
$\mathbf{r}_m$	location vector for the neighbouring neuron
S_{NN}	set of Nearest Neighbour neurons
$\tilde{A}$	fuzzy set for the base set A
$\mu_{\tilde{A}}$	membership function for fuzzy set
$\mathbf{v}_i$	cluster centre
s_i	within-cluster distance (deviation) for cluster i
$\mathbf{g}$	gradient
$\mathbf{J}$	Jacobian matrix
$\mathbf{I}$	identity matrix
μ	weighting constant
P_i	ith predicted value
O_i	ith observed value
P()	probability
p()	probability function

CONTENTS

1. INTRODUCTION

The problem encountered frequently in today's world is that of information overflow. To be precise, there is a lot of data available but too sparse information or knowledge about the subject inspected often in a limited time. Therefore, the enrichment process where the measured data is first turned into problem specific information and then into more general knowledge is a very important subject of study. This was also the starting point of this thesis with focus on environmental and bioprocesses.

Nowadays, there is an increasing amount of data available from our environment. Meteorological data is one of the oldest sources of this kind. During the last two decades, urban air quality measurements have been carried out routinely in most of the cities or even towns. Together, these two create an excellent opportunity to monitor the workings of the nature and human activities mixed in complex ways. Moreover, as the datasets are now reaching many years back in time it is possible to use computational models for atmospheric phenomena described through these time-series.

Another important trend in this century has been the advancement of medicine and biochemistry. This has culminated to the innovations of being able to monitor the genes using so called micro-array technology. Thus, based on those and more traditional measurements, the complex behaviour of living creatures can be analysed and modelled enabling even individual medication. This opens remarkable prospects also in environmental sciences where the monitoring of environmental processes could be brought into a new level of accuracy.

The large amounts of (on-line) data requires automated data processing schemes to be developed, which can then deliver information for planning, maintenance and exception handling (alarm) activities. The first attempts of this kind were carried out in the 70's and 80's when a computational scheme called artificial intelligence (AI) was developed and applied in many sectors. The basic approach of AI was to capture human knowledge in the form of objects and rules in order to create an expert system for the data processing tasks required. However, it turned out that this approach is not able to capture the true nature of the phenomena in a sufficient way. Starting from the 80's a new brand of computation was developed and it is nowadays called computational intelligence (CI). The basic approach of CI is to create the data processing system by utilizing measurements of the study subjects and then constructing more abstracted objects algorithmically mimicking the processes found in the brain (e.g. neural networks) or in the nature (e.g. evolutionary computation).

During the last two decades, the methods of CI have been successfully applied in many industrial and everyday applications including process control and consumer electronics like intelligent washing machines for example. Meanwhile, it has become clear that these methods are useful only in limited areas of application, due to the fact that they are not really intelligent in the human sense. However, each method can be seen to dominate its own dimension of intelligence, which makes them complementary rather than competitive in respect to each other. Naturally, this has led to attempts to combine two or several methods of CI in a co-operative way, leading to active research in hybrid computational intelligence in recent years.

Multi-disciplinary research is becoming increasingly common in science and this is reflected in education at the university level, especially in environmental sciences and biosciences. New disciplines are being created, and the most exciting ones are usually based on applying the new information sciences to other fields of research. Among them are environmental informatics and bioinformatics. This thesis is the result of studies in the field of environmental informatics and bioinformatics. It is hoped that this dissertation will serve as an introduction to those who

want to enlarge their knowledge of scientific methods in those fields.

Our research group has decided to focus our research and teaching in environmental informatics at the University of Kuopio on the following themes: (i) using computational intelligence for analyzing and modelling environmental data; (ii) developing methods for continuous monitoring of the environment; and (iii) utilizing software engineering for delivering the solutions to end-users. As discussed later, this approach can also be applied to the field of bioinformatics. The central theme of this work is the first, which implies the need to understand how data can be utilized by refining them into information and knowledge. On the other hand, continuous monitoring adds tools and devices which can be used to yield data continuously and in real time, for example the applications based on the 'electronic nose' in two of the studies of this thesis. Finally, the solutions thus created can be delivered to end-users by utilizing the best of modern software engineering to package the solutions in an effective and user-friendly way. This approach has led to three commercial Visipoint Ltd. softwares called Visual Data, Visual Gene and Visual Nose (based on the NDA library). Their development was initiated in the studies for this thesis, and the design and guidance of their implementation were carried out by the author.

The aim of this thesis was to evaluate self-organizing maps (SOMs) and other related methods for analysing and modelling environmental and bioinformatics problems, using a number of selected case studies. For analysing the data, SOM was selected as the main method because of its properties, which are suited for both clustering and visualization. The natural choice for regression was multi-layer perceptron (MLP), which has already been applied in numerous applications. Other methods were then added, to complement the other two. The domains of application (air quality and bioinformatics) were selected because these areas had suitable data available, as well as experts in the field who could participate in guiding the work. These case studies are covered in detail in the publications and also referred to in the introductory part of the thesis.

The introductory part of the thesis is divided as follows. Chapter 1 is a short introduction to the themes to be discussed. Chapter 2 introduces environmental informatics and bioinformatics together with the challenges posed by data collected from the processes of those fields. This is followed in Chapter 3 by a presentation of computational intelligence and other important concepts needed for understanding the discussion in the rest of the thesis. Chapter 4 describes the methods used in the studies, enabling a reader new to CI to follow the discussion and the articles. Chapter 5 goes into the heart of the topic, discussing the results of the practical works in the thesis and also introducing the software created. Finally, Chapter 6 draws everything together with concluding remarks.

In addition to the novel applications covered in the publications, the main innovation of the thesis was to show how the performance of self-organizing maps can be enhanced by combining them with Sammon's mapping. Even though this idea has been presented earlier (e.g. Chang and Lee, 1973), it has not been previously studied to this extent with real world problems and datasets. Additionally, the Visual Data software developed and tested in these studies takes the approach to a practical level, where end-users who are not familiar with the computational methods can produce results in their own disciplines.

2. THE DOMAIN OF APPLICATION

2.1. Environmental informatics

Environmental informatics (EI) is a new discipline which has been defined by Page and Hilty (1995) as follows:

Environmental Informatics is a special subdiscipline of Applied Informatics dealing with the methods and tools of computer sciences for analyzing, supporting and setting up those information processing procedures which are contributing to the investigation, removal, avoidance and minimization of environmental burden and damages.

Another definition has been proposed by Green and Klomp (1998), focusing more on the integration of global data sources and even on aspects of life sciences:

The application of information technology to environmental issues is changing both theory and practice. The idea of "natural computation" provides new ways to understand environmental complexity across the entire range of scales, from individual phenotype to biogeography. Understanding the ways in which local interactions affect the global composition and dynamics of whole communities is crucial to the viability of strategies to manage ecosystems, especially in landscapes altered by human activity. Also environmental planning and management are increasingly dependent on accurate, up-to-date information that sets local decisions within a global context. The Internet makes it possible to combine environmental data from many different sources, raising the prospect of creating a global information warehouse that is distributed amongst many contributing sites.

It can thus be seen that there are different interpretations and points of view concerning what is included in environmental informatics. For the practical purposes of this thesis, this is my own definition:

Environmental informatics is based on applying information technology to environmental issues using data-driven methods.

On the other hand, environmental informatics can be seen as a mediator between environmental sciences and modern informatics, offering novel solutions based on collected data and processing them into the information and knowledge needed for problem solving in that domain. This is illustrated in Figure 1.

Naturally, such a problem domain is impossible to solve in its general form, so represen-

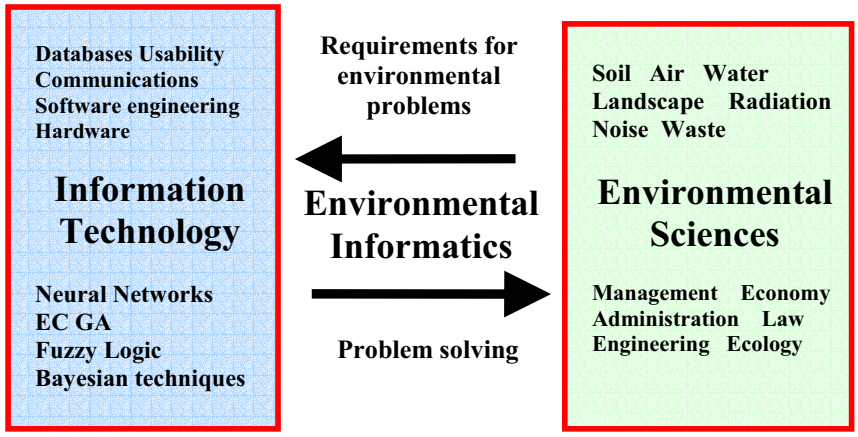

Figure 1. The role of environmental informatics as a mediator between environmental sciences and modern informatics (modified from Page and Rautenstrauch, 2001).

tative examples are needed in the form of case studies. In this thesis, they were found in the fields of urban air quality and bioinformatics. This is mostly due to the research work that has been carried out in the University of Kuopio in various research groups, which meant that representative data and trained experts in the discipline were available, and the experts were able to spot the most important targets for multi-disciplinary studies of this kind.

2.2. Urban air quality

In recent years, urban pollution has emerged as the most acute problem in environmental health, because of its negative effects on health and living conditions. To prevent the further decline of air quality, scientific planning of analysis methods and pollution control are required. Within this framework it is necessary to (i) analyse and specify all pollution sources and their contribution to air quality; (ii) study the different factors which cause the pollution; and (iii) develop tools to reduce pollution by introducing control measures and alternatives to existing practices (Kolehmainen et al., 2000).

The research carried out thus aims at attaining a better understanding of the phenomena associated with gaseous atmospheric pollutants, aerosol particles and meteorological parameters. The specific goal is to develop air quality models which can forecast the next day's urban air quality. The modelling is typically based on the historical data concerning air pollutants and predicted meteorological variables.

Air quality forecasting is a typical problem domain of environmental informatics in many respects. Firstly, ample data have been measured and collected for both the air quality parameters (concentrations of gaseous and particle pollutants) and the closely related meteorological parameters. This enables the use of data-driven methods such as neural networks. Secondly, the data originate from open processes which render the chaotic and noisy characteristics of nature. This is different from the more limited and controllable circumstances of industrial processes, which can, of course, also exhibit complex behaviour. Thirdly, urban air quality is also connected to the main themes of environmental sciences, namely to the consequences of human activities for nature, and also for humans themselves in the form of deteriorating health and living conditions.

2.3. Challenges with environmental data

Environmental datasets are usually comprised of time series measured over several years. The primary factors influencing the modelling thus originate from the cycles of nature (seasonality and daily variations) and from human activity, e.g. daily and weekly variations (Kolehmainen et al., 2000). Seasonality usually leads to the requirement of having at least 3-5 years of measured data for the modelling. In this way at least 1-2 years can be used for validating the modelling results. Consequently, urban air quality is a prominent object for such studies, as air quality data have been monitored for one or two decades in many western cities.

The objective of urban air quality modelling, for example, is usually to forecast the concentrations of pollutants for the next day, in order to make recommendations and issue warnings about bad air quality episodes (see review by Gardner and Dorling, 1998). In this case, the sampling time needed is usually hours or days, even though the measurements may be carried out in two- or three-minute sampling intervals. Thus, the number of data points available is usually tens of thousands, which is enough for most of the methods used in computational intelligence.

Consequently, the quality of the data is usually of more concern. The most often encountered problem is missing data, due to measure-

ment device failures or human errors. In case of time-series data, most CI methods require complete datasets, which leads to imputation of the missing values (Bishop, 1995). Another typical quality problem is outliers, which are due to measurement errors either by devices or human operators. Severe challenges can also be caused by systematic errors originating from erroneous calibration of measurement devices, for example.

The most often encountered environmental datasets (Gunther, 1998) can be characterized as multivariate time series (MTS). This means that there are several variables measured at the same sampling sites or that they can be treated as such by calculating hourly averages, for example. This leads to a data matrix, where the columns represent different variables of the same process and the rows correspond to different time points. Even if there is only one variable (time series) directly available, auxiliary variables can often be generated from other sources of information. The most obvious one is to create new variables based on the time of the measurements. This is because most environmental processes are subject to the changes of nature (seasonality) and also human activities (Kolehmainen et al., 2000). Thus, variables such as hour of the day, day of the week and day of the year can be utilized to connect the periodicity of these affecting factors to the process inspected. For the same reasons, it is often helpful to consider meteorological variables to be added to the dataset, whenever available from the same geographical region.

On the other hand, the multitude of variables can also cause problems, due to the correlations between variables, which lead to the same information being presented to the learning algorithm (see Chapter 4) several times. This in turn can mislead the algorithm to pay more attention to some features than to others. A natural solution to this is to apply feature selection in the pre-processing stage in one way or another (Tucker et al., 2001). Secondly, some variables may contain noise, which can hamper the performance of the learning methods.

Additionally, seasonal components in time series can burden or prevent the working of the learning algorithm by introducing non-stationarity into the data (Dorffner, 1996). They should be handled explicitly, for example by differencing or by fitting linear regression models to the data in the pre-processing stage. An example of this can be found in one of the works included in this thesis (Kolehmainen et al., 2001).

The processes themselves usually have an inner structure, where different parts of the process influence each other (Haykin and Principe, 1998). These complex interactions usually include lags, i.e. the changes in one part of the system affect some other part of it with a delay. This in turn is reflected in the corresponding measured time series as lagged values. Complexity in many natural phenomena arises from massive interactions among different parts of a non-linear dynamical system (Haykin and Principe, 1998). If the system is additionally sensitive to initial conditions, it can express chaotic behaviour. In such a system, nearby states of the system are separated rapidly. This leads to diminished ability to forecast the future evolution of the system, thus effectively preventing the accomplishment of the short time goals of time-series forecasting. Chaos is also expressed in the time-series in a similar fashion as random noise, making it difficult to distinguish the former from the latter. Therefore, it is important to identify the nature of the time series early in the data processing chain, since it can determine which models can effectively be used in that context.

A problem is defined to be an inverse problem of another problem if the formulation of the former requires full or partial knowledge of the latter (Kirsch, 1996). On the other hand, Hadamard (1902) has introduced the concept of well-posed problem; problem is said to be well-posed in the sense of Hadamard if it satisfies three conditions, namely existence,

uniqueness and continuity. If any of these conditions is not satisfied, the problem is called ill-posed. Consequently, a dynamic reconstruction problem (such as weather-related problems in environmental sciences) is in reality an ill-posed inverse problem (Haykin and Principe, 1998). In order to be able solve such problems, some prior knowledge is usually imported to the model.

The challenges and their possible solutions presented above lead easily to a situation where the model to be created is very complex. To summarize, the following properties of environmental processes and the corresponding time series can be found to create this complexity:

- The seasonal and other cyclic dependencies in them require that the model can adapt to changing conditions, i.e. the processes are not stationary.
- The functions to be modelled with CI methods are complex, and the corresponding model requires a large number of free parameters to fit it.
- The processes are described through multitudes of variables, which leads to large input space.
- There are lagged interactions between variables, which makes the input space larger by one or two orders of magnitude in the worst case.
- The environmental processes are often non-linear and potentially chaotic, which needs special attention.
- The processes can be ill-defined inverse problems, which require prior knowledge to be imported to the model.

A large part of the modelling process can thus be seen as an optimization problem, where the most effective variables and their combinations are selected and the model parameters are set to optimal values. However, this optimization can potentially create a large search space and lead to unacceptable execution times for the algorithms. Therefore, the optimization process must be somehow divided so that the search space can be split into manageable pieces. Basically, this could be done in several ways, which are examined in more detail in Chapter 5.3.

2.4. Bioinformatics

Bioinformatics is a new discipline based on the same driving forces of development as environmental informatics, namely the advent of computing power, software engineering and utilization of computational intelligence. Additionally, the breakthrough in gene technology gives it a flavour of its own. Bioinformatics can be defined as follows (Nilges and Linge, 2003):

Bioinformatics derives knowledge from computer analysis of biological data. These can consist of the information stored in the genetic code, but also experimental results from various sources, patient statistics, and scientific literature. Research in bioinformatics includes method development for storage, retrieval, and analysis of the data. Bioinformatics is a rapidly developing branch of biology and is highly interdisciplinary, using techniques and concepts from informatics, statistics, mathematics, chemistry, biochemistry, physics, and linguistics. It has many practical applications in different areas of biology and medicine.

This practical approach to bioinformatics can be deepened in the following way.

In the search for unravelling the organization of life, the crucial factor is understanding temporal and spatiotemporal dynamics of biological processes. Traditionally, this has been tackled by the reductionist approach, which means assuming that once the individual molecules have been identified, the complete function of the whole system or organism can be derived from the sum of individual molecular actions. Despite of the success so far, however, the consensus is growing that the reductionist paradigm cannot answer to the following three questions namely i) how living systems function as a whole, ii) how they transduce and

process dynamical information and iii) how they respond to external perturbations (Walleczek, 2000).

The central concept in these fundamental questions is self-organization, which refers to emergent features possessed by a dynamical system as a whole but not by its constituent parts. The features in biological systems that enable self-organization are openness and nonlinearity. Openness is due to the fact that the living systems are in the state of permanent flux and they are continuously interchanging matter and energy with their environment. Respectively, nonlinearity results from a complex organization of a vast network of molecular interactions. Therefore, Walleczek (2000) suggests that biomedical scientists should adopt an information-based and whole-systems approach to biological understanding in order to develop advanced biomedical technology.

The main ideas here are clearly in the last sentence, namely information-based and whole-system approaches. This was also the starting point for the bioinformatics-related studies of this thesis. Hence, the analysis was based on measured data and the target was to describe the possible configurations of whole systems, namely human beings, yeast or biotechnical processes. The following definition describes and summarizes the bioinformatics-related goals of this thesis:

Bioinformatics applies information technology to biological issues using data-driven methods in order to reveal the working of different abstraction and complexity levels of the entity studied.

2.5. The nature of bioinformatics data

When we are considering datasets originating from living nature, i.e. vegetation, animals or human beings, the structure of the data is different from that of data found in the environment. Firstly, the time-series properties are practically absent, since usually only a few points of time are available. This naturally restricts the number of methods applicable, and the model reflects only a static state of the system. Also, the number of individuals and variables is often limited. However, in the datasets used in two of the works of this thesis, the number of genes available for gene expression analysis (Törönen et al., 1999) was over 6000, and the number of subjects available from the epidemiological survey (Valkonen et al., 2002) was 1650. Consequently, the use of methods of CI, such as neural networks, was valid still. For the third study (Kolehmainen et al., 2003) the data originating from the electronic nose (MGD-1) are time-series data, which makes it possible to account for the dynamical aspects of the process. However, the measurements of more traditional variables such as glucose and ethanol concentrations were carried out by sampling the medium, which leads to a sparse data matrix.

The restrictions in bioinformatics are mostly due to limitations of the corresponding measurement techniques, which rely on costly and labour-oriented laboratory practise. Unfortunately, this also holds for micro-array data, which are gathered using expensive chip technology. However, it can be anticipated that the development which has led to cheap microcomputer technology will also enable cost-effective measurements of gene expression in the future. This would make it possible to measure and analyse the time-series properties of biological phenomena.

During the last decade, a vast amount of bioinformatics data has been collected in numerous databases, which can be utilized using web technology. Most of the data are related to gene sequences, but data related to gene expression and protein expression of various species are also accumulating. However, the data are in various formats in different databases, which makes it tedious to aggregate data from various sources (Stein, 2002), so it is necessary to find ways to manage these data in coherent ways that permit the modelling of biological phenomena using the best available measurements.

2.6. The link between environmental informatics and bioinformatics

Environmental informatics and bioinformatics are similar in their methods in many respects. Firstly, both use methods of computational intelligence as well as statistics for yielding information and knowledge from measured data of the phenomenon studied. Secondly, the role of software engineering is important in the form of database management, modelling tool development and user interface design, for example. Additionally, both disciplines study phenomena which are often based on biology, i.e living things. The similarity of these new disciplines is illustrated in Figure 2.

