JYVÄSKYLÄ STUDIES IN COMPUTING 33

Sami Saalasti

Neural Networks for Heart Rate Time Series Analysis

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi Villa Ranan Paulaharju-salissa
joulukuun 19. päivänä 2003 kello 12.

Academic dissertation to be publicly discussed, by permission of
the the Faculty of Information Technology of the University of Jyväskylä,
in the Building Villa Rana, Paulaharju Hall, on December 19, 2003 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2003

Neural Networks for Heart Rate Time Series Analysis

JYVÄSKYLÄ STUDIES IN COMPUTING 33

Sami Saalasti

Neural Networks for Heart Rate Time Series Analysis

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2003

Editors
Tommi Kärkkäinen
Department of Mathematical and Information Technology, University of Jyväskylä
Pekka Olsbo, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

URN:ISBN 951391707X
ISBN 951-39-1707-X (PDF)

ISBN 951-39-1637-5 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä
and ER-Paino Ky, Lievestuore 2003

ABSTRACT

Saalasti, Sami
Neural Networks for Heart Rate Time Series Analysis
Jyväskylä: University of Jyväskylä, 2003, 192 p.
(Jyväskylä Studies in Computing
ISSN 1456-5390; 33)
ISBN 951-39-1707-X
Finnish summary
Diss.

The dissertation introduces method and algorithm development for nonstationary, nonlinear and dynamic signals. Furthermore, the dissertation concentrates on applying neural networks for time series analysis. The presented methods are especially applicable for heart rate time series analysis.

Some classical methods for time series analysis are introduced, including improvements and new aspects for existing data preprocessing and modeling procedures, e.g., time series segmentation, digital filtering, data-ranking, detrending, time-frequency and time-scale distributions. A new approach for the creation of hybrid models with a discrete decision plane and limited value range is illustrated. A time domain peak detection algorithm for signal decomposition, i.e., estimation of a signal's instantaneous power and frequency, is presented.

A concept for constructing reliability measures, and the utilization of reliability to improve model and signal quality with postprocessing are grounded. Also a new method for estimating the reliability of instantaneous frequency for time-frequency distributions is presented. Furthermore, error tolerant methods are introduced to improve the signal-to-noise ratio in the time series.

Some new principles are grounded for the neural network theory. Optimization of a time-frequency plane with a neural network as an adaptive filter is introduced. The novelty of the method is the use of a neural network as an inner function inside an instantaneous frequency estimation function. This is an example of a new architecture called a transistor network that is introduced together with the general solution for its unknown parameters. Applicability of the dynamic neural networks and model selection using physiological constraints is demonstrated with a model estimating excess post-exercise oxygen consumption based on heart rate time series. Yet another application demonstrates the correlation between the training and testing error and usage of the neural network as a memory to repeat the different RR interval patterns.

Keywords: heart rate time series, neural networks, preprocessing, postprocessing, feature extraction, respiratory sinus arrhythmia, excessive post-exercise oxygen consumption

Author Sami Saalasti
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Supervisors Professor Tommi Kärkkäinen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Professor Pekka Neittaanmäki
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Professor Pekka Orponen
Department of Computer Science and Engineering
Helsinki University of Technology
Finland

Professor Heikki Rusko
Research Institute for Olympic Sports
Jyväskylä, Finland

Opponent Research Professor Ilkka Korhonen
VTT Information Technology
Tampere, Finland

ACKNOWLEDGMENTS

The dissertation is based on years of collaboration with Ph.D Joni Kettunen, who has inspired and attended the work in various ways. His peculiarity is the number of ideas he provides daily, challenging me to find better mathematical solutions for given problems.

I would like to express my sincere gratitude to professors Tommi Kärkkäinen and Pekka Neittaanmäki for their trust and support. Without their intervention the process to complete the dissertation would not have begun. The work of Professor Kärkkäinen in merging neural networks and the optimization theory has been a great inspiration and has provided new insights into the research.

I also wish to thank all the staff in Firstbeat Technologies, especially Aki Pulkkinen and Antti Kuusela. Furthermore, I wish to express my gratitude to Professor Heikki Rusko and M.Sc Kaisu Martinmäki from the Research Institute for Olympic Sports.

This doctoral thesis is partially based on my licentiate thesis, "Time series prediction and analysis with neural networks", published in the year 2001. The work is partially reprinted in this dissertation. The work was supervised by Professor Pekka Orponen, who I wish to express my gratitude. Furthermore, the physiological interpretation is greatly affected by the work of our multi-scientific, skillful team, and several publications [76, 77, 78, 97, 98, 139, 140, 141, 149, 150, 151, 152] are exploited for this work.

This work was financially supported by COMAS Graduate School from the University of Jyväskylä. The author has participated in two TEKES-projects at the Research Institute for Olympic Sports and Firstbeat Technologies. Both of these projects have provided much of the experience, data and results presented in this dissertation.

Finally, I want to express my appreciation to my wife Piia for her support, assistance with medical terminology, patience and understanding.

Jyväskylä, 9th December 2003

Sami Saalasti

CONTENTS

NOTATIONS AND ABBREVIATIONS

Matrix and vector operations

Real numbers are presented as $a, b, c, \ldots, w, x, y, z$

Vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots, \mathbf{w}, \mathbf{x}, \mathbf{y}, \mathbf{z}$

Matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{W}, \mathbf{X}, \mathbf{Y}, \mathbf{Z}$

Matrix vector multiplication

$$\mathbf{Y} = \mathbf{Ax} \Leftrightarrow y_i = \sum_{j=1}^{m} a_{ij} x_j \Leftrightarrow \begin{cases} a_{11}x_1 & + \ldots + & a_{1m}x_m & = & y_1 \\ \vdots & + \ldots + & \vdots & = & \vdots \\ a_{n1}x_1 & + \ldots + & a_{nm}x_m & = & y_m \end{cases}$$

,

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1m} \\ a_{21} & a_{22} & \ldots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nm} \end{bmatrix} \in \mathbf{R}^{n \times m}, \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} \in \mathbf{R}^m \equiv \mathbf{R}^{m \times 1}, \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbf{R}^n$$

Matrix transpose $\mathbf{B} = \mathbf{A}^T \Leftrightarrow b_{ij} = a_{ij}$

Element by element multiplication $\mathbf{A} \cdot \mathbf{B} = a_{ij} b_{ij}, \mathbf{A}, \mathbf{B} \in \mathbf{R}^{n \times m}$

Hessian matrix contains second order derivatives of a function y respect to variable x_i. The element in the ith row and jth column of the matrix is

$$\frac{\partial^2 y}{\partial x_i \partial x_j}.$$

Euclidean norm

$$||\mathbf{x}|| = \sqrt{\sum_{k=1}^{N} x_k^2}, \ \mathbf{x} \in \mathbf{R}^N.$$

Physiology, Heart rate variability

ULF Ultra-low-frequency band of the power spectrum.

	The frequency range is 0.0001-0.001 Hz.
VLF	Very low-frequency band of the power spectrum.
	The frequency range is 0.003-0.03 Hz.
LF	Low-frequency band of the power spectrum.
	The frequency range is 0.04-0.15 Hz.
HF	High-frequency band of the power spectrum.
	The frequency range is 0.15-0.4 Hz or 0.15-0.5 Hz.
LF+HF	Both low- and high-frequency bands of the power spectrum.
HR	Heart rate.
HRV	Heart rate variability.
RRI	RR interval in milliseconds.
HR	Heart rate in beats per minute.
IHR	Instantaneous heart rate in beats per minute.
IBI	Inter-beat interval in milliseconds.
HP	Heart period in milliseconds.
NNI	Normal-to-normal interval.
ECG	Electrocardiogram.
RSA	Respiratory sinus arrhythmia.
bpm	Beats per minute.
ms	Milliseconds.
EPOC	Excess Post-exercise Oxygen Consumption.
HRmax	Maximal heart rate.
VO2	Oxygen consumption.
VO2max	Maximal oxygen consumption.
pVO2	VO2 proportional to VO2max, $pVO2 = \frac{VO2}{VO2max}$.
pHR	HR proportional to HRmax, $pHR = \frac{HR}{HRmax}$.
EB	Extra beat.
MB	Missing beat.

Data preprocessing and modeling

FFT	Fast-Fourier transformation.
STFT	Short-time Fourier transformation.
SPWV	Smoothed pseudo Wigner-Ville transformation.
TFRD	Time-frequency distribution.
Hz	Hertz, how many times per second, $1\ Hz = \frac{1}{s}$.
PSD	Power spectral density.
SSE	Sum of squared errors.
MSE	Mean-squared error.
MRE	Mean relative error.
NMSE	Normalized mean-squared error.
RMSSD	The square root of the mean of the sum of the squares of differences.
MUSIC	MUltiple SIgnal Classification method.

AR	Autoregressive.
MA	Moving average.
ARIMA	Autoregressive integrated moving average model.

Segmentation

CP	Change point.
GLR	Generalized likelihood ratio test.
LLR	Log-likelihood ratio.
ISR	Initial search region length.
MRL	Minimum region length.

Neural networks

$w_{ij}^{(l)}$	A weight connection from unit (neuron) i in layer l to unit j in layer $l+1$.
$s_j^{(l)}$	Excitation of unit j in layer l.
$f(s_j^{(l)})$	Activation of unit j in layer l.

epoch	One epoch means the training of the network with entire data once.
NN	A neural network.
FFNN	A feed-forward neural network.
HMM	A hidden markov model.
MLP	A multilayer perceptron.
FIR	Finite impulse response.
TDNN	A Time delayed neural network.
RBFN	A radial basis function neural network.
GRNN	A generalized regression neural network.
LRNN	A family of neural networks called locally recurrent neural networks.
ORNN	Output-recurrent neural network.
PCA	Principal component analysis.
SOM	Self-organized map (Kohonen network).

Hybrid model with discrete decision plane

CC	Credibility coefficients.
#CC	Number of credibility coefficients.
DC	Discrete coordinates.
DDP	Discrete decision plane.
HMDD	Hybrid model with discrete decision plane.

1 INTRODUCTION

Physiological time series are challenging: they require methods which are tolerant for signal artifacts and methods providing temporal dynamics (nonstationarity) and nonlinearity. Examples of various physiological time series include a heart rate time series, diastolic- and systolic blood pressure, skin conductance, ventilation, oxygen consumption, electromyograph, electroencephalograph, electrocardiograph, etc. In this dissertation, the focus will be on the heart rate time series.

Korhonen [86, p. 14] lists alterations in heart rate variability to be linked to the various physiological and medical provocations, including changes in posture, hypovolemic stress, isometric and dynamic exercise, mental stress, introduction of vasoactive drugs and pharmacological autonomic blocking. Furthermore, a decrease in the heart rate variability has been linked to pathologies, e.g., sudden infant death syndrome, diabetes mellitus, myocardial infarction, myocardial dysfunction, and reinnervation after cardiac transplantation. Alterations in the heart rate variability has also been linked to different sleep stages, levels of workload, personal fitness and smoking. In the 1990s an increased interest in heart rate variability studies has provided new insights into human physiology, but clinical standards and applications are yet to be developed.

The emphasis of this dissertation is on neural networks. They are often linked to artificial intelligence, but rather perhaps, should be treated as powerful data-driven numerical methods applied to a variety of problems, e.g., to model phenomena or time series, or to construct expert systems for classification, decision and detection. In classical artificial intelligence, expert knowledge is used to construct inference rules with semantics similar to programming languages. With neural networks, network training happens at the syntactic level and semantically the network is tried to be proven reasonable only after training.

Nevertheless, the modeling of expert knowledge, extraction of signal characteristics or pure time series modeling requires a variety of mathematical tools. Figure 1 illustrates a set of numerical methods presented in this work applicable for physiological modeling. Furthermore, the map illustrates different dimensions and classes for the methods resulting in different applicability.

The x-coordinate of the map illustrates method applicability for real-time, or on-line processing. Requirements for such methods include, e.g., optimized and CPU-friendly complexity. If a method is to be applied for embedded systems the CPU-requirements become even more important. It is notable that, in general, the methods available for on-line processing are also capable for large dataset processing.

The y-coordinate illustrates the method's capability to tolerate temporal variation in the system, i.e., how much the system parameters vary in time. This is equivalent for examining stationarity assumptions of the model. The methods high in the hierarchy also bear nonlinearity much better. Naturally, the classical

linear models localize low in temporal hierarchy.

It should be noted that the mind map is illustrative and should not be interpreted as absolute. For example, a neural network model called multi-layer perceptron is a model that is both trained and used with the solved parameters, and the calculation complexity for the usage and training are not similar. The training may consist of a complicated numerical optimization process requiring much computational time and memory, e.g., calculation of Hessian-matrix in Newton's method. The training may also be implemented in an on-line manner resulting in faster computation time for one iteration in the optimization algorithm. The drawback, then, is that the optimization will require more iterations to find a local minimum, compared to Newton's method. The resulting network is just a computational unit that is fast to execute. To be more precise, the network complexity may affect the computational speed, and a large number of network parameters may result in slow computation. Hence, it is important not to take the two-dimensional figure literally; the author acknowledges that it is not a mathematically exact presentation and various definitions and concepts may be interpreted in different ways. For example, K-means clustering is usually considered as a clustering algorithm, but in our context it is utilized to find network parameters for the radial basis function network.

Integration of algorithms, methods and models

One possibility of describing the integration of different methods is a forward flow, where signal preprocessing and signal decomposition provides characteristics of a signal to be further analyzed or modeled by, for example, a neural network. Different methods integrate as preprocessing techniques are used to segment and decompose the signal, observations are drawn and a model is constructed and optimized with a proper strategy. The model may produce estimates for another signal, classify states or perhaps predict future values in a time series.

However, artificial division of mathematical techniques, for example, to modeling or preprocessing may be questioned. For example, a neural network model may be used as a filter, such as a signal preprocessing technique, to elicit desired signal characteristics. Furthermore, we described the process to be a flow forward. This only describes the simple applications, since a complicated system may include several steps with different preprocessing, postprocessing, linear and nonlinear methods and parallel or recursive processes. Such an iterative and incremental development also underlines the current state-of-the-art methodologies for software development in general (e.g., [74]).

Signal preprocessing is often used to improve signal's explanatory value or signal-to-noise ratio. Preprocessing techniques may also be used for signal decomposition, for example, to its frequency and power contents. Furthermore, signal characterization (or feature extraction) may be used to build quantitative measures of the signal. For example, with the heart rate time series, the low-

and high-frequency bands of the power spectrum are construed to be noninvasive measures of autonomic control. Decomposition of the signal may include several methods, e.g., Wavelet transform, peak detection, or short-time Fourier transformation.

Reliability of a given measure or model estimate may be exploited in various ways. Reliability may guide to seek an estimate from an alternative model or it may be used to improve the accuracy of the model in the time domain. Reliability may also be exploited with hybrid models to decide the use of a proper method or to focus data preprocessing and artifact detection or correction in invalid regions.

Segmentation may be used to guide different methods or models to process different parts of the signal. Identification of a segment is based on signal characteristics. The methods interact and, for example, decomposition information may be used for both model construction and segmentation. The process may also be recursive, in that the model outputs can be used to focus preprocessing and segmentation. The system recursively improves until a steady state is achieved.

Author's contributions

The author wishes to give new insights and perspective in the physiological time series analysis and furthermore contributes the following:

1. An extensive methodological review.

2. An approach to creating hybrid models with discrete decision-plane and limited value range. Examples and analysis of the new method are provided.

3. A new concept called a transistor network. The architecture is introduced together with a general analytic solution for the network parameters.

4. Optimization of a three-dimensional discrete plane to provide optimal centre mass. Application for adaptive neural network filtering with efficient use of neural network parameters. A neural network is used as an inner function of objective function. The methodology may be applied in to the detection of breathing frequency strictly from the heart rate time series.

5. The extension of a segmentation method called generalized likelihood ratio test, to multivariate on-line applications with simple estimation and error functions. General properties of the algorithm is investigated, including, the algorithm's sensitivity to its own parameters.

6. A geometric approach (a.k.a. "peak detection") for the estimation of a signal's instantaneous power and frequency. The method may be utilized to estimate respiration frequency from chest expansion data.

7. Concepts for automated control of signal artifacts: error tolerant models and improving signal-to-noise ratio with data preprocessing and postprocessing.
8. Construction of reliability measures on various models and exploiting the estimates to form time domain corrections to the estimated time series.
9. Use of constraints in neural network model selection. An application to model excess post-exercise oxygen consumption strictly from the heart rate via oxygen consumption modeling with a temporal neural network.

Structure of the dissertation

The introduction is presented in the First Section. The 2nd Section outlines the characteristics of the heart rate time series. Section three covers the concepts which form the framework for model building of the physiological phenomena: feature extraction, signal preprocessing and postprocessing.

In Section four a detailed description of the neural networks applied in this dissertation is provided. Furthermore, the section illustrates the author's perspective in neural network optimization, providing the founded decisions made regarding the selection of the optimization methods that are later used in the applications section.

The fifth Section describes a new general concept for constructing hybrid models with a discrete decision plane. In addition, examples are provided to illustrate the justification of the method and failure of the divide and conquer methodology.

Section six describes in a very detailed manner a generation of two physiological neural network based models for to estimate the excessive post-exercise oxygen consumption and the detection of respiratory frequency strictly from the heart rate time series. In addition, a neural network training simulation is provided to illustrate the coupling of training and testing error with large datasets. Finally, the conclusions of the work are presented in Section seven.

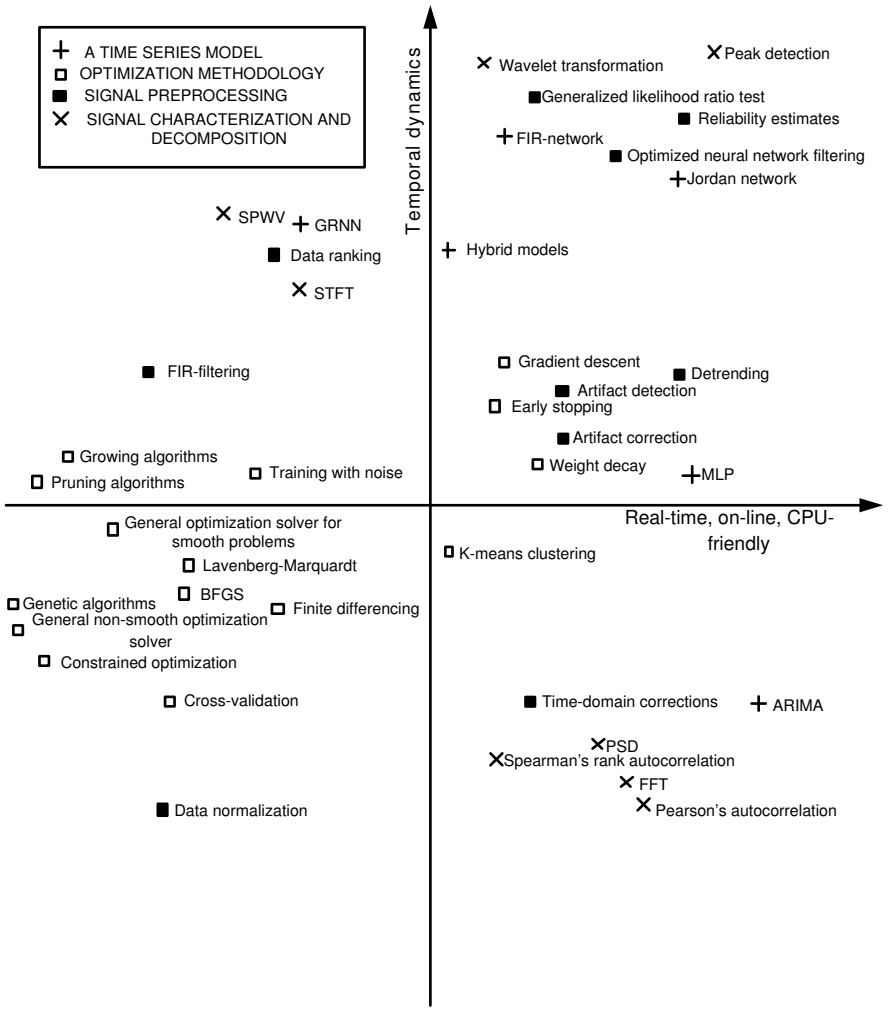

Figure 1: Mind map of different methods and models of the dissertation. The x-coordinate illustrates how well a method is applicable for real-time processing. The strength of stationarity assumptions is described with the y-coordinate.

2 HEART RATE TIME SERIES

In this dissertation, the link between the methodology and the examined phenomena is the human physiology. The autonomic nervous system has primary control over the heart's rate and rhythm. Heart rate variability has been suggested as a noninvasive measure of autonomic control. A short introduction to cardiovascular- and the autonomic nervous system is presented in Section 2.1, although a thorough study of this topic is outside the scope of this dissertation. The intention of this section is to provide sufficient background and characteristics of the heart rate time series dynamics for the applications presented later in the dissertation. The heart rate time series are complex, unpredictable, nonlinear and nonstationary with temporal cyclic and asyclic components. They can be derived from electrocardiograph and can contain electrocardiograph-specific and heart-rate-monitor-related artifacts. Both resampling and nonlinear transformation of RR intervals to heart rate will be demonstrated to distort the heart rate interpretation and statistics.

Heart rate has a connection to other physiological measures, such as oxygen consumption, tidal volume, respiration frequency, ventilation and blood pressure. To form the essence of the phenomena with mathematical modeling, all the information, such as multivariate signals, may be used to improve understanding. Heart rate and blood pressure are influenced by the respiration cycle that is visible in the time series. A preliminary example of respiration coupling with heart rate and blood pressure is discussed in Section 2.5.

2.1 Autonomic nervous system and heart rate variability

The cardiovascular system consists of myocardium (the heart) (see Figure 2), veins, arterias and capillaries. The main function of cardiovascular system is to transmit oxygen to the muscles and remove carbon dioxide. Furthermore, it transmits waste products to the kidneys and liver, white blood cells to tissues, and controls the acid base balance of the body.

The autonomic nervous system (see Figure 3) has primary control of the heart's rate and rhythm. It also has control over the smooth muscle fibers, glands, blood flow to the genitals during sexual acts, gastrointestinal tract, sweating and the pupillary aperture. The autonomic nervous system consists of *parasympathetic* and *sympathetic* parts. They have contrary effects on the human body, for example, parasympathetic activation preserves the blood circulation in muscles, while sympathetic activation accelerates it. The primary research topic in the field of *heart rate variability* (HRV) research is to quantify and interpret the autonomic process of the human body and the balance between parasympathetic and sympathetic activation [46, 120].

The monitoring of the heart rate and, especially, its variability, is an attempt

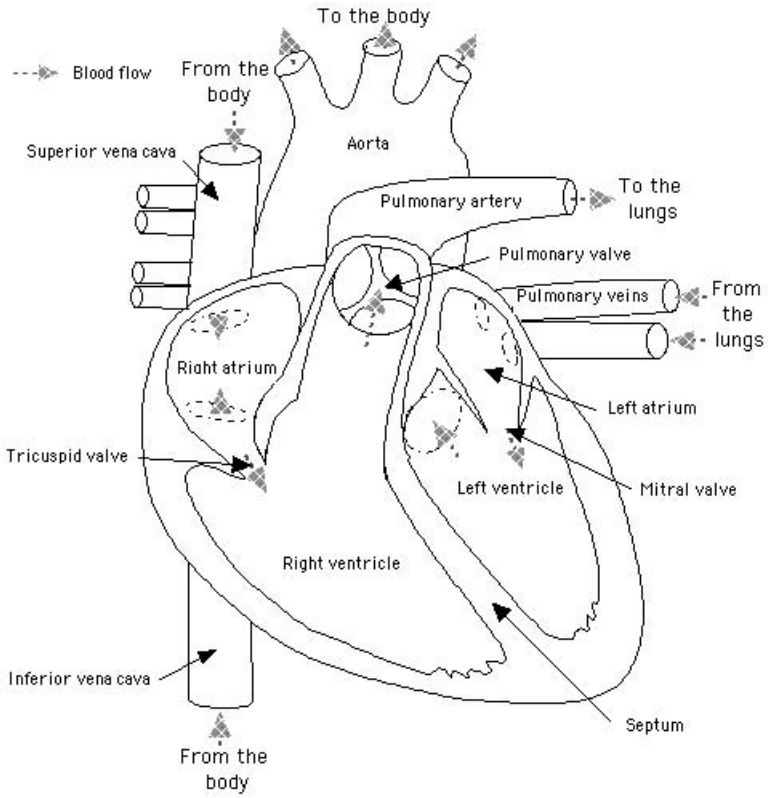

Figure 2: Interior view of the heart's anatomy. Original figure from *EnchantedLearning.com*.

to apply an indirect measurement of autonomic control. Hence, HRV could be applied as a general health index, much like noninvasive diastolic and systolic blood pressure. The (clinical) applications for such a system would include, for instance the monitoring of hypovolemic or mental stress, preventing and predicting myocardial infarction or dysfunction, the monitoring of isometric and dynamic exercise, the prediction and diagnosis of overreaching, measuring vitality, the monitoring of recovery from exercise or injury, etc. However, the diagnostic products are yet to come, since at the present no widely approved clinical system for the monitoring of the autonomic nervous system via heart rate exists. Commercial manufacturers using a variety of methods and HRV indices do exist, but the development of such systems have not been guided in any way and has been left to free market forces. As a result, no standardization has been established [61].

An example of HRV indices include spectral parameters derived from the recording of the heart rate. However, several difficulties exist in interpreting HRV in such a way. Especially the respiratory component in the *high-frequency band* (HF) of the heart rate signal (0.15-0.4 Hz or 0.15-0.5 Hz) has a substantial impact

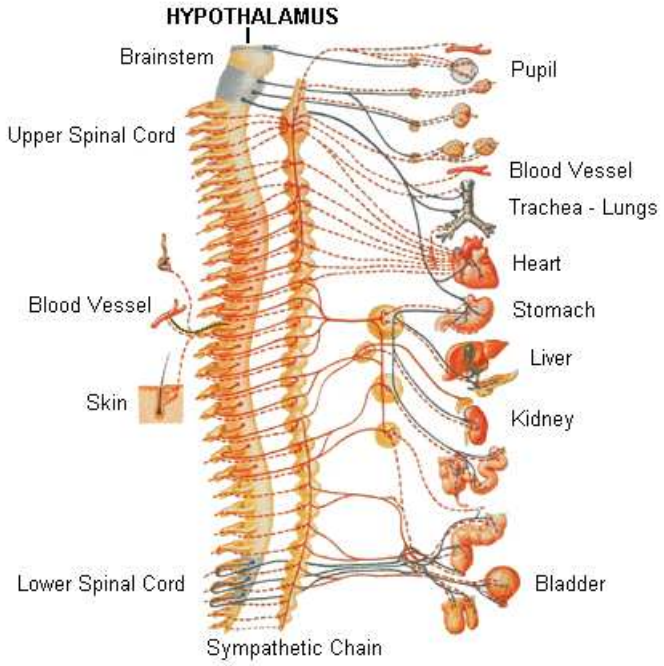

Figure 3: Autonomic nervous system. Original figure from *National Parkinson Foundation*, *www.parkinson.org*.

on HF independent of changes in parasympathetic activation. The respiratory component of the heart rate signal may also overlap the *low-frequency band* (LF) of the heart rate signal (0.04-0.15 Hz) resulting in a complicated *power spectrum* and interpretation. The power spectrum provides the information of how the power or energy distributes as a function of frequency. In HRV analysis the energy unit is expressed in a ms^2. Spectral analysis is further discussed in Section 3.1.1. LF power has been linked to cardiac sympathetic activation but there are reports that exist which have failed to find the link between them [6].

The invasive research of the autonomic nervous system is founded through experiments on animals, and using drugs stimulating or blocking sympathetic and parasympathetic activation in a direct or indirect manner on human subjects. Drugs stimulating sympathetic activation, known as *symphathomimetic drugs*, include norepinephrine, epinephrine and methoxamine. Blocking may be achieved with drugs such as reserpine, guanethidine, phenoxybenzamine or phentolamine. The list of drugs is quite large and they may affect different points in the stimulatory process. For example, hexamethonium may be used to block the transmission of nerve impulses through the autonomic ganglia and hence the drug blocks both sympathetic and parasympathetic transmissions [46, p. 696]. The blocking

of the vagal system and its effect on heart rate variability has been studied, e.g., in [97, 98, 132].

A variety of noninvasive research on HRV also exists. In Pitzalis et al. [133] noninvasive methods, so called alpha-index and sequence analysis, are compared to evaluate the correlation and agreement between the baroreflex sensitivity[1] obtained with invasive measures (drug stimulation).

2.2 Time series categories

"A time series is a set of observations generated sequentially in time", [15, p. 21]. *A time series model* is a system of definitions, assumptions, and equations set up to describe particular phenomena expressed in a time series. *Time series modeling* describes the process in building the model.

According to Chatfield [24], a time series is said to be *deterministic* if its future values are determined by some mathematical function of its past values. *Statistical* or *stochastic* time series can be described by some probability distribution. The time series is said to be *static* or *stationary* if its statistics, usually mean and variance, do not change in time. On the other hand, nonstationary signals can contain many characteristics: A time series has a *trend* if it has a long-term change in its mean. In a time series having *seasonal fluctuation*, there is some annual, monthly, or weekly variation in the series. An *outlier* means an observation which differs or is unexpected compared to other values in the series.

A *chaotic* time series is generated by some dynamical nonlinear deterministic process which is critically dependent on its initial conditions. A classic example is the *Henon map*:

$$x(k) = 1 + w_1 x(k-2) - w_2 x(k-1)^2, \tag{1}$$

where w_1 and w_2 are free parameters. Lehtokangas has demonstrated [92, p. 5-7] that if two implementations of the Henon map are written as *Matlab*[2] programs, then changing the order of the last two terms, i.e., $x(k) = 1 - w_2 x(k-1)^2 + w_1 x(k-2)$, results in two different time series. The absolute error between the results grows exponentially between the first and 90th iteration and then settles down. The difference between the implementations is the result of rounding errors due to changing the order of the terms.

The presented characteristics apply mostly to the classical time series analysis. In general, a heart rate time series does not purely belong to any of the given classes. The category depends on the observed time series, the length of the signal and the nature of the recording. For example, a heart rate time series produced by a metronome-spaced breathing test under steady conditions appears to

[1] Depressed baroreflex sensitivity plays a prognostic role in patients with a previous myocardial infarction.

[2] Matlab is a language for technical computing. It integrates computation, visualization, and programming in an environment where problems and solutions are expressed in mathematical notation [100].

be stationary for individuals having a strong respiratory component in their HR. The short-term oscillations corresponding to the stationary breathing frequency results in a well-behaved signal. In an *ambulatory recording*, the outcome is quite different: all the major frequency and power components in the signal have a considerable temporal variation. Ambulatory recording is performed outside the controlled laboratory environment or laboratory protocols, so that the nonstationary changes in the signal are rather a rule than an exception in such free measurement. Movements, postural change, etc. will result in the adjusting of blood pressure and muscular blood circulation. Hence, the actions cause alterations to HR. If only the HR time series is observed, then the changes appear nonstationary and unpredictable in nature.

2.3 From continuous electrocardiogram recording to heart rate time series

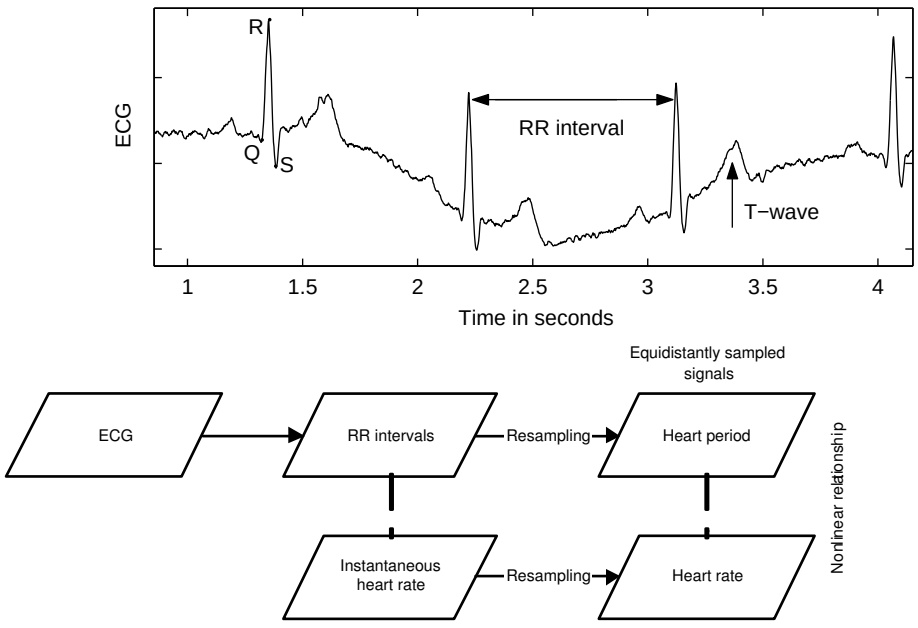

Figure 4: Different signals derived from electrocardiograph and an example ECG time series.

Abbreviations, synonyms and expressions used for signals derived from *electrocardiogram* (ECG) recording are presented in Table 1. Electrocardiogram represent the recording of the electrical potential of the heart, carried out using sensors positioned on the surface of the body. *RR interval* (RRI), inter-beat interval and cycle interval are synonyms representing different names for the same non-

Abbreviation	Explanation	Unit
NNI	normal-to-normal interval	ms
RRI	RR interval	ms
	cyclic interval	ms
IBI	inter-beat interval	ms
HP	heart period	ms
	beat-to-beat time series	ms
HR	heart rate	bpm
IHR	instantaneous heart rate	bpm

Table 1: Abbreviations and synonyms used in the dissertation for signals derived from electrocardiograph recording.

equidistantly sampled signal. RR interval is expressed as a time between consecutive QRS-waves of the electrocardiogram (see Figure 4). Instantaneous heart rate is a nonlinear transformation of RRI and has beats per minute (bpm) as its unit. A heart rate time series is resampled from RRI to have equidistant sampling and transformed to bpm unit. In this dissertation a heart period and beat-to-beat time series are regularly sampled counterparts of RRI. A normal-to-normal interval is defined as the interval between two successive normal, non-artifactual, complexes in ECG [120].

ECG may be recorded with a variety of commercial equipment. For clinical use the Holter ECG is most frequently used [167]. There exists mobile and event monitors, able to record ECG for various time periods. Heart rate monitors do not store ECG but rather the RRI or average HR, for example, average of the HR for the last 15 seconds.

Scientifically used ECG recorders, like the Holter ECG, do not have memory limitations, and such devices use a high sampling rate to increase the precision of the signal representation. Sampling rates between 125 to 4096 Hz are used by the commercial manufacturers. There also exists ECG recorders sampling at a variable rate [1, 2, 18]. Furthermore, there exists a number of methods for the QRS-wave recognition in ECG [42, 109, 135, 167].

RR intervals and heart period are commonly expressed in milliseconds (ms), and (instantaneous) heart rate as beats per minute (bpm). The nonlinear transformation, $bpm = 60000/ms$, between the signals is presented in Figure 5. The conversion from RRI to IHR, or HP to HR, may distort statistics and influence the physiological interpretation in experimental design and tests. For example, in Quigley and Berntson [142] the differences in the interpretation of autonomic control with heart period versus heart rate is studied.

Time series analysis, e.g., calculation of the power spectrum, is based on equidistantly sampled signals. Two main approaches are used to transform a sequence of RR intervals into equidistantly sampled heart period time series:

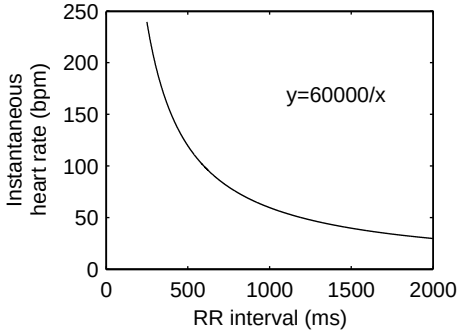

Figure 5: The nonlinear relationship between instantaneous heart rate and RRI.

the interpolation- and window-average-resampling methods. The interpolation[3] methods may be carried out in a step-wise manner, linearly or by spline function [86, p. 23]. The step-wise linear interpolation resampling method is described in Algorithm 2.1 and illustrated in Figure 6. A sampling frequency of 5 Hz (200 ms) is used in this dissertation.

The resampled signal may also be desampled back to a RR interval sequence without information loss as illustrated in Algorithm 2.2. This property allows us to store only one of the signals, equidistant or a non-equidistant signal, as the transformation between the signals is enabled. Notice that the window average resampling does not contain this property.

Resampling changes the statistics of the ECG derived signals. Even if the sampling accuracy is perfect and no information is lost, the procedure will affect the basic statistics such as mean and variation. This is illustrated in Figure 7. To demonstrate this let us consider two beats lasting 500 and 300 milliseconds, respectively. When sampled with Algorithm 2.1 and 5 Hz sampling frequency, the time series results into 500, 500, 400, 300 milliseconds. The mean values of the two RR intervals and resulting resampled heart period time series are 400 and 425 milliseconds, respectively.

Algorithm 2.1 *Resampling with step-wise linear interpolation.*

0. Let x present the sequence of RR intervals and y the resampled output vector. Set the remainders to zero:

$$r_1 = r_2 = 0.$$

Then set input and output vector indices to one

$$i = j = 1.$$

[3]The dictionary of mathematics [31] defines interpolation as follows: "For known values $y(1), y(2), \ldots, y(n)$ of a function $f(x)$ corresponding to values $x(1), x(2), \ldots, x(n)$ of the independent variable, interpolation is the process of estimating any value y' of the function for a value x' lying between two of the values of x, e.g., $x(1)$ and $x(2)$. *Linear interpolation* assumes that $(x(1), y(1))$, (x', y'), and $(x(2), y(2))$ all lie on a straight-line segment".

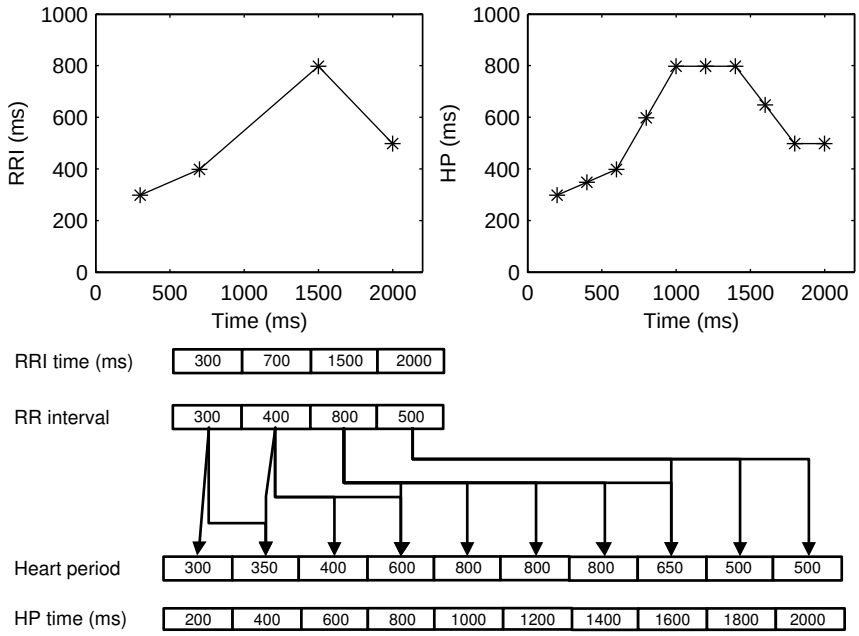

Figure 6: Resampling with step-wise linear interpolation. RR intervals stored in a vector are transformed to equidistantly sampled heart period signal, where the time difference between each vector position is 200 milliseconds long (5 Hz sampling). The time vector for RRI in milliseconds is a cumulative sum of the RRI.

Notice that the sampling interval $\triangle T$ should not exceed the minimum value of the RR intervals.

Length of the input vector is n. Then the maximum length of the output vector is

$$\frac{1}{\triangle T}\sum_{k=1}^{n} x(k).$$

In computer implementation the output length may be truncated after the algorithm execution.

1. *Calculate the full times the sampling interval $\triangle T$ goes to difference of the current beat and the time r_2 reserved in the previous iteration:*

$$c = \lfloor \frac{x_i - r_2}{\triangle T} \rfloor,$$

where $\lfloor \cdot \rfloor$ is an operator for rounding down a real number to an integer value.

2. *Set*

$$y_{j \ldots j+c-1} = x_i$$

and

$$j = j + c.$$

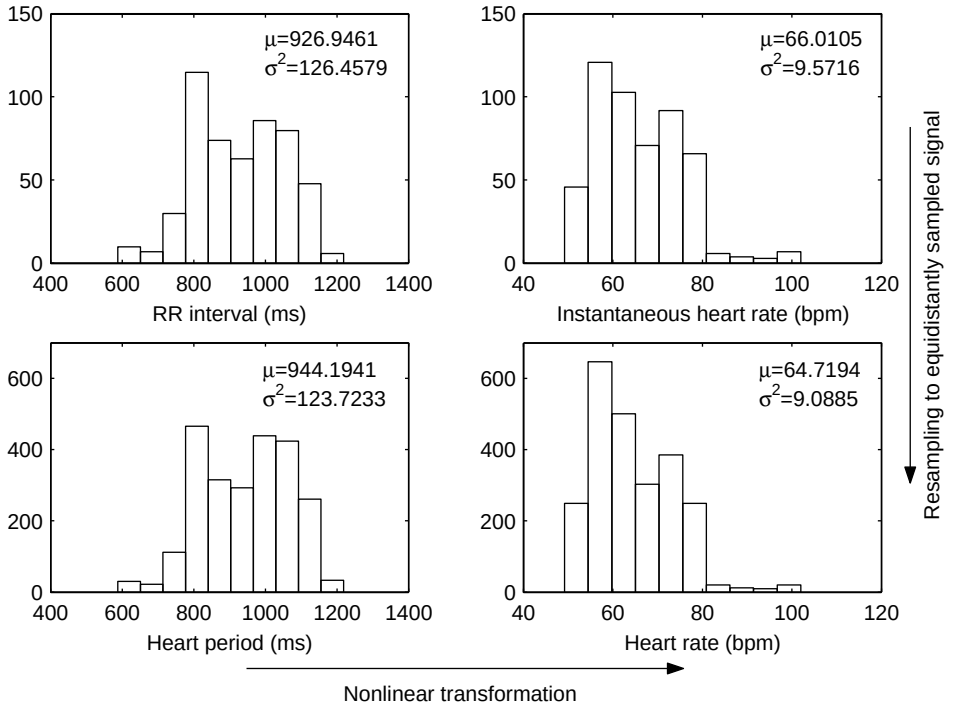

Figure 7: Histograms presenting an eight minute RRI recording refined to three different signals. The nonlinear transformation and resampling both affect the statistics (mean μ and variation σ^2) of the series.

3. *If i is less than n, then calculate the beats left over:*

$$r_1 = x_i - r_2 - \triangle T \cdot c.$$

Then reserve the beats from x_{i+1} to fill in one full interval:

$$r_2 = \triangle T - r_1.$$

Finally, calculate the transition beat y_j between the two beats x_i and x_{i+1}:

$$y_j = \frac{x_i \cdot r_1 + x_{i+1} \cdot r_2}{\triangle T}.$$

4. *If i equals n, then the calculation is ready. Else increase the indices i and j*

$$i = i + 1,\ j = j + 1$$

and return to step 1.

Algorithm 2.2 *Desampling of the time series.*

0. *Let x present equidistantly sampled time series input vector, y the output vector of RR intervals and $\triangle T$ the sampling interval of the time series. Set remainder to zero*

$$r = 0.$$

Then set input and output vector indices to one

$$i = j = 1.$$

The maximum length of the output vector and length of the input vector are n.

1. *Set current output value to current input*

$$y_j = x_i.$$

Calculate

$$c = \frac{x_i - r}{\triangle T}.$$

If the beat is evenly divisible, i.e., c has no remainder, then set

$$i = i + c,\ r = 0.$$

Else set

$$r = \triangle T - (x_i - r - \lfloor c \rfloor) \cdot \triangle T)$$

and

$$i = i + \lfloor c \rfloor) + 1,$$

to reserve time from the next beat.

2. *If i is greater than n, then all the beats are processed and the calculation is ready. Else increase the index j*

$$j = j + 1$$

and repeat from the first step.

The terminology is sometimes used loosely in HRV-related publications. For example, heart rate variability is often used to also express RR variability and instantaneous heart rate variability [120]. This may become problematic, as will be demonstrated with the following example: A statistic that is greatly affected by the used signal is the *square root of the mean of the sum of the squares of differences*[4] (RMSSD) expressed with the following formula:

$$RMSSD = \frac{1}{N-1}\sqrt{\sum_{k=1}^{N-1} (x(k) - x(k+1))^2}, \tag{2}$$

[4]In heart rate variability analysis, RMSSD is a time domain estimate of the short-term components of HRV [120].

where $x(k)$ is an N-length time series. Basically, RMSSD may be calculated for both heart period time series or RR intervals, but the interpretation is not the same. If, for example, Algorithm 2.1 is used in resampling, then the scale of the result is diminished in long RR intervals because of the zero differences, while short RR intervals are less affected. The number of zero differences increases as a function of the sampling frequency. Thus, in such a case the RMSSD of a heart period time series results in an index that has little value for the analysis.

2.4 Heart rate time series artifacts

Heart rate time series artifacts are caused by several sources. They are common, and often characteristic, for healthy and clinical subjects, in both laboratory and field monitoring, from sleep to sports. In the measurement environment, magnetic, electric, and RF noise may disturb the device, especially heart rate monitors. Furthermore, the contact difficulties of electroids, such as the lack of moisture, a problem in the measurement equipment, or spikes produced by body movements may trigger errors.

Also internal "artifacts" exist that are initiated by the body. These arrhythmias are not actual artifacts in the technical sense but look peculiar, alter computations, and are thus treated as artifacts. Different instantaneous arrhythmias are normal also for healthy subjects and could be considered characteristic for ECG and the heart rate time series. Arrhythmias like tachycardia and bradycardia are pathological and may cause *extra* (EB) or *missing beats* (MB) in the corresponding RR intervals [113]. Missing beats originate from unrecognized QRS-waves in the ECG, while extra beats originate from false detection of QRS-waves resulting in the splitting of the corresponding RRI into several. Measurement and triggering errors may originate from false detection of QRS-waves caused by a concurrence of amplitude modulation and respiratory movement, large T-wave related to QRS-wave, bad electrode contact, or spikes produced by body movements [136].

Computer automated correction of the heart rate signal artifacts are discouraged, and manual editing should be performed instead [120]. However, the combination of manual editing and computer aided detection may be feasible with large datasets [113].

Artifact detection procedures are often based on thresholds, such as beats exceeding or falling below twice the mean RRI in a predefined window. Also thresholds based on windowed standard deviation or the difference between successive RR intervals are used. Another perspective is to use a model to fit the time series and predict the following beats. Yet another threshold is utilized to define appropriate differences between the estimates and target values.

The seriousness and amount of corrupted data must be considered when editing the data, and the number of corrected beats is advised to report in connection with the analysis. The correction procedures and rules are combinations of adding the extra beat to neighbouring beats or splitting the artifact beats. Miss-

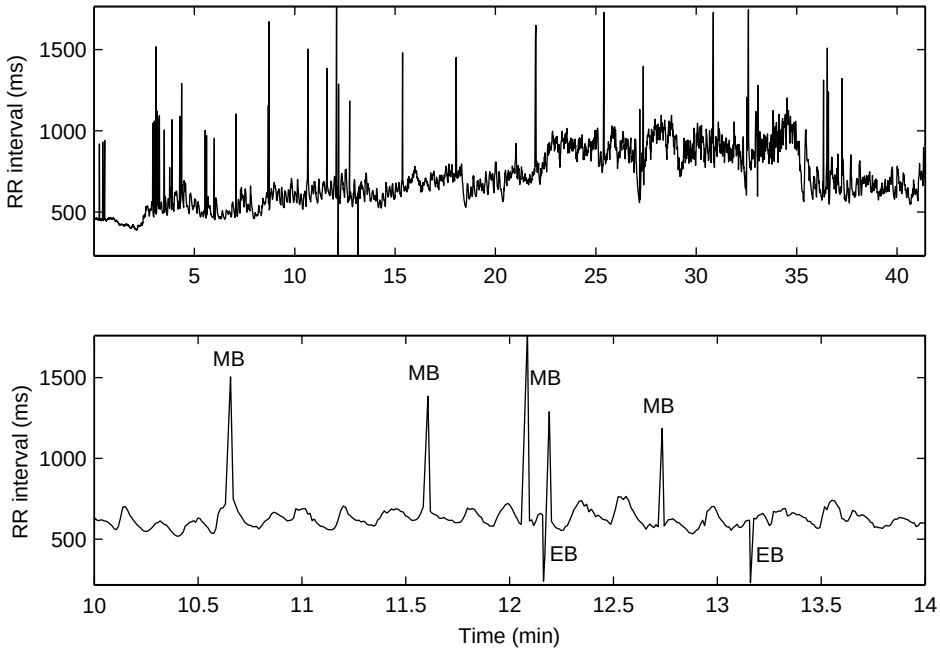

Figure 8: Upper figure presents a sequence of RR intervals recorded with a heart rate monitor containing measurement errors. The lower figure presents part of the series with missing- and extra beats marked.

ing beats are evenly split, meaning if the mean level of the RRI sequence is 2000 milliseconds, a 6000 ms artifact is split three times. Also noise may be added to create artificial variation in the corrected sequence. However, the total time of the series should stay unchanged. If the beat is not integer divisible, then it may have adjacent artifact beats and they have to be added before division. The beat may also be caused by a transient arrhythmia such as bradycardia.

It should be noted that missing beats may never be accurately corrected, since the exact time instant is lost forever. However, when the extra beats are added to the neighbouring beat, it results in a correct reparation if and only if the neighbour is chosen properly and the neighbor is not an artifact itself.

The impact of artifacts on heart rate variability estimates is severe for both frequency and time domain analysis [8, 113]. The correction procedures are not able to restore the true beat-to-beat variation but the influence on variability estimates is less dramatic when considering occasional corrected artifact beats. Highly corrupted sections of data are advised to be left out of the analysis.

Heart rate monitors may produce a large number of artifacts during exercise because, for example, of body movements. This is illustrated in Figure 8. Some monitors record RR intervals up to 30000 beats and construct heart rate variability measures to estimate maximal oxygen uptake or relaxation [172]. However, a

more common measure is the heart rate level used to produce estimates such as energy usage or to guide exercise intensity. Hence, the correction error does not cause a significant problem in these applications, since it mainly affects the beat-to-beat variation.

In this dissertation the heart rate time series are corrected by an expert physiologist. Different detection and correction heuristics and rules, as well as types of artifacts and the influence of artifacts on heart rate variability estimates, are considered by several authors, e.g., Berntson, Quigley, Jang, and Boysen [7], Berntson and Stonewell [8], Mulder [113], Porges and Byrne [136].

2.5 Respiratory sinus arrhythmia

The human body contains multiple cyclic processes such as the monthly menstrual cycle caused by a females sex hormones; daily cycles including body temperature, hormonal cycles (cortisol, testosterone), sleeping rhythm, hemoglobin quantity, acid base balance of blood and urine; and weekly cycles, like the fluid balance. Even ones's height has a daily variation caused by compression of the intervertebral disks.

In the cardiovascular system, the short-time fluctuation of blood pressure and heart rate are connected to respiratory sinus arrhythmia (RSA). In normal, healthy subjects, inhalation increases the heart rate and decreases blood pressure. In expiration, the heart rate decreases and blood pressure increases.

The sinusoidal breathing oscillations in heart rate are apparent in Figure 9, illustrating a metronome-spaced breathing test. The test starts with one minute of spaced breathing at a frequency of 0.5 Hz. Then the breathing rate is stepped down by 0.1 Hz every minute until it reaches 0.1 Hz. After this, the procedure is reversed to the starting frequency. The total test time is nine minutes. Each new step is indicated by a computer-generated sound.

Eight distinct measures were recorded during the test: skin conductivity, RR intervals, systolic and diastolic blood pressure, electromyogram presenting muscle activity from both the biceps and the triceps, respiration using a spirometer and respiration from the chest expansion. The systolic- and diastolic blood pressure time series are presented in Figure 10 where both the low- and high-frequency breathing patterns are distinctive. Blood pressure is usually recorded in three different beat-to-beat series: systolic- and diastolic blood pressure (the maximum and minimum blood pressure during each beat). Also the mean arterial pressure (true mean pressure between two successive diastolic time instants) may be stored.

Respiration rate and volume is known to influence RSA regardless of parasympathetic activation [6, 146]. Furthermore, Kollai and Mizsei conclude that the amplitude of RSA does not necessarily reflect the proportional changes in

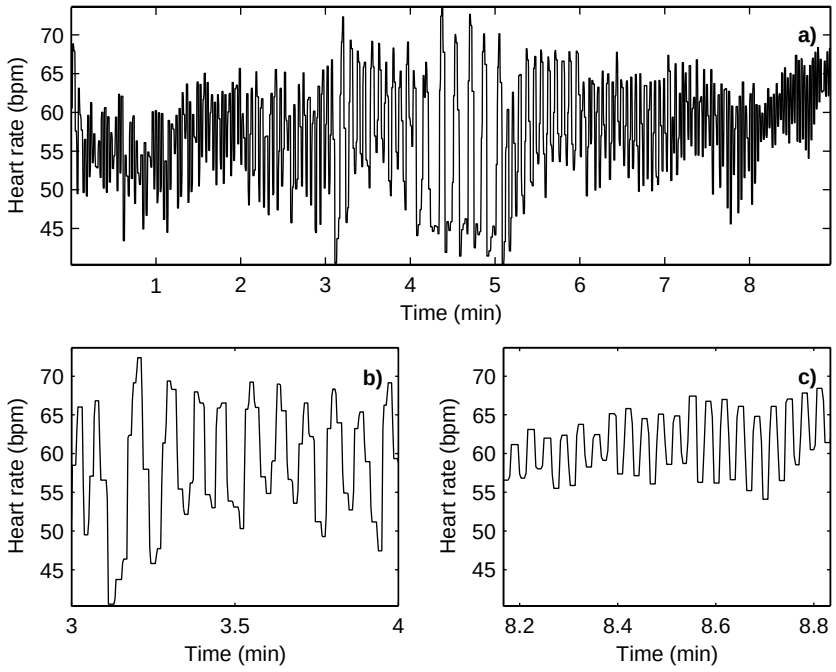

Figure 9: Figure a) presents a spaced breathing test heart rate time series. Figures b) and c) are snapshots of the test in 0.2 and 0.5 Hz breathing rhythm, respectively. Notice the decrease of heart rate amplitude as a function of breathing frequency especially in figures b) and c) while the mean level of the heart rate between the oscillation remains rather stable.

parasympathetic control[5] [84].

Figure 9 proposes that the heart rate amplitude decreases as a function of the breathing frequency. In addition, inter-individual clinical studies has demonstrated reduced RSA with cardiac disease, hypertension, anxiety and depression. Intra-individual research have demonstrated reduced RSA in physiological stress and physical exercise and increased RSA with psychological relaxation [59].

Similar experiments as the breathing test have been studied, e.g., to understand the influence of respiration on heart rate and blood pressure [117] and to examine the effects of paced respiration on the heart rate and heart rate variability [164].

[5]Historical remark: Katona and Jih claimed respiratory sinus arrhythmia as a noninvasive measure of parasympathetic cardiac control. The conclusions were based on a study of anaesthetized dogs [72]. The generalization to human subjects was later questioned by Kollai and Mizsei [84].

Interpretive caveats of the RSA

As demonstrated, the respiratory component of the RSA is visible in steady conditions, e.g., during metronome-spaced breathing. However, the relationship between the RSA frequency and respiratory period may be inflated by several known and unknown sources of naturally occurring nonstationarities and inconsistencies in the cardiac activity and respiratory patterns. In patent by Kettunen and Saalasti [77], a list of challenges in the interpretation is given as follows:

> Even the breathing oscillation may stay at relatively fixed levels during stable conditions, such as rest or different phases of sleep, fast changes are typical in the rate of respiration rate and may unfold, within a single breathing cycle, a substantial change in the adjacent periods. Thus, the respiratory period may show a three-fold increase from 3 seconds to 9 seconds within single respiratory cycle.
>
> It is generally known that several incidents that evoke naturally during non-controlled measurement, such as movement and postural change, speech, physical exercise, stress and sleep apnea may produce significant alterations in the respiratory patterns.
>
> The respiratory pattern of HRV may be overshadowed by phasic accelerative and decelerative heart period responses to both physical and mental incidents, such as postural change, motor control, cognitive stimulation, and emotional arousal. These incidents are frequent, unpredictable from a physiological point of view, may have great amplitude and are often located in the frequency bandwidth of respiratory control.
>
> low-frequency component of the HR, reflecting the HR and blood pressure rhythms is often dominant in the HR. This pattern is most visible in the centre frequency of about 0.1 Hz, but is often considerably broader from 0.04 to 0.15 Hz. The broader bandwidth allows the 0.1 hertz rhythm to overlap with the RSA component, when respiration rate is lower than about 10 breaths per minute.
>
> The amplitude of both the RSA and 0.1 hertz rhythms are sensitive to changes in overall physiological state. For example, when compared to resting conditions, the RSA amplitude may show almost complete disappearance during maximal exercise and certain clinical conditions.
>
> The amplitude of the respiratory period coupled heart period oscillations is modulated by the respiratory period. Accordingly, the amplitude of the RSA increases towards lower frequencies (< 0.20 Hz). Furthermore, the respiratory coupled rhythm is not often exactly sinusoidal but may be composed of several periodic components at different phases of the respiratory cycle.

These characteristics of the HR impose several difficulties in the interpretation of the HR and HRV data. The detailed description of these difficulties forms the basis

and motivation for the application presented in Section 6.3. In the application, the detection of respiratory frequency strictly from the heart rate time series is demonstrated. In addition, the discussion is important to emphasize the affect of the oscillatory components characteristic on the heart rate.

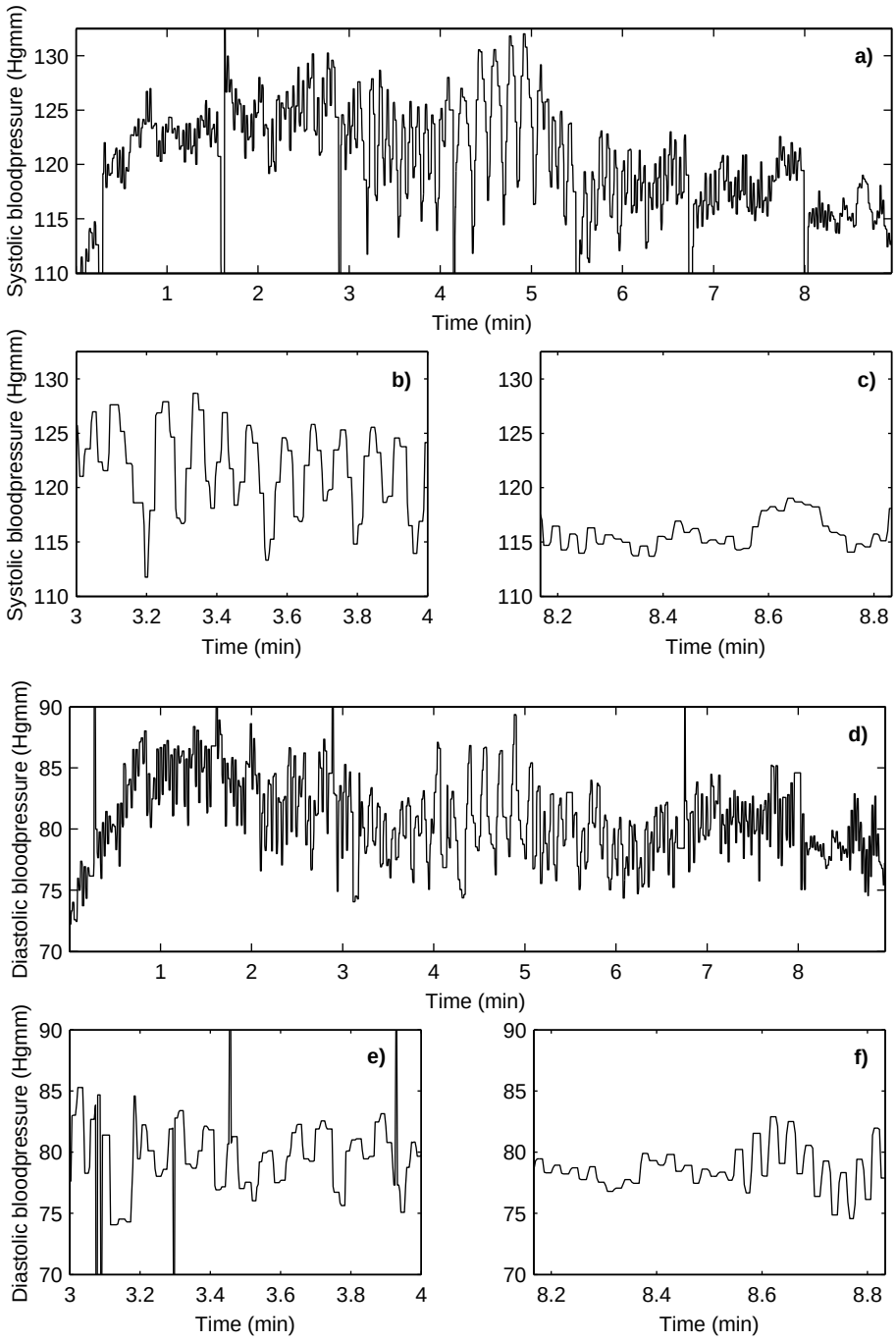

Figure 10: Figures a) and d) presents systolic and diastolic blood pressure time series of a metronome-spaced breathing test. Figures b) and e) present the systolic and diastolic blood pressure time series with spaced breathing of 0.2 Hz and c) and f) 0.5 Hz breathing rhythm.

2.6 Heart rate dynamics

Heart rate is a complex product of several physiological mechanisms. That poses a challenge to a valid interpretation of the HR. This is especially the case within the ambulatory measurement.

Effect of an extreme mental response and stress to the heart rate

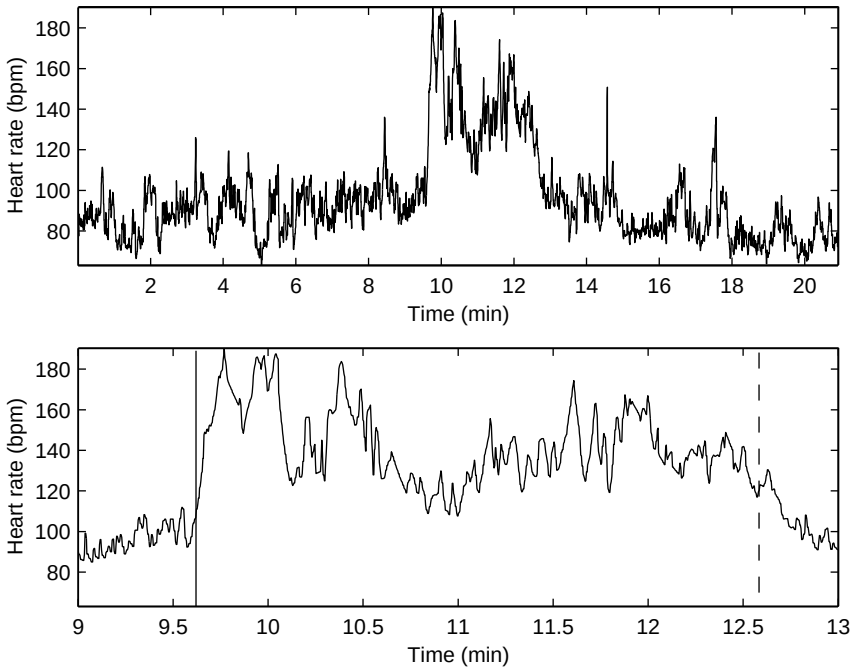

Figure 11: An abrupt heart rate level increase due to anxiety and excitement in a stressful performance. The upper figure presents the entire time series and lower figure the time series during the speech. The beginning of the speech is pointed with a vertical solid line and the end with a dashed line.

In Figure 11, a heart rate time series of an individual performing to an audience is presented. The sudden burst in the heart rate level occurs within seconds after the subject stands up to move in front of the audience. During the presentation, the heart rate starts to decrease as the excitement moderates. The nervousness before the speech is shown in an increased resting heart rate, as the normal mean resting heart rate of the subject is around fifty beats per minute. The figure suggests that, after the presentation, the heart rate level continues to decrease until a relaxed state is achieved.

The example illustrates how emotions and stress may have an instant effect on the heart rate. The recovery from stress may be moderately rapid, but con-

tinuous stress may also appear as a long time alteration to heart rate level and variability.

Effect of the exercise to the heart rate

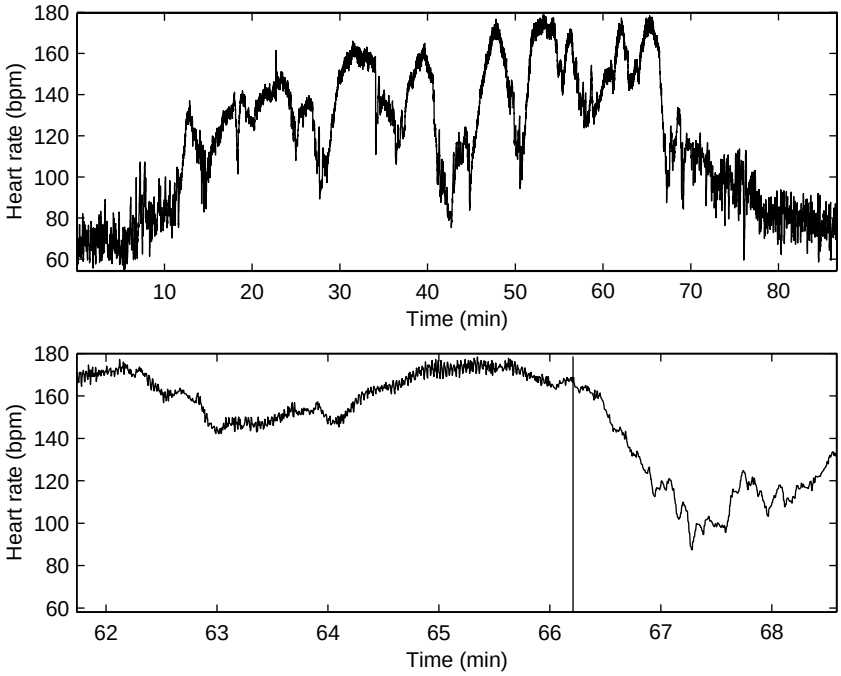

Figure 12: Heart rate time series with a base line, sixty minute roller skating exercise and recovery. The upper figure presents the entire exercise and the lower figure closer inspection at the end of exercise. The time moment indicating the end of the exercise is indicated with a vertical solid line.

The characteristics and properties of the heart rate change considerably if the heart rates of a resting or exercising individual are compared. This appears in the temporal dynamics and characteristics of the signal. For example, an acceleration of the heart rate from a resting level to an individual's maximal heart rate may be relatively rapid as a maximal exercise response. However, the recovery from the maximum heart rate level back to the resting level is not as instantaneous and may take hours, or even days after heavy exercise, e.g., a marathon race. After intense exercise, the body remains in a metabolic state to remove carbon dioxide and body lactates; this process accelerates the cardiovascular system. Furthermore, the body has to recovery from the oxygen deficit induced by the exercise.

Figure 12 illustrates a 60-minute roller skating exercise with the immediate recovery presented in a separate graph. The figure demonstrates how the resting level is not achieved during the recorded recovery time. The illustrated exercise is

an example of fitness training with a relatively steady exercise intensity. A more rapid increase in the heart rate may be achieved with more intense sports, e.g., 400 meter running.

Inter- and intra-individual variation of the heart rate

Characteristics of heart rate time series are heavily influenced by inter- and intra-individual variation. Macro-level intra-individual heart rate characteristic fluctuation is illustrated in Figure 13. The scatter plot of a 28-hour heart rate recording shows the variation of the two measures during different activities. The difference between two successive RR intervals decreases as the heart rate increases. During sleep the difference is at its highest. In a micro-level, heart rate fluctuations appear, for instance, by body movements, position changes, temperature alterations (vasodilator theory, see [45, p. 232]), pain or mental responses (as shown in Figure 11).

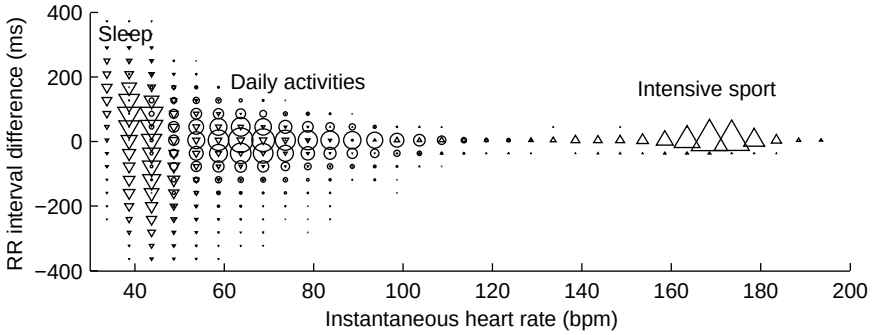

Figure 13: Variation of two variables expressed as a scatter plot between instantaneous heart rate and RRI difference. The dataset is based on a 28-hour RRI recording of an individual. The plot has three dimensions, as occurrence of the observations in predefined accuracy is visualized with the marker size. The x- and y-axis resolutions are five beats per minute and fifty milliseconds.

Inter-individual variation in a heart rate time series is illustrated in Figure 14. The time series presents four sitting-to-standing tests of different individuals after morning awakening. The heart rate, its variation, recovery from position change and standing responses differ among the individuals. In Section 2.5, several alterations to the heart rate RSA component were discussed. Furthermore, the individual's age, gender, mental stress, vitality and fitness are reported to affect the heart rate variability.

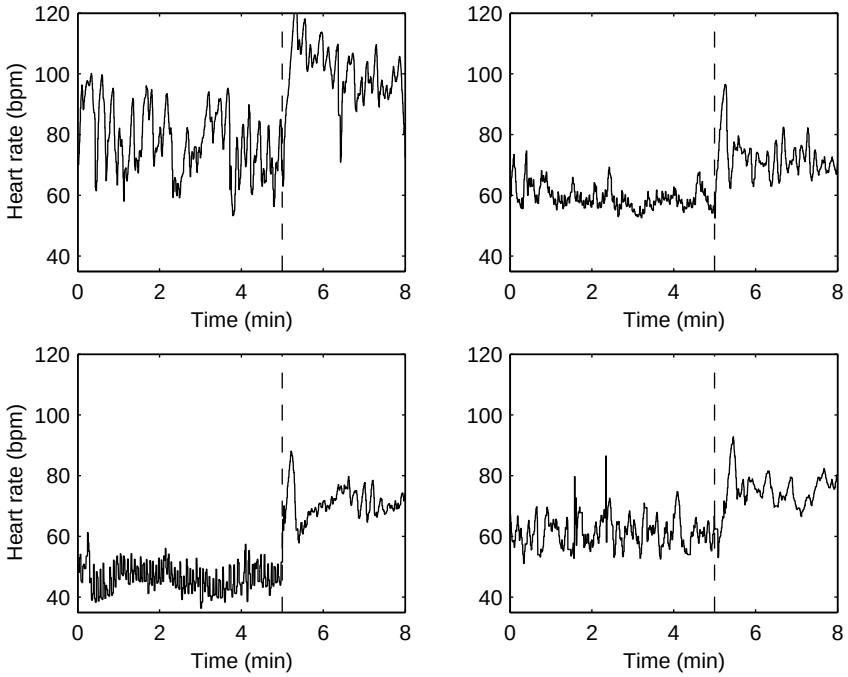

Figure 14: Sitting-to-standing tests of four individuals after morning awakening. The dashed line indicates the moment when the alarm signal requests stand up.

3 TIME SERIES ANALYSIS

This dissertation concentrates on applying neural networks for physiological signals and, especially, for heart rate time series analysis. Although applications vary, the general modeling process for physiological signals is illustrated in Figure 15.

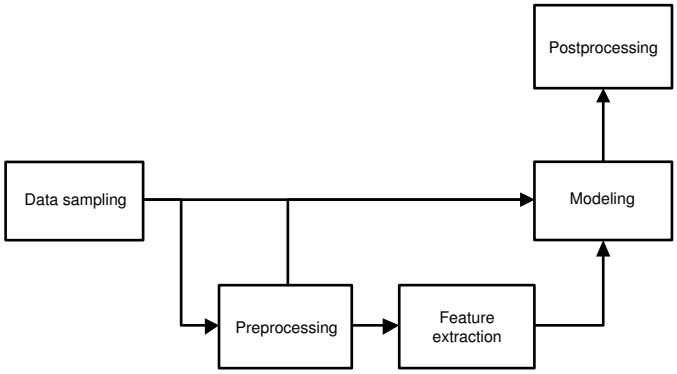

Figure 15: A common physiological time series modeling process.

Four steps are presented in the figure: data sampling, preprocessing, feature extraction and modeling. Sampling is executed by a device recording the physiological time series; we may only obtain a discrete presentation of the human physiology with predefined sampling accuracy. To choose an appropriate sampling the *Shannon sampling theorem* has to be taken into account [23]. It states that the sampling rate has to be at least twice the frequency of the highest frequency component in the signal, if we wish to recover the signal exactly. The *Nyquist frequency* f_N is the highest frequency of which we can get meaningful information from the data:

$$f_N = \frac{1}{2\triangle t},$$

where $\triangle t$ is the equal interval between the observations in seconds. If the sampling frequency is not high enough, then the frequencies above the Nyquist frequency will be reflected and added to the frequency band between 0 and f_N hertz. This phenomena is known as *aliasing* or *folding* [23, 159, 165].

The data may be used directly to construct a model of it. However, in more complicated applications the data is preprocessed with methods capable of, for instance, denoising, detrending, filtering or segmenting the signal. The process may also include feature extraction, creating, for example, a new set of time series including the signal characteristics in separate signals. The modeling step may include linear or nonlinear models or hybrid models, the combination of several models. Furthermore, various postprocessing steps may be executed on the model estimate, e.g., time domain corrections through smoothing or interpolation (missing data).

These are the basic steps in a methodological point of view. In addition, the expert knowledge of both psychophysiology and mathematical expertise are required for the model generation. One possibility for the model generation and, finally, validation of the model results is to visualize the outcome with several perspectives and various empirical data. The visual inspection will also ease the communication between the experts in different fields.

Neural networks are not isolated from classical linear and nonlinear methods. In this section, some classical methods for time series analysis are introduced, including improvements and new aspects for existing data preprocessing and modeling procedures, e.g., time series segmentation, data-ranking, detrending, time-frequency and time-scale distributions, and geometric modeling. Some linear and nonlinear techniques for time series analysis and numerical methods, including standard digital signal processing procedures, are also reviewed.

3.1 Linear and nonlinear time series analysis

This section briefly reviews the classical linear models and their dual counterparts in the frequency domain, i.e., autocorrelation, autoregressive and moving average models versus spectral density estimation. The underlying dependency (or connection) between the models is called *time-frequency dualism*.

The weakness of classic linear models lies in their restrictive assumptions, especially the stationarity assumptions of the signal. The strength is the comprehensive theoretical understanding of linear systems. Regardless of the restrictions, linear time series analysis is widely used even in complex model reconstruction. Furthermore, linear models may be applied and modified to describe nonlinearity and unstationarity, e.g., piecewise linear models in the time domain or short-time Fourier transformation in the frequency domain. The latter is an example of a time-frequency distribution that will be introduced in Section 3.1.2.

Time-frequency distributions may be utilized for the decomposition of a signal to its temporal frequency and power contents. Also time-scale presentations, e.g., Wavelet-transformation, can be used to exclude temporal frequency contents of a signal. A time domain algorithm is illustrated in Section 3.1.9 for the estimation of frequency and power moments.

System or signal modeling is generally performed by means of some quantitative measure. We wish to estimate the goodness of fit for the given empirical model. Thus, different error functions are reviewed in Section 3.1.4.

3.1.1 Spectral analysis

Spectral analysis is used to explore the periodic nature of a signal. In classic spectral analysis the signal is supposed to be stationary. Furthermore, parametric spectral methods assume that the signal is produced by some predefined model. Examples of parametric methods are the Yule-Walker method [174, 186] and the

MUSIC, or MUltiple SIgnal Classification method [10, 158] (cited in [165]). The Yule-Walker method assumes that the time series can be described by an autoregressive process (see Section 3.1.7). The MUSIC method assumes that the signal is a composition of a complex sinusoidal model.

Most commonly used nonparametric spectral methods are based on the *discrete-time Fourier transformation* of the signal $x(k)$, defined as

$$X(f) = \sum_{k=-\infty}^{\infty} x(k)e^{-ifk},$$

where f denotes the frequency of interest and e^{-ifk} is a complex term defined as

$$e^{-ifk} = \cos(fk) + i\sin(fk).$$

Power spectral density (PSD) of infinite signal $x(k)$ is defined as

$$S(f) = \lim_{N\to\infty} \mathcal{E}\left[\frac{1}{N}\left|\sum_{k=1}^{N} x(k)e^{-ifk}\right|^2\right],$$

where $\mathcal{E}$ is the *expectation operator*. PSD describes how power distributes as a function of frequency f for the given signal $x(k)$. Independent of the method used, only an estimate of the true PSD of the signal can be obtained from a discrete signal.

An example of a finite time PSD estimate of the signal, called a *periodogram*, is given by the following formula:

$$P(f) = \frac{1}{N}\left|\sum_{k=1}^{N} x(k)e^{-ifk}\right|^2.$$

A periodogram can be enhanced by using various kinds of windowing which leads to different methods, for example the Blackman-Tukey, Bartlett, Welch and Daniell methods. In addition, the definition of PSD may also be based on discrete time Fourier transformation of the covariance sequence of the signal. The corresponding finite time PSD estimate is called a *correlogram* [23, 159, 165].

When periodic components of the signal are found, they can be useful in constructing a model of the signal. With neural networks, they can be used in a similar manner as the autocorrelation function to determine the number of effective inputs required for the network. However, real life signals, such as heart rate time series, are often nonstationary and are compositions of periodic and nonperiodic components.

Power and frequency moments

Characterization, quantification or feature extraction of power spectrum may be executed in several ways depending on the application. Instead of using the

whole spectrum we may compose one or more features to define its frequency and power contents.

A basic PSD feature is the *mode frequency*, the frequency of maximum power, in other words, the frequency where the highest power peak of PSD is:

$$f_{MOD} = \arg\max_{f} S(f). \tag{3}$$

Another commonly used feature is the *mean frequency*, the centre of gravity of the spectrum defined as

$$f_{MEAN} = \frac{\int_{f=0}^{\infty} fS(f)df}{\int_{f=0}^{\infty} S(f)df}. \tag{4}$$

In time-frequency distributions, the mean frequency is also known as *instantaneous frequency* or *centre frequency*.

The *median frequency* divides the PSD into two equal-sized power regions:

$$\int_{f=0}^{f_{MED}} S(f)df = \int_{f_{MED}}^{\infty} S(f)df. \tag{5}$$

Mode, mean and median powers may also be defined in a similar manner to characterize the PSD:

$$p_{MOD} = \max_{f} S(f), \tag{6}$$

$$p_{MEAN} = \lim_{N \longrightarrow \infty} \frac{\sum_{f=0}^{N} S(f)}{N}, \tag{7}$$

$$p_{MED} = \int_{f=0}^{f_{MED}} S(f)df. \tag{8}$$

We have defined the power spectrum features for the full power spectrum but naturally the inspection may also be applied to a partial area, or band, of the spectrum. For example, in a heart rate variability analysis we may also wish to define the mean frequency and power for both low- and high-frequency bands separately.

The mean and median frequency has shown to have a high correlation with empirical data, such as a myoelectric signal, and there is only little reason to select one over another [56]. Perhaps, since the median frequency is more complex to calculate compared to the mean frequency, and as they both give similar empirical results, the latter is more commonly used. However, the median frequency has claimed to be least sensitive to noise [166] (cited in [86, p. 34]). As discussed in Section 2.4, the heart rate time series artifacts affect both the time and frequency domain features considerably. Hence, the proper correction procedures should be applied before utilizing the spectral analysis.

3.1.2 Time-frequency distributions

A straightforward idea to localize the spectral information in time is to use only a windowed part of the data to present the local spectral contents and move this window through the data. As a result we get a *time-frequency distribution* (TFRD), where for each time instant we have a local spectrum [28]. Due to the local nature of the spectrum it will no longer be so affected by the nonstationarities of the signal.

With time-frequency distributions we can follow how the frequency and amplitude contents of the signal change through time (or remain the same for a stationary signal). An easy implementation of this idea is the *short-time Fourier transformation* (STFT) (a.k.a. *Gabor transformation*), defined in the infinite discrete case as

$$STFT(f,k) = \sum_{n=-\infty}^{\infty} x(k+n)h_N(n)e^{-ifn},$$

where $h_N(n)$ is a symmetric, data window with N nonzero samples. The corresponding periodogram reads as

$$P(f,k) = \frac{1}{N}\left|STFT(f,k)\right|^2.$$

Another popular TFRD is the *Smoothed Pseudo Wigner transformation* (SPWV), defined as

$$SPWV(f,k) = \sum_{m=-\infty}^{\infty} g_M(m) \sum_{n=-\infty}^{\infty} h_N(n)x(k+n+m)x^*(k-n+m)e^{-i2fn}.$$

The summation with the window $g_M(m)$ is used to smooth the estimate over time. This is to reduce the *cross-terms* often appearing in the middle of two periodic components, which makes the interpretation difficult. The enhancement of the time resolution, however, leads to the reduction of the frequency resolution and vise versa: A short data window will result in a time-sensitive model but it will also cut off periodic components below the Nyquist frequency.

With a SPWV, the *digital filtering*, for example FIR filtering introduced in Section 3.2.3, may help to reduce the cross-terms. Digital filtering is used to remove frequency components not of interest from the time series. Especially with the heart rate data, where there are two clear high and low-frequency components, such as the RSA- and 0.1 hertz components, digital filtering may alter the quality of the presentation. When the signal is reduced to only one component, the cross-terms are not likely to have as great an effect on the TFRD as when there are more periodic components in the signal.

The advantage of the SPWV over the STFT is that it is two-dimensional, leading to an excellent time resolution. STFT, on the other hand, is a more robust estimate of the local spectra [134, p. 49, 57].

The power and frequency moments for time-frequency distributions are defined for each time instant and, for example, the mean frequency results in an

instantaneous frequency time series. For nonstationary and multi-cyclic component signals, such as heart rate, the mode frequency may become very unstable and oscillate when observed over time (see Korhonen 1997 [86, p. 34]). The instantaneous frequency is often more stable and appears continuous. However, in the presence of multiple cyclic components, it describes the signal frequency contents poorly. One alternative is to use preliminary information of the signal, if any, and build the frequency and power features based on separate frequency bands.

Other time-frequency distributions exist as well, for example the Wavelet transforms (see Section 3.1.3) and parametric methods like the AR block model algorithm. A comprehensive theoretical and historical review of time-frequency distributions is given by Cohen [28].

3.1.3 Time-scale distributions

A different perspective to signal decomposition is given by the *wavelet transformation* [26, 44, 96]. Instead of power, the wavelet transformation is based on *coefficients*, which describe the correlation between the wavelet and the signal at any given time instant. The frequency is replaced with a concept of scale. The scales may be conversed to analogous frequencies.

The basic principle of the wavelet transformation is illustrated in Figure 16. A *mother wavelet* is moved across the signal with different scales to measure the correlation between the wavelet and the signal at each time instant. Different shapes of the wavelet may be used, enabling other than sinusoidal composition of the signal. For each wavelet scale we produce a wavelet coefficient time series

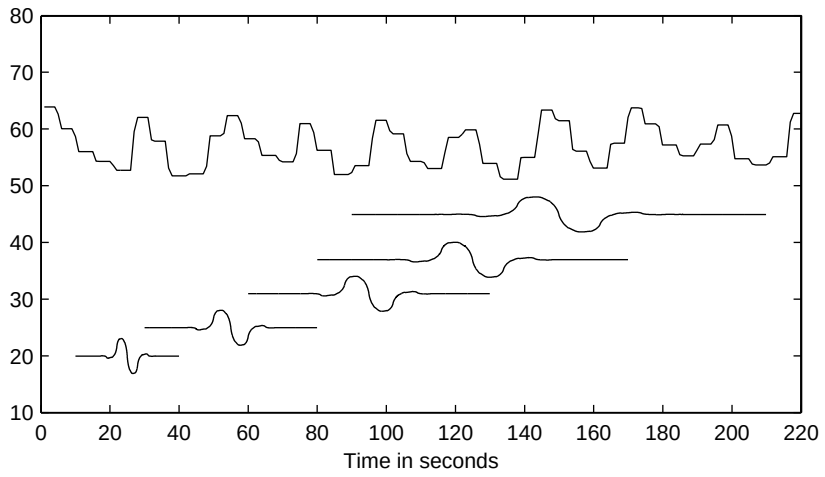

Figure 16: The concept of time-scale representation of a signal: wavelet shapes with different scales are moved across the signal to calculate wavelet coefficients, the correlation between the signal and wavelet shape at a given time instant.

and all together they produce the time-scale distribution of the signal.

The wavelet transformation gives a time-scale distribution of the signal in continuous or discrete form. In time-scale representations the concept of continuous and discrete differ from standard or intuition. The *discrete wavelet transformation* is used for analysis where the wavelet scale is restricted to powers of two. *Continuous wavelet transformation* is applied to a discrete time series but the scale of the wavelet is "continuous", i.e., the accuracy of the scale is unlimited.

In short-time Fourier transformation the time- and frequency resolution are dominated by the window size used. In continuous wavelet transformation both may be set arbitrary. Another important strength is the abandonment of sinusoidal presumption of the signal content. The mother wavelet may have different shapes, improving the power estimation properties in some applications. For example, Pichot et al. [131] claim wavelet transformation to be superior to short-time Fourier transformation in quantitative and statistical separation of the heart rate time series during atropine blocking. The base level before the atropine is compared to progressive atropine doses over time. The STFT power is unable to give statistical difference between the base level and atropine doses. The wavelet coefficients produce notable and quantitative change in heart rate variability.

The family of mother wavelets is already wide and more may be defined. The wavelet shape should contain certain properties to be invertible [104]. One modification to the wavelet transformation would be to use different wavelet shapes for different scales [131].

The use of wavelet transformation to measure power contents and abrupt changes in the system seems promising [131]. However, the scale presentation is more difficult to interpret since the correlation between the wavelet and the signal oscillates. This deficiency is visualized with a simple sine function in Figure 17. The correlation between the wavelet and the signal is close to one in the time instants the two signals cross. In the middle, between the transition from positive to negative correlation of one, the wavelet coefficients go to zero. This oscillation results in a difficult interpretation of the frequency contents of the signal. In particular, the use of mean, mode and median frequencies become unstable.

Other wavelet applications include ECG artifact detection [27, p. 894-904]; detection of discontinuities, breakdown points and long-term evolution (trends); signal or image denoising; image compression; and fast multiplication of large matrices [104].

3.1.4 Error functions

Error functions are used to measure the difference between true and model-generated data. Various error functions are used, depending on the purpose of the analysis. In time series modeling, the error function is often the *objective function*, the function we wish to minimize. Hence, the modeling is based on empirical input data fed into the model and the target data. The model-produced output is compared to the target values and a measure of distance is calculated. This mea-

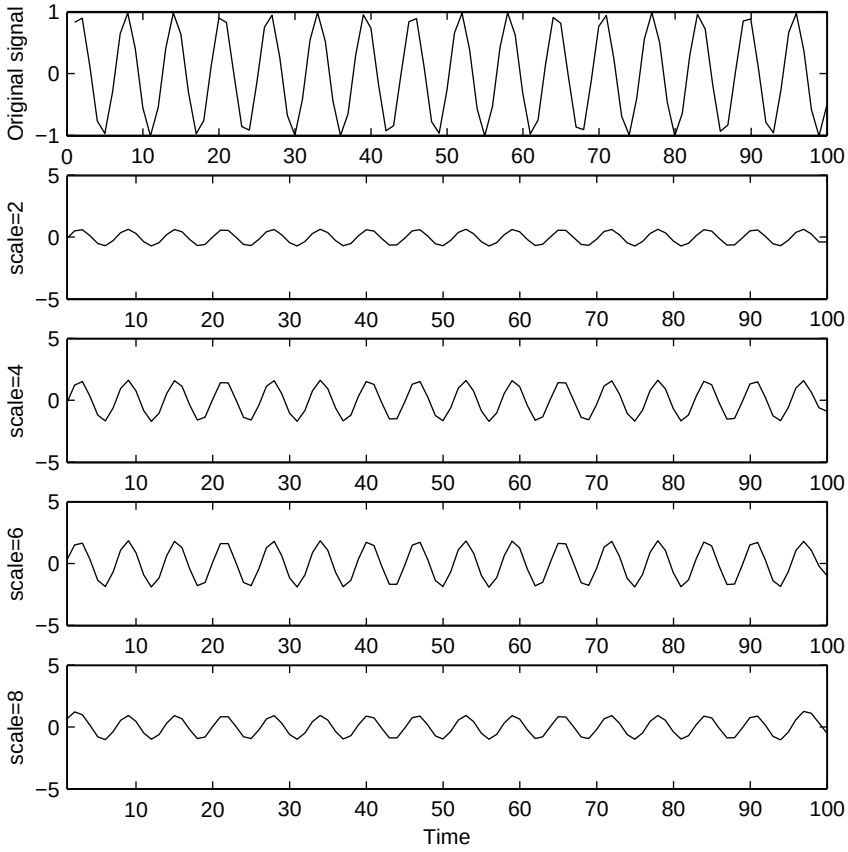

Figure 17: Discrete wavelet transformation of a sine function. The wavelet coefficients oscillate with the signal's sine rhythm.

sure, or error, is used to guide the optimization process to find an optimal set of parameters for the model.

Let $\mathcal{T}$ be a set of N indices for which we want to compare the predicted values $\hat{x}(k)$ and the real observed values $x(k)$. Let $\sigma_{\mathcal{T}}^2$ be the variance of the observed set $\{x(k) : k \in \mathcal{T}\}$. The *sum of squared errors* of the predictors $\hat{x}(k)$ is defined as

$$SSE = \frac{1}{2} \sum_{k \in \mathcal{T}} (x(k) - \hat{x}(k))^2. \tag{9}$$

The SSE measure can be derived from the principle of maximum likelihood on the assumption of a Gaussian distribution of target data (see [13, p. 195-198]). The *mean-squared error*, MSE, is defined as $2\text{SSE}/N$.

Normalized mean-squared error is

$$NMSE = \frac{1}{\sigma_{\mathcal{T}}^2 N} \sum_{k \in \mathcal{T}} (x(k) - \hat{x}(k))^2. \tag{10}$$

If the NMSE equals one, the error corresponds to predicting the average value of the set.

If the targets are genuinely positive, we may also define *mean relative error*,

$$MRE = \frac{1}{N} \sum_{k \in \mathcal{T}} \frac{|x(k) - \hat{x}(k)|}{x(k)}. \tag{11}$$

Notice that optimization methods based on derivative information of the objective function require continuous and differentiable functions [116]. Thus, MRE is not suitable in these applications. However, optimization methods do exist, like genetic algorithms, that do not contain these restrictions [94, 107, 124, 173].

MRE results in a different distribution of residuals than, say MSE, as MRE gives a relative error between the target and the estimate rather than absolute error. This may be illustrative when the error of the model in different target space regions is observed. In general, the various functions reveal different aspects of the model error.

If some of the samples should have more weight in the optimization process, a *weighted squared error* may be constructed [13, 87]:

$$WSE = \sum_{k \in \mathcal{T}} w(k) \cdot \left(x(k) - \hat{x}(k)\right)^2, \tag{12}$$

where $w(k)$ is the positive weighting. Weighting may also be chosen in such a way that the total sum of the weights equals one.

3.1.5 Correlation functions

Some authors also prefer to use correlation between the estimate and target data to illustrate model fitness. However, one difference to error functions is that correlation is not used in optimization steps.

Pearson's correlation coefficient is defined as follows:

$$C_P = \frac{\sum_{k \in \mathcal{T}} \left(x(k) - \mu_x\right) \left(\hat{x}(k) - \mu_{\hat{x}}\right)}{\sqrt{\sum_{k \in \mathcal{T}} \left(x(k) - \mu_x\right)^2 \cdot \sum_{k \in \mathcal{T}} \left(\hat{x}(k) - \mu_{\hat{x}}\right)^2}}, \tag{13}$$

where μ_x and $\mu_{\hat{x}}$ are the respective sample means for estimate x and target $\hat{x}$ in the defined set $k \in \mathcal{T}$.

Spearman's rank correlation coefficient is defined as

$$C_S = 1 - \frac{12 \cdot SSE}{N^3 - N}, \tag{14}$$

where the sum of squared errors is calculated for *ranked* counterparts of $x(k)$ and $\hat{x}(k)$. Ranked data is ordinal data. Real valued data is arranged and labeled, for instance, to integer numbers, depending on their order in the sequence. Data ranking may improve the correlation estimation in the presence of outliers and artifacts. Furthermore, data ranking may elicit a minor variability of the signal in

the presence of greater values. There is also Kendall's rank and biserial correlation coefficients [25, 31].

Notice that the use of a correlation alone to describe *model fitness* is somewhat questionable since, e.g., Pearson correlation between the time series $[1\ 2\ 3\ 4\ 5]$ and $[22\ 23\ 24\ 25\ 26]$ results in one. The analysis of equation (13) also reveals that outliers will mislead the correlation estimation considerably: consider two random signals drawn from uniformly distributed interval $[0\ 1]$ having Pearson correlation close to zero. If some time instance for both signals is replaced with a number large enough, similarly to missing- and extra beats in sequence of RR intervals, the Pearson correlation will go to one.

Since correlation estimates assume stationarity of the time series, a nonstationary signal with varying variance will diminish the correlation effect for time instances having a decreased variance. Thus, the instances in the signal having greater variance will dominate the results, even if they constitute a shorter period of time in the whole signal.

In statistical sciences, a Pearson correlation assumes a normally distributed and large dataset. The significance of the correlation is tested via t-test or ANOVA [185]. If the assumptions are not valid, nonparametric methods, like Spearman's correlation, are evaluated. In this dissertation, the precise statistical analysis is avoided and the statistical indices are merely descriptive. The stationary assumption, especially, is invalid for the presented applications and, thus, statistical indices have to be treated with caution.

3.1.6 Autocorrelation function

Autocorrelation is used to estimate periodic behavior of the time series in the time domain. An alternative and perhaps more illustrative device is the spectral analysis, presented in Section 3.1.1.

By calculating the correlation between the signal and its delayed version, we can study the periodic nature of the signal. For example, an autocorrelation of one at defined delay, or time lag, suggests that the signal includes similar oscillations after the defined time lag.

The *autocovariance* at *lag* $m = 0, 1, \ldots$ of $x(k)$ versus $x(k-m)$ is defined as

$$cov(x(k), x(k-m)) = \mathcal{E}[(x(k) - \mu)(x(k-m) - \mu)]. \tag{15}$$

The corresponding *autocorrelation* at lag m is given by

$$\rho_m = \frac{cov(x(k), x(k-m))}{\sqrt{\mathcal{E}[(x(k) - \mu)^2]\mathcal{E}[(x(k-m) - \mu)^2]}}. \tag{16}$$

If the process is stationary, the variances do not depend on time. This means that the correlation between $x(k)$ and $x(k-m)$ reduces to

$$\rho_m = \frac{cov(x(k), x(k-m))}{\sigma^2},$$

where σ is the standard deviation and σ^2 the variance.

A sample estimation of the autocorrelation function for stationary processes suggested in Box, Jenkins and Reinsel [15, p. 31] is

$$\hat{\rho}_m = \frac{\sum_{k=1}^{N-m}(x(k)-\hat{\mu})(x(k+m)-\hat{\mu})}{\sum_{k=1}^{N}(x(k)-\hat{\mu})^2}.$$

Autocorrelation relies on the stationarity of the time series. For a nonstationary signal, autocorrelation may be comprehended as a measure of average lag (in a cycle or period), main or most distinctive lag, in the signal.

One use for the autocorrelation function is to estimate the number of effective inputs for a neural network. Eric Wan [178, p. 209] used this approach together with single step residuals of the linear autoregressive models (see Section 3.1.7) to estimate the number of inputs of a laser data set for a FIR network, presented in Section 4.2.2.

Notice that autocorrelation may be constructed in a similar manner for the correlation estimates introduced in Section 3.1.5: Spearman's rank, Kendall's rank and biserial correlation coefficients. In addition, the deficiencies discussed in the section are also valid with the autocorrelation estimation.

3.1.7 Linear models

Linear models assume that the time series can be reproduced from a linear relationship between the model parameters. Although linear models are not very powerful, and often not even suitable when forecasting complex time series, they still have some desirable features. Their theory is well investigated and can be understood in great detail. Also implementation of the model is straightforward. This is not the case with more complex models, like neural networks. Linear models can also be used to offer a point of comparison against more sophisticated models. Linear models for time series analysis have been considered, for example, by Box, Jenkins and Reinsel [15] or Chatfield [24].

Autoregressive models

In an *autoregressive model* of order p, or AR(p) model, it is assumed that the future values of the time series are a weighted sum of the past values of the series:

$$x(k) = w_1 x(k-1) + w_2 x(k-2) + \cdots + w_p x(k-p) + \epsilon(k), \tag{17}$$

where $\epsilon(k)$ is an error term assumed to be a white noise process or some controlled input.

One important theoretical result for the AR(p) model is the *stationarity condition*: an AR(p) model is stationary if and only if the roots of the equation

$$1 - w_1 z - w_2 z^2 - \cdots - w_p z^p = 0 \tag{18}$$

lie outside the unit circle in the complex plane [15, p. 55]. Moreover, if the error term vanishes, the output of the model can only go to zero, diverge or oscillate periodically. Take for example an AR(1) model

$$x(k) = wx(k-1).$$

If $|w| < 1$, then $x(k)$ decays to zero. For $|w| > 1$ the value of $x(k)$ grows exponentially without limit.

For autoregressive models the autocorrelations ρ_m can be represented as [15, p. 57]

$$\rho_m = \sum_{i=1}^{p} w_i \rho_{m-i}, \qquad m = 1, \ldots, p, \tag{19}$$

in terms of the parameters w_i. This formula is known as the *Yule-Walker equation*. If the autocorrelations are estimated from the data, the Yule-Walker equations can be used to approximate the unknown parameters w_i.

Moving average models

A *moving average model* of order q, MA(q) model, presupposes that the time series is produced by some external input $e(k)$:

$$x(k) = e(k) + \hat{w}_1 e(k-1) + \cdots + \hat{w}_q e(k-q). \tag{20}$$

The name of the model can be misleading, since the sum of the weight parameters $\hat{w}_i$ is not restricted to unity. If the external inputs are uncorrelated and time independent, the MA(q) models are always stationary [15, p. 70].

Mixed autoregressive-moving average models

A natural step to gain more model flexibility is to join the AR(p) and the MA(q) models together. The result is the *mixed autoregressive-moving average*, or ARMA(p,q) model:

$$\begin{aligned} x(k) = {} & w_1 x(k-1) + w_2 x(k-2) + \cdots + w_p x(k-p) + e(k) + \\ & + \hat{w}_1 e(k-1) + \cdots + \hat{w}_q e(k-q). \end{aligned} \tag{21}$$

Autoregressive integrated moving average models

The *autoregressive integrated moving average models*, ARIMA(p,d,q) models, are an attempt to linearly control nonstationary signals. The model has slightly weaker assumption than the AR(p) model: the dth difference of the model, $\nabla^d x(k) = x(k) - x(k-d)$, is stationary. This leads to a model of the form

$$\begin{aligned} \nabla^d x(k) = {} & w_1 \nabla^d x(k-1) + w_2 \nabla^d x(k-2) + \cdots + w_p \nabla^d x(k-p) + e(k) + \\ & + \hat{w}_1 e(k-1) + \cdots + \hat{w}_q e(k-q). \end{aligned} \tag{22}$$

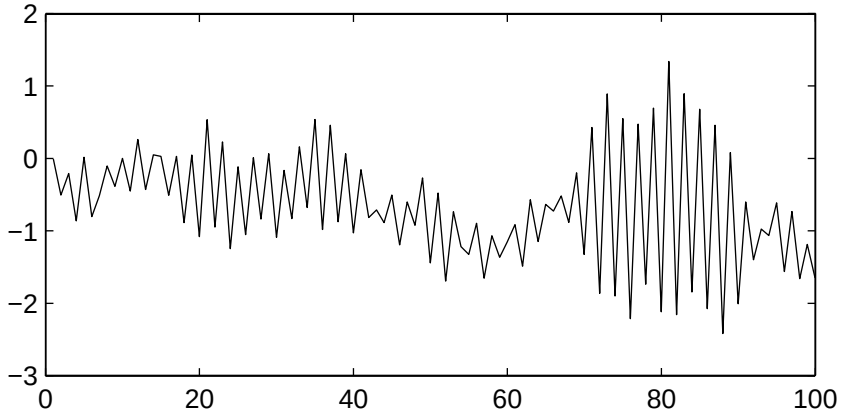

Figure 18: First twenty points of ARIMA(1,2,2) model given by equation (23).

For an example we generated an ARIMA(1,2,2) model with random white noise as an external input $e(k)$. The formula of the model was

$$\begin{aligned} x(k) = x(k-2) + 0.1(x(k-1) - x(k-3)) + \\ + e(k) - 0.5e(k-1) + 0.2e(k-2). \end{aligned} \tag{23}$$

The result is shown in Figure 18. Random noise was drawn from the uniform interval $[-0.5, 0.5]$.

Discussion

One question not answered was how to select the order of the model when we are presented with some data. Some heuristics have been developed but they usually rely heavily on the linearity of the model and on assumptions of the white noise distribution [178, p. 15]. Many of the techniques are variations of the idea that part of the data is retained from the modeling and then used to compare the efficiency of the model by comparing model output and retained data. In such a manner several models with a different number of parameters may be evaluated and compared.

We did not give any procedure to find coefficients for MA(q), ARMA(p,q) or ARIMA(p,d,q) models. There are some standard techniques [15, section 6.3]. Basically these techniques reduce to the solving of a suitable system of linear equations.

Linear models have a good theoretical background and they have been widely used for almost half a century. However, it turns out that if the system from which the data is drawn has a complicated power spectrum, the linear models will fail [178]. A power spectrum contains same information as the ARMA(p,q) model. Thus, if and only if the time series is well characterized by its power spectrum it can be approximated with ARMA(p,q) model. One example of such a

system is the *logistic map*, which is a simple parabola

$$x(k) = wx(k-1)(1-x(k-1)). \tag{24}$$

This system is known to describe many laboratory systems such as hydrodynamic flows and chemical reactions. It is not possible to give any suitable linear fit to a system of this kind [178, p. 16-17].

3.1.8 Nonlinear models

Nonlinear models became recognized and used in practice by the scientific community in the early 1980s. Models like the Volterra series, threshold AR models (TAR) and exponential AR models are restricted to present some particular model structure. This explains why there are so many different nonlinear models in the literature.

If the phenomenon we are observing has a structure that is a special case of the nonlinear model, the model estimates can be very accurate. Lehtokangas [92, p. 58-63] used different kinds of models including the radial basis function network, autoregressive models, threshold AR models and Volterra series, for the estimation of logistic and Henon maps defined in (1) and (24). It appeared, that the Volterra series outperformed other methods. Both maps were possible to be modeled without error, since both maps are special cases of the Volterra series. Notice, however, that in this situation the optimal solution also includes the model structure, i.e., the number of model parameters. Due to the universal approximation theory presented in Section 4.1.1 we may always construct a large two-layered neural network that can repeat the data without an error.

There are some theoretical benefits if we restrict the class of nonlinearity. Often, for example, the model parameters can be optimized with efficient algorithms. However, there are many different kinds of nonlinearity in the world. A neural network can often offer a more flexible and powerful tool for function approximation. Yet, neural networks often require a lot of computer time for determining the unknown parameters. Another deficiency is that the local training algorithms do not always find the optimal solution. In addition, their extensive theory is still to be constructed. Nevertheless, neural networks are interesting and in many cases may provide a suitable model for the observed system.

Nonlinear heart rate models include models for oxygen consumption estimation. The oxygen consumption estimation based on the heart rate level will be presented in Section 6.2. In addition, HRV analysis and research has utilized some nonlinear quantitative measures. The methods include approximate entropy, detrended fluctuation analysis, power-law relationship analysis, the Lyapunov exponent, Haussdorff correlation dimension D and Kolmogorov entropy K, see e.g. [108, p. 20-24]. The development of these methods has its origin in the chaos theory.

3.1.9 Geometric approach in the time domain to estimate frequency and power contents of a signal

An alternative for time-frequency and time-scale distributions to detect the main frequency components in a signal is presented next. The algorithm is based on peak detection in the time domain. The method results in perfect time and frequency resolution. Furthermore, it allows the measurement of reliability of the frequency and power estimates, as will be demonstrated later in Section 3.3. The algorithm is efficient, taking less CPU-time than, for instance, any time-frequency distribution. It may also be applied to on-line applications and embedded systems. A deficiency of the method is that it may not work well with natural signals with multiple cyclic components. The principles of the algorithm are first presented in 3.1 and then further analysis and examples are provided.

Algorithm 3.1 *Down peak detection algorithm.*

1. *Calculate a moving average of the signal, e.g., with a Hanning window.*
2. *Define a maximal frequency (MF) allowed by the algorithm. This specifies a local minimum range.*
3. *Choose all local minimums in the signal which are below the moving average of the signal. These anchor points are called peaks of the signal.*
4. *If there exists two or more anchor points inside a local minimum range, only one is chosen.*
5. *Two adjacent anchor points define one instantaneous frequency of the signal as inverse of the time difference in seconds between the peaks.*

Algorithm 3.1 seeks local minimums, called down peaks, of the sinusoid signal. Detection of local maxima is executed in a similar manner.

When the peaks are detected the instantaneous frequency is formed by calculating the time in seconds between two successive peaks. The frequency between the peaks is the inverse of the time distance in seconds between the peaks. The mean power of a complex time series is defined with the following formula:

$$F(t) = \frac{1}{N} \sum_{k=1}^{N} |f(k)|^2. \tag{25}$$

Thus, the instantaneous power is calculated by applying the equation (25) for each peak interval.

Figure 19 demonstrates the applicability of the algorithm for simulated data. Data has a trend component, a single dynamic sinusoidal component and random noise. The model behind the data is given by:

$$f(t) = 100 + 70 \sin \frac{t}{20} + (100 - t) \left(e(t) + \sin \frac{\pi t^2}{500} \right), \tag{26}$$

where $e(t)$ is random noise drawn from the uniform distribution on $[0, 1]$.

The algorithm relies heavily on the gradient information of the signal. The signal noise or more frequent sinusoidal components with small amplitude may generate adjacent peaks within the main component we wish to observe. This is controlled by limiting the frequency range, by choosing only one local minimum inside the defined region. This procedure, combined with prior knowledge of the signal, may be applied to filter out some of the periodicity introduced by other oscillations or to filter out noise.

The basic algorithm does not give an exact time location, since the frequency is estimated between two adjacent local minimums, thus leading to quarter displacement of instantaneous frequency compared to analytic sinuisoid from zero to 2π. This may be corrected by applying the anchor points between two adjacent up and down peaks. It is also possible to assess the amplitude of a sinus component as the difference between the adjacent local minimum and the local maximum divided by two.

For some signals, a less complicated approach may be applied. Instead of detecting up or down peaks, an intersection between the window-averaged mean and the signal may be used to define anchor points. In this procedure only every second anchor point is labeled and the time difference between adjacent points declare the instantaneous frequency between them. In addition to simplicity, the algorithm estimates the exact time location.

This section outlined the subject of the geometrical approach for estimating instantaneous frequency, amplitude and power. In the application presented in 6.3, the algorithm is utilized to estimate the respiration frequency from chest expansion data.

The algorithm can be further developed by utilizing hybrid models. For example, the wavelet transformation could be applied to peak detection. Furthermore, time-frequency distributions provide average frequency contents in a predefined time range, which could be applied to the selection of the peaks. A selective search among different instantaneous frequencies and their probabilities could be used.

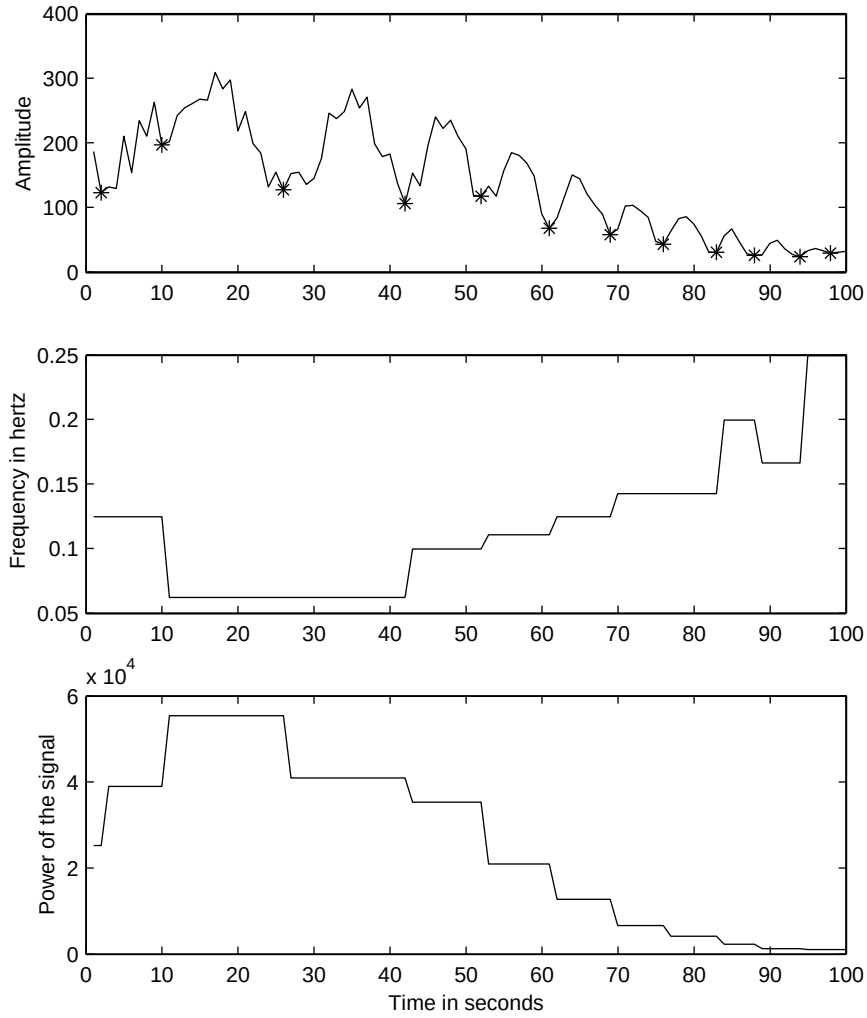

Figure 19: A signal decomposed to its frequency and power components with the peak detection algorithm. The upper figure is the original signal with asterisks in detected lower peaks (anchor points). The middle figure illustrates the instantaneous frequency through time and bottom figure is the corresponding power for each frequency cycle. In this example maximal frequency was set to Nyquist frequency: MF=2.5 Hz.

3.2 Basic preprocessing methods

Neural networks are famous for their fault-tolerance, which means that the phenomenon to be captured is not needed to be described precisely and completely by the measured data. However, better data results in better distribution and improved models.

With neural networks, preprocessing may have great impact on network performance. The simplest case of preprocessing could be a reduction of data if there is redundant information. Also smoothing, e.g., with a moving Hanning window, may improve the signal-to-noise ratio of the data. In general, the use of data preprocessing techniques is application dependent and different methods should be empirically experimented and validated.

3.2.1 Moving averaging of the signal

Smoothing corresponds to moving averaging of the signal with a predefined window shape. Naturally, the optimal window length and shape has to be explored through experimentation. A general smoothing procedure of a discrete time series $x(t)$ for a single time instant t is expressed with the following formula:

$$\hat{x}(t) = \frac{\sum_{n=-k}^{k} x(t) h_{2k+1}(n+k)}{\sum_{n=-k}^{k} h_{2k+1}(n+k)}, \tag{27}$$

where $h(\cdot)$ is the window, such as a Hanning window, of odd length $2k+1$. The window is usually chosen in such a way that the current time instant has a relative weighting of one and the time instants before and after are symmetric and have decreasing weighting as a function of distance to the centre. Typical moving average windows are presented in Figure 20 [99, 165]. For example, an N-point Hanning window is constructed with the following equation:

$$h_N(t) = 0.5 \left(1 - \cos\left(\frac{2\pi t}{t+1}\right)\right),\ t \in \{1, \ldots, N\}. \tag{28}$$

In Kettunen and Keltinkanas-Järvinen [75] smoothing is shown to improve the signal-to-noise ratio of physiological data. This information is suggested to be exploited to enhance the quality of the input signals for the given time series model.

3.2.2 Linear and nonlinear trends and detrending

A loose definition of a trend was given by Chatfield [24]: a trend is a long-term change in the mean level of the time series. When creating a synthetic model of the empirical time series, we may presume the model consists of components

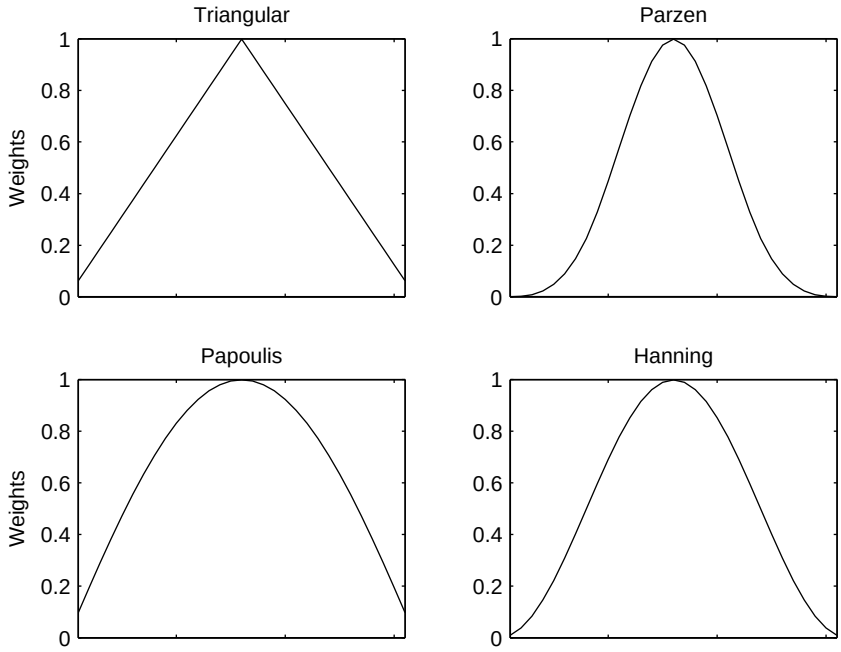

Figure 20: Examples of different moving average windows each having a total length of 31. Different averaging windows are presented, e.g., in [99, 165].

such as cyclic components, level, trend or noise terms. Thus, the trend estimation may be part of the model construction. A trend may also be considered a cyclic component with long cyclic length.

The process of removing trend components not of interest from a time series is called *detrending*. The procedure basically simplifies the signal by removing one or more linear components. Detrending may also improve the time series stationarity conditions, leading to enhanced estimation properties. This also applies to a frequency domain analysis, where detrending may improve the PSD estimate.

Linear detrending may be performed in its simplest form by subtracting a fitted line from the time series. To expand this idea to nonlinear trends, we may use any curve-fitting approach for the trend removal. However, these approaches are not yet practical for a natural time series with many visible trends, such as the time series having several local trends instead of one global trend. For example, in Figure 12 there exists first an increasing nonlinear trend in the heart rate during exercise (first phase) and decreasing nonlinear trend when the subject is recovering from the exercise (second phase).

A linear estimate is too simple, and for the nonlinear models we should know the number of local trends in advance to choose the appropriate model order. A more automated process is to remove local trends with digital filtering, introduced in the next section, which may be used to remove desired low-frequency

components from the time series (so called *high-pass filtering*).

There are also other alternatives to trend removal. A neural network may be constructed for filtering and trend removal with autoassociative learning [173, p. 42-44]. Also a Wavelet transformation may be applied to the trend removal [104]. Smoothing, or moving average methods as well as convolution, for filtering and trend removal, are described by Chatfield [24].

3.2.3 Digital filtering

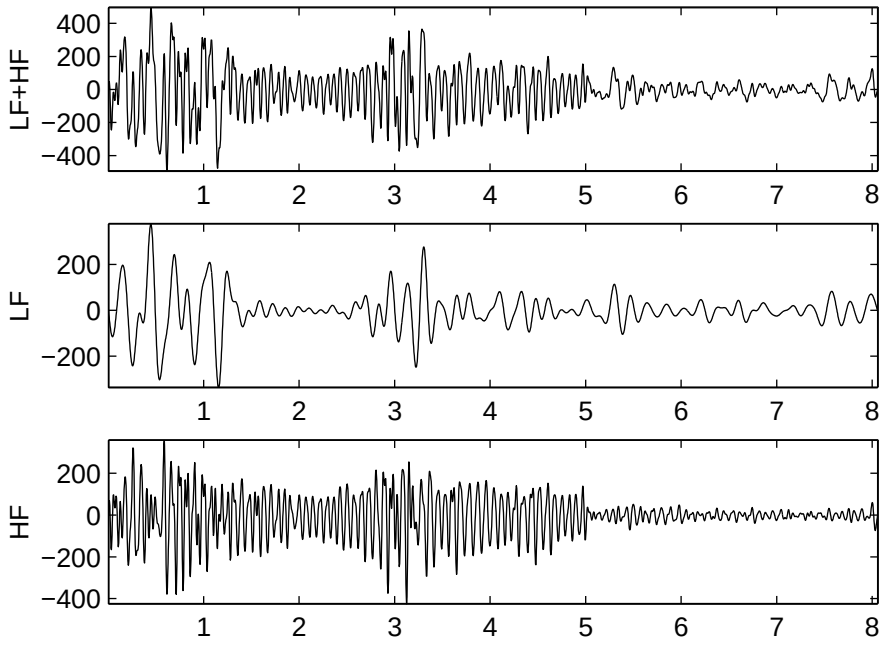

Figure 21: First figure presents the outcome of a 500th order FIR digital band-pass filter for frequencies $0.04 - 0.5$ Hz (LF+HF). The second and third figures present the band-pass filters for frequencies $0.04 - 0.15$ Hz (LF) and $0.15 - 0.5$ Hz (HF), respectively.

Digital filtering is a normal data preprocessing technique used to reject the periodic components not of interest. Examples of digital filtering procedures are infinite impulse response (IIR), and finite impulse response (FIR) filters. They are standard signal processing techniques and are well described, e.g., in [121, 103, 159, 161].

Figure 21 presents the outcome of different filtering procedures applied to the orthostatic test data presented in Figure 14 in the second row left. In the experiment, the five hertz sampled heart period time series was filtered with a 500th order FIR digital band-pass filter to extract the frequency bands between $0.04-0.5$ Hz (both low- and high-frequency components), $0.15 - 0.5$ Hz (high-frequency

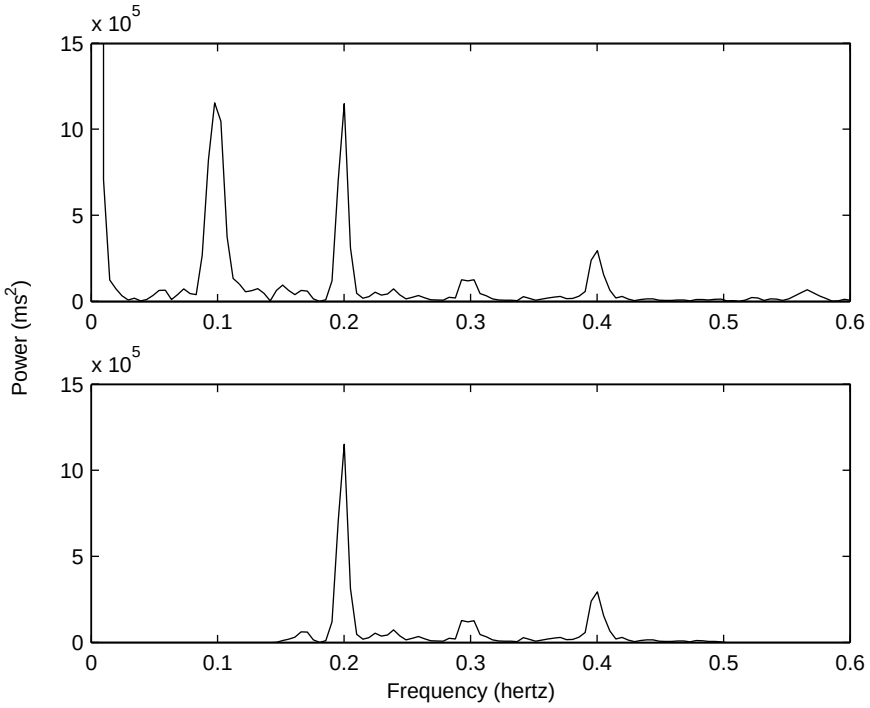

Figure 22: The first figure illustrates the power spectrum of a breathing test data introduced in Section 2.5 lasting a total of 9 minutes. The second figure is the power spectrum of the same data but the data is filtered to a high-frequency band between $0.15 - 0.5$ Hz.

component), and $0.04 - 0.15$ Hz (the low-frequency band). As proposed in Section 3.2.2, digital filtering can be applied for long-term and also short-term trend removal as can be verified from the figures. The *passband* refers to those frequencies that are passed, while the *stopband* contains those frequencies that are blocked. The *transition band* is between them. Furthermore, the *cut-off frequency* is the one dividing the passband and transition band [161].

An important feature of FIR digital filtering is that it may be constructed in a way that it does not change the phase of the signal. It also offers a reliable cut-off between frequencies, as can be seen in Figure 22; the spectral contents within the frequency band seem to remain unchanged compared to the unfiltered data. The accuracy of the frequency cut-off depends on the filter order. With a small number of filter coefficients, the band-pass filtering results in a wide transition band.

To achieve a clear frequency cut-off a high filter order is required, but also enough data points. This is a deficiency in digital filtering if applied to on-line applications. To declare a zero phase filter, three times the filter order of data points is required. Such filter calculus for time t also requires future points, which effects on-line applicability [103].

In practice, the digital filter coefficients are resolved in advance to define proper frequency and power modulation. Thus, only a weighted average through the filter coefficients and the signal is calculated for each time instant.

An alternative for digital filtering is a direct power weighting in the frequency domain. The weighting may be used to elicit certain frequency components by using proportional weighting of the power spectrum. Simple filtering is conducted just to ignore or cut off the power spectrum frequencies not of interest. Furthermore, to bring forth some power components, a pre-knowledge of the signal may be used to construct *adaptive filters* where the filtering (or direct power spectrum weighting) is dynamic and controlled by an algorithm using, perhaps, multi-signal and signal noise information [27, 53]. An application for direct power weighting is later presented in Section 6.3, where a neural network adaptive filter is constructed for breathing frequency detection strictly from the heart rate time series.

3.2.4 Data normalization

If a time series with different statistical properties, such as mean and variance, are analyzed or modeled, the interpretation may be distorted. For example, simultaneous visual interpretation of the signals is difficult if a signal exists that has a considerably higher range than the other signals.

The rescaling can be done with the following formula for each data point:

$$\tilde{x}(k) = \frac{x(k) - \mu}{\sigma} = \frac{1}{\sigma} \cdot x(k) - \frac{\mu}{\sigma} = \alpha \cdot x(k) + \beta, \tag{29}$$

where μ is the sample mean and σ^2 the variance. The latter equality emphasizes the fact that the normalization is only a scaling procedure, meaning it does not differ from transforming the signal to a certain interval. For example, forcing a time series to an interval from zero to one is obtained by choosing

$$\alpha = \frac{1}{\max_k \{x(k)\} - \min_k \{x(k)\}},$$

$$\beta = -\frac{\min_k \{x(k)\}}{\max_k \{x(k)\} - \min_k \{x(k)\}}.$$

Normalization, or scaling, may become problematic with on-line applications and chaotic signals, since the signal characteristics change over time. With signals like a heart rate time series, prior knowledge of the signal may be exploited, for example the minimum and maximum values, to transform the input and output space instead of using only the available data.

3.2.5 Data ranking

A signal-to-noise ratio may be improved for certain applications by ranking the signal, e.g., by sorting the observations to positive integer numbers, allowing the

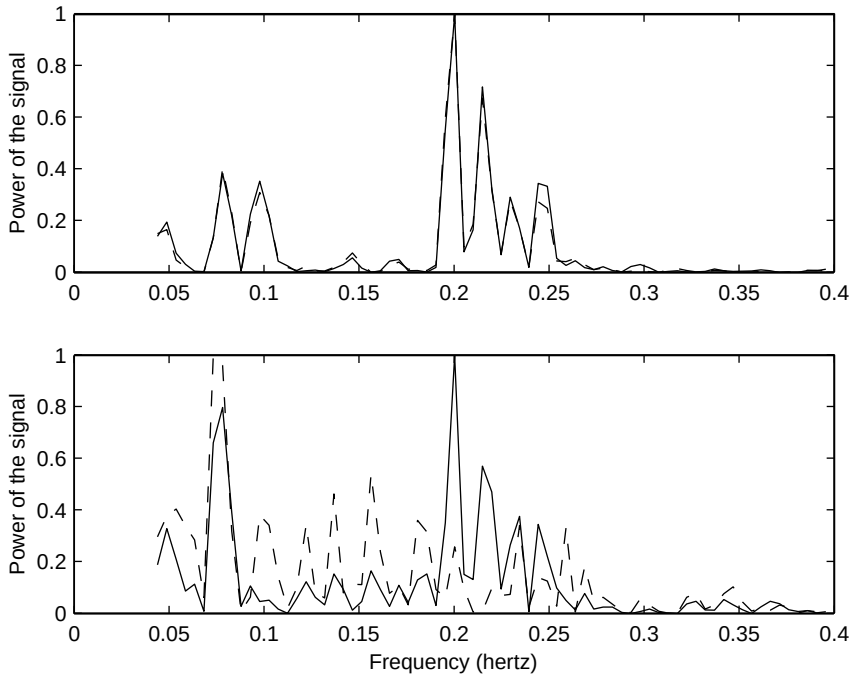

Figure 23: The upper figure presents two normalized power spectrums of a heart period time series with spectrum calculated from original data (dashed) and ranked data (solid line). The bottom figure illustrates power spectrums produced in a similar way but now ten missing beat artifacts are assigned to the time series. The heart period time series is presented in the bottom left of Figure 14.

same occurrences. Algorithm 3.2 presents an example implementation of this approach. Data ranking preserves the signal rhythm and oscillations but it deletes the acceleration of the signal.

Algorithm 3.2 *Data ranking.*

0. *Let x present a vector, e.g., a time series, containing n observations $x(k),\ k = 1,\ \ldots, n$. Vector y will present the output of the algorithm.*

1. *Sort the vector x to increasing order and store the position of each element in the original vector to z. The resulting sorted time series is presented by vector $\overline{x}$. Hence, the following equality will hold:*

$$\overline{x}(k) = x(z(k))$$

for all k.

2. *Set the indices to one,*

$$i = 1,\ j = 1.$$

3. *Set*

$$y(z(i)) = j.$$

4.1 *If* i *equals* n*, then the calculation is ready.*

4.2 *Else if* $\overline{x}(i)$ *equals* $\overline{x}(i+1)$ *then set*

$$i = i + 1$$

and return to step 3.

4.3 *Else set*

$$i = i + 1,\ j = j + 1$$

and return to step 3.

The upper graph in Figure 23 illustrates a normalized power spectrum of original and data-ranked signals. The difference between the signals is insignificant. The interesting result is achieved when ten missing beats are introduced to the heart period time series. As demonstrated by the power spectrums, the data-ranked signal better preserves the original frequency and power contents and is less sensitive to the artifacts.

Data ranking has a great impact in correlation coefficient calculations in presence of artifacts and may be used to improve the estimation. Naturally this applies to the variance and standard deviation of the signal. Modifications based on rank and sign are applied to the correlation coefficients estimation by Möttönen et al. [112, 111]. They also applied the method to the MUSIC algorithm, briefly introduced in Section 3.1.1. However, it seems that the implementation is very expensive to calculate and not practical for larger data processing or embedded systems with inefficient CPU.

3.2.6 Remarks

Artifact correction, digital filtering, detrending, power spectrum weighting, data normalization, as well as other data preprocessing techniques, often improve the data quality in statistical significance tests or other direct quantitative analysis, such as model building and direct error measurement between the model and target output. This may be a result of improved signal-to-noise ratio of the signal. However, they may have a side effect to simplify the signal and observed phenomenon, especially with certain statistical tests, the assumptions of linearity and stationarity may have a great impact in the selection of data preprocessing techniques. After all the data manipulation it is necessary to ask whether the preprocessing is executed only to improve applicability of a mathematical or statistical model rather than understanding the underlying phenomena that the signal represents. The presumptions of the signal nature should drive the analysis, not the techniques.

3.3 Postprocessing

In this dissertation two different signal *postprocessing* approaches are suggested: moving averaging of the model output and interpolation approach (cf. Section 2.3). Postprocessing refers here to a signal processing method applied to the model output or signal estimate. Both methods produce a *time domain correction* of the given model to form enhanced estimates. A time domain correction may utilize the local information of the temporal signal to decide whether the *observed instant appears fit based on its surrounding*. Hence, the presumption is that each time instant is related to its neighbours and should not differ significantly from its close surroundings. Abrupt changes are considered as outliers or artifacts. This presumption is suggested, because a change in the heart rate time series has certain physiological limits, e.g, the acceleration or recovery of the heart is restricted with physiological laws. The applicability of postprocessing is demonstrated later with an application presented in Section 6.3.

To gain local information we must have some objective measure to quantify the reliability of the estimate at given time instances. It will be shown that reliability information of the signal estimate may be produced by some models based on, for example, the model error, distribution of residuals or properties of the input signals. Such models include a generalized regression neural network presented in Section 4.3.2 and the hybrid models introduced in Section 5. However, in this section we have to presume that such information is available for these models and the information may be used to enhance the quality of the model. The time domain correction will also be called a *reliability correction*, as reliability is the main tool to improve the model.

The *reliability estimate* $rb(t)$ is assumed to be a discrete presentation of the reliability of the model output $y(t)$ at a given time instant t. It is scaled in such a way that the higher the value, the higher the reliability of the signal is. Thus, it gives quantified local information of the fit of the model estimate. An example reliability estimate for an instantaneous frequency of the time-frequency and time-scale distributions is presented in Section 3.3.1. Yet another example is the reliability estimate for the peak detection algorithm presented in Section 3.3.2.

3.3.1 Reliability of an instantaneous frequency

Both time-frequency and time-scale distributions are able to elicit instantaneous frequency moments of the signal (see Sections 3.1.1, 3.1.2 and 3.1.3). It appears that the mode frequency may produce fast oscillations of the instantaneous frequency estimate. This will be later demonstrated in Section 6.3. The question is whether these oscillations can be controlled and perhaps reliabilities could be constructed for the given instantaneous frequency estimate.

In this section we outline a concept that is not, to our knowledge, discussed in literature. It is quite a simple observation and is formulated as follows: in

instantaneous frequency estimation *each cycle should last a certain period*, e.g., if the instantaneous frequency gives 0.1 Hz in certain time instant, a stable presentation should have at least ten seconds of the frequency 0.1 Hz in the surrounding time points.

Consider, for example, the Gabor transformation. It produces average spectral contents of the signal defined within the used time window. If a signal has several nonstationary frequency components, then the components with similar amplitude may produce oscillating frequency estimates from one to another and the estimates may not last the required time frame.

We suggest the following error to be calculated for the cyclic length deviation:

$$E(t) = 2f(t) \cdot \min\{ \sum_{k=1}^{\frac{1}{2f(t)}} (f(t) - f(t+k))^2, \sum_{k=1}^{\frac{1}{2f(t)}} (f(t) - f(t-k))^2 \}, \qquad (30)$$

where $E(t)$ is the estimated squared cyclic error of the instantaneous frequency measure $f(t)$ at time instant t. The formula is a heuristic and a compromise to control the uncertaintity where the given frequency component should start. Hence, it is the squared error of the estimate to its neighbours right before and after it, lasting half the cycle length $\frac{1}{2f(t)}$. Analyzing the formula reveals that a frequency component lasting its full length will always result in zero error.

This error information may also be used to construct a reliability measure of the instantaneous frequency. Sudden jumps into frequencies that do not last their respective cyclic length could be considered "artifacts". We presumed that reliability should produce high values for time instants including better reliability. Now the error $E(t)$ produces small values for a more reliable time instant. To override this, we may transform the error $E(t)$ to follow the presumption. An example of a nonlinear transformation function is defined as follows:

$$rb(t) = \exp(-c \cdot E(t)),\ c > 0, \qquad (31)$$

where c is a positive constant. The transformation maps the function $E(t)$ to an interval $(0, 1]$; a small error will now result in high reliability. Zero errors will result in the reliability of one. Notice that the constant c may be optimized for the application. Since the function in (31) is differentiable, the optimal constant c may be optimized, e.g., with the nonlinear optimization methods discussed in Section 4.

An example linear transformation is defined as

$$rb(t) = c - E(t),\ c > 0, \qquad (32)$$

where the constant c could be chosen large enough to keep the reliability positive.

3.3.2 Reliability of the peak detection algorithm

In Section 3.1.9 a geometric approach in the time domain was presented to estimate the frequency and power contents of the signal. If we presume local stability

of the signal, such as similarity of three adjacent cyclic components, we may improve the algorithm by choosing between alternative peaks with a reliability measure. Reliability needs the algorithm modification to detect both up and down peaks (adjacent local minimum and maximum) of the signal. At time moment t_2, the observed down peak's reliability is a measure of the distance and amplitude similarity between the adjacent up peaks labeled to occur at time moments t_1 and t_3:

$$r(t_2) = \frac{\min\{|x(t_1) - x(t_2)|, |x(t_3) - x(t_2)|\}}{\max\{|x(t_1) - x(t_2)|, |x(t_3) - x(t_2)|\}} \tag{33}$$

The reliability should be interpreted as a utilization of amplitude information in the signal. Clear and steady (similar) amplitudes simulate a perfect sine wave. Clearly, the measure in (33) gives full reliability of one to an analytic sinusoid signal.

3.3.3 Moving averaging of the model output

In Section 3.2.1 smoothing was suggested to be exploited to enhance the quality of the input signals for the given time series model. Furthermore, a similar approach can be utilized for the postprocessing of the model estimate to smooth the model output. This may improve the model especially if the model itself may produce reliability information that can be utilized with the smoothing. The resulting smoothing generates a weighting that is relative not only to distance of the centre but also relative to the corresponding reliability of each time instant. The procedure results in the following equation:

$$\hat{y}(t) = \frac{\sum_{n=-K}^{K} h_{2K+1}(n+K) rb(t+n) y(t)}{\sum_{n=-K}^{K} h_{2K+1}(n+K) rb(t+n)}, \tag{34}$$

where $rb(t)$ is the reliability of the estimate $y(t)$ at time instant t.

Smoothing with and without reliability weighting may also be used as an empirical test for whether the produced reliabilities are reasonable. If smoothing without the reliability weighting produces better estimations, then the reliability information is not valid. Notice also that there exists an optimal window length and shape for the given application. It is also suggested that some pre-knowledge, e.g., model inputs, could be exploited to form a dynamic window length having non-constant length.

3.3.4 Interpolation approach

The reliability information may also be used to improve the model estimate by interpolating instants where the reliability falls below a predefined threshold. The threshold is chosen empirically to optimize the model. The threshold may be optimized by any line-search algorithm, e.g., golden section search [154], backtracking algorithm [33], Brent's algorithm [154], hybrid bisection-cubic interpolation algorithm [154] and algorithm described by Charalambous [22] (all cited in [101]).

3.3.5 Remarks

The proposed time domain corrections have a heavy assumption: they both assume that the time instants having poor reliability may be improved by shifting them towards the values of the surrounding time moments with higher reliability. In many time series, such as the heart rate signal, the adjacent time instants are coupled and do not differ substantially. Hence, a correction towards adjacent values having a higher reliability seems reasonable. Naturally, the effect of the heuristic should be empirically evaluated.

Furthermore, the moving average methods will smooth the signal to have a lower variance and instantaneous changes. Hence, we assume that there are some limits for the instantaneous changes of the signal and the changes should be reduced. This leads to an idea that the information on the difference limits itself could be used to evaluate the reliability of the model. As discussed in Section 2.4, the physiological limits can be utilized to detect artifacts and outliers in the heart rate time series.

3.4 Time series segmentation

Fancourt and Principe define the basic signal segmentation problem as follows: "given a single realization of a piecewise ergodic random process, segment the data into continuous stationary regions" [37]. The dictionary of mathematics defines ergodicity as a property of time-depended processes, such as Markov chains (a stochastic process that depends only on the previous state of the system), in which the eventual distribution of states of the system is independent of the initial state [31].

In this concept (heart rate time series analysis), we may conclude that the purpose of the time series segmentation is to segment the data into continuous stationary regions. Furthermore, we may use the segmentation information, such as the beginning and end of each segment together with suitable segment-wise statistics, to detect and analyze changes in time series level, trend or cyclic behavior. In detrending, we could apply a data segmentation routine to improve the curve-fitting approach to divide the data into linear or nonlinear segments and treat each segment with the detrending routine.

Segmentation information may be used for data modeling, e.g., to construct a piecewise nonlinear model of the system. Each segment is reproduced with a different parameter set for a given model or models. For the frequency domain analysis, methods like the Fourier transformation assume stationarity of the time series. Thus, segmentation enables us to use the stationary frequency domain methods to analyze nonstationary data.

Another application for time series segmentation is *state detection* or the classification of the time series. In the state detection procedure, a set of features is calculated for each segment. A feature-vector contains, for example, time series

statistics such as mean and variance. Also frequency domain features like mean power or central frequency may be considered. After feature construction, each data segment is defined and labeled, for example, as a shortest Euclidean distance between the state prototypes and the feature vectors. State prototypes illustrate an "ideal" set of features for each possible state.

The described state detection heuristic is related to signal classification. The combination of signal segmentation and classification has been experimented, e.g., by Kehagias [73]. Kohlmorgen et al. present algorithms to utilize neural networks and time series segmentation to model physiological state transitions (for example wake/sleep, music/silence) with an EEG signal [81, 82, 93, 114]. Furthermore, in articles [62, 80, 79, 127] Hidden Markov Models are exploited to model switching dynamics.

Two different time series segmentation heuristics are presented next, the moving of a PSD template and a generalized likelihood ratio test (GLR). The first is presented for its simplicity, the latter to describe enhancements developed to apply GLR in physiological on-line multivariate time series segmentation. The common factor for the methods is that they are applied, in this dissertation, in the time-frequency domain.

In a system described in [76, 78] the GLR-algorithm is used to segment HR time series to detect the physiological state of a body. The overall system is applied for the daily monitoring of physiological resources. HR is segmented, based on the HR level and time-frequency distribution of the signal. Different statistical features are calculated for each segment and exploited to detect rest, recovery, physical exercise, light physical activity or postural changes.

The selection of a segmentation algorithm or a specific attribute set for the method is application dependent. No universal segmentation algorithm allowing segmentation of any nonstationary time series exists. Furthermore, there is a compromise between the accuracy and computational complexity among the methods. Thus, a variety of algorithms should be considered, depending on the purpose and characteristic properties of the application.

3.4.1 Moving a PSD template across the signal to detect change points

Cohen [27, p. 825-827] presents a simple approach for segmenting biomedical nonstationary signals. A reference window is constructed by calculating a PSD estimate from the beginning of the signal. Suitable prior knowledge is used to define the appropriate and reliable window length. The algorithm proceeds with a comparison between a sliding window, which is shifted along the signal, and the reference window. A threshold and an appropriate distance measure are used to decide if the two windows are considered to be close enough.

Cohen chooses a relatively normalized spectral distance to measure the difference between the windows:

$$D_t = \int \left(\frac{S_R(w) - S_t(w)}{S_R(w)} \right)^2 dw,$$

where $S_R(w)$ and $S_t(w)$ are the PSD estimates of the two windows. When a threshold is exceeded, the last window is defined as the reference window (or template), a new segment is started and the algorithm continues.

The algorithm may be modified with a growing reference window instead of a fixed one. Also the way PSD or distance measure is calculated may vary, depending on the application. In addition, preprocessing and selection of a suitable frequency band may be required if, for instance, you wish for the affect of long-term trends or signal noise to be eliminated.

The algorithm is sensitive to signal artifacts, which affect the power spectrum, and the squared distance between the PSD estimates. Hence, the correction of signal artifacts and outliers are essential.

3.4.2 Signal decomposition and generalized likelihood ratio test

Another approach for nonstationary and nonlinear time series segmentation is the use of the generalized likelihood ratio test to detect changes in signals. We follow the ideas presented by Fancourt et al. [37][6], but introduce two enhancements to this algorithm. The first improvement applies the GLR to multivariate signals. The second discussion considers a hybrid model, where a signal is decomposed and further processed with a simple model to find the proper segmentation with GLR.

We will briefly discuss one possible setup for the GLR algorithm. A full description, alternative setups, discussion of implementation issues, as well as a theoretical backround of the GLR algorithm, are provided by the article of Fancourt and Principe[7] [37].

We define the time index N relative to the last detected change point (CP) to keep the notation simple. The GLR algorithm is based on the following *log-likelihood ratio* (LLR):

$$L(T,N) = \frac{(T-1)}{2}\log\left(\frac{E_1}{E_2}\right) + \frac{(N-T+1)}{2}\log\left(\frac{E_1}{E_3}\right). \tag{35}$$

It is used to test whether a change has occurred inside the window $\{1\ldots N\}$ or not. The variable T is the intersection point dividing the first (whole) region, $\{1\ldots N\}$, to second, $\{1\ldots T-1\}$, and third, $\{T\ldots N\}$, with respective estimation errors E_1, E_2 and E_3. The mean-squared estimation errors (see Section 3.1.4) are computed between the model and signal in their respective regions. Figure 24 illustrates a *three model GLR*.

The *initial search region length* (ISR) defines the minimum range in which the algorithm is applied. It will also define the initial window range after the CP

[6]The article of Willsky and Jones [181] presented first application of GLR to detection of abrupt nonstationary changes in signal.

[7]Fancourt and Principe apply neural networks to GLR as they produce the log-likelihood ratio from neural network forecast errors of the signal. A time-delay neural network, trained with Lavenberg-Marquardt algorithm, was applied for signal estimation.

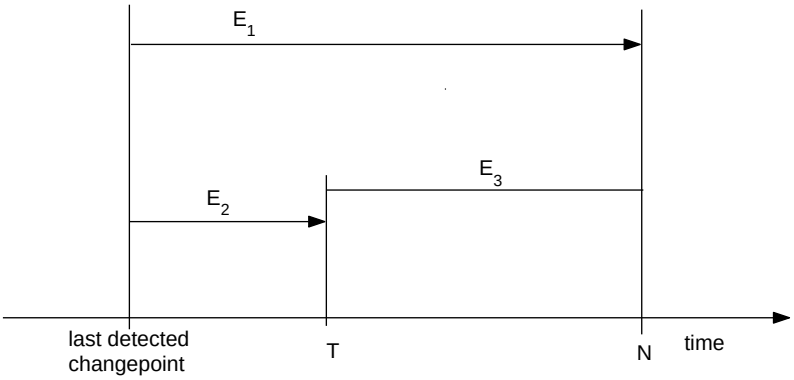

Figure 24: Three model GLR.

has been detected, and the search is continued from window $\{CP+1,\ldots,CP+ISR\} \equiv \{1,\ldots,N\}$. Notice that ISR affects the system's accuracy, since if two change points are inside the initial search region, only one of them can be detected.

A *minimum region length* (MRL) is defined to avoid having the LLR estimate constructed with too few samples. It is applied to each variance estimate E_1, E_2 and E_3 while the function of ISR is used to limit the initial window length. Hence, the variance estimates are calculated in the following regions:

E_1: $[1,N]$, where $N \geq ISR$,

E_2: $[1,T-1]$, where $T > MRL$,

E_3: $[T,N]$, where $N \geq ISR$ and $T > MRL$ and $N-T+1 \geq MRL$.

The limits reveal that MRL also defines the *dead-zone*: a window area where the change points will not be detected.

In the *outside loop* of the GLR algorithm N is increased for each iteration with a predefined *step-size* and the LLR is recalculated. The position of the intersection point T can follow the middle of the window. A threshold will determine if a change in the signal has occurred ($L(T,N) > threshold$).

In the *inside loop*, the log-likelihood ratio is used to estimate the change point inside the window by moving the intersection point T and re-calculating LLR in each instance for the three regions. In this stage the parameter N remains fixed. The CP is the intersection point where the minimum value of the ratio is achieved.

Thus, the algorithm is a two-stage procedure: in the inner iteration a minimum of $L(T,N)$ respect to intersection point T is recovered while the outer loop enlarges the search interval N or accepts a new segment.

GLR modifications

The basic GLR algorithm can also be applied to multivariate signals. One alternative is to run the algorithm separately for each signal and use the union of the

outputs to produce final segmentation. If we presume the signals to be dependent, we could run parallel GLR processes where a detected change in one signal would also reset other processes to continue from the change point.

Multivariate signals may also be treated with a single GLR run, when signal estimation errors are combined in the likelihood calculation. A simple way to unite the errors is to use the average over the errors. However, if the signal variances are not homogeneous, then the signal with the highest variance dominates. This may be prevented with the normalization of the signals to unit variance and zero mean. Normalization may fail for time-dynamic signals in on-line applications if the required statistics (mean and variance) change in time[8]. Our suggestion for modifying the GLR is to form the error function as *Mth root of product of errors*:

$$E_k = \sqrt[M]{\prod_{j=1}^{M} E_{kj}}, \tag{36}$$

where M presents the number of signals and E_{kj} their respective estimation errors in region k (see Figure 24).

With the modification presented above, the log-likelihood ratio equation results in the following:

$$\begin{aligned} L(T,N) &= \frac{(T-1)}{2} \log \left(\sqrt[M]{\prod_{j=1}^{M} \frac{E_{1j}}{E_{2j}}} \right) + \frac{(N-T+1)}{2} \log \left(\sqrt[M]{\prod_{j=1}^{M} \frac{E_{1j}}{E_{3j}}} \right) \\ &= \frac{(T-1)}{2M} \sum_{j=1}^{M} \log \left(\frac{E_{1j}}{E_{2j}} \right) + \frac{(N-T+1)}{2M} \sum_{j=1}^{M} \log \left(\frac{E_{1j}}{E_{3i}} \right). \end{aligned} \tag{37}$$

The formula in (37) results in a multivariate generalization of equation (35) by producing the log-likelihood ratio as an average variance between the error regions. If $M = 1$, then the two formulas conclude.

Notice, that the denominator in equations (35) and (37) may become zero. To prevent zero division in the equations, we may add a computer epsilon to the corresponding error estimations.

The above modification may also be applied to form an advanced implementation of the GLR calculation for a univariate signal. The idea is to represent the signal as a multivariate signal, by using multiple feature sequences. For example, in applications where the signal level and cyclic fluctuation is of interest, such as heart rate time series, we may chop the signal into several signals by using time-frequency distribution moments.

TFRD moments, such as instantaneous frequency or power in a predefined frequency band, may be used to estimate changes in the frequency domain. The

[8]If normalization is applied in an on-line application, the mean and variance is set before-hand based on pre-data and assumptions of the system's behavior. Hence, the normalization is used to scale the data to follow approximately the wished statistics and we presume that the statistics do not change considerably during the on-line process. However, if the observed system is nonstationary, the presumptions will fail.

original signal may be used to present the signal level. Furthermore, it may also be high-pass filtered or moving averaged to emphasize the level. To these multivarate signals we may apply the multivariate GLR algorithm. Each signal is estimated with a simple model, such as mean or median, resulting in fast estimation of the change point in GLR algorithm. If, for instance, the sequence mean is chosen, a mean-squared error at region k for feature sequence j, is calculated with the following equations:

$$E_{1j} = \frac{1}{N}\sum_{t=1}^{N}\left(x_j(t) - \overline{x}_{1j}\right)^2,\ \overline{x}_{1j} = \frac{1}{N}\sum_{t=1}^{N} x_j(t),$$

$$E_{2j} = \frac{1}{T-1}\sum_{t=1}^{T-1}\left(x_j(t) - \overline{x}_{2j}\right)^2,\ \overline{x}_{2j} = \frac{1}{T-1}\sum_{t=1}^{T-1} x_j(t),$$

$$E_{3j} = \frac{1}{N-T+1}\sum_{t=T}^{N}\left(x_j(t) - \overline{x}_{3j}\right)^2,\ \overline{x}_{3j} = \frac{1}{N-T+1}\sum_{t=T}^{N} x_j(t).$$

A simple estimation function may be utilized, since the signal dynamics are dispersed to the feature sequences, and the modeling of the dynamics are no longer a problem of the estimation function: *the cyclic changes in the original signal are level changes in TFRD's frequency moments*.

The use of the median to estimate a region inside a segment may be more stable than the use of the mean in presence of signal artifacts and outliers. Notice that implementation of median does not necessarily require sorting of the array (see, e.g., [29, 60, 138, 182]). For example, histogram or tree-based methods do not require full sorting of the array to calculate the median. Choosing a suitable algorithm depends on the array length, typical values, and whether we wish to save memory or CPU-time.

In time-frequency distributions, the compromise between time and frequency resolution must be considered. It is clear that the presented method suffers from time sensitivity issues if a method such as the STFT is applied for signal decomposition and calculation of, for instance, instantaneous frequency moments. STFT's time resolution depends on the used window size and is proportional to the frequency resolution: STFT with a small window has better time resolution but poor frequency resolution. However, if a larger window is used for a nonstationary time series, STFT may offer a more stable presentation of the signal. A large window gives an average estimate of power or frequency moments in a given window. Methods like Wavelet transformation have perfect time resolution but they suffer from other effects, such as instability of instantaneous frequency measures. Hence, the signal decomposition method for GLR and its usage depends on the application. For example, an application calculating TFRD, regardless of the GLR algorithm, may naturally use effectively the information for time series segmentation.

The GLR algorithm has some theoretical assumptions we have not considered in this discussion. One is an assumption of Gaussian errors in the signal

predictors and the use of the mean-squared error function. Another assumption is the use of a parametric model in signal estimation. In our experience the presented modifications to use signal decomposition and a simplified model for the estimation, seem to work and improve the algorithm implementation with decreased calculation time. The justification of a multivariate log-likelihood ratio seems reasonable as introduced in (37), although a complete theoretical justification of the enhancements is subject to future research. Next, an example of the modified GLR to decompose a signal is presented.

An example with a nonstationary sinusoid signal

Figure 25 illustrates a sinusoid signal composed with the following set of equations:

$$\hat{y}(c_1, c_2, c_3, t) = c_1 \sin\left(5 \cdot (c_2 + c_3 \cdot 2\pi t)\right) \tag{38}$$

$$x(t) = \begin{cases} \hat{y}(0.9, 0, 0.2, t) + \hat{y}(0.5, 12.5, 0.8, t) + \hat{y}(1.5, 6.25, 0.1, t), & t \leq 20 \\ \hat{y}(0.5, 0, 0.2, t) + \hat{y}(1.0, 12.5, 0.8, t) + \hat{y}(0.5, 6.25, 0.1, t), & t > 20 \ \wedge \ t \leq 40 \\ \hat{y}(0.5, 0, 0.2, t) + \hat{y}(2.5, 12.5, 0.8, t) + \hat{y}(0.8, 6.25, 0.1, t), & t > 40 \end{cases} \tag{39}$$

where the sampling frequency of the signal is set to five hertz and time t is presented in seconds. The variables c_1, c_2 and c_3 represent the time series amplitude, phase and frequency, respectively. Notice that no noise or artifacts are presented in the equation.

The signal consists of three stationary signals each containing three distinct sinusoid components with defined amplitudes and frequencies. Furthermore, Figure 25 presents the mean frequency and power of the short-time Fourier transformation (STFT) applied to the dataset. In this example, the STFT is calculated with a ten second Hanning window.

The mean frequency and power estimates are used as a multivariate input for the GLR algorithm to search for change points in the signal using formula (37). The true change points we wish to detect are in $t = 20$ and $t = 40$. A median function is used to fit the signal decomposition features, mean frequency and power, to their respective median values inside each segment candidate. Estimation errors E_{kj} for the three regions are calculated as a mean-squared error between the median of the feature signal j in region k and the feature signal j. The resulting segmentation, together with medians of each feature inside the segment, are illustrated with horizontal and vertical lines in Figure 25.

Visual inspection indicates that the setting of the first change point is not consistent and thus the algorithm may be considered to behave well. A human expert would perhaps place the second change point later if only the raw signal would be considered. However, evaluation of the features indicate an abrupt increase in the mean power just before the 40-second mark. Thus, based on the visual inspection of the features, the second labeling is reasonably accurate.

ISR	MRL=1			MRL=3		
	TH=10	TH=30	TH=50	TH=10	TH=30	TH=50
4	22/0.6	8/3.1	3/2.4			
12	7/2.9	3/2.4	2/2.5	6/2.8	3/2.5	2/2.5
20	4/0.9	2/2.4	2/2.5	4/1.0	2/2.5	2/2.5
28	2/2.6	2/2.6	2/2.6	2/2.3	2/2.3	2/2.3
36	2/2.2	2/2.2	2/2.2	2/2.0	2/2.0	2/2.0

Table 2: The sensitivity of GLR method on its own parameters with an example dataset presented in (39) and Figure 25. The table contains a number of change points and a mean absolute error presented as $\#CP/MAE$-pairs. The MRL and ISR are presented in seconds. Furthermore, the abbreviation TH stands for the threshold value.

GLR sensitivity to its own parameters

The GLR algorithm has some attributes and variables of its own that must be set for a given application. The step size of the algorithm affects the computational load of the method and should be exploited in the outside loop if less precision is tolerated. Naturally the step size should be small enough to avoid two change points to appear in the observed window $\{1 \dots N\}$. The inside loop of the algorithm searches the change point with a step size equal to one.

Also the minimum region length in the segmentation must be considered, since too small a range, i.e., not enough data points, may result in poor model estimation[9]. The initial search region length, i.e., the initial size of the region after each detected change point, should be small enough to avoid several change points situated inside the observed region. Furthermore, the relationship between the MRL and the ISR affects the precision of the GLR method and declares the dead-zone. Thus, if there is a change point inside the initial region it may be accurately discovered only if it is inside the interval $[1 + MRL, N - MRL]$.

The search of GLR parameters for the segmentation of the signal presented in Figure 25 is demonstrated in Table 2. The table contains the free parameters, the number of change points and the mean absolute error of the segmentation. More precisely, the error is defined as a *mean absolute distance between the closest CP and the true change point* in seconds.

The analysis of Table 2 reveals that the correct number of change points and the minimum error is achieved with three different attribute sets, where the ISR$=$ 36 seconds. The best result is somewhat undesirable: the chosen ISR will start (and restart) the outside loop of the algorithm in an optimal window, where exactly one change point is located. Since the signal length is 60-seconds, the outside loop will only be executed twice. In this example, other parameter settings with smaller ISR could execute better with new data, as the result does not indicate optimal

[9]The stability of the median may help as the median is already a stable statistic for small amounts of data.

parameter set but only optimal ISR.

With a smaller initial range, the algorithm reveals more change points when used with a small threshold. However, when a higher threshold value is set, the algorithm has a better chance to discover the true number of segments regardless of the small ISR.

Since the algorithm is sensitive to its own free parameters, an expert analysis must be considered before an automated use of the method. The sensitivity analysis also indicates that the signal must contain some kind of stability: the signal has to behave reasonably well for the algorithm to work on-line. If the attributes are set with an initial signal, an explosion or a diminishing of the signal amplitude or variance compared to the original signal will result in poor performance of the algorithm. This is the price to pay, however, since an algorithm without any attributes giving a reasonable segmentation for any temporal changes in the signal would be a universal segmentation machine.

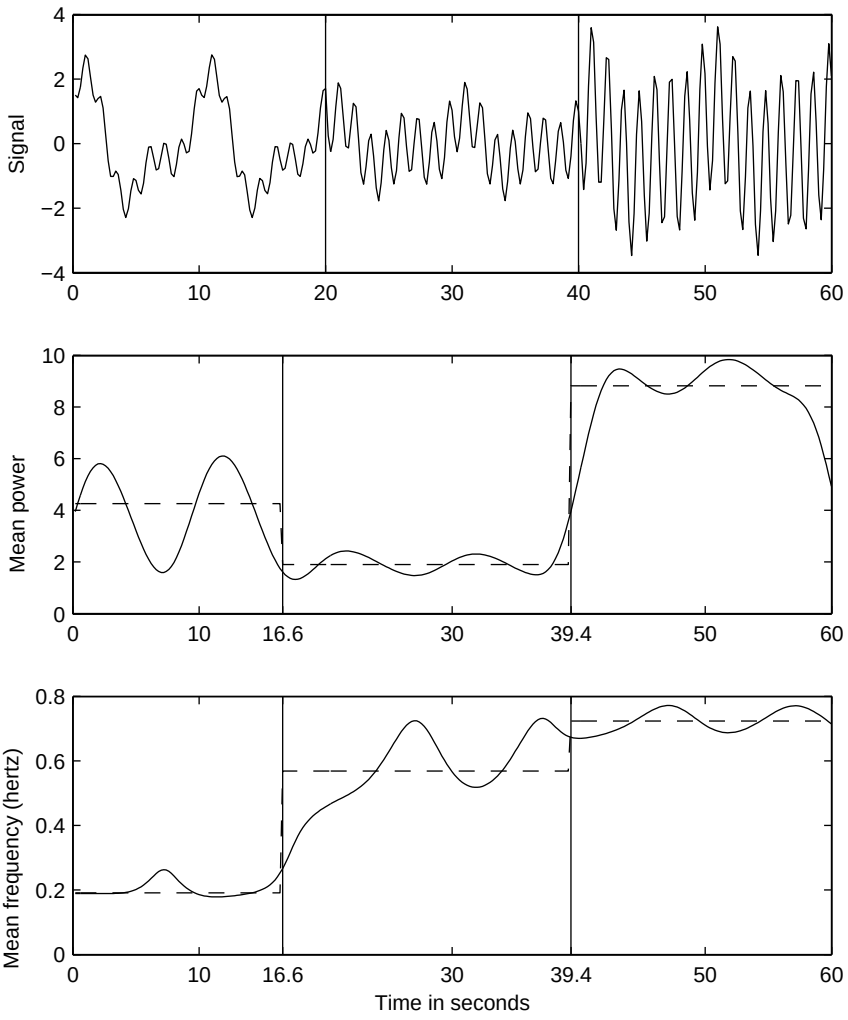

Figure 25: Segmentation of an example dataset with the GLR algorithm. The upper figure illustrates the original signal, the middle the mean power of the STFT of the signal, and the bottom figure the corresponding mean frequency of the STFT. STFT is calculated with a ten-second Hanning window and the signal is sampled in five hertz. Furthermore, the vertical dashed lines illustrate median of the feature inside the segment. In the upper figure, horizontal lines illustrate the true change points, while the middle and bottom figures illustrate the estimated change points, the result of the GLR algorithm. The GLR algorithm was applied with following parameters: threshold=10, ISR=36, MRL=3 and step size of one second.

4 NEURAL NETWORKS

Neural networks provide a powerful and flexible nonlinear modeling tool for time series analysis. They may also be utilized for classification, autoassociative filtering, prediction, system control or image compression, just to mention a few applications. See, e.g., [4, 20, 49, 95, 122, 147, 170, 173, 178].

In this dissertation we concentrate on the second generation neural networks [173] and especially the feed-forward neural network. The basic principle for a *feed-forward neural network* (FFNN) (a.k.a. multilayer perceptron) is to train a network with real world empirical data with input-output samples to construct a nonlinear relationship between the samples and to generalize this to outside observations. However, the generalization is limited, since for common problems extrapolation may be harder than interpolation between the training points.

The universal approximation theory, presented in Section 4.1.1, provides the grounds for the practical observation that, for the stationary time series, the selection of the correct neural network architecture is not the main problem when modeling a system. In our experience the most complex process is choosing an appropriate neural network *training* algorithm. Training refers to the adaptation process by which the neural network learns the relationship between the inputs and targets. This process is often guided by an optimization algorithm [144].

Often the whole neural network concept seems less complex than the variety of optimization methods and heuristics that may be utilized for the training. Still a number of articles are published proposing a new training algorithm or an improvement to the existing one, see, e.g., [14, 21]. In this dissertation we outline some principles for network optimization and refer to common optimization steps familiar within neural network literature.

The neural network is heavily influenced by the training samples, and thus, a valid sampling of observations is necessary. The network optimization is usually executed in a mean-squared-error sense using the error functions (9) or (10). This specializes the network in learning the observations occurring most often in the set. It is therefore important to have even sampling of the function range. Thus, the distribution of the output space should be generally smooth. Notice that, as we mentioned, the neural network may be used to interpolate between the training points. This may be exploited with sampling to reduce the data in some applications.

Artifacts will affect the network performance as for any linear or nonlinear model. If the noise in the signal is non-Gaussian with a mean other than zero, then the model will include a bias towards the noise [88].

4.1 Feed-forward neural networks

The unquestionably most popular neural network architecture is the family of feed-forward networks, together with the backpropagation training algorithm introduced by Rumelhart, Hinton and Williams in 1986 [148] [10].

Feed-forward networks are widely used by neural network researchers and they give a theoretical basis for constructing more sophisticated models. In time series modeling, feed-forward networks can give good results if the observed phenomenon is stationary. This will be shown in the examples presented at the end of this section. For time-varying or chaotic time series the network must contain some temporal information to enable good performance.

4.1.1 Motivation

If the values of the time series are determined by some mathematical function, then the system is said to be deterministic. For such systems *Takens' theorem* [168] implies that there exists a *diffeomorphism*, a one to one differentiable mapping with a differentiable inverse, between a sufficiently large window of the time series

$$x(k-1), x(k-2), \ldots, x(k-T),$$

and the underlying state of the dynamic system which gives rise to the time series. This implies that there exists, in theory, a nonlinear autoregression of the form

$$x(k) = g[x(k-1), x(k-2), \ldots, x(k-T)],$$

which models the series exactly (assuming there is no noise). The function g is the appropriate diffeomorphism.

Another important result, *the universal approximation theorem*, is the one shown by Irie and Miyake [63], Hornik, Stinchcombe and White [58], Cybenko [30] and Funahashi [43]: a FFNN with a arbitrary number of neurons is capable of approximating any uniformly continuous function to an arbitrary accuracy [144, 178][11].

4.1.2 The network architecture

Figure 26 illustrates the structure of a multilayer feed-forward network. The data flows strictly forward and no *feedback* connections exist, that is, connections from the output units to the previous or same layers.

[10] It appears that the history of the backpropagation algorithm can be tracked to Paul Werbos and his doctoral thesis at Harvard University in August 1974 [54, p. 41].

[11] Notice that universal approximation is not a rare property. Polynomials, Fourier series, wavelets, etc. have similar capabilities, so that only a lack of the universal approximation capability would be an issue [144].

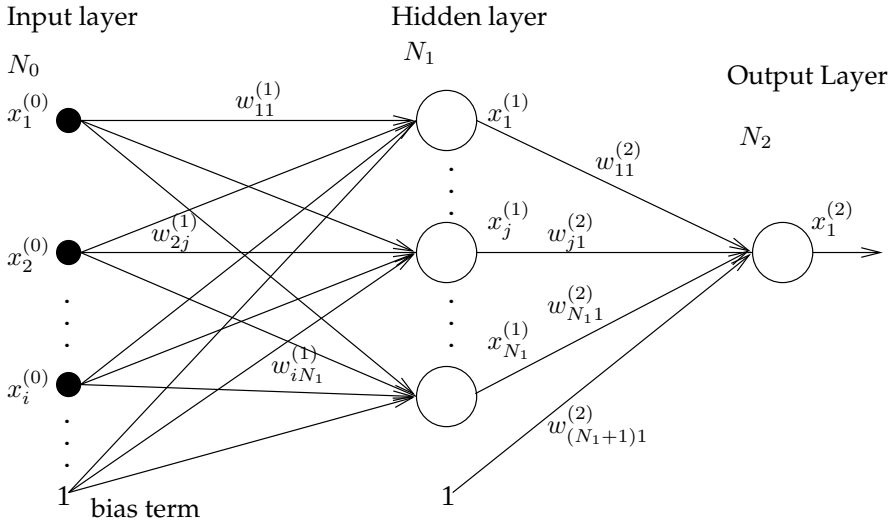

Figure 26: A multilayer feed-forward network with one hidden layer.

To investigate the architecture more closely let us take a look at a single unit j (or neuron) in layer l of the network (Figure 27). The unit receives N_{l-1} real-valued inputs from the previous layer, which are multiplied by weight parameters $w_{ij}^{(l)}$. Layer 0 is taken to consist of the input variables, thus the input layer has N_0 units, hidden layer l has N_l units and output layer L has N_L units. For weight parameter $w_{ij}^{(l)}$ the indices i and j notate a one-way directed connection between unit i in layer $l-1$ and unit j in layer l. Weight parameters are combined using the integration function g, which (in the case of standard FFNN) is a sum of the inputs

$$g(x_1^{(l-1)}, x_2^{(l-1)}, \ldots, x_{N_{l-1}}^{(l-1)}) = \sum_{i=1}^{N_{l-1}} w_{ij}^{(l)} x_i^{(l-1)} + w_{(N_{l-1}+1)j}^{(l)}.$$

This sum of the inputs multiplied by the weights is also called the *excitation* of the jth unit. Haykin [54] refers to this as the *net activation potential of neuron j*.

As a more practical notation we define excitation of unit j in layer l as

$$s_j^{(l)} = \sum_{i=1}^{N_{l-1}} w_{ij}^{(l)} x_i^{(l-1)} + w_{(N_{l-1}+1)j}^{(l)}. \tag{40}$$

The extra parameter $w_{(N_{l-1}+1)j}^{(l)}$ in the preceding equations is a *bias*-term (a.k.a. *threshold, offset*). Note that the inputs to a unit in layer l define an N_{l-1}-dimensional space where the weights of the unit determine a hyperplane through the space. Without a bias input, this separating hyperplane is constrained to pass through the origin.

By setting

$$x_{N_{l-1}+1}^{(l)} = 1 \qquad \text{for } 0 \leq l \leq L-1$$

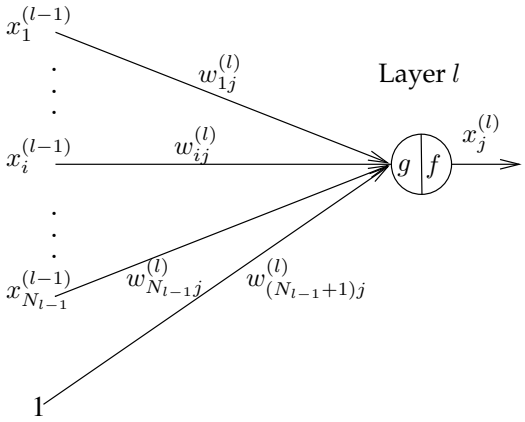

Figure 27: A single unit in a feed-forward network.

we may write equation (40) in a shorter form:

$$s_j^{(l)} = \sum_{i=1}^{N_{l-1}+1} w_{ij}^{(l)} x_i^{(l-1)}.$$

Other types of integration functions, for instance multiplication, could be foreseen, but addition is used to preserve the locality of the neuron information in backpropagation, introduced in Section 4.1.3 [147, p. 170].

After computation of the integration function the result is directed to the *activation* function f. If the activation function is $f(x) = x$, then the neuron simply computes a linear combination of the inputs. Since the composition of linear functions is again a linear function, the network would only be a plain AR-net (see Section 3.1.7). To add nonlinear properties we use a sigmoid function, mapping

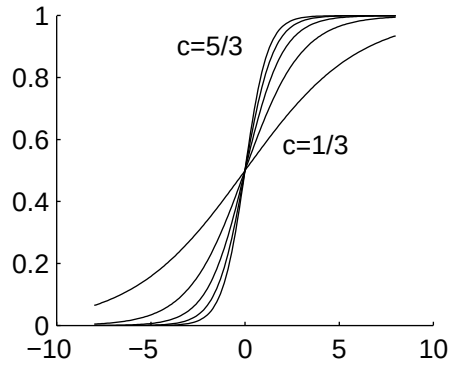

Figure 28: Sigmoid function $f_c(x)$ with different values of parameter c.

the real numbers to the interval $[0, 1]$ (see Figure 28):

$$f_c(x) = \frac{1}{1 + e^{-cx}}. \tag{41}$$

The activation function must be a nonlinear differentiable map to allow the backpropagation-algorithm to work. The *logistic*, *tanh* and *Gaussian*[12] functions are commonly used. Sigmoid and the tanh function have the same shape but tanh defines a mapping from the real axis to the interval $[-1, 1]$:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{42}$$

The output $x_j^{(l)}$ of the unit j is now

$$x_j^{(l)} = f(s_j^{(l)}).$$

If we use only one unit with a nonlinear activation function, then the network is a representation of a generalized linear model [106].

In time series prediction (Figure 29) the feed-forward network has a single output unit, T input units and $(L - 1)$ hidden layers. To use previous notation,

$$N_0 = T, N_L = 1.$$

The T inputs are the previous values of the time series

$$x(k-1), x(k-2), \ldots, x(k-T) = \mathbf{x}^{(0)} = x_1^{(0)}, x_2^{(0)}, \ldots, x_T^{(0)},$$

where k denotes time. These are used to predict the output value

$$\hat{x}(k) = x_1^{(L)}.$$

The vector of inputs is sometimes referred to as the *data window*. Teaching is done over all known times k. When teaching, real values $x(k)$ are used in inputs, not the network generated approximations $\hat{x}(k)$. This type of learning is also known as *teacher forcing*, *equation error formulation* or *open-loop adaptation scheme* [54, p. 516].

When predicting future points, approximations $\hat{x}(k)$ must be used. Haykin names this type of teaching a *closed-loop adaptation scheme* [54, p. 516] . Bishop calls this *multi-step ahead* prediction and when predicting only one future point it is called *one-step ahead* prediction [13, p. 303] .

4.1.3 Backpropagation algorithm

Backpropagation is the most commonly used method for training feed-forward neural networks and is presented by several authors, e.g., [13, 19, 55, 144, 147]. It should be noted that the term backpropagation refers to two different things.

[12] Notice that the Gaussian function is mainly used with the radial basis function network presented in Section 4.3.1.

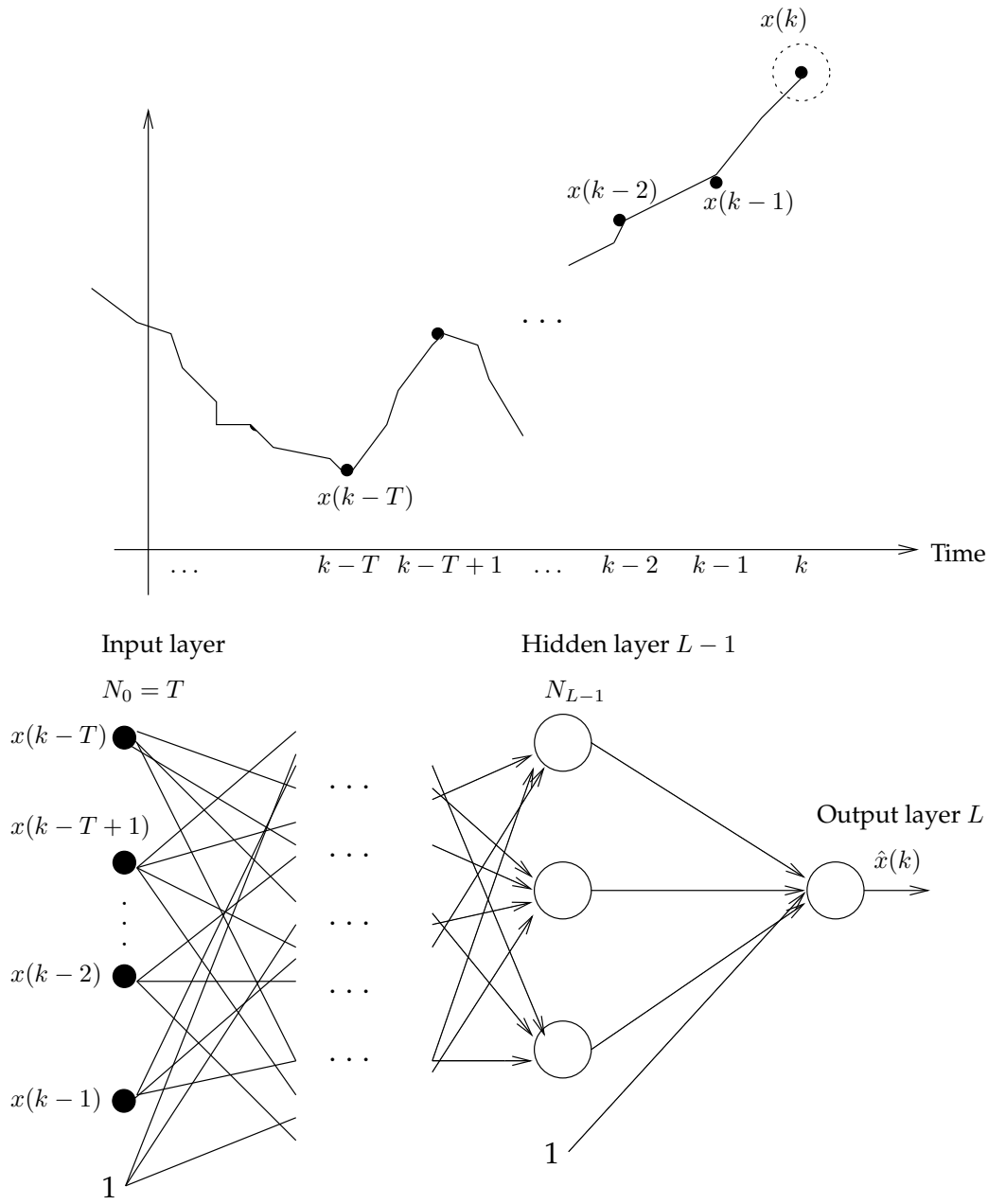

Figure 29: Feed-forward network in time series prediction.

First, a backpropagation algorithm describes a method to calculate the derivatives of the network training error with respect to the weights by utilizing a derivative chain-rule. Second, the concept is used for the training algorithm that is basically equivalent to the gradient descent optimization method [144, p. 49]. Both purposes of use are presented in this section.

The backpropagation algorithm looks for the local minimum of the error function in weight space using the gradient descent method. The combination of weights that minimizes the error function is considered to be the solution of the learning problem. The error function for p training patterns is defined as

$$E = \frac{1}{2} \sum_{k} ||\mathbf{x}^{(L)}(k) - \mathbf{t}(k)||^2,$$

where $\mathbf{x}^{(L)}(k)$ is the output generated by the network and $\mathbf{t}(k)$ is the desired target vector of dimension N_L (cf. Section 3.1.4). Since we use a differentiable activation function and addition as the integration function, this error function will be differentiable.

Next we restrict the error function to contain only one training pattern. The error function may, in this case, be written as

$$E = \frac{1}{2} \sum_{i=1}^{N_L} (x_i^{(L)} - t_i)^2.$$

To minimize the error function with respect to the weight parameters we use an iterative process of gradient descent for which we need to calculate the partial derivatives

$$\frac{\partial E}{\partial w_{ij}^{(l)}}.$$

Each weight is updated using the increment

$$\triangle w_{ij}^{(l)} = -\gamma \frac{\partial E}{\partial w_{ij}^{(l)}} \Longleftrightarrow w_{ij}^{(l)} = w_{ij}^{(l)} - \gamma \frac{\partial E}{\partial w_{ij}^{(l)}},$$

where γ is a *learning rate* that defines the step length of each iteration in the negative gradient direction.

Let us take a closer look at the process in the example of a two-layer FFNN. We will show the precise formulas to calculate each weight update. This example can then be generalized to more complex structures.

The activation function $f(x)$ will be fixed as the sigmoid, in (41), with parameter c set to 1. Its derivative evaluates to the simple form $f(x)(1 - f(x))$. The backpropagation algorithm can be decomposed into four steps: Feed-forward computation, backpropagation to the output layer, backpropagation to the hidden layer and finally computation of the weight updates.

In the first step the input vector $\mathbf{x}^{(0)} = (x_1^{(0)}, \ldots, x_{N_0}^{(0)})$ is presented to the network. The vectors $\mathbf{x}^{(1)} = (x_1^{(1)}, \ldots, x_{N_1}^{(1)})$ and $\mathbf{x}^{(2)} = (x_1^{(2)}, \ldots, x_{N_2}^{(2)})$ are computed

and stored. The evaluated derivatives of the activation functions are also stored at each unit.

In the second step we calculate the first set of partial derivatives $\partial E/\partial w_{ij}^{(2)}$.

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = [x_j^{(2)}(1 - x_j^{(2)})(x_j^{(2)} - t_j)]x_i^{(1)} = \delta_j^{(2)} x_i^{(1)},$$

where we defined the *backpropagated error*

$$\delta_j^{(2)} = x_j^{(2)}(1 - x_j^{(2)})(x_j^{(2)} - t_j).$$

Next we have to calculate backpropagation to the hidden layer. The partial derivatives are

$$\frac{\partial E}{\partial w_{ij}^{(1)}} = \delta_j^{(1)} x_i^{(0)},$$

where

$$\delta_j^{(1)} = x_j^{(1)}(1 - x_j^{(1)}) \sum_{q=1}^{N_2} w_{jq}^{(2)} \delta_q^{(2)}.$$

The final step is to calculate the weight updates. The corrections to the weights are given by

$$\triangle w_{ij}^{(2)} = -\gamma x_i^{(1)} \delta_j^{(2)}, \qquad \text{for } i = 1, \ldots, N_1 + 1; j = 1, \ldots, N_2,$$

and

$$\triangle w_{ij}^{(1)} = -\gamma x_i^{(0)} \delta_j^{(1)}, \qquad \text{for } i = 1, \ldots, N_0 + 1; j = 1, \ldots, N_1,$$

where the bias terms are included by setting $x_{N_0+1}^{(0)} = x_{N_1+1}^{(1)} = 1$.

More than one training pattern

To achieve higher accuracy in the model, multiple training patterns are used. Corrections can be made using *on-line-* or *off-line updates*. For p training patterns the off-line method gives updates in the gradient direction in the form

$$\triangle w_{ij}^{(l)} = \triangle_1 w_{ij}^{(l)} + \triangle_2 w_{ij}^{(l)} + \cdots + \triangle_p w_{ij}^{(l)}.$$

As gradient direction is mathematically a linear operator, the off-line update is an analytically valid operation. An alternative is to use on-line training where weight updates are made sequentially after each pattern presentation. On-line training can be seen as adding noise to the gradient direction and, thus, it may help the procedure to avoid falling into shallow local minima of the error function [147, p. 167].

4.1.4 Some theoretical aspects for a feed-forward neural network

The universal approximation theory implies that any continuous function can be approximated to arbitrary accuracy with a two-layered network. This does not, however, mean that a two-layered network is optimal, e.g., in the sense of learning time or the number of network parameters. In addition, there exists functions, which may not be approximated with a two-layer network with any number of units, but that can be approximated with three-layered networks [162, 163] (cited in [144]).

Another theoretical result presented, for example, in Bishop [13], is that a function presented by a two-layered feed-forward network with sigmoid activation with fixed c in (41) and N_1 units in the hidden layer has $N_1!2^{N_1}$ different parameter combinations that result in the same function.

Yet another result is, shown by Barron [3] and Jones [69], that the residual error of the network function decreases as $\mathcal{O}(1/N_1)$ as the number of hidden units is increased[13].

Kolmogorov's theorem

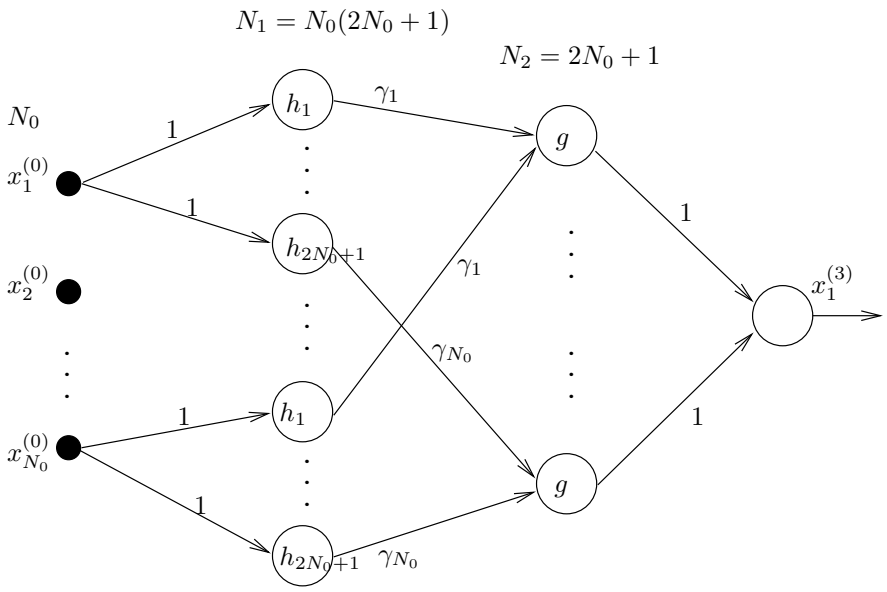

Figure 30: A feed-forward network to implement Kolmogorov's theorem.

A theoretical result obtained by Kolmogorov [13, 85, 147] says that every continuous function of N_0 variables can be presented as the superposition of a

[13]For positive functions f and g, we use the notation $f = \mathcal{O}(g)$, if $f(N) < ag(N)$ for some positive constant a and sufficiently large N.

small number of functions of one variable. In neural networks this means that any continuous function can be presented exactly by a three-layered network having $N_1 = N_0(2N_0 + 1)$ and $N_2 = 2N_0 + 1$ units in the hidden layers. The network architecture is presented in Figure 30. Given the functions $h_j(x)$ and $g(x)$ the output of the network is

$$x_1^{(3)} = \sum_{j=1}^{2N_0+1} g\left(\sum_{i=1}^{N_0} \gamma_i h_j(x_i^{(0)})\right). \tag{43}$$

Function h_j is strictly monotonic and g is real valued and continuous. The function g depends on the function we wish to approximate but h_j does not. Kolmogorov's theorem is an existence result; we do not have any method to find the unknown functions h_j and g.

4.2 Introducing temporal dynamics into neural networks

The underlying presumption for feed-forward neural network is that the input-output sample dynamics do not change in time, i.e., the same input always maps to a similar output. To override this limitation network architectures developed for temporal, dynamic time series include delayed, or recurrent synapses between the neurons allowing the networks internal state to change in time resulting in different outputs between equal inputs for different time instants.

Even if the feed-forward neural network is able to present any input-output mapping to arbitrary accuracy, the network may not be optimal in the sense of architecture (number of parameters), learning time or in terms of generalization. In this section we introduce two different networks applicable for the modeling of time dynamic systems: the Jordan network and FIR network. Later in Section 6.2 the networks are applied to excess post-oxygen consumption modeling, which also demonstrates the significance of the dynamic neural network structure.

There are also other popular recurrent network architectures not discussed in this work: Hidden Markov Models, Elman network, Hopfield network, Boltzmann machines, the mean-field-theory machine and methods for real-time nonlinear adaptive prediction of nonstationary signals [55].

4.2.1 An output recurrent network, the Jordan Network

Jordan presented his recurrent neural network model in 1986 [70, 71]. A *Jordan network* has recurrent connections from the output layer to the input layer. These delayed values are called *state units*. State units also have self-connections, making their total output to be a weighted sum of the past $k-2$ output values. Figure 31 shows the basic structure of the Jordan network.

State units at time k are defined as

$$x_{i+N_0-N_L}^{(0)}(k) = \hat{w}_{ii} x_i^{(L)}(k-2) + x_i^{(L)}(k-1), \qquad \text{for } 0 < i \leq N_L. \tag{44}$$

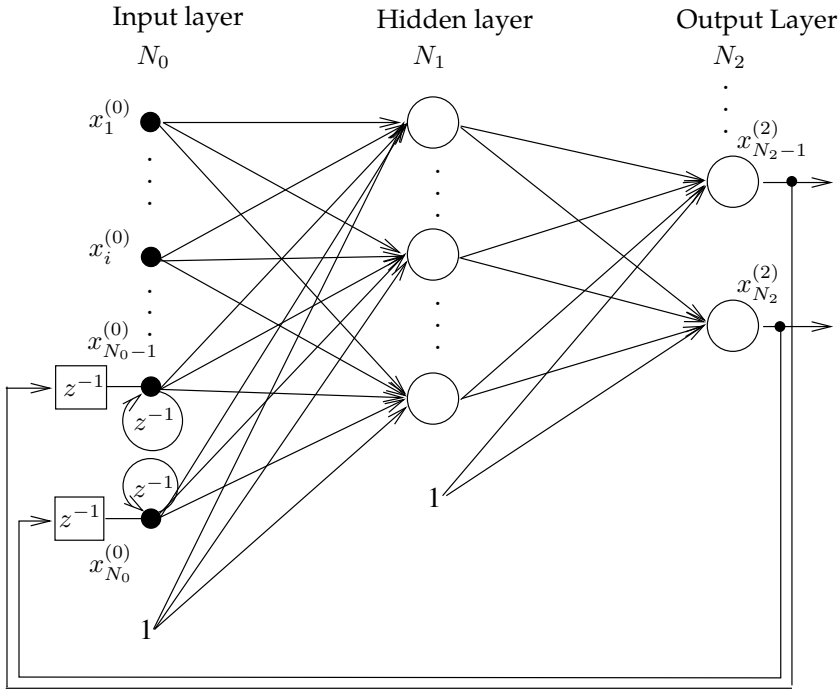

Figure 31: Jordan Network. The *unit delay operator* z^{-1} expresses a reduction of time index by one, $z^{-1}x(k) = x(k-1)$ and $z^{-1}(z^{-1}x(k)) = x(k-2)$.

The total excitation of unit j (including bias) in layer l is

$$s_j^{(l)} = \sum_{i=1}^{N_{l-1}+1} w_{ij}^{(l)} x_i^{(l-1)}, \qquad \text{where } x_{N_{l-1}+1}^{(l-1)} = 1. \tag{45}$$

Net excitation is directed to activation function $f(\cdot)$ and we get an output of unit j in layer l:

$$x_j^{(l)} = f(s_j^{(l)}), \qquad \text{for } 0 < l \leq L. \tag{46}$$

It is possible to solve the unknown network parameters, e.g., by unfolding the network to its static representation. This training procedure is called temporal backpropagation and it is introduced in Section 4.2.3.

4.2.2 Finite Impulse Response Model

The FIR, or Finite-Duration Impulse Response model (or simply Finite Impulse Response model) is also a feed-forward network. It attains dynamic behavior by introducing FIR linear filters to each weight connection.

Output at time k in each FIR linear filter corresponds to a weighted sum of

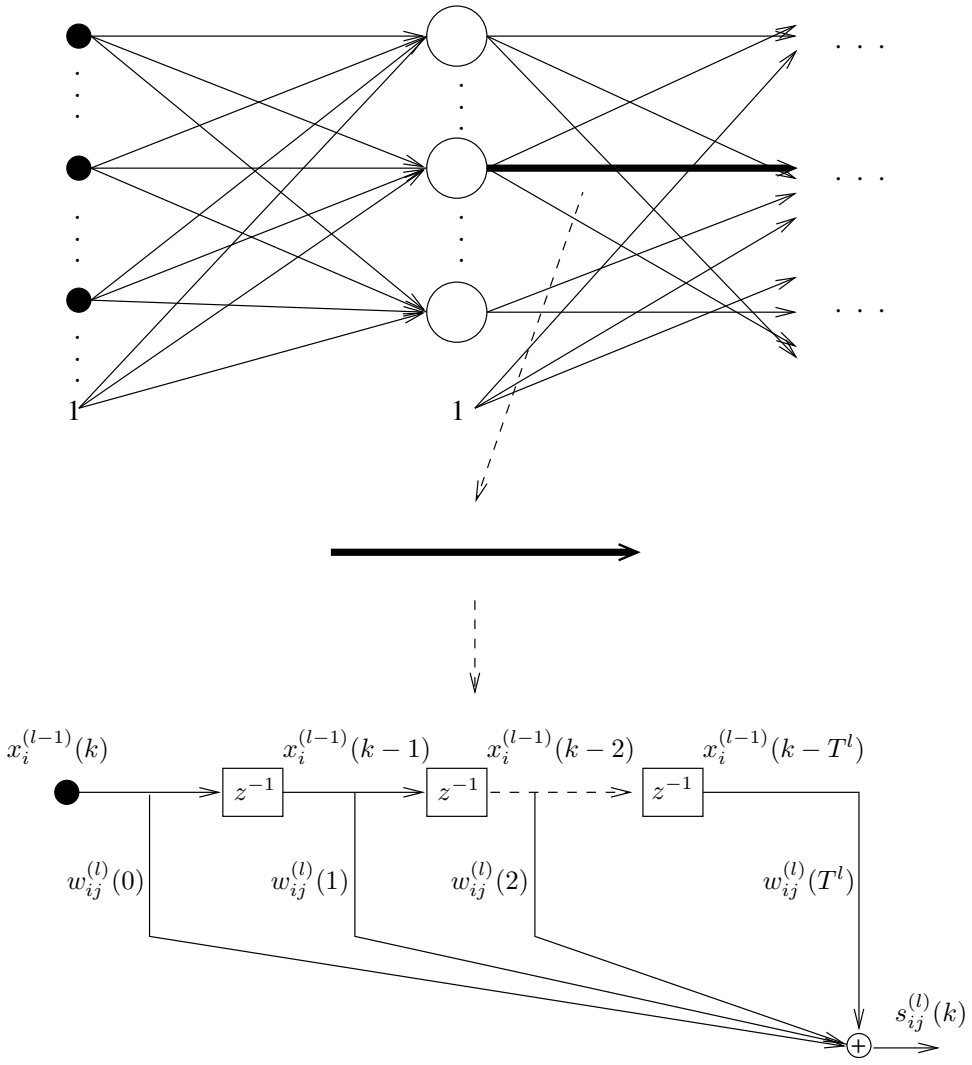

Figure 32: FIR multilayer network and linear filter. z^{-1} is a unit delay operator.

past delayed values of the input:

$$y(k) = \sum_{n=0}^{T} w(n)x(k-n). \tag{47}$$

Note that this is a result of one filter. Next we use notation introduced in the previous section and generalize (47) into a multilayer perceptron (see Figure 32).

We may write the excitation of neuron j in layer l given by input i in layer $l-1$ as

$$s_{ij}^l(k) = \sum_{n=0}^{T^l} w_{ij}^l(n) x_i^{l-1}(k-n)$$

We may write this in a matrix form by introducing the following definitions:

$$\mathbf{w}_{ij}^l = (w_{ij}^l(0), w_{ij}^l(1), \ldots, w_{ij}^l(T^l)),$$

$$\mathbf{x}_i^l(k) = (x_i^l(k), x_i^l(k-1), \ldots, x_i^l(k-T^{l+1})).$$

Now the excitation takes form

$$s_{ij}^l(k) = \mathbf{w}_{ij}^l(\mathbf{x}_i^{l-1}(k))^T. \tag{48}$$

Total excitation of neuron j in layer l at time k may now be written as

$$s_j^l(k) = \sum_{i=1}^{N_{l-1}+1} s_{ij}^l(k), \tag{49}$$

where

$$s_{(N_{l-1}+1)j}^l(k) = \theta_j^l \qquad \text{for all } k$$

is the bias term. The output of neuron j at time k is

$$x_j^l(k) = f(s_j^l(k)), \tag{50}$$

where $f(\cdot)$ is an activation function, for example a sigmoid.

A note must be given to a fact that FIR networks can be shown to be functionally equivalent to time-delay neural networks (TDNN). FIR (and TDNN) can be formulated to follow static structure by removing all time delays [178, p. 199-202]. This technique is known as unfolding-in-time. The resulting static network is much larger and has perhaps mostly a theoretical value, as the network size is proportional to the number of training samples. However, this shows that a FIR network can be considered a compact representation of a larger static network and its network parameters may also be solved by using a standard backpropagation algorithm presented in Section 4.2.3.

Temporal backpropagation

As discussed in the previous section, it would be possible to train a FIR network using standard backpropagation after unfolding. However, the technique has some undesirable characteristics, e.g., it requires global bookkeeping to keep track of which static weights are the same in the equivalent original network. Furthermore, unfolding will grow the resulting static network size as a function of the training samples [54, p. 510].

As an alternative, a more attractive *temporal backpropagation* algorithm, first introduced by Wan [175], is presented next. The starting point for the temporal

backpropagation algorithm is similar to standard backpropagation: we wish to calculate partial derivatives of the error function with respect to the weight vector and update the weight parameters according to the negative gradient direction.

The error function E is given by the following equations:

$$e_j(k) = x_j^L(k) - t_j(k),$$

where $e_j(k)$ is the error of output node j,

$$E(k) = \frac{1}{2}\sum_{j=1}^{N_L} e_j^2(k),$$

$$E = \sum_k E(k),$$

where summation of $E(k)$ is taken over all time.

The gradient of the error function with respect to a synaptic filter is expanded using the chain rule[14]:

$$\frac{\partial E}{\partial \mathbf{w}_{ij}^l} = \sum_k \frac{\partial E}{\partial s_j^l(k)} \frac{\partial s_j^l(k)}{\partial \mathbf{w}_{ij}^l}.$$

Note that the equality holds only if the summation is taken over all k. Now we can write down the corrections of the synaptic filters:

$$\mathbf{w}_{ij}^l(k+1) = \mathbf{w}_{ij}^l(k) - \gamma \frac{\partial E}{\partial s_j^l(k)} \frac{\partial s_j^l(k)}{\partial \mathbf{w}_{ij}^l},$$

where γ is the learning-rate parameter. Furthermore, from the definition of $s_j^l(k)$ in equations (48) and (49) we calculate

$$\frac{\partial s_j^l(k)}{\partial \mathbf{w}_{ij}^l} = \mathbf{x}_i^{l-1}(k),$$

where $\mathbf{x}_i^{l-1}(k)$ is the input vector applied to a neuron j in layer l. Defining

$$\delta_j^l(k) \equiv \frac{\partial E}{\partial s_j^l(k)}$$

leads to more familiar notation (see Section 4.1.3)

$$\mathbf{w}_{ij}^l(k+1) = \mathbf{w}_{ij}^l(k) - \gamma \delta_j^l(k) \mathbf{x}_i^{l-1}(k).$$

Next we derive the explicit formulas for $\delta_j^l(k)$. For the output layer L we get

$$\delta_j^L(k) \equiv \frac{\partial E}{\partial s_j^L(k)} = \frac{\partial E(k)}{\partial s_j^L(k)} = e_j(k) f'(s_j^L(k)).$$

[14] A good introduction for the use of the chain rule with backpropagation is given by Paolo Campolucci [19, p. 22-26]

For the hidden layer, we use the chain rule twice:

$$\begin{aligned}
\delta_j^l(k) \equiv \frac{\partial E}{\partial s_j^l(k)} &= \sum_{m=1}^{N_{l+1}} \sum_t \frac{\partial E}{\partial s_m^{l+1}(t)} \frac{\partial s_m^{l+1}(t)}{\partial s_j^l(k)} \\
&= \sum_{m=1}^{N_{l+1}} \sum_t \delta_m^{l+1}(t) \frac{\partial s_m^{l+1}(t)}{\partial s_j^l(k)} \\
&= \sum_{m=1}^{N_{l+1}} \sum_t \delta_m^{l+1}(t) \frac{\partial s_m^{l+1}(t)}{\partial x_j^l(k)} \frac{\partial x_j^l(k)}{\partial s_j^l(k)} \\
&= \sum_{m=1}^{N_{l+1}} \sum_t \delta_m^{l+1}(t) \frac{\partial \left[\sum_{j'=1}^{N_l} s_{j'm}^{l+1}(t)\right]}{\partial x_j^l(k)} \frac{\partial f(s_j^l(k))}{\partial s_j^l(k)} \\
&= f'(s_j^l(k)) \sum_{m=1}^{N_{l+1}} \sum_t \delta_m^{l+1}(t) \frac{\partial s_{jm}^{l+1}(t)}{\partial x_j^l(k)}.
\end{aligned}$$

But since

$$s_{jm}^{l+1}(t) = \sum_{k'=0}^{T^{l+1}} w_{jm}^{l+1}(k') x_j^l(t-k'),$$

the partial derivative is

$$\frac{\partial s_{jm}^{l+1}(t)}{\partial x_j^l(k)} = \begin{cases} w_{jm}^{l+1}(t-k), & \text{for } 0 \le t-k \le T^{l+1}, \\ 0, & \text{otherwise.} \end{cases}$$

Now we may continue and find the final formula for $\delta_j^l(k)$ in the hidden layer:

$$\begin{aligned}
\delta_j^l(k) &= f'(s_j^l(k)) \sum_{m=1}^{N_{l+1}} \sum_{t=k}^{T^{l+1}+k} \delta_m^{l+1}(t) w_{jm}^{l+1}(t-k) \\
&= f'(s_j^l(k)) \sum_{m=1}^{N_{l+1}} \sum_{n=0}^{T^{l+1}} \delta_m^{l+1}(k+n) w_{jm}^{l+1}(n) \\
&= f'(s_j^l(k)) \sum_{m=1}^{N_{l+1}} \bar{\delta}_m^{l+1}(k) (\mathbf{w}_{jm}^{l+1})^T,
\end{aligned}$$

where

$$\bar{\delta}_m^{l+1}(k) = [\delta_m^{l+1}(k), \delta_m^{l+1}(k+1), \ldots, \delta_m^{l+1}(k+T^{l+1})].$$

Finally, the algorithm takes the form

$$\mathbf{w}_{ij}^l(k+1) = \mathbf{w}_{ij}^l(k) - \gamma \delta_j^l(k) \mathbf{x}_i^{l-1}(k), \tag{51}$$

where

$$\delta_j^l(k) = \begin{cases} e_j(k) f'(s_j^l(k)), & l = L, \\ f'(s_j^l(k)) \sum_{m=1}^{N_{l+1}} \bar{\delta}_m^{l+1}(k) (\mathbf{w}_{jm}^{l+1})^T, & 1 \le l < L. \end{cases} \tag{52}$$

Notice that equations (51) and (52) can be seen as a vector generalization of the standard backpropagation algorithm. If we replace the vectors $\mathbf{x}_i^{l-1}(k)$, $\mathbf{w}_{jm}^{l+1}$ and $\bar{\delta}_m^{l+1}(k)$ by their scalar counterparts, then the temporal backpropagation algorithm reduces to the standard backpropagation algorithm.

Computations of $\delta_j^l(k)$ require future values of δ's. This is obvious when examining equation (52) (for $l \neq L$), definition of the vector $\bar{\delta}_j^l(k)$ and time index k. To rewrite the algorithm in a causal form we use only a finite number of future values of δ and do some reindexing. This leads to the following equations

$$\mathbf{w}_{ij}^{L-n}(k+1) = \mathbf{w}_{ij}^{L-n}(k) - \gamma \delta_j^{L-n}(k-nT)\mathbf{x}_i^{L-1-n}(k-nT), \tag{53}$$

$$\delta_j^{L-n}(k-nT) = \begin{cases} e_j(k)f'(s_j^L(k)), & n=0, \\ f'(s_j^{L-n}(k-nT)) \sum_{m=1}^{N_{L-n+1}} \bar{\delta}_m^{L+1-n}(k-nT)(\mathbf{w}_{jm}^{L-n+1})^T, & 1 \leq n < L. \end{cases} \tag{54}$$

Furthermore, in equations (53) and (54) we assumed that each synaptic filter is of order T in each layer. For the general case let T_{ij}^l be the order of the synaptic filter connecting neuron i in layer $l-1$ to neuron j in layer l. Then in the previous equations we must replace the terms nT by

$$\sum_{l=L-n+1}^{L} \max\{T_{ij}^l, \text{ for all suitable } i \text{ and } j\}.$$

The idea is that the time shift for the δ associated with a given neuron must be made equal to the total number of tap delays along the longest path to the output of the network [178, p. 216-217].

FIR in practice

Erik Wan successfully used the FIR network in the Santa Fe time series competition to forecast one hundred points of the laser data (see Figure 33). The data size available for the training was one thousand. He used the first nine hundred points for training and one hundred for validation. The network architecture consisted of three layers, including twelve hidden units in both hidden layers and 25 delays for each neuron in the first layer and five on hidden layers. The FIR network was able to give one of the best results for this dataset [178].

Also Camps-Valls et al. [20] utilized the FIR network for time series prediction. They compared three different neural network models (FFNN, FIR and Elman recurrent network) for a prediction of cyclosporine dosage in patients after kidney transplantation. They also experimented with a committee network for the given task. The FIR network was chosen for the prediction of blood concentration and the Elman-network for dosage prediction.

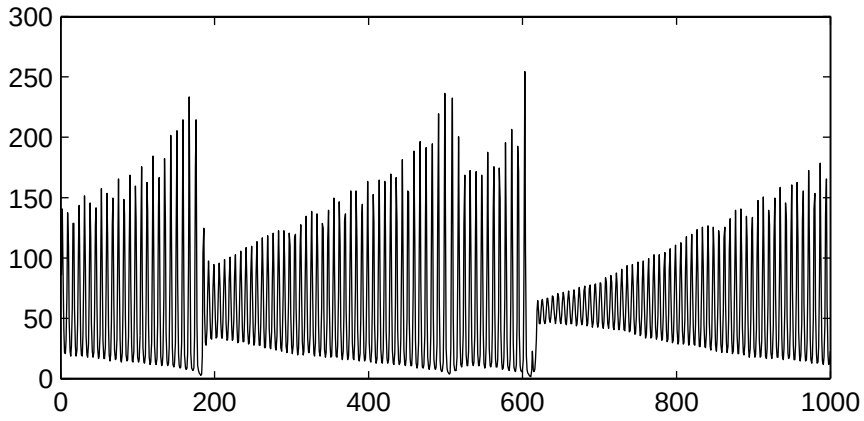

Figure 33: First one thousand points of the laser data.

4.2.3 Backpropagation through time

A time-depended neural network architecture may be transformed to its equivalent static structure. The *backpropagation through time algorithm* can be derived by *unfolding* a recurrent network into FFNN. The idea is presented in terms of an example in Figure 34. Each time step presents a new layer to the network. To train this unfolded network we may use the backpropagation through time algorithm [180].

In the procedure, layers correspond to time intervals. Time (or the number of layers) runs from 0 to L:

$$0 \leq l \leq L.$$

Instantaneous error of unit j in layer l is

$$e_j(l) = x_j^{(l)} - t_j,$$

where t_j is the desired response of unit j which is, naturally, same for all layers. The total error is

$$E = \frac{1}{2} \sum_{l=0}^{L} \sum_{j=1}^{N} e_j^2(l),$$

where N is the number of neurons in the network.

As in standard backpropagation we want to compute the partial derivatives of the error function with respect to synaptic weights w_{ij} of the network. The algorithm takes the following form:

1. Feed-forward computation for time interval $[0, L]$ is performed. All the necessary variables are stored.

2. Backpropagation to the output-, hidden- and input layers is performed to calculate local gradients. The following equations define a recursive formula

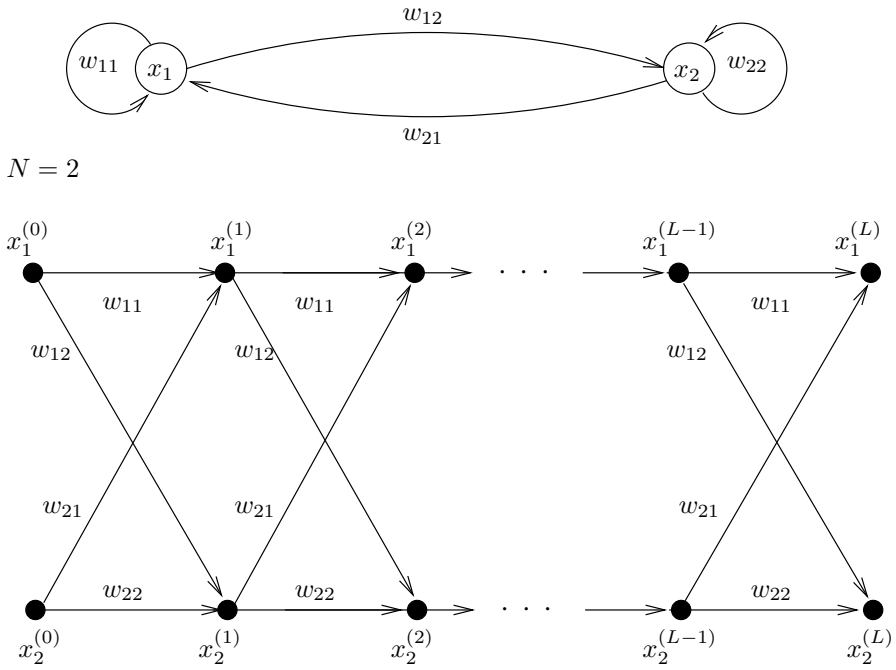

Figure 34: A two-neuron recurrent network and a corresponding network unfolded in time.

for $\delta_j^{(l)}$:

$$\delta_j^{(l)} = -\frac{\partial E}{\partial s_j^{(l)}} = \begin{cases} f'(s_j^{(l)})e_j(l) & \text{if } l = L \\ f'(s_j^{(l)})\left[e_j(l) + \sum_{m=1}^{N} w_{jm}\delta_m^{(l+1)}\right] & \text{if } 0 < l < L \end{cases}$$

where $f'(\cdot)$ is the derivative of activation function and $s_j^{(l)}$ the total excitation at time l for unit j. Index j runs from 1 to N and l from 1 to L.

3. Network weights are updated:

$$\triangle w_{ij} = -\gamma\frac{\partial E}{\partial w_{ij}} = \gamma\sum_{l=1}^{L}\delta_j^{(l)}x_i^{(l-1)},$$

where γ is the learning-rate parameter and $x_i^{(l-1)}$ in layer $l-1$ is the ith input of neuron j at layer l.

4.2.4 Time dependent architecture and time difference between observations

Sometimes the time difference between the observations is important in the modeling of a phenomena. Examples are, for example, the reduction or increase of lactates and glycogen as a function of time and exercise intensity.

If the time series is evenly sampled but contain a few missing observations, then there are basically two possible solutions. The first is natural for a static neural network, e.g., feed-forward network. In the network architecture, the time difference between the observations is fed in as a network input. However, this may not be optimal for temporal neural networks as equal inputs will result in the same output.

The alternative is to insert synthetic observations to reconstruct an even sampled time series. A generation of a new observation may be based on, for example, interpolation between samples. Furthermore, the error function may be modified to assess the reliability weighting of the samples according to equation (12).

An opposite problem occurs when the training data is sampled with a higher rate than necessary, resulting in repeated samples. For time series analysis and cyclic patterns one possibility is to use the frequency domain analysis to discover an adequate sampling rate based on the frequency-power distribution of the sequence. If the resulting spectrum does not contain power after a certain threshold frequency, then the sampling rate may be adjusted in accordance with the threshold. Naturally the Nyquist frequency and aliasing has to be taken into account.

A recurrent network may be applied to model the system based on equally sampled data. If the sampling interval is not fed into the network, or it is constant for all samples, then the network will not generalize for sequences containing a different sampling interval. One solution is to generate samples with different intervals to train the network and use the time difference between the samples as one network input. An alternative is to train one network specialized in a certain sampling and to interpolate the input or the output sequences.

4.3 Radial basis function networks

In this section we introduce a radial basis function (RBFN) and generalized regression neural networks (GRNN). GRNN is a modification of RBFN which is better suited for regression estimation. Both networks have the advantage of natural interpretation of reliability estimates as presented at the end of this section. RBFN networks have been applied to a variety of applications, see, e.g., [4, 11, 176].

4.3.1 Classical radial basis function network

One approach to function approximation and regression problems is to use radial basis function networks. For neural networks they were first introduced by Broomhead and Lowe [16] (cited in [54, p. 236]). RBFN networks have been shown to be able to approximate any function to arbitrary accuracy by Hartman, Keeler and Kowalski [51], and by Park and Sandberg [125, 126] (cited in [13, p. 168]).

RBFN architecture

In radial basis function networks, a two-layer architecture is primarily used [13, p. 168]. The basic form of the network can be presented with the formula (see Figure 35)

$$x_1^{(2)} = y(x) = \sum_{j=1}^{N_1+1} w_j g_j \left(\left(x_2^{(0)} - \mu_j \right)^2 \right). \tag{55}$$

The activation function $g_j(x)$ is also called a *basis function* and μ_j is the *prototype*. The basis function gives excitation as a Euclidean distance between the input $x_1^{(0)}$ and the prototype. Extra activation $g_{N_1+1} = 1$ is used to include the bias term.

Equation (55) is a presentation of a one-dimensional regression approximation where the approximated function is a map from one-dimensional real space to another. The generalization for a multidimensional function $\mathbf{R}^n \rightarrow \mathbf{R}^m$ takes the form

$$y_k(x) = \sum_{j=1}^{N_1+1} w_{jk} g_j(||\mathbf{x} - \mu_j||), \tag{56}$$

where $k = 1, \ldots, m$ and $\mathbf{x}, \mu_j \in \mathbf{R}^n$. The most common form of basis functions is the Gaussian

$$g_j(x) = exp\left(-\frac{x^2}{2\sigma_j^2} \right), \tag{57}$$

where σ_j^2 introduces another free parameter for the basis function. It controls the smoothness properties of the function [13, p. 165]. If very small values of the σ_j^2 are used, then the resulting network function will act like a higher order polynomial. For large values, the network function presents a simple function, a line in the extreme case.

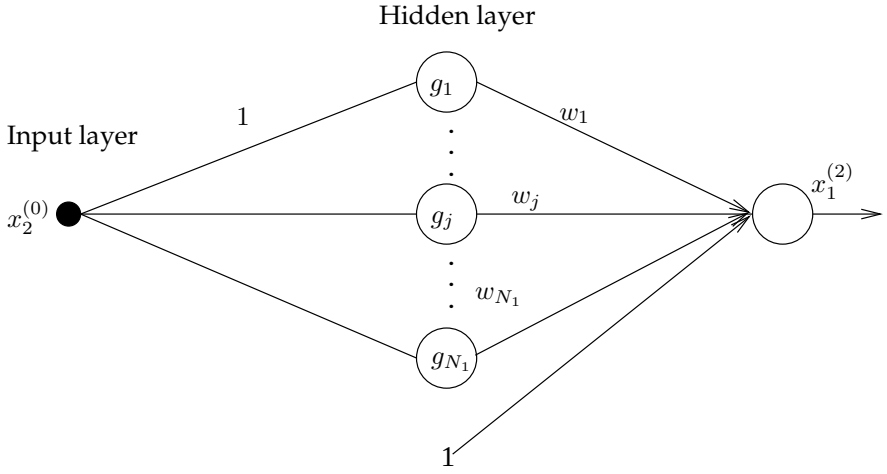

Figure 35: A two-layer radial basis function network.

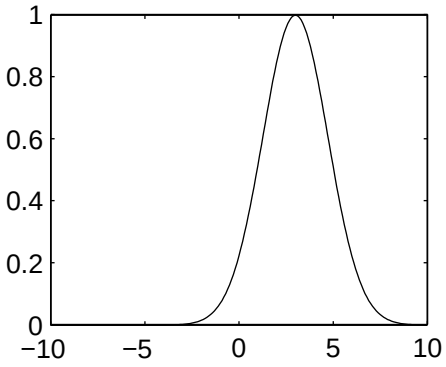

Figure 36: Normalized Gaussian activation, with $\mu = \sigma^2 = 3.0$.

Bishop [13, p. 165-176] presents other possible basis functions of the form

$$g(x) = (x^2 + \sigma^2)^{-\alpha}, \qquad \alpha > 0,$$

$$g(x) = x^2 \ln x,$$

and

$$g(x) = (x^2 + \sigma^2)^{\beta}, \qquad 0 < \beta < 1.$$

The simplest form of the activation is, naturally, the linear function

$$g(x) = x.$$

A common form of the Gaussian basis function [147, p. 422] is to use the normalized form:

$$\hat{g}_j(x) = \frac{g_j(x)}{\sum_{k=1}^{N_1} g_k(x)}, \tag{58}$$

where $g_j(x)$ is the basis function of equation (57).

To conclude, various basis functions can be chosen for different hidden units. In practice, however, the same basis functions are usually applied and the prototype, or *centre*, will be the variable to specialize hidden units in a specific input.

Learning

One big advantage with RBFN is the fast optimizing of the free parameters. The optimizing is done in two stages. During the first stage the basis function parameters μ_j and σ_j^2 are evaluated. This can be done in an *unsupervised* manner, where target outputs are not needed for the evaluation of the parameters. In *supervised* learning, the target outputs are used in the calculus. This is done at the cost of simplicity and nonlinear optimization strategies must be used. The benefit is a more accurate evaluation of the parameters. Notice that if the prototypes μ_j are known, then equation (56) will result in a linear system and the parameters w_{jk} can be solved with linear programming.

Unsupervised learning techniques for the basis function parameters

One simple approach to choose the prototypes for basis functions is to use a subset of the training data. The set can be chosen randomly. This is of course a fast approach, and easy to implement, but might give sub-optimal results. The σ_j can be set the same for all j and calculated as a multiple of the average distance between centres. Another approach would be to determine σ_j from the distance of the prototype to its L nearest neighbours.

Another unsupervised learning technique is to use clustering algorithms. An easy-to-implement batch-version of the *K-means clustering algorithm* [13, 110] can be used to evaluate centres for the basis function:

1. Choose K disjoint sets S_j randomly. Each set contains N_j data points.
2. Calculate the mean $\mu_j = \frac{1}{N_j} \sum_{k \in S_j} \mathbf{x}_k$, for all $j = 1, \ldots, K$.
3. Reconstruct each set S_j to have the nearest neighbours with respect to the distance $||\mathbf{x}_k - \mu_j||$. If some of the sets became different, then return to step two.

Yet another way to separate features of the data is to use the Kohonen network, also known as a self-organizing feature map [83, 147].

Supervised learning of the network parameters

For a one-dimension regression problem, using the Gaussian basis function of the form (57), and the sum of squared error E, we can solve the unknown basis function parameters with backpropagation following the negative gradient direction:

$$\begin{aligned} \triangle\sigma_j &= -\gamma \frac{\partial E}{\sigma_j} \\ &= -\gamma \sum_k (x(k+1) - y(x(k))) w_j \exp\left(-\frac{(x(k)-\mu_j)^2}{2\sigma_j^2}\right) \frac{(x(k)-\mu_j)^2}{\sigma_j^3} \end{aligned} \tag{59}$$

and

$$\begin{aligned} \triangle\mu_j &= -\gamma \frac{\partial E}{\mu_j} \\ &= -\gamma \sum_k (x(k+1) - y(x(k))) w_j \exp\left(-\frac{(x(k)-\mu_j)^2}{2\sigma_j^2}\right) \frac{(x(k)-\mu_j)}{\sigma_j^2}, \end{aligned} \tag{60}$$

where γ is the learning rate [13, p. 190-191].

The weight parameters w_j may be solved with backpropagation using the following equation [147, p. 423]:

$$\triangle w_j = -\gamma \frac{\partial E}{w_j} = -\gamma g_j((x(k)-\mu_j)^2)(x(k+1) - y(x(k))).$$

However, if the basis function parameters are estimated using unsupervised training, then equation (56) will result in a linear system and linear programming can be utilized. For more than one training pattern, on-line or off-line updates can be used.

4.3.2 A generalized regression neural network

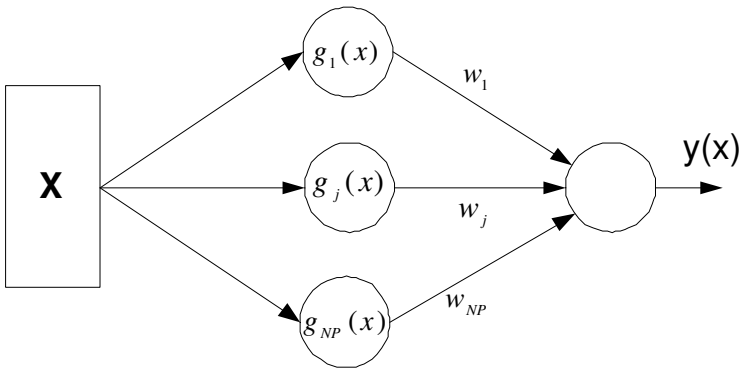

Figure 37: A generalized regression neural network.

Figure 37 illustrates the architecture of the generalized regression neural network [177] (cited in [101]). GRNN is basically a radial basis function network with a normalized linear output. The overall network output $y(\mathbf{x})$ with a given $N \times 1$-vector $\mathbf{x}$ is

$$y(\mathbf{x}) = \frac{\sum_{j=1}^{NP} w_j \cdot g_j(\mathbf{x})}{\sum_{j=1}^{NP} g_j(\mathbf{x})} + b, \tag{61}$$

where b is a bias term, NP the number of prototypes in the network and w_j are the network weights. The function $g_j(\mathbf{x})$ is defined as

$$g_j(\mathbf{x}) = \exp\left(-\frac{||\mathbf{x} - \mu_j||^2_{\mathbf{v}_j}}{2\sigma_j^2}\right) + \epsilon, \tag{62}$$

where μ_j is the jth prototype and σ_j^2 is the width parameter. The constant $\epsilon > 0$ is used to create a *forced activation*. The forced activation is introduced for two purposes: to prevent the denominator of the function in (61) to go zero and to compute an average of the network weights if not a single prototype is active. An alternative is to add the ϵ-constant to the denominator. Then, if none of the network prototypes is active, the overall output $y(x)$ will be equal to the bias.

Instead of Euclidean distance, we use a *weighted Euclidean distance* of the prototype μ_j and vector $\mathbf{x}$ as

$$||\mathbf{x} - \mu_j||^2_{\mathbf{v}_j} = \sum_{k=1}^{N} v_{kj}^2 (x_k - \mu_{kj})^2, \tag{63}$$

where v_j is squared to allow negative values. This is necessary because of the supervised training of the network introduced in the next subsection. The weighted Euclidean distance is a modification to the presentation of the GRNN in [177]. The weighting gives different inputs different emphasis in the calculus. This may be replaced by scaling the inputs to a desired value-range in unsupervised learning. This requires knowledge of the used inputs and their respective order. Assumption is that through supervised learning suitable weighting can be recovered empirically.

Supervised learning of the network parameters

K-means clustering algorithm can be used as an initialization for supervised learning techniques with GRNN. Next we will present the error function derivatives in respect to the network parameters. These formulas can be directly used with the backpropagation algorithm to iteratively find a local solution of the network parameters.

Let us first consider the case with a single sample presented to the network. The squared error of the sample is

$$E = \frac{1}{2}(y(\mathbf{x}) - t)^2. \tag{64}$$

The derivative of (64) respect to the network weight parameters w_j is given by the equation

$$\frac{\partial E}{\partial w_j} = r\frac{g_j(\mathbf{x})}{\sum_{k=1}^{NP} g_k(\mathbf{x})}, \tag{65}$$

where r is the residual of the sample:

$$r = y(\mathbf{x}) - t.$$

Notice that the second-layer parameters w_j may be linearly solved since equation (61) is a linear system if all but w_j are considered constant.

Next we define δ_j as

$$\delta_j = r\left(g_j(\mathbf{x}) - \epsilon\right)\frac{w_j - y(\mathbf{x}) + b}{\sum_{k=1}^{NP} g_k(\mathbf{x})}. \tag{66}$$

Now we can list the derivatives of the error function in respect to the remaining network parameters:

$$\frac{\partial E}{\partial b} = r, \tag{67}$$

$$\frac{\partial E}{\partial \sigma_j} = \delta_j\frac{||\mathbf{x} - \mu_j||^2_{\mathbf{v}_j}}{\sigma_j^3}, \tag{68}$$

$$\frac{\partial E}{\partial \mu_{ij}} = \delta_j v_{ij}^2\frac{x_i - \mu_{ij}}{\sigma_j^2}, \tag{69}$$

$$\frac{\partial E}{\partial v_{ij}} = -\delta_j v_{ij} \frac{(x_i - \mu_{ij})^2}{\sigma_j^2}, \tag{70}$$

where $\mu_j = [\mu_{1j}, \ldots, \mu_{Nj}]^T$ and $\mathbf{v}_j = [v_{1j}, \ldots, v_{Nj}]^T$.

Reliability of the network estimates

In Section 3.3 the concept of reliability and time domain corrections were introduced. Next, two intuitive heuristics for the reliability of the estimates produced by the GRNN are suggested. The concept of reliability for GRNN is comprehended as a measure of *localized firing intensity* in the network. This corresponds to the idea of local neurons that together map the whole input space but also act locally: it is assumed that there are no repeated prototypes and the prototypes are in hierarchical order, i.e., they have neighbours but are also more apart from the other neurons. Each neuron has a corresponding weight w_j, which stands for the overall output of the system if the input is equal to the corresponding prototype. Prototypes close to each other also fire similar weighting w_j.

If an input vector is distant to all prototypes, the total firing intensity of the network is lower compared to the familiar input. Hence, the GRNN reliability estimate is based on the mean firing intensity of the network:

$$rb_1(t) = \frac{1}{NP} \sum_{j=1}^{NP} g_j(\mathbf{x}_t), \tag{71}$$

where NP is the number of prototypes and $rb_1(t)$ is the reliability estimate for time instant t.

We assumed that the neurons act locally and give similar weighting w_j between similar prototypes. Thus, another reliability estimate would be the calculation of deviation of the prototype weights and network output:

$$rb_2(t) = \frac{\sum_{j=1}^{NP} g_j(\mathbf{x}_t)(w_j - y(\mathbf{x}_t))^2}{NP \sum_{j=1}^{NP} g_j(\mathbf{x}_t)}. \tag{72}$$

This can be read as a measure of similarity between those weights w_j that constitute most to the overall output. If the deviation is high, the locality assumption is invalid.

The two reliability estimates differ in their interpretation. The first measures the similarity of the input to the prototypes while the second is a measure of locality of the network output. The reliability concept gives a tool for the analysis of the trained GRNN; to investigate empirically how it utilizes its neurons. Reliability estimates constituted by the GRNN may also be utilized for time domain corrections, as is discussed in Section 3.3.

The generalized regression neural network, reliability estimates and time domain corrections are demonstrated later in Section 6.3.3, where they are applied to respiratory frequency detection strictly from the heart rate signal.

4.4 Optimization of the network parameters; improvements and modifications

In neural networks, finding network parameters which give the smallest training error and best fit is not a proper approach. There is a danger of encountering an *overfitting* of the data if nothing else than a small training error is of interest. Data may have noise in it. When the network reproduces the training set exactly, the noise will also be reproduced. What we are really interested in is good *generalization*. Generalization refers to the neural network producing reasonable outputs for inputs not encountered during training [54]. In Section 4.4.2 the network training is modified to avoid an overfit of the data and to avoid an over complex neural network architecture. Methods of weight decay, early stopping and training with noise are introduced.

In Section 3.2 the data preprocessing techniques were briefly presented, which may be used to improve network performance. For example, data scaling in Section 3.2.4, may be utilized to transform the network inputs and targets to the order of unity. FFNN has a linear transformation in the first layer, which is similar to the scaling procedure. If, however, the input and target scaling is not executed the network weights may have markedly different values from each other and this will result in problems, for instance, with the weight initialization [13, 87]. In addition to this, some classical improvements and modifications in network optimization are introduced in the next subsection.

