Information

45

Model-based Algorithm Development

Yu Yao

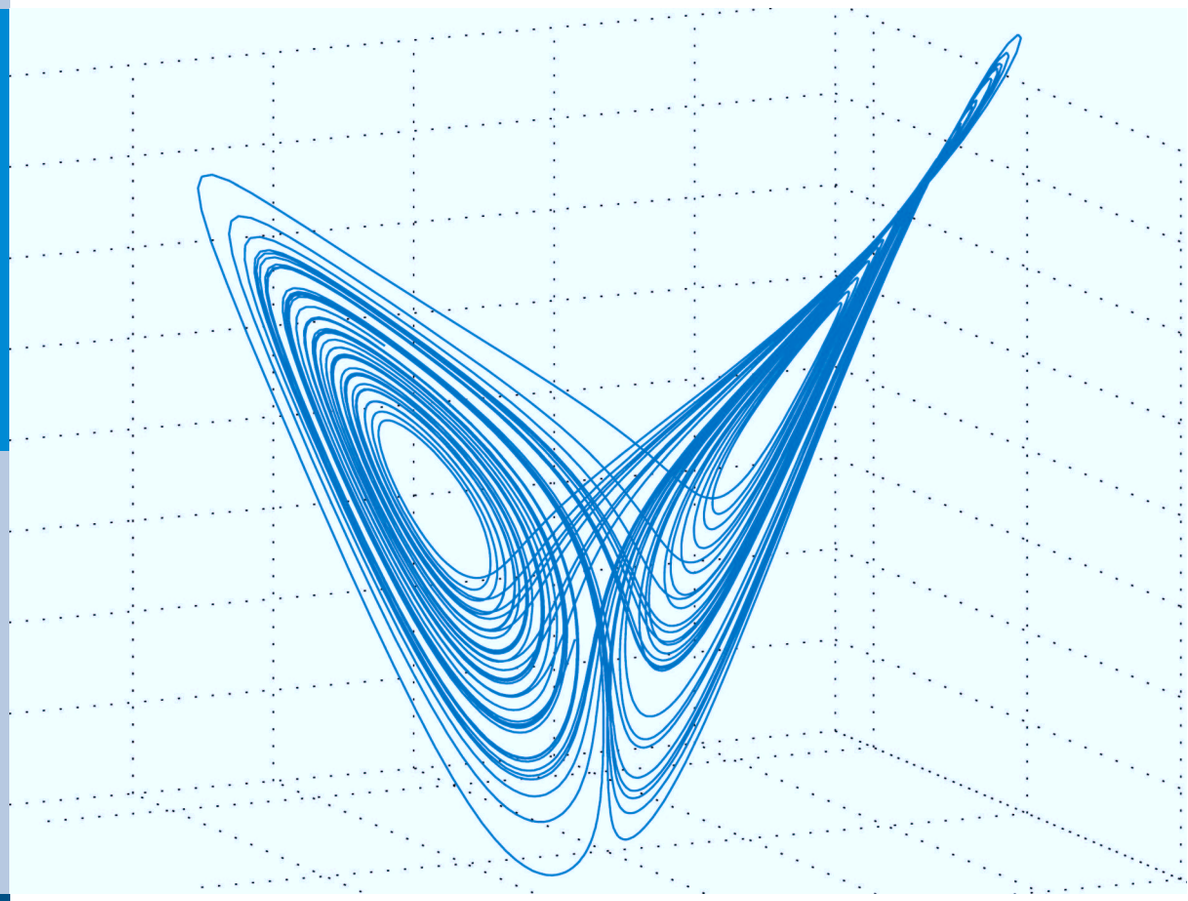
Member of the Helmholtz Association

Information
Band / Volume 45
ISBN 978-3-95806-080-7

JÜLICH
FORSCHUNGSZENTRUM

Model-based Algorithm Development with Focus on Biosignal Processing

Yu Yao



Information
Band / Volume 45
ISBN 978-3-95806-080-7

JÜLICH
FORSCHUNGSZENTRUM

Schriften des Forschungszentrums Jülich
Reihe Information / Information Band / Volume 45

Forschungszentrum Jülich GmbH
Zentralinstitut für Engineering, Elektronik und Analytik (ZEA)
Systeme der Elektronik (ZEA-2)

Model-based Algorithm Development with Focus on Biosignal Processing

Yu Yao

Schriften des Forschungszentrums Jülich
Reihe Information / Information Band / Volume 45

ISSN 1866-1777 ISBN 978-3-95806-080-7

Bibliographic information published by the Deutsche Nationalbibliothek.
The Deutsche Nationalbibliothek lists this publication in the Deutsche
Nationalbibliografie; detailed bibliographic data are available in the
Internet at http://dnb.d-nb.de.

Publisher and Distributor: Forschungszentrum Jülich GmbH
Zentralbibliothek
52425 Jülich
Tel: +49 2461 61-5368
Fax: +49 2461 61-6103
Email: zb-publikation@fz-juelich.de
www.fz-juelich.de/zb

Cover Design: Grafische Medien, Forschungszentrum Jülich GmbH

Printer: Grafische Medien, Forschungszentrum Jülich GmbH

Schriften des Forschungszentrums Jülich
Reihe Information / Information, Band / Volume 45

D 468 (Diss., Wuppertal, Univ., 2015)

ISSN 1866-1777
ISBN 978-3-95806-080-7

The complete volume is freely available on the Internet on the Jülicher Open Access Server (JuSER)
at www.fz-juelich.de/zb/openaccess.

Abstract

In recent years, the development of cheap and robust sensors combined with the ever increasing availability of the internet led to a revolution in information technology, giving rise to an amount of data, which was unimaginable just a decade ago. This explosion in data lead to an increased demand for algorithms for processing this data. However, an often overlooked aspect is that with ever sophisticated algorithms there is associated a demand for equally sophisticated mathematical modelling. In this thesis, we explore the interaction between algorithm design and modelling.

Although, the models and methods discussed here are not limited to any single domain of application, we will base our discussion on example applications from the domain of biomedical engineering. This is because the analysis of physiological time series is characterised by two problems which help to highlight the importance of modelling. First, the high noise level of biological signals requires strong regularization, which can be provided via a model. Second, in many medical applications the value of interest is not directly observable. Thus, these latent variables have to be estimated, e.g. with the help of a model.

In the course of our discussion, we will encounter two major modalities. The first one is Ballistocardiography (BCG), a modality often used in home monitoring applications, which is based on simple pressure sensors, yielding a scalar signal. The second modality is functional magnetic resonance imaging (fMRI), a complex and highly sophisticated method, capable of generating images of brain functionality.

In the first half of this thesis, we will focus on signal separation and denoising methods for BCG. The performance of these methods is then verified with model generated data, which provides a very common example of how modelling interacts with algorithm design. However, the relationship between algorithms and modelling goes much deeper, since new insights

gained through better signal processing methods can also inspire new models. This can be seen from the improved probabilistic BCG model, which emerged from the results of the BCG signal separation and denoising method. Finally, the new model opens the possibility for probabilistic higher level analysis of the BCG signal, which exemplifies how improvements in modelling leads to improved algorithms.

In the latter half of the thesis, we will focus on embedded clustering for fMRI data, which allows us to perform model inversion and clustering at the same time. Here we see that although there are great differences between the two modalities BCG and fMRI, the model based approach reveals how methods developed for BCG can be applied to fMRI. This again demonstrates the importance of a model based view on algorithm design.

Contents

Acknowledgement

This work would not have been possible without the dedicated effort and generous support of many people.

First of all, I would like to thank my supervisor Michael Schiek for his continued support and excellent supervision throughout all the years at ZEA-2.

Furthermore, I would like to thank my doctoral adviser Uwe Pietrzyk for providing the opportunity to conduct the research which made this thesis possible, as well as the numerous discussion sessions, which provided invaluable input for my research.

I would also like to thank Andreas Klümper and Thomas Lippert for the interesting discussion and their helpful advice, as well as Johannes Lindemeyer from INM-4 for generously providing the MRI scans in chapter 4.

In addition, I would like to thank Stefan van Waasen and all my colleagues at ZEA-2 for their support and encouragement.

Many thanks to Christoph Brüser and his colleagues from MedIt at RWTH Aachen University for their support and the fruitful collaboration.

Special thanks to Sudhir Shankar Raman for supervising my project at TNU in Zurich, which was one of the most interesting and exciting projects during my PhD. Also, many thanks to Klaas Enno Stephan and the other colleagues at TNU for the interesting discussion, and for introducing me to many exciting ideas and concepts, which had a lasting impact on my research.

Chapter 1

Introduction

Modelling and signal processing are two closely related topics. Every signal processing technique is based, to a certain degree, on a model, either explicitly or implicitly. Thus, increasingly accurate models can help improving or verifying signal processing techniques, open up new perspectives on existing algorithms or inspire new algorithms.

In this context, the validation of algorithms with model-generated data is of special importance, since in some cases only artificial data can provide the ground truth information, which is necessary to quantify the error of the signal processing technique under assessment [1, 2].

Since many physiological time series are very noisy, models are needed to provide constraints and regularization for signal processing techniques. Furthermore, in some cases, the information of interest cannot be measured directly and model-based methods are necessary to estimate these hidden or latent variables. This is especially true for medical imaging applications like magnetic resonance imaging, computed tomography or positron emission tomography [3]. In addition, the emerging field of data fusion, i.e. the combination of data from different modalities, represents another scenario where modelling is an essential part of signal processing [4, 5, 6].

On the other hand, many biological systems are not fully understood and signal processing techniques can be used to analyse the raw data, providing new insights into the structure and working principle of these systems. This process can lead to better models being developed, but it can also benefit from the emergence of novel modelling concepts. A prime example is the concept of synchronization in nonlinear dynamics, which is based on a mathematical treatment of coupled oscillators [7], but went on to revolutionize numerous fields in biology and biomedicine [8].

One of the most well-known biomedical applications of this concept is cardio-respiratory synchronization [9, 10, 11], which is a measure combining information from two modalities: electrocardiogram (ECG) and respiratory effort. It has been shown to be a potential indicator for cardio-vascular health and a very reliable predictor for sleep apnoea in infants [9, 10].

At the same time the concept of phase synchronization has also led to the development of a highly effective processing technique for magnetoencephalography data, used for cardiac and ocular artefact removal and identification of relevant signal components [12, 13]. Thus, advances in modelling can lead to improvements in very different fields of application. This is also one of the central themes in this thesis.

In addition, one of the key observations from nonlinear dynamics is that complex behaviour can emerge from systems described by a few low-dimensional, but nonlinear laws [8]. This raises the question, if it is possible to find simple low-dimensional models describing the complex behaviour of physiological systems, by extending our model space to include nonlinear models. And some results in this area seem to indicate that this is indeed a promising approach [9, 10].

In this context, it should be mentioned that model-based signal processing is essentially a cross-disciplinary topic. This is especially obvious for the field of nonlinear dynamics [8, 14], where many models developed in physics were adapted by the biomedical community [9, 10, 12, 13]. However, it also applies to numerous other fields, like e.g. the emerging discipline of computational psychiatry, which combines methods from high performance computing [15], statistical inference [16, 17] and neurobiology to address open questions in psychiatry [18].

Interestingly, many concepts used in statistical inference, like the information theoretic entropy or the variational free energy, have their origin in statistical and quantum physics [19, 20, 21], where computational aspects also play an important role [22]. Therefore, statistical inference itself can be viewed as a cross-disciplinary field.

Another important theme discussed in this thesis is that signal processing methods may help us to refine signals and enhance the understanding of the underlying principles and mechanisms of the system under investigation, which allows us to build new models or improve and refine existing models.

We will investigate this iterative aspect of modelling and signal processing based on several examples from biomedical signal processing, where the interaction between modelling and algorithm design is of special relevance. The idea behind this approach is that the cycle of model refinement and

algorithm development should always be driven by experimental data, in order to allow the results to be validated. Without the guidance through measured data, it is difficult to relate the models and algorithms to meaningful applications. For this reason, we will base the discussion in this thesis on two example modalities from biomedical applications.

The first modality is ballistocardiography (BCG), which is a method for recording the mechanical heartbeat activity via measurement of tiny body movements caused by the heart and blood flow. We have chosen this modality to demonstrate techniques for source separation, i.e. the extraction of several signal components from a noisy scalar superposition. Thus, we will deal with the problem of extracting high-dimensional dynamics from a low-dimensional source.

The second modality is the blood-oxygenation-level dependent (BOLD) contrast used in functional magnetic resonance imaging (fMRI) studies on brain activity patterns. Data from this modality is investigated in the context of embedded clustering (EC), which combines a clustering model with the estimation of effective brain coupling via a model known as dynamic causal modelling (DCM). In contrast to the previous modality, embedded clustering deals with the extraction of a low-dimensional representation from high-dimensional data.

These two modalities reflect two distinct directions in biomedical engineering. BCG is currently researched as a simple, robust and cheap modality for unobtrusive data acquisition in potential home monitoring applications. The simplicity of the sensor and the uncertainty about the environmental conditions pose significant challenges since it leads to very noisy data, contamination by artefacts and missing segments in recordings. When dealing with data of this kind, there is a need for models providing prior information to guide reconstruction and regularize algorithms.

In contrast, the fMRI BOLD signal is an example for a highly sophisticated imaging method, which is used in clinical research and practice. This highly complex method indirectly measures the hidden brain activity, by recording a signal related to differences in blood-oxygenation-level. As a consequence, the algorithms, used to derive the hidden variables of interest from the measured data, have to be based on a model of the underlying process that generated the measurement.

The above description of the two modalities already contain good examples for the interaction between aspects of modelling and signal processing. The processing of data in the presence of noise and uncertainty about the system, or the estimation of hidden variables are both cases which require a model of the underlying process of data generation. Another example is the

verification of an algorithm with data generated using a model.

Although, the discussion in this thesis often focuses on the two modalities BCG and fMRI, the overall aim of this work is to discuss the afore-mentioned interaction between modelling and signal processing based on examples from biosignal processing. In the course of this discussion, we will be exposed to a number of concepts which show that the methods used to deal with the two example modalities BCG and fMRI have many aspects in common, although they represent entirely different directions in biomedical research. These concepts include nonlinear dynamical systems for modelling, as well as probabilistic inference used for model inversion. Throughout the thesis, we will also encounter the topic of mixture modelling, which serves as another link between the different chapters.

As mentioned, probability theory will play an important role throughout the whole thesis. Biosignals are characterized by a high degree of uncertainty about the generation process and the recording of biosignals is especially prone to noise contamination. This necessitates the use of probabilistic inference methods, which are ubiquitous throughout this thesis. Therefore, we dedicate chapter 2 to an introduction of concepts from probability theory and techniques for probabilistic inference including methods for approximate inference such as Monte Carlo and variational Bayes, which originated in the field of statistical physics. The remainder of the chapters is organized as follows.

In chapter 3, we will focus on nonlinear source separation applied to BCG as an example modality. We will demonstrate how model generated data can be used to verify the accuracy of the source separation algorithms. Furthermore, we will also see how the results of the source separation enables us to build a better model of the BCG. The improved model allows us to perform more sophisticated processing tasks, e.g. heartbeat detection in the BCG.