It should be noted that besides common methods, new ways of problem solving can also be found by utilizing bioinformatics as a tool in the environmental domain (Geer et al., 2001). This leads to new definitions or even sub-disciplines in the form of environmental bioinformatics and genomics.

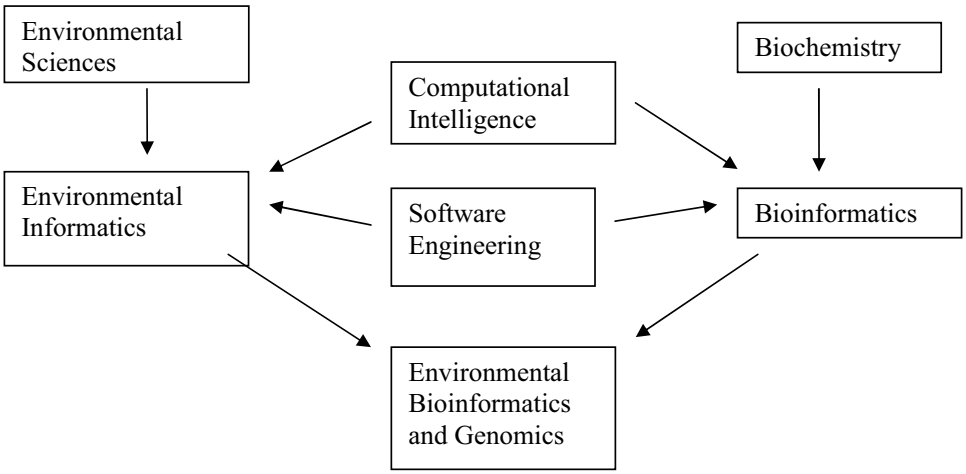

Figure 2. Environmental informatics and bioinformatics are basically similar. Both disciplines use software engineering and computational intelligence as their building blocks. By combining these two, interesting combinations can be created for monitoring environmental processes and finding solutions for environmental problems.

3. COMPUTATIONAL INTELLIGENCE

3.1. What is computational intelligence?

One of the earliest definitions of computational intelligence (CI) can be found in Bezdek (1994):

A system is computationally intelligent when it deals with only numerical (low-level) data, has pattern recognition components, and does not use knowledge in the AI sense; and additionally when it (begins to) exhibit (i) computational adaptivity; (ii) computational fault tolerance; (iii) speed approaching human-like turnaround; and (iv) error rates that approximate human performance.

This may seem a rather complex definition and it also leans on the traditional definition of artificial intelligence (AI) of symbolic representation in an exclusionary way. Fogel (1995) offers another definition:

…a relatively new term offered to generally describe methods of computation that can be used to adapt solutions to new problems and do not rely on explicit human knowledge.

The most common element in these two definitions seems to be adaptivity. A link to human knowledge is also made in a sense, which implies use of measured data instead of explicit rules constructed by human operators. Pal and Pal (2002) combine these definitions by requiring the following characteristics of a CI component (neural network, fuzzy logic etc.):

- Considerable potential in solving real world problems
- Ability to learn from experience
- Capability of self-organizing
- Ability to adapt in response to dynamically changing conditions and constraints

They summarize this by requiring the component to display intelligent behaviour as observed in humans. Interestingly, this last definition refers to the same attributes of intelligence that Feigenbaum (2003) considers in his article. He starts by reminding readers of Turing's test (Turing, 1950), where judgements about the intelligent behaviour of artificial systems should be based on the extent to which their behaviour resembles that of humans. However, Feigenbaum points out that human behaviour is very multidimensional, and that most of the achievements of AI and CI result from the study of different dimensions separately. Thus, the label "partially intelligent" could be justified when considering the methods of CI.

Another perspective to CI and intelligent behaviour can be found in King (1998), who starts by defining an intelligent system as having the attributes of intelligence, namely the capability for any of the following: comprehension, reasoning, perception, communication and learning. However, King focuses on knowledge representation, which is the process through which any intelligent system stores knowledge about the problem domain. In this way the following definition (King, 1998) can be seen as distinguishing between symbolic AI and computation- oriented CI:

Symbolic approaches focus on representing how the brain reasons, irrespective of how this is accomplished biologically, whereas the computationally intelligent system focuses on mimicking the brain's biological processes for accomplishing learning, memory, and pattern recognition.

The difficulty in defining CI in a short and yet comprehensive way is probably the reason why it is frequently defined in terms of its components. The most often encountered list includes artificial neural networks (ANN), fuzzy logic, genetic algorithms (GA) and Bayesian networks.

This also brings up another concept, soft computing (SC). It has been defined, by for example Pal and Pal (2002), Dote and Ovaska (2001), and Zadeh (1994), as a consortium of different computing tools that can exploit tol-

erance for imprecision, uncertainty, and partial truth to achieve robustness, tractability, and low cost. Its core methods are fuzzy logic, (artificial) neural networks, and evolutionary computation (EC). It should be noted that the role of these methods is not competitive but synergistic and complementary. Interestingly, the same idea is also a central issue of this work, namely that of combining several methods of CI.

3.2. Hybrid computational intelligence

Hybrid computational intelligence is defined by Tsakonas and Dounias (2002) as any effective combination of intelligent techniques that performs better than or as well as simple standard intelligent techniques. In their review, the following hybrid computational intelligence schemes are acknowledged:

- Neural networks and fuzzy logic
- Neural networks and evolutionary computation
- Fuzzy logic and evolutionary algorithms
- Machine learning and fuzzy logic
- Machine learning and evolutionary algorithms
- Hybrid neural network systems
- Hybrid genetic algorithms

The first of these (neural networks and fuzzy logic) is probably the most successful example of hybrid CI (Tsakonas and Dounias, 2002): another term used is neuro-fuzzy systems and techniques. The most frequently encountered principles of the combination of these two are use of fuzzy functions in neural network nodes and application of neural-like training for the fuzzy membership functions in fuzzy systems.

It has also been pointed out (Tsakonas and Dounias, 2002) that the increasing computation power and technology make it possible to use increasingly complex intelligent architectures. It should also be noted that the list of methods included in CI varies from author to author. This can be seen in another review of hybrid soft computing systems applications by Bonissone et al. (1999), where they include Bayesian belief networks instead of machine learning. In this thesis, several combinations of CI methods as well as traditional methods such as linear regression were used together:

- SOM and Sammon's mapping (gradient search)
- SOM, MLP and fuzzy logic
- MLP and linear regression
- SOM and linear regression
- SOM, MLP and traditional clustering algorithms

However, it should be noted that hybrid methods are generally understood to work together, either simultaneously or performing the same function. In this sense, Yao (1999) has reviewed different combinations of ANNs and evolutionary algorithms, which is a very important subtopic of hybrid CI. He concluded that the two forms of adaptation, namely evolution and learning, make the adaptation of systems using both of these much more effective in a dynamic environment. However, as Beliakov and Abraham (2002) reminded us, deterministic approaches can yield near optimal results much faster than EC.

3.3. Data-mining

A more practical perspective to CI can be found under the term data-mining. It stems from an important problem of the age, information overflow. Humans today find themselves in a situation where there is too much data and too sparse information (Adriaans and Zantinge, 1996). This creates a challenge for finding methods and tools for enriching information from the large amounts of data. Data-mining has been defined by Hand et al. (2001) as follows:

Data-mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in

novel ways that are both understandable and useful for the data owner.

It is usually regarded as being the key element in a more elaborate process known as knowledge discovery in databases (KDD). KDD is linked to another important development, data warehousing. A data warehouse is a central store of data that has been extracted from an operational database (Adriaans and Zantinge, 1996). The term KDD has been used by Fayyad (1996) to denote the overall process of extracting high-level knowledge from low-level data. He presents this in detail, including data-mining as part of this process. This is explained in more detail at the beginning of Chapter 4. However, he also points out that data-mining and KDD are often used interchangeably.

According to Hand et al. (2001), the data-mining process can be regarded as consisting of the following steps:

1) Exploratory data analysis (EDA)
2) Descriptive modelling
3) Predictive modelling: classification and regression
4) Discovering patterns and rules
5) Retrieval by content.

The first three steps are concerned with model building, which is the focus of this thesis. They are briefly introduced below, to help in understanding the different aspects of data analysis and modelling. The last two, on the other hand, are outside the scope of this thesis.

3.3.1. Exploratory data analysis

The goal of EDA is to visualize and explore data without a clear idea of what to look for. It has also been called data-driven hypothesis generation (Hand et al., 2001). The basic idea is to use the strong image processing capabilities of the human being to detect novel patterns in the data. The tools for EDA include simple plotting of one or two variables, histograms and box plots. However, as the dimension of the data increases, multivariate methods are needed to reduce the dimensionality, usually to 2 dimensions. Suitable methods for this include principal components analysis (PCA), self-organizing maps (SOM) and Sammon's mapping (see Chapter 4 for detailed discussion).

3.3.2. Descriptive modelling

In descriptive modelling some intrinsic properties of the data are uncovered by algorithmic techniques. The most common ways to describe the data are density estimation and cluster analysis.

Density estimation can basically be achieved with two types of methods, parametric and non-parametric (Bishop, 1995). The parametric methods are based on using some probability distribution such as the normal distribution. Once the form of the model has been decided on, the goal is then to set the parameters, based on observed data. The best known method is expectation maximization (EM), which is one of the maximum-likelihood techniques (Dempster et al., 1977). Another well-known approach is Bayesian inference (Gelman, 2003), where the parameters are described using a distribution set initially to some prior. This is then converted to a posterior distribution using Bayes' theorem, which links the prior beliefs and measured data together.

The non-parametric methods for density estimation do not assume any prior form for the distribution function (Bishop, 1995). Histograms can be considered one way of achieving the density estimation but their representation becomes awkward when the number of dimensions grows. Other well-known examples of this class are kernel-based methods and K-nearest-neighbours.

The SOM also describes the underlying density to some extent, but the representation is not exact. In other words, the SOM undersamples regions of high probability and oversamples those of low probability (van Hulle, 2000). Algorithmic ways to respond to this

have been suggested, but a practical way is to apply Sammon's mapping to the weight vectors of the SOM, which is an important topic of this thesis.

The aim of cluster analysis is to decompose or partition data into groups where the items in one group are as similar as possible to each other while being as dissimilar as possible to those in other groups (Hand et al., 2001). The traditional methods for this include c-means clustering (MacQueen, 1967), hierarchical clustering (Johnson, 1967) and fuzzy c-means clustering (Bezdek, 1981). Note that the SOM algorithm can also be used for this purpose. One problem often encountered here is to determine the number of clusters. This is due to the fact that most methods require the number of clusters to be preset. Another problem is due to difficulties in constructing a validity criterion for the partitioning of the data achieved. Thus, it is difficult to compare different methods against each other and the final criterion is often determined by visual inspection. This has led to the situation where numerous algorithms have been developed for different kinds of data (Hand et al., 2001).

Basically, there are two kinds of clustering algorithms: partition-based and agglomerative (Hand et al., 2001). The former methods (including c-means and fuzzy c-means clustering) minimize or maximize the so-called score function, which is calculated using centres of the clusters as reference points. Consequently, a distance measure has to be defined between input points according to some metric (e.g. Euclidean metric). The agglomerative clustering methods (including hierarchical clustering) are based on measures of distance between clusters. Given the initial clustering, they merge those clusters which are nearest to each other, in order to reach a reduced number of clusters. In this thesis, clustering methods were applied in one of the studies (Niska et al., 2003b) in order to split the problem space into local domains.

3.3.3. Classification

The aim of predictive modelling is to build a model which is able to estimate a value of (usually) one variable from other variables. In discriminative classification, the model outputs a class label stating membership of the given sample. Well-known examples of this are nearest neighbour classification and learning vector quantization (LVQ), the latter being a supervised version the SOM algorithm. The earliest formal approach to classification can be found in Fisher (1936), who introduced the idea of seeking a linear combination of variables that are able to discriminate between two classes. This kind of discriminative classification is thus based on finding decision boundaries between the classes.

In contrast to this, a probabilistic model for classification outputs a probability that the sample belongs to a certain class (Hand et al., 2001). The objects belonging to a class are supposed to have measurement vectors that are distributed according to some density function. Once the functional form and its parameters have been estimated, the posterior probability that a sample belongs to a certain class can be calculated using Bayes' theorem.

Besides the theoretical viewpoint described above, a prescriptive framework can be created by dividing the building of classifiers into three approaches (Hand et al., 2001). The discriminate approach tries to model the decision boundaries directly. Examples of this approach include perceptrons (Rosenblatt, 1958) and Support Vector Machines (SVM) (Boser et al., 1992). The regression approach models the posterior class probabilities explicitly. Logistic regression (Press and Wilson, 1978) and decision trees (Breiman et al., 1984) are examples of this approach. The classifiers created using the class-conditional approach are usually called Bayesian, but they are not necessarily Bayesian in the formal sense of parameter estimation. The main difference between these approaches is that the first two focus on the differences between classes while

the last one focuses on the distributions of measurement vectors for the classes.

3.3.4. Regression

In regression the model outputs a numeric value or values. The simplest and most widely used approach is linear regression (Draper and Smith, 1981), where the parameters in linear position are estimated with the least squares technique, for example. This is discussed in more detail in Chapter 4.4.

One method that is important nowadays for non-linear regression is the feed-forward neural network. It is able to learn the relationships between input and output variables using non-linear elements and their weights to capture the essential features of the data. Sequential parameter estimation, which is based on iteration, is normally used for this purpose (Bishop, 1995). Multi-layer perceptron (MLP) and its learning algorithms are described below in Chapter 4.

Besides the problem of effective parameter estimation with non-linear regression models, the main concern more universally is generalization, i.e. whether the constructed model can estimate values based on data not used in training it. Linked to this, it is also important to use proper statistical parameters to describe the goodness of the model as well as creating error estimates for them. These issues are discussed in more detail in Chapter 4 and they have also been addressed in the publications included in this thesis.

3.3.5. Predictive modelling for time-series data

Environmental processes are usually described through time-series which has been measured over some interval of time. Even though time-series processing can be dealt with as a classification or regression problem to a large extent (Dorffner, 1996), the temporal dimension usually requires special attention.

A time-series is a series of values of a variable at successive times, $\boldsymbol{x}(t) = (x(t_1), x(t_2), x(t_3), \ldots$ (Papoulis, 2002). This means that the samples of the variable (or data of several variables) are ordered in time. Usually the variables are sampled uniformly so that the time interval $t_i - t_{i-1}$ between any two samples is the same. In this case, we are using a so-called implicit representation of time (Haykin, 1999). The opposite is explicit representation, where the sampling period can be non-uniform.

Time-series analysis and processing is a traditional and developed field of its own, having many elements in common with the signal processing field (Dorffner, 1996). The classical methods include the autoregressive (AR) and moving average (MA) models, as well as enhanced versions and combination of them (Box and Jenkins, 1976). However, they are outside the scope of this thesis.

One of the most important things to bear in mind in time-series modelling is that what is actually modelled is (or should be) the process we are inspecting. Thus, the data available describe that process only in some limited way, so, knowledge about the process is important. Operational use of time-series modelling aims at producing practically usable information about the value of some variable in the future. Thus, the words most often encountered in that context are forecasting and prediction. Despite differences in definition, they are used as synonyms in the context of time-series modelling.

In general, time-series analysis has three goals: forecasting (predicting), modelling, and characterization (Weigend and Gershenfeld, 1994). The aim of forecasting is to accurately predict the short-term evolution of the system. For example: if we know the concentration of NO_2 now and what is has been during the several past years, is it possible to forecast the concentration for tomorrow morning?

On the other hand, the goal of modelling is to find a description that accurately captures features of the long-term behaviour of the system. This could be exemplified by the following procedure: using the least squares technique and sine and cosine functions, one can fit the NO_2 data of the past several years to show the seasonal, weekly and daily variations in the time-series due to meteorological conditions and human activities.

Note that the goals of forecasting and modelling are not necessarily identical. Thus, finding governing equations with proper long-term properties may not be the most reliable way to determine parameters for good short-term forecasts. Correspondingly, a model that has good short-term properties may have incorrect long-term ones.

The third part of time-series analysis is time series characterization. This attempts, with little or no a priori knowledge, to determine fundamental properties such as degrees of freedom of a system or the amount of randomness.

3.4. Learning and understanding

One viewpoint of modelling using CI methods is to consider strong models and weak models (Weigend and Gershenfeld, 1994). Thus, the concepts of learning and understanding can also be sharpened and given a practical meaning for modelling different kinds of phenomena.

Strong models are based on strong assumptions about the phenomenon inspected. The process is thus usually described with few equations (such as partial differential equations) with few parameters. Such models are able to explain the phenomenon and are thus related to understanding.

On the other hand, weak models make only a few domain-specific assumptions, which leads to more parameters. Thus, clear interpretation of the phenomenon is difficult because the parameter set learned is not usually connected directly to the phenomenon inspected. In that case we can be learning but not understanding.

The neural network-based models used extensively in this work are weak models and thus include a lot of tuneable parameters. This can cause another side effect, memorization (Weigend and Gershenfeld, 1994). In practise this means that the training examples are learned by heart. This is one form of overfitting, where the model captures noise in addition to the real function to be learned. Overfitting is thus possible if the model is too complex, i.e. it has a lot of tuneable parameters. On the other hand, if the model is too simple it cannot capture all the characteristics of the target function.

Therefore, the real goal of learning is generalization (Bishop, 1995). In practise this means that the model can be successfully applied to new data. Thus, the following undesirable chain action is of most concern when trying to make good forecast models for time-series data: weak models => many parameters => overfitting => no generalization. The traditional way to avoid overfitting with neural networks is early stopping, where the original data are usually split into the three datasets of training, testing and validating. The training set is used to construct the model, whereas the testing set sets the limit for stopping the training before overfitting occurs. Finally, the validation set is used to check the generalization properties of the model. However, during recent years, regularization has been used increasingly for the same purpose. It is covered in more detail in Chapter 4.11.

4. METHODS FOR INTELLIGENT DATA PROCESSING

The data processing chain in knowledge discovery (Fayyad, 1996) consists of several steps, illustrated in Figure 3. The process is initiated by selecting the data from the database. Cleaned data are achieved by pre-processing the target data using several optional methods, which are described below. The cleaned data are then prepared for the data-mining step by transforming the variables or data-rows into more suitable scales of representation. The data can also be reduced to fewer variables using feature-selection techniques. The data-mining stage can then utilize CI methods as well as traditional methods such as linear regression. The stages before data-mining are briefly described below, followed by a description of the methods in the data-mining stage used in this thesis. Note that the articles in this thesis contain descriptions of the corresponding methods used. In most cases, however, the methods are described in general terms, because it was thought that the editors and readers would prefer to have results of the discipline itself rather than formal mathematical descriptions. Moreover, the journals selected for publishing the articles are in different disciplines and it was thus necessary to provide some details of the methods, and this has led to similarities between the methods parts in the articles.

4.1. Pre-processing the data

The target data are usually given as a data matrix **X**, consisting of data rows and columns. The columns correspond to measurement variables and are also called attributes or feature values. The rows correspond to units of measurement (e.g. gene or study subject) or to different points of time. They are also called samples or data lines. The format of the data matrix is described in Equation 1.

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \cdots \\ \mathbf{x}_n \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \cdots & \cdots & \cdots & \cdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \quad (1)$$

where **X** is the data matrix, n is the number of samples and p is the number of variables. Missing or erroneous data complicate the modelling of large data sets, because most data processing applications require complete data matrices before analysis (Bishop, 1995). For this reason, imputation techniques are widely needed but it can be very difficult to select a suitable imputation method and using wrong

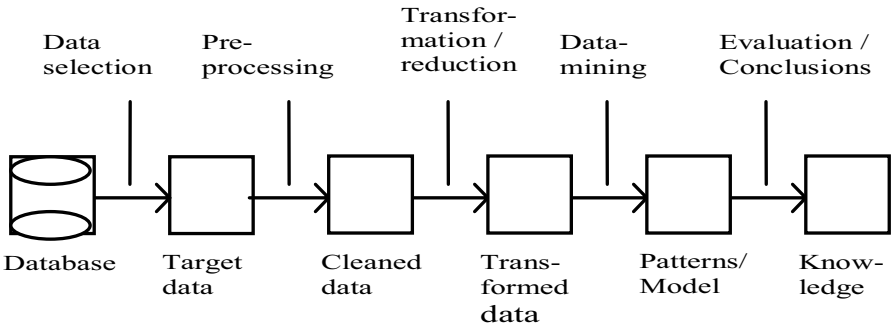

Figure 3. The data processing chain in knowledge discovery. Note that for simplification any possible loopbacks in the chain have been omitted (modified from Fayyad, 1996).

methods will lead to defective interpretations. The problem of missing data in most environmental and bioinformatics data sets is thus complicated by the large amount of missing values caused by measurement errors, breakdowns and human mistakes.

Tens of different types of imputing techniques have been developed since 1980 (Little and Rubin, 1987; Schafer, 1997). The two main ways to handle missing data are case deletion and imputing. In case deletion, all incomplete rows (cases) of the data matrix are discarded, whereas in imputing they are filled in by some technique. One commonly used method for imputing missing data is to fill in the missing values by setting them to the mean value or other statistical parameter. However, these simple techniques are not recommended since using them can severely alter the data distribution (Ghahramani and Jordan, 1994). More advanced imputing techniques are based on history data or a model such as density estimation, regression or neural networks. This is discussed in more detail in Kolehmainen et al. (2002) and Junninen et al. (2002) in the context of air quality data.

4.2. Data transformation and dimensionality reduction

Data transformations are usually needed before applying the methods themselves, for the following reasons:

- The magnitudes of the variables differ so the variables with largest numeric values tend to dominate the learning process.
- The variable is cyclic, so it includes discontinuities.
- The data distribution does not enable the algorithm to recover important features.
- There are outliers, which suppress important features.

Note that by applying the transformations, different aspects of the data can be recovered. The transformations can be applied to different pieces of the data. The most often encountered are:

- Transformation of one value only (e.g. logarithmic transformation and transformations using sine and cosine functions for handling cyclic variables).
- Transformation of one variable (column), usually based on statistical properties of the variable such as min, max, mean and variance.
- Transformation of one vector (row), usually based on the length of the vector or on its statistical properties.
- More advanced transformations such as data whitening, taking the correlations of the variables also into account.

The most often encountered types of transformations are the ones for one variable or for one vector. In the first case, either of the following two transformations is usually applied. In the equalization of variables, each column is scaled linearly so that its minimum is zero and maximum is one. It is defined as

$$c_i' = \frac{c_i - c_{\min}}{c_{\max} - c_{\min}} \tag{2}$$

where c_i is the original value, c_i' is the transformed value, c_{min} is the minimum element of the column vector **c** and c_{max} is the maximum element of the column vector **c**. Note that it is assumed here that $c_{min} \neq c_{max}$. Equalization of variables is used to transform the original values of several variables into a comparable range, thus preventing any from dominating the learning process. It keeps all the values in a range where they have an effect on learning. Therefore, if there are bad outliers, variance scaling of variables should be used instead. It is defined as follows:

$$c_i' = \frac{c_i - \bar{c}}{c_\sigma} \tag{3}$$

where $\bar{c}$ is the mean of the values in vector **c** and c_σ is the standard deviation of the values in it. Note that it is assumed here that $c_\sigma \neq 0$. Variance scaling of a variable is thus an important tool for diminishing the effect of outliers in the data while also equalizing the effect of all variables transformed. However, it is sometimes necessary to suppress the magnitude of individual variables and focus on the relative values of variables in each data vector (row). This can be achieved by normalizing each row of data, i.e. the length of the vector is scaled to one. It is defined as:

$$r_i' = \frac{r_i}{\|\mathbf{r}\|} \tag{4}$$

where r_i is an original element of the vector **r**, r_i' is the transformed element and $\|\mathbf{r}\|$ is length of vector **r**.

It is often the case that some of the measured data are not useful for the information processing being carried out: some of them can even be misleading. This can happen basically in two ways. First, some variables or sensors are not needed at all. On the other hand, some of the feature vectors may be outliers so that they are not part of any cluster formed by the rest of the vectors. Feature selection and feature extraction aim at reducing the dimensionality of the data in order to solve the previously mentioned problems (Bishop, 1995). It can be achieved by several algorithms based on multivariate statistical analysis or using the constructed model itself iteratively with GA, for example.

4.3. Metrics

The metric used for comparing two vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ has a crucial effect on the learning algorithm due to its ability to account for different aspects of the data. Most of the time, Euclidean distance metric can be used:

$$\mathrm{d}_{\mathrm{Euc}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{x}_i - \mathbf{x}_j)} \tag{5}$$

On other occasions other alternatives must be considered (Diday and Simon, 1980). Note also that the transformations used in the pre-processing stage can have the same or similar effects as changing the metric used. Euclidean distance metric assumes some commensurability between the variables and this must usually be assured by pre-processing using some suitable transformation, typically by variance scaling. Note that Euclidean distance metric is a special case of Minkowski L_λ metric with $\lambda = 2$. Correspondingly, other commonly used metric is city-block or Manhattan distance namely L_1.

Pearson's product moment correlation coefficient (Diday and Simon, 1980) can be used instead of Euclidean metric. It is a measure of the linear association between two variables. Given vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ defined as above, it can be defined as

$$d_{\mathrm{PCC}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{k=1}^{p} (x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j) / p}{\sigma_i \sigma_j} \tag{6}$$

where $\bar{x}_i$ is the mean and σ_i the standard deviation of the values in the vector $\mathbf{x}_i$ and p is the number of elements in the vector.

Moreover, in order to compensate for different variances of the variables and for correlations between variables, the Mahalanobis distance (Diday and Simon, 1980) can be used. It is based on using the covariance matrix $\mathbf{\Sigma}$ as part of the equation for calculating the distance between two vectors:

$$\mathrm{d}_{\mathrm{MH}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{\Sigma}^{-1} (\mathbf{x}_i - \mathbf{x}_j)} \tag{7}$$

4.4. Linear regression

Linear regression aims at fitting an equation to some data (Berthouex and Brown, 2002). An example of such equation is as follows:

$$z = a + bx + cx^2 + dxy + e \tag{8}$$

where z is the dependent variable, x and y are independent variables, a, b, c and d are the parameters to be estimated and e is the residual error. Thus, even though there are non-linear terms (x^2 and xy) their values can be calculated from measurements and therefore they pose no problem in fitting. In contrast, the parameters to be estimated must be in linear positions and thus justify the term "linear regression". Note that regression in general is also called curve fitting or parameter estimation.

Another example, used in one of the author's articles (Kolehmainen et al., 2001), fits a basic level, line (trend) and several sine and cosine functions of different multifolds to the signal consisting of NO_2 concentration levels measured with an interval of one hour:

$$z_t = \sum_{k=1}^{m} (a_k \sin k\omega_0 t + b_k \cos k\omega_0 t) + \sum_{l=0}^{q} c_l t^l + e_t \quad (9)$$

where z_t is NO_2 concentration level, m is maximum multifold of the sine and cosine terms, q is maximum multifold of curve fitting, ω_0 is basic frequency (e.g. $\omega_{\text{year}} = 2 * \pi / (365 * 24)$, since the resolution of measurement is 1 hour), a_k, b_k and c_l are coefficients to be resolved by fitting and t is time.

The most commonly used approach in estimating the parameters is the method of least squares (LS). It is based on minimizing the sum of squared residuals:

$$S = \sum_{i=1}^{n} (e_i)^2 = \sum_{t=1}^{n} (z_t - \hat{z}_t)^2 \quad (10)$$

where z_t are the measured values of n measurements and $\hat{z}_t$ are the computed values from the model. Note that in order for the parameter estimates to be unbiased, the residual errors should be random, have a zero mean and be independent with constant variance. Consequently, this leads to normal equations, which have an algebraic solution. However, as the number of parameters increases, it is customary to use matrix algebra in handling the data and the parameters. Using matrix notation, the model can be written as follows:

$$\mathbf{z} = \mathbf{H}\boldsymbol{\beta} + \mathbf{e} \quad (11)$$

where $\mathbf{z}$ includes the values for dependent variable, $\mathbf{H}$ includes the independent variables, $\boldsymbol{\beta}$ includes the parameters and $\mathbf{e}$ includes the residual errors. Consequently, the least squares estimate for the parameters is (Berthouex and Brown, 2002):

$$\hat{\boldsymbol{\beta}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{z} \quad (12)$$

For non-linear equations no algebraic solution exists and parameter estimation is usually carried out using iterative methods such as Gauss-Newton iteration (Berthouex and Brown, 2002).

4.5. Self-organizing map

4.5.1. The basic SOM

The self-organizing map (SOM) is one of the best known unsupervised learning methods (Kohonen, 1997). It can be defined as follows.

Let $\mathbf{X}$ be the data matrix of p measured variables for n samples as defined in Equation 1. The self-organizing map consists of M neurons organized as a 2-dimensional lattice, each having in its weight vector, $\mathbf{w}_m = (w_{m1}, w_{m2},\ldots,w_{mp})$, ($m = 1,\ldots,M$) as many weights as there are measured variables. The weight vectors of the SOM are first initialized to random values. With each vector $\mathbf{x}_i$ the winning neuron (Best-Matching Unit, BMU) is first found. The Best-Matching Unit is the neuron which is closest, or most similar, to the input vector. The Best-Matching Unit for the ith input vector is found by comparing the input vector $\mathbf{x}_i$ and weight vectors $\mathbf{w}_m$ of M neurons by the Euclidean distance metric defined in Equation 5.

The Best Matching Unit (BMU) is thus the neuron being at the smallest Euclidean distance from the input vector:

$$c(\mathbf{x}_i, \mathbf{W}) = \arg\min_j \left\| \mathbf{x}_i - \mathbf{w}_j \right\| \quad (13)$$

where **W** includes the weight vectors of the SOM. The weights of the BMU and the neighbouring neurons (according to the neighbourhood function) are corrected towards the input vector using equation 14 (update rule).

$$\mathbf{w}_{\mathrm{m}}(t+1) = \mathbf{w}_{\mathrm{m}}(t) + h_{cm}(t)\left[\mathbf{x}_{\mathrm{i}} - \mathbf{w}_{\mathrm{m}}(t)\right] \quad (14)$$

where t is a counter for iterations, c is the index for the BMU and m is the index for the neuron to be updated. The neighbourhood function can be defined as follows for a Gaussian function (Kohonen, 1997):

$$h_{cm}(t) = \alpha(t) \exp\left(-\frac{\left\| \mathbf{r}_c - \mathbf{r}_m \right\|^2}{2\sigma^2(t)}\right) \quad (15)$$

where $\mathbf{r}_c$ and $\mathbf{r}_m$ are the location vectors for the corresponding nodes, $\sigma(t)$ defines the width of the kernel and $\alpha(t)$ is a learning rate factor. The learning rate factor and kernel width decrease monotonically towards the end of learning as a function of time. Note that in practical implementations of up to hundreds of neurons, the selection of these parameters is not crucial (Kohonen, 1997). Also, a simple neighbourhood function consisting of just a set of neurons around the BMU can be used in those cases instead of the Gaussian function.

However, the neighbourhood radius must be large enough at the beginning of the learning process to enable global ordering to take place. A value larger than half of the radius of the network is suggested (Kohonen, 1997). Additionally, it is suggested as a general procedure that the learning is split into two phases, with the first consisting of 1000 steps. During this phase, both the neighbourhood radius and learning rate factor shrink, with the latter starting from a value near 1. During the second phase, fine adjustment takes place with a learning rate factor of the order 0.02 or less, and the neighbourhood consisting only of the nearest neighbours of the BMU. The convergence typically needs a step number of at least 500 times the number of neurons in the network.

To summarize, the SOM algorithm proceeds as follows:

1) Find the BMU for one input vector according to the minimum Euclidean distance.
2) Move the weight vector of the BMU towards that input vector, using the update rule.
3) Move the weight vectors of neighbouring neurons (according to the neighbourhood function) towards that input vector, using the update rule.
4) Repeat steps 1-3 for the next input vector until all input vectors have been used.
5) Repeat steps 1-4 until convergence.
6) Find the final BMU (the neuron which the individual belongs to) for each input vector according to the Euclidean distance.

4.5.2. Examples of the SOM

Most CI methods work in p-dimensional space, which consists of the measured variables in the data. Thus, one variable (such as wind speed) represents one dimension. As the dimensionality (number of variables in the data) grows beyond three up to tens or hundreds of variables, it is more appropriate to think of each data row (all the variables measured at the same time, for example) as a vector of p components. Figure 4 shows how a gene expression data set consisting of two time points is transformed into vectors in two dimensions, and how the weight vectors of the self-organizing map stabilize from originally random locations to represent the density of the data. Consequently, the mental action needed for understanding the working of most CI methods is to be able to

imagine the data as vectors of this kind but now in p dimensions instead of the normal two or three.

The basic idea behind the SOM algorithm is that the weight vectors of neurons gradually, during the learning process, come to represent a number of original measurement vectors (see Figure 5). They can then be used as a basis of further analysis more easily than the original measurement vectors. This is because of the reduced number of data. Also, the weight vectors represent the centre-of-mass points of the clusters of measurement data and they can thus be used to find the grouping of the data.

An example of a SOM is illustrated in Figure 6, where the study subjects (1650 participants) of the Kuopio Ischemic Heart Disease Risk Factor Study (KIHD) were grouped according to 25 biochemical and physiological variables. The bar graph represents six central variables of the study and it can be seen that their distribution on the SOM map varies consider-

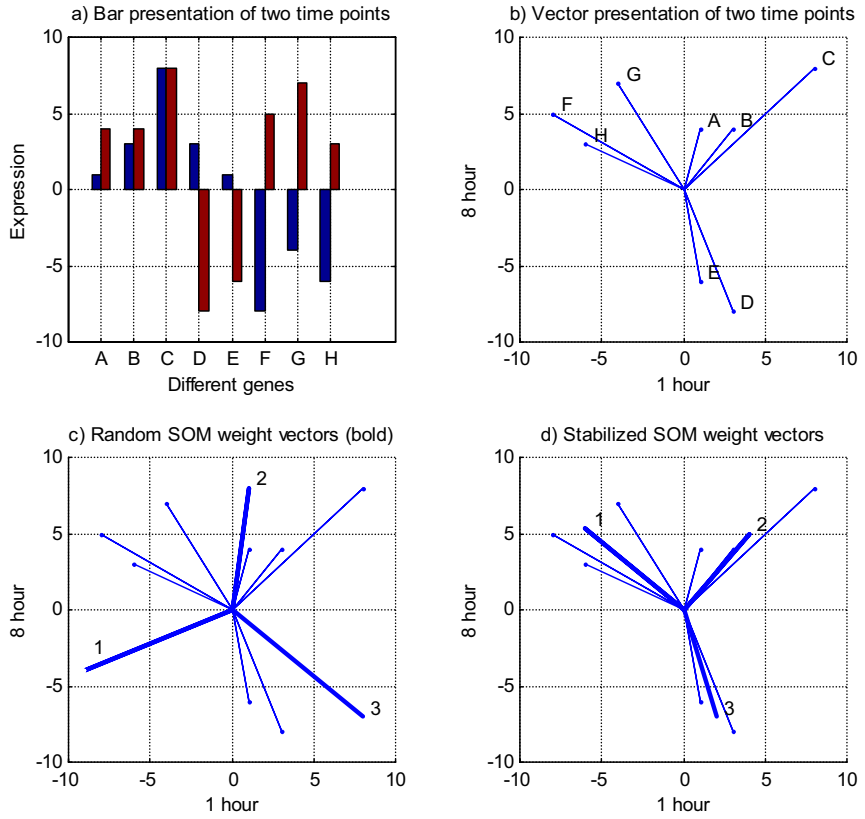

Figure 4. A simple representation of a gene-expression profile with two time points as coordinate vectors. (a) Eight genes are labelled A-H. The bar graphs correspond to expression levels measured at time points of 1 and 8 hours. (b) The expression levels have been transformed into vectors. A vector (line) which has a length and a direction now represents each gene. (c) The weight vectors of the SOM (bold lines numbered 1, 2 and 3) are first assigned to random values. (d) The weight vectors are moved towards the input vectors using an iteration process. The weight vectors stabilize themselves into configurations where they each represent a group of the original measured vectors, i.e. gene profiles A-H (modified from an original idea of Garry Wong, University of Kuopio).

ably. Note especially, how the SOM is able to create a continuous mapping where different combinations of high and low values of the variables presented have been spread smoothly around the map.

4.5.3. The tree-structured SOM

A variation of the SOM, the tree-structured SOM, has also been constructed (Koikkalainen, 1994). The software implementation consists of several SOMs organized hierarchically in a pyramid-like fashion in several layers. The number of neurons at a lower level is four times the number at the previous level. However, visual inspection of the measurement data is directed to one level at a time and it can thus be used for data exploration similarly to a "standard" SOM.

The learning rule of the TS-SOM has been modified and is described in Equation 16.

$$\mathbf{w}_{\mathrm{m}}(t+1) = \mathbf{w}_m(t) + a(\mathrm{t})\left[\mathbf{x}_i - \mathbf{w}_m(t)\right] \qquad (16)$$

Thus, its neighbourhood function has been reduced to a fixed form where the neighbourhood is always defined to be the four adjacent neurons of the BMU. This is possible since the role of the altering neighbourhood radius is handled by the previous (more coarse) levels of the pyramid-like structure of several SOMs. This structure is illustrated in Figure 7.

As found by Koikkalainen (1994), there is no difference in the topological ordering of a well trained "standard" SOM and TS-SOM. The vector quantization performance (i.e. how well the weights of the network represent the original data), however, is worse to some degree with the TS-SOM, but this can be compensated for with the full search for assigning the data to the neurons after training. The discriminant surfaces of the TS-SOM are convexes that are similar to the "standard" SOM and, again, the full search can be applied after training to produce true Voronoi convexes, i.e. areas in the input space for which the weight vector is the BMU.

4.6. Sammons's mapping

Sammon's mapping (Sammon Jr, 1969), is a non-linear mapping algorithm, which is closely related to the metric version of multi-dimensional scaling (MDS) (Torgerson, 1952). The aim of the algorithm is to represent points of *p*-dimensional space in 2-dimensions. The algorithm tries to find the locations in the 2-dimensional target space so that the conservation of the original structure of the measurement vectors in *p*-dimensional space is maximized. The error of the structure conservation is measured as follows:

$$E = \frac{1}{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} d_{ij}} \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \frac{(d_{ij} - d'_{ij})^2}{d_{ij}} \qquad (17)$$

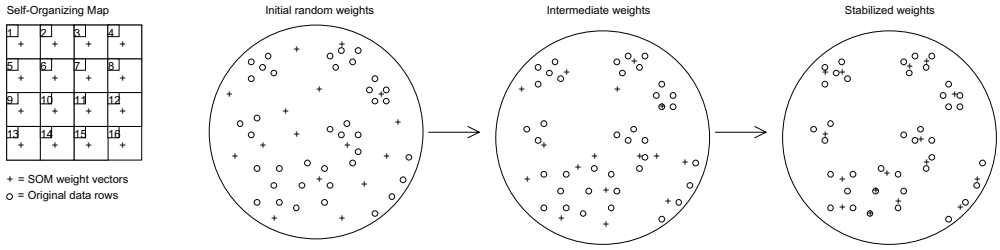

Figure 5. The learning procedure of the self-organizing map. Initially randomly distributed weight vectors (marked by +) are driven into a stable configuration defined by the input sample vectors (marked by o) (modified from Kolehmainen et al., 2003).

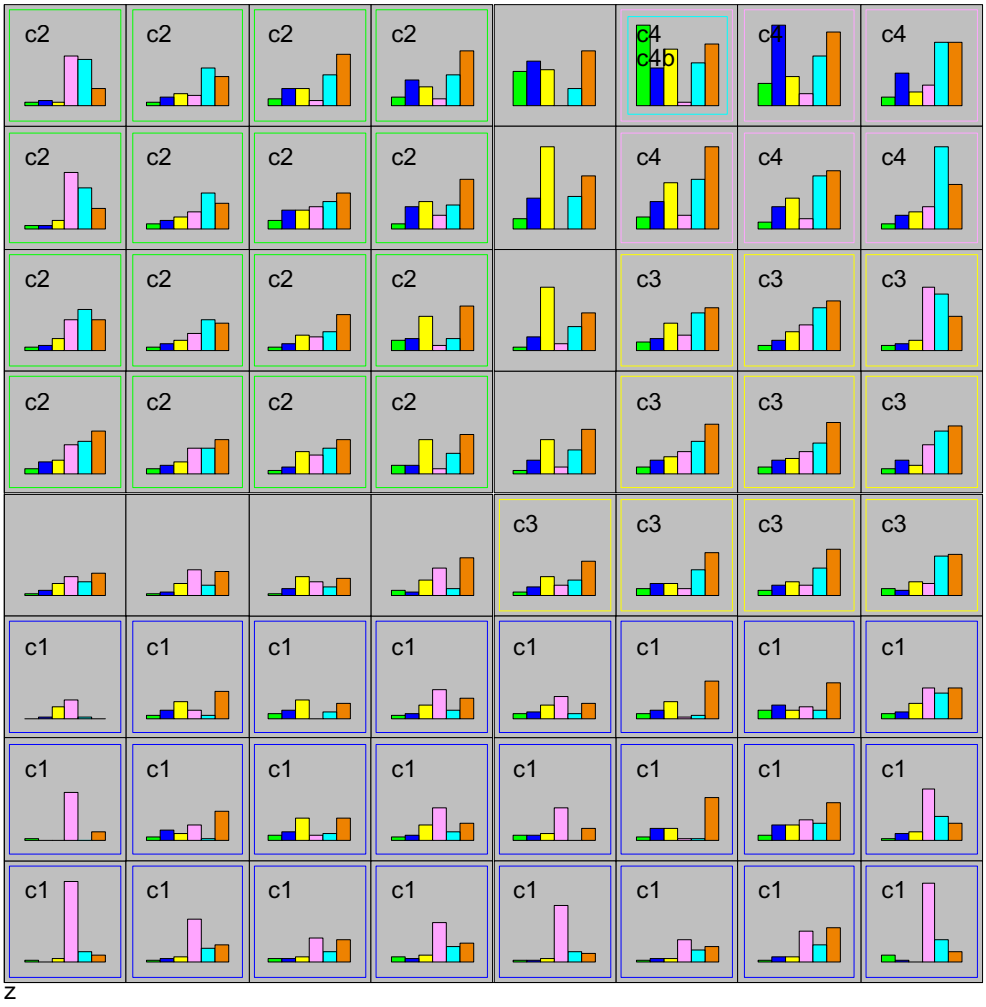

Figure 6. Self-organizing map with bar graphs showing the distribution of the central physiological variables: blood glucose (1st bar), serum insulin (2nd bar), triglycerides (3rd bar), HDL cholesterol (4th bar), systolic blood pressure (5th bar) and waist-to-hip ratio (6th bar) (modified from Valkonen et al., 2002).

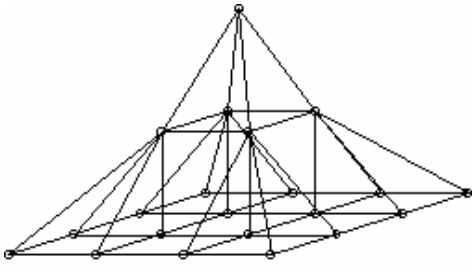

Figure 7. The pyramid-like structure of several SOMs in the TS-SOM implementation. Each node represents one neuron so that the SOM in level 1 consists of 4 neurons and that in level 2 consists of 16 neurons. Additionally, there is level 0, which consists of only one neuron (modified from the NDA user's guide).

where n is the number of data points, d_{ij} is the (Euclidean) distance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ in the original space, and d'_{ij} is the (Euclidean) distance between the corresponding points $\mathbf{x}'_i$ and $\mathbf{x}'_j$ in the lower dimensional target space. Minimization is usually based on pseudo-Newton minimization (Becker and Le Cun, 1989), which is also called steepest descent in some publications (e.g. Kohonen, 1997). The positions in the target space are thus updated as follows:

$$x'_{ip}(t+1) = x'_{ip}(t) - \alpha \frac{\partial E(t)/\partial x'_{ip}(t)}{\left|\partial^2 E(t)/\partial x'_{ip}(t)^2\right|} \qquad (18)$$

where x'_{ip} is the pth coordinate of the position of the point in the target space. The factor α must be set experimentally and has an effect on the convergence of the algorithm. A value of 0.3-0.4 is usually suggested for it even though this value is not suitable for all data sets (de Ridder and Duin, 1997). The problem with the update rule is the second derivative in areas of low curvature with a small value of the derivative. Thus, other minimization techniques, such as normal gradient descent, could be used (de Ridder and Duin, 1997).

Sammon's mapping is able to represent the relative distances of vectors in a measurement space and is thus useful in determining the shape of clusters and the relative distances between them. However, due to the normalization of distance preservation errors, small errors are slightly emphasized (Kaski, 1997). Even though the SOM and Sammon's mapping work somewhat similarly, the numerical calculation of the latter is more time consuming. As this can be a problem with a massive data set, it is of benefit to combine these two algorithms. This is discussed in more detail in Chapter 5.

The initialization of Sammon's mapping has been studied by Lerner et al. (2000). They used a neural network implementation of the algorithm, as explained in Chapter 5.2. The results of their study showed that principal components (PCs) based initialization of the algorithm can yield shorter training period, lower mapping error and higher classification accuracy. However, in the implementation used in this thesis, the starting point of Sammon's mapping algorithm is the weights of the SOM, which essentially results in similar initialization as in Lerner et al. (2000).

4.7. U-matrix representation

The U-matrix representation (Ultsch and Siemon, 1990) is a way of describing the relative distances between the neurons in a SOM map. It is calculated by taking the mean of the distances between the p-dimensional weight vectors of neighbouring neurons of the target neuron $\mathbf{w}_i$ as follows:

$$d_{umat}(\mathbf{w}_i) = \frac{1}{N}\sum_{j=1}^{N} \left\|\mathbf{w}_i - \mathbf{w}_j\right\| \;;\; j \in \mathrm{S_{NN}} \qquad (19)$$

where N is the number of nearest neighbour neurons for $\mathbf{w}_i$ and $\mathrm{S_{NN}}$ is the set of nearest neighbour neurons of $\mathbf{w}_i$.

The result of the calculation is a number assigned to each neuron on the map. These numbers can then be used as grey level values or as a z-axis dimension in visualization, for example. Use of the U-matrix is illustrated in Figure 8, where different phases of a bio-process have been presented using the SOM and U-matrix representation.

4.8. Fuzzy logic

Fuzzy logic and fuzzy set theory are a mathematical way to handle uncertainty (Zimmermann, 1991). The basic difference between fuzzy logic and ordinary logic is based on taking into account more truth values than just 'true' or 'false'. This can be understood as a kind of grey-scale property instead of using just black and white. The background for formally handling this comes from the basic property of fuzzy sets. Fuzzy sets include their elements wholly or just partially. This makes it possible to model the uncertainty and vague-

ness that is confronted in real life (Niemi, 1996). It should also be noted that fuzziness and probability are different concepts even though they sometimes seem similar and are often confused with each other.

Formally, a fuzzy set is defined as

$$\tilde{\mathrm{A}} = \{(x, \mu_{\tilde{A}}(x)) | \, x \in \mathrm{S}\} \tag{20}$$

where x is an object, S is a set, and $\mu_{\tilde{\mathrm{A}}}$ is a membership function (see also Figure 10), which for a normal fuzzy set maps object x in set S into interval [0,1]:

$$\mu_{\tilde{A}} : \mathrm{S} \rightarrow [0,1] \tag{21}$$

The use of fuzzy logic in practical implementations can easily be understood in process control. The basic structure of such a fuzzy system is illustrated in Figure 9. The fuzzification module takes the crisp (non-fuzzy) values measured from the process and transforms them into fuzzy values. The rulebase module includes the knowledge that is needed to handle the process. This is in the form of if-then rules and is usually formed from the expertise of a human operator. The inference engine takes the fuzzified input values, and calculates the output using a rulebase. This output is then transformed back to a crisp value that is used to initiate a control action to the process.

The fuzzification process is based on the two concepts of linguistic variable and membership function. A linguistic variable tries to model terms used in everyday language. It is further divided into adjectives that are described by membership functions. This structure is illustrated in Figure 10. The linguistic variable presented here as an example is "temperature". It consists of the adjectives "cold", "cool", "warm", "hot" and "very hot". The membership functions give the degree of membership of each adjective in the base set.

Fuzzy system applications can be found in a variety of different areas ranging from transportation and industrial applications to those in consumer electronics (Bonissone et al., 1999). In these applications, fuzzy logic is normally used as an expert system, implementing human operators or process engineer's

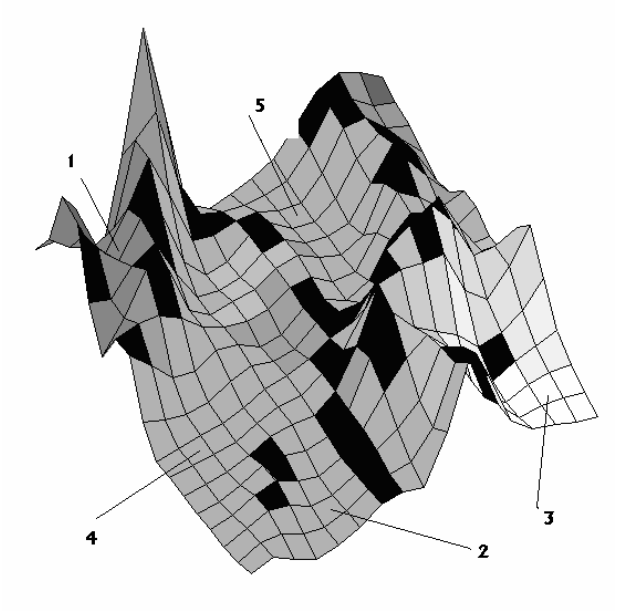

Figure 8. Phases of a bio-process illustrated using the SOM and U-matrix representation. The regions numbered 1-5 each represent one process phase where the measurement device signal varies less than between the phases. Thus, the "valleys" of the U-matrix representation are coherent areas of the SOM where the neighboring neurons are similar in their content (modified from Kolehmainen et al., 2003).

expertise. This is expressed in the form of if-then rules.

If there are a large number of input variables, it is not always feasible to create membership functions for each input variable separately. Instead, the input space can first be transformed into a low dimensional space (Jang et al., 1997). In this work the transformation was accomplished by calculating the Euclidean distance between an input vector and the reference vectors forming the kernel of the set. Thus, the use of fuzzy logic was in the form of fuzzy group membership evaluation. This is illustrated in Figure 11. The kernels of the sets are areas of the input space where a certain phenomenon was supposed to be true (e.g. an air quality episode where the pollutant concentrations are above certain limit). For a detailed example of this, see Kolehmainen et al. (2000) (included in this thesis).

4.9. C-means and fuzzy c-means clustering

C-means (MacQueen, 1967) is an algorithm for partitioning (clustering) *n* data points into *C* disjoint subsets S_j containing n_j data points so as to minimize the sum-of-squares criterion:

$$E = \sum_{j=1}^{C} \sum_{i \in S_j} \left\| \mathbf{x}_i - \mathbf{u}_j \right\|^2 \qquad (22)$$

where $\mathbf{x}_i$ is a vector representing the *i*th data point and $\mathbf{u}_j$ is the centroid of the data points in S_j. The algorithm consists of a simple re-estimation procedure, as follows (Bishop, 1995). First, the data points are assigned at random to the *C* sets. Then the centroid is computed for each set. These two steps are alternated until a stopping criterion is met, i.e., when there is no further change in the assignment of the data points. An example of the result of applying the c-means algorithm to air quality data is shown in Figure 12. Note that the c-means algorithm is also called k-means in some sources. However, the term c-means underlines the connection between this algorithm and fuzzy c-means clustering (see below), which is a fuzzy extension of c-means clustering.

Fuzzy c-means (FCM) is a data clustering technique wherein each data point belongs to a cluster to some degree specified by a membership grade. This technique was originally introduced by Bezdek (1981) as an improvement on earlier clustering methods such as c-means. The algorithm is initiated by assigning the cluster centres to random locations. The number of clusters and the degree of fuzziness are decided by the user. Next, the membership values for each data point to each of the cluster centres are calculated. New locations for the cluster centres are then calculated as a mean

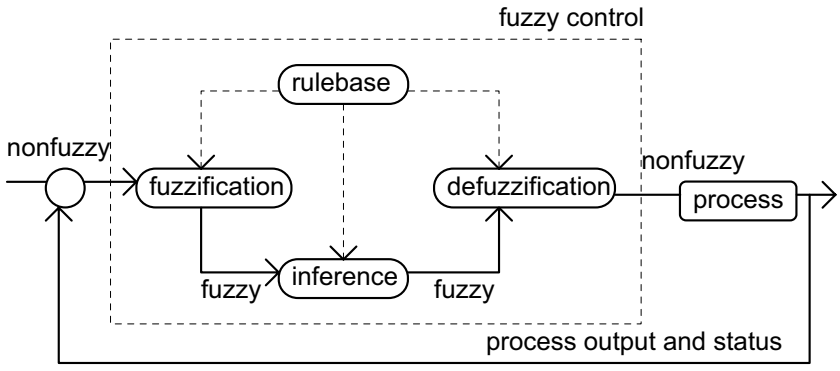

Figure 9. General structure of fuzzy system (modified from Fool & Fox, 1999).

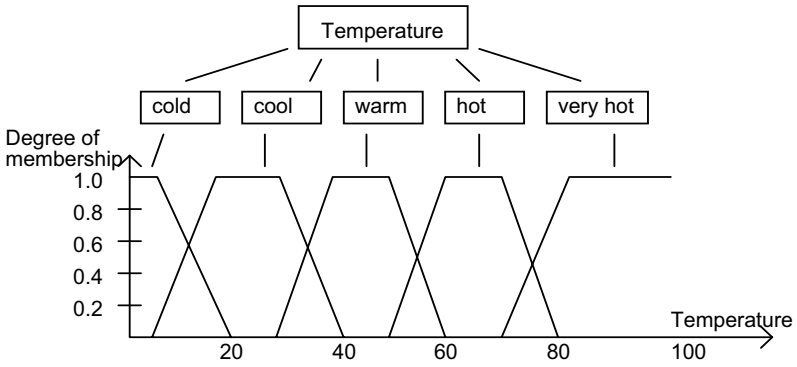

Figure 10. A linguistic variable and membership functions (modified from Niemi, 1996).

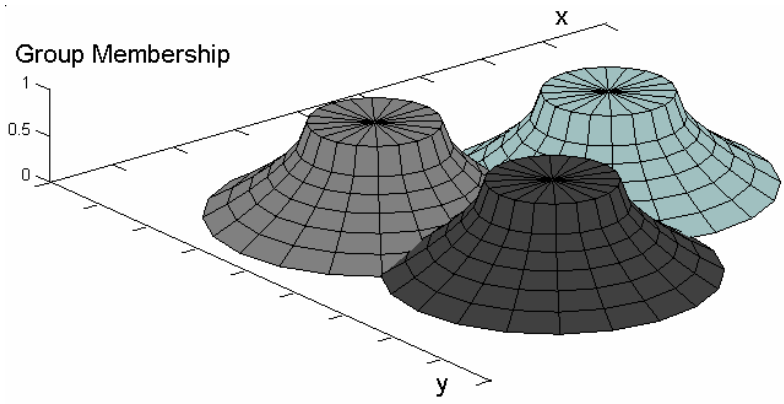

Figure 11. A general illustration of multi-dimensional kernels and their membership functions. The kernels can be seen as plateaus, and the membership value of each data vector diminishes as its distance from the kernel grows. Note how the different membership functions can overlap with each other (modified from the NDA user's guide).

value of data points, weighting it with the membership values. These two steps are iterated until the change of partition is under a preset value. The algorithm thus corresponds to the minimization of the following problem (Zimmermann, 1991):

$$E = \sum_{i=1}^{C} \sum_{k=1}^{n} (\mu_{ik})^m \left\| \mathbf{x}_k - \mathbf{v}_i \right\|^2 \qquad (23)$$

$$\mathbf{v}_i = \frac{1}{\sum_{k=1}^{n} \mu_{ik}} \sum_{k=1}^{n} (\mu_{ik})^m \mathbf{x}_k \; ; \mathrm{m} > 1 \qquad (24)$$

where μ_{ik} is the membership value of data point k to cluster centre i, $\mathbf{x}_k$ is a data point, $\mathbf{v}_i$ is a cluster centre, C is the number of clusters, m is the degree of fuzziness, and n is the number of data points.

Measuring the goodness of the clustering achieved is essential for at least two reasons: to permit the selection of the right number of

clusters, and the comparison of different models. For c-means clustering, the Davies-Bouldin index (Davies and Bouldin, 1979) can be used. It is calculated according to the following equation:

$$DB_C = \sum_{k=1}^{C} \max_{l \neq k} \left\{ \frac{s_k + s_l}{d_{kl}} \right\} \quad (25)$$

where s_k is the within-cluster distance for cluster k, and d_{kl} is the between cluster distance for clusters k and l. For fuzzy c-means clustering, the PC index (Bezdek, 1981) can be used. It is defined as follows:

$$PC = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{C} (\mu_{ij}(\mathbf{x}_i))^2 \quad (26)$$

where μ_{ij} is the membership function for cluster j.

Different indexes are used for these two clustering algorithms due to their different construction, i.e. in c-means clustering a vector belongs to only one cluster, whereas in fuzzy clustering all the vectors belong to all clusters according to some membership value. Consequently, it is not possible to compare c-means and fuzzy c-means clustering directly, but it is, however, possible indirectly if some model is based on the clustering and the goodness of the model is used instead. An example of this can be found in one study of this thesis (Niska et al., 2003b).

4.10. Neurocomputing

We already saw an example of an artificial neural network in the SOM algorithm. It provides one way of neurocomputing, which is a new way of computing that does not require constructing algorithms or explicit rules. In principle, it is a fundamentally different approach to information processing and the first alternative to programmed computing. Neural networks are the primary information processing structures used in neurocomputing. The processing capabilities are developed by exposing the neural network to an information environment, in contrast to explicit programming (Hecht-Nielsen, 1991).

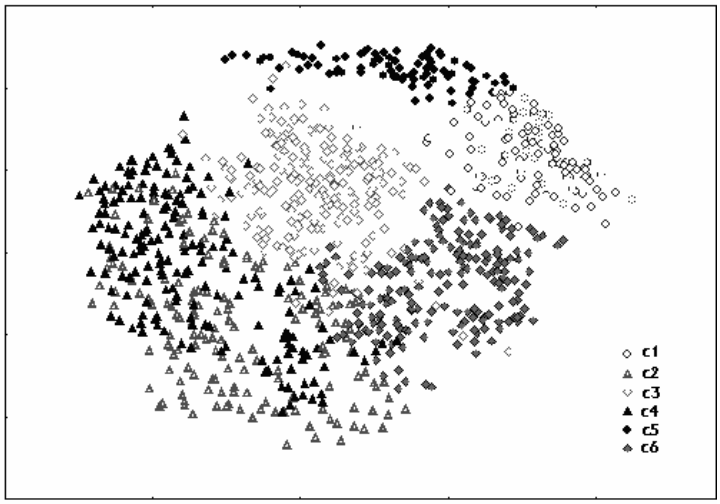

Figure 12. Example of c-means clustering. The data consist of 43 air quality variables measured over one year in Kuopio, Finland, clustered into six clusters named c1-c6. Note that the 43-dimensional cluster structure was reduced to a 2-dimensional surface using Sammon's mapping in order visualize it (unpublished results).

4.10.1. The multi-layer perceptron

Multi-layer perceptrons (MLP), which are one type of feed-forward neural networks, consist of processing elements and connections (Haykin, 1999). Processing elements are usually called neurons, and are arranged as layers. There are three kinds of layers: input layers, hidden layers and output layers (see Figure 13). The input layer serves as a buffer that distributes the input signals to the hidden layer. Each unit in the hidden layer sums its input, handles it with a transfer function and distributes the result to the next layer, which is usually the output layer. The units in the output layer compute their output similarly. Note also that there can be more than one hidden layer. The connections have weights associated to them, which are adjusted during learning using a learning rule (e.g. Equation 33) which states the adjustments of the weights to be done at that instant.

An introduction and overview of MLP applications in the atmospheric sciences can be found in Gardner and Dorling (1998).

4.10.2. Training the MLP

The most common supervised learning algorithm is the back-propagation (BP) algorithm, also called the generalized (Widrow-Hoff) delta rule (Haykin, 1999). Firstly, the error of the network is defined as follows:

$$e_l = \|\mathbf{y}_l - \mathbf{d}_l\| \tag{27}$$

where, $\mathbf{y}_l$ is the network output with input l and $\mathbf{d}_l$ is the corresponding desired network output.

The performance index (error function to be minimized) of the MLP network then can be written as follows (Hagan and Menhaj, 1994):

$$E = \sum_{l=1}^{n} e_l^2(\mathbf{w}) \tag{28}$$

where n is the number of data input rows, vector $\mathbf{w}$ contains the weights and biases of the network, and $e_l(\mathbf{w})$ contains the error of the network for input row l. By partial differentiation of the performance index in respect of the weights and biases we get Equation 29:

$$\frac{\partial E}{\partial w_m} = \sum_{l=1}^{n} 2e_l(\mathbf{w}) \frac{\partial e_l(\mathbf{w})}{\partial w_m} \tag{29}$$

The corresponding Jacobian matrix contains the first derivatives of these network errors with respect to the weights and biases, as follows:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial e_1}{\partial w_1} & \frac{\partial e_1}{\partial w_2} & \cdots & \frac{\partial e_1}{\partial w_N} \\ \frac{\partial e_2}{\partial w_1} & \frac{\partial e_2}{\partial w_2} & \cdots & \frac{\partial e_2}{\partial w_N} \\ \cdots & \cdots & \cdots & \\ \frac{\partial e_n}{\partial w_1} & \frac{\partial e_n}{\partial w_2} & \cdots & \frac{\partial e_n}{\partial w_N} \end{bmatrix} \tag{30}$$

where N is the number of weights and biases in the network. Thus, by denoting $\mathbf{e} = [e_1, e_2, \ldots e_n]^T$ and by $\mathbf{J}_m$ the m:th column of the Jacobian matrix we get Equation 31:

$$\frac{\partial E}{\partial w_m} = 2\mathbf{e}^T \mathbf{J}_m \tag{31}$$

Consequently, omitting the constant factor 2, the gradient (vector giving the steepest descent of the error) of the performance function E can

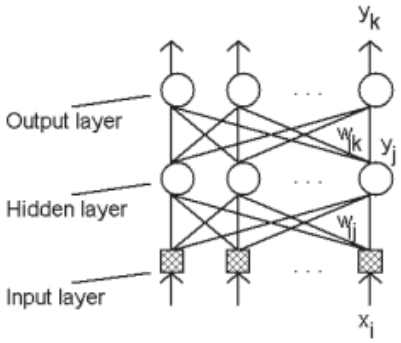

Figure 13. The principle of the MLP network (modified from Pham, 1995).

be calculated (Hagan and Menhaj, 1994) using Equation 32.

$$\mathbf{g} = \mathbf{J}^{T}\mathbf{e} \tag{32}$$

Using the equations above, the back-propagation algorithm updates the current weights as follows, in Equation 33:

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \alpha(t)\mathbf{g}(t) \tag{33}$$

where $\alpha(t)$ is a learning rate factor, $\mathbf{w}(t)$ is a vector of current weights and biases, $\mathbf{g}(t)$is the current gradient for the weights and biases, and t is a counter for iterations.

To summarize the above, the back-propagation algorithm works in two phases, with the first consisting of evaluating the derivatives of the error function in respect to the weights and biases (Bishop, 1995). In the second phase, the adjustments to the weights are calculated using the derivates. The BP algorithm solves the credit assignment problem by propagating the errors backwards in the network. This is possible because the sum-of-squares error function (Equation 28) is a differentiable function of the network outputs, and therefore the error itself is a differentiable function of the network weights (Bishop, 1995).

A major problem with the BP algorithm is that the suitable learning rate and number of nodes in the hidden layer(s) must be determined experimentally. Other issues that have been addressed with a number of variations of BP are poor generalization, slow learning, and local minima (Tsaptsinos, 1995). To overcome these problems, enhanced versions of the BP algorithm have been developed. The simplest ones are based on adding a momentum term to the steepest descent learning rule, which then uses the previous value of the gradient to guide the search (Haykin, 1999). Another method is based on an adaptive learning rate for different parts of the network. Both of these methods try to avoid local minima, which can slow down the learning rate.

More enhanced methods are based on numerical optimization (Haykin, 1999). They usually utilize Hessian matrices, where the Hessian matrix contains the second derivatives of the network errors with respect to the weights and biases. In the conjugate gradient algorithms (Charalambous, 1992), a search is performed along conjugate directions, which produces generally faster convergence than steepest descent directions. Each of the conjugate gradient algorithms requires a line search at each iteration, which is computationally expensive since it requires that the network response to all training inputs be computed several times for each search. The scaled conjugate gradient algorithm (SCG), developed by Moller (1993), was designed to avoid the time-consuming line search. Newton's method is an alternative to the conjugate gradient methods for fast optimization (Battiti, 1992), as it often converges faster than conjugate gradient methods. Unfortunately, it is complex and expensive to compute the Hessian matrix for feedforward neural networks. A class of algorithms based on Newton's method, called quasi-Newton (or secant) methods, have been developed which do not require the calculation of second derivatives.

The Levenberg-Marquardt (LM) algorithm (Hagan and Menhaj, 1994), used in the comprising works of this thesis, was also designed to approach a second-order training speed without requiring computation of the Hessian matrix. The Levenberg-Marquardt algorithm uses an approximation to the Hessian matrix in the following Newton-like update shown in Equation 34:

$$\mathbf{w}(t+1) = \mathbf{w}(t) - [\mathbf{J}^{T}\mathbf{J} + \mu\mathbf{I}]^{-1}\mathbf{J}^{T}\mathbf{e}(t) \tag{34}$$

where μ is a scalar and $\mathbf{I}$ is the identity matrix. The parameter μ is multiplied by some factor β when a step would increase the network performance index and divided by it when a step

decreases the performance index. Consequently, with low μ, the LM becomes Gauss-Newton, and with high μ it becomes steepest descent with a step $1/\mu$. Suitable starting point values could be $\mu = 0.01$ and $\beta = 10$, as suggested by Hagan and Menhaj (1994). Thus, the LM algorithm can be considered as a model-trust modification of the Gauss-Newton algorithm (Battiti, 1992).

4.10.3. Other types of neural networks

Radial-Basis Function Networks (RBFN) are based on Cover's theorem (Cover, 1965), according to which a pattern-recognition problem cast into a high dimensional space is more likely to be linearly separable. RBFNs have three layers: input layer, hidden layer with non-linear functions, and linear output layer. The non-linear hidden layer differs from the MLP in using radial basis functions not used with the MLP. Despite the differences between the MLP and RBFN, it is always possible to find a MLP capable of mimicking the corresponding RBFN and vice versa (Haykin, 1999).

A standard way of utilizing past values with neural networks is to supply them as inputs to the network, as with the MLP. In contrast, recurrent neural networks (RNN) have connections from output or hidden layers to the previous ones. This creates a context memory for the network, which helps it to distinguish between different operating conditions. Several authors have claimed that RNNs are superior to standard ones in time-series processing, for example Gan (2002), who argues that there is a relationship between recurrent and regularized neuro-fuzzy networks. Ulbricht (1994) based his judgment on real-world applications using multi-recurrent networks.

Dorffner (1996) presented an overview of different neural network types for time-series processing, identifying the following RNN types:

- Jordan networks, which have connections from the output layer to an additional input layer (context layer).
- Elman networks, which have an additional input layer called the state layer. It receives connections from the hidden layer.
- Multi-recurrent networks (MRN), which can have connections from both the output and hidden layer into an additional input layer (state layer). Additionally, self-recurrent loops at the state layer can be used.

The dynamic behaviour of MRN poses challenges concerning whether the network is controllable, i.e. can be driven to a certain state with a finite number of steps (Haykin, 1999). Also, the training algorithm of all the recurrent networks must be at least modified (e.g. BPTT = Back-Propagation-Through-Time) or be different from the ones for static networks (e.g. RTRL = Real-Time-Recurrent-Learning).

Support vector machines (SVM) are based on the idea of constructing a hyperplane as a decision surface so that the margin of separation between positive and negative examples is maximized (Boser et al., 1992). This is called structured risk minimization. RBFN and two-layer MLP can be handled as special cases of the SVM. SVMs have good generalization capabilities despite missing domain-specific knowledge (Haykin, 1999).

The use of the SVM for regression has been studied by several authors. Muller et al. (1997) compared them with the RBF networks in time-series prediction, and found the SVM to exceed the performance of RBFN clearly. Lu et al. (2002) made a similar comparison between SVM and RBFN in predicting air pollutant concentrations, and their results agree with those of Muller et al. (1997). Mukherjee et al. (1997) included the chaotic behaviour of time-series in their study, and their results confirm the good performance of the SVM compared with polynomial approximation techniques, RBFN and neural networks.

4.11. Regularization and Bayesian techniques

To understand the basic principle of regularization, some additional concepts are needed. Consider a regression model fitted to a set of points. Firstly, if a model has a high bias (average error), it implies that the model function is on the average different from the actual function. On the other hand, if the model has high variance, it means that the model function is very sensitive to different data sets originating from the actual function. The importance of these concepts can be seen by considering their connection to minimizing the generalization error, which is the main goal in forecast modelling. The generalization error can be decomposed as follows (Haykin, 1999):

$$E_{generalization} = \sqrt{Bias^2 + variance} \qquad (35)$$

Consequently, for any given size of data set, there is some optimal balance between bias and variance, which gives the smallest average generalization error. In other words, one must use some method to control the effective complexity of the model (Bishop, 1995).

One way of controlling the effective complexity of the model (the bias part) is to compare a range of models with different numbers of free parameters (hidden units with neural networks). This is called structural stabilization. The variance part (with neural networks) is usually handled with a technique called early stopping, in which a separate test set is used to detect the point when overfitting begins (Bishop, 1995).

An alternative approach for reducing both bias and variance is to use some additional information about the problem domain (Bishop, 1995). This is called regularization. With neural networks, this is based on adding a penalty term to the error function, thus importing some additional information to the model, e.g. that the actual function modelled is smooth (Foresee and Hagan, 1997). The effective complexity of the model can be controlled by an adjustable parameter which regulates the balance between the training error function and the regularization term. Weight decay is one way to implement regularization with neural networks. It is based on the idea that large weight values correspond to the possibility of overfitting (Bishop, 1995).

Bayes' theorem (Gelman, 2003) gives a formula for calculating the posterior probability for a model, given a prior probability and evidence. Thus, for classifying the measurement vector **x** into one of the *C* classes, minimizing the probability of misclassification, the posterior probability for class C_k can be written as follows (Bishop, 1995):

$$P(C_k \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_k) P(C_k)}{p(\mathbf{x})} \qquad (36)$$

where $p(\mathbf{x}|C_k)$ is the class-conditional probability density function for class C_k, $P(C_k)$ is the (prior) probability of class C_k, and p(x) is the unconditional probability function for **x**.

The prior probability corresponds to our beliefs before making any measurements, and can therefore be used to import domain knowledge into the model. The evidence corresponds to the measurements been made of the phenomenon. Thus, Bayes' theorem provides a practical guide for combining data-driven approaches to information expressed at a symbolic level, such as rules.

A Bayesian approach can be utilized as such Bayesian belief networks (BBN) (e.g. Singh and Valtorta, 1995). The advantage of these compared with many computational methods is that the use of prior knowledge compensates for the need for large amounts of training data for the model.

Bayesian techniques can also be used with neural networks, including the MLP network., This can be done in two ways. Firstly, the network can be regularized by setting the network parameters to some prior distribution (see Liang et al., 2002). Secondly, the Bayesian approach can be used in developing the training algorithms for the MLP. Neal (1996) de-

scribes such learning in detail. Note that learning algorithms of this kind can diminish the amount of data needed for training and thus enable applications where only limited measurements have been made or are possible.

4.12. Measuring the goodness of the estimate

Measuring the goodness of the estimate is an important part of model development, and it can be achieved by visual or by numerical methods. Visual methods make it possible to get an intuitive hold of the model performance, whereas numerical methods provide a more solid ground for comparing and enhancing the models in a scientific way. The visual methods include simple plotting of time-series (predicted vs. observed), histograms of observed-predicted and scatter plots of observed vs. predicted. Examples of these visualizations can be found in the studies included in this thesis (e.g. Kolehmainen et al., 2001) so they are not discussed here. Numerical performance indicators and the procedure for calculating a standard error for them are discussed in detail below.

4.12.1. Numerical performance indicators

The use of different operational performance indicators for evaluating and comparing models has been discussed in Willmott (1982) and Willmott et al. (1985). Their recommendations were followed in our studies and the indicators adapted are explained below. Here we consider prediction models and denote by O_i an observed data point whose prediction by the model is P_i.

Mean Absolute Error (MAE) is the simplest of the numerical goodness measures. It is simply the mean of the absolute errors taken over the set of the estimates. It is calculated according to equation 37:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|P_i - O_i\right| \tag{37}$$

where n is number of data points. The benefits of MAE are that it is less sensitive than squared errors to extreme values and it also reports the error in the original magnitude and scale. Bias is similar to that in MAE but omits the taking of the absolute value (Equation 38). It describes how much the model underestimates or overestimates the situation.

$$Bias = \frac{1}{n}\sum_{i=1}^{n}[P_i - O_i] \tag{38}$$

The Coefficient of Determination (R^2) tells us how much of the observed variability is accounted for by the estimate model. Even though it has defects in certain situations (Comrie, 1997), this measure is usually used in order to maintain compatibility with other studies. The R^2 is calculated according to Equation 39:

$$R^2 = \frac{\sum_{i=1}^{n}\left[P_i - \overline{O}\right]^2}{\sum_{i=1}^{n}\left[O_i - \overline{O}\right]^2} \tag{39}$$

where $\overline{O}$ is the average of observed data. The Root Mean Square Error (RMSE) is one of the most common indicators used with neural networks. The RMSE is calculated according to Equation 40:

$$RMSE = \left(\frac{1}{n}\sum_{i=1}^{n}[P_i - O_i]^2\right)^{\frac{1}{2}} \tag{40}$$

Note that this also preserves the original units. However, extreme error values have more effect than small errors due to the exponentiation. Note additionally that RMSE is usually used in the error function guiding the learning of a MLP network. The RMSE can be divided into systematic and unsystematic components using LS fitting and decomposing the RMSE, using the following two equations:

$$RMSE_S = \left(\frac{1}{n}\sum_{i=1}^{n}(\hat{P}_i - O_i)^2\right)^{\frac{1}{2}} \tag{41}$$

$$RMSE_U = (\frac{1}{n}\sum_{i=1}^{n}(\hat{P}_i - P_i)^2)^{\frac{1}{2}} \quad (42)$$

where $\hat{P}$ is the least squares estimate for P_i yielded by regression of predicted values on observed values.

$RMSE_S$ (systematic) describes the part of the error due to the model (linear bias). Thus, a low value implies a good model. $RMSE_U$ (unsystematic) describes the part of the error which is due to random noise and cannot be captured by the model.

The Proportion of Systematic Error (PSE) gives the ratio of the squared systematic and unsystematic errors. Thus, a lower value implies a better model. It can be used to compare models of different levels of accuracy. It is calculated according to Equation 43:

$$PSE = \frac{RMSE_S^2}{RMSE_U{}^2} \quad (43)$$

Index-of-Agreement (*IA*) is a relative measure limited to the range 0..1. It is thus dimensionless, giving a relative size of the difference. Therefore, it is ideal for making cross-comparisons between models. It is calculated using the following equations:

$$IA = 1 - \frac{\sum_{i=1}^{n}(P_i - O_i)^2}{\sum_{i=1}^{n}(|P_i'| + |O_i'|)^2} \quad (44)$$

where

$$P_i' = P_i - \overline{O} \quad (45)$$

and

$$O_i' = O_i - \overline{O} \quad (46)$$

The basic step in determining the goodness indicators for episodes (e.g. periods of time with extreme values occurring in air quality data) is to calculate all the indicators needed for a limited data set, i.e. take all the values above a certain limit. On the other hand, there are indicators based on calculating the number of times that a given episode has occurred or been predicted. Note that an episode can here be defined as exceeding a given limit value for health effects, for example. Thus, the following indicators can be determined:

- The number of episodes that occurred
- The number of episodes predicted
- The number of correctly predicted episodes
- The number of false alarms
- The number of episodes not found

The natural question arising is, what constitutes an episode? The (computationally) easiest way to answer is to look at every time point separately. A more complicated method would be to construct the episodes by tracking the change above the limit value.

Furthermore, the previous indicators can be unified into a single indicator called the Success-index (*Si*). It is calculated according to Equation 47:

$$Si = \left(\frac{e_c}{e} - \frac{e_p - e_c}{n - e}\right)100 \quad (47)$$

where e is the number of actual episodes, e_p is the number of predicted episodes, e_c is the number of correctly predicted episodes, and n is the number of rows.

In general, IA is the best operational measure, i.e. if it is not good then it is unlikely that the model can be used in practise. Systematic and unsystematic components of the RMSE tell us about the condition of the model compared with the noise level. Then, if the other indicators are good, the R^2 should be high. From the other perspective, a high R^2 value usually implies low RMSE and high *IA*, but does not guarantee it. An example of the use of these indicators is shown in Table 1. It can be seen

that the MLP model yielded the best results in terms of *IA*, RMSE (both systematic and unsystematic) and PSE. However, the SOM model had the lowest bias, and the Periodic+MLP the highest R^2 value. Thus, it could be concluded that the MLP model would most probably provide the best operational performance of these models.

4.12.2. Estimating the standard error

Bootstrapping (Efron and Tibshirani, 1993; Mooney and Duval, 1993) is a way of calculating whether we can trust the indicator(s). This is especially important with methods such as neural networks, where the outcome depends on the random initializations of weights. Bootstrapping is a non-parametric method and does not need any strong assumptions. It is evaluated by resampling the validation set with replacement and calculating the indicator(s) for each set separately. Thus, the model itself is trained only once. A graphical example of the use of bootstrapping in estimating the standard error of the IA parameter is shown in Figure 14.

Using the bootstrapping algorithm, the standard error for the estimated parameter (e.g. index-of-agreement) is now the standard deviation of the separate estimates calculated by resampling with replacements. If the histogram of the bootstrap estimates is approximately normal in shape, normal theory for finding confidence intervals for the unknown parameter can be used.

The use of bootstrapping to estimate the standard error for a neural network model is exemplified in Figure 14. The original outcome of the model validation process gave the value 0.9063 for the IA parameter. The uncertainty of this result was estimated using the bootstrapping algorithm, and the distribution of the re-sampling process is illustrated in Figure 14. The standard error was determined from this distribution and the results could be given as IA = 0.9063 ± 0.0021

Table 1.
Estimators for five models for the validation data (modified from Kolehmainen et al., 2001).

ESTIMATOR	Periodic	Periodic+SOM	SOM	Periodic+MLP	MLP
R^2	0.61	0.77	0.72	1.00	0.96
Bias	3.93	3.93	2.17	4.05	3.28
IA	0.73	0.77	0.85	0.89	0.90
RMSE	15.38	15.21	12.45	11.41	10.72
$RMSE_s$	10.86	9.34	7.16	5.43	4.72
$RMSE_u$	10.88	12.00	10.19	10.03	9.45
PSE	0.50	0.38	0.33	0.23	0.19

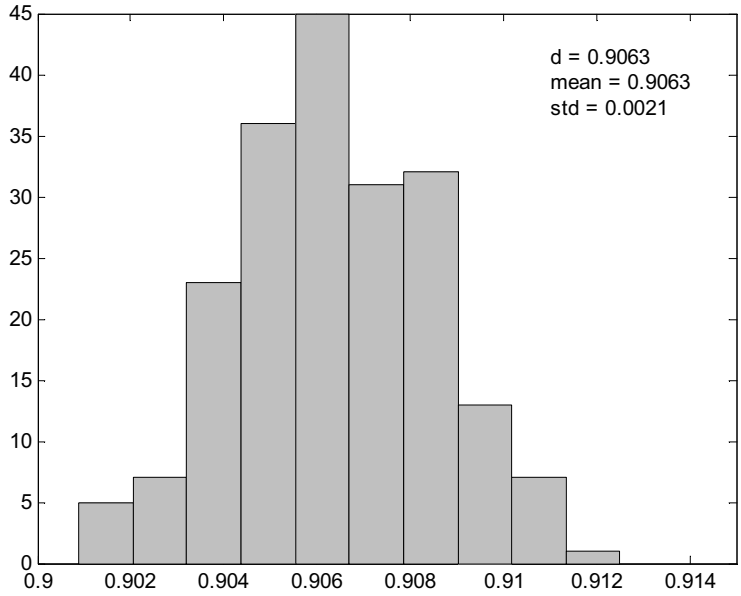

Figure 14. Histogram based on the bootstrapping algorithm. The histogram shows the variation around the estimated index-of-agreement value d = IA = 0.9063. The standard error can now be estimated and IA = 0.9063 ± 0.0021.

5. CASE STUDIES

5.1. Method Selection

The work carried out in this thesis involved approaches which cover the first three steps of the data-mining process (Hand et al., 2001). The goal of the first series of studies was to test methods which might be suitable for visualizing the structures of the measured data in order to reveal the relationships within the process modelled. In practise, this means exploring the dependencies among the variables and related cluster structures. The importance of visualization in real-world decision making environments, especially in multi-objective cases, is acknowledged by, among others, Parmee (2001).

A natural choice for data exploration is the SOM. An important feature of the SOM is the continuity of the lattice, which enables natural visualization of the data. This is in contrast to some other methods such as c-means and hierarchical clustering, in which the relationship between clusters remains unclear to some extent. As a SOM is not strictly a faithful representation (van Hulle, 2000), some regions of the data space are under-sampled and some over-sampled. A comprehensive way to correct this would be to modify the SOM algorithm itself. However, a more practical implementation is to use the weight vectors of the SOM as a starting point for Sammon's mapping algorithm, which is able to clarify the initial cluster structure created. This speeds up the calculation of the Sammon's mapping algorithm, enabling fast and user-oriented working.

The other part of the thesis aimed at covering the modelling phase of the data-mining process. The MLP neural network was chosen as a backbone algorithm because of its wide range of known applications. However, it soon became clear that auxiliary algorithms are needed for handling the challenges posed by environmental data (see 2.3 for details). The main aim in these extensions was to test methods that can simplify the modelling process by splitting it into phases. One approach selected was to model the time-series using linear regression and then to apply the neural networks to the residual of the first step. The second approach was to simplify the function to be modelled by clustering the problem space for local models. This leads naturally to testing the usability of well-known algorithms such as c-means and fuzzy c-means clustering as well as application of fuzzy group membership evaluation in this simplification process.

5.2. Visualization using SOM and Sammon's mapping

The basic aim behind four of the studies of this thesis (Törönen et al., 1999; Kolehmainen et al., 2001b; Valkonen et al., 2002; Kolehmainen et al., 2003) was to test the applicability of SOMs and Sammon's mapping in visualizing the multivariate cluster structure of different environmental and bioinformatics datasets. This idea was also used in the fifth study (Kolehmainen et al., 2000). Sammon's mapping was thus applied in the stage where the SOM algorithm has already achieved a substantial data reduction by replacing the original data vectors with fewer representative prototype vectors. This chain of action is illustrated in Figure 15. A literature survey of related approaches will be given at the end of this Section.

In the first study of this kind (Kolehmainen et al., 2001), the data exploration procedure by SOM and Sammon's mapping was applied to ion mobility spectrometry (IMS) data measured from a pulp mill process. The calibration measurements were carried out in a laboratory to determine the connection between them and field conditions. The results show that the datasets are comparable to a certain extent in the sense of what is known about the overall development of the process, i.e. sulphur compounds first dominate the IMS measurement and then, as concentrations of such compounds decrease, terpene compounds can

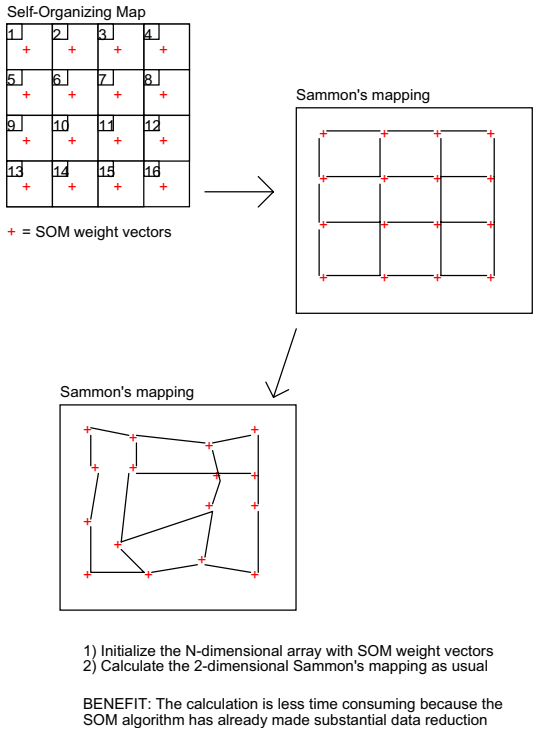

Figure 15. Principle of using a SOM and Sammon's mapping in combination. The learned SOM (first grid) is first used to initialize the Sammon's mapping data structure (intermediate grid). Sammon's mapping algorithm is then applied, which results in a final visualization (last grid). Note how the 2-dimensional result can still resemble the SOM grid in some ways.

be detected. More importantly, the method of applying CI methods to measurement data of this kind was shown to reveal important features of the measurement technique itself, e.g. the separability of different sulphur compounds, and their concentrations, as well as provide a clear and intuitive view of the complex and multivariate problem domain of IMS in general.

The same measurement technique (IMS) and computation methods were also applied to monitor yeast fermentation process (Kolehmainen et al., 2003). CI methods have been applied earlier in process discovery, e.g. in die-casting machines (Cass and DePietro, 1998). However, we extended the domain of application to biochemical processes, which created an extra layer of complexity. The results show that different phases of one fermentation batch and different batches can be distinguished by their distinct IMS profile. Interestingly, the phases detected by this method and those found in standard textbooks differed substantially in the early phases of the process: our method was able to detect two phases instead of the known one. This shows how an innovative application of CI methods can reveal new information in a way that is novel to the whole discipline of bioprocess engineering, for example.

The combination of SOM and Sammon's mapping was also applied to gene expression data (Törönen et al., 1999). In this study, we were able to show how these methods can be used to find and identify the most interesting genes of a certain phenomenon from thousands or potentially tens of thousands of genes. The use of such software in screening potential genes speeds up the research work at the laboratory level, enabling a faster cycle of experiments. The study was one of the most important starting points for gene expression based bioinformatics research, which has become a very active field of research recently. Fuzzy clustering (e.g. Futschik and Kasabov, 2002) and Bayesian neural networks (e.g. Liang et

al., 2002) have also been used recently for this purpose, and these two studies focused especially on handling the noise component of the detected gene expression levels. However, the frontier of gene expression-based bioinformatics is currently moving from data exploration to reverse engineering of gene networks, which introduces ill-defined inverse problems into this field (e.g. Wahde and Hertz, 2000; Chen et al., 2001)

The same method was then applied to epidemiological data of 1650 subjects (Valkonen et al., 2002). Using 25 biochemical and physiological variables alone, we were able to show how the inspection of multivariate data of this kind can lead to identification of important cluster structures consisting of healthy individuals, groups of individuals with severe symptoms, and two intermediate groups (see Figure 16). In this way, it was demonstrated how this method can act as a hypothesis generator by finding new groupings and unidentified groups from multivariate research data. These findings can then act as a starting point for more focused research with traditional statistical tools, for example. Kusiak et al. (2000)

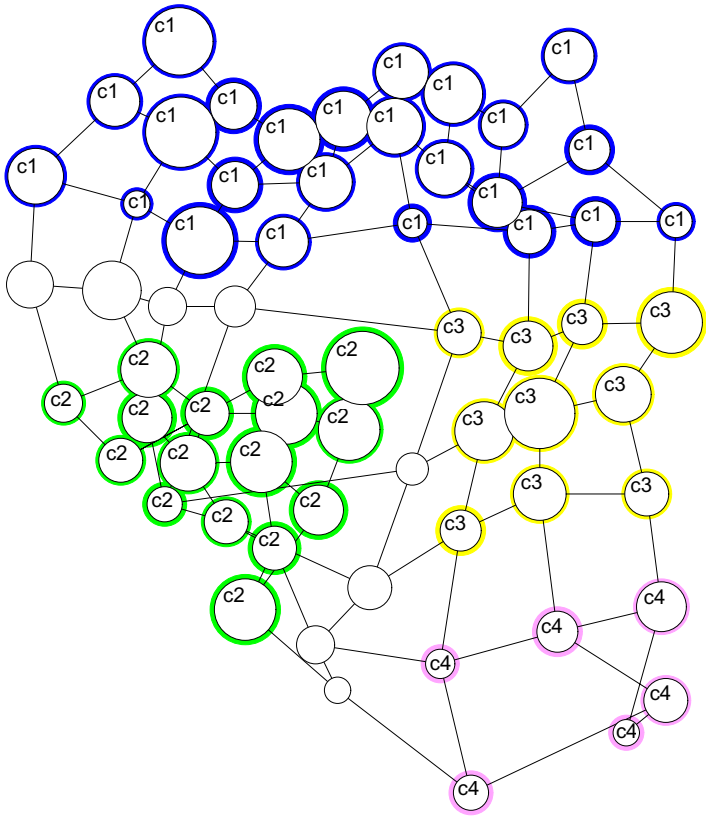

Figure 16. An example clustering revealed by the Sammon's mapping algorithm. Each circle corresponds to one neuron in the SOM map. Relative distances between neurons describe the distance in the original 25-dimensional space of the training variables. The diameter of each neuron corresponds to the number of subjects included in the neuron. The labels correspond to groups identified from the study cohort: C1 comprises healthy study subjects, C4 corresponds to characteristic phenomena in the insulin resistance syndrome, and C2 and C3 are intermediate groups, the status of which is still under study (modified from Valkonen et al., 2002).

have also suggested that research in the future will be data driven. They acknowledged the role of data mining as an important area of CI which offers tools for the analysis of large datasets.

In the fifth study (Kolehmainen et al., 2000), SOM and Sammon's mapping were used to cluster air quality data. The results show that data exploration provides clear and intuitive visualization about the air quality episodes (intervals of bad air quality) for the area inspected. Moreover, it was shown how the episodes can be detected from the rest of the measurement data, identified using standard statistical indicators and assigned a textual description for later use (e.g. in communication).

All the previously mentioned studies introduced new methods to the fields of their application. Consequently, comparative works are not easy to find yet, even though linear methods, such as the factor analysis and PCA, have been applied earlier in some cases, e.g. Edwards et al. (1994) and Lempiäinen et al. (1999) in epidemiology, Paterson et al. (1999) and Nagendra and Khare (2003) in air quality, and Yeung and Ruzzo (2001) in gene expression data analysis. In bioprocess engineering (Kolehmainen et al., 2003) at least one previous study can be found where supervised learning using feed-forward neural networks were used to detect the phases of a bioprocess (Simon et al., 1998). However, such an approach skips the crucial step of data exploration, in which novel patterns and phenomena can be found. The most direct comparison can be found in gene expression data analysis, where a similar study by Tamayo et al. (1999) was published almost at the same time as our study. Their results agree with ours in general, but their implementation was less enhanced as they excluded genes with less activity from the SOM analysis with a preset limit value. Additionally, traditional clustering algorithms (such as the hierarchical clustering) have been used to cluster genes according to their gene expression profile (e.g. Eisen et al., 1998). However, such approaches miss the crucial feature of the SOM, which enables the creation of a continuous visualization of different clusters in respect to each other.

Consequently, it is appropriate to look for the results achieved by combining the SOM and Sammon's mapping in other disciplines, as well as other characteristics studied by other research groups. At a theoretical level, Dzemyda (2001) tested and discussed the benefits of combining the SOM and Sammons' mapping as compared with using them separately. The thesis of Kaski (1997) is mentioned as a source for the idea itself by Dzemyda but he also remarks that Kaski does not supply theoretical or experimental support for his claim. More generally, the original idea of reducing the computational complexity of Sammon's mapping by applying it to only a representative subset of the data can be traced to Chang and Lee (1973). This creates, however, the problem of how to map the rest of the data to the lower dimensional space for visualization, for example.

One solution to this problem has been suggested by Mao and Jain (1995), who introduced a neural network replicating the Sammon's mapping functionality in an unsupervised form. More specifically, the backpropagation algorithm is extended to include the stress term of Sammon's mapping and it does not need any categorical information as in supervised learning. However, this approach was questioned by de Ridder and Duin (1997) because of the convergence characteristics of the unsupervised method. Similarly, Pal et al. (2002) suggest the use of fuzzy clustering and fuzzy rule base to capture the essential part of Sammon's mapping, thus enabling new data to be visualized in respect to the original training data. However, these approaches do not include the use of the SOM in data reduction.

Schemes for structure preserving dimensionality reduction have been suggested by Pal and Eluri (1998). Firstly, they selected a representative sample from the data by simple random sampling without replacement, followed by

Sammon's mapping algorithm. Secondly, the SOM was used to generate representative prototypes for the Sammon's mapping algorithm. Both schemes then included the use of the MLP to create a mapping for new points of data. Comparing these two approaches with each other and with that of Mao and Jain (1995) and the original Sammon's mapping, they concluded that their approaches reduced computation time and exhibit good prediction capability for new data points. The work of Pal and Eluri was later enhanced by König (2000), who suggested the use of a recall mechanism in the Sammon's mapping phase and also the use of an additional neural network as a final phase. König concluded that the enhancements provide reliability, more general applicability and better performance for various data sets.

It can thus be concluded that the idea of combining the SOM and Sammon's mapping is not new and involves important topics not accounted for in this thesis. However, the combination in data visualization has not been applied to this extent to real world problems. Moreover, the results achieved in this thesis had novelty in the respective disciplines and yielded significant results for those areas, showing the usefulness of the approach in real applications. Additionally, software was created, which implements the ideas in a user-friendly way, as explained in detail in Chapter 5.4. Consequently, it can be assumed that the method tested could also be applied in several other areas in a straightforward fashion.

5.3. Regression of time-series data

The aim of the rest of the studies (three altogether) was to test the applicability of different CI methods in the modelling phase of data mining. The central modelling tool in all the cases was the MLP. Additionally, a comparison was also made with SOM, even though it is basically an unsupervised method.

Firstly, the use of traditional methods combined with CI methods was investigated in Kolehmainen et al. (2001). In that study we tested an idea found in textbooks (e.g. Haykin, 1999): that of capturing and subtracting the cyclic dependencies from time-series data before applying methods such as neural networks. Additionally, we tested the performance of the SOM in this regression problem. The simulation studies showed, contrary to expectations that direct application of the MLP performed better that the corresponding hybrid method, where the cyclic structures were first suppressed with linear regression. However, it was confirmed that a supervised method such as the MLP is indeed superior to unsupervised methods as well as to linear regression alone. Additionally, the use of linear regression for inspecting the seasonal components in a time-series was clearly demonstrated. More importantly, it was also demonstrated that even though the forecasting accuracy was good in general, the models were not able not account for episodes of the air quality phenomenon studied. This is due to the inability of empirical models to capture extreme values, which are usually under-represented in training data.

Comparison of the prediction accuracy of the models used in our study with those of others can be performed to a certain extent. The limiting factor here is the fact that several authors apply only one or two methods for calculating the success of the forecast model. We have used several statistical parameters (Willmott, 1982; Willmott et al., 1985; see 4.13 for details), as well as descriptive graphics to describe and compare the goodness of the models. When comparison has been possible, our results have been on the same level as those of others, i.e. superior results have not been achieved yet in our studies. Such a comparison was most directly possible in two publications (Kukkonen et al., 2003 ; Schlink et al., 2003) which are the result of the APPETISE EU project (Appetise, 2004), in which we also participated. The publications reported rigorous inter-comparison of different forecasting models for NO_2, PM_{10} (particles) and ozone, including our model.

The controversial issue of the relative performance of direct applications of the MLP versus a two phase model where the MLP is applied to the residual of the linear regression using sine and cosine components needs further clarification. Unfortunately, there appears to be no study that would permit direct comparison, so we must content ourselves here with the opinions of the authors of general textbooks such as Masters (1995), who discusses topics relevant to this question. Firstly, he notes that the use of linear regression with sine and cosine functions in time-series regression is not recommended, because of the large number of terms required for exact coverage. On the other hand, he acknowledges the usability of the two phase approach applied in our study, not naming the exact implementation, however. Finally, the use of seasonal differencing (e.g. subtracting the value exactly one year before from the current value) is recommended as a more feasible solution to this kind of problems. Thus, it can be concluded that our study had weaknesses that could have been removed with repeated simulations, and its validity remains to be confirmed with new and more comprehensive studies.

The last two studies (Kolehmainen et al., 2000 ; Niska et al., 2003b) focused on investigating the possibilities of dividing the search space into regions, which could then be modelled with simpler models. The basic idea was to use different clustering methods: the SOM, c-means clustering and fuzzy c-means clustering. Note that this was done by first reducing the data with the SOM in order to make the other clustering algorithms work in a feasible time. Fuzzy group membership evaluation was used to describe the non-crisp features of the cluster borders. The results show that c-means clustering had some advantages in forecasting episode concentrations. The model based on fuzzy c-means clustering achieved best accuracy concerning overall performance. The results in general, however, showed that only marginal benefit can be gained by applying this multi-model structure to air quality data. It was thus concluded that the application of one method alone cannot compensate for other possible weaknesses of the whole data processing chain.

The problem domain of these two studies has also been investigated by others. Several authors have shown the usefulness of voting classification algorithms (Breiman, 1996; Freund and Schapire, 1996; Quinlan, 1996). Bauer and Kohavi (1999) compared these algorithms (bagging and boosting), where several submodels compete with each other. Kim et al. (2002) proposed a two-level evolutionary environment called meta-evolutionary ensembles. Besides the ensembles (groups of classifiers) competing with each other, the individual classifiers compete and are rewarded especially for getting difficult data items classified correctly. Guerra-Salcedo and Whitley (1999) broaden this discussion into feature selection and they claim that this optimization of the classifiers leads to better performance. One approach to optimizing the model has also been studied by our team but it is not included in this thesis (Niska et al., 2003).

A different approach to this model simplification, called variable grouping, has been systematically studied by Tucker et al. (2001). The basic idea is to divide the input space into groups, where the dependency between variables within the same group is high, but the dependencies between groups are as low as possible. Note that this inspection is ranged to lag values of the MTS. As a result of that study, two approaches have been suggested, that of hill climb strategy and the grouping genetic algorithm by Falkenauer (1998). A separate model for each variable group is created and the results are unified at the later stage of the model application.

5.4. Visual Data Software

To enable data exploration of the studies in this thesis in an efficient and user-friendly way, software for this purpose was developed. The

prototype of the software (called DANA) was developed in our research group during the years 1997-1999 as a result of the author's initial programming work and then Teri Hiltunen as the main software developer and the author as a designer and supervisor of the work. In 1999 Visipoint Ltd. (http://www.visipoint.fi) was founded and the final software was created there under the name Visual Data. Two limited versions of the software were also developed, Visual Gene (called GenePoint initially) and Visual Nose. As founders of the company, the author and Teri Hiltunen were also mainly responsible for the software engineering efforts on the commercial side. The Visual Data software was then created in its current form using Visual Basic programming in the MS Windows environment.

The software was based on the Neural Data Analysis (NDA) package (http://erin.mit.jyu.fi) utilizing its central functionality which enables the use of the SOM and Sammon's mapping algorithm. The NDA package is described in detail in the thesis by Häkkinen (2001).

Visual Data software has two very strong properties. Firstly, it is possible to use large data sets (at least millions of data lines) for interactive analysis. This a direct consequence of the TS-SOM approach used in the NDA library. Secondly, the software is extremely easy to use even by a person who is not familiar with the concepts of neural networks. The typical time for learning the basic operations under guidance is 4-8 hours. This is a consequence of successful user-interface design, described below in more detail.

The starting point for data analysis is to select the variables used for training the SOM. This has been streamlined in the form of a dialog shown in Figure 17. The approach enables the user to quickly test different SOM models trained with different sets of variables.

Thereafter, the data needs to be pre-processed using suitable transformations or a combination of them, and a separate dialog box was designed, enabling the user to select the transformation for each variable or any combina-

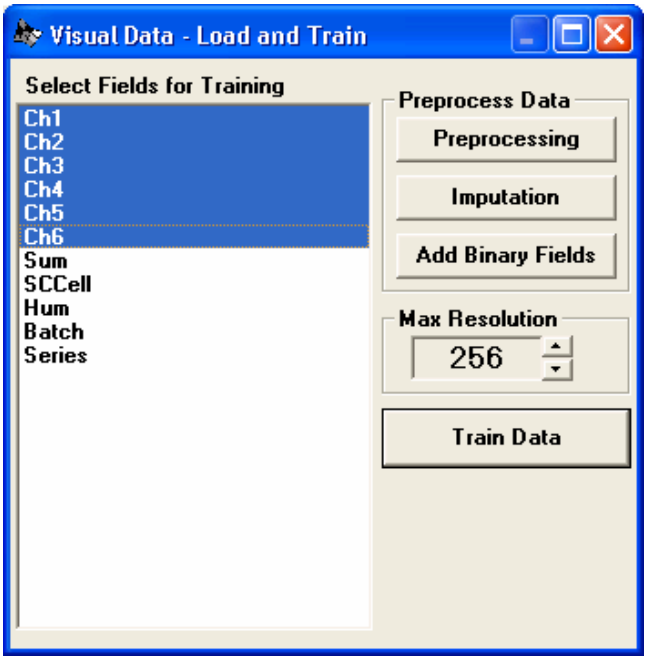

Figure 17. The dialog for selecting the training variables for the SOM in Visual Data software. The six channels of the MGD-1 artificial nose were selected in this case.

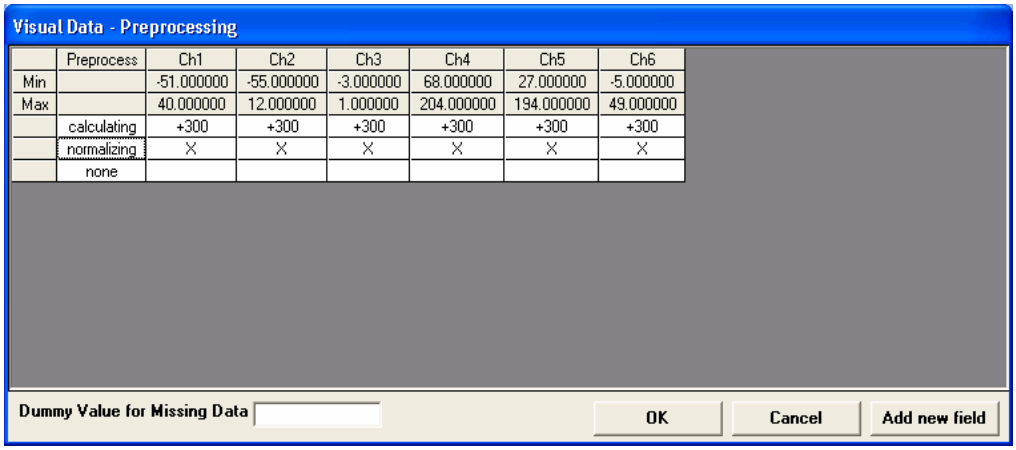

Figure 18. Transformations for the training variables can be selected using a dedicated dialog box in Visual Data software. In this case, a constant of +300 is first added to all the channel values. The learning vector is then normalized by its length.

tion of them in a clear and concise way. This is illustrated in Figure 18.

After pre-processing the data, the SOM is created by just pressing the "Train Data" button. Only the level and thus the number of neurones for the network need to be defined. This is a result of the properties of the TS-SOM (Koikkalainen, 1994), which enables default parameter values to be used in most cases. For advanced users, however, the parameters can also be set using a separate dialog box. The result of the SOM calculation is presented in the main window of Visual Data, as illustrated in Figure 19. The main window contains the controls in the left panel for setting the properties of the SOM, i.e. selecting the variable for grey tone and z-axis, resolution of the SOM, dimension of the graphics and zooming factor. The bottom panel can be set to present more detailed data about the selected neuron or cluster. The bottom left panel allows the user to select the variables for this detailed presentation and its format.

Using a menu bar selection of the main window, an auxiliary window for Sammon's mapping (see Figure 20) can be created. The starting point of this computation is the weight vectors of the SOM. This makes the computation time feasible even for very large datasets. Moreover, it permits a direct connection between both windows, allowing the user to inspect and select (cluster) neurons using either window. Thus, the software supports the complementary use of these two modelling algorithms, as also suggested by, for example, Dzemuda (2001), in an extremely user friendly way.

5.5. Outlook

Although the results of using SOM and Sammon's mapping in data exploration have been promising with static data (e.g. gene expression and epidemiologic datasets), time-series properties have not been taken into account in an adequate way. This can be done by adapting methods from signal processing, which is another field very closely related to time-series processing (Dorffner, 1996). More specifically, methods for integrating continuous, on-line data with off-line laboratory measurements for data exploration purposes should be developed. The signal processing methods could also be used to compensate for lagged

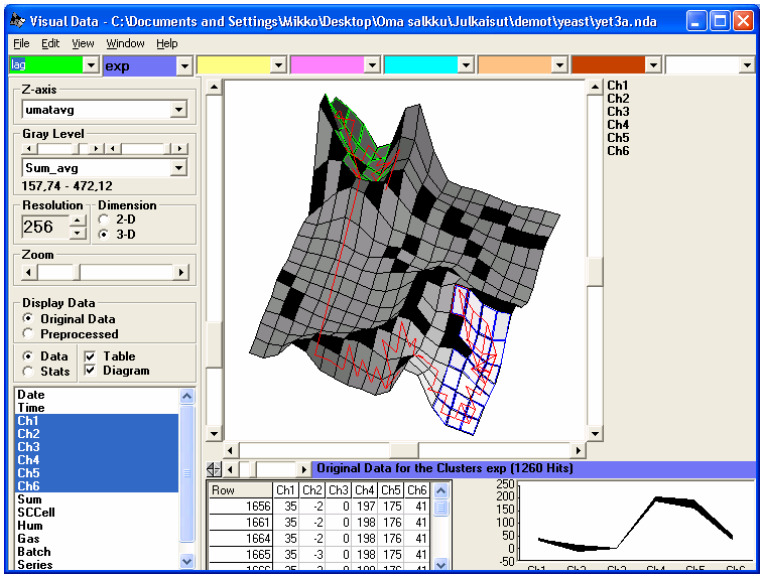

Figure 19. The main window of the Visual Data software. The graphics area presents the SOM in three dimensional format, where the z-axis corresponds to the U-matrix value in this example. Two phases of yeast fermentation process measured using the MGD-1 artificial nose have been clustered and marked according to a trajectory which shows the time development of the measurement data. The left panel of the window contains the control buttons for setting the graphics properties. The bottom panel allows the user to inspect any neuron or cluster of the SOM in more detail, in either numeric and/or diagram format.

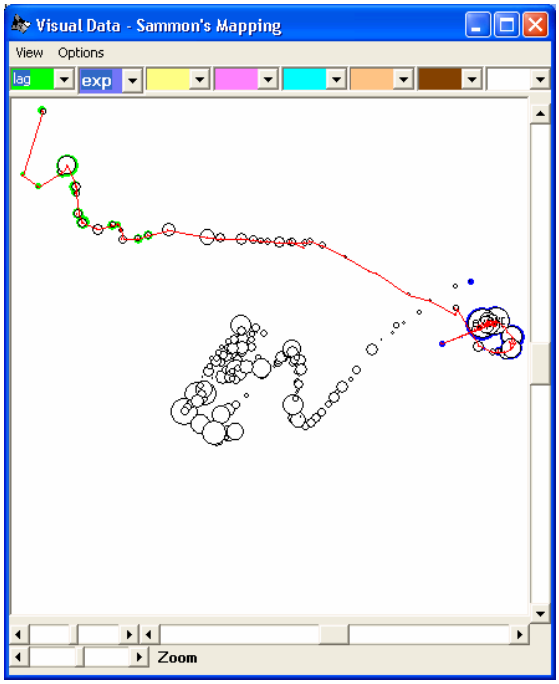

Figure 20. Sammon's mapping in an auxiliary window of the Visual Data software. Each neuron of the SOM is shown as a circle. The radius of the neurons corresponds to the number of original data vectors in each neuron. This enables rough density estimation of the original p-dimensional space in respect to the phenomenon inspected. The trajectory shows the time development from the lag phase of the yeast fermentation to the next phase, exponential growth.

values in multivariate time-series of process data. However, even bigger challenges can be found if the spatial dimensions are added in the form of location information. This would require the use of different kinds of statistics as a basis of the algorithms.

The elements used in time-series prediction for air quality data need to be tested more comprehensively. This can be done by creating a software environment where all the aspects inspected are available for testing together. At least the elements described below should be taken into account.

Optimization of the inputs and (neural network) model parameters should be tested. Given the limitations caused by large search spaces, it should be divided into subtasks both by variable grouping (Tucker et al., 2001) and by clustering the problem space into local domains. The optimization should focus more on the episodes (i.e. periods of bad air quality), which can be done by developing the fitness function. This could also be boosted by duplicating the data representing the episodes in a systematic way. Moreover, our latest results (not published) show that multi-objective optimization (e.g. prediction accuracy as the first objective and minimizing the number of variables as the other) can lead to better performing models.

The model itself could also be developed further. Firstly, the initialization of the weights of the MLP should be investigated. Possible solutions include clustering the problem space and the use of the GA. Secondly, regularization techniques should be tested in order to include additional knowledge in the model. The most promising solutions for this are based on Bayesian techniques. Finally, replacing the MLP with recurrent neural networks or with the SVM could enhance the capabilities of the model to account for the non-stationary and cyclic nature of the environmental data.

As already discussed, modelling of environmental and related data easily leads to ill-defined inverse problems. Their solution usually requires that domain-specific knowledge is incorporated into the model, which is usually based on Bayes theorem. An example of such a development can be found in bioinformatics, where data exploration studies of gene expression data have inspired active research on reverse engineering the gene regulatory networks (Wahde and Hetrtz, 2000; Chen et al., 2001). Interestingly, recurrent neural networks, GA and simulated annealing have been proposed as suitable methods in these studies. Thus, the solutions suggested in bioinformatics could also be tested in environmental informatics.

6. SUMMARY AND CONCLUSIONS

The aim of this thesis was to evaluate self-organizing maps together with other related methods of computational intelligence for analysing and modelling environmental and bioinformatics problems, using a number of selected case studies. The thesis comprises seven studies which have been published as six peer-reviewed articles and one extended abstract in conference proceedings. All of the studies were based on using more than one traditional or computationally intelligent method, with the SOM as the central modelling tool. The main contribution of this thesis was testing the applicability of data exploration methods using the SOM and Sammon's mapping in real world problems and showing how significant results in the corresponding discipline could be yielded. Overall, it can be concluded that if the method tested is implemented as user-friendly software (Visual Data), excellent results can be achieved, as explained in more detail below.

Firstly, environmental informatics and bioinformatics were defined and their connection with each other discussed. The challenges posed by environmental informatics and bioinformatics data were then identified. It was found that the quantity of data is usually adequate but the quality can be a problem, especially in the form of missing values. The multivariate nature of time-series data requires that dependencies among variables be accounted for in order to prevent similar variables dominating the learning algorithms. On the other hand, the multivariate aspect can also be used in compensating for missing values or unreliable variables. The complex behaviour of natural processes induces additional characteristics, including non-linear and potentially chaotic phenomena, ill-defined inverse problems and multitudes of lagged dependencies among the variables. It is concluded that the above issues often lead to large search spaces in modelling and optimizing the complicated functions corresponding to these processes.

CI has been defined in several ways in different sources but the common elements of these definitions include adaptivity to changing conditions and the ability to learn from experience. It is also often required that the behaviour of the system can be described at least partly as intelligent in respect to human behaviour. On the basis of this, hybrid CI can then be defined as any effective combination of intelligent techniques that performs better or in a competitive way to simple standard intelligent techniques. As a practical implementation for CI, the concept of data-mining was introduced and its main steps relevant to this work were identified as exploratory data analysis, descriptive modelling and regression modelling. The special characteristics of time-series data needing special attention in predictive modelling were explained. This discussion was then concluded by focusing attention on the central topic of modelling, that of generalization.

The methods needed in intelligent data processing were then described in detail. Firstly, the data processing chain in knowledge discovery was described. The first steps (pre-processing, transformation and dimensionality reduction) were briefly described. Linear regression was also briefly introduced, followed by a more detailed introduction to the central methods of data mining, i.e. SOM, Sammon's mapping, c-means and fuzzy c-means clustering, U-matrix, fuzzy logic, and neurocomputing, including the MLP. Regularization and Bayesian techniques were explained as a means of enhancing the previously mentioned methods, and finally the measuring of the goodness of the estimates was described.

The studies comprising this thesis were then evaluated and discussed. The large potential of combining SOM and Sammon's mapping were demonstrated in several applications. The results show that such hybrid use can (i) reveal important features of the measurement

technique itself, such as separability of compounds and their concentrations, (ii) reveal new information in a way that is novel to the whole discipline, (iii) speed up the research work at the laboratory level, and so enabling a faster cycle of experiments, (iv) act as a hypothesis generator for more focused research with traditional statistical tools, and (v) supply clear and intuitive visualization about environmental phenomena, such as air quality episodes, for the area inspected.

In time-series processing the use of traditional methods combined with CI methods was investigated. Contrary to expectations, direct application of the MLP performed better in our study than the corresponding hybrid method, where the cyclic structures were first suppressed with linear regression. It was demonstrated, however, that the hybrid use of the above methods leads to better understanding of the process modelled (e.g. by visualization of the seasonal components of the time-series), which is an important goal of time-series modelling. The two other studies focused on investigating the possibilities of dividing the search space into regions, which could then be modelled with simpler models. The results in general, however, show that only marginal benefit can be gained by applying this multimodel structure to air quality data.

As a recommendation for future work it was suggested that the data exploration solution should be enhanced with methods from signal processing to enable the handling of measurements with different time scales and lagged multivariate time-series. For the regression modelling of time-series data, it was suggested that an integrated environment be created for testing all the methods discussed in a coherent way and simultaneously. The list of most potential issues includes optimization of the inputs and model parameters by variable grouping and clustering, boosting by duplicating the data representing the episodes, applying recurrent neural networks or SVM instead of MLP, and applying regularization techniques to import prior knowledge into the model. Accounting for non-linear chaotic behaviour and problems of ill-defined inverse problems also needs further investigation

REFERENCES

Adriaans, P. and Zantinge, D., Data mining, Addison-Wesley, Harlow, England, 1996.

Appetise home pages, http://www.uea.ac.uk/env/appetise/, 21.1.2004.

Battiti, R., First and second order methods for learning: between steepest descent and Newton's method, Neural Computation, 1992, 4, 141-166.

Bauer, E. and Kohavi R., An empirical comparison of voting classification algorithms: Bagging, Boosting and variants, Machine Learning, 1999, 36, 105-139.

Becker, S. and Le Cun, Y., Improving the convergence of back-propagation learning with second order methods. In Proceedings of the 1988 connectionists models summer school, Carnagie-Mellon University, 1989, Morgan Kaufmann, Los Altos, CA.

Beliakov, G. and Abraham, A., Global optimisation of neural networks using deterministic hybrid approach, Hybrid information systems, Proceedings of the First International Workshop on Hybrid Intelligent Systems, HIS 2001, 2002, Springer Verlag, Germany, 79-92.

Berthouex, P.M. and Brown, L.C., Statistics for environmental engineers, CRC Press Company, N.Y, 2nd edition, 2002.

Bezdek, J.C., Pattern recognition with fuzzy objective function algorithms, 1981, Plenum Press, New York.

Bezdek, J.C., What is computational Intelligence, Computational Intelligence: imitating life, eds., Zurada, J., Marks, J.M., and Robonson, C., Piscataway, IEEE Press, 1994, 1-12.

Bishop, C., Neural Networks for Pattern Recognition, Clarendon Press, Oxford, 1995.

Bonissone, P.P., Chen, Y-U., Goebel, K., and Khedkar P.S., Hybrid soft computing systems: Industrial and commercial applications, Proceedings of the IEEE, 1999, 87, 1641-67.

Boser, B., Guyon, I., and Vapnik, V., A training algorithm for optimal margin classifiers, 1992, Fifth Annual Workshop on Computational Learning Theory, 144-152, San Mateo, CA, Morgan Kauffmann.

Box, G. and Jenkins, F., Time Series Analysis: Forecasting and Control, 1976, 2nd edition, Oakland, CA: Holden-Day.

Breiman, L., Stone, C., Olshen, R., and Friedman, J., Classification and Regression Trees, 1984, Brooks/Cole.

Breiman, L., Bagging predictors, Machine learning, 1996, 24, 123-140.

Cass, R. and DePietro, J., Computational intelligence methods for process discovery, Engineering Applications of Artificial Intelligence, 1998, 11, 675-681.

Chang, C. and Lee, R., A heuristic relaxation method for nonlinear mapping in cluster analysis, IEEE Transactions on Systems, Man, and Cybernetics, 1973, 3, 197-200.

Charalambous, C., Conjugate gradient algorithm for efficient training of artificial neural networks, IEE Proceedings-G Circuits Devices and Systems, 1992, 139, 301-310.

Chen T., Filkov V., and Skiena S.S., Identifying gene regulatory networks from experimental data, Parallel Computing, 2001, 27, 141-162.

Comrie A., Comparing neural networks and regression models for ozone forecasting, Journal of Air and Waste Management Association, 1997, 47, 653-663.

Cover, T., Geometrical and statistical properties of systems of linear inequalities

with applications on pattern recognition, IEEE Transactions on electronic computers, 1965, EC-14, 326-334.

Davies, D. and Bouldin, D., A cluster separation measure, IEEE Transaction on Pattern Analysis and Machine Intelligence, 1979, PAMI-1, No 2, 159-179.

Deb, K., Multi-objective Optimization Using Evolutionary Algorithms, Wiley, 2001.

Dempster, A., Laird, N., and Rubin, D., Maximum likelihood from incomplete data via the EM algorithm, Journal of Royal Statistical Societ, 1977, Ser. B, 39, 1-38.

Diday, E. and Simon, J.C., Clustering Analysis, In Fu, K.S. (ed.), Digital pattern recognition, Springer-Verlag, 1980.

Dote, Y. and Ovaska, S.J., Industrial applications of soft computing: A review, Proceedings of the IEEE, 2001, 89, 1243-65.

Dorffner, G., Neural Networks for time series processing, Neural Network World, 1996, 4, 447-468.

Draper, N. and Smith, H., Applied regression analysis, 1981, New York, Wiley.

Dzemyda, G., Visualization of a set of parameters characterized by their correlation matrix, Computational Statistics & Data analysis, 2001, 36, 15-30.

Edwards, K., Austin, M., Newman, B., Mayer, E., Krauss, R., and Selby J., Multivariate analysis of the insulin resistance syndrome in women, *Arterioscler Thromb,* 1994, 14, 1940-45.

Efron, B. and Tibshirani, R., An Introduction to the Bootstrap, 1993, Chapman and Hall, New York.

Eiceman, G.A. and Karpas, Z., Ion Mobility Spectrometry, CRC Press, Inc., 1994.

Eisen, M., Spellman, P., Brown, P., and Botstein, D., Cluster analysis and display of genome-wide expression patterns, Proceedings of the National Academy of Sciences of the United States of America, 1999, 96, 10943-10943.

Falkenauer, E., Genetic Algorithms and Grouping Problems, Wiley, 1998.

Fayyad, U., Data mining and knowledge discovery: Making sense out of data, IEEE Expert Intelligent Systems & their Applications, 1996, 11, 20-25.

Feigenbaum, E.A., Some challenges and grand challenges for computational intelligence, Journal of the ACM, 2003, 50, 32-40.

Fisher, R., The use of multiple measurements in taxonomic problems, Ann. Eugen., 1936, 7, 178-188.

Fogel, D., Review of "Computational intelligence: imitating life", IEEE Transactions on Neural Networks, 1995, 6, 1562-1565.

Fool & Fox Users Manual, Fuzzy system development tools, Version 1.6, 1999, (ftp://ftp.informatik.uni-oldenburg.de/pub/fool/), 18.9.2003.

Foresee, F.D. and Hagan M.T, Gauss-Newton approximation to Bayesian regularization, Proceedings of the 1997 International Joint Conference on Neural Networks, 1997, 1930-1935.

Freund, Y and Schapire, R., Experiments with a new boosting algorithm, In Machine learning: Proceedings of the Thirteenth National Conference, 1996, 148-156, Morgan Kaufmann.

Futschik, E.M., and Kasabov N.K., Fuzzy clustering of gene expression data, Proceedings of the 2002 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'02), 2002, 414-419.

Gan, Q., Exploring recurrent learning for neurofuzzy networks using regularization theory, Proceedings of the 2002 International Joint Conference on Neural Networks (IJCNN'02), 2002, 1763-1766.

Gardner, M.W. and Dorling, S.R., Artificial neural networks (the multi-layer perceptron) - a review of applications in the atmospheric sciences, Atmospheric Environment, 1998, 32, 2627-2636.

Geer, C., Whyte, L., Lawrence, J., Masson, L., and Brousseau, R., Genomics technologies for environmental science, Environmental Science & Technology, 2001, A, 364-370.

Gelman, A., Bayesian Data Analysis, 2003, 2nd edition, Boca Raton, FL Chapman & Hall/ CRC.

Ghahramani, Z. and Jordan, M.I., Supervised learning from incomplete data via an EM approach, In Dowan, J.D., Tesauro, G.T. and Alspector, J. (eds.), Advances in Neural Information Processing Systems, 1994, 6, 120-127, San Mateo, CA, Morgan-Kaufmann.

Goldberg, D.E., Genetic Algorithms in Search, Optimization, and Machine Learning, Reprinted with corrections, Addison-Wesley Publishing Company, Inc., 1989.

Green, D.G. and Klomp N.I., Environmental informatics - a new paradigm for coping with complexity in nature. Complexity International, 1998, 6, (http://journal-ci.csse.monash.edu.au/ci/vol06/green/green.html)

Guerra-Salcedo, C. and Whitley, D., Genetic approach to feature selection for ensemble creation, In GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference, 1999, Morgen Kaufmann, 236-243.

Gunther, O., Environmental Information Systems, Springer, 1998.

Hadamard, J., Sur les problèmes aux dérivées partielles et leur signification physique, Princeton University Bulletin, 1902, 49-52.

Hagan, M.T. and Menhaj, M., Training feedforward networks with the Marquardt algorithm, IEEE Transactions on Neural Networks, 1994, 5, 989-993.

Hand, D., Mannila, H., and Smyth, P., Principles of Data Mining, Cambridge (MA), MIT Press, 2001.

Haykin, S., Neural Networks: A Comprehensive Foundation, 2nd edition, Prentice Hall, New Jersey, 1999.

Haykin, S. and Principe, J.C., Making sense of complex world, IEEE Signal Processing Magazine, 1998, May, 66-81.

Hecht-Nielsen, R., Neurocomputing, Reprinted with corrections, Addison-Wesley Publishing Company, Inc. 1991.

Häkkinen, E., Design, implementation and evaluation of Neural Data Analysis environment, Doctoral dissertation, 2001, University of Jyväskylä.

Jang, J.-S., Sun, C.-T., and Mizutani, E., Neuro-fuzzy and Soft Computing, 1997, Prentice Hall.

Johnson, S., Hierarchical clustering schemes, Psychometrika, 1967, 2, 241-254.

Junninen, H., Niska H., Ruuskanen A., Patama T., Tuppurainen K., Kolehmainen M., and Ruuskanen J., The performance of different imputation methods for air quality data with missing values, Submitted to Atmospheric Environment, 2002, currently under revisions.

Kaski, S., Data exploration using self-organizing maps, Thesis for the degree of Doctor of Technology, 1997, Helsinki University of Technology, Neural Networks Research Centre.

Kim, Y.S., Street, W.N., and Menczer, F., Meta-Evolutionary ensembles, Proceedings of the 2002 International Joint Conference on Neural Networks (IJCNN'02), 2002, 2791-2796.

King, R.L., Artificial neural networks and computational intelligence, IEEE Computer Applications in Power, 1998, 11, 14-25.

Kirsch, A., An Introduction to the Mathematical Theory of Inverse Problems, Springer-Verlag, 1996.

Kohonen, T., Self-Organizing Maps, 2nd edition, Springer-Verlag Berlin Heidelberg, Germany, 1997.

Koikkalainen, P., Progress with the tree-structured self-organizing map, ECAI'94, Proceedings of the 11th European Conference on Artificial Intelligence, 1994, Cohn, A. (Ed.), Wiley&Sons, 211-215.

Kolehmainen, M., Martikainen, H., Hiltunen, T., and Ruuskanen, J., Forecasting air quality parameters using hybrid neural network modelling, Environmental Monitoring and Assessment, 2000, 65, 277-286.

Kolehmainen, M., Martikainen, H., and Ruuskanen, J., Neural networks and periodic components used in air quality forecasting, Atmospheric Environment, 2001, 35, 815-825.

Kolehmainen, M., Rissanen, E., Raatikainen, O., and Ruuskanen, J., Monitoring odorous sulfur emissions using Self-organizing maps for handling ion mobility spectrometry data, Journal of Air and Waste Management, 2001b, 51, 966-971.

Kolehmainen, M., Junninen, H., Niska, H., Patama, T., Ruuskanen, A., Tuppurainen, K., and Ruuskanen, J., Missing data toolbox for air quality datasets, Environmental Communication in the Information Society, 16th International Conference "Informatics for Environmental Protection", 445-451, September 25-27, 2002, Vienna University of Technology.

Kolehmainen M., Rönkkö P., and Raatikainen O., Monitoring of yeast fermentation by ion Mobility spectrometry measurement and data visualisation with Self-organizing maps, Analytica Chimica Acta, 2003, 484, 93-100.

Kukkonen, J., Partanen, L., Karppinen, A., Ruuskanen, J., Junninen, H., Kolehmainen, M., Niska, H., Dorling, S., Chatterton, T., Foxall, R., and Cawley, G., Extensive evaluation of neural network models for the prediction of NO_2 and PM_{10} concentrations, compared with a deterministic modelling system and measurements in Central Helsinki, Atmospheric Environment, 2003, 37, 4539-4550.

Kusiak, A., Kern, J.A., Kernstine, K.H., and Tseng, B.T.L, Autonomous decision-making: a data mining approach, IEEE Transactions on Information Technology in Biomedicine, 2000, 4, 274-284.

König, A., Interactive visualization and analysis of hierarchical neural projections for data mining, IEEE Transactions on Neural Networks, 2000, 11, 615-624.

Lempiäinen, P., Mykkänen, L., Pyörälä, K., Laakso, M., and Kuusisto, J., Insulin resistance syndrome predicts coronary heart disease events in elderly nondiabetic men, Circulation, 1999, 100, 123-28.

Lerner, B., Guterman, H., Aladjem, M., and Dinstein, I., On the initialization of Sammon's nonlinear mapping, Pattern Analysis & Applications, 2000, 3, 61-68.

Liang, Y., George, E.O., and Kelemen, A., Bayesian neural network for microarray data, Proceedings of the 2002 Congress on Evolutionary Computation (CEC'02), 2002, 193-197.

Little, R.J.A and Rubin, D.B., Statistical Analysis with Missing Data, John Wiley & Sons, New York, 1987.

Lu, W., Wang, W., Leung, A.Y.T, Lo, S-M., Yuen, R.K.K, Xu, Z., and Fan H., Air pollutant parameter forecasting using Support Vector Machines, Proceedings of the 2002 International Joint Conference on Neural Networks (IJCNN'02), 2002, 630-635.

MacQueen, J., Some methods for classification and analysis of multivariate observations. Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, 1967, Volume I, Statistics, University of California Press.

Mao, J. and Jain, A., Artificial neural networks for feature extraction and multivariate data projection, IEEE Transactions of Neural Networks, 1995, 6, 296-317.

Masters, T., Neural, Novel & Hybrid Algorithms for Time Series Prediction, 1995, John Wiley & Sons, Inc.

Moller, M.F., A scaled conjugate gradient algorithm for fast supervised learning, Neural Networks, 1993, 6, 525-533.

Mooney, C. and Duval, R., Bootstrapping: A Nonparametric Approach to Statistical Interference, 1993, SAGE Publications Inc.

Mukherjee, S., Osuna, E., and Girosi., Nonlinear prediction of chaotic time series using a support vector machine, Proc. NNSP, 1997, 511-520.

Muller, K.R., Smola, A.J., Rätsch, G., Schölkopf, B., Kohlmorgen, J., and Vapnik V., Predicting time series with support vector machines, Proc. ICANN, 1997, 999-1004.

Munoz, A. and Muruzabal, J., Self-organizing maps for outlier detection, Neurocomputing, 1998, 18, 33-60.

Nagendra, S. and Khare, M., Principal components analysis of urban traffic characteristics and meteorological data, Transportation Research Part D - Transport and Environment, 2003, 8, 285-297.

NDA user's guide, Neural Data Analysis Environment User's Guide, http://erin.mit.jyu.fi/projects/nda/usrman/usrman.html, 14.8.2003.

Neal, R.M., Bayesian learning for neural networks, Springer-Verlag New York, Inc, 1996.

Niemi, A., Johdatus sumeisiin joukkoihin ja sumeaan logiikkaan (in Finnish), 1996, Hakapaino Oy, Helsinki.

Nilges, M. and Linge, J.P., A definition of bioinformatics, Unité de Bio–Informatique Structurale, Institut Pasteur, 25–28 rue du Docteur Roux, F–75015 Paris, France, http://www.pasteur.fr/recherche/unites/Binfs/definition/bioinformatics_definition.html, 6.8.2003

Niska H., Hiltunen T., Karppinen A., and Kolehmainen M., Evolving the neural network model for forecasting air pollution time-series, IFAC International Conference on Intelligent Control Systems and Signal Processing (ICONS'03), Faro, Portugal, 310-315, April 08-11, 2003.

Niska H., Hiltunen T., Kolehmainen M., and Ruuskanen J., Hybrid models for forecasting air pollution episodes, International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA'03), University Technical Institute of Roanne, 80-84, France April 23-25, 2003b, Springer-Verlag/Wien.

Pal, N. and Eluri, V., Two efficient connectionist schemes for structure preserving dimensionality reduction, IEEE Transactions on Neural Networks, 1998, 9, 1142-1154.

Pal, N., Eluri, V., and Mandal, G., Fuzzy logic approaches to structure preserving dimensionality reduction, IEEE Transactions of Fuzzy Systems, 2002, 10, 277-286.

Pal, N.R. and Pal, S., Computational intelligence for pattern recognition, International Journal of Pattern Recognition and Artificial Intelligence, 2002, 16, 773-79.

Page, B. and Hilty, L.M. (eds.), Umweltinformatik – Informatikmethoden für Umweltschutz und Umweltforchung, Handbuch der Informatik, 1995, 13, München, 2nd edition.

Page, B. and Raustenstrauch, C., Introduction to environmental information systems, In (Eds.) Raustenstrauch, C and Patig, S., Environmental Information Systems in Industry and Public Administration, Idea Group Publishing, 2001.

Panella, M., Rizzi, A., Mascioli, F.M., and Martinelli, G., Constructive MoG neural networks for pollution data forecasting, Proceedings of the 2002 International Joint Conference on Neural Networks (IJCNN'02), 2002, 417-422.

Papoulis, A., Probability, Random Variables and Stochastic Processes, 4th edition, 2002, Boston, McGraw-Hill.

Parmee, I.C, Poor-definition, uncertainty, and human factors - satisfying multiple objectives in real-world decision-making environments, Evolutionary Multi-Criterion Optimization, Proceedings of EMO, 2001, 52-66.

Paterson, K., Sagady, J., and Hooper, D., Analysis of air quality data using positive matrix factorization, Environmental Science & Technology, 1999, 33, 635-641.

Pham, D.T., An introduction to artificial neural networks, Neural Networks for Chemical Engineers, (Ed.) Bulsari, A.B., Elsevier Science B. V., 1995, 1-19.

Press, S. and Wilson, S., Choosing between logistic regression and discriminant analysis, Journal of the American Statistical Association, 1978, 73, 699-705.

Quinlan, J., Bagging, boosting and c4.5, In Proceedings of the Thirteenth National Conference on Artificial Intelligence, 1996, 725-730, AAAI Press and the MIT Press.

de Ridder, D. and Duin, R., Sammon's mapping using neural networks: a comparison, Pattern Recognition Letters, 1997, 18, 1307-1316.

Rosenblatt, F., The perceptron: A probabilistic model for information storage and organization in the brain, 1958, Psychological Review, 65, 386-408.

Sammon Jr., J.W., A nonlinear mapping for data structure analysis, IEEE Transactions on Computers, 1969, C-18, 401-409.

Schafer, J.L., Analysis of Incomplete Multivariate Data, Monographs on Statistics and Applied Probability, 1997, No 72, Chapman & Hall, London.

Simon, L., Karim, M., and Schreiweis, A., Prediction and classification of different phases in a fermentation using neural networks, Biotechnology Techniques, 1998, 12, 301-304.

Schlink, U., Dorling, S., Pelikan, E., Nunnari, G., Cawley, G., Junninen, H., Greig, A., Foxall, R., Eben, K., Chatterton, T., Vondracek, J., Richter, M., Dostal, M., Bertucco, L., Kolehmainen, M., and Doyle, M., A rigorous inter-comparison of ground-level ozone predictions, Atmospheric Environment, 2003, 37, 3237-3253.

Singh, M. and Valtorta M., Construction of Bayesian network structures from data - a brief survey and an efficient algorithm,

International Journal of Approximate Reasoning, 1995, 12, 111-31.

Stein, L., Creating a bioinformatics nation, Nature, 2002, 417, 119-120.

Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E.S., and Golub, T., Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation, Proceedings of the National Academy of Sciences of the United States of America, 1999, 96, 2907-12.

Torgerson, W., Multidimensional scaling: I. Theory and method, Psychometrika, 1952, 17, 401-419.

Tsakonas, A. and Dounias, G., Hybrid computational intelligence schemes in complex domains: An extended review, Methods and Applications of Artificial Intelligence, 2002, 2308, 494-511.

Tsaptsinos, D., Back-propagation and its variations, Neural Networks for Chemical Engineers, (Ed.) Bulsari, A.B., Elsevier Science B. V., 1995, 33-76.

Tucker, A., Swift, S., and Liu, X., Variable grouping in multivariate time series via correlation, IEEE Transactions on Systems Man and Cybernetics, Part B-Cybernetics, 2001, 31, 235-245.

Turing, A.M., Computing machinery and intelligence, Mind, 1950, 433-460.

Törönen P., Kolehmainen M., Wong G., and Castren E., Analysis of gene expression data using self-organizing maps, FEBS Letters, 1999, 451, 142-146.

Ulbricht, C., Multi-Recurrent networks for traffic forecasting, Proceedings of the Twelfth National Conference on Artificial Intelligence, AAAI Press / MIT Press, Cambridge, MA, 1994, 883-888.

Ultsch, A. and Siemon, H.P., Kohonen's self organizing feature maps for exploratory data analysis, In Proc. INNC'90, Int. Neural Network Conf., Dordrecht, Netherlands, 1990, 305-308.

Valkonen V-P., Kolehmainen M., Lakka H-M., and Salonen J., Insulin resistance syndrome revisited: application of Self-organizing maps, International Journal of Epidemiology, 2002, 31, 864-871.

Van Hulle M., Faithful Representations and Topographic Maps: from Distortion- to Information-Based Self-Organization, John Wiley & Sons, INC., New York, U.S.A, 2000.

Wahde, M. and Hertz, J., Coarse-grained reverse engineering of genetic regulatory networks, BioSystems, 2000, 55, 129-136.

Walleczek, L.(ed.), Self-Organized Biological Dynamics & Nonlinear Control, 2000,Cambridge University Press.

Ward, O.P., Fermentation Biotechnology, reprinted by John Wiley & Sons Ltd, West Sussex, England, 1995.

Weigend, A. and Gershenfeld N. (eds.), Time Series Prediction, Proceedings of the NATO Advanced Research Workshop on Comparative Time Series Analysis held in Santa Fe, New Mexico, May 14-17, 1992.

Willmott, C., Some comments on the evaluation of the model performance, Bulleting of American Meteorological Society, 1982, No 63, 1309-1313.

Willmott, C., Ackleson, S., Davis, R., Feddema, J., Klink, K., Legates, D., O'Donnell, J., and Rowe, C., Statistics for the evaluation and comparison of models, Journal of Geophysical Research, 1985, 90, 8995-9005.

Yao, X., Evolving artificial neural networks, Proceedings of the IEEE, 1999, 87, 1423-1447.

Yeung, K. and Ruzzo, W., Principal components analysis for clustering gene expression data, Bioinformatics, 2001, 17, 763-774

Zadeh, L.A., Fuzzy Logic and soft computing issues, contention and perspectives, Proc. 3rd Int. Conf. Fuzzy Logic, Neural Nets and Soft Computing, IIZUKA, Japan, 1994, 1-2.

Zimmermann H.-J., Fuzzy Set Theory and its Applications, Second Edition, Kluwer Academic Publishers, 1991