An automated procedure to find the right network architecture does not exist. One must have some knowledge of the data and the network in advance. Even so, some algorithms are developed to find an optimal architecture. Often referred to in the literature are the growing (e.g., cascade correlation) and pruning algorithms (e.g., optimal brain damage, optimal brain surgeon). Growing algorithms include the model order selection during the training process. One simple method introduced by Bello [5] is to start with a few hidden units, train the network, and use the optimized weights as the initial weights for a larger network. An opposite approach to growing is to start with a large network and remove the weights or nodes which are less important. Pruning algorithms differ in the way of how the weights or nodes to be eliminated are selected [13, 52, 91, 143].

In this dissertation a different approach is chosen: the model is selected based on the evaluation of several local minima, which are locally optimal in respect to the error [87], with different initial conditions and network architecture, e.g., the number of hidden units and the number of inputs, and a cross-validation method presented in Section 4.4.3 to estimate the expected (general) error of the network.

4.4.1 Classical improvements to backpropagation convergence

There is plenty of literature describing various methods to make backpropagation converge better and faster. Unfortunately, these improvements only work in restricted applications and are not universal. In many cases standard backpropagation begins to perform better than its improvements after a certain level of complexity and size of the training set are achieved [147, p. 183].

Backpropagation with momentum

Rojas [147, p. 184] presents an improvement called *backpropagation with momentum*. The idea is to calculate a weighted average of the current gradient and the previous correction direction. This should help to avoid oscillations in narrow valleys of the error function. The updates in the backpropagation algorithm take the form

$$\triangle w_{ij}^{(l)}(k) = -\gamma \frac{\partial E}{\partial w_{ij}^{(l)}} + \alpha \triangle w_{ij}^{(l)}(k-1),$$

where γ is the learning rate and α a *momentum rate*. Both learning parameters affect convergence greatly and so they also become parameters which need to be optimized.

Adaptive step algorithms

It is not trivial to fix universal learning rates, or design an adaptive algorithm to find them. Too low learning rates will result in slow convergence of the algorithm. If the learning rate is too large, the optimization process can fall into oscillatory traps where updates "jump" over the optimum and soon turn back in the same direction only to get lost again. Adaptive approaches increase the step size whenever the error function decreases over several iterations. The step size is decreased when the algorithm jumps over a valley of the error function. In learning algorithms with a global learning rate, all weights are updated with this step size. In algorithms with local learning rates, a different constant is used for each weight. Depending on the information used to decide whenever to increase or decrease the learning rate, different algorithms are developed, for example Silva and Almeida's algorithm [160], Delta-bar-delta [65], the dynamic adaptation algorithm [153] and Rprop [145].

Offset terms in derivative

Exceedingly low derivatives in nodes can lead to a slow convergence. One solution is to force $|f'(x)| \geq \epsilon$. Another approach is to introduce an offset-term: $|\triangle f'(x)| = \epsilon$. However, using this approach raises the question as to what the training is based on, since the analytic gradient information is no longer valid and is manipulated.

Initial weight selection

One question is where the iterative learning process is started, i.e., what are the best initial weights to start with. Usually weights are taken randomly from an interval $[-\alpha, \alpha]$. Very small values of α paralyze training since the corrections will become very small. Very large values can lead to saturation of the nodes in the network and to flat zones of the error function, resulting in a slow convergence. Choosing the right α value is usually not a great problem and $\alpha = 1$ is used in many neural network software packages. Perhaps analyzing and learning to know one's data will give much better results when fixing α [147, p. 197].

Second-order algorithms

Second-order algorithms include more information about the shape of the error function than the mere value of the gradient. Newton's method is one example of a pure second-order algorithm. However, the problem with such an approach is the complexity when calculating inverse of the Hessian matrix of the error function. In *pseudo-Newton methods* this can be avoided using a simplified form of the Hessian. Other second-order algorithms are Quickprop [36] (cited in [147]) and QRprop [130, 129]. It is also possible to rework the standard backpropagation algorithm to use second-order information [13, 147]. The Hessian matrix for the feed-forward neural network is presented, for example, by Bishop [13].

A second order algorithm commonly used in MathWorks products (optimization and neural network toolboxes) is the Levenberg-Marquardt backpropagation. The algorithm uses an approximated Hessian matrix to update the network parameters [13, p. 290-292]. The use of Levenberg-Marquardt algorithm in neural network training is described in [48] and [47] (both cited in [101]).

4.4.2 Avoiding overfit of the data

Next we present, in more detail, three different approaches to prevent overfit, thus improving the generalization of a network with noisy data.

Penalising model complexity

A network with a high number of parameters may often result in overfit and poor performance. *Requlatisation* techniques [57] (cited in [13]) add a penalty term ω to the error function:

$$\tilde{E} = E + \alpha\omega.$$

Penalty parameter α controls the effect of the model complexity to the error function.

In *weight decay* regularization (a.k.a. *ridge regression*), the penalty term consists of the sum of squares of all the network parameters w_i:

$$\omega = \frac{1}{2}\sum_i w_i^2. \tag{73}$$

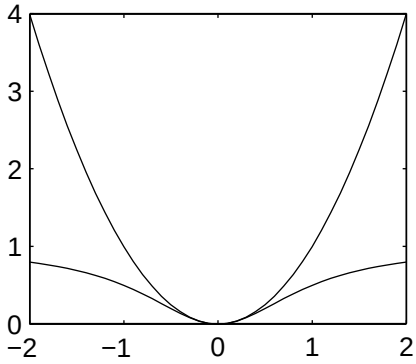

Figure 38: Decay terms of equations (73) and (74) ($\hat{w} = 1$) respect to a single weight w_i.

Since the central region of the sigmoid function is close to linear, the units give linear mapping for small values of weights, [13, p. 318-330]. If some of the units become linear during training, the overall network complexity will reduce (remember, the composition of linear units can be replaced with a single linear unit).

Weight decay given in equation (73) favours many small values of weight parameters rather than a few large ones. The following modification for the decay term

$$\omega = \frac{1}{2} \sum_i \frac{w_i^2}{\hat{w}^2 + w_i^2} \tag{74}$$

will help to avoid this problem [50, 90, 179] (cited in [13, p. 363]). The parameter $\hat{w}^2$ must be fixed in advance. Figure 38 shows how the decay terms of equations (73) and (74) behave. As can be seen, the function corresponding to formula (74) is non-convex, thus increasing the amount of local minima for the regularized error function [87].

Weight decay techniques penalize model complexity of neural networks. There are also other measures of complexity, e.g., the so-called information criteria, e.g., Akaike's (AIC), Bayesian (BIC), network (NIC) and deviance information criteria (DIC), which can be used for this purpose [171, p. 49-55].

Training with noise

One approach to avoid overfit is to add small noise to the training data. Heuristically, this could make it harder for the network to make an exact match of the data. In [12] it is shown by Bishop that training with noise is closely related to the technique of regularization.

Early stopping

In the early stopping method (see, e.g., [137]) the network is trained with many parameters. During training typically the sum of squared errors will decrease and an effective number of parameters, whose values differ sufficiently from zero will grow. However, at some point in the training, the generalization capacity of the network will start to decrease. With early stopping learning is ended in an optimum state where the generalization is at its best.

In practice the early stopping method is used with a *validation set*, a set of observations held back from the training and used to measure network performance. During training the network error is also calculated with the validation set. The training is stopped when a good match for the validation set is achieved. Then we expect to attain good data representation with the network.

4.4.3 Expected error of the network; cross-validation

The most common method for estimating a generalization error in neural networks is to reserve part of the data as a test set, which is not used during training. After training the test set is fed to the network. The resulting testing error will give us an estimate of the generalization error. The problem with this is that, from a training perspective, part of the data is lost. Cross-validation is also known as *split-sample* or *hold-out validation* [55, p. 213-218].

4.4.4 FFNN and FIR in matrix form: through training samples, forward and backward

Feed-forward neural network and the network derivatives have frequently been presented in matrix form as a single sample presentation (see, e.g., [147]). It appears that both the feed-forward neural network and finite impulse neural networks may be presented with the same compact matrix representation where the FIR network gives a feed-forward neural network as a special case if no tapped delays are present in the network. Furthermore, it will be shown that all the training samples may be included in the matrix presentation resulting in a simplified presentation of both the network output and weight gradients.

The advantage of the matrix presentation is the abandonment of overpopulation of indices in the network's forward and backward calculation. The matrix form improves the analytic presentation value and interpretation since much of the optimization and control theory is illustrated in the matrix form. Furthermore, the matrix presentation enables fast implementation with software packages supporting a matrix presentation, especially the Matlab programming environment, and software libraries available for common programming languages, such as Fortran, C and C++ dedicated to matrix computation.

Feed-forward

Let N_l present the number of units in the layer l and L the number of layers in the network. Thus, the number of inputs is presented as N_0, and N_L illustrates the number of units in the output layer. N_s is the number of (training) samples and K_l the number of tapped delays, FIR linear filters, in layer l. We may define the layer l network weights in a matrix form with the following equations

$$\mathbf{W}^l(k) = \begin{bmatrix} w^l_{11}(k) & w^l_{12}(k) & \dots & w^l_{1N_l}(k) \\ w^l_{21}(k) & w^l_{22}(k) & \dots & w^l_{2N_l}(k) \\ \vdots & \vdots & \ddots & \vdots \\ w^l_{N_{l-1}1}(k) & w^l_{N_{l-1}2}(k) & \dots & w^l_{N_{l-1}N_l}(k) \end{bmatrix} \in \mathbf{R}^{N_{l-1} \times N_l}, \tag{75}$$

$$\mathbf{B}^l = \overbrace{\begin{bmatrix} b_1 & \dots & b_1 \\ b_2 & \dots & b_2 \\ \vdots & \ddots & \vdots \\ b_{N_l} & \dots & b_{N_l} \end{bmatrix}}^{N_s} \in \mathbf{R}^{N_l \times N_s}, \tag{76}$$

where the transpose of the bias vector $[b_1 b_2 \dots b_{N_l}]^T$ is repeated N_s times in the matrix. This duplication of bias values is necessary when the bias is added through samples to the network excitation.

The excitation $\mathbf{S}^l$ and activation $\mathbf{X}^l$ of layer l with N_s samples is defined with the following equations

$$\mathbf{S}^l = \sum_{k=0}^{K_l} (\mathbf{W}^l(k))^T \mathbf{X}^{l-1}(k) + \mathbf{B}^l, \tag{77}$$

$$\mathbf{X}^l = f(\mathbf{S}^l), \tag{78}$$

where the function $f(\cdot)$ is the activation function, e.g., sigmoid, and the activation function is calculated for each element in the matrix. Thus, the matrix dimension remains unchanged.

The excitation and activation of layer l is dependent on the past K_l activations of layer $l-1$. The delayed activation in the matrix form may be constructed with a special matrix padded with as many zero elements in columns as there are delays in the activation $\mathbf{X}^l(k)$:

$$\mathbf{X}^l(k) = \begin{bmatrix} \overbrace{0 \ \dots \ 0}^{k} & x^l_{11} & x^l_{12} & \dots & x^l_{1(N_s-k+1)} \\ 0 \ \dots \ 0 & x^l_{21} & x^l_{22} & \dots & x^l_{2(N_s-k+1)} \\ \vdots \quad \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 \ \dots \ 0 & x^l_{N_l 1} & x^l_{N_l 2} & \dots & x^l_{N_l(N_s-k+1)} \end{bmatrix} \in \mathbf{R}^{N_l \times N_s}, \tag{79}$$

where k is from 0 to K_l expressing the delays of layer l and $\mathbf{X}^l(0) \equiv \mathbf{X}^l$.

The "zero-layer" is the input layer and therefore we may express the input vector with N_s samples as

$$\mathbf{X}^0(0) \equiv \mathbf{X}.$$

Furthermore, if the network output is linear, then the activation of the last layer L is equal to the excitation of the layer:

$$\mathbf{X}^L = \mathbf{S}^L. \tag{80}$$

Feed-backward, solving the network weight gradients

In case of a linear output, presented in (80), and mean-squared error between the output and training target vector $\mathbf{Y}$ defined in vector form as

$$\mathbf{E} = \frac{1}{2N_0N_s}\sum_{n=1}^{N_s}||\mathbf{X}_n^L - \mathbf{Y}_n||^2, \tag{81}$$

we may solve the backpropagation error matrices δ^l through N_s samples with the following equations:

$$\delta^L = \mathbf{X}^L - \mathbf{Y}, \tag{82}$$

$$\delta^l = \sum_{k=0}^{K_{l+1}} f'(\mathbf{S}^l) \cdot (\mathbf{W}^{l+1}(k)\delta^{l+1}(k)), \tag{83}$$

where $f'(\cdot)$ is the activation function derivative processed for each element in the matrix $\mathbf{S}^l$. The multiplication between the derivative and total backpropagation error from the layer $l+1$ is executed element-by-element for the equal-sized matrices, which results in an unchanged matrix dimension.

The delayed backpropagation error $\delta^l(k)$ is a reduction of a matrix $\delta^l(0) \equiv \delta^l$, defined with the following zero-padded matrix

$$\delta^l(k) = \begin{bmatrix} \delta^l_{1(k+1)} & \cdots & \delta^l_{1N_s} & \overbrace{0 & \ldots & 0}^{k} \\ \delta^l_{2(k+1)} & \cdots & \delta^l_{2N_s} & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ \delta^l_{N_l(k+1)} & \cdots & \delta^l_{N_lN_s} & 0 & \ldots & 0 \end{bmatrix}. \tag{84}$$

The weight and bias derivatives $D\mathbf{W}^l(k)$ and $D\mathbf{B}^l$ may now be given as

$$D\mathbf{W}^l(k) = \frac{1}{N_0N_s}\mathbf{X}^{l-1}(k)(\delta^l)^T, \tag{85}$$

$$D\mathbf{B}^l = \frac{1}{N_0N_s}\overbrace{[1\ldots1]}^{N_s}(\delta^l)^T, \tag{86}$$

where the N_s length row vector $[1\ldots1]$ contains ones. Hence, the vector matrix multiplication in (86) adds the backpropagated errors δ^l for each training sample in layer l, and the result is the gradient bias row vector of length N_l.

Discussion

The matrix implementation requires more memory than conventional programming of the forward-backward computation since all the computational stages for backpropagation error, excitation, activation and activation derivatives must be stored for each sample and each neuron. Clearly, without matrix storage, the forward and backward stages may be performed in loops storing only the sum of the weight gradients when running the whole training sequence through the backward calculation.

In the matrix presentation, the number of computational operations is not increased and, therefore, the computer implementation for systems optimized for the matrix calculation may give good performance. However, the presentation is not given justice if only the implementation performance is considered. The easier interpretation may become important in a theoretical analysis. Furthermore, for most of the practical applications memory usage is not considered a problem [88].

A question arises as to whether more networks could be presented in this manner, giving network forward and backward calculus and FIR- or feed-forward neural networks as a special case. This could be a situation with networks with local feedback, generally referred as locally recurrent neural networks (LRNN) or local feedback multilayer networks. The base of these models lies in the adaptation of ARMA model in the network. FIR network is also a LRNN [19]. A comprehensive theoretical foundation for different temporal network architectures enabled for the matrix treatment is left for future research.

4.4.5 Backpropagation alternatives

A variety of improvements for neural network training have one certain outcome: the number of alternatives makes the decision of the right procedure complex. Since network training is basically an optimization problem it could also be treated as one. There are general nonlinear optimization programs that use different methods and algorithms depending on the problem and available extra information, e.g., if approximations of the gradient or Hessian exist. For example, Matlab Optimization Toolbox offers general functions for multivariate constrained and unconstrained nonlinear smooth- or non-smooth optimization [102]. However, a general ”best” approach for nonlinear programming does not exist and the choice between various methods depends on the application.

The use of the general optimization program simplifies network training. Only network output and, optionally, gradient evaluation are required. Gradient or Hessian information decreases the calculation time but is optional, since the program may use finite differencing, later presented in this section, to numerically approximate function derivatives. In addition, the numerical derivatives may be used to verify the analytic derivatives.

With a general optimization solver we can construct controversial network architectures much easier, e.g., hybrid models or use the network as an inner func-

tion, as presented in Section 5.2, since the emphasis is no longer in complicated neural network optimization: the minimum requirement is to provide cost function with error between the (continuous) function estimates, such as neural network, and target samples.

Numerical gradients

For common network architectures the analytic gradients are available. For example Campolucci [19] presents a signal-flow-graph approach to solve network gradients for family of neural networks called locally recurrent neural networks. Also the Jordan-network, FIR network and feed-forward neural network belong to this family of networks. However, if the network is heavily modified for a specific application, the resolving of analytic gradients may become time consuming. Some authors also prefer non-gradient methods or a combination of non-gradient and gradient methods as a general approach to find a "global"[15] solution for a nonlinear problem.

If the approximated function is continuous and thus, in theory, has analytically solvable derivatives, a numerical estimate of the gradient may be produced with *finite differencing*. A derivative of a function $f(t)$ is defined with the following formula:

$$f'(t) = \lim_{t \to 0} \frac{f(x+t) - f(x)}{t}. \tag{87}$$

The numerical gradient estimate for a parameter is assessed by estimating equation (87) with small value for t, which in computer programs is related to ϵ-precision.

The above derivative is solved with forward differencing. Naturally backward differencing may be applied and in addition central differencing is defined with the following formula

$$f'(t) = \lim_{t \to 0} \frac{f(x+t) - f(x-t))}{2t}. \tag{88}$$

Forward/backward differences require one and central differencing two extra function evaluations *for each parameter*. Thus, numerical derivatives requires extra computing time compared to the use of analytic derivatives. The applicability of the method depends on the complexity of the problem, i.e., the number of parameters and samples. Numerical derivatives may also be used to verify the analytic derivatives.

Genetic algorithms

Another popular optimization strategy is introduced by a variety of genetic algorithms, also applicable for neural network optimization. The basic principle in

[15]Generally the global optimum is difficult to prove. However, global optimization conditions and solutions may be find, e.g., to convex or linear problems.

genetic algorithms is to have a population of solution candidates for the problem, where the best candidates are kept and modified, combined or perturbed to produce new candidates (see, e.g., [94, 107, 124, 173]). Genetic algorithms are natural for non-smooth problems where the approximated function is not continuous.

Nelder-Mead simplex method

A general unconstrained and non-smooth nonlinear problem solver in Matlab uses Nelder-Mead simplex method to minimize an object function. The method is applicable to problems with a small number of parameters and may handle discontinuity if it does not occur near the solution [89] (cited in [102]).

Constrained optimization

With a general nonlinear constrained optimization solver it is possible to introduce constraints in neural networks optimization. Constraints could be used, for example, to restrict the function range or to build a strictly increasing neural network function.

However, since the constraints often make the optimization plane more complex, it becomes harder to find a suitable local solution. An alternative would be to resolve multiple local solutions with unconstrained optimization and afterwards use constraints to select a valid local minimum.

5 HYBRID MODELS

A hybrid model is a system where the system output is formed by several models, different, e.g., in their structure or parameters. For example, neural networks often interpolate well between the training points but the extrapolation maybe completely undesirable. More precisely, the underlying phenomena to be modeled by a network can increase naturally but the extrapolation shows a decrease in values. Another example is the estimation of some natural system with positive output space but the estimation results in negative values in the input boundaries.

Another observation is that different methods may work well within a certain input-target space. Observed phenomena may be linear in some regions and nonlinear in others, or the system dynamics change depending on the input space. This raises the question, if different models could be specialized in different regions in the input-target space.

In neural network literature the hybrid models combining different *expert functions* to one overall output are called *committee machines*. An individual expert function is a model, e.g. a neural network, that is specialized in a specific input space. Haykin presents an introduction to committee machines in [55, Chapter 7]. He divides different approaches of the expert function combination to static and dynamic structures. In the dynamic structure the input signal affects the expert combination unit while in the static structure the combination mechanism does not involve the input signals.

Furthermore, the static structure category includes *ensemble averaging* [184, 115, 128, 183] and *boosting methods* [35, 34, 38, 39, 40, 41, 155, 156, 157] (all cited in [55]). In both methodologies the integration function is a linear combination of different experts. In ensemble averaging experts are trained with the same data, for example with different initial conditions, while in boosting the experts are trained on data sets with different distributions. Bishop [13, p. 364-369] uses the concept *committees of networks* equivalent to ensemble averaging.

The dynamic structure category by Haykin contains a *mixture of experts* [118, 66, 67] (cited in [55]) and *hierarchical mixture of experts*, where the latter is a generalization of the first method. Dynamic methods include training of the entire committee with the input depended integration function. The system is supposed to divide (and conquer) the input space and experts to a modular network[16], where the integration unit "decides" which experts should learn which training patterns. The experts are expected to specialize in simple tasks and are formed as simple linear units. The nonlinearity is achieved with the *gating network* constituting the in-

[16] In [123] (cited in [55]) a modular network is defined as follows: "A neural network is said to be modular if the computation performed by the network can be decomposed into two or more modules (subsystems) that operate on distinct inputs without communicating with each other. The outputs of the modules are mediated by an integrating unit that is not permitted to feed information back to the modules. In particular, the integrating unit both (1) decides how the outputs of the modules should be combined to form the final output of the system, and (2) decides which modules should learn which training patterns".

put space depended weighting of the experts (for details see [55, p. 368-369]). The parameters for the experts and integration function are searched in the same optimization routine with gradient descent or exceptation-maximization approach (EM-algorithm) [32] (cited in [55]).

The committee machines share and improve some theoretical results and properties of single neural networks [55]:

1. Committees are universal approximators.
2. The bias of the ensemble averaged function is the same and the variance is less than the one achieved with a single neural network.
3. Suppose that three experts have an error rate of $\epsilon < 1/2$ with respect to the distributions on which they are individually trained (boosting algorithm). Then the overall error rate of the committee machine is bounded by $g(\epsilon) = 3\epsilon^2 - 2\epsilon^3$ [155].

The straightforward optimization of free parameters for dynamic structures results in multiple experts being "mixed" or "switched between" in order to map the input-target space. Hence, modularity is not achieved. Tang et al. [169] describes a procedure to enhance the mixture of experts by applying classification methods, e.g., self-organized maps [83], to divide the input space and to feed the "correct" inputs to the individual experts.

Next we will present a general method to construct a hybrid model. The proposed approach is not intended to be used solely with neural network expert functions, but rather with any individual models to be combined. The method is closely related to the one presented in [169] since it is a compromise between the static and dynamic committee machine categories: the integration function is optimized with the input-target-space but the expert functions are not part of the optimization. The expert functions may be formed with applying different initial conditions to training or using different subsets of the data as described with committee machines with static structures. Furthermore, input space mapping may also include the output space of the expert functions. It is assumed that not only the input space but also the output space can be utilized in the integration unit. Moreover, the algorithm will be applied to form a reliability measure of the overall output as well as a time domain correction for the modeled time series. It will also be illustrated how to define a cost function in optimization to prevent the mixing of the expert functions.

In Section 5.2 a new concept and method called *transistor network* is introduced. It is also a hybrid model where the neural network will be used as an inner function in a larger system. The architecture may be utilized for neural network optimized adaptive filtering introduced in Section 5.2.1.

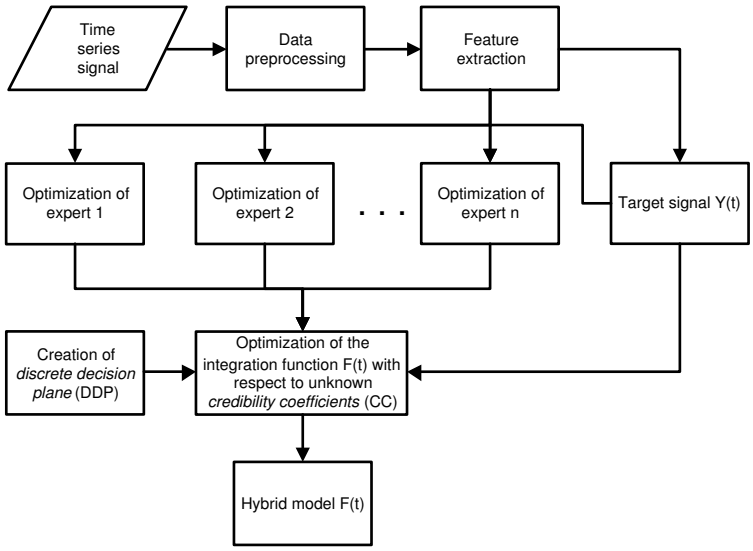

Figure 39: A flow chart illustrating the overall view and optimization steps required by the HMDD. Optimization of experts $f_k(t)$ is isolated from the integration function $F(t)$. Expert functions may differ, e.g., in the way they are preprocessed, modeled, trained (optimized), etc. The integration function combines different experts with respect to the credibility coefficients of the discrete decision plane.

5.1 A hybrid model with discrete decision plane

Next we suggest an optimization strategy for a limited input-target space to construct a hybrid model designed for time series modeling. The system contains credibility of each expert model corresponding to its input-output mapping and gives a discrete decision plane for each expert function. We assume that each expert is capable of forming it's own mapping.

Figure 39 illustrates the overall view and optimization steps required by the *hybrid model with discrete decision plane* (HMDD). It is emphasized that the integration function and experts are optimized in separate steps to preserve the modularity.

5.1.1 General presentation of the HMDD

A *discrete decision plane* (DDP) of the HMDD is defined as

$$\mathbf{A}_k = \begin{bmatrix} a_{k11} & a_{k12} & \dots & a_{k1L_k} \\ a_{k21} & a_{k22} & \dots & a_{k2L_k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{kM_k1} & a_{kM_k2} & \dots & a_{kM_kL_k} \end{bmatrix}, \ \mathbf{a}_{ki} = \begin{bmatrix} a_{k1i} \\ a_{k2i} \\ \vdots \\ a_{kM_ki} \end{bmatrix}, \ \mathbf{b}_k = \begin{bmatrix} b_{k1} \\ b_{k2} \\ \vdots \\ b_{kL_k} \end{bmatrix}, \qquad (89)$$

where the matrix $\mathbf{A}_k$ defines the *discrete coordinates* (DC) of the system, and vector $\mathbf{b}_k$ the corresponding *credibility coefficients* (CC). L_k is the *number of credibility coefficients* (#CC) and M_k the dimension of the discrete coordinates for the kth model. Furthermore, the integration function $F(t)$ producing the final output of the general HMDD reads as:

$$F(t) = \frac{\sum_{k=1}^{N} e^{g_k(\mathbf{x}_k(t))} \cdot f_k(t)}{\sum_{k=1}^{N} e^{g_k(\mathbf{x}_k(t))}}, \tag{90}$$

where N is the number of experts and $f_k(t) \in \mathbf{R}$ is the output of an expert k at time instant t. The exponential transformation in (90) is used to keep the relative weighting of each expert function positive. Another possibility for example, is to use the sigmoid function in (41), which would give a more natural interpretation of the weights, as the transformation would result in real numbers inside the interval $[0, 1]$. The relationship between $g_k(\mathbf{x}_k(t))$, discrete coordinate $\mathbf{a}_{ki}$, and credibility coefficients $\mathbf{b}_k$ is defined as

$$\begin{aligned} i = \underset{i \in \{1,\ldots,L_k\}}{\arg\min} \; ||\mathbf{x}_k(t) - \mathbf{a}_{ki}||, \\ g_k\left(\mathbf{x}_k(t)\right) = b_{ki}. \end{aligned} \tag{91}$$

Hence, the discrete decision plane is defined for all model-wise reference points $\mathbf{x}_k(t) \in \mathbf{R}^{M_k}$, also between or outside the defined discrete coordinates defined with the matrix $\mathbf{A}_k$ in (89).

The discrete coordinate system may be set by hand based on the knowledge of the modeled system. An alternative is to search for a suitable division with a clustering algorithm, e.g., SOM [83] or K-means clustering introduced in Section 4.3.1. Notice that the discrete coordinates may be different for each expert. In addition, the expert inputs and outputs, the discrete coordinates and the credibility coefficients are connected in time, but the coordinates do not necessary have to include the expert inputs or outputs. The construction of the coordinates may be based on any division; the only requirement is that each expert output $f_k(t)$ for each time moment t can be unambiguously connected with a discrete coordinate $\mathbf{a}_{ki}$ and the respective credibility coefficient b_{ki}.

The credibility coefficients $\mathbf{b}_k$ are the free parameters of the system optimized with supervised learning. The derivatives of the credibility coefficients with respect to an objective function is presented later in the section.

Notice that if we define DDP by means of expert outputs and set $L_k \equiv 1, \; \forall k$ in (89), then the system describes one setup for ensemble averaging. Hence, output of the model is a weighted average of the expert outputs and the normalized weights are the free parameters of the system.

5.1.2 Deviation estimate of the HMDD

We may construct a *deviation estimate* of the hybrid model in equation (90), based on the deviation between the hybrid model output $F(t)$ and experts $f_k(t)$,

weighted with the discrete decision plane:

$$rb(t) = \frac{\sum_{k=1}^{N} e^{g_k(\mathbf{x}_k(t))} \cdot (F(t) - f_k(t))^2}{\sum_{k=1}^{N} e^{g_k(\mathbf{x}_k(t))}}. \quad (92)$$

Hence, the deviations between experts may be estimated as a weighted distance between each expert and the HMDD at each time instant. Notice that expert outputs with a large variance result in to higher deviation estimates. Thus, the deviation may not be absolutely interpreted but is expert dependent.

If the hybrid model produces high deviation estimates, the model may only use its extra parameters to combine the experts to decrease the overall error. This may suggest a rejection of the hybrid model, or just the time moments with a high deviation estimate, since the improvement of the error is only due to free parameters and expert combination, instead of modularity. Thus, the expert deviation may be interpreted as a reliability estimate of the hybrid model. Deviation estimate may also be applied to postprocessing with time domain corrections presented in Section 3.3. The transformation of the deviation estimate to the reliability measurement may be executed in several manners, for example with the exponential scaling or linear transformation presented in equations (31) and (32). Transformation is required, for example, to invert the deviation estimates close to zero to reliability of one.

The applicability of the deviation estimate can be evaluated by correlation between the expert deviations and squared model estimate residuals $r(t)^2 = (F(t) - Y(t))^2$ or absolute residuals $|r(t)|$. Here $Y(t)$ presents the target signal. Positive correlation suggests that in time instants where deviation is high the residuals will also increase. High deviation is a result of expert combination, i.e., there is no single expert that would give a distinctive output in this target-space region. Also if the residuals are high in these time instants then the combination of the experts does not result in lowered residual. Negative correlation can suggest that the hybrid model is able to reduce the overall error by combining the expert outputs. Notice, however, that the correlation is not an exact measure in this concept and may only be used to guide the analysis.

5.1.3 Optimization of the credibility coefficients

Determination of credibility coefficients is realized using a gradient descent algorithm with supervised learning. The error function E is defined as follows:

$$E = \frac{1}{2} \sum_{t=1}^{T} (F(t) - Y(t))^2 + w \cdot \sum_{t=1}^{T} rb(t), \quad (93)$$

where $F(t)$ and $rb(t)$ are defined in (90) and (92). Here w defines a regularization parameter to control the effect of the deviation estimate $rb(t)$ on the optimization. The deviation estimate contributes the idea of penalizing mixing of experts in the overall model. This concept is close to the idea of penalising model complexity of a neural network discussed in Section 4.4.2.

Derivative of the error function E respect to the credibility coefficients is given by the following equation:

$$\frac{\partial E}{\partial b_{ki}} \equiv \frac{\partial E}{\partial g_k\left(\mathbf{x}_k(t)\right)} = \frac{e^{g_k(\mathbf{x}_k(t))}\left(f_k(t) - F(t)\right)\left(r(t) + w \cdot \left(f_k(t) - F(t) - rb(t)\right)\right)}{\sum_{m=1}^{N} e^{g_m(\mathbf{x}_k(t))}}, \tag{94}$$

where the relationship between $g_k\left(\mathbf{x}_k(t)\right)$ and b_{ki} is given in (91).

Smoothing of the credibility coefficient derivatives

The interpretation of the discrete decision plane may be improved by smoothing the credibility coefficient derivatives in (94). If the discrete coordinates are stored in a vector arranged in equidistant and increasing order, then a moving averaging can be utilized to smooth the corresponding derivatives. This is illustrated with the following equation:

$$\widehat{\frac{\partial E}{\partial b_{ki}}} = \sum_{j=i-\frac{N}{2}}^{i+\frac{N}{2}} h_N(j) \frac{\partial E}{\partial b_{k(i+j)}}, \tag{95}$$

Here $h_N(\cdot)$ is a symmetric window, for example the Hanning window, with N nonzero samples. Generalization to multi-dimensional coordinate system requires smoothing window to be defined to a multi-dimensional space, for example, as a weighting diminishing as a function of Euclidean distance from the observed coordinate.

The procedure results in a smoother discrete decision plane. This may improve the interpretation of the DDP and, furthermore, enhance generalization of the hybrid model. It may also affect those coordinate positions that are inactive with the current data, by directing the respective passive credibility coefficients towards their neighbours. The credibility coefficients connected with such coordinates are recommended to be interpolated or set to some pre-defined constant, as the supervised learning leaves them to their initialized values.

Notice that the derivatives solved in (94) are no longer valid, when smoothing is used. In practice, however, a solution of the optimization problem is found, as the smoothing only perturbs the derivatives. Direct smoothing of the coordinate coefficients is not recommended since it will result in a suboptimal solution. The smoothing approach will later be demonstrated with an example.

5.1.4 Deterministic hybrid model

The optimization of general HMDD results in several local minima, and thus to many solutions depending on the random initialization of the credibility coefficients. An alternative deterministic heuristic is to calculate the model errors at each coordinate to decide the best expert. To generate such *deterministic hybrid model*, the experts have to share the same coordinate system.

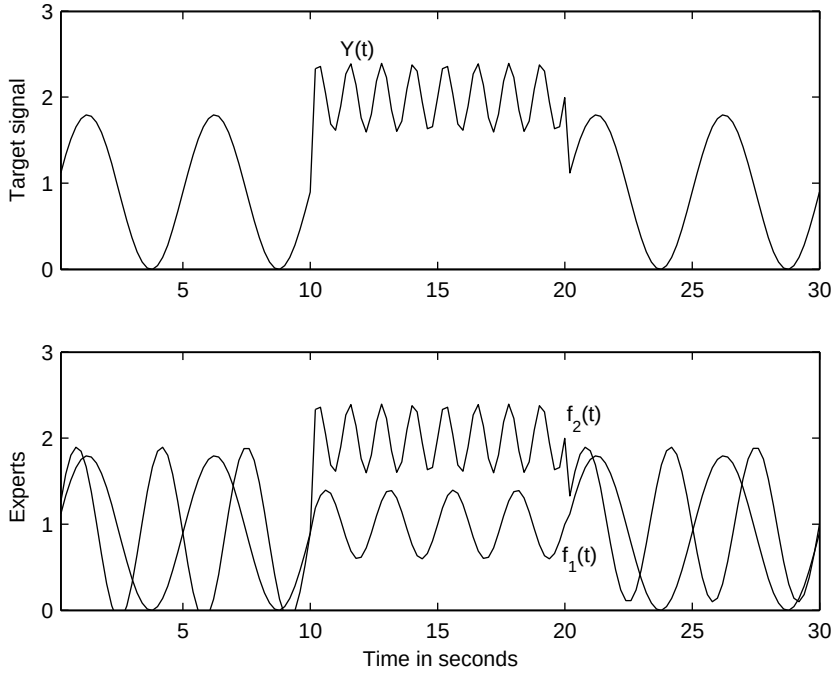

Figure 40: The target signal and expert functions of the hybrid model.

The deterministic hybrid is an example of hard-decision integration function. It cannot use the decision plane to declare compromise between the experts: only a single model will determine the output for each time instant. Furthermore, the deviation estimate in (92) may not be applied, neither the reliability corrections.

5.1.5 An example of hybrid models optimized to output space mapping

The HMDD optimized to output space mapping is a special case of the general HMDD. The model is obtained by setting $M_k \equiv 1$ and $\mathbf{x}_k(t) \equiv f_k(t)$. Thus, the discrete coordinates are defined based on the output range of the expert functions.

Figure 40 demonstrates the use of the hybrid model which is optimized to output space mapping. The target data is a combination of two expert function outputs $f_1(t)$ and $f_2(t)$. Table 3 presents the outcome of hybrid models optimized and postprocessed with different approaches. Two- and three-dimensional discrete decision planes ($M_k = 1$ or $M_k = 2$) were experimented. The resulting HMDDs resulted in a total of 62 and 512 credibility coefficients, respectively. In addition, the effect of the regularization parameter w was tested. The mean-squared errors and correlations between the squared deviation estimates and model residuals are presented. The various models are analyzed in the following subsection.

	$\#CC = 62, M_k = 1$				
	R_0	R_1	R_2	R_3	R_4
w=0.0	0.0178/0.3389	0.0103/0.2222	0.0117/-0.0042	0.0094/0.1307	0.0084/-0.0196
w=0.3	0.0297/0.0716	0.0194/0.2765	0.0468/0.3148	0.0197/0.2826	0.0232/0.3544
Derivative smoothing	0.0257/-0.0185	0.0146/-0.0333	0.0257/-0.0185	0.0146/-0.0333	0.0146/-0.0333
	$\#CC = 512, M_k = 2$				
	R_0	R_1	R_2	R_3	R_4
w=0.0	0.0067/0.8491	0.0061/0.4178	0.0006/0.3079	0.0052/0.2854	0.0042/-0.0017
Deterministic integration	0.0322	0.0183			

Table 3: Results of various HMDD optimized to output space mapping to estimate the signal with the two expert functions presented in Figure 40. The abbreviations $R_1 - R_4$ presents the results of different time domain post-correction heuristics applied to the HMDD output. R_0 expresses the MSE/C_P between the model output and target without any post-correction. The Pearson's correlation, C_P, is calculated between the squared model residuals and deviation estimates of the HMDD. R_1 corresponds to the results obtained in moving averaging of the model output with a Hanning window of length three. R_2 is a time domain correction with interpolation method, presented in Section 3.3.4, and threshold value 0.01. R_3 and R_4 presents the outcome of the reliability weighted moving average time domain correction in (34), with a length three Hanning window, and exponential and linear transformations of the deviation estimate, presented in (31) and (32). In addition, w presents the regularization parameter in (93).

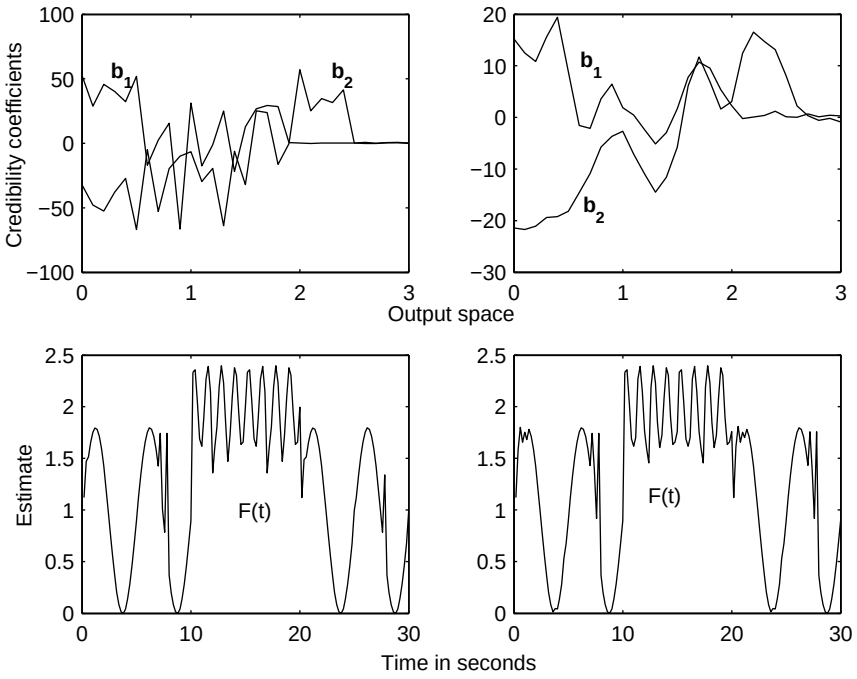

Figure 41: The upper figures present the two-dimensional discrete decision planes of the hybrid model and bottom figures estimate the corresponding model estimates. The left column is optimized without derivative smoothing while the right column is optimized with derivatives smoothed with a five point Hanning window $h_5(\cdot)$.

Hybrid models with two- and three-dimensional discrete decision planes

Figure 41 illustrates the results with a HMDD optimized with and without smoothing of the derivatives in (94). The two-dimensional DDP had discrete coordinates set as follows: $\mathbf{a}_1 = \mathbf{a}_2 = [0\ 0.1\ 0.2\ \ldots\ 2.8\ 2.9\ 3.0]$. Thus, the number of credibility coefficients for one expert was 31 and total was 62. As may be verified from the right column of the figure and Table 3, the smoothing results in a higher estimation error but more of a construed discrete decision plane. Figure 42 illustrates the corresponding deviation estimates of the HMDD presented in (92). As may be noticed the deviations become larger in the target space areas where the two experts interact and the decision between the models is not possible.

Notice that in (90) the credibility coefficients are transformed with an exponential function. Thus, the negative credibility coefficients illustrate output regions where an individual expert has little or no weight in the overall result, as the weighting approaches zero.

It appears from Figure 42 that the deviation estimates with smoothing of the derivatives are smaller but become active more often compared to the opti-

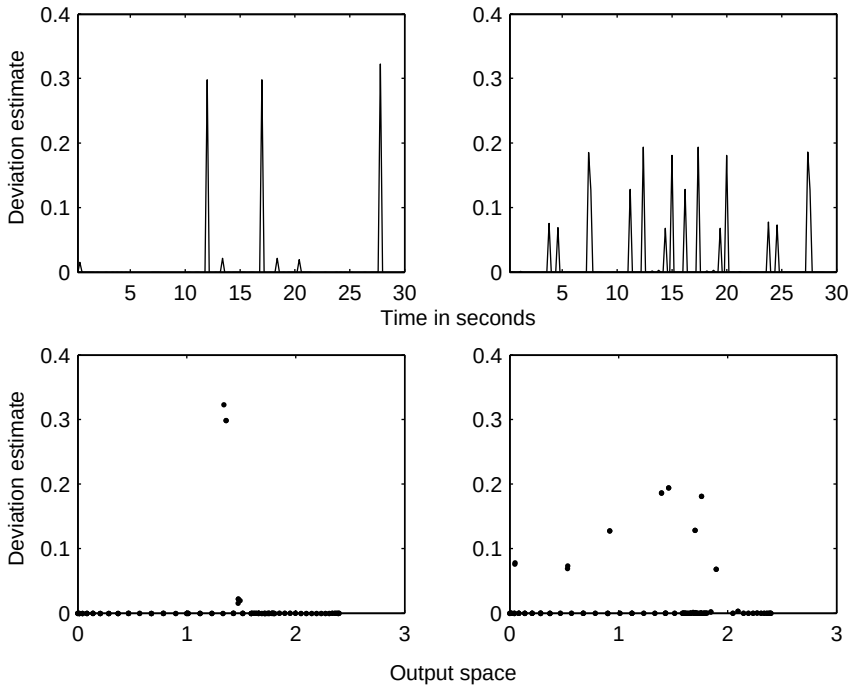

Figure 42: The upper figures presents the deviation estimates of the HMDD with two-dimensional DDP as a function of time while the bottom figure illustrates the scatter plot between the output space and deviation. The left column is optimized without smoothing of the derivatives and the right column is optimized with derivatives smoothed with a five point Hanning window.

mization without derivative smoothing. The result is clear when the credibility coefficients presented in the right column of Figure 41 are observed, e.g., in the region between 1.3 and 2.0. In this particular region the two models intersect. This is also verified with the scatter plot in Figure 42. This demonstrates how smoothing does not decrease the deviation but rather separates the experts in order to yield a more interpretable presentation. As may be compared from Figure 41, the right column, is somewhat more clear to interpret: the output-space below 1.3 is mapped by the first expert while the middle part from 1.3 to 2.0 is combined by both models. The remaining part is modeled by the second expert. The described division is harder to apply for the left column in Figure 42.

The hybrid model reliability may also be visually interpreted from the credibility coefficients. If the target space is separated into distinct regions where at each region only one expert has positive credibility coefficients while others remain negative, the hybrid model is well founded. Furthermore, the deviation will also be close to zero.

Figure 43 presents the outcome of the example optimized with deviation

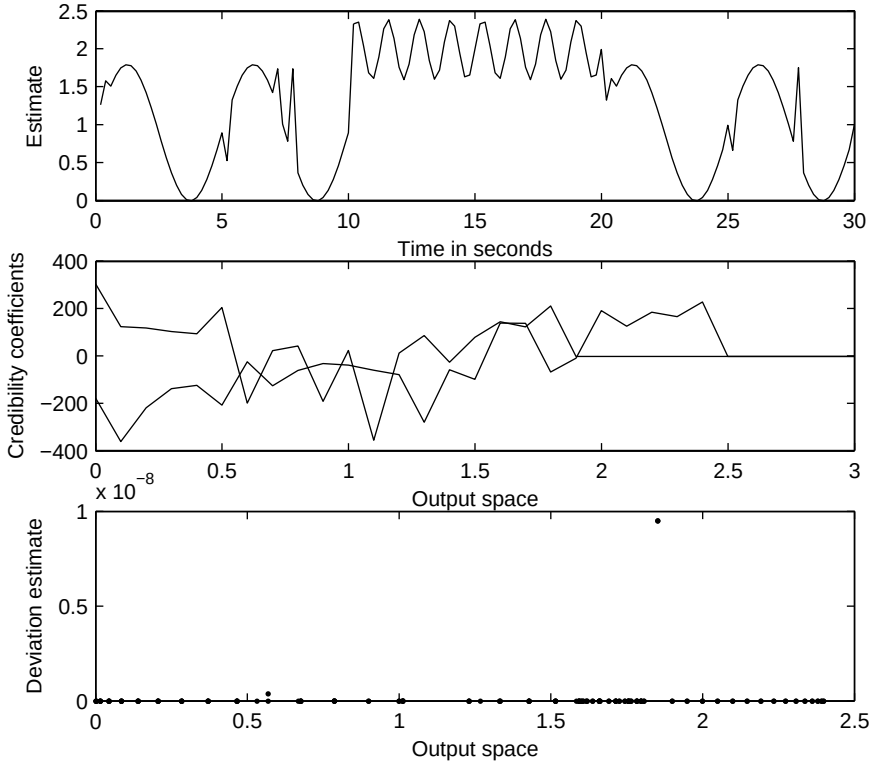

Figure 43: The upper figure presents the result of a hybrid model optimized with deviation term in the error function and regularization parameter set to $w = 0.3$. The middle figure presents the corresponding two-dimensional discrete decision plane. The bottom figure illustrates the scatter plot between the output space and the deviation estimates.

term and regularization parameter $w = 0.3$. As may be compared, the two-dimensional decision plane is more distinct and expert deviation has decreased. However, this is done at a cost of increased error of the estimate.

Figure 44 presents the outcome of a deterministic hybrid in a three-dimensional discrete decision plane. Now both expert outputs are used to determine the DDP for both experts. The discrete coordinate matrix in (89) is the following:

$$\mathbf{A}_1 = \mathbf{A}_2 = \overbrace{\begin{bmatrix} 0 & 0 & \dots & 0 & 0.2 & 0.2 & \dots & 3.0 & 3.0 \\ 0 & 0.2 & \dots & 3.0 & 0 & 0.2 & \dots & 2.8 & 3.0 \end{bmatrix}}^{256}$$

Thus, the resulting discrete decision plane has a total of 512 credibility coefficients.

Figure 45 illustrates the corresponding HMDD. The decision planes in Figure 44 are very distinctive but the model error is higher (MSE=0.0322) compared

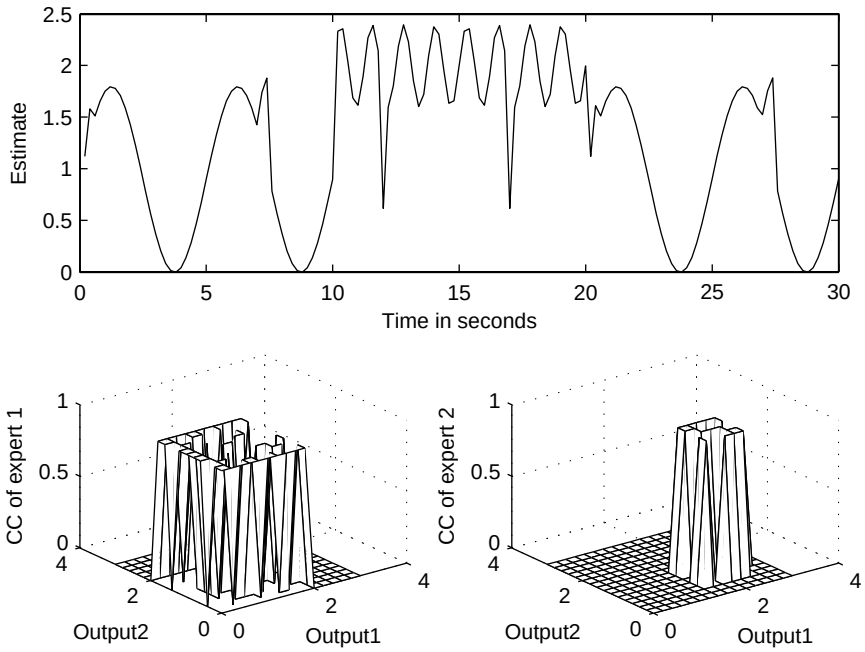

Figure 44: The upper figure presents the deterministic hybrid model estimate of the signal in Figure 40. The bottom figures illustrate the three dimensional discrete decision planes.

to those achieved with the HMDD (MSE=0.0067).

Table 3 reveals that the correlations between squared residuals and deviation estimates are a good quantitative measure of the model quality and mixing of experts. The hybrid models optimized with regularization parameter and derivative smoothing result in a small deviation, but also to uncorrelated deviation estimates and residuals. Hence, the time domain corrections using reliability information are unable to improve the model, and pure moving averaging results in better outcome. Furthermore, the models optimized without any attempt to control mixing of the experts resulted in improved estimates with the reliability information. The there-dimensional DDP with interpolation time domain post-correction outperforms other estimates.

5.1.6 Mixing of the expert functions

Next we will demonstrate the problem of expert function mixing also recognized in [169]. In the mixture of experts model it is assumed that the expert functions will self-organize to find a suitable partitioning of the input space so that each expert does well at modeling of its own subspace. The procedure is called divide and conquer [55]. The presumption is that the expert functions will learn to map a specific input space and the integration function $F(t)$ will only combine the results

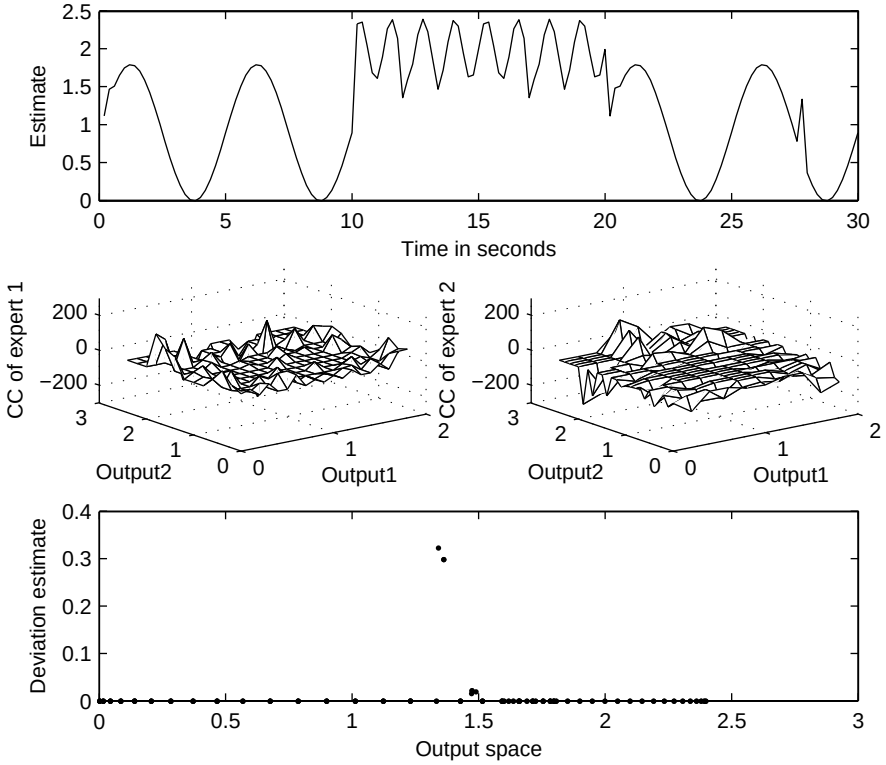

Figure 45: The upper figure illustrates the HMDD estimate of the signal in Figure 40. The middle figures illustrate the discrete decision plane and the bottom figure the hybrid model deviation estimate. The correlation between the absolute residuals and the deviation estimate was 0.9262. The mean-squared error between the estimate and target signal was 0.0067.

emphasizing the correct experts according to the input. Even if the parameters of the expert functions and the integration function are optimized simultaneously it is expected that they will act apart.

A HMDD optimized to one dimensional input space mapping is defined with the following parameterizations respect to the general HMDD:

$$M_k \equiv 1,\ \mathbf{x}_k(t) \equiv t.$$

In the following demonstration the expert functions are all linear:

$$f_k(t) = a_k t + b_k.$$

The general solution of the credibility coefficient derivatives in (94) are applied and the simultaneous optimization of the parameters a_k and b_k of the expert func-

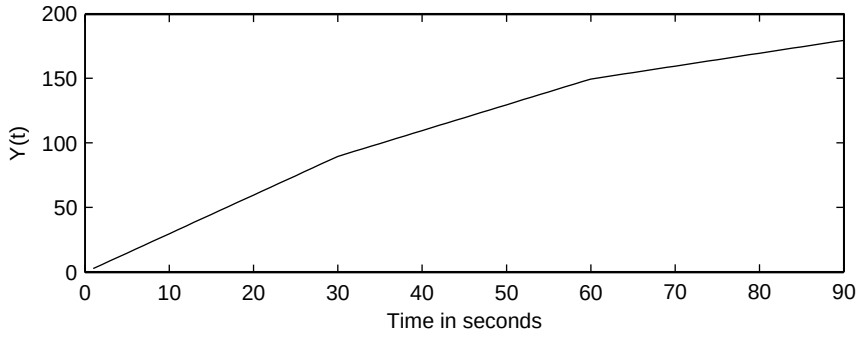

Figure 46: A piecewise-linear function defined in (97).

tions is performed with the following equations:

$$\begin{aligned} \frac{\partial E}{\partial a_k} &= \frac{t \cdot e^{g_k(t)} \left(r(t) + 2w\left(F(t) - a_k t - b_k\right)\right)}{\sum_{m=1}^{N} e^{g_m(t)}}, \\ \frac{\partial E}{\partial b_k} &= \frac{e^{g_k(t)} \left(r(t) + 2w\left(F(t) - a_k t - b_k\right)\right)}{\sum_{m=1}^{N} e^{g_m(t)}}. \end{aligned} \tag{96}$$

For this example we define the target as follows (see Figure 46):

$$Y(t) = \begin{cases} 3t, & t > 0 \ \wedge \ t \leq 30 \\ 2t + 30, & t > 30 \ \wedge \ t \leq 60 \\ t + 90, & t > 60 \ \wedge \ t \leq 90 \end{cases} \tag{97}$$

The discrete coordinates are defined as $[0\ 10\ 20\ \ldots\ 90]$ for all experts.

A HMDD was searched with 4648 local minima, where optimization started with different initial conditions and regularization parameters. The results are presented in Table 4. The best fit resulted in MSE=4.5124 (see Figure 47) with regularization parameter $w = 0$. The deviation term was utilized to force better separation of the input space between the three experts. Notice the increase of error and decrease of the mean deviation as a function of the regularization parameter.

As may be verified from Figure 47 and Table 4 the overall result is not as expected with a clear division of the input space and separation of the experts. In-

MSE	w	Mean deviation	a_1	b_1	a_2	b_2	a_3	b_3
4.5124	0	5.2e+002	1.8	37.0	0.9	18.5	3.3	-3.2
12.9762	1	2.0e-001	1.8	31.4	1.5	52.9	3.0	-0.4
13.1909	5	1.8e-003	2.3	14.7	3.0	0.2	1.5	53.6
13.2200	10	4.3e-005	1.5	52.2	2.9	0.3	3.0	0.2
13.4549	20	5.8e-007	3.0	-0.3	1.5	52.0	2.9	1.9

Table 4: Optimal results of a HMDD, total of 4648 local minima were searched.

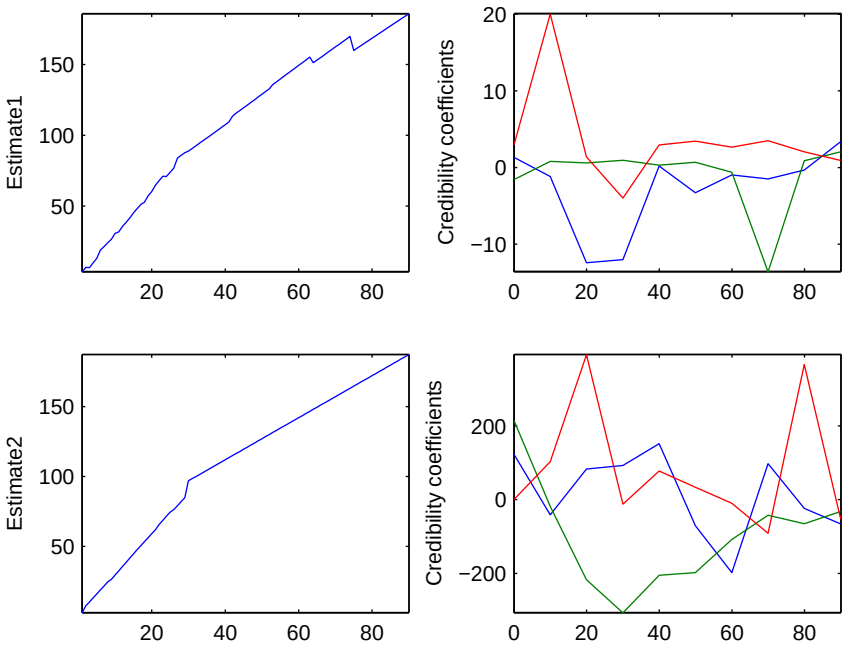

Figure 47: Two estimates of the piecewise-linear function defined in (97) modeled with a HMDD. The upper Figure is optimized with deviation term excluded and the bottom Figure with a deviation term and $w = 20$ as the regularization parameter. The left column illustrates the model estimates and the right column the corresponding discrete decision planes.

stead the optimization without deviation term results in a combination of experts. The complicated optimization plane due to several parameters will not result in the desired solution with the deviation term either.

This simplified example illustrates that the optimal solution of a complicated nonlinear system may not be managed with divide and conquer. The gradient descent algorithm can not differentiate between different parameters in the optimization plane but just finds an optimal solution regardless of the architecture of the system, especially, when there are too many free parameters compared to the complexity of the phenomenon.

5.1.7 Generalization capacity of the HMDD

The integration function in (90) is basically a weighted average of the expert outputs. Hence, the model may not extrapolate outside the boundaries defined by the experts. Furthermore, the integration may allow a combination of the experts based only on extra parameters the model introduces. Thus, the number of credibility coefficients affects not only the accuracy, but also the generalization capacity of the HMDD. To find proper balance between the number of credibility coeffi-

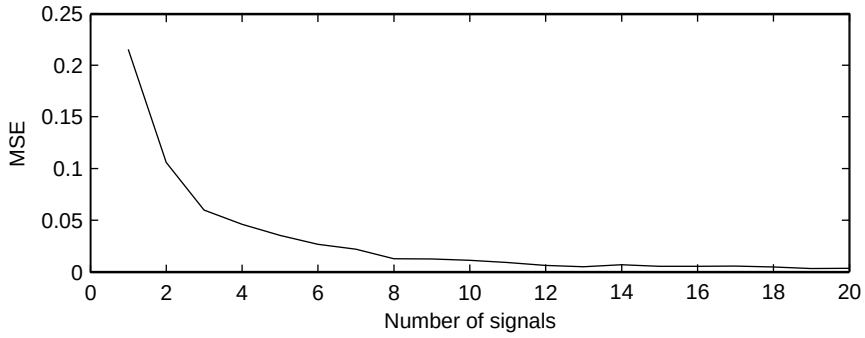

Figure 48: A mean-squared errors of a different number of random signals estimating a random target signal. All the signals are drawn from an interval $[0, 1]$ and they are uncorrelated. The example demonstrates how the HMDD begins to make convex combinations to reproduce the target signal, even when the signals are random.

cients and generalization, the discrete decision plane has to be post-analyzed.

To further investigate the problem with generalization, we present the following example. Consider multiple random uncorrelated time series signals that are used to estimate another random target signal. If the signals are drawn from the same interval and overlap with the target, the hybrid model can decrease the overall error as a function of the number of experts. This is demonstrated in Figure 48, where a hundred point long random signals drawn from uniform distribution in the interval $[0, 1]$ is used to estimate target random signal from the same function range. For illustrative purposes the optimizations are performed separately for a different number of signals to demonstrate the decrease of the hybrid model error. In this way the increase in the number of experts may be exploited to decrease the error of the HMDD. However, the hybrid model will not give as good MSE with a new data.

5.1.8 Summary

The current section introduces a general concept of constructing hybrid models with a discrete decision plane. Heavy assumption of the model is that, for example the input or output space, can be utilized in the integration function and the discrete decision plane will divide the expert functions to act separately in different regions of input-output space of the experts.

In Kettunen et al. [77] the preferred embodiment of respiratory frequency detection from the heart rate signal included HMDD optimized to output space mapping, where different time-frequency features were combined to decrease overall error. The examples presented in this section demonstrated the potential of the HMDD with an artificial dataset. Furthermore, several general properties of the HMDD were outlined:

1. Ensemble averaging, HMDD optimized to output or input space mapping are special cases of the general HMDD.

2. Also a deterministic hybrid model with hard-decision integration function was presented. The deficiency of the model is that it may not fully utilize the information loaded in the experts. Neither can any reliability measures or corrections be exploited. The benefit of the model is a very distinct decision plane that is easy to interpret. Also optimization of the model is straightforward and unambiguous for the given dataset.

3. The integration function and experts are optimized in separate steps to preserve modularity of the hybrid. This premise was demonstrated to be correct in Section 5.1.6. A simple example demonstrated how the optimal solution of a nonlinear system may not be managed with divide and conquer, if the integration function and expert functions are optimized simultaneously.

4. A deviation estimate of the HMDD may be utilized to control modularity of the system. It may also be interpreted as a reliability estimate and used in time domain post-corrections of the model output. However, care must be taken in the exploitation of the deviation estimate as it should not be interpreted as absolute.

5. Validity of the reliability estimates should be evaluated. Some possible heuristics were introduced, e.g., studying the correlation between the squared model residuals and deviation estimates. In addition, the post-correction methods may be indirectly used to evaluate validity of the reliability estimates. For example, if reliability weighted moving averaging decreases the model error, it should be compared to pure moving averaging with the same window template.

6. The number of extra parameters HMDD includes, affects on the accuracy and generalization capacity of the model. As a warning example, it was demonstrated how a HMDD can combine random signals on a desired target. In addition, other deficiencies were linked with the HMDD: as the integration function is basically a weighted average of the expert function outputs, the model may not extrapolate outside the boundaries defined by the experts. The integration may also allow a combination of the experts based only on extra parameters the model introduces resulting in poor generalization.

   Hence, it is emphasized that the discrete decision plane has to be post-analyzed with, e.g., visualization and studying deviation estimates of the HMDD. Also cross-validation may be utilized to estimate the generalization capacity of the model.

7. Smoothing of the credibility coefficient derivatives may be exploited to enhance generalization of the HMDD. Smoothing may also affect the inactive

coordinate positions, by directing otherwise passive credibility coefficients towards their neighbours.

8. Including a deviation term in optimization, and smoothing of the derivatives, are both modifications that are made in the cost of increased training error. However, instead of the training error, the overall testing error should be examined, as it gives a better estimate of the model quality and generalization.

5.2 A transistor network; a neural network as an inner function

A common usage for a neural network in time series analysis is to form a model by optimizing it with respect to a target signal. In the optimization process the network input is fed to the network and the output is directly compared to the target signal and, e.g., a squared error between the network output and the target is calculated. Furthermore, the error function is used to update the network parameters towards the negative gradient directions of the parameters. For example, for the feed-forward neural network, the derivatives in respect to the error function may be solved with the backpropagation algorithm presented in Section 4.1.3.

Now let us consider a system where the network is calculated several times with a different input to form the overall output. This system will be called a transistor network. A transistor network output is defined as follows:

$$G = \mathcal{F}(g(\mathbf{x}_1), \ldots, g(\mathbf{x}_K)),$$

where the function $\mathcal{F}$ is the integration function gathering K network outputs to a single instance. The vector $\mathbf{x}_k$ illustrates the kth input of the neural network g. We will limit the inspection to real valued network and integration function. Furthermore, to simplify the notation we will illustrate the results for a single input-target sample. The analysis may be extended to several samples, multiple outputs in the network output layer and to vector valued integration function.

In electronics a transistor is a circuit that transmits the electricity forward after enough power is supplied into it. A gate is launched when enough tension is reached. The analogy is apparent as the transistor network fires only after enough inputs are fed to the neural network. The neural network g receives several inputs and produces several outputs before the actual output G is produced with the integration function $\mathcal{F}$.

The applicability of the transistor network is shown especially in Section 6.3.2, where a transistor network is utilized for the respiration frequency detection from the heart rate time series. The general concept of adaptive filtering is presented in Section 5.2.1. To our knowledge, the concept of a transistor network is new and may find other useful applications in the future.

Next we will show the general solution of the derivatives of the transistor network parameters. Let us consider a single transistor output G and target Y.

The squared error (divided by two) is defined as

$$E = \frac{1}{2}\left(G - Y\right)^2 .$$

The derivative of the neural network parameter w_{ij}^l in layer l respect to the error function E may be calculated as follows:

$$\begin{aligned}\frac{\partial E}{\partial w_{ij}^l} &= (G-Y)\frac{\partial G}{\partial w_{ij}^l} \\ &= (G-Y)\sum_{k=1}^{K}\frac{\partial G}{\partial g(\mathbf{x}_k)}\frac{\partial g(\mathbf{x}_k)}{\partial w_{ij}^l} \\ &= \sum_{k=1}^{K}\frac{\partial G}{\partial g(\mathbf{x}_k)}\frac{\partial E}{\partial w_{ij}^l}\left(k\right),\end{aligned} \tag{98}$$

where we have used the chain rule once. Furthermore, we use the identity

$$\frac{\partial E}{\partial w_{ij}^l}\left(k\right) = (G-Y)\frac{\partial g(\mathbf{x}_k)}{\partial w_{ij}^l}, \tag{99}$$

which includes the derivative of the network function g in respect to the parameter w_{ij}^l for the kth input. Thus, the general solution in (98) may be exploited by utilizing the traditional backpropagation derivatives resulting in a simple analytic solution for the problem.

The formulation in (99) appears simple but yet a sophisticated application may be built around it as will be shown in Section 6.3.2. The basic principle of the derivative solution is powerful as may be verified from equation (109), where quite a complex objective function is derived based on equation (99).

5.2.1 A neural network optimized adaptive filter

Next an optimized adaptive filter is constructed with a neural network to modulate a discrete time-frequency distribution. The general procedure may be applied to optimize the time-frequency plane for respiratory frequency detection from the heart rate time series. This application is later presented in Section 6.3.2. The benefit of this approach is its capability to utilize the information controlling the filter's adaptation inside the neural network architecture. A special ingredient of the method is the use of the neural network as an inner function inside an instantaneous frequency estimation function. Thus, the described system is a transistor network introduced in the previous section. Furthermore, the procedure results in a relatively small number of network parameters processing large number of inputs to form a single output.

Digital filter design with a neural network has also been demonstrated in [9]. The article mainly focuses on designing a FIR filter for a desired amplitude response and is not frequency moment optimized filter as the system presented here.

In this method the time-frequency distribution is presumed to be positive and it must contain at least one non-zero element per time instant. Furthermore, for each time instant t there exists a target frequency $y(t)$ we wish to estimate.

A neural network function $g(k,t)$ is used to weight the time-frequency distribution[17]. It may include time- and frequency depended variables, but at least discrete frequency variables $w(k)$ in its input. In general, the time depended variables are utilized to modify the filter shape: they form the adaptive part of the neural network. For example, a filter shape depending only on a frequency instant $w(k)$ would be defined as $g(k,t) = g\left(w\left(k\right)\right)$. Thus, the neural network would have a single input $w(k)$ at frequency instant k.

A neural network weighted TFRD is defined as follows

$$\hat{F}(k,t) = F(k,t)^{g(k,t)}, \tag{100}$$

where the neural network g has a linear output and the discrete time-frequency matrix $F(k,t)$ is computed with T time instants and K frequency bins. Alternative weighting is given by equation

$$\hat{F}(k,t) = g(k,t)F(k,t), \tag{101}$$

which may be applied when the network g has sigmoidal output resulting in positive weighting inside the interval $[0\ 1]$.

The relationship of weighting to digital filtering is as follows: digital filtering (for example FIR filtering) is used to remove specified frequency components of the time series signal. The number of parameters in the filter specifies the sharpness of the filtering. Thus, a small number of parameters results in lowered amplitude of the nearby frequencies, if the spectrum of the signal is analyzed. In addition, e.g., FIR filtering results in an amplitude response to the interval $[0\ 1]$.

In a similar manner, the weighting of the TFRD or a single spectrum may be comprehended as a filtering of the signal. However, the manipulated spectrum and the corresponding amplitude response may often be impossible to re-create in the time domain, as the FIR filter with similar, complicated, amplitude response may not be generated. As the weighting defined in (100) has filter response outside the interval $[0\ 1]$ the interpreting is liberal. However, the weighting is still used to diminish signal frequency components not of interest. Thus, we will refer the weighting procedure as filtering of the signal.

Discrete instantaneous, or mean, frequency of weighted TFRD $\hat{F}(k,t)$ is defined as (see equation (4))

$$\hat{f}_{MEAN}(t) = \frac{\sum_{k=1}^{K} w(k)\hat{F}(k,t)}{\sum_{k=1}^{K} \hat{F}(k,t)}. \tag{102}$$

Similarly, mode frequency of weighted TFRD at time instant t, is defined as

$$\hat{f}_{MOD}(t) = \{w(k);\ \arg\max_{k} \hat{F}(k,t)\}. \tag{103}$$

[17]The notation $g(k,t)$ should be read as a description of a network function which the input is varied depending on the frequency and time instants.

Equations (102) and (103) together with (100) and (101) form the transistor network and optimization problem is to solve the unknown network parameters in respect to, e.g., mean-squared error (divided by two) between the target $y(t)$ and the frequency moment estimate $\hat{f}(t)$:

$$E = \frac{1}{2T} \sum_{t=1}^{T} \Big(\hat{f}(t) - y(t) \Big)^2 , \tag{104}$$

where the frequency moment estimate $\hat{f}(t)$ may be, e.g., $\hat{f}_{MEAN}$ or $\hat{f}_{MOD}$. For mode frequency analytic derivatives in respect to the error function do not exist, since it results in a discontinuous function and non-smooth optimization problem.

In the procedure, the neural network function g is inside the object function in (104) and it is calculated K times for each time instant. Only the network inputs containing frequency information vary. Time depended variables remain and change only when new time moments are estimated. Hence, the benefit of the architecture is a small number of network parameters compared to the number of time-frequency variables it processes.

In principle, any neural network architecture may be applied for the method. In case of temporal dynamics the time depended neurons need special focusing. Temporal neurons or connections should remain their state inside the inner loop while the frequency information is processed. Time depended states should be allowed to change only after the algorithm moves to a new time instant.

Figure 49 illustrates one possible overall view of the system. The target and input time series signals are first both preprocessed, e.g., outliers and artifacts are removed. For validation signal the target frequency is revealed to construct a supervised learning setup. Time-frequency distribution of the input signal is calculated and time, frequency and time-frequency features are extracted. Notice that several distributions may be utilized with different parameterizations but only one is optimized and weighted. Probably, a more complicated system could be formed with the decision function between different distribution estimates.

For one time instant the network feed-forward stage is repeated K times with varying frequency and time-frequency features, but with constant time depended features. The resulting K-length vector is used to weight the time-frequency distribution $F(\cdot, t)$ resulting in filtered spectrum $\hat{F}(\cdot, t)$ (see equations (100) and (101)). The instantaneous frequency of the spectrum is calculated with, e.g., frequency moments defined in (102) or (103). The result is the instantaneous frequency estimate of the system.

Off-line optimization process also requires output of all time instants, and the network is run $K \times T$-times total. For on-line updates the network gradient is updated every Kth feed-forward stage.

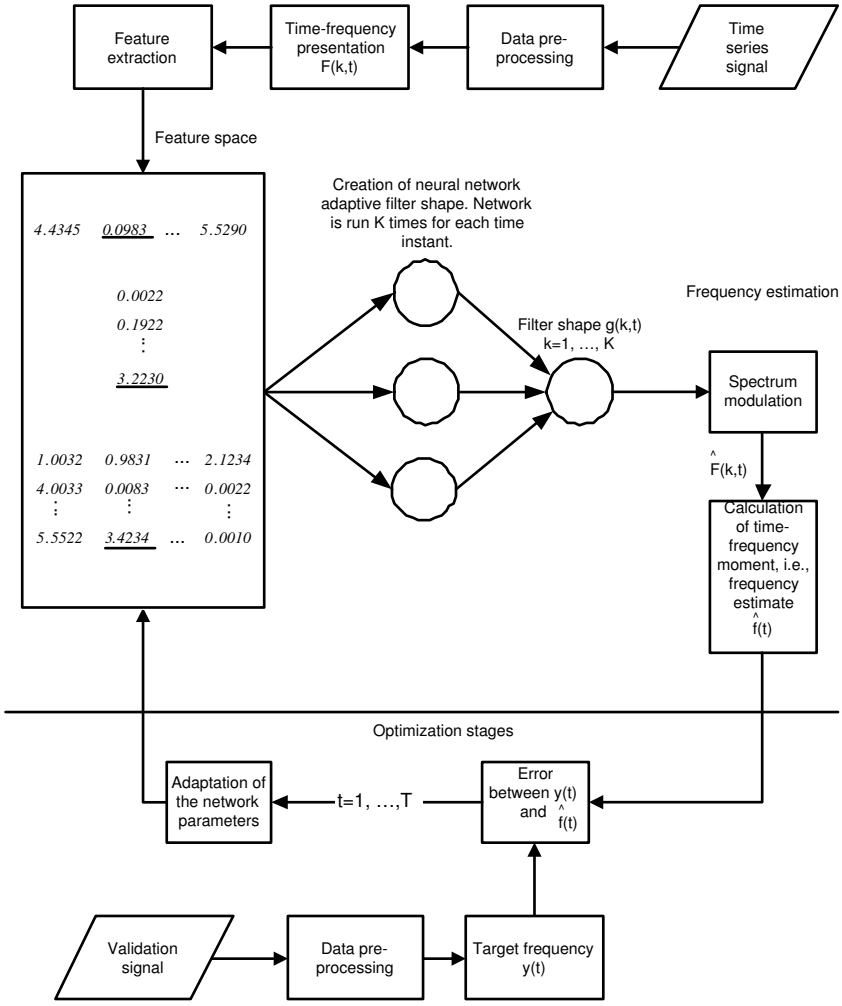

Figure 49: A flow chart illustrating the system view of neural network optimized adaptive digital filtering.

6 APPLICATIONS

In Section 6.1 we demonstrate with a large heterogeneous dataset that there may exist a high correlation between training and testing errors, even with a large number of network parameters. The neural network may use its neurons as "memory" to repeat different RRI patterns.

Two neural network based physiological models are presented in sections 6.2 and 6.3. The first application presents a dynamic neural network applied to modeling of excess post-exercise oxygen consumption while the second introduces the detection of breathing frequency strictly from the heart rate time series with a transistor neural network.

6.1 Training with a large dataset; correlation of training and testing error

Large datasets may become problematic for optimization routines. Second order algorithms often require more memory than basic backpropagation and the memory usage is proportional to the training set sizes. Also calculation time increases as more samples are introduced as the number of function calculations increases in the optimization algorithm.

Nevertheless, the use of large datasets is required for certain applications, e.g., if we wish to cover interindividual laws from physiological signals and represent them with good generalization. If we only use part of the data, then some dynamics and individual information may be missed. Another benefit with large datasets could be that the signal-to-noise ratio may be improved for certain data sets. The signal noise may start nearing the Gaussian distribution with zero mean as more data is introduced. This might lead to better generalization since the network does not bias towards nonzero error.

In the following experiment the RR intervals of the orthostatic tests were used to train a feed-forward network with five input units and one hidden layer. Only one person's data was used. The number of orthostatic tests was 51 each lasting 8 minutes. The number of input units was not optimized and was based on intuition. This modeling scene is very challenging; the network should learn and remember the past RRI sequence $x(t-5), \ldots, x(t-1)$ to predict the next RRI $x(t)$. It is not necessarily clear that any deterministic patterns exist. The hypothesis is that the number of hidden units may be used to increase the neural network's "memory". We do not believe that there is any system that can be modeled, rather there might exist some repeating patterns that can be memorized by the network.

We trained the network with seventy percent of the data with one to forty hidden units, the rest of the data, thirty percent, was used for testing. Validation set and early stopping were not used since the amount of data is heterogeneous and large enough (total of 26162 RR intervals) to prevent overfitting. The training

was repeated ten times for each case resulting in 400 training sessions. The full experiment took one week of computer time.

The feed-forward network was trained with Levenberg-Marquardt back-propagation. The ending criteria for the network training were set as follows: the MSE goal was 0.001 and maximum number of *epochs* was set at 300 to decrease overall computation time. One epoch means the training of the network with the entire data once. Sigmoid units were used in the hidden layer and linear unit in the output.

Notice that in the more general framework when cross-validation together with early stopping is used, the Levenberg-Marquardt based optimizing strategy might lead to poor generalization. It gives a fast convergence even in one algorithm step that might lead to overfitting of the data.

The results

Figure 50 presents the best training and testing records as a function of the number of hidden units. Also the histograms of the training sessions are presented. It appeared that in 365 out of 400 training sessions the mean-squared error reached a value less than 3400. For the test data 350 out of 400 resulted in MSE less than 4600. A few training sessions resulted in MSE higher than 10000 showing the failure of the local optimizing strategy. These "outliers" appeared randomly and were unrelated to any specific network architecture.

Increasing the number of hidden units decreased the training and testing errors. This suggests that the hypothesis in the beginning was justified: adding more "memory", i.e., hidden units in the architecture, results in better performance with both the testing and training data.

Figure 51 presents statistics of the best test data fit. In addition, the corresponding MSE of the training data is illustrated in the top left corner of the figure. To have a better interpretation the mean-squared errors are scaled to a minimum of zero and maximum of one.

The mean absolute value of the network parameters (or weights) as a function of the number of hidden units is presented in the middle of Figure 51. Only the networks resulting in the best testing error are included. The "netmean" describes the effective number of parameters. As can be seen the minimum test error is achieved with 31 hidden units where the "netmean" is high locally compared to the surroundings.

The scatter plot between the number of network parameters and "netmean" presented at Figure 51 does not show any clear pattern. However, with few parameters the effective number of parameters is relatively higher with less variance in interpretation.

The most clear result is presented with the scatter plot between the testing and training mean-squared errors at figure 51. The relationship is very linear suggesting that the optimizing strategy without early stopping seems justified: a good training performance results in a relatively good test data fit.

The average training time measured as epochs is presented in the bottom left corner of Figure 51. The number of epochs seemed quite arbitrary and did not decrease or increase as a function of the number of hidden units.

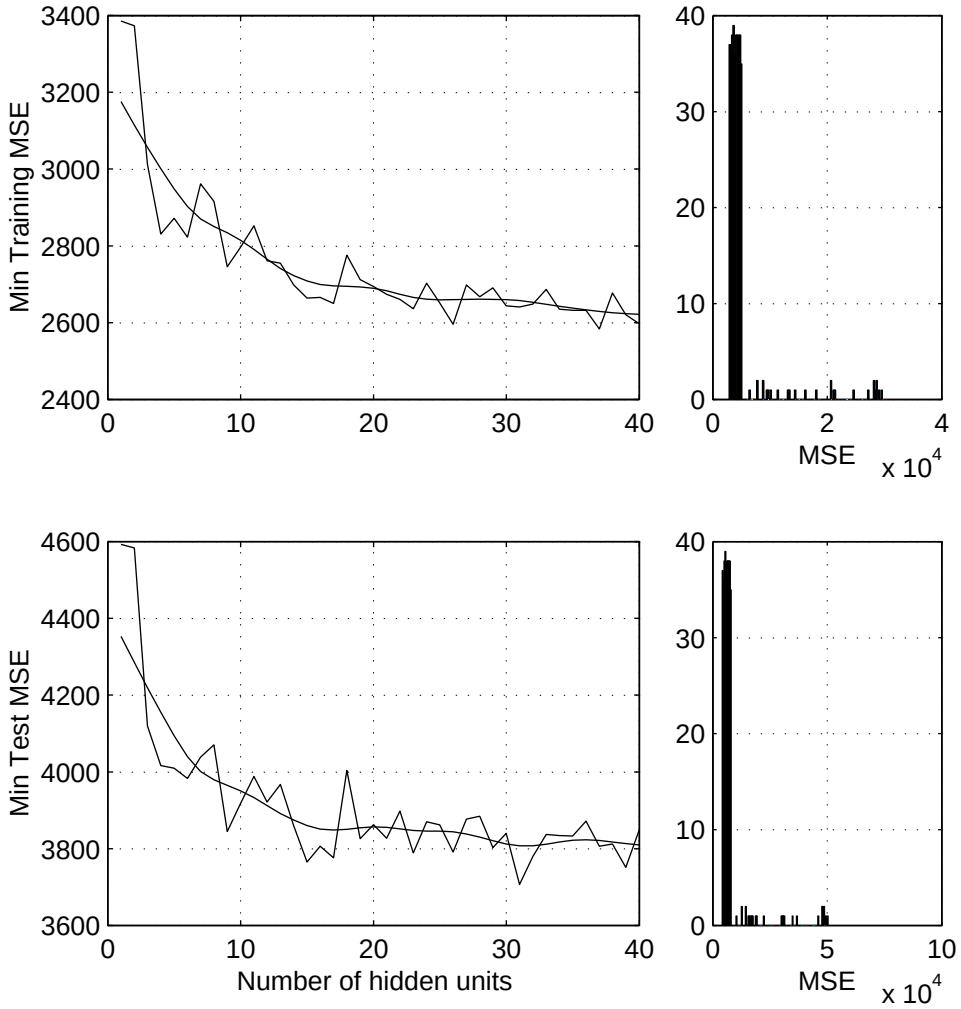

Figure 50: The minimum training and testing errors as a function of the number of hidden units. The smoothed line interprets the decreasing trend of the mean-squared error. The histograms present the total distribution of the training and testing mean-squared errors. Networks including one to forty hidden units and five input units were each trained ten times with the orthostatic test data.

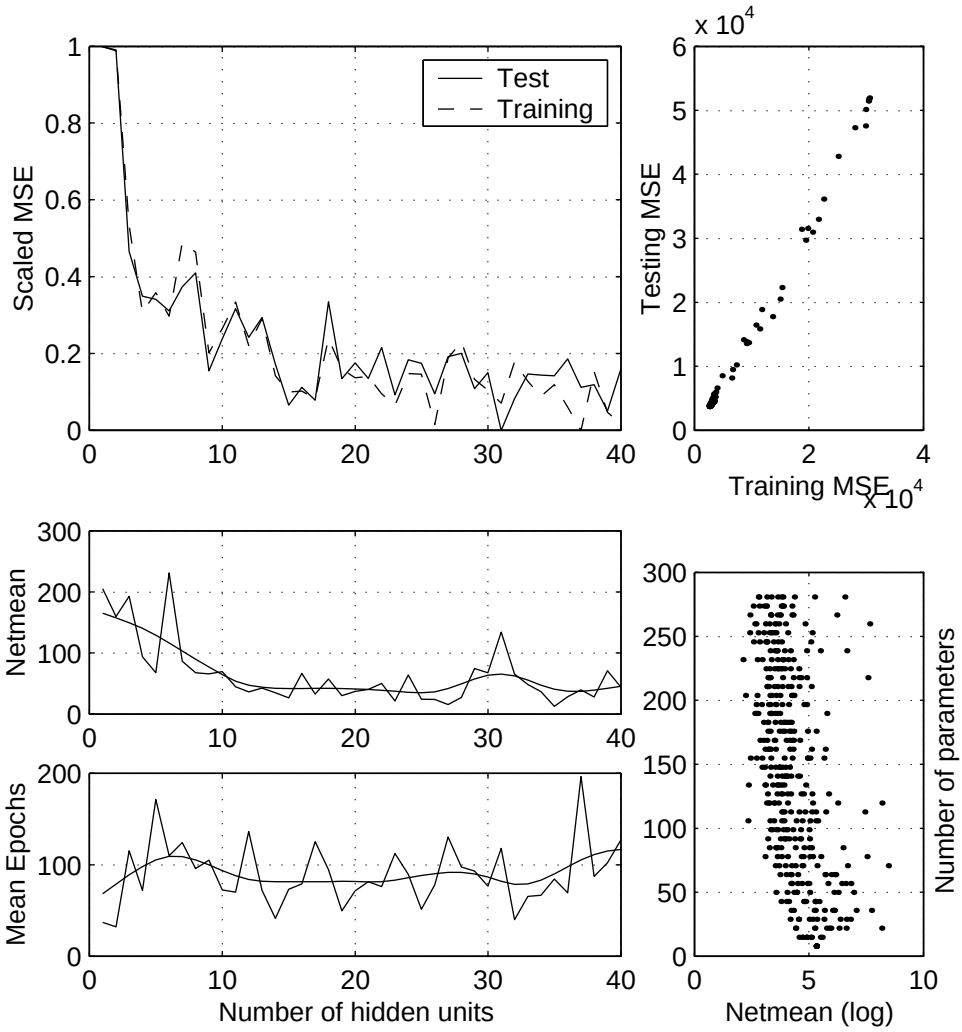

Figure 51: The figure in top left corner illustrates the best test data fit (solid) as a function of network hidden units and the corresponding scaled MSE of the training data (dashed). The average absolute value of the network parameters, ”netmean”, as a function of hidden units is presented in the middle figure. The upper scatter plot illustrates the linear correlation between the testing and training MSE. The scatter plot below presents the correlation between the number of parameters and ”netmean”. Both scatter plots include all the training cases.

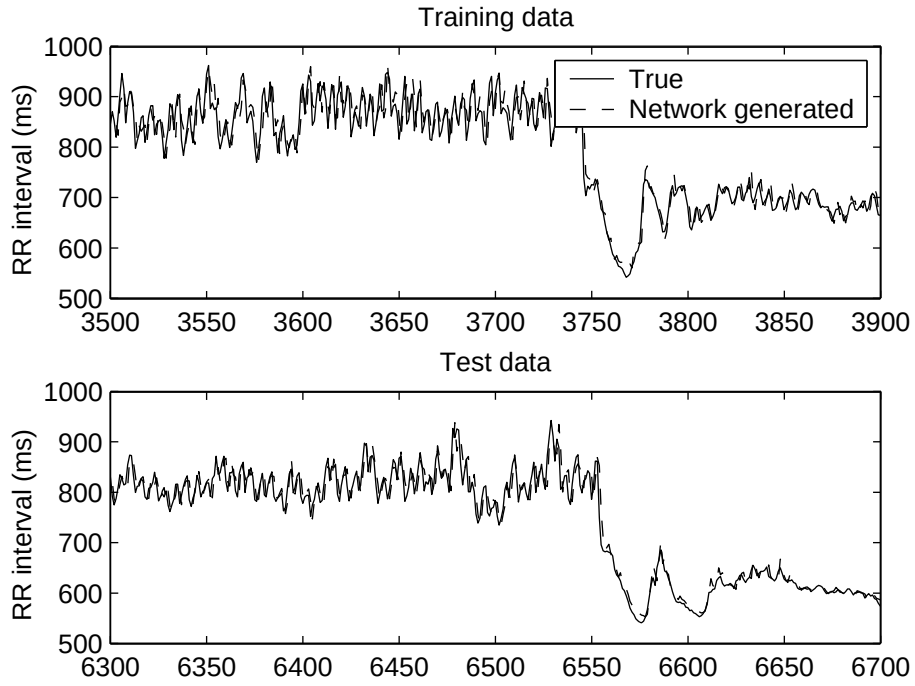

Figure 52: The upper figure is an example of the training data (solid) together with the network fit achieved with one-step-ahead predictions (dashed). The lower figure presents the same information with test data.

Figure 52 presents an example of the training and testing data with the corresponding network fit, with an architecture of forty hidden units.

Discussion

The current neural network experiment leaves open questions to further investigation. It will be interesting to see the neural network prediction on different stages of the test and if there is any repeated patterns before or after the subject starts to stand.

The experiment was continued to forty hidden units. The upper scatter plot at Figure 51 shows the training results of all the cases. The training cases resulting in a good training fit also resulted in a good testing fit. This means that we did not encounter overfitting. Overfitting happens when the training MSE is small but the testing MSE is high. If we continued to add hidden units to the system, then overfitting would probably happen at some point. In this experiment the unsuccessful training cases corresponded to optimization failure with a poor local minimum.

By studying the first layer weights of the network, the amount of effective network inputs could be estimated. If some input has very small outgoing

weights, it could be erased. The network should be trained several times to make reliable conclusions.

One possible application for RRI modeling could be detection and correction of signal artifacts. A one step-ahead prediction with a threshold could be used to detect artifacts in the signal. Furthermore, a neural network model could be used to replace the missing or corrupted beats and to simulate local variance of the signal.

6.2 Modeling of continuous Excess Post-exercise Oxygen Consumption

Excess post-exercise oxygen Consumption (EPOC) is the extent of physical activity induced by a heightened level of oxygen consumption after the cessation of physical activity, or briefly, the extent of additional oxygen consumption after exercise [105, p. 133].

After exercise, oxygen consumption (VO2) does not return to it's resting level immediately but rather in a curvilinear fashion (see, e.g., *short-term recovery and oxygen depth* [45, p. 1011]). The causes of EPOC after exercise may not be totally clear but based on literature it is hypothesized that the greater the fatigue accumulation during exercise the greater the EPOC and the longer the time required for VO2 to recover to the pre-exercise level.

Excess post-exercise oxygen consumption may accurately be measured after the exercise with machinery providing respiratory gases. The total amount of oxygen consumption above the base resting level gives the amount of EPOC for the exercise. To measure EPOC the respiratory gases are recorded until the base level is reached. The integral of the oxygen consumption during the resting phase is the quantity of EPOC. In Figure 53 three different exercises with 70% ex-

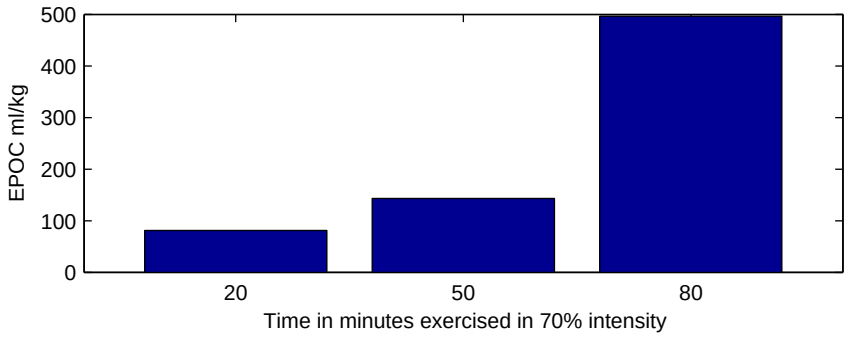

Figure 53: Measured EPOC of a 70 kilogram weighted individual exercising for different durations with 70% intensity. The figure suggests that the EPOC measured as a function of exercise time and intensity is not linear in respect to time.

ercise intensity lasting twenty, fifty and eighty minutes are illustrated. The figure demonstrates that the amount of EPOC is not linear in respect to time.

The heart rate during exercise gives information on the intensity of the exercise but it does not take into account the cumulative effect of the exercise duration. In [140, 149, 151] heart rate derived EPOC is suggested as a noninvasive measure of body fatigue, and furthermore, a system for the prediction of EPOC, recovery and exhaustion time is proposed. The innovation offers a method for continuously tracking the influence of exercise on body fatigue and the recovery from exercise without the restrictions of the laboratory environment or equipment. The procedure is claimed to be useful for providing real time feedback on exercise status to optimize physical exercise, sports training and recovery, and to provide predictions of time requirements for body recovery and exhaustion.

In this dissertation an alternative neural network based model is constructed for continuous modeling of EPOC as a function of accumulated body fatigue and current exercise intensity. The example is used to illustrate the benefits of dynamic neural network modeling compared to its static counterpart. It will also demonstrate the importance of the physiologically based presumptions in the model building. Furthermore, the example will demonstrate the use of constraints in the model selection and it will illustrate how to generate extra data to produce evenly sampled dataset. Moreover, the presentation is a completion for the implementation described in Saalasti, Kettunen and Pulkkinen 2002 [151], but may also be considered as a separate, isolated, presentation for modeling of body fatigue, or EPOC. The physiological context and interpretation is mainly described in [140, 149, 151] and is partly produced here to provide sufficient physiological background for the computational system.

The quantity of EPOC depends on the intensity and duration of the exercise. As the dissertation is concentrated on heart rate time series analysis, the relationship between heart rate and oxygen consumption is founded next to define the appropriate exercise intensity estimation.

6.2.1 Oxygen consumption and heart rate level as estimates for exercise intensity

The rate of oxygen intake, oxygen consumption (VO2), is a central mechanism in exercise and provides a measure to describe the intensity of the exercise. Oxygen is needed in the body to oxidize the nutrition substrates into energy and, therefore, VO2 is very tightly coupled with the energy consumption requirements triggered by exercise and physical activity. Thus, VO2 is an indirect measurement of burnt calories during the exercise. The American College of Sports Medicine Position Stand of recommendations for exercise prescription [119] suggests the use of VO2 for measuring physiological activity.

The level of oxygen consumption can be measured by different methods. The most accurate methods rely on the measurement of heat production or analysis of respiratory gases. The disadvantage of precise measurement is the require-

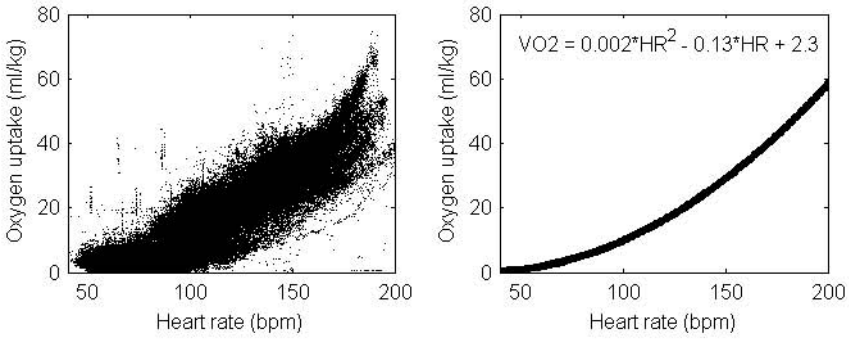

Figure 54: A scatter plot between absolute VO2 and HR-values together with a polynomial fit based on the data. The data is a collection of 158 recordings with different individuals and different tasks.

ment of heavy equipment restricting the measurement to the laboratory environment.

Given the relative difficulty in measuring oxygen consumption directly, we may estimate VO2 on the basis of heart beat. Heart rate is a major determinant of the circulatory volume and often provides a reasonable estimate of the oxygen consumption. This is empirically illustrated in Figure 54, where a nonlinear relationship between VO2 and heart rate level is demonstrated together with a polynomial fit to the data. A raw estimation of transformation from heart rate to oxygen consumption may be expressed with the following equation:

$$VO2 = 0.002 \cdot HR^2 - 0.13 \cdot HR + 2.3. \tag{105}$$

The proposed model in (105) is inaccurate (MAE=3.6982 ml/kg), and for increased precision additional information, like individual maximal oxygen consumption or heart rate level or resting heart rate level, could be exploited. Furthermore, other bio-signals, like respiratory activity, may improve the model's accuracy. The effect of heart rate derived respiration frequency into VO2 estimation is presented in [139, 141].

Maximal oxygen consumption (VO2max) is defined as the maximal oxygen intake during exhaustive exercise. It describes a person's ultimate capacity of aerobic energy production. This may be achieved by stepwise exercise protocol where body stress is taken into voluntary exhaustion (maximal stress test). During the test the oxygen uptake is measured with suitable laboratory equipment. Also non-exercise methods are available to estimate a person's aerobic capacity. They are often based on individual characteristics such as, for example, age, sex, anthropometric information, history of physical activity, or resting level physiological measurement (e.g. Jackson et al. [64], or [172]). In a similar manner, via maximal stress test or via mathematical formulation, maximal heart rate level (HRmax) may be evaluated. An example heuristic for determination of HRmax is

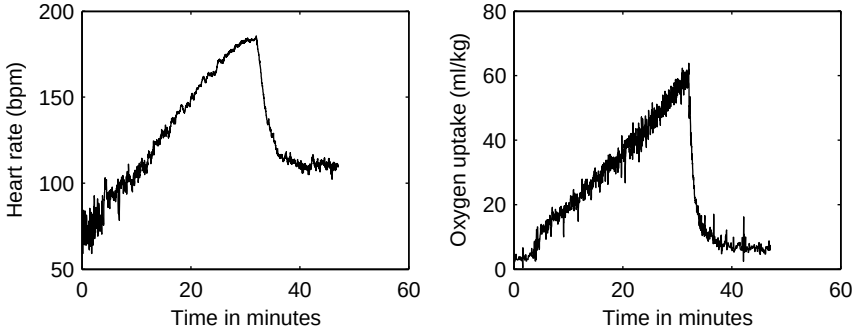

Figure 55: A maximal stress test illustrating the relationship between the oxygen consumption and heart rate level.

a raw formulation $220 - age$ expressing linear relationship between HRmax and the age of an individual. Figure 55 illustrates an example HR and VO2 time series in a maximal oxygen uptake test. A close relationship between the measurements is expressed with a high correlation coefficient $C_P = 0.9185$.

The database illustrated in Figure 54 also contains the laboratory recorded HRmax and VO2max values. Constructing a second order polynomial fit between HR proportional to HRmax (pHR) and VO2 proportional to VO2max (pVO2) results in a formula

$$pVO2 = 1.459 \cdot pHR^2 - 0.49 \cdot pHR + 0.04. \tag{106}$$

The resulting pVO2 may be transformed to absolute scale by multiplying the result with individual VO2max. The error of the fit was MAE=3.1558 ml/kg, decreasing the error 15%, compared to (105).

The transformation in (106) gives a foundation for expressing the individual exercise intensity by means of the proportional heart rate. As the exercise intensity is expressed in percentages, the conversion to relative scale is both intuitive and practical, since the intensity measure may be directly compared between different exercises, and in some degree between individuals having different physiological attributes. Thus, two people that differ in their maximal VO2 but exercise at the same relative intensity have the similar exercise impact on their bodies.

Notice that the error measurements are absolute and the models are fitted to the whole database without taking the data distribution into account. The error distribution is considerably higher in low intensities. If only part of the data is selected, e.g., a partition of data consisting of oxygen the consumption levels between $1 - 5$ ml/kg, the error estimates are MAE=2.1895 ml/kg and MRE=82% for model presented in (105). For the VO2 estimate in (106) the corresponding errors reduce to MAE=2.0124 ml/kg and MRE=75%. The selection was composed of 47% of the data. The selected VO2 level corresponds to the resting oxygen consumption level of a young adult man [45, p. 1014].

In the higher end of the distribution we selected exercise intensities pVO2>40% consisting of 25% of the data. The mean HR in this area was 140 bpm± 24 bpm. The corresponding model errors in this partition were MAE=6.2726 ml/kg, MRE=23% and MAE=4.6950 ml/kg, MRE=19%, for the models in (105) and (106), respectively.

The above analysis demonstrates that the relative error for both models is high in VO2 level between $1-5$ ml/kg and considerably lower for higher exercise intensities. This also reveals that different error measurements should be exploited in the analysis: a mean absolute error suggests that both models map the lower exercise intensities better, but when evaluating relative errors it appears that both models work better with the higher exercise intensities.

Furthermore, the modeling reveals that the oxygen consumption level has a high inter-individual variation. Guyton reports average VO2max levels for an untrained average male to follow 3600 ml/min, an athletically trained average male 4000 ml/min, and male marathon runners 5100 ml/min [45, p. 1014].

Naturally the high exercise intensities result in an increasing effect on measures like body fatigue or energy consumption. Our interest is to measure the body fatigue accumulated during exercise, where the effect of lower intensities on the index is less dramatic. Furthermore, we are modeling the system giving a response of an average individual, so that some modeling error must be tolerated.

As discussed earlier, the quantity of EPOC depends, at least, on the intensity and duration of the exercise (see Figure 53). The above analysis concludes that HR may be used as an indirect measure of exercise intensity for a person. Next the foundations of the EPOC model are built.

6.2.2 Building the EPOC model

A presumption for the model is that the EPOC may be estimated as a function of the current exercise intensity and accumulated body fatigue. Furthermore, to build a discrete model, the time difference between consecutive sampling points, $\triangle t$, has an effect on the index. This may be mathematically formulated as follows:

$$EPOC_t = f(EPOC_{t-1}, exercise\_intensity_t, \triangle t). \tag{107}$$

The recursive modeling of the accumulation of the body fatigue has the benefit of not having requirements for knowing a priori the beginning time of the exercise and different durations of exercise at varying intensities.

Let us emphasize that the amount of EPOC may not be continuously recorded in a laboratory. EPOC may only be accurately measured by finishing the exercise and recording the oxygen usage during the recovery until a base level of oxygen consumption is reached. The formulation in (107) has to be considered as a model that is able to predict the amount of post-oxygen consumption if the exercise finished at any given moment.

The described restrictions will have the effect on the availability of consistent data. In a laboratory we are able to measure exercise intensity as VO2 and

the amount of EPOC after the exercise. We may control the duration of the exercise and the intensity. The state before exercise is presumed to be the base line, i.e., normal inactive oxygen consumption rate (EPOC equals zero). For the EPOC modeling a dataset of 49 sessions were gathered consisting of different individuals, exercise durations, and intensities. In all datasets, the intensity of the exercise was kept constant during the training. Different sets of data consisted of exercises lasting between 2 to 180 minutes and exercise intensities between $18 - 108\%$ of VO2max. Figure 53 illustrates the amount of EPOC in three different exercises with different durations with a constant, 70%, exercise intensity.

The pre-model in (107) introduces the properties we wish to have: the model should not require the starting time of the exercise, but rather be a pure function of the current intensity and accumulated fatigue. In addition, we limit the inspection to strictly increasing functions; the estimation of recovery is not demonstrated. To estimate EPOC continuously the model has to interpolate each sample from the beginning to the end, from zero to the recorded fatigue. To generate equidistantly sampled signal we linearly interpolate the data to a one minute sampling. Thus, adjacent samples will have a one minute difference ($\triangle t = 1$). As the phenomena itself is not necessary linear we will base the optimization of the model to weighted squared error defined in (12). Model predictions inside the sampling interval may be obtained using interpolation.

An output recurrent neural network (ORNN), with current exercise intensity as input, is chosen to model the system. The network will give the current amount of EPOC as output, and the output is fed back to the network as an input in the next iteration. Sigmoid units are used in the hidden layer and linear unit in the output. The network is a special case of Jordan-network without recurrent self-connections in the input layer. It is apparent that ORNN architecture may be used to follow the characteristics formed in (107).

As has been discussed in Section 4.2, a recurrent network has a static counterpart. However, the static equivalence is only apparent for the set data length. Static networks will also map equal inputs to the same output. If exercise time and intensity were both used as an input for, e.g., FFNN, the resulting model would only give an average response for a certain input pair. Also for the model to be strictly increasing, the current state of the system should be fed back to the model. The latter is the final drawback that will prevent static modeling of this phenomena based on the pre-model. This also implies that other recurrent models could be applied here, as they are able to store the internal state of the system.

6.2.3 Results with the output recurrent neural network

Output-recurrent neural network was trained with varying $(3 - 14)$ number of hidden units 1595 times altogether starting with different initial conditions. No constraints were used during the optimization. The weighted squared error was constructed as follows: the beginning (EPOC=0) and end of each exercise had a weight of one and all the linearly interpolated time instants in between were

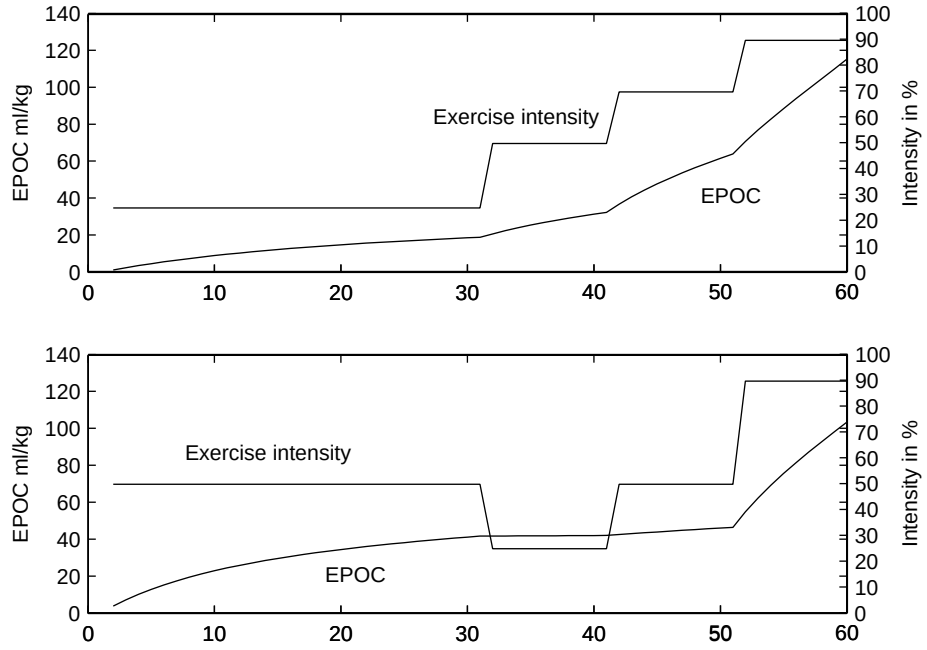

Figure 56: Two simulated intensity time series and the corresponding EPOC estimate based on output recurrent neural network model. The continuous time series is the EPOC and step-function is the corresponding exercise intensity.

weighted with 0.00001. The latter parameter was not optimized but rather chosen by intuition and trial and error. If the constants were set to zero, the optimization failed to find a strictly increasing model.

Only twelve local minima resulted to strictly increasing functions with the given test data. The training data, two artificial datasets and a maximal stress test time series were used to test the constraint. Naturally all possible inputs were not experienced and it may not be guaranteed that the model would increase in all possible setups. Thus, the model is only *empirically* strictly increasing.

Surprisingly all the twelve local minima were found with 6 hidden units. This empirically suggests a correct model complexity for the given phenomena.

The artificial datasets together with the resulting EPOC are presented in Figure 56. It appears that the model behaves well as a function of intensity and previous EPOC estimate; higher intensity results in an increased EPOC. In addition, the past intensities affect the final result as both datasets include the same intensity in the end resulting in differing total EPOC.

Figure 57 illustrates a heart rate time series dataset together with the exercise intensity estimated with the transformation in (106) and the continuous presentation of EPOC. The simulation indicates that EPOC model continuously tracks the body fatigue during exercise and is sensitive to different exercise intensities.

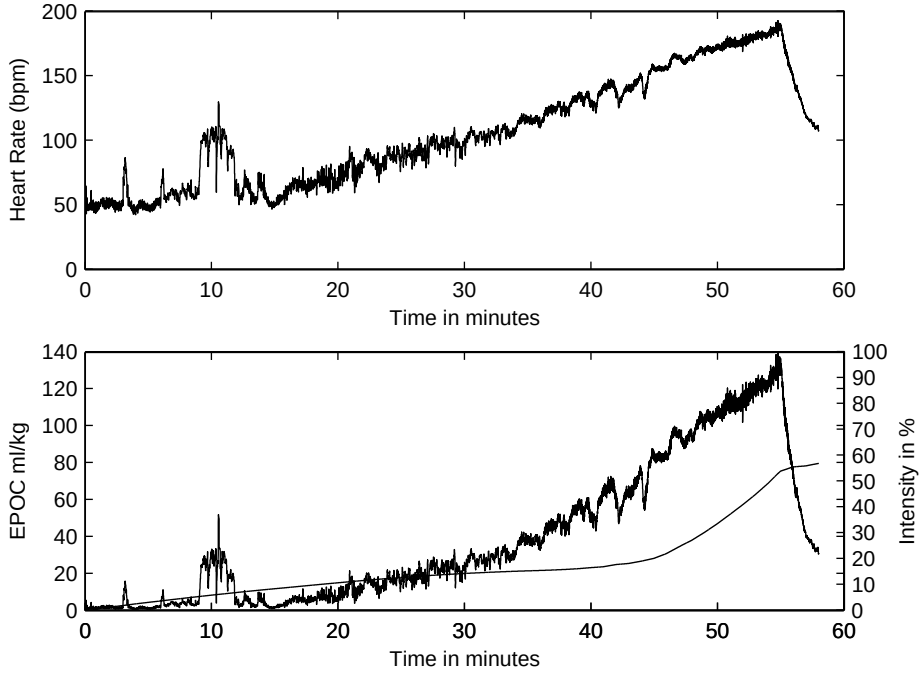

Figure 57: Upper figure illustrates a heart rate time series during a maximal stress test. The bottom figure illustrates the corresponding pVO2 estimate transformed with equation (106) together with a continuous presentation of EPOC. The EPOC estimate is based on output recurrent neural network model. The exercise intensity at current time instant and the previous EPOC quantity were used to predict the current quantity of EPOC.

The chosen EPOC model resulted to MAE=32.7 ml/kg, MRE=27.5 % for the 49 original, true, EPOC samples. The corresponding residuals together with the sample labels are gathered in Figure 58. The figure proposes that the network is less accurate for the samples having a higher intensity, EPOC or exercise duration. The data distribution is concentrated on lower exercise durations and EPOC. Since the optimization is also affected by the distribution, the model predictions favour the most frequent samples. Longer lasting exercises are iterated more often, i.e., the estimations are fed back to the network more frequently resulting in, perhaps, increased error and stability problems. The inter-individual variation may also affect the residual distribution, suggesting that the differences between individuals tend to grow as more exhausting exercise is performed.

6.2.4 Revisiting the presumptions; experiment with a FIR network

It was discussed in Section 6.2.2 that a static neural network may not be used to model the strictly increasing and continuous EPOC model. It was also presumed

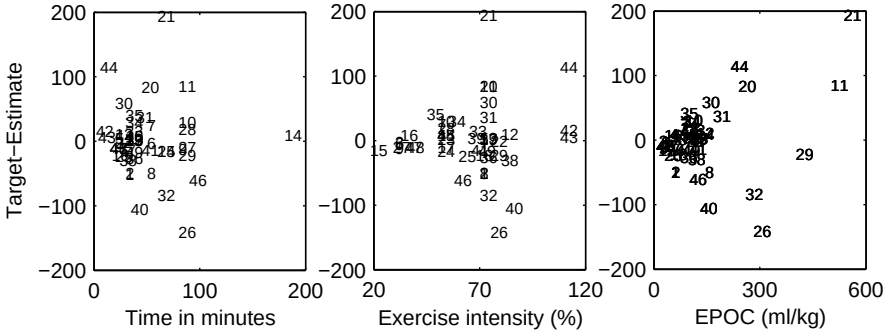

Figure 58: Residual plot of the EPOC model between different dimensions, where each sample is labeled with a number. The analysis reveals, with some exceptions, that the model error increases as a function of exercise intensity, time and quantity of EPOC. This may be result of the data distribution, since low intensity exercises are more common. In addition, network is ran multiple iterations, which may result in stability problems of recurrent neural network. Furthermore, the problem may be result of inter-individual variation, or finally a combination of all three deficiencies.

in (107) that the EPOC model should use the current intensity and previous EPOC to predict the current amount of EPOC. The last presumption is now re-evaluated: it is possible that other dynamic neural networks could be applied, since they are able to store the internal state of the system, which allows a strictly increasing function to be constructed mapping similar input to different output. Perhaps an alternative model could be utilized, which does not rely on the recurrent connection between the output and input layers.

A FIR network, presented in Section 4.2.2, was trained in a similar concept as ORNN in the previous section. The whole procedure consisted of calculating over one thousand different local minima for different three-layered FIR networks including linear output, initialized with different initial conditions. The number of sigmoid hidden units in the both hidden layers varied between three to eight and the delays in the first hidden layer varied between two to four.

The best MAE=53.9 ml/kg and MRE=63.1% were achieved with a FIR network including three hidden units in the second and third layer. The number of delays was two in the second layer. The model selection was based on both the estimation error and the number of occasions the constraints were violated, as there did not exist such a network that would have realized all the empirical constraints. Hence, the resulting network was not strictly increasing with the test datasets. Furthermore, the network's output soon became constant, as constant intensity was fed to the network. It is reminded that for the ORNN the corresponding errors were MAE=32.7 ml/kg and MRE=27.5%. Furthermore, the ORNN was able to satisfy the empirical constraints.

Using elapsed time from the beginning of the measurement as another input, decreased the error to MAE=30.0 ml/kg and MRE=62.4%. The network had three hidden units in the second and third layer, and four delays in the second layer. Still, however, the constraints were left unsatisfied. The network seemed to operate similar to FFNN, and the delays in the hidden layer did not direct the model estimates to follow the constraints. Thus, it may be concluded that the ORNN was superior to the FIR network. Furthermore, the presumptions of the model structure in (107) seems valid.

6.2.5 Discussion

In athletic training and sports the balance between training load and recovery is crucial to achieve, improve and maintain good physical fitness. Enough rest to recover from the exercise is required and the load and timing of the training bouts have to be optimal to gain positive training effect. Too frequent and strenuous training bouts may lead to a negative training effect [140].

Control of the training load is conventionally based mainly on the previous personal experience about the effect of exercise on the body. Current methods that may be used to obtain objective information on body fatigue due to exercise requires invasive procedures (e.g., lactate measurement) and are, thus, restricted to a laboratory environment demanding professional aid.

A physiology based measure revealing the accumulation of exercise-induced fatigue during different intensities and phases of exercise was established. The accumulated body fatigue, or EPOC, is suggested to be utilized to optimize exercise and fitness training. Requiring only heart rate monitoring makes the proposed approach especially suitable for field use.

In innovation presented in [76, 78] EPOC information is exploited for the detection of different states of the human body. The overall system is applied for daily monitoring of physiological resources. A key part of the system is segmentation of the HR signal with the GLR-algorithm. The segmentation information together with calculated features and chosen statistics, are used to detect rest, recovery, physical exercise, light physical activity and postural changes from HR.

From a mathematical point of view, the presentation demonstrated an applicability of a dynamic neural network, the output recurrent network, to a biosignal. The overall process of data collection, pre-model generation, transformation of heart rate to exercise intensity, and model building for continuous excess post-exercise oxygen consumption estimation were presented. Furthermore, a heuristic for searching a strictly increasing function was presented and a procedure to re-generate the missing data with an appropriate optimization heuristic.

The behavior and properties of the resulting EPOC model were found satisfactory. The overall error was tolerable as the inter-individual variation of the modeled system was considerably high. In addition, an alternative experiment with a FIR network was illustrated. The experiment suggested that the presumptions of the model structure in (107) were valid; the current EPOC estimate should

be fed back to the network.

The application was not presented in all its dimensions, and further work may be required to find the optimal model architecture. The recovery of the exercise was not modeled as it introduces another complicated dimension in the phenomena. The solution for estimating and combining both recovery and fatigue components to model EPOC is introduced in Saalasti et al. [151].

It is not claimed that the presented output recurrent neural network is optimal for the given problem. However, we may say that to model the problem in a physiologically sensible way, only a dynamic network may be applied, not its static counterpart. The current state of the system has to be presented in the model for the model to strictly increase. Otherwise equal input will result in equal output and accumulated EPOC will not affect the system. Furthermore, the temporal memory in the network should exploit the output estimates when constructing recurrent connections.

6.3 Modeling of respiratory sinus arrhythmia

Even if the relationship between the respiratory frequency and heart rate is a well known phenomena (see Section 2.5), a methodology to accurately reveal the breathing component from the heart rate is yet to be constructed. Only under optimal conditions, for example, during spaced breathing, the breathing frequency is distinct enough to be expressed with time-frequency analysis. The identification and accuracy of the respiration frequency diminishes considerably whenever the heart period signal obtained during ambulatory monitoring includes nonstationary changes in either the breathing cycle or heart rate variability. Such nonstationarities may occur, for instance, due to movement, postural change, speech, physical exercise, stress or sleep apnea.

Heart rate monitoring has been successfully used in managing exercise training in field use since the introduction of heart rate monitors in the 1980's, but at present it offers only limited information to the individual engaged in exercise training. Information on respiratory activity would certainly provide new perspectives in optimizing training in field.

In Kettunen et al. [77] a general approach for the detection of respiratory frequency based on heart rate time series is created. The target of the research was to derive a reliable measure of respiratory information based solely on the heart period signal. Furthermore, in [152, 141] the heart rate derived respiration frequency is exploited for oxygen consumption estimation. The solutions presented in this section may be considered as alternative implementations or completions for these studies, but may also be read as a solid presentation.

In this section three different models are applied to respiratory detection: a feed-forward neural network, a transistor network and generalized regression neural network. The purpose of this study is not to compare the methods but rather show their varying properties. Limiting the use of different models to a

few applications does not provide full proof for general use of the methods. Thus, an analytic approach is attempted instead of an empirical comparison between the methods.

6.3.1 Time-frequency analysis on the breathing test data

The effect of respiration on the heart rate high-frequency component ($0.15 - 0.5$ Hz) is an acknowledged phenomenon [17, 117] (see also Section 2.5). To examine the relationship, a dataset is created and analyzed using time frequency presentations. The dataset[18] consists of a metronome-spaced breathing test followed by data generated under spontaneous conditions. The test starts with one minute of spaced breathing at a frequency of 0.5 Hz. Then the breathing rate is stepped down by 0.1 Hz every minute until it reaches 0.1 Hz. After this, the procedure is reversed to the starting frequency. The total test time is nine minutes. Each new step is indicated and controlled by a computer-generated sound.

Eight different measures were recorded during the test: skin conductivity, RR intervals, systolic and diastolic blood pressure, electromyogram presenting muscle activity from the biceps and triceps, respiration using a spirometer (to measure tidal volume) and respiration from the chest expansion.

After the breathing test spontaneously generated data was recorded for 40 minutes. The subject was sitting and allowed to speak and read. During this part the spirometer was not used.

Similar experiments as the breathing test have been studied to understand the influence of respiration on heart rate and blood pressure (see Novak et al. [117]).

Figure 59 presents the time-frequency distributions of the heart rate and respiration in the high-frequency band calculated with a short-time Fourier transformation (see Section 3.1.2). Figure 60 shows the corresponding instantaneous frequencies for the signals.

Inspection of the figures reveals that the respiration frequency cannot be followed purely with the mode frequency. The fast frequency changes presented in the top of Figure 60 suggest that the method is not completely reliable as some of the changes are not physiologically valid. The breathing frequency oscillates and is noisy.

The mean frequency does not give the true frequency either, since it does not have a sharp frequency resolution. There is a lot of power in the frequencies close to the maximum frequency component (see Figure 59). However it gives the most smooth and continuous performance. If the periodic components were very sharp and concentrated on one frequency the mean frequency would give its best performance. The more spread the power spectrum is, the less accurate the mean frequency is.

[18] Dataset was produced at the Research Institute for Olympic Sports as part of the StateMate-project (years 2000-2001). The project was developing an on-line physiological monitoring system as part of a personal health monitoring service.

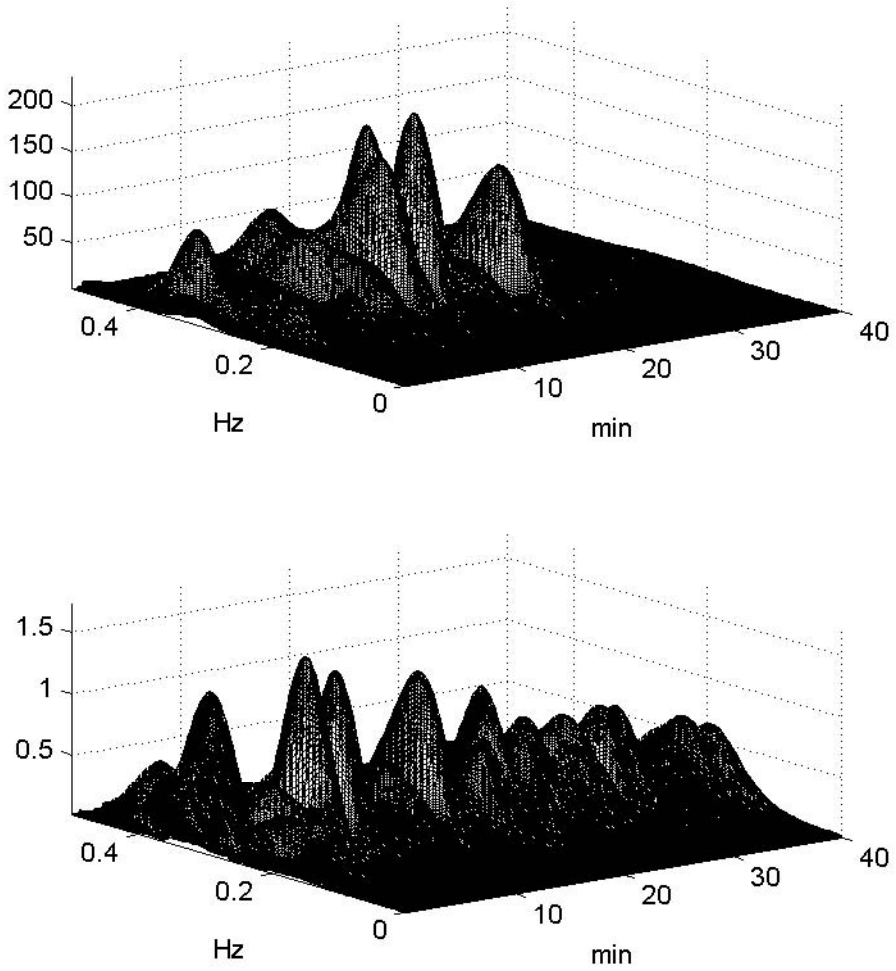

Figure 59: Upper figure presents the time-frequency distribution of the respiration measured from chest expansion. The lower figure presents the time-frequency distribution of the heart rate.

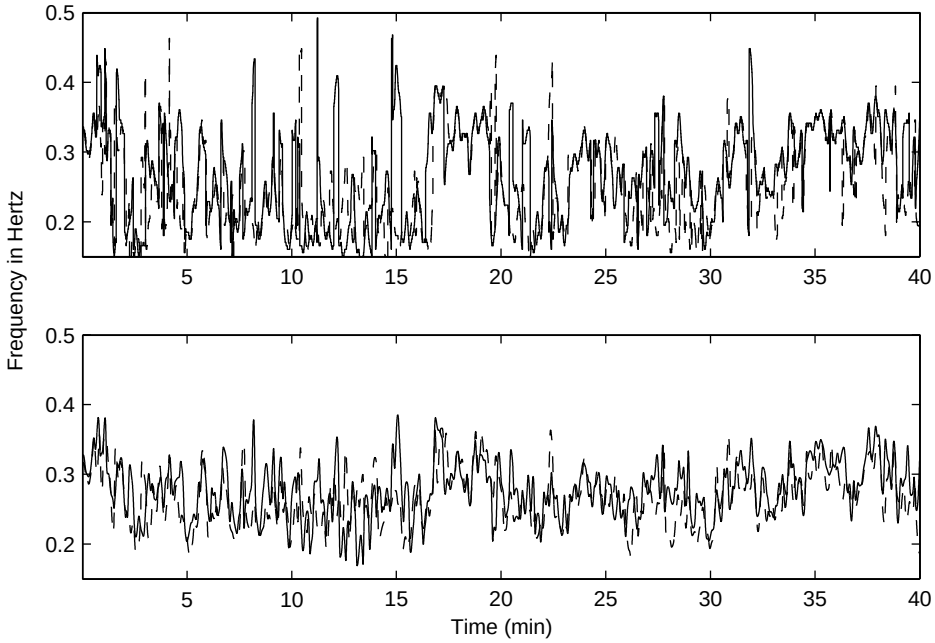

Figure 60: Instantaneous frequencies calculated for the respiration (solid line) and heart rate (dashed line) with two different methods. The figure at the top presents the mode frequency of the time-frequency distribution while the second figure presents the calculations with mean instantaneous frequency.

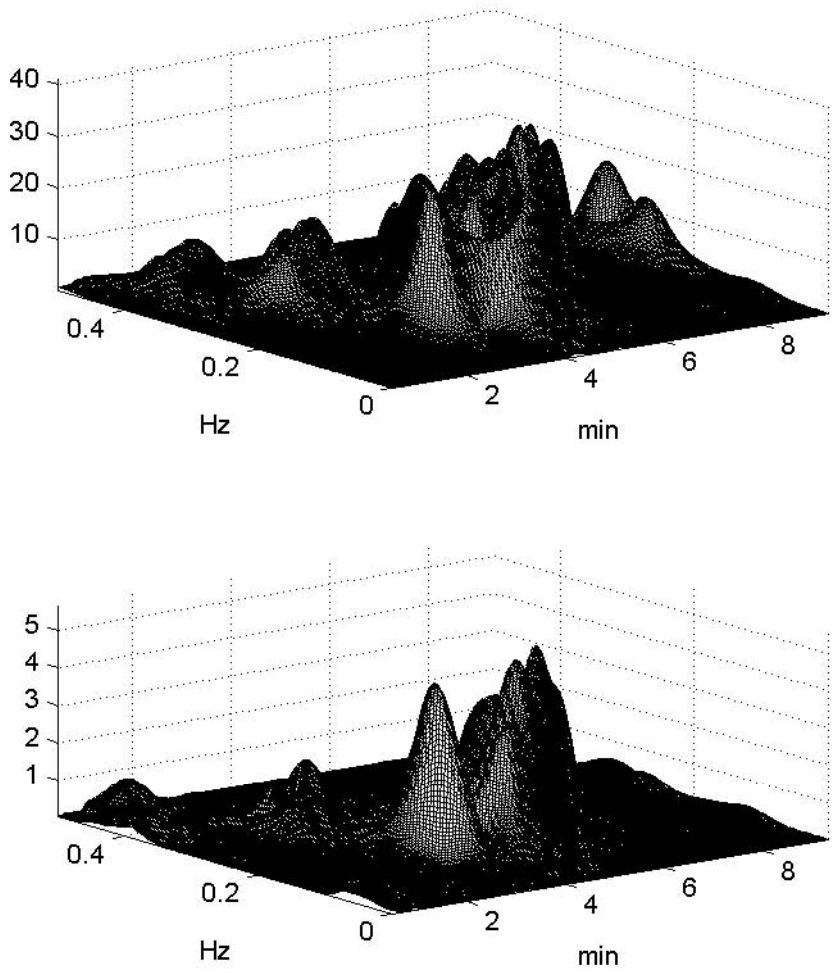

Figure 61: Upper figure is the time frequency presentation of the respiration during the breathing test. The lower figure presents the time frequency distribution of the heart rate during the test.

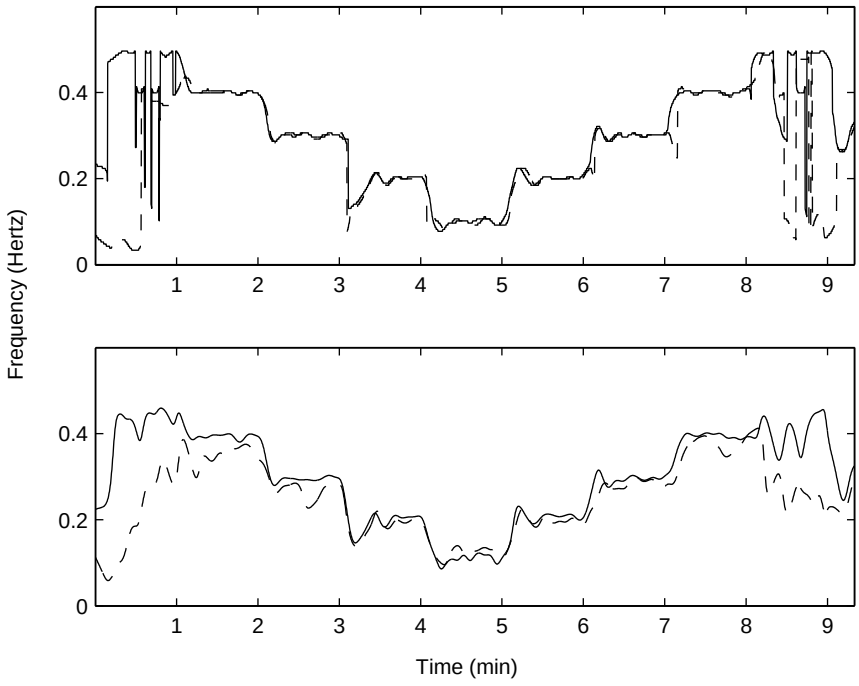

Figure 62: Instantaneous frequencies of the breathing test calculated for the respiration (solid line) and heart rate (dashed line). The first figure is the instantaneous frequency calculated with mode frequency and the second figure presents the calculations with mean frequency.

Figures 61 and 62 present the results achieved in the breathing test. Notice from Figure 62 that the breathing frequency at 0.5 Hz is noisy. The breathing frequency and the corresponding heart rate power have strong negative correlation. The heart rate's low-frequency component is always present and may have more power compared to the high-frequency band where the breathing power exists. Another reason for the failure could be the difficulty to breath at this high phase.

Figure 62 also shows the failure of the mean frequency: it cannot follow the true respiration frequency as reliably as the mode frequency when heart rate is considered. At lower frequencies the two instantaneous frequencies almost overlap, as the relative power is higher and the breathing pattern is more clear.

6.3.2 Optimizing a time-frequency plane to detect respiratory frequency from heart rate time series

In Section 2.5 the oscillations of the heart rate time series were linked to respiratory sinus arrhythmia. In the previous Section time-frequency distributions were utilized for the analysis of the phenomena. It appeared that the link between the heart rate and respiratory oscillations was apparent and possible to reveal with the correct mathematical methodology. However, some questions remained as neither of the instantaneous frequency estimates were able to follow the correct respiratory frequency.

It was discussed that the respiratory frequency has a strong negative correlation with the total power of the heart rate. Hence, high breathing frequency results in a lowered total power in heart rate. Furthermore, the heart rate low-frequency component is always present and may have more power compared to the high-frequency band where the breathing power exists. To make it less predictable, the respiratory frequency may also appear in the LF-band. Actually the breathing frequency may empirically range from 0.03 to 1.3 Hz[19]. This raises the question if the described variation and balance between the LF- and HF-powers could be modeled and controlled to follow the respiratory frequency from the heart rate. Perhaps, by giving an optimized weighting for the whole frequency-band, the respiratory detection could be improved.

In this section we will utilize a feed-forward neural network for adaptive filtering to detect respiratory frequency from the heart rate time series. The general concept of neural network adaptive filtering was presented in Section 5.2.1.

Creation of the target data

The adaptive filtering procedure presented in Section 5.2.1 requires target time series $y(t)$ providing the true respiratory frequency. This cannot be extracted accurately from the heart rate time series. The task is to dynamically filter the TFRD of the heart rate time series in such a way that the respiratory oscillations may be estimated from it.

[19]The empirical breathing range is based on the database used in this section.

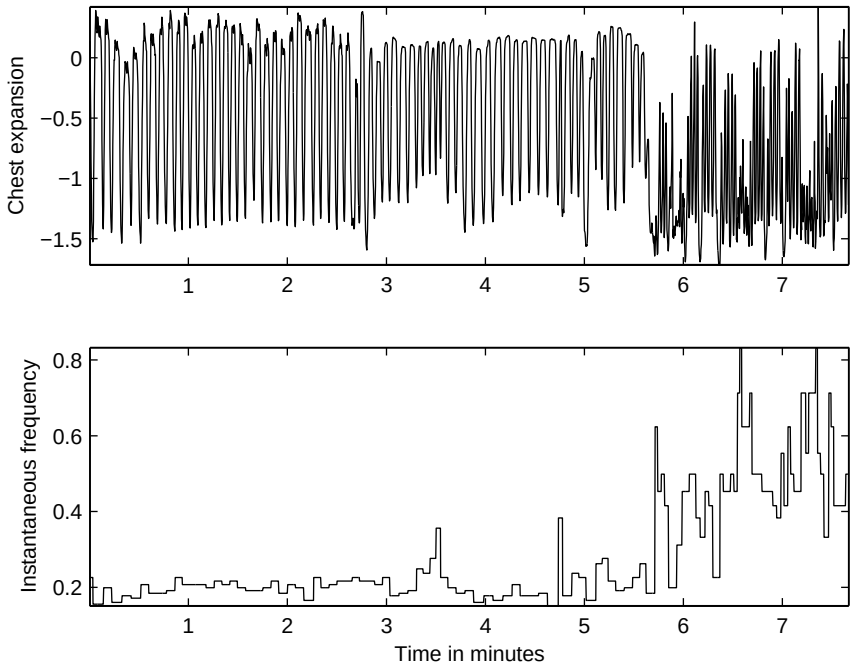

Figure 63: The upper figure illustrates the chest expansion time series presenting the expiration and inspiration as sinusoidal oscillations. The lower figure presents the corresponding instantaneous frequency derived from the upper signal with the peak detection algorithm presented in Section 3.1.9. The peaks were visually verified and corrected by a human expert.

The target respiratory frequency time series may be produced from the information achieved from spirometer or respiration derived from the chest expansion. Here the latter is used for practical issues, since it is more convenient for long time recordings.

Figure 63 demonstrates an example chest expansion time series together with the instantaneous frequency derived from it. For automated processing the algorithm presented in Section 3.1.9 was utilized to detect lower peaks of the chest expansion time series together with visual inspection of a human expert to derive instantaneous respiratory frequency target time series.

A total of 35 hours of heart rate and respiratory oscillation data was produced in the described manner to provide sufficient dataset for the experiment[20]. Distribution of the data along different dimensions is presented in Figure 64.

[20]The database is property of Research Institute for Olympic Sports and Firstbeat Technologies Ltd.

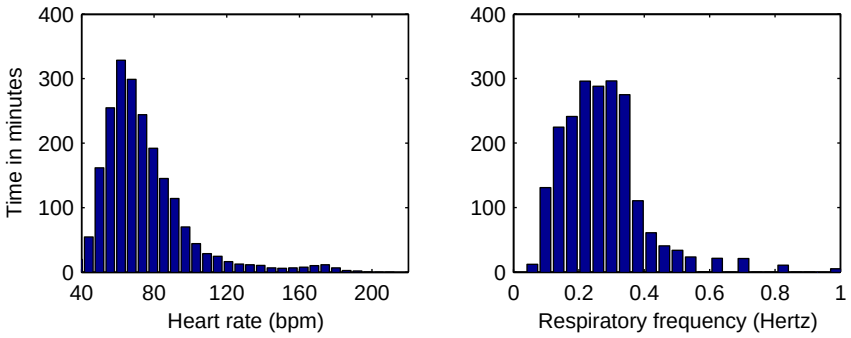

Figure 64: A database consisting of over 35 hours of heart period recordings and chest expansion time series derived respiratory frequency. Data was sampled in five hertz and it consisted of over 50 different individuals with varying age, sex and physiological condition, performing different tasks from sleeping to maximal exercise performance.

Model building

We have chosen a feed-forward neural network with a linear output neuron as the adaptive filter function. This results in a real-valued filter presented in (100). Furthermore, we will use the mean frequency moment presented in (102). The network g has five inputs: a single time-frequency input; normalized power $\frac{F(k,t)}{\max_k F(k,t)}$, three time series inputs; heart rate divided by two-hundred and moving averaged with ten seconds Hanning window, instantaneous frequencies; the mode and mean frequencies of the original TFRD calculated from the heart rate time series (see Section 3.1.1) and finally the frequency input $w(k)$. The value range of each input is already from zero to two, so the linear combination in the network input layer is reasonable and no further normalization is required.

The network input for frequency $w(k)$ and time instant t is defined as

$$g(k,t) \equiv g\left(w(k), \frac{F(k,t)}{\max_k F(k,t)}, f_{MOD}(t), f_{MEAN}(t), \frac{hr(t)}{200}\right). \qquad (108)$$

The three time series inputs were considered to have coupling with the true respiration frequency. This may also be quantitatively verified as a correlation between the respiration frequency and the corresponding input. The normalized TFRD contains information about the power distribution over frequencies. The frequency bins $w(k)$ are required for the network to produce the correct weighting for each frequency instant.

Since in the original TFRD derived from heart rate time series the power distribution concentrates in the very and ultra low-frequency bands (ULF-VLF), the mode frequency may be misplaced. The high-frequency band (HF) is usually considered to be mostly affected by the respiratory sinus arrhythmia, so the mode frequency may be chosen to be calculated only for frequencies higher than 0.15

Hz.

A squared error $E(t)$ (divided by two) between the target respiration frequency $y(t)$ and the frequency moment estimate $\hat{f}_{MEAN}(t)$ reads as:

$$E(t) = \frac{\left(\hat{f}_{MEAN}(t) - y(t)\right)^2}{2},$$

and the mean-squared error is

$$E = \frac{1}{T}\sum_{t=1}^{T} E(t).$$

Unknown network parameters may be solved by using the general result in (98), since the error function is continuous and has analytic derivatives. The gradient of a network weight w_{ij}^l respect to the error function E reads as follows:

$$\begin{aligned}
\frac{\partial E(t)}{\partial w_{ij}^l} &= \sum_{k=1}^{K} \frac{\partial \hat{f}_{MEAN}(t)}{\partial g(k,t)} \frac{\partial E(t)}{\partial w_{ij}^l}(k) \\
&= \sum_{k=1}^{K} \frac{w(k)\frac{\partial \hat{F}(k,t)}{\partial g(k,t)} \sum_{m=1}^{K} \hat{F}(m,t) - \frac{\partial \hat{F}(k,t)}{\partial g(k,t)} \sum_{m=1}^{K} w(m)\hat{F}(m,t)}{(\sum_{m=1}^{K} \hat{F}(m,t))^2} \cdot \frac{\partial E(t)}{\partial w_{ij}^l}(k) \\
&= \frac{\sum_{k=1}^{K} \hat{F}(k,t) \log F(k,t) \left(w(k) - \hat{f}_{MEAN}(t)\right) \frac{\partial E(t)}{\partial w_{ij}^l}(k)}{\sum_{k=1}^{K} \hat{F}(k,t)},
\end{aligned} \tag{109}$$

$$\frac{\partial \hat{F}(k,t)}{\partial g(k,t)} = \hat{F}(k,t) \log F(k,t),$$

$$\frac{\partial E}{\partial w_{ij}^l} = \frac{1}{T}\sum_{t=1}^{T} \frac{\partial E(t)}{\partial w_{ij}^l},$$

where $\frac{\partial E(t)}{\partial w_{ij}^l}(k)$ is the kth derivative corresponding to the kth input, kth output and time instant t with respect to the squared error $E(t)$.

Results

Since the distribution of the respiration frequencies is concentrated to lower breathing frequencies training and testing sets were sampled from smoothed distribution with an equal amount of data between respiration frequencies from 0.03 to 1.3 Hz. A total of 1000 randomly drawn samples (time instants) were used for both training and testing. Thus, the total number of training and testing samples contained only 0.3% of the data. Training was performed with a different number of hidden units to find the optimal network architecture. Testing error was utilized for model selection.

	$\hat{f}_{MEAN}$	$\hat{f}_{MOD}$	hr
$\hat{f}_{MEAN}$	1	0.5907	0.2602
$\hat{f}_{MOD}$	0.5907	1	0.5298
hr	0.2602	0.5298	1

Table 5: Cross-correlations between different features in the respiration detection procedure.

The optimization was performed with Matlab Optimization Toolkit's FMINUNC-function specialized in unconstrained nonlinear optimization using Levenberg-Marquardt-algorithm to approximate the Hessian matrix [102].

The TFRD was calculated with 255 frequency bins but only the frequencies between 0.03 to 1.3 Hz were considered. Thus, the resulting time-frequency matrix dimensions were 65×631500 for the full dataset. This is equal to 1.3 gigabytes of information with double-precision (32 bits). Thus, the optimization of the whole dataset would be computationally very expensive, so that training and testing sets were utilized. Also using the whole dataset would danger generalization of the model, since the model would be biased towards the lower breathing frequencies. Furthermore, the length of the short-time Fourier transformation Hanning-window length was chosen to be 255 (or 51 seconds).

The time series correlations between the true respiratory frequency and neural network inputs were calculated for the dataset. The correlations were 0.6415, 0.4995 and 0.6652 for the averaged heart rate, mean and mode frequency, respectively. If the mode frequency was calculated including ULF-LF frequency bands, then the correlation would be left to 0.1277. Furthermore, the mean-squared errors between the mean and mode frequency and true respiratory frequency were 0.0202 and 0.0111, respectively. Without filtering these instantaneous frequency moments can be considered as the best pre-estimates for the respiratory frequency.

The correlation between the features presents the additional information a feature contributes. A high, close to one, positive or negative correlation between the features suggests that the two features are similar. The cross-correlations between feature combinations are presented in Table 5. The analysis suggests that the features are uncorrelated and contribute additional information to the system.

The normalized TFRD feature may not be comprehended in a similar manner as the discussed time features. Each time feature is a pre-estimate for the breathing frequency. Instead, the normalized TFRD contributes the overall shape of the instantaneous spectrum providing the amplitude information to the calculus. Furthermore, the frequency bins contribute the location of the spectrum amplitude.

The whole procedure consisted of calculating over thousand different local minima for different two-layered feed-forward neural networks initialized with different initial conditions. The optimal network chosen with the test set had fifteen hidden units, with a total of $(5+1) \cdot 15 + (15+1) = 106$ network parameters.

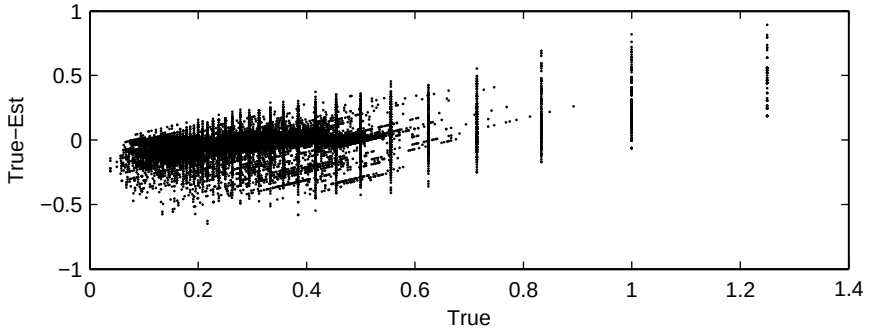

Figure 65: A scatter plot illustrating the distribution of residuals as a function of a true respiration frequency.

The error for the whole dataset was MSE=0.0047 and correlation between the estimated and true respiration frequency was 0.8579. This shows that the optimized TFRD outperforms the pre-estimates.

Notice that if the feed-forward neural network would have been used in the conventional manner by feeding all the inputs to the network at once, the number of inputs would have been $65 + 65 + 1 + 1 + 1 = 133$. Furthermore, the shape of the filter would have been calculated instantly requiring 65 outputs. The resulting network with fifteen hidden units would have had $(133+1)\cdot 15+(15+1)\cdot 65 = 3050$ parameters instead of 106.

Figure 65 demonstrates a scatter plot illustrating the distribution of residuals as a function of a true respiration frequency. The plot includes the whole dataset. The result seems to give linear bias towards a positive difference. This suggests that we have a suboptimal solution for the problem and further analysis is required. However, this analysis is left for future work. The demonstration is satisfactory as the results indicate an improved system for the breathing frequency estimation. The object of this study is to illustrate the properties and applicability of the transistor network to physiological modeling, not to claim an optimal solution.

Figure 66 illustrates an example heart rate time series, true respiration frequency and estimated respiration frequency of the system. As may be verified the system gives an average frequency response depending on the time resolution set for the TFRD. Diminishing of the time window would probably result in oscillations and increase the overall error. However, optimal time resolution was not searched for in this demonstration.

Figures 67-69 illustrate the shape of the neural network adaptive filter with different input conditions together with the original and weighted spectrums. The adaptive nature of the filter depending on its input is apparent. Even the relative number of the network parameters was small compared to the traditional approach, fifteen hidden units is quite many. This high number of network param-

eters is a result of the requirement of the complicated and dynamically changing filter shapes. To include different shapes, the network requires more parameters to adapt into the various inputs.

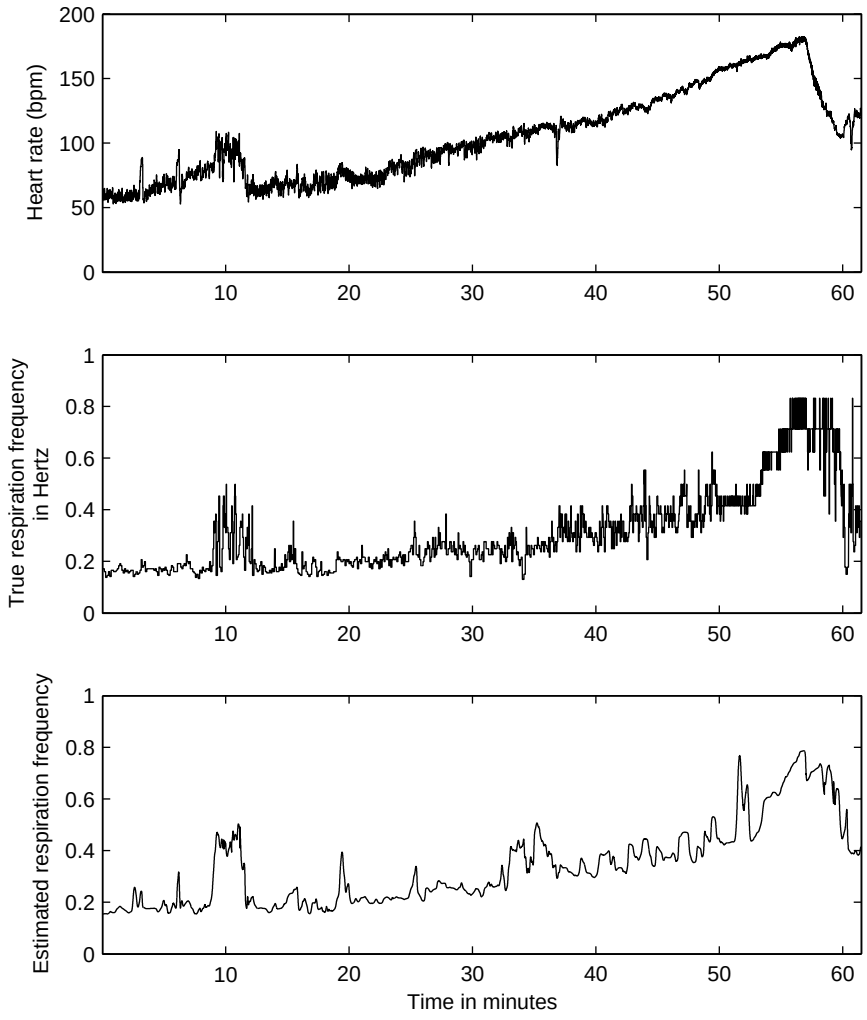

Figure 66: The upper figure illustrates a heart rate time series of a maximal uptake test of a person. The middle figure presents the true respiration frequency during the exercise while the bottom figure is the estimation derived from the heart rate time series.

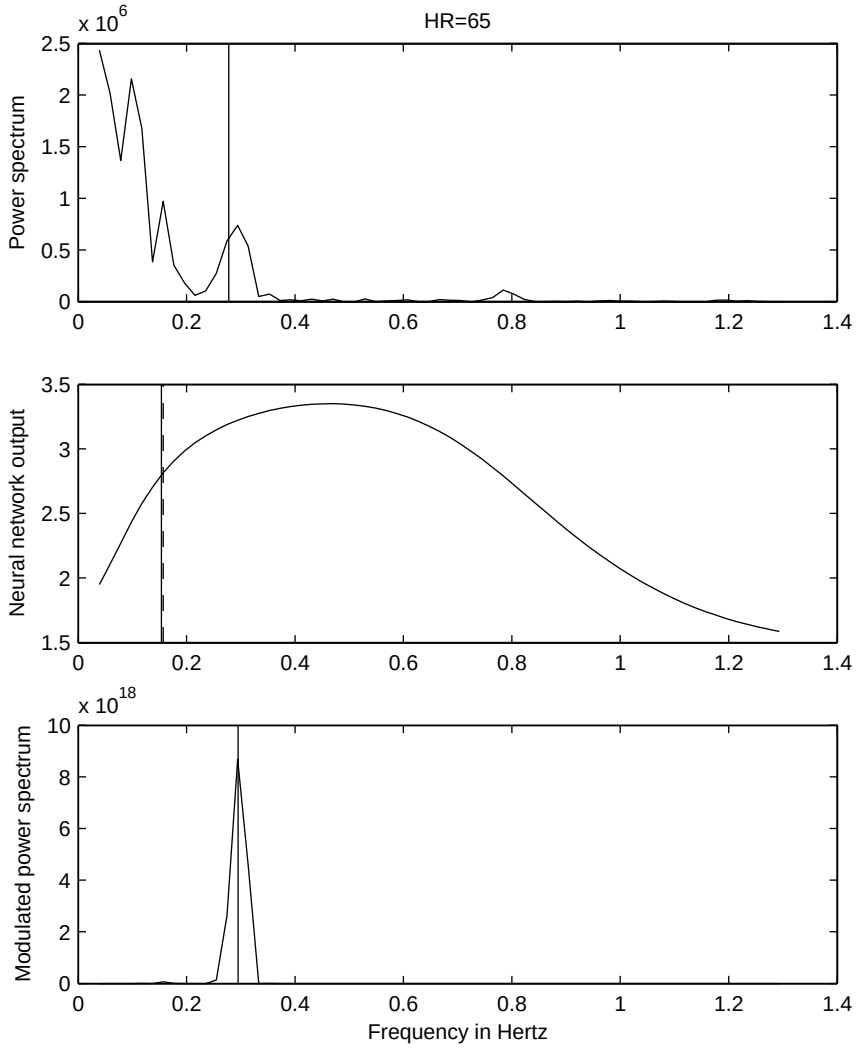

Figure 67: A snapshot of the respiratory detection procedure. The upper figure presents the original TFRD together with the true respiratory frequency for the given time moment illustrated by a horizontal line. Above the figure is presented the mean heart rate for this time instant. The middle figure illustrates the neural network produced time-frequency weighting together with mean and mode frequencies of the original TFRD (solid and dashed lines). The bottom figure demonstrates the resulting weighted spectrum with the mean frequency presented by a horizontal line.

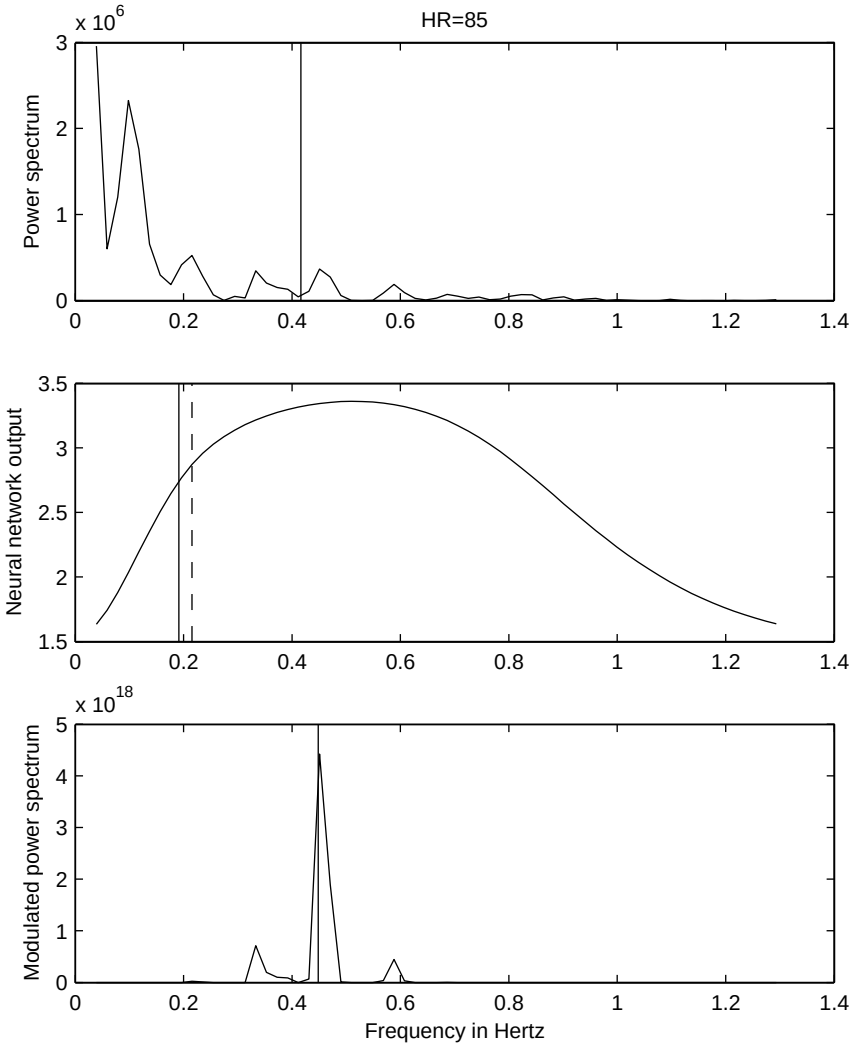

Figure 68: A snapshot of the respiratory detection procedure. The upper figure presents the original TFRD together with the true respiratory frequency for the given time moment illustrated by a horizontal line. Above the figure is presented the mean heart rate for this time instant. The middle figure illustrates the neural network produced time-frequency weighting together with mean and mode frequencies of the original TFRD (solid and dashed lines). The bottom figure demonstrates the resulting weighted spectrum with the mean frequency presented by a horizontal line.

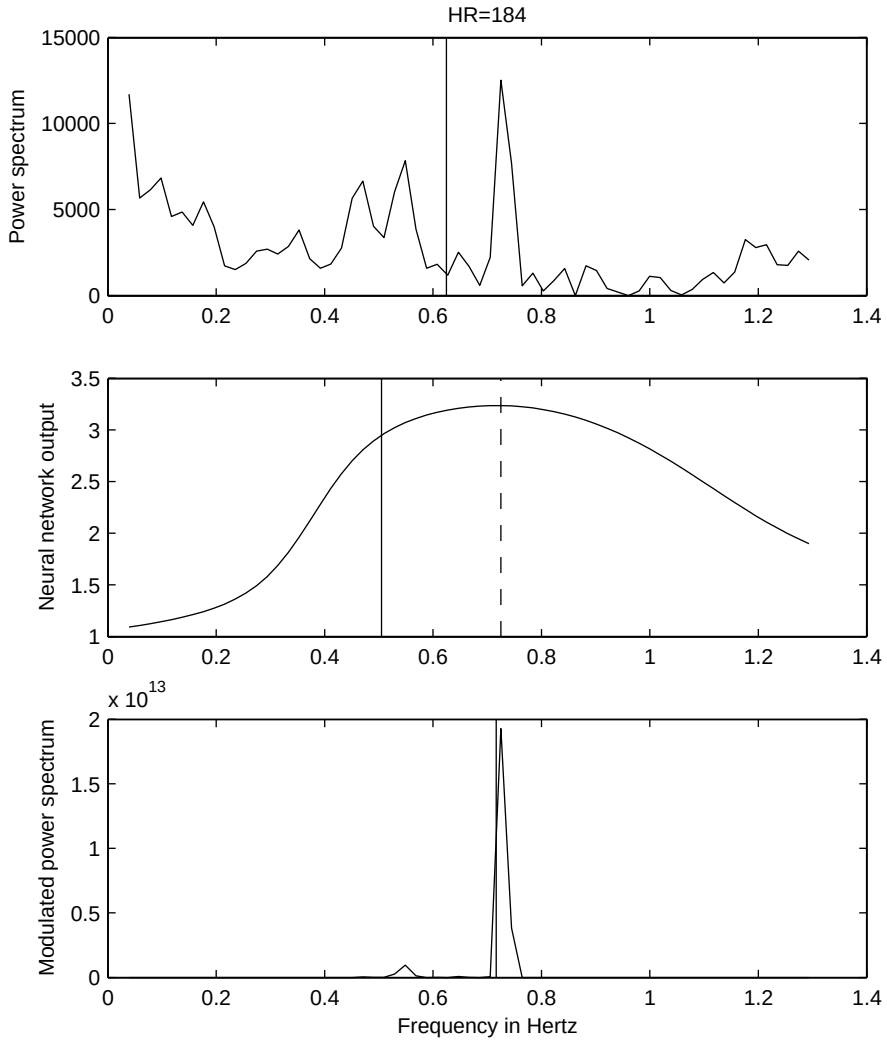

Figure 69: A snapshot of the respiratory detection procedure. The upper figure presents the original TFRD together with the true respiratory frequency for the given time moment illustrated by a horizontal line. Above the figure is presented the mean heart rate for this time instant. The middle figure illustrates the neural network produced time-frequency weighting together with mean and mode frequencies of the original TFRD (solid and dashed lines). The bottom figure demonstrates the resulting weighted spectrum with the mean frequency presented by a horizontal line.

6.3.3 Applying generalized regression neural network for respiratory frequency detection

Another setup for the respiratory frequency detection was utilized with a generalized regression neural network introduced in Section 4.3.2. The data presented in the previous section was utilized in the model building.

The peculiarity of the GRNN, and radial basis function networks in general, is that we may construct a reliability measure, e.g., based on a mean firing intensity of the network at the given time instant (see equation (71)), for the network output. The reliability estimation in (71) basically measures the similarity of the network inputs and the prototypes. Thus, it is assumed that the network is trained with an ideal, unambiguous set and similar inputs should be mapped to the same output. If an input is distant from all the prototypes it is "unfamiliar" resulting in small reliability.

In addition to the reliability estimation, the GRNN introduces two different methods for the possible training procedures, which could also be combined: The unknown network weights may be found in a supervised or unsupervised manner or by using the network weights found with unsupervised learning as initialization for the supervised training. In this demonstration, the K-means clustering algorithm was applied to the training data and different error estimates were calculated. The second step included supervised training of the GRNN with given input-output samples based on the network trained with the K-means clustering in the previous step. Supervised learning was based on analytic gradients solved in Section 4.3.2 and gradient descent algorithm.

In the pre-analysis different inputs were experimented for the GRNN. The initial model included similar inputs as the transistor network model in the previous section. However, it appeared that it was not possible to utilize the spectral information of the time series in the modeling. This may be the result of several factors. Perhaps prototyping the normalized spectrum is impossible as the non-stationary of the heart rate time series results in an infinite number of different spectral shapes. In addition, the short-time Fourier-transformation operates with a pre-defined window length resulting in an average spectrum containing several distinguishable frequency components. The innovation in the dynamic filtering approach was to reduce the number of these components based on pre-inputs.

As the spectral information was not exploitable only three network inputs were chosen (presented in the previous section): average heart rate, mean, and mode instantaneous frequencies. Furthermore, the training and testing data was selected from a smooth distribution to prevent the model from specializing to the most frequent samples, in order to achieve better generalization. A total of 2000 training samples were generated. The testing set consisted of the whole database, 35 hours of data.

The reliability estimate was applied to the time domain corrections, or post-correction, of the network's output. Three different correction heuristics were applied for the whole dataset. In the first method, the time instants, where the

deviation estimates $rb(t)$ are below the defined threshold, are interpolated by the surrounding values having higher reliability. In the second correction heuristic the reliability weighted average defined in (34) is utilized. The third correction is pure Hanning averaging, as smoothing is assumed to decrease the overall error.

Results

Table 6 presents the outcome of the experiment. A total of 400 different local minima were calculated with a different number of network prototypes. In addition, a suitable averaging window was searched for the post-correction heuristics. The optimal result for each possible setup is presented as a mean-squared error between the estimate and target respiratory frequency.

The training error decreased in both supervised and unsupervised learning as more network prototypes were introduced. However, only in unsupervised learning the resulting model also decreases the overall (testing) error. This may be due to the preservation of locality of the neurons in K-means clustering. In gradient descent optimization, the locality of the neurons is not maintained and the network may overfit increasing the overall error.

All the correction heuristics were able to diminish the error for the dataset. The post-correction with interpolation resulted in a minor improvement. Smoothing averaging appeared to be the optimal post-correction method for this application. As the optimal window lengths were quite large the difference between the reliability and pure Hanning smoothing was insignificant. This suggests that in the present case the average response is not affected much by the local differences between the reliability estimates as the smoothing operates in moving windows larger than two minutes.

In this application the reliability based corrections did not appear to be very effective or had only a minor effect. In the example presented in Section 5 the time domain corrections were able to diminish the overall error considerably with the artificial data. It seems that neither of the examples should be considered as proof for the applicability of the post-correction in the time domain. It may only be speculated that for some time series the post-correction may appear to be valuable and should be considered.

(#Prototypes/ #params)	K-means clustering					Gradient Descent				
	E_{tr}	E_{all}	E_{C_1}	E_{C_2}	E_{ave}	E_{tr}	E_{all}	E_{C_1}	E_{C_2}	E_{ave}
5/41	0.0111	0.0087	0.0083	0.0077/120	0.0078/160	0.0092	0.0072	0.0070	0.0063/120	0.0064/140
10/81	0.0102	0.0080	0.0077	0.0071/140	0.0072/160	0.0087	0.0069	0.0068	<u>0.0060</u>/120	0.0061/140
15/121	0.0100	0.0082	0.0078	0.0071/120	0.0072/160	0.0080	0.0067	0.0067	0.0061/120	<u>0.0060</u>/120
20/161	0.0098	0.0078	0.0074	0.0067/120	0.0069/160	0.0081	0.0068	0.0068	0.0061/120	0.0061/120
30/241	0.0089	0.0074	0.0071	0.0064/120	0.0065/160	0.0081	0.0068	0.0067	0.0061/120	0.0061/120
50/401	0.0085	0.0071	0.0070	0.0064/120	0.0064/120	0.0080	0.0071	0.0070	0.0063/120	0.0062/140

Table 6: Table presenting the results of the generalized regression neural network applied to respiratory frequency detection. Mean-squared errors of the training, testing, interpolation- , weighted average- and average corrected outputs (E_{tr}, E_{all}, E_{C_1}, E_{C_2}, E_{ave}, respectively) are compared between the two training heuristics, K-means clustering (unsupervised learning) and gradient descent optimization (supervised learning). The minimum errors are underlined. In the Hanning window and reliability weighted averaging the corresponding window lengths (in seconds) are presented as MSE/window length-pairs.

6.3.4 PCA and FFNN for respiratory frequency estimation

A classical approach for multivariate time series modeling with a feed-foward neural network is to use the *principal component analysis* (PCA) to reduce the dimensions of the input vectors and then to train the network. PCA has three effects: First it orthogonalizes the components of the input vectors, so that they are uncorrelated with each other. In addition to this, it orders the resulting orthogonal components, i.e., principal components, so that those with the largest variation come first. At last it eliminates those components that contribute the least to the variation in the data set [68, 101].

To apply the idea we use PCA to reduce the inputs presented in (108) and estimate the respiration frequency with a FFNN directly. The initial input vector included the following features for each time instant t:

$$\frac{F(1,t)}{\max_k F(k,t)}, \ldots, \frac{F(K,t)}{\max_k F(k,t)}, f_{MOD}(t), f_{MEAN}(t), \frac{hr(t)}{200}.$$

Here the frequency bin vector $w(k)$ was not included, as it would have contained only the same information for each time instant. Hence, the total number of network inputs were 68 before applying the PCA.

Principal component analysis was carried out with the Matlab Neural Network Toolkit's PREPCA-function. It eliminates those principal components that contribute less than p% to the total variation in the data set. Different values for p were experimented in the training. Over 2000 local minima were calculated for the various number of hidden units (between 6 and 20) in the network. Training and testing sets were used in a similar manner as in Section 6.3.2. In addition, the inputs were normalized before applying the PCA. Table 7 gathers the results. The best MSE resulted in 0.0070 for the FFNN model.

PCA(p)	#inputs	#hidden units	#params	C_P	MSE
0.5%	38	20	801	0.7470	0.0083
1.0%	26	14	393	0.7498	0.0082
2.0%	12	20	281	0.7529	0.0083
3.0%	6	12	97	0.7888	0.0070
4.0%	3	16	81	0.7836	0.0071
5.0%	2	10	41	0.7625	0.0077

Table 7: Pearson correlations and the mean-squared errors between the FFNN estimates and the true respiration frequencies. The correlations and errors were calculated for the whole dataset. Various architectures for FFNN and different values for PCA parameter p were experimented.

6.3.5 Discussion

In the first subsection time-frequency distributions of the heart rate time series was analyzed to introduce the base numerical approach to reveal respiratory frequency component of the signal. It appeared that in some steady conditions, as in metronome-spaced breathing, the RSA component of the heart rate was distinct. However, under spontaneous or ambulatory recording the resulting heart rate time series appears nonstationary adding several major frequency components in the signal. Especially the 0.1 hertz component of the HR-signal is often dominant reflecting the rhythmic changes in blood pressure control.

Also the RSA component itself is nonstationary. For example, speech irregulates the breathing pattern resulting in a difficult instantaneous detection of the RSA with, e.g., Gabor transformation. Gabor transformation only gives average periodic spectral shape of the signal for a given time instant. Methods like Wavelet transformation or smoothed Pseudo Wiegner-Ville may offer sharper time-resolution but are less stable and in our experience are not suitable for the given problem.

Three different approaches for detection of respiratory frequency from the heart rate time series were introduced. The neural network architectures contained different properties and perspectives for the modeling. The transistor-network based dynamic filtering attempted optimization and weighting of the TFRD to expose the hidden respiratory component. With the GRNN an assumption was that a set of features can present the input-output mapping of the phenomena by prototyping an adequate set of input-space combinations. The FFNN was used to estimate the respiration frequency directly. GRNN was optimized with both in a supervised and unsupervised manner, while the other two models were optimized with a supervised learning strategy.

Table 8 lists the best results of the models. Naturally we may apply the time domain correction, not only to the GRNN network, but also to the transistor network and FFNN by using two minutes Hanning window to moving average the results. With the transistor network the resulting MSE decreases slightly from 0.0047 to 0.0044. Smoothing the FFNN respiration frequency estimate with the same window decreases the mean-squared error from 0.0070 to 0.0060. Hence, the FFNN and GRNN estimates produce a similar estimation error.

It appeared that for this application dynamic filtering was an advantageous method, able to process the time- and frequency domain information in a compact

Model	MSE	#parameters
Transistor network	0.0044	106
FFNN	0.0060	97
GRNN	0.0060	81

Table 8: The best post-corrected results of the three models presented for respiration detection.

and efficient way resulting in a decreased error. As a result an average breathing frequency is revealed from a full breathing scale as Figure 66 suggests.

Even if the GRNN was not as effective as the transistor network, the analysis revealed two important factors: moving averaging may be used to enhance the quality of the signal and resulting estimate. In addition, the reliability estimate may be exploited in the time domain correction.

In Section 5.1.2 optimization of the objective function including the deviation estimate was introduced. Naturally GRNN optimized with supervised learning could include the deviation estimate in the error function. The approach could better preserve the locality of the prototypes, thus resulting in enhanced reliability estimates. However, the overall error of the system could increase. The interesting question is, if there could be an optimal regularization parameter contributing enhanced reliability and optimal time domain correction (cf. [88]).

Notice that the number of prototypes in GRNN was 81, thus one additional feature would result in $2 \cdot 81 = 162$ extra parameters[21]. This is the disadvantage of the GRNN: each new feature increases the number of network parameters relative to the number of prototypes. As complex maps require a large number of prototypes, the networks may become quite large and impractical as more memory and CPU-time is required.

Hybrid models with a discrete decision plane were introduced in Section 5.1. In the patent by Kettunen and Saalasti [77] the preferred embodiment included output space optimized hybrid model, where different time-frequency features were combined to decrease the overall error. For example, the moving window length in Gabor transformation may introduce one set of features. Short window length may appear optimal to reveal the breathing frequency, e.g., during heavy exercise when the breathing pattern is relatively high. Increased time resolution is achieved with the shorter window length. Also different frequency bands could be used to generate features. Clearly, this could be exploited in the GRNN by introducing new set of features based on different parameterizing of the given TFRD. In the patent by Kettunen et al. [77] the GRNN is suggested as one alternative integration function for breathing frequency feature combination.

[21] As the GRNN uses the weighted Euclidean distance, additional feature produces two extra parameters for each prototype.

7 CONCLUSIONS

The physiological time series are often complex, nonstationary, nonlinear and unpredictable. Especially, the ambulatory measurement offers challenge for the analysis and used methods, by introducing an increased measurement error and signal artifacts. Furthermore, interpretation and statistics of heart rate data are distorted by mathematical operations like nonlinear transformations or resampling of the data. The complexity of the heart rate signals were brought forth with discussion, examples and visualization. In spite of the difficulties, we introduced methodology which were able to quantify and model the heart rate data. Thus, several innovations combining human physiology and mathematical modeling were presented in the examples:

1. Utilization of individual physiological parameters like maximal heart rate and oxygen consumption were demonstrated to improve the explanation value of the proportionally scaled signal. Furthermore, the physiological constraints were proposed to be exploited in on-line applications to form a normalization, or scaling, of nonstationary signals.

2. Data ranking was demonstrated to preserve signal rhythm diminishing the heart rate signal acceleration, resulting in an improved estimation of frequency components in the spectral analysis of the signal.

3. A new peak detection algorithm was applied for the estimation of the respiration frequency from chest expansion data resulting in perfect time-frequency resolution.

4. Postprocessing and time domain corrections appeared valuable for physiological time series as the adjacent time instants are coupled and do not differ substantially. Reliability estimates were successfully exploited with the post-corrections and a new heuristic for estimating reliability of instantaneous frequency for time-frequency distributions was presented. In addition, a generalized regression neural network, peak detection algorithm and HMDD were demonstrated to include a natural interpretation of the reliability estimation.

5. A transistor neural network, feed-forward neural network and generalized regression neural network were successfully utilized to extract the respiratory frequency from the heart rate time series.

6. Physiological constraints were utilized for model selection in neural network training. The resulting EPOC model was able to extrapolate to unseen values in a physiologically valid way.

These innovations offer new insights to physiological time series modeling, and to our knowledge, are new and as yet not published.

Neural network architectures and optimization

Neural networks are universal approximators introducing powerful and flexible nonlinear models with several applications. However, the most complex processes is the choice of the appropriate network architecture and optimization method to find the unknown weights of the network.

In this dissertation the presented neural network architectures included two dimensions: static vs. dynamic network architecture and local vs. global neurons. Global neurons include models like FFNN, FIR network and Jordan network that use sigmoidal activation. Network architectures including local neurons are radial basis function- and generalized regression neural networks operating with Gaussian activation functions and Euclidean distance between the network input and prototypes. It was demonstrated how these networks include the different characteristic properties and strengths, for example, networks with local neurons offer a natural interpretation of reliability estimates.

Temporal neural networks, like FIR network and Jordan network, may be unfolded to follow the static structure. However, this should be applied only to network training. Temporal neural networks are very different to their static counterparts when they are run for new data and especially with different lengths of data. The applicability of dynamic networks, generation of synthetic observations and the use of reliability weighting of training samples in the objective (error) function were demonstrated with the modeling of excess post-exercise oxygen consumption.

Several classical methods to improve backpropagation or network performance were reviewed. Moreover, modifications and improvements for network training are constantly published. Non of these improvements were utilized for the examples presented in the dissertation: We wish to emphasize that network training is basically a nonlinear optimization problem and it should be treated as one. Hence, we used a general optimization solver and searched several local minima to find the appropriate parameter set using cross-validation. The number of hidden units varied during the model search to select an appropriate model complexity. In addition, we used physiological constraints for the model selection with the EPOC model instead of using constraints in the objective function. Also a formulation of FFNN and FIR network in a matrix form was presented improving the analytic presentation value of the backpropagation equations.

The signal artifacts and distribution of the target signal biases the neural network model towards the outliers and most frequent samples. Hence, we select an even distribution of the training data to improve the generalization of the neural network model.

It was demonstrated how the divide and conquer-approach for classical hybrid models does not lead to modularity of the resulting network, when the parameters of the expert and integration functions are optimized simultaneously. It may be speculated that it is possible to find a local minimum that includes

the wished properties. However, this may realize improbable and instead of a hybrid model with a discrete decision plane was introduced, where the integration function is optimized separately from the experts improving the possibility of constructing a modular system. Also tools for measuring and monitoring the modularity were presented.

A common usage for a neural network model is to form a model by optimizing it in respect to some defined target signal. A new concept, a transistor neural network, was introduced to expand the applicability and flexibility of neural network architectures. It was also successfully applied for the modeling of a respiratory frequency from the heart rate time series, where traditional approaches were outperformed.

REFERENCES

[1] P. Augustyniak. Recovering the precise heart rate from sparsely sampled electrocardiograms. w materialach konferencji Computers in Medicine, Lódz, 23-25.09.1999, str. 59-65, 1999.

[2] P. Augustyniak and A. Wrzesniowski. Ecg recorder sampling at the variable rate. In proceedings of 6th International Conference SYMBIOSIS 2001, Szczyrk, Poland 11-13 September, 2001.

[3] A. R. Barron. Universal approximation bounds for superposition of a sigmoidal function. *IEEE Transactions on Information Theory*, 39:930–945, 1993.

[4] L. Behera. Query based model learning and stable tracking of a robot arm using radial basis function network. *Computers and Electrical Engineering*, 29:553–573, 2003.

[5] M. G. Bello. Enhanced training algorithms, and integrated training/architecture selection for multilayer perceptron networks. *IEEE Transactions on Neural Networks*, 3:864–875, 1992.

[6] G. G. Berntson, T. Bigger, D. Eckberg, P. Grossman, P. G. Kaufmann, M. Malik, H. N. Nagaraja, S. W. Porges, P. J. Saul, P. H. Stone, and M. Van Der Molen. Heart rate variability: Origins, methods, and interpretative caveats. *Psychophysiology*, 34:623–648, 1997.

[7] G. G. Berntson, K. S. Quigley, J. F. Jang, and S. T. Boysen. An approach to artifact identification: application to heart period data. *Psychophysiology*, 27(5):586–598, 1990.

[8] G. G. Berntson and J. R. Stonewell. ECG artifacts and heart period variability: Don't miss a beat! *Psychophysiology*, 35:127–132, 1998.

[9] D. Bhattacharya and A. Antoniou. Design of equirriple FIR filters using a feedback neural network. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 45(4):527–531, 1998.

[10] G. Bienvenu. Influence of spatial coherence of the background noise on high resolution passive methods. In *Proceedings of the International Conference on Acoustics, Speecs, and Signal Processing, Washington, DC*, pages 306–309, 1979.

[11] S. A. Billings and X. Hong. Dual-orthogonal radial basis function networks for nonlinear time series prediction. *Neural Networks*, 11:479–493, 1998.

[12] C. M. Bishop. Training with noise is equivelant to tikhonov regularization. *Neural Computation*, 7(1):108–116, 1995.

[13] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Somerset 1997.

[14] A. Bortoletti, C. D. Fiore, S. Fanelli, and P. Zellini. A new class of quasi-newtonian methods for optimal learning in MLP-networks. *IEEE Transactions on Neural Networks*, 14(2):263–273, 2003.

[15] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice-Hall, Inc., USA 1994.

[16] D. S. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1988.

[17] E. T. Brown, L. Beightol, J. Koh, and D. Eckberg. Important influence of respiration on human r-r interval power spectra is largely ignored. *Appl. Physiol.*, 75(5):2310–2317, 1993.

[18] L. Burattini, W. Zareba, J. P. Couderc, J. A. Konecki, and A. J. Moss. Optimizing ecg signal sampling frequency for t-wave alternans detection. *Computers in Cardiology*, 25:721–724, 1998.

[19] P. Campolucci. *A Circuit Theory Approach to Recurrent Neural Network Architectures and Learning Methods*. PhD thesis, Universita Degli Studi Di Bologna, Dottorato di Ricerca in Ingegneria Elettrotecnica, 1998.

[20] G. Camps-Valls, B. Porta-Oltra, E. Soria-Olivas, J. D. Martin-Guerrero, A. J. Serrano-López, J. Pérez-Ruixo, and N. V. Jiménez-Torres. Predicion of cyclosporine dosage in patients after kidney transplantation using neural networks. *IEEE Transactions on Biomedical Engineering*, 50(4):442–448, 2003.

[21] D. Chakraborty and N. R. Pal. A novel training scheme for multilayered perceptrons to realize proper generalization and incremental learning. *IEEE Transactions on Neural Networks*, 14(1):1–14, 2003.

[22] C. Charalambous. Conjugate gradient algorithm for efficient training of artificial neural networks. *IEEE Proceedings*, 139(3):301–310, 1992.

[23] C. Chatfield. *The Analysis of Time Series: An Introduction*. Chapman and Hall, 5 edition, 1999.

[24] C. Chatfield. *The Analysis of Time Series: An Introduction*. Chapman and Hall, Great Britain 1991.

[25] Y. P. Chen and P. M. Popovich. *Correlation: Parametric and Nonparametric measures*. Sage University Papers Series on Quantitative Applications in the Social Sciences, 07-139, Thousand Oaks, CA: Sage, 1999.

[26] C. Chui. *An introduction to wavelets*. Academic Press, San Diego, 1992.

[27] A. Cohen. Biomedical signals: Origin and dynamic characteristics; frequency domain analysis. In J. D. Bronzino, editor, *The biomedical engineering*, pages 805–827. CRC Press, Inc., 1995.

[28] L. Cohen. Time-frequency distributions - a review. *Proceedings of the IEEE*, 77:941–981, 1989.

[29] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to algorithms*. Cambridge (Mass.) : MIT Press, 20 edition, 1998.

[30] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Control, Signals, and Sys.*, 2(4), 1989.

[31] J. Daintith and R. D. Nelson, editors. *Dictionary of mathematics*. Penguing Group, 1989.

[32] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.

[33] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. New York: Prentice-Hall, 1983.

[34] H. Drucker, C. Cortes, L.D. Jackel, and Y. LeCun. Boosting and other ensemble methods. *Neural Computation*, 6:1289–1301, 1994.

[35] H. Drucker, R. E. Schapire, and P Simard. Improving performance in neural networks using a boosting algorithm. *Advances in Neural Information Processing Systems*, 5:42–49, 1993.

[36] S. Fahlman. Faster learning variations on back-propagation: An empirical study. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 38–51. Morgan Kaufmann, 1989.

[37] C. L. Fancourt and J. C. Principe. On the use of neural networks in the generalized likelihood ratio test for detecting abrupt changes in signals. *Intl. Joint Conf. on Neural Networks*, pages 243–248, 2000.

[38] Y. Freund. Boosting a week learning algorithm by majority. *Information Computation*, 121:256–285, 1995.

[39] Y. Freund and R. E Schapire. Experiments with a new boosting algorithm. *Machine Learning: Proceedings of the Thirteenth International Conference, Bari, Italy*, pages 148–156, 1996.

[40] Y. Freund and R. E Schapire. Game theory, on-line prediction and boosting. *Proceedings of the Ninth Annual Conference on Computational Learning Theory, Desenzano del Garda, Italy*, pages 325–332, 1996.

[41] Y. Freund and R. E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

[42] G. M. Friesen, T. C. Jannett, M. A. Jadallah, S. L. Yates, S. R. Quint, and H. T. Nagle. A comparison of the noise sensitivity of nine qrs detection algorithms. *IEEE Transactions on Biomedical Engineering*, 37(1):85–98, 1990.

[43] K. I. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2:183–192, 1989.

[44] A. Grossmann and J. Morlet. Decomposition of hardy functions into square integrable wavelets of constant shape. *SIAM Journal of Mathematical Analysis*, 15:723–736, 1984.

[45] A. C. Guyton. *Textbook of medical physiology*. W.B. Saunders company, 7 edition, 1986.

[46] A. C. Guyton and J. E. Hall. *Textbook of medical physiology*. W.B. Saunders company, 9 edition, 1996.

[47] M. T. Hagan, H. B. Demuth, and M. H. Beale. *Neural Network Design*. PWS Publishing, 1996.

[48] M. T. Hagan and M. Menhaj. Training feedforward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6):989–993, 1994.

[49] L. O. Hall, A. M. Bensaid, L. P. Clarke, R. B. Velthuizen, M. S. Silbiger, and J. C. Bezdek. A comparison of neural network and fuzzy clustering techniques in segmenting magnetic resonance images of the brain. *IEEE Transactions on Neural Networks*, 3(5):672–682, 1992.

[50] S. J. Hanson and L. Y. Pratt. Comparing biases for minimal network construction with back-propagation. *Advances in Neural Information Processing Systems*, 1:177–185, 1989.

[51] E. J. Hartman, J. D. Keeler, and J. M. Kowalski. Layered neural networks with Gaussian hidden units as universal approximations. *Neural Computation*, 2:210–215, 1990.

[52] B. Hassibi and D. G. Stork. Second order derivatives for network pruning: optimal brain surgeon. *Advances in Neural Information Processing Systems*, 5:164–171, 1993.

[53] S. Haykin. *Addaptive Filter Theory*. Prentice Hall, 2002.

[54] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice-Hall, Inc., New Jersey 1994.

[55] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice-Hall, Inc., 2 edition, New Jersey 1999.

[56] G. M. Hägg. Comparison of different estimators of electromyographic spectral shifts during work when applied on short test conditions. *Med Biol Eng Comp*, 29:511–516, 1991.

[57] G. E. Hinton. Learning translation invariant recognition in massively parallel networks. In J. W. de Bakker, A. J. Nijman, and P. C. Treleaven, editors, *Proceedings PARLE conference on parallel architectures and Languages Europe*, pages 1–13. Berlin: Springer-Verlag, 1987.

[58] K. Hornik, M. Stinchombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[59] J. H. Houtveen, S. Rietveld, and E. J. C. De Geus. Contribution of tonic vagal modulation of heart rate, central respiratory drive, respiratory depth, and respiratory frequency to respiratory sinus arrhythmia during mental stress and physical exercise. *Psychophysiology*, 39:427–436, 2002.

[60] T. S. Huang, G. J. Yang, and G. Y. Tang. A fast two-dimensional median filtering algorithm. *IEEE transactions on acoustics, speech and signal processing*, 27:13–18, February 1979.

[61] H. V. Huikuri, T. Mäkikallio, J. Airaksinen, R. Mitrani, A. Castellanos, and R. Myerburg. Measurement of heart rate variability: A clinical tool or a research toy? *Journal of the American College of Cardiology*, 34(7):1878–1883, 1999.

[62] D. Husmeier. Learning non-stationary conditional probability distributions. *Neural Networks*, 13:287–290, August 2000.

[63] B. Irie and S. Miyake. Capabilities of three-layered perceptrons. *Proceedings IEEE Second International Conference on Neural Networks*, 1:641–647, 1988.

[64] A. S. Jackson, S. N. Blair, M. T. Mahar, L. T. Wier, R. M. Ross, and J. E. Stuteville. Prediction of functional aerobic capacity without exercise testing. *Medicine & Science in Sports and Exercise*, 22(6):863–870, 1990.

[65] R. A. Jacobs. Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1:295–307, 1988.

[66] R. A. Jacobs. *Task Decomposition Through Computation in a Modular Connectionist Architecture*. PhD thesis, University of Massachusetts, 1990.

[67] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991.

[68] I. T. Jollife. *Principal Component Analysis*. New York: Springer-Verlag, 1986.

[69] L. K. Jones. A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training. *Annals of Statistics*, 20:608–613, 1992.

[70] M. I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. *Proceedings 8th Annual Conference of the Cognitive Science Society*, pages 531–546, 1986.

[71] M. I. Jordan. *A Parallel Distributed Processing Approach*. University of California, technical report 8604, 1986.

[72] P. G. Katona and F. Jih. Respiratory sinus arrhythmia: noninvasive measure of parasympathetic cardiac control. *Journal of Applied Physiology*, 39(5):801–805, 1975.

[73] A. Kehagias and V. Petridis. Time-series segmentation using predictive modular neural networks. *Neural Computation*, 9:1691–1709, 1997.

[74] S. Kendall. *The Unified Process Explained*. Addison-Wesley, 2001.

[75] J. Kettunen and L. Keltinkangas-Järvinen. Smoothing enhances the detection of common structure from multiple time series. *Behaviour Research Methods, Instruments & Computers*, 33(1):1–9, 2001.

[76] J. Kettunen, J. Kotisaari, S. Saalasti, A. Pulkkinen, P. Kuronen, and H. Rusko. A system for daily monitoring of physiological resources: A pilot study. Science for Success congress, Jyväskylä, Finland, October, 2002.

[77] J. Kettunen and S. Saalasti. Procedure for deriving reliable information on respiratory activity from heart period measurement. Patent number FI20011045 (pending), 2002.

[78] J. Kettunen, S. Saalasti, and A. Pulkkinen. Patent number FI20025039 (pending), 2002.

[79] J. Kohlmorgen and S. Lemm. A dynamic HMM for on-line segmentation of sequential data. *Advances in Neural Information Processing Systems*, 14:793–800, 2001.

[80] J. Kohlmorgen, S. Lemm, K.-R. Müller, S. Liehr, and K. Pawelzik. Fast change point detection in switching dynamics using a hidden markov model of prediction experts. *Proc. of the Int. Conf. on Artificial Neural Networks*, pages 204–209, 1999.

[81] J. Kohlmorgen, K.-R. Müller, and K. Pawelzik. Segmentation and identification of drifting dynamical systems. *Neural Networks for Signal Processing*, 7:326–335, 1997.

[82] J. Kohlmorgen, K.-R. Müller, J. Rittweger, and K. Pawelzik. Identification of nonstationary dynamics in physiological recordings. *Biological Cybernetics*, 83:73–84, 2000.

[83] T. Kohonen. *Self-Organizing Maps*. Springer, 1995.

[84] M. Kollai and G. Mizsei. Respiratory sinus arrhythmia is a limited measure of cardiac parasympathetic control in man. *Journal of Physiology*, 424:329–342, 1990.

[85] A. N. Kolmogorov. On the representation of continuous functions of several variables by superposition of continuous functions of one variable and addition. *Dijkadt Akademiia Nauk SSSR*, 114:953–956, 1957.

[86] I. Korhonen. *Methods for the analysis of short-term variability of heart rate and blood pressure in frequency domain*. PhD thesis, VTT-Technical Research Centre of Finland, 1997.

[87] T. Kärkkäinen. MLP-network in a layer-wise form with applications to weight decay. *Neural Computation*, 14(6):1451–1480, 2002.

[88] T. Kärkkäinen and E. Heikkola. Robust formulations for training multilayer perceptrons. *To appear in Neural Computation*, 2003.

[89] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9(1):112–147, 1998.

[90] K. J. Lang and G. E. Hinton. Dimensionality reduction and prior knowledge in e-set recognition. *Advances in Neural Information Processing Systems*, 2:178–175, 1990.

[91] Y. Le Cun, J. S. Denker, and S. A. Solla. Optimal brain damage. *Advances in Neural Information Processing Systems*, 2:598–605, 1990.

[92] M. Lehtokangas. *Neural Networks in Time Series Modelling*. Tampere University of Technology Electronics Laboratory, Tampere 1994.

[93] S. Liehr, K. Pawelzik, J. Kohlmorgen, and K.-R. Müller. Hidden markov mixtures of experts with an application to EEG recordings from sleep. *Theory in Biosciences*, 118:246–260, 1999.

[94] H. Maaranen, K. Miettinen, and M. M. Mäkelä. Training multi layer perceptron using a genetic algorithm as a global optimizer. In M. G. C. Resende and J. P. de Sousa, editors, *Metaheuristics: Computer Decision-Making*, pages 421–448. Kluwer Academic Publishers B.V., 2003.

[95] S. Makeig, T-P. Jung, and T. J. Sejnowski. Using feedforward neural networks to monitor alertness from changes in EEG correlation and coherence. *Advances in Neural Information Processing Systems*, 8:931–937, 1996.

[96] S. A. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:674–693, 1989.

[97] K. Martinmäki, L. Kooistra, J. Kettunen, S. Saalasti, and H. Rusko. Cardiovascular indices of vagal activation as a function of recovery from vagal blockade. ACSM Congress, St. Louis, May 28 - June 1. Abstract: Medicine and Science in Sports and Exercise 34(5), Supplement: S60., 2002.

[98] K. Martinmäki, L. Kooistra, J. Kettunen, S. Saalasti, and H. Rusko. Intraindividual validation of heart rate variability indices to measure vagal activity. Science for Success congress, Jyväskylä, Finland, October, 2002.

[99] Matlab. *Time-Frequency Toolbox for use with Matlab*, 1996.

[100] Matlab. *The Language of Technical Computing*, 1999.

[101] Matlab. *Neural Network Toolbox for use with Matlab*, 2000.

[102] Matlab. *Optimization Toolbox for use with Matlab*, 2000.

[103] Matlab. *Signal Processing Toolbox for use with Matlab*, 2000.

[104] Matlab. *Wavelet Toolbox for use with Matlab*, 2002.

[105] W. D. McArdle, I. Katch, F., and V. L. Katch. *Exercise Physiology: Energy, nutrition and human performance*. Williams & Wilkins, 4 edition, 1996.

[106] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman and Hall, Great Britain 1985.

[107] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, 1994.

[108] T. Mäkikallio. *Analysis of heart rate dynamics by methods derived from nonlinear mathematics: Clinical applicability and prognostic significance*. PhD thesis, Department of Internal Medicine, University of Oulu, 1998.

[109] S. Mohsin, Y. Kurimoto, Y. Suzuki, and J. Maeda. Extraction of the qrs wave in an electrocardiogram by fusion of dynamic programming matching and a neural network. *Trans. of Institute of Electrical Engineers of Japan*, 122-C(10):1734–1741, 2002.

[110] J. Moody and C. J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989.

[111] J. Möttönen, V. Koivunen, and H. Oja. Sign and rank based methods for autocorrelation coefficient estimation. Tampere International Center for Signal Processing (TICSP) Seminar Presentation, 2000.

[112] J. Möttönen, H. Oja, and V. Koivunen. Robust autocovariance estimation based on sign and rank correlation coefficients. IEEE HOS'99, 1999.

[113] L. J. M. Mulder. *Assessment of cardiovascular reactivity by means of spectral analysis*. PhD thesis, Instituut voor Experimentele Psychologie van de Rijksuniversiteit Groningen, 1988.

[114] K. R. Müller, J. Kohlmorgen, A. Ziehe, and B. Blankertz. Decomposition algorithms for analysing brain signals. *IEEE Symposium 2000 on adaptive Systems for Signal Processing, Communications and Control*, pages 105–110, 2000.

[115] U. N Naftaly and D. Horn. Optimal ensemble averaging of neural networks. *Network*, 8:283–296, 1997.

[116] J. Nocedal and S. J. Wright, editors. *Numerical optimization*. Springer-Verlag, New York, 1999.

[117] V. Novak, P. Novak, J. DeChamplain, A. R. LeBlanc, R. Martin, and R. Nadeau. Influence of respiration on heart rate and blood pressure fluctuations. *Journal of Applied Physiology*, 74:617–626, 1993.

[118] S.J. Nowlan. Maximum likelihood competetive learning. *Advances in Neural Information Processing Systems*, 2:574–582, 1990.

[119] American College of Sports Medicine Position Stand. The recommended quantity and quality of exercise for developing and maintaining cardiorespiratory and muscular fitness, and flexibility in healthy adults. *Medicine & Science in Sports and Exercise*, 30(6):975–991, 1998.

[120] Task Force of the European Society of Cardiology, the North American Society of Pacing, and Electrophysiology. Heart rate variability: standards of measurement, physiological interpretation, and clinical use. *European Heart Journal*, 17:354–381, 1996.

[121] A. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall, 1999.

[122] E. Oropesa, H. L. Cycon, and M. Jobert. Sleep stage classification using wavelet transform and neural network. Technical Report 8, International Computer Science Institute, 1947 Center St., Suite 600, Berkeley, California 94704-1198, 1999.

[123] D. N. Osherson, S. Weinstein, and M. Stoli. Modular learning. *Computational Neuroscience*, pages 369–377, 1990.

[124] P. M. Pardalos and E. H. Romeijn, editors. *Handbook of Global Optimization Volume 2*. Kluwer Academic Publishers, 2002.

[125] J. Park and I. W. Sandberg. Universal approximation using radial basis function networks. *Neural Computation*, 3:246–257, 1991.

[126] J. Park and I. W. Sandberg. Approximation and radial basis function networks. *Neural Computation*, 5:305–316, 1993.

[127] W. D. Penny and S. J. Roberts. Dynamic models for nonstationary signal segmentation. *Computers and Biomedical Research*, 32:483–502, 1999.

[128] M. P. Perrone and L. N. Cooper. When networks disagree: ensemble methods for hybrid neural networks. *Artificial Neural Networks for Speech and Vision*, pages 126–142, 1993.

[129] M. Pfister. *Hybrid learning algorithms for neural networks*. PhD thesis, Free University Berlin, 1995.

[130] M. Pfister and R. Rojas. Speeding-up backpropagation - a comparison of orthogonal techniques. *International Joint Conference on Neural Networks*, pages 517–523, 1993.

[131] V. Pichot, J. M. Gaspoz, S. Molliex, A. Antoniadis, T. Busso, F. Roche, F. Costes, L. Quintin, J. R. Lacour, and J. C. Barthelemy. Wavelet transform to quantify heart rate variability and to assess its instantaneous changes. *Journal of Applied Physiology*, 86(3):1081–1091, 1999.

[132] M. V. Pitzalis, F. Mastropasqua, F. Massari, A. Passantino, P. Totaro, C. Forleo, and P. Rizzon. belta-blocker effects on respiratory sinus arrhythmia and baroreflex gain in normal subjects. *The Cardiopulmonary and Critical Care Journal*, 114(1):185–191, 1998.

[133] M. V. Pitzalis, F. Mastropasqua, A. Passantino, F. Massari, L. Ligurgo, C. Forleo, C. Balducci, F. Lombardi, and P. Rizzon. Comparison between noninvasive indices of baroreceptor sensitivy and the phenylephrine method in post-myocardial infarction patients. *Circulation*, 97(14):1362–1367, 1998.

[134] S. Pola, A. Macerata, M. Emdin, and C. Marchesi. Estimation of the power spectral density in nonstationary cardiovascular time series: Assessing the role of the time-frequency representations. *IEEE Transactions of Biomedical Engineering*, 43:46–59, 1996.

[135] R. Poli, S. Cagnoni, and G. Valli. Genetic design of optimum linear and nonlinear QRS detectors. *IEEE Transactions on Biomedical Engineering*, 42(11):1137–41, 1995.

[136] S. W. Porges and E. A. Byrne. Research methods for measurement of heart rate and respiration. *Biological Psychology*, 34:93–130, 1992.

[137] L. Prechelt. Early stopping - but when? In G. B. Orr and K. R. Müller, editors, *Neural networks; Tricks of the Trade*, pages 55–70. Berlin Heidelberg. Springer-Verlag, 1998.

[138] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The art of scientific computing*. Cambridge University Press, 2 edition, 2002.

[139] A. Pulkkinen. Uusien sykkeeseen perustuvien hapenkulutuksen arviointimenetelmien tarkkuus. Master's thesis, University of Jyväskylä, Department of Biology of Physical Activity, 2003.

[140] A. Pulkkinen, J. Kettunen, S. Saalasti, and H. Rusko. New method for the monitoring of load, fatigue and recovery in exercise training. Science for Success congress, Jyväskylä, Finland, October, 2002.

[141] A. Pulkkinen, J. Kettunen, S. Saalasti, and H. Rusko. Accuracy of VO2 estimation increases with heart period derived measure of respiration. 50th Annual Meeting of the American College of Sports Medicine, San Francisco, California, USA, May 28-31, 2003.

[142] K. S. Quigley and G. G. Berntson. Autonomic interactions and chronotropic control of the heart: Heart period versus heart rate. *Psychophysiology*, 33:605–611, 1996.

[143] R. D. Reed. Pruning algorithms - a survey. *IEEE Transactions on Neural Networks*, 4(5):740–744, 1993.

[144] R. D. Reed and R. J. Marks II. *Neural smithing: Supervised learning in Feedforward Artificial Neural Networks*. Cambridge (Mass.) : MIT Press, 1 edition, 1999.

[145] M. Riedmiller and H. Braun. Speeding-up backpropagation. In R. Eckmiller, editor, *IEEE International Conference on Neural Networks*, pages 586–591, 1993.

[146] T. Ritz, M. Thöns, and B. Dahme. Modulation of respiratory sinus arrhythmia by respiration rate and volume: Stability across posture and volume variations. *Psychophysiology*, 38:858–862, 2001.

[147] R. Rojas. *Neural Networks: A Systematic Introduction*. Springer Berlin, 1996.

[148] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[149] H. Rusko, A. Pulkkinen, S. Saalasti, and J. Kettunen. Pre-prediction of EPOC: A tool for monitoring fatigue accumulation during exercise. 50th Annual Meeting of the American College of Sports Medicine, San Francisco, California, USA, May 28-31, 2003.

[150] S. Saalasti. Time series prediction and analysis with neural networks. Licentiate thesis, University of Jyväskylä, Department of Mathematics and Statistics, 2001.

[151] S. Saalasti, J. Kettunen, and A. Pulkkinen. Patent number FI20025038 (pending), 2002.

[152] S. Saalasti, J. Kettunen, A. Pulkkinen, and H. Rusko. Monitoring respiratory activity in field: applications for exercise training. Science for Success congress, Jyväskylä, Finland, October, 2002.

[153] R. Salomon. *Verbesserung konnektionistischer Lernverfahren die nach der Gradientenmethode arbeiten*. PhD thesis, Technical University of Berlin, 1992.

[154] L. E. Scales. *Introduction to Non-Linear Optimization*. New York: Springer-Verlag, 1985.

[155] R. E Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

[156] R. E Schapire. Using output codes to boost multiclass learning problems. *Machine Learning: Proceedings of the Fourteenth International Conference, Nashville, TN*, 1997.

[157] R. E Schapire, Y. Freund, and P. Bartlett. Boosting the margin: A new explanation for the effectiveness of voting methods. *Machine Learning: Proceedings of the Fourteenth International Conference, Nashville, TN*, 1997.

[158] R. O. Scmidt. Multiple emitter location and signal parameter estimation. In *Proc. RADC, Spectral Estimation Workshop, Rome*, pages 243–258, 1979.

[159] H. R. Shumway and S. D. Stoffer. *Time Series Analysis and Its Applications*. Springer-Verlag, 2000.

[160] F. Silva and L. Almeida. Speeding-up backpropagation. In R. Eckmiller, editor, *Advanced Neural Computers*, pages 151–156. North-Holland, 1990.

[161] S. W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1997.

[162] E. D. Sontag. Feedback stabilization using two-hidden-layer nets. Technical report, Rutgers Center for Systems and Control, 1990.

[163] E. D. Sontag. Feedback stabilization using two-hidden-layer nets. *IEEE Transactions on Neural Networks*, 3(6):981–990, 1992.

[164] R. Stark, A. Schienle, B. Walter, and D. Vaitl. Effects of paced respiration on heart period and heart period variability. *Psychophysiology*, 37:302–309, 2000.

[165] P. Stoica and R. Moses. *Introduction to Spectral Analysis*. Prentice Hall, 1997.

[166] F. B. Stulen and C. J. DeLuca. Frequency parameters of the myoelectric signal as a measure of muscle conduction velocity. *IEEE Trans Biomed Eng*, 28:515–523, 1981.

[167] Y. Suzuki. Self-organizing qrs-wave recognition in ecg using neural networks. *IEEE Transactions on Neural Networks*, 6(6):1469–1477, 1995.

[168] F. Takens. Detecting strange attractors in turbulence. *Dynamical Systems and Turbulence*, 898:336–381, 1981.

[169] B. Tang, M. I. Heywood, and M. Shepherd. Input partitioning to mixture of experts. *IEEE World Congress on Computational Intelligence (IEEE WCCI 2002)*, 2002.

[170] M. Till and S. Rudolph. Optimized time-frequency distributions for signal classification with feed-forward neural networks. *Proceedings SPIE Conference on Applications and Science of Computation al Intelligence III, Orlando, Florida, April 24-28th*, 2000.

[171] A. Vehtari. *Bayesian Model Assessment and Selection Using Expected Utilities*. PhD thesis, Department of Electrical and Communications Engineering, Helsinki University of Technology, 2001.

[172] K. Väinämö, S. Nissilä, T. Mäkikallio, M. Tulppo, and J. Röning. Artificial neural networks for aerobic fitness approximation. International conference on Neural Networks (ICNN '96), Washington DC, USA, June 3-6, 1996.

[173] P. Virtanen. *Neuro-fuzzy expert systems in financial and control engineering*. PhD thesis, Department of Mathematical Information Technology, University of Jyväskylä, 2002.

[174] G. Walker. On periodicity in series of related terms. In *Proceedings of the Royal Society of London, 131*, pages 518–532, 1931.

[175] E. A. Wan. Temporal backpropagation for FIR neural networks. *Proceedings IEEE International Joint Conference on Neural Networks*, 1:575–580, 1990.

[176] Z. Wang and T. Zhu. An efficient learning algorithm for improving generalization performance of radial basis function neural networks. *Neural Networks*, 13:545–553, 2000.

[177] P.D. Wasserman. *Advanced Methods in Neural Computing*. New York: Van Nostrand Reinhold, 1993.

[178] A. S. Weigend and N. A. Gershenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past*. Addison-Wesley Publishing Company, USA 1994.

[179] A. S. Weigend, B. A. Huberman, and D. E. Rumelhart. Predicting the future: a connectionist approach. *International Journal of Neural Systems*, 1(3):193–209, 1990.

[180] R. J. Williams and J. Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2:490–501, 1990.

[181] A. S. Willsky and H. L. Jones. A generalized likelihood ratio approach to detection and estimation of jumps in linear system. *IEEE Trans. Automatic Control*, AC-21(1):108–112, 1976.

[182] N. Wirth. *Algorithms + data structures = programs*. Englewood Cliffs (N.J.) : Prentice-Hall, 1 edition, 1976.

[183] D. H Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

[184] D. H Wolpert. Optimal linear combinations of neural networks. *Neural Networks*, 10:599–614, 1997.

[185] T. H. Wonnacott and R. J. Wonnacott. *Introductory statistics*. John Wiley & Sons, 4 edition, 1990.

[186] G. U. Yule. On a method of investigating periodicities in disturbed series, with special reference to wolfer's sunspot numbers. In *Phil. Trans. Royal Society of London, 226*, pages 267–298, 1927.

YHTEENVETO (Finnish summary)

Tämä väitöskirjatutkimus luo katsauksen moderneihin matemaattisiin menetelmiin fysiologisen aikasarjamallinnuksen viitekehyksessä. menetelmät luovat kokonaisuuden, jota voidaan käyttää hyväksi analysoitaessa fysiologista aineistoa. Erityisesti keskitytään sykkeen mallintamiseen käyttäen hermoverkkomallinnusta.

Fysiologinen aineisto voitaisiin määritellä ihmisen sisäistä fysiologiaa kuvaaviksi objektiivisesti mitattaviksi muuttujiksi. Täten esimerkiksi ihmisen ikä ei ole fysiologinen muuttuja, vaan sitä nimitetään taustamuuttujaksi. Psykologiset muuttujat ovat subjektiivisia ja niitä ei voida todentaa mittaamalla. Psykologinen aineisto kerätään haastatteluilla ja kyselylomakkeilla. Niiden luotettavuuteen vaikuttavat monet häiriötekijät, kuten sosiaalisesti hyväksyttävät vastaukset, huolimattomuus tai haastattelijan virhearvioinnit.

Psykologisen ja fysiologisen aineiston yhtymäkohdat ovat tutkimuksissa, joissa halutaan fysiologian perusteella määritellä ihmisen psykologista tilaa. Tämäntyyppisessä tutkimuksessa psykologinen aineisto muutetaan numeerisen muotoon siten, että sitä voidaan tilastollisesti käsitellä. Fysiologisista muuttujista pyritään löytämään piirteet, jotka pystyvät selittämään psykologisia muuttujia. Tutkimuksen sovellutuksia on esimerkiksi ihmisen resurssien seuraaminen. Yhteiskunnallisesti voitaisiin säästää huomattavia rahamääriä, jos ihmisen työuupumus voitaisiin havaita ajoissa suorittamalla päivittäinen yksinkertainen tehtävä. Tehtävässä voitaisiin mitata esimerkiksi suoritustasoa, sykettä, havainto- tai reaktionopeutta.

Fysiologisessa aineistossa on omat erityispiirteensä, jotka tulee ottaa huomioon käytettäessä ja kehitettäessä matemaattisia menetelmiä. Aineistoa on usein mahdollista tallentaa suuria määriä, esimerkiksi sykeväliaikasarjaa voidaan tallentaa kuluttajamarkkinoille suunnatuilla sykemittareilla n. 30000 lyöntiä ennen mittalaitteen muistin täyttymistä. Fysiologisen aineiston erityispiirteissä on myös yhtymäkohtia mm. biologiseen ja finanssiaineistoon, jolloin esitettäviä ratkaisumenetelmiä voidaan mahdollisesti hyödyntää myös yleisemmin.

Aineiston visualisointi tuottaa asiantuntijalle mallinnuksen pohjana käytettävää tietoa. Väitöstyössä demonstroidaan eri tapoja havainnollistaa aineistoa ja fysiologisia ilmiöitä, sekä fysiologisten ilmiöiden kompleksisuutta.

Tutkimuksessa esitetään menetelmälaajennuksia käsiteltäessä mitattavia muuttujia suoraan mittalaitteelta. Tämä näkökohta palvelee mittalaiteteollisuutta ja etenkin kuluttajamarkkinoille suunnattuja sulautettuja tuotteita, joiden koko ei mahdollista tehokkaita suorittimia tai suuria muistikapasiteetteja. Edelliset ovat tietysti myös kustannuskysymyksiä. Ohjelmistotasolla tehdyt ratkaisut ovat kertakustanteisia, joita voidaan monistaa tuotteisiin.

Fysiologinen aikasarja-analyysi

Fysiologiset aikasarjat ovat usein kaoottisia, epälineaarisia ja epästationaarisia, johtaen signaalin huonoon ennustettavuuteen ja fysiologisen tulkinnan vaikeuteen. Etenkin vapaat kontrolloimattomat mittaukset ovat haastavia mittauslaitteistojen herkistyessä erilaisille ulkopuolisille häiriöille tuottaen signaaliin virheitä.

Fysiologian ennustettavuus riippuu tarkasteltavasta kohteesta ja mittausvälistä. Mitattaessa sykettä ei keskisykettä mallintamalla voida ennustaa tulevia arvoja. Kuitenkin tiedetään henkilön hapenkulutuksen jäävän korkeammalle tasolle kovan fyysisen suorituksen jälkeen. Myös lihaksen puristusvoiman voidaan tilastollisesti ennustaa aidosti laskevaksi vanhuusiällä mittausvälin ollessa esimerkiksi kymmenen vuotta.

Fysiologinen aineisto sisältää lainalaisuuksia, joita tulisi pystyä hyväksikäyttämään luotaessa matemaattisia malleja. Muuttujat ovat järkeviä vain tietyissä fysiologisissa rajoissa. Esimerkiksi syke ei voi olla alle kahtakymmentä tai yli kolmeasataa lyöntiä minuutissa. Aineisto on ajassa etenevää ja mitattavat muuttujat korreloivat keskenään. Matemaattisen mallin tulee käyttäytyä myös aineiston ulkopuolella järkevästi. Esimerkiksi seurattaessa saman henkilön vitaalikapasiteettia voidaan sen arvioida tietyn iän jälkeen jatkuvan vähenevänä. Malleja voidaan yleistää skaalaamalla mitattavia muuttujia taustamuuttujiinsa. Taustamuuttujia ovat mm. ihmisen ikä, paino, pituus, fysiologiset minimit ja maksimit (minimaalinen tai maksimaalinen hapenkulutus, -syke, -ventilaatio, tai -lihaksen puristusvoima). Väitöksessä esitetään mm. sovellus happivelan (EPOC) mallintamiseen, jossa käytetään hyväksi fysiologisia rajoitteita, sekä maksimaalista sykettä arvioitaessa suhteellista hapenkulutusta. Fysiologisen aineiston rajoitettua skaalaa voidaan käyttää hyväksi myös esim. aineiston skaalaamisessa (tai normalisoinnissa) mittauslaitteistoissa tietämättä etukäteen todellista aineiston jakaumaa.

Asiantuntijatiedon siirtäminen matemaattiseen malliin voidaan toteuttaa useilla eri tavoilla. Sumeat asiantuntijajärjestelmät pohjautuvat asiantuntijalauseista koottaviin totuuslauseisiin, joilla pystytään koostamaan asiantuntijoiden monimutkaista päättelyä. Sumeaa logiikkaa voidaan käyttää myös yleisemmin sumeuttamaan taustamuuttujia, kun jatkuva esitys tahdotaan tiivistää. Esimerkiksi syötettäessä neuroverkolle tietoa ihmisen painosta, saattaa järjestelmän kannalta olla tarkoituksenmukaisempaa sumeuttaa muuttuja, siten että se antaa totuusarvon henkilön ylipainosta nollan ja ykkösen välille.

Puhtaassa aikasarjamallinnuksessa käytettävien menetelmien kirjo on laaja. Eri menetelmät tarjoavat ominaisuuksia, joita voidaan käyttää hyväksi mallinnettaessa fysiologista aikasarjaa. Klassisten lineaaristen ja epälineaaristen mallien vahvuus on niiden teoriakehyksen laaja ymmärrys, sekä mallien tarkkailtavuus ja ominaisuuksien parempi hallinta. Esimerkiksi ekstrapoloitaessa aineiston ulkopuolelle lineaarisen mallin käytös on ennustettavissa. Neuroverkot pystyvät esit-

tämään hyvin monimutkaisia pintoja, mutta ne ovat ns. mustia laatikoita joiden käyttäytymistä ei aina voida täysin ennakoida tai hallita.

Klassisia ja moderneja menetelmiä voidaan käyttää rinnakkain luomalla hybridejä, jotka yhdistetään käyttämällä asiantuntijatietoa tai muodostamalla ns. päätösfunktio.

Väitöksessä käsitellään erilaisia aineiston esikäsittelyrutiineja, kuten segmentointia, aineiston järjestäminen (eng. ”data ranking”), normalisointi/skaalaus, digitaalinen suodatus, suora aikataajuusmatriisin painotus, sekä lineaarisen tai epälineaarisen trendin poistaminen. Esikäsittelymenetelmien vaikutus mallinnuksen laadun paranemiseen on usein kriittistä. On kuitenkin tärkeää ymmärtää esikäsittelyrutiinien periaatteet, sillä ne saattavat yli-yksinkertaistaa aineistoa ja poistaa todellisia esim. epälineaarisia ilmiöitä käytettäessä lineaarisia menetelmiä.

Aikasarjasta voidaan irrottaa piirteitä automatisoidusti segmentoimalla aikasarja homogeenisiin väleihin. Segmentointiin voidaan käyttää myös useita muuttujia. Tutkimuksessa esitetään laajennettu versio ns. GLR-algoritmista aikasarja segmentointiin. Mitattavista muuttujista laskettavia piirteitä voidaan käyttää selittämään fysiologista tai psykologista tilaa.

Käytettävien mallien ennustavuuden, yleistyvyyden, teoreettisen hallinnan ja tarkkuuden lisäksi oleellista on pystyä laskemaan luottamus saadulle tulokselle. Luottamusta voidaan käyttää temporaalisessa aineistossa tulosten korjaamiseen tai aineiston karsimiseen. Tilantunnistuksessa luottamus kertoo mallin kyvystä tunnistaa ilmiö. Mallin tuottamien arvojen luottamus tuo myös epäsuoraa tietoa syötemuuttujien laadusta. Tietoa voidaan käyttää syötesignaalin virheiden tunnistamiseen. Luokitus-algoritmit, kuten Kohosen itseorganisoituva kartta, pystyy tuottamaan luottamustietoa laskemalla sisääntulovektorin etäisyyden prototyypeistä. Hybridi-järjestelmistä luottamusta voidaan mitata mm. laskemalla asiantuntijafunktioiden ja lopullisen tuloksen välinen varianssi. Aikataajuusjakaumissa ajallinen taajuushavainto tarkoittaa sitä, että kyseisen taajuuskomponentin tulisi jatkua komponentin määräämän ajan aikatasossa. Luottamusmuuttujalla tulisi myöskin olla tiettyjä ominaisuuksia. Väitöksessä pohditaankin luottamuskertoimien ongelmakenttää fysiologisessa viitekehyksessä esitetyille malleille.

Esikäsittelyn lisäksi fysiologisen mallin tuottamaa ulostuloa voidaan jälkikäsitellä mm. liukuvalla keskiarvostamisella tai käyttämällä hyväksi mallin luottamusestimaatteja. Tämänkaltaisen jälkikäsittelyn perusoletuksena on, että aikasarja on lokaalisti riippuvaista, ts. vierekkäiset havainnot eivät poikkea suuresti toisistaan. Esimerkiksi sykkeen kiihtyvyys ja palautuminen on rajoittunutta, ja jälkikorjaus menetelmät osoittautuvatkin toimiviksi fysiologisessa viitekehyksessä.

Erilaiset aikataajuusjakaumat, sekä aika-skaalajakaumat luovat pohjan epästationaaristen aikasarjojen taajuusinformaation laskennalle. Väitöksessä esitetään uusi geometrinen menetelmä, jolla saavutetaan täydellinen aikataajuusresoluutio. menetelmää sovelletaan hengitysvenymäpannan tuottaman signaalin

käsittelyyn hengitystaajuuden laskemiseksi.

Neuroverkkoarkkitehtuurit ja optimointi

Neuroverkolla voidaan arvioida mitä tahansa yhdenmukaisesti jatkuvaa funktiota mielivaltaisen tarkasti lisäämällä riittävä määrä verkon parametreja. Neuroverkot ovatkin joustavia yleismalleja, joita on käytetty useissa tosielämän sovellutuksissa.

Neuroverkkojen käytön pääongelmat ovat oikean arkkitehtuurin ja optimointimenetelmän valinta. Erilaiset neuroverkko arkkitehtuurit voidaan jakaa ajallisesti staattisiin ja aikadynaamisiin järjestelmiin. Staattisissa verkoissa on lokaalisti ja globaalisti toimivia neuroneja, joista jälkimmäistä edustaa esim. FFNN. Edellistä edustaa mm. radiaalifunktio-neuroverkko.

Oleellinen osa neuroverkko-julkaisuista edelleen käsittelee erilaisia parannuksia ns. ”backpropagation”-algoritmiin ja tapaan, millä neuroverkon tuntemattomat parametrit voidaan optimoida. Tutkimuksessa kuvataankin ”backpropagation”-algoritmin soveltamista FFNN- ja FIR-neuroverkoille matriisimuodossa, sekä esitellään erilaisia vaihtoehtoisia menetelmiä neuroverkko-parametrien ratkaisemiseksi. Oleellista on se, että neuroverkko tulisi ajatella vain yhtenä epälineaarisena järjestelmänä, jonka ratkaisumekanismit löytyvät epälineaarisen optimoinnin teorioista. neuroverkkoteorian rinnalle syntynyt optimointiteoria ja erilaiset optimointi menetelmät eivät ole tarkoituksenmukaisia. Sen sijaan analyyttisten derivaattojen sujuva ja yleinen ratkaisumalli on tärkeää, koska klassiset epälineaarisen optimoinnin menetelmät tehokkaimmillaan käyttävät tavalla tai toisella derivaattoja tai toisen asteen derivaattoja hyväkseen.

Fysiologisissa sovellutuksissa esitelläänkin optimointi yleisen optimointiratkaisijan avulla. Pääajatuksia optimointiin ovat useiden lokaalien minimien läpikäyminen, eri piiloneuronien määrien kokeileminen oikea verkko kompleksisuuden löytämiseksi, sekä fysiologisten rajoitteiden käyttö mallin valinnasta. Lisäksi esitellään puuttuvien havaintojen korvaaminen, sekä näiden painotettu optimointi.

Signaalin häiriöt ja jakauma vaikuttavat neuroverkon opetukseen siten, että malli harhautuu kohti aineistoa, joka on virhemielessä tärkein. Tämä voi johtaa mallin huonoon yleistettävyyteen, ts. huonoon käyttäytymiseen uuden aineiston kanssa. Ongelmaa voidaan vähentää valitsemalla neuroverkolle tasaisesti jakautunut opetusaineisto. Testiaineistoa voidaan käyttää parhaan lokaalin minimin valitsemisessa.

Erilaiset klassiset hybridi-mallit perustuvat oletukseen, että malli voidaan optimoida modulaariseksi optimoimalla yhtäaikaisesti eri asiantuntijafunktiot ja integraatiofunktio. Rinnakkainen optimointi johtaa kuitenkin hyvin epätodennäköisesti modulaariseen malliin. Sen sijaan väitöksessä esitetyssä diskreetissä päätöspinta hybridi-mallissa asiantuntijafunktiot ja integraatiofunktio optimoidaan erikseen. Lisäksi mallille esitetään eri työkaluja modulaarisuuden ja

yleistettävyyden mittaamiseen ja havainnollistamiseen.

Väitöskirjassa esitetään kaksi laajempaa fysiologisen mallinnuksen esimerkkiä. Hengitystaajuuden tunnistaminen sykeaikasarjasta tuottaa informaatiota, jota voidaan käyttää hyväksi mm. hapenkulutuksen arvioinnissa. Happivelkaa mallinnetaan palautuvalla verkolla, jossa osoitetaan dynaamisen neuroverkon mahdollisuudet ennennäkemättömän aineiston ekstrapoloinnissa, sekä staattisten verkkojen epäonnistuminen tehtävässä. Happivelan ja hengitystaajuuden arviointi perustuu täysin sykkeestä laskettaviin muuttujiin.

Tutkimuksessa esitettään uusi yleinen neuroverkko-arkkitehtuuri, jota nimitetään transistori-verkoksi. Transistori-verkossa järjestelmän lopullinen ulostulo saadaan integroimalla neuroverkon antamat ulostulot yhdeksi. Tätä sovelletaan dynaamisen suodattimen rakentamiseen ja hengitystaajuuden estimointiin. Transistori-verkossa neuroverkko on sisäfunktiona ja se prosessoi useita sisääntuloja muodostaakseen yhden. Kuvatulla menetelmällä pystytään tuottamaan paras hengitys-estimaatti, sekä prosessoimaan suuri määrä aineistoa pienemmällä määrällä parametreja verrattuna klassisiin neuroverkkomenetelmiin.

JYVÄSKYLÄ STUDIES IN COMPUTING

1 ROPPONEN, JANNE, Software risk management - foundations, principles and empirical findings. 273 p. Yhteenveto 1 p. 1999.
2 KUZMIN, DMITRI, Numerical simulation of reactive bubbly flows. 110 p. Yhteenveto 1 p. 1999.
3 KARSTEN, HELENA, Weaving tapestry: collaborative information technology and organisational change. 266 p. Yhteenveto 3 p. 2000.
4 KOSKINEN, JUSSI, Automated transient hypertext support for software maintenance. 98 p. (250 p.) Yhteenveto 1 p. 2000.
5 RISTANIEMI, TAPANI, Synchronization and blind signal processing in CDMA systems. - Synkronointi ja sokea signaalinkäsittely CDMA järjestelmässä. 112 p. Yhteenveto 1 p. 2000.
6 LAITINEN, MIKA, Mathematical modelling of conductive-radiative heat transfer. 20 p. (108 p.) Yhteenveto 1 p. 2000.
7 KOSKINEN, MINNA, Process metamodelling. Conceptual foundations and application. 213 p. Yhteenveto 1 p. 2000.
8 SMOLIANSKI, ANTON, Numerical modeling of two-fluid interfacial flows. 109 p. Yhteenveto 1 p. 2001.
9 NAHAR, NAZMUN, Information technology supported technology transfer process. A multi-site case study of high-tech enterprises. 377 p. Yhteenveto 3 p. 2001.
10 FOMIN, VLADISLAV V., The process of standard making. The case of cellular mobile telephony. - Standardin kehittämisen prosessi. Tapaustutkimus solukkoverkkoon perustuvasta matkapuhelintekniikasta. 107 p. (208 p.) Yhteenveto 1 p. 2001.
11 PÄIVÄRINTA, TERO, A genre-based approach to developing electronic document management in the organization. 190 p. Yhteenveto 1 p. 2001.
12 HÄKKINEN, ERKKI, Design, implementation and evaluation of neural data analysis environment. 229 p. Yhteenveto 1 p. 2001.
13 HIRVONEN, KULLERVO, Towards Better Employment Using Adaptive Control of Labour Costs of an Enterprise. 118 p. Yhteenveto 4 p. 2001.
14 MAJAVA, KIRSI, Optimization-based techniques for image restoration. 27 p. (142 p.) Yhteenveto 1 p. 2001.
15 SAARINEN, KARI, Near infra-red measurement based control system for thermo-mechanical refiners. 84 p. (186 p.) Yhteenveto 1 p. 2001.
16 FORSELL, MARKO, Improving Component Reuse in Software Development. 169 p. Yhteenveto 1 p. 2002.
17 VIRTANEN, PAULI, Neuro-fuzzy expert systems in financial and control engineering. 245 p. Yhteenveto 1 p. 2002.
18 KOVALAINEN, MIKKO, Computer mediated organizational memory for process control. Moving CSCW research from an idea to a product. 57 p. (146 p.) Yhteenveto 4 p. 2002.
19 HÄMÄLÄINEN, TIMO, Broadband network quality of service and pricing. 140 p. Yhteenveto 1 p. 2002.
20 MARTIKAINEN, JANNE, Efficient solvers for discretized elliptic vector-valued problems. 25 p. (109 p.) Yhteenveto 1 p. 2002.
21 MURSU, ANJA, Information systems development in developing countries. Risk management and sustainability analysis in Nigerian software companies. 296 p. Yhteenveto 3 p. 2002.
22 SELEZNYOV, ALEXANDR, An anomaly intrusion detection system based on intelligent user recognition. 186 p. Yhteenveto 3 p. 2002.
23 LENSU, ANSSI, Computationally intelligent methods for qualitative data analysis. 57 p. (180 p.) Yhteenveto 1 p. 2002.
24 RYABOV, VLADIMIR, Handling imperfect temporal relations. 75 p. (145 p.) Yhteenveto 2 p. 2002.
25 TSYMBAL, ALEXEY, Dynamic integration of data mining methods in knowledge discovery systems. 69 p. (170 p.) Yhteenveto 2 p. 2002.
26 AKIMOV, VLADIMIR, Domain Decomposition Methods for the Problems with Boundary Layers. 30 p. (84 p.). Yhteenveto 1 p. 2002.
27 SEYUKOVA-RIVKIND, LUDMILA, Mathematical and Numerical Analysis of Boundary Value Problems for Fluid Flow. 30 p. (126 p.) Yhteenveto 1 p. 2002.
28 HÄMÄLÄINEN, SEPPO, WCDMA Radio Network Performance. 235 p. Yhteenveto 2 p. 2003.
29 PEKKOLA, SAMULI, Multiple media in group work. Emphasising individual users in distributed and real-time CSCW systems. 210 p. Yhteenveto 2 p. 2003.
30 MARKKULA, JOUNI, Geographic personal data, its privacy protection and prospects in a location-based service environment. 109 p. Yhteenveto 2 p. 2003.
31 HONKARANTA, ANNE, From genres to content analysis. Experiences from four case organizations. 90 p. (154 p.) Yhteenveto 1 p. 2003.
32 RAITAMÄKI, JOUNI, An approach to linguistic pattern recognition using fuzzy systems. 165 p. Yhteenveto 1 p. 2003.
33 SAALASTI, SAMI, Neural networks for heart rate time series analysis. 192 p. Yhteenveto 5 p. 2003.