Chapter 4 deals with the problem of embedded clustering, which provides a unified framework for model inversion and clustering. This topic is discussed in the context of psychological fMRI studies with groups of subjects. And although the application differs strongly from that of the previous chapter, we will see that on the level of modelling many parallels to the methods from chapter 3 can be observed.

We conclude this thesis in chapter 5, with a summary presenting a unifying viewpoint on the different modelling and signal processing aspects exemplified in chapters 3 and 4, as well as an outlook on new and old challenges which still remain.

While these chapter provide a detailed discussion of the relevant problems, some mathematical and technical details are presented for the interested reader in the appendix. These topics include mathematical basics, such as a short introduction to Lagrange multiplier or a list of important probability distributions, as well as technical and mathematical details of the Monte Carlo method used in chapter 3 and solutions to the integrals in chapter 4.

Chapter 2

Probability Theory and Inference

An essential aspect in model building is probability theory, which plays an important role for modelling uncertainty and chance. However, the mathematical foundations of probability theory and statistics are too broad to be discussed here. In this chapter, we will provide an introduction to probability theory with focus on aspects of applied probability theory and methods for approximate inference, which will be needed in later chapters.

We start with an introduction to basic concepts of probability theory and continue with a discussion on inverse problems in section 2.2. Other topics include Monte Carlo and variational Bayes, two classes of methods for approximating solutions for inverse problems. In addition, we will introduce a graphical representation for probabilistic models called Bayesian networks, which will be used on many occasions throughout this thesis.

2.1 Basics of Probability Theory

In this section, we will introduce basic concepts and definitions of probability theory. In order to keep the discussion simple and easy to understand, we follow the example of other authors [23, 24] by introducing the concepts using a simple random experiment.

The random experiment, we have chosen as a running example for this section consists of rolling two fair six-sided dice. Each of the dice rolls has six equally probable outcomes and, assuming the two dices are distinguishable, the whole experiment has 36 different outcomes. This leads to our first

definition, namely the definition of a random variable.

Formally, a random variable is defined as a function, which assigns a number to each outcome of a random experiment. A simple example of a random variable in our dice experiment would be the sum of eyes on both dice. Sometimes it will be important to distinguish between the random variable itself and the value of the variable. In these cases, we will use capital letters for the name of a random variable and lower case letters for its value. Thus, we will call the sum of eyes random variable X, while x denotes the value of X.

The set of all values the random variable can take, which in our example includes the integers from 2 to 12:

$$x \in \{2, 3, \ldots, 12\},$$

is called the sample space Ω. The sample space is required to be a non-empty set [23] and subsets A of the sample space are called events. For example, the subset

$$A = \{2, 4, 6, \ldots, 12\}$$

corresponds to the event "the eye count is even".

In addition, we define a sigma algebra $\mathcal{A}$ as a set of events A possessing the following properties:

$$\Omega \in \mathcal{A} \tag{2.1}$$

$$A \in \mathcal{A} \Rightarrow \bar{A} \in \mathcal{A} \tag{2.2}$$

$$A_i \in \mathcal{A} \Rightarrow \bigcup_i A_i \in \mathcal{A}. \tag{2.3}$$

Equation (2.1) means that the sigma algebra $\mathcal{A}$ must always contain the entire sample space as an element. Equation (2.2) means that if an event A is in the sigma algebra $\mathcal{A}$, the complementary event $\bar{A} = \Omega / A$, containing all outcomes that are not in A, must also be in the sigma algebra $\mathcal{A}$. From these two axioms, it immediately follows that $\mathcal{A}$ must also contain the empty set $\emptyset$. Lastly, equation (2.3) means that if a set of events A_i is in $\mathcal{A}$, their union must also be in $\mathcal{A}$.

The probability is formally defined as a function p which assigns each event in the sigma algebra a number and which satisfying three conditions [23, 24] known as the Kolmogorov axioms:

$$p(A) \in \mathbb{R} \text{ and } p(A) \geq 0 \quad \forall A \in \mathcal{A} \tag{2.4}$$

$$p(\Omega) = 1 \tag{2.5}$$

$$p\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} p(A_i), \tag{2.6}$$

with the events A_i being mutually exclusive.

In our example, $p(X{=}12)$ is the probability that the sum of eyes is 12 and $p(X{=}x)$ the probability that the sum of eyes is x. If the meaning is clear from the context, we can also simplify the notation by writing $p(x)$ instead of $p(X{=}x)$.

The introduction given above applies to random variables with discrete sample space. Random variables with continuous sample space are often treated as a separate case, with modifications to the definitions given above, including the introduction of probability via the cumulative distribution function. In addition, a unified theory of discrete and continuous random variables requires a measure theoretic approach, which is outside the scope of this work. For an in depth treatment of the foundations of probability theory, we refer the reader to standard textbooks like [23, 24].

Instead, we will continue with the concept of joint and conditional probabilities, which will become important in later chapters. For this purpose, we assume that the dice are labelled A and B and we define a second random variable Y as the number of eyes on dice A. Now, it is possible to form a joint sample space for (X, Y)-pairs: $(X, Y) \in \{2, 3, \ldots, 12\} \times \{1, 2, \ldots, 6\}$. The joint probability of the variables X and Y $p(X{=}x, Y{=}y)$ is then defined as the probability of X taking the value of x and Y taking the value of y at the same time.

However, the value of the joint probability function has to be consistent with the value of $p(X{=}x)$ defined above. In this context, $p(X{=}x)$ is called the marginal probability of X and can be interpreted as the probability for X taking the value x, while the value of the variable Y is unknown or not of interest. The joint probability function must be defined in a way such that the value of the marginal probability is obtained by "summing out" Y:

$$p(X) = \sum_{Y} p(X, Y), \tag{2.7}$$

where $\sum_Y$ means summing over all possible values of Y. For continuous variables, summation has to be replaced by integration.

The conditional probability $p(X{=}x|Y{=}y)$ is the probability of X taking on the value of x, when Y is known to have the value y. It is defined as:

$$p(X{=}x|Y{=}y) = \frac{p(X{=}x, Y{=}y)}{p(Y{=}y)}, \text{ if } p(Y{=}y) \neq 0. \tag{2.8}$$

In the dice example, $p(X{=}9|Y{=}5) = \frac{1}{6}$ is the probability that the sum of eyes is nine, when dice A, which has already been thrown, is showing five eyes. On the other hand, $p(X{=}9, Y{=}5) = \frac{1}{36}$ is the probability that the sum of eyes is nine and dice A shows five eyes, before any dice have been thrown and $p(Y{=}5) = \frac{1}{6}$ is the probability that dice A shows five eyes, while the sum of eyes does not matter, or is not observed.

When rewriting (2.8) as

$$p(x, y) = p(x)p(y|x) = p(y)p(x|y), \tag{2.9}$$

the equation is also known as chain rule. However, in the pattern recognition community equations (2.7) and (2.9) are also referred to as sum and product rules, respectively. They represent the two most fundamental rules in inference, from which most of the remaining rules can be derived [16, 17].

Inserting equations (2.7) and (2.9) into equation (2.8), we obtain Bayes law:

$$p(Y{=}y|X{=}x) = \frac{p(X{=}x|Y{=}y)p(Y{=}y)}{p(X{=}x)} = \frac{p(X{=}x|Y{=}y)p(Y{=}y)}{\sum_Y p(X{=}x|Y)p(Y)}, \tag{2.10}$$

which sometimes is called the law of inverse probability [17]. Again the summation has to be replaced by an integral for continuous random variables. As the name already implies, this equation will play an important role when solving inverse problems, which is the main topic in the next section, where we deal with inverse problems.

In the dice example, equation (2.10) can be applied to the scenario where somebody tells us that the sum of eyes is nine and asks for the probability that the number of eyes on dice A is five. The marginal probability for the sum of eyes being nine is $p(X{=}9) = \frac{1}{9}$, while the probability that dice A shows five eyes at the same time is $p(X{=}9|Y{=}5) = \frac{1}{6}$. The marginal probability that dice A shows five eyes is $p(Y{=}5) = \frac{1}{6}$, since we assumed fair dice. Thus, according to equation (2.10), the solution is $p(Y{=}5|X{=}9) = \frac{1}{4}$.

Note that calculating the denominator in equation (2.10) requires a summation over all possible values of Y, which in the dice example is limited to only six possibilities. However, in general, calculating the denominator in Bayes law is the most challenging part of many model inversion problems, and we dedicate section 2.4 to the discussion of approximation methods for solving this problem.

2.2 Inverse Probability

When dealing with probabilistic models, it is often convenient to distinguish between forward problems and inverse problems. A forward problem consists of the task of calculating the observation or probabilities for possible observations generated by a system, under the condition that one knows the properties of the system.

An example is ballistocardiography, which was mentioned in the introduction. If the timing of the heartbeats of a subject is known, one can attempt to build a model which calculates the measurable BCG based on the assumed heartbeat times. A model that solves the forward problem is sometimes called a generative model, because it is capable of generating simulated observations. We will encounter examples of generative models for BCG in section 3.5.

On the other hand, in inverse problems, one is given the observation and the task is to infer properties of the system or other unobservable variables in the problem. In the BCG example, the inverse problem is to estimate the unobservable heartbeat times from the measured BCG time series. Many problems of practical interest, including several of the problems in chapter 3 and 4, are inverse problems and one way of solving these kinds of problems is to apply Bayes law.

We have already seen a very simple example of how to apply Bayes law at the end of the previous section. In this section, we will introduce a slightly more complex example, in order to provide additional insight and familiarize the reader with the concepts involved in solving inverse problems. The example in this section is a variation on a very common toy problem [16, 17], which we call the two boxes of fruits.

2.2.1 The Boxes of Fruits Example

Two boxes with apples and bananas are prepared for a guessing game. One of the boxes has a dark colour and contains two apples and six bananas. The other box is painted in a light colour and contains four apples and three bananas. The position of the two boxes are randomized by flipping a fair coin and a cover is put above both boxes, which prevents the player from seeing the positions, but allows the player to reach into the boxes. The player has to pick an opening at random, take out a fruit and guess the colour of the box based on the type of the fruit taken out. If the guess is correct the player can keep the fruit as a reward. Figure 2.1 illustrates the setting of the problem.

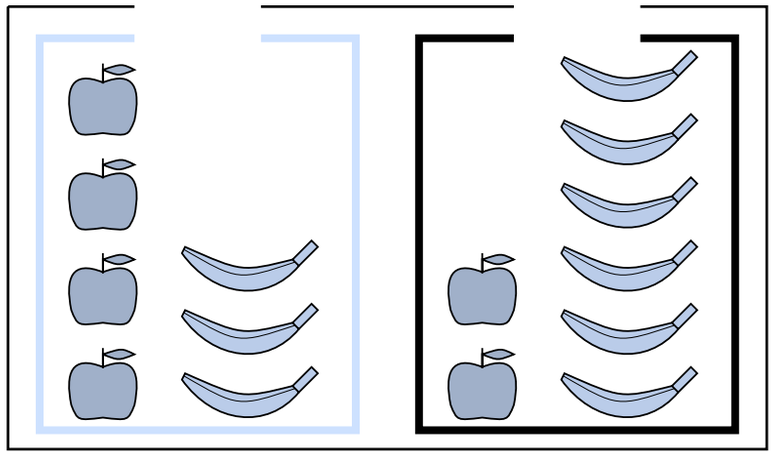


Figure 2.1: The setting for the boxes of fruits example. The light box (on the left side) contains four apples and three bananas and the dark box (on the right side) contains two apples and six bananas. Both boxes are covered such that their colour and position cannot be seen from the outside. The player has to reach into one of the boxes, take out a fruit and guess the colour of the box.

In this example, the forward problem consists in predicting the type of fruit being taken out, from the number of fruits in each box and the probability of picking the boxes. However, this is not the problem the player has to solve. The player has to solve the inverse problem, i.e. inferring the colour of the box from observing the type of the fruit. The correct answer to this question is very intuitive: Due to the high ratio of bananas in the dark box and the high ratio of apples in the light box, the answer should be dark when the fruit taken out is a banana and light if it is an apple. In the following, we will derive this solution mathematically to provide insight into the way of solving inverse problems.

To model this problem, we introduce two variables: B is the colour of the box which can be either dark (d) or light (l), and which is to be inferred. Such a variable that cannot be observed is called a latent variable. The second variable F is the type of fruit, which is either apple (a) or banana (b). This variable can be observed and is therefore called observation or data variable.

The generative model for this example consists of two steps: In the first step, we pick a value for B by flipping a fair coin. Mathematically, this corresponds to assuming a uniform probability of $p(B=d) = p(B=l) = 0.5$.

The probability function $p(B)$ is called the prior probability of the box colour B, since it describes the probability of B prior to observing the data F [17].

In the second step, we have to model the action of taking out a fruit from the chosen box. This can be described using the conditional probability function $p(F|B)$, where we plug in the value of B from step one and obtain the probability of F conditioned on B. This means that if $B = d$, we flip a biased coin which lands heads three out of four times and assign $F = b$ if the coin comes up head and $F = a$ for tail. Conversely, if $B = l$, we flip a coin which lands heads three out of seven throws. If we fix the value of F and interpret $p(F|B)$ as a function of B, then $L(B) = p(F|B)$ is called the likelihood function [17]. All in all, there are four different configurations for $p(F|B)$, which are summarized in table 2.1.

Above, we have specified a procedure to simulate the boxes of fruits experiment, which first draws a value for B from the prior probability and based on that value generates the observation F from $p(F|B)$. However, the full forward model requires a third and important step in which we specify the joint probability over all variables in the problem, observed or latent [16]. In the current example, this is achieved simply by applying the chain rule (eq (2.9)): $p(B, F) = p(B)p(F|B)$. For more complex problems, it can become necessary to apply the chain rule repeatedly.

Having specified the forward model, we can now turn to the inverse problem. In Bayesian statistics, solving the inverse problem corresponds to calculating the probability of some or all of the latent variables conditioned on the observed variable. In our example, this would be the probability of B conditioned on F: $p(B|F)$. In the context of inverse problems, $p(B|F)$ is called the posterior probability, since it is the probability of B after observing the value of F. It can be obtained by applying Bayes law (eq (2.10)):

$$p(B|F) = \frac{p(F|B)p(B)}{\sum_B p(F|B)p(B)}. \tag{2.11}$$

The sum in the denominator is equivalent to the marginal probability of the observation, which is often simply called the marginal

$$p(F) = \sum_B p(F|B)p(B). \tag{2.12}$$

When comparing different competing models this term plays an important role and is called the model evidence. This topic is discussed in more depth in section 2.5.1.

Here, we will carry out the calculation for a concrete example. Assuming

$p(F\|B)$		B	
		d	l
F	a	$\frac{1}{4}$	$\frac{4}{7}$
	b	$\frac{3}{4}$	$\frac{3}{7}$

$p(B\|F)$		B	
		d	l
F	a	$\frac{7}{23}$	$\frac{16}{23}$
	b	$\frac{7}{11}$	$\frac{4}{11}$

Table 2.1: Likelihoods (left) and posterior probabilities (right) for the boxes of fruits example.

the fruit taken from the box is an apple, the marginal for $F=a$ is:

$$\begin{aligned} p(F=\mathrm{a}) &= \sum_B p(F=\mathrm{a}|B)p(B) \\ &= p(F=\mathrm{a}|B=\mathrm{d})p(B=\mathrm{d}) + p(F=\mathrm{a}|B=\mathrm{l})p(B=\mathrm{l}) \\ &= \frac{4}{7}0.5 + \frac{1}{4}0.5 \\ &= \frac{23}{56}. \end{aligned} \tag{2.13}$$

Similarly, if the fruit had been a banana, the marginal probability would be:

$$\begin{aligned} p(F=\mathrm{b}) &= p(F=\mathrm{b}|B=\mathrm{d})p(B=\mathrm{d}) + p(F=\mathrm{b}|B=\mathrm{l})p(B=\mathrm{l}) \\ &= \frac{3}{7}0.5 + \frac{3}{4}0.5 \\ &= \frac{33}{56}, \end{aligned} \tag{2.14}$$

which sums to one with the marginal for $F=a$.

With the values for the likelihood from table 2.1, we can now evaluate the posterior probability (eq 2.11). For example, if the fruit is an apple the posterior probability of the dark box $p(B=\mathrm{d}|F=\mathrm{a})$ is given by:

$$\begin{aligned} p(B=\mathrm{d}|F=\mathrm{a}) &= \frac{p(F=\mathrm{a}|B=\mathrm{d})p(B=\mathrm{d})}{p(F=\mathrm{a})} \\ &= \frac{\frac{1}{4}0.5}{\frac{23}{56}} = \frac{7}{23}, \end{aligned}$$

which is much smaller than 0.5. Thus, the posterior probability confirms the intuitive solution that if the fruit is an apple, it most likely did not come from the dark box. The remaining cases can be calculated analogously and the resulting posterior probabilities are summarized in table 2.1, on the right side.

2.2.2 General Discussion

Before concluding this section, we have to mention a few important points. Aside from confirming the intuitive result, the posterior probability calculated from Bayes law provides us with the additional information of how reliable the solution is. In the previous example, a posterior probability close to 0.5 would indicate that the result is not very reliable. This is one of the key differences distinguishing Bayesian model inversion from methods which only provide a point estimate of the variable of interest, i.e. either $B=l$ or $B=d$.

We have also seen from the solution to the boxes of fruits problem that specifying a forward model precedes the solution of the inverse problem. This generative approach can be extended to many other problems, where building a forward model and inverting it using Bayes law is more complex, but holds the advantage of being more easily interpretable [17]. We will see more examples in the following chapters.

Lastly, we should point out a few critical aspects of Bayesian model inversion. In the Bayesian framework, it is always necessary to specify a prior probability function or distribution over the latent variables in the model. The prior probability encodes the information on the latent variables before any data is observed and it will have an impact on the conclusions of the model inversion process.

The prior is typically used to encode model assumptions and for many critics this is a weakness of Bayesian statistics, since the assumptions in the prior will bias the result of the inference. On the other hand, supporters of the Bayesian approach to inference argue, that every inference process, no matter what method it uses, is always based on assumptions [16]. The difference is that in Bayesian statistics the assumptions are stated explicitly via the prior distribution, while in other methods they are hidden, but present nonetheless.

Despite the impact on the inference result, prior probabilities can be very useful when building models. In addition to encoding model assumptions, they can be used to enforce constraints on the variables in the model. The classic example is a model parameter which has to be positive. By choosing a prior distribution for this parameter with support limited to the positive real numbers, the posterior distribution is automatically zero for all negative numbers, since it contains the prior as a multiplicative factor.

In addition, the Bayesian framework, where assumptions or prior knowledge is encoded by a prior distribution and the information from the data is encoded in the likelihood function, offers an intuitive approach to handling

sequential measurements. Assume that in the boxes of fruits example, the player draws multiple times from the box (with replacement) yielding a sequence of observed fruit types $F_1, F_2, \ldots, F_N$. One can of course build a more complex model which treats the N successive observations as one vector observation. However, the complexity of such a model would increase exponentially with N.

A much simpler and more intuitive approach offered by Bayesian inference is to calculate the posterior based on the first observation $p(B|F_1)$. After obtaining the second observation F_2, we simply update the posterior distribution $p(B|F_2, F_1)$ by incorporating the new measurement via the likelihood function, while the posterior from the first step $p(B|F_1)$ becomes the prior of the second step. This scheme can be continued for each subsequent measurement. Thus, the prior is not only used for encoding assumptions but can also be used for incorporating past measurements. This idea forms the basis of many successful algorithms like e.g. the Kalman filter or the particle filter [25].

Another more subtle aspect is the use of priors to break symmetries in the model. For example, if a time series model contains two identical parts for modelling random oscillations on two different time scales and the model parameter are set by fitting the model to training data, then it can happen that for some data sets the first part fits the fast oscillation and the second part fits the slow oscillation, while for other data sets the situation is reversed. This ambiguity caused by the symmetry in the model can be avoided by selecting different prior distributions for the parameters of the two parts of the model.

In the following chapters, we will encounter all the aspects mentioned above. On the one hand, we will see how to use prior distributions to encode model assumptions and enforce constraints. However, we will also witness cases where the prior has an impact on the results of the inference process.

Aside from the controversy about the prior distribution, there is a more fundamental topic of criticism on Bayesian methods, which revolves around the use and interpretation of probability. The reader may have noticed that throughout this section, we have treated every latent variable as a random variable. This is due to the Bayesian interpretation of probability as a measure of uncertainty, which means that unobserved but deterministic variables, like the colour of the box that has been chosen, are treated as random, just like the outcome of a coin flip. For some people, this is a violation of the definition of probability, which should only be used to describe repeatable random experiments, like flipping a coin or rolling a dice, and not for deterministic variables with unknown value.

The debate between Bayesian and non-Bayesian is a complicated issue, which sometimes leads to heated arguments. In this thesis, we will not delve further into this topic. Instead, we will apply Bayesian concepts whenever they offer practical or conceptual benefits, but also acknowledge the weaknesses.

As already hinted at the end of the previous section, the calculation of the marginal probability is the most computationally challenging step in the application of Bayes law, since it involves a summation or integration over all possible values or configurations of all latent variables. In equation (2.13) and (2.14) this summation included only two summands, because there was only one binary variable B. However, the number of possibilities will increase exponentially with the number of variables, which can become computationally infeasible even for models of moderate size. For this reason, there exists a multitude of methods for approximating the marginal probability and in section 2.4 we will introduce two families of approximations which we will apply in chapter 3 and 4 respectively.

However, before doing that, we will first introduce the concept of graphical models, a tool for illustrating the structure of models using graphs. This form of illustration will be used extensively throughout the entire document.

2.3 Graphical Models

In the previous section, we have seen that forward modelling and inverse problems are closely connected. For example, the joint probability density defined in the forward model is required for the application of Bayes law. In the following, we will introduce a tool called graphical models, which can be used to visualize the structure of a model by representing the joint probability density as a graph.

For the box of fruits example, the joint probability density was obtained by applying the chain rule. This applies to many models, with the only difference that for more complex models, the chain rule has to be applied repeatedly. The result is an expression for the joint probability density consisting of a product of probability densities defined over subgroups of the variables involved in the model.

In the above example, the factors in the product are the prior probability for the box colour $p(B)$ and the conditional probability of the type of fruit $p(F|B)$, which depends on the colour. However, for larger models, like e.g. the embedded clustering model in chapter 4, it is difficult to grasp the structure of the model and the dependencies between the variables just by

looking at the factors in the joint density. For this reason, graphical models were proposed as a method to visualize the structure of the model and help understanding the dependencies among the variables in the joint probability density.

The visual representation of probabilistic models is a vast topic and for a detailed treatment we refer the reader to [17, 26, 27]. Different types of graphical models exist, which can be divided into two groups, depending on whether the graph used to represent the model is directed or undirected. Each type has its own advantages and disadvantages and in the following, we will focus on one type of directed graphical models called Bayesian networks, which is ideally suited for visualizing the causal structure of a model.

A Bayesian network represents the joint probability density of a model using a directed acyclic graph (DAG) [17], i.e. a graph with directed edges but without loops. A loop is a series of edges which, when traversed in the direction they are pointing, leads back to the node one started at. This, of course, precludes self-connections in a DAG.

Each node in a Bayesian network represents one factor of the joint probability distribution and the variables it is defined over. As noted above, each factor is a probability distribution, conditioned or unconditioned, which is defined over a subgroup of variables. If the distribution is unconditioned, like e.g. $p(B)$ in the box example, it is represented by a node without parents. On the other hand, nodes representing a conditional probability distribution have parents. The parent nodes are those which represent the variables that the child probability distribution is conditioned upon. In the box example, this means that $p(F|B)$ is represented by a node which has one parent, namely the node representing $p(B)$. Thus, the Bayesian network for the boxes of fruits example is given by the graph in figure 2.2, which contains one node for B and another node for F, with one edge pointing from B to F.

A common convention is that nodes of latent variable like B are represented by open circles, while nodes of variables which can be observed like F are represented by filled circles. Filled nodes are also used to indicate that a variable is fixed to a known value. An example are the parameters in the generative mixture of Gaussians model in figure 4.6.

In the Bayesian network, the structure of the model and the dependencies between the factors in the joint probability distribution are encoded in visual form. However, the main advantage of Bayesian networks is that they visualize the causal relationship among the variables in a model. This can be seen as follows.

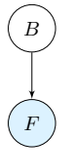


Figure 2.2: The directed graphical model for the boxes of fruits example consists of two nodes representing $p(B)$ and $p(F|B)$. Nodes are labelled with the variables the distribution is defined over and for latent variables like B, the nodes are represented by open circles, while the node for F is filled to indicate that the value of F can be observed. The edge from B to F indicates that the distribution over F is conditioned on B.

The symmetric definition of the chain rule in equation (2.9) suggests that a mathematically equivalent way of writing the joint probability distribution in the boxes of fruits example is to multiply the posterior probability with the evidence: $p(B, F) = p(F)p(B|F)$. However, aside from the fact that the likelihood $p(F|B)$ can be easily specified, while the posterior $p(B|F)$ cannot, this way of writing the joint probability is also counter-intuitive. The interpretation given by the model is that the colour of the box B is an internal state of the system, which cannot be observed, but which influences, or one could say causes the observation F. To express this relation between cause and effect, we prefer to write the joint probability as $p(B, F) = p(B)p(F|B)$, which is represented visually in the graphical model by having the edge point towards the observation.

The above considerations apply for most models, where the edges in the graphical model point from a variable to the variables under its influence. By employing directed edges, Bayesian networks offer an intuitive way to specify the causal relationship between the variables in the model. We will find this property useful when visualizing the embedded clustering model in chapter 4.

Before closing this section, we will introduce another notational aspect of Bayesian networks, which will make the graphical representation of large models more compact. Imaging that in the box of fruits experiment, the player is allowed to draw repeatedly with replacement from the chosen box. This means that there is now a series of N observations represented by the variables $F_1, F_2, \ldots, F_N$. Since the fruit is replaced into the box, the probability distribution for each of the observations is identical. Additionally, we assume that the content of the box is shuffled before each draw, such that the observations are also independent. In this case, the variables $F_1, F_2, \ldots, F_N$ are called independent and identically distributed (iid.) and

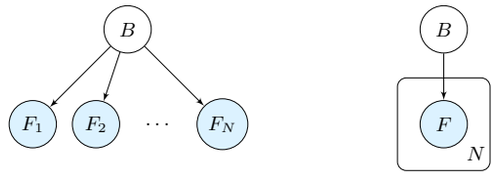


Figure 2.3: Demonstration of the plate notation in graphical models: The graphical model for the boxes of fruits example with repeated drawing with replacement shown on the left has been reproduced in much more compact form with plate notation on the right.

the joint distribution over the box colour and all observations is given by:

$$p(B, F_1, F_2, \ldots, F_N) = p(B) \prod_{n=1}^{N} p(F_n|B). \tag{2.15}$$

This can be represented by a Bayesian network with one node for each observation F_n and a node for B with edges pointing towards each of the observations. However, the nodes for the observations all have the same structure consisting of one node connected to B by an edge pointing towards the observation. In order to avoid repeating part of the model with identical structure, we will introduce the plate notation.

A plate is a rectangle in the graphical model that has a number or variable in the lower right corner. Any nodes or edges contained inside the plate are replicated as often as the number in the corner of the plate suggests. To illustrate this idea, figure 2.3 shows two equivalent versions of the Bayesian network of the boxes of fruits example with repeated drawing. The graph on the left side shows the model without plate notation, while in the graph on the right side plate notation was used to make the network more compact.

Note that plate notation is not limited to expressing multiple observations. Rather, it can be used to represent any part of the model which consists of a repetition of nodes and edges with identical structure. This will allow us to draw compact graphs representing large models like the mixture of Gaussians model or the embedded clustering model in chapter 4.

2.4 Approximate Inference

As previously mentioned, the computationally most demanding part of solving inverse problems is to calculate the marginal probability in the denominator of Bayes law. Equation (2.10), indicates that in the general case

one has to sum (or integrate) over every combination between all latent variables. In the boxes of fruits example, there was only one binary latent variable, which meant that the sum had only two summands. However, for larger models the number of terms in the sum grows exponentially with the number of latent variables, i.e. for a model with two latent binary variables the sum contains four terms.

As an example, consider a model consisting of a humble 30×30 grid of particles, each described by a binary spin variable [16]. The internal state of this model is described by a vector of 900 binary variables, which means that to calculate the evidence, one has to sum over 2^{900} terms. By way of comparison, the age of the universe is estimated to be 2^{58} seconds and the number of electrons in the universe is approximately 2^{266} [16]. Thus, we see that the exact evaluation of the marginal probability or evidence is only possible for the simplest models, while for most practical models approximations are necessary.

The approximation methods at our disposal roughly fall into two categories. The first one is the class of Monte Carlo methods, which tries to find a numerical approximation to the posterior distribution by drawing a set of representative samples. The second class consists of methods that try to find an analytic approximation to the model evidence and posterior distribution. These methods are summarized under the name variational Bayes, since they are based on minimizing a functional using calculus of variations. In the following chapters, we will apply methods from both classes to solve a range of practical model inversion problems. Here, we will introduce these two categories of approximation methods, starting with Monte Carlo.

2.4.1 Monte Carlo

Monte Carlo is a class of very powerful and popular methods for performing inference in intractable models. Here, intractable means that exact inversion of the model by directly applying and evaluating Bayes law is computationally infeasible. The popularity of Monte Carlo methods is partly due to its broad applicability. A model needs to fulfil only few prerequisites in order to be invertible via Monte Carlo.

The general idea behind Monte Carlo methods is to approximate a probability distribution $p(X)$, called the target distribution, by drawing a set of representative samples $\{x_1, x_2, \ldots, x_N\}$ from it. This set of samples can then be used to calculate values of interest, e.g. to approximate the expected value of X using the sample mean $\bar{x} = \sum_n x_n$. In order to invert a model with observation Y and latent variable X using Monte Carlo, one simply

specifies the posterior distribution $p(X|Y)$ as the target distribution.

At this point, the reader might point out that to obtain $p(X|Y)$, we first need to calculate the marginal probability $p(Y)$, which is assumed to be intractable. However, a property of many Monte Carlo methods, which makes these methods so powerful, is that direct access to the target distribution is not necessary. Instead, it is sufficient if one can evaluate, for any value of X, a function $p^*(X) = const \cdot p(X)$ which is proportional to the target distribution up to a constant factor. In the case of the posterior probability, the function $p^*(X)$ is given by the joint probability density $p(X,Y) = p(X|Y)/p(Y)$ (eq (2.8)), which is proportional to the posterior density up to the model evidence $p(Y)$. And since $p(Y)$ does not depend on X, it can be treated as a constant factor by the Monte Carlo method.

In fact, the statement above means that Monte Carlo methods can even be used to invert models where the joint probability distribution does not have a closed-form representation. Since we only need to evaluate the function $p^*(X)$ for any input argument, even a model in which the joint distribution is only given implicitly, e.g. via a black box computer program, can be inverted using Monte Carlo.

Monte Carlo methods which possess the property mentioned above include importance sampling, rejection sampling and Metropolis-Hastings, as well as Markov chain Monte Carlo (MCMC), a generalization of Metropolis-Hastings. While a special case of importance sampling, called the particle filter, is widely used in practice [25], importance sampling in general does not scale well to high-dimensional models [16]. Similarly, rejection sampling is well suited for certain one-dimensional problems, but also does not scale well with the dimensionality of the problem [16]. However, rejection sampling holds the advantage of producing independent samples, which is not the case for Markov chain Monte Carlo. In MCMC the output is a set of samples $\{x_1, x_2, \ldots, x_N\}$ where x_{n+1} is correlated with x_n. On the other hand, MCMC is applicable to high-dimensional models and requires less knowledge on the target distribution [16].

MCMC will be used in section 3.5.2 for inverting the probabilistic ballistocardiogram model to detect heartbeats. In the following, we will introduce the general idea behind MCMC in more detail and introduce the Metropolis-Hastings algorithm, which is a special case of MCMC.

Markov Chain Monte Carlo

Markov chain Monte Carlo (MCMC) is a special kind of Monte Carlo, which is very popular due to its broad applicability. The only condition for MCMC

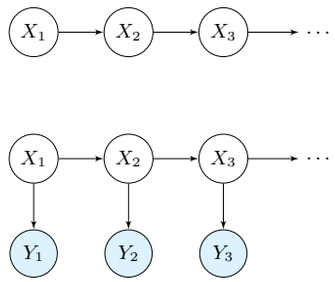


Figure 2.4: Top: graphical model of the most basic version of a Markov chain without observation. Bottom: graphical model of a Markov chain with observables Y_n.

to be applicable is that the target distribution $p(X)$ can be evaluated up to a constant for any given input argument [16].

MCMC works by generating a sequence of samples $x_1, x_2, \ldots, x_N$ where each sample depends on the previous. This sequence of samples can be viewed as realizations of the random variables in a so called Markov chain, which is a sequence of random variables in which each variable depends on its predecessor. A Markov chain is described by a distribution like:

$$p(X_1, X_2, \ldots, X_N) = p(X_1)p(X_2|X_1)\cdots p(X_n|X_{n-1})\cdots p(X_N|X_{N-1}). \tag{2.16}$$

If the so called transition probability $p(X_n|X_{n-1})$ is the same for all n, the Markov chain is said to be homogeneous.

Figure 2.4 shows graphical models of Markov chains, which are often used as simple models for time series. The famous Kalman filter algorithm, as well as the particle filter are both based on the Markov chain with observations [25], which is represented by the lower graph in figure 2.4. The upper graph shows the basic Markov chain without observation, which forms the basis for MCMC.

The sequence of samples generated using MCMC has the special property that if one ignores the ordering of the samples and treat them as multiple realizations of one random variable X, they will be distributed according to the target distribution $p(X)$ [16]. However, one has to keep in mind that these samples are not independent, which means that they should not be used to calculate second order statistics directly, i.e. the variance of the target distribution is not equal to the sample variance:

$$\mathrm{var}\big(p(X)\big) \neq \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^2. \tag{2.17}$$

On the other hand, the correlation between samples has no impact on the calculation of first order statistics [28].

In order to achieve the property described above, the Markov chain constructed in MCMC must satisfy certain conditions. One of them is the so called detailed balance condition on the transition probability [28]:

$$p(x)p(x'|x) = p(x')p(x|x'). \tag{2.18}$$

It means that the transition probability has to be chosen in such a way that the probability of starting from a sample x, which is drawn from the target distribution $p(X)$, and moving to another sample x' via the transition probability distribution $p(x'|x)$ must be equal to the probability of starting at x' and moving to x.

A transition probability distribution that satisfies the detailed balance condition ensures that the target distribution $p(X)$ is an invariant distribution of the Markov chain constructed using that transition distribution [16]. For $p(X)$, being an invariant distribution means that given a sample x_n from $p(X)$ and moving to another sample x_{n+1} via the transition distribution $p(x_{n+1}|x_n)$ will ensure that x_{n+1} is also a sample from $p(X)$.

The other condition that the Markov chain has to satisfy is called ergodicity, which means that no matter from which distribution the first sample x_1 is drawn, $p(X_n)$ will converge to the target distribution $p(X)$ for $n \to \infty$ [16]. In [29], Neal *et al.* proof that any homogeneous Markov chain is ergodic under a few weak conditions on the invariant distribution and transition probability. For the mathematical details, we would like to refer the reader to the original publication [29].

Above, we have described the idea behind MCMC and the conditions that the transition density of a Markov chain has to satisfy in order to be useable in an MCMC scheme. However, the question of how to construct such a chain in practice remains open. In the subsequent section, we will offer an answer to this question by introducing the Metropolis-Hastings algorithm as an example of an MCMC method.

Metropolis-Hastings

The Metropolis-Hastings (MH) algorithm is named after Metropolis, who first proposed the method in [30] and Hastings, who subsequently generalized the algorithm [31]. The MH algorithm provides a way to construct a transition distribution such that the target distribution is the invariant distribution of the Markov chain.

The scheme works as follows: Given the sample from the last step x_n, we first generate a sample x' by drawing from a proposal distribution $q(x'; x_n)$, which is a probability distribution over x' that depends on x_n. The so called proposal x' is accepted with probability a calculated using the formula:

$$a = \min\Big(1, \frac{p^*(x')}{p^*(x_n)} \frac{q(x_n; x')}{q(x'; x_n)}\Big), \tag{2.19}$$

where $p^*(\cdot)$ is a function proportional to the target distribution. Intuitively, the above formula says that if in a long chain the transition $x' \to x_n$ is more likely to occur than $x_n \to x'$, then we always accept. Otherwise, we accept with probability equal to the ratio between the probabilities of the transitions.

If the proposal is accepted, it becomes the new sample in the Markov chain: $x_{n+1} = x'$. If x' is rejected, the old sample becomes the new sample: $x_{n+1} = x_n$. This distinguishes MH from rejection sampling, where only acceptance will lead to gaining a sample.

The standard choice is to obtain the proposal x' by perturbing the old sample x_n with Gaussian noise [16], in which case the proposal distribution is given by:

$$q(x'; x_n) = \mathcal{N}(x_n, \sigma I). \tag{2.20}$$

It can be shown that the scheme described above satisfies detailed balance [17]. And that for the correct choice of the proposal distribution q, $p(X_n)$ indeed converges to the target distribution for $n \to \infty$ [16]. However, the samples in the Markov chain are not independent, but highly correlated with their predecessor. This means that the accuracy of the result, which for Monte Carlo methods increases with the number of independent samples, does not depend directly on the length of the Markov chain in MH.

In order to obtain samples using MCMC which are effectively independent, one has to pick samples in the chain which are separated by a certain number of steps m. A simple calculation on a toy problem [16, 17] shows that the number of steps m, one has to wait for two samples x_n and x_{n+m} in the chain to become effectively independent, scales quadratic with $m \simeq \sigma_{max}/\sigma_{min}$, which is the ratio between the largest (σ_{max}) and smallest (σ_{min}) length scales of the target distribution $p(X)$.

In other words, if the target distribution p(X) is a function with a long and thin mode and if the ratio between length and width of the mode is about 100, one has to run a MCMC algorithm for $N = 10\,000$ steps in order to obtain a result with an accuracy equivalent to that of a result obtained with 100 independent samples. The low speed of MCMC is one of its drawbacks.

Summary

In this section, we have introduced Monte Carlo methods as a way to approximate the posterior distribution for intractable model, i.e. models where the direct inversion with Bayes law is computationally infeasible. Additionally, we have described the Metropolis-Hastings algorithm, which is an example for a Markov chain Monte Carlo method. Due to their complexity and the number of different variations, Monte Carlo methods represent an own field of research and the introduction given here can only present the most basic ideas. For a more detailed introduction to Monte Carlo methods in general, we refer the reader to [16, 17], while [28] provides an in depth treatment of MCMC methods.

The disadvantage of Monte Carlo methods in general and MH in particular is their slow speed and high demand for computational resources. On the other hand, they possess the asymptotic exactness property, which means that as the number of samples drawn ($N \to \infty$) increases, the result converges to the exact solution [16, 17].

In the next section, we will introduce a completely different approach to the approximate inference problem, which is based on finding an analytic approximation to the marginal probability. These methods, which are known as variational approximations or variational Bayes, lack both the universal applicability and the asymptotic exactness of Monte Carlo, but have the advantage of being less computationally intensive. The savings in computing time can be immense, as we will see in chapter 4.

2.4.2 Variational Methods

The reason for the intractability of many model inversion problems is the computational complexity of calculating the marginal probability. Monte Carlo methods, introduced in the previous section, solve this problem by drawing representative samples from the posterior distribution, which does not require calculating the marginal probability. In this section, we will introduce a variational approximation method known as variational Bayes, which pursues the alternative method of approximating the marginal probability itself [17, 32, 33, 34, 35].

The marginal probability, also called the model evidence, consists of a sum or an integral, the computational complexity of which is often the reason for the intractability of the model inversion. For sums, it is mostly the number of terms which is too high to be evaluated in a timely manner using the available computational resources. For integrals, the problem lies in the

integrand which is too complicated to allow for an analytic solution [16, 17].

In variational Bayes, the marginal probability is approximated by deriving a lower bound on the sum or integral that represents it. This lower bound depends on parameters, which we call variational parameters. Maximizing the bound with respect to the variational parameters will provide an approximation to the marginal probability and at the same time also yields an approximation to the posterior distribution [17]. In the following, we will derive the variational Bayes approximation for a general setting. The results presented here will be applied to a real world problem in chapter 4.

Derivation of the Variational Bound

We start by defining the Kullback-Leibler (KL) divergence as a way of specifying dissimilarity between two probability distributions. If $p(X)$ and $q(X)$ are both probability distributions over X, the KL divergence between q and p is defined by the integral:

$$\mathrm{KL}\left(q(X)||p(X)\right) = \int_X q(X) \log \frac{q(X)}{p(X)} dX. \tag{2.21}$$

For a discrete random variable, the integral is replaced by a sum over all possible values of the variable.

Note that the KL divergence is not invariant against exchanging the order of its arguments, i.e. $\mathrm{KL}(q||p) \neq \mathrm{KL}(p||q)$. However, it is easily proven that the KL divergence is always larger or equal zero ($\mathrm{KL}(q||p) \geq 0$), with equality only if both probability distributions are equal [16, 17]. This is an important property, which we will need later on.

If we assume that our model contains observable variables X, as well as latent variables Z, which we would like to infer from the observation X, Bayes law is given by:

$$p(Z|X) = \frac{p(X|Z)p(Z)}{p(X)}. \tag{2.22}$$

As mentioned, the most difficult part is the calculation of the marginal probability or model evidence in the denominator:

$$p(X) = \int_Z p(X|Z)p(Z)dZ, \tag{2.23}$$

which often contains an intractable integral or an intractable sum if the latent variable is discrete. Here, X and Z represent the set of observed and

latent variables, respectively, and the integral over Z means integrating (or summing) over all latent variables in the set.

In variational Bayes, the aim is to derive an approximation which is a lower bound to the marginal probability. For this purpose, we rearrange equation (2.22) and take the logarithm on both sides:

$$\log p(X) = \log \frac{p(X,Z)}{p(Z|X)}. \tag{2.24}$$

Remember that the product between likelihood and prior equals the joint probability: $p(X,Z) = p(Z|X)p(Z)$.

Now, we choose an arbitrary distribution $q(Z)$, called the variational distribution, over the latent variables Z. We multiply both sides of equation (2.24) with $q(Z)$ and integrate over Z:

$$\int_Z q(Z) \log p(X) dZ = \int_Z q(Z) \log \frac{p(X,Z)q(Z)}{p(Z|X)q(Z)} dZ. \tag{2.25}$$

Since $\log p(X)$ does not depend on Z, it can be pulled out of the integral and $\int_Z q(Z)dZ = 1$ by definition. Thus, after expanding the fraction on the right hand side with $q(Z)$, we have:

$$\log p(X) = \int_Z q(Z) \log \frac{p(X,Z)}{q(Z)} dZ - \int_Z q(Z) \log \frac{p(Z|X)}{q(Z)} dZ \tag{2.26}$$

$$= \mathcal{F} + \mathrm{KL}\left(q(Z)||p(Z|X)\right). \tag{2.27}$$

Comparing this equation with the definition of the KL divergence (eq (2.21)), we recognize the second term in the first line on the right hand side as the KL divergence between $q(Z)$ and the posterior distribution $p(Z|X)$. The first term is called the free energy $\mathcal{F}$, the value of which depends on our choice of q.

When considering that the KL divergence is always greater or equal zero, we can draw a few conclusions about the free energy. First, the free energy is always smaller than the logarithm of the evidence, i.e. $\mathcal{F}$ is the lower bound we are looking for. Second, when maximizing the value of $\mathcal{F}$ with respect to $q(Z)$, i.e. varying the form of the distribution $q(Z)$ as to increase $\mathcal{F}$, one also minimizes $\mathrm{KL}\left(q(Z)||p(Z|X)\right)$ at the same time. This means that the larger $\mathcal{F}$ becomes, the closer $q(Z)$ is to the posterior probability density $p(Z|X)$. In the extreme case, if we choose q equal to the posterior density, $\mathcal{F}$ would be equal to the evidence. However, this trivial solution does not work in practice, since we do not have access to the posterior density $p(Z|X)$ due to the intractability of evaluating the marginal likelihood.

In this context, it is important to note that a consequence of our ignorance about $p(Z|X)$ is the inability to evaluate the KL divergence between q and $p(Z|X)$. This is also the reason why the optimum setting for q must be found by maximizing $\mathcal{F}$ and not by minimizing KL $\big(q(Z)||p(Z|X)\big)$, although both operations are mathematically equivalent.

The name variational Bayes comes from the fact that we are maximizing the free energy $\mathcal{F}$, which is a functional, with respect to the function q. In practice however, one has to make sure that the integration involved in calculating $\mathcal{F}$ stays tractable. The common strategy is to restrict $q(Z)$ to a certain class of (simple) distributions such that the integral remains analytically solvable, and maximize the free energy with respect to the parameters on which $q(Z)$ depend. A prominent example is to restrict q to a Gaussian distribution $q(Z) = \mathcal{N}(Z|\mu, \sigma)$ and maximize $\mathcal{F}$ with respect to the mean and covariance. This method of restricting q to a Gaussian distribution is used extensively in [36], where it is called the Laplace approximation.

On the other hand, if Z comprises a set of variables $Z = \{Z_1, Z_2, \ldots\}$, another way of restricting $q(Z)$ is to divide Z into subgroups and assume that q can be factorized into distributions over the subgroups: $q(Z_1, Z_2, Z_3, \ldots) = q(Z_1)q(Z_2, Z_3)\cdots$. This kind of restriction is related to a method known as mean field approximation in statistical physics [17] and unlike in the previous case, there are no restrictions on the functional form of the distribution. In practice, often both kinds of restrictions, i.e. restricting the functional form and assuming a factorization on q, have to be applied in order to make the calculation of the free energy $\mathcal{F}$ tractable.

Optimization of the Free Energy

Maximizing the free energy with respect to the parameters of $q(Z)$ can in principle be done with any method. Like in [36], one can simply take the derivative of $\mathcal{F}$ with respect to the parameters of $q(Z)$ and use a standard gradient optimizer to maximize $\mathcal{F}$.

As an alternative, we will present an iterative scheme which updates the parameters of $q(Z)$ in a way that is guaranteed to increase $\mathcal{F}$. This scheme has been applied to many model inversion problems, including learning the parameters of clustering models [17, 33, 37] and we will use it in chapter 4 to solve the embedded clustering problem.

We assume a mean field approximation, where Z is divided into I subgroups $\{Z_1, Z_2, \ldots, Z_I\}$ and $q(Z)$ is factored into a product of independent distributions over the I subgroups: $q(Z) = \prod_{i=1}^{I} q(Z_i)$. Here, each Z_i stands for a whole subgroup of latent variables. When plugging this relation into

the expression for the free energy from equation (2.26) and isolating the distribution $q(Z_j)$ over one of the subgroups, we arrive at:

$$\mathcal{F} = \int q(Z) \log p(X, Z) dZ - \int q(Z) \log q(Z) dZ \tag{2.28}$$

$$= \int \prod_{i=1}^{I} q(Z_i) \log p(X, Z) dZ - \sum_{i=1}^{I} \int q(Z_i) \log q(Z_i) dZ_i \tag{2.29}$$

$$= \int q(Z_j) \Big(\int \prod_{i \neq j} q(Z_i) \log p(X, Z) dZ_i \Big) dZ_j$$

$$- \int q(Z_j) \log q(Z_j) dZ_j + const, \tag{2.30}$$

where *const* stands for all terms in the summation that do not depend on $q(Z_j)$ and $\prod_{i \neq j}$ denotes a product over all $i \in \{1, \ldots, j-1, j+1, \ldots, I\}$.

The inner integral in equation (2.30) can be interpreted as the logarithm of a probability distribution $\tilde{p}(X, Z_j)$ over the observation X and the current group of latent variables Z_j:

$$\log \tilde{p}(X, Z_j) := \int \prod_{i \neq j} q(Z_i) \log p(X, Z) dZ_i + const. \tag{2.31}$$

Here, *const* denotes the logarithm of the unknown normalization constant of $\tilde{p}(X, Z_j)$. Note also that the integral on the right hand side can be interpreted as the expectation of $\log p(X, Z)$ under the distribution $\prod_{i \neq j} q(Z_i)$.

With this, we can express the free energy as:

$$\mathcal{F} = \int q(Z_j) \log \tilde{p}(X, Z_j) dZ_j - \int q(Z_j) \log q(Z_j) dZ_j + const \tag{2.32}$$

$$= \int q(Z_j) \log \frac{\tilde{p}(X, Z_j)}{q(Z_j)} dZ_j + const \tag{2.33}$$

$$= -\operatorname{KL}\big(q(Z_j) || \tilde{p}(X, Z_j)\big) + const, \tag{2.34}$$

which we recognize as the negative KL divergence between $q(Z_j)$ and $\tilde{p}(X, Z_j)$, up to an additive constant. Since we know that the KL divergence is minimized when the distributions in its arguments are equal, we see that equation (2.34) is maximized with respect to $q(Z_j)$ for $q(Z_j) = \tilde{p}(X, Z_j)$. Thus, by choosing:

$$q(Z_j) = \frac{\exp\Big(\int \prod_{i \neq j} q(Z_i) \log p(X, Z) dZ_i \Big)}{\int \exp\Big(\int \prod_{i \neq j} q(Z_i) \log p(X, Z) dZ_i \Big) dZ_j}, \tag{2.35}$$

or equivalently:

$$\log q(Z_j) = \int \prod_{i \neq j} q(Z_i) \log p(X, Z) dZ_i + const \tag{2.36}$$

we can maximize the free energy $\mathcal{F}$ with respect to the factor $q(Z_j)$. Here, the *const*-term corresponds to the negative logarithm of the denominator of equation (2.35), which does not depend on $q(Z_j)$.

However, we also see that the maximization with respect to one factor $q(Z_j)$ depends on the other factors, since the right hand side of equation (2.36) involves an expectation with respect to all factors expect for $q(Z_j)$. Therefore, equation (2.36) has to be evaluated successively for each of the factors $q(Z_j) : \ j = 1, \ldots, I$. That is, we have to use the current value of the parameters of $q(Z_i) : \ i = 1, \ldots, j-1, j+1, \ldots, I$ to evaluate the integral on the right hand side of equation (2.36). The result can then be used to update the parameters of $q(Z_j)$. Then, we need to increment j and repeat the process for the next factor. This successive evaluation has to be iterated until the factors $q(Z_j)$ and their parameters converge to a stable value.

It can be shown that each time equation (2.36) is evaluated, the parameters of $q(Z_j)$ will change in a way that the free energy increases [17]. Since $\mathcal{F}$ is also upper bounded by the marginal probability, this means that the update scheme described above is guaranteed to converge, although the fix point found can be a local maximum. By repeating the update scheme with different initial conditions, one can increase the likelihood to find the global optimum.

It should be noted that the accuracy of the approximation, i.e. how close the global maximum of $\mathcal{F}$ is to the true value of $p(X)$, is determined only by how many restrictions are put on the variational distribution $q(Z)$. In general, less restrictions lead to better accuracy, but care must be taken that all integrals involved stay analytically solvable. On the other hand, increasing the number of iterations of the variational update scheme does not increase the achievable accuracy of the variational approximation, but nevertheless, it can affect the convergence of the scheme.

As mentioned, a variational Bayes approximation using this maximization strategy has been applied successfully to parameter estimation in mixture modelling [17, 33, 37], which will be one of the components of the embedded clustering model we focus on in chapter 4. Aside from the variational free energy maximization introduced here, there are also other variational approximation methods e.g. expectation propagation. However, these methods are outside the scope of this introduction and we refer the reader to [17] for a more comprehensive documentation on variational approximation methods.

Connection to Statistical Physics

The concept of minimizing free energy originated from the fields of thermodynamics and statistical physics, where the so called Helmholtz free energy F is defined as:

$$F = U - TS. \tag{2.37}$$

Here, U is the internal energy, T is the (absolute) temperature and S is the entropy of the system [38, 39]. In this context, we should point out that the variational free energy, as introduced in equation (2.28), should more accurately be called the negative free energy [40], since it can be written as:

$$\begin{aligned} \mathcal{F} &= \int q(Z) \log p(X, Z) dZ - \int q(Z) \log q(Z) dZ && (2.38) \\ &= \big\langle \log p(X, Z) \big\rangle_{q(Z)} + \mathcal{H}\left(q(Z)\right), && (2.39) \end{aligned}$$

where the angular brackets $\langle \cdot \rangle_q$ denote the expectation of the term inside the brackets with respect to the distribution q.

The first term in equation (2.38) can be interpreted as the expectation over the logarithm of a probability distribution, which has a similar functional form as the negative average energy in statistical physics [38, 39]. The second term is the information theoretic entropy $\mathcal{H}$ [41]. Compared with equation (2.37), the functional form is very similar except for the sign.

Summary

Before closing this section, we would like to remark that Monte Carlo and variational Bayes represent the two extremes in the field of approximate inference, with Monte Carlo being a computationally intensive numerical method, while variational Bayes offers a much faster analytic alternative. Monte Carlo is applicable to many problems and requires almost no effort for adapting it to the specific model, although we will see a counter example in chapter 3, where the structure of the model does indeed require us to adapt the MCMC scheme to our needs. On the other hand, the update steps in variational Bayes requires solving the integral in equation (2.36), which should possess an analytic solution due to the restrictions to the variational distribution $q(Z)$, but which still can prove to be hard to solve. In exchange, one gains a much faster alternative to Monte Carlo.

The current direction of research in this field is to develop approximate inference methods which combine advantages from both Monte Carlo and variational methods. An example for this is hybrid Monte Carlo, also known

as Hamiltonian Monte Carlo, where additional information on the target distribution $p(X)$ is used to obtain independent samples with shorter Markov chains. Due to the broadness of this topic, we have to refer the interested reader to other texts [16, 17, 28], which cover the topic in a more comprehensive way than it is possible within the scope of this thesis.

In the next section, we will cover the topics of model comparison and the relation between parametric and non-parametric methods. These topics will help the reader to understand the relation between the methods introduced in the different chapters of this thesis. These topics represent an overarching theme encountered throughout the entire thesis, which demonstrates how modelling can help us link seemingly unrelated problems and applications to a common basis.

2.5 Advanced Topics

In this section, we introduce various advanced topics, which deal with the relationship between different models. First, we will delve into the topic of model comparison and model selection. This topic deals with the question of how to select, based on the observed data, the most appropriate model out of several competing hypothesis. Then, we will introduce a class of models known as non-parametric models and discuss the differences between these models and their parametric counterparts.

The topics in this section are less technical in nature than those of the previous sections. Instead, they will explore some important concepts and notions, which we will need in order to understand the common theme underlying the different aspects in this thesis.

2.5.1 Model Comparison

In section 2.2, we have introduced Bayes law as a method to invert a probabilistic model and obtain the posterior distribution over the latent variables, which can e.g. be model parameters, given the observation. However, in some situations there are more than one hypothesis about how the observed data has been generated, leading to a multitude of models. In these cases, we have to solve a model selection problem, in addition to the model inversion problem.

One approach to this problem is to introduce an additional variable

$$M \in \{m_1, m_2 \ldots, m_I\}, \tag{2.40}$$

which acts as a label for the model [42]. Thus, if we denote the observed data as X, the posterior distribution over the latent variables Z_i in model m_i is given by:

$$p(Z_i|X, M=m_i) = \frac{p(X|Z_i, M=m_i)p(Z_i|M=m_i)}{p(X|M=m_i)}. \tag{2.41}$$

This equation is Bayes law (eq (2.10)), with the inclusion of an additional variable M, indicating that we have selected model m_i.

It is now possible to extend the principles behind Bayesian model inversion to the problem of model selection by simply applying Bayes law on the level of models instead of on the level of latent variables and parameters within each model [16, 17]. Consequently, we can try to calculate the posterior distribution over the model label M given the data $p(M|X)$, which tells us how probable model M is in the light of the data X observed so far:

$$p(M|X) = \frac{p(X|M)p(M)}{p(X)}. \tag{2.42}$$

Here, $p(M)$ is the prior distribution over models and the marginal probability of the data $p(X)$ can be obtained by summing over all models:

$$p(X) = \sum_{M=m_1} p(X|M)p(M). \tag{2.43}$$

According to the sum rule (eq (2.7)), we can obtain the likelihood $p(X|M)$ by summing or integrating out the latent variables Z_i of the respective model:

$$p(X|M) = \int_{Z_i} p(X|Z_i, M)p(Z_i|M)dZ_i. \tag{2.44}$$

This approach to model comparison is also known as Bayesian model selection [16, 17] and the reader might have noticed that the expression $p(X|M)$, which we called likelihood on the model selection level, is nothing else than the marginal probability in the model inversion step (eq (2.41)). In light of this observation, it becomes clear why the marginal probability is also known under the name model evidence and why, throughout this chapter, we have stressed the importance of calculating the marginal probability.

Under normal circumstances, there is no reason to prefer one of the models over the others. Therefore, most of the time, the prior over models $p(M)$ is chosen flat, i.e. $p(M) = const$ across all models. In this case, the only difference comes from the model evidence and two model can be compared to each other via the ratio between their evidence $p(X|m_i)/p(X|m_j)$, which is called the Bayes factor [43].

There are no limits to what kind of models can be compared. The set of models $\{m_1, m_2 \ldots, m_I\}$ can be a simple list of different versions of the same model with varying complexity or degrees of freedom. The classic example would be polynomials with different degrees or auto regressive models with different order. On the other hand, nothing prevents us from comparing completely different hypothesis about how the data was generated, as long as the model evidence or an approximation thereof can be calculated [42].

In chapter 4, we will see a concrete example of the application of this technique, where versions of a model are compared, which differ in some of their parameters. These parameters, when interpreted in the context of the application, corresponds to different hypothesis about the system underlying data generation.

The ideas introduced in this section, are a straight forward extension of Bayesian inference to the model comparison problem. However, just like in the case of model inversion, there are scientists who oppose the use of the Bayesian approach, preferring the classical approach to hypothesis testing based on p-values [16, 17]. As mentioned, the debate between Bayesians and non-Bayesians is beyond the scope of this thesis. At this point, we will only note that despite some shortcomings, Bayesian model comparison offers a flexible and conceptually simple approach to the problem of model comparison, which has been applied to problems similar to those discussed in this thesis and which gives the impetus for some of the approaches taken, especially in chapter 4.

2.5.2 Parametric and Non-Parametric Models

When dealing with modelling and algorithms, one often comes across a common distinction between so called parametric and non-parametric models or methods. In fact, most models or methods can be classified as either parametric or non-parametric [16, 17]. In this section, we will provide a short overview on this topic.

The name non-parametric is a misnomer, as it does not denote a model or method that has no parameters. Rather, it denotes a model where the number of parameters is infinite or grows with the number of observed data points. On the other hand, the term parametric denotes a model with a fixed number of parameters. This is best illustrated via an example.

A classic example for a parametric method is a polynomial of fixed degree in the context of curve fitting. In this case the parameters are the coefficients of the polynomial. Their number is determined by the degree of the polynomial

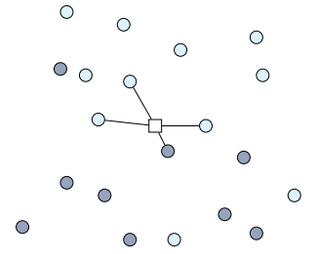


Figure 2.5: kNN classification in two dimensions with $k = 4$. Filled circles are the vectors in the training data set, where light grey denotes class 1 and dark grey denotes class 2. The rectangle is a test data point with unknown class label. The lines connect it to the four closest points in the training set consisting of three points from class 1 and one point from class 2. Thus, the label assigned to the test data point by kNN would be class 1.

and not by the number of data points used for fitting the coefficients. Once the coefficients are fitted, they are sufficient to make predictions, i.e. to extrapolate the curve, while the data point can be discarded [17].

With non-parametric methods, the number of parameters grows with the data. In section 3.5.2, we will encounter the Gaussian process, which is the non-parametric counterpart of the polynomial for curve fitting [44]. Here, we will introduce the well-known k-nearest neighbours (kNN) classification method as an example of a non-parametric method.

In kNN, the task is to classify vectors of fixed dimension into one of several categories [17]. For this purpose, one needs a training database consisting of a set of example vectors for which the correct category label is known. A new vector is classified by finding the k closest vectors among all vectors in the training database and assigning the new vector to the category which holds the majority among these k nearest neighbours [17]. To measure distance, any metric can be used. Popular choices are the Euclidean distance or the Mahalanobis distance, which is obtained by taking the square root of a quadratic form: $\sqrt{(\boldsymbol{x}_1 - \boldsymbol{x}_2)^T A^{-1} (\boldsymbol{x}_1 - \boldsymbol{x}_2)}$, with A being a suitable positive definite matrix [16]. Figure 2.5 illustrates the working principle of kNN.

The reason why kNN is considered a non-parametric method is that the parameters of kNN are the vectors in the training database. The number of neighbours k is a so called hyper-parameter, which governs the overall characteristics of the algorithm [17]. The actual classification result is determined mainly by the parameters, i.e. the vectors in the training database and their associated labels. Their number has no upper limit, which means the number of parameters grows with the size of the training dataset.

From the examples given above, we see that the advantage of non-parametric methods is that they are not limited by an underlying model with finite degree of freedom. In the case of parametric methods, the flexibility of the model is limited by the number of parameters, which has to be fixed in advance. This can act as a bottleneck when fitting a complex dataset [17].

Taking polynomial curve fitting as an example, a polynomial of degree P can have at most $P - 1$ extrema. If the true function underlying the data generation process has more than $P - 1$ extrema, the polynomial will not be able to capture the detailed structure of the underlying curve. If one tries to counter this problem by setting P to a very high value by default, two problems will arise. First, the complexity of the true function that generated the data is not known in practice. Second, a very flexible model with many degrees of freedom will tend to concentrate on capturing small details and overfit the data in the process, leading to poor generalization performance [17]. In the case of polynomial curve fitting, a too high degree results in wild oscillations of the polynomial [17]

A non-parametric method does not suffer from these problems, because it keeps the entire dataset, which grants access to all available information about the underlying structure of the problem, even if this structure is highly nonlinear and difficult to describe [16, 44]. This means that the method adapts its flexibility to the problem. At the same time overfitting can be avoided by tuning the hyperparameters [17]. For kNN, the hyperparameter k has to be chosen large enough such that looking at the k closest neighbours give stable results, and small enough such that the result is not dominated by the category with the most data points [17].

However, the disadvantage of non-parametric methods is the increase in computational complexity with growing size of the dataset. This leads to a dilemma for non-parametric methods, where, in order to obtain as much information as possible, one would like to increase the size of the dataset, but in doing so one also increases the time the algorithm takes to compute the result [17]. For parametric methods, only the complexity of the fitting step increases with the size of the dataset, while the time needed for computing results or making predictions only depends on the number of parameters, which is fixed [17].

In the coming chapters, it will become clear that the methods and models introduced throughout this thesis all fall into one of the categories introduced here. However, this categorization will also help us to realize common themes connecting the models, despite the different ways they were derived and the seemingly unrelated fields of applications from which they originated.

We start with the topic of nonlinear source separation, where we encounter several modelling aspects. Some of these models were developed specifically for improving the source separation algorithm, while others were inspired by the separation result. We will also see that these model are, in fact, related to the models from chapter 4, which deal with an entirely different application, namely clustering.

Chapter 3

Source Separation

We start the main part of this thesis with the introduction of a nonlinear source separation method. By source separation we mean the task of extracting high dimensional multivariate data from noisy scalar time series. The approach presented in this chapter is based on a nonlinear noise reduction method called locally projective noise reduction (LPNR) [14]. Iterated application of this noise reduction method yields a separation algorithm with unique properties not encountered in linear methods [45].

In the course of the discussion, it will become apparent that the LPNR method, which was originally developed to denoise chaotic time series, is closely connected to mixture modelling, a topic that also plays an important role later in chapter 4. However, this is not the only modelling aspect in this chapter. We show that modelling the time series helps verify and improve source separation results. The ability to simulate data with properties that can be chosen by the experimenter leads to a better understanding of the properties of the separation algorithm. On the other hand, source separation allows creating and improving models for each of the sources. This can lead to improvement in the subsequent higher level signal processing steps.

Using ballistocardiography (BCG) as an example modality, we demonstrate the afore-mentioned points. However, these methods are not limited to BCG and application to other modalities and domains are possible. Examples include electrocardiography (ECG), magnetocardiography, other mechanical cardiography methods like seismocardiography or even radar based methods.

The structure of this chapter is as follows. After a short introduction to BCG, we present the LPNR and discuss the connection of LPNR to mixture modelling. Then, we introduce the LPNR-based signal separation and apply it to the BCG components separation problem. Furthermore, we

demonstrate how a model of the BCG can be used to verify and improve the source separation algorithm. After verification, the result of the source separation algorithm can be used to build an improved model of the cardiac component of the BCG. This second BCG model can then be used to solve higher level signal processing tasks, e.g. heartbeat detection. We demonstrate this by presenting a Monte Carlo beat detection method for BCG which is built around the improved BCG model.

3.1 Ballistocardiography

In this chapter, we use ballistocardiography (BCG) as an example modality to demonstrate how the methods introduced here can be applied to real world problems. Therefore, we will provide a short introduction to this modality in this section.

BCG is a non-invasive method used to measure mechanical cardiac activity through measurement of body movements caused by the heartbeat motion. In contrast to ECG, which assesses the cardiac activity indirectly by measuring the electric excitation that accompanies the activities of the heart muscles, BCG provides a more direct way to assess the heartbeat activity.

The BCG modality was investigated intensively during the 1960's [46] and in this early phase, it consisted of measuring the movement of the suspended body frame caused as a reaction to the contraction motion of the heart and the blood flow. For this purpose the subject was attached to a BCG table, which was suspended from the ceiling using cables [46]. These BCG tables were carefully calibrated with the aim to derive, from the measured body movements, quantitative estimates of measures like cardiac output, the blood volume pumped per beat. Unfortunately, these calibrated BCG tables were also large and difficult to operate. Along with the development of alternative methods like ultrasound and echocardiography, this was one of many factors contributing to the decline in BCG research during the 1980's [47].

However, with advances in sensor technology and signal processing, there seems to be a renewal of interest in BCG in the past decade, with a strong focus on home monitoring applications. Modern sensors allow BCG to be recorded using very compact devices, e.g. pressure sensitive films or piezo based sensors [47], allowing BCG recording devices to be built into everyday objects like weighing scales [48], beds [49, 50] or chairs [51, 52]. Even wearable BCG devices have been developed [53, 54].

Most of these modern BCG devices are not calibrated, but they open the

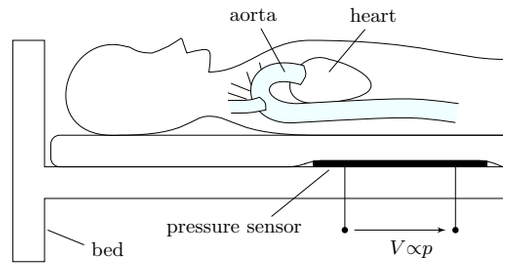


Figure 3.1: Schematic of an uncalibrated bed-mounted BCG acquisition system used for long-term monitoring. The subject is lying on a normal bed, while a pressure sensitive sensor is placed under the mattress. BCG recorded with systems of this type often contain a strong respiratory component.

possibility for unobtrusive long-term BCG acquisition. This reflects the shift in focus from the classical approach to BCG with calibrated recording devices for diagnosis towards the modern approach with uncalibrated devices designed for unobtrusive monitoring [47]. Figure 3.1 shows a schematic illustration of a bed-mounted uncalibrated BCG acquisition system, typically used for long-term home monitoring.

Another important differences between classical and modern BCG devices lies in the morphology of the observed signal. Similar to ECG, one can observe in BCG recorded with classical devices a typical signal morphology with groups of characteristic peaks. Although slight discrepancies exist between different types of BCG tables, the morphology is consistent within each type of BCG tables [47] and the peaks in the BCG waveform can be linked to certain physiological events in the cardiac cycle, e.g. the ejection of blood from the ventricle [46, 47].

However, as a result of the unobtrusive measurement concept, BCG recorded with modern devices have no predictable signal morphology. This is because the sensor is embedded into different objects, e.g. BCG recorded from chair-mounted sensors differ significantly from those recorded using bed-mounted sensors. In addition, the subject is free to move during BCG acquisition, which introduces motion artefacts and which also means that the subject's posture and position relative to the sensor may change considerably during a recording. This makes it difficult to link peaks observed in the BCG to events in the cardiac cycle. It also has serious implications for modelling the BCG, and in section 3.5 we present two approaches to this problem. The first one is based on a hand-optimized custom template and the second approach is based on a non-parametric model which can adapt itself to the signal.

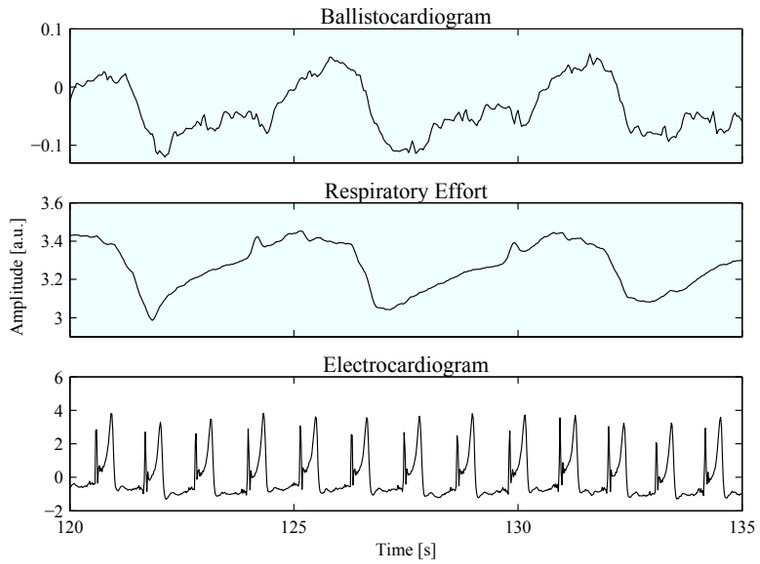


Figure 3.2: A section from a BCG recorded with a bed mounted sensor (top) with gold standard reference for respiration (middle) and heartbeat (bottom). Note that the shape of the BCG is dominated by the respiratory component, while the cardiac component is visible as low amplitude ripples superimposed on the respiratory component and in synchrony with the ECG reference.

As mentioned, movements of the subject can cause artefacts in the signal. Artefacts caused by irregular motion like limb movement are typically characterized by short duration and a large amplitude, often causing clipping of the signal. Affected segments have to be rejected before analysis of the signal. On the other hand, respiratory movements are ubiquitous but regular. They can be considered as a component of the BCG which have to be filtered out before analysis. However, since the respiratory component contains physiological information on its own, there is considerable interest in separating it from the cardiac component of the BCG for further analysis. In section 3.2, we will use this problem to demonstrate the application of the LPNR-based source separation.

Figure 3.2 shows a section from a BCG recorded with a modern bed-mounted system. As a reference, simultaneous recordings of ECG and respiratory effort via respiration belt are provided. The BCG in this sample has a strong respiratory component with an amplitude much higher than that of the cardiac component. This is typical for BCG acquired with bed-mounted

sensors and it means that the BCG often shows a strong resemblance to the respiratory reference. The cardiac BCG component in this sample is visible as a series of small deflections superposed on the respiratory component and appearing to be in synchrony with the ECG reference. As we will see in the next section, these two components of the BCG are characterized by overlapping spectra, which makes them difficult to separate with linear filters. However, using nonlinear methods this problem can be solved.

3.2 Nonlinear Noise Reduction

In order to understand the nonlinear signal separation scheme, which we present in this chapter, we need to first introduce a nonlinear noise reduction method called locally projective noise reduction (LPNR). Thus, we dedicate this section to an introduction of the LPNR algorithm and a discussion of the connections between LPNR and the topic of mixture modelling, which will play an important role later in chapter 4.

In the following, we use bold font to denote vectors and subscripts to denote samples in a time series. For instance, in a scalar time series x, we denote the i-th sample with x_i and $\boldsymbol{x}_j$ denotes the j-th delay vector in the trajectory obtained by embedding [55, 56] x:

$$\boldsymbol{x}_j = (x_j, x_{j+1}, \ldots, x_{j+M})^T. \tag{3.1}$$

Whenever possible, we use the corresponding capital letter to denote limits of subscripts. In the above example, i runs from 1 to I and j runs from 1 to J. In addition, we use square brackets to index elements of vectors and matrices: $\boldsymbol{a}[i]$ is the i-th element of vector $\boldsymbol{a}$ and $A[i, j]$ is the element in row i and column j of the matrix A.

3.2.1 Locally Projective Noise Reduction

LPNR was first proposed in [57] as an algorithm to denoise chaotic time series, which often have a broad spectrum with non-negligible components at high frequencies. This poses problems for denoising, since the broad spectrum of these signals often overlap with the spectrum of the noise, making it difficult to reduce noise with low-pass filters without damaging the signal itself. Hence, the authors of [57] presented a noise reduction algorithm which did not rely on representing the signal in frequency domain. Instead, LPNR embeds the signal into delay space and takes advantage of the geometrical properties of deterministic time series in delay space [55, 56].

Derivation of LPNR

Early works on projective noise reduction methods include [58, 59, 60] and [61]. The name locally projective noise reduction goes back to Kantz and Schreiber who published a textbook which contains a detailed description of the LPNR algorithm [14], and lists several interesting examples for its application to denoising chaotic signals, but also for application to problems outside the field of nonlinear dynamics. These applications include ECG denoising [62, 63], EEG denoising for the analysis of event related potentials [64, 65] or denoising speech signals [66, 67].

Furthermore, these applications demonstrate that LPNR, which was originally developed for chaotic signals, can also be used to denoise stochastic signals. The prerequisite is that the trajectory obtained by embedding the signal must lie on or near a low dimensional attractor in delay space [14]. For an M_0-dimensional deterministic system, Takens *et al.* showed that even if the systems is only observed through a scalar time series, one can reconstruct an attractor equivalent to the attractor characterizing the original system, by delay embedding the scalar time series in $M \geq 2M_0 + 1$ dimensions [55]. However, Kantz *et al.* pointed out that even for signals which contain stochastic components, like ECG, the delay space trajectory often still lies near a low dimensional attractor, which justifies the application of LPNR [14].

The geometrical meaning of this is illustrated in figure 3.3, where the left panel shows a noisy time series embedded into two-dimensional delay space. However, the delay vectors approximately follow a one-dimensional trajectory, which is blurred due to the noise. In the right panel, the same signal is shown after noise reduction with LPNR and the one-dimensional nature of the trajectory is more clearly visible.

This already brings us to the basic idea of LPNR: If we know (or assume) that the trajectory obtained by embedding the noiseless version of a signal into a high-dimensional delay space lies on or near a low dimensional attractor, then the deviation of the actual delay vectors from the low-dimensional attractor must be due to noise. We can attempt to reduce the noise by estimating the position of the noise free trajectory and moving the delay vectors towards that position. In [57], Grassberger *et al.* proposed a method for estimating a local approximation to the noise free trajectory, which will be described below.

The LPNR algorithms assumes that the delay vector obtained by embedding the noiseless input time series into M-dimensional delay space will follow a trajectory which lies on an attractor with only M_0-dimensions and that

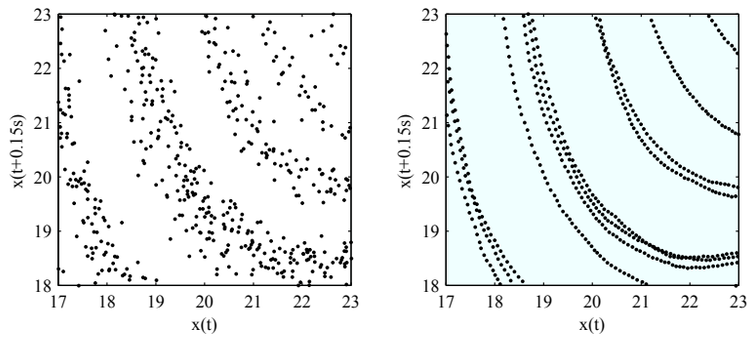


Figure 3.3: A scalar time series is embedded into two-dimensional delay space. The delay vectors follow a one-dimensional trajectory, which is heavily blurred by noise in the original time series (left). After noise reduction with LPNR the one-dimensional trajectory is clearly visible (right).

M_0 is known. This means that, when making a first order approximation to the attractor at a certain point $\boldsymbol{x}_0$ on the trajectory, the delay vectors surrounding $\boldsymbol{x}_0$ will occupy only an M_0-dimensional subspace of the M-dimensional space they are embedded in. This M_0-dimensional subspace will be spanned by M_0 vectors $\boldsymbol{t}_1$ to $\boldsymbol{t}_{M_0}$ which are tangent to the attractor.

On the other hand, there will be a space of $Q = M - M_0$ dimensions into which the nearby delay vectors do not extend. This space is called the nullspace [14] and it is spanned by Q vectors $\boldsymbol{a}_1$ to $\boldsymbol{a}_Q$, which fulfil the following relationship:

$$\boldsymbol{a}_q(\boldsymbol{x}_k - \boldsymbol{x}_0) = 0, \quad q = 1, \ldots, Q. \tag{3.2}$$

Here, $\boldsymbol{x}_0$ is the point on the attractor around which we like to linearize the attractor, similar to the expansion point of a Taylor series expansion.

We also assume that there is a set of delay vectors $X = \{\boldsymbol{x}_k,\ k = 1, \ldots, K\}$ containing all delay vectors $\boldsymbol{x}_k$ which are closer to $\boldsymbol{x}_0$ than a certain threshold ε. This threshold has to be small enough so that the curvature of the attractor is negligible within a distance of ε around $\boldsymbol{x}_0$. We will discuss the meaning of ε in a moment. Note that here, the index k is used to identify the delay vectors in the set X and is not related to the order of samples in the time series x.

In geometrical terms, equation (3.2) means that the $\boldsymbol{a}_q$ are perpendicular to the attractor. However, the actual time series will contain measurement noise: $y_i = x_i + \eta_i$. Here, we assume that the noise process η has zero mean

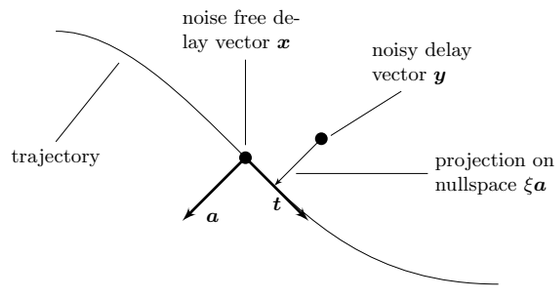


Figure 3.4: Schematic illustration of the basic principle of LPNR for an one-dimensional trajectory in two-dimensional delay space. $\boldsymbol{t}$ is the tangent and $\boldsymbol{a}$ spans the nullspace.

but is not necessarily white. In this case, the above relationship is perturbed by a noise term:

$$\boldsymbol{a}_q(\boldsymbol{y}_k - \boldsymbol{x}_0) = \xi_{kq} \neq 0, \quad q = 1, \ldots, Q. \tag{3.3}$$

The simple idea of projective noise reduction is to obtain estimates for ξ_{kq} and $\boldsymbol{a}_q$, and correct the noisy delay vector by subtracting $\xi_{kq} \cdot \boldsymbol{a}_q$ from $\boldsymbol{y}_k$, which is equivalent to projecting $\boldsymbol{y}_k$ onto a first order approximation of the noise free attractor. In figure 3.4, this is illustrated schematically for an example in two-dimensional delay space.

In order to obtain the estimates for $\boldsymbol{a}_q$, Grassberger *et al.* made the assumption that the values of ξ_{kq} are small compared to $\boldsymbol{t}_{m_0}(\boldsymbol{y}_k - \boldsymbol{x}_0)$. Geometrically, this means that the trajectory is dominated by the shape of the attractor and the noise is only causing a small perturbation ξ_{kq} away from the attractor. Without loss of generality, the $\boldsymbol{a}_q$ can also assumed to be normalized and orthogonal to each other. Thus, the task is to find Q vectors which fulfil equation (3.2), as well as the relationship:

$$\|\boldsymbol{a}_q\| = 1 \qquad \text{and} \tag{3.4}$$

$$\boldsymbol{a}_p^T \boldsymbol{a}_q = \begin{cases} 0 & \text{if } p \neq q \\ 1 & \text{if } p = q \end{cases} \tag{3.5}$$

for $p, q = 1, \ldots, Q$ and at the same time minimize the average perturbation in the neighbourhood [14]:

$$L = \sum_{k=1}^{K} \sum_{q=1}^{Q} \xi_{kq}^2. \tag{3.6}$$

Grassberger *et al.* solved this task using Lagrange multiplier with the result that the $\boldsymbol{a}_q$ are the eigenvectors corresponding to the Q smallest eigenvalues

of the (empirical) covariance matrix of the delay vectors $\boldsymbol{y}_1$ to $\boldsymbol{y}_K$. The details of the Lagrange multiplier approach from [57] are summarized in appendix A.1. In practice, one would obtain the eigenvectors of the covariance matrix by applying principal components analysis (PCA) to the delay vectors in the set $Y = \{\boldsymbol{y}_k,\ k = 1, \ldots, K\}$.

With this, we have the ingredients of the algorithm, allowing us to give a summary of all steps needed for LPNR. We start with the noisy time series $y = x + \eta$ which needs to be denoised. The first step is to choose an embedding dimension M, which is larger than the known or assumed dimension of the attractor M_0, and to apply M-dimensional delay embedding to the time series y. M is one of the parameters of LPNR and we will discuss its meaning in the subsequent paragraphs. The embedding will give rise to a delay vectors series, denoted $\boldsymbol{y}$.

Since projective noise reduction has to be applied to each delay vector separately, the following steps are repeated for all delay vectors. For a particular delay vector $\boldsymbol{y}_i$ in the series, the second step consists in choosing a value for ε and finding all other delay vectors which are closer to $\boldsymbol{y}_i$ than ε. The set of these neighbouring delay vectors is denoted by $Y_i = \{\boldsymbol{y}_k,\ k = 1, \ldots, K\}$ and its elements satisfy $||\boldsymbol{y}_k - \boldsymbol{y}_i|| \leq \varepsilon$.

Given this neighbourhood of $\boldsymbol{y}_i$, we have to estimate the linear approximation to the nullspace $\boldsymbol{a}_q$, as well as a point $\boldsymbol{x}_0$ which lies on the attractor and is close to $\boldsymbol{y}_i$. Because we assume that the noise process η has zero mean, the estimate of $\boldsymbol{x}_0$ is simply given by the mean of all delay vectors in Y_i. To estimate $\boldsymbol{a}_q$, we first calculate the empirical covariance matrix C of the $\boldsymbol{y}_k$:

$$C = \frac{1}{K} \sum_{k=1}^{K} (\boldsymbol{y}_k - \boldsymbol{x}_0)(\boldsymbol{y}_k - \boldsymbol{x}_0)^T. \tag{3.7}$$

Then, we perform a PCA on C and set $\boldsymbol{a}_q$ equal to the eigenvectors corresponding to the Q smallest eigenvalues of C.

The next step is to calculate the component in $\boldsymbol{y}_i$ which is due to noise and subtract it from $\boldsymbol{y}_i$. This noise component $\boldsymbol{\xi}_i$ is a vector consisting of Q contributions, one from each $\boldsymbol{a}_q$, which are given in equation (3.3). Adding these components gives us:

$$\boldsymbol{\xi}_i = \sum_{q=1}^{Q} \boldsymbol{a}_q \boldsymbol{a}_q^T (\boldsymbol{y}_k - \boldsymbol{x}_0). \tag{3.8}$$

Thus, the corrected delay vector is given by $\tilde{\boldsymbol{y}}_i = \boldsymbol{y}_i - \boldsymbol{\xi}_i$.

Now, we obtained a noise reduced version of the delay vector series, which we call $\tilde{\boldsymbol{y}}$. However, what we need is the noise reduced version of the scalar

Algorithm 1 Pseudo code for locally projective noise reduction

function LPNR(y, ε, M, Q)
  $\boldsymbol{y} \leftarrow \text{delayEmbedding}(y, M)$
  for $i \leftarrow 1, \ldots, I$ **do**
    $Y_i \leftarrow \{||\boldsymbol{y}_k - \boldsymbol{y}_i|| \leq \varepsilon\}$
    $\boldsymbol{x}_0 \leftarrow \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{y}_k$
    $C \leftarrow \frac{1}{K} \sum_{k=1}^{K} (\boldsymbol{y}_k - \boldsymbol{x}_0)(\boldsymbol{y}_k - \boldsymbol{x}_0)^T$
    $V \leftarrow \text{pca}(C)$ ▷ columns of V contain eigenvectors
    $V \leftarrow \text{sort}(V)$ ▷ sort in ascending order of eigenvalues
    for $q \leftarrow 1, \ldots, Q$ **do**
      $a_q \leftarrow V[:, q]$
    $\boldsymbol{\xi}_i \leftarrow \sum_{q=1}^{Q} \boldsymbol{a}_q \boldsymbol{a}_q^T (\boldsymbol{y}_k - \boldsymbol{x}_0)$
    $\tilde{\boldsymbol{y}}_i = \boldsymbol{y}_i - \boldsymbol{\xi}_i$
    $\tilde{y}_i \leftarrow \frac{1}{M} \sum_{m=1}^{M} \tilde{\boldsymbol{y}}_{i-m+1}[m]$
  return $\tilde{y}$

time series y. When transforming $\tilde{\boldsymbol{y}}$ back into a scalar signal, we encounter the problem that each sample y_i in the original time series is part of M consecutive delay vectors:

$$\begin{aligned} \boldsymbol{y}_{i-M+1} &= (y_{i-M+1}, \ldots, y_i)^T \\ &\vdots \\ \boldsymbol{y}_i &= (y_i, \ldots, y_{i+M-1})^T, \end{aligned}$$

each receiving an independent correction. In order to obtain a consistent estimate for the noise reduced scalar time series, we average over the contributions from each of the M corrected delay vectors:

$$\tilde{y}_i = \frac{1}{M} \sum_{m=1}^{M} \tilde{\boldsymbol{y}}_{i-m+1}[m]. \tag{3.9}$$

The steps given above provide a comprehensive description of the LPNR. They are visualized in a flow chart in figure 3.5 and additionally, a pseudo code listing is provided in algorithm 1.

To enhance the geometrical interpretation of LPNR, figure 3.6 illustrates how the LPNR steps look like in two-dimensional delay space. The four

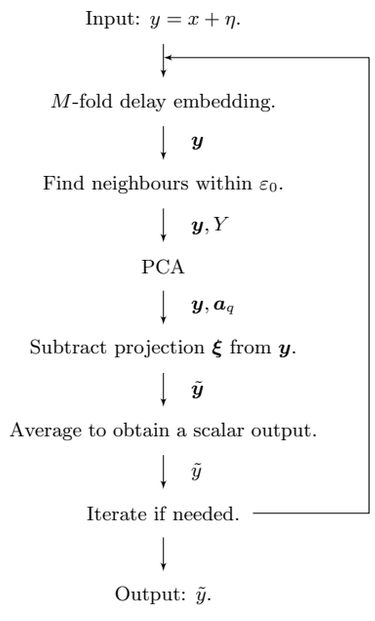


Figure 3.5: Flowchart of the locally projective noise reduction algorithm. The noise process η does not have to be white, but is assumed to have zero mean. The symbols on the right side of each arrow denote the variables which are passed between the respective steps.

panels illustrate the key steps of finding the local linear approximation via PCA and projecting towards the attractor, which have to be repeated for every delay vector.

Parameters of LPNR

The LPNR algorithm has several key parameters, which are discussed in the following. We begin with the threshold for the neighbour search ε, which determines the range of the local linear approximation. As stated earlier, the value of ε has to be small enough such that second order effects are negligible. This can be seen from figure 3.6: If the radius for neighbour search were e.g. twice as large, the curvature of the trajectory would influence the PCA and the eigenvectors would not be aligned tangential and normal to the attractor any more. In addition, the mean of the neighbouring delay vectors would not be on the attractor any more. Instead, $\boldsymbol{x}_0$ would be perturbed away from the attractor in the direction of the curvature.

Figure 3.6: Illustration of the steps in LPNR for one delay vector in the trajectory of the time series embedded in two-dimensional delay space. Top left: position of the target delay vector in the noisy trajectory. Top right: neighbour search. Bottom left: PCA, the large axis of the ellipse corresponds to the first eigenvector, while the second eigenvector is not shown. Bottom right: applying the correction to the target delay vector.

On the other hand, ε should also not be too small. This can also be explained in an intuitive way using figure 3.6. Imagine the value of ε being only half as large as shown in the second panel. Then, the neighbourhood would not range beyond the noise level, but form a circular cloud of delay vectors instead. Thus the covariance matrix C would have eigenvalues of almost equal size, which cannot be used to distinguish between tangent and normal directions.

The above considerations show that when selecting a value for ε, one has to strike a compromise between limiting second order effects and noise reduction. This also illustrates the limits of LPNR: The distance on the attractor at which second order effects become non-negligible is an upper limit on the

amplitude of the noise that can be removed with LPNR without distorting the original signal. In practice, one would first visually inspect the signal and choose ε to be slightly larger than the peak to peak amplitude of the noise and after noise reduction check the output signal for distortions [14]. The last step is important, since distortion of the signal has been shown to be among the consequences of careless application of projective filtering techniques [68].

The parameters M and Q or equivalently M and M_0 are closely related. M is the embedding dimension and must be higher than M_0. Embedding is a complex topic by itself and for time series generated by deterministic systems there are theoretic results stating that even if we only observe a scalar time series and the attractor characterizing the original system has M_0 dimensions, the attractor can be reconstructed by delay embedding the scalar time series in $M \geq 2M_0 + 1$ dimensions [55, 56]. For example, the famous Lorenz system [69] is characterized by an attractor with a fractal dimension between 2 and 3. Consequently, to denoise a noisy Lorenz time series, we should choose $M \geq 7$ and $Q = M - 3$.

However, in the case of stochastic signals encountered in practical applications like ECG, it is difficult to define the value of M_0, since it is unclear if there even exists an attractor underlying the generation of these signals. Although, limit cycle based ECG models [70] have been used with great success for ECG processing [71, 72]. This would suggest that, at least for ECG signals, using limit cycle attractors seem to be a good way to model the signal.

For the practical application of LPNR to denoising experimental time series which are not strictly deterministic, the recommendation is to set M equal to the number of samples which corresponds to the duration of deflections caused by the noise process [14, 45]. Thus, the role of M is complementary to that of ε: While ε describes the amplitude scale of the noise process, M describes its time scale.

After choosing M, Q is set to adjust the strength of filtering. The larger we choose Q, the lower the dimension of the attractor we restrict the time series to be on. E.g. for ECG, good results were obtained by setting $Q = M - 1$ [62]. As with the case of ε, it is important to inspect the signal after noise reduction and adjust the parameters if necessary.

The choice of parameters for LPNR requires manual effort, which is one of the disadvantages of LPNR. However, the unusual behaviour of LPNR with respect to its parameters offers an alternative approach to noise reduction compared to linear filtering. While linear filters distinguish between noise and signal by spectral components in frequency domain, LPNR stays in

time domain and distinguishes signal from noise via the typical amplitude scale and time scale of the noise component. This means that even when the spectrum of signal and noise overlap in frequency domain, which would pose a problem for linear filters, LPNR can still supress the noise, if the typical amplitude scale and time scale of the noise component differ from those of the signal. However, we would like to point out that this does not mean LPNR is superior to linear filtering. Rather, the two methods are complementary. We will see an example for this in the next section.

A Toy Example

Identifying a signal component via the amplitude scale and time scale is an unusual concept, especially in the classical signal processing domain, which has been dominated by the success of linear filtering in frequency domain. Therefore, we provide an illustrative example, in order to enhance the understanding of this concept and the behaviour of LPNR.

The top left panel in figure 3.7 shows a series of Gaussian shaped peaks with two distinct amplitudes. The peaks are shaped after Gaussian bell curves, which are described by the following equation:

$$f(t) = a \exp\Big(-\frac{t^2}{2\tau^2}\Big). \tag{3.10}$$

a determines the amplitude of the peak, which, in our example, can take on two distinct values and τ is the width of the peak, which is constant for all peaks in the top left panel of figure 3.7. The Fourier transform of equation (3.10) is given by [73]:

$$F(\omega) = \sqrt{2\pi}\tau\, a \exp\Big(-\frac{\tau^2\omega^2}{2}\Big), \tag{3.11}$$

which shows that the width of the spectrum (also called bandwidth) is determined by the inverse width of the bell curve in time domain. This is a consequence of a fundamental principle of the Fourier transform, which states that the shorter a signal is in time domain, the broader its spectrum is in frequency domain [73].

However, the important implication for our example is that the spectrum of all peaks have the same extend in frequency domain, since τ is constant for all peaks. Therefore, if one would apply a linear filter to the signal in the top left panel of figure 3.7, all peaks would be affected in the same way, since linear filters, targeting a certain range in frequency domain, cannot distinguish between signals with the same spectrum. This can be seen in the bottom panel of figure 3.7. In addition, the FIR filtering introduces a baseline oscillation that was not present in the original signal.

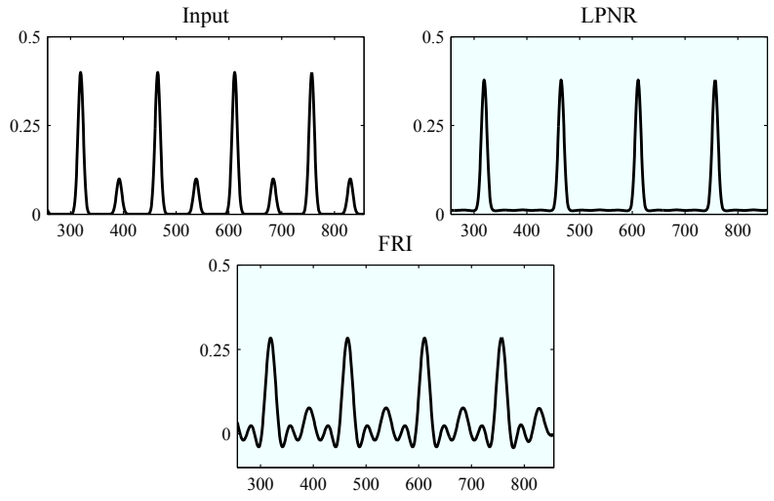


Figure 3.7: Demonstrating the properties of LPNR. A mixture of Gaussians shaped peaks with different amplitudes but identical variance (top left) is filtered using LPNR (top right) and FIR filters (bottom).

On the other hand, the top right panel in figure 3.7 shows that LPNR removed all of the small peaks without affecting the large peaks much. The reason is that we have chosen for ε a value which is slightly larger than the amplitude of the small peaks but much smaller than the amplitude of the large peaks. This tells the LPNR algorithm to remove peaks smaller than ε, but leave peaks larger than ε untouched. In contrast to linear filters, the fact that both small and large peaks have the same spectrum does not prevent LPNR from successfully targeting the smaller peaks, as the difference in amplitude is sufficient to distinguish them from the large peaks.

Inspecting figure 3.7 very carefully, one will notice that the baseline in the right panel is slightly elevated. This elevation of the baseline will grow with increasing value of ε and is due to the distortion of the attractor mentioned at the beginning of this section.

3.3 Connection to Mixture Modelling

An interesting observation is that the noise reduction method, which is a core component of the source separation scheme introduced in the next section, is based on local application of PCA, a dimensionality reduction method.

Thus, dimensionality reduction is a key component of our source separation scheme.

However, when looking at LPNR on a higher level, it will become clear that although LPNR has its roots in nonlinear dynamics, where it was developed as a noise reduction method, there are close connections to other methods belonging to entirely different fields. Thus, we will dedicate this section to shedding light on the relationship between LPNR and mixture modelling, which is a key topic of chapter 4.

We will begin by discussing LPNR in the context of the parametric versus non-parametric categorization introduced in section 2.5.2, which leads us to a version of LPNR optimized for online execution. Given this point of view on LPNR, we will be able to realize the close relationship between LPNR and mixture modelling. In fact, we will see that LPNR and the mixture of principal components analysers model can be interpreted as subcategories of the same model family.

3.3.1 LPNR as a non-parametric method

The application of PCA to the neighbours of each delay vector is what makes LPNR a non-parametric method. This can be seen by comparing LPNR to the k-nearest neighbours algorithm, which was introduced in section 2.5.2 as an example of a non-parametric method, and observing the parallels between these two methods.

First, the parameters discussed in the previous section, ε, M and Q, should actually be referred to as hyper-parameters of LPNR. In this context, the parameters of LPNR are the Q eigenvectors $\boldsymbol{a}_q$ calculated for each delay vector, the number of which grows with the size of the input. This is the key property of a non-parametric method.

Second, for each input, LPNR is extracting the local information contained in the data to obtain the result, just as kNN. This way the, the algorithm is not limited by an underlying model with finite degree of freedom. Instead, the information about the global structure of the problem, is retained by storing the entire dataset, while at the same time, the result for each single point is influenced by detailed information of the local structure. For kNN, the local approximation consist in finding the k closest neighbours and for LPNR it consist in finding the neighbours within a range of ε.

However, LPNR also suffers from the disadvantages of non-parametric methods described in section 2.5.2. Just like kNN, LPNR has to search the entire dataset when processing each input sample. Not only is this time consuming,

but it also introduces a dependence of the processing time on the possibly varying size of the dataset, which is undesirable for an algorithm that has to run under real-time constraints. Therefore, a version of LPNR optimized for online execution has been developed [74].

3.3.2 An online version of LPNR

The high computational complexity of LPNR and the fact that in its original formulation LPNR is non-causal, meaning that it needs the entire input time series before it can start processing, was the motivation for developing an online version of LPNR [74]. One of the key modifications proposed in [74] is to store the eigenvectors $\boldsymbol{t}_{m_0}$ and $\boldsymbol{a}_q$, as well as the mean delay vectors $\boldsymbol{x}_0$, for certain representative points distributed over the attractor. For each delay vector, online LPNR first checks if there is a representative point nearby. If so, the stored eigenvectors are used to make the correction, otherwise the neighbour search and PCA are performed for the current delay vector, which is then included into the list of representative points. By limiting the number of representative points, the LPNR algorithm has been effectively turned into a parametric method.

The set of eigenvectors $\boldsymbol{t}_{m_0}$ and $\boldsymbol{a}_q$ associated with each of the representative points form a model of the attractor. The information on the attractor, which in the original LPNR was implicitly contained in the entirety of the dataset, is now contained explicitly in a parametric model. This model describes the attractor by covering it with locally computed eigenvectors, which contain the information on the local tangential and normal directions of the attractor. Such a model has already been used in [75] for learning attractors. It has also been proposed independently in the machine learning and image processing communities, where it is referred to as a manifold learning technique and used for a variety of applications including handwritten digit recognition [76, 77], dimensionality reduction [78], image coding [79] and image interpolation [80].

However, all methods based on modelling the attractor using a set of local approximations have to face the assignment problem, which deals with the question of which local approximation to assign each delay vector or data point to. Assignments can be based on the Euclidean distance measure, but also the reconstruction error can be used as a criterion [81]. When making hard assignments, i.e. assigning each delay vector to a single approximation, the chosen assignments define a partitioning of the attractor. This can lead to problems for certain applications, since e.g. the criterion used for finding the optimum assignment becomes non-differentiable with hard assignments. Also, the treatment for delay vectors at the boundary between

two approximations is non-optimal [81].

To address these problems, Tipping *et al.* proposed a probabilistic version of PCA [81]. Using their probabilistic interpretation of PCA, the attractor model can be formulated as a mixture model, where data points are assigned based on the probability that they were generated by a certain component. These soft assignments allow the model to assign each data point to more than one component, eliminating many problems caused by hard assignments. In the following, we provide a short description of the probabilistic principal components analysis (PPCA) model introduced in [81] and discuss the parallels between the mixture of PPCA model and the LPNR algorithm.

3.3.3 Probabilistic Principal Components Analysis

The PPCA model can be viewed as a latent variable model with a continuous latent variable [17, 81]. The generative process is as follows [81]: First, a random vector $\boldsymbol{z}$ is drawn from an isotropic Gaussian distribution

$$p(\boldsymbol{z}) = \frac{1}{(2\pi)^{d/2}} \exp\Big(-\frac{\boldsymbol{z}^T\boldsymbol{z}}{2}\Big), \tag{3.12}$$

with d being the dimension of $\boldsymbol{z}$. However, $\boldsymbol{z}$ is latent, which means it is not observed. Instead, what we observe is the vector $\boldsymbol{y}$ given by:

$$\boldsymbol{y} = W\boldsymbol{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}. \tag{3.13}$$

Here, $\boldsymbol{\epsilon}$ is a Gaussian random vector with variance σ modelling the measurement noise: $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma I)$. Thus, $\boldsymbol{z}$ is first transformed by multiplying it with the $e \times d$ matrix W and adding the mean $\boldsymbol{\mu}$, as well as measurement noise, before we observe it. The conditional probability of $\boldsymbol{y}$ given $\boldsymbol{z}$ is:

$$p(\boldsymbol{y}|\boldsymbol{z}) = \frac{1}{(2\pi\sigma^2)^{e/2}} \exp\Big(-\frac{||\boldsymbol{y} - W\boldsymbol{z} + \boldsymbol{\mu}||^2}{2\sigma^2}\Big), \tag{3.14}$$

and by marginalizing out $\boldsymbol{z}$, we obtain the marginal distribution over $\boldsymbol{y}$:

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{z}) &= \int p(\boldsymbol{y}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \\ &= \frac{1}{|2\pi C|^{1/2}} \exp\Big(-\frac{1}{2}(\boldsymbol{y} - \boldsymbol{\mu})^T C(\boldsymbol{y} - \boldsymbol{\mu})\Big), \end{aligned} \tag{3.15}$$

with $C = \sigma^2 I + WW^T$.

After observing a number of data points $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N\}$, one way to estimate the parameters W, $\boldsymbol{\mu}$ and σ is by maximizing the log-likelihood function:

$$\mathcal{L}(W, \boldsymbol{\mu}) = \sum_{n=1}^{N} \log(p(\boldsymbol{y}_n)), \tag{3.16}$$

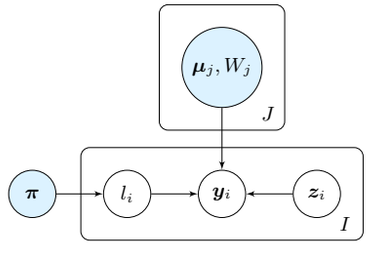


Figure 3.8: Graphical model for the mixture of probabilistic principal components analysers model. Note the similarity to the graphical model of the mixtures of Gaussians model in figure 4.6.

with respect to W and $\boldsymbol{\mu}$. Here, $p(\boldsymbol{y}_n)$ denotes the function defined in equation (3.15). Tipping *et al.* derived two solutions to this maximum likelihood estimator, which are both explained in detail in [81]. The first is a closed-form solution, while the second one is an iterative solution based on a method called the expectation maximization (EM) algorithm.

3.3.4 Mixture of PPCA and LPNR

Although the PPCA model introduced in the previous section extends the classical PCA to the domain of probabilistic models, it is still a linear model. In order to model nonlinear systems, Tipping *et al.* introduced the mixture of PPCA model [81], which approximates a nonlinear manifold using a combination of local linear PPCA models. This is accomplished by adding a discrete latent variable l to the model, which indicates which component is responsible for generating the current observation.

The complete mixture of PPCA model can be described as follows. There are J PPCA components, each with its own set of parameters W_j, $\boldsymbol{\mu}_j$ and σ_j ($j = 1, \ldots, J$). As with the simple PPCA model, $\boldsymