

Diss. ETH No. 17038

A Factor Graph Approach to Model-Based Signal Separation

A dissertation submitted to
ETH Zurich
for the degree of
Doctor of Sciences ETH Zurich

presented by

Volker Maximillian Koch

Dipl.-Ing., Universität Karlsruhe (TH)
MAS ETH MTEC/BWI, ETH Zurich
born on October 3, 1971
citizen of the Federal Republic of Germany

accepted on the recommendation of
Prof. Dr. Hans-Andrea Loeliger, examiner
Prof. Dr. Kevin C. McGill, co-examiner

Hartung-Gorre Verlag, Konstanz, February 2007

Series in Signal and Information Processing

Vol. 18

Editor: Hans-Andrea Loeliger

Bibliographic Information published by Die Deutsche Bibliothek

Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the internet at <http://dnb.ddb.de>.

Copyright © 2007 by Volker Maximilian Koch

First Edition 2007

HARTUNG-GORRE VERLAG KONSTANZ

ISSN 1616-671X
ISBN-10 3-86628-140-4
ISBN-13 978-3-86628-140-0

**Seite Leer /
Blank leaf**

Seite Leer /
Blank leaf

Preface

Even long before I had finished my *Dipl.-Ing.* degree at the Universität Karlsruhe, I had received several offers from industrial research labs, ETH professors, and from professors at other great universities to do my doctoral research with them. Some of the areas they were working in were rather exciting. For example, some projects were in the emerging field of wearable computing, others in a field, which I enjoyed already during my diploma thesis at Stanford University: the wide and ever-expanding domain of functional magnetic resonance imaging. I was even offered positions with almost no teaching duties.

However, I preferred to focus on methodology rather than on one specific application during my doctoral studies. I also wanted to be a teaching assistant since I considered this to be interesting and a great learning experience. Both a serious focus on signal processing methodology and the possibility to teach was offered by Professor Hans-Andrea Loeliger of the Signal and Information Processing Laboratory (ISI) at ETH Zurich.

After I had visited him, I was convinced that there was some serious research going on at the ISI, where Professor James L. Massey, Professor George S. Moschytz, and Professor Fritz Eggimann were just replaced by Professor Amos Lapidoth and Professor Hans-Andrea Loeliger. I also had the feeling that Andi was very excited about his work. In fact, he was so excited, that I did not understand every detail of the many things he showed me and explained during my first visit.

Since everyone seemed also pretty friendly at the ISI, I accepted Andi's offer, moved to Switzerland, joined the ISI, and stayed there for about 5

years. During this time, I was fortunate to do interesting research, teach, supervise, learn a lot in many fields, and develop myself both professionally as well as personally in a quite international environment. Last but not least, I met many interesting people. The next few paragraphs are dedicated to them.

First and foremost I would like to express my deepest gratitude to Andi. In many regards, I consider him as an idealist—in the most positive meaning of the word. As everyone should do, he questions mainstream opinions and tries to convey the importance of critical thinking to other people. He has a passion for engineering, math, quality, and design. And he is a curious, enthusiastic, energetic, excited, and sometimes impatient researcher. On the other hand, I experienced him as exceptionally patient, trusting, and understanding when it comes to people. I very much appreciate the freedom he gave me, which allowed me to do research in Palo Alto, California and at the RIKEN Brain Science Institute in Japan. He also encouraged me to teach for two semesters at a college in Lucerne, which was an exceptional and rewarding experience, as it has been to work with Andi.

I am also very grateful to Kevin McGill. After I had met him at several conferences and we had exchanged some emails, he invited me to visit his lab and Stanford University for three months at the beginning of 2006. Besides many interesting discussions about EMG signal decomposition algorithms, Kevin also made sure that I enjoyed life outside of the workplace. For example, he invited me to his home, introduced me to his family, and together with his wife we went to a crab feast dinner. I also like to remember the occasion where we went to a science fair in San Jose to be judges. Kevin made sure that I had a fantastic time—not only on these occasions—but during my whole stay in the Palo Alto region. Finally, he kindly agreed to be the co-examiner of this doctoral dissertation. This means that he had to fly all the way from sunny California to a less sunny and rather dark and cold winter in Zurich.

Feeling already cold myself while writing this, a warm *Thank you!* goes to all my colleagues at the ISI. In particular I would like to mention my former office neighbors and/or collaborators and/or conspirators Justin Dauwels (who also provided first-class feedback on this dissertation), Markus Hofbauer, Daniel Hösli, and Sascha Korl for a great time inside and outside of the lab.

During about five years at the ISI, I also got acquainted with Renate Ago-

tai, Francesco Amatore, Dieter Arnold, Raphael Berner, Jonas Biveroni, Marion Brändle, Murti Devarakonda, Max Dünki, Matthias Frey, Qun Gao, Kurt Heutschi, Junli Hu, Tobias Koch, Ralf Kretzschmar, Amos Lapidoth, Dani Lippuner, Heinz Mathis, Patrick Merkli, Natalia Miliou, Stefan Moser, Maja Ostojevic, Christoph Reller, Bernadette Röösli, Thomas Schärer, Patrik Strelbel, Stephan Tinguely, Pascal Vontobel, Ligong Wang, Michèle Wigger, as well as many short-term visitors and alumni of the lab. All of them enriched my ETH experience, for which I am thankful.

It was a pleasure to have also met so many other people during my time as a doctoral student, research assistant, and teaching assistant. In particular I would like to mention the active members of VMITET and AVETH, my colleagues in the IEEE Student Branch and several other academic associations, Zoia Lateva and other fellows at the Rehabilitation R&D Center in Palo Alto, Andrzej Cichocki, Tomasz Rutkowski, and many more at RIKEN, and Helmut Krueger, Thomas Läubli, Peter Schenk, and Daniel Zennaro of the former IHA. It is also a pleasure to acknowledge the contributions of the students I supervised in more than 20 projects.

I don't want to forget to remember the people who have motivated and positively influenced my way even many years before I joined the ISI. The list of names would clearly be too long to present here, but it would include my parents and friends as well as certain professors, teachers, co-workers, and supervisors. I especially thank those teachers and professors, who not only wanted to convey as much information as possible during their classes, but rather motivated and encouraged me, so that I continued to be interested in a topic even long after the exam.

Volker Koch, February 2007

Seite Leer /
Blank leaf

Abstract

This doctoral thesis is about a rather new model-based signal processing methodology that is based on factor graphs and message-passing algorithms. Using this, we have developed exemplary signal processing algorithms for various biomedical applications. The main application to evaluate and demonstrate this new methodology was in the field of electromyographic (EMG) signal analysis. EMG signals are electrical signals that are generated during muscle contractions. They can be measured using various kinds of electrodes, e.g., needle or fine-wire electrodes. Their analysis provides valuable information for the diagnosis of neuromuscular disorders, the study of neuromuscular control mechanisms, and the verification of anatomic hypotheses.

EMG signals are essentially made up of superimposed action potential trains from several sources. For a thorough analysis, the measured EMG signals need to be decomposed into their constituent trains. However, this task can become difficult if action potentials from different sources overlap. Many EMG signal decomposition methods have been proposed. Traditional algorithms often use heuristic segmentation and clustering approaches. Especially in the case of difficult superpositions with many overlapping action potentials, these algorithms could often not decompose muscle signals correctly. Many other approaches are computationally not feasible when superpositions of many action potentials are to be resolved.

To be able to decompose EMG signals with difficult superpositions, new signal processing algorithms based on factor graphs were developed. Factor graphs allow the systematic derivation of advanced model-based signal processing algorithms. A factor graph is a graphical model that represents a factorization of a function. Instead of starting with such

a function, we explain how a factor graph for EMG signal decomposition can intuitively be obtained from a block diagram. Here, the block diagram is a simulation model for EMG signals. The factor graph approach allowed us to integrate action potential shape information, firing statistics, and other properties of EMG signals into the same model.

Finally, we show how decomposition is achieved by means of the sum-product algorithm. It performs inference by passing messages along the edges of a factor graph. Since our factor graphs have cycles, we get sub-optimal iterative algorithms that allow to handle complex models, which could not be used just a few years ago. We have developed several algorithms that differ, e.g., in the representation of the messages propagating in factor graphs. Our new algorithms allow the fast resolution of single and multi-channel superpositions consisting of many overlapping action potentials.

We have also used the factor graph language to derive novel message-passing algorithms for several other biomedical applications. One example is the extraction of heart beats from pressure sensors that are located under bedposts (seismosomnography). The advantage of this method is that the signals can be recorded without attaching electrodes to a human subject while sleeping. Other examples include the important topic of multi-channel neural spike sorting and blind-source separation for electroencephalographic signals. We present exemplary factor graphs for all these applications.

Keywords: graphical models, factor graphs, signal modeling, model-based signal processing, sum-product algorithm, message passing, superpositions, signal decomposition, resolving superpositions, electromyography, seismosomnography, multi-channel neural spike sorting, blind source separation.

Kurzfassung

In dieser Doktorarbeit geht es um Faktorgraphen und Message-Passing-Algorithmen. Dies sind Bestandteile einer relativ neuen modellbasierte Signalverarbeitungsmethodik, unter deren Verwendung exemplarische Signalverarbeitungsalgorithmen für verschiedene biomedizinische Anwendungen entwickelt wurden. Die Hauptanwendung war im Bereich der elektromyographischen (EMG) Signalanalyse. EMG-Signale sind elektrische Signale, die während Muskelkontraktionen erzeugt werden. Sie können mit verschiedenartigen Elektroden gemessen werden, zum Beispiel mit Nadel- oder feinen Drahtelektroden. Ihre Analyse liefert wertvolle Informationen zur Diagnose neuromuskulärer Krankheiten, zum Studium neuromuskulärer Regelmechanismen und zur Überprüfung anatomischer Hypothesen.

EMG-Signale bestehen im Wesentlichen aus überlagerten Einzelsignalen, welche von verschiedenen Quellen erzeugt werden. Jedes dieser Einzelsignale enthält Aktionspotentiale. Zur genauen Analyse werden EMG-Signale häufig in ihre Einzelsignale zerlegt. Dieses kann schwierig sein, wenn sich Aktionspotentiale verschiedener Quellen überlappen. Traditionelle EMG-Signalzerlegungsalgorithmen verwenden häufig heuristische Segmentations- und Clustering-Ansätze. Insbesondere im Falle von schwierigen Überlagerungen mit vielen überlappenden Aktionspotentialen können diese Algorithmen oft keine zufriedenstellenden Zerlegungsresultate liefern. Viele Ansätze sind außerdem zu rechenaufwändig für den Fall vieler sich überlappender Aktionspotentiale.

Mit dem Ziel, EMG-Signale mit schwierigen Superpositionen zerlegen zu können, haben wir neue Signalverarbeitungsalgorithmen basierend auf Faktorgraphen und Message-Passing-Algorithmen entwickelt. Faktorgraphen erlauben eine systematische Herleitung von modellbasierten

Signalverarbeitungsalgorithmen. Ein Faktorgraph ist ein grafisches Modell, das eine Faktorisierung einer Funktion repräsentiert. Anstelle mit so einer Funktion zu beginnen erklären wir, wie ein Faktorgraph zur EMG-Signalzerlegung intuitiv aus einem Blockdiagramm hergeleitet werden kann. Das Blockdiagramm stellt in diesem Falle ein Simulationsmodell für EMG-Signale dar. Dieser Ansatz erlaubt es sowohl die Form der Aktionspotentiale, als auch Feuerungsstatistiken sowie weitere Eigenschaften von EMG-Signalen in ein und dasselbe Modell zu integrieren.

Schliesslich zeigen wir, wie die Zerlegung mit Hilfe des Summe-Produkt-Algorithmus erreicht wird. Dieser Ansatz findet die Lösung des Signalzerlegungsproblems durch das Berechnen von Messages, die Entlang der Kanten im Faktorgraphen verschickt werden. Da unsere Faktorgraphen Kreise enthalten, ergeben sich suboptimale iterative Algorithmen, welche aber erst den Umgang mit den gegebenen komplexen Modellen erlauben. Solche Modelle konnten vor einigen Jahren noch nicht verwendet werden. Wir haben verschiedene Algorithmen entwickelt, welche sich z.B. bezüglich der Repräsentation der Messages unterscheiden, die innerhalb des Faktorgraphen verschickt werden. Unsere neuen Algorithmen erlauben die schnelle ein- und mehrkanalige Zerlegung von Superpositionen, welche aus vielen überlappenden Aktionspotentialen bestehen können.

Wir haben den Faktorgraph-Ansatz ebenfalls verwendet, um neue Message-Passing-Algorithmen für andere Anwendungen aus dem Bereich der biomedizinischen Signalverarbeitung herzuleiten. Ein Beispiel ist die Extraktion des Herzschlags aus Signalen, die mit Hilfe von Drucksensoren unter den Pfosten eines Betts aufgenommen werden (Seismosomnographie). Bei dieser Vorgehensweise können die physiologischen Signale ohne das Anbringen von Elektroden am Körper des Menschen während des Schlafs gemessen werden. Andere behandelte Anwendungen sind die mehrkanalige neuronale Spikeklassifizierung sowie die blinde Signalzerlegung für elektroenzephalographische Signale. In dieser Arbeit stellen wir exemplarische Faktorgraphen für all diese Anwendungen vor.

Stichworte: Graphische Modelle, Faktorgraphen, Signalmodellierung, modellbasierte Signalverarbeitung, Summe-Produkt-Algorithmus, Message-Passing, Überlagerungen, Signalzerlegung, Superpositionszerlegung, Elektromyographie, Seismosomnographie, mehrkanalige neuronale Spike-Klassifizierung, blinde Signalzerlegung.

Contents

Preface	v
Abstract	ix
Kurzfassung	xi
1 Introduction	1
1.1 Motivation and Objective	1
1.2 Main Contributions	3
1.3 Target Group	7
1.4 Outline	8
2 Electromyography	11
2.1 Anatomy and Physiology	12
2.2 Origin of EMG Signals	15
2.3 Measuring EMG Signals	18
2.4 Pre-Processing of EMG Signals	22
2.5 Applications	24
2.6 Quantitative Analysis of EMG Signals	26
2.7 Summary, Conclusions, and Outlook	27
3 EMG Signal Decomposition	29
3.1 Pictorial Outline of Decomposition	30
3.2 Block Diagram Model	31
3.3 Generating Synthetic EMG Signals	33
3.4 Methods for EMG Signal Decomposition	35
3.5 Resolving Superpositions	37
3.6 Previous Work	38
3.7 Problems and Difficulties of EMG Signal Decomposition .	40
3.8 Summary, Conclusions, and Outlook	43

4 Resolving Superpositions Using Factor Graphs	45
4.1 Motivation	46
4.2 Block Diagram Model for EMG Signals	47
4.3 Problem Statement	53
4.4 Introduction to Factor Graphs	54
4.4.1 Example 1: A Simple Factor Graph	55
4.4.2 Factor Graph Definitions and Rules	55
4.4.3 Example 2: Multivariate Probability Distribution .	57
4.4.4 Example 3: Sum Constraint Node	58
4.4.5 Example 4: Equality Constraint Node	59
4.5 Factor Graph Model for EMG Signals	60
4.6 Resolving Superpositions as a MAP Problem	62
4.6.1 Factor Graph and Factorization	64
4.6.2 Message-Passing Algorithm Using the Sum-Product Rule	65
4.7 Introduction to the Sum-Product Algorithm	65
4.8 Sum-Product Algorithm for Resolving Superpositions . .	72
4.9 Symbol MAP Decoding Versus Block MAP Decoding . .	73
4.10 Summary, Conclusions, and Outlook	75
5 Three New Algorithms for Resolving Superpositions	77
5.1 Message Types Used in This Chapter	78
5.1.1 Discrete Messages	79
5.1.2 Scalar-Gaussian Messages	80
5.1.3 Multivariate-Gaussian Messages	83
5.2 Resolving Superpositions Using Discrete Messages Only .	84
5.2.1 Factor Graph	84
5.2.2 Variables and Message Types	84
5.2.3 Node Functions and Message Update Rules . . .	86
5.3 Resolving Superpositions Using Scalar-Gaussian Messages	107
5.3.1 Factor Graph	107
5.3.2 Variables and Message Types	107
5.3.3 Node Functions and Message Update Rules . . .	109
5.4 Resolving Superpositions Using Multivariate-Gaussian Messages	115
5.4.1 Factor Graph	116
5.4.2 Variables and Message Types	116
5.4.3 Node Functions and Message Update Rules . . .	118
5.5 Improved Readout	123
5.5.1 Standard Symbol-Wise MAP Estimate	123
5.5.2 Improved Readout	125

5.6	Results for the Known-Constituent Problem	126
5.6.1	Simulation Set 1: Varying σ_{detect}	126
5.6.2	Simulation Set 2: Varying the Number of Iterations	138
5.7	Computation Times	141
5.8	Improvements	144
5.8.1	Trying Different Noise Levels	144
5.8.2	Exception Throwing	144
5.9	Improved Results for the Known-Constituent Problem .	146
5.9.1	Simulation Set 3: Varying σ_{simul}	146
5.9.2	Simulation Set 4: Varying the Number of Sources .	150
5.10	The Unknown-Constituent Problem	160
5.11	Results for the Unknown-Constituent Problem	163
5.11.1	Simulation Set 5: Varying σ_{detect}	163
5.11.2	Simulation Set 6: Varying σ_{simul}	173
5.12	Summary, Conclusions, and Outlook	175
6	Decomposing EMG Signals Using Factor Graphs	177
6.1	Overview	178
6.2	Segmenting EMG Signals	180
6.2.1	Previous Segmentation Methods	180
6.2.2	Factor-Graph-Based Segmentation	181
6.3	Estimating EMG Signal Noise Power	190
6.4	Estimating MUAP Shapes	192
6.5	Using Source Models	195
6.6	Using Block Processing	197
6.7	Adapting MUAP Shapes	200
6.8	Dealing with Recruitment and Decruitment	201
6.9	Summary, Conclusions, and Outlook	202
7	Signal Processing Applications Beyond EMG	203
7.1	Seismosomnography	204
7.1.1	The Seismosomnography System	204
7.1.2	Heart Beat Detection Using Factor Graphs	207
7.1.3	Results	207
7.2	Neural Spike Sorting	211
7.3	Blind Source Separation	213
7.3.1	Applications	213
7.3.2	Definition	214
7.3.3	BSS Using Factor Graphs	215
7.3.4	Conclusions	216
7.4	Summary, Conclusions, and Outlook	220

8 Summary, Discussion, Conclusions, and Outlook	221
8.1 EMG Signal Decomposition	221
8.2 Applications Beyond EMG	224
8.3 Overall Conclusions and Outlook	224
A Background Material	227
A.1 Probability Theory	227
A.2 Gaussian Distribution	231
A.2.1 Scalar Case	231
A.2.2 Vector Case	233
B Message Update Rules	237
C MATLAB Program EMG-View	239
D Java Program EMG-Belief	243
E Java Package factorgraph	249
Abbreviations	253
Glossary	257
List of Symbols	261
List of Figures	266
Bibliography	275
Index	291
About the Author	299

Chapter 1

Introduction

Reading, after a certain age, diverts the mind too much from its creative pursuits. Any man who reads too much and uses his own brain too little falls into lazy habits of thinking.

Albert Einstein

1.1 Motivation and Objective

The main application that we wanted to consider during this project was in the field of electromyographic (EMG) signal analysis, in particular the decomposition of electromyographic signals. Many EMG signal decomposition algorithms had already been developed. However, decomposing complex EMG signals could still be improved, especially for the case of high levels of muscle force, which may lead to difficult superpositions consisting of many action potentials from different sources.

For example, already in 1992, Richard Gut, an alumni of our lab, had finished his thesis [49, 50] on the optimal decomposition of EMG signals using the Viterbi algorithm. Like many other optimal approaches, this approach could not deal with difficult superpositions consisting of many action potentials.

Another example is the decomposition algorithm developed by Peter Wellig, who finished writing his doctoral dissertation at our lab in the

year 2000 [135]. His work concentrated on the decomposition of long-term EMG signals. For this, he developed a tool called EMG-LODEC, which is also described in a paper by Zennaro et. al. [142]. Zennaro writes: “The complete decomposition is limited to two superimposed MUAPs, since the computation time is prohibitive for more. The algorithm is based on the peel off approach, which forms composite waveforms from all combinations and determines the best match residual form of it.”

So, we wanted to develop an EMG signal decomposition algorithm that can resolve superpositions consisting of many¹ action potential. Usually, when a specific application-oriented problem—like this one—is to be solved, one can proceed in the following way:

- 1) State and analyze the problem.
- 2) Choose a suitable methodology to solve the problem.
- 3) Implement.
- 4) Test and optimize.

However, in our case, the basic methodology was already a constant given the research interests and projects at our lab. So, instead of selecting *some* methodology, we wanted to apply, test, and evaluate iterative message-passing algorithms on factor graphs.

Factor graphs [90] are graphical models and a versatile language to derive advanced model-based signal processing algorithms. In fact, many algorithms in signal processing, digital communications, coding, and artificial intelligence can be described as message-passing algorithms on factor graphs.

Whenever the underlying factor graphs contain cycles, the corresponding message-passing algorithms are sub-optimal iterative algorithms. Therefore, message-passing algorithms on factor graphs with cycles will not always yield the correct results. They may, in fact, fail to converge in this case².

¹Definitely more than 2 MUAPs, preferably up to about 6...8.

²An often used message-passing algorithm is the sum-product algorithm [90], since it has shown to have better convergence properties on graphs with cycles than, e.g., the max-product algorithm. The sum-product algorithm is sometimes also referred to as belief propagation algorithm or probability propagation algorithm. Since the messages sent by the sum-product algorithm are often non-normalized probability distributions, we usually avoid both of these terms.

Although not optimal, message-passing algorithms on factor graphs with cycles yield excellent results in some applications, especially in the decoding of turbo codes [77]. This led to several doctoral dissertations at our lab, e.g., in the fields of iterative decoding [134], analog electronic circuits [107], noise reduction for speech signals [76], analog decoders [43], as well as in communications and machine learning [28]. The present thesis was therefore intended to be one of several others to investigate message-passing algorithms for various applications.

The primary objective of this dissertation was therefore the modeling of EMG signals using factor graphs and the development of completely new EMG signal decomposition algorithms using (sub-optimal) iterative message passing. We hoped that this novel approach, which, as far as we know, has never been used for this class of problems, would allow us to deal with difficult superpositions that could not be resolved before using other practical algorithms.

Apart from this rather application-oriented objective, we hoped to gain experience with signal modeling and algorithm development using factor graphs. The problem of EMG signal decomposition seemed rather suitable for this purpose, since the *basic* problem formulation is clear, simple, and can easily translated into a factor graph. We wanted to use the gained experience in various other applications—and for the improvement of the basic methodology.

1.2 Main Contributions

New Models for EMG Signals

Factor graphs allow the systematic derivation of advanced model-based signal processing algorithms. We developed factor graphs that model EMG signals, which has never been done before. This completely new approach allowed us to integrate action potential shape information, firing statistics, multiple channels, and other properties of EMG signals very naturally into the same model.

New EMG Signal Decomposition Algorithms

Novel algorithms for EMG signal decomposition were developed using the factor graph language. The algorithms are based on iterative message-passing. By changing graph topologies, node functions, message types and representations, update schedules, and other parameters, various completely new or extended EMG signal decomposition algorithms were designed, implemented, validated, and compared. Although iterative message passing leads to sub-optimal algorithms, we have achieved good decomposition performance and low computational complexity, even for the case of high levels of noise and/or difficult superpositions. These algorithms are explained in detail in this thesis.

According to Kevin McGill (personal communication), it may be unrealistic to expect to be able to correctly resolve all superimpositions involving 14 units in real signals, but it is still very important to be able to resolve superimpositions of 6 or 8, which our new algorithms are able to do.

Developed Software

- a) We have developed two main programs with graphical user interfaces (GUIs). These tools are easy to use by other signal processing and/or EMG researchers and maybe even by clinicians. The first one is a MATLAB tool, called **EMG-View**, for loading, saving, importing, exporting, displaying, pre-processing, and modifying EMG signals. It also allows to analyze and to modify decomposition results and to evaluate decomposition algorithm performance. More information on **EMG-View** can be found in Appendix C.
- b) The second main program with a GUI was written in Java and is called **EMG-Belief**. It allows the simulation of synthetic EMG signals as well as the segmentation and decomposition of synthetic and measured EMG signals. In addition, it contains tools for analyzing the developed message-passing algorithms by plotting messages and by providing other information. Finally, it contains a simulation environment to generate and decompose many signals with pre-defined properties and selected algorithms. All this helps to develop and to evaluate decomposition algorithms. More information on **EMG-Belief** can be found in Appendix D.

- c) We have also developed a Java package called `factorgraph`. It facilitates the design and implementation of factor graphs and message-passing algorithms. It provides building blocks that algorithm designers can use without having to derive every detail by themselves first. Its building blocks can simply be combined (modular assembly principle), which allows rapid prototyping. In addition, new building blocks can easily be included. This generic package especially supports the implementation of discrete-time signal processing algorithms, where one often has one factor graph slice per discrete time index. This fact is explicitly supported by the `factorgraph` package. Since it was implemented in Java, it is also an object oriented, modular, open, reusable, and cross-platform toolbox. More information on the Java package `factorgraph` is provided in Appendix E.
- d) In addition, several other utilities were designed and implemented or extended, for example, utilities to generate synthetic EMG signals and to evaluate decomposition algorithm performance. We have also implemented an interface [9] so that EMG-Belief can work with EMG-LODEC. EMG-LODEC is a long-term EMG signal decomposition tool written by Peter Wellig [135].

Applications Beyond Electromyography

During the development process of the previously mentioned algorithms and software, valuable experiences were made that can be used in various other applications. In fact, we have used factor graphs to derive message-passing algorithms for several other biomedical applications. Examples include seismosomnography (heartbeat detection using pressure sensors, which boils down to irregular pulse train and multichannel estimation), spike sorting for neural signals, and blind source separation. More information on these topics is given in Chapter 7.

Student Projects

The author of this thesis supervised and co-supervised more than 20 student projects, i.e., semester theses, diploma theses, master's theses, and other student projects. Supervisors or co-supervisors of some of these projects were: Simon Bovet, Mark Brink, Justin Dauwels, Miriam

Fend, Daniel Hösli, Junli Hu, Sascha Korl, Thomas Läubli, Hans-Andrea Loeliger, Patrick Merkli, and Peter Schenk. Both these projects and the main research profited from the collaboration.

Most projects were in the field of electromyography and EMG signal decomposition:

- Audrey Schaufelberger, “EMG-signal decomposition using factor graphs: noise variance estimation” [121]
- Marc Andre and Christoph Rüegg, “Emglink: Programmierung eines Interfaces zwischen EMG-LODEC und EMG-Signal” [9]
- Lukas Bolliger, “EMG signal decomposition using factor graphs and message-passing algorithms with joint-Gaussian and discrete messages” [15]
- Yves Kunz, “Algorithmen zur Zerlegung überlagerter EMG-Signale” [78]
- Georg Böcherer, “Adaption of a priori models in EMG signal decomposition” [11]
- Johan Peterson, “Estimation of filters for electromyographic (EMG) signal decomposition” [114]
- Markus Wenk, “Segmentation of EMG signals” [136]
- Christian Zimmermann, “Decomposition of an EMG signal using an a-priori model” [144]
- Christian Rijke, “Simulation of EMG signals to test decomposition algorithms” [119]
- Martin Byrød, “Segmentation and decomposition of EMG signals” [20]
- Samuel Bruhin and Benjamin Amsler, “Zerlegung von EMG-Signalen mittels graphischen Modellen und iterativen Algorithmen” [19]
- Rolf Grüninger, “Optimization of an iterative EMG signal decomposition algorithm” [48]
- Samuel Bruhin and Benjamin Amsler, “Zerlegung von EMG-Signalen mit Faktorgraphen” [18]

Some projects were related to seismosomnography:

- Christian Lüthi, “Seismosomnography: a factor graph approach” [97]
- Thomas Hug and Mirco Rossi, “Seismosomnography: heartbeat detection from pressure sensors” [62]
- Pascal Wildbolz, “Seismosomnography: heartbeat detection from pressure sensors” [138]
- Conraddin Merk, “Seismosomnography: a factor graph approach” [106]

The remaining projects were related to various other fields:

- Martin Renold, “Spike sorting of tetrode data and EMG signal decomposition using loopy belief propagation algorithms” [118]
- Simon Schneiter, “Störgeräuschbefreiung für Sprachsignale mittels Faktorgraphen” [124]
- Roland Abt and Marco Diefenbacher, “Whisker-based obstacle avoidance in the AMouse project” [8]
- Nicolas Cedraschi, “Computing information rates of channels with phase noise” [22]
- Franziska B. Pfisterer and Felix Bürgin, “QRS detection for automated arrhythmia analysis” [115]

1.3 Target Group

Since this thesis deals both with new signal processing methods and biomedical applications, readers with a wide range of backgrounds are addressed:

Factor graph experts: Individuals who already have a good background in factor graphs and message-passing algorithms can learn about interesting and important applications. At the same time

they will see how the factor graph approach can be applied in those applications. All this might give them ideas for their own research.

Signal processing experts: This thesis should also be beneficial for signal processing experts without a specific factor graph background. They get an easy to understand and application-oriented introduction to factor graphs and they can also learn about interesting applications.

Non signal processing researchers and users: Users, such as clinicians, might also find this thesis useful since they can learn about a rather new signal processing approach.

Students: Another goal is to give students (maybe even PhD students), who do projects and theses in the field, an “easy” and intuitive introduction to signal modeling using factor graphs and message-passing algorithms. To achieve this goal, many examples were included³.

Reading committee: Last but not least, since this is a doctoral thesis, this report is also intended to give an overview of my work so that it can be evaluated by the doctoral dissertation reading committee.

1.4 Outline

Since this thesis should not only be read by factor graph experts but also by researchers with a focus on electromyography and other applications, and for didactic reasons, this thesis starts with an application (electromyographic signal decomposition) rather than with factor graph theory. The background material on factor graphs and message-passing algorithms is introduced when and as needed. In some sense, this book is also a tutorial on how to derive message-passing algorithms based on factor graphs for signal processing problems.

It is one goal that most people with some knowledge in signal processing should be able to read this thesis and understand the derivations and basic ideas without having to read many additional papers. Therefore,

³The author of this thesis would have found such a collection of easy-to-understand examples quite useful at the beginning of his doctoral research.

certain basic concepts, which might be clear to some readers without explanation, were included in this dissertation.

This thesis is structured as follows:

Chapter 2 gives an introduction to electromyography, the main application considered in this thesis.

Chapter 3 introduces the reader to electromyographic signal decomposition. The focus lies on the problem of resolving superpositions. Previous approaches to solve this problem are sketched.

Chapter 4 introduces and derives a new approach to resolve superpositions. The developed algorithms are based on factor graphs and the sum-product algorithm. The factor graph theory is introduced and explained here as far as it is needed for this application.

Chapter 5 introduces and compares three message-passing algorithms to resolve superpositions based on the fundamentals established in Chapter 4. These algorithms basically differ in the types of messages they use.

Chapter 6 investigates what is necessary to use message-passing algorithms not only to resolve single superpositions, but to decompose whole EMG signals. Treated are various topics such as the segmentation of EMG signals, the estimation of MUAP shapes, the adaptation of MUAP shapes, and noise level estimation.

Chapter 7 introduces graphical models for three other applications beyond electromyography. Namely, these applications are: heart beat detection, neural spike sorting, and blind source separation.

Chapter 8 summarizes, discusses, and concludes this theses. It also gives an outlook.

The appendix contains some basic concepts for readers without a background in signal processing, e.g., some probability theory. In addition to message update rules based on the sum-product algorithm, the appendix also gives an overview of some of the developed software.

Seite Leer /
Blank leaf

Chapter 2

Electromyography

‘Then no wonder I can catch up with you so fast after you’ve had four years of biology.’ They had wasted their time memorizing stuff like that, when it could be looked up in fifteen minutes.

From *Surely You’re Joking, Mr. Feynman!* [41]
by Richard P. Feynman

Muscle fiber contraction goes along with electrical activity. Electromyography is a method for measuring, displaying, and analyzing such electrical signals using various kinds of electrodes. An electromyographic (EMG) signal consists of signal contributions of the muscle fibers that are within a certain distance from the electrode.

Electromyography aids in the diagnosis of neuromuscular disease. It is also used by researchers to study neuromuscular control mechanisms and to verify anatomic hypotheses. Therefore, the analysis of EMG signals yields valuable information for clinicians and researchers.

This chapter gives an introduction to electromyography as far as it is needed for the aim of electromyographic (EMG) signal decomposition. More detailed information can be found in [10, 61, 98, 111]. Further literature is listed in the corresponding sections.

Electromyography

Electromyography deals with the measurement, display, and analysis of electrical signals that are generated when muscles contract.

2.1 Anatomy and Physiology

This section deals with *human* anatomy and physiology that is relevant to this thesis, such as the nervous system, muscle types, muscle anatomy, and muscle contraction.

Nervous System

For a voluntary muscle (see below) to contract, a stimulation from the nervous system is required. Table 2.1 shows the various parts of the human nervous system. The central nervous system consists of the brain and the spinal cord. The spinal cord connects the brain with the body. The peripheral nervous system consists of nerves (groups of axons) carrying impulses to and from the central nervous system, e.g., it carries information from the spinal cord to muscle fibers [5, 145].

Muscles

In humans, muscles can be classified into three categories. Skeletal (or striated) muscles are attached to the skeleton and exert forces on it.

Table 2.1: The human nervous system.

Central		
Peripheral	Somatic	
	Autonomic	Sympathetic Parasympathetic Enteric

They consist of multinucleated fibers (long, slender cells), which are under the voluntary control of the somatic nervous system¹. These contractile fibers convert chemical energy (adenosine triphosphate, ATP) into mechanical energy (motion) [3].

The two other types of muscles are cardiac muscles and smooth muscles. Smooth muscles line the viscera, blood vessels and dermis. Cardiac muscles and smooth muscles are operated by the autonomic nervous system, see Table 2.1. Therefore, they are not under voluntary control [3]. From now on, this thesis assumes skeletal muscles if nothing else is mentioned.

Motor Unit

The motor unit (see Figure 2.1) is the smallest functional unit of the neuromuscular system. It consists of an alpha motor neuron², which is a nerve cell as in Figure 2.2, and all the muscle fibers it innervates. The number of motor units per muscle ranges from about 100 to more than 1700 [112, 123]. The number of muscle fibers of a motor unit also varies. Motor units in muscles of the eye might have 5...7 muscle fibers, a motor unit of the hand has about 100 [135], and large limb muscles can have more than 500 muscle fibers [133].

Motor unit

A motor unit, the smallest functional unit of the neuromuscular system, is the collection of an alpha motor neuron and all the muscle fibers innervated by this neuron.

Muscle Fiber Contraction

Voluntary muscles require action potentials from their nerves to initiate a contraction. For example, the sequence of events can begin in an upper alpha motor neuron, whose cell body is located in the cerebral cortex of the brain. Then, an action potential reaches the lower alpha motor neuron in the spinal cord. From there, an action potential is sent along an axon towards the neuromuscular junction, which is the synapse

¹The somatic nervous system is that part of the peripheral nervous system that is not autonomic, see Table 2.1.

²A motor neuron is sometimes also called a motoneuron.

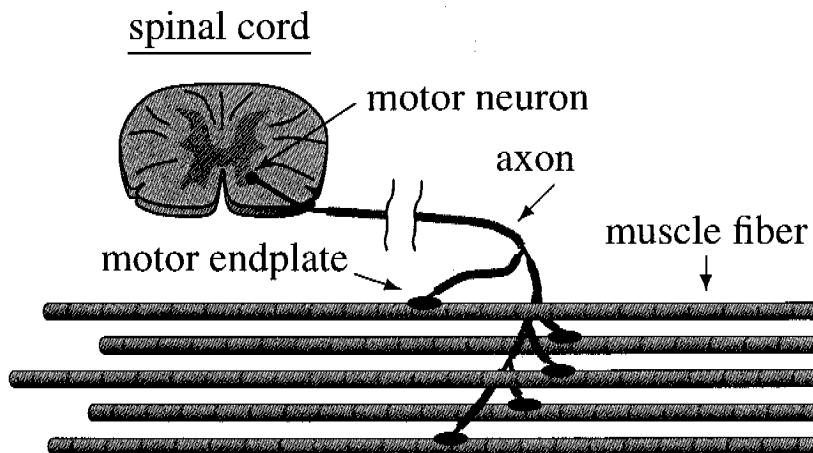


Figure 2.1: A motor unit consisting of a nerve cell (motor neuron), which is located in the spinal cord, and a group of muscle fibers innervated by the axon of the nerve cell. Modified from [56] with permission.

between an axon terminal and a motor end plate of a muscle fiber. When the action potential arrives here, the neurotransmitter acetylcholine is released. Acetylcholine stimulates receptors in the cell membrane of the muscle fiber and the muscle fiber membrane depolarizes. The end result is a contraction of the muscle fiber [4].

Muscle Contraction

Motor units are repeatedly active. To increase the muscle force, active motor units increase their firing rates and idle motor units become active, they are recruited. Smaller motor units, i.e., those with fewer muscle fibers, are usually recruited first. Firing rates depend on various factors, such as level of contraction, muscle size, axonal damage, etc. Typical firing frequencies of motor units are in the range between 5 and 50 Hz. For 25% maximum voluntary contraction (MVC), a motor unit firing frequency of about 15 Hz is not abnormal [12, 26]. For the purpose of EMG signal decomposition, typical values for the firing frequency are between 5 and 20 Hz.

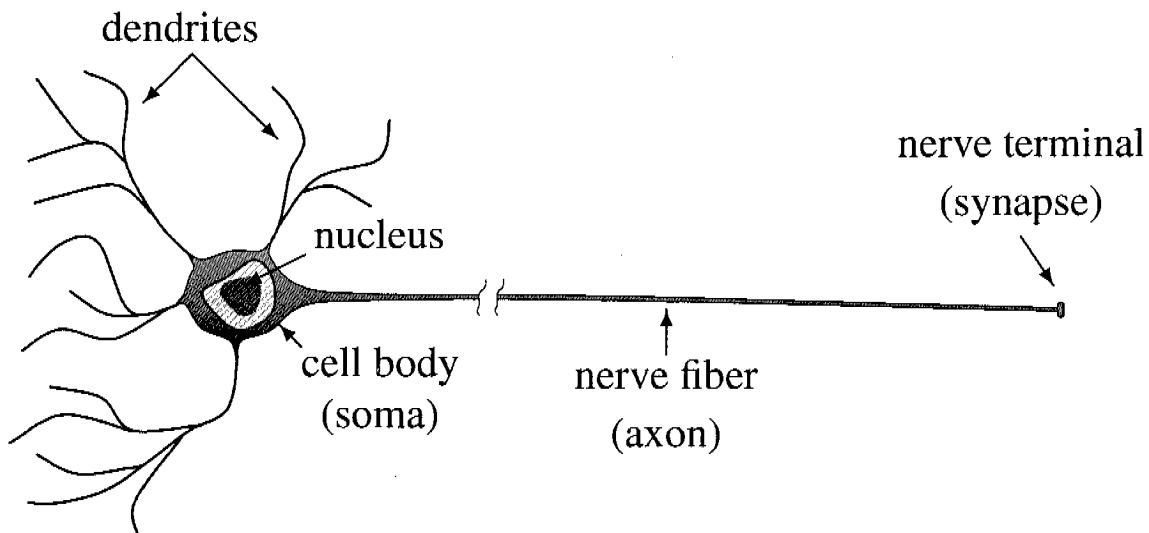


Figure 2.2: A nerve cell. From [56] with permission.

Muscle Disease

Since a disease such as muscle weakness can be caused by a failure in any part of this chain of events, clinicians need to find the exact cause of a disease before they can treat a patient [4]. For example, in motor neurone disease (MND), motor nerves stop working. The muscles that the damaged nerves supply are no longer activated and therefore lose their strengths. EMG signal analysis can help with the diagnosis of disease by helping to determine the exact cause of it. But before we explain more about EMG signal analysis, we treat the origin and recording of EMG signals in the following sections.

2.2 Origin of EMG Signals

An EMG signal, which we can record by using various types of electrodes (see Section 2.3), originates from the membranes of contracting muscle fibers that are close to the recording electrode. In this section, the basic mechanisms of the signal creation process are sketched. More information is available in [39, 112, 133].

Resting Membrane Potential

Remember that a muscle fiber is a long, slender cell. The resting membrane potential, measured from the inside to the outside of a muscle fiber, is about -90 mV [122]. Therefore, the inside of the muscle fiber is more negative than the outside. This is caused and maintained by the differences in concentrations of ions on the inside and outside of the cell and by properties of the cell membrane, namely the selective ion permeability and active transport mechanisms for ions (ion pumps).

Muscle Fiber Action Potential

When an alpha motor neuron becomes active, it generates an action potential that propagates along its axon to all muscle fibers belonging to that motor unit. At the neuromuscular junction, the neuro transmitter acetylcholine is released. This transmitter causes an action potential in the motor end plate region of the muscle fiber. This corresponds to a change in the voltage across the muscle fiber membrane from about -90 mV to about $20\dots 50$ mV.

Starting at the motor end plate region, the action potential propagates along the muscle fiber in both directions towards the tendons. This goes along with a contraction of the muscle fiber. The propagation of the action potential corresponds to a propagation of a local depolarization of the muscle fiber membrane. The resulting signal that could be measured if only one muscle fiber was active a single time is called a muscle fiber action potential (MFAP).

Due to the previously mentioned membrane properties, the original ion concentrations and the resting membrane potential is reestablished after a few milliseconds at each location on the muscle fiber membrane.

Motor Unit Action Potential

Since one activation of an alpha motor neuron causes all innervated muscle fibers to contract, a sum signal, which consists of contributions from all muscle fiber action potentials, is usually measured. This spike of electrical activity, caused by one activation of a motor unit, is called a motor unit action potential (MUAP). The duration of such a MUAP depends

on several factors such as the filtering process, see Section 2.4. Typical MUAP durations are in the range 5 ms...10 ms. For the often used sampling frequency of 20 kHz ($T=50\ \mu s$), this corresponds to 100...200 samples.

MUAP

A motor unit action potential (MUAP) is a waveform that is measured when a motor unit is activated one time.

Consecutive MUAPs from the same motor unit have similar shapes. However, the MUAP shapes depend on many factors, e.g., the type of muscle fibers, the anatomical structure of the muscle (e.g., its size and the number of muscle fibers per motor unit), the relative location of the electrode to the source of the electrical activity (muscle fiber membranes), the properties of the volume conductor between the recording electrode and the source, the propagation velocity, the size, shape, and orientation of the recording electrodes [128], and the EMG signal processing methods used, see Section 2.4. Since some of these factors can change over time, consecutive MUAPs from the same motor unit will not be exactly identical in shape.

EMG Signal

A single motor unit fires repeatedly and therefore generates a signal with a train of action potentials, which is called a MUAP train.

An EMG signal consists of contributions from several motor units and is defined as the sum of all MUAP trains plus noise and artifacts. For example, artifacts can be caused by a moving electrode. Figure 2.3 shows a measured EMG signal with two MUAPs from different motor units, and noise.

EMG signal

An EMG signal contains MUAPs from different motor units, artifacts, and noise. MUAPs from different motor units can overlap.



Figure 2.3: An EMG signal with two MUAPs from different motor units.

2.3 Measuring EMG Signals

Various types of electrodes can be used to measure EMG signals. The main classes are needle electrodes, wire electrodes, and surface electrodes. Here, only short descriptions are given. More information is available in [129, 133]. Figure 2.4 shows an EMG experiment for the investigation of neck and shoulder strain using surface electrodes and fine-wire electrodes.

Needle Electrodes

Needle electrodes can be used for intramuscular recordings. By placing a needle electrode within a muscle, one can record extracellular muscle potentials. Very fine needle electrodes placed close to single muscle fibers measure muscle fiber action potentials. Needle electrodes with a larger active surface detect superpositions of several muscle fiber actions potentials, i.e., MUAPs. There are different kinds of needle electrodes. Well known and commonly used are concentric needle electrodes. Other examples are monopolar needle electrodes and special multi-channel needle electrodes.

Fine-Wire Electrodes

When using needle electrodes during dynamic contractions, the distance between the electrode tip and the muscle fibers changes during muscle movement. This leads to a poor signal quality and can hurt patients. An alternative to needle electrodes are fine-wire electrodes. Fine wires are inserted into a muscle using a needle, the needle is removed, and the wires stay inside of the muscle with the help of a small barb. Figure 2.5 shows such a needle with a fine wire-electrode. Like needle electrodes, fine-wire electrodes also measure extracellular electrical activity. An advantage of

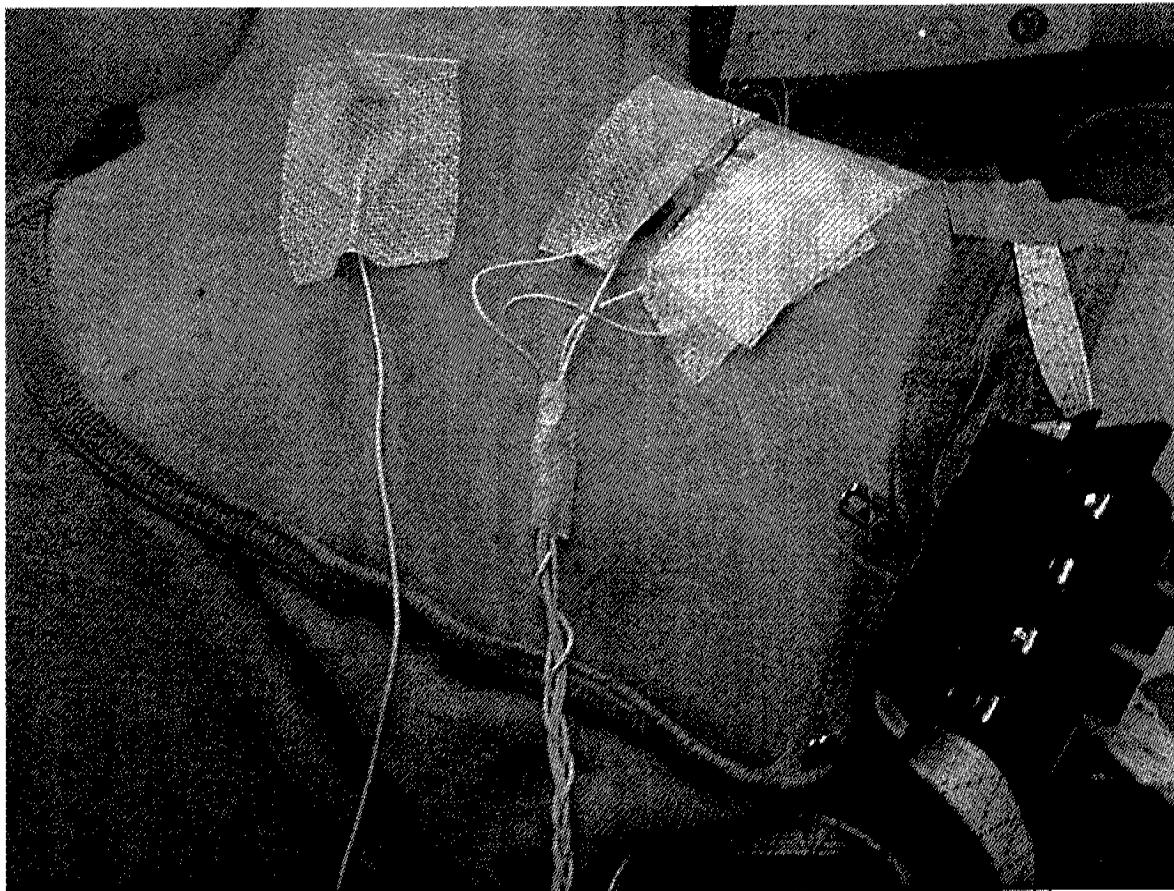


Figure 2.4: EMG experiment for the investigation of neck and shoulder strain using surface electrodes and fine-wire electrodes at the same time. Picture provided by Thomas Läubli and his group of ETH Zurich.

fine-wire electrodes over needle electrodes is that they hurt less when the patient has to contract the muscle during an experiment. In addition, the wire moves with the muscle, which is important for a good signal quality.

Surface Electrodes

Intra-muscular recordings can be painful to the patient, which is especially problematic when experimenting on or diagnosing children. For certain applications surface electrodes can be used. They do not cause pain since they are not inserted into a muscle but rather placed non-invasively on the skin surface. However, due to the low-pass filtering properties of skin and muscle tissue, the applications are limited. In

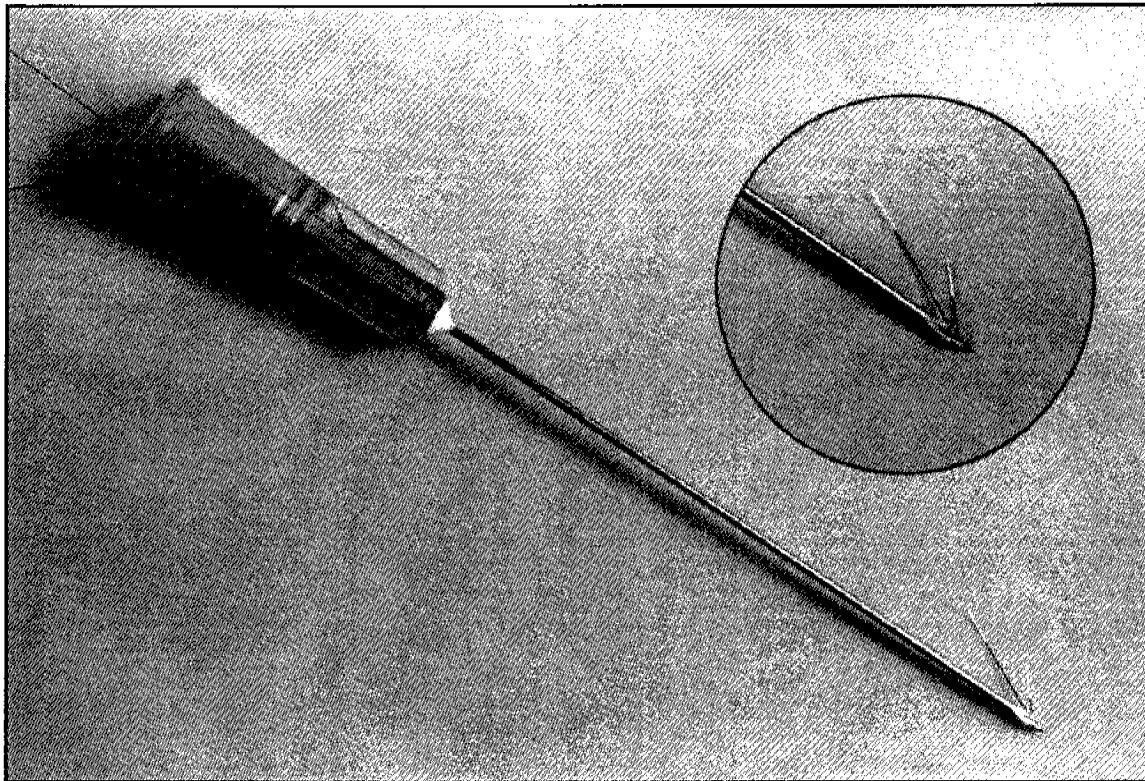


Figure 2.5: Needle with two fine-wire electrodes. Picture provided by Thomas Läubli and his group of ETH Zurich [143].

addition, the amount of information from muscles lying deep below the surface of the skin is also very limited. More details on surface electrodes can be found in [25, 109].

Surface Electrode Arrays

Several surface electrodes making up an array can be used to increase the amount of information from surface recordings. Several groups have obtained relevant clinical data and good results using this approach [36, 37, 100, 108]. More information on surface electrode arrays can be found in [38]. Figure 2.6 shows an EMG experiment for the investigation of back pain using surface electrode arrays.

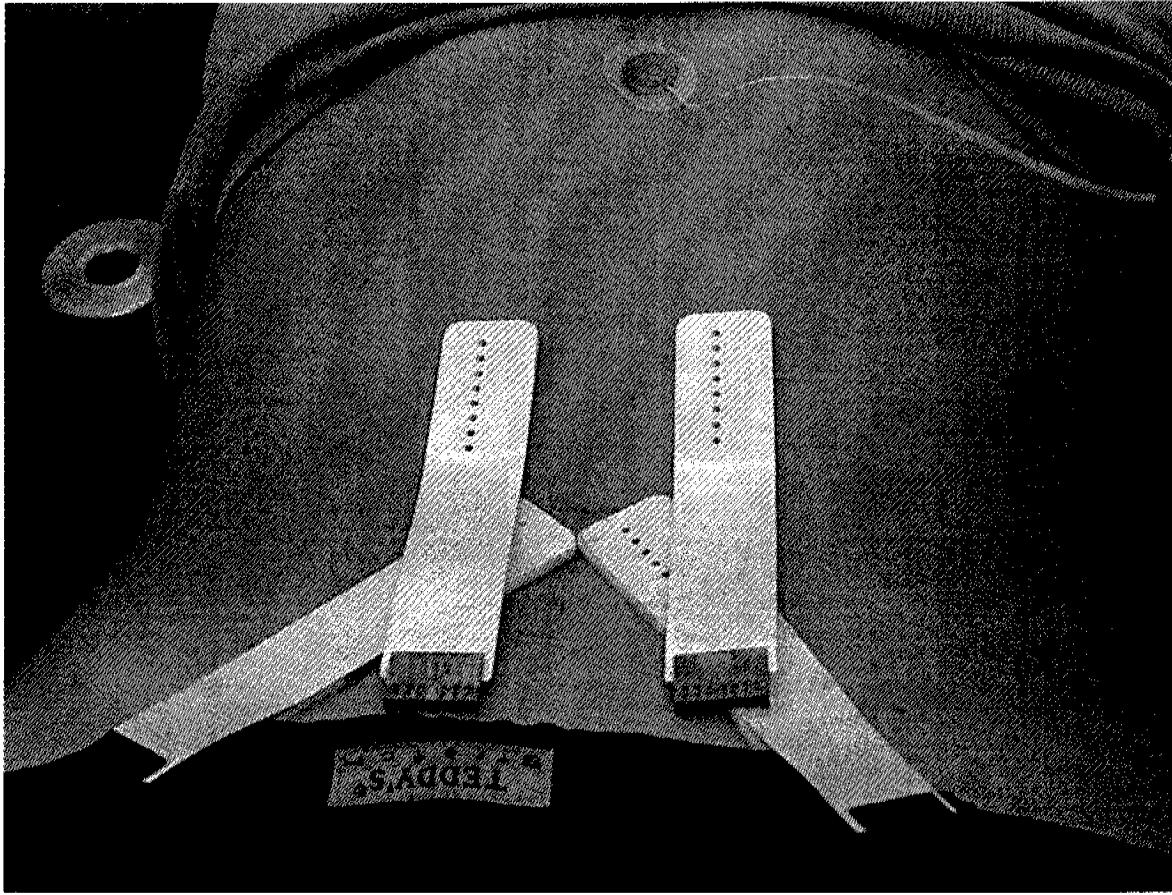


Figure 2.6: EMG experiment using surface electrode arrays. Picture provided by Thomas Läubli and his group of ETH Zurich.

Electrodes for EMG Signal Decomposition

For the purpose of EMG signal decomposition, which is the main topic of this dissertation, signals with uniquely shaped MUAPs for each active and detected motor unit are required. This can be achieved with electrodes that have small detection surfaces since they make the electrodes very selective. However, a small detection surface also leads to large MUAP shape changes when an electrode moves. Sudden MUAP shape changes can be a difficult problem for the purpose of EMG signal decomposition.

In contrast, surface electrodes are not very selective; they reduce the differences between MUAPs from different motor units. They also increase their durations, which makes superpositions more likely and a correct decomposition more difficult [129]. On the other hand, it has been shown that even single motor unit contributions can be extracted from multi-

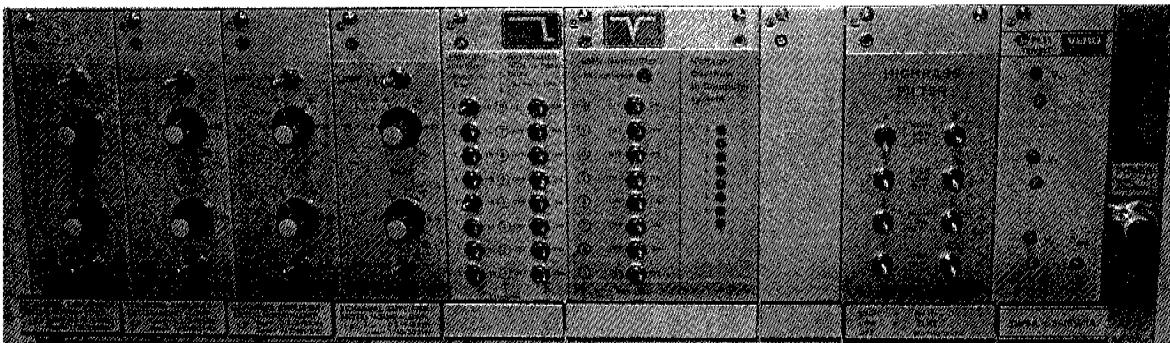


Figure 2.7: Front of an 8-channel EMG signal amplifier built at our laboratory by Thomas Schärer.

channel surface electrode recordings using advanced detection systems and algorithms that were especially developed to extract information from multiple channels. Up to hundreds of channels have been used [38].

Electrodes Used Here

The EMG signals used during this dissertation are intramuscular EMG signals recorded with concentric needle, monopolar needle, or fine-wire electrodes. From now on, signals originating from these kinds of electrodes are assumed in this dissertation if nothing else is mentioned.

2.4 Pre-Processing of EMG Signals

In many modern EMG systems, the measured signals are amplified (see Figure 2.7) in an analog way, filtered with an anti-aliasing filter, converted using an analog-to-digital (AD) converter, and then digitally processed and displayed, often using standard PC hardware with Microsoft Windows Software [133], see Figure 2.8.

AD Conversion

A typical sampling frequency is 10 kHz. However, our lab often uses signals recorded with a sampling frequency of 20 kHz. According to the Nyquist theorem, the upper frequency limit is therefore 10 kHz. Typical

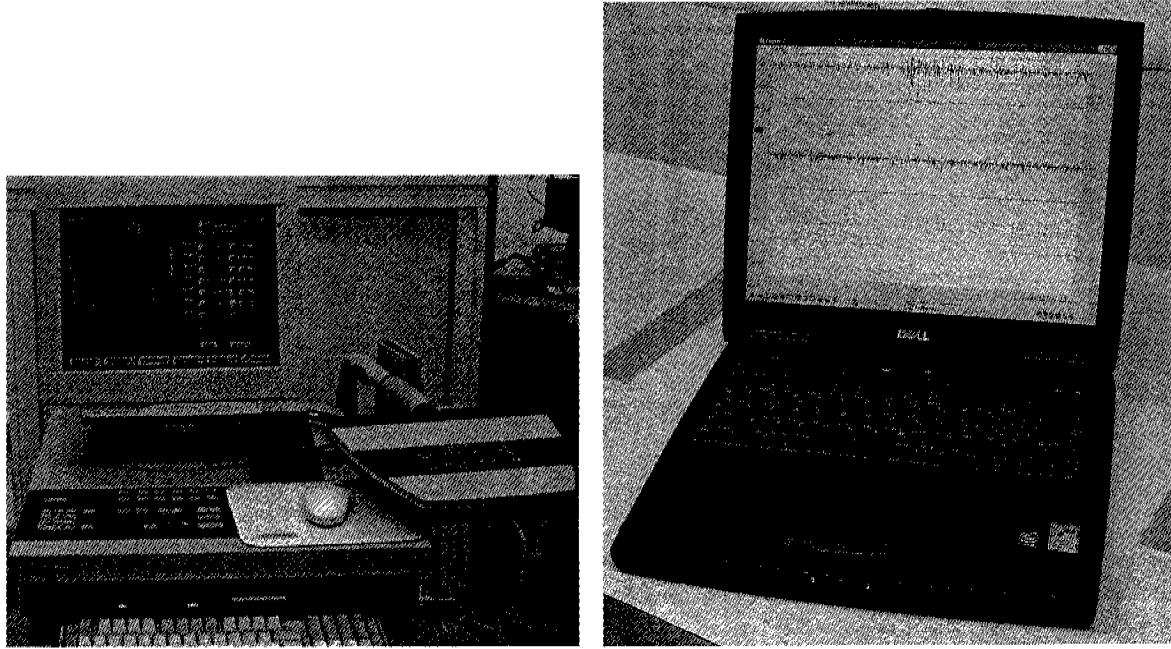


Figure 2.8: Left: Dantec Keypoint system for electromyography; right: standard notebook with Microsoft Windows displaying EMG signals.

modern AD converters allow an amplitude resolution of 12 or more bits, yielding $2^{12} = 4096$ or more amplitude levels.

Filtering

In addition to being low-pass filtered (anti-aliasing filter), EMG signals are often high-pass filtered before further processing occurs, for example at 1 kHz [101]. Such filtering reduces non-discriminatory low-frequency information and shortens the duration of MUAPs, which leads to fewer superpositions. Filtering a signal can also prevent self superpositions of MUAPs from the same source. A frequency band that is often used when band-pass filtering an EMG signal is the band between 1.2 and 2.5 kHz.

Up-Sampling

Kevin McGill has previously up-sampled signals to 80 kHz that were originally recorded with a sampling rate of 10 kHz using FFT-based interpolation. The decomposition performance could be improved in this way [101, 103].

Data Formats

The digitized EMG data often needs to be stored for further analysis and exchanged with other researchers and clinicians. For this, standardized data formats are helpful. A recent web site, called EMGLAB [1], which was set up by Kevin McGill, attempts to help in this regard. It also provides information related to electromyography and promotes “decomposition as a research tool, exchange and discussion of EMG data, attention to accuracy and precision, and algorithm innovation”.

2.5 Applications

The analysis of EMG signals yields valuable information for the use in many applications. In the early decades of the 20th century, electromyography was mainly used to differentiate between different neuromuscular disorders. But electromyography can not only help in the diagnosis of disease, it can also be used for the study of the normal human muscle. Examples include the investigation of the interplay between agonists and antagonists, the role of different muscles in synergistic action, or the study of muscle fatigue [59]. Today, many more applications are known. In this section, a brief overview and some exemplary applications are presented. This can not be a complete list. Refer to [111] for a detailed description of several applications.

Applications in Neurology

Neurologists have various means to diagnose patients. They take a history, do muscle tests, coordination tests, reflex tests, CT and MRI scans, blood tests, EEG measurements, nerve conduction tests, and EMG tests. Surface EMG signal analysis can be used to measure reflex activity and conduction velocities of motor nerves. Recently, high-density multichannel electrode grids have been developed to non-invasively measure motor unit characteristics [145]. Beyond diagnosing diseases, EMG signal analysis can also help evaluating the progression of a disease.

Applications in Ergonomics

According to the International Ergonomics Association, the term *ergonomics* is defined as follows [2]:

“Ergonomics (or human factors) is the scientific discipline concerned with the understanding of interactions among humans and other elements of a system, and the profession that applies theory, principles, data and methods to design in order to optimize human well-being and overall system performance.

Ergonomists contribute to the design and evaluation of tasks, jobs, products, environments and systems in order to make them compatible with the needs, abilities and limitations of people.”

For example, researchers have investigated whether and why prolonged computer work can lead to neck and/or shoulder pain [141]. To investigate this question, intramuscular and surface EMG signals were recorded and analyzed, see Figure 2.4. The aim was to determine whether there are continuously active motor units during normal computer work at low levels of muscle contraction (Cinderella hypothesis). If this was the case, these motor units may become metabolically overloaded resulting in pain. To answer this question, multi-channel long-term muscle signals needed to be decomposed. This was achieved by using a special software package called EMG-LODEC, which was developed by Peter Wellig at our lab [135, 142, 143].

Applications in work physiology and biofeedback are other examples: some patients experience muscle pain due to muscle tension, e.g., back pain. To reduce their pain, feedback based on EMG signals is given to those patients. The EMG signals reflect the level of muscle contraction and the feedback can be auditory and/or visual. This can help patients to learn to control muscle tension, i.e., to relax certain muscles or to improve work techniques [27, 59].

Surface EMG signals can also be used to provide an objective assessment of muscle fatigue; with potential applications in basic research and clinical studies [110].

Neuromuscular Anatomy and Physiology Research

One can study the muscle architecture by estimating the location of the motor endplate and muscle/tendon junctions [80] using EMG signal analysis. Another example is the study of doubly innervated muscle fibers as in [81]. Refer also to [79, 82].

Other Applications

There are many more applications, such as in exercise physiology [40], in movement and gait analysis [44], in rehabilitation medicine [117], and in the control of prostheses [113]. A detailed overview is beyond the scope of this report.

2.6 Quantitative Analysis of EMG Signals

A non-quantitative analysis of EMG signals may be sufficient for some applications. However, for many of the questions arising in the applications sketched above, a quantitative analysis of EMG signals is necessary in order to estimate important parameters. Some examples are presented in the following list:

- Number of active motor units;
- Motor unit firing rates;
- Motor unit recruitment and decruitment;
- Innervation time statistics;
- Synchronization of motor units, i.e., two motor units firing not completely independently;
- MUAP shapes; in particular of interest for diagnostic purposes are the number of zero-crossings and the maximal amplitudes of the MUAPs;
- Muscle fiber conduction velocity;
- MUAP shape changes;

- Power spectrum.

To estimate many of these parameters, such as motor unit firing times, innervation time statistics, motor unit recruitment, MUAP shape changes, etc., a reliable decomposition of EMG signals is necessary, which is the main topic of this dissertation.

2.7 Summary, Conclusions, and Outlook

The EMG signal recorded by a needle or fine-wire electrode is made up of contributions from several motor units. Such EMG signals can give useful information that help to answer questions related to pathology, physiology, and anatomy. To obtain this information, it is often necessary to separate the contributions from the simultaneously active motor units. This process is known as EMG signal decomposition, which is the topic of the following chapters.

Seite Leer /
Blank leaf

Chapter 3

EMG Signal Decomposition

...but I refused to study the problem of intelligence as others have before me. I believe the best way to solve this problem is to use the detailed biology of the brain as a constraint and as a guide, yet think about intelligence as a computational problem—a position somewhere between biology and computer science. Many biologists tend to reject or ignore the idea of thinking of the brain in computational terms, and computer scientists often don't believe they have anything to learn from biology.

From *On Intelligence* [58] by Jeff Hawkins

As stated before, an EMG signal consists of signal contributions of the muscle fibers that are within a certain distance from the measuring electrode. Also as a reminder, an alpha motor neuron together with the muscle fibers it innervates is called a motor unit, which is the smallest functional unit of the neuromuscular system. One activation of a motor unit results in a waveform that is called a motor unit action potential (MUAP). A single motor unit generates a train of such waveforms, a MUAP train. A general EMG signals consists of the sum of several MUAP trains, noise, and artifacts.

An important analysis method for EMG signals is EMG signal decomposition. The aim of EMG signal decomposition is to separate an EMG signal into its MUAP trains. Since EMG signal decomposition separates the activity of multiple simultaneously active motor units, it yields several important parameters, such as the firing times of the motor units. This information is valuable both for clinicians and researchers.

EMG signal decomposition

EMG signal decomposition aims at separating an EMG signal into its constituent MUAP trains.

This chapter gives an introduction to electromyographic signal decomposition. After a pictorial outline of decomposition and a definition using a block diagram model for EMG signals, a selection of existing EMG signal decomposition methods is briefly described and discussed. We will focus on the resolution of superpositions. Finally, some problems and difficulties of EMG signal decomposition are presented.

3.1 Pictorial Outline of Decomposition

Figure 3.1 schematically depicts the origin, detection, and decomposition of an EMG signal. In the lower left, a cross section of the spinal cord is shown. This is where the cell bodies of the alpha motor neurons are located. Each alpha motor neuron axon innervates a bundle of muscle fibers. When an alpha motor neuron fires repeatedly, it causes the motor unit to generate a MUAP train. As explained in Chapter 2, an EMG signal consists of contributions from several motor units, since several alpha motor neurons are usually active.

Figure 3.1 depicts the overlapping single MUAP trains in the upper right plot. This is to emphasize that the measured signal consists of components from all active and detected motor units. When we measure an EMG signal inside of a muscle, we actually do not measure signal contributions from these single motor units individually, but rather the sum of their contributions plus noise and artifacts. Decomposing an EMG signal means separating signal contributions from these different active motor units, i.e., extracting the individual MUAP trains from the raw EMG signal.

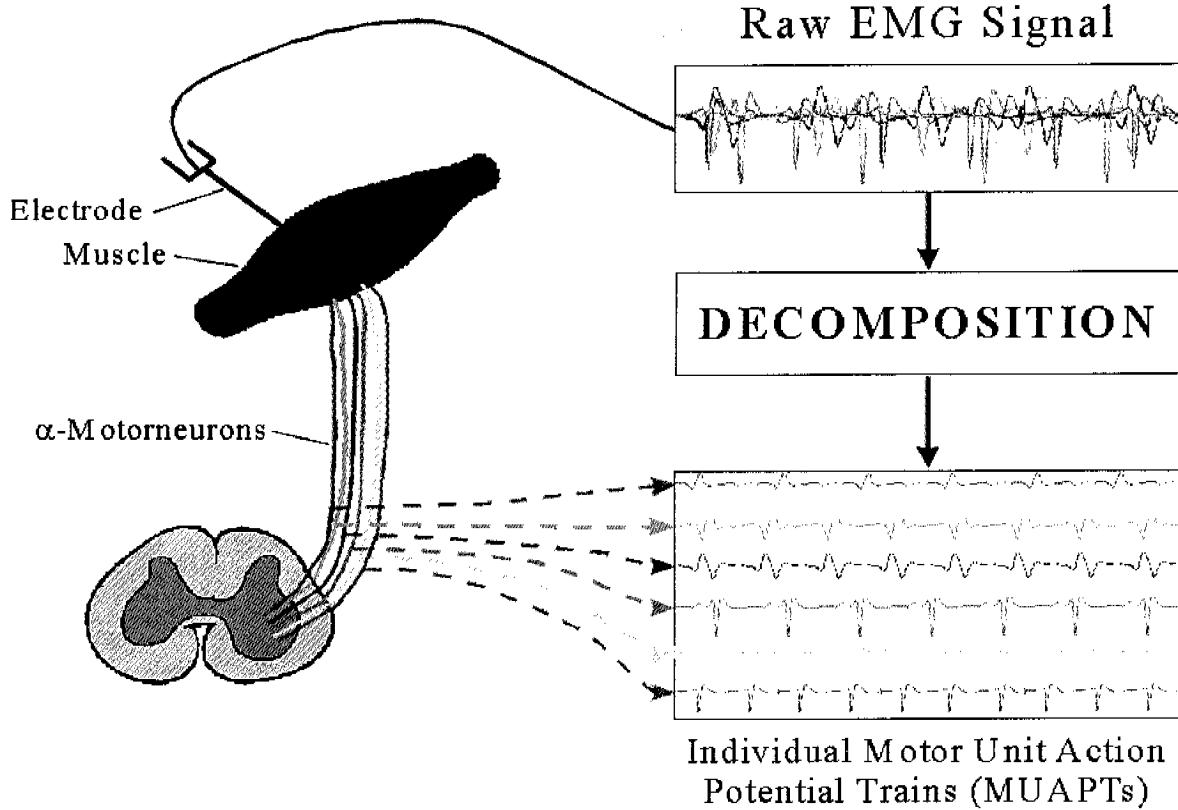


Figure 3.1: Pictorial outline of EMG signal decomposition. Reproduced from http://nmrc.bu.edu/tutorials/motor_units/decomp.html with permission from Prof. Carlo J De Luca.

3.2 Block Diagram Model

We can model an EMG signal and its generation by using a block diagram such the one as in Figure 3.2. The signals

$$X_i \triangleq (X_{i,1}, X_{i,2}, X_{i,3}, \dots), \quad (3.1)$$

with $X_{i,k} \in \{0, 1\}$, are discrete-time and binary signals modeling motor neuron activity. If $X_{i,k} = 1$, we say that motor neuron i fires at time k . This means that the firing time is the time at the start of a MUAP, not when it has the highest amplitude¹.

The MUAP shapes are modeled by finite impulse response (FIR) filters. An FIR filter produces a MUAP at its output when a 1 is applied at its

¹The location where the MUAP has its highest amplitude is sometimes used in the literature as the firing time.

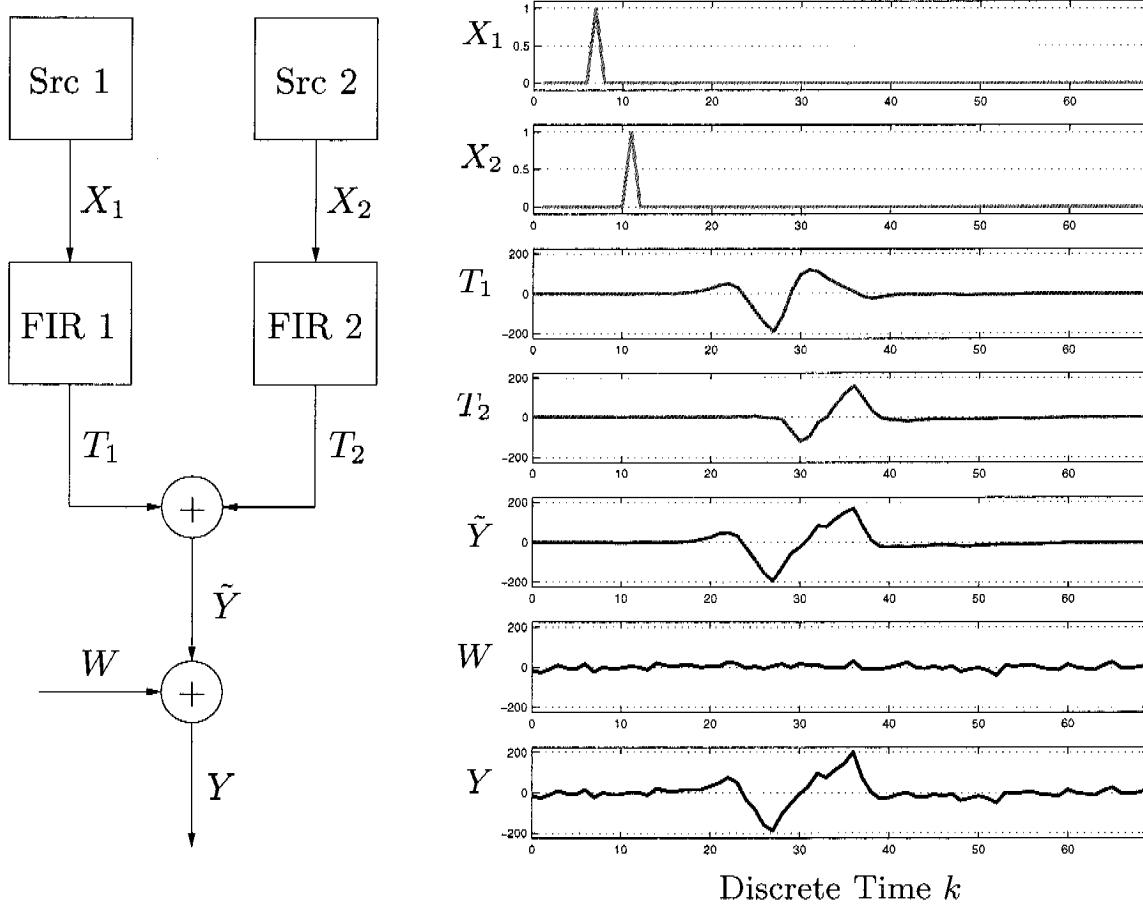


Figure 3.2: Left: A block diagram modeling the EMG signal generation process for the case of two sources. Right: Example signals corresponding to the block diagram.

input. The signals T_i , see Figure 3.2, are therefore the MUAP trains, W is additive noise, and Y is the measured EMG signal. In this example, the two signals X_1 and X_2 each contain a one at the locations of the onsets of the MUAPs in the MUAP trains T_1 and T_2 .

Given this notation, decomposing an EMG signal means estimating the source signals X_i given the EMG signal Y . In other words, we are looking for the firing times of the alpha motor neurons. More detailed information about this block diagram will be given in the next chapter.

If the MUAP templates (the filter coefficients of the FIR filters in Figure 3.2) are known and constant, it is easy to find those MUAPs in an EMG signal Y that do not overlap, such as the MUAPs in the left part

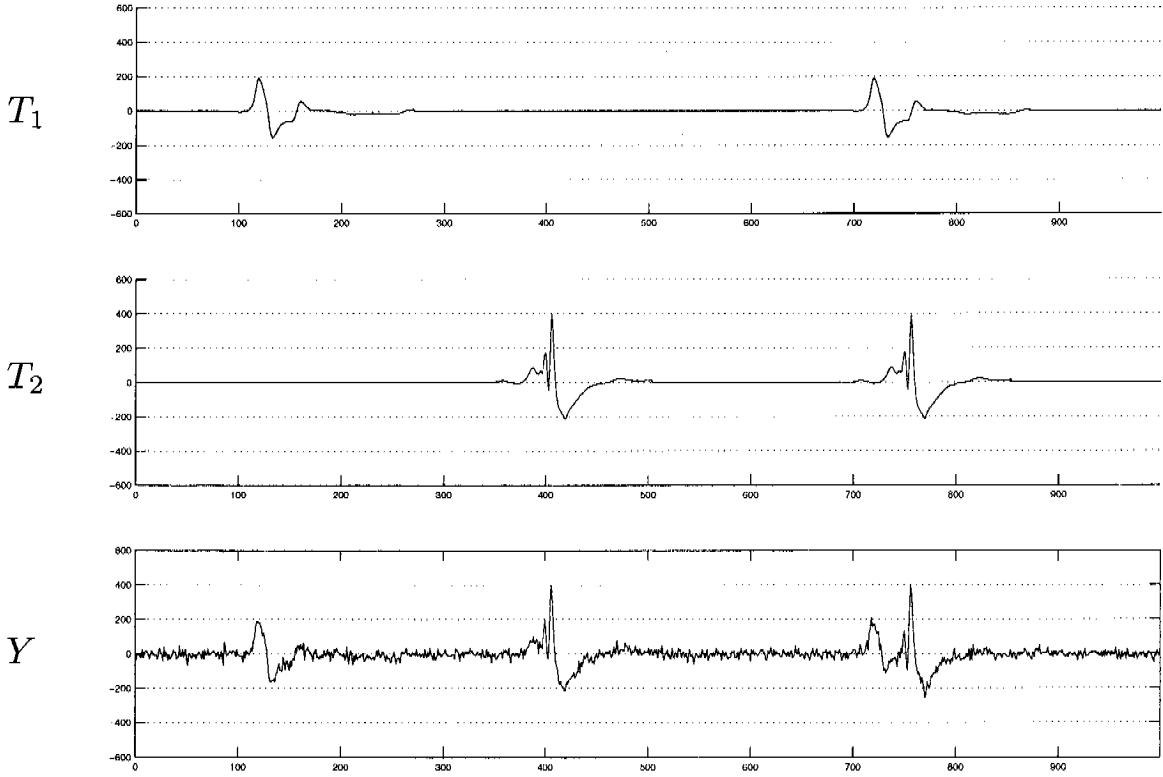


Figure 3.3: Example signals corresponding to the block diagram in Figure 3.2.

of Figure 3.3. However, if MUAPs from different motor units do overlap, as in the right part of Figure 3.3, the decomposition problem becomes more complicated. This is especially true in the case of destructive superpositions and/or when more than two action potentials overlap.

Many decomposition algorithms have been proposed. However, many practical tools are limited to a few superimposed MUAPs, e.g., only two in [142], since the computation time needed for the decomposition of stronger superpositions is prohibitive for more.

3.3 Generating Synthetic EMG Signals

To evaluate and to compare the performances of EMG signal decomposition algorithms, it is useful to start with decomposing synthetic EMG signal having known properties before using real measured EMG signals

with unknown and/or changing properties.

The simple block diagram in Figure 3.2 can already be used to generate EMG signals. In fact, the EMG signal Y in Figure 3.3 is a synthetic signal generated using such a block diagram. The following list contains a few parameters and properties that the user might want to change to test specific aspects of an EMG signal decomposition algorithm:

- Number of detected motor unit action potential trains;
- Firing frequencies and inter-spike-interval variations;
- MUAP shapes, their degree of similarity both for MUAPs generated by the same motor unit and for MUAPs generated by different motor units;
- Degree of MUAP superposition;
- Number of channels (number of detection surfaces);
- Channel offset and channel-offset variability;
- Type of noise and noise level;
- Filter properties of the skin and underlying tissue;
- Degree and manner of MUAP shape changes over time.

Stochastic Model of Single Cells

A spike train of an individual motor neuron may be approximated by a stochastic point process with independent and identically distributed inter-spike intervals (ISI), sometimes also called inter-pulse intervals (IPI). In the literature [130], a Gaussian distribution function was often used as an approximation of a real IPI histogram, but also other distributions were used, such as the Weibull distribution, the gamma distribution, and the Poisson distribution. Especially the Poisson process is a simple random process with no memory and only one parameter [63]. Overall, a Gaussian distribution seems to model the IPI of a motor unit well [130].

Noise

In simulated signals, the noise is often modeled as additive white Gaussian noise (AWGN) [49]. Another more realistic way, while still using simulated EMG signals, is to add the residual from measured EMG signals to a superposition of artificially created MUAP trains. A residual is the difference signal between an EMG signal and the corresponding reconstructed signal².

Changing MUAP Shapes

MUAP shapes from the same source change over time. Especially when decomposing long-term recordings, decomposition algorithms need to track the changing waveforms. To test decomposition algorithms in this regard, we adapted and extended [119] an EMG signal simulation tool by Farina [35].

3.4 Methods for EMG Signal Decomposition

Before computer algorithms were widely used to help with the decomposition of EMG signals, EMG signals were sometimes decomposed by identifying distinctively shaped MUAPs manually. In the meantime, several computer programs have been developed to automate the decomposition of EMG signals [29, 52, 84, 102, 104, 126, 142]. Today, many algorithms are not completely automatic but require some user interaction.

Algorithms do not only differ by the extent of automation, but also by what information they use and by the extent to which they are able to decompose EMG signals.

- Many modern EMG signal decomposition algorithms now use MUAP shape information and firing time information.
- There are algorithms that track changing MUAP shapes over time, which makes them suitable for the decomposition of long-term

²Refer to the glossary for an explanation on what the reconstructed signal is.

EMG signals [135].

- Some algorithms can deal with multiple channels.
- Some algorithms do not decompose superpositions at all, others aim at a full decomposition of EMG signals. Doing a full decomposition means detecting all firings of all active and detected motor units, which is especially important if one wants to not only detect global parameters like the mean firing rate of motor units, but also study complete motor unit firing patterns. This information is helpful, e.g., in studies of motor unit coordination, short-term synchronization, muscle architecture, and discharge irregularities [104].

Basic Steps

Many EMG signal decomposition algorithms use the following basic steps—or at least some of them [129, 135]:

Signal acquisition. See Chapter 2.

Segmentation. Most EMG signal decomposition algorithms segment an EMG signal into active and non-active parts first. Active parts may contain single MUAPs, superpositions of MUAPs, artifacts, noise, and combinations of these components. An easy way to segment an EMG signal is to use a threshold and some other criteria. See Section 6.2.

MUAP clustering. Similar active segments are grouped into clusters. Motor unit classes are then created for each cluster. The similarity measure depends on the MUAP representation in a feature space. In this way the MUAP shapes of individual motor units can be found.

Classification. If an active segment contains an individual MUAP, this MUAP is classified into its corresponding motor unit class. There is one class for each motor unit.

Resolution of superimposed MUAPs. In this step, which is not implemented in all EMG signal decomposition algorithms, the not already assigned active segments are analyzed. An algorithm checks,

if they are superpositions of two or more MUAPs. If a superposition can be resolved, the individual MUAPs are assigned to their corresponding classes.

Discovery of temporal relationships. Sometimes long MUAPs are split into smaller parts and the individual parts are treated temporarily as separate MUAPs. This may simplify the clustering and classification procedures, but it also leads to multiple linked MUAP trains. At the end, these trains need to be identified and merged.

Plausibility check. Some algorithms explicitly check the results to make sure that the decomposition is plausible. This can be done based on firing statistics, missing MUAPs, and additional MUAPs. Based on this evaluation, the decomposition result might be changed.

3.5 Resolving Superpositions

When two or more motor units generate action potentials that overlap in an EMG signal, the decomposition task becomes harder since the single MUAPs can no longer be classified easily. Especially in the case of partly destructive superpositions involving many MUAPs, sophisticated algorithms are necessary to resolve such superpositions.

There are different strategies to resolve superpositions [126, 129].

Peel-off approach. The peel-off approach, also called the sequential approach, matches MUAP templates individually to superpositions in an EMG signal. If a match is good enough, the MUAP template is subtracted from the superposition and the next MUAP template is tried. This approach does not work well if the superpositions look very different from any single MUAP template.

Modeling approach. Here, superpositions are synthesized from a collection of MUAP templates by adding these templates with varying relative time shifts. The resulting superpositions are then compared to a superposition in an EMG signal, which is to be resolved. This approach is capable to deal with destructive properties, but it is computationally much more expensive due to the large search space, especially when several MUAP templates can be involved

in a superposition. Practical algorithms that use the modeling approach try to reduce the computational effort by limiting the number of contributing MUAPs, initially aligning them, and then using optimization techniques in the reduced search space to find the correct alignment.

Combinations of peel-off and modeling approach. It is also possible to combine the two previous approaches [34, 96]. One approach is to create sorted lists of possible motor units that contribute to a superposition. Knowledge from the EMG signal that is decomposed, such as average firing rates, can be used to generate such a sorted list. Both rule-based expert systems and the template energy can also be used to sort the list of possible motor units creating a superposition.

When developing algorithms that resolve superpositions, it is often differentiated between the unknown-constituent problem and the known-constituent problem. In the known-constituent problem, the algorithm knows which sources are active within a superposition. In the unknown-constituent problem, the algorithm has all MUAP shapes but does not know which MUAPs make up a given superposition.

3.6 Previous Work

Electromyographic signal decomposition has been a research topic at our lab for about 30 years, resulting in 7 doctoral theses [33, 46, 49, 51, 52, 131, 135]. Also beyond our lab, many EMG signal decomposition methods have been proposed. Refer to [101, 126, 126, 127, 129, 140] for a good overview. Only a small sample of the many approaches can be briefly mentioned here to illustrate a few approaches. Also, we will concentrate on recent algorithms for resolving superpositions in EMG signals.

- De Luca and his collaborators have worked a long time on electromyography: already in 1972 they described in [30] the quadri-filar, three channel, differential EMG electrode. Many papers on EMG signal decomposition followed, such as [83–85, 99]. The algorithm in [84] uses combinations of two MUAPs to reconstruct superpositions. The best approximations are weighted with their probabilities of occurring.

- More than 20 years ago, McGill, Cummins, and Dorfman developed the system ADEMG (Automatic Decomposition Electromyography) [102]. However, like many systems, it was unable to resolve superpositions.
- Haas developed an algorithm [52] that tests whether single MUAP templates occur in an active segment. If this is not the case, the algorithm reconstructs superpositions using two and three MUAP templates. Finally, the algorithm chooses the combination that matches the active segment best. The algorithm can decompose superpositions of up to three MUAPs.
- Gut [49,50] used communication techniques to decompose superimposed action potentials in EMG signals. He interpreted an EMG signal as the output of a channel. Superpositions were decomposed by either a maximum-likelihood (ML) sequence detector or by a maximum-a-posteriori (MAP) sequence detector. For this, the Viterbi algorithm, which calculates the most probable path through a trellis in an efficient way [116], was chosen to estimate the firing patterns of the motor units (sources). Although this approach is precise (an optimal approach), it is computationally very expensive, especially in the case of strong superpositions.
- Etawil and Stashuk [34] developed an algorithm for resolving superimposed MUAPs. It is based on the peel-off approach and reduces the number of possible combinations of MUAPs making up a superposition. For this, temporal relationships between and within motor unit action potential trains and MUAP energy information is used.
- Wellig [135, 142] developed a tool that is tailored towards the decomposition of long-term EMG signals. This tool was named EMG-LODEC (electromyogram long-term decomposition). Superpositions are resolved by using the modeling approach, i.e., by creating reconstructed signals using reference signals (averaged and tracked MUAPs). By minimizing an error between the superposition and the reconstructed signal, the superposition can either be resolved or it can be classified as not resolvable. No firing time information is used. For practical applications [142], only superpositions of a maximum of two MUAPs were decomposed.
- Recently, artificial intelligence algorithms were introduced to improve the decomposition quality [53–55, 60]. They use a probabilis-

tic framework, i.e., the IPUS framework for integrated processing and understanding of signals [86]. The resolution of superpositions, which does not need a segmentation step³, could be improved using this approach.

- In 2002, McGill described an optimal algorithm for resolving superimposed action potentials [101]. It aligns a set of MUAP templates with a given superposition in an EMG signal by minimizing the Euclidean distance between the sum of the aligned templates and the superposition in the EMG signal. The algorithm solves this optimization problem in an artful way using a recursive approach. It searches all possible discrete-time alignments, starting with the most likely ones. Once it can be verified that the optimal alignment has been found, the algorithm stops. This approach leads to an optimal and fast algorithm for the resolution of superpositions. The algorithm always finds the smallest MSE between the superposition and the reconstructed signal. The superpositions can involve six or more templates [104], which might be similarly shaped. The algorithm can also deal with destructive interference and added noise.
- Later, McGill and colleagues developed a computer-aided decomposition program for single- and multi-channel signals called **EMGLAB** [104], which they validated in [105]. This program can decompose signals recorded with needle or fine-wire electrodes during low and moderate levels of muscular contraction. Besides providing a powerful user interface that allows manual editing and verification of the results, it uses advanced algorithms for template matching and it can resolve superimpositions [101]. The open-source computer program EMGLAB is available for download [1].

3.7 Problems and Difficulties of EMG Signal Decomposition

In this section, some problems and difficulties are listed that one faces when decomposing measured EMG signals. More information on this topic is presented in Chapter 6.

³Personal communication with S. Hamid Nawab, EMBC, August 31, 2006.

Superpositions of MUAPs

We have just stated that MUAPs from different motor units can overlap. As the level of muscle force increases, the number of active motor units increases as well as their activation rates. This leads to more overlapping MUAPs, which makes the signal decomposition task harder. In an extreme case, two or more MUAPs may cancel out or yield a waveform that is similar to another MUAP template. Also, as a general rule, resolving a superposition becomes more difficult the more MUAPs make up this superposition.

MUAP Shapes and Nonstationary Firing Statistics

For a good decomposition, MUAPs of the same motor unit should be similar over time and MUAPs from different motor units should be different. Unfortunately, MUAP shapes may change over time due to noise, artifacts, time offsets from the sampling process, moving electrodes, physiological jitter, and other physiological processes. In addition, MUAPs from different motor units may not be very different. In this case, the decomposition algorithm has to rely on firing statistics for inter-spike intervals. However, such firing statistics may also change over time, especially during experiments with dynamic contractions. This may be a problem if the decomposition algorithm has to rely on firing statistics, e.g., in the presence of high levels of noise and/or not distinct MUAP shapes.

Douplets

One example of a nonstationary firing behavior are douplets. A doublet is a collection of two MUAPs from the same motor unit, where the second MUAP occurs immediately after the first one. Therefore, the inter-spike interval between the two action potentials is much shorter than it is normally the case. In addition, the second MUAP usually has a different shape than the first one. A decomposition algorithm that heavily relies on firing statistics, e.g., because of similar MUAP templates and/or high noise levels, can easily miss douplets.

Dynamic Contractions

In case of dynamic contractions, both inter-spike intervals and MUAP shapes change rapidly over time, which makes the decomposition task hard. One way to reduce the problem is to use multiple electrodes/channels.

Noise and Artifacts

Although we usually use an AWGN model, real EMG signals contain several other components, such as artifacts due to moving electrodes. Also, MUAPs generated by distant fibers have usually a small amplitude and they are similar in shape. It is therefore not easy to classify them correctly. For this reason, they become a part of the background noise. These small MUAPs can change the shapes of large MUAPs when firing occurs at a similar time.

Ill-Posed Problem / Not Well Conditioned Problem

EMG signal decomposition is an inverse problem, i.e., model parameters (firing times, MUAP shapes) are estimated based on measured EMG signals. Like many inverse problems, the problem of EMG signal decomposition is ill-posed since the model parameters may not be unique. The case where two motor units generate exactly identical MUAP shapes is one extreme example. Others are where two or more superimposed MUAPs cancel each other completely or create the shape of a third motor unit.

In general, noise and artifacts can make the decomposition problem less well conditioned, and the optimal solution with the minimum mean squared error (MMSE) may no longer be the correct one. Noise is especially a problem if superposition have amplitudes well below the noise level.

To deal with or to lessen these problems, decomposition algorithms must not only rely on MUAP shape information but also use appropriate a priori information, for example a source model with suitable firing statistics. However, this can lead to wrong results, e.g., if doublets occur.

Recruitment and Decruitment

Idle motor units may start firing, usually when the muscle force level is increased. They can also stop firing. This has to be considered, especially when developing EMG signal decomposition algorithms for long-term recordings.

Many Active Motor Units

If there are many active motor units, the decomposition algorithm can fit EMG signal parts closely by creating strong superpositions of many action potentials. This is especially a problem for a high noise level case and many small MUAP waveforms. This problem can be lessened by using appropriate source models that impose a cost each time a motor unit fires.

3.8 Summary, Conclusions, and Outlook

In this chapter we have introduced and defined the problem of EMG signal decomposition. We have also outlined previous work with a focus on the resolution of superpositions in EMG signals. Next we will explain a new model-based approach for resolving superpositions.

Seite Leer /
Blank leaf

Chapter 4

Resolving Superpositions Using Factor Graphs

Until the problem is better understood, a more formal definition of consciousness is likely to be either misleading or overly restrictive, or both. If this seems evasive, try defining a gene.

Is it a stable unit of hereditary transmission? Does a gene have to code for a single enzyme? What about structural and regulatory genes? Does a gene correspond to one continuous segment of nucleic acid? What about introns? And wouldn't it make more sense to define a gene as the mature mRNA transcript after all the editing and splicing have taken place? So much is now known about genes that any simple definition is likely to be inadequate. Why should it be any easier to define something as elusive as consciousness?

Historically, significant scientific progress has commonly been achieved in absence of formal definitions. For instance, the phenomenological laws of electric current flow were formulated by Ohm, Ampère, and Volta well before the discovery of the electron in 1892 by Thompson.

From *The Quest for Consciousness* [64] by Christof Koch

This chapter explains how superpositions in electromyographic signals can be resolved using a novel model-based signal processing approach. We used recent ideas from signal processing, in particular, iterative “turbo” signal processing based on graphical models (“factor graphs”) [90].

Graphical models such as factor graphs allow to model complex systems and signals in a consistent, unified, and systematic way. They help to derive a wide variety of signal processing algorithms. A factor graph is a graphical representation of a mathematical model. The edges in the factor graph correspond to the variables in the model. The nodes in the factor graph represent relationships between variables [94].

We use factor graphs to model a class of signals, in our case EMG signals. Based on that, we have developed non-optimal message-passing algorithms that can decompose difficult superpositions in EMG signals. The superpositions may consist of many overlapping action potentials. Using these non-optimal message-passing algorithms on factor graphs allowed us to work with more complex models than it has previously been possible [94].

After a motivation, this chapter starts with the already introduced block diagram representing a simulation model for EMG signals. To develop an algorithm for resolving superpositions, we introduce the theory of factor graphs and develop a factor graph based on the given block diagram. Such a factor graph represents a factorization of the probability distribution of our EMG signal model.

Then we show how superpositions in EMG signals can be resolved by using message-passing algorithms. These algorithms can efficiently solve inference problems by passing local messages. In our case this means finding an approximation of the MAP estimate of motor unit firing times, i.e., we try to estimate the source signals given the observed EMG signals.

4.1 Motivation

Electromyography is an essential research method and a vital clinical diagnostic tool applied in many procedures every day. Several algorithms for the resolution of superpositions in EMG signals have been proposed. But especially in the case of many overlapping action potentials, these

algorithms could often not resolve superpositions correctly. Other (optimal) approaches are in many cases computationally not feasible when superpositions of many action potentials are to be decomposed.

Also, many traditional EMG signal decomposition algorithms use basically two steps: segmentation and classification. In the case of long superpositions and/or high levels of noise or artifacts in an EMG signal, segmenting an EMG signal can be problematic.

Our new algorithms, which are based on factor graphs, allow the decomposition of superpositions consisting of many overlapping action potentials, as we have shown, e.g., in [66–72]. We also use a one-step approach so that no segmentation of EMG signals is necessary.

4.2 Block Diagram Model for EMG Signals

Many models of EMG signals have been proposed. In this section, a very simple model is presented. It can be used to better understand EMG signals as well as to generate synthetic EMG signals. It will also be the basis for the factor graph that we used to develop our new EMG signal decomposition algorithms. For further information on more sophisticated EMG signal modeling and simulation refer to [130].

Alpha Motor Neuron or Source Model

If an alpha motor neuron sends an action potential to its muscle fibers, we say that the alpha motor neuron fires. We can model such an alpha motor neuron (source) by using a random number generator. It generates random numbers 0 (idle) and 1 (firing) according to some probability distribution, e.g., a Gaussian or Poisson distribution. In other words, an action potential train of an individual alpha motor neuron may be approximated by a stochastic point process with independent and identically distributed inter-spike intervals, see Section 3.3.

Model for the MUAP Shapes

Each MUAP shape is modeled by a finite impulse response (FIR) filter with filter coefficients h_ℓ , $\ell = 0 \dots M$, where M is the order of the FIR filter. The filter performs a convolution of the binary input signal X with the filter impulse response function h :

$$T_k = \sum_{\ell=0}^M X_{k-\ell} \cdot h_\ell. \quad (4.1)$$

The filter coefficients correspond to the MUAP shape so that the impulse response of the filter is identical to the MUAP modeled. In simple words, a 1 in the input signal X at time k leads to a MUAP in the output signal T starting at time k .

Noise Model

We usually use an additive white Gaussian noise (AWGN) model. Later, in the decomposition algorithm, this corresponds to a least-squares fit. This means that given an observed EMG signal Y and all FIR filter coefficients modeling the MUAP shapes, we seek binary source signals X (i.e., firing times) that minimize the sum of squared differences between the EMG signal Y and the reconstructed EMG signal \tilde{Y} (see also Figure 4.1).

Formally, this means that we wish to choose the model parameters X as follows:

$$\hat{X} = \underset{X}{\operatorname{argmin}} \sum_k (Y_k - \tilde{Y}_k(X))^2. \quad (4.2)$$

Although the above equation looks rather innocent, in practice it is hard to find the “optimal” parameters X . Much of the research in signal processing (and, in fact in this thesis) is about finding suitable approximations.

Block Diagram for One Electrode

As briefly introduced in Chapter 3, Figure 4.1 shows the block diagram for EMG signal generation. It contains the source model, the model for

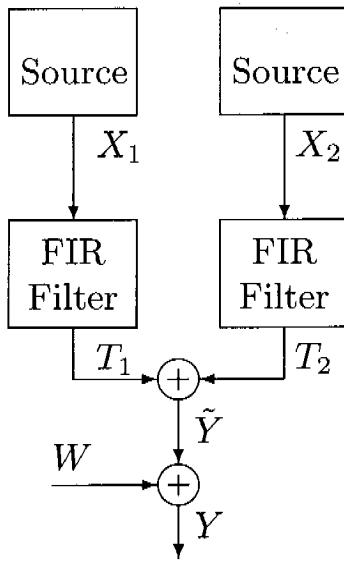


Figure 4.1: Model of EMG signal generation for two sources and one channel.

the MUAP shapes, and the noise model. The model also incorporates *multiple* sources; in Figure 4.1 two sources are included. A block diagram such as the one in Figure 4.1 can be used to model and to generate synthetic EMG signals. The generation process works as follows: source i emits a discrete-time binary signal

$$X_i \triangleq (X_{i,1}, X_{i,2}, X_{i,3}, \dots) \quad (4.3)$$

with $X_{i,k} \in \{0, 1\}$. If $X_{i,k} = 1$, we say that source i “fires” at time k .

Each filtered source signal gives a motor unit action potential train. For source i this is

$$T_i \triangleq (T_{i,1}, T_{i,2}, T_{i,3}, \dots) \quad (4.4)$$

with

$$T_{i,k} \triangleq \sum_{\ell=0}^{M_i} X_{i,k-\ell} \cdot h_{i,\ell}, \quad (4.5)$$

where M_i is the order of the FIR filter corresponding to source i and where $h_{i,\ell} \in \mathbb{R}$ is the ℓ -th filter coefficient expressing the waveform of the MUAP generated by source i .

In general, more than one source is active. Therefore, the EMG signal without noise

$$\tilde{Y} \triangleq (\tilde{Y}_1, \tilde{Y}_2, \tilde{Y}_3, \dots) \quad (4.6)$$

is the summation of all MUAP trains with

$$\tilde{Y}_k \triangleq \sum_{i=1}^{N_{\text{src}}} T_{i,k}, \quad (4.7)$$

where N_{src} is the number of sources.

Finally, the electrode picks up a noisy version of this sum signal

$$Y \triangleq (Y_1, Y_2, Y_3, \dots), \quad (4.8)$$

with

$$Y_k \triangleq \tilde{Y}_k + W_k \quad (4.9)$$

$$= \sum_{i=1}^{N_{\text{src}}} T_{i,k} + W_k \quad (4.10)$$

$$= \sum_{i=1}^{N_{\text{src}}} \sum_{\ell=0}^{M_i} X_{i,k-\ell} \cdot h_{i,\ell} + W_k, \quad (4.11)$$

where

$$W \triangleq (W_1, W_2, W_3, \dots) \quad (4.12)$$

is additive white Gaussian noise (AWGN). The noise samples are i.i.d., each one is normally distributed:

$$W_k \sim \mathcal{N}(w_k \mid m, \sigma^2). \quad (4.13)$$

Here, m is the mean, which is usually zero, and σ is the standard deviation of the normal distribution.

Therefore, the electrode picks up a noisy and filtered superposition of the source signals X_i .

Block Diagram for More Than One Electrode

Multi-channel EMG signals

Each electrode picks up a noisy and differently filtered superposition of the source signals X_i .

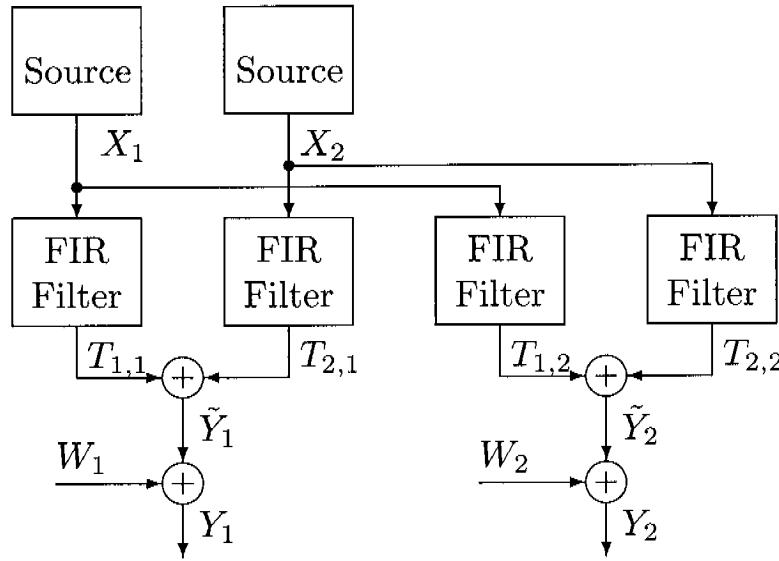


Figure 4.2: Model of EMG signal generation for two sources and two channels.

If we have more than one recording electrode (or sensor, or channel) that measure the activity of the active motor units, we have to extend the model in Figure 4.1. The block diagram in Figure 4.2 shows the two-electrode case.

Both electrodes pick up noisy and differently filtered superpositions of the source signals X_i . We now re-define the variables h , T , \tilde{Y} , W , and Y .

The motor unit action potential train for source i and electrode j is

$$T_{i,j} \triangleq (T_{i,j,1}, T_{i,j,2}, T_{i,j,3}, \dots) \quad (4.14)$$

with

$$T_{i,j,k} \triangleq \sum_{\ell=0}^{M_i} X_{i,k-\ell} \cdot h_{i,j,\ell}, \quad (4.15)$$

where M_i is the order of the FIR filter corresponding to source i and where $h_{i,j,\ell} \in \mathbb{R}$ is the ℓ -th filter coefficient expressing the waveform of the MUAP generated by source i in channel (electrode) j .

In general, more than one source is active. Therefore, the EMG signal without noise

$$\tilde{Y}_j \triangleq (\tilde{Y}_{j,1}, \tilde{Y}_{j,2}, \tilde{Y}_{j,3}, \dots) \quad (4.16)$$

is the summation of all MUAP trains with

$$\tilde{Y}_{j,k} \triangleq \sum_{i=1}^{N_{\text{src}}} T_{i,j,k}, \quad (4.17)$$

where N_{src} is the number of sources.

We define¹ the multi-channel EMG signal Y as

$$Y \triangleq (Y_1, Y_2, Y_3, \dots), \quad (4.18)$$

where the index refers to the channel number. Electrode j , which corresponds to channel j , picks up a noisy version of this sum signal

$$Y_j \triangleq (Y_{j,1}, Y_{j,2}, Y_{j,3}, \dots), \quad (4.19)$$

with

$$Y_{j,k} \triangleq \tilde{Y}_{j,k} + W_{j,k} \quad (4.20)$$

$$= \sum_{i=1}^{N_{\text{src}}} T_{i,j,k} + W_{j,k} \quad (4.21)$$

$$= \sum_{i=1}^{N_{\text{src}}} \sum_{\ell=0}^{M_i} X_{i,k-\ell} \cdot h_{i,j,\ell} + W_{j,k}, \quad (4.22)$$

where

$$W_j \triangleq (W_{j,1}, W_{j,2}, W_{j,3}, \dots) \quad (4.23)$$

is additive white Gaussian noise (AWGN) for electrode j . The samples are i.i.d., each one is normally distributed:

$$W_{j,k} \sim \mathcal{N}(w_{j,k} \mid m_j, \sigma_j^2). \quad (4.24)$$

Here, m_j is the mean, which is usually zero, and σ_j is the standard deviation of the normal distribution of electrode j .

¹Remember that we re-define variable Y here.

Block diagram model of EMG signals

The block diagram in Figure 4.2 explicitly models

- Several motor unit sources,
- The MUAP shapes,
- Additive superposition of the contributions from the motor units,
- Additive noise, and
- Multiple electrodes.

4.3 Problem Statement

Decomposition in General

Given the block diagram in Figure 4.2, decomposing a multi-channel EMG signal

$$Y \triangleq (Y_1, Y_2, Y_3, \dots) \quad (4.25)$$

means estimating the binary source signals

$$X_i = (X_{i,1}, X_{i,2}, X_{i,3}, \dots) \quad (4.26)$$

for all sources i .

Remark 4.1. (Non-Linear Problem)

Note that, although the filtering process is linear, the problem of EMG signal decomposition is non-linear. The reason for this is that the source signals X_i are binary. The factor graph approach, which we are going to explain next, can explicitly model the binary source signals. \square

In this chapter and in Chapter 5 we assume the following to make the problem simpler:

- Only a single superposition is given. It is not our goal to decompose EMG signals consisting of many active segments.

- All MUAP shapes, i.e., the filter coefficients $h_{i,j,\ell}$, are known to the algorithm.
- The filter coefficients are also expected to be constant over time.

4.4 Introduction to Factor Graphs

The block diagram in Figure 4.2 can be used to simulate synthetic EMG signals. However, it can not be used directly to decompose EMG signals, i.e., to estimate the binary source signals X_i given the EMG signals Y . For this, we developed a factor graph based on the block diagram in Figure 4.2. Message passing on this factor graph then allows decomposing EMG signals.

Before the actual message-passing algorithm is derived and explained, we give a short introduction to factor graphs. The factor graph framework offers a language for signal and system modeling. This is helpful for the development of complex and practical detection and estimation algorithms.

Due to the graphical representation, which electrical engineers and others find familiar since they are used to block diagrams and electric circuits, the problem at hand is comprehended easier. Also, once a mathematical model is expressed in the factor graph language, modifications and extensions can easily and intuitively be made.

In this thesis we use a factor graph notation as in [42]. We sometimes refer to these factor graphs as Forney-style factor graphs (FFG).

For further information on factor graphs refer to the tutorial paper by Loeliger [90] and to [42, 77, 134, 137].

Factor graphs

Factor graphs are graphical representations of complex mathematical models. They have a modular structure, which helps comprehending, changing, and extending the models themselves as well as the algorithms derived based on factor graphs. Factor graphs allow a unified approach to many signal processing problems and beyond.

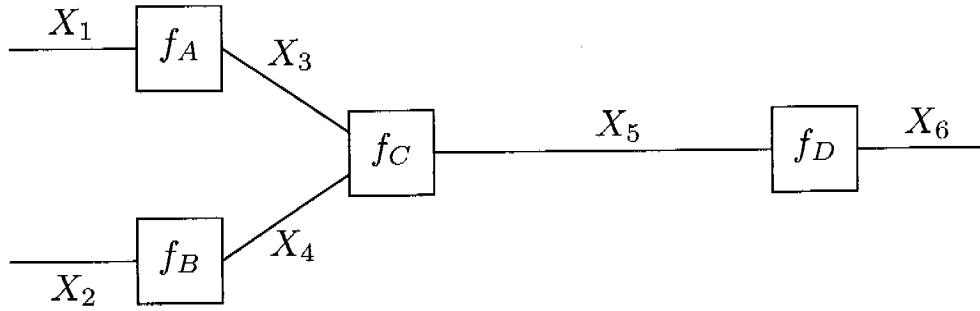


Figure 4.3: Example of a simple factor graph representing the factorization $f_A(x_1, x_3) f_B(x_2, x_4) f_C(x_3, x_4, x_5) f_D(x_5, x_6)$.

4.4.1 Example 1: A Simple Factor Graph

In general, a factor graph is a graphical model, which represents factorizations of arbitrary multivariate functions and dependencies among variables. For example, let the function $f(x_1, x_2, x_3, x_4, x_5, x_6)$ factor as

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = f_A(x_1, x_3) f_B(x_2, x_4) f_C(x_3, x_4, x_5) f_D(x_5, x_6). \quad (4.27)$$

where all variables are discrete variables, i.e., $x_i \in \mathbb{Z}$. This factorization is represented by the factor graph in Figure 4.3.

Given the exemplary factorization in (4.27) and its factor graph representation in Figure 4.3, we now present some definitions and rules for interpreting and drawing factor graphs.

4.4.2 Factor Graph Definitions and Rules

Factor graph: A Forney-style factor graph (FFG), such as the one in Figure 4.3, represents a factorization of a global function. In the example, the global function is $f(x_1, x_2, x_3, x_4, x_5, x_6)$ and the factorization is $f_A(x_1, x_3) f_B(x_2, x_4) f_C(x_3, x_4, x_5) f_D(x_5, x_6)$. A factor graph consists of nodes and edges.

Node: There is a node for every factor. In the example, the factors are f_A , f_B , f_C , and f_D . These factors are also called local functions.

Edge: There is an edge or a half-edge for every variable. Edges in the factor graph above are given for x_3 , x_4 , and x_5 , half-edges for the

variables x_1 , x_2 , and x_6 . Each edge is connected to exactly two nodes, each half-edge is connected to exactly one node only.

Connection: An edge (or half-edge) is connected to a node if and only if the factor (corresponding to the node) is a function of the variable (corresponding to the edge). For example, the half-edge that corresponds to x_1 is connected to the node that corresponds to f_A since $f_A(x_1, x_3)$ is a function of x_1 .

Number of connections per edge: An edge is connected to no more than two nodes. If three or more local functions are functions of the same variable, an equality constraint node can be used to “clone” variables. See Section 4.4.5 for further information on this topic.

Configuration: A configuration ω is a particular assignment of values to all variables. For the exemplary factor graph in Figure 4.3, one configuration would be

$$\omega = (x_1, x_2, x_3, x_4, x_5, x_6) = (2, 4, 3, 2, 1, 3). \quad (4.28)$$

Configuration space: The configuration space Ω is the set of all possible configurations, i.e., it is the domain of the global function f . The configuration space for our example above is

$$\Omega = \mathbb{Z}^6. \quad (4.29)$$

In this chapter and in Section 5.2 of the next chapter we will assume discrete variables only. In later sections, we will also use continuous variables.

Valid configuration: We say that a configuration ω is valid if $f(\omega) \neq 0$.

Arrows: Sometimes arrows are included on the edges of the factor graph. However, this is basically redundant since the relationships among variables is already defined by the node functions, for example in (4.34). On the other hand, it is often convenient to draw a factor graph with arrows to indicate how the system model could be simulated; see Figure 4.11. Arrows can also be used to name messages in a simple way; see (4.59).

Three simple rules to draw a factor graph [89]

- There is a node for every factor.
- There is an edge (or half edge) for every variable.
- The node representing some factor g is connected with the edge (or half edge) representing some variable x if and only if x is an argument of g .

Given these simple rules and explanations, a factor graph can now easily be drawn given a factorization of a function. As a reminder, it is our goal to develop a factor graph for EMG signal decomposition based on the block diagram in Figure 4.2. Before we do this, we first present a few more simple examples.

4.4.3 Example 2: Multivariate Probability Distribution

As we have stated earlier, factor graphs represent factorizations of arbitrary multivariate functions and dependencies among variables. We will often use factor graphs to represent probabilistic models. In this case the following holds:

- The global function f as well as the local functions are probability distributions.
- The sample space Ω of probability theory², which is the set of all possible outcomes of an experiment, corresponds to the configuration space Ω .
- $f : \Omega \rightarrow [0, 1]$.

For example, a node function can be the multivariate probability distribution

$$f(x, y) = p_{XY}(x, y) \quad (4.30)$$

²Refer to Appendix A.

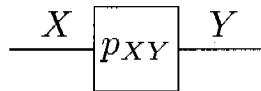


Figure 4.4: Factor graph depicting a multivariate probability distribution $p(x, y)$.

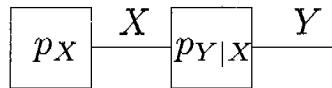


Figure 4.5: Factor graph depicting the factorization $p(x)p(y|x)$.

of two discrete random variables X and Y . This function can be depicted by the simple factor graph in Figure 4.4, consisting of only one node, corresponding to the node function $p_{XY}(x, y)$, and two edges, corresponding to the random variables X and Y .

The multivariate probability distribution $p_{XY}(x, y)$ can factor in different ways. According to the definition of the conditional probability, which holds in general, the factorization

$$p(x, y) = p(x)p(y|x) \quad (4.31)$$

is always possible³. This factorization is depicted in Figure 4.5.

If the random variables X and Y are independent, then the multivariate probability distribution factors as

$$p(x, y) = p(x)p(y) \quad (4.32)$$

and the factor graph consists of two unconnected components as in Figure 4.6.

4.4.4 Example 3: Sum Constraint Node

A node function in a factor graph can also represent a deterministic relationship found in a block diagram. For example, consider the sum node in Figure 4.7, which represents a deterministic relationship. To incorporate such a relationship into a factor graph representing a probabilistic

³Refer to Appendix A.

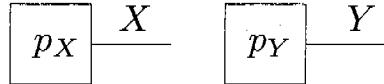


Figure 4.6: Factor graph depicting the factorization $p(x)p(y)$.

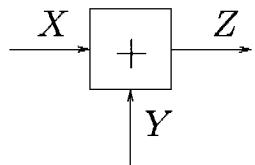


Figure 4.7: Block diagram for $z = x + y$.

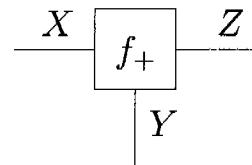


Figure 4.8: Corresponding factor graph with node function f_+ .

model, the corresponding node function has to evaluate to 1 if the configuration is valid, i.e., if the variables have values that are consistent with the node function. Hence, for this example, the node function for the sum constraint node is

$$f_+(x, y, z) \triangleq \begin{cases} 1, & \text{if } z = x + y \\ 0, & \text{else} \end{cases} \quad (4.33)$$

$$= \delta[z - x - y] \quad (4.34)$$

and the corresponding factor graph is shown in Figure 4.8. $\delta[.]$ is the Kronecker delta, which evaluates to 1 if its discrete argument is 0, otherwise the Kronecker delta evaluates to 0.

4.4.5 Example 4: Equality Constraint Node

Another example for an important deterministic relationship is the equality node. The block diagram of the equality node is given in Figure 4.9. As for the sum constraint node, the corresponding node function has to evaluate to 1 if the configuration is valid, i.e., if the variables have values that are consistent with the node function. For this example, the node function for the equality constraint node is given by

$$f_=(x, y, z) \triangleq \begin{cases} 1, & \text{if } z = x = y \\ 0, & \text{else} \end{cases} \quad (4.35)$$

$$= \delta[z - x]\delta[z - y] \quad (4.36)$$

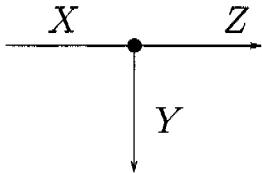


Figure 4.9: Block diagram for $z = x = y$.

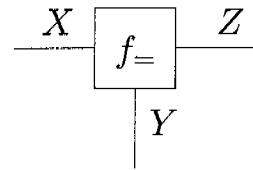


Figure 4.10: Corresponding factor graph with node function $f=$.

and the corresponding factor graph is given in Figure 4.10.

This node is important since the factor graph rules assume that no variable appears in more than two factors. If we have a variable that appears in more than two factors, we need to introduce auxiliary variables and an auxiliary factor, which is the equality constraint node. We call this process variable cloning. It allows us to always enforce the condition that no variable appears in more than two factors. In other words, the equality constraint node is a branching point in the factor graph [92].

4.5 Factor Graph Model for EMG Signals

After we introduced the basic factor graph theory that is needed for the next steps, we now explain how a discrete time signal, such as an EMG signal, can be modeled using the factor graph notation.

The new factor graph is based on the block diagram in Figure 4.2. Figure 4.11 shows one section of this factor graph. Note the topological similarities and differences between the block diagram in Figure 4.2 and the complete factor graph. An important difference between the block diagram and the factor graph is that the complete factor graph has one section—such as the one in Figure 4.11—for each discrete time index k in the EMG signal. This means that every variable reappears in each time slice. This allows sequential processing of the whole signal, which is computationally of advantage.

Next, we briefly describe some components of the factor graph in Figure 4.11.

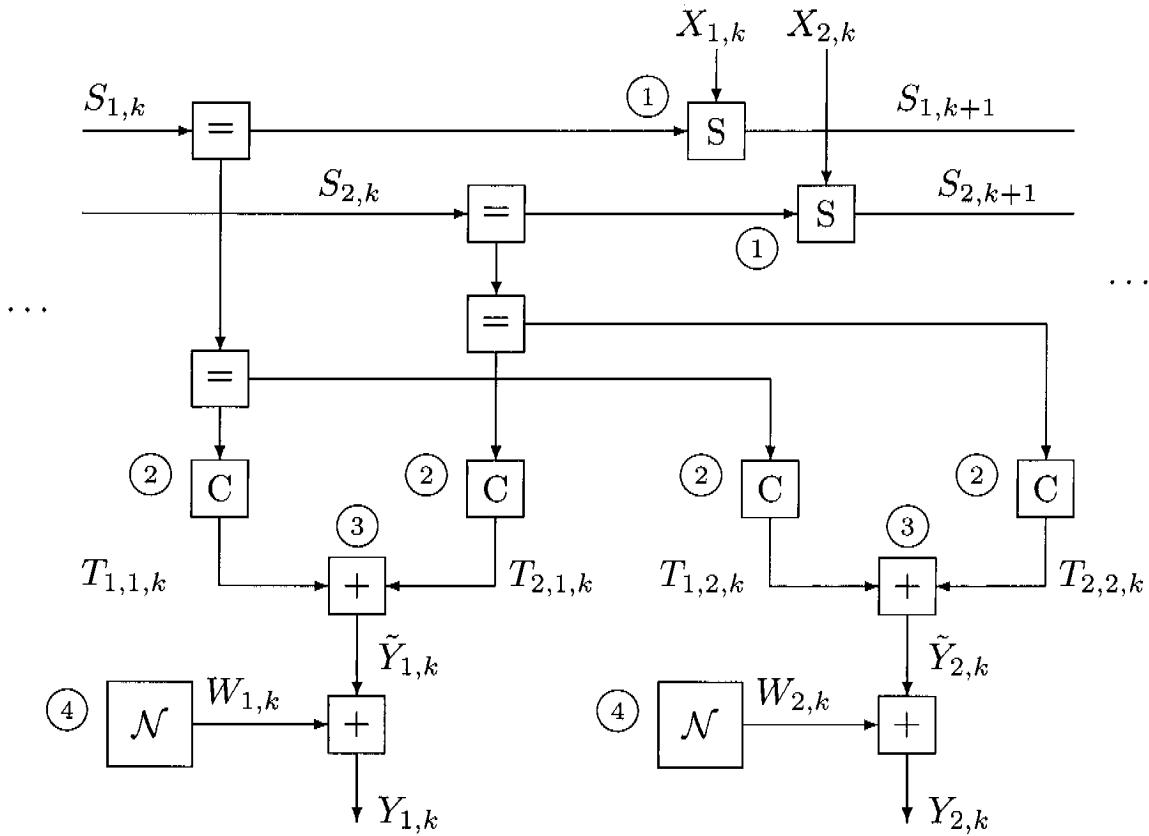


Figure 4.11: Time- k section of a factor graph for EMG signal decomposition.

- The boxes labeled ① represent the source models. They are basically state transition probabilities $p(s_{i,k+1}|s_{i,k})$, which are defined by a finite state model. Therefore, these nodes represent a probabilistic model such as the one in the example in Figure 4.5.
- The boxes labeled ② represent the coefficient nodes. They translate between FIR filter state values S and MUAP values T .
- The boxes labeled ③ are the sum constraint nodes as in Figure 4.8, modeling the summation of the MUAP trains.
- Finally, the boxes labeled ④ model additive noise.

These and the other boxes in Figure 4.11 correspond local functions and all local functions give the global function as introduced in Section 4.4.2.

More detailed definitions and explanations regarding the individual nodes of the factor graph are given in the next chapter.

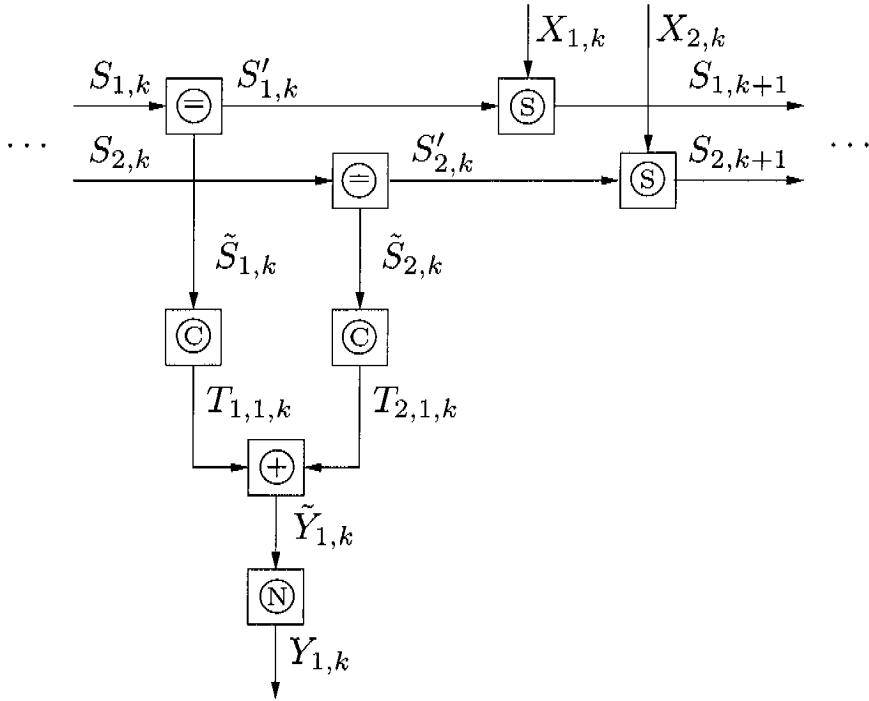


Figure 4.12: One section of the whole factor graph for the case of one channel and two sources.

4.6 Resolving Superpositions as a MAP Problem

Now that we have developed a factor graph, how can we use it to resolve superpositions or to decompose whole EMG signals? Let us consider a simplified one-channel version of the factor graph that is partly shown in Figure 4.11. This simpler factor graph is given in Figure 4.12.

Given the filter coefficients representing the MUAPs, decomposing a whole EMG signal or resolving single superpositions in an EMG signal means estimating the firing times of the sources. In other words, we need to estimate the binary source signals x given EMG signals y .

We can state this estimation problem as a classification problem⁴. Let us first assume that we wanted to classify a sample $x_{i,k}$ of a binary source signal into either of the two classes *no firing* (if $x_{i,k} = 0$) or *firing* (if $x_{i,k} = 1$) without actually seeing the EMG signals y . The best choice

⁴The classification problem mentioned here is different from the classification problem mentioned previously, where we assign MUAPs to motor unit classes.

would be to choose the class with the higher prior probability. The prior probabilities $p(x_{i,k})$ are the probabilities of belonging to each of the two classes *firing* and *no firing*. These correspond to the fraction of samples in each class, in the limit of an infinite number of observations. This approach minimizes the error of misclassification [13].

If our classifier is allowed to use the EMG signals y in addition to the prior probabilities of the classes to perform a classification, then the decision should no longer be based on the prior probabilities only, but also on the EMG signals y . How can we formally combine the prior probabilities $p(x_{i,k})$ with the additional information from the EMG signals y ?

The solution is the posterior probability $p(x_{i,k}|y)$. This is the probability that a sample belongs to a certain class given that we observe EMG signals y . When we assign a sample to the class having the largest posterior probability, we minimize the probability of misclassifying that sample. In this sense, the posterior probability $p(x_{i,k}|y)$ allows us to make optimal decisions regarding the class membership of new samples [13].

A formal way of stating what we have just explained shows the equation for symbol-wise maximum a-posteriori probability (MAP) estimation of the binary source variables $x_{i,k}$ [93]:

$$\hat{x}_{i,k}(y) = \operatorname{argmax}_{x_{i,k}} p(x_{i,k}|y). \quad (4.37)$$

This equation means that we choose the binary source signal sample $x_{i,k}$ so that it maximizes $p(x_{i,k}|y)$.

With the definition of conditional probability it follows:

$$\hat{x}_{i,k}(y) = \operatorname{argmax}_{x_{i,k}} \frac{p(x_{i,k}, y)}{p(y)}. \quad (4.38)$$

Since $p(y)$ is constant for different $x_{i,k}$, it does not change the $x_{i,k}$ for which $\frac{p(x_{i,k}, y)}{p(y)}$ becomes maximal. Therefore, it follows:

$$\hat{x}_{i,k}(y) = \operatorname{argmax}_{x_{i,k}} p(x_{i,k}, y). \quad (4.39)$$

Now, let us assume that the EMG signals y are given and fixed. $\hat{x}_{i,k}(y)$ is therefore no longer a function of y . Therefore, our problem of estimating a binary source signal sample $x_{i,k}$ can be summarized by the following

equation:

$$\hat{x}_{i,k} = \operatorname{argmax}_{x_{i,k}} p(x_{i,k}). \quad (4.40)$$

$p(x_{i,k})$ can be obtained by marginalizing $p(x, s, s', \tilde{s}, t, \tilde{y})$, where $p(x, s, s', \tilde{s}, t, \tilde{y})$ is the multivariate distribution describing our problem. Marginalization is performed by “summing out” all variables except $x_{i,k}$:

$$\hat{x}_{i,k} = \operatorname{argmax}_{x_{i,k}} \sum_{\sim x_{i,k}} p(x, s, s', \tilde{s}, t, \tilde{y}) \quad (4.41)$$

The essential part of this problem is the computation of the marginals

$$p(x_{i,k}) = \sum_{\sim x_{i,k}} p(x, s, s', \tilde{s}, t, \tilde{y}), \quad (4.42)$$

which can be calculated efficiently using message passing on the factor graph that is partly depicted in Figure 4.12. We will sketch next

- 4.6.1) The relationship between the multivariate distribution $p(x, s, s', \tilde{s}, t, \tilde{y})$ and the factor graph depicted in Figure 4.12 and
- 4.6.2) How message passing on a factor graph can be used to calculate marginals.

4.6.1 Factor Graph and Factorization

In general, a factor graph depicts a factorization of a function. In our case, the factor graph that is partly given in Figure 4.12 depicts a factorization of the multivariate distribution $p(x, s, s', \tilde{s}, t, \tilde{y})$ ⁵. The factorization of the multivariate distribution that is represented by the full factor graph is:

$$\begin{aligned} p(x, s, s', \tilde{s}, t, \tilde{y}) = & \prod_k f_=(s_{1,k}, s'_{1,k}, \tilde{s}_{1,k}) \cdot f_=(s_{2,k}, s'_{2,k}, \tilde{s}_{2,k}) \cdot \\ & f_S(s'_{1,k}, x_{1,k}, s_{1,k+1}) \cdot f_S(s'_{2,k}, x_{2,k}, s_{2,k+1}) \cdot \\ & f_C(\tilde{s}_{1,k}, t_{1,1,k}) \cdot f_C(\tilde{s}_{2,k}, t_{2,1,k}) \cdot \\ & f_+(t_{1,1,k}, t_{2,1,k}, \tilde{y}_{1,k}) \cdot f_N(\tilde{y}_{1,k}). \end{aligned} \quad (4.43)$$

⁵Remember that y is given and fixed.

In the context of factor graphs, the multivariate distribution $p(x, s, s', \tilde{s}, t, \tilde{y})$ is also called the global function and the factors, such as $f_i = (s_{1,k}, s'_{1,k}, \tilde{s}_{1,k})$, are called local functions.

4.6.2 Message-Passing Algorithm Using the Sum-Product Rule

Message-passing algorithms pass messages along the edges of a factor graph. In the case of the sum-product message-passing algorithm, the sum-product rule is used to calculate the messages. The resulting algorithm is also called the sum-product algorithm.

The sum-product algorithm applied to the factor graph in Figure 4.12 can be used to estimate marginals, i.e., to solve our classification problem:

$$\hat{x}_{i,k} = \operatorname{argmax}_{x_{i,k}} \underbrace{\sum_{\sim x_{i,k}} \underbrace{p(\text{"all variables"})}_{\substack{\text{Factor graph} \\ (\text{written as product})}}}_{\text{Sum}} \quad (4.44)$$

Sum-product algorithm

To show and explain how marginals can be calculated/estimated using message passing, we step back from our EMG signal decomposition problem in the next section, where we derive and explain the sum-product algorithm.

4.7 Introduction to the Sum-Product Algorithm

In this section we will derive the sum-product algorithm [90] and show how it can be used to efficiently calculate all marginals in a factor graph without cycles. To do this, we use the simple example from Section 4.4.1. There we had the following factorization of a discrete function:

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = f_A(x_1, x_3)f_B(x_2, x_4)f_C(x_3, x_4, x_5)f_D(x_5, x_6). \quad (4.45)$$

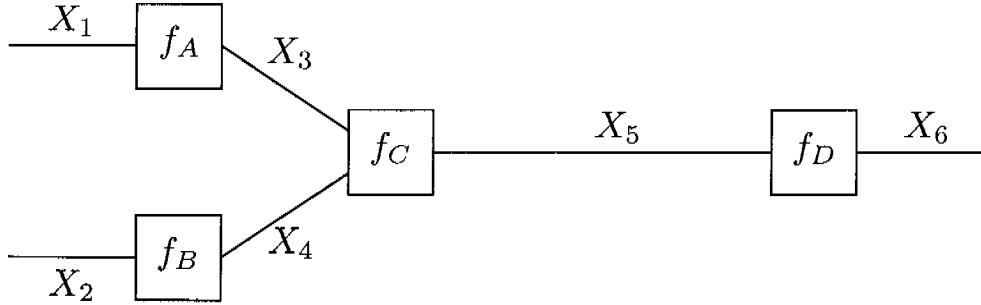


Figure 4.13: Example of a simple factor graph representing the factorization $f_A(x_1, x_3)f_B(x_2, x_4)f_C(x_3, x_4, x_5)f_D(x_5, x_6)$.

Let us now assume that $f(x_1, x_2, x_3, x_4, x_5, x_6)$ is a probability mass function. The factorization $f_A(x_1, x_3)f_B(x_2, x_4)f_C(x_3, x_4, x_5)f_D(x_5, x_6)$ is represented by the factor graph in Figure 4.13. We would like to calculate the marginal probability $p(x_5)$. This can be done by marginalization. For this we use the distributive law to simplify the summations. This means that we pull the factors that do not depend on the summation variable out of the sum so that we do not have to sum over them, e.g.,

$$\sum_a f(a)f(b) = f(b) \sum_a f(a). \quad (4.46)$$

This makes the calculation more efficient. For our example it follows:

$$f(x_5) = \sum_{x_1, x_2, x_3, x_4, x_6} f(x_1, x_2, x_3, x_4, x_5, x_6) \quad (4.47)$$

$$= \sum_{x_1} \sum_{x_2} \sum_{x_3} \sum_{x_4} \sum_{x_6} f_A(x_1, x_3) f_B(x_2, x_4) f_C(x_3, x_4, x_5) f_D(x_5, x_6) \quad (4.48)$$

$$= \sum_{x_3} \sum_{x_4} f_C(x_3, x_4, x_5) \sum_{x_1} f_A(x_1, x_3) \sum_{x_2} f_B(x_2, x_4) \sum_{x_6} f_D(x_5, x_6) \quad (4.49)$$

We will now show that this calculation can be interpreted in a factor graph.

First, we note that the factorization in

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = [f_A(x_1, x_3)f_B(x_2, x_4)f_C(x_3, x_4, x_5)] \cdot [f_D(x_5, x_6)] \quad (4.50)$$

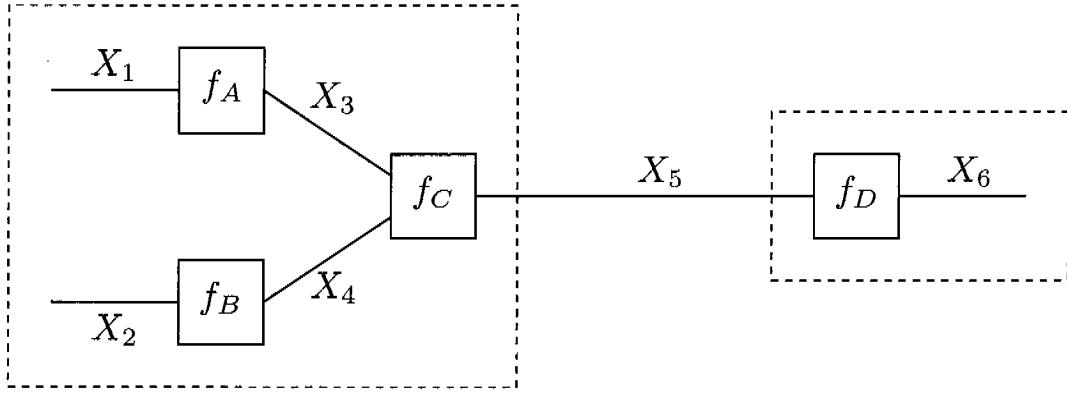


Figure 4.14: Factor graph depicting the factorization $[f_A(x_1, x_3)f_B(x_2, x_4)f_C(x_3, x_4, x_5)] \cdot [f_D(x_5, x_6)]$. Note that the dashed boxes correspond to the terms in brackets.

can be depicted by the factor graph in Figure 4.14.

Second, we see that some parts of (4.49) can be interpreted as messages in the factor graph, flowing along the edges of the factor graph. This is visualized in Figure 4.15 and (4.51).

$$p(x_5) = \sum_{x_3} \sum_{x_4} f_C(x_3, x_4, x_5) \underbrace{\sum_{x_1} f_A(x_1, x_3)}_{\mu_3} \underbrace{\sum_{x_2} f_B(x_2, x_4)}_{\mu_4} \underbrace{\sum_{x_6} f_D(x_5, x_6)}_{\mu_{5L}} \underbrace{\sum_{x_5} \mu_{5R}}$$

(4.51)

All that was done in (4.51) was the definition of names, i.e., μ_3 , μ_4 , μ_{5L} , and μ_{5R} . We then interpreted these objects as messages in the factor graph of Figure 4.15.

For example, the object μ_{5R} can be seen as a summary that is sent out of the left dashed box. Similarly, the object μ_{5L} can be seen as a summary that is sent out of the right dashed box.

From (4.51) it can easily be seen that

$$p(x_5) = \mu_{5R} \cdot \mu_{5L}. \quad (4.52)$$

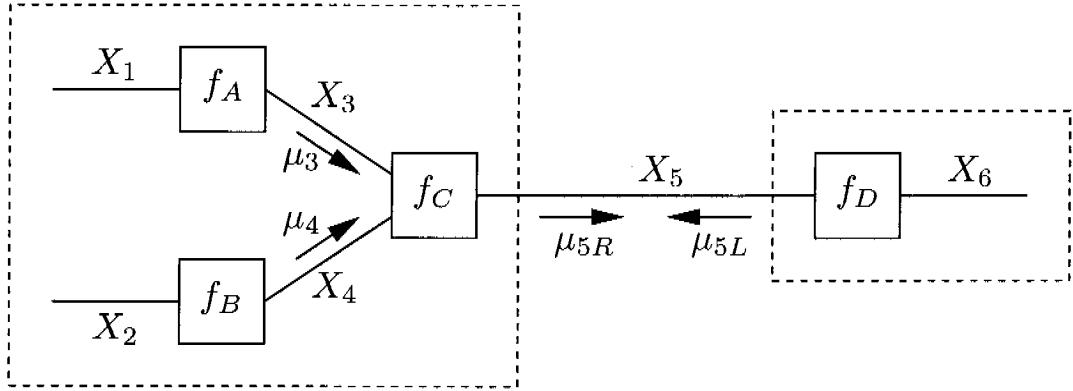


Figure 4.15: Factor graph with messages.

Therefore, multiplying the two messages that flow along the edge corresponding to x_5 yields the desired marginal $p(x_5)$.

Based on the previous example, we can derive a general rule for calculating messages that allows us to calculate the marginals. For example, consider the message μ_{5R} . From (4.51) we know that

$$\mu_{5R} = \sum_{x_3} \sum_{x_4} f_C(x_3, x_4, x_5) \cdot \mu_3 \cdot \mu_4. \quad (4.53)$$

Therefore, the general rule that we have just derived for calculating messages out of nodes, called the sum-product rule, states the following: an outgoing message, here μ_{5R} , is calculated by summing over the product of the node function from which the message is sent, here $f_C(x_3, x_4, x_5)$, and all incoming messages, here μ_3 and μ_4 .

In simple words, the outgoing message of a node depends on the node function and on the messages flowing into that node. Since we sum out all variables except the variable corresponding to the outgoing edge, here x_5 , the message depends only on the one remaining variable corresponding to the outgoing edge, here x_5 . We often write $\mu_{5R}(x_5)$ to make this explicit.

Having derived the general sum-product rule, it remains to be explained why we have identified the following three terms in (4.51) with the corresponding messages:

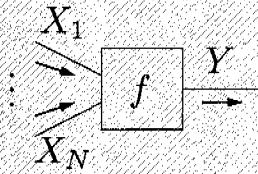
$$\mu_3 = \sum_{x_1} f_A(x_1, x_3) \quad (4.54)$$

$$\mu_4 = \sum_{x_2} f_B(x_2, x_4) \quad (4.55)$$

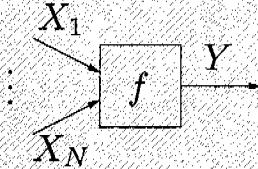
$$\mu_{5L} = \sum_{x_6} f_D(x_5, x_6) \quad (4.56)$$

This can be interpreted in two ways. One way is to assume that there are no incoming messages. This is consistent with the sum-product rule, i.e., we just do not have that message factor. The other way is to assume that the incoming messages from half edges are uniformly distributed. This means that the whole outgoing message (e.g., μ_3) is multiplied by a constant factor. This is, however not a problem, since messages can always be normalized so that the sum of all entries becomes one.

In fact, for computational reasons, messages are usually not normalized, i.e., their elements do not have to sum to one as in the case of probability distributions. However, we can still calculate marginals by first multiplying the two not-normalized messages along any edge in the factor graph, and then scaling the result so that the sum of the elements becomes one.

Sum-product rule (notation 1)


$$\mu_{f \rightarrow y}(y) \triangleq \sum_{x_1, \dots, x_N} f(y, x_1, \dots, x_N) \cdot \mu_{x_1 \rightarrow f}(x_1) \cdots \mu_{x_N \rightarrow f}(x_N) \quad (4.57)$$

Sum-product rule (notation 2)


Here, in this alternative notation, the arrow points to the right ($\overrightarrow{\mu}_Y(y)$) if the message is sent in the same direction as the arrow on the edge in the factor graph. Otherwise, the arrow would point to the left ($\overleftarrow{\mu}_Y(y)$).

$$\overrightarrow{\mu}_Y(y) \triangleq \mu_{f \rightarrow y}(y) \quad (4.58)$$

$$= \sum_{x_1, \dots, x_N} f(y, x_1, \dots, x_N) \cdot \overrightarrow{\mu}_{X_1}(x_1) \cdots \overrightarrow{\mu}_{X_N}(x_N) \quad (4.59)$$

Note that in this notation, an arrow on the edge in the factor graph does not give the actual direction in which the message propagates along the edge. It is only a reference arrow.

In summary, the following general rule applies to calculate the marginal $p(y)$ given the two messages along the edge associated with y :

$$p(y) = \gamma \cdot \vec{\mu}_Y(y) \cdot \overleftarrow{\mu}_Y(y), \quad (4.60)$$

where the normalizing factor γ is calculated as

$$\gamma = \frac{1}{\sum_y \vec{\mu}_Y(y) \cdot \overleftarrow{\mu}_Y(y)}. \quad (4.61)$$

The product of the two message is sometimes abbreviated by

$$\mu_{\text{tot}}(y) \triangleq \vec{\mu}_Y(y) \cdot \overleftarrow{\mu}_Y(y). \quad (4.62)$$

It follows

$$p(y) = \gamma \cdot \mu_{\text{tot}}(y). \quad (4.63)$$

The algorithm that uses the sum-product rule to calculate messages is called the sum-product algorithm [91, 92]. Applying this algorithm to cycle-free factor graphs allows to compute the correct marginals using (4.63).

Sum-product algorithm

The sum-product algorithm, which uses the sum-product rule to calculate messages, allows to simultaneously and efficiently compute all marginals in a factor graph without cycles.

Remark 4.2. (Factor graphs with cycles)

If a factor graph has cycles, then the sum-product algorithm can still be used by following the sum-product rule. This is due to the fact that the messages are computed locally. Because of the cycles in the factor graph, an iterative algorithm results. Messages may be passed multiple times on a given edge until some stop criterium is satisfied. Note that there is no natural termination when the factor graph has cycles (as it is the case for cycle-free graphs), nor is there a unique schedule for the order of message calculations. In this case, the algorithm might not give the correct marginals and it might even fail to converge. However, this sub-optimal approach allows very complex models (factor graphs) and is known to give excellent results in some important applications, such as the decoding of turbo codes or LDPC codes [77]. \square

4.8 Sum-Product Algorithm for Resolving Superpositions

Now we come back to the factor graph for EMG signal decomposition that is partly depicted in Figure 4.12. We have already stated in Section 4.6 that we can decompose an EMG signal by symbol-wise MAP estimation. As we have explained there, the binary source signal samples $x_{i,k}$ are estimated by the following equation [134]:

$$\hat{x}_{i,k} = \underbrace{\operatorname{argmax}_{x_{i,k}}}_{\text{Decision taking}} \left(\underbrace{\sum_{\sim x_{i,k}}}_{\text{Sum}} \underbrace{p(x, s, s', \tilde{s}, t, \tilde{y})}_{\substack{\text{Factor graph} \\ (\text{written as product})}} \right) .$$

$\underbrace{\quad\quad\quad}_{\text{Sum-product algorithm}}$

Decision about source signals based on symbol-wise decoding
(4.64)

As we have seen in the previous section, we can calculate the marginals

$$p(x_{i,k}) = \sum_{\sim x_{i,k}} p(x, s, s', \tilde{s}, t, \tilde{y}) \quad (4.65)$$

by using the sum-product algorithm. Therefore, the decomposition problem can be solved by passing messages along the edges of the factor graph that is partly depicted in Figure 4.16. The message passing is done according to some schedule⁶. The resulting messages $\overleftarrow{\mu}_{X_{i,k}}(x_{i,k})$ are then used to estimate the binary source variables $x_{i,k}$:

$$\begin{aligned} \hat{x}_{i,k} &= \operatorname{argmax}_{x_{i,k}} p(x_{i,k}) \\ &\approx \operatorname{argmax}_{x_{i,k}} \overleftarrow{\mu}_{X_{i,k}}(x_{i,k}) \end{aligned}$$

Note that the complete factor graph has cycles across different time slices, as can easily be seen in Figure 4.17. Extending the factor graph so that

⁶The details of this procedure are explained in Chapter 5.

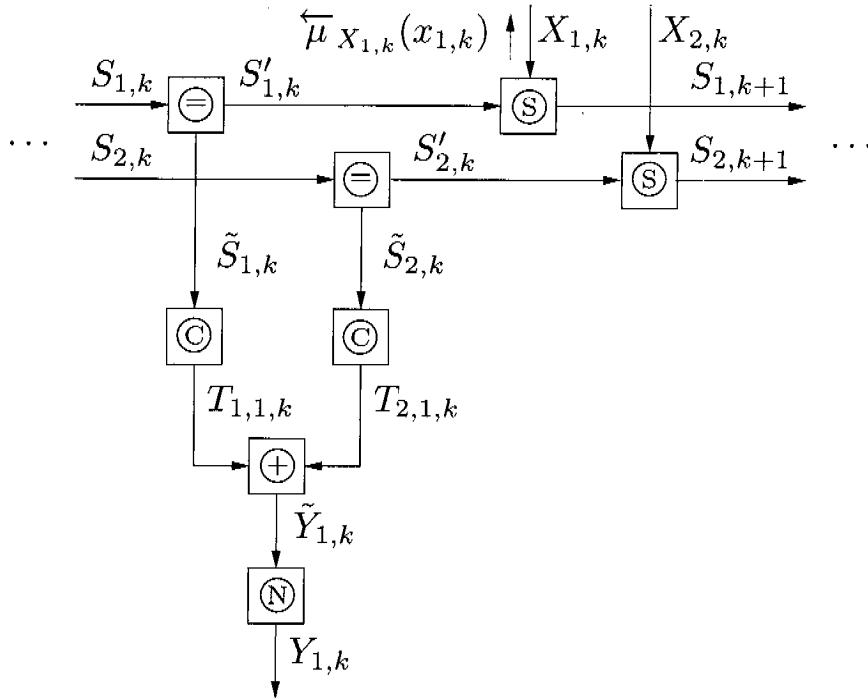


Figure 4.16: Factor graph with resulting message.

it can deal with two channels leads to cycles in individual time slices, as can be seen in Figure 4.18.

Because of the cycles, the resulting messages $\overleftarrow{\mu}_{X_{i,k}}(x_{i,k})$ can at best approximate the desired marginals $p(x_{i,k})$. If the factor graph had no cycles, then the sum-product algorithm would yield symbol-wise maximum a-posteriori (MAP) estimates. Since the factor graph has cycles, the algorithm yields only an approximation of these symbol-wise MAP estimates and may even fail to converge.

4.9 Symbol MAP Decoding Versus Block MAP Decoding

Instead of using a symbol MAP decoder as in (4.37), which does not maximize the global function of the full factor graph, another approach would be to find those binary source signals that maximize the global function. This can efficiently be achieved by using the Viterbi algorithm [49, 134], which performs block MAP decoding. The same result can

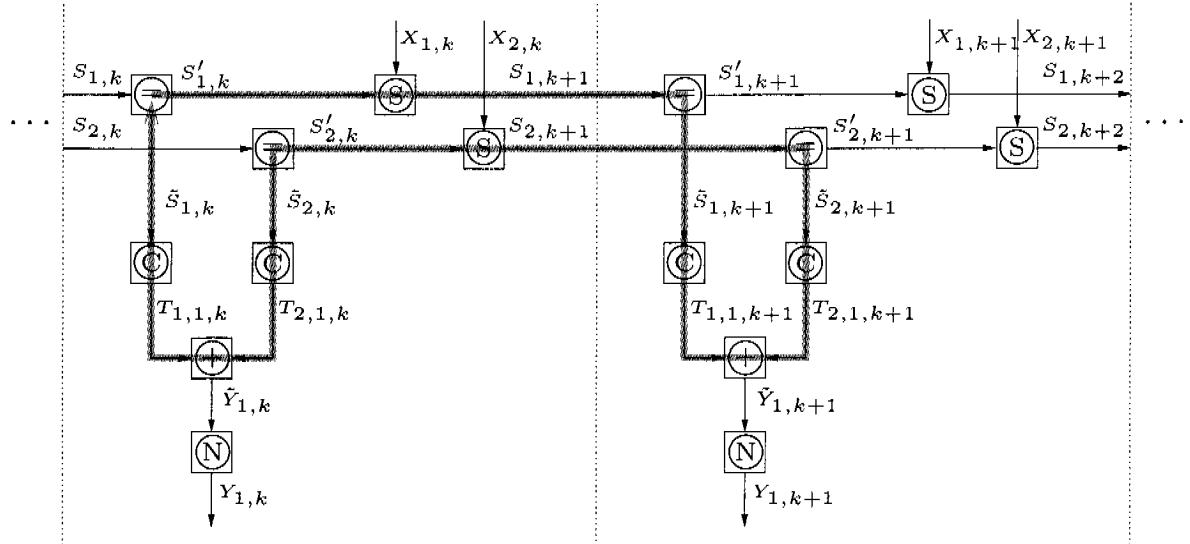
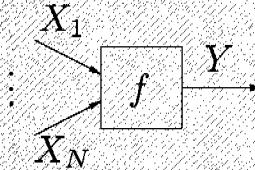


Figure 4.17: Cycle across two time slices for the case of only one channel.

be achieved by using the max-product algorithm⁷ [95, 134]. The max-product algorithm uses the max-product rule instead of the sum-product rule:

Max-product rule



$$\vec{\mu}_Y(y) \triangleq \max_{x_1, \dots, x_N} f(y, x_1, \dots, x_N) \cdot \vec{\mu}_{X_1}(x_1) \cdots \vec{\mu}_{X_N}(x_N) \quad (4.66)$$

However, the Viterbi algorithm is an optimal algorithm, which is computationally expensive [49]. It has also empirically been shown that the sum-product algorithm converges better on graphs with cycles than the

⁷If the global function is a Gaussian distribution, then the max-product algorithm is equivalent to the sum-product algorithm. In other words, in the case of Gaussian messages (see Section 5.1), the max-product algorithm is equivalent to the sum-product algorithm.

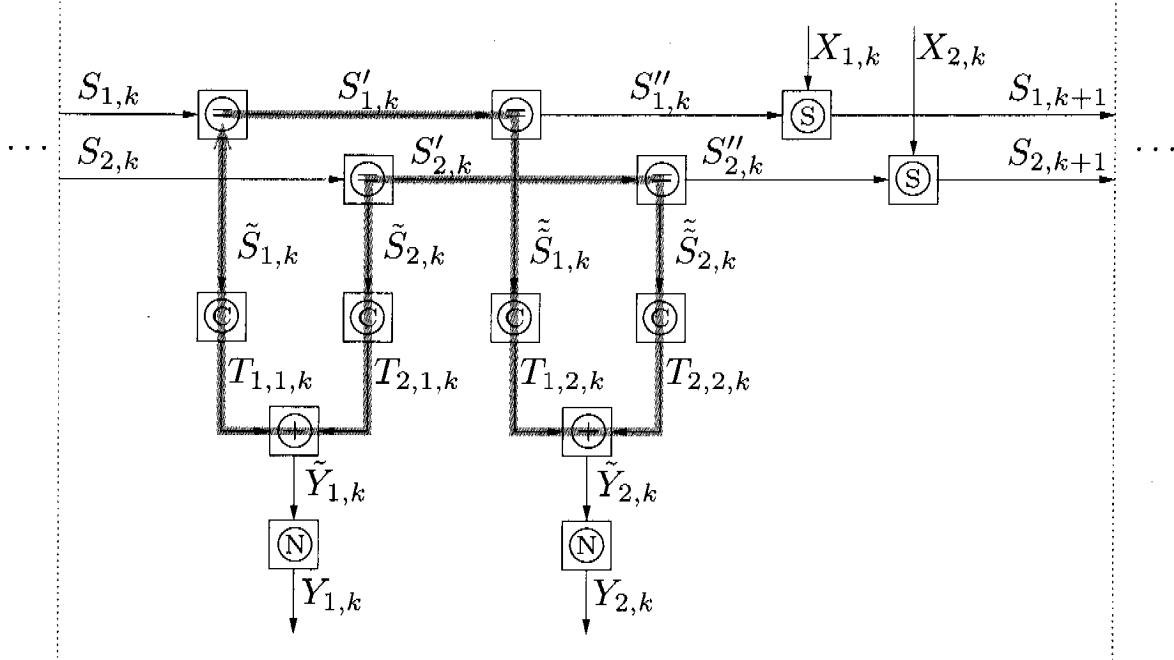


Figure 4.18: Cycle in one time slice for the case of two channels.

max-product algorithm. We have therefore used a symbol MAP decoding approach and the sum-product algorithm to estimate firing times.

4.10 Summary, Conclusions, and Outlook

Top-level development steps for iterative message-passing algorithms based on factor graphs

- 1) Draw a block diagram.
- 2) Derive a factor graph based on the block diagram.
- 3) Specify variable/message types.
- 4) Define node functions.
- 5) Derive message update rules.
- 6) Define a message-update schedule.

Running a message-passing algorithm on a factor graph

- 1) Initialize messages.
- 2) Run the message-passing algorithm according to the schedule until a pre-defined number of iterations or until some other stop criterion is reached.
 - For numeric stability, normalize or scale the messages after each calculation or iteration.

So far we have explained some factor graph theory and how a factor graph for EMG signal decomposition can be derived. We have also shown how message passing on a factor graph can solve the problem of calculating an approximation of a MAP estimate on a factor graph with cycles.

However, the details have still to be explained. For example, each node in the factor graph needs to be defined and the message update schedules have to be given. We will do this in the next chapter, where we present three different algorithms for resolving superpositions in EMG signals.

Chapter 5

Three New Algorithms for Resolving Superpositions

Thus my central theme is that complexity frequently takes the form of hierarchy and that hierarchic systems have some common properties independent of their specific content. Hierarchy, I shall argue, is one of the central structural schemes that the architect of complexity uses.

From *The Sciences of the Artificial* [125] by Herbert Simon

Based on the fundamentals introduced in the previous chapter, in this chapter we present and explain three novel algorithms for the resolution of superpositions in EMG signals in more detail. As before, we always use the sum-product rule [90] to calculate messages in factor graphs, and ultimately an approximation of the marginal conditional probability distributions of the binary source signals

$$X \triangleq (X_1, X_2, X_3, \dots), \quad (5.1)$$

with

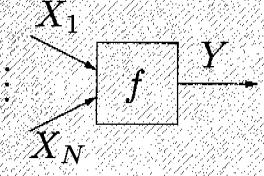
$$X_i = (X_{i,1}, X_{i,2}, X_{i,3}, \dots), \quad (5.2)$$

where index i represents the source number as before. The problem statement from Section 4.3 also holds for this chapter: all characteristic

MUAP shapes of all active motor units are known and constant. We also assume that MUAPs from the same motor unit do not overlap, i.e., we do not allow self-superpositions¹. The algorithms in this chapter basically differ in the topologies of the underlying factor graphs and in the message types used.

5.1 Message Types Used in This Chapter

Sum-product rule for discrete and continuous variables



Discrete messages: $X_1 \dots X_N$ and Y are discrete random variables. (See also Section 4.7.)

$$\overrightarrow{\mu}_Y(y) = \sum_{x_1} \dots \sum_{x_N} f(y, x_1, \dots, x_N) \cdot \overrightarrow{\mu}_{X_1}(x_1) \cdots \overrightarrow{\mu}_{X_N}(x_N) \quad (5.3)$$

Continuous messages: $X_1 \dots X_N$ and Y are continuous random variables.

$$\overrightarrow{\mu}_Y(y) = \int_{x_1} \dots \int_{x_N} f(y, x_1, \dots, x_N) \cdot \overrightarrow{\mu}_{X_1}(x_1) \cdots \overrightarrow{\mu}_{X_N}(x_N) dx_N \dots dx_1 \quad (5.4)$$

In Chapter 4 we have explained the sum-product rule for discrete variables (and messages), see (5.3). However, the sum-product rule has also a generalized continuous form for continuous random variables, see (5.4). In fact, the messages that would ideally be used in message-passing algorithms are often scaled continuous probability density functions, such as $\overrightarrow{\mu}_{X_1}(x_1)$ in (5.4).

¹Assuming that there are no self-superpositions is a reasonable assumption given the refractory periods and the band-pass filters that are used to filter the EMG signals.

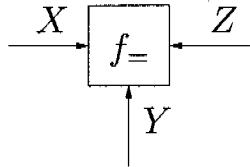


Figure 5.1: Equality constraint node.

5.1.1 Discrete Messages

Since such general continuous functions, such as $\vec{\mu}_Y(y)$ in (5.4), can not be stored in memory, they have to be represented/approximated in some way. For example, continuous messages may be approximated by discrete messages, which we can get by discretizing variables. In this case, the already known sum-product rule in (5.3) is applied to calculate messages. This approach is used in Section 5.2 to develop an algorithm for resolving superpositions in EMG signals.

Example 5.1. Consider the equality-constraint node in Figure 5.1. Let the variables X , Y , and Z be discrete, binary random variables taking on values 0 and 1. The node function is given by

$$f_=(x, y, z) = \delta[z - x]\delta[z - y], \quad (5.5)$$

where $\delta[.]$ represents the Kronecker delta function as introduced in Section 4.4.5. The message $\overleftarrow{\mu}_Z(z)$ can be calculated using the discrete version of the sum-product rule as given in (5.3):

$$\overleftarrow{\mu}_Z(z) = \sum_X \sum_Y \delta[z - x]\delta[z - y] \cdot \vec{\mu}_X(x) \cdot \vec{\mu}_Y(y) \quad (5.6)$$

$$= \vec{\mu}_X(z) \cdot \vec{\mu}_Y(z). \quad (5.7)$$

Let the incoming messages $\vec{\mu}_X(x)$ and $\vec{\mu}_Y(y)$ be defined as follows:

$$\vec{\mu}_X(0) = 0.9 \quad (5.8)$$

$$\vec{\mu}_X(1) = 0.1 \quad (5.9)$$

$$\vec{\mu}_Y(0) = 0.5 \quad (5.10)$$

$$\vec{\mu}_Y(1) = 0.5. \quad (5.11)$$

Then the outgoing message $\overleftarrow{\mu}_Z(z)$ is:

$$\overleftarrow{\mu}_Z(0) = \overrightarrow{\mu}_X(0) \cdot \overrightarrow{\mu}_Y(0) \quad (5.12)$$

$$= 0.9 \cdot 0.5 \quad (5.13)$$

$$= 0.45, \quad (5.14)$$

and

$$\overleftarrow{\mu}_Z(1) = \overrightarrow{\mu}_X(1) \cdot \overrightarrow{\mu}_Y(1) \quad (5.15)$$

$$= 0.1 \cdot 0.5 \quad (5.16)$$

$$= 0.05. \quad (5.17)$$

The message can be normalized so that the sum over all elements becomes one:

$$\overleftarrow{\mu}_Z(0) = 0.9 \quad (5.18)$$

$$\overleftarrow{\mu}_Z(0) = 0.1. \quad (5.19)$$

Note that this is the same distribution as $\overrightarrow{\mu}_X$, which could be expected since $\overrightarrow{\mu}_Y(y)$ is a uniform distribution. \square

5.1.2 Scalar-Gaussian Messages

We have just pointed out that continuous messages may be approximated by discrete message. Another way to represent continuous messages is by using functions that can be described by a few parameters. In this context we often use Gaussian density functions. The Gaussian distribution $\mathcal{N}(x | m_X, \sigma_X^2)$ of a random variable X with $x \in \mathbb{R}$, which is also called a “normal distribution”, is defined as:

$$\mathcal{N}(x | m_X, \sigma_X^2) \triangleq \frac{1}{\sqrt{2\pi}\sigma_X} \exp\left(-\frac{(x - m_X)^2}{2\sigma_X^2}\right), \quad (5.20)$$

where m_X is the mean and σ_X is the standard deviation. The variance is $V_X = \sigma_X^2$.

Parameterizing messages using Gaussian distributions has advantages. First, Gaussian distributions can be described by only two parameters, the mean and the variance. Second, when using Gaussian distributed input messages, the output message is often also a Gaussian density². In

²Whether the output message of a node with Gaussian distributed input messages is also a Gaussian density depends on the node function.

such a case, the message update calculations can be described in terms of the means and variances only, which makes the message updates very simple and computationally inexpensive.

Example 5.2. Now consider again the equality constraint node in Figure 5.1. This time, let the variables X , Y , and Z be continuous random variables with

$$\overrightarrow{\mu}_X(x) \propto \mathcal{N}\left(x | \overrightarrow{m_X}, \overrightarrow{V_X}\right) \quad (5.21)$$

$$= \frac{1}{\sqrt{2\pi\overrightarrow{V_X}}} \exp\left(-\frac{(x - \overrightarrow{m_X})^2}{2\overrightarrow{V_X}}\right) \quad (5.22)$$

$$\overrightarrow{\mu}_Y(y) \propto \mathcal{N}\left(y | \overrightarrow{m_Y}, \overrightarrow{V_Y}\right) \quad (5.23)$$

$$= \frac{1}{\sqrt{2\pi\overrightarrow{V_Y}}} \exp\left(-\frac{(y - \overrightarrow{m_Y})^2}{2\overrightarrow{V_Y}}\right). \quad (5.24)$$

Similar to the discrete case, the node function is given by

$$f_=(x, y, z) = \delta(z - x)\delta(z - y), \quad (5.25)$$

where $\delta(\cdot)$ represents the Dirac delta. The message $\overleftarrow{\mu}_Z(z)$ can be calculated using the continuous version of the sum-product rule as given in (5.4):

$$\begin{aligned} \overleftarrow{\mu}_Z(z) &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \delta(z - x)\delta(z - y) \cdot \overrightarrow{\mu}_X(x) \cdot \overrightarrow{\mu}_Y(y) dx dy \\ &= \overrightarrow{\mu}_X(z) \cdot \overrightarrow{\mu}_Y(z) \\ &= \frac{1}{\sqrt{2\pi\overrightarrow{V_X}}} \exp\left(-\frac{(z - \overrightarrow{m_X})^2}{2\overrightarrow{V_X}}\right) \cdot \frac{1}{\sqrt{2\pi\overrightarrow{V_Y}}} \exp\left(-\frac{(z - \overrightarrow{m_Y})^2}{2\overrightarrow{V_Y}}\right) \\ &\propto \exp\left(-z^2 \frac{1}{2\overrightarrow{V_X}} + z \frac{2\overrightarrow{m_X}}{2\overrightarrow{V_X}} - z^2 \frac{1}{2\overrightarrow{V_Y}} + z \frac{2\overrightarrow{m_Y}}{2\overrightarrow{V_Y}}\right) \\ &= \exp\left(-z^2 \left(\frac{1}{2\overrightarrow{V_X}} + \frac{1}{2\overrightarrow{V_Y}}\right) + z \left(\frac{2\overrightarrow{m_X}}{2\overrightarrow{V_X}} + \frac{2\overrightarrow{m_Y}}{2\overrightarrow{V_Y}}\right)\right) \quad (5.26) \end{aligned}$$

$$= \exp\left(-z^2 \frac{1}{2\overrightarrow{V_Z}} + z \frac{2\overleftarrow{m_Z}}{2\overrightarrow{V_Z}}\right) \quad (5.27)$$

$$\begin{aligned}
&\propto \frac{1}{\sqrt{2\pi\overleftarrow{V}_Z}} \exp\left(-z^2 \frac{1}{2\overleftarrow{V}_Z} + z \frac{2\overleftarrow{m}_Z}{2\overleftarrow{V}_Z}\right) \cdot \exp\left(-\frac{\overleftarrow{m}_Z^2}{2\overleftarrow{V}_Z}\right) \\
&= \frac{1}{\sqrt{2\pi\overleftarrow{V}_Z}} \exp\left(-z^2 \frac{1}{2\overleftarrow{V}_Z} + z \frac{2\overleftarrow{m}_Z}{2\overleftarrow{V}_Z} - \frac{\overleftarrow{m}_Z^2}{2\overleftarrow{V}_Z}\right) \\
&= \frac{1}{\sqrt{2\pi\overleftarrow{V}_Z}} \exp\left(-\frac{(z - \overleftarrow{m}_Z)^2}{2\overleftarrow{V}_Z}\right).
\end{aligned}$$

As it is the case for the input messages $\overrightarrow{\mu}_X(x)$ and $\overrightarrow{\mu}_Y(y)$, we see that the outgoing message $\overleftarrow{\mu}_Z(z)$ is also proportional to a Gaussian density function:

$$\overleftarrow{\mu}_Z(z) \propto \mathcal{N}(z | \overleftarrow{m}_Z, \overleftarrow{V}_Z). \quad (5.28)$$

Comparing coefficients of (5.26) and (5.27)

$$\frac{1}{2\overleftarrow{V}_Z} = \frac{1}{2\overrightarrow{V}_X} + \frac{1}{2\overrightarrow{V}_Y} \quad (5.29)$$

$$\frac{2\overleftarrow{m}_Z}{2\overleftarrow{V}_Z} = \frac{2\overrightarrow{m}_X}{2\overrightarrow{V}_X} + \frac{2\overrightarrow{m}_Y}{2\overrightarrow{V}_Y} \quad (5.30)$$

gives:

$$\overleftarrow{m}_Z = \frac{\overrightarrow{V}_X \overrightarrow{V}_Y}{\overrightarrow{V}_X + \overrightarrow{V}_Y} \left(\frac{\overrightarrow{m}_X}{\overrightarrow{V}_X} + \frac{\overrightarrow{m}_Y}{\overrightarrow{V}_Y} \right) \quad (5.31)$$

$$\overleftarrow{V}_Z = \frac{\overrightarrow{V}_X \overrightarrow{V}_Y}{\overrightarrow{V}_X + \overrightarrow{V}_Y}. \quad (5.32)$$

For example, if $\overrightarrow{V}_X = 0, \overrightarrow{V}_Y \neq 0$, this becomes

$$\overleftarrow{m}_Z = \overrightarrow{m}_X \quad (5.33)$$

$$\overleftarrow{V}_Z = 0. \quad (5.34)$$

□

In cases as in the previous example, each message can be represented by the two parameters mean and variance only. The actual messages that we send along the edges of the factor graph are therefore vectors with two

components for each message. The first component stores the mean and the second component stores the variance of the Gaussian distribution. Calculating the outgoing message is then based on these parameters only, as can be seen in (5.31) and (5.32). Because of this, the computational effort when calculating Gaussian messages is small. Refer to Appendix B for a table with update rules for standard building blocks. This table is actually made for the general case of multivariate-Gaussian messages, which we will cover in the next section.

A disadvantage of using Gaussian messages is that the described messages are not always Gaussian functions, in contrast to the previous example. In such cases, a parametrization using Gaussian messages is an approximation which may lead to sub-optimal results. Nevertheless, we use a combination of Gaussian messages and discrete messages in Section 5.3 for resolving superpositions in EMG signals.

5.1.3 Multivariate-Gaussian Messages

The last message type that we used for EMG signal decomposition are multivariate-Gaussian messages. With $\mathbf{x} \in \mathbb{R}^n$, the multivariate normal distribution is defined as

$$\mathcal{N}_W(\mathbf{x} | \mathbf{m}_x, \mathbf{W}_x) = \sqrt{\frac{|\mathbf{W}_x|}{(2\pi)^n}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{m}_x)^\top \mathbf{W}_x (\mathbf{x} - \mathbf{m}_x)\right), \quad (5.35)$$

where \mathbf{W}_x is called the weight matrix, $|\mathbf{W}_x|$ is the determinant of \mathbf{W}_x , \mathbf{m}_x is the mean vector of \mathbf{x} , and $n = \dim(\mathbf{x})$. If $\mathbf{V}_x = \mathbf{W}_x^{-1}$ exists, we can also write:

$$\mathcal{N}(\mathbf{x} | \mathbf{m}_x, \mathbf{V}_x) = \frac{1}{\sqrt{(2\pi)^n |\mathbf{V}_x|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{m}_x)^\top \mathbf{V}_x^{-1} (\mathbf{x} - \mathbf{m}_x)\right), \quad (5.36)$$

where \mathbf{V}_x is the covariance matrix. As for the scalar-Gaussian case, the messages that we send along the edges of a factor graph consist only of the parameters \mathbf{m}_x and \mathbf{W}_x or \mathbf{m}_x and \mathbf{V}_x . For many standard node functions, messages can be calculated by using simple update rules. Some are given in Table B.1.

We use a combination of multivariate-Gaussian and discrete messages in Section 5.4.

5.2 Resolving Superpositions Using Discrete Messages Only

In the previous section we have presented three different message types: discrete messages, scalar-Gaussian messages, and multivariate-Gaussian messages. These are the message types that we have used in our new EMG signal decomposition algorithms. There are other message types beyond these three types introduced here. Refer to [28] and [76] for more information on this topic.

In this section, we use discrete messages only, i.e., we assume that all variables are discrete. As an additional restriction, we allow only integer values. For a discrete random variable Y this means

$$y \in \mathbb{Z}, \quad (5.37)$$

where y is a value of the random variable Y . Therefore, a discrete message along the edge that corresponds to variable Y has the form

$$\mu_Y \in \mathbb{R}^n, \quad (5.38)$$

where n is the number of integers that Y can take on.

5.2.1 Factor Graph

Figure 5.2 shows one slice of the factor graph for EMG signal decomposition for the case where we use discrete messages³. Next we explain the variables, messages, and nodes in the graph in more detail.

5.2.2 Variables and Message Types

The variables in the factor graph determine the format of the messages that we send along its edges. For a list of symbols used here refer also to page 261.

³A very similar graph was already introduced in Section 4.5

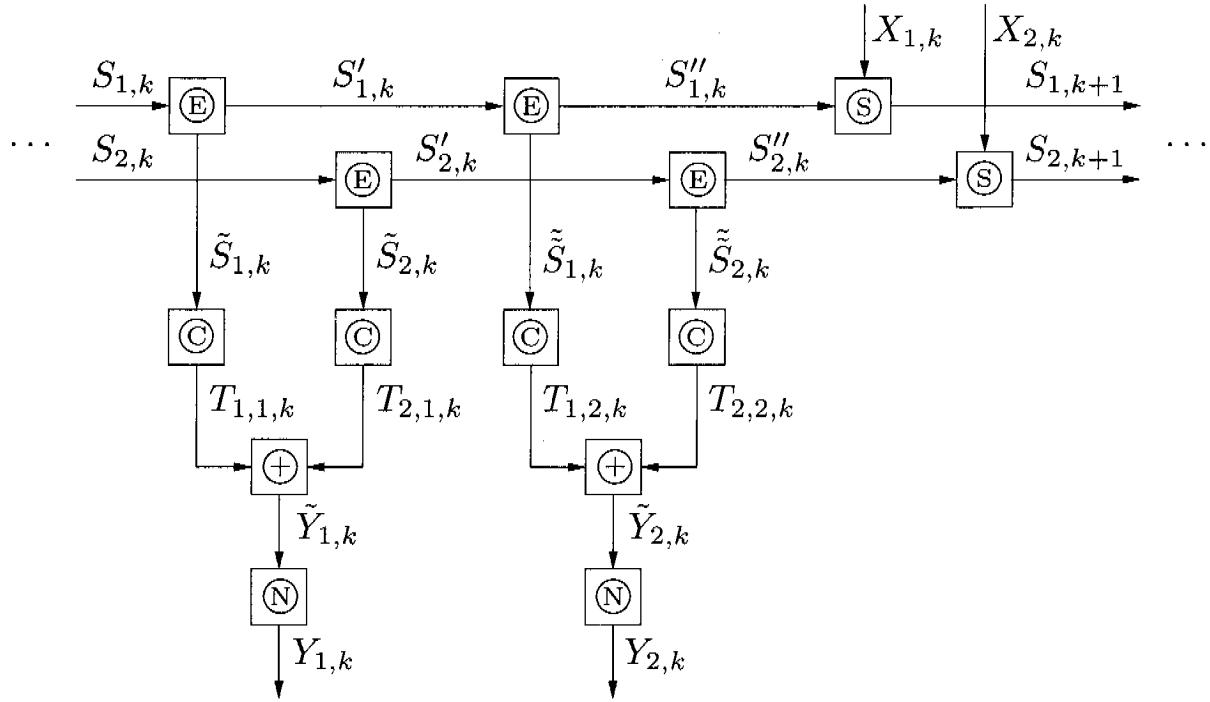


Figure 5.2: One section of the whole factor graph for the case where we use only discrete messages.

Variable $X_{i,k}$: The variable $X_{i,k}$ represents one sample of the source signal of source i at time k .

Message: A message along the edge corresponding to $X_{i,k}$ is a vector with two elements representing the discrete function $\overleftarrow{\mu}_{X_{i,k}}(x_{i,k})$. The first element of the vector holds the value $\overleftarrow{\mu}_{X_{i,k}}(x_{i,k} = 0)$ (no firing of source i at time k) and the second element of the vector holds the value $\overleftarrow{\mu}_{X_{i,k}}(x_{i,k} = 1)$ (firing of source i at time k).

Variables $S_{i,k}$: The variable $S_{i,k}$ represents the state of the finite state source model. We have used different source models. The general source model has N_a active states and N_i idle states. The active states code the state of the FIR filter producing a MUAP. In other words, an active state gives the number of the tap of the FIR filter. When in an idle state, there is no MUAP produced. The variables $S'_{i,k}$, and $S''_{i,k}$ are defined in the same way as $S_{i,k}$.

Message: A message along the edge corresponding to $S_{i,k}$ is a vector with $N_a + N_i$ elements.

Variable $\tilde{S}_{i,k}$: The variable $\tilde{S}_{i,k}$ represents also states. The active states correspond to the active states of $S_{i,k}$. In contrast to $S_{i,k}$, $\tilde{S}_{i,k}$ has only one idle state since the FIR filters do not need the information in which idle state the source model is.

Message: A message along the edge corresponding to $\tilde{S}_{i,k}$ is a vector with $N_a + 1$ elements.

Variable $T_{i,j,k}$: The variable $T_{i,j,k}$ represents an amplitude value of the motor unit action potential train of source i and channel j at time k . The variable can take on integer values in the interval $[T_{i,j,\min}, T_{i,j,\max}]$.

Message: A message along the edge corresponding to $T_{i,j,k}$ is a vector of length $T_{i,j,\max} - T_{i,j,\min} + 1$.

Variable $\tilde{Y}_{j,k}$: The variable $\tilde{Y}_{j,k}$ represents a sample of the discrete-time EMG signal without noise of channel j at time k .

Message: A message along the edge corresponding to $\tilde{Y}_{j,k}$ is a vector of appropriate length⁴.

Variable $Y_{j,k}$: The variable $Y_{j,k}$ represents the discrete-time EMG signal with noise of channel j at time k .

Message: A message along the edge corresponding to $Y_{j,k}$ stores the sample value $Y_{j,k} = \bar{y}_{j,k}$.

5.2.3 Node Functions and Message Update Rules

We now explain the nodes in Figure 5.2 and the corresponding message update rules. For this purpose, we use a simple example. At the beginning we will consider only one source, which produces only one MUAP. Given is an EMG signal, which is $L_S = 5$ samples long. It contains exactly one complete MUAP of duration $L_M = 3$ that we would like to find.

Figure 5.3 shows the possible locations of the MUAP in the signal. The numbers give the states of the FIR filter producing the MUAP. “I” stands

⁴See later sections.

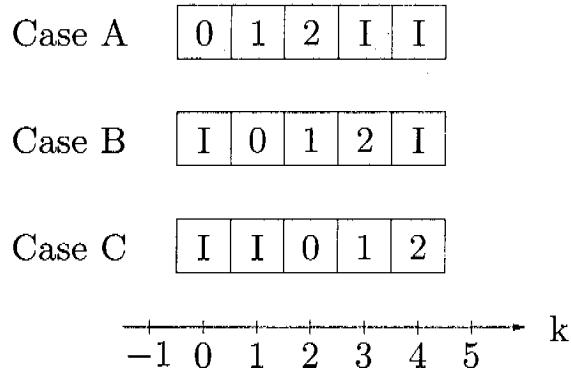


Figure 5.3: Possible locations of a MUAP in the signal. The numbers give the states of the FIR filter, which are also the sample numbers of the MUAP. “I” stands for an idle state.

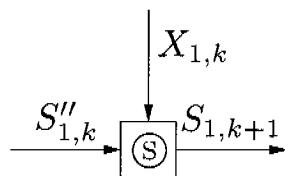


Figure 5.4: Source node.

for an idle state. Therefore, the active states code the taps of the filter. When in an idle state, there is no MUAP produced.

Source Nodes

The boxes labeled \circled{S} in Figure 5.2 are the source nodes. A source node is also shown in Figure 5.4. These nodes represent the state transition probabilities $p(s_{i,k+1}|s''_{i,k}, x_{i,k})$, which are defined by a finite state model. Therefore, the source nodes represent a probabilistic model such as the one in the example in Figure 4.5.

The corresponding state transition diagram is given in Figure 5.5. The active states correspond to the states of the FIR filter modeling the MUAP. The FIR filter coefficients associated with these active states, named here h_0 , h_1 , and h_2 , are also visualized in the figure.

At time $k = -1$, we are in state “S”. From there we go along one of the ways “A”, “B”, or “C” with equal probability. The way determines

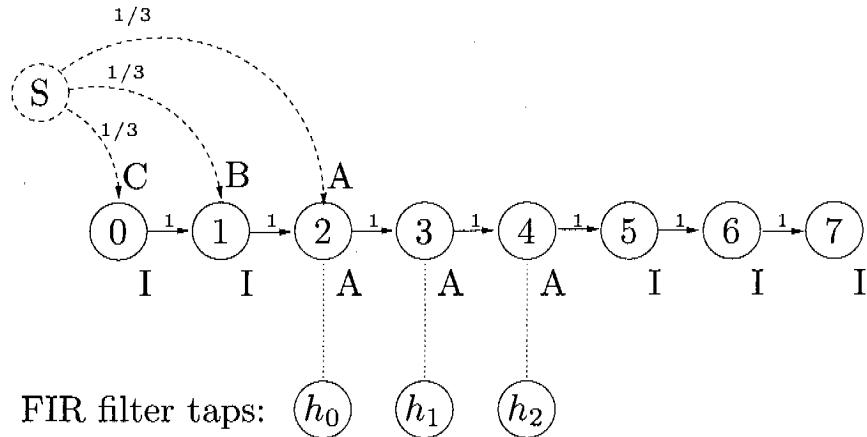


Figure 5.5: State transition diagram of the source model. “A” stands for an active state, “I” for an idle state.

when the source fires, e.g., it fires at $k = 0$ for way “A”. The source model has 2 idle states at the beginning (necessary for case “C”), then 3 active states, since the MUAP is 3 samples long. Finally, there are 3 more idle states on the right (necessary for case “A”)⁵.

Table 5.1 gives the state transition table, i.e., the state transition probabilities for all combinations of $s''_{1,k}$ and $s_{1,k+1}$. This table also defines the node function $p(s_{1,k+1}|s''_{1,k}, x_{1,k})$.

⁵On the right there are 2 states plus 1 state since the right terminal node, which we initialize as explained later, is at $k = 5$, i.e., one discrete time step beyond the last sample in the EMG signal.

$s''_{1,k}$	$s_{1,k+1}$	$x_{1,k}$	$p(s_{1,k+1} s''_{1,k}, x_{1,k})$
0	1	0	1
1	2	0	1
2	3	1	1
3	4	0	1
4	5	0	1
5	6	0	1
6	7	0	1
all other			0

Table 5.1: State transition table and definition of the node function.

The message $\vec{\mu}_{S_{1,k+1}}(s_{1,k+1})$ in Figure 5.2 can be calculated by using the discrete sum-product rule, the node function $p(s_{1,k+1}|s''_{1,k}, x_{1,k})$, and the incoming message $\vec{\mu}_{S''_{1,k}}(s''_{1,k})$ ⁶:

$$\vec{\mu}_{S_{1,k+1}}(s_{1,k+1}) = \sum_{s''_{1,k}} \sum_{x_{1,k}} p(s_{1,k+1}|s''_{1,k}, x_{1,k}) \vec{\mu}_{S''_{1,k}}(s''_{1,k}). \quad (5.39)$$

With $S''_{1,k} \in \{0, 1, \dots, 7\}$, $S_{1,k+1} \in \{0, 1, \dots, 7\}$, $X_{1,k} \in \{0, 1\}$, and Table 5.1 it follows, e.g., for $S_{1,k+1} = 1$:

$$\vec{\mu}_{S_{1,k+1}}(1) = \sum_{s''_{1,k}} \sum_{x_{1,k}} p(1|s''_{1,k}, x_{1,k}) \vec{\mu}_{S''_{1,k}}(s''_{1,k}) \quad (5.40)$$

$$= p(1|0, 0) \vec{\mu}_{S''_{1,k}}(0) \quad (5.41)$$

$$= \vec{\mu}_{S''_{1,k}}(0) \quad (5.42)$$

The message update rule can also be written as a matrix multiplication⁷:

$$\begin{pmatrix} \vec{\mu}_{S_{1,k+1}}(0) \\ \vec{\mu}_{S_{1,k+1}}(1) \\ \vec{\mu}_{S_{1,k+1}}(2) \\ \vec{\mu}_{S_{1,k+1}}(3) \\ \vec{\mu}_{S_{1,k+1}}(4) \\ \vec{\mu}_{S_{1,k+1}}(5) \\ \vec{\mu}_{S_{1,k+1}}(6) \\ \vec{\mu}_{S_{1,k+1}}(7) \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \vec{\mu}_{S''_{1,k}}(0) \\ \vec{\mu}_{S''_{1,k}}(1) \\ \vec{\mu}_{S''_{1,k}}(2) \\ \vec{\mu}_{S''_{1,k}}(3) \\ \vec{\mu}_{S''_{1,k}}(4) \\ \vec{\mu}_{S''_{1,k}}(5) \\ \vec{\mu}_{S''_{1,k}}(6) \\ \vec{\mu}_{S''_{1,k}}(7) \end{pmatrix}. \quad (5.43)$$

The message $\overleftarrow{\mu}_{S''_{1,k}}(s''_{1,k})$ in the opposite direction can be calculated similarly:

$$\overleftarrow{\mu}_{S''_{1,k}}(s''_{1,k}) = \sum_{s_{1,k+1}} \sum_{x_{1,k}} p(s''_{1,k+1}|s''_{1,k}, x_{1,k}) \overleftarrow{\mu}_{S_{1,k+1}}(s_{1,k+1}). \quad (5.44)$$

⁶The source is already modeled by the node function. Therefore, the edge corresponding to variable $X_{1,k}$ is only necessary for the final read-out. No incoming message $\vec{\mu}_{X_{1,k}}(x_{1,k})$ is needed. As a marginalia: Having no incoming message is equivalent to having an incoming message with a uniform distribution (for finite-alphabet variables). To see this, consider the sum-product rule in (5.3). A uniform message can be seen as a constant factor that can be pulled in front of the sums. Now remember that messages can be scaled arbitrarily.

⁷In practice, this message update is not implemented as an explicit matrix multiplication, which would be inefficient due to the many zero-entries in the matrix. The message update is rather implemented as a shift operation.

With $S''_{1,k} \in \{0, 1, \dots, 7\}$, $S_{1,k+1} \in \{0, 1, \dots, 7\}$, $X_{1,k} \in \{0, 1\}$, and Table 5.1 it follows, e.g., for $S''_{1,k} = 1$:

$$\overleftarrow{\mu}_{S''_{1,k}}(1) = \sum_{s_{1,k+1}} \sum_{x_{1,k}} p(s_{1,k+1}|1, x_{1,k}) \overleftarrow{\mu}_{S_{1,k+1}}(s_{1,k+1}) \quad (5.45)$$

$$= p(2|1, 0) \overleftarrow{\mu}_{S_{1,k+1}}(2) \quad (5.46)$$

$$= \overleftarrow{\mu}_{S_{1,k+1}}(2) \quad (5.47)$$

As before, the update rule can be written as a matrix multiplication:

$$\begin{pmatrix} \overleftarrow{\mu}_{S''_{1,k}}(0) \\ \overleftarrow{\mu}_{S''_{1,k}}(1) \\ \overleftarrow{\mu}_{S''_{1,k}}(2) \\ \overleftarrow{\mu}_{S''_{1,k}}(3) \\ \overleftarrow{\mu}_{S''_{1,k}}(4) \\ \overleftarrow{\mu}_{S''_{1,k}}(5) \\ \overleftarrow{\mu}_{S''_{1,k}}(6) \\ \overleftarrow{\mu}_{S''_{1,k}}(7) \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} \overleftarrow{\mu}_{S_{1,k+1}}(0) \\ \overleftarrow{\mu}_{S_{1,k+1}}(1) \\ \overleftarrow{\mu}_{S_{1,k+1}}(2) \\ \overleftarrow{\mu}_{S_{1,k+1}}(3) \\ \overleftarrow{\mu}_{S_{1,k+1}}(4) \\ \overleftarrow{\mu}_{S_{1,k+1}}(5) \\ \overleftarrow{\mu}_{S_{1,k+1}}(6) \\ \overleftarrow{\mu}_{S_{1,k+1}}(7) \end{pmatrix}. \quad (5.48)$$

Note that the transformation matrix from (5.43) is the transpose of the matrix from (5.48):

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}^T. \quad (5.49)$$

Also note that the state “S” in Figure 5.5 has not been mentioned in the update rules. We take care of this state by initializing the terminal nodes of the factor graph appropriately. Figure 5.6 shows the factor graph with terminal nodes, which are drawn in bold.

For the forward direction, where we update messages according to (5.43),

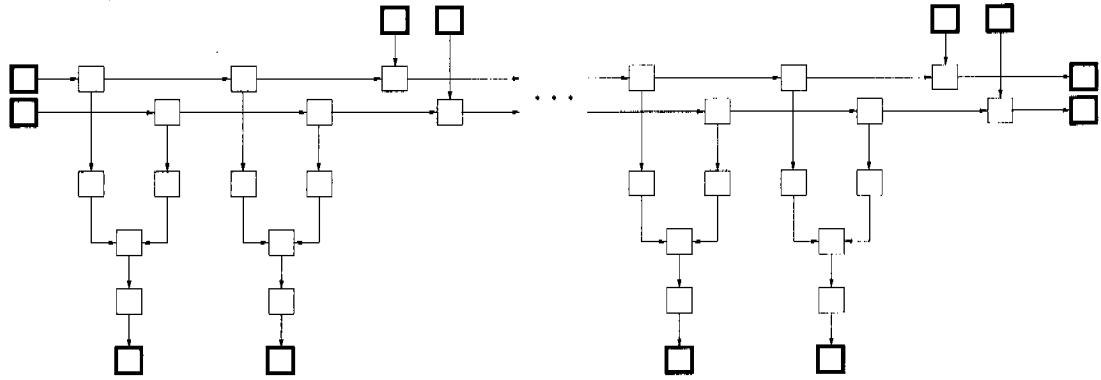


Figure 5.6: Factor graph with terminal nodes, which are necessary for the implementation of the factor graph using the `factorgraph` package.

the left terminal node is initialized so that its outgoing message is

$$\vec{\mu}_{S_{1,0}}(s_{1,0}) = \begin{pmatrix} \vec{\mu}_{S_{1,0}}(0) \\ \vec{\mu}_{S_{1,0}}(1) \\ \vec{\mu}_{S_{1,0}}(2) \\ \vec{\mu}_{S_{1,0}}(3) \\ \vec{\mu}_{S_{1,0}}(4) \\ \vec{\mu}_{S_{1,0}}(5) \\ \vec{\mu}_{S_{1,0}}(6) \\ \vec{\mu}_{S_{1,0}}(7) \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \quad (5.50)$$

This initialization allows the finite state model in Figure 5.5 to be in the state $S_{1,0} \in \{0, 1, 2\}$ at time $k = 0$. Given this initialization, the MUAP is equally likely to start at time $k = 0$ (if $S_{1,0} = 2$), at time $k = 1$ (if $S_{1,0} = 1$), or at time $k = 2$ (if $S_{1,0} = 0$). It cannot start earlier or later, since we expect the MUAP to lie completely within the signal. Therefore, the other elements of $\vec{\mu}_{S_{1,0}}(s_{1,0})$ are zero.

For the backward direction, where we update messages according to (5.48), the right terminal node is initialized so that its outgoing mes-

sage is

$$\overleftarrow{\mu}_{S_{1,5}}(s_{1,5}) = \begin{pmatrix} \overleftarrow{\mu}_{S_{1,5}}(0) \\ \overleftarrow{\mu}_{S_{1,5}}(1) \\ \overleftarrow{\mu}_{S_{1,5}}(2) \\ \overleftarrow{\mu}_{S_{1,5}}(3) \\ \overleftarrow{\mu}_{S_{1,5}}(4) \\ \overleftarrow{\mu}_{S_{1,5}}(5) \\ \overleftarrow{\mu}_{S_{1,5}}(6) \\ \overleftarrow{\mu}_{S_{1,5}}(7) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}. \quad (5.51)$$

Calculating the number of states^a

- Given length of the short EMG signal in samples: $L_S (= 5)$
- Given duration of the characteristic MUAP: $L_M (= 3)$
- Number of idle states, left: $N_{IL} = L_S - L_M (= 2)$
- Number of active states: $N_A = L_M (= 3)$
- Number of idle states, right: $N_{IR} = L_S - L_M + 1 (= 3)$
- Number of states: $N_S = 2 \cdot L_S - L_M + 1 (= 8)$

^aHere we assume that the MUAP is fully contained in the short EMG signal.

The source model presented here is only suitable for the special case where we have exactly one firing per source in the signal. However, the number of sources can be larger than one.

Finally, the message $\overleftarrow{\mu}_{X_{1,k}}(x_{1,k})$ are calculated by using the following update rule:

$$\overleftarrow{\mu}_{X_{1,k}}(x_{1,k}) = \sum_{s''_{1,k}} \sum_{s_{1,k+1}} p(s_{1,k+1}|s''_{1,k}, x_{1,k}) \overrightarrow{\mu}_{S''_{1,k}}(s''_{1,k}) \overleftarrow{\mu}_{S_{1,k+1}}(s_{1,k+1}). \quad (5.52)$$

Since $X_{1,k} \in \{0, 1\}$, each message $\overleftarrow{\mu}_{X_{1,k}}(x_{1,k})$ can be represented as a vector with two elements. A “firing” of source 1 means:

- $X_{1,k} = 1$;
- $S''_{1,k}$ encodes the first active state: $S''_{1,k} = 2$ in this example; and
- $S_{1,k+1}$ encodes the second active state: $S_{1,k+1} = 3$ in this example.

With this and Table 5.1, the message elements are calculated as follows:

$$\overleftarrow{\mu}_{X_{1,k}}(1) = \sum_{s''_{1,k}} \sum_{s_{1,k+1}} p(s_{1,k+1}|s''_{1,k}, x_{1,k}) \overrightarrow{\mu}_{S''_{1,k}}(s''_{1,k}) \overleftarrow{\mu}_{S_{1,k+1}}(s_{1,k+1}) \quad (5.53)$$

$$= p(3|2, 1) \overrightarrow{\mu}_{S''_{1,k}}(2) \overleftarrow{\mu}_{S_{1,k+1}}(3) \quad (5.54)$$

$$= \overrightarrow{\mu}_{S''_{1,k}}(2) \overleftarrow{\mu}_{S_{1,k+1}}(3) \quad (5.55)$$

$$\overleftarrow{\mu}_{X_{1,k}}(0) = \sum_{s''_{1,k}} \sum_{s_{1,k+1}} p(s_{1,k+1}|s''_{1,k}, x_{1,k}) \overrightarrow{\mu}_{S''_{1,k}}(s''_{1,k}) \overleftarrow{\mu}_{S_{1,k+1}}(s_{1,k+1}) \quad (5.56)$$

$$\begin{aligned} &= \overrightarrow{\mu}_{S''_{1,k}}(0) \overleftarrow{\mu}_{S_{1,k+1}}(1) + \\ &\quad \overrightarrow{\mu}_{S''_{1,k}}(1) \overleftarrow{\mu}_{S_{1,k+1}}(2) + \\ &\quad \overrightarrow{\mu}_{S''_{1,k}}(3) \overleftarrow{\mu}_{S_{1,k+1}}(4) + \\ &\quad \overrightarrow{\mu}_{S''_{1,k}}(4) \overleftarrow{\mu}_{S_{1,k+1}}(5) + \\ &\quad \overrightarrow{\mu}_{S''_{1,k}}(5) \overleftarrow{\mu}_{S_{1,k+1}}(6) + \\ &\quad \overrightarrow{\mu}_{S''_{1,k}}(6) \overleftarrow{\mu}_{S_{1,k+1}}(7) \end{aligned} \quad (5.57)$$

To estimate the source signal x_1 , we can use the approach explained in Section 4.8:

$$\hat{x}_1 = \operatorname{argmax}_{x_1} \overleftarrow{\mu}_{X_{1,k}}(x_{1,k}). \quad (5.58)$$

Refer to Section 5.5 for an alternative procedure that leads to better results in practice.

Extraction Nodes

Since the coefficient nodes \odot in Figure 5.2 do not need the information in which idle state the source model is, the idle states are combined into

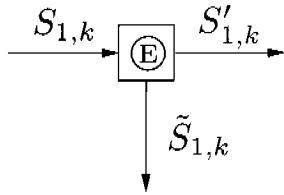


Figure 5.7: Extraction node.

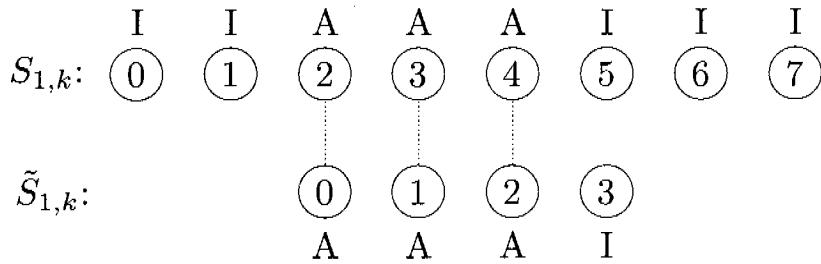


Figure 5.8: Source variable $S_{1,k}$ and FIR state variable $\tilde{S}_{1,k}$. “A” stands for an active state, “I” for an idle state.

one idle state⁸. This is done by the extraction nodes, which are denoted by \circled{E} in Figure 5.2. For example, let us consider the node in Figure 5.7. This node extract the information from the source variables $S_{1,k}$ and $S'_{1,k}$ for the coefficient node, which is supplied with the variable $\tilde{S}_{1,k}$.

Figure 5.8 shows the domain of the variables $S_{1,k}$ and $\tilde{S}_{1,k}$ as well as their relationship:

- The domains are: $S_{1,k} \in \{0, 1, 2, \dots, 7\}$ and $\tilde{S}_{1,k} \in \{0, 1, 2, \dots, 3\}$.
- The active states coded by $S_{1,k}$ correspond to the active states of $\tilde{S}_{1,k}$.
- The idle states coded by $S_{1,k}$ are combined into one idle state that is coded by $\tilde{S}_{1,k}$.

Table 5.2 defines the node function $f_{\circled{E}}(s_{1,k}, s'_{1,k}, \tilde{s}_{1,k})$ of the extraction node. With this node function and the incoming messages, the outgoing

⁸Here we assume that all characteristic MUAPs from the same source have the same durations in different channels, which can easily be achieved, e.g., by zero-padding.

message $\overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})$ in direction of the coefficient node can be computed:

$$\overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \sum_{s_{1,k}} \sum_{s'_{1,k}} f_{\textcircled{E}}(s_{1,k}, s'_{1,k}, \tilde{s}_{1,k}) \overrightarrow{\mu}_{S_{1,k}}(s_{1,k}) \overleftarrow{\mu}_{S'_{1,k}}(s'_{1,k}). \quad (5.59)$$

With $\tilde{S}_{1,k} \in \{0, 1, \dots, 3\}$ it follows, e.g., for $\tilde{S}_{1,k} = 0$:

$$\overrightarrow{\mu}_{\tilde{S}_{1,k}}(0) = \sum_{s_{1,k}} \sum_{s'_{1,k}} f_{\textcircled{E}}(s_{1,k}, s'_{1,k}, 0) \overrightarrow{\mu}_{S_{1,k}}(s_{1,k}) \overleftarrow{\mu}_{S'_{1,k}}(s'_{1,k}) \quad (5.60)$$

$$= f_{\textcircled{E}}(2, 2, 0) \overrightarrow{\mu}_{S_{1,k}}(2) \overleftarrow{\mu}_{S'_{1,k}}(2) \quad (5.61)$$

$$= \overrightarrow{\mu}_{S_{1,k}}(2) \overleftarrow{\mu}_{S'_{1,k}}(2). \quad (5.62)$$

We get the message values for the other active states of $\tilde{S}_{1,k}$ similarly:

$$\overrightarrow{\mu}_{\tilde{S}_{1,k}}(1) = \overrightarrow{\mu}_{S_{1,k}}(3) \overleftarrow{\mu}_{S'_{1,k}}(3) \quad (5.63)$$

$$\overrightarrow{\mu}_{\tilde{S}_{1,k}}(2) = \overrightarrow{\mu}_{S_{1,k}}(4) \overleftarrow{\mu}_{S'_{1,k}}(4). \quad (5.64)$$

And the message value for the idle state $\tilde{S}_{1,k} = 3$ is:

$$\begin{aligned} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(3) &= \sum_{s_{1,k}} \sum_{s'_{1,k}} f_{\textcircled{E}}(s_{1,k}, s'_{1,k}, 3) \overrightarrow{\mu}_{S_{1,k}}(s_{1,k}) \overleftarrow{\mu}_{S'_{1,k}}(s'_{1,k}) \quad (5.65) \\ &= \overrightarrow{\mu}_{S_{1,k}}(0) \overleftarrow{\mu}_{S'_{1,k}}(0) + \\ &\quad \overrightarrow{\mu}_{S_{1,k}}(1) \overleftarrow{\mu}_{S'_{1,k}}(1) + \end{aligned}$$

$s_{1,k}$	$s'_{1,k}$	$\tilde{s}_{1,k}$	$f_{\textcircled{E}}(s_{1,k}, s'_{1,k}, \tilde{s}_{1,k})$
0	0	3	1
1	1	3	1
2	2	0	1
3	3	1	1
4	4	2	1
5	5	3	1
6	6	3	1
7	7	3	1
all other			0

Table 5.2: Definition of the node function of the extraction node.

$$\begin{aligned} & \overrightarrow{\mu}_{S_{1,k}}(5) \overleftarrow{\mu}_{S'_{1,k}}(5) + \\ & \overrightarrow{\mu}_{S_{1,k}}(6) \overleftarrow{\mu}_{S'_{1,k}}(6) + \\ & \overrightarrow{\mu}_{S_{1,k}}(7) \overleftarrow{\mu}_{S'_{1,k}}(7). \end{aligned} \quad (5.66)$$

The message $\overrightarrow{\mu}_{S'_{1,k}}(s'_{1,k})$ is computed as follows:

$$\overrightarrow{\mu}_{S'_{1,k}}(s'_{1,k}) = \sum_{s_{1,k}} \sum_{\tilde{s}_{1,k}} f_{\textcircled{E}}(s_{1,k}, s'_{1,k}, \tilde{s}_{1,k}) \overrightarrow{\mu}_{S_{1,k}}(s_{1,k}) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \quad (5.67)$$

With $S'_{1,k}(s'_{1,k}) \in \{0, 1, \dots, 7\}$ it follows, e.g., for $S'_{1,k} = 0$:

$$\begin{aligned} \overrightarrow{\mu}_{S'_{1,k}}(0) &= \sum_{s_{1,k}} \sum_{\tilde{s}_{1,k}} f_{\textcircled{E}}(s_{1,k}, 0, \tilde{s}_{1,k}) \overrightarrow{\mu}_{S_{1,k}}(s_{1,k}) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \end{aligned} \quad (5.68)$$

$$= f_{\textcircled{E}}(0, 0, 3) \overrightarrow{\mu}_{S_{1,k}}(0) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(3) \quad (5.69)$$

$$= \overrightarrow{\mu}_{S_{1,k}}(0) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(3). \quad (5.70)$$

The other message values are computed accordingly:

$$\overrightarrow{\mu}_{S'_{1,k}}(1) = \overrightarrow{\mu}_{S_{1,k}}(1) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(3) \quad (5.71)$$

$$\overrightarrow{\mu}_{S'_{1,k}}(2) = \overrightarrow{\mu}_{S_{1,k}}(2) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(0) \quad (5.72)$$

$$\overrightarrow{\mu}_{S'_{1,k}}(3) = \overrightarrow{\mu}_{S_{1,k}}(3) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(1) \quad (5.73)$$

$$\overrightarrow{\mu}_{S'_{1,k}}(4) = \overrightarrow{\mu}_{S_{1,k}}(4) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(2) \quad (5.74)$$

$$\overrightarrow{\mu}_{S'_{1,k}}(5) = \overrightarrow{\mu}_{S_{1,k}}(5) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(3) \quad (5.75)$$

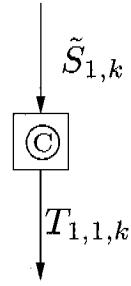
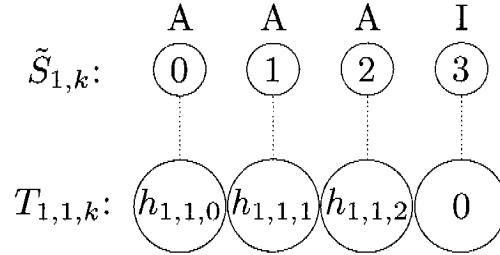
$$\overrightarrow{\mu}_{S'_{1,k}}(6) = \overrightarrow{\mu}_{S_{1,k}}(6) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(3) \quad (5.76)$$

$$\overrightarrow{\mu}_{S'_{1,k}}(7) = \overrightarrow{\mu}_{S_{1,k}}(7) \overleftarrow{\mu}_{\tilde{S}_{1,k}}(3) \quad (5.77)$$

The message $\overleftarrow{\mu}_{S_{1,k}}(s_{1,k})$ is computed similarly.

Coefficient Nodes

The coefficient nodes, which are denoted by \textcircled{C} in Figure 5.2, translate the states into amplitude values and vice versa. For example, let us consider the node in Figure 5.9. The relationship between the variables $\tilde{S}_{1,k}$ and $T_{1,1,k}$ is visualized in Figure 5.10 (compare also to Figure 5.5 and Figure 5.8). It can also be described by the following equation:

**Figure 5.9:** Coefficient node.**Figure 5.10:** FIR state variable $\tilde{S}_{1,k}$ and FIR coefficients $T_{1,1,k}$. “A” stands for an active state, “I” for an idle state.

$$T_{1,1,k} = \begin{cases} h_{1,1,\tilde{S}_{1,k}} & \text{if } \tilde{S}_{1,k} \in \{0, 1, 2\} \\ 0 & \text{if } \tilde{S}_{1,k} = 3 \end{cases}$$

Therefore, the node function for the node in Figure 5.9 can be stated as:

$$f_{\textcircled{C}}(\tilde{s}_{1,k}, t_{1,1,k}) = \begin{cases} \delta[t_{1,1,k} - h_{1,1,\tilde{S}_{1,k}}] & \text{if } \tilde{S}_{1,k} \in \{0, 1, 2\} \\ \delta[t_{1,1,k}] & \text{if } \tilde{S}_{1,k} = 3 \end{cases}$$

With this node function and the incoming message $\vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})$, the outgoing message $\vec{\mu}_{T_{1,1,k}}(t_{1,1,k})$ in direction of the coefficient node can be computed:

$$\vec{\mu}_{T_{1,1,k}}(t_{1,1,k}) = \sum_{\tilde{s}_{1,k}} f_{\textcircled{C}}(\tilde{s}_{1,k}, t_{1,1,k}) \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \quad (5.78)$$

$$\begin{aligned}
 &= \sum_{\substack{\text{active states } \tilde{S}_{1,k}}} \delta[t_{1,1,k} - h_{1,1,\tilde{S}_{1,k}}] \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) + \\
 &\quad \sum_{\substack{\text{idle state } \tilde{S}_{1,k}}} \delta[t_{1,1,k}] \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \quad (5.79) \\
 &= \delta[t_{1,1,k} - h_{1,1,0}] \vec{\mu}_{\tilde{S}_{1,k}}(0) +
 \end{aligned}$$

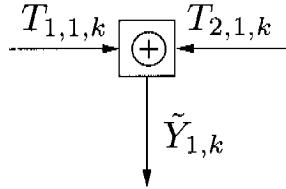


Figure 5.11: Sum constraint node.

$$\begin{aligned} & \delta[t_{1,1,k} - h_{1,1,1}] \overrightarrow{\mu}_{\tilde{S}_{1,k}}(1) + \\ & \delta[t_{1,1,k} - h_{1,1,2}] \overrightarrow{\mu}_{\tilde{S}_{1,k}}(2) + \\ & \delta[t_{1,1,k}] \overrightarrow{\mu}_{\tilde{S}_{1,k}}(3) \end{aligned} \quad (5.80)$$

The message in the opposite direction can be computed similarly:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \sum_{t_{1,1,k}} f_{\odot}(\tilde{s}_{1,k}, t_{1,1,k}) \overleftarrow{\mu}_{T_{1,1,k}}(t_{1,1,k}) \quad (5.81)$$

For the active states $\tilde{S}_{1,k} \in \{0, 1, 2\}$ it follows:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \sum_{t_{1,1,k}} \delta[t_{1,1,k} - h_{1,1,\tilde{S}_{1,k}}] \overleftarrow{\mu}_{T_{1,1,k}}(t_{1,1,k}) \quad (5.82)$$

$$= \overleftarrow{\mu}_{T_{1,1,k}}(h_{1,1,k}) \quad (5.83)$$

and for the idle state $\tilde{S}_{1,k} = 3$:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \overleftarrow{\mu}_{T_{1,1,k}}(0) \quad (5.84)$$

Sum Constraint Nodes

The sum constraint nodes, which are denoted by \oplus in Figure 5.2, combine the information from the sources. For example, let us consider the node in Figure 5.11 with the relationship

$$\tilde{y}_{1,k} = t_{1,1,k} + t_{2,1,k}. \quad (5.85)$$

As already introduced in Section 4.4.4, the node function is:

$$f_{\oplus}(t_{1,1,k}, t_{2,1,k}, \tilde{y}_{1,k}) = \delta[\tilde{y}_{1,k} - t_{1,1,k} - t_{2,1,k}]. \quad (5.86)$$

The variable $T_{1,1,k}$ represents an amplitude value of the motor unit action potential train of source 1 and channel 1 at time k . The variable can take on integer values in the interval $[T_{1,1,\min}, T_{1,1,\max}]$. Therefore, a message $\vec{\mu}_{T_{1,1,k}}(t_{1,1,k})$ is a vector of length $T_{1,1,\max} - T_{1,1,\min} + 1$.

Similarly for the variable $T_{2,1,k}$, which, in contrast to $T_{1,1,k}$, represents an amplitude value of the motor unit action potential train of source 2. The variable can take on integer values in the interval $[T_{2,1,\min}, T_{2,1,\max}]$. Therefore, a message $\vec{\mu}_{T_{2,1,k}}(t_{2,1,k})$ is a vector of length $T_{2,1,\max} - T_{2,1,\min} + 1$.

The variable $\tilde{Y}_{1,k}$ represents the discrete-time EMG signal without noise of channel 1 at time k . Its domain are all integer values in the interval $[T_{1,1,\min} + T_{2,1,\min}, T_{1,1,\max} + T_{2,1,\max}]$.

Using the discrete sum-product rule, the message $\vec{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k})$ becomes a convolution of the two incoming messages $\vec{\mu}_{T_{1,1,k}}(t_{1,1,k})$ and $\vec{\mu}_{T_{2,1,k}}(t_{2,1,k})$:

$$\begin{aligned}\vec{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k}) &= \sum_{t_{1,1,k}} \sum_{t_{2,1,k}} \delta[\tilde{y}_{1,k} - t_{1,1,k} - t_{2,1,k}] \\ &\quad \cdot \vec{\mu}_{T_{1,1,k}}(t_{1,1,k}) \cdot \vec{\mu}_{T_{2,1,k}}(t_{2,1,k}) \quad (5.87) \\ &= \sum_{t_{2,1,k}} \vec{\mu}_{T_{1,1,k}}(\tilde{y}_{1,k} - t_{2,1,k}) \cdot \vec{\mu}_{T_{2,1,k}}(t_{2,1,k}).\end{aligned}\quad (5.88)$$

The update rules for $\overleftarrow{\mu}_{T_{1,1,k}}(t_{1,1,k})$ and $\overleftarrow{\mu}_{T_{2,1,k}}(t_{2,1,k})$ are derived accordingly:

$$\begin{aligned}\overleftarrow{\mu}_{T_{1,1,k}}(t_{1,1,k}) &= \sum_{\tilde{y}_{1,k}} \sum_{t_{2,1,k}} \delta[\tilde{y}_{1,k} - t_{1,1,k} - t_{2,1,k}] \\ &\quad \cdot \overleftarrow{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k}) \cdot \vec{\mu}_{T_{2,1,k}}(t_{2,1,k}) \quad (5.89)\end{aligned}$$

$$= \sum_{t_{2,1,k}} \overleftarrow{\mu}_{\tilde{Y}_{1,k}}(t_{1,1,k} + t_{2,1,k}) \cdot \vec{\mu}_{T_{2,1,k}}(t_{2,1,k}) \quad (5.90)$$

and

$$\begin{aligned}\overleftarrow{\mu}_{T_{2,1,k}}(t_{2,1,k}) &= \sum_{\tilde{y}_{1,k}} \sum_{t_{1,1,k}} \delta[\tilde{y}_{1,k} - t_{1,1,k} - t_{2,1,k}] \\ &\quad \cdot \overleftarrow{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k}) \cdot \vec{\mu}_{T_{1,1,k}}(t_{1,1,k}) \quad (5.91)\end{aligned}$$

$$= \sum_{t_{1,1,k}} \overleftarrow{\mu}_{\tilde{Y}_{1,k}}(t_{1,1,k} + t_{2,1,k}) \cdot \vec{\mu}_{T_{1,1,k}}(t_{1,1,k}). \quad (5.92)$$

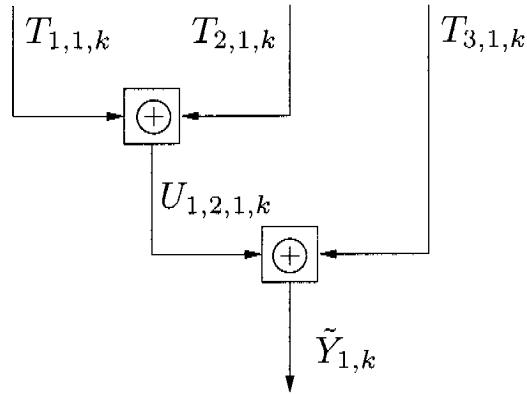


Figure 5.12: Two sum constraint nodes for three sources.

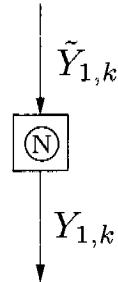


Figure 5.13: Noise node.

Finally, let us consider the general case of N sources. In this case we use multiple sum constraint nodes each having three edges, like the one in Figure 5.11. The factor graph part containing the sum constraint nodes might look as in Figure 5.12. The factor graph topology therefore depends on the number of sources. Since the number of sources can vary, our algorithm has to be able to deal with a variable number of sources, for which the factor graph has to be constructed automatically.

Noise Nodes

The noise nodes, which are denoted by \textcircled{N} in Figure 5.2, model the amplitude noise in the EMG sample values. For example, let us consider the node in Figure 5.13. This node can be seen as a compound node, which is given in Figure 5.14. Here,

$$\tilde{Y}_1 \triangleq (\tilde{Y}_{1,1}, \tilde{Y}_{1,2}, \tilde{Y}_{1,3}, \dots) \quad (5.93)$$

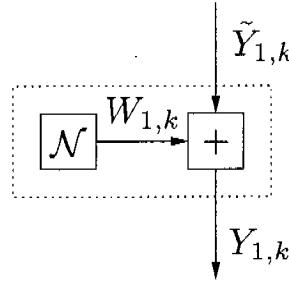


Figure 5.14: Noise node in detail.

is the discrete-time EMG signal without noise of channel 1,

$$Y_1 \triangleq (Y_{1,1}, Y_{1,2}, Y_{1,3}, \dots) \quad (5.94)$$

is the discrete-time EMG signal of channel 1, and

$$W_1 \triangleq (W_{1,1}, W_{1,2}, W_{1,3}, \dots) \quad (5.95)$$

is zero-mean an i.i.d. additive white Gaussian noise of channel 1, i.e.,

$$W_{1,k} \sim \mathcal{N}(w_{1,k} \mid 0, \sigma_1^2), \quad (5.96)$$

where σ_1 is the standard deviation for this channel.

We usually need to calculate only the message $\overleftarrow{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k})$ and not $\overrightarrow{\mu}_{Y_{1,k}}(y_{1,k})$. We can derive the update rule for $\overleftarrow{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k})$ by using the standard sum-product rule:

$$\overleftarrow{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k}) = \sum_{y_{1,k}} \sum_{w_{1,k}} \delta[y_{1,k} - \tilde{y}_{1,k} - w_{1,k}] \cdot \overleftarrow{\mu}_{Y_{1,k}}(y_{1,k}) \cdot \overrightarrow{\mu}_{W_{1,k}}(w_{1,k}). \quad (5.97)$$

Since the message $\overleftarrow{\mu}_{Y_{1,k}}(y_{1,k})$ represents a fixed sample value $y_{1,k} = \bar{y}_{1,k}$, where $\bar{y}_{1,k}$ is a fixed value, it can be written as

$$\overleftarrow{\mu}_{Y_{1,k}}(y_{1,k}) = \delta[y_{1,k} - \bar{y}_{1,k}]. \quad (5.98)$$

Since the node marked with \mathcal{N} is a terminal node, the message $\overrightarrow{\mu}_{W_{1,k}}(w_{1,k})$ represents the distribution of the random variable $W_{1,k}$, which we defined to be Gaussian distributed:

$$\overrightarrow{\mu}_{W_{1,k}}(w_{1,k}) = \frac{1}{\sqrt{2\pi}\sigma_1} \cdot \exp\left(-\frac{w_{1,k}^2}{2\sigma_1^2}\right). \quad (5.99)$$

With this, (5.97) becomes

$$\overleftarrow{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k}) = \sum_{y_{1,k}} \sum_{w_{1,k}} \delta[y_{1,k} - \tilde{y}_{1,k} - w_{1,k}] \cdot \overleftarrow{\mu}_{Y_{1,k}}(y_{1,k}) \cdot \overrightarrow{\mu}_{W_{1,k}}(w_{1,k}) \quad (5.100)$$

$$= \sum_{y_{1,k}} \sum_{w_{1,k}} \delta[y_{1,k} - \tilde{y}_{1,k} - w_{1,k}] \cdot \delta[y_{1,k} - \bar{y}_{1,k}] \cdot \frac{1}{\sqrt{2\pi\sigma_1}} \cdot \exp\left(-\frac{w_{1,k}^2}{2\sigma_1^2}\right) \quad (5.101)$$

$$= \sum_{w_{1,k}} \delta[\bar{y}_{1,k} - \tilde{y}_{1,k} - w_{1,k}] \cdot \frac{1}{\sqrt{2\pi\sigma_1}} \cdot \exp\left(-\frac{w_{1,k}^2}{2\sigma_1^2}\right) \quad (5.102)$$

$$= \frac{1}{\sqrt{2\pi\sigma_1}} \cdot \exp\left(-\frac{(\tilde{y}_{1,k} - \bar{y}_{1,k})^2}{2\sigma_1^2}\right). \quad (5.103)$$

Message Update Schedule

The order of message calculations is defined by an update schedule. If the factor graph had no cycles, then an optimal update schedule would exist. Since our factor graph has cycles, the update schedule is not unique and it is an important degree of freedom. One possible schedule that yields good results is presented next.

First, the messages of all terminal nodes, which are marked in bold in Figure 5.15, are initialized as follows:

- Each terminal node at the top is initialized with a uniform distribution so that both message entries have identical values.
- The terminal nodes on the left and on the right side of the full factor graph are initialized according to (5.50) and (5.51).
- The terminal nodes at the bottom are initialized with the corresponding sample values of the EMG signals⁹.

Second, the messages out of the noise nodes are calculated. All this, the initialization of the terminal node messages and the calculation of the

⁹The messages leaving the terminal nodes at the bottom of the graph are actually Gaussian messages, which we will use in the next section more widely.

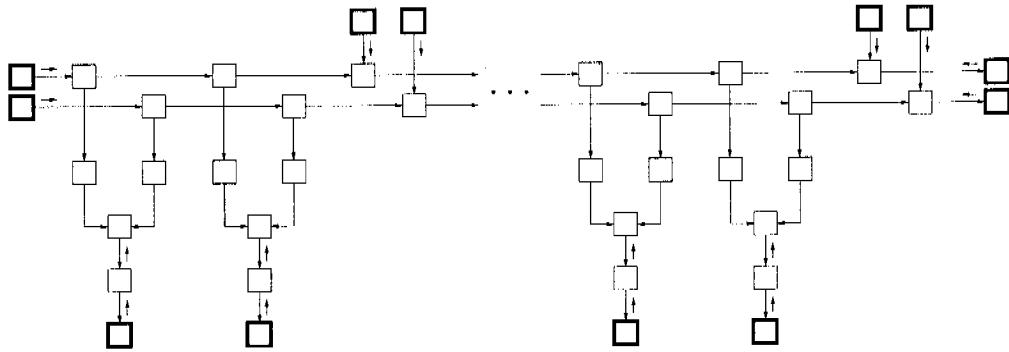


Figure 5.15: Messages that are applied/computed during the initialization phase.

messages coming out of the noise nodes, is done only one time and it is not repeated¹⁰.

After this initialization phase, the messages are calculated time slice by time slice in forward direction (k increases). For each time slice, an update schedule as visualized in Figure 5.16 is used:

- 1) Messages are sent from the extraction nodes towards the coefficient nodes.
- 2) Messages are sent from the coefficient nodes towards the sum constraint nodes.
- 3) If there were more than two sources, additional sum constraint nodes would be necessary and messages between the sum constraint nodes would have to be computed. This is visualized by the dashed factor graph part.
- 4) Messages are sent from the sum constraint nodes toward the coefficient nodes.
- 5) Messages are sent from the coefficient nodes towards the extraction nodes.
- 6) Messages are sent from the extraction nodes towards the source nodes.

¹⁰If a different noise variance is to be tried (see Section 5.8), the graph is re-created and newly initialized.

- 7) Messages are sent from the source nodes towards the next time slice.

This is done for all time slices from $k = 0$ to $k = k_{\max}$.

Next, messages are calculated in the backward direction, i.e., from $k = k_{\max}$ to $k = 0$. The update schedule for each time slice is visualized in Figure 5.17.

These forward and backward sweeps are repeated several times until a maximum number of iterations is reached. Also other stop criteria can be used. In fact, the message update process is often stopped as soon as it can be assumed that the results are good enough or when no further improvements are likely to occur. For example, the mean squared error between the reconstructed¹¹ signal and the given EMG signal can be used. Here we assume that the decomposition performance is good if the reconstructed signal is similar to the given EMG signal. Another example for a stop criterion is a measure of how strongly the messages in the factor graph change from iteration to iteration. Almost constant messages mean that the algorithm has converged and further iterations are unlikely to change the outcome.

After enough forward and backward sweeps have been performed, the messages indicated in Figure 5.18 are computed. Based on these messages, the binary source signals X can be estimated. This is described in Section 5.5.

The schedule presented here can be modified in many ways. However, we have not found another schedule yielding a better performance [121].

Remark 5.1. (Multiple Channels)

As we can see in Figure 5.2, our message-passing algorithms can naturally and simultaneously deal with multiple EMG channels recorded with different electrodes. Our approach also ensures that noisy channels that are associated with a high noise model variance automatically influence the final decomposition results less than the other channels. This is due to the fact that the messages reaching the extraction nodes from below are more flat in the case of a high noise variance. For example, in the extreme case of a flat message, the outgoing message of an equal node will not be influenced by this equal-valued message. □

¹¹Refer to the glossary.

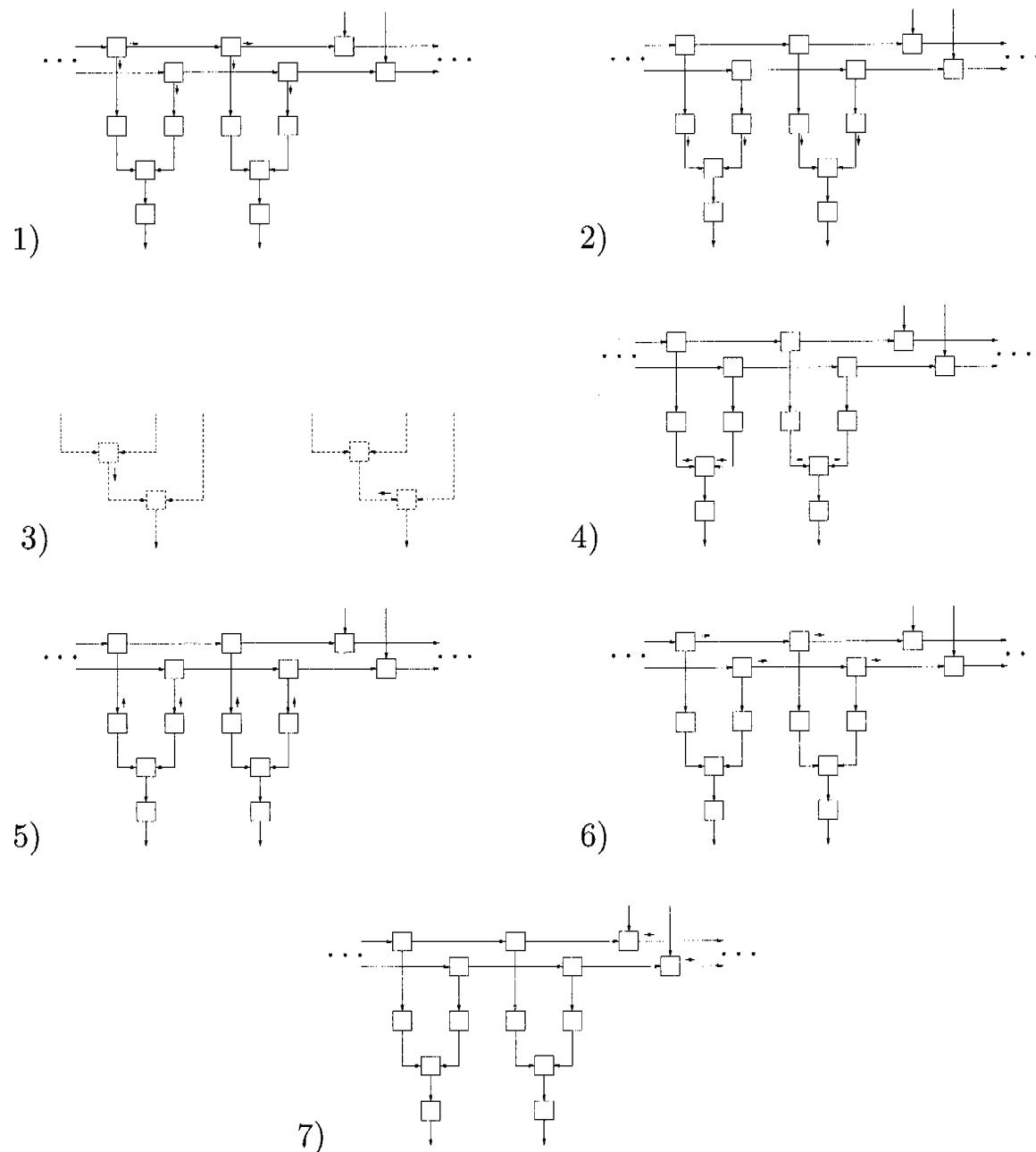


Figure 5.16: Update schedule for the forward direction (increasing discrete time k).

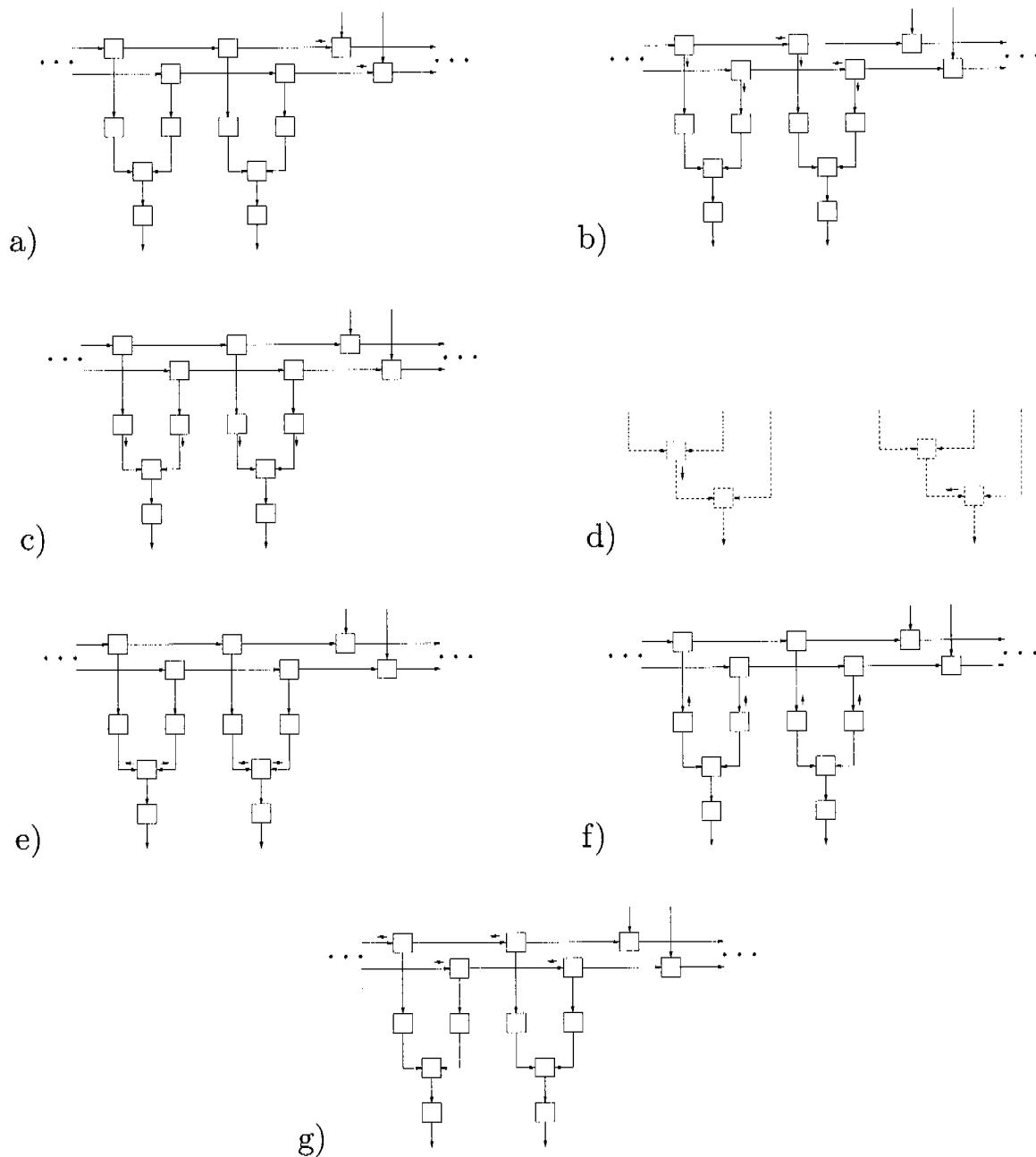


Figure 5.17: Update schedule for the backward direction (decreasing discrete time k).

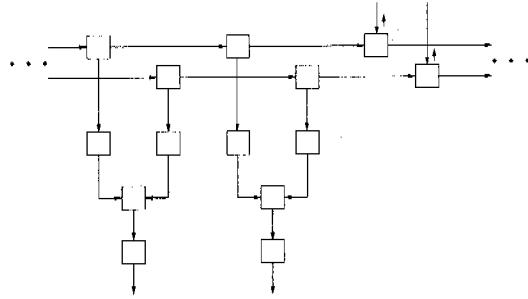


Figure 5.18: Messages that are computed during the read-out phase.

5.3 Resolving Superpositions Using Scalar-Gaussian Messages

In this section, we assume that the variables above the coefficient nodes \mathbb{C} are discrete and the variables below are continuous. The messages above the coefficient nodes are discrete messages as in the previous section and the messages below are scalar-Gaussian messages.

5.3.1 Factor Graph

Figure 5.19 shows one slice of the factor graph. Note that the topology of the factor graph has not changed compared to the factor graph in the previous section. Next we explain the variables, messages, and nodes in the graph in more detail.

5.3.2 Variables and Message Types

Variable $X_{i,k}$: The same as in Section 5.2.

Message: The same as in Section 5.2.

Variables $S_{i,k}$: The same as in Section 5.2.

Message: The same as in Section 5.2.

Variable $\tilde{S}_{i,k}$: The same as in Section 5.2.

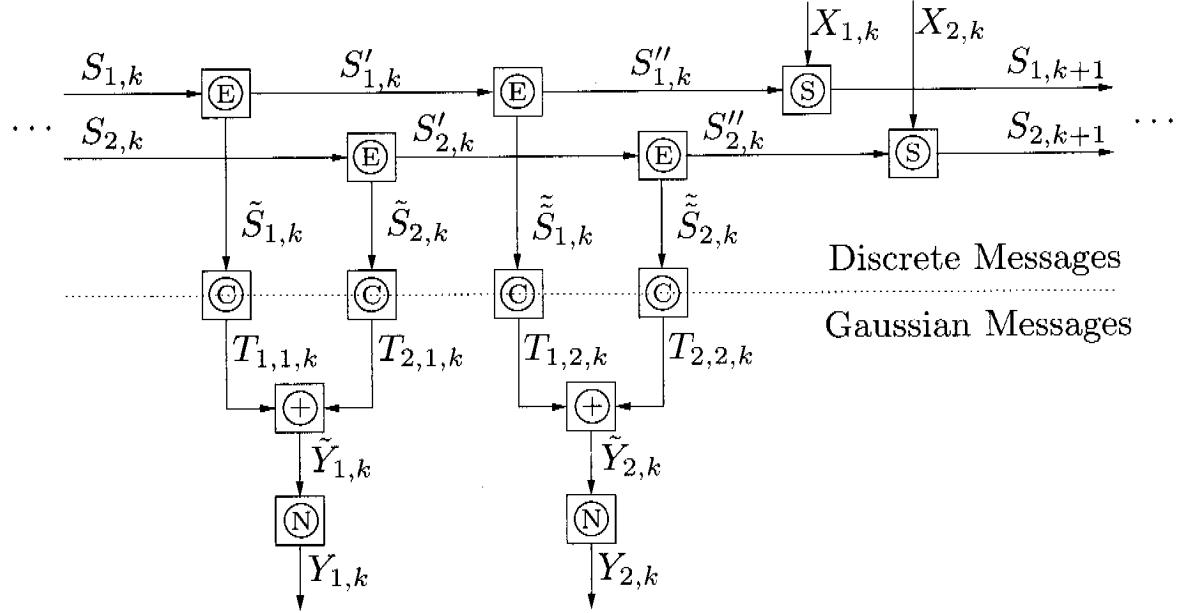


Figure 5.19: One section of the whole factor graph for the case where we use discrete and Gaussian messages.

Message: The same as in Section 5.2.

Variable $T_{i,j,k}$: The variable $T_{i,j,k}$ represents an amplitude value of the motor unit action potential train of source i and channel j at time k . The variable can take on real values, i.e., $T_{i,j,k} \in \mathbb{R}$.

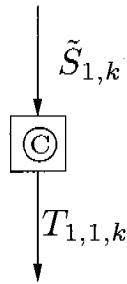
Message: A message along the edge corresponding to $T_{i,j,k}$ is a scalar-Gaussian message as introduced in Section 5.1. It is represented by the Gaussian distribution

$$\mathcal{N}(t | m_T, \sigma_T^2) \triangleq \frac{1}{\sqrt{2\pi}\sigma_T} \exp\left(-\frac{(t - m_T)^2}{2\sigma_T^2}\right), \quad (5.104)$$

where m_T is the mean and σ_T is the standard deviation. The variance is $V_T = \sigma_T^2$.

Variable $\tilde{Y}_{j,k}$: The variable $\tilde{Y}_{j,k}$ represents the discrete-time EMG signal without noise of channel j at time k . The variable can take on real values, i.e., $\tilde{Y}_{j,k} \in \mathbb{R}$.

Message: A message along the edge corresponding to $\tilde{Y}_{j,k}$ is a scalar-

**Figure 5.20:** Coefficient node.

Gaussian message represented by the Gaussian distribution

$$\mathcal{N}(\tilde{y} | m_{\tilde{Y}}, \sigma_{\tilde{Y}}^2) \triangleq \frac{1}{\sqrt{2\pi}\sigma_{\tilde{Y}}} \exp\left(-\frac{(\tilde{y} - m_{\tilde{Y}})^2}{2\sigma_{\tilde{Y}}^2}\right). \quad (5.105)$$

Variable $Y_{j,k}$: The variable $Y_{j,k}$ represents the discrete-time EMG signal with noise of channel j at time k . The variable can take on real values, i.e., $Y_{j,k} \in \mathbb{R}$.

Message: A message along the edge corresponding to $Y_{j,k}$ is a scalar-Gaussian message represented by the Gaussian distribution

$$\mathcal{N}(y | m_Y, \sigma_Y^2) \triangleq \frac{1}{\sqrt{2\pi}\sigma_Y} \exp\left(-\frac{(y - m_Y)^2}{2\sigma_Y^2}\right). \quad (5.106)$$

5.3.3 Node Functions and Message Update Rules

We now explain the nodes in Figure 5.19 and the corresponding message update rules. The source nodes \textcircled{S} and the extraction nodes \textcircled{E} in Figure 5.19 are the same as in Figure 5.2 of Section 5.2.

Coefficient Nodes

The coefficient nodes, which are denoted by \textcircled{C} in Figure 5.19, translate the states into amplitude values and vice versa. For example, let us consider the node in Figure 5.20.

As in the previous section, the relationship between the variables $\tilde{S}_{1,k}$ and $T_{1,1,k}$ is visualized in Figure 5.10. It can also be described by the following equation:

$$T_{1,1,k} = \begin{cases} h_{1,1,\tilde{S}_{1,k}} & \text{if } \tilde{S}_{1,k} \in \{0, 1, 2\} \\ 0 & \text{if } \tilde{S}_{1,k} = 3 \end{cases}$$

Therefore, the node function for the node in Figure 5.9 can be stated as:

$$f_{\odot}(\tilde{s}_{1,k}, t_{1,1,k}) = \begin{cases} \delta[t_{1,1,k} - h_{1,1,\tilde{S}_{1,k}}] & \text{if } \tilde{S}_{1,k} \in \{0, 1, 2\} \\ \delta[t_{1,1,k}] & \text{if } \tilde{S}_{1,k} = 3 \end{cases}$$

With this node function and the incoming message $\vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})$, an intermediate discrete message $\vec{\mu}_{\tilde{T}_{1,1,k}}(\tilde{t}_{1,1,k})$ in direction of the coefficient node can be computed as in the previous section:

$$\vec{\mu}_{\tilde{T}_{1,1,k}}(\tilde{t}_{1,1,k}) = \sum_{\tilde{s}_{1,k}} f_{\odot}(\tilde{s}_{1,k}, \tilde{t}_{1,1,k}) \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \quad (5.107)$$

$$= \sum_{\substack{\text{active states } \tilde{S}_{1,k}}} \delta[\tilde{t}_{1,1,k} - h_{1,1,\tilde{S}_{1,k}}] \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) + \sum_{\substack{\text{idle state } \tilde{S}_{1,k}}} \delta[\tilde{t}_{1,1,k}] \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \quad (5.108)$$

$$= \delta[\tilde{t}_{1,1,k} - h_{1,1,0}] \vec{\mu}_{\tilde{S}_{1,k}}(0) + \delta[\tilde{t}_{1,1,k} - h_{1,1,1}] \vec{\mu}_{\tilde{S}_{1,k}}(1) + \delta[\tilde{t}_{1,1,k} - h_{1,1,2}] \vec{\mu}_{\tilde{S}_{1,k}}(2) + \delta[\tilde{t}_{1,1,k}] \vec{\mu}_{\tilde{S}_{1,k}}(3) \quad (5.109)$$

The variable $T_{1,1,k}$ is now—in contrast to the previous section—continuous and the message $\vec{\mu}_{T_{1,1,k}}(t_{1,1,k})$ is approximated using a Gaussian parameterizations:

$$\vec{\mu}_{T_{1,1,k}}(t_{1,1,k}) = \mathcal{N}\left(t_{1,1,k} \mid \vec{m}_{T_{1,1,k}}, \vec{\sigma}_{T_{1,1,k}}^2\right) \quad (5.110)$$

$$= \frac{1}{\sqrt{2\pi} \vec{\sigma}_{T_{1,1,k}}} \exp\left(-\frac{(t_{1,1,k} - \vec{m}_{T_{1,1,k}})^2}{2\vec{\sigma}_{T_{1,1,k}}^2}\right), \quad (5.111)$$

where $\vec{m}_{T_{1,1,k}}$ is the mean and $\vec{\sigma}_{T_{1,1,k}}$ is the standard deviation. The variance is $\vec{V}_{T_{1,1,k}} = \vec{\sigma}_{T_{1,1,k}}^2$. Therefore, we need to compute the parameters $\vec{m}_{T_{1,1,k}}$ and $\vec{V}_{T_{1,1,k}}$ of the scalar-Gaussian message $\vec{\mu}_{T_{1,1,k}}(t_{1,1,k})$ based on $\vec{\mu}_{\tilde{T}_{1,1,k}}(\tilde{t}_{1,1,k})$:

$$\vec{m}_{\tilde{T}_{1,1,k}} = \mathbb{E}[\tilde{T}_{1,1,k}] \quad (5.112)$$

$$\approx \frac{1}{\sum_{\tilde{t}_{1,1,k}} \vec{\mu}_{\tilde{T}_{1,1,k}}(\tilde{t}_{1,1,k})} \sum_{\tilde{t}_{1,1,k}} \vec{\mu}_{\tilde{T}_{1,1,k}}(\tilde{t}_{1,1,k}) \tilde{t}_{1,1,k} \quad (5.113)$$

$$\vec{V}_{\tilde{T}_{1,1,k}} = \mathbb{E}[(\tilde{T}_{1,1,k} - \vec{m}_{\tilde{T}_{1,1,k}})^2] \quad (5.114)$$

$$\approx \frac{1}{\sum_{\tilde{t}_{1,1,k}} \vec{\mu}_{\tilde{T}_{1,1,k}}(\tilde{t}_{1,1,k})} \sum_{\tilde{t}_{1,1,k}} \vec{\mu}_{\tilde{T}_{1,1,k}}(\tilde{t}_{1,1,k}) (\tilde{t}_{1,1,k} - \vec{m}_{\tilde{T}_{1,1,k}})^2 \quad (5.115)$$

We set the parameters of the scalar-Gaussian message $\vec{\mu}_{T_{1,1,k}}(t_{1,1,k})$ to

$$\vec{m}_{T_{1,1,k}} = \vec{m}_{\tilde{T}_{1,1,k}} \quad (5.116)$$

$$\vec{V}_{T_{1,1,k}} = \vec{V}_{\tilde{T}_{1,1,k}}. \quad (5.117)$$

It is also possible to avoid the intermediate variable $\tilde{t}_{1,1,k}$. In this case, we calculate $\vec{m}_{T_{1,1,k}}$ and $\vec{V}_{T_{1,1,k}}$ directly based on $\vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})$ (cf., Section A.2):

$$\vec{m}_{T_{1,1,k}} \approx \frac{1}{\sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \tilde{h}_{1,1,\tilde{S}_{1,k}} \quad (5.118)$$

$$= \frac{1}{\sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\text{active states } \tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) h_{1,1,\tilde{S}_{1,k}} \quad (5.119)$$

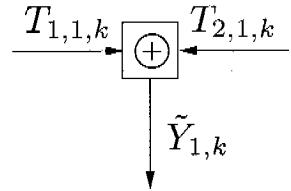
$$\vec{V}_{T_{1,1,k}} \approx \frac{1}{\sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) (\tilde{h}_{1,1,\tilde{S}_{1,k}} - \vec{m}_{T_{1,1,k}})^2, \quad (5.120)$$

where $\tilde{h}_{1,1,k}$ is defined as

$$\tilde{h}_{1,1,s} \triangleq \begin{cases} h_{1,1,s} & \text{if } s \in \{0, 1, 2\} \\ 0 & \text{if } s = 3. \end{cases} \quad (5.121)$$

The message in the opposite direction is determined by using the same approach as in the previous section:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \sum_{t_{1,1,k}} f_{\odot}(\tilde{s}_{1,k}, t_{1,1,k}) \overleftarrow{\mu}_{T_{1,1,k}}(t_{1,1,k}) \quad (5.122)$$

**Figure 5.21:** Sum constraint node.

For the active states $\tilde{S}_{1,k} \in \{0, 1, 2\}$ it follows:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \sum_{t_{1,1,k}} \delta[t_{1,1,k} - h_{1,1,\tilde{S}_{1,k}}] \overleftarrow{\mu}_{T_{1,1,k}}(t_{1,1,k}) \quad (5.123)$$

$$= \overleftarrow{\mu}_{T_{1,1,k}}(h_{1,1,k}) \quad (5.124)$$

and for the idle state $\tilde{S}_{1,k} = 3$:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \overleftarrow{\mu}_{T_{1,1,k}}(0) \quad (5.125)$$

Therefore, the upward message $\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})$ is computed by sampling the scalar-Gaussian distribution $\overleftarrow{\mu}_{T_{1,1,k}}(t_{1,1,k})$ at the values of the filter coefficients of the corresponding states.

Sum Constraint Nodes

The sum constraint nodes, which are denoted by \oplus in Figure 5.19, combine the information from the sources. As in the previous section, let us consider the node in Figure 5.21 with the relationship

$$\tilde{y}_{1,k} = t_{1,1,k} + t_{2,1,k}. \quad (5.126)$$

For continuous variables $T_{1,1,k}$, $T_{2,1,k}$, and $\tilde{Y}_{1,k}$, the node function is:

$$f_{\oplus}(t_{1,1,k}, t_{2,1,k}, \tilde{y}_{1,k}) = \delta(\tilde{y}_{1,k} - t_{1,1,k} - t_{2,1,k}). \quad (5.127)$$

Let us determine the message $\overrightarrow{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k})$ given that the incoming messages are the Gaussian messages

$$\overrightarrow{\mu}_{T_{1,1,k}}(t_{1,1,k}) \propto \mathcal{N}\left(t_{1,1,k} \mid \vec{m}_{T_{1,1,k}}, \vec{V}_{T_{1,1,k}}\right) \quad (5.128)$$

$$= \frac{1}{\sqrt{2\pi} \vec{\sigma}_{T_{1,1,k}}} \exp \left(-\frac{(t_{1,1,k} - \vec{m}_{T_{1,1,k}})^2}{2\vec{V}_{T_{1,1,k}}} \right) \quad (5.129)$$

$$\vec{\mu}_{T_{2,1,k}}(t_{2,1,k}) \propto \mathcal{N}(t_{2,1,k} | \vec{m}_{T_{2,1,k}}, \vec{V}_{T_{2,1,k}}) \quad (5.130)$$

$$= \frac{1}{\sqrt{2\pi} \vec{\sigma}_{T_{2,1,k}}} \exp \left(-\frac{(t_{2,1,k} - \vec{m}_{T_{2,1,k}})^2}{2\vec{V}_{T_{2,1,k}}} \right) \quad (5.131)$$

by using the continuous version of the sum-product rule:

$$\begin{aligned} \vec{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k}) &\propto \int_{t_{1,1,k}} \int_{t_{2,1,k}} \delta[\tilde{y}_{1,k} - t_{1,1,k} - t_{2,1,k}] \\ &\quad \cdot \vec{\mu}_{T_{1,1,k}}(t_{1,1,k}) \cdot \vec{\mu}_{T_{2,1,k}}(t_{2,1,k}) dt_{2,1,k} dt_{1,1,k} \end{aligned} \quad (5.132)$$

$$\begin{aligned} &\vdots \\ &\propto \exp \left(-\frac{((t_{1,1,k} + t_{2,1,k}) - (\vec{m}_{T_{1,1,k}} + \vec{m}_{T_{2,1,k}}))^2}{2(\vec{V}_{T_{1,1,k}} + \vec{V}_{T_{2,1,k}})} \right) \\ &\propto \frac{1}{\sqrt{2\pi} \vec{\sigma}_{\tilde{Y}_{1,k}}} \exp \left(-\frac{(\tilde{y}_{1,k} - \vec{m}_{\tilde{Y}_{1,k}})^2}{2\vec{V}_{\tilde{Y}_{1,k}}} \right) \end{aligned} \quad (5.133)$$

$$\propto \mathcal{N}(\tilde{y}_{1,k} | \vec{m}_{\tilde{Y}_{1,k}}, \vec{V}_{\tilde{Y}_{1,k}}). \quad (5.134)$$

The complete derivation is omitted here, but (5.134) indicates that the resulting outgoing message $\vec{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k})$ is again a Gaussian distribution. To easily compute the outgoing messages of the sum constraint node given Gaussian input messages, we can use the following simple rules:

Summary of the message update rules for the sum constraint node (scalar-Gaussian messages)

$$\vec{\mu}_{\tilde{Y}_{1,k}}(\tilde{y}_{1,k}) \propto \mathcal{N}\left(\tilde{y}_{1,k} \mid \vec{m}_{\tilde{Y}_{1,k}}, \vec{V}_{\tilde{Y}_{1,k}}\right) \quad (5.135)$$

$$= \mathcal{N}\left(\tilde{y}_{1,k} \mid \vec{m}_{T_{1,1,k}} + \vec{m}_{T_{2,1,k}}, \vec{V}_{T_{1,1,k}} + \vec{V}_{T_{2,1,k}}\right) \quad (5.136)$$

$$\overleftarrow{\mu}_{T_{1,1,k}}(t_{1,1,k}) \propto \mathcal{N}\left(t_{1,1,k} \mid \overleftarrow{m}_{T_{1,1,k}}, \overleftarrow{V}_{T_{1,1,k}}\right) \quad (5.137)$$

$$= \mathcal{N}\left(t_{1,1,k} \mid \overleftarrow{m}_{\tilde{Y}_{1,k}} - \vec{m}_{T_{2,1,k}}, \overleftarrow{V}_{\tilde{Y}_{1,k}} + \vec{V}_{T_{2,1,k}}\right) \quad (5.138)$$

$$\overleftarrow{\mu}_{T_{2,1,k}}(t_{2,1,k}) \propto \mathcal{N}\left(t_{2,1,k} \mid \overleftarrow{m}_{T_{2,1,k}}, \overleftarrow{V}_{T_{2,1,k}}\right) \quad (5.139)$$

$$= \mathcal{N}\left(t_{2,1,k} \mid \overleftarrow{m}_{\tilde{Y}_{1,k}} - \vec{m}_{T_{1,1,k}}, \overleftarrow{V}_{\tilde{Y}_{1,k}} + \vec{V}_{T_{2,1,k}}\right) \quad (5.140)$$

The message update is much faster when dealing only with the parameters of Gaussian messages compared to the discrete case in the previous section. This fact makes our decomposition algorithm much faster.

We can derive similar update rules also for other nodes, and for the general case of multivariate-Gaussian messages. A list of such rules is given in Table B.1.

Noise Nodes

As before, the noise nodes, which are denoted by \textcircled{N} in Figure 5.19, model the amplitude noise in the EMG sample values. For example, let us consider the node in Figure 5.22.

The message $\overleftarrow{\mu}_{Y_{1,k}(y_{1,k})}$ leaving the terminal nodes¹² at the bottom of the graph is a Gaussian message with the two parameters initialized to

$$\overleftarrow{m}_{Y_{1,k}} = y_{1,k} \quad (5.141)$$

¹²The terminal nodes are not depicted in Figure 5.19. Refer to Figure 5.6 instead.

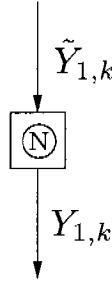


Figure 5.22: Noise node for the scalar-Gaussian case.

$$\overleftarrow{V}_{Y_{1,k}} = 0, \quad (5.142)$$

where $y_{1,k}$ is a fixed sample of the EMG signal at time k and channel 1.

The parameters of the scalar-Gaussian outgoing message $\overleftarrow{\mu}_{\tilde{Y}_{1,k}(\tilde{y}_{1,k})}$ can easily be determined as

$$\overleftarrow{m}_{\tilde{Y}_{1,k}} = \overleftarrow{m}_{Y_{1,k}} \quad (5.143)$$

$$\overleftarrow{V}_{\tilde{Y}_{1,k}} = \sigma_1^2, \quad (5.144)$$

where σ_1^2 is the noise variance of channel 1.

Message Update Schedule

The message update schedule is defined as in the previous section.

5.4 Resolving Superpositions Using Multivariate-Gaussian Messages

As in the previous section, in this section we assume that the variables above the coefficient nodes \odot are discrete and the variables below are continuous. The messages above the coefficient nodes are discrete messages as in the previous section and the messages below are now multivariate-Gaussian messages.

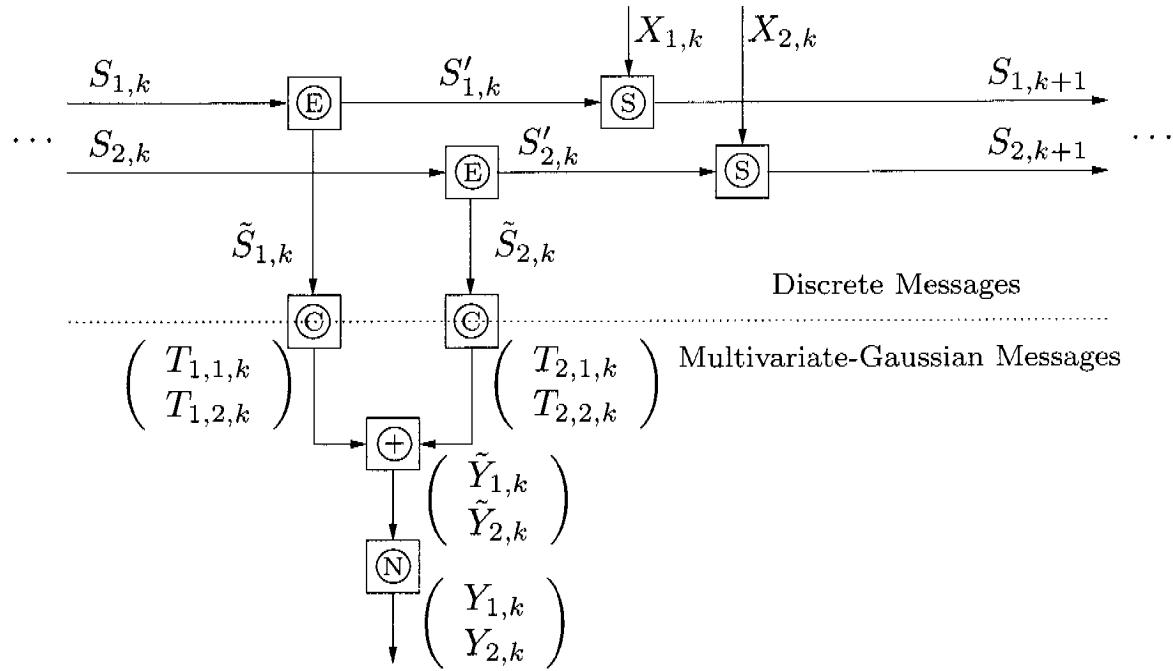


Figure 5.23: One section of the whole factor graph for the case where we use discrete and multivariate-Gaussian messages.

5.4.1 Factor Graph

To avoid short cycles within one time slice (see Figure 4.18), we developed an algorithm where several channels are handled simultaneously by using vector variables instead of scalars. This lead to multivariate-Gaussian messages below the coefficient nodes. Figure 5.23 shows one slice of the factor graph for this case.

Next we explain the variables, messages, and nodes in the graph in more detail.

5.4.2 Variables and Message Types

Variable $X_{i,k}$: The same as in Section 5.2 and in Section 5.3.

Message: The same as in Section 5.2 and in Section 5.3.

Variables $S_{i,k}$: The same as in Section 5.2 and in Section 5.3.

Message: The same as in Section 5.2 and in Section 5.3.

Variables $\tilde{S}_{i,k}$: The same as in Section 5.2 and in Section 5.3.

Message: The same as in Section 5.2 and in Section 5.3.

Variable $\mathbf{T}_{i,k}$: The variable $\mathbf{T}_{i,k} \triangleq \begin{pmatrix} T_{i,1,k} \\ T_{i,2,k} \end{pmatrix}$ consists of the components as introduced in Section 5.3.

Message: A message along the edge corresponding to $\mathbf{T}_{i,k}$ is a multivariate-Gaussian message as introduced in Section 5.1. It is represented by the multivariate-Gaussian distribution

$$\begin{aligned} & \mathcal{N}(\mathbf{t}_{i,k} | \mathbf{m}_{\mathbf{T}_{i,k}}, \mathbf{V}_{\mathbf{T}_{i,k}}) \\ &= \frac{1}{\sqrt{(2\pi)^n |\mathbf{V}_{\mathbf{T}_{i,k}}|}} \exp \left(-\frac{1}{2} (\mathbf{t}_{i,k} - \mathbf{m}_{\mathbf{T}_{i,k}})^\top \mathbf{V}_{\mathbf{T}_{i,k}}^{-1} (\mathbf{t}_{i,k} - \mathbf{m}_{\mathbf{T}_{i,k}}) \right), \end{aligned} \quad (5.145)$$

where $\mathbf{m}_{\mathbf{T}_{i,k}}$ is the mean vector, $\mathbf{V}_{\mathbf{T}_{i,k}}$ is the covariance matrix, and $n = \dim(\mathbf{T}_{i,k})$.

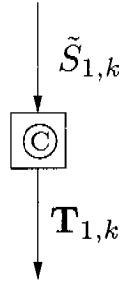
Variable $\tilde{\mathbf{Y}}_k$: The variable $\tilde{\mathbf{Y}}_k \triangleq \begin{pmatrix} \tilde{Y}_{1,k} \\ \tilde{Y}_{2,k} \end{pmatrix}$ consists of the components as introduced in Section 5.3.

Message: A message along the edge corresponding to $\tilde{\mathbf{Y}}_k$ is a multivariate-Gaussian message as introduced in Section 5.1. It is represented by the multivariate-Gaussian distribution

$$\begin{aligned} & \mathcal{N}(\tilde{\mathbf{y}}_k | \mathbf{m}_{\tilde{\mathbf{Y}}_k}, \mathbf{V}_{\tilde{\mathbf{Y}}_k}) \\ &= \frac{1}{\sqrt{(2\pi)^n |\mathbf{V}_{\tilde{\mathbf{Y}}_k}|}} \exp \left(-\frac{1}{2} (\tilde{\mathbf{y}}_k - \mathbf{m}_{\tilde{\mathbf{Y}}_k})^\top \mathbf{V}_{\tilde{\mathbf{Y}}_k}^{-1} (\tilde{\mathbf{y}}_k - \mathbf{m}_{\tilde{\mathbf{Y}}_k}) \right), \end{aligned} \quad (5.146)$$

where $\mathbf{m}_{\tilde{\mathbf{Y}}_k}$ is the mean vector, $\mathbf{V}_{\tilde{\mathbf{Y}}_k}$ is the covariance matrix, and $n = \dim(\tilde{\mathbf{Y}}_k)$.

Variable \mathbf{Y}_k : The variable $\mathbf{Y}_k \triangleq \begin{pmatrix} Y_{1,k} \\ Y_{2,k} \end{pmatrix}$ consists of the components as introduced in Section 5.3.

**Figure 5.24:** Coefficient node.

Message: A message along the edge corresponding to \mathbf{Y}_k is a multivariate-Gaussian message as introduced in Section 5.1. It is represented by the multivariate-Gaussian distribution

$$\begin{aligned} & \mathcal{N}(\mathbf{y}_k | \mathbf{m}_{\mathbf{Y}_k}, \mathbf{V}_{\mathbf{Y}_k}) \\ &= \frac{1}{\sqrt{(2\pi)^n |\mathbf{V}_{\mathbf{Y}_k}|}} \exp\left(-\frac{1}{2}(\mathbf{y}_k - \mathbf{m}_{\mathbf{Y}_k})^\top \mathbf{V}_{\mathbf{Y}_k}^{-1} (\mathbf{y}_k - \mathbf{m}_{\mathbf{Y}_k})\right), \end{aligned} \quad (5.147)$$

where $\mathbf{m}_{\mathbf{Y}_k}$ is the mean vector, $\mathbf{V}_{\mathbf{Y}_k}$ is the covariance matrix, and $n = \dim(\mathbf{Y}_k)$.

5.4.3 Node Functions and Message Update Rules

We now explain the nodes in Figure 5.23 and the corresponding message update rules. The source nodes \textcircled{S} and the extraction nodes \textcircled{E} in Figure 5.19 are the same as in Figure 5.2 of Section 5.2.

Coefficient Nodes

Let us consider the coefficient node in Figure 5.24. The relationship between the variables $\tilde{S}_{1,k}$ and $\mathbf{T}_{1,k}$ is visualized in Figure 5.25. We calculate the parameters of the multivariate-Gaussian message

$$\begin{aligned} \vec{\mu}_{\mathbf{T}_{1,k}} &= \mathcal{N}\left(\mathbf{t}_{1,k} | \vec{\mathbf{m}}_{\mathbf{T}_{1,k}}, \vec{\mathbf{V}}_{\mathbf{T}_{1,k}}\right) \\ &= \frac{1}{\sqrt{(2\pi)^n |\vec{\mathbf{V}}_{\mathbf{T}_{1,k}}|}} \exp\left(-\frac{1}{2}(\mathbf{t}_{1,k} - \vec{\mathbf{m}}_{\mathbf{T}_{1,k}})^\top \vec{\mathbf{V}}_{\mathbf{T}_{1,k}}^{-1} (\mathbf{t}_{1,k} - \vec{\mathbf{m}}_{\mathbf{T}_{1,k}})\right) \end{aligned} \quad (5.148)$$

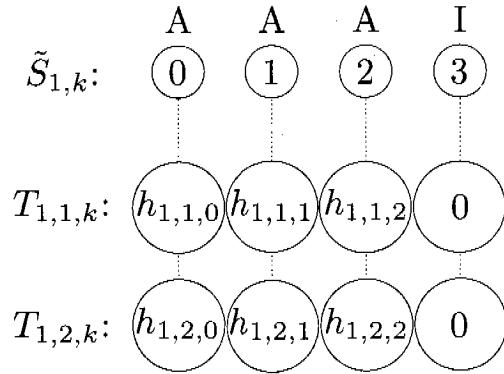


Figure 5.25: FIR state variable $\tilde{S}_{1,k}$ and FIR coefficients $\mathbf{T}_{1,k} \triangleq \begin{pmatrix} T_{1,1,k} \\ T_{1,2,k} \end{pmatrix}$. “A” stands for an active state, “I” for an idle state.

in downward direction as follows (cf., Section A.2 and previous section for the scalar-Gaussian case), where $\tilde{\mathbf{h}}_{1,s}$ is defined as

$$\tilde{\mathbf{h}}_{1,s} = \begin{pmatrix} \tilde{h}_{1,1,s} \\ \tilde{h}_{1,2,s} \end{pmatrix} \quad (5.149)$$

with

$$\tilde{h}_{1,j,s} \triangleq \begin{cases} h_{1,j,s} & \text{if } s \in \{0, 1, 2\} \\ 0 & \text{if } s = 3. \end{cases} \quad (5.150)$$

The mean vector then becomes:

$$\vec{\mathbf{m}}_{\mathbf{T}_{1,k}} = \mathbb{E}[\mathbf{T}_{1,k}] \quad (5.151)$$

$$= \mathbb{E}\left[\begin{pmatrix} T_{1,1,k} \\ T_{1,2,k} \end{pmatrix}\right] \quad (5.152)$$

$$= \begin{pmatrix} \mathbb{E}[T_{1,1,k}] \\ \mathbb{E}[T_{1,2,k}] \end{pmatrix} \quad (5.153)$$

$$= \begin{pmatrix} m_{T_{1,1,k}} \\ m_{T_{1,2,k}} \end{pmatrix} \quad (5.154)$$

$$\approx \frac{1}{\sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \tilde{\mathbf{h}}_{1,\tilde{S}_{1,k}} \quad (5.155)$$

Therefore, the components of $\vec{\mathbf{m}}_{\mathbf{T}_{1,k}}$ are:

$$m_{T_{1,1,k}} \approx \frac{1}{\sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\tilde{s}_{1,k}} \vec{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \tilde{h}_{1,1,\tilde{S}_{1,k}} \quad (5.156)$$

$$= \frac{1}{\sum_{\tilde{s}_{1,k}} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\text{active states } \tilde{s}_{1,k}} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) h_{1,1,\tilde{S}_{1,k}} \quad (5.157)$$

$$m_{T_{1,2,k}} \approx \frac{1}{\sum_{\tilde{s}_{1,k}} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\tilde{s}_{1,k}} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \tilde{h}_{1,2,\tilde{S}_{1,k}} \quad (5.158)$$

$$= \frac{1}{\sum_{\tilde{s}_{1,k}} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\text{active states } \tilde{s}_{1,k}} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) h_{1,2,\tilde{S}_{1,k}} \quad (5.159)$$

The covariance matrix of the downward message is:

$$\begin{aligned} \overrightarrow{\mathbf{V}}_{\mathbf{T}_{1,k}} &= \frac{1}{\sum_{\tilde{s}_{1,k}} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k})} \sum_{\tilde{s}_{1,k}} \overrightarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) \cdot \\ &\quad (\tilde{\mathbf{h}}_{1,\tilde{S}_{1,k}} - \overrightarrow{\mathbf{m}}_{\mathbf{T}_{1,k}})(\tilde{\mathbf{h}}_{1,\tilde{S}_{1,k}} - \overrightarrow{\mathbf{m}}_{\mathbf{T}_{1,k}})^T. \end{aligned} \quad (5.160)$$

Refer to Section A.2.2 for a derivation of how to calculate the components of the matrix $\overrightarrow{\mathbf{V}}_{\mathbf{T}_{1,k}}$.

The message in the opposite direction is computed as follows:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \sum_{\mathbf{t}_{1,k}} f_{\odot}(\tilde{s}_{1,k}, \mathbf{t}_{1,k}) \overleftarrow{\mu}_{\mathbf{T}_{1,k}}(\mathbf{t}_{1,k}) \quad (5.161)$$

For the active states $\tilde{S}_{1,k} \in \{0, 1, 2\}$ it follows:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \sum_{\mathbf{t}_{1,k}} \delta[\mathbf{t}_{1,k} - \mathbf{h}_{1,\tilde{S}_{1,k}}] \overleftarrow{\mu}_{\mathbf{T}_{1,k}}(\mathbf{t}_{1,k}) \quad (5.162)$$

$$= \overleftarrow{\mu}_{\mathbf{T}_{1,k}}(\mathbf{h}_{1,\tilde{S}_{1,k}}) \quad (5.163)$$

$$\begin{aligned} &= \mathcal{N}\left(\mathbf{h}_{1,\tilde{S}_{1,k}} \mid \overleftarrow{\mathbf{m}}_{\mathbf{T}_{1,k}}, \overleftarrow{\mathbf{V}}_{\mathbf{T}_{1,k}}\right) \\ &= \frac{1}{\sqrt{(2\pi)^n |\overleftarrow{\mathbf{V}}_{\mathbf{T}_{1,k}}|}} \cdot \\ &\quad \exp\left(-\frac{1}{2}(\mathbf{h}_{1,\tilde{S}_{1,k}} - \overleftarrow{\mathbf{m}}_{\mathbf{T}_{1,k}})^T \overleftarrow{\mathbf{V}}_{\mathbf{T}_{1,k}}^{-1} (\mathbf{h}_{1,\tilde{S}_{1,k}} - \overleftarrow{\mathbf{m}}_{\mathbf{T}_{1,k}})\right) \end{aligned} \quad (5.164)$$

and for the idle state $\tilde{S}_{1,k} = 3$:

$$\overleftarrow{\mu}_{\tilde{S}_{1,k}}(\tilde{s}_{1,k}) = \overleftarrow{\mu}_{\mathbf{T}_{1,k}}(\mathbf{0}), \quad (5.165)$$

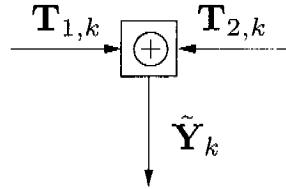


Figure 5.26: Sum constraint node.

where $\mathbf{0} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

Therefore, the upward message $\overleftarrow{\mu}_{\tilde{s}_{1,k}}(\tilde{s}_{1,k})$ is computed by sampling the multivariate-Gaussian distribution $\overleftarrow{\mu}_{\mathbf{T}_{1,k}}(\mathbf{t}_{1,k})$ at the values of the filter coefficients of the corresponding states.

Sum Constraint Nodes

The sum constraint nodes, which are denoted by \oplus in Figure 5.23, combine the information from the sources. As in the previous section, let us consider the node in Figure 5.26 with the relationship

$$\tilde{\mathbf{y}}_k = \mathbf{t}_{1,k} + \mathbf{t}_{2,k}. \quad (5.166)$$

To easily compute the outgoing messages of the sum constraint node given multivariate-Gaussian input messages, we can use the following simple rules¹³:

¹³The derivation is omitted here. Refer also to Table B.1 for a list of update rules for other node functions.

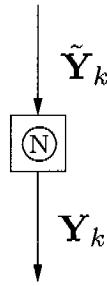


Figure 5.27: Noise node for the multivariate-Gaussian case.

Summary of the message update rules for the sum constraint node (multivariate-Gaussian messages)

$$\overrightarrow{\mu}_{\tilde{Y}_k}(\tilde{y}_k) \propto \mathcal{N}\left(\tilde{y}_k \mid \overrightarrow{m}_{\tilde{Y}_k}, \overrightarrow{V}_{\tilde{Y}_k}\right) \quad (5.167)$$

$$= \mathcal{N}\left(\tilde{y}_k \mid \overrightarrow{m}_{T_{1,k}} + \overrightarrow{m}_{T_{2,k}}, \overrightarrow{V}_{T_{1,k}} + \overrightarrow{V}_{T_{2,k}}\right) \quad (5.168)$$

$$\overleftarrow{\mu}_{T_{1,k}}(t_{1,k}) \propto \mathcal{N}\left(t_{1,k} \mid \overleftarrow{m}_{T_{1,k}}, \overleftarrow{V}_{T_{1,k}}\right) \quad (5.169)$$

$$= \mathcal{N}\left(t_{1,k} \mid \overleftarrow{m}_{\tilde{Y}_k} - \overrightarrow{m}_{T_{2,k}}, \overleftarrow{V}_{\tilde{Y}_k} + \overrightarrow{V}_{T_{2,k}}\right) \quad (5.170)$$

$$\overleftarrow{\mu}_{T_{2,k}}(t_{2,k}) \propto \mathcal{N}\left(t_{2,k} \mid \overleftarrow{m}_{T_{2,k}}, \overleftarrow{V}_{T_{2,k}}\right) \quad (5.171)$$

$$= \mathcal{N}\left(t_{2,k} \mid \overleftarrow{m}_{\tilde{Y}_k} - \overrightarrow{m}_{T_{1,k}}, \overleftarrow{V}_{\tilde{Y}_k} + \overrightarrow{V}_{T_{1,k}}\right) \quad (5.172)$$

Noise Nodes

As before, the noise nodes, which are denoted by \textcircled{N} in Figure 5.23, model the amplitude noise in the EMG sample values. For example, let us consider the node in Figure 5.27.

The message $\overleftarrow{\mu}_{Y_k(y_k)}$ leaving the terminal nodes¹⁴ at the bottom of the graph is a multivariate-Gaussian message with the two parameters

¹⁴The terminal nodes are not depicted in Figure 5.23. Refer to Figure 5.6 instead.

initialized to

$$\mathbf{m}_{\mathbf{Y}_k} = \begin{pmatrix} Y_{1,k} \\ Y_{2,k} \end{pmatrix} \quad (5.173)$$

$$\mathbf{V}_{\mathbf{Y}_k} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}. \quad (5.174)$$

The parameters of the multivariate-Gaussian outgoing message $\overleftarrow{\mu}_{\tilde{\mathbf{Y}}_k}(\tilde{\mathbf{y}}_k)$ can easily be determined as

$$\mathbf{m}_{\tilde{\mathbf{Y}}_k} = \mathbf{m}_{\mathbf{Y}_k} \quad (5.175)$$

$$\mathbf{V}_{\tilde{\mathbf{Y}}_k} = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix}, \quad (5.176)$$

where σ_j^2 is the noise variance of channel j .

Message Update Schedule

The message update schedule is defined as in the previous sections.

5.5 Improved Readout

5.5.1 Standard Symbol-Wise MAP Estimate

Consider the factor graph in Figure 5.28. Let us assume for now that the factor graph has no cycles¹⁵. Using the sum-product algorithm, we could then calculate the exact¹⁶ marginals $p(x_{i,k})$ for each source $i \in \{1, 2\}$ and

¹⁵In fact, the time slice depicted in Figure 5.28 has no cycles. However, the full factor graph has cycles, as we can see in Figure 4.17.

¹⁶Since the factor graph has cycles, we get at best only an approximation of the marginals.

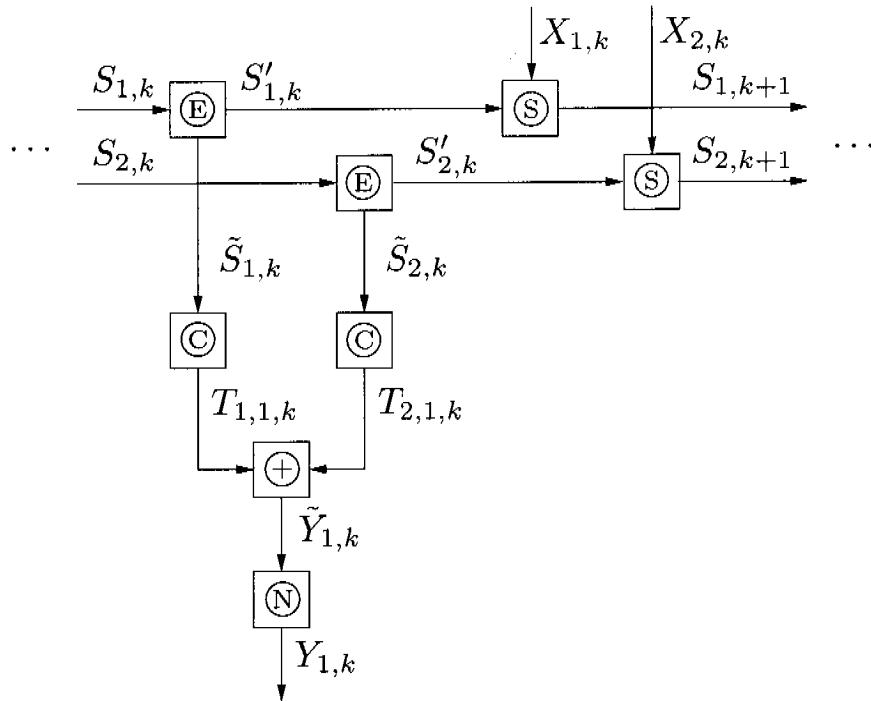


Figure 5.28: One section of the whole factor graph for the case where we use only discrete messages and one electrode.

each discrete time k , cf., Section 4.8:

$$p(x_{i,k}) = \underbrace{\sum_{\sim x_{i,k}} p(\text{all variables in the FG})}_{\substack{\text{Sum} \\ \text{Factor graph} \\ (\text{written as product})}} \quad (5.177)$$

Sum-product algorithm

where “ $\sim x_{i,k}$ ” means “all variables in the factor graph except variable $x_{i,k}$ ”. What does this mean? How do we get, e.g., $p(x_{1,k})$ by message passing in Figure 5.28? Let us assume that, after a few iterations, we get a stable discrete message $\overleftarrow{\mu}_{X_{1,k}}(x_{1,k})$. Since there is no message $\overrightarrow{\mu}_{X_{1,k}}(x_{1,k})$ being sent in downward direction along edge $X_{1,k}$, which corresponds to sending a uniform distribution, we can calculate $p(x_{1,k})$ as follows¹⁷:

$$p(x_{1,k}) = \gamma \cdot \mu_{\text{tot}}(x_{1,k}) \quad (5.178)$$

¹⁷Refer to Section 4.7 and in particular to (4.63) for an introduction to μ_{tot} .

$$= \gamma \cdot \overrightarrow{\mu}_{X_{1,k}}(x_{1,k}) \cdot \overleftarrow{\mu}_{X_{1,k}}(x_{1,k}) \quad (5.179)$$

$$= \gamma \cdot \overleftarrow{\mu}_{X_{1,k}}(x_{1,k}). \quad (5.180)$$

From (4.64) in Section 4.8 we know how to get the symbol-wise maximum a-posteriori (MAP) estimate $\hat{x}_{1,k}$ for $x_{1,k}$ given $p(x_{1,k})$:

$$\hat{x}_{1,k} = \operatorname{argmax}_{x_{1,k}} p(x_{1,k}). \quad (5.181)$$

Remember that we have cycles in the factor graph. Therefore, there is no guarantee that the marginal $p(x_{1,k})$ is correct.

5.5.2 Improved Readout

We can use the approach that we have just described to estimate the binary source signals $x_{i,k}$. However, the results can be improved. One reason for this might be that the theoretically optimal readout method described above might not be optimal for our case in which we use non-optimal iterative message-passing algorithms.

We have achieved good results by using the following approach to detect firings. For example, to detect the firings for source $i = 1$, we first normalize the messages $\overleftarrow{\mu}_{X_{1,k}}(x_{1,k})$ so that the sum of its two elements for each k becomes one. Then we analyze $\tilde{x}_{1,k} = \overleftarrow{\mu}_{X_{1,k}}(x_{1,k} = 1)$, which is a signal over k . We detect a firing by searching for local maxima in this signal. The main reasons for not using a constant threshold to detect firings are:

- At the firing locations, there is sometimes not one peak, but rather an increased amplitude in the signal $\tilde{x}_{1,k}$ across several samples. This might partly be caused by EMG signal noise and the fact that the message $\overleftarrow{\mu}_{X_{1,k}}(x_{1,k})$ is calculated based on the two messages $\overrightarrow{\mu}_{S_{1,k}}(s_{1,k})$ and $\overleftarrow{\mu}_{S_{1,k+1}}(s_{1,k+1})$. These two messages do not always have their peaks at the same k .
- The amplitudes of local maxima differ.

Using a fixed threshold would detect too many firings¹⁸ if the threshold was too low. On the other hand, if the threshold was too high, too few or no firing would be detected.

¹⁸Self-superpositions of MUAPs from the same source are not allowed. Our readout procedure ensures this by using a window that is almost as long as one MUAP.

5.6 Results for the Known-Constituent Problem

5.6.1 Simulation Set 1: Varying σ_{detect}

We tested our three new algorithms using single superpositions in two-channel signals. Each signal is 70 samples long and consists of four known MUAPs of duration 50. Figure 5.29 and Figure 5.30 show example signals¹⁹. For each data point in the result plots presented next, we actually generated and decomposed 1000 signals (algorithm 1 from Section 5.2) or 10000 signals (algorithm 2 from Section 5.3 and algorithm 3 from Section 5.4). The same MUAPs were used to generate all signals. The integer firing times were generated at random for each signal using a uniform distribution. The firing can occur at discrete times $\{0, 1, 2, \dots, 20\}$, i.e., each MUAP lies completely within each signal. We generated additive white Gaussian noise with a standard deviation of $\sigma_{\text{simul}} = 13$ for each signal.

The gold firing times in Figure 5.29 and in Figure 5.30 are marked by large dots. The firing times that the decomposition algorithm detected are marked by small dots. For this example, since the small dots lie on top of the large dots, the algorithm found all firing times without making a mistake. In fact, all example signals in this chapter were completely correctly decomposed by at least one of our new algorithms.

Figure 5.31 shows the fraction of completely correct decompositions vs. the decomposition parameter σ_{detect} . σ_{detect} is the standard deviation of the AWGN noise model of the decomposition algorithm.

¹⁹Note that the MUAPs from different sources are very similar in shape, which makes decomposing the signals difficult.

Correct decomposition of a signal

A simulated signal is completely correctly decomposed if the estimated source signals $\hat{X}_{i,k}$ match the source signals $X_{i,k}$ from the simulation^a. Therefore, no missed detections and no additional detections are allowed^b. Note also that the fraction of correctly found MUAPs is usually higher than the fraction of completely correct decompositions.

^aCompare to Figure 4.2.

^bA distance of up to two sampling intervals between a gold firing and a detected firing is allowed.

Figure 5.32 shows the relative RMSE r vs. the decomposition parameter σ_{detect} . The relative RMSE r is defined as the RMSE for the decomposition result over the RMSE for the correct decomposition:

$$r = \frac{e_{\text{detect}}}{e_{\text{simul}}}. \quad (5.182)$$

The root mean squared error (RMSE) e_{detect} between a simulated multi-channel EMG signal Y and a reconstructed multi-channel EMG signal \hat{Y} is defined as

$$e_{\text{detect}} = \sqrt{\frac{1}{L_s N_c} \sum_{k=0}^{L_s-1} \sum_{j=1}^{N_c} (Y_{j,k} - \hat{Y}_{j,k})^2}, \quad (5.183)$$

where L_s is the length of the EMG signal, and N_c is the number of channels.

Similar to (4.22), the components of the reconstructed EMG signal \hat{Y} are

$$\hat{Y}_{j,k} = \sum_{i=1}^{N_{\text{src}}} \sum_{\ell=0}^{M_i} \hat{X}_{i,k-\ell} \cdot h_{i,j,\ell}. \quad (5.184)$$

The RMSE for the correct decomposition e_{simul} is defined as

$$e_{\text{simul}} = \sqrt{\frac{1}{L_s N_c} \sum_{k=0}^{L_s-1} \sum_{j=1}^{N_c} (Y_{j,k} - \tilde{Y}_{j,k})^2} \quad (5.185)$$

$$= \sqrt{\frac{1}{L_s N_c} \sum_{k=0}^{L_s-1} \sum_{j=1}^{N_c} W_{j,k}^2}. \quad (5.186)$$

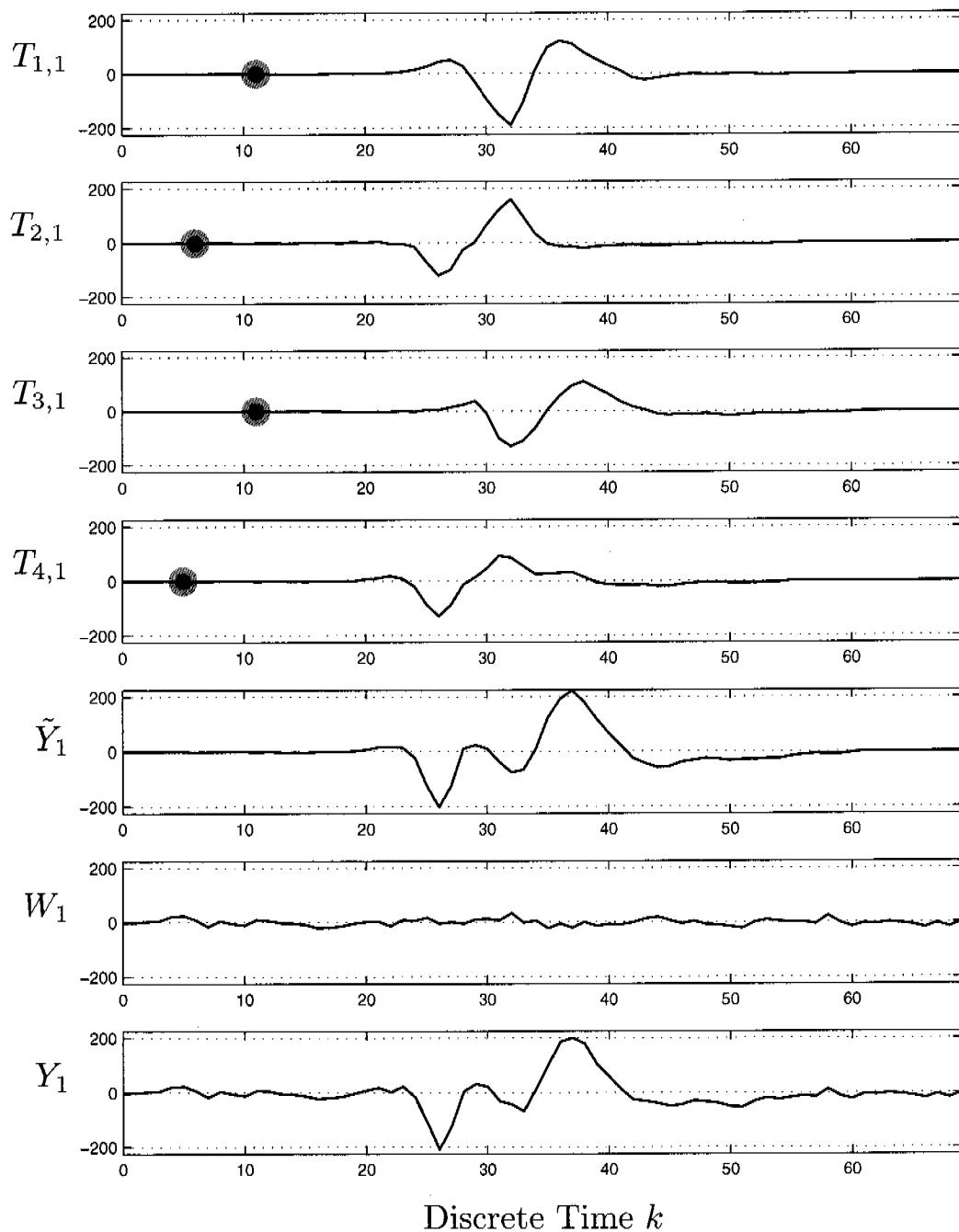


Figure 5.29: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 13$, and the superposition with noise Y_1 . Compare to Figure 4.2.

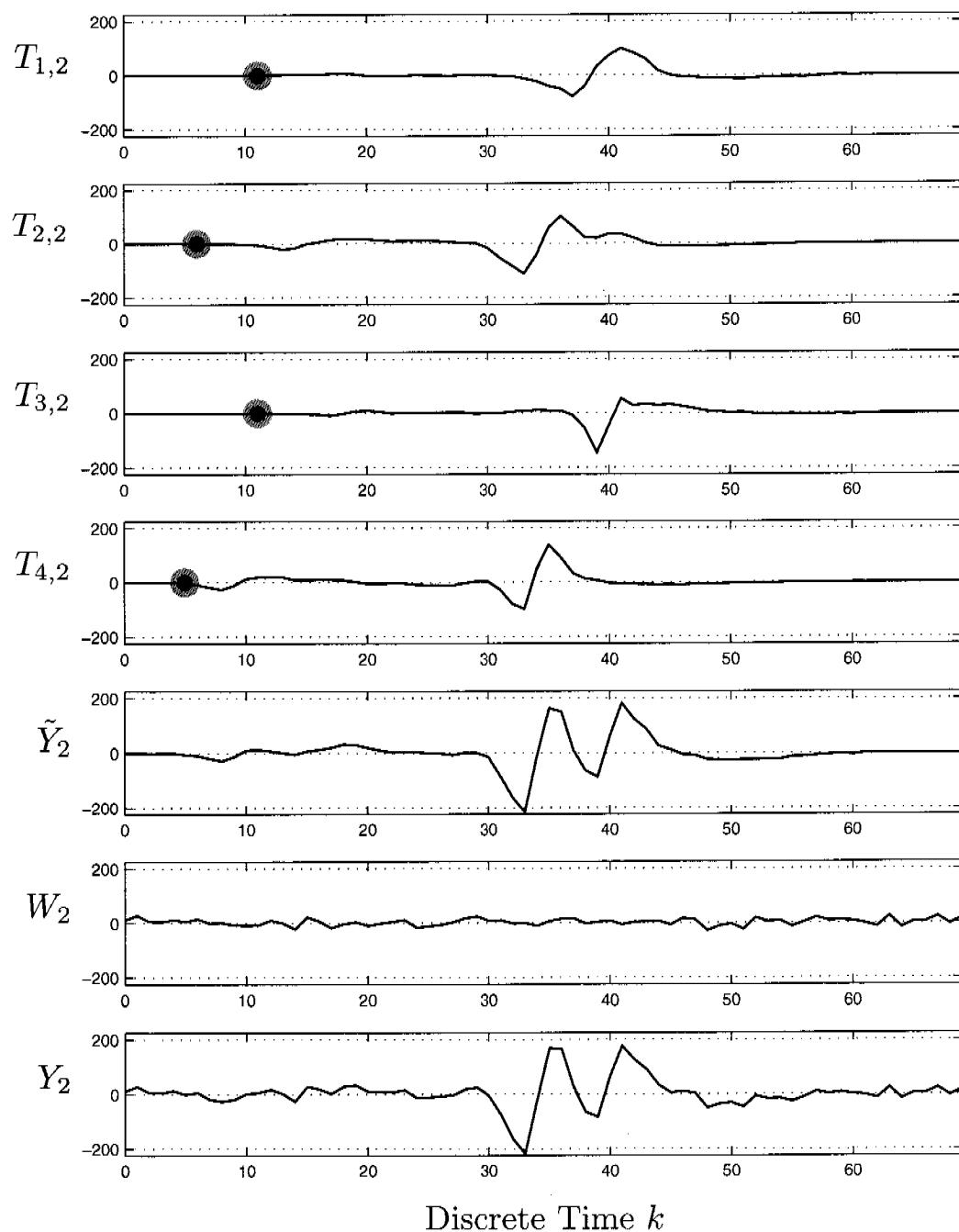


Figure 5.30: As 5.29, but for the second channel.

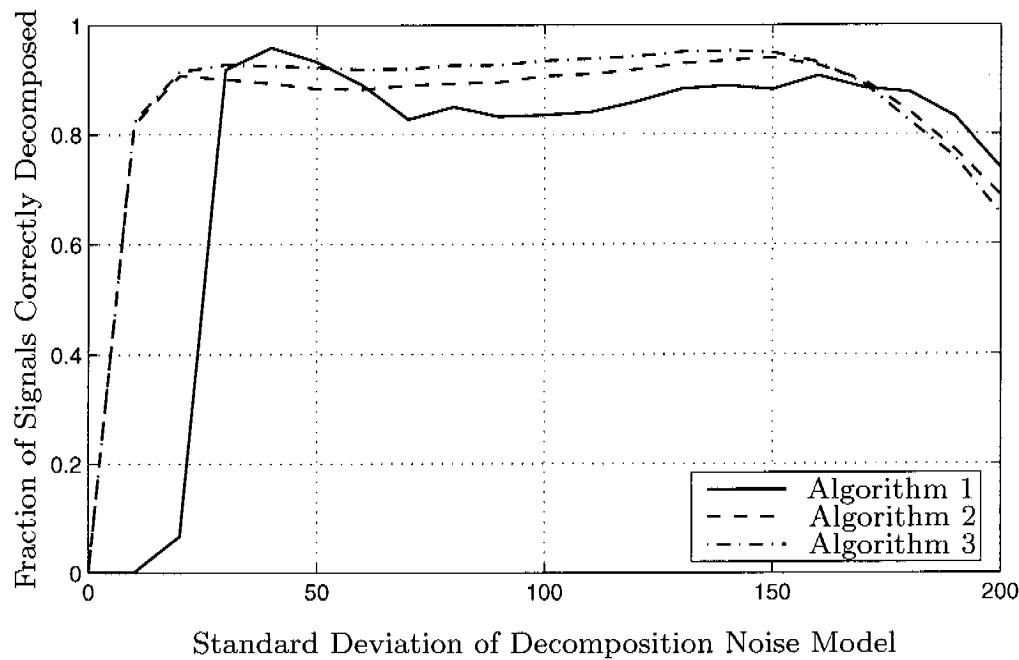


Figure 5.31: Fraction of completely correctly decomposed superpositions vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}} = 13$.

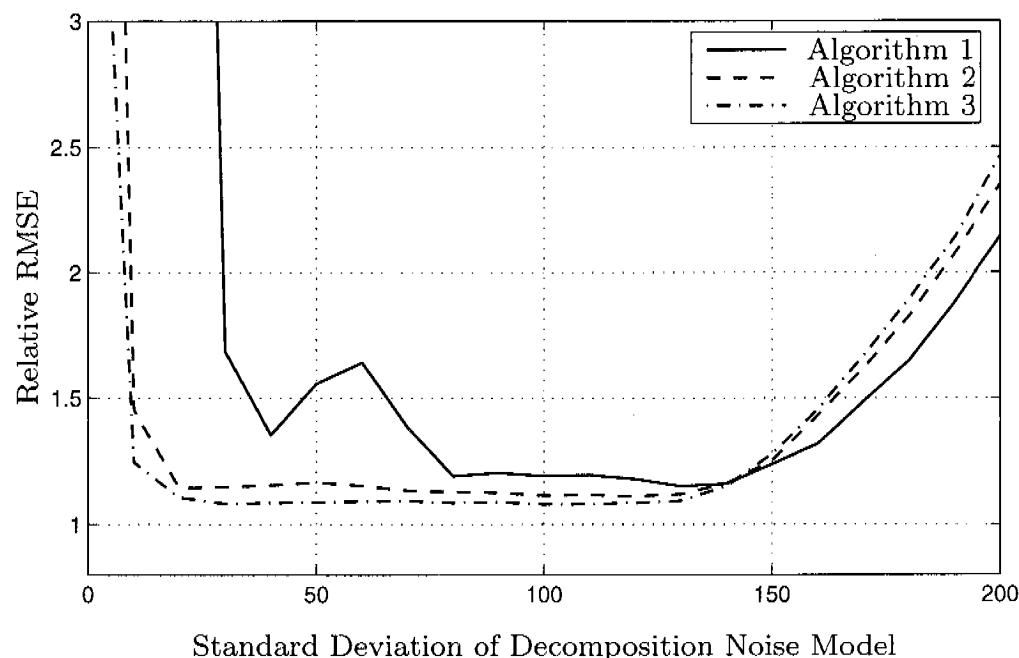


Figure 5.32: Relative RMSE vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}}=13$.

Figure 5.33 and Figure 5.34 show simulated signals for the case of a noise standard deviation $\sigma_{\text{simul}} = 30$. The corresponding decomposition results are given in Figure 5.35 and Figure 5.36.

Figure 5.37 and Figure 5.38 show simulated signals for the case of a noise standard deviation $\sigma_{\text{simul}} = 50$. The corresponding decomposition results are given in Figure 5.39 and Figure 5.40.

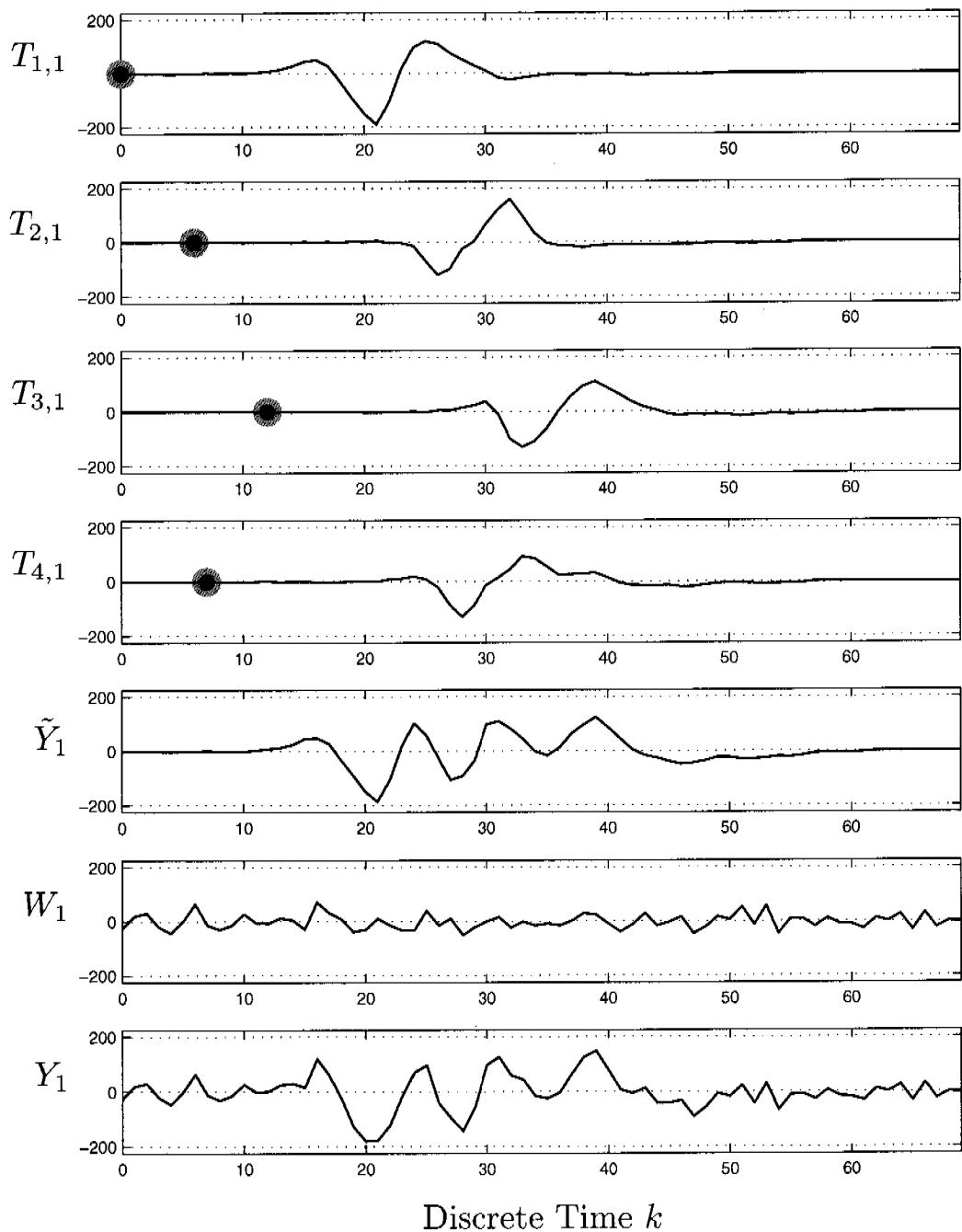


Figure 5.33: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 30$, and the superposition with noise Y_1 . Compare to Figure 4.2.

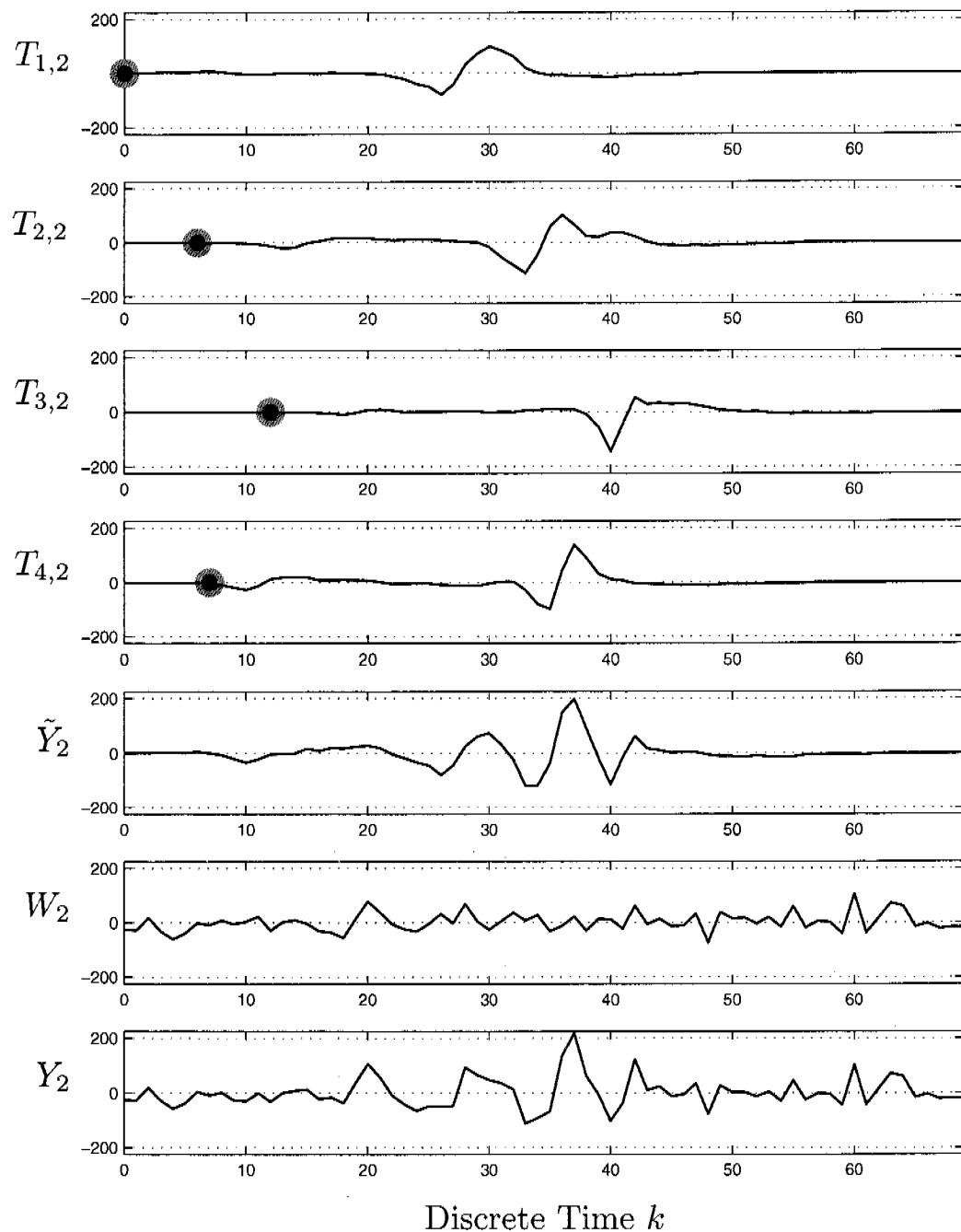


Figure 5.34: As 5.33, but for the second channel.

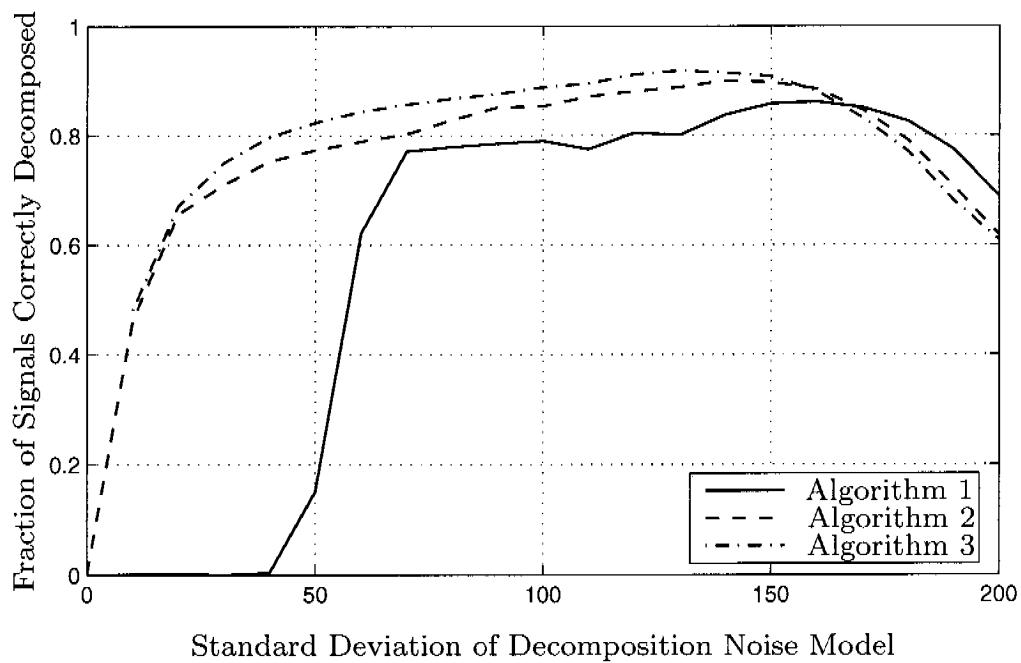


Figure 5.35: Fraction of completely correctly decomposed superpositions vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}} = 30$.

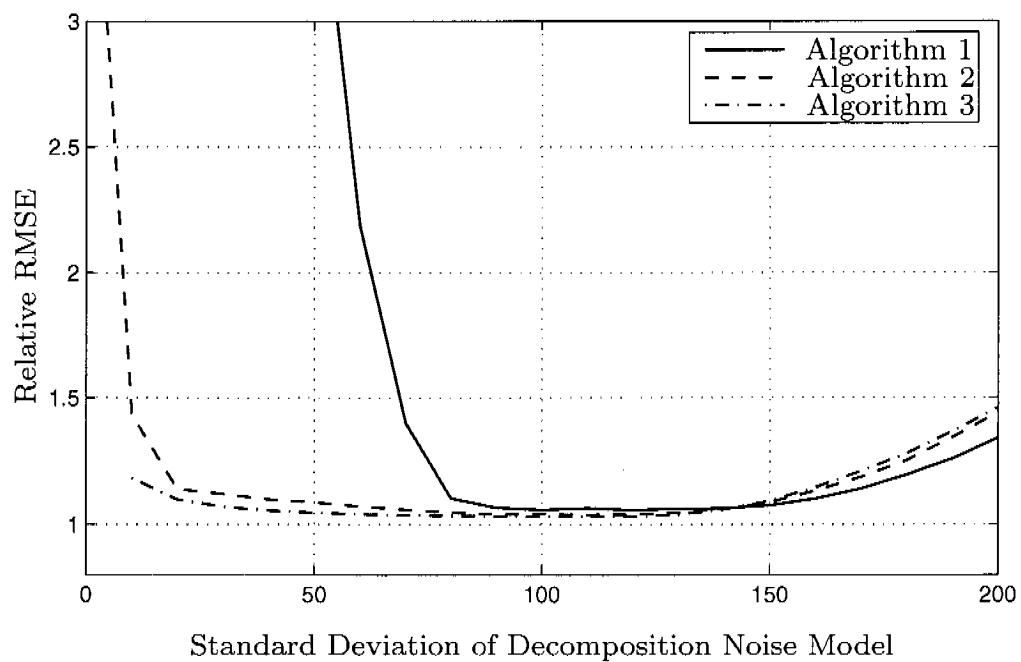


Figure 5.36: Relative RMSE vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}}=30$.

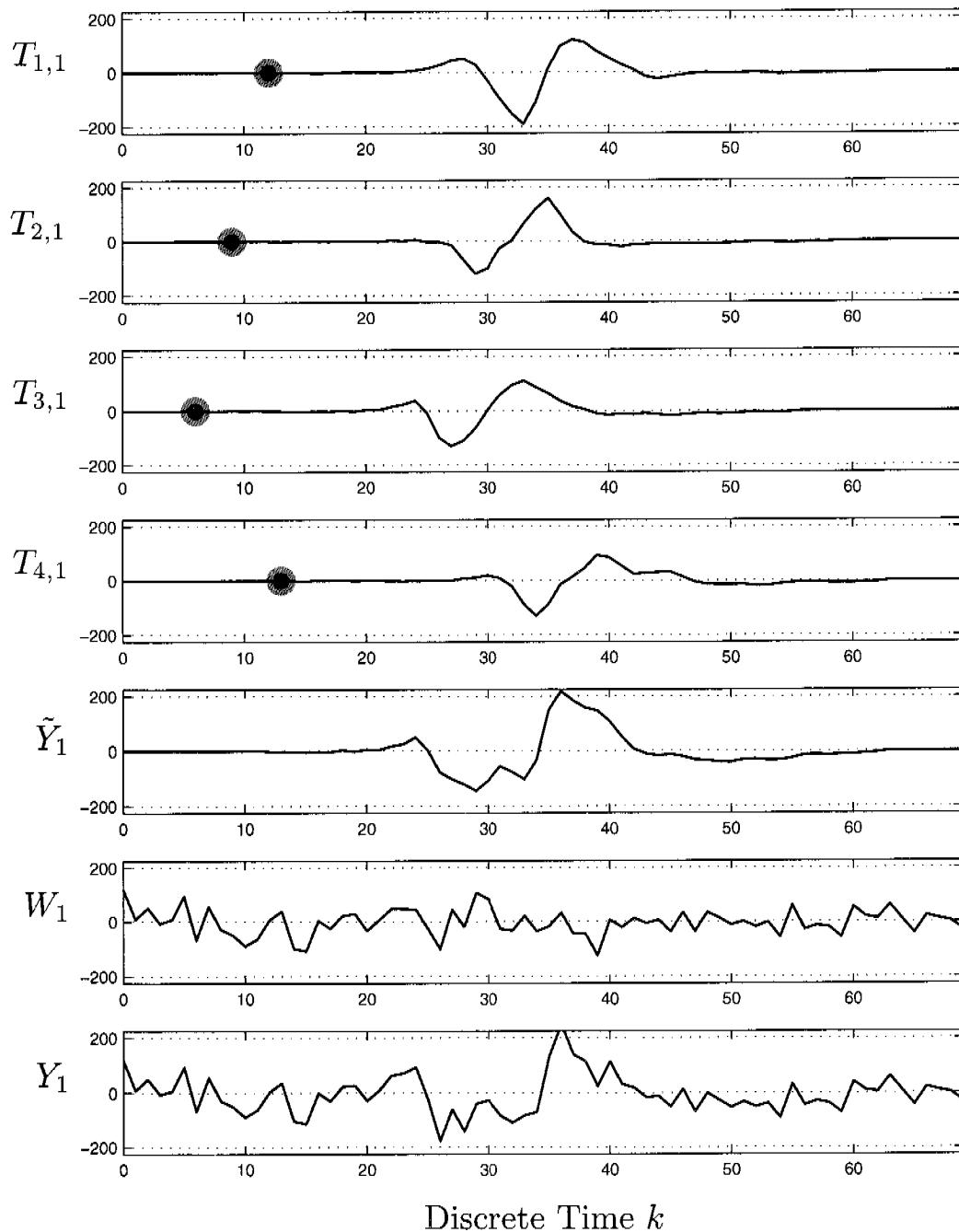


Figure 5.37: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 50$, and the superposition with noise Y_1 . Compare to Figure 4.2.

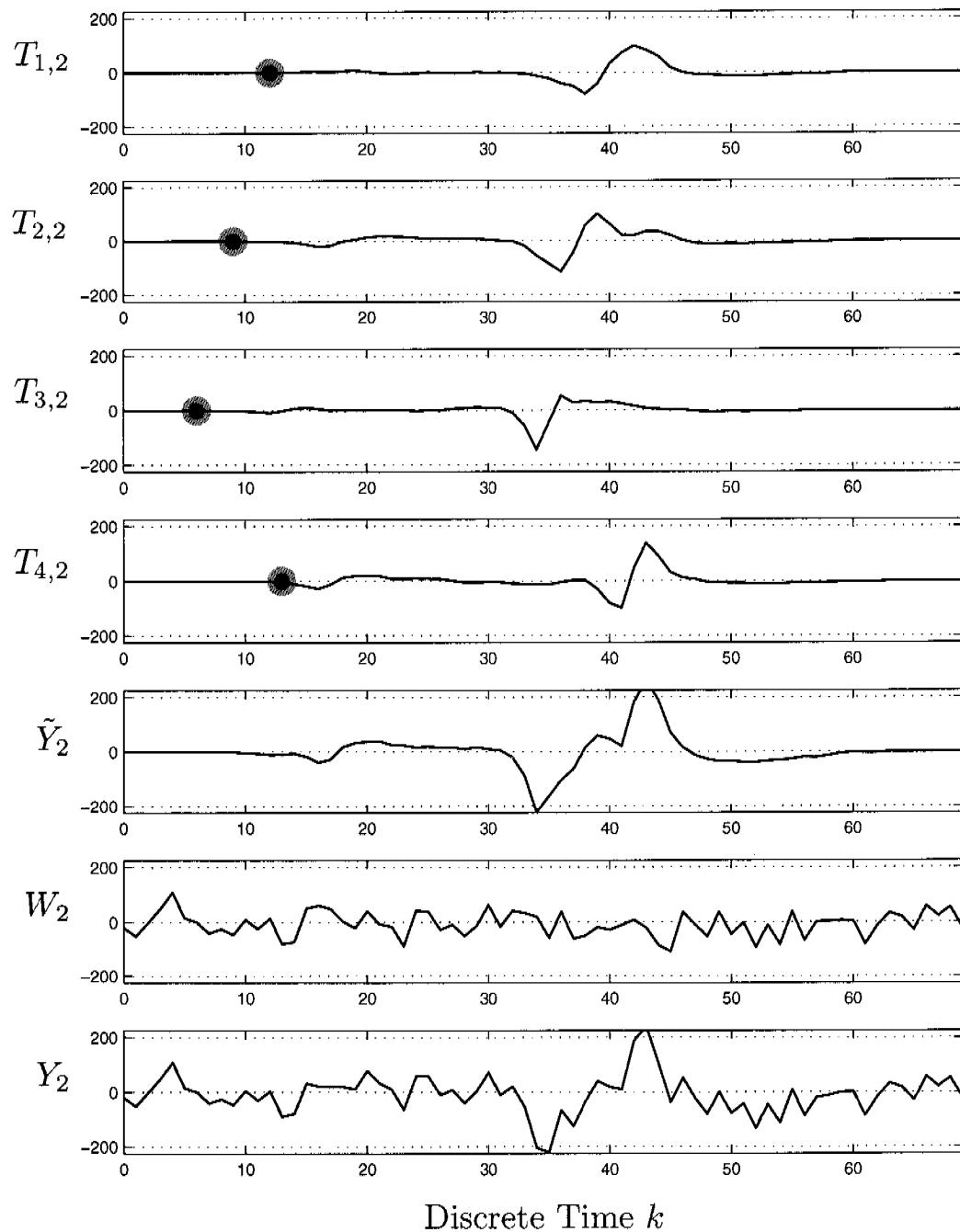


Figure 5.38: As 5.37, but for the second channel.

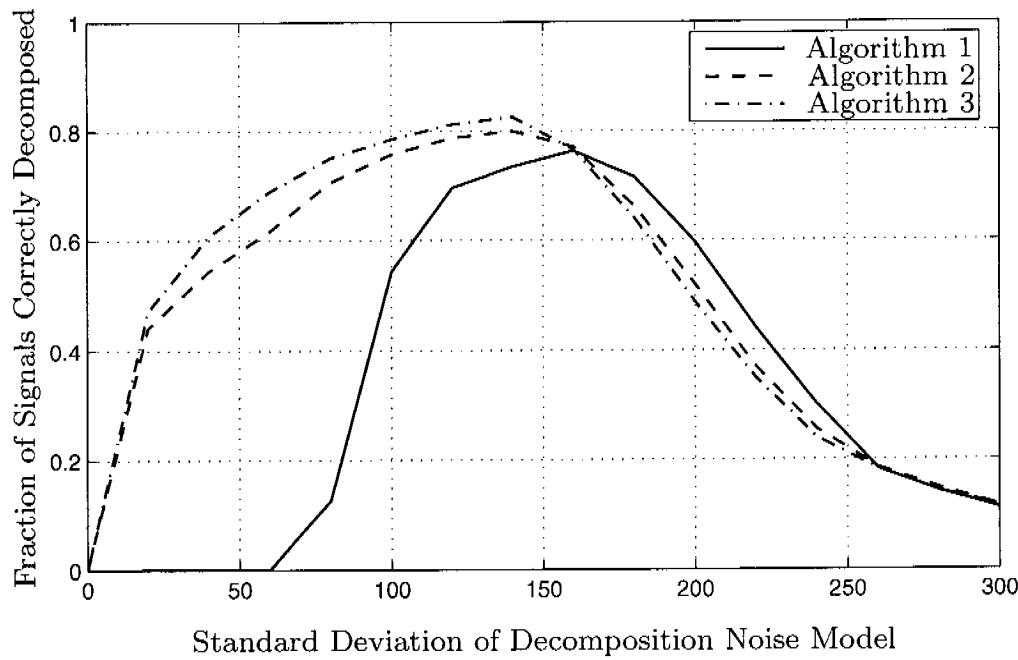


Figure 5.39: Fraction of completely correctly decomposed superpositions vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}} = 50$.

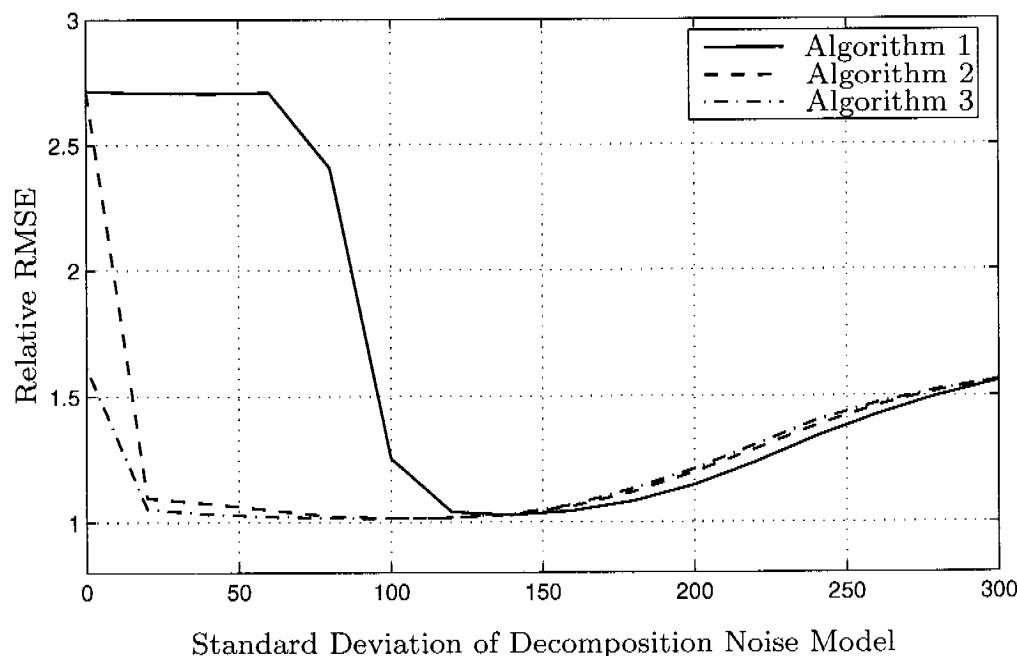


Figure 5.40: Relative RMSE vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}}=50$.

5.6.2 Simulation Set 2: Varying the Number of Iterations

Next we consider how the decomposition performance depends on the number of iterations used.

Figure 5.41 shows the fraction of completely correct decompositions vs. σ_{detect} for three different number of iterations. Figure 5.42 shows the corresponding plot for the relative RMSE. Only algorithm 3 was used here. An exemplary two-channel signal with the here used $\sigma_{\text{simul}} = 50$ can be seen in Figure 5.37 and in Figure 5.38.

Figure 5.43 shows the fraction of completely correct decompositions vs. the number of iterations for the three algorithms. Figure 5.44 shows the corresponding plot for the relative RMSE. An exemplary two-channel signal with the here used $\sigma_{\text{simul}} = 50$ can be seen in Figure 5.37 and in Figure 5.38.

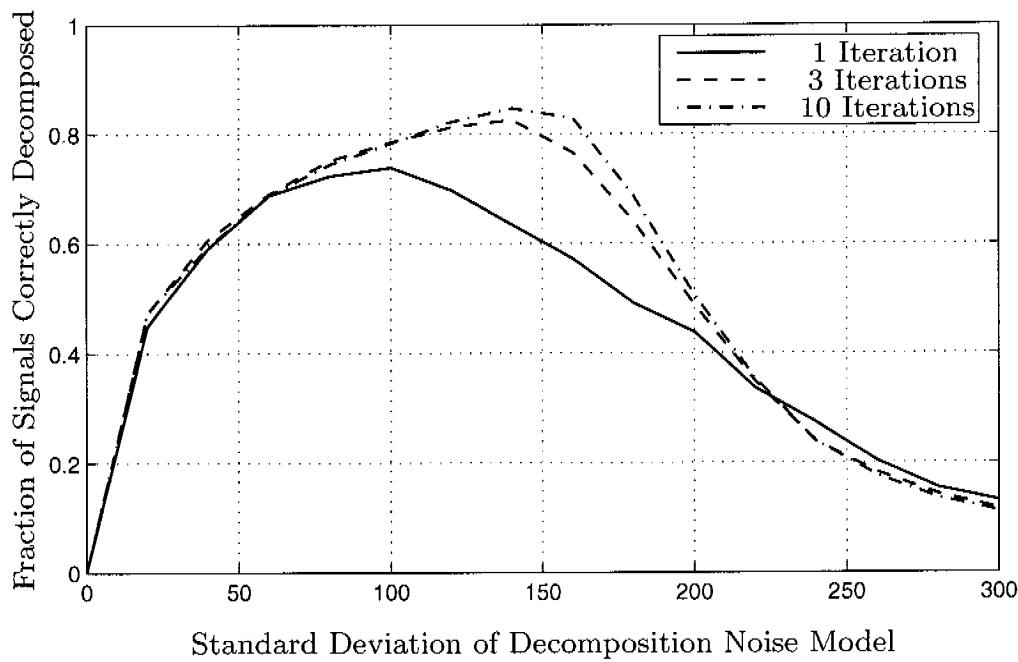


Figure 5.41: Fraction of completely correctly decomposed superpositions vs. the detection parameter σ_{detect} for 1, 3, and 10 iterations.

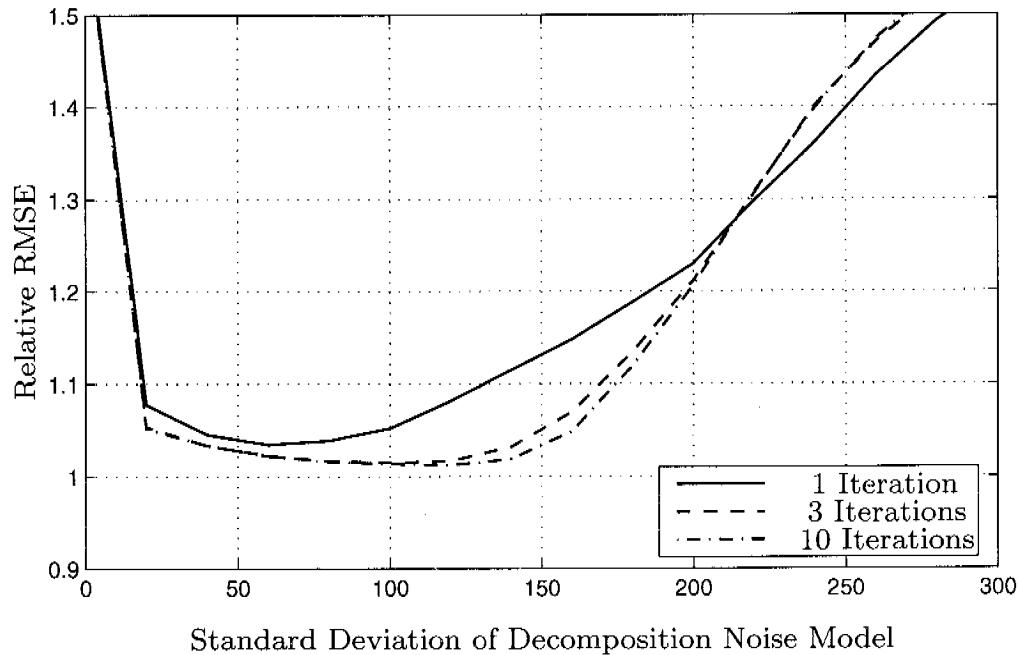


Figure 5.42: Relative RMSE vs. the detection parameter σ_{detect} for 1, 3, and 10 iterations.

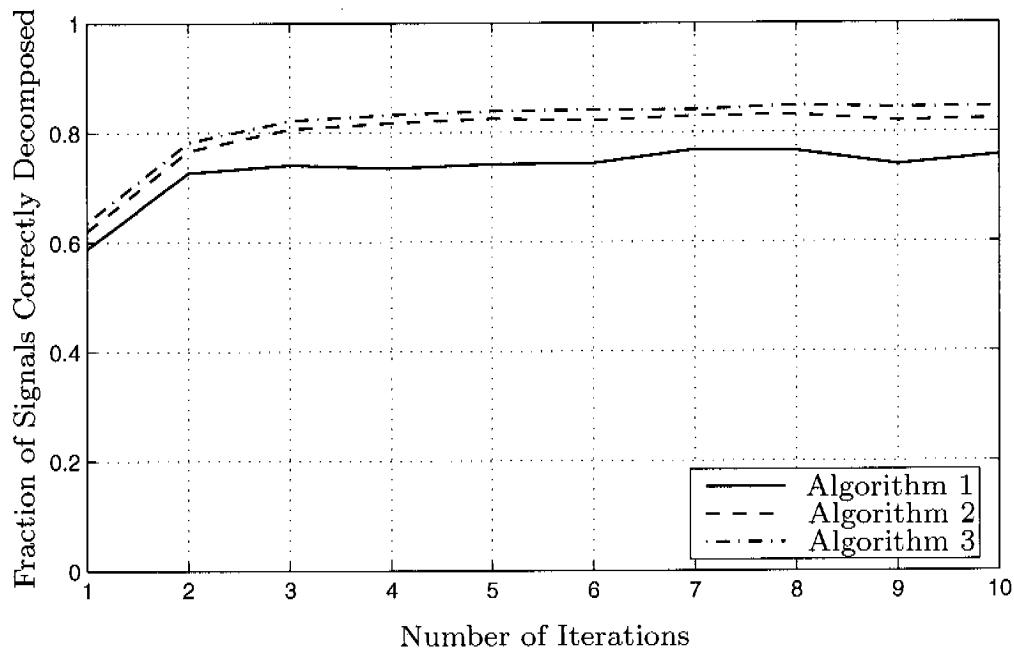


Figure 5.43: Fraction of completely correctly decomposed superpositions vs. the number of iterations.

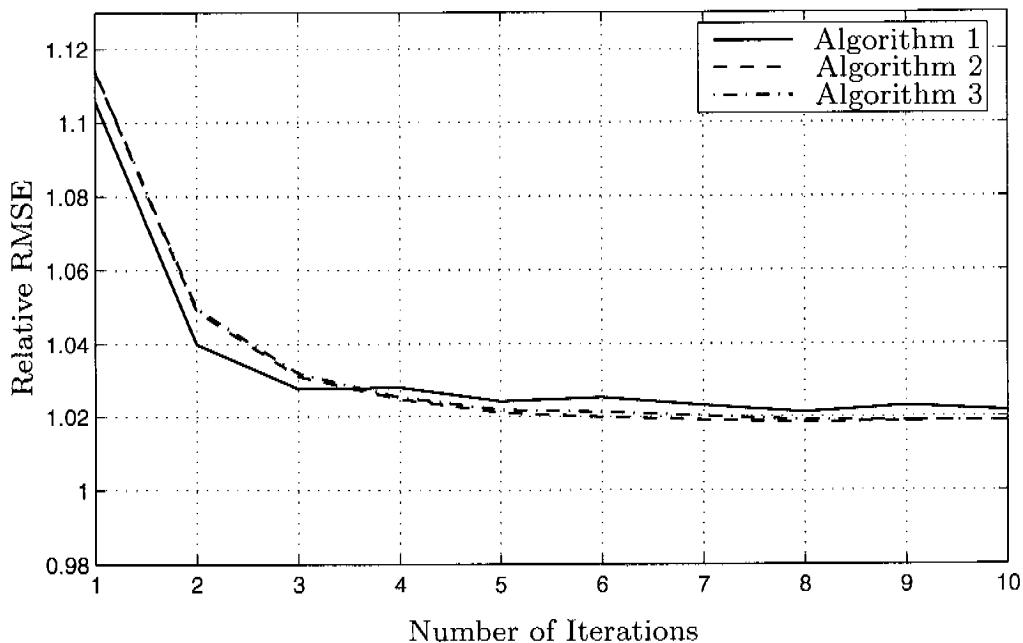


Figure 5.44: Relative RMSE vs. the number of iterations.

5.7 Computation Times

In this section we show how the computation time depends on the number of sources and the duration of the MUAPs. All time measurements were performed on a standard personal computer²⁰.

To calculate the average time per signal decomposition, we decomposed 100 test signals for each data point. The tests were done based on the previously described known-constituent problem. The noise standard deviation of the signals is $\sigma_{\text{simul}} = 10$. An exemplary two-channel test signal can be seen in Figure 5.53 and in Figure 5.54. Note that we used a different set of MUAPs here than we used in the previous section.

Exactly one iteration was used to decompose each test signal. The time was only included in the calculation of the average time per signal if the decomposition result was correct in terms of the firing times.

Figure 5.45 shows the average time per correctly decomposed signal in seconds vs. the number of sources for algorithm 1. Figure 5.46 shows the corresponding plots for algorithms 2 and 3.

Next, we plotted the average time per correctly decomposed signal vs. the duration of four MUAPs. For this we changed the MUAP durations between 25 and 50. As before, exactly one iteration was used to decompose each test signal, $\sigma_{\text{simul}} = 10$, and only correct decompositions were considered. Figure 5.47 shows the results for algorithm 1 and Figure 5.48 for algorithms 2 and 3.

²⁰Dell Dimension DIM4600 Desktop, Intel Pentium 4 processor with 2.66 GHz, 1 GB RAM, Windows XP Professional Version 2002 with Service Pack 2. The decomposition algorithms were implemented in Java as described in Appendix D and Appendix E.

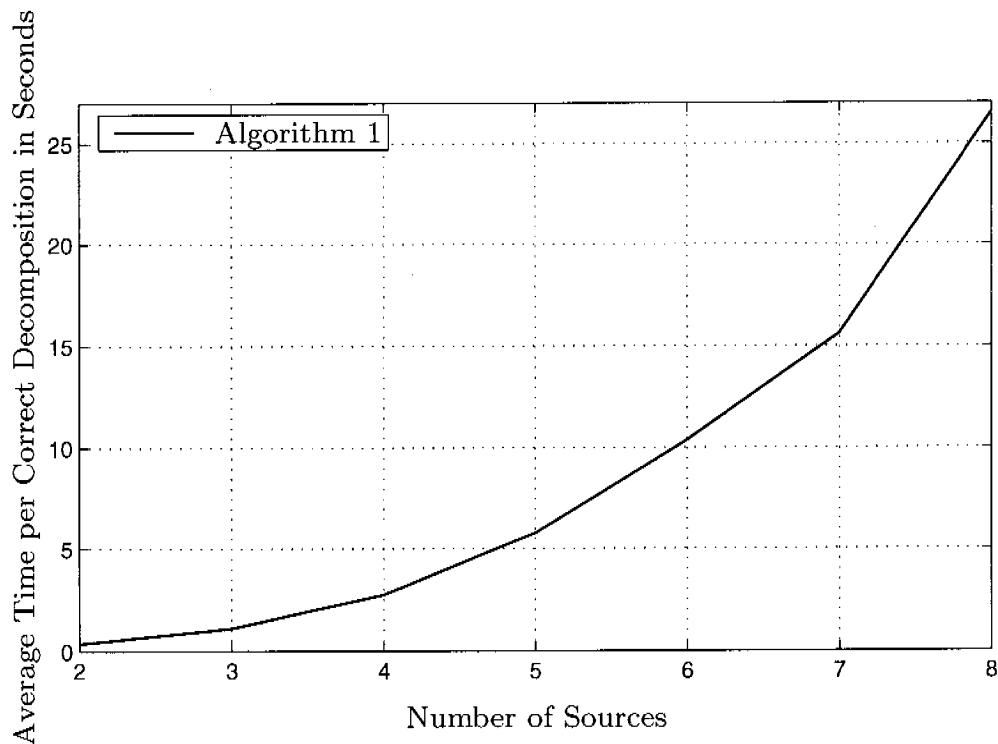


Figure 5.45: Average time for correctly decomposing a signal vs. the number of sources for algorithm 1.

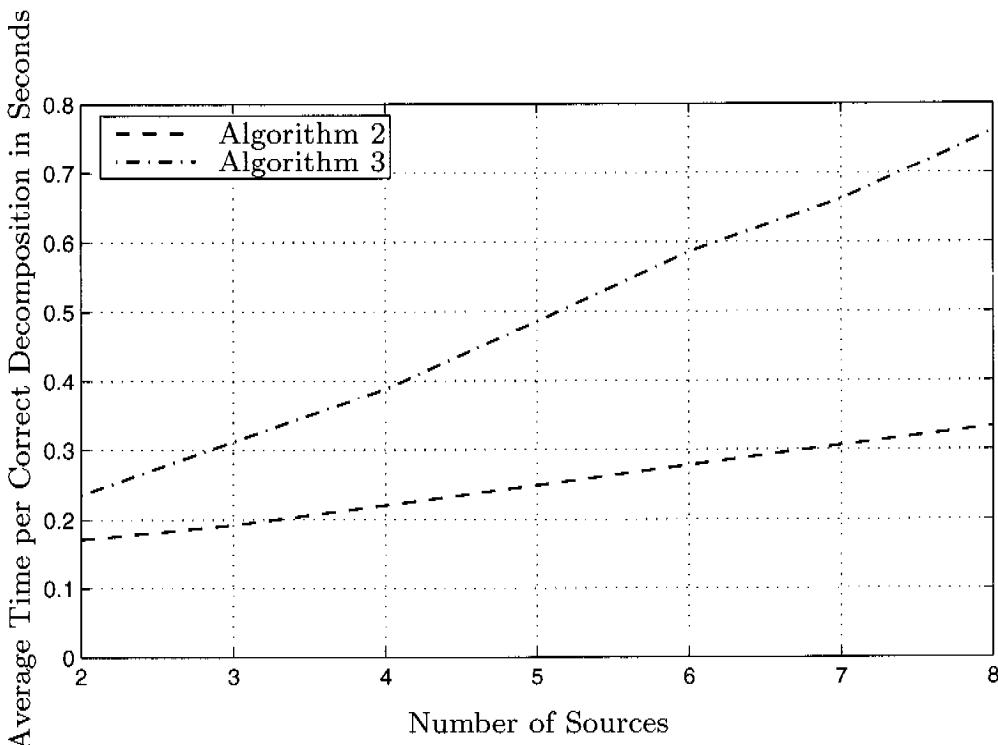


Figure 5.46: Average time for correctly decomposing a signal vs. the number of sources for algorithms 2 and 3.

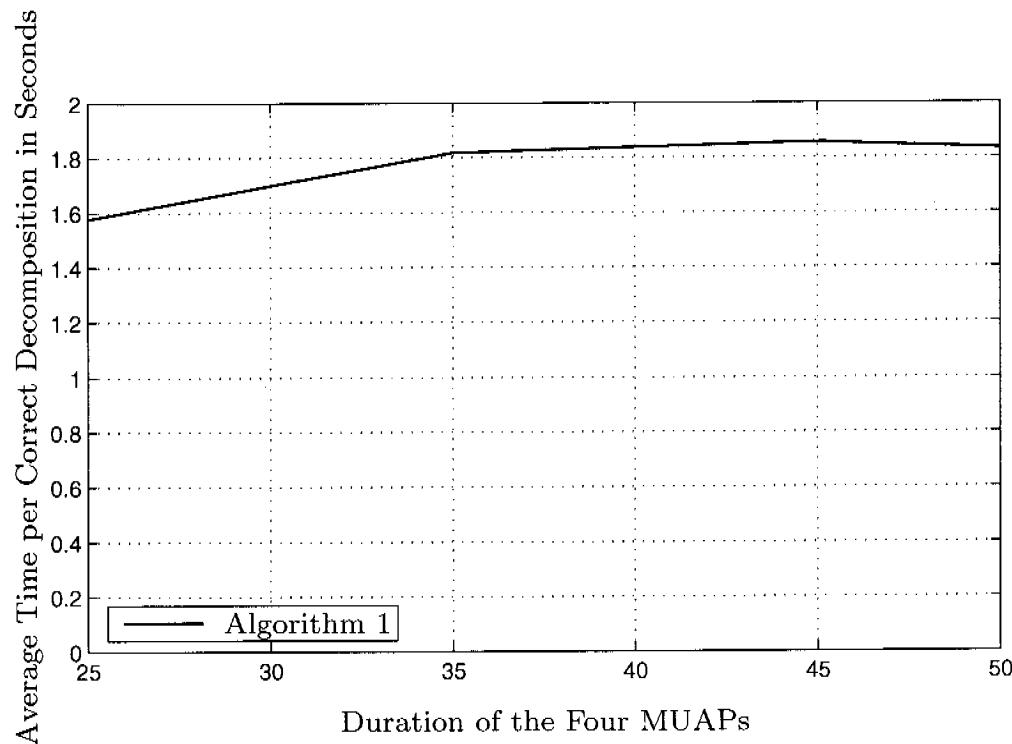


Figure 5.47: Average time for correctly decomposing a signal vs. the duration of the MUAPs for algorithm 1.

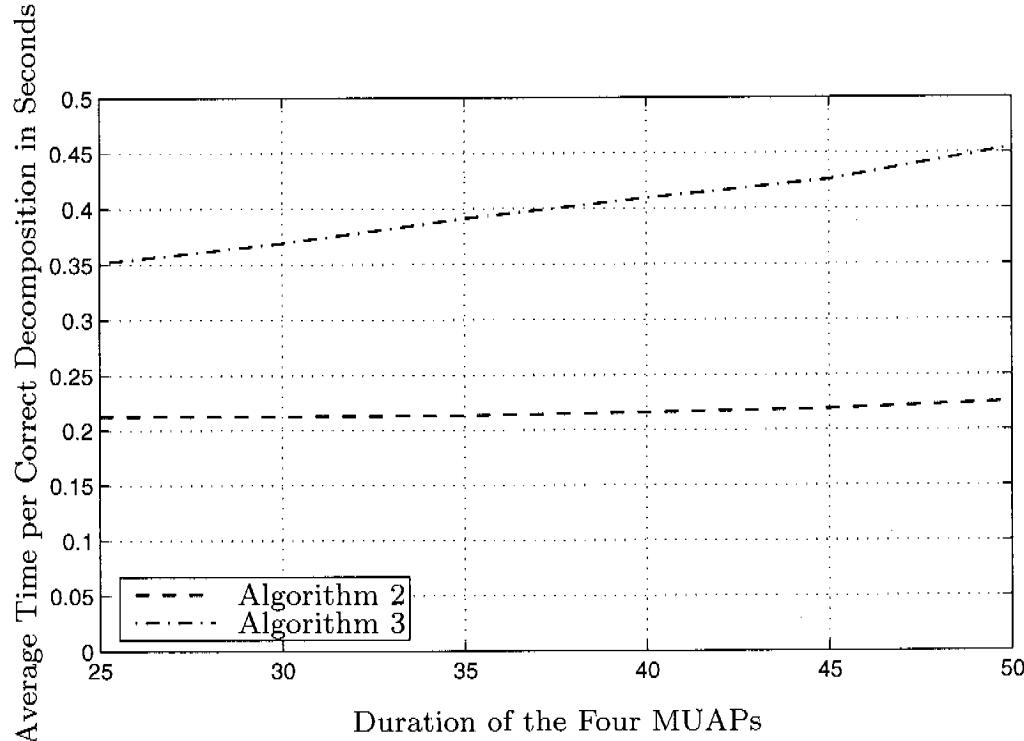


Figure 5.48: Average time for correctly decomposing a signal vs. the duration of the MUAPs for algorithms 2 and 3.

5.8 Improvements

The three algorithms described in Sections 5.2, 5.3, and 5.4 work, i.e., they can resolve superpositions. However, we have developed several methods to improve the performance and the speed of our decomposition algorithms. Some of them are outlined in this section.

5.8.1 Trying Different Noise Levels

As we have seen, the standard deviation σ_{detect} of the AWGN noise models in our factor graphs has a great influence on the performance of our EMG signal decomposition algorithms. It is usually not the average noise power of the EMG signal that yields the best results, although this measure can give a good starting point to determine a suitable σ_{detect} .

We could improve our decomposition results by partly²¹ performing the decomposition process several times with different, pre-defined and σ_{simul} -dependent noise parameters σ_{detect} . The decomposition results were compared based on the RMSE between the EMG signal and the reconstructed signal. The decomposition result yielding the smallest RMSE was finally selected.

5.8.2 Exception Throwing

When the noise variance of our AWGN noise model is set to a low value, numerical problems during the decomposition can occur. For example, single message values may become very small even when using `double` variables in Java.

Our Java algorithms check for small values²², `NaN`²³, and `Inf` values when

²¹The algorithm compares the results already after the first iteration and performs all iterations only for the noise variance yielding the best result after the first iteration.

²²Apart from a reduced precision, small values in discrete messages can lead to zero values, which often propagate and make all values of a message zero. In practice, a problem is identified if a message value of a discrete message is smaller than three times the smallest positive nonzero value of type `double`, which is `Double.MIN_VALUE = 2-1074 = 4.9 · 10-324`.

²³In agreement with the IEEE 754 standard, Java uses constants for representing special values, including positive infinity (`+Inf`), negative infinity (`-Inf`), and non-numeric results (`NaN`, not a number).

normalizing messages after each update. A special exception is thrown if problematic messages are detected. The exception is caught and a higher noise variance is automatically set before the algorithm tries the decomposition again. Trying different noise variances leads to improved results. Throwing an exception and trying another noise variance *immediately* in the case of numerical problems leads to shorter decomposition times.

5.9 Improved Results for the Known-Constituent Problem

5.9.1 Simulation Set 3: Varying σ_{simul}

Figure 5.51 and Figure 5.52 show results for the case where we used the improvements described in Section 5.8.1 and Section 5.8.2. Exemplary two-channel EMG signals with four sources firing can be seen in Section 5.6.1 and for the extreme case of $\sigma_{\text{simul}} = 100$ in Figure 5.49 and Figure 5.50.

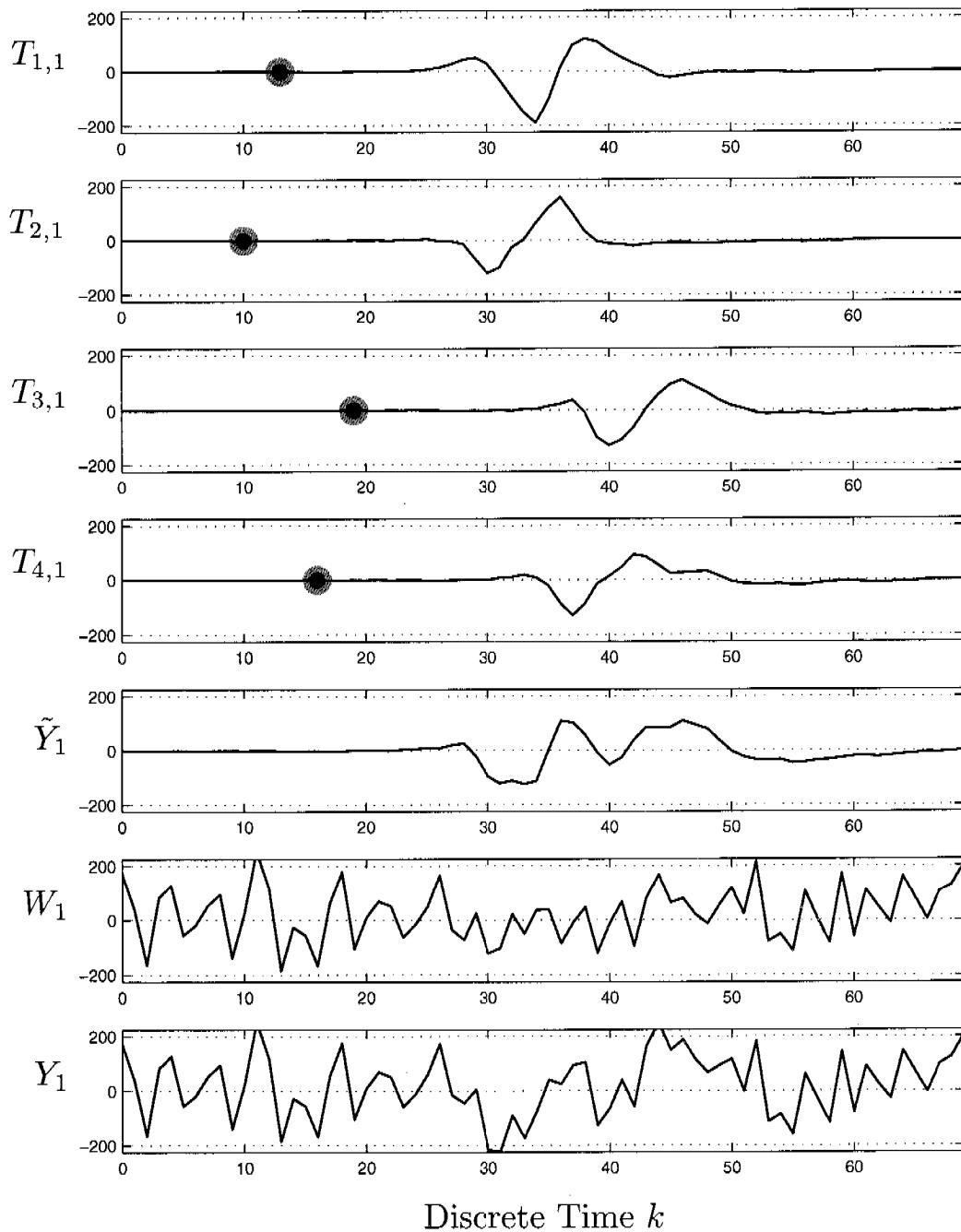
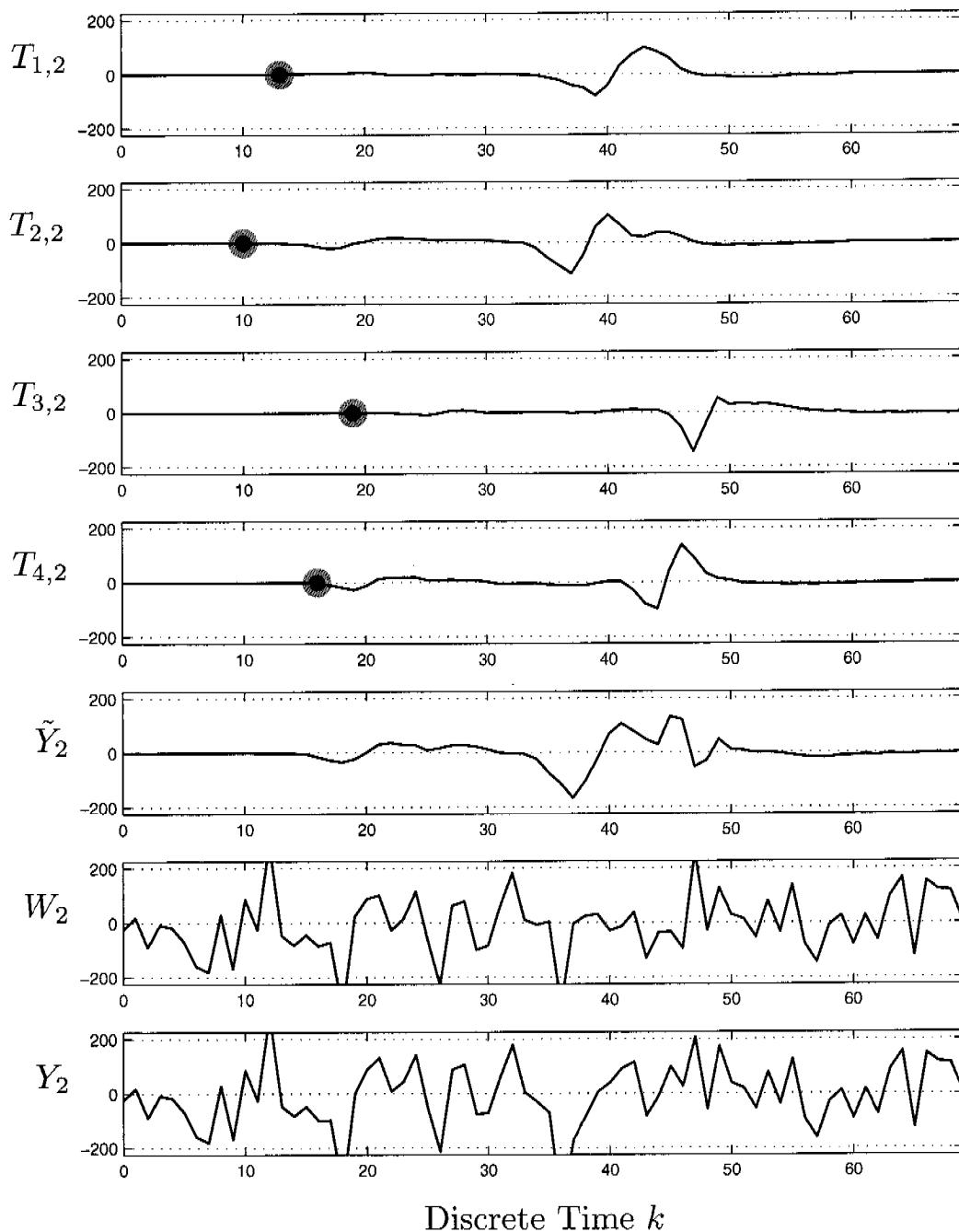


Figure 5.49: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 100$, and the superposition with noise Y_1 . Compare to Figure 4.2.

**Figure 5.50:** As 5.49, but for the second channel.

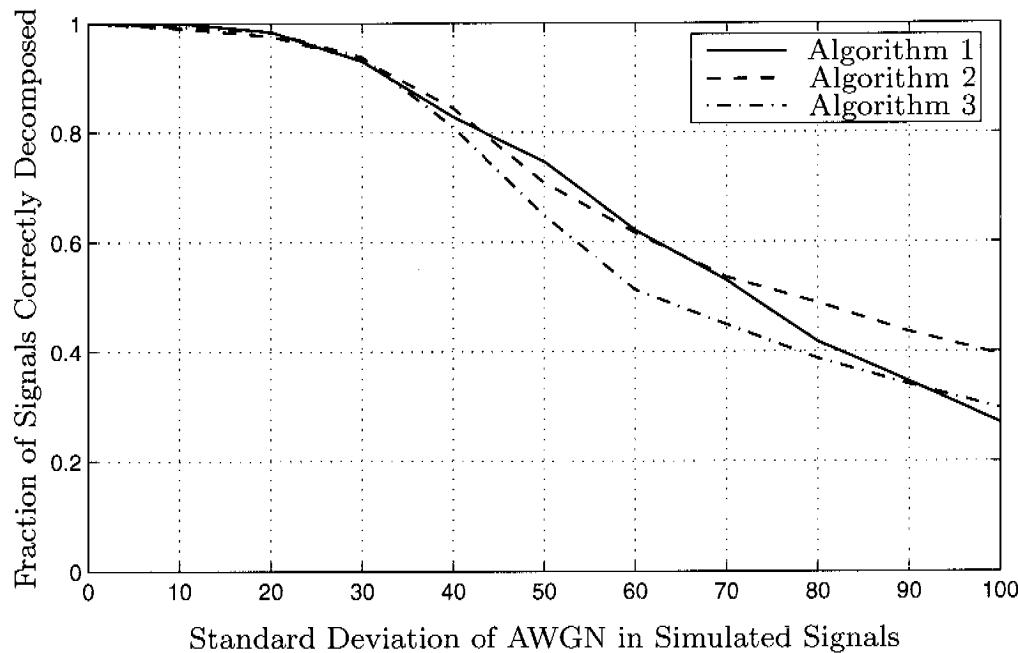


Figure 5.51: Fraction of completely correctly decomposed superpositions vs. the standard deviation of the AWGN in the simulated signals.

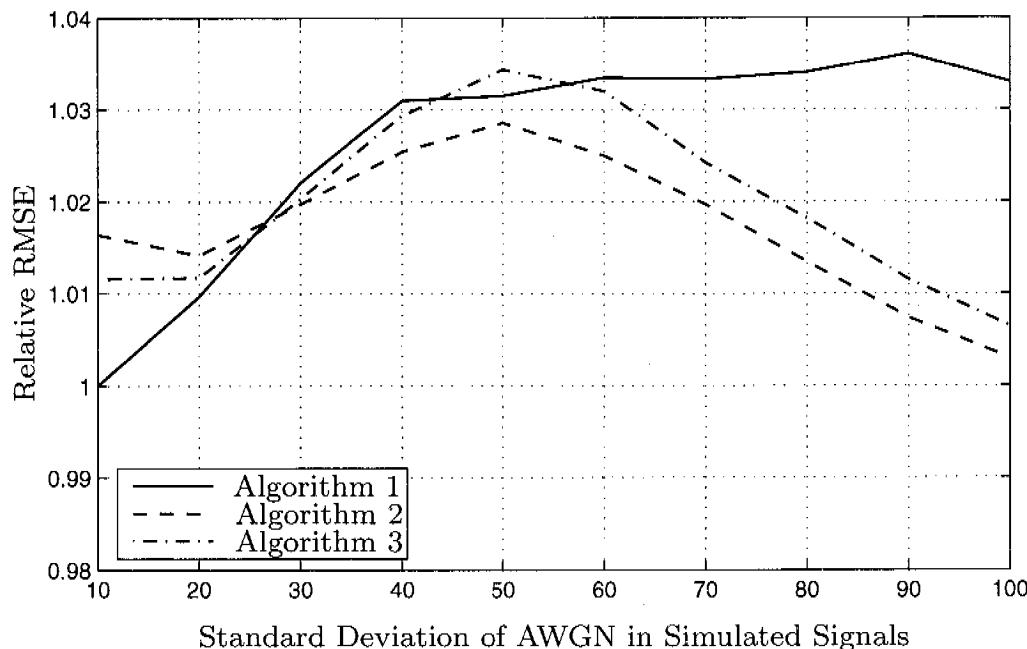


Figure 5.52: Relative RMSE vs. the standard deviation of the AWGN in the simulated signals.

5.9.2 Simulation Set 4: Varying the Number of Sources

In this section we present results for a varying number of sources contributing to a superposition. As before, we used the improvements described in Section 5.8.1 and Section 5.8.2 and a maximum number of three iterations. Note that, for each signal, n sources with their corresponding MUAP shapes were chosen at random from a set of $N_{\text{src}} = 8$ sources. The order of the sources were also chosen at random.

Example signals for the case of $\sigma_{\text{simul}} = 10$ and $n = 8$ sources are given in Figure 5.53 and in Figure 5.54. The corresponding results can be found in Figure 5.55 and Figure 5.56.

Example signals for the case of $\sigma_{\text{simul}} = 30$ and $n = 8$ sources are given in Figure 5.57 and in Figure 5.58. The corresponding results can be found in Figure 5.59 and Figure 5.60.

Example signals for the case of $\sigma_{\text{simul}} = 50$ and $n = 8$ sources are given in Figure 5.61 and in Figure 5.62. The corresponding results can be found in Figure 5.63 and Figure 5.64.

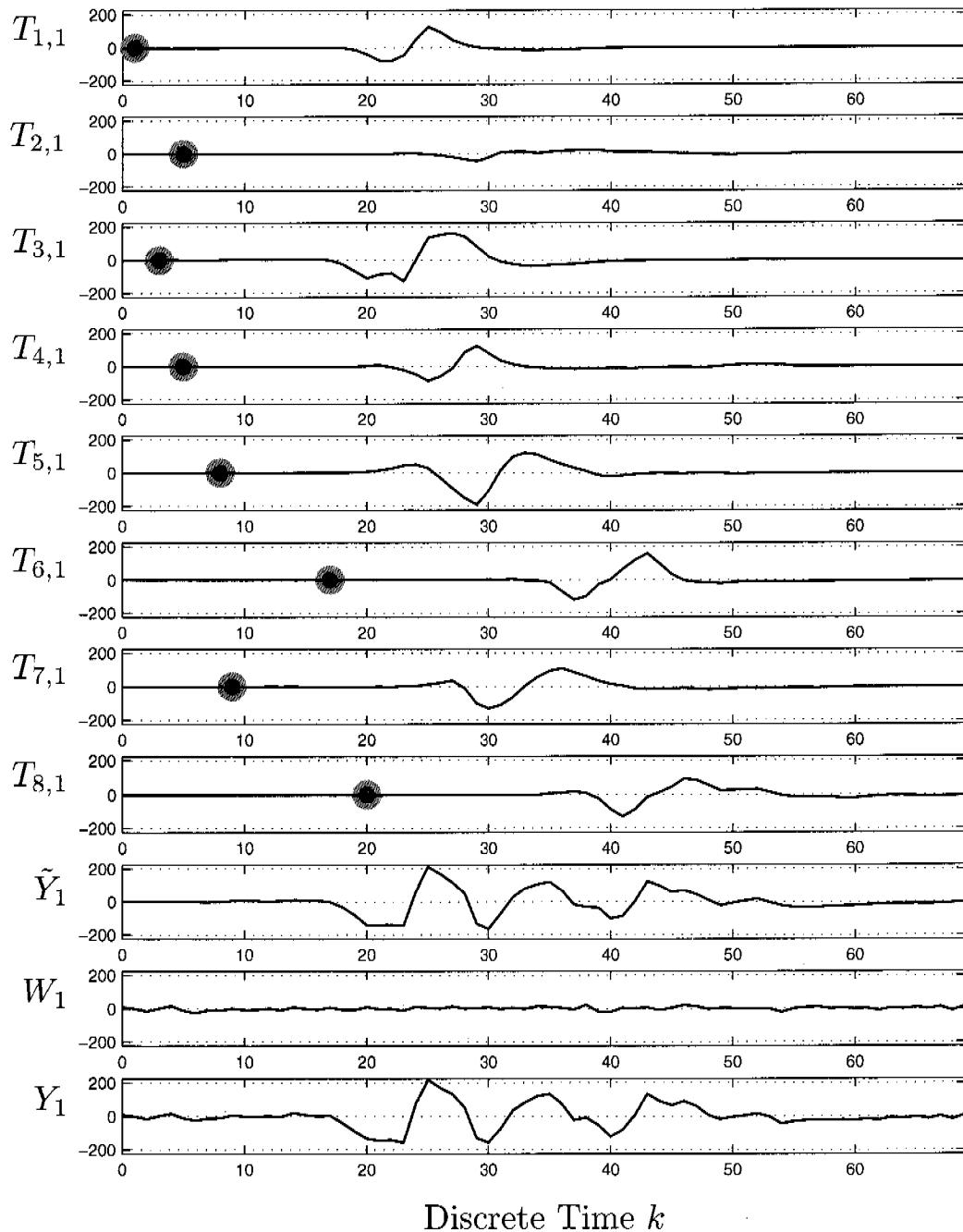


Figure 5.53: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4, 5, 6, 7, 8\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 10$, and the superposition with noise Y_1 . Compare to Figure 4.2.

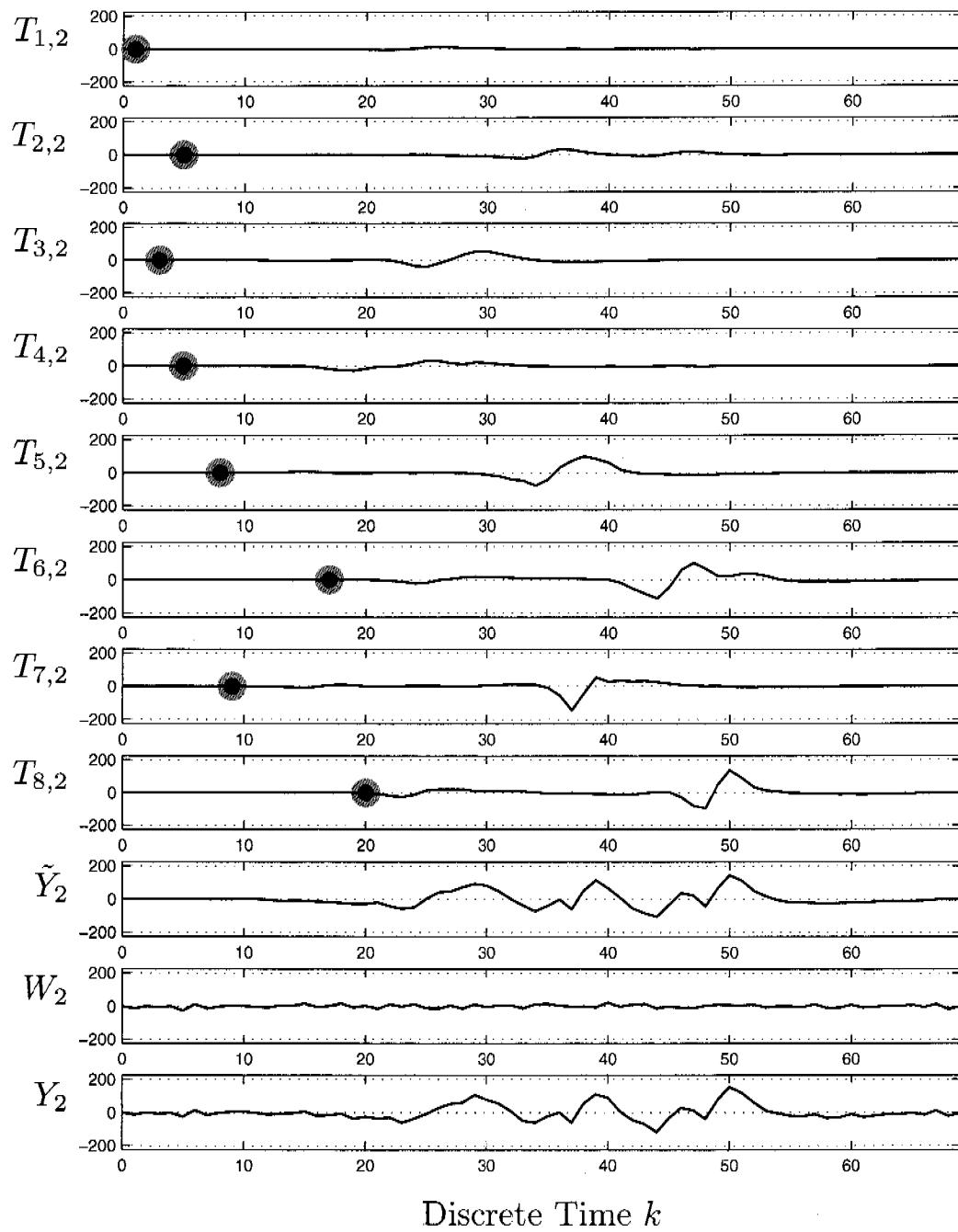


Figure 5.54: As 5.53, but for the second channel.

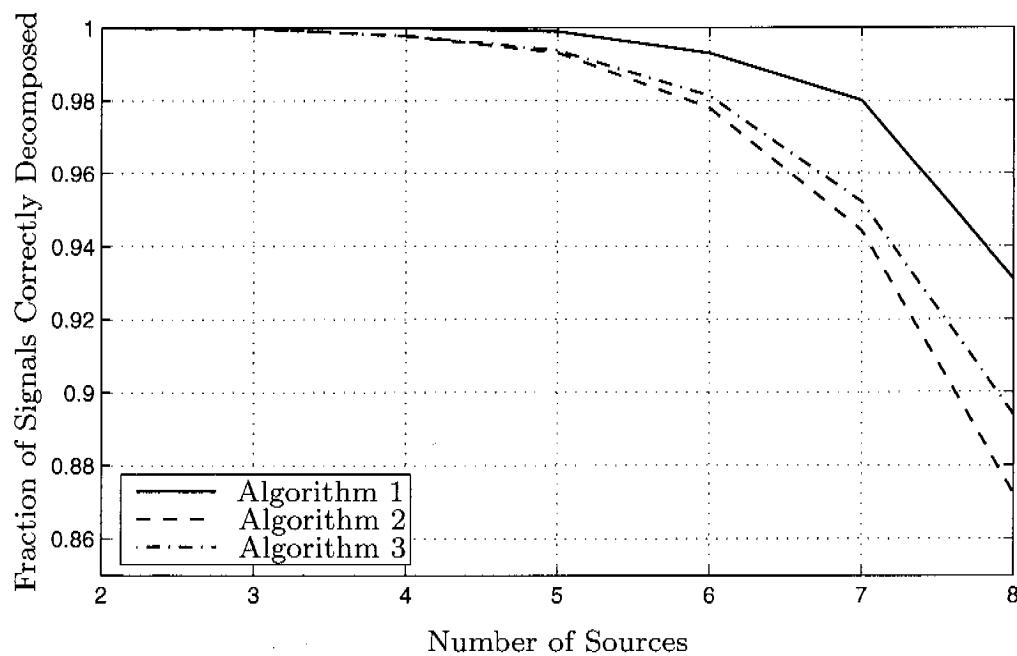


Figure 5.55: Fraction of completely correctly decomposed superpositions vs. the number of sources for a $\sigma_{\text{simul}} = 10$.

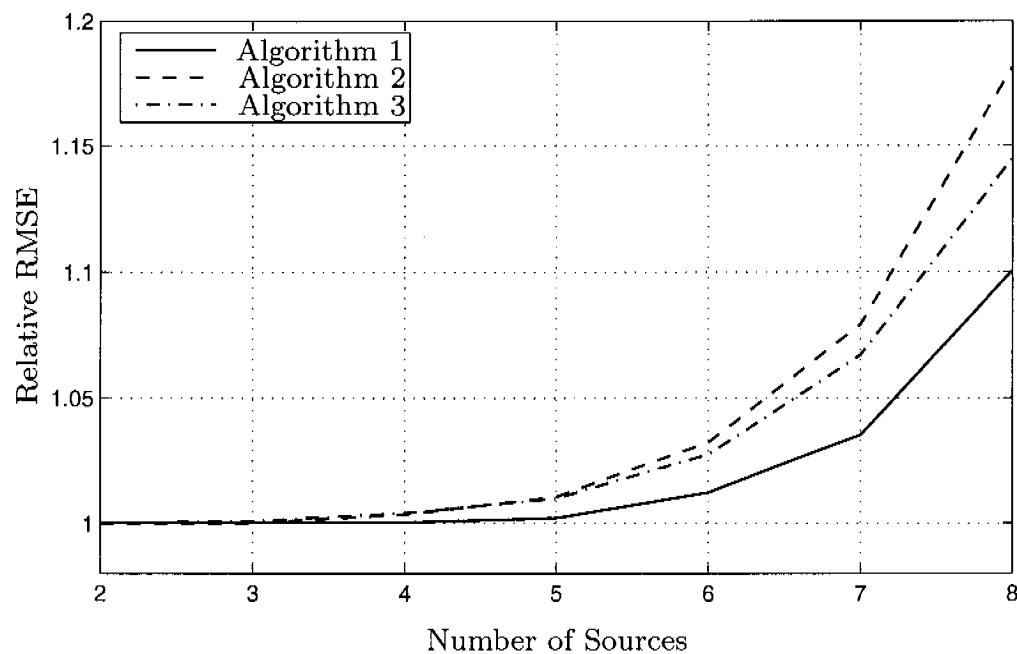


Figure 5.56: Relative RMSE vs. the number of sources for a $\sigma_{\text{simul}}=10$.

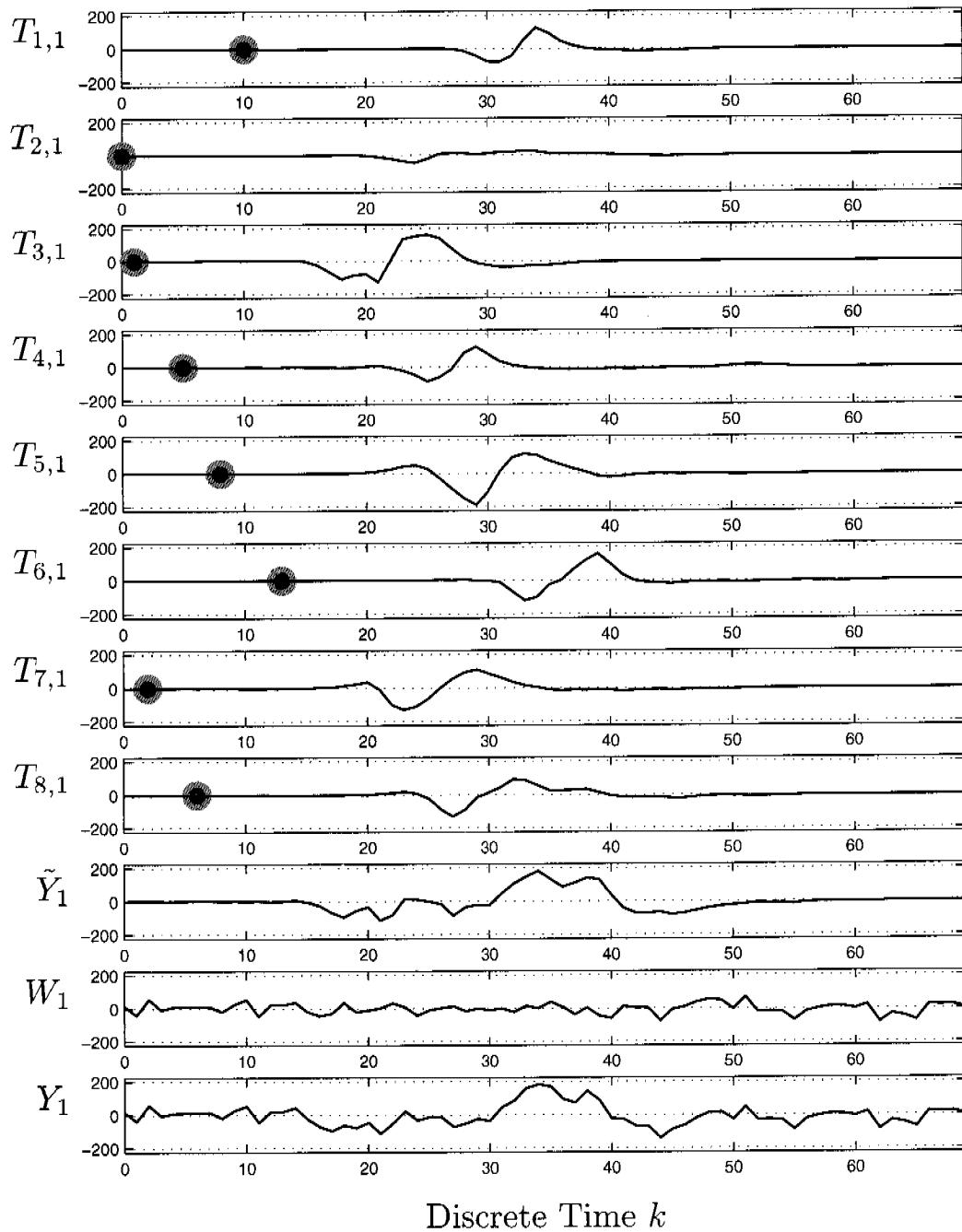


Figure 5.57: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4, 5, 6, 7, 8\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 30$, and the superposition with noise Y_1 . Compare to Figure 4.2.

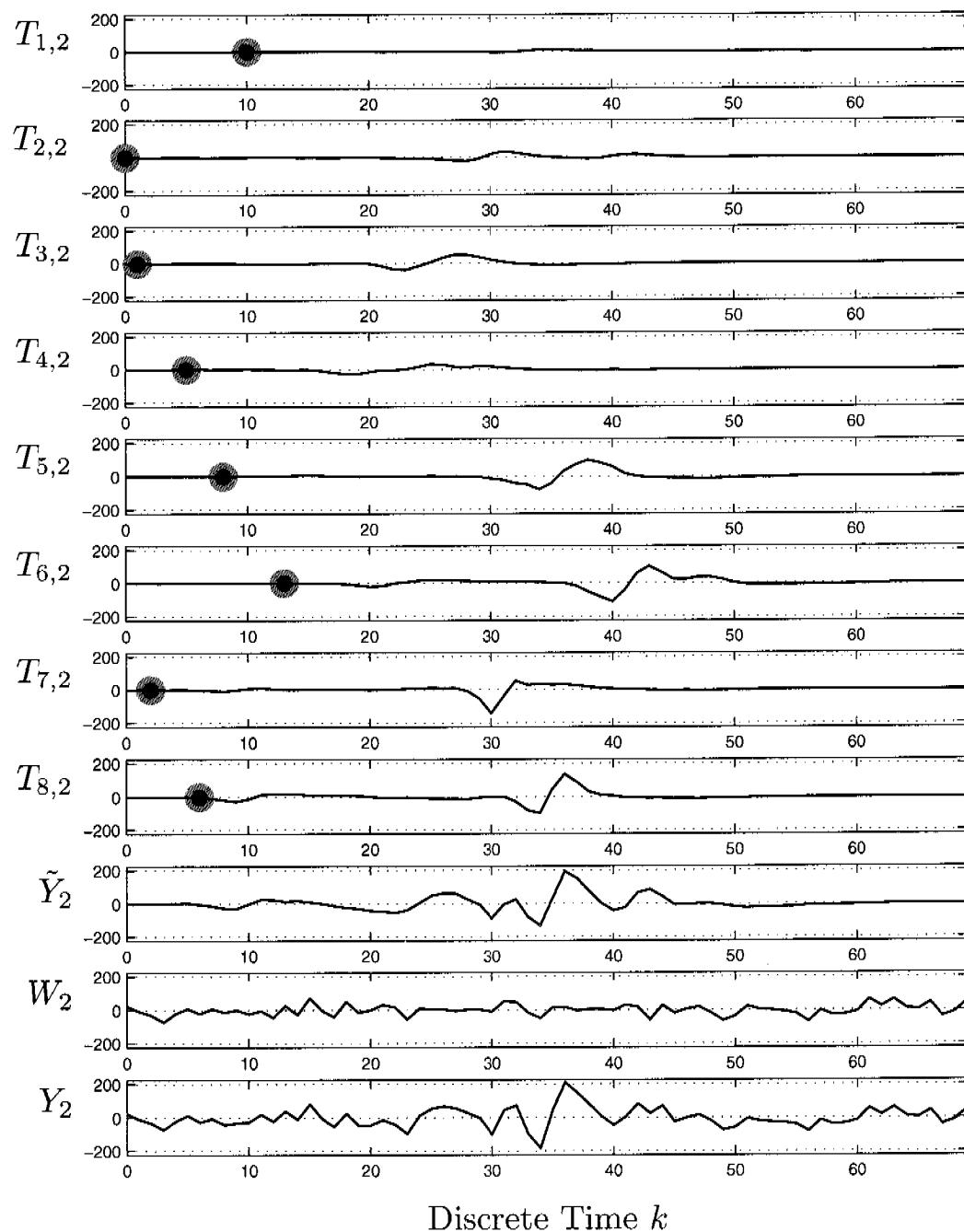


Figure 5.58: As 5.57, but for the second channel.

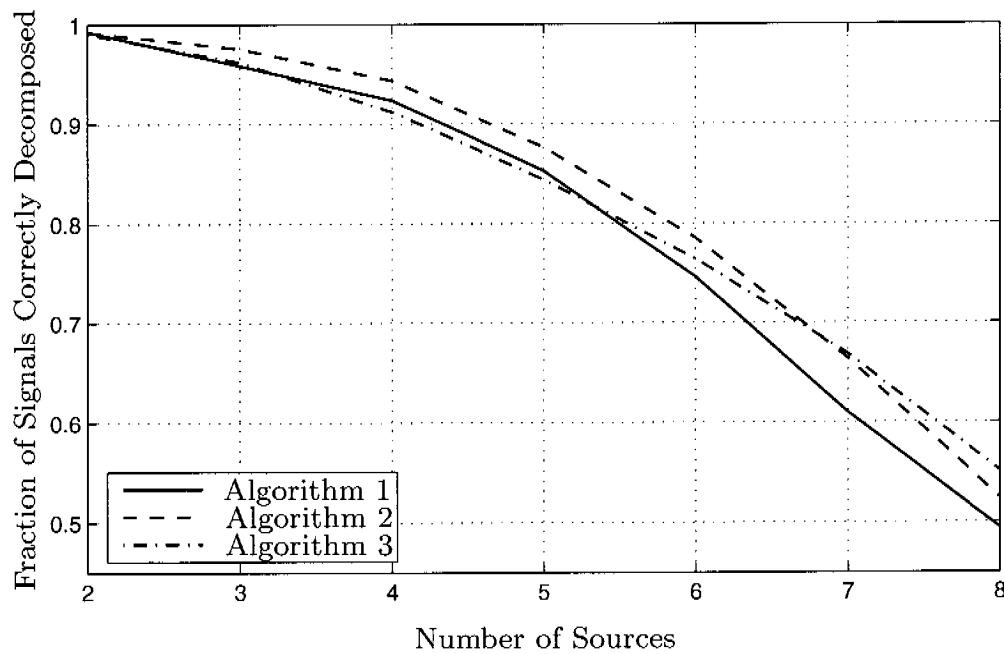


Figure 5.59: Fraction of completely correctly decomposed superpositions vs. the number of sources for a $\sigma_{\text{simul}} = 30$.

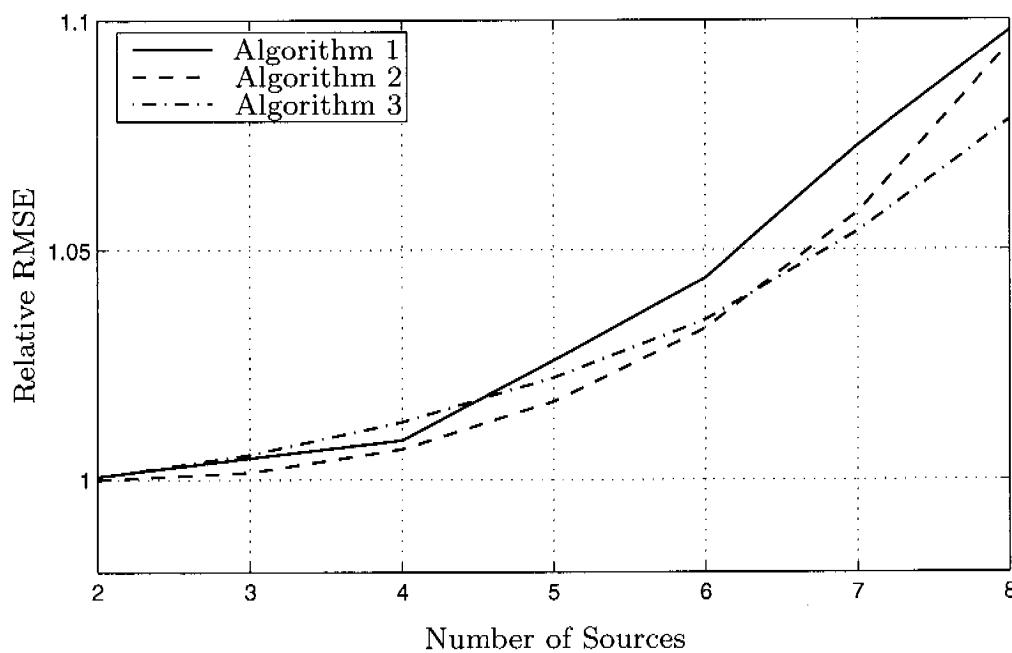


Figure 5.60: Relative RMSE vs. the number of sources for a $\sigma_{\text{simul}}=30$.

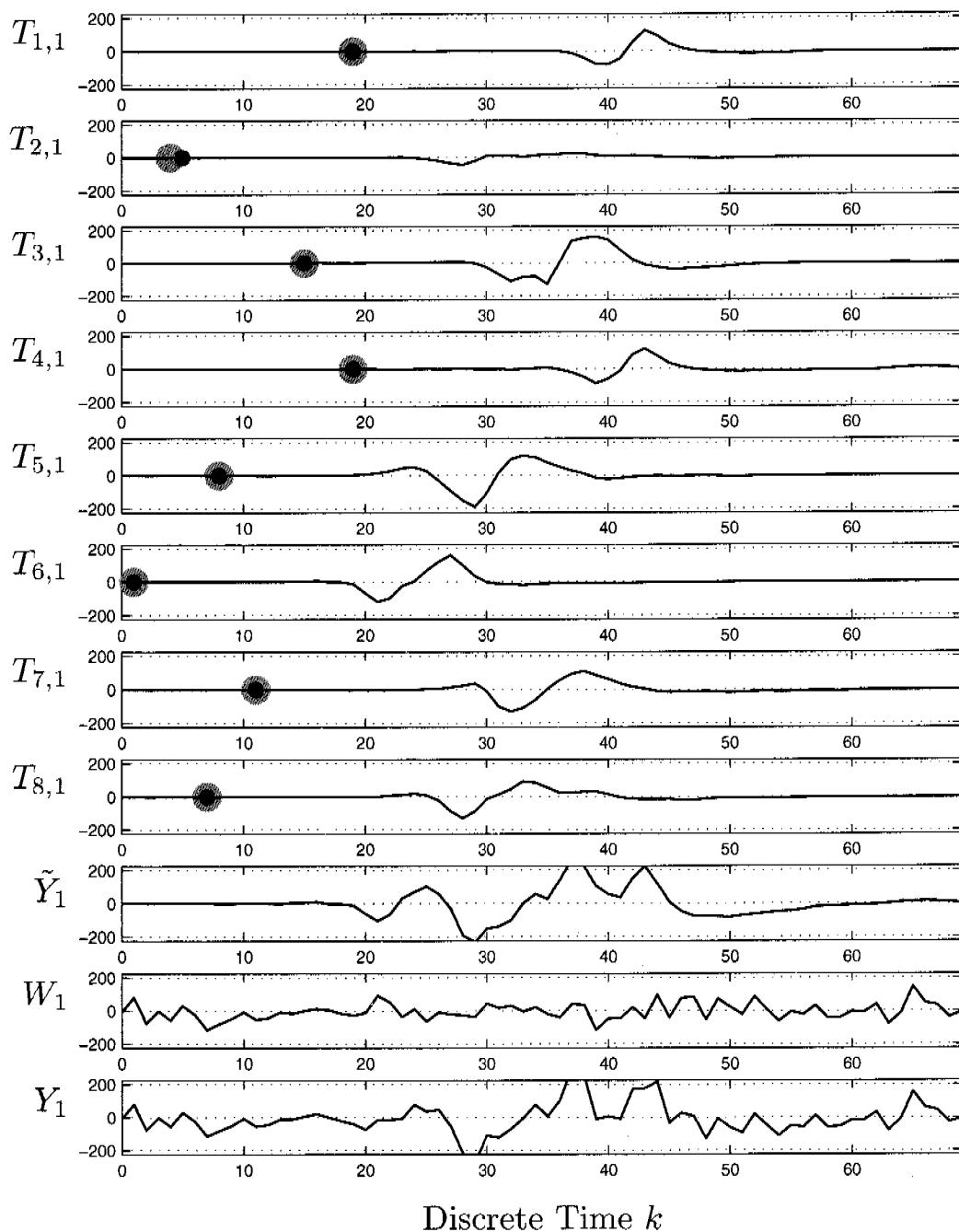


Figure 5.61: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4, 5, 6, 7, 8\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 50$, and the superposition with noise Y_1 . Compare to Figure 4.2.

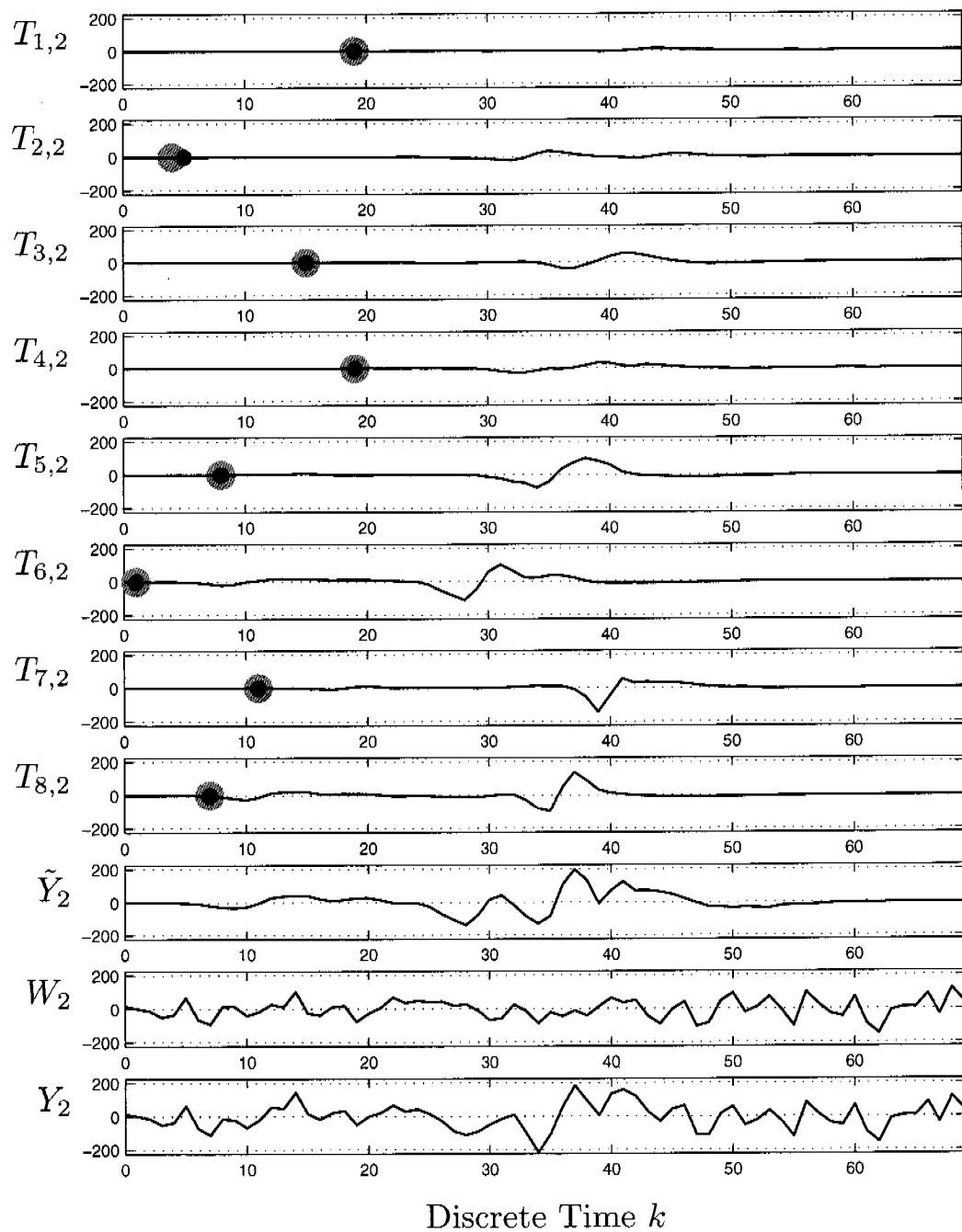


Figure 5.62: As 5.61, but for the second channel.

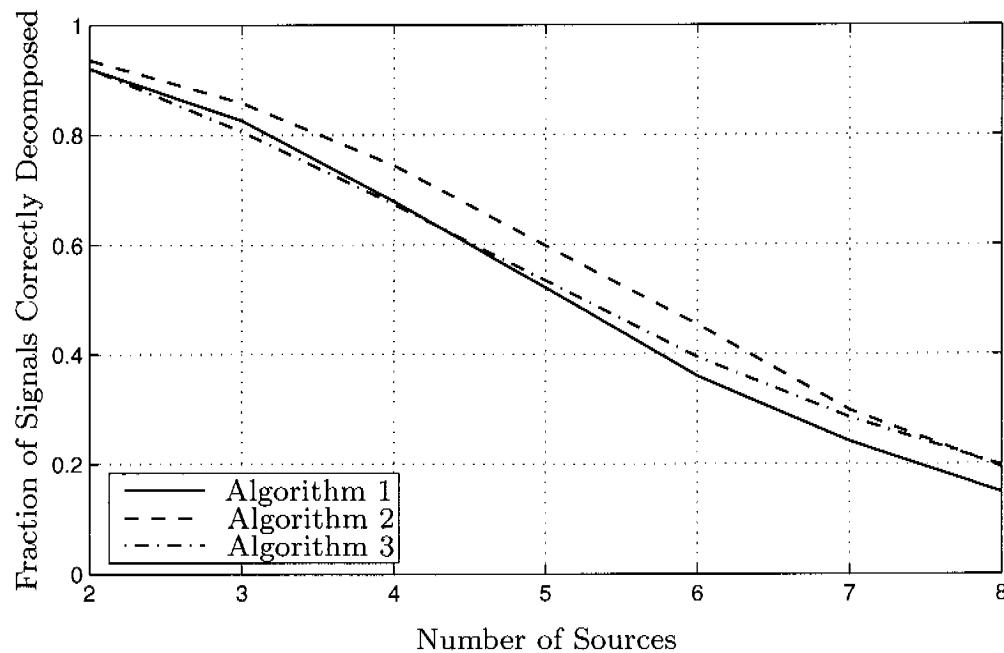


Figure 5.63: Fraction of completely correctly decomposed superpositions vs. the number of sources for a $\sigma_{\text{simul}} = 50$.

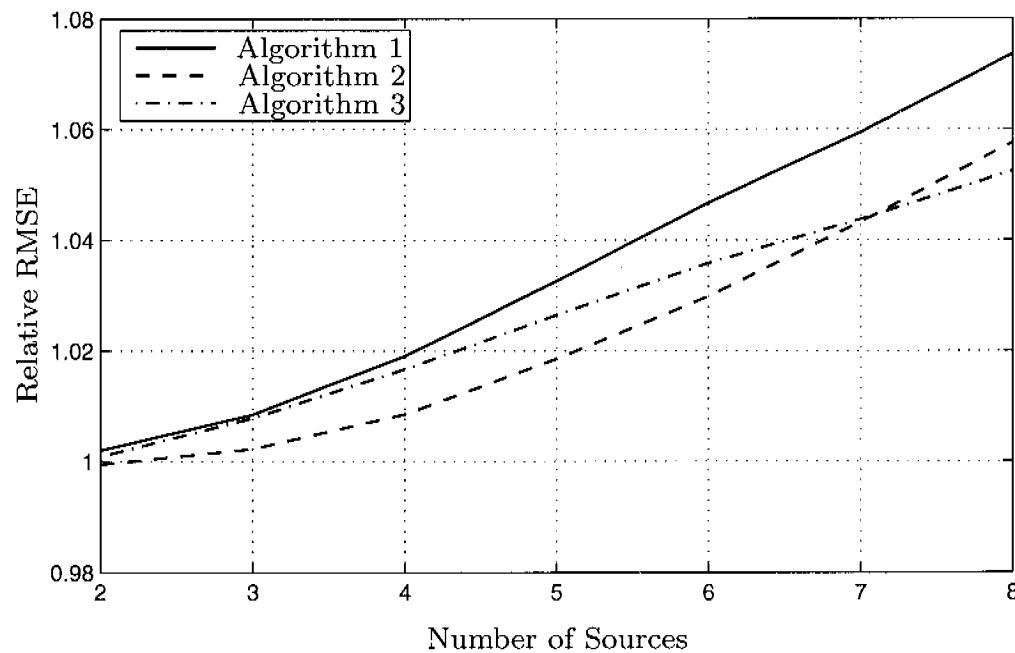


Figure 5.64: Relative RMSE vs. the number of sources for a $\sigma_{\text{simul}}=50$.

5.10 The Unknown-Constituent Problem

In the previous sections, we assumed that all sources become active during a superposition. In the unknown-constituent problem, it is not known which MUAPs make up a superposition. We now consider the case where superpositions consist of three out of four possible MUAPs.

The source model introduced in Section 5.2.3 can not handle such a situation since it requires all four sources to fire. We therefore modified the source model so that it allows sources to not fire.

Figure 5.65 shows the possible locations of the MUAP in the signal. In contrast to Figure 5.3, the new case “D” represents the situation where the source is not active.

We also have to extend the state transition diagram given in Figure 5.5. Figure 5.66 shows the new version for the unknown-constituent problem. As before, at time $k = -1$ the source model is in state “S”. From there we go along one of the ways “A”, “B”, or “C”, or “D”. In the case of way “D”, the source does not fire. Compared to the source model for the known-constituent problem, three new idle states had to be added for case “D”.

The left terminal node is initialized so that its outgoing message is

$$\vec{\mu}_{S_{1,0}}(s_{1,0}) = \begin{pmatrix} \vec{\mu}_{S_{1,0}}(0) \\ \vec{\mu}_{S_{1,0}}(1) \\ \vec{\mu}_{S_{1,0}}(2) \\ \vec{\mu}_{S_{1,0}}(3) \\ \vec{\mu}_{S_{1,0}}(4) \\ \vec{\mu}_{S_{1,0}}(5) \\ \vec{\mu}_{S_{1,0}}(6) \\ \vec{\mu}_{S_{1,0}}(7) \\ \vec{\mu}_{S_{1,0}}(8) \\ \vec{\mu}_{S_{1,0}}(9) \\ \vec{\mu}_{S_{1,0}}(10) \end{pmatrix} = \begin{pmatrix} p_1 \\ p_1 \\ p_1 \\ 0 \\ 0 \\ p_2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad (5.187)$$

where p_1 and p_2 are constants, which were determined empirically due to the suboptimal nature of our message-passing algorithms.

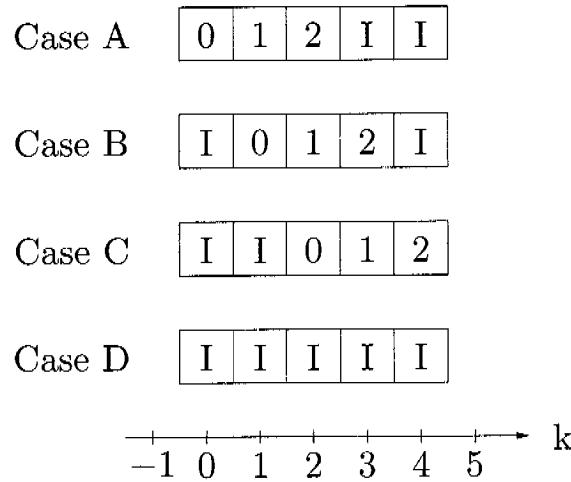


Figure 5.65: Possible locations of a MUAP in the signal. The numbers give the states of the FIR filter, which are also the sample numbers of the MUAP. “I” stands for an idle state.

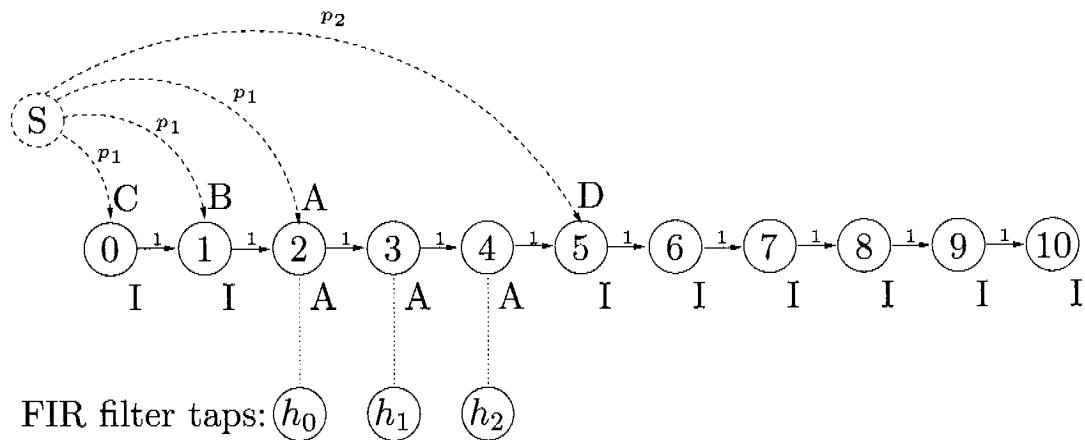


Figure 5.66: State transition diagram of the source model. “A” stands for an active state, “I” for an idle state.

The right terminal node is initialized so that its outgoing message is

$$\overleftarrow{\mu}_{S_{1,5}}(s_{1,5}) = \begin{pmatrix} \overleftarrow{\mu}_{S_{1,5}}(0) \\ \overleftarrow{\mu}_{S_{1,5}}(1) \\ \overleftarrow{\mu}_{S_{1,5}}(2) \\ \overleftarrow{\mu}_{S_{1,5}}(3) \\ \overleftarrow{\mu}_{S_{1,5}}(4) \\ \overleftarrow{\mu}_{S_{1,5}}(5) \\ \overleftarrow{\mu}_{S_{1,5}}(6) \\ \overleftarrow{\mu}_{S_{1,5}}(7) \\ \overleftarrow{\mu}_{S_{1,5}}(8) \\ \overleftarrow{\mu}_{S_{1,5}}(9) \\ \overleftarrow{\mu}_{S_{1,5}}(10) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ p_3 \\ p_3 \\ p_3 \\ 0 \\ 0 \\ p_4 \end{pmatrix}, \quad (5.188)$$

where p_3 and p_4 are constants, which were determined empirically due to the suboptimal nature of our message-passing algorithms.

5.11 Results for the Unknown-Constituent Problem

5.11.1 Simulation Set 5: Varying σ_{detect}

Next we present results for the unknown-constituent problem. The setting is similar as in Section 5.6.1: we decomposed single superpositions in two-channel signals. Each signal is 70 samples long. In contrast to Section 5.6.1, each signal consists of a superposition of three out of four possible MUAPs. The MUAP shapes of all four sources are given to the algorithms. However, the algorithm does not get the information which three out of the four MUAPs are active. The improvements described in Section 5.8.1 and Section 5.8.2 were not used here.

Figure 5.67 and Figure 5.68 show simulated signals for the case of a noise standard deviation $\sigma_{\text{simul}} = 10$. The corresponding decomposition results are given in Figure 5.69 and Figure 5.70.

Figure 5.71 and Figure 5.72 show simulated signals for the case of a noise standard deviation $\sigma_{\text{simul}} = 30$. The corresponding decomposition results are given in Figure 5.73 and Figure 5.74.

Figure 5.75 and Figure 5.76 show simulated signals for the case of a noise standard deviation $\sigma_{\text{simul}} = 50$. The corresponding decomposition results are given in Figure 5.77 and Figure 5.78.

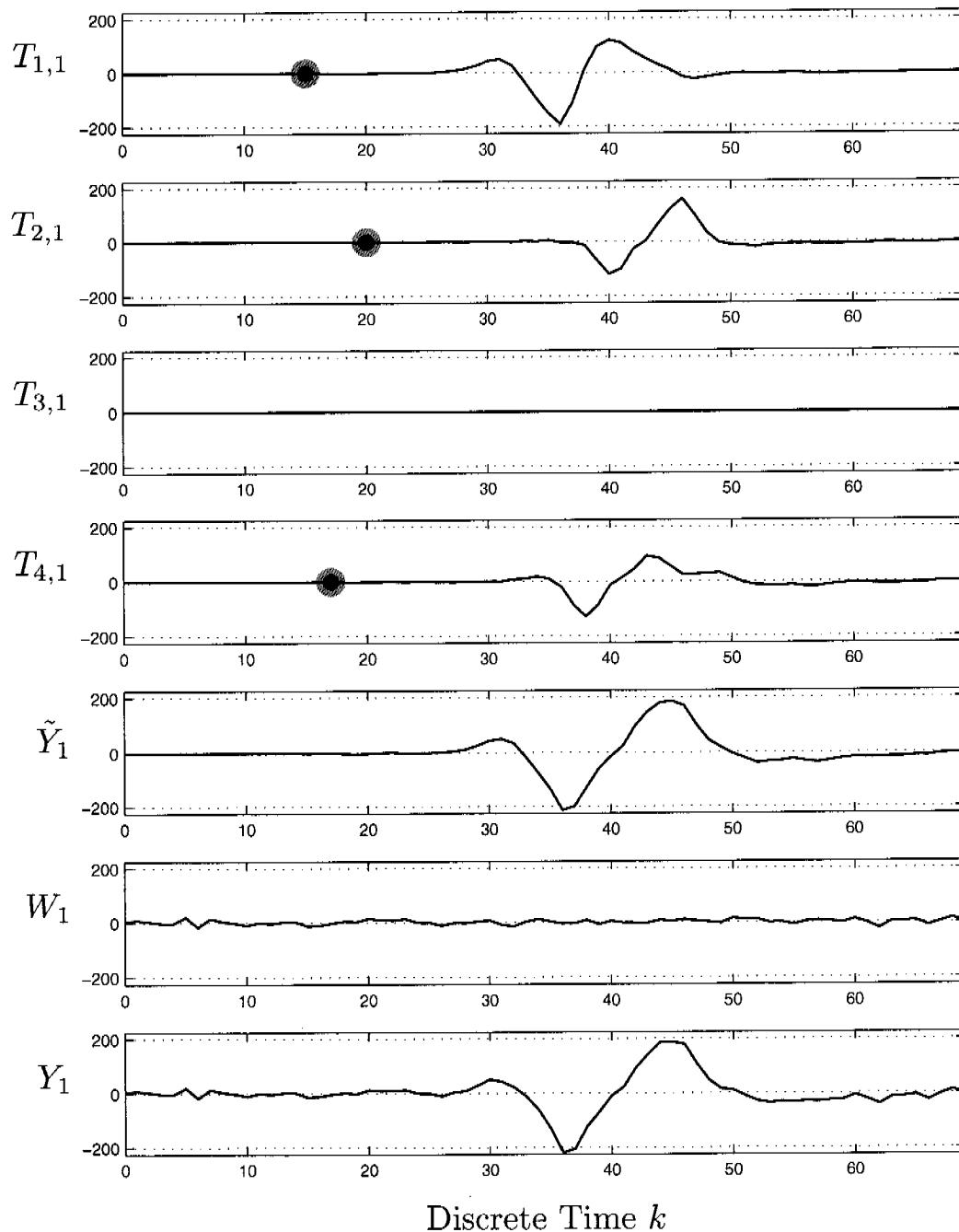


Figure 5.67: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 10$, and the superposition with noise Y_1 . Compare to Figure 4.2.

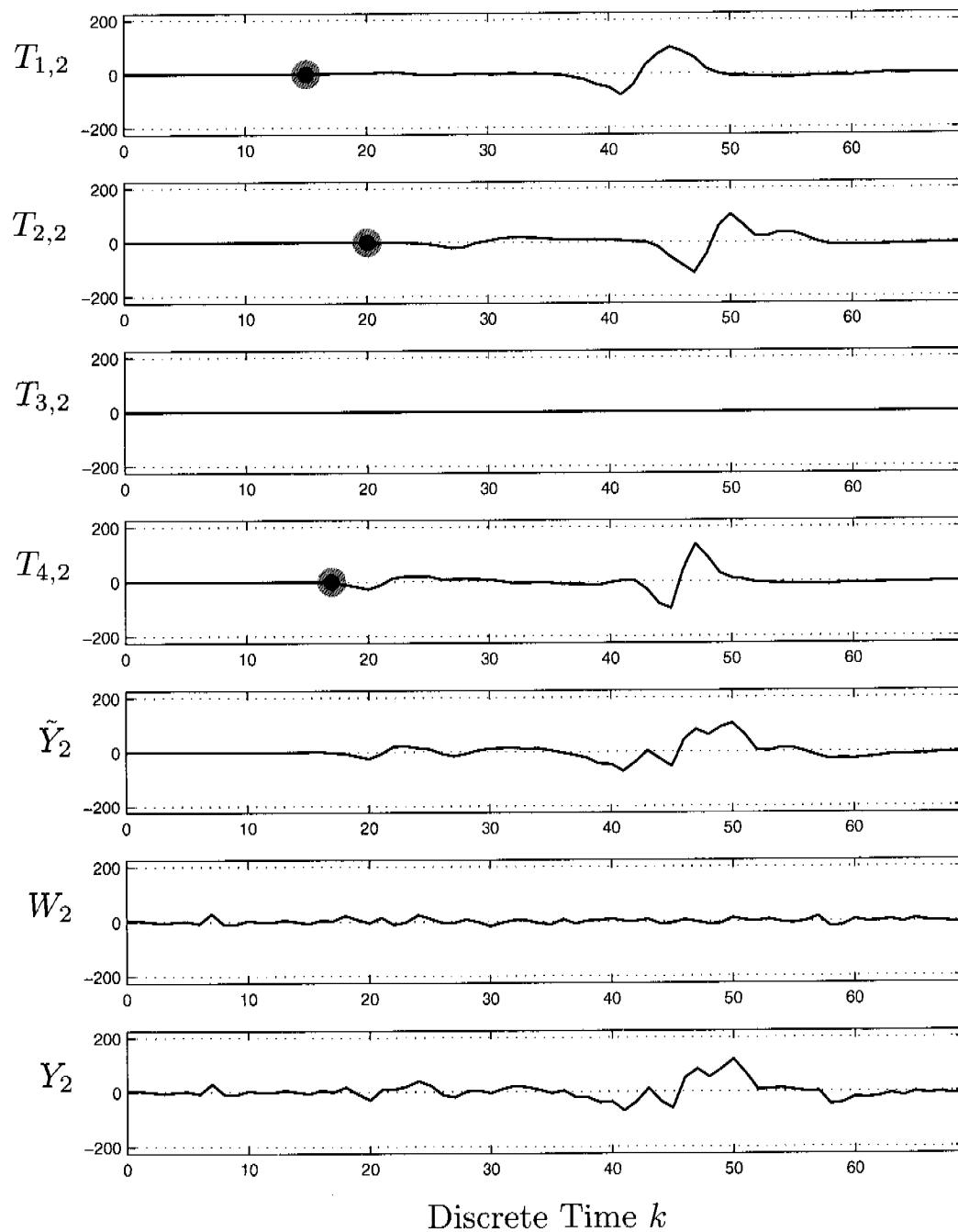


Figure 5.68: As 5.67, but for the second channel.

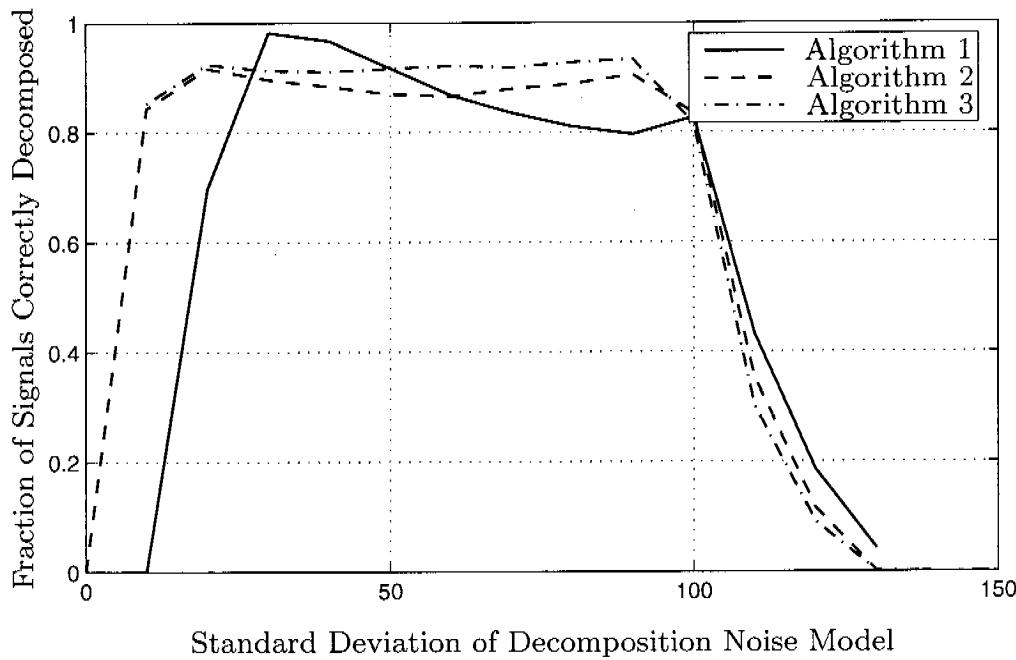


Figure 5.69: Fraction of completely correctly decomposed superpositions vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}} = 10$.

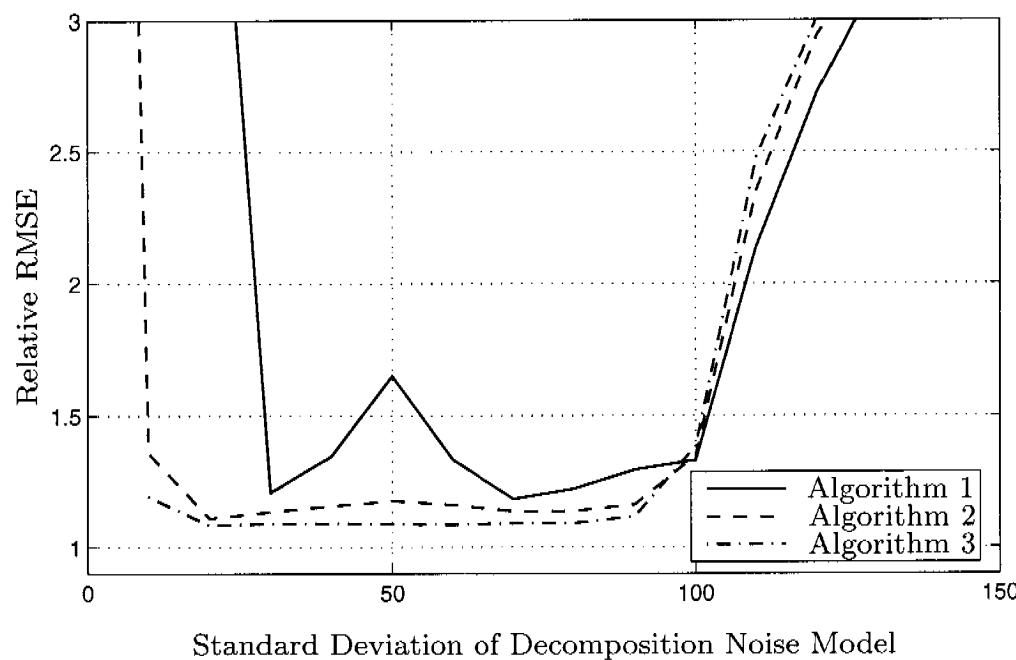


Figure 5.70: Relative RMSE vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}}=10$.

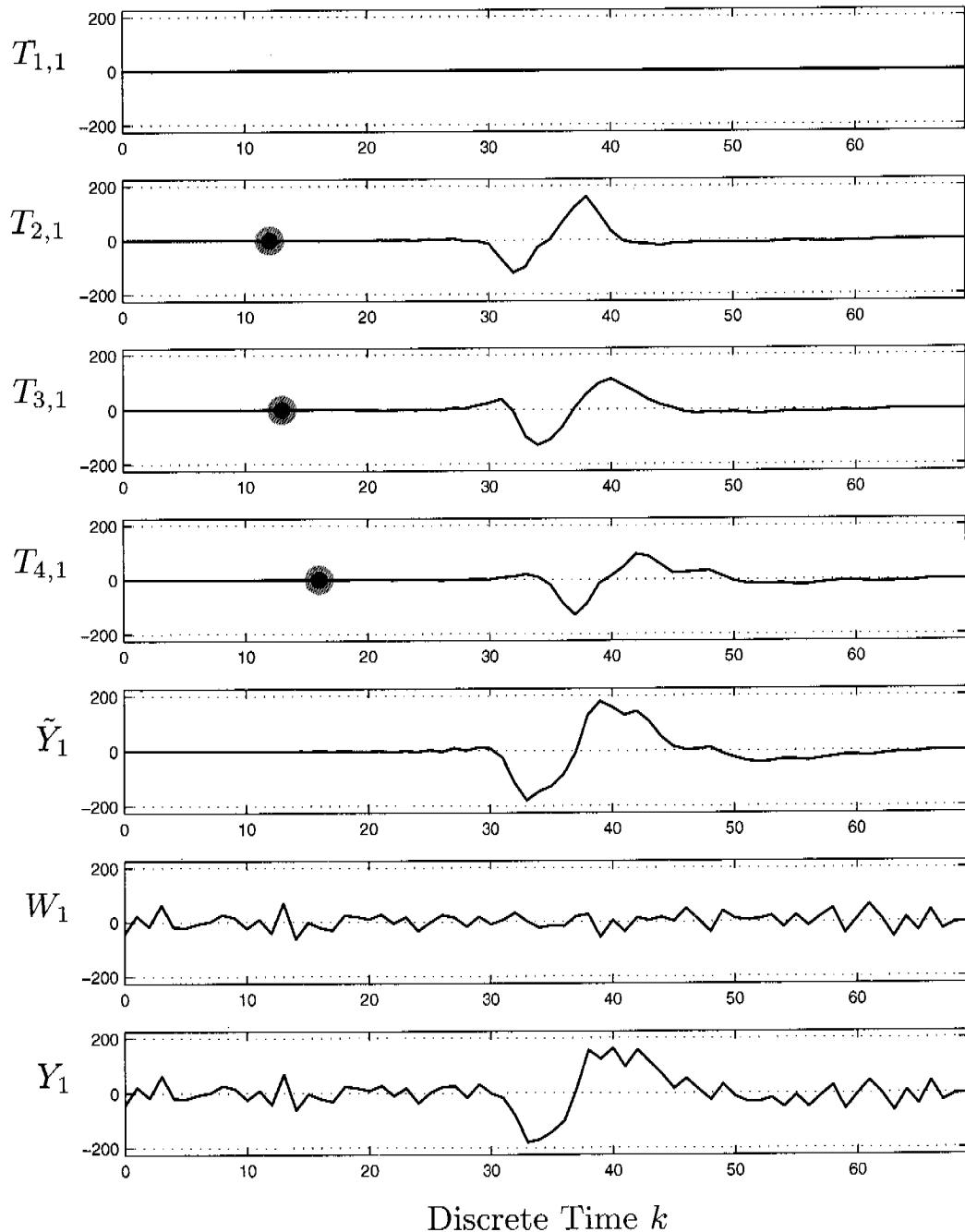


Figure 5.71: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 30$, and the superposition with noise Y_1 . Compare to Figure 4.2.

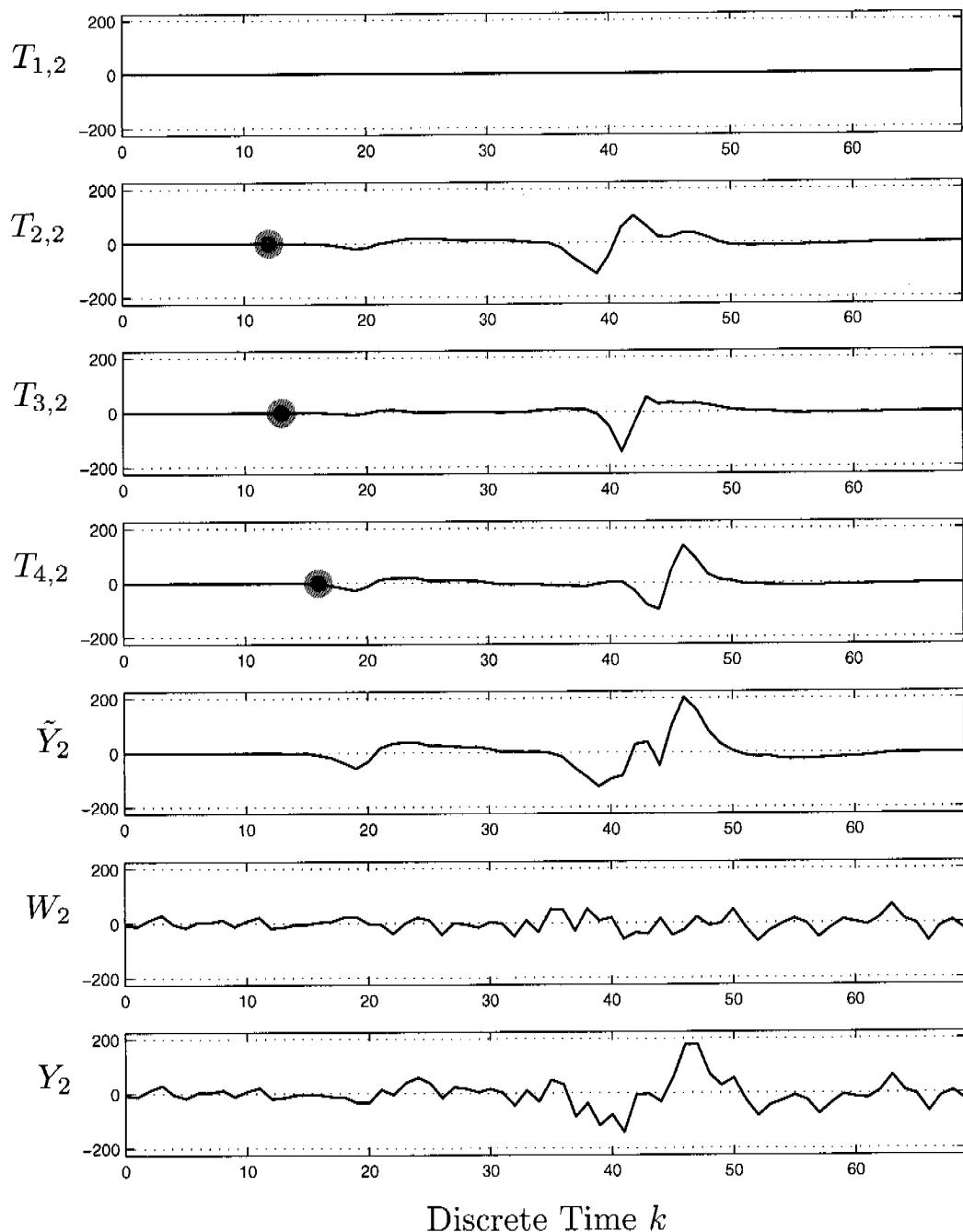


Figure 5.72: As 5.71, but for the second channel.

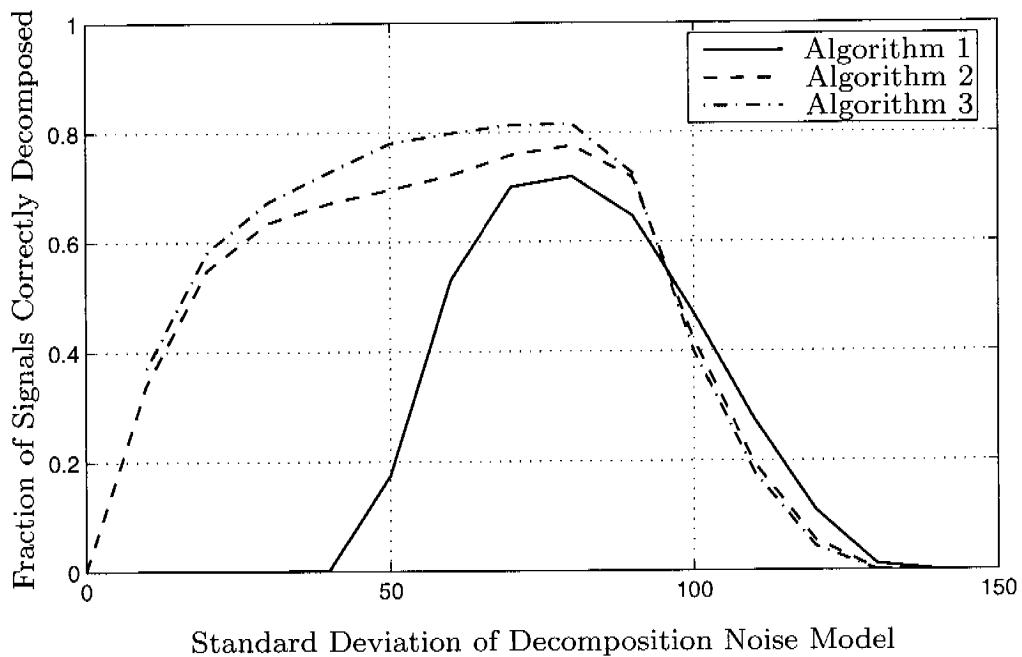


Figure 5.73: Fraction of completely correctly decomposed superpositions vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}} = 30$.

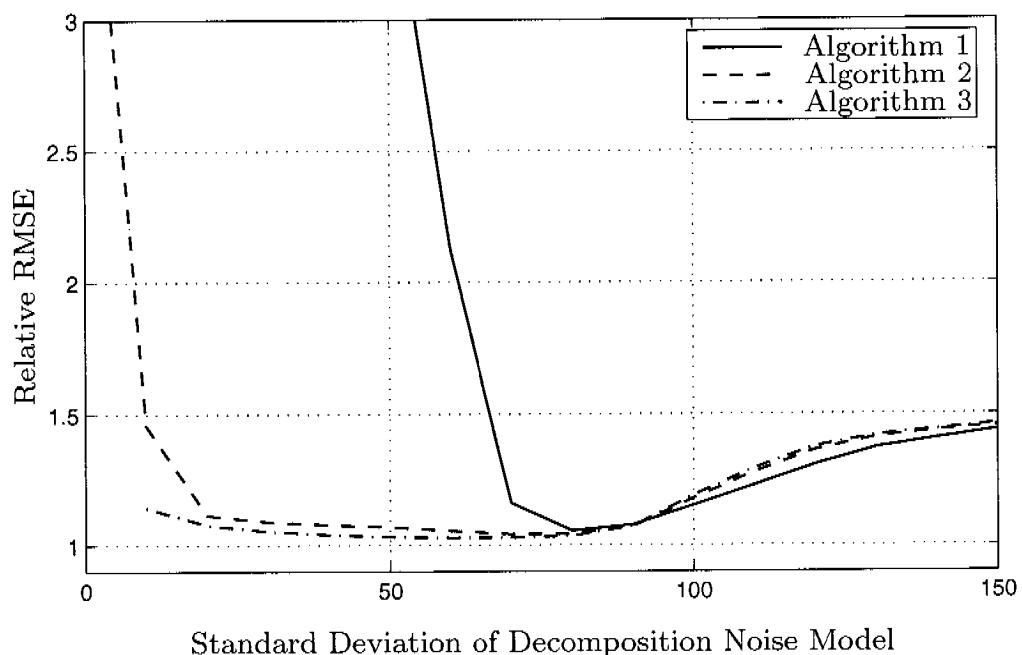


Figure 5.74: Relative RMSE vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}}=30$.

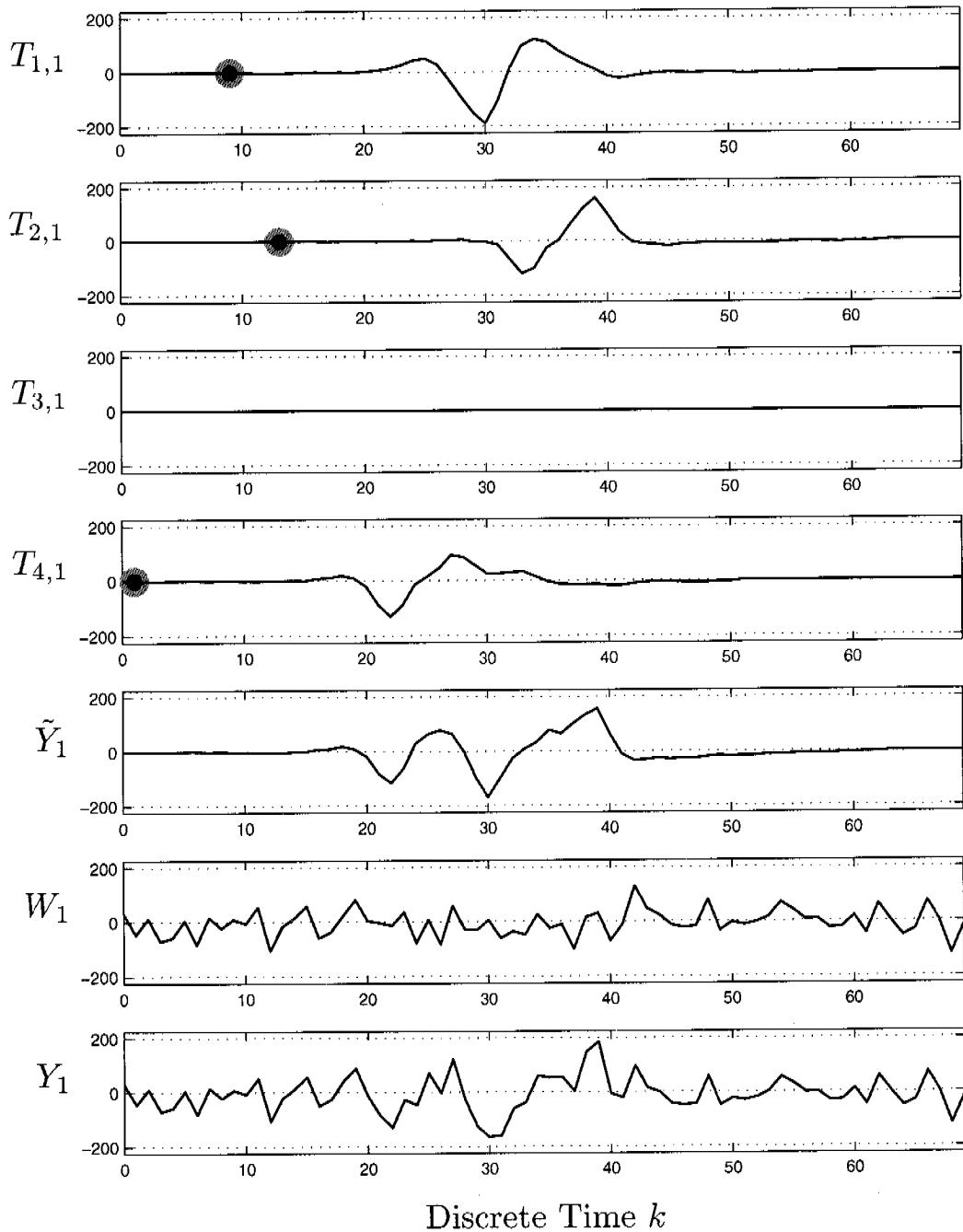


Figure 5.75: Four MUAP trains $T_{i,1}$, generated by sources $i \in \{1, 2, 3, 4\}$ with marked firing times, the superposition of these MUAP trains \tilde{Y}_1 , additive white Gaussian noise W_1 with a standard deviation of $\sigma_{\text{simul}} = 50$, and the superposition with noise Y_1 . Compare to Figure 4.2.

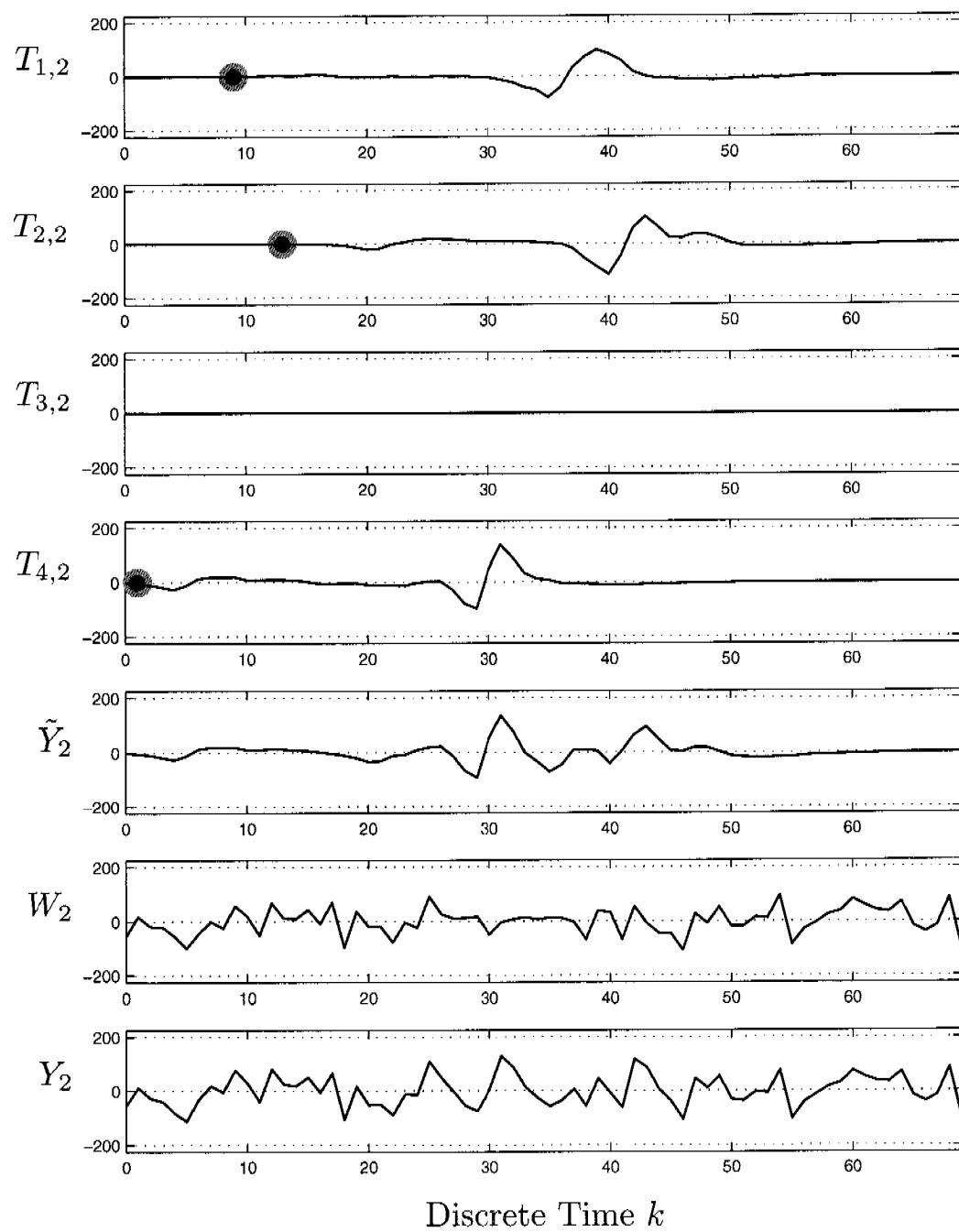


Figure 5.76: As 5.75, but for the second channel.

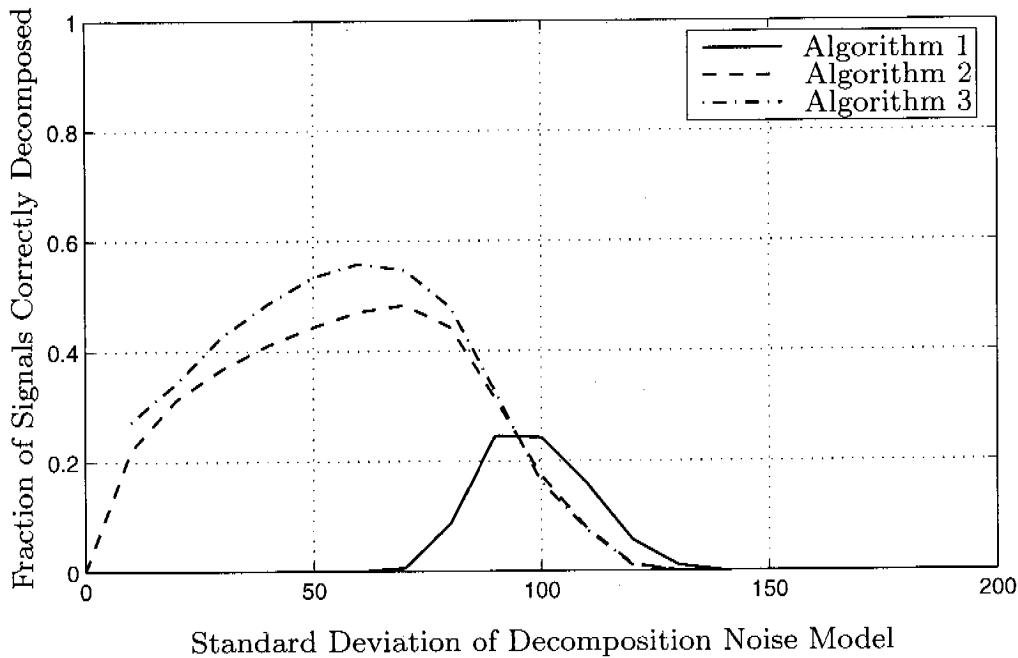


Figure 5.77: Fraction of completely correctly decomposed superpositions vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}} = 50$.

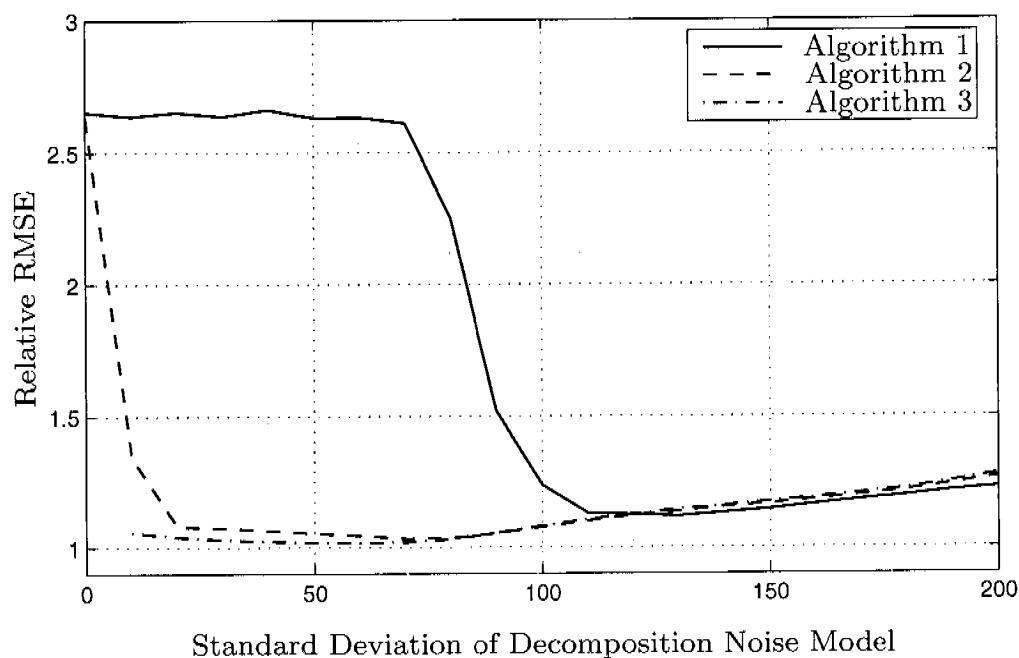


Figure 5.78: Relative RMSE vs. the detection parameter σ_{detect} for a $\sigma_{\text{simul}} = 50$.

5.11.2 Simulation Set 6: Varying σ_{simul}

Figure 5.79 and Figure 5.80 show results for the case where we used the improvements described in Section 5.8.1 and Section 5.8.2. Exemplary two-channel EMG signals with three out of four possible sources firing can be seen in Section 5.11.1.

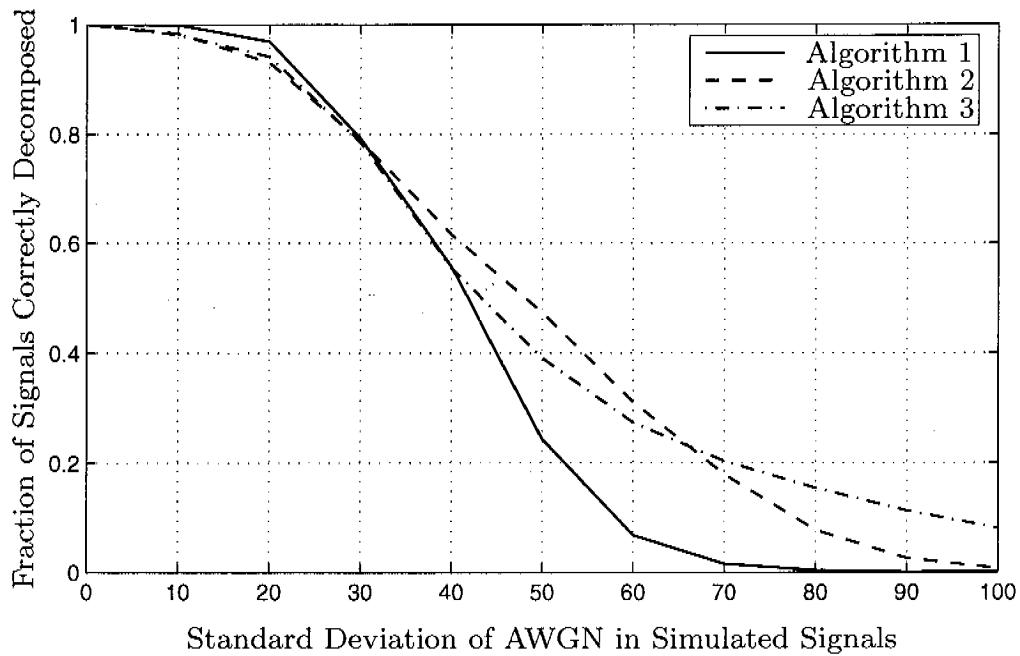


Figure 5.79: Fraction of completely correctly decomposed superpositions vs. the standard deviation of the AWGN in the simulated signals.

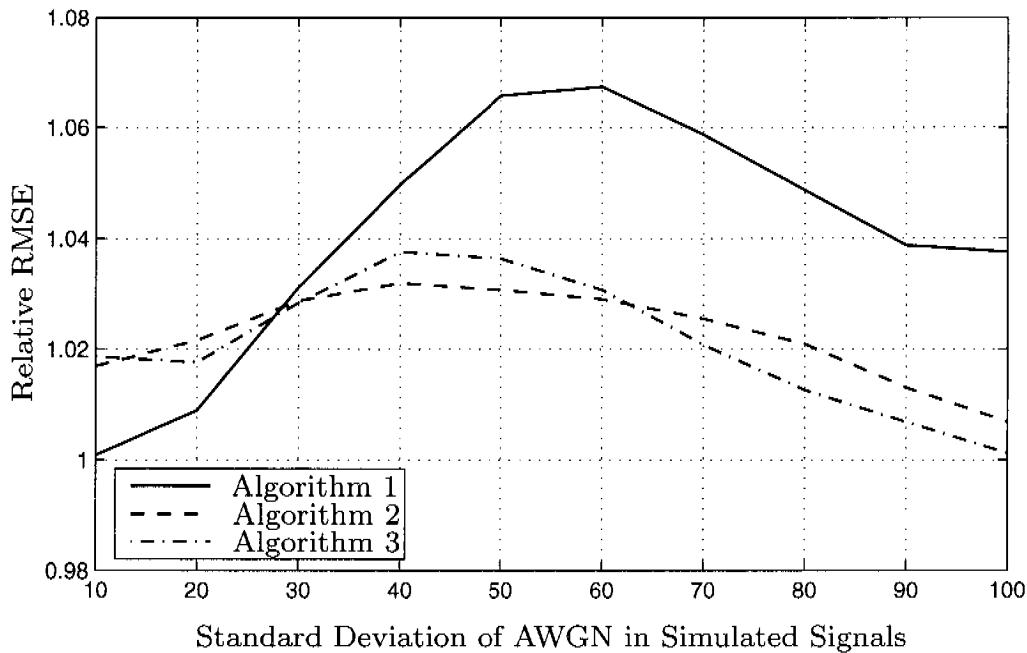


Figure 5.80: Relative RMSE vs. the standard deviation of the AWGN in the simulated signals.

5.12 Summary, Conclusions, and Outlook

In this chapter we have presented three new factor-graph-based algorithms for resolving superpositions in EMG signals. These algorithms mainly differ in their factor graph topologies and in the message types used.

Algorithm 1: The first algorithm uses **only discrete messages**.

Algorithm 2: The second algorithm uses **discrete messages** for the variables above the coefficient nodes, and **scalar-Gaussian messages** for the variables below the coefficient nodes.

Algorithm 3: The third algorithm uses **discrete messages** for the variables above the coefficient nodes and **multivariate-Gaussian messages** for the variables below the coefficient nodes.

Like algorithm 2, algorithm 3 also uses parameterized messages. But it has the advantage of fewer short cycles in the factor graph since it processes the signals of different electrodes simultaneously, cf., Figure 4.18.

The decomposition algorithm described by McGill in [101] is optimal in the sense that it finds the firing times that minimize the mean squared error between the reconstructed signal and the EMG signal. In contrast, the three message-passing algorithms described here are sub-optimal since the underlying factor graphs have cycles. Message-passing algorithms that pass messages along the edges of factor graphs with cycles are iterative algorithms, which might not converge to the correct solution. However, the results presented in this chapter show that the decomposition quality is still quite good. They could be further improved by automatically changing the important parameter σ_{detect} of the decomposition algorithms and then taking the result with the smallest RMSE.

Results for the known-constituent problem: Given synthetic two-channel superpositions with four similar MUAPs, all three improved algorithms decomposed more than 99 % of the signals without a mistake for a $\sigma_{\text{simul}} = 10$. For a $\sigma_{\text{simul}} = 20$, still more than 97 % of the signals were decomposed correctly by all algorithm; see Figure 5.51. For synthetic two-channel superpositions of 8 MUAPs,

all three algorithms decomposed more than 87 % of the signals correctly for a $\sigma_{\text{simul}} = 10$. Algorithm 1 even decomposes more than 93 % of the signals correctly in this case; see Figure 5.55.

Results for the unknown-constituent problem: Given synthetic two-channel superpositions with three out of four similar MUAPs, all three improved algorithms decomposed more than 98 % of the signals without a mistake for a $\sigma_{\text{simul}} = 10$. For a $\sigma_{\text{simul}} = 20$, still more than 93 % of the signals were decomposed correctly by all algorithms; see Figure 5.79.

For the known-constituent problem, the average time to correctly decompose a 70 sample long superposition containing 4 MUAPs of length 50 with AWGN having a standard deviation of 10 was 2.75 s for algorithm 1, 221 ms for algorithm 2, and 388 ms for algorithm 3; see Figure 5.45 and Figure 5.46. In general, the computational complexity of algorithm 1 is substantially higher than those of algorithms 2 and 3. This is especially true for superpositions with many overlapping MUAPs and for high levels of noise.

Our new message-passing algorithms show a roughly linearly growing computational demand with the durations of the MUAPs; see Figure 5.47 and Figure 5.48. Algorithms 2 and 3 also show a roughly linearly growing computational demand with the number of sources; see Figure 5.46. In contrast, the computational effort in McGill's optimal algorithm [101] increases exponentially with the duration of the MUAPs and the number of MUAPs.

In this chapter we have considered the problem of resolving superpositions. The next chapter deals with decomposing longer EMG signals.

Chapter 6

Decomposing EMG Signals Using Factor Graphs

Perfection is achieved, not when there is nothing more to add,
but when there is nothing left to take away.

Antoine de Saint-Exupery

Everything should be made as simple as possible, but not
simpler.

Albert Einstein

In the previous chapters, we presented algorithms for resolving individual superpositions in EMG signals using factor graphs and message-passing algorithms. However, often we would like to decompose longer measured EMG signals and not just superpositions. We already sketched some problems and difficulties one faces in this case in Section 3.7. In this chapter, we are going to discuss and explain in more detail what needs to be done and what has to be considered when decomposing longer EMG signals with recording times of several minutes. We will also explain how our algorithms may be extended to decompose such longer EMG signals.

In particular we will elucidate which problems we decided to solve using message-passing algorithms on factor graphs and which problems we chose to solve by other means. For example, should MUAP shapes be estimated by message passing? Estimating MUAP shapes, which are modeled by FIR filters with up to 600 coefficients per filter, using message passing can easily yield computationally expensive algorithms that may not even converge. Although this might change in the future as we become more experienced, we decided to not use message-passing algorithms to estimate MUAP shapes. The same holds for the adaption of filter coefficients to changing MUAP shapes, which is especially important when decomposing long-term EMG recordings: we decided to do this by traditional averaging rather than by message passing.

6.1 Overview

When we would like to decompose EMG signals and not just single superpositions, we need to consider the following:

Segmenting EMG signals: If we assume that the characteristic MUAP shapes of all motor units are given¹, then our algorithm does not need a segmentation step. Nevertheless, we will sketch methods for segmenting EMG signals. The segmentation result can be used to estimate filter coefficients if they are not already given. Also, our factor-graph-based decomposition algorithms may be made more efficient if we run them only on the active segments rather than on the full EMG signal.

Estimating EMG signal noise power: We have observed that the standard deviation of the additive white Gaussian noise model has an important influence on the performance of our decomposition algorithms. It is therefore advisable to estimate the noise variance based on the EMG signals that we want to decompose. Since the noise level changes over time, the variance parameter should be adapted when an EMG signal is decomposed. The estimated noise variance can also be used as an important parameter for EMG signal segmentation algorithms.

¹The characteristic MUAP shapes might be provided by some other algorithm, e.g., EMG-LODEC [135], of which our decomposition algorithms may be a plug-in to deal with difficult superpositions [9].

Estimating MUAP shapes: We will often assume that the filter coefficients of the FIR filters that model the MUAP shapes are given a-priori. However, in this chapter we sketch how filter coefficients can be estimated in principle.

Using source models: We know that every motor unit fires repeatedly. The inter-spike intervals are distributed according to some probability distribution. This knowledge can be used to improve our models, our algorithms, and ultimately the decomposition algorithm performance. However, using a strong source model can also lead to mistakes. We will discuss this issue in more detail in this chapter.

Using block processing: As stated before, our factor graph consists of many slices; one slice for each discrete time index in the EMG signal. Since computer RAM is limited, we can only have a certain number of slices. To decompose long EMG signals anyway, we have developed a block processing method. This means that we do not create a factor graph for the full EMG signal, which may not fit into RAM, but rather a factor graph for a shorter block of the EMG signal. Then we process the full EMG signal by dividing it up into overlapping blocks that the algorithms process separately.

Adapting MUAP shapes: MUAPs of the same motor unit change their characteristic shapes over time. Especially when dealing with long-term recordings, decomposition algorithms need to track these changes and adapt the filter coefficients of the FIR filters modeling the MUAPs accordingly.

Dealing with recruitment and decruitment: When the level of muscle force increases, more motor units may become active. This is called recruitment of new motor units, whereas decruitment means the opposite. A good decomposition algorithm needs to be able to deal with recruitment and decruitment of motor units.

In the next sections, we will explain and discuss the topics outlined here in more detail.

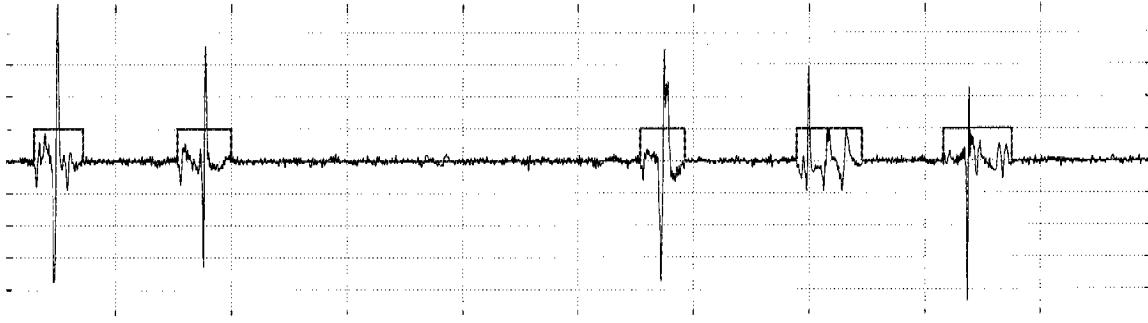


Figure 6.1: EMG signal and the corresponding segmentation result.

6.2 Segmenting EMG Signals

Problem Statement

Segmenting an EMG signal means the classification of signal parts into two classes: idle segments and active segments. As explained in Section 3.4, active segments are made up of single MUAPs, superpositions of MUAPs, artifacts, noise, and combinations of these components. Figure 6.1 shows an EMG signal and the result of a segmentation process.

Overview

In this section we will present methods for EMG signal segmentation. After sketching some traditional approaches, we will describe a factor-graph-based algorithm. We have developed this algorithm to compare it to traditional methods and to learn more about the development of factor-graph-based algorithms and their applications to various problems.

6.2.1 Previous Segmentation Methods

Peter Wellig describes different criteria to segment EMG signals in his doctoral thesis [135]:

Steepness-based criteria: Steepness-based criteria are useful to detect active segments since the signal value usually increases at the beginning of an active segment. A simple criteria can be based

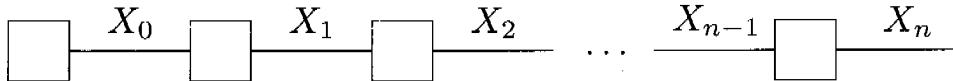


Figure 6.2: Factor graph of a Markov chain.

on the sum of differences between neighboring samples in a certain time interval. If the value is larger than some threshold, an active segment is detected.

Amplitude-based criteria: Even better for segmentation purposes are amplitude-based criteria [51]. They consider the signal energy and are especially advantageous in cases where active segments increase slowly in amplitude. A threshold for the amplitude can be set based on the estimated standard deviation of the noise. Wellig [135] used such amplitude-based criteria with success.

6.2.2 Factor-Graph-Based Segmentation

Markov Chain

The random variables X_0, X_1, \dots, X_n form a Markov chain if, for $0 < k < n$ conditioned on $X_k = x_k$, the random vectors (X_0, \dots, X_{k-1}) and (X_{k+1}, \dots, X_n) are independent [92].

For example, the Markov chain

$$p(x_0, \dots, x_n) = p(x_0)p(x_1|x_0)p(x_2|x_1) \cdots p(x_n|x_{n-1}) \quad (6.1)$$

has a factor graph as in Figure 6.2.

Hidden Markov Model

A Hidden Markov Model (HMM) is a widely used class of statistical models [92]. We can represent a HMM using the factor graph notation. Figure 6.3 shows the factor graph of a HMM with the probability distri-

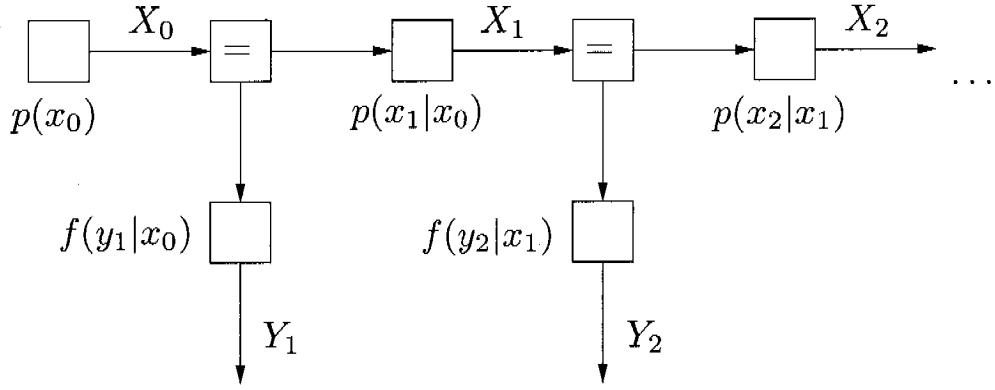


Figure 6.3: Factor graph of a Hidden Markov Model.

bution

$$p(x_0, x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n) = p(x_0) \prod_{k=1}^n p(x_k|x_{k-1})f(y_k|x_{k-1}), \quad (6.2)$$

where X_0, \dots, X_n are discrete random variables, called the states. They form a Markov chain and take values in some finite set \mathcal{X} , the state space. The variables Y_1, \dots, Y_n are real-valued random variables. They are directly observable. Note that Y_k depends only on X_{k-1} .

Factor Graph for EMG Signal Segmentation

We would like to segment an EMG signal into idle and active segments. The idle state corresponds to real-valued output signal parts with low energy; the active state corresponds to real-valued output signal parts with high energy.

We have designed and implemented a new factor-graph-based algorithm to segment EMG signals [136]. Figure 6.4 shows the factor graph for EMG signal segmentation, which is basically a HMM. Next we define the variables and then the individual node functions in the factor graph.

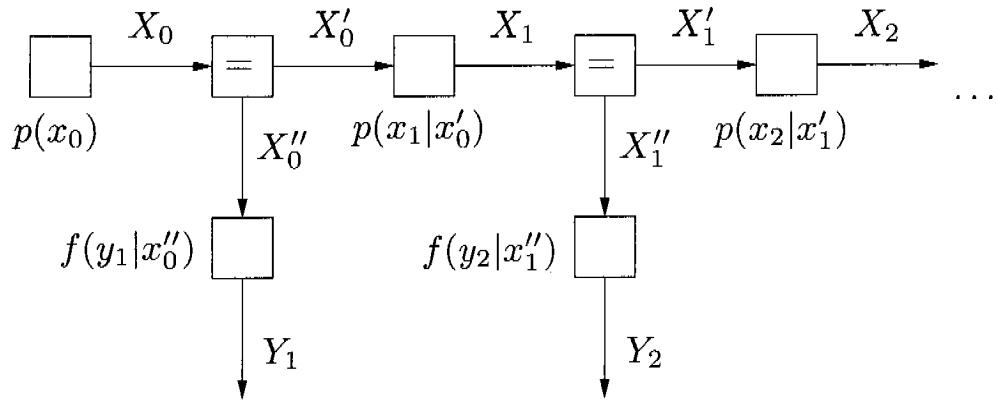


Figure 6.4: Factor graph for EMG signal segmentation with auxiliary variables.

Variables in the Factor Graph

We model an EMG signal $Y \triangleq (Y_1, \dots, Y_n)$ as an ordered collection of random variables Y_1, \dots, Y_n , where n is the number of samples, i.e., the length of the EMG signal.

The finite state space $\mathcal{X} = \{0, 1\}$ models the two states of the real-valued EMG signal: idle and active. We define $X_{k-1} = 0$ if a sample Y_k belongs to an idle segment, and $X_{k-1} = 1$ if Y_k belongs to an active segment.

The variables X'_k and X''_k in Figure 6.4 are auxiliary variables. The equality constraint node denoted with “=” makes sure that $X_k = X'_k = X''_k$.

Node Functions in the Factor Graph

Figure 6.5 shows a section of the factor graph. We will use the variables in this figure in the upcoming explanations, definitions, and derivations.

The functions $f(y_{k+1}|x''_k)$ give the relationship between the observed variables Y_{k+1} and the state variables X''_k , where

$$f(y_{k+1}|x''_k = 0) = \frac{1}{\sqrt{2\pi}\sigma_0} e^{-y_{k+1}^2/2\sigma_0^2} \quad (6.3)$$

and

$$f(y_{k+1}|x''_k = 1) = \frac{1}{\sqrt{2\pi}\sigma_1} e^{-y_{k+1}^2/2\sigma_1^2} \quad (6.4)$$

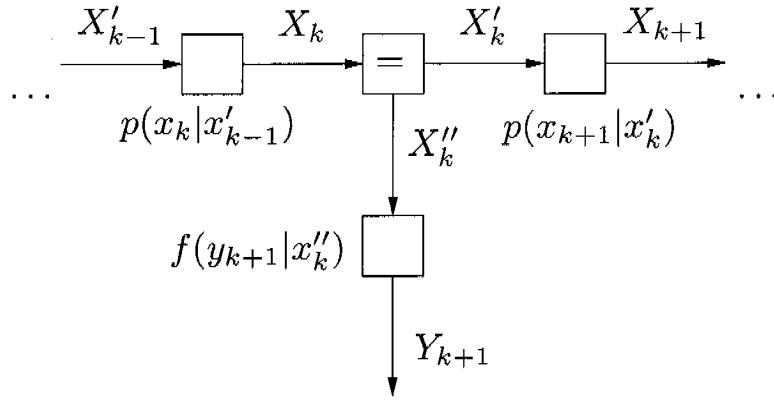


Figure 6.5: A section of the factor graph for EMG signal segmentation.

with

$$\sigma_0 < \sigma_1. \quad (6.5)$$

We therefore define the node functions of the nodes denoted with $f(y_{k+1}|x''_k)$ in Figure 6.5 as

$$f(y_{k+1}|x''_k) = \frac{1}{\sqrt{2\pi}\sigma_0} e^{-y_{k+1}^2/2\sigma_0^2} \cdot \delta(x''_k - 0) + \frac{1}{\sqrt{2\pi}\sigma_1} e^{-y_{k+1}^2/2\sigma_1^2} \cdot \delta(x''_k - 1). \quad (6.6)$$

The node denoted with “=” in Figure 6.5 is a equality constraint node as introduced in Section 4.4.5, i.e.,

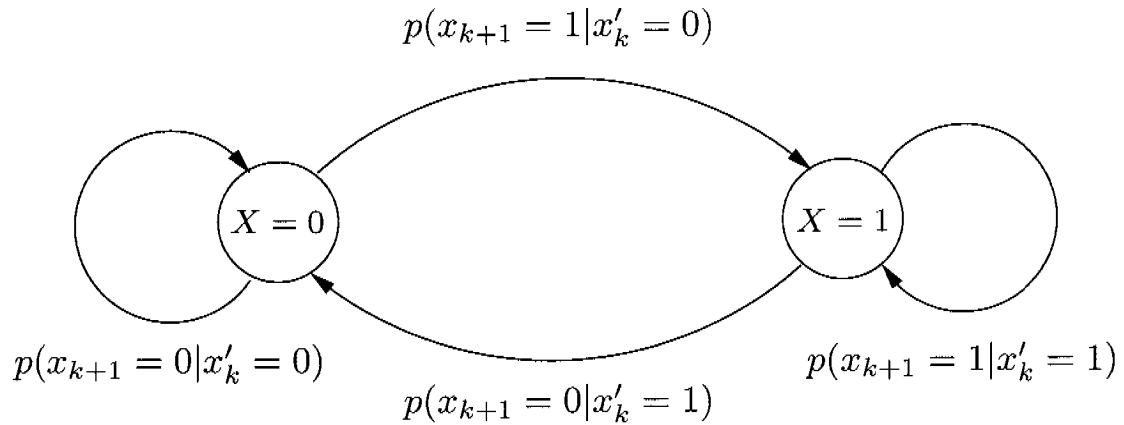
$$f_{=} = (x_k, x'_k, x''_k) \triangleq \begin{cases} 1, & \text{if } x_k = x'_k = x''_k \\ 0, & \text{else.} \end{cases} \quad (6.7)$$

$$(6.8)$$

Next we will explain the state transition nodes, for example the node with the node function $p(x_{k+1}|x'_k)$ in Figure 6.5. The state transition nodes model the state transitions from one time slice to the next time slice. Figure 6.6 shows the corresponding state transition diagram and Table 6.1 shows the state transition table. This finite state model has 2 parameters that have to be chosen: ϵ_0 and ϵ_1 . They are defined as follows:

$$\epsilon_0 \triangleq p(x_{k+1} = 1|x'_k = 0) \quad (6.9)$$

$$\epsilon_1 \triangleq p(x_{k+1} = 0|x'_k = 1). \quad (6.10)$$

**Figure 6.6:** State transition diagram.

The other two state transition probabilities are then:

$$p(x_{k+1} = 0|x'_k = 0) \triangleq 1 - \epsilon_0 \quad (6.11)$$

$$p(x_{k+1} = 1|x'_k = 1) \triangleq 1 - \epsilon_1. \quad (6.12)$$

These parameters determine the average durations of the two states. We assume them to be independent of k .

Calculating Messages in the Factor Graph

Figure 6.7 shows arrows that represent messages, which need to be calculated for EMG signal segmentation. We can calculate messages in this factor graph by using the standard sum-product rule (5.4).

Given the node function as in (6.6) and by using the sum-product rule, we can calculate the message going out of the node $f(y_{k+1}|x''_k)$ towards

x'_k	x_{k+1}	$P(x_{k+1} x'_k)$
0	0	$1 - \epsilon_0$
0	1	ϵ_0
1	0	ϵ_1
1	1	$1 - \epsilon_1$

Table 6.1: State transition table.

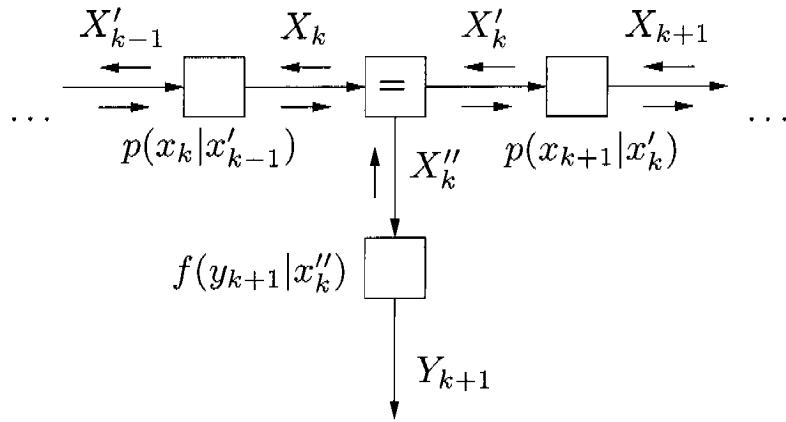


Figure 6.7: A section of the factor graph with messages that need to be calculated for EMG signal segmentation.

edge X''_k . We name this message $\overleftarrow{\mu}_{X''_k}(x''_k)$ according to our naming convention² in (4.59):

$$\overleftarrow{\mu}_{X''_k}(x''_k) = \int_{\tilde{y}_{k+1}} f(\tilde{y}_{k+1}|x''_k) \overleftarrow{\mu}_{\tilde{Y}_{k+1}}(\tilde{y}_{k+1}) d\tilde{y}_{k+1}. \quad (6.13)$$

Since the message $\overleftarrow{\mu}_{\tilde{Y}_{k+1}}(\tilde{y}_{k+1})$ represents a fixed sample value $Y_{k+1} = y_{k+1}$, $\overleftarrow{\mu}_{\tilde{Y}_{k+1}}(\tilde{y}_{k+1})$ can be written as

$$\overleftarrow{\mu}_{\tilde{Y}_{k+1}}(\tilde{y}_{k+1}) = \delta(\tilde{y}_{k+1} - y_{k+1}). \quad (6.14)$$

With this, (6.13) becomes

$$\overleftarrow{\mu}_{X''_k}(x''_k) = \int_{\tilde{y}_{k+1}} f(\tilde{y}_{k+1}|x''_k) \overleftarrow{\mu}_{\tilde{Y}_{k+1}}(\tilde{y}_{k+1}) d\tilde{y}_{k+1} \quad (6.15)$$

$$= \int_{\tilde{y}_{k+1}} f(\tilde{y}_{k+1}|x''_k) \delta(\tilde{y}_{k+1} - y_{k+1}) d\tilde{y}_{k+1} \quad (6.16)$$

$$= f(y_{k+1}|x''_k). \quad (6.17)$$

Since $x''_k \in \{0, 1\}$, message $\overleftarrow{\mu}_{X''_k}(x''_k)$, which is a function, can be represented as a vector with two numbers:

$$\overleftarrow{\mu}_{X''_k}(x''_k) \equiv \begin{pmatrix} \overleftarrow{\mu}_{X''_k}(0) \\ \overleftarrow{\mu}_{X''_k}(1) \end{pmatrix}. \quad (6.18)$$

²With this convention, the message is named according to the arrow on the edge of the factor graph in Figure 6.7. The name does not depend on the additional arrow besides the edge, which actually represents the message. The arrows besides the edges of the factor graph are only included to visualize which messages need to be calculated for EMG signal segmentation.

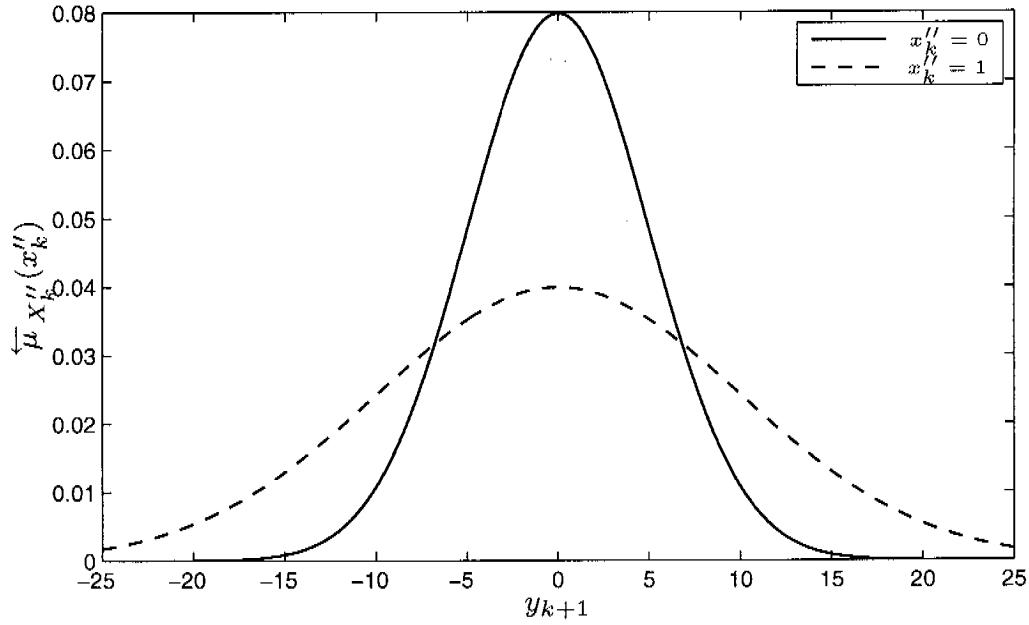


Figure 6.8: Messages for $x''_k = 0$ and $x''_k = 1$.

With the node function in (6.6) and with equation (6.17), the components of this message vector are calculated according to the following two equations:

$$\overleftarrow{\mu}_{X''_k}(0) = \frac{1}{\sqrt{2\pi}\sigma_0} e^{-y_{k+1}^2/2\sigma_0^2} \quad (6.19)$$

$$\overleftarrow{\mu}_{X''_k}(1) = \frac{1}{\sqrt{2\pi}\sigma_1} e^{-y_{k+1}^2/2\sigma_1^2}. \quad (6.20)$$

Figure 6.8 shows these two messages.

The messages in the equal nodes are calculated by using the discrete version of the sum-product rule. For example, message $\overrightarrow{\mu}_{X'_k}(x'_k)$ can be determined as follows:

$$\overrightarrow{\mu}_{X'_k}(x'_k) = \sum_{x_k} \sum_{x''_k} f_=(x_k, x'_k, x''_k) \overrightarrow{\mu}_{X_k}(x_k) \overleftarrow{\mu}_{X''_k}(x''_k) \quad (6.21)$$

$$= \sum_{x_k} \sum_{x''_k} \delta(x_k - x'_k) \delta(x_k - x''_k) \overrightarrow{\mu}_{X_k}(x_k) \overleftarrow{\mu}_{X''_k}(x''_k) \quad (6.22)$$

$$= \overrightarrow{\mu}_{X_k}(x'_k) \overleftarrow{\mu}_{X''_k}(x'_k). \quad (6.23)$$

Message $\overleftarrow{\mu}_{X'_k}(x'_k)$ in the other directions are calculated accordingly.

More interesting than the equal node is the state transition node. The message $\vec{\mu}_{X_{k+1}}(x_{k+1})$ in Figure 6.5 or Figure 6.7 can be calculated by using the discrete sum-product rule, the node function $p(x_{k+1}|x'_k)$, and the incoming message $\vec{\mu}_{X'_k}(x'_k)$:

$$\vec{\mu}_{X_{k+1}}(x_{k+1}) = \sum_{x'_k} p(x_{k+1}|x'_k) \vec{\mu}_{X'_k}(x'_k). \quad (6.24)$$

With $X'_k \in \{0, 1\}$, $X_{k+1} \in \{0, 1\}$, and Table 6.1 it follows:

$$\vec{\mu}_{X_{k+1}}(0) = \sum_{x'_k} p(0|x'_k) \vec{\mu}_{X'_k}(x'_k) \quad (6.25)$$

$$= p(0|0) \vec{\mu}_{X'_k}(0) + p(0|1) \vec{\mu}_{X'_k}(1) \quad (6.26)$$

$$= (1 - \epsilon_0) \vec{\mu}_{X'_k}(0) + \epsilon_1 \vec{\mu}_{X'_k}(1) \quad (6.27)$$

$$\vec{\mu}_{X_{k+1}}(1) = \epsilon_0 \vec{\mu}_{X'_k}(0) + (1 - \epsilon_1) \vec{\mu}_{X'_k}(1). \quad (6.28)$$

This can be written as a matrix multiplication:

$$\begin{pmatrix} \vec{\mu}_{X_{k+1}}(0) \\ \vec{\mu}_{X_{k+1}}(1) \end{pmatrix} = \begin{pmatrix} 1 - \epsilon_0 & \epsilon_1 \\ \epsilon_0 & 1 - \epsilon_1 \end{pmatrix} \cdot \begin{pmatrix} \vec{\mu}_{X'_k}(0) \\ \vec{\mu}_{X'_k}(1) \end{pmatrix}. \quad (6.29)$$

The message $\overleftarrow{\mu}_{X'_k}(x'_k)$ in the opposite direction can be calculated similarly:

$$\overleftarrow{\mu}_{X'_k}(x'_k) = \sum_{x_{k+1}} p(x_{k+1}|x'_k) \overleftarrow{\mu}_{X_{k+1}}(x_{k+1}). \quad (6.30)$$

With $X'_k \in \{0, 1\}$, $X_{k+1} \in \{0, 1\}$, and Table 6.1 it follows:

$$\overleftarrow{\mu}_{X'_k}(0) = \sum_{x_{k+1}} p(x_{k+1}|0) \overleftarrow{\mu}_{X_{k+1}}(x_{k+1}) \quad (6.31)$$

$$= p(0|0) \overleftarrow{\mu}_{X_{k+1}}(0) + p(1|0) \overleftarrow{\mu}_{X_{k+1}}(1) \quad (6.32)$$

$$= (1 - \epsilon_0) \overleftarrow{\mu}_{X_{k+1}}(0) + \epsilon_1 \overleftarrow{\mu}_{X_{k+1}}(1) \quad (6.33)$$

$$\overleftarrow{\mu}_{X'_k}(1) = \epsilon_1 \overleftarrow{\mu}_{X_{k+1}}(0) + (1 - \epsilon_1) \overleftarrow{\mu}_{X_{k+1}}(1). \quad (6.34)$$

This can be written as a matrix multiplication:

$$\begin{pmatrix} \overleftarrow{\mu}_{X'_k}(0) \\ \overleftarrow{\mu}_{X'_k}(1) \end{pmatrix} = \begin{pmatrix} 1 - \epsilon_0 & \epsilon_1 \\ \epsilon_1 & 1 - \epsilon_1 \end{pmatrix} \cdot \begin{pmatrix} \overleftarrow{\mu}_{X_{k+1}}(0) \\ \overleftarrow{\mu}_{X_{k+1}}(1) \end{pmatrix}. \quad (6.35)$$

Note that the transformation matrix from (6.29) is the transpose of the matrix from (6.35):

$$\begin{pmatrix} 1 - \epsilon_0 & \epsilon_1 \\ \epsilon_0 & 1 - \epsilon_1 \end{pmatrix} = \begin{pmatrix} 1 - \epsilon_0 & \epsilon_0 \\ \epsilon_1 & 1 - \epsilon_1 \end{pmatrix}^T. \quad (6.36)$$

Before these calculated messages are sent along the edges of the factor graph, they are scaled. This is necessary for numeric reasons. We have had good experience with normalizing the messages so that the sum of the components becomes 1, i.e.,

$$\overrightarrow{\mu}_{X_{k+1}}(x_{k+1}) = \frac{\overrightarrow{\mu}_{X_{k+1}}(x_{k+1})}{\sum_{x_{k+1}} \overrightarrow{\mu}_{X_{k+1}}(x_{k+1})}. \quad (6.37)$$

Segmenting an EMG Signal Using Message Passing

To obtain the information whether sample Y_{k+1} belongs to an active or an idle segment, we need to calculate $\mu_{\text{tot}}(x_k)$ as in (4.62):

$$\mu_{\text{tot}}(x_k) \triangleq \overrightarrow{\mu}_{X_k}(x_k) \cdot \overleftarrow{\mu}_{X_k}(x_k). \quad (6.38)$$

In fact, we compute $\mu_{\text{tot}}(x_k)$ for all k and then make a decision based on the following rule: if

$$\mu_{\text{tot}}(x_k = 1) > \mu_{\text{tot}}(x_k = 0), \quad (6.39)$$

then Y_{k+1} belongs to an active segment.

The schedule for computing the necessary messages $\overrightarrow{\mu}_{X_k}(x_k)$ and $\overleftarrow{\mu}_{X_k}(x_k)$ is as follows. After all messages $\overleftarrow{\mu}_{X''_k}(x''_k)$ have been calculated, the messages in forward and backward directions can be computed independently in two steps:

- First, one forward sum-product sweep (k is increasing) that calculates the messages $\overrightarrow{\mu}_{X_k}(x_k)$ for all k ;
- Second, one backward sum-product sweep (k is decreasing) that calculates the messages $\overleftarrow{\mu}_{X_k}(x_k)$ for all k .

Performance

The segmentation results that we obtained with this factor-graph based algorithm are very good, cf. Figure 6.1. However, for good results, the four parameters σ_0 , σ_1 , ϵ_0 , and ϵ_1 have to be carefully chosen, e.g., based on the noise level in the EMG signal.

Finally, we compared our factor-graph based segmentation algorithm to a traditional threshold-based algorithm. Both algorithms needed the same number of parameters. We found no performance differences between these two algorithms.

6.3 Estimating EMG Signal Noise Power

Estimating the Noise Power

We sometimes need to estimate the noise level in an EMG signal, for example to determine parameters for EMG signal segmentation, as mentioned in the previous section. We also need to know the noise level when resolving superpositions, as explained in Chapter 5.

A simple, reliable, and automatic algorithm for the estimation of the noise power in a band-pass filtered discrete-time EMG signal had already been explained in [135]. We sketch this approach next.

Assuming a zero mean³, the algorithm works by first calculating the mean signal power for several signal intervals:

$$\sigma_i^2 = \frac{1}{L} \sum_{k=i}^{i+L-1} s_k^2. \quad (6.40)$$

Here, s is a discrete-time EMG signal, L is the length of each interval, and i is the discrete start time of an interval.

By assuming that the smallest σ_i^2 is that of an idle segment, we can estimate the noise power of the signal:

$$\hat{\sigma}^2 = \min_i \sigma_i^2. \quad (6.41)$$

³We can make the assumption that EMG signals have a zero mean since we always bandpass filter raw EMG signals before processing them further.

Estimating the Standard Deviation of an AWGN Noise Model

When decomposing EMG signals, we assume an AWGN noise model. This means that the samples of the additive noise signal are drawn from a normal distribution. As introduced in Section 5.1,

$$\mathcal{N}(x | m_X, \sigma_X^2) \triangleq \frac{1}{\sqrt{2\pi}\sigma_X} \exp\left(-\frac{(x - m_X)^2}{2\sigma_X^2}\right), \quad (6.42)$$

is the normal or Gauss distribution with mean m_X and standard deviation σ_X . We assume $m_X = 0$ and would like to estimate the parameter σ_X^2 . A good estimate for this parameter is the sample variance σ_S^2 , which is defined as the power of the signal s with its mean μ removed:

$$\sigma_S^2 = \frac{1}{L} \sum_{k=0}^{L-1} |s_k - \mu|^2. \quad (6.43)$$

Assuming a mean of zero, we get

$$\sigma_S^2 = \frac{1}{L} \sum_{k=0}^{L-1} s_k^2. \quad (6.44)$$

When we compare (6.40) with (6.44), we see that we can use $\hat{\sigma}$ as an estimate for the parameter σ_X of our AWGN model in the factor graph for EMG signal decomposition. In practice we had to adjust this estimate to get better results⁴.

Conclusions

To estimate parameters such as the standard deviation σ_X of our AWGN model, we could also use a factor graph-based approach where we estimate parameters using expectation maximization. However, the presented simple method above works very reliably and we therefore chose it over a factor-graph based approach.

⁴We suppose that this is due to quantization noise and sub-optimal message passing.

6.4 Estimating MUAP Shapes

In Chapters 4 and 5 we assumed that all characteristic MUAP shapes were given. We also assumed that no new motor units were recruited. In practice, we often⁵ need to extract the characteristic MUAP waveforms before we run our decomposition algorithm on a measured EMG signal, see Section 3.4.

Problem Statement

Given an EMG signal, we would like to estimate the number of active motor units and their characteristic MUAP shapes. Here we assume that no recruitment of motor units occurs. We also assume short EMG signals where MUAP shapes do not change substantially over time.

Factor-Graph-Based Approach?

Although it would be nice to estimate the firing times of motor units and their corresponding MUAP shapes simultaneously within the same factor graph and message-passing framework, this is not easy. The reason for it being difficult is that we model the MUAPs using FIR filters. These filters can have up to 600 coefficients each, and there may be more than 10 such filters. The fact that there are so many parameters (filter coefficients, firing times, noise levels) to be estimated makes the problem hard.

The number of parameters could be reduced by modeling the problem at hand more carefully, for example, by using wavelets or other means to represent the MUAPs with fewer parameters.

However, given the current implementation with its many parameters, the message-passing algorithm would become computationally more expensive, more complicated, and more unstable. All this may lead to longer computation times and cases where the algorithm does not converge to the correct solution, leading to less good results.

⁵However, sometimes we can assume the characteristic MUAP shapes to be given. For example, if our algorithms from Chapters 4 and 5 are used as a plug-in of another program, e.g., EMG-LODEC, where they merely resolve (difficult) superpositions.

Since it is usually possible to easily estimate the filter coefficients without message passing, we chose to not use the latter approach here⁶. However, with more experience in the future, it might be advisable to come back to this problem.

An Intuitive Approach

There are many ways and algorithms to estimate the number of active motor units and the corresponding characteristic MUAP waveforms in an EMG signal. A simple and intuitive algorithm for estimating MUAP shapes is presented next⁷.

First, the EMG signal is segmented into active and idle segments, as explained in Section 6.2. Then the active segments are processed one after the other. For the first active segment, we create a motor unit class. Then we process the second active segment. If this segment fits into the motor unit class, it is assigned to it. Otherwise, a second motor unit class is created.

Fitting: A similarity or distance measure is used to determine whether a segment fits into a motor unit class. A good measure is the Euclidean distance between the class signal⁸, which might be the average signal of all segments already assigned to this class, and the active segment compared to the class.

Assigning: Assigning an active segment to a class means that the active segment is labeled with the number of the class. In addition, the class signal that defines the class can be

⁶In other applications we have investigated the case where both firing times and filter coefficients were estimated simultaneously using factor graphs and message-passing algorithms. The number of filter coefficients were substantially smaller in these cases. A good example is our seismosomnography project. Refer to Chapter 7 for a short review of this application.

⁷There are more sophisticated approaches such as cluster analysis to deal with this problem. This simple algorithm is partly presented here for didactic reasons, i.e., to show how characteristic MUAP shape and the number of active motor units can be estimated in principle.

⁸A class or reference signal of a motor unit class can be one MUAP generated by this motor unit. Alternatively, it may also be the average of several MUAPs generated by this motor unit [135]. We sometimes refer to a reference signal also as a characteristic MUAP of a motor unit or as a template.

updated, for example by averaging all segments that are assigned to this class.

The third active segment is then compared to all given motor unit classes. If it fits, it is assigned. If it does not fit, a new class is created. This procedure is continued until all active segments belong to one motor unit class.

To be exact, the motor unit classes at this stage are only candidates for motor unit classes. This is due to the fact that some classes may contain not single MUAPs, but superpositions or artifacts.

Therefore, in the next step we need to identify those candidate motor unit classes that may contain superpositions of MUAPs and/or artifacts. This can easily be achieved since motor units fire repeatedly and a real motor unit class will have many active segments assigned to it. A class that has only one segment assigned to it is probably not a real motor unit class.

Next, all segments from those classes that are considered not real motor unit classes are processed and the corresponding classes are deleted. Processing the segments means that the algorithm tries to decompose them given the already identified class signals.

Other Algorithms for the Estimation of MUAP Shapes

One approach was already briefly mentioned in Section 3.4: after the signal acquisition and segmentation steps, a MUAP cluster analysis is performed. The cluster analysis is a block-oriented method. Block oriented methods yield better results than sequential methods in determining the number of active motor units and their reference signals [46, 52, 135].

Since the problem of extracting MUAP templates from EMG signals was not a main part of this dissertation, we refer to already existing methods, e.g., to an algorithm described by Wellig in [135]. Wellig developed a block-oriented wavelet-based cluster analysis algorithm.

Conclusions

The described segmentation-based approaches fail if a segmentation can not be performed, e.g., if we see only superpositions in an EMG signal. In such a case, a simultaneous estimation of MUAP shapes and firing times using message passing might give a solution. But even in such an extreme case, one could design an EMG experiment so that there are only a few motor units active at the beginning, which could be identified. Then one could slowly increase the force level and identify the additional motor units one by one. However, this is a highly artificial problem and an EMG signal with only superpositions and no individual MUAPs would be very difficult to decompose in general.

6.5 Using Source Models

Although the algorithms presented in Chapters 4 and 5 can already be used directly to decompose EMG signals and not just for the resolution of superpositions in EMG signals⁹, the decomposition results are often sub-optimal. An example that illustrates this statement is given next.

We have tested our algorithms from the previous chapters with long, measured signals of up to 10 seconds¹⁰. The resulting mean squared error (MSE) between the reconstructed¹¹ signal and the measured signal was very small. The algorithm was therefore able to find model parameters (firing times in this case) that fit the measured signal well. However, the MSE was in fact smaller than the MSE for the solution that we consider to be the “correct” solution in terms of the firing times. Although the MSE was small, the firing times were not all correct.

In this case, where no explicit source model was used for the average firing frequencies, the algorithm had the freedom to fit MUAPs—in particular MUAPs with a small signal energy—to the signal to explain noise and artifacts. This often leads to a better fit than the fit when using the correct firing times to reconstruct the signal. This is especially a prob-

⁹Given the MUAP templates of all active motor units, our factor-graph-based decomposition algorithm does not need a segmented EMG signal. In principle, it can decompose an EMG signals consisting of several MUAPs and superpositions directly.

¹⁰The sampling frequency was 10 kHz in this example.

¹¹Refer to the glossary.

lem when there are many motor units with MUAPs that have a small signal energy, in cases of high noise levels, and when MUAP shapes from different motor units can not easily be distinguished.

Source Models

As we have seen in Section 3.3 and in Section 4.2, a spike train of an individual alpha motor neuron can be modeled by a stochastic point process with independent and identically distributed interspike intervals. This knowledge can be used to improve our model and ultimately the decomposition algorithm performance: we could utilize source models that favor MUAP firings at or close to the average firing rates of the corresponding motor units. In this way, the detection of irregular firing patterns of motor units can be discouraged. In fact, we have already seen that we can implement source models by using state transition nodes that represent a finite state model with state transition probabilities $p(s_{i,k+1}|s_{i,k})$.

Discussion

In general, the question of how much a-priori knowledge should be used when decomposing EMG signals is important. The answer to this question really depends on what we would like to extract from the EMG signals.

For example, if we are interested in average firing frequencies, the exact locations of individual MUAPs is not very important. In this case, we could use a source model that uses the fact that every motor unit fires repeatedly. The source model then assumes some probability distribution for the inter-spike intervals, e.g., a Gaussian or Poisson distribution.

On the other hand, since such a source model encourages regular firing patterns, unusual events such as missing MUAPs or additional MUAPs are associated with a cost and they are therefore discouraged. Hence, they might not be detected by the algorithm. So especially when we would like to discover unusual events such as doublets exactly, a source model can become a problem.

Finally, if we model the distribution of the inter-spike intervals, we have

to be aware of the fact that these statistics can change over time. The corresponding parameters should therefore be estimated and adapted, especially when decomposing long-term signals.

Should Doublets be Modeled?

A general question arises here. To what extent should we explicitly model special features in EMG signals using the factor graph language? For example, should we model events such as doublets explicitly? Doublets do not occur often and when they occur, the MUAP shapes are usually different from the normal shapes¹². All this makes the explicit modeling of doublets difficult and problematic¹³.

The quotes at the beginning of this chapter say it nicely: trying to model too many details within the factor graph framework can be difficult. Comprehensive models also often result in more complicated algorithms that need longer computation times and they also may not work as well as simpler ones¹⁴.

6.6 Using Block Processing

Although long EMG signals fit well into the RAM of modern personal computers, the number of samples in an EMG signal that can be processed “simultaneously” is limited¹⁵. Remember that a factor graph for EMG signal decomposition has one slice for each discrete time index.

There are different ways to deal with the given memory limitation. One way would be, when we process a signal from left to right, to add slices on the right end of the factor graph and delete the same number of slices on the left end. However, the number of slices that is present in memory

¹²The second MUAP of a doublet usually has a smaller amplitude and is wider than the other MUAPs of the same motor unit.

¹³Personal communication with Zoia Lateva, Rehabilitation R&D Center, VA Palo Alto Health Care System, Palo Alto, CA 94304, U.S.A., February 2006.

¹⁴More complicated models often have more parameters that need to be estimated. This can make the task harder and the message-passing algorithms that estimate the parameters may even fail to converge to the correct solution.

¹⁵Here we assume a standard PC and no use of hard disk memory. Virtual RAM that uses a hard disk would make our algorithms much slower.

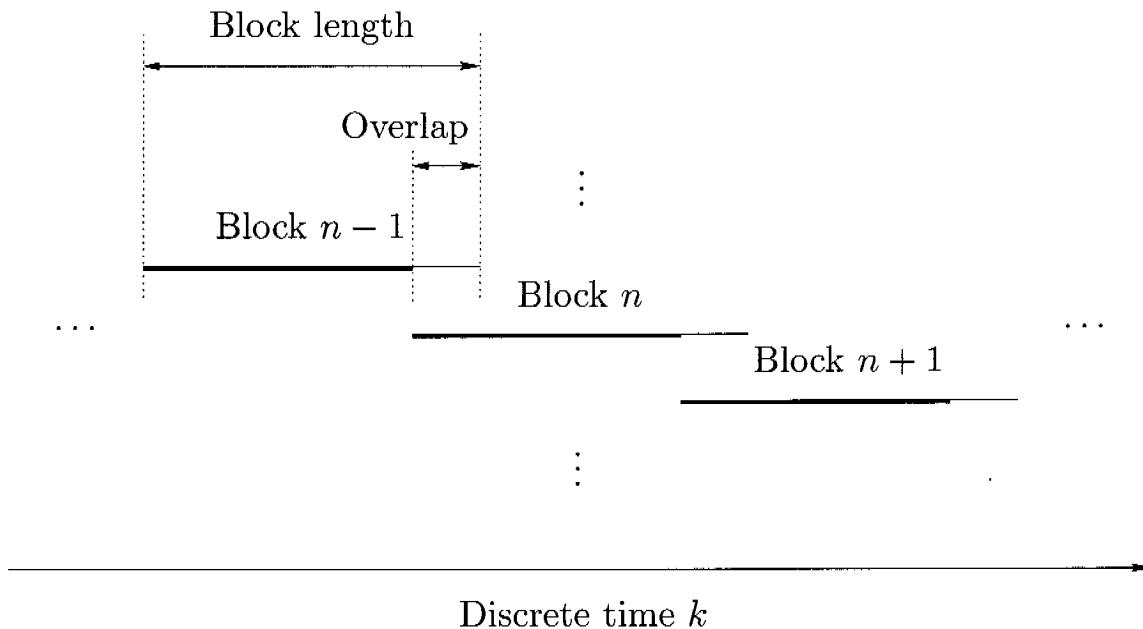


Figure 6.9: Three overlapping signal blocks. The decomposition results (firing times) from the bold parts of each block are used to make up the final (overall) decomposition result.

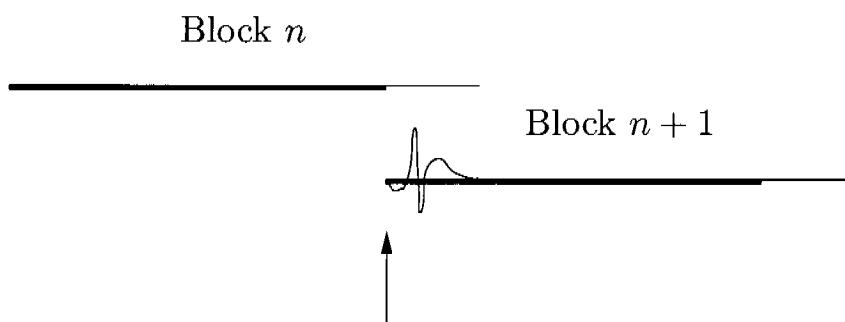
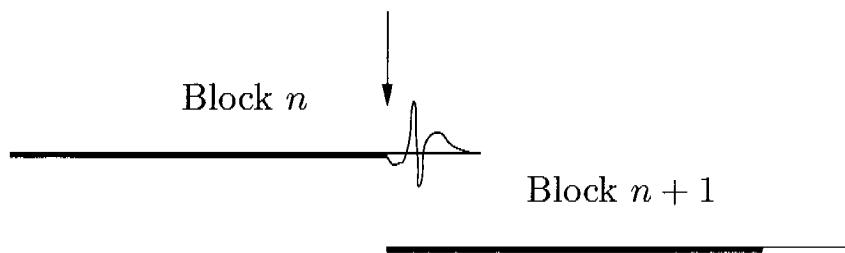
should not be too small since they store messages that we may want to use when iterating. We decided to use a block processing method instead of this *sliding factor graph* method.

Block processing means that we do not create a factor graph for the full EMG signal, which may not fit into RAM, but rather a factor graph for a shorter block of the EMG signal. Then we process the full EMG signal by dividing it up into overlapping blocks that the message-passing algorithm processes one after the other. Figure 6.9 shows the idea of our block processing method, where at any point in time, there is only one block in RAM.

The bold lines mark the parts that are used for the overall decomposition result: Up to one sample before the overlap with block $n + 1$ begins, the firing time information is taken from block n . Beginning with the sample where the overlap starts, the firing time information is taken from block $n + 1$. In this way we minimize¹⁶ the length of the necessary overlap as can be seen in Figure 6.10.

¹⁶The reason why this asymmetric readout procedure is advantageous lies in the fact that we define the firing time to be at the beginning of a MUAP.

Firing time: last sample of block n



Firing time: first sample of block $n + 1$

Figure 6.10: The firing time of a motor unit is defined to be at the location of the first sample of the MUAP. This figure shows two cases; top: the firing time is located at the last sample of block n ; bottom: the firing time is located at the first sample of block $n + 1$. We can see that the overlap should be as long as the expected duration of the longest MUAP/superposition so that the full MUAP/superposition can be processed within one block.

Remark 6.1. (Block Length and Source Model)

For example, each signal block may contain 2000 samples (for each channel). Given a sampling frequency of 20 kHz and the fact that there is one slice for each discrete time index, each signal block contains a signal that was recorded in 100 ms. As stated in Section 2.1, a typical motor unit firing frequency is 10 Hz. On average we therefore have one firing per block. Hence, when we use a source model as explained in Section 6.5, it is important to convey the information about the last firing time(s) from the current block to the next block rather than process each block completely independently of the others. \square

Remark 6.2. (Length of Overlap)

As a general rule, for a good decomposition performance the overlap should rather be too long than too short. On the other hand, it should be reasonable since a longer overlap costs additional computation time. But what is reasonable? When reading out the firing times as in Figure 6.9, the overlap should be at least as long as the longest MUAP/superposition, see also Figure 6.10. In this case, the complete MUAP/superposition can be processed within one block. Since, at least in theory, the maximum length of a superposition is the length of the EMG signal, a reasonable assumption has to be made. As mentioned in Section 2.1, a typical MUAP duration is 100...200 samples for a sampling frequency of 20 kHz. For this sampling frequency we have often used an overlap of about 300...500 samples. \square

6.7 Adapting MUAP Shapes

Due to different reasons, such as biological variability and moving electrodes, the shapes of MUAPs from the same source change over time. These changes are usually only minor from one MUAP to the next one. In such a case, the difference between the shapes can be regarded as noise and the general noise model is usually sufficient to deal with it. However, when decomposing long-term recordings that are many minutes long, such as in [135], the shapes often vary gradually but substantially over time. In this case, the filter coefficients of the FIR filters modeling the MUAPs need to be adapted.

This adaptation process could be done using a message-passing algorithm on a factor graph. For the same reasons as given in Section 6.4, we decided to adapt the FIR filter coefficients not by message passing. Instead we have implemented a weighted averaging algorithm with good results [11]: It assumes that MUAP shapes remain almost unchanged within a short signal block¹⁷. After a block has been processed, all N MUAPs from the same source within this block are aligned, resulting in the aligned MUAPs c_i . These aligned MUAPs are then averaged, e.g.,

¹⁷The signal blocks mentioned here are usually much longer than the blocks that we referred to in the context of block processing in Section 6.6.

by using a weighted arithmetic average:

$$\bar{c}_k = \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i c_{i,k}. \quad (6.45)$$

In this equation, \bar{c}_k is the resulting averaged MUAP, the w_i are the weights, and k is the discrete time index. The resulting average signal is then used to update the FIR filter coefficients, which are utilized for decomposing the next signal block.

An alternative approach would be to constantly keep the filter coefficients updated based on a weighted average of the last N MUAPs of each source. However, this approach is computationally more expensive than the block-oriented approach. Also, we found the block-oriented approach to be sufficient since the MUAP shape changes within each block are only minor if the block length is not too long.

Remark 6.3. (Averaging with Aging)

Why did we not just take the last MUAP of the current block to update the filter coefficients for the next block? The *averaging* in general is necessary due to EMG signal noise and because of MUAP variability. A *weighted* averaging method is especially useful if the amplitudes of MUAPs are either increasing or decreasing. In this way MUAPs that occurred a long time ago will have less influence on the average than those that occurred recently. This approach, which we also call *averaging with aging*, is quite common since it is well suited for adapting to slow signal changes [135]. \square

6.8 Dealing with Recruitment and Decruitment

The number of active motor units can change over time. Especially when the level of muscle force increases, the number of active motor units also increases. This is called recruitment of motor units. A comprehensive decomposition tool for measured EMG signals also needs to be able to deal with recruitment of new motor units. This includes detecting newly active motor units, creating new motor unit classes, and estimating the corresponding characteristic MUAP shapes and source parameters.

On the other hand, when the muscle force decreases, motor units are decriuted. This does not have to be explicitly modeled as long as the source model does not force firings after a certain time of inactivity. However, the performance will be better if only currently active motor units are considered since this avoids accidentally detecting a firing of a non-active motor unit due to noise, shape changes, shape similarities of MUAPs from different motor units, or superpositions that are similar to the MUAP shape of a non-active motor unit.

6.9 Summary, Conclusions, and Outlook

As we have seen in this chapter, developing a tool that can decompose measured EMG signals with high quality involves many aspects. Newly recruited motor units have to be reliably identified and their characteristic MUAP shapes need to be estimated. Especially in long-term recordings, the changing MUAP shapes of the motor units need to be tracked and the filter coefficients of the FIR filters that model the MUAPs need to be adapted accordingly. A source model is useful to deal with ambiguities but it can also bias in favor of desired/expected results. For example, a good source model should neither prevent the detection of doublets, nor should it fill in firings when they are actually missing.

Instead of implementing another comprehensive tool for EMG signal decomposition that comprises all aspects sketched in this section, it was the goal of this work to develop, implement, and evaluate completely new approaches to EMG signal decomposition with a focus on resolving superpositions. These algorithms may be used as a plug-in for more comprehensive EMG signal decomposition tools that already deal with many or all of the issues described in this chapter, e.g., EMG-LODEC [135], EMGLAB [104], Precision Decomposition [54], or others. A plug-in for EMG-LODEC has already been developed by us [9].

Chapter 7

Signal Processing Applications Beyond EMG

Jeff Hawkins says he's mapped out the way the human brain works, and has begun to fashion thinking machines to emulate the process. It comes down to Hierarchical Temporal Memory (HTM). Basically, he says, our brains take sensory inputs from the world and build a set of beliefs around the causes of those inputs. . . . Hawkins layers his machine brains with nodes that make inferences about outside sensory data, and then pass these hunches on up a hierarchy of nodes until a consensus – a belief – evolves about the source of the data. The use of “belief propagation techniques”, says Hawkins, enables an entire system to reach the best overall consensus swiftly. . . . “Stable beliefs at the top lead to changing predictions and behavior at the bottom,” says Hawkins. Where does this lead? Possibly to “machines that are much smarter than humans,” says Hawkins, computers whose abilities extend beyond sense biology and provide a means to expand such complex fields as weather, cosmology and genetics.

From MIT World's *Can a New Theory of the Neocortex Lead to Truly Intelligent Machines?* [57]

In this chapter we present three applications outside of the field of electromyography. Since these were only side-projects, we mainly concentrate on a brief description of the applications and the factor graph models.

7.1 Seismosomnography

Seismosomnography is a recording technique for non-invasive and contact-free measurements of physiological parameters during sleep. Of special interest are heart rate, respiration rate, and body movements. Contact-free means that no cables have to be attached to the human body so that the person can move freely during the recording process. In our case, the signals are measured by pressure sensors located under the four posts of a bed as can be seen in Figure 7.1. Figure 7.3 shows measured seismosomnographic signals from the four pressure sensors. Refer to [16] for more information.

7.1.1 The Seismosomnography System

The recording system by Mark Brink of ETH Zurich [6] consists of the sensors, a so-called d-box¹, a power supply unit, and input/output devices, see also Figure 7.2 and Figure 7.4. Each sensor has an AD converter. From the sensors, the digital information is sent to a controller circuit within the d-box. The controller circuit collects the data and sends them via a standard serial binary data interconnection (RS-232) interface to an embedded single board computer in the d-box. The main task of this d-box is to control the system, to store sensor data on a hard disk, and to display them on a screen. By connecting the d-box to a monitor, a keyboard, and a mouse, the d-box can be controlled and pre-programmed. The memory is sufficient to record raw data for up to 400 entire nights.

¹The name "d-box" originated from an old version of the device, which was called *Dormograph*. Personal communication with Mark Brink, ETH Zurich, Switzerland, October 2006.

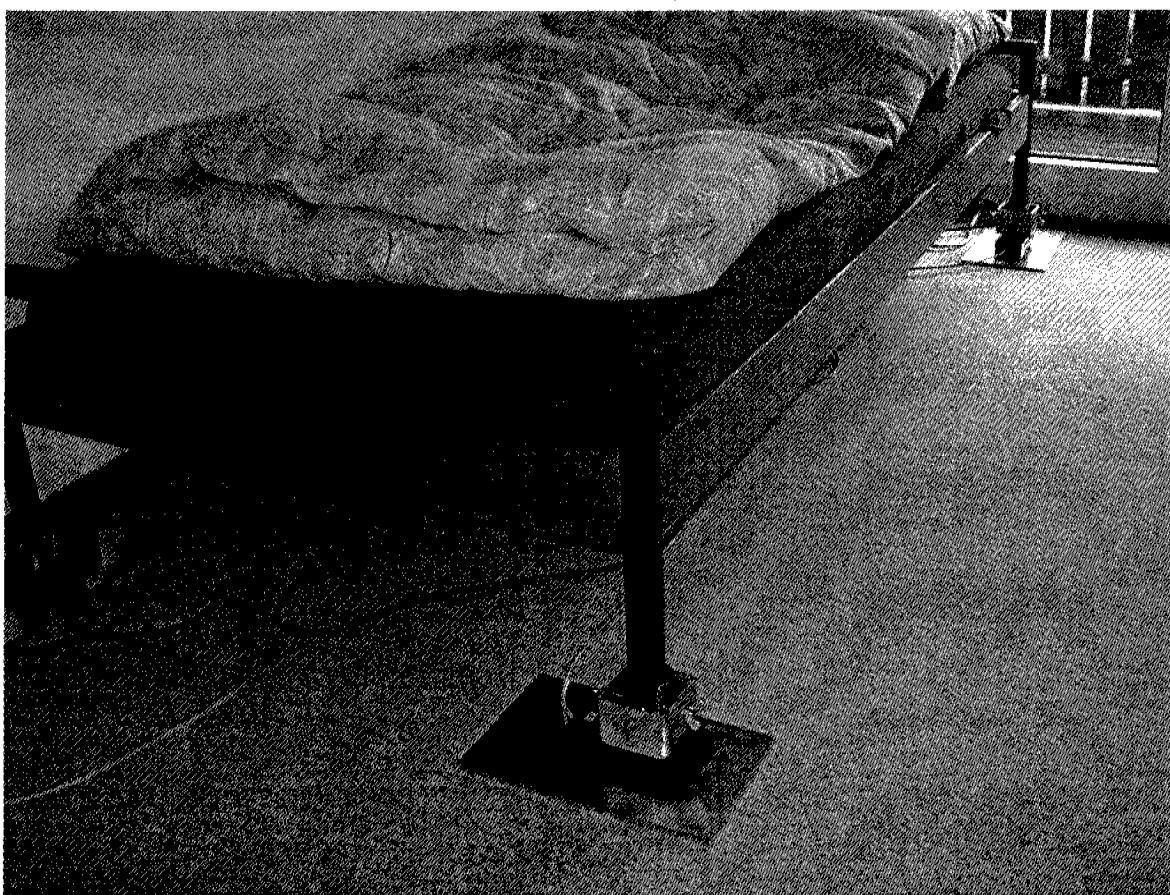


Figure 7.1: Setup of the seismosomnography system from [6] with permission (Mark Brink, ETH Zurich)

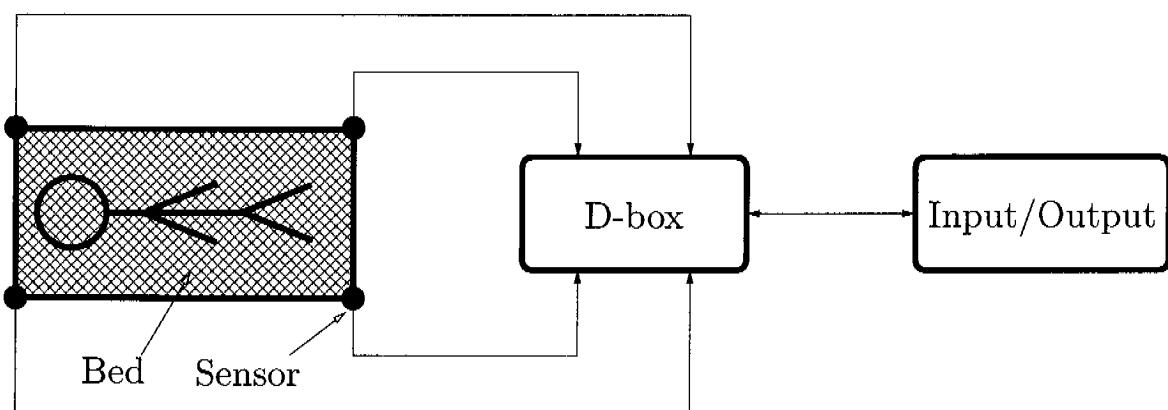


Figure 7.2: Outline of the seismosomnography system.

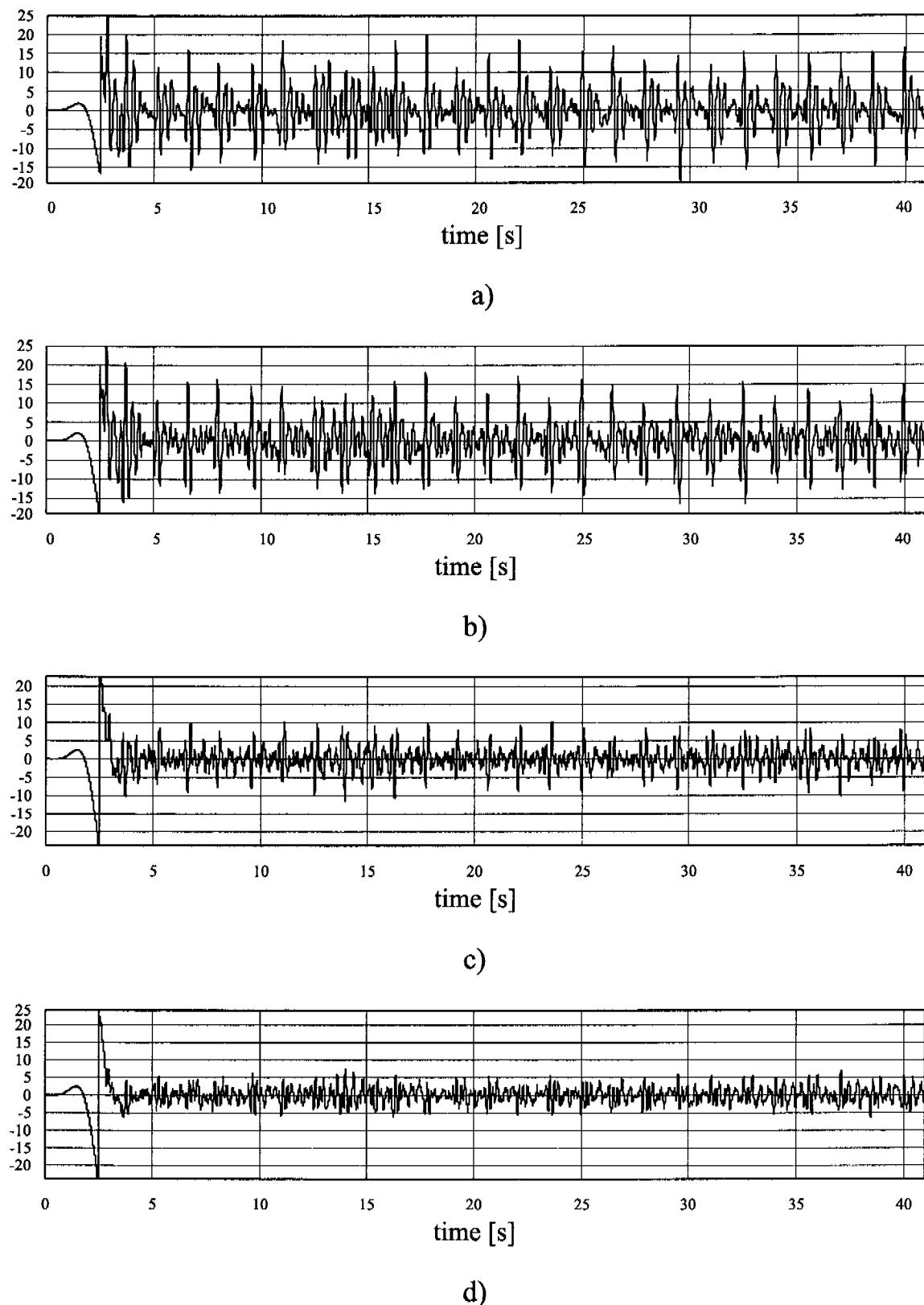


Figure 7.3: Measured seismosomnographic signals from [106].

The block diagram model is given in Figure 7.5.

Node ① stands for the heart beat pulse-generation model, which is similar to the source models that we have used to generate EMG signals. This node emits a binary signal.

Nodes ② are filters² that model the characteristic shapes of the action potentials in the four channels. Since the way from the heart through the body and the bed to each of the four sensors is different, the shapes of the action potentials will also be different and we need to model them using four different filters.

Nodes ③ model the noise in the four channels.

7.1.2 Heart Beat Detection Using Factor Graphs

We try to detect the irregular pulse train that is generated by the heart by analyzing the signals recorded by the d-box. For that we³ have developed novel multi-channel factor-graph-based message-passing algorithms. A factor graph that facilitates the development of such algorithms, and which is based on the block diagram in Figure 7.5, is given in Figure 7.6.

7.1.3 Results

We have recently [97] achieved good results when simultaneously estimating filter coefficients and the locations of heart beat spikes. The estimation is done by message passing. For estimating the filter coefficients we use expectation maximization in the form of message passing. Instead of modeling the spike shapes by IIR filters, we use an FIR filter model.

Using this approach for the decomposition of EMG signals is not straightforward:

²We have used both FIR and IIR filters to model the action potential shapes originating from the heart beat.

³Besides the author of this thesis, the following people contributed to this research project: Mark Brink, Junli Hu, Sascha Korl, Hans-Andrea Loeliger, Patrick Merkli, as well as the students Thomas Hug and Mirco Rossi [62], Christian Lüthi [97], Conradin Merk [106], and Pascal Wildbolz [138].

- In seismosomnography, we have typically used FIR filters with a filter order of $N = 7$. The computational complexity strongly increases for longer filters. In contrast, when decomposing EMG signals, we have used FIR filters with orders of up to $N = 599$.
- In EMG signal decomposition, superpositions of spikes from different sources can occur. In contrast, there are no superpositions of spikes when detecting a heart beat.
- The recent results are promising for synthetic signals. It has yet to be shown how our new algorithms perform in the case of measured data.

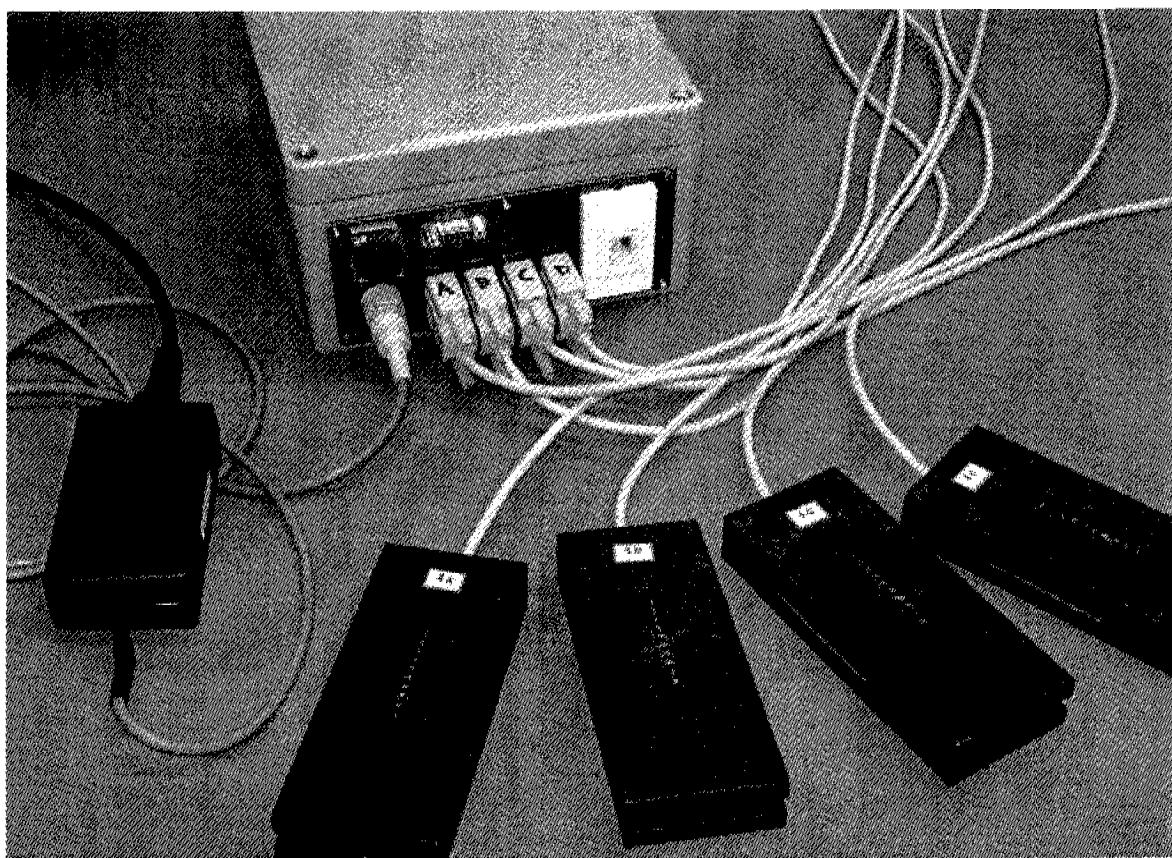


Figure 7.4: D-box, power supply, and pressure sensors of the seismosomnography system from [6] with permission (Mark Brink, ETH Zurich)

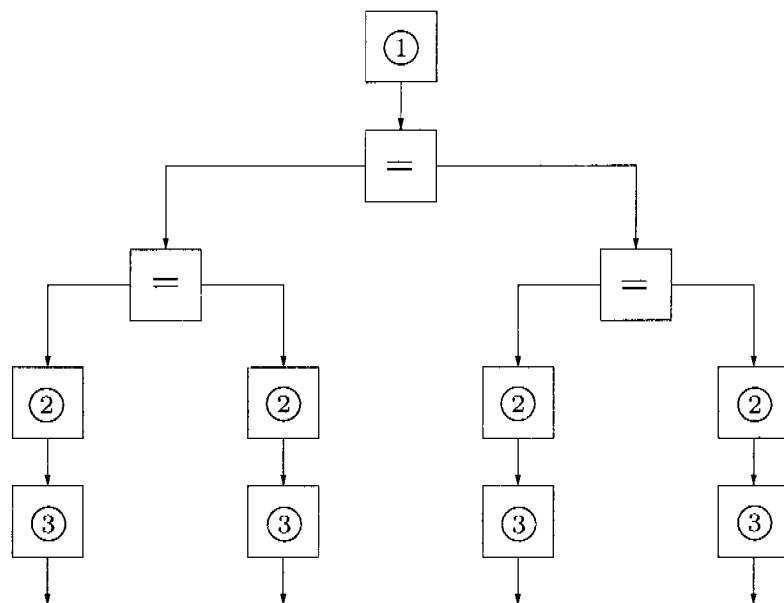


Figure 7.5: Block diagram of the seismosomnography system.

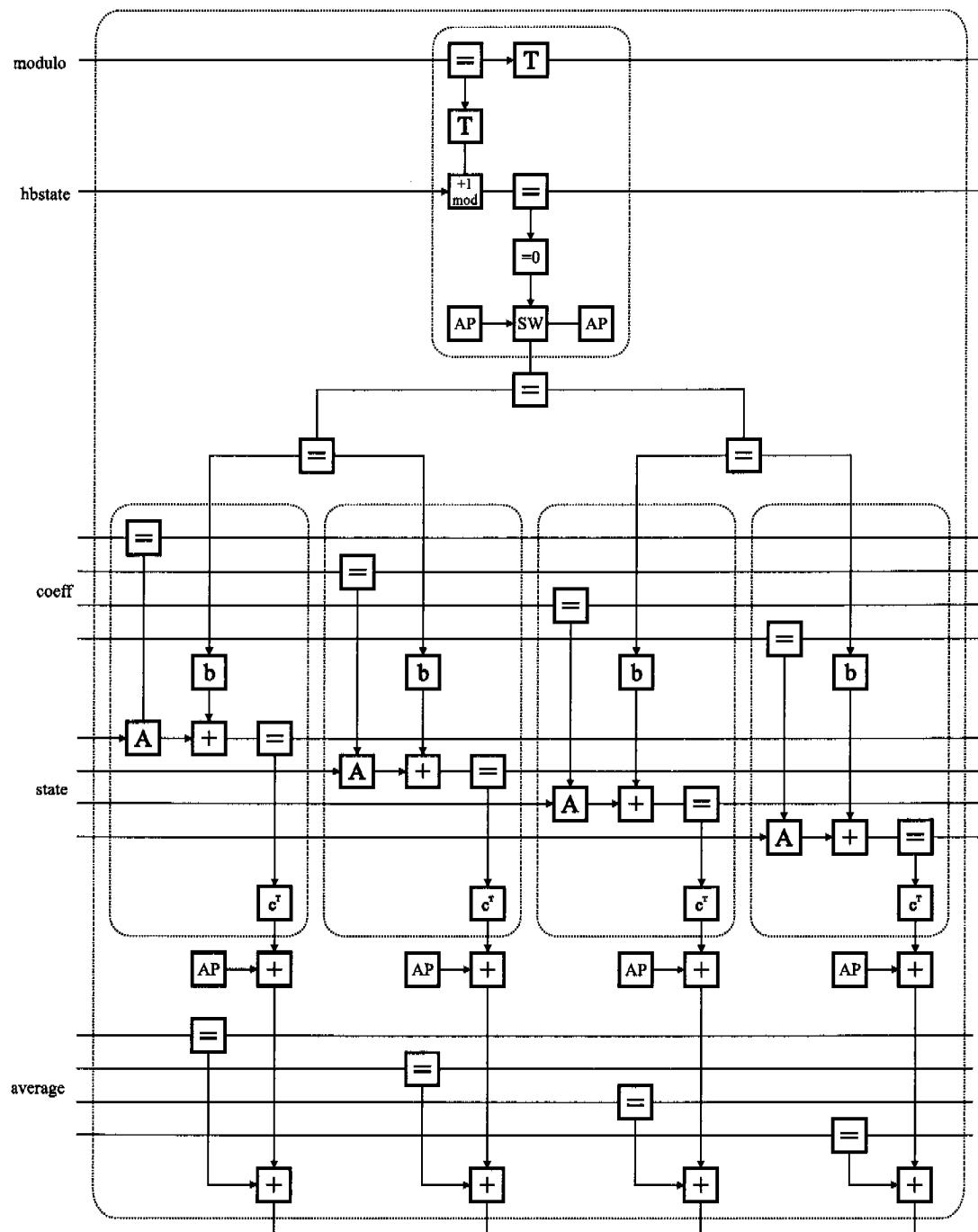


Figure 7.6: Factor graph of the seismosomnography system from [106].

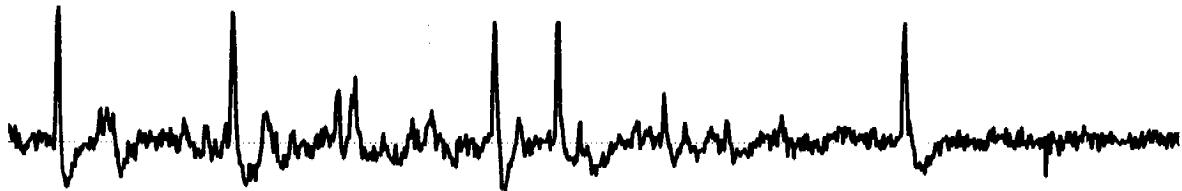


Figure 7.7: One channel of a multi-channel signal recorded with a tetrode, which was provided by Nicholas A. Lesica, Harvard University, U.S.A., September 2005.

7.2 Neural Spike Sorting

The goal of spike sorting is to identify and assign each spike (action potential) in a neural signal to the particular neuron that produced it. The neural signals are often recorded using extracellular electrodes, e.g., tetrodes, in brain tissue. Therefore, these electrodes record an extracellular voltage that is produced by an unknown number of active neurons. Figure 7.7 shows a short tetrode signal.

Spike sorting of signals recorded in neurophysiological experiments is an important topic and a big problem⁴ in neuroscience research [17]. It is a necessary first step whenever multiple spike trains are analyzed. Hence, the quality of the spike sorting process has a strong influence on the accuracy of all subsequent analyses, and therefore on the overall analysis [17].

The problem of spike-sorting is very similar to that of EMG signal decomposition. There are sources, the neurons, and each neuron creates spikes with a distinct shape. We also face many challenges here that were discussed in this thesis. For example, the characteristic spike shapes change over time since neuronal properties and experimental conditions evolve. Special tests⁵ have also shown that the classes of spike shapes from different neurons overlap partly. This fact can make error-free spike sorting impossible.

Many algorithms have been developed for spike sorting. We have used

⁴Personal communication with Nicholas A. Lesica, Harvard University, U.S.A., September 2005: “Spike sorting is a big problem: people work 1 week manually on a signal that was recorded in 4 hours.”

⁵Dual intracellular-extracellular recording studies [17]

our factor-graph-based message-passing algorithms for EMG signal decomposition for spike sorting signals⁶ that were recorded with tetrodes [118]. We found that the neural signals that we used were more difficult to deal with than many EMG signals that we have usually decomposed. The main reason for this is that the noise in our tetrode signals was higher. We hope to achieve better results in the future when our algorithms will have been further improved.

⁶The signals were provided by Nicholas A. Lesica, Harvard University, U.S.A., September 2005.



Figure 7.8: EEG research for a brain-computer interface at the RIKEN Brain Science Institute in Japan.

7.3 Blind Source Separation

7.3.1 Applications

Blind Source Separation (BSS) has many potential applications in telecommunications, medical signal processing, and signal processing in general. For example, BSS is often used to separate acoustical mixtures, e.g., two speakers or one speaker and music. BSS has also been used in EMG signal analysis [45, 87] and in the analysis of electroencephalographic (EEG) signals [23], see Figure 7.8. In electroencephalography, electrical activity of the brain is measured by recording from up to 512 electrodes [120], usually placed on the scalp. The measured signals are created by postsynaptic potentials from a large number of neurons. A comprehensive list of BSS applications is beyond the scope of this thesis.

7.3.2 Definition

Consider a linear MIMO (multiple-input/multiple-output) system

$$\mathbf{x}(t) = \mathbf{A} \cdot \mathbf{s}(t) + \mathbf{n}(t), \quad (7.1)$$

where

$$\mathbf{x}(t) \triangleq \begin{pmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_N(t) \end{pmatrix} \quad (7.2)$$

denote the N observed signals,

$$\mathbf{A} \triangleq \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,M} \\ a_{2,1} & a_{2,2} & \dots & a_{2,M} \\ \vdots & \vdots & & \vdots \\ a_{N,1} & a_{N,2} & \dots & a_{N,M} \end{pmatrix} \quad (7.3)$$

is the mixing matrix,

$$\mathbf{s}(t) \triangleq \begin{pmatrix} s_1(t) \\ s_2(t) \\ \vdots \\ s_M(t) \end{pmatrix} \quad (7.4)$$

are the M source signals, and

$$\mathbf{n}(t) \triangleq \begin{pmatrix} n_1(t) \\ n_2(t) \\ \vdots \\ n_N(t) \end{pmatrix} \quad (7.5)$$

is additive noise. Equation (7.1) is depicted in Figure 7.9.

The aim of blind source separation (BSS) is to estimate the source signals $\mathbf{s}(t)$ based on multiple sensor signals $\mathbf{x}(t)$ and some a-priori information (e.g., independence, sparsity, non-negativeness, or smoothness) about the source signals. Refer to [24] for more information on BSS.

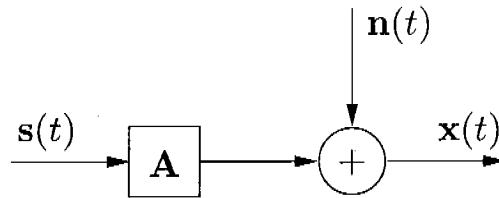


Figure 7.9: A block diagram modeling source mixing with additive noise: $\mathbf{x}(t) = \mathbf{A} \cdot \mathbf{s}(t) + \mathbf{n}(t)$.

7.3.3 BSS Using Factor Graphs

There are different methods to perform blind source separation, e.g., principal components analysis and independent component analysis. Many more blind source separation techniques have been proposed in the literature [24]. We present a new approach, where the linear mixing of source signals is modeled by using factor graphs. Iterative message-passing algorithms that are based on the sum-product algorithm pass messages along the edges of these factor graphs in order to estimate the source signals.

We consider a simple discrete-time case with two sources and two electrodes. We define

$$\mathbf{x}[k] \triangleq \begin{pmatrix} x_{1k} \\ x_{2k} \end{pmatrix} \quad (7.6)$$

$$\mathbf{A} \triangleq \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \quad (7.7)$$

$$\mathbf{s}[k] \triangleq \begin{pmatrix} s_{1k} \\ s_{2k} \end{pmatrix} \quad (7.8)$$

$$\mathbf{n}[k] \triangleq \begin{pmatrix} n_{1k} \\ n_{2k} \end{pmatrix} \quad (7.9)$$

with the following index definitions:

$$x_{\text{electrode, sample}}$$

$$a_{\text{electrode, source}}$$

$$s_{\text{source, sample}}.$$

With this, we can re-write equation (7.1):

$$\mathbf{x}[k] = \mathbf{A} \cdot \mathbf{s}[k] + \mathbf{n}[k] \quad (7.10)$$

$$\begin{pmatrix} x_{1k} \\ x_{2k} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \cdot \begin{pmatrix} s_{1k} \\ s_{2k} \end{pmatrix} + \mathbf{n}[k] \quad (7.11)$$

$$= \begin{pmatrix} a_{11}s_{1k} + a_{12}s_{2k} \\ a_{21}s_{1k} + a_{22}s_{2k} \end{pmatrix} + \mathbf{n}[k]. \quad (7.12)$$

Equation (7.12) is modeled by the factor graph in Figure 7.10 for the general case of time slice k . Note that, in this section, we use a factor graph notation as in Figure 7.10, which facilitates their implementation using our `factorgraph` package. In contrast to the factor graphs in previous chapters, we have explicitly defined port identifiers, e.g., P0, P1, and P2. We have also included terminal nodes in Figure 7.10 since they need to be implemented when using our `factorgraph` package.

Without considering the noise term $\mathbf{n}[k]$, and given that the mixed signal samples x_{1k} and x_{2k} are known, equation (7.12) has six unknown parameters for this time slice: a_{11} , a_{12} , a_{21} , a_{22} , s_{1k} , and s_{2k} . Note that these six parameters cannot be uniquely determined given only the two equations in (7.12). To be able to separate the signals one can assume certain properties of the source signals, e.g., independence. Figure 7.11 shows a factor graph for BSS. The lower part, which models the signal mixing, is identical to the factor graph in Figure 7.10. In addition, the factor graph in Figure 7.11 contains an a-priori model of the source signals, which corresponds in this case to a simple linear predictor.

7.3.4 Conclusions

In the case of EMG signal decomposition, the source signals are binary signals (firing vs. no firing). This is already an important restriction on the source signals, which can explicitly be modeled when using our factor-graph-based message-passing algorithms for EMG signal decomposition. We further restricted/modeled the source signals by using finite-state source models.

In contrast, in this section, the source signals \mathbf{s} are real-valued signals. Therefore, a source model is even more important to reduce the degrees of freedom.

Although many blind source separation algorithms are already available, our factor-graph-based approach has the advantage that it can easily be extended, e.g., by using different a-priori models for both the source signals $\mathbf{s}[k]$ and the mixing matrix \mathbf{A} . Therefore, an important difference

between conventional BSS methods and our factor graph approach is that the latter allows/requires an explicit source model.

Understanding the blind source separation problem in terms of factor graphs and iterative message passing is also of theoretical interest and might lead to powerful novel (maybe nonlinear) approaches and algorithms. We are especially interested in the cases where the number of sources is larger than the number of observed signals and where the mixing matrix $\mathbf{A}[k]$ changes over time.

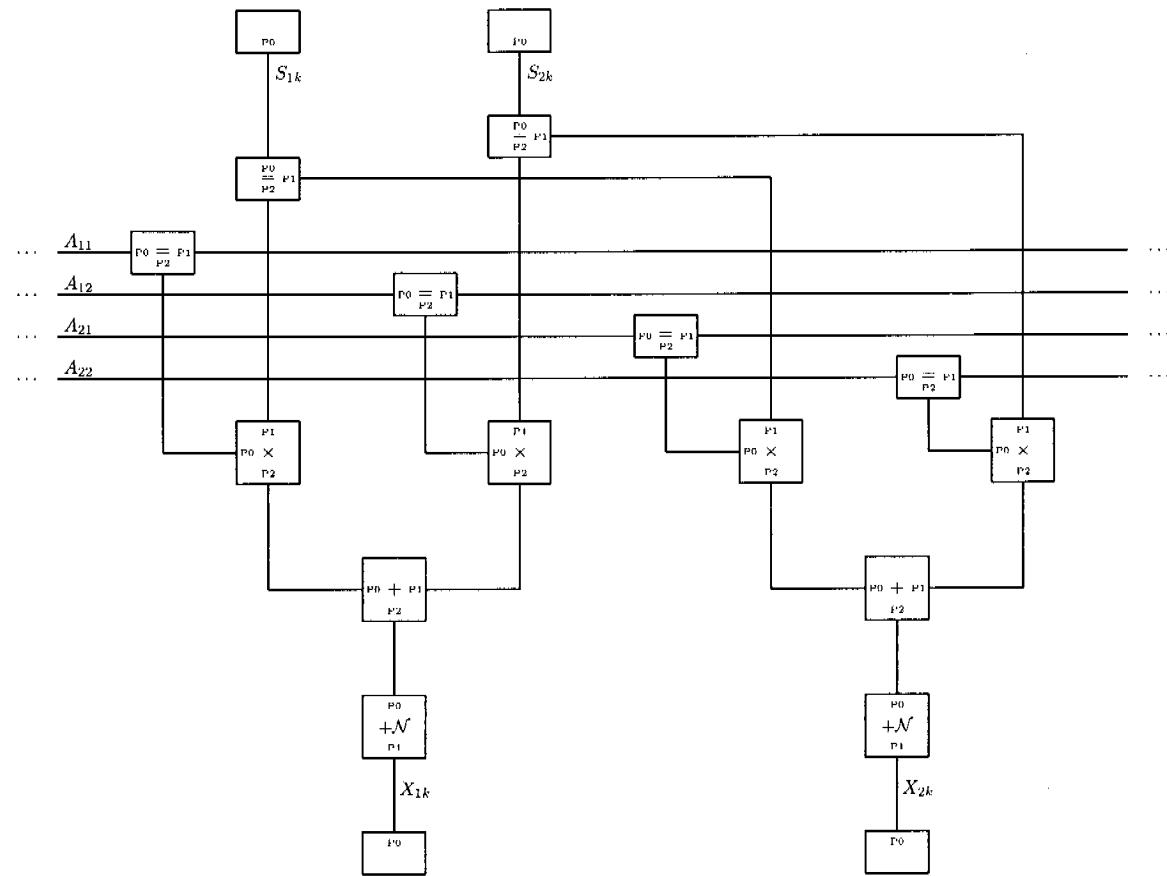


Figure 7.10: One time slice (for time k) of the factor graph for the blind source separation problem modeling $\mathbf{x}[k] = \mathbf{A} \cdot \mathbf{s}[k] + \mathbf{n}[k]$ for two sources and two electrodes.

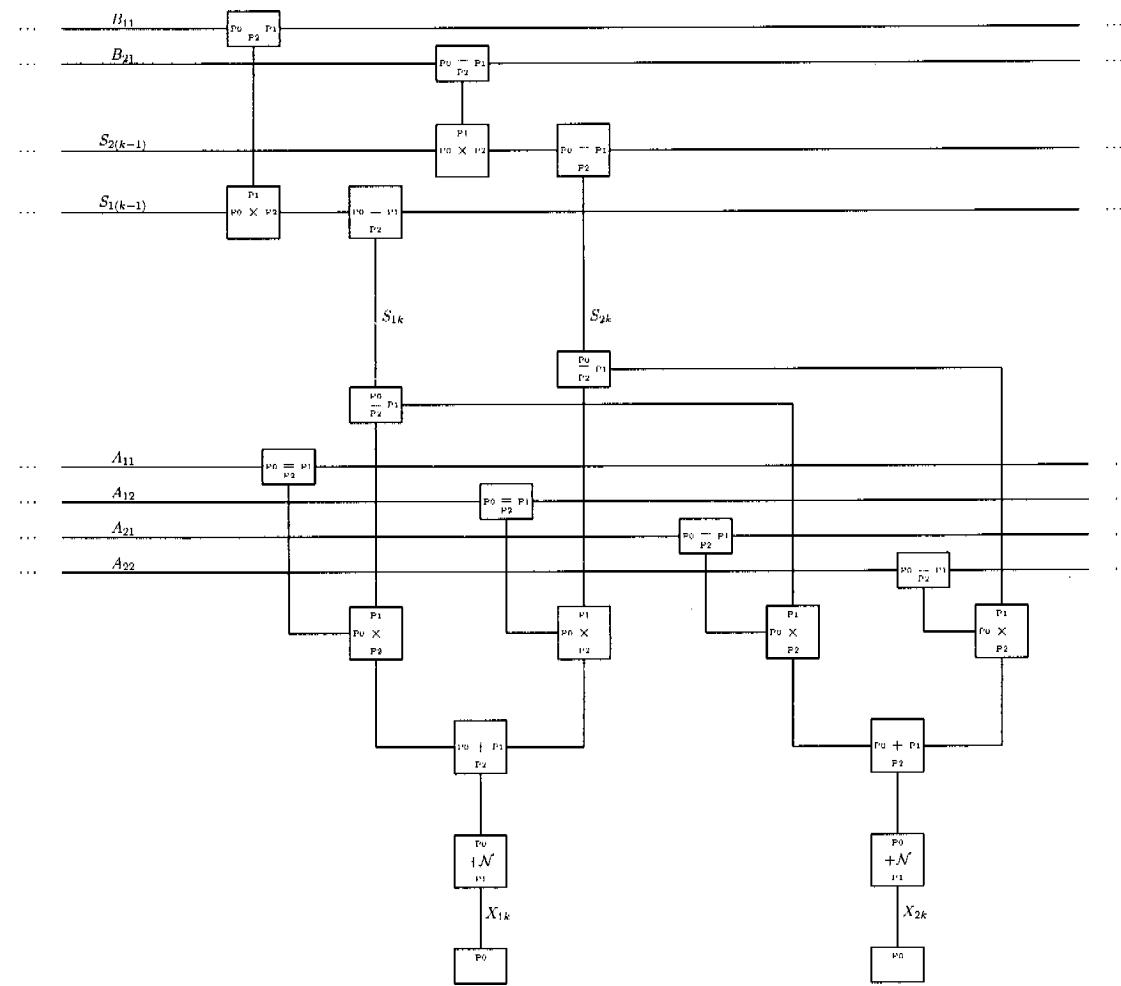


Figure 7.11: One time slice of a factor graph for BSS with a-priori model for the source signals (linear predictor), two sources, and two electrodes.

7.4 Summary, Conclusions, and Outlook

In this chapter we have presented applications beyond electromyographic signal decomposition, where factor graphs and message-passing algorithms can be applied. Although these topics were only considered briefly during this dissertation, we have gained valuable insights that also helped to improve the algorithms for EMG signal decomposition. In particular, we re-implemented and extended our `factorgraph` package⁷ so that it can deal also with scalar-Gaussian and multivariate-Gaussian message types—in addition to discrete messages. Since the applications presented here are very popular and important, it might be worthwhile to investigate them further and improve our factor-graph-based message-passing algorithms.

⁷Refer to Appendix E for more information on the `factorgraph` package.

Chapter 8

Summary, Discussion, Conclusions, and Outlook

Only those who will risk going too far can possibly find out how far one can go.

Thomas Stearns Eliot

8.1 EMG Signal Decomposition

We have developed and presented new algorithms for resolving superpositions in EMG signals. These message-passing algorithms are based on factor graphs. Factor graphs are a versatile language for signal and system modeling. They allow to naturally integrate various signal and system characteristics into a single coherent model. For example, the factor graph for EMG signal decomposition allows us to include source models, MUAP shape models, noise models, the fact that EMG signals are made up of MUAP trains and that they can be recorded by multiple electrodes, and—last but not least—the binary nature of the source signals.

The three developed message-passing algorithms for resolving superpositions mainly differ in the message types used. We started with discrete messages only (algorithm 1). By assuming scalar-Gaussian (algorithm

2) or multivariate-Gaussian distributions (algorithm 3) for certain variables in the factor graph, we could parameterize our messages. This allowed us to just send a few parameters representing scalar-Gaussian or multivariate-Gaussian distributions, e.g., mean vectors and covariance matrices in the multivariate-Gaussian case. Compared to sending discrete messages, the use of parameterized messages yielded faster algorithms.

When comparing our decomposition results with those of the (probably) best existing one by McGill [101], it turns out that McGill's algorithm is optimal in the sense that it finds the firing times which minimize the mean squared error between the reconstructed signal and the corresponding given EMG signal. In contrast, the iterative message-passing algorithms that we have developed are sub-optimal since the underlying factor graphs have cycles. They might not converge to the correct solution. In our case this means that the algorithms do not always find the solution with the smallest mean squared error.

However, the computational demand of McGill's optimal algorithm [101] increases exponentially with the duration of the MUAPs and the number of MUAPs making up a superposition. In contrast, our new message-passing algorithms show a roughly linearly growing computational demand with the durations of the MUAPs. Algorithms 2 and 3 also show a roughly linearly growing computational demand with the number of sources.

Having superpositions of many MUAPs in measured EMG signals means that many motor units are active. This makes the decomposition problem hard, independent of what algorithm one uses. One reason for this is that there are always small residuals between templates and corresponding MUAPs in an EMG signal. Such residuals might be due to MUAP shape changes such as MUAP jitter, noise, or artifacts. Also, when many MUAP shapes are available, our algorithms will wrongly find many strong superpositions in the signal. For these cases, the MSE might even be lower than the MSE for the correct solution since the algorithm fits small MUAPs so that they explain the residuals.

Nevertheless, we have decomposed very strong/difficult superpositions using our algorithms. Such superpositions cannot be resolved by most other known algorithms. Due to the sub-optimality of our iterative message-passing algorithms, the computational complexity is relatively low, even for very strong superpositions.

We have shown how our approach can in principle be used for the decomposition of EMG signals—not just for the resolution of relatively short (yet difficult) superpositions. In this case, the model may become more sophisticated. For example, a source model that represents inter-spike interval statistics can help to improve decomposition results, especially when MUAP shape information is unreliable (not distinct) and firing statistics are reliable.

Unfortunately, such explicit modeling might also have disadvantages. In general, an explicit model may be inflexible, i.e., it may discourage certain special—but possible—events. For example, when we use a source model for inter-spike intervals, we might discourage the detection of irregular firing patterns, e.g., doublets. Also, when modeling many more aspects in the factor graph, such as changing MUAP shapes, varying noise levels, or even the estimation of the FIR filter coefficients representing the MUAPs, the model may become rather complex with many free parameters. This leads to a high computational complexity and also often to poor convergence results. So it is important to be careful when deciding what to model using factor graphs and what to do outside of this framework.

It was originally hoped to develop an EMG signal decomposition algorithm that could deal with long-term EMG signals generated during dynamic muscle contractions. Such signals may be generated during tapping experiments to determine whether there are continuously active motor units during normal computer work (Cinderella hypothesis). However, the exact decomposition of such EMG signals is very difficult and currently not possible given our existing algorithms. The reason for this is that motor units are constantly recruited and derecruited. In addition, dynamic muscle contractions may lead to changing MUAP shapes and changing motor unit firing statistics. Doublets also occur more often in the case of dynamic contraction. Especially such doublets are difficult to classify correctly since the second MUAP of a doublet has often a lower amplitude and is wider than the first one, so template matching is problematic. In addition, firing statistics will be more misleading than helpful, since the inter-spike interval is very short between the two MUAPs of a doublet.

8.2 Applications Beyond EMG

We have also sketched other biomedical signal processing problems in this thesis where the factor graph approach can and has been applied.

For example, the problem of neural spike sorting, which is basically the same problem as EMG signals decomposition, is of interest to many researchers in the field of neuroscience. We tried to spike-sort neural signals that were recorded with tetrodes using our existing factor-graph-based EMG signal decomposition tools. However, the signals that we tried contained more noise than typical intra-muscular EMG signals, which makes their decomposition harder. This impression was confirmed by several experts in the field. The results were therefore not very reliable.

The problem of detecting physiological parameters such as the heart beat from pressure sensors (seismosomnography) is interesting both from an application as well as from a methodological point of view. We have developed factor graphs and message-passing algorithms for this purpose.

Finally, we have also looked at the important method of blind source separation, for which we developed both factor graphs and message-passing algorithms. The insights gained from this short-term side project helped us to better understand and to improve the algorithms used for EMG signal decomposition.

8.3 Overall Conclusions and Outlook

From applying the factor graph methodology to the problems presented and discussed in this dissertation, we have learned that the development of message-passing algorithms for real-world applications is not yet straight-forward. In fact, a lot of testing and tweaking is necessary to make an algorithm work with real, measured signals.

Despite the modular structure of factor graphs, developing message-passing algorithms based on these graphical models—to solve real-world signal processing problems—is not straight-forward, yet.

However, given our experiences with factor graphs, we believe that this new approach is promising. There are already applications where iterative message-passing algorithms yield excellent results, such as the iterative decoding of turbo codes. Recent methodological improvements and new ideas let us hope that we will soon be able to even further improve their performance in several applications, in particular for real, measured signals.

The tools that we have implemented proved helpful during the development process of new algorithms. For example, the object-oriented **factorgraph** package provides the functionality for checking certain properties of factor graphs and for debugging message-passing algorithms. We hope that this software will be found useful in future projects.

**Seite Leer /
Blank leaf**

Appendix A

Background Material

This thesis may not only be of interest to signal processing experts, but also to application-oriented readers, such as electromyography experts. Therefore, this section reviews some basic concepts in probability theory. The information is given in a simplified and intuitive way to make basic concepts clear. Generalizations are avoided. For the same reason, special restrictions are usually not noted, e.g., that a certain probability can not be zero for an equality to hold. For further information and overviews refer to [7, 14, 32, 47, 93, 139].

A.1 Probability Theory

Probability of an Event

Consider an experiment that generates random outcomes. The set of all possible outcomes is the sample space Ω . One outcome is denoted as $\omega \in \Omega$.

Example A.1. When rolling a fair dice, the sample space is $\Omega = \{1, 2, 3, 4, 5, 6\}$. \square

A probability is a number in the closed interval $[0, 1]$. We can assign probabilities to individual outcomes and to sets of outcomes, i.e., to

subsets of the sample space. A subset of a sample space is called an event and the set of all events is denoted by \mathcal{E} . Each event $A \in \mathcal{E}$ is therefore a subset of the sample space: $A \subseteq \Omega$.

Probabilities

Probabilities are defined for events only.

Example A.2. We would like to determine the probability of rolling a 3 with a fair dice. Here, the event A is defined as “rolling a 3” and $P(A) = 1/6$. \square

Example A.3. Let event B be defined as “rolling an odd number”. Then, $B = \{1, 3, 5\}$ and $P(B) = 1/2$. \square

Joint Probability

The joint probability $P(A, B)$ is the probability that both events, A and B will occur.

$$P(A, B) = P(A \cap B) \quad (\text{A.1})$$

Marginal Probability

Consider the joint probability $P(A, B)$. The marginal probability $P(A)$ is the probability of event A , ignoring any information about event B . The marginal probability is obtained by summing the joint probability over all possible outcomes of event B .

$$P(A) = \sum_B P(A, B) \quad (\text{A.2})$$

This process of “summing out a variable” is called marginalization.

Conditional Probability

The conditional probability $P(A|B)$ represents the chance that event A will occur given that event B has already occurred.

$$P(A|B) = \frac{P(A, B)}{P(B)} \quad (\text{A.3})$$

In general, since $0 \leq P(B) \leq 1$, the following inequality holds:

$$P(A|B) \geq P(A, B). \quad (\text{A.4})$$

Mutually Exclusive Events

Two events A and B are mutually exclusive, if they cannot both occur. In this case the following equation holds:

$$P(A \text{ or } B) = P(A) + P(B). \quad (\text{A.5})$$

Total Probability Theorem

Consider events A , B , and C . If the events B and C are mutually exclusive and $P(B) + P(C) = 1$, then

$$P(A) = P(A, B) + P(A, C) \quad (\text{A.6})$$

$$= P(A|B)P(B) + P(A|C)P(C). \quad (\text{A.7})$$

In general if follows for event A and mutually exclusive events B_1, B_2, \dots, B_n with $\sum_{i=1}^n P(B_i) = 1$:

$$P(A) = \sum_i P(A|B_i)P(B_i). \quad (\text{A.8})$$

Independent Events

The two events A and B are independent if

$$P(A, B) = P(A)P(B). \quad (\text{A.9})$$

If A and B are independent, then, with (A.3) and (A.9), the conditional probability $P(A|B)$ becomes

$$P(A|B) = \frac{P(A, B)}{P(B)} = \frac{P(A)P(B)}{P(B)} = P(A), \quad (\text{A.10})$$

since event B does not give any information about event A in this case. Therefore, independence means that knowing about the outcome of A does not give any information about the outcome of B , and vice versa.

Bayes' Theorem

$$P(A, B) = P(B, A) \quad (\text{A.11})$$

$$P(A|B)P(B) = P(B|A)P(A) \quad (\text{A.12})$$

$$\Rightarrow P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (\text{A.13})$$

Bayes' theorem in (A.13) [13] is widely used in probability theory. $P(A|B)$ is called the posterior probability, which is expressed by the prior probability $P(A)$ and the class-conditional probability $P(B|A)$. The denominator $P(B)$ is a normalization factor that makes the posterior probabilities sum to one.

Example A.4. (MAP estimation) The posterior probability is often used in decision and estimation problems. For example, in Section 4.6 we used symbol-wise maximum a posteriori estimation (MAP) estimation to estimate the source signal samples $x_{i,k}$:

$$\hat{x}_{i,k} = \operatorname{argmax}_{x_{i,k}} p(x_{i,k}|y). \quad (\text{A.14})$$

In general, we can use MAP estimation to estimate unobserved variables on the basis of observed data. In contrast to maximum likelihood (ML) estimation, MAP estimation uses a prior distribution over the quantity one wants to estimate, see (A.18). We can easily see this by using Bayes' theorem to rewrite the posterior probability $p(x_{i,k}|y)$ using the prior probability $p(x_{i,k})$, the class-conditional probability $p(y|x_{i,k})$, and a normalization factor $p(y)$:

$$p(x_{i,k}|y) = \frac{p(y|x_{i,k})p(x_{i,k})}{p(y)} \quad (\text{A.15})$$

We would like to sketch three examples on how we may perform MAP estimation:

Way 1: We can use Bayes' theorem to write the MAP problem in terms of the prior probability $p(x_{i,k})$ and the class-conditional probability

$p(y|x_{i,k})$ by plugging (A.15) into (A.14):

$$\hat{x}_{i,k} = \operatorname{argmax}_{x_{i,k}} p(x_{i,k}|y) \quad (\text{A.16})$$

$$= \operatorname{argmax}_{x_{i,k}} \frac{p(y|x_{i,k})p(x_{i,k})}{p(y)} \quad (\text{A.17})$$

$$= \operatorname{argmax}_{x_{i,k}} p(y|x_{i,k})p(x_{i,k}). \quad (\text{A.18})$$

Way 2: An alternative approach is to estimate the $p(x_{i,k}|y)$ directly, for example from the outputs of a neural network [13].

Way 3: As explained in Section 4.6, for EMG signal decomposition we utilized symbol-wise MAP estimation:

$$\hat{x}_{i,k} = \operatorname{argmax}_{x_{i,k}} p(x_{i,k}|y) \quad (\text{A.19})$$

$$= \operatorname{argmax}_{x_{i,k}} \frac{p(x_{i,k}, y)}{p(y)} \quad (\text{A.20})$$

$$= \operatorname{argmax}_{x_{i,k}} p(x_{i,k}, y) \quad (\text{A.21})$$

We obtain $p(x_{i,k}, y)$ by marginalization using message passing on a factor graph as in Section 4.7.

□

A.2 Gaussian Distribution

A.2.1 Scalar Case

The Gaussian distribution $\mathcal{N}(x | m_X, \sigma_X^2)$ of a random variable X with $x \in \mathbb{R}$, which is also called a normal distribution, is defined as:

$$\mathcal{N}(x | m_X, \sigma_X^2) \triangleq \frac{1}{\sqrt{2\pi}\sigma_X} \exp\left(-\frac{(x - m_X)^2}{2\sigma_X^2}\right), \quad (\text{A.22})$$

where m_X is the mean and σ_X is the standard deviation.

The parameters of this Gaussian distribution can be identified as:

$$m_X = \mathbb{E}[X] \quad (\text{A.23})$$

$$\sigma_X^2 = \mathbb{V}[X] \quad (\text{A.24})$$

$$= \mathbb{E}[X^2] - \mathbb{E}[X]^2, \quad (\text{A.25})$$

where $\mathbb{E}[X]$ is the expected value of X and $\mathbb{V}[X]$ is the variance of X .

Given samples that are drawn from a Gaussian distribution, the parameters m_X and σ_X^2 of the normal distribution can be estimated by the sample mean

$$m_X \approx \frac{1}{N} \sum_{i=1}^N x_i \quad (\text{A.26})$$

and the sample variance

$$\sigma_X^2 \approx \frac{1}{N} \sum_{i=1}^N (x_i - m_X)^2, \quad (\text{A.27})$$

where N is the number of samples.

For the case of weighted samples x_i , we get

$$m_X \approx \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_i \quad (\text{A.28})$$

$$\sigma_X^2 \approx \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i (x_i - m_X)^2 \quad (\text{A.29})$$

$$= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i (x_i^2 - 2x_i m_X + m_X^2) \quad (\text{A.30})$$

$$= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_i^2 - \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N 2w_i x_i m_X \\ + \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i m_X^2 \quad (\text{A.31})$$

$$= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_i^2 - 2m_x^2 + m_X^2 \quad (\text{A.32})$$

$$= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_i^2 - m_x^2, \quad (\text{A.33})$$

where w_i is the weight for sample x_i .

A.2.2 Vector Case

The multivariate normal distribution $\mathcal{N}_W(\mathbf{x} | \mathbf{m}_\mathbf{X}, \mathbf{W}_\mathbf{X})$ of a random variable \mathbf{X} with $\mathbf{x} \in \mathbb{R}^n$ is defined as

$$\mathcal{N}_W(\mathbf{x} | \mathbf{m}_\mathbf{X}, \mathbf{W}_\mathbf{X}) = \sqrt{\frac{|\mathbf{W}_\mathbf{X}|}{(2\pi)^n}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{m}_\mathbf{X})^\top \mathbf{W}_\mathbf{X} (\mathbf{x} - \mathbf{m}_\mathbf{X})\right), \quad (\text{A.34})$$

where $\mathbf{W}_\mathbf{X}$ is called the weight matrix, $\mathbf{m}_\mathbf{X}$ is the mean vector, and $n = \dim(\mathbf{x})$. If $\mathbf{V}_\mathbf{X} = \mathbf{W}_\mathbf{X}^{-1}$ exists, we can also write:

$$\mathcal{N}(\mathbf{x} | \mathbf{m}_\mathbf{X}, \mathbf{V}_\mathbf{X}) = \frac{1}{\sqrt{(2\pi)^n |\mathbf{V}_\mathbf{X}|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{m}_\mathbf{X})^\top \mathbf{V}_\mathbf{X}^{-1} (\mathbf{x} - \mathbf{m}_\mathbf{X})\right), \quad (\text{A.35})$$

where $\mathbf{V}_\mathbf{X}$ is the covariance matrix.

The parameters of this multivariate-Gaussian distribution can be identified as:

$$\mathbf{m}_\mathbf{X} = \mathbb{E}[\mathbf{X}] \quad (\text{A.36})$$

$$\mathbf{V}_\mathbf{X} = \mathbb{V}[\mathbf{X}] \quad (\text{A.37})$$

$$= \mathbb{E}[\mathbf{X} \mathbf{X}^T] - \mathbb{E}[\mathbf{X}] \mathbb{E}[\mathbf{X}]^T, \quad (\text{A.38})$$

where $\mathbb{E}[\mathbf{X}]$ is the expected value of \mathbf{X} and $\mathbb{V}[\mathbf{X}]$ is the covariance matrix of \mathbf{X} .

Given samples that are drawn from a multivariate-Gaussian distribution, the parameters $\mathbf{m}_\mathbf{X}$ and $\mathbf{V}_\mathbf{X}$ of the multivariate-Gaussian distribution can be estimated by the sample mean vector and the sample covariance matrix

$$\mathbf{m}_\mathbf{X} \approx \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i \quad (\text{A.39})$$

$$\mathbf{V}_\mathbf{X} \approx \frac{1}{N} \sum_{i=1}^N (\mathbf{x}_i - \mathbf{m}_\mathbf{X})(\mathbf{x}_i - \mathbf{m}_\mathbf{X})^T, \quad (\text{A.40})$$

where \mathbf{x}_i is a sample and N is the number of samples.

For the case of weighted samples \mathbf{x}_i , we get

$$\mathbf{m}_X \approx \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i \mathbf{x}_i \quad (\text{A.41})$$

$$\mathbf{V}_X \approx \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i (\mathbf{x}_i - \mathbf{m}_X)(\mathbf{x}_i - \mathbf{m}_X)^T, \quad (\text{A.42})$$

where w_i is the weight for sample \mathbf{x}_i .

For computing \mathbf{m}_X and \mathbf{V}_X with a computer program, let us define the following:

$$\mathbf{x}_i \triangleq \begin{pmatrix} x_{i,1} \\ x_{i,2} \\ \vdots \\ x_{i,l_{\max}} \end{pmatrix} \quad (\text{A.43})$$

$$\mathbf{m}_X \triangleq \begin{pmatrix} m_{X,1} \\ m_{X,2} \\ \vdots \\ m_{X,l_{\max}} \end{pmatrix} \quad (\text{A.44})$$

$$\mathbf{V}_X \triangleq \begin{pmatrix} V_{X,1,1} & V_{X,1,2} & \dots & V_{X,1,m_{\max}} \\ V_{X,2,1} & V_{X,2,2} & \dots & V_{X,2,m_{\max}} \\ \vdots & \vdots & & \vdots \\ V_{X,l_{\max},1} & V_{X,l_{\max},2} & \dots & V_{X,l_{\max},m_{\max}} \end{pmatrix}, \quad (\text{A.45})$$

where $l_{\max} = m_{\max}$.

The components of \mathbf{m}_X and \mathbf{V}_X for the case of weighted samples \mathbf{x}_i are calculated as follows:

$$m_{X,l} \approx \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_{i,l} \quad (\text{A.46})$$

$$V_{X,l,m} \approx \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i (x_{i,l} - m_{X,l})(x_{i,m} - m_{X,m}) \quad (\text{A.47})$$

$$= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i (x_{i,l}x_{i,m} - x_{i,l}m_{X,m} - m_{X,l}x_{i,m} + m_{X,l}m_{X,m})$$

(A.48)

$$= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_{i,l} x_{i,m} - \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_{i,l} m_{X,m} \\ - \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i m_{X,l} x_{i,m} + \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i m_{X,l} m_{X,m} \quad (\text{A.49})$$

$$= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_{i,l} x_{i,m} - m_{X,l} m_{X,m} - m_{X,l} m_{X,m} + m_{X,l} m_{X,m} \quad (\text{A.50})$$

$$= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i x_{i,l} x_{i,m} - m_{X,l} m_{X,m} \quad (\text{A.51})$$

Compare (A.51) to (A.38). Equations (A.46) and (A.51) are actually used in our Java program for message updates. Refer to Section 5.4.3.

Seite Leer /
Blank leaf

Appendix B

Message Update Rules

Using parameterized messages has advantages since the message update operation can often be described by means of the parameters of the distributions only. This makes message calculations computational inexpensive.

An example for parameterized messages are scalar-Gaussian and multivariate-Gaussian messages. A summary of update rules for multivariate-Gaussian messages can be found in Table B.1 on page 238. The detailed derivation is given in [88].

Refer to Appendix E for a list of symbols.

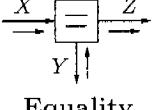
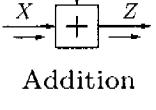
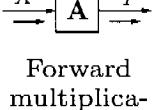
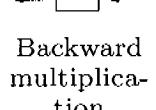
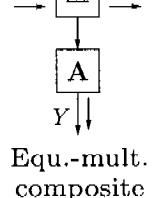
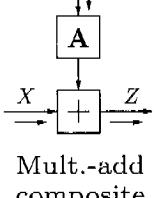
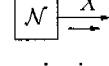
	Node	Update rule
1		$m_Z = (W_X + W_Y)^\#(W_X m_X + W_Y m_Y)$ $V_Z = V_X (V_X + V_Y)^\# V_Y$ $W_Z = W_X + W_Y$ $\xi_Z = \xi_X + \xi_Y$
2		$m_Z = m_X + m_Y$ $V_Z = V_X + V_Y$ $W_Z = W_X (W_X + W_Y)^\# W_Y$ $\xi_Z = (V_X + V_Y)^\# (V_X \xi_X + V_Y \xi_Y)$
3		$m_Y = Am_X$ $V_Y = AV_X A^\top$ $W_Y \stackrel{3}{=} A^{-\top} W_X A^{-1}$ $\xi_Y = (AV_X A^\top)^\# AV_X \xi_X \stackrel{1}{=} A^{-\top} \xi_X$
4		$m_X = (A^\top W_Y A)^\# A^\top W_Y m_Y \stackrel{2}{=} A^{-1} m_Y$ $V_X \stackrel{3}{=} A^{-1} V_Y A^{-\top}$ $W_X = A^\top W_Y A$ $\xi_X = A^\top \xi_Y$
5		$m_Z = m_X + V_X A^\top G(m_Y - Am_X)$ $V_Z = V_X - V_X A^\top G A V_X$ $W_Z = W_X + A^\top W_Y A$ $\xi_Z = \xi_X + A^\top \xi_Y$ <p>with $G = (V_Y + AV_X A^\top)^{-1}$</p>
6		$m_Z = m_X + Am_Y$ $V_Z = V_X + AV_Y A^\top$ $W_Z = W_X - W_X A H A^\top W_X$ $\xi_Z = \xi_X + W_X A H (\xi_Y - A^\top \xi_X)$ <p>with $H = (W_Y + A^\top W_X A)^{-1}$</p>
7		$m_x = m$ $V_x = V$ $W_x = W$
$\#$ denotes the Moore-Penrose pseudoinverse 1 if A and V_x are positive definite 2 if A and W_x are positive definite 3 if A is invertible		

Table B.1: Update equations for standard building blocks from [76].

Appendix C

MATLAB Program EMG-View

EMG-View is a MATLAB¹ program for loading, saving, viewing, evaluating, and editing EMG signals and EMG signal decomposition results. This program was written by the author of this thesis as a part of his doctoral research.

The topmost signal in Figure C.1 shows one channel of an eight-channel simulated EMG signal as well as the reconstructed signal, which is based on the detected firing times. It is annotated with the detected (top) and gold (bottom) “firing times” and the durations of the MUAPs of the selected sources².

The second plot shows the residual signals, i.e., the difference signals between the EMG signal and the reconstructed signals. The residual signal based on the correct (=gold) firing times from the simulation are hidden behind the corresponding signals for the detected firing times since the firing times are identical in this example.

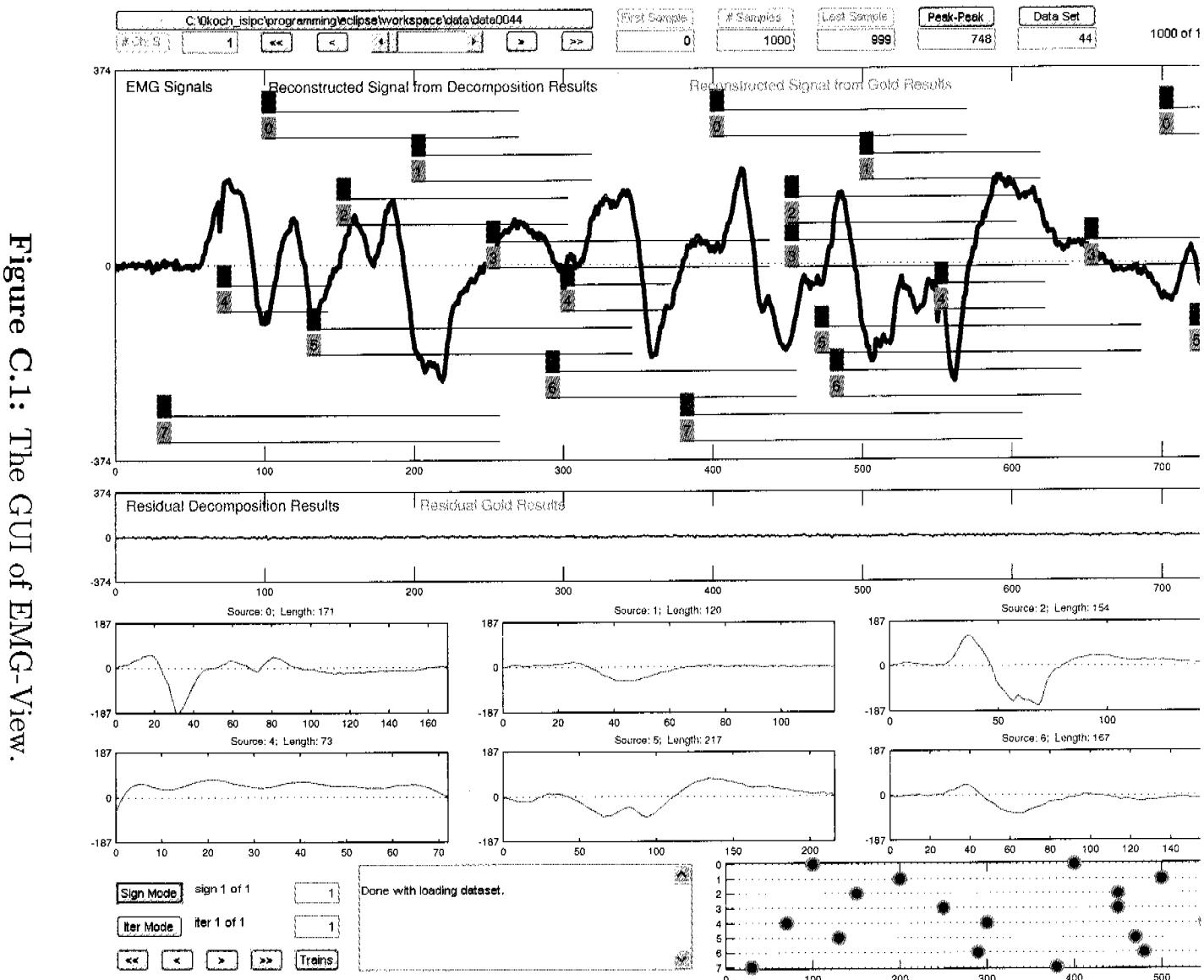
Then, the eight MUAPs are show, which were used to simulate the EMG signal. The user can use buttons to determine which MUAPs are displayed if there are more than eight MUAPs.

¹MATLAB is from The MathWorks, Natick, MA, U.S.A.

²All sources are selected in this example.

The plot at the bottom shows gold and detected firing times for the selected sources—in general for a longer signal part than the plots above.

Figure C.1: The GUI of EMG-View.



Seite Leer /
Blank leaf

Appendix D

Java Program EMG-Belief

EMG-Belief is a Java¹ program for EMG signal decomposition, which was written by the author of this thesis as a part of his doctoral research. It implements the algorithms introduced in Chapter 5 and allows:

- Setting options;
- Generating synthetic EMG signals;
- Selecting and starting decomposition algorithms;
- Running automatic decompositions on a set of signals using the same or different parameters for each signal;
- Displaying algorithm-specific information and system information at runtime;
- Evaluating decomposition algorithm performance and saving/displaying decomposition results.

The graphical user interface (GUI) provides several windows. The most important windows are described next.

Figure D.1 shows the options window. Each set of options is saved as a separate text file. Since the options are not part of the Java code,

¹Java is from Sun Microsystems, Santa Clara, CA, U.S.A.

they can be changed at runtime without restarting the program. Some options are:

- Data set number, which specifies a directory with EMG signals, MUAP shapes, gold firing times, and decomposition results;
- Logging mode, which is especially useful for debugging purposes;
- Simulation mode for decomposing many signals automatically;
- Parameters for generating synthetic EMG signals;
- Parameters for decomposing EMG signals (decomposition algorithm, message type, update schedule, stop criterion);
- Parameters for evaluating decomposition results.

Figure D.2 shows the console window, which displays information at runtime as well as final decomposition results.

Figure D.3 shows the window for selecting and inspecting current and stored—in particular discrete—messages. This is especially useful for debugging purposes. The message to plot is selected by choosing the desired time slice, the port number, and the direction of the message. Sliders allow the easy selection of messages in the factor graph, e.g., for different slices or iterations. It is also possible to record and play movies of changing messages.

Figure D.4 shows how a single discrete message is presented by EMG-Belief.

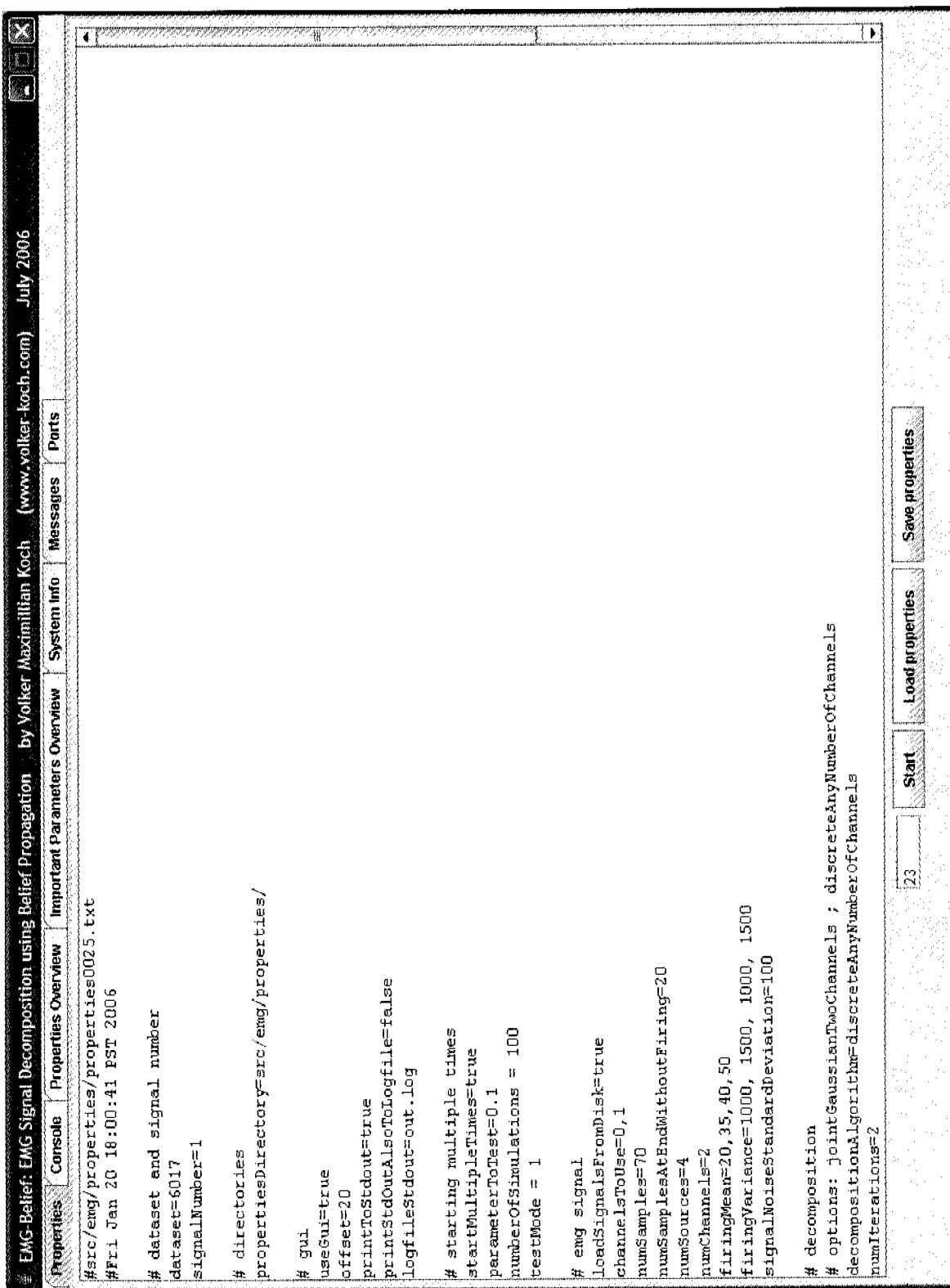


Figure D.1: The GUI of EMG-Belief. Here the properties (options) window is shown. For example, the user can determine, which signal(s) to decompose and how the decomposition is done.

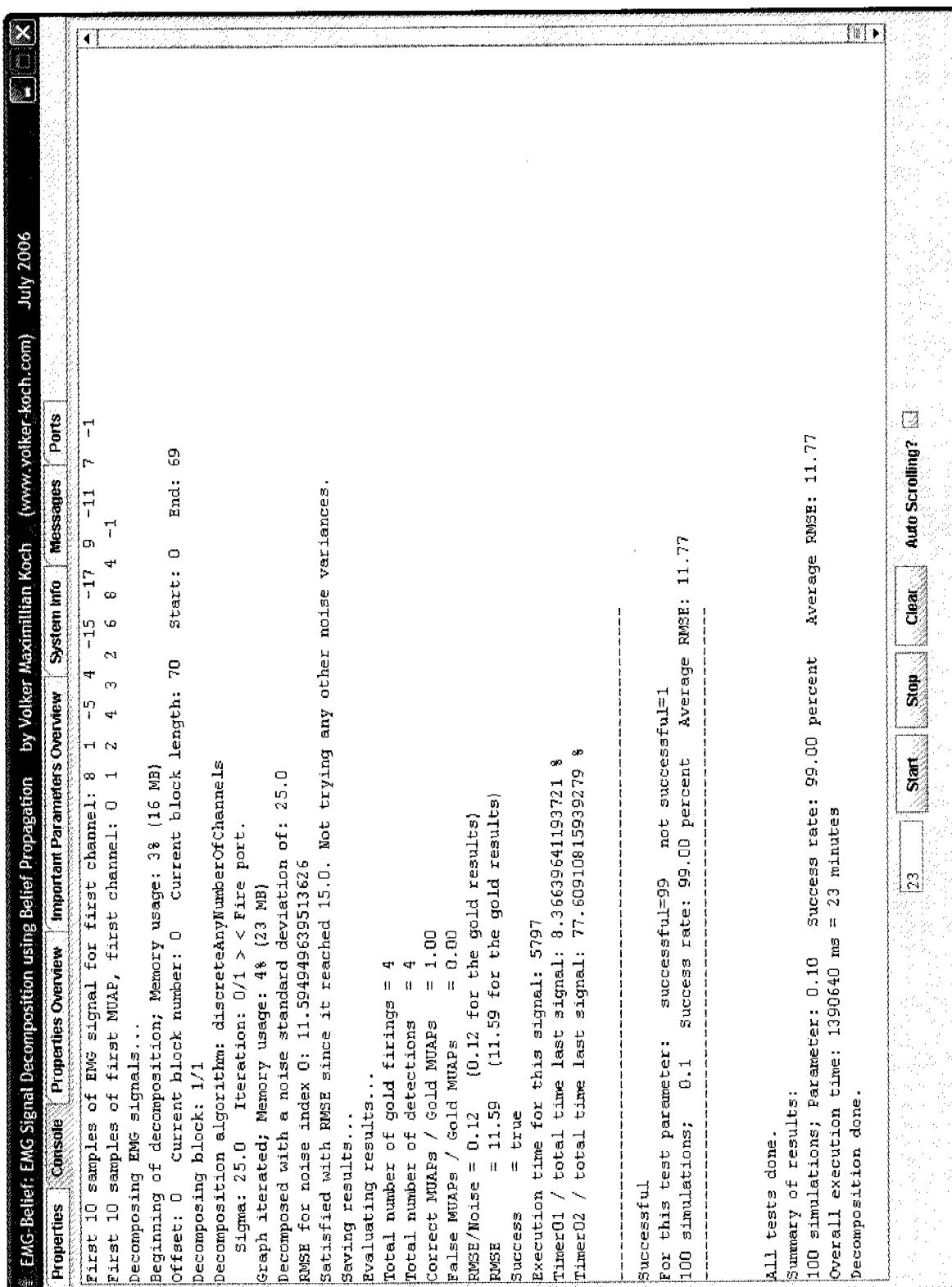


Figure D.2: The GUI of EMG-Belief. Here the console window is shown, where the user gets information during and after the decomposition, e.g., decomposition results.

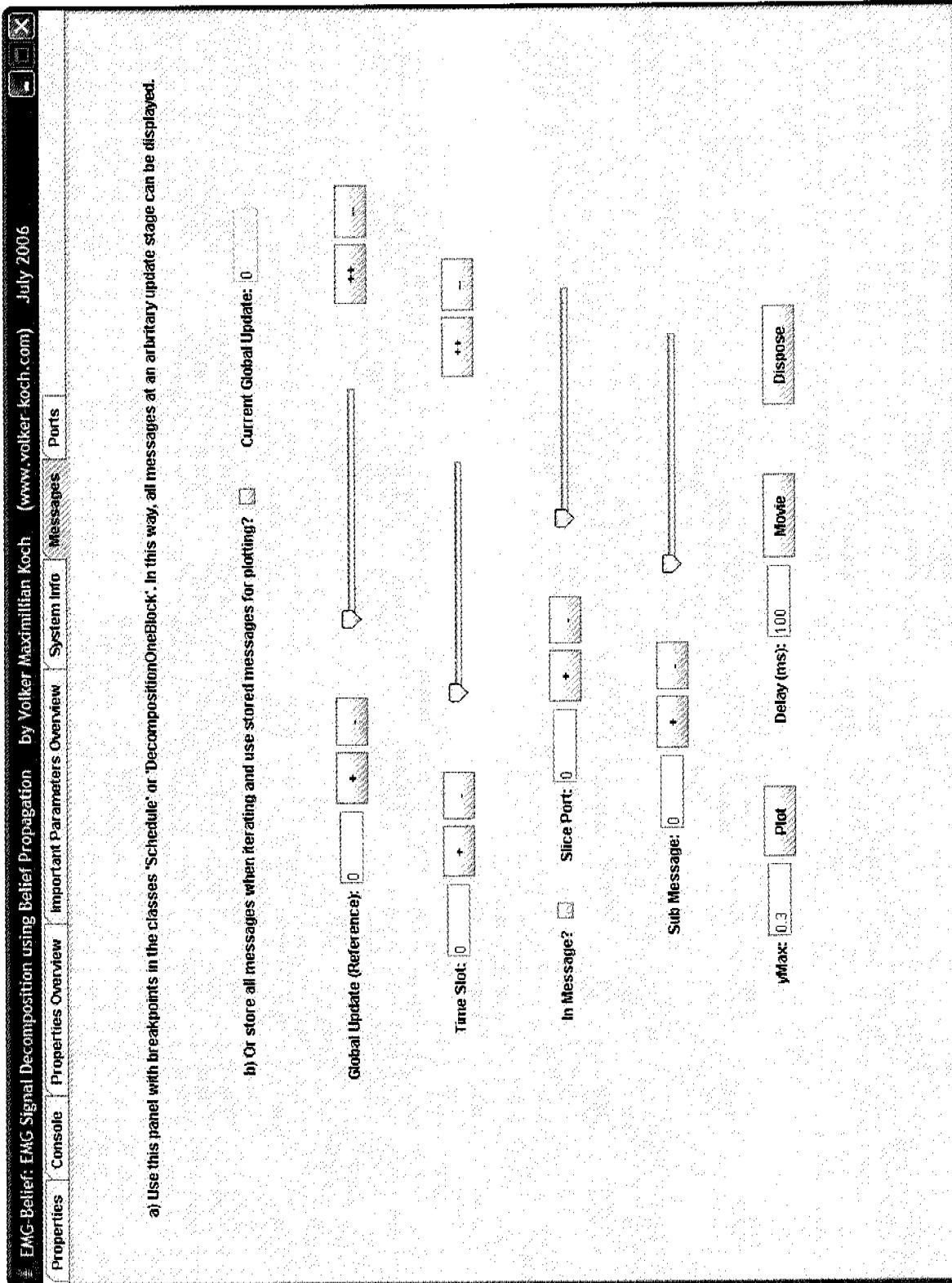


Figure D.3: The GUI of EMG-Belief. Here the window for inspecting messages in the message-passing algorithm is shown, see also Figure D.4 for a single message. This is especially useful for debugging message-passing algorithms.

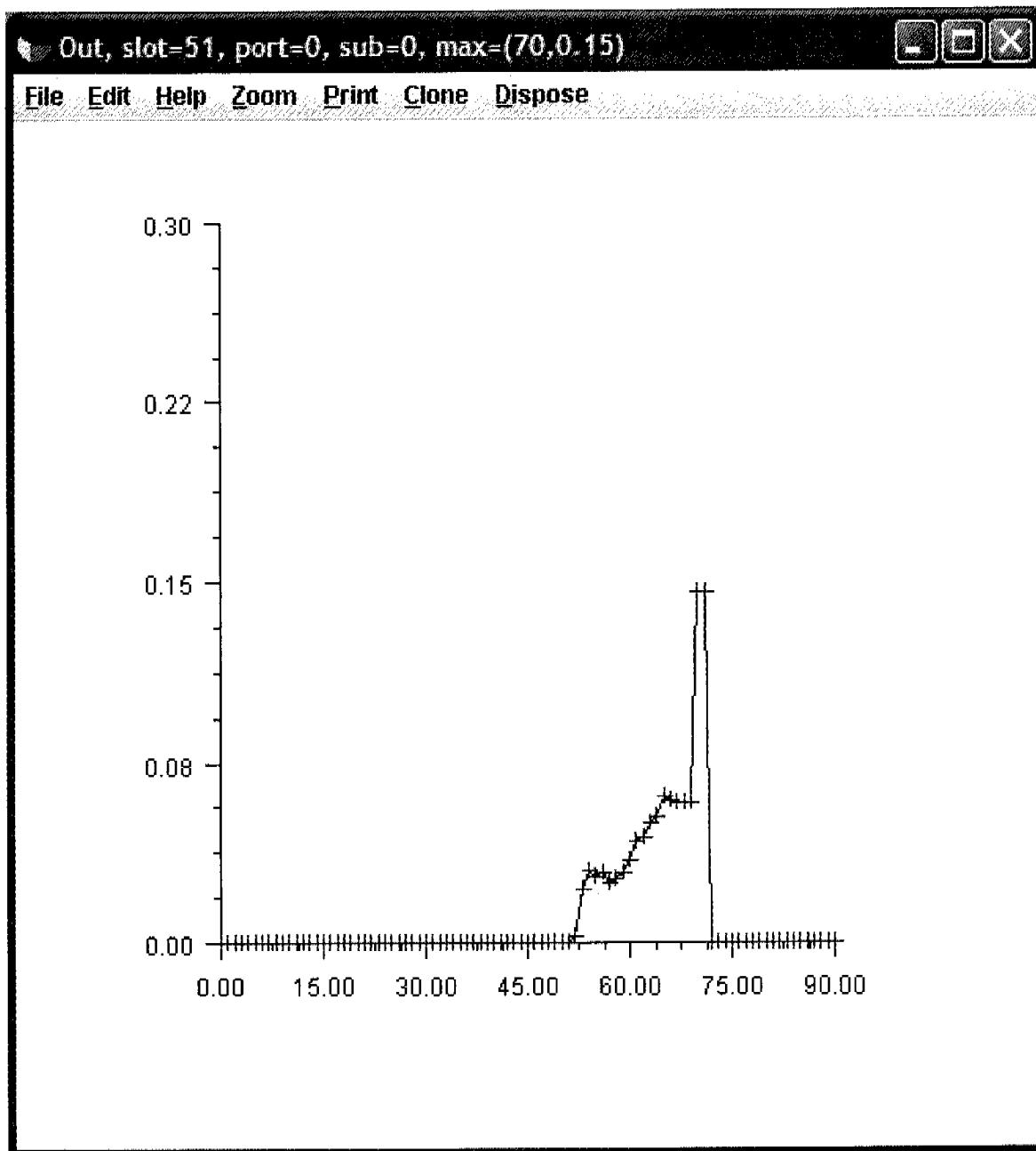


Figure D.4: The GUI of EMG-Belief. Here a single message is shown.

Appendix E

Java Package `factorgraph`

A previously developed and implemented Java¹ package [132] for factor graphs and message-passing algorithms was completely re-developed and re-implemented. The new Java package was called `factorgraph`. It facilitates the implementation of message-passing algorithms that are based on factor graphs. This package is briefly described in this section.

One basic class is called `Node`. The package allows to define the factor graph topology by connecting nodes. Since every node has a node function, the second basic class is called `Function`. Its subclasses implement specific node functions for different message types.

We have separate classes for nodes and its functions to separate the topology of the factor graph from the algorithm that runs on it. In this way we can easily change node functions without touching the topology of the factor graph.

Nodes compute messages that are sent out of these nodes along the edges of the factor graph. Instances of the class `Schedule` give the order of message computations. To define such a schedule, the user assigns a unique number to each port in the factor graph. The class `Port` is used to do this. Then the schedule is just a sequence of port numbers. Figure E.2 contains a code snipped that shows how the very simple factor graph in Figure E.1 as well as an update schedule can be defined using the package

¹Java is from Sun Microsystems, Santa Clara, CA, U.S.A.

`factorgraph`.

The class `Message` deals with storing and processing various kinds of messages. The Java package is very flexible in that it can easily be extended to deal with arbitrary message formats. It currently supports discrete messages, Gaussian messages, and multivariate-Gaussian messages.

When designing factor graphs for signal processing applications, like the ones in Chapter 5, there are many identical slices in the factor graph, one for each discrete time index. After implementing only one such slice in a subclass of the class called `FgOneTimeSlotAbstract`, objects of this class can then be instantiated for each slice. The single sections are connected to make up the full factor graph.

The Java package `factorgraph` is a general package. It provides building blocks that algorithm designers can use without having to derive every detail by themselves first (modular assembly principle). This helps researchers and developers to implement various signal processing or machine learning algorithms in a more or less straightforward way.

In summary, the `factorgraph` package is an object oriented, modular, open, reusable, and cross-platform toolbox. It facilitates the implementation of factor graphs and message-passing algorithms. Its building blocks can simply be combined, which allows rapid prototyping. In addition, new building blocks can easily be included.

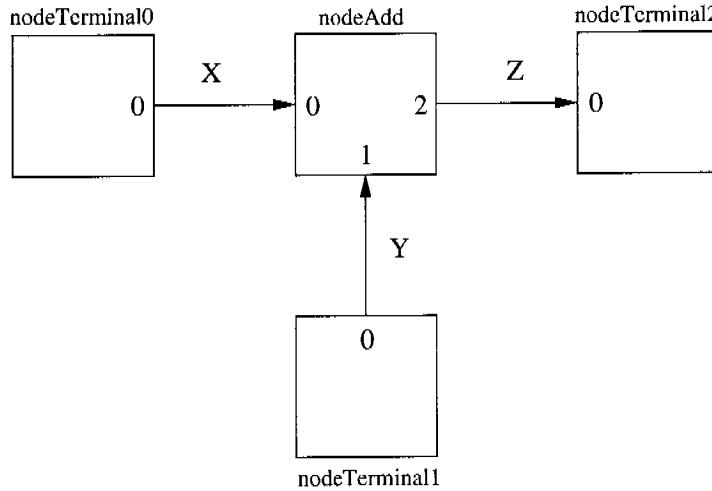


Figure E.1: A simple factor graph with one add node ($Z = X + Y$) and three terminal nodes.

```

public void createGraph(){

    // Instantiate nodes
    nodeAdd      = new Node("nodeAdd", functionAdd);
    nodeTerminal0 = new Node("nodeTerminal0", functionTerminal0);
    nodeTerminal1 = new Node("nodeTerminal1", functionTerminal1);
    nodeTerminal2 = new Node("nodeTerminal2", functionTerminal2);

    // Establish connections between nodes
    Node.connect(nodeTerminal0, 0, nodeAdd, 0);
    Node.connect(nodeTerminal1, 0, nodeAdd, 1);
    Node.connect(nodeTerminal2, 0, nodeAdd, 2);

    // Define slice port numbers
    port = new Port[numberOfPortsMax];
    addPort(nodeAdd, 0);
    addPort(nodeAdd, 1);
    addPort(nodeAdd, 2);

    // Define update schedule
    schedule0 = new Schedule(3, port, numberOfPorts);
    schedule0.addPortToSchedule(0);
    schedule0.addPortToSchedule(1);
    schedule0.addPortToSchedule(2);
}

```

Figure E.2: Source code that shows how the factor graph in Figure E.1 as well as a message update schedule can be defined using the package **factorgraph**.

Seite Leer /
Blank leaf

Abbreviations

AD converter	Analog-to-digital converter
AWGN	Additive white Gaussian noise
BSS	Blind source separation
CT	Computed tomography
EM	Expectation maximization
EEG	Electroencephalogram
EMG	Electromyography or electromyographic
EMG-Belief	Software tool for EMG signal decomposition using message-passing algorithms
EMG-LODEC	Software tool for electromyogram long-term decomposition
EMG-View	Software tool for loading, saving, viewing, and editing EMG signals
ETH Zurich	Short English version of <i>Eidgenössische Technische Hochschule Zürich</i> (ETH Zürich)
factorgraph	Java package for implementing message-passing algorithms on factor graphs
FFG	Forney-style factor graph

FG	Factor graph
FIR	Finite impulse response
FSM	Finite state model
GUI	Graphical user interface
HMM	Hidden Markov model
iff	If and only if
i.i.d.	Independent identically distributed
IRR	Infinite impulse response
IPI	Inter-pulse interval
ISI	Inter-spike interval
SI	Signal and Information Processing Laboratory
MAP	Maximum a posteriori
MMSE	Minimum mean squared error
MFAP	Muscle fiber action potential
MIMO system	Multiple-input/multiple-output system
ML	Maximum likelihood
MRI	Magnetic resonance imaging
MSE	Mean squared error
MUAP	Motor unit action potential
MVC	Maximum voluntary contraction
pdf	Probability density function
RAM	Random access memory

RMSE	Root mean squared error
RS-232	A standard for serial binary data interconnection
SEMG	Surface EMG
SNR	Signal-to-noise ratio
SPA	Summary-propagation algorithm
w.r.t.	With respect to

Seite Leer /
Blank leaf

Glossary

Correct decomposition	A simulated signal is completely correctly decomposed if the estimated source signals $\hat{X}_{i,k}$ match the source signals $X_{i,k}$ from the simulation. Therefore, no missed detections and no additional detections are allowed.
Doublet	A doublet is a collection of two MUAPs from the same motor unit, where the second MUAP occurs immediately after the first one.
Electromyography	Electromyography deals with the measurement, display, and analysis of electrical signals that are generated when muscles contract.
EMG-Belief	EMG-Belief is a Java program for EMG signal decomposition. It uses message-passing algorithms on factor graphs and it was written by the author of this thesis as a part of his doctoral research.
EMG-LODEC	EMG-LODEC (electromyogram long-term decomposition) is a software tool for decomposing long-term EMG signals. It was developed by Peter Wellig.
EMG signal	An EMG signal contains MUAPs from different motor units, artifacts, and noise. MUAPs from different motor units can overlap.

EMG signal decomposition	EMG signal decomposition aims at separating an EMG signal into its constituent MUAP trains.
EMG-View	EMG-View is a MATLAB program for loading, saving, viewing, evaluating, and editing EMG signals and EMG signal decomposition results. This program was written by the author of this thesis as a part of his doctoral research.
factorgraph Java package	A Java package, which facilitates the implementation of message-passing algorithms that are based on factor graphs.
Factor graphs	Factor graphs are graphical representations of complex mathematical models. They have a modular structure, which helps comprehending, changing, and extending the models themselves as well as the algorithms derived based on factor graphs. Factor graphs allow a unified approach to many signal processing problems and beyond.
Known-constituent problem	In the known-constituent problem, all MUAP shapes are known and all of these shapes contribute to the given superposition.
Motor unit	A motor unit, the smallest functional unit of the neuromuscular system, is the collection of an alpha motor neuron and all the muscle fibers innervated by this neuron.
MUAP	A motor unit action potential (MUAP) is a waveform that is measured when a motor unit is activated one time.
Reconstructed signal	A reconstructed signal is created by placing the MUAPs at the locations of the corresponding firings in a signal. It is therefore the EMG signal minus noise and artifacts containing only the MUAPs. (This is, of course, only the case if all MUAPs were identified correctly.)

Recruitment	To increase muscle force, idle motor units may become active, they are recruited.
Spike	We define a spike to be a short waveform. For example, an action potential such as a MUAP is a spike. A “1” in a binary source signal is also a spike.
Spike train	We define a spike train to be a signal with a series of spikes.
Sum-product rule	The sum-product rule states that an outgoing message of a node is calculated by summing over the product of the node function (corresponding to the node from which the message is sent) and all incoming messages.
Sum-product algorithm	The sum-product algorithm, which uses the sum-product rule to calculate messages, allows to simultaneously and efficiently compute all marginals in a factor graph without cycles.
Unknown-constituent problem	In the unknown-constituent problem, all MUAP shapes are known but not all of them contribute to the given superposition.

**Seite Leer /
Blank leaf**

List of Symbols

Elementary and Various

\hat{x}	Estimate of variable (or parameter) x
\triangleq	Definition
\propto	Proportional to
$\stackrel{!}{=}$	Set to equality
\square	End of a proof, remark, example, etc.
$f_m = \max_x f(x)$	f_m is the largest value of the function $f(x)$
$x_m = \operatorname{argmax}_x f(x)$	x_m is the value x that maximizes $f(x)$
\in	Set membership
\subseteq	Subset
$\sum_{\sim x}$	Summing out of all variables except x
$\delta[x]$	Kronecker delta with discrete variable x
$\delta(x)$	Dirac delta with continuous variable x
\mathbb{Z}	Commutative ring of integers [21,91], which is the set $\{\dots, -2, -1, 0, 1, 2, \dots\}$

\mathbb{Z}^n	n -dimensional space of integers
\mathbb{R}	Field of real numbers [21, 91]
$[a, b]$	Closed interval: $x \in [a, b] \Leftrightarrow a \leq x \leq b$

Matrices and Vectors

\mathbf{x}	Vector
$\dim(\mathbf{x})$	Dimension of vector \mathbf{x} , i.e., the number of its components
\mathbf{X}	Vector of random variables
\mathbf{A}	Matrix
\mathbf{A}^T	Transpose of matrix \mathbf{A}
$ \mathbf{A} $	Determinant of matrix \mathbf{A}
\mathbf{A}^{-1}	Inverse of matrix \mathbf{A}
$\mathbf{A}^\#$	Moore-Penrose pseudoinverse of matrix A
$\mathbf{0}$	A vector of zero elements

Probability Theory

Ω	The set of all possible outcome of an experiment, also called the sample space
ω	An outcome of an experiment: $\omega \in \Omega$
\mathcal{E}	The set of events
A	An event: $A \in \mathcal{E}, A \subseteq \Omega$

$P(A)$	Probability of event A
X	Random variable
x	Value of random variable X
$\mathbb{E}[X]$	Expectation of random variable X
$\text{Var}[X]$	Variance of random variable X
$p_X(x)$	Probability distribution of random variable X
$p_{X Y}(x y)$	Conditional probability distribution: “ p of x given y ”
$p_{X,Y}(x,y)$	Joint probability distribution: “ p of x and y ”

Factor Graphs

$\mu_{f \rightarrow x}(x)$	Message from node f towards edge x (old notation)
$\mu_{x \rightarrow f}(x)$	Message from edge x towards node f (old notation)
$\overrightarrow{\mu}_X(x)$	Message in direction of the arrow on the edge corresponding to variable X in the factor graph (new notation)
$\overleftarrow{\mu}_X(x)$	Message in opposite direction of the arrow on the edge corresponding to variable X in the factor graph (new notation)
$\mu_{\text{tot}}(x)$	$\overrightarrow{\mu}_X(x) \cdot \overleftarrow{\mu}_X(x)$

Gaussian Messages

$\mathcal{N}(x m_X, \sigma_X^2)$	Scalar-Gaussian distribution with mean m_X and variance σ_X^2
m_X	Mean of a Gaussian distribution
σ_X	Standard deviation of a Gaussian distribution
V_X	Variance of a Gaussian distribution: $V_X = \sigma_X^2$

Multivariate-Gaussian Messages

$\mathcal{N}(\mathbf{x} \mathbf{m}_X, \mathbf{V}_X)$	Multivariate-Gaussian distribution with mean vector \mathbf{m}_X and covariance matrix \mathbf{V}_X
$\mathcal{N}_W(\mathbf{x} \mathbf{m}_X, \mathbf{W}_X)$	Multivariate-Gaussian distribution with mean vector \mathbf{m}_X and weight matrix \mathbf{W}_X
\mathbf{m}_X	Mean column vector of a multivariate-Gaussian distribution
\mathbf{W}_X	Weight matrix of a multivariate-Gaussian distribution
\mathbf{V}_X	Covariance matrix of a multivariate-Gaussian distribution: $\mathbf{V}_X = \mathbf{W}_X^{-1}$
ξ_X	Weighted mean: $\xi_X = \mathbf{W}_X \mathbf{m}_X$

Signals and Indices

i	source or alpha motor neuron i
j	channel or electrode or sensor j
$h_{i,j,\ell}$	ℓ -th filter coefficient expressing the waveform of the MUAP generated by source i in electrode j
X_i	$X_i \triangleq (X_{i,1}, X_{i,2}, X_{i,3}, \dots)$; discrete-time binary signal emitted by source i
$T_{i,j}$	$T_{i,j} \triangleq (T_{i,j,1}, T_{i,j,2}, T_{i,j,3}, \dots)$; motor unit action potential train of source i and channel j
W_j	$W_j \triangleq (W_{j,1}, W_{j,2}, W_{j,3}, \dots)$; additive white Gaussian noise of channel j
Y_j	$Y_j \triangleq (Y_{j,1}, Y_{j,2}, Y_{j,3}, \dots)$; discrete-time EMG signal of channel j

**Seite Leer /
Blank leaf**

List of Figures

2.1	Motor unit.	14
2.2	Nerve cell.	15
2.3	EMG signal.	18
2.4	EMG experiment with surface and fine-wire electrodes. . .	19
2.5	Needle with fine-wire electrodes.	20
2.6	EMG experiment with surface electrodes.	21
2.7	EMG signal amplifier.	22
2.8	EMG recording and analysis systems.	23
3.1	Pictorial outline of decomposition.	31
3.2	Left: A block diagram modeling the EMG signal generation process for the case of two sources. Right: Example signals corresponding to the block diagram.	32
3.3	Example signals corresponding to the block diagram in Figure 3.2.	33
4.1	Block diagram model for EMG signal generation.	49
4.2	Block diagram model for two channels.	51
4.3	Factor graph.	55

4.4	Factor graph depicting a multivariate probability distribution.	58
4.5	Factor graph depicting a factorization.	58
4.6	Factor graph depicting a factorization.	59
4.7	Block diagram for $z = x + y$	59
4.8	Factor graph of an add-constraint node.	59
4.9	Block diagram for $z = x = y$	60
4.10	Factor graph of an equal-constraint node.	60
4.11	Factor graph for EMG signal decomposition.	61
4.12	Factor graph for EMG signal decomposition.	62
4.13	Exemplary factor graph.	66
4.14	Factor graph depicting a factorization.	67
4.15	Factor graph with messages.	68
4.16	Factor graphs with resulting message.	73
4.17	Factor graph with marked cycles.	74
4.18	Factor graph with marked cycles.	75
5.1	Equality constraint node.	79
5.2	Factor graph for EMG signal decomposition with discrete messages.	85
5.3	Possible locations of a MUAP in the signal.	87
5.4	Source node.	87
5.5	State transition diagram of the source model.	88
5.6	Factor graph with terminal nodes.	91
5.7	Extraction node.	94

5.8	Source and FIR state variables.	94
5.9	Coefficient node.	97
5.10	Coefficient node variables.	97
5.11	Sum constraint node.	98
5.12	Sum constraint nodes.	100
5.13	Noise node.	100
5.14	Noise node in detail.	101
5.15	Message initialization.	103
5.16	Update schedule for the forward direction.	105
5.17	Update schedule for the backward direction.	106
5.18	Message calculations during the read-out phase.	107
5.19	Factor graph for EMG signal decomposition with discrete and Gaussian messages.	108
5.20	Coefficient node.	109
5.21	Sum constraint node.	112
5.22	Noise node.	115
5.23	Factor graph for EMG signal decomposition with discrete and multivariate-Gaussian messages.	116
5.24	Coefficient node.	118
5.25	Coefficient node variables.	119
5.26	Sum constraint node.	121
5.27	Noise node.	122
5.28	Factor graph for EMG signal decomposition with discrete messages for only one electrode.	124
5.29	Known-constituent problem: exemplary superposition of four MUAPs for $\sigma_{\text{simul}}=13$	128

5.30 Known-constituent problem: exemplary superposition of four MUAPs for $\sigma_{\text{simul}}=13$	129
5.31 Known-constituent problem: decomposition results for $\sigma_{\text{simul}} = 13$	130
5.32 Known-constituent problem: decomposition results for $\sigma_{\text{simul}} = 13$	130
5.33 Known-constituent problem: exemplary superposition of four MUAPs for $\sigma_{\text{simul}}=30$	132
5.34 Known-constituent problem: exemplary superposition of four MUAPs for $\sigma_{\text{simul}}=30$	133
5.35 Known-constituent problem: decomposition results for $\sigma_{\text{simul}} = 30$	134
5.36 Known-constituent problem: decomposition results for $\sigma_{\text{simul}} = 30$	134
5.37 Known-constituent problem: exemplary superposition of four MUAPs for $\sigma_{\text{simul}}=50$	135
5.38 Known-constituent problem: exemplary superposition of four MUAPs for $\sigma_{\text{simul}}=50$	136
5.39 Known-constituent problem: decomposition results for $\sigma_{\text{simul}} = 50$	137
5.40 Known-constituent problem: decomposition results for $\sigma_{\text{simul}} = 50$	137
5.41 Known-constituent problem: decomposition results for different numbers of iteration.	139
5.42 Known-constituent problem: decomposition results for different numbers of iteration.	139
5.43 Known-constituent problem: decomposition results for different number of iterations.	140
5.44 Known-constituent problem: decomposition results for different number of iterations.	140

5.45 Known-constituent problem: average time for correctly decomposing a signal vs. the number of sources.	142
5.46 Known-constituent problem: average time for correctly decomposing a signal vs. the number of sources.	142
5.47 Known-constituent problem: average time for correctly decomposing a signal vs. the duration of the MUAPs. . .	143
5.48 Known-constituent problem: average time for correctly decomposing a signal vs. the duration of the MUAPs. . .	143
5.49 Known-constituent problem: exemplary superposition of four MUAPs for $\sigma_{\text{simul}}=100$	147
5.50 Known-constituent problem: exemplary superposition of four MUAPs for $\sigma_{\text{simul}}=13$	148
5.51 Known-constituent problem: improved decomposition results for varying σ_{simul}	149
5.52 Known-constituent problem: improved decomposition results for varying σ_{simul}	149
5.53 Known-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=10$	151
5.54 Known-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=10$	152
5.55 Known-constituent problem: improved decomposition results for $\sigma_{\text{simul}} = 10$	153
5.56 Known-constituent problem: improved decomposition results for $\sigma_{\text{simul}} = 10$	153
5.57 Known-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=30$	154
5.58 Known-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=30$	155
5.59 Known-constituent problem: improved decomposition results for $\sigma_{\text{simul}} = 30$	156

5.60 Known-constituent problem: improved decomposition results for $\sigma_{\text{simul}} = 30$	156
5.61 Known-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=50$	157
5.62 Known-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=50$	158
5.63 Known-constituent problem: improved decomposition results for $\sigma_{\text{simul}} = 50$	159
5.64 Known-constituent problem: improved decomposition results for $\sigma_{\text{simul}} = 50$	159
5.65 Possible locations of a MUAP in the signal.	161
5.66 State transition diagram of the source model.	161
5.67 Unknown-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=10$	164
5.68 Unknown-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=10$	165
5.69 Unknown-constituent problem: decomposition results for $\sigma_{\text{simul}} = 10$	166
5.70 Unknown-constituent problem: decomposition results for $\sigma_{\text{simul}} = 10$	166
5.71 Unknown-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=30$	167
5.72 Unknown-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=30$	168
5.73 Unknown-constituent problem: decomposition results for $\sigma_{\text{simul}} = 30$	169
5.74 Unknown-constituent problem: decomposition results for $\sigma_{\text{simul}} = 30$	169
5.75 Unknown-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=50$	170

5.76 Unknown-constituent problem: exemplary superposition for $\sigma_{\text{simul}}=50$	171
5.77 Unknown-constituent problem: decomposition results for $\sigma_{\text{simul}} = 50$	172
5.78 Unknown-constituent problem: decomposition results for $\sigma_{\text{simul}} = 50$	172
5.79 Unknown-constituent problem: improved decomposition results for varying σ_{simul}	174
5.80 Unknown-constituent problem: improved decomposition results for varying σ_{simul}	174
6.1 EMG signal and the corresponding segmentation result. .	180
6.2 Factor graph of a Markov chain.	181
6.3 Factor graph of a Hidden Markov Model.	182
6.4 Factor graph for EMG signal segmentation with auxiliary variables.	183
6.5 A section of the factor graph for EMG signal segmentation.	184
6.6 State transition diagram.	185
6.7 Factor graph with messages for EMG signal segmentation.	186
6.8 Gaussian messages.	187
6.9 Block processing of EMG signals.	198
6.10 Block processing of EMG signals.	199
7.1 Setup of the seismosomnography system.	205
7.2 Outline of the seismosomnography system.	205
7.3 Measured seismosomnographic signals.	206
7.4 D-box, power supply, and pressure sensors of the seismo- somnography system.	209

7.5	Block diagram model of the seismosomnography system. . .	209
7.6	Factor graph of the seismosomnography system.	210
7.7	Tetrode signal.	211
7.8	EEG research for a brain-computer interface	213
7.9	A block diagram modeling source mixing with additive noise.	215
7.10	Factor graph for BSS.	218
7.11	Factor graph for BSS.	219
C.1	The GUI of EMG-View.	241
D.1	The GUI of EMG-Belief: options window.	245
D.2	The GUI of EMG-Belief: console window.	246
D.3	The GUI of EMG-Belief: inspecting messages.	247
D.4	The GUI of EMG-Belief: display of a message.	248
E.1	Factor graph.	251
E.2	Java source code representing a factor graph.	251

Bibliography

- [1] “EMGLAB,” *World Wide Web*, July 2006. [Online]. Available: <http://emglab.stanford.edu/>
- [2] “International Ergonomics Association,” *World Wide Web*, July 2006. [Online]. Available: <http://www.iea.cc/ergonomics/>
- [3] “muscle,” *Encyclopædia Britannica*, July 2006. [Online]. Available: <http://search.eb.com/eb/article-9110701>
- [4] “muscle disease,” *Encyclopædia Britannica*, July 2006. [Online]. Available: <http://search.eb.com/eb/article-58884>
- [5] “nervous system, human,” *Encyclopædia Britannica*, July 2006. [Online]. Available: <http://search.eb.com/eb/article-75525>
- [6] “Seismosomnography,” *World Wide Web*, July 2006. [Online]. Available: <http://www.seismosomnography.ethz.ch/>
- [7] “Wolfram MathWorld,” *World Wide Web*, August 2006. [Online]. Available: <http://mathworld.wolfram.com/>
- [8] R. Abt and M. Diefenbacher, “Whisker-based obstacle avoidance in the AMouse project,” Semester thesis, ETH Zurich, Zurich, Switzerland, December 2003.
- [9] M. Andre and C. Rüegg, “Emglink: Programmierung eines Interfaces zwischen EMG-LODEC und EMG-Signal,” MTU thesis, ETH Zurich, Zurich, Switzerland, March 2006.
- [10] J. V. Basmajian and C. J. De Luca, *Muscles Alive: Their Functions Revealed by Electromyography*, 5th ed. Baltimore, MD: Williams & Wilkins, 1985.

- [11] G. Böcherer, “Adaption of a priori models in EMG signal decomposition,” Semester thesis, ETH Zurich, Zurich, Switzerland, July 2005.
- [12] F. Bellemare, J. J. Woods, R. Johansson, and B. Bigland-Ritchie, “Motor-unit discharge rates in maximal voluntary contractions of three human muscles,” *J Neurophysiol*, vol. 50, no. 6, pp. 1380–1392, 1983. [Online]. Available: <http://jn.physiology.org/cgi/content/abstract/50/6/1380>
- [13] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, USA, 1996.
- [14] I. F. Blake, *An introduction to applied probability*. Wiley, 1979.
- [15] L. Bolliger, “EMG signal decomposition using factor graphs and message-passing algorithms with joint-Gaussian and discrete messages,” Semester thesis, ETH Zurich, Zurich, Switzerland, December 2005.
- [16] M. Brink, C. H. Müller, and C. Schierz, “Contact-free measurement of heart rate, respiration rate, and body movements during sleep,” *Behavior Research Methods*, vol. 38, no. 3, pp. 511–521, 2006.
- [17] E. N. Brown, R. E. Kass, and P. P. Mitra, “Multiple neural spike train data analysis: state-of-the-art and future challenges,” *Nat Neurosci*, vol. 7, no. 5, pp. 456–461, May 2004. [Online]. Available: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1469612/>
- [18] S. Bruhin and B. Amsler, “Zerlegung von EMG-Signalen mit Faktorgraphen,” Semester thesis, ETH Zurich, Zurich, Switzerland, July 2003.
- [19] S. Bruhin and B. Amsler, “Zerlegung von EMG-Signalen mittels graphischen Modellen und iterativen Algorithmen,” Master’s thesis, ETH Zurich, Zurich, Switzerland, May 2004.
- [20] M. Byr öd, “Segmentation and decomposition of EMG signals,” Semester thesis, ETH Zurich, Zurich, Switzerland, May 2004.
- [21] P. J. Cameron, *Introduction to Algebra*. Oxford University Press, USA, 1998.
- [22] N. Cedraschi, “Computing information rates of channels with phase noise,” Diploma (Master’s) thesis, ETH Zurich, Zurich, Switzerland, September 2003.

- [23] A. Cichocki, S. L. Shishkin, T. Musha, Z. Leonowicz, T. Asada, and T. Kurachi, "EEG filtering based on blind source separation (BSS) for early detection of Alzheimer's disease," *Clin Neurophysiol*, vol. 116, no. 3, pp. 729–737, Mar 2005. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=15721088>
- [24] A. Cichocki and S.-i. Amari, *Adaptive Blind Signal and Image Processing*. John Wiley & Sons, 2002.
- [25] E. A. Clancy, D. Farina, and G. Filligoi, "Single-channel techniques for information extraction from the surface EMG signal," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 133–168.
- [26] D. M. Connelly, C. L. Rice, M. R. Roos, and A. A. Vandervoort, "Motor unit firing rates and contractile properties in tibialis anterior of young and old men," *J Appl Physiol*, vol. 87, no. 2, pp. 843–852, 1999. [Online]. Available: <http://jap.physiology.org/cgi/content/abstract/87/2/843>
- [27] J. R. Cram, "Biofeedback applications," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 435–451.
- [28] J. H. G. Dauwels, "On graphical models for communications and machine learning: algorithms, bounds, and analog implementation," Doctoral Thesis, ETH Zurich, May 2006.
- [29] C. J. De Luca, "Precision decomposition of EMG signals," *Methods in Clinical Neurophysiology*, vol. 4, pp. 1–28, 1993.
- [30] C. J. De Luca and W. J. Forrest, "An electrode for recording single motor unit activity during strong muscle contractions," *IEEE Trans Biomed Eng*, vol. 19, no. 5, pp. 367–372, Sep 1972. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=5038391>
- [31] R. F. Dougherty, V. M. Koch, A. A. Brewer, B. Fischer, J. Modersitzki, and B. A. Wandell, "Visual field representations and locations of visual areas V1/2/3 in human visual cortex," *Journal of Vision*, vol. 3, no. 10, article 1, pp. 586–598, October 2003. [Online]. Available: <http://journalofvision.org/3/10/1/>

- [32] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.
- [33] K. J. Erb, "Erkennung von Signalen motorischer Einheiten im Elektromyogramm," Doctoral Thesis, ETH Zurich, 1978.
- [34] H. Etawil and D. Stashuk, "Resolving superimposed motor unit action potentials," *Med Biol Eng Comput*, vol. 34, no. 1, pp. 33–40, Jan 1996. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=8857310>
- [35] D. Farina, A. Crosetti, and R. Merletti, "A model for the generation of synthetic intramuscular EMG signals to test decomposition algorithms," *IEEE Trans Biomed Eng*, vol. 48, no. 1, pp. 66–77, Jan 2001. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=11235593>
- [36] D. Farina, E. Fortunato, and R. Merletti, "Noninvasive estimation of motor unit conduction velocity distribution using linear electrode arrays," *IEEE Trans Biomed Eng*, vol. 47, no. 3, pp. 380–388, Mar 2000. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=10743780>
- [37] D. Farina and R. Merletti, "Estimation of average muscle fiber conduction velocity from two-dimensional surface EMG recordings," *J Neurosci Methods*, vol. 134, no. 2, pp. 199–208, Apr 2004. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=15003386>
- [38] D. Farina, R. Merletti, and C. Disselhorst-Klug, "Multi-channel techniques for information extraction from the surface EMG," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 169–203.
- [39] D. Farina, R. Merletti, and D. F. Stegeman, "Biophysics of the generation of EMG signals," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 81–105.
- [40] F. Felici, "Applications in exercise physiology," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 365–379.

- [41] R. P. Feynman, R. Leighton, E. Hutchings, and A. R. Hibbs, ‘*Surely You’re Joking, Mr. Feynman!*’ (*Adventures of a Curious Character*). W. W. Norton & Company, 1997.
- [42] J. Forney, G.D., “Codes on graphs: normal realizations,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 520–548, 2001.
- [43] M. U. Frey, “On analog decoders and digitally corrected converters,” Doctoral Thesis, ETH Zurich, April 2006.
- [44] C. Frigo and R. Shiavi, “Applications in movement and gait analysis,” in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 381–401.
- [45] G. A. Garcia, R. Nishitani, R. Okuno, and K. Akazawa, “Independent Component Analysis as preprocessing tool for decomposition of surface-electrode array electromyogram,” in *4th International Symposium on Independent Component Analysis and Blind Source Separation (ICA2003)*, Nara, Japan, April 2003.
- [46] A. Gerber, “Detection and estimation for the decomposition of electromyograms,” Doctoral Thesis, ETH Zurich, 1984.
- [47] L. Gonick and W. Smith, *Cartoon Guide to Statistics*. Collins, 1994.
- [48] R. Grüninger, “Optimization of an iterative EMG signal decomposition algorithm,” Semester thesis, ETH Zurich, Zurich, Switzerland, December 2003.
- [49] R. Gut, “Exakte Zerlegung von Elektromyogrammen mittels Viterbi Algorithmus,” Doctoral Thesis, ETH Zurich, 1992.
- [50] R. Gut and G. S. Moschytz, “High-precision EMG signal decomposition using communication techniques,” *IEEE Transactions on Signal Processing*, vol. 48, no. 9, pp. 2487–2494, September 2000.
- [51] A. Gygi, “Analyse des Nadel- und Oberflächen-Elektromyogramms mittels Statistiken höherer Ordnung,” Doctoral Thesis, ETH Zurich, 1994.
- [52] W. F. Haas, “Methoden der Mustererkennung zur automatischen Zerlegung des Elektromyogramms,” Doctoral Thesis, ETH Zurich, 1989.

- [53] S. Hamid Nawab, R. Wotiz, and C. De Luca, "Improved resolution of pulse superpositions in a knowledge-based system EMG decomposition," in *26th Annual International Conference of the Engineering in Medicine and Biology Society (EMBC)*, San Francisco, CA, U.S.A., 2004, pp. 69–71.
- [54] S. Hamid Nawab, R. Wotiz, and C. De Luca, "Multi-receiver precision decomposition of intramuscular EMG," in *28th Annual International Conference of the Engineering in Medicine and Biology Society (EMBC)*, New York, NY, U.S.A., August 30–September 3 2006, pp. 1252–1255.
- [55] S. Hamid Nawab, R. Wotiz, L. Hochstein, and C. D. Luca, "Next-generation decomposition of multi-channel EMG signals," in *24th Annual International Conference of the Engineering in Medicine and Biology Society (EMBC)*, 2002, pp. 36–37.
- [56] B. Hammarberg, "A signal processing approach to practical neurophysiology—a search for improved methods in clinical routine and research," Doctoral Thesis, Uppsala University, 2002. [Online]. Available: <http://www.signal.uu.se/Research/rneurophys.html>
- [57] J. Hawkins, "Can a new theory of the neocortex lead to truly intelligent machines?" *MIT World*, September 2005. [Online]. Available: <http://mitworld.mit.edu/video/316/>
- [58] J. Hawkins and S. Blakeslee, *On Intelligence*. Times Books, 2004.
- [59] G. M. Hägg, B. Melin, and R. Kadefors, "Applications in ergonomics," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 343–363.
- [60] L. Hochstein, S. H. Nawab, and R. Wotiz, "An AI-based software architecture for a biomedical application," in *Proceedings of The 6'th World Multiconference on Systemics, Cybernetics and Informatics*, vol. XI, Orlando, Florida, U.S.A., July 2002, pp. 60–64.
- [61] H.-C. Hopf and A. Struppner, *Elektromyographie. Lehrbuch und Atlas*. Thieme Georg Verlag, 1974.
- [62] T. Hug and M. Rossi, "Seismosomnography: Heartbeat detection from pressure sensors," Semester thesis, ETH Zurich, Zurich, Switzerland, July 2006.

- [63] C. Koch, *Biophysics of Computation: Information Processing in Single Neurons (Computational Neuroscience Series)*. New York, NY, USA: Oxford University Press, Inc., 1999.
- [64] C. Koch, *The Quest for Consciousness: A Neurobiological Approach*. Roberts & Company Publishers, 2004.
- [65] V. M. Koch, “Novel methods for the visualization and analysis of functional maps in cortex,” Diploma (Master’s) thesis, Universität Karlsruhe and Stanford University, Zurich, Switzerland, July 2001.
- [66] V. M. Koch and H.-A. Loeliger, “Decomposition of electromyographic signals by iterative message passing in a graphical model,” in *26th Annual International Conference Engineering in Medicine and Biology Society (EMBC 2004)*, San Francisco, California, U.S.A., September 1-5 2004.
- [67] V. M. Koch and H.-A. Loeliger, “A new EMG signal decomposition approach using factor graphs,” in *XVth Congress of the International Society of Electrophysiology & Kinesiology (ISEK 2004)*, Boston, MA, U.S.A., June 18-21 2004.
- [68] V. M. Koch and H.-A. Loeliger, “A novel EMG signal decomposition approach using a graphical model and an iterative algorithm,” in *Fifth international scientific conference on prevention of workrelated musculoskeletal disorders (Premus 2004)*, Zurich, Switzerland, July 11-15 2004.
- [69] V. M. Koch and H.-A. Loeliger, “EMG signal decomposition by loopy belief propagation,” in *30th IEEE International Conference on Acoustics, Speech, and Signal Processing (IEEE ICASSP 2005)*, Philadelphia, PA, U.S.A., March 18-23 2005, pp. 397–400.
- [70] V. M. Koch and H.-A. Loeliger, “Resolution of superpositions in EMG signals using belief propagation,” in *Gemeinsame Jahrestagung der Deutschen, Österreichischen und Schweizerischen Gesellschaften für Biomedizinische Technik*, Zurich, Switzerland, September 6-9 2006, p. V193.
- [71] V. M. Koch, K. C. McGill, and H.-A. Loeliger, “Resolution of superpositions in EMG signals using belief propagation: results for the known constituent problem,” in *28th Annual International*

Conference of the IEEE Engineering in Medicine and Biology Society (IEEE EMBC 2006), New York City, NY, U.S.A., August 30-September 3 2006, pp. 1260–1263.

- [72] V. M. Koch, K. C. McGill, and H.-A. Loeliger, “Resolution of superpositions in EMG signals using belief propagation: results for the unknown constituent problem,” in *Symposium on Electromyography—Principals and Applications*, Zurich, Switzerland, April 27 2006, p. 15.
- [73] V. M. Koch, A. R. Wade, R. F. Dougherty, and B. A. Wandell, “Automatic identification of retinotopic visual areas,” presented at the Society for Neuroscience’s 31st Annual Meeting, Orlando, Florida, U.S.A., 2001.
- [74] V. Koch, “Quantitative clinical electromyography using EMG signal decomposition—electromyographic signal detection, analysis and decomposition,” Course project, University of Waterloo, Waterloo, ON, Canada, April 1997.
- [75] V. Koch, “Quantitative clinical electromyography using EMG signal decomposition: evaluation of decomposition algorithm performance,” Semester thesis, University of Waterloo, Waterloo, ON, Canada, August 1997.
- [76] S. Korl, “A factor graph approach to signal modelling, system identification and filtering,” Doctoral Thesis, ETH Zurich, July 2005.
- [77] F. Kschischang, B. Frey, and H.-A. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.
- [78] Y. Kunz, “Algorithmen zur Zerlegung überlagerter EMG-Signale,” Semester thesis, ETH Zurich, Zurich, Switzerland, July 2005.
- [79] Z. C. Lateva and K. C. McGill, “Satellite potentials of motor unit action potentials in normal muscles: a new hypothesis for their origin,” *Clin Neurophysiol*, vol. 110, no. 9, pp. 1625–1633, Sep 1999. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=10479030>
- [80] Z. C. Lateva and K. C. McGill, “Estimating motor-unit architectural properties by analyzing motor-unit action potential morphology,” *Clin Neurophysiol*, vol. 112, no. 1, pp. 127–135,

- Jan 2001. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=11137670>
- [81] Z. C. Lateva, K. C. McGill, and M. E. Johanson, "Electrophysiological evidence of adult human skeletal muscle fibres with multiple endplates and polyneuronal innervation," *J Physiol*, vol. 544, no. Pt 2, pp. 549–565, Oct 2002. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=12381826>
- [82] Z. C. Lateva, K. C. McGill, and M. E. Johanson, "Increased jitter and blocking in normal muscles due to doubly innervated muscle fibers," *Muscle Nerve*, vol. 28, no. 4, pp. 423–431, Oct 2003. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=14506713>
- [83] R. S. LeFever and C. J. De Luca, "Decomposition of action potential trains," in *8th Annual Meeting of the Society for Neuroscience*, 1978, p. 229.
- [84] R. S. LeFever and C. J. De Luca, "A procedure for decomposing the myoelectric signal into its constituent action potentials—part i: Technique, theory, and implementation," *IEEE Trans Biomed Eng*, vol. 29, no. 3, pp. 149–157, Mar 1982. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=7084948>
- [85] R. S. LeFever, A. P. Xenakis, and C. J. De Luca, "A procedure for decomposing the myoelectric signal into its constituent action potentials—part ii: Execution and test for accuracy," *IEEE Trans Biomed Eng*, vol. 29, no. 3, pp. 158–164, Mar 1982. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=7084949>
- [86] V. Lesser, H. Nawab, and F. Klassner, "IPUS an architecture for the integrated processing and understanding of signals," *Artificial Intelligence*, vol. 77, pp. 129–171, 1995.
- [87] Q. Li, J. hai Yang, X. Chen, Z. Liang, and Y. xuan Ren, "The decomposition of surface EMG signals based on blind source separation of convolved mixtures," in *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, Sept. 2005, pp. 5912–5915.
- [88] H.-A. Loeliger, "Least squares and Kalman filtering on Forney graphs," *Codes, Graphs, and Systems*, pp. 113–135, 2002. [Online]. Available: <http://www.isi.ee.ethz.ch/~loeliger/>

- [89] H. A. Loeliger, *Lecture Notes: Algebra, Codes, and Signal Processing*, ETH Zurich, Switzerland, 2003.
- [90] H.-A. Loeliger, “An introduction to factor graphs,” *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 28–41, January 2004.
- [91] H. A. Loeliger, *Lecture Notes: Algebra, Codes, and Signal Processing*, ETH Zurich, Switzerland, 2005.
- [92] H. A. Loeliger, *Lecture Notes: Signal and Information Processing: Modeling, Filtering, Learning*, ETH Zurich, Switzerland, 2005.
- [93] H. A. Loeliger, *Lecture Notes: Stochastische Modelle und Signalverarbeitung*, ETH Zurich, Switzerland, 2005.
- [94] H.-A. Loeliger, J. Dauwels, V. M. Koch, and S. Korl, “Signal processing with factor graphs: examples,” in *First International Symposium on Control, Communications and Signal Processing (ISCCSP 2004)*, Hammamet, Tunisia, March 2004.
- [95] H.-A. Loeliger, J. Dauwels, J. Hu, S. Korl, L. Ping, and F. Kschischang, “The factor graph approach to model-based signal processing,” Draft version of September 5, 2006. [Online]. Available: <http://www.isi.ee.ethz.ch/~loeliger/>
- [96] G. H. Loudon, N. B. Jones, and A. S. Sehmi, “New signal processing techniques for the decomposition of EMG signals,” *Med Biol Eng Comput*, vol. 30, no. 6, pp. 591–599, Nov 1992. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=1297013>
- [97] C. Lüthi, “Seismosomnography: A factor graph approach,” Master’s thesis, ETH Zurich, Zurich, Switzerland, December 2006.
- [98] H.-P. Ludin, *Praktische Elektromyographie*. Enke Ferdinand, 1993.
- [99] B. Mambrito and C. J. De Luca, “A technique for the detection, decomposition and analysis of the EMG signal,” *Electroencephalogr Clin Neurophysiol*, vol. 58, no. 2, pp. 175–188, Aug 1984. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=6204844>
- [100] T. Masuda, H. Miyano, and T. Sadoyama, “A surface electrode array for detecting action potential trains of single motor units,” *Electroencephalogr Clin Neurophysiol*, vol. 60, no. 5, pp. 435–443, May 1985. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=2580695>

- [101] K. C. McGill, "Optimal resolution of superimposed action potentials," *IEEE Transactions on Biomedical Engineering*, vol. 49, no. 8, pp. 640–650, July 2002.
- [102] K. C. McGill, K. L. Cummins, and L. J. Dorfman, "Automatic decomposition of the clinical electromyogram," *IEEE Trans Biomed Eng*, vol. 32, no. 7, pp. 470–477, Jul 1985. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=3839488>
- [103] K. C. McGill and L. J. Dorfman, "High-resolution alignment of sampled waveforms," *IEEE Trans Biomed Eng*, vol. 31, no. 6, pp. 462–468, Jun 1984. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=6735418>
- [104] K. C. McGill, Z. C. Lateva, and H. R. Marateb, "EMGLAB: an interactive EMG decomposition program," *J Neurosci Methods*, vol. 149, no. 2, pp. 121–133, Dec 2005. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=16026846>
- [105] K. McGill, Z. Lateva, and M. Johanson, "Validation of a computer-aided EMG decomposition method," in *Engineering in Medicine and Biology Society, 2004. EMBC 2004. Conference Proceedings. 26th Annual International Conference of the*, vol. 2, 2004, pp. 4744–4747.
- [106] C. R. Merk, "Seismosomnography: A factor graph approach," Semester thesis, ETH Zurich, Zurich, Switzerland, July 2004.
- [107] P. P. Merkli, "Message-passing algorithms and analog electronic circuits," Doctoral Thesis, ETH Zurich, 2005.
- [108] R. Merletti, D. Farina, and M. Gazzoni, "The linear electrode array: a useful tool with many applications," *J Electromyogr Kinesiol*, vol. 13, no. 1, pp. 37–47, Feb 2003. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=12488085>
- [109] R. Merletti and H. J. Hermens, "Detection and conditioning of the surface EMG signal," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 107–131.
- [110] R. Merletti, A. Rainoldi, and D. Farina, "Myoelectric manifestations of muscle fatigue," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 233–258.

- [111] R. Merletti and P. Parker, Eds., *Electromyography: Physiology, Engineering and Non-Invasive Applications*. Wiley-IEEE Press, July 2004.
- [112] T. Moritani, D. Stegeman, and R. Merletti, “Basic physiology and biophysics of the EMG signal generation,” in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 1–25.
- [113] P. A. Parker, K. B. Englehart, and B. S. Hudgins, “Control of powered upper limb prostheses,” in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 453–475.
- [114] J. Peterson, “Estimation of filters for electromyographic (EMG) signal decomposition,” Master’s thesis, ETH Zurich, Zurich, Switzerland, May 2005.
- [115] F. B. Pfisterer and F. Bürgin, “QRS detection for automated arrhythmia analysis,” Semester thesis, ETH Zurich, Zurich, Switzerland, July 2002.
- [116] J. Proakis, *Digital Communications*. Singapore: McGraw-Hill, 1983.
- [117] A. Rainoldi, R. Casale, P. Hodges, and G. Jull, “Applications in rehabilitation medicine and related fields,” in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 403–433.
- [118] M. Renold, “Spike sorting of tetrode data and EMG signal decomposition using loopy belief propagation algorithms,” Semester thesis, ETH Zurich, Zurich, Switzerland, February 2006.
- [119] C. Rijke, “Simulation of EMG signals to test decomposition algorithms,” Semester thesis, ETH Zurich, Zurich, Switzerland, July 2004.
- [120] O. Ryynänen, J. Hyttinen, and J. Malmivuo, “Effect of skull resistivity and measurement noise on the spatial resolution of EEG,” *International Journal of Bioelectromagnetism*, vol. 7, no. 1, 2005. [Online]. Available: <http://www.ijbem.org/volume7/number1/toc.htm>

- [121] A. Schaufelberger, "EMG signal decomposition using factor graphs: Noise variance estimation," Semester thesis, ETH Zurich, Zurich, Switzerland, July 2006.
- [122] R. F. Schmidt and H.-G. Schaible, *Neuro- und Sinnesphysiologie*. Springer, Berlin, 2001.
- [123] R. F. Schmidt and G. Thews, *Physiologie des Menschen*, 27th ed. Springer, Berlin, 1997.
- [124] S. Schneiter, "Störgeräuschbefreiung für Sprachsignale mittels Faktorgraphen," Semester thesis, ETH Zurich, Zurich, Switzerland, July 2004.
- [125] H. A. Simon, *The Sciences of the Artificial—3rd Edition*. The MIT Press, 1996.
- [126] D. Stashuk, "EMG signal decomposition: how can it be accomplished and used?" *Journal of Electromyography & Kinesiology*, vol. 11, no. 3, pp. 151–173, June 2001.
- [127] D. W. Stashuk, "Decomposition and quantitative analysis of clinical electromyographic signals," *Med Eng Phys*, vol. 21, no. 6-7, pp. 389–404, Jul-Sep 1999. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=10624736>
- [128] D. W. Stashuk, "Detecting single fiber contributions to motor unit action potentials," *Muscle & Nerve*, vol. 22, no. 2, pp. 218–229, 1999.
- [129] D. W. Stashuk, D. Farina, and K. Sogaard, "Decomposition of intramuscular EMG signals," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 47–80.
- [130] D. F. Stegeman, R. Merletti, and H. J. Hermens, "EMG modeling and simulation," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 205–231.
- [131] R. Studer, "Computergestützte Analyse der Signale motorischer Einheiten im Elektromyogramm," Doctoral Thesis, ETH Zurich, 1984.

- [132] J. Treichler, "Simulation of analog decoders," Semester thesis, ETH Zurich, Zurich, Switzerland, July 2002.
- [133] J. V. Trontelj, J. Jabre, and M. Mihelin, "Needle and wire detection techniques," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 27–46.
- [134] P. O. Vontobel, "Algebraic coding for iterative decoding," Doctoral Thesis, ETH Zurich, July 2003.
- [135] P. T. Wellig, "Zerlegung von Langzeit-Elektromyogrammen zur Prävention von arbeitsbedingten Muskelschäden," Doctoral Thesis, ETH Zurich, November 2000.
- [136] M. Wenk, "Segmentation of EMG signals," Semester thesis, ETH Zurich, Zurich, Switzerland, February 2005.
- [137] N. Wiberg, H.-A. Loeliger, and R. Kotter, "Codes and iterative decoding on general graphs," in *Information Theory, 1995. Proceedings., 1995 IEEE International Symposium on*, Whistler, BC, 1995.
- [138] P. Wildbolz, "Seismosomnography: Heartbeat detection from pressure sensors," Semester thesis, ETH Zurich, Zurich, Switzerland, February 2005.
- [139] P. Woolf, C. Burge, A. Keating, and M. Yaffe, *Statistics and Probability Primer for Computational Biologists, BE 490/Bio7.91*, Massachusetts Institute of Technology, MA, U.S.A., Spring 2004.
- [140] D. Zazula, S. Karlsson, and C. Doncarli, "Advanced signal processing techniques," in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 259–304.
- [141] D. Zennaro, T. Läubli, D. Krebs, A. Klipstein, and H. Krueger, "Continuous, intermittent and sporadic motor unit activity in the trapezius muscle during prolonged computer work," *J Electromyogr Kinesiol*, vol. 13, no. 2, pp. 113–124, Apr 2003. [Online]. Available: <http://www.hubmed.org/display.cgi?uids=12586517>
- [142] D. Zennaro, P. Wellig, V. M. Koch, G. S. Moschytz, and T. Läubli, "A software package for the decomposition of long-term multichannel EMG signals using wavelet coefficients," *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 1, pp. 58–69, January 2003.

- [143] D. Zennaro, *Analysis and Interpretation of Long-Term Intramuscular Electromyography Signals During Prolonged Computer Work*. Shaker Verlag GmbH, 2003.
- [144] C. Zimmermann, “Decomposition of an EMG signal using an a-priori model,” Semester thesis, ETH Zurich, Zurich, Switzerland, February 2005.
- [145] M. J. Zwarts, D. F. Stegeman, and J. G. van Dijk, “Surface EMG applications in neurology,” in *Electromyography: Physiology, Engineering and Non-Invasive Applications*, R. Merletti and P. Parker, Eds. Wiley-IEEE Press, July 2004, pp. 323–342.

Index

- double, 144
- ADEMG, 39
- EMG-Belief, **257**
- EMG-LODEC, 25, 39, **257**
- EMG-View, **258**
- EMGLAB, 24, 40
- Emglink, 6
- factorgraph, **258**
- acetylcholine, 14
- acoustical mixtures, 213
- AD conversion, 22
- AD converter, **253**
- adaptation of MUAP shapes, 200
- adapting MUAP shapes, 179
- additive white Gaussian noise, 35, 48
- ADEMG, 39
- algorithm 1, 84
- algorithm 2, 107
- algorithm 3, 115
- alpha motor neuron, 13, 47
- analog-to-digital conversion, 22
- anatomy, 12
- anatomy research, 26
- applications, 24, 26
- artifacts, 42
- autonomic nervous system, 13
- auxiliary factor, 60
- auxiliary variables, 60
- averaging, 201
- averaging with aging, 201
- AWGN, 35, 48, **253**
- axon, 13
- axon terminal, 14
- back pain, 25
- Bayes' Theorem, 230
- belief propagation algorithm, 65
- biofeedback, 25
- blind source separation, 213–215
- block diagram, 47, 48, 50
- block diagram model, 31
- block diagram model of EMG signals, 53
- block length, 199
- block MAP decoding, 73
- block processing, 179, 197
- block-wise MAP, 73
- brain signals, 211
- brain tissue, 211
- BSS, 213–215, **253**
 - applications, 213
 - definition, 214
 - using factor graphs, 215
- cell membrane, 14
- central nervous system, 12
- changing MUAP shapes, 35
- changing shapes, 41
- channel, 51
- Cichocki, A., 301
- Cinderella hypothesis, 25, 223
- class signal, 193

- class-conditional probability, 230
 classification problem, 62
 classifier, 63
 cloning of variables, 56, 60
 coefficient nodes (discrete messages), 96
 coefficient nodes (multivariate-Gaussian messages), 118
 coefficient nodes (scalar-Gaussian messages), 109
 computation times, 141
 concentric needle electrodes, 18
 conditional probability, 228
 configuration, 56
 configuration space, 56
 connection, 56
 contraction, 14
 contributions, 3
 convolution, 99
 correct decomposition, 127, **257**
 covariance matrix, 83
 CT, **253**
 cycles, 72
 cycles in factor graphs, 71
 data formats, 24
 Dauwels, J., 3
 De Luca, C. J., 38
 decomposition, 30
 problems, 40
 decomposition in general, 53
 decomposition methods, 35
 decruitment, 43, 179, 201
 depolarization, 14, 16
 determinant, 83
 difficulties in decomposition, 40
 Dirac delta, 81
 discrete messages, 79, 84
 doublet, **257**
 doublets, 41
 modeling, 197
 dynamic contractions, 42
 dynamic muscle contractions, 42
 edge, 55, 56
 EEG, 213, **253**
 Eggimann, F., v.
 electrode, 18, 51
 choosing, 21
 fine-wire, 18
 needle, 18
 surface, 19
 surface array, 20
 electrode grids, 24
 electroencephalography, 213
 electromyography, 11, 12, **257**
 applications, 24
 anatomy research, 26
 ergonomics, 25
 neurology, 24
 quantitative analysis, 26
 EM, **253**
 EMG, **253**
 EMG signal, 11, 17, **257**
 block diagram, 48
 firing rates, 14
 model, 47
 MUAPs, 48
 noise, 48
 source, 47
 origin, 15
 pre-processing, 22
 AD conversion, 22
 filtering, 23
 up-sampling, 23
 EMG signal decomposition, 30, **258**
 basic steps, 36
 block diagram, 31
 difficulties, 40
 methods, 35

- pictorial outline, 30
EMG signal generation, 33
EMG signals
 measuring, 18
EMG-Belief, 243, **253**
EMG-LODEC, 39, **253**
EMG-View, 239, **253**
EMGLAB, 24, 40
Emglink, 6
enteric, 12
equality constraint node, 59
ergonomics, 25
estimating EMG signal noise
 power, 178
estimating MUAP shapes, 179, 192
estimating noise power, 190
ETH Zurich, v, **253**
event, 227, 228
exception throwing, 144
exercise physiology, 26
extracellular voltage, 211
extraction nodes (discrete mes-
 sages), 93
- factor graph**, 54, **258**
 arrows, 56
 connection, 56
 edge, 55
 EMG signal decomposition, 61
 example
 a simple factor graph, 55
 equality constraint node, 59
 multivariate probability dis-
 tribution, 57
 sum constraint node, 58
 introduction, 54
 model for EMG signals, 60
 node, 55
 notation, 54
 rules, 55, 57
 factor graphs with cycles, 53, 71
- factorgraph Java package, 249,
253, **258**
Farina, D., 35
fatigue, 25
FFG, 54, 55, **253**
FG, **254**
filtering, 23
fine-wire electrodes, 18
finite impulse response filters, 31
FIR, **254**
FIR filter, 31, 48
 order, 48
FIR filter order, 48
firing, 47, 62
Forney-style factor graph, 54, 55
FSM, **254**
- gait analysis, 26
gamma distribution, 34
Gaussian distribution, 34, 80, 231
gaussian distribution, 231
 scalar case, 231
 vector case, 233
generation of EMG signals, 33
global function, 55, 61, 65
glossary, 257
GUI, **254**
Gut, R., 1, 39
- Haas, W. F., 39
half-edge, 55, 56
heart beat detection
 using factor graphs, 207
Hidden Markov Model, 181
HMM, 181, **254**
human factors, 25
human nervous system, 12
- i.i.d., **254**
iff, **254**
IIR, **254**

- ill-posed problem, 42
 improved readout, 123
 independence, 229
 independent component analysis, 215
 independent events, 229
 Inf , 144
 inter-pulse interval, 34
 interspike interval, 34
 inverse problem, 42
 ion pumps, 16
IPI, 34, **254**
IPUS framework, 40
ISI, v, 34, **254**
 joint probability, 228
 known-constituent problem, 38, **258**
 results, 126
 Koch, V. M., biography, 299
 Korl, S., 3
 Kronecker delta, 59
 Kschischang, F., 3
 Lapidoth, A., v
 least-squares fit, 48
 list of abbreviations, 253
 list of symbols, 261
 local function, 55, 65
 local functions, 61
 Loeliger, H.-A., v, vi, 301
 loops in factor graphs, 71
MAP, 39, 73, **254**
 MAP decoding
 block MAP decoding, 73
 symbol MAP decoding, 73
 MAP estimate, 46
 MAP estimation, 62, 63
 MAP problem, 62
 marginal probability, 228
 marginalization, 64, 228
 Markov chain, 181
 Massey, J. L., v
 max-product algorithm, 74
 max-product rule, 74
 maximum a-posteriori probability estimation, 62, 63
 maximum voluntary contraction, 14
 McGill, K. C., vi, 39, 40, 301
 mean squared error, 195
 measuring of EMG signals, 18
 message types, 78
 message update rules, 237
 message update schedule (discrete messages), 102
 message-passing algorithm, 65
 running, 76
 message-passing algorithm development steps, 75
MFAP, 16, **254**
MIMO, 214
MIMO system, **254**
 minimum mean squared error, 42
ML, 39, **254**
MMSE, 42, **254**
 monopolar needle electrodes, 18
 Moschytz, G. S., v
 motivation, 1
 motoneuron, 13, 47
 motor end plate, 14, 16
 motor neuron, 47
 motor unit, 13, **258**
 firing, 47
 motor unit action potential, 16
MRI, **254**
MSE, 195, **254**
MUAP, 16, 17, **254**, **258**
 MUAP shape, 17
 MUAP shape changes, 200
 MUAP shape estimation, 192

- factor graph approach, 192
intuitive approach, 193
problem statement, 192
- MUAP shapes, 48
MUAP train, 17
multi-channel EMG signals, 50
multi-channel needle electrodes, 18
multichannel electrode grids, 24
multiple channels, 104
multivariate normal distribution, 83
multivariate probability distribution, 57
multivariate-Gaussian messages, 83, 115
- muscle, 12
contraction
dynamic, 42
disease, 15
- muscle architecture, 26
muscle contraction, 14
muscle fatigue, 25
muscle fiber action potential, 16
muscle fiber contraction, 13
muscle tension, 25
muscle/tendon junction, 26
mutually exclusive events, 229
MVC, 14, 254
- NaN, 144
Nawab, S. H., 40
neck pain, 25
needle electrodes, 18
nerve, 12
nervous system, 12
neural signal, 211
neural spike sorting, 211
neurology, 24
neuromuscular junction, 13
neuron, 211
neuroscience research, 211
- neurotransmitter, 14
noise, 35, 42
noise in synthetic signals, 35
noise model, 48
noise nodes (discrete messages), 100
noise nodes (multivariate-Gaussian messages), 122
noise nodes (scalar-Gaussian messages), 114
noise power, 190
noise power estimation, 190
normal distribution, 80, 231
- objective, 1
order of FIR filter, 48
origin of EMG signals, 15
outcome of an experiment, 57, 227
outline of this thesis, 8
overlap, 200
- pain, 25
parasympathetic, 12
pdf, **254**
peel-off approach, 39
peripheral nervous system, 12
physiology, 12
physiology research, 26
Poggio, T., 300
Poisson distribution, 34
Poisson process, 34
posterior probability, 63, 230
postsynaptic potentials, 213
pre-processing of EMG signals, 22
previous work, 38
principal components analysis, 215
prior probabilities, 63
prior probability, 230
probabilities, 228
probability, 227
probability of an event, 227

- probability propagation algorithm, 65
probability theory, 227
problem statement, 53
prosthesis control, 26
quantitative analysis of EMG signals, 26
RAM, 254
random number generator, 47
readout, 123
 improved, 125
reconstructed EMG signal, 127
reconstructed signal, **258**
recruitment, 14, 43, 179, 201, **259**
reference signal, 193
rehabilitation medicine, 26
residual signal, 35
resolution of superpositions, 37
resolving superpositions, 37
 MAP problem, 62
 sum-product algorithm, 72
resting membrane potential, 16
results
 known-constituent problem, 126
 unknown-constituent problem, 163
RMSE, 127, 255
root mean squared error, 127
RS-232, 255
sample space, 57, 227, 228
scalar-Gaussian messages, 80, 107
segmentation
 emg signals, 189
 factor graph, 182
 messages, 185
 node functions, 183
 variables, 183
segmentation of EMG signals, 180
segmenting EMG signals, 178
seismosomnography, 204
SEMG, 255
sensor, 51
shoulder pain, 25
Signal and Information Processing Laboratory, v
skeletal muscles, 12
SNR, 255
software overview, 4
somatic nervous system, 13
source models, 179, 195
source nodes (discrete messages), 87
SPA, 255
spike, 211, **259**
spike sorting, 211
spike train, **259**
spike train analysis, 211
spinal cord, 13
Stashuk, D. W., 39, 300
stochastic model of single cells, 34
stochastic point process, 47
striated muscles, 12
student projects, 5
sum constraint node, 58
 message update rules, 114, 122
sum constraint nodes (discrete messages), 98
sum constraint nodes (multivariate-Gaussian messages), 121
sum constraint nodes (scalar-Gaussian messages), 112
sum-product algorithm, 65, 71, **259**
sum-product algorithm for resolving superpositions, 72
sum-product rule, 65, 68, 70, **259**

- continuous, 78
- summing out, 64
- superposition, 41
 - resolving
 - modeling approach, 37
 - peel-off approach, 37
- superpositions
 - resolving, 37
- surface electrode arrays, 20
- surface electrodes, 19
- symbol MAP decoder, 73
- symbol-wise MAP, 73
- symbol-wise MAP estimate, 123
- sympathetic, 12
- synapse, 13
- synthetic EMG signals, 33

- target group of this thesis, 7
- tetrode, 211
- tetrode signal, 211
- total probability theorem, 229
- tracking MUAP shape changes, 200
- trellis, 39
- Tsujino, K., 300

- unknown-constituent problem, 38,
 160, **259**
 - results, 163
- up-sampling, 23

- valid configuration, 56
- variable cloning, 60
- Viterbi algorithm, 39, 73
- Vontobel, P., 3

- w.r.t., **255**
- Wandell, B. A., 300
- Weibull distribution, 34
- weight matrix, 83
- Wellig, P. T., 2, 39
- Willersinn, D., 300

Seite Leer /
Blank leaf

About the Author

Dare to be naive.

Richard Buckminster Fuller

Volker Koch was born on October 3, 1971 in Paderborn, Germany. Already at age 4 he liked playing with soldering irons, burning himself at least once. Despite this painful experience, he has never lost his early passion for technology. Only a few years later, while still in primary school, he installed 220 V cables and appliances using large electric saws and drilling machines, much to the concern of his mother, who was worried he would electrocute or hurt himself. Also, taking apart old TVs and radios was both interesting and beneficial—there were resistors, capacitors, coils, loudspeakers, and other devices to be found that could easily be reused. Over time, his depot became larger than that of many professionals.

Whenever he wasn't reading *Perry Rhodan*, he developed and manufactured his own (sometimes printed) circuit boards. He built alarm systems, remote controls, and RS232 interfaces to connect his *Commodore 64* and those of his customers to 300 baud acoustic couplers. In those days, it was not unusual to observe the presence of a very long cable, reaching from his room on the second floor along the stairway to the telephone on the first floor. He also programmed, repaired, and damaged computers. Like *Abenteuer Forschung* with Joachim Bublath, the *WDR-Computerclub* with Wolfgang Back und Wolfgang Rudolph was a must-see event. And whenever an electrician or another craftsman showed up, he closely observed and helped (while other kids enjoyed open swimming pools)—an apprenticeship in electrical engineering was clearly a good choice.

Education

After he had successfully completed such an industrial apprenticeship, high school, and mandatory national service in the technical service department of a hospital, he started his university studies in electrical engineering and information technology at the Universität Paderborn. Having passed his intermediate examination, he became increasingly interested in biomedical engineering. But before moving to Karlsruhe to specialize in this field, he broadened his horizons by studying at the University of Waterloo in Canada for one year, where he also got in touch with EMG signal decomposition for the first time [74, 75].

Despite he had a great time in Ontario, he came back to Germany in 1997 to finish his studies at yet another university, the Universität Karlsruhe. In 2001 he graduated from this university with a diploma degree in electrical engineering and information technology. His area of specialization was biomedical engineering with a focus on medical imaging.

From 2001 until 2006 he was a doctoral student at ETH Zurich in Switzerland, where he was also a member of the Doctoral School in Computer, Control, and Communications. In addition, from 2002 to 2004 he studied part-time for the Master of Advanced Studies ETH in Management, Technology, and Economics/BWI degree, which he obtained in the fall of 2004. Since 2003 he has also been a part-time student for the ETH teaching certificate in electrical engineering.

Research

During his studies he worked at the Department of Systems Design Engineering at the University of Waterloo (EMG signal decomposition with Professor Stashuk), at the Fraunhofer Institute for Information and Data Processing in Germany (automated image interpretation with Dr. Willersinn), at Mitsubishi Electric Corporation in Japan (knowledge acquisition system with Dr. Tsujino), and at the Massachusetts Institute of Technology (face detection with Professor Poggio). He did his diploma thesis at Stanford University (medical imaging, vision, and fMRI with Professor Wandell) [31, 65, 73].

From 2001 to 2006 he was a research and teaching assistant at the Signal

and Information Processing Laboratory of ETH Zurich in Switzerland (biomedical signal processing with Professor Loeliger). In 2005 he was a research intern at the prestigious Brain Science Institute of RIKEN in Japan (biomedical signal processing with Professor Cichocki). Then, in 2006, he was a visiting biomedical engineer at the VA Palo Alto Rehabilitation R&D Center in California (EMG signal decomposition with Professor McGill), where he was also associated with the Department of Bioengineering of Stanford University.

Teaching

Besides doing research at ETH Zurich, he also enjoyed supervising more than 20 student projects and being a teaching assistant of 5 different courses: assembler programming of DSPs (6 semesters), algebra, codes, and signal processing (3 semesters), stochastic models and signal processing (1 semester), adaptive filters and neural networks (1 semester), and signal and information processing (1 semester). In 2005 and in 2006 he was a college lecturer at HTA Lucerne in Switzerland, where he developed and taught a new course on digital image processing.

Other Activities, Hobbies, and Interests

From 2003 to 2007 he was chair of the IEEE Student Branch Zurich and president of an academic association at ETH Zurich. In addition, he represented the scientific staff in various committees, such as the teaching committee. In his remaining spare time he enjoys *doing* sports (aerobics, running, swimming, badminton), reading good books or watching videos from *MIT World*, reading news about cutting edge and future technologies, being actively involved in the community, socializing, and a game of chess with a good glass of wine. He neither enjoys American series or movies dubbed into German nor discussions about operating systems.

Series in Signal and Information Processing

edited by Hans-Andrea Loeliger

- Vol. 1: Hanspeter Schmid, **Single-Amplifier Biquadratic MOSFET-C Filters.** ISBN 3-89649-616-6
- Vol. 2: Felix Lustenberger, **On the Design of Analog VLSI Iterative Decoders.** ISBN 3-89649-622-0
- Vol. 3: Peter Theodor Wellig, **Zerlegung von Langzeit-Elektromyogrammen zur Prävention von arbeitsbedingten Muskelschäden.** ISBN 3-89649-623-9
- Vol. 4: Thomas P. von Hoff, **On the Convergence of Blind Source Separation and Deconvolution.** ISBN 3-89649-624-7
- Vol. 5: Markus Erne, **Signal Adaptive Audio Coding using Wavelets and Rate Optimization.** ISBN 3-89649-625-5
- Vol. 6: Marcel Joho, **A Systematic Approach to Adaptive Algorithms for Multichannel System Identification, Inverse Modeling, and Blind Identification.** ISBN 3-89649-632-8
- Vol. 7: Heinz Mathis, **Nonlinear Functions for Blind Separation and Equalization.** ISBN 3-89649-728-6
- Vol. 8: Daniel Lippuner, **Model-Based Step-Size Control for Adaptive Filters.** ISBN 3-89649-755-3
- Vol. 9: Ralf Kretzschmar, **A Survey of Neural Network Classifiers for Local Wind Prediction.** ISBN 3-89649-798-7
- Vol. 10: Dieter M. Arnold, **Computing Information Rates of Finite State Models with Application to Magnetic Recording.** ISBN 3-89649-852-5
- Vol. 11: Pascal O. Vontobel, **Algebraic Coding for Iterative Decoding.** ISBN 3-89649-865-7
- Vol. 12: Qun Gao, **Fingerprint Verification using Cellular Neural Networks.** ISBN 3-89649-894-0
- Vol. 13: Patrick P. Merkli, **Message-Passing Algorithms and Analog Electronic Circuits.** ISBN 3-89649-987-4

- Vol. 14: Markus Hofbauer, **Optimal Linear Separation and Deconvolution of Acoustical Convulsive Mixtures.** ISBN 3-89649-996-3
- Vol. 15: Sascha Korl, **A Factor Graph Approach to Signal Modelling, System Identification and Filtering.** ISBN 3-86628-032-7
- Vol. 16: Matthias Frey, **On Analog Decoders and Digitally Corrected Converters.** ISBN 3-86628-074-2
- Vol. 17: Justin Dauwels, **On Graphical Models for Communications and Machine Learning: Algorithms, Bounds, and Analog Implementation.** ISBN 3-86628-080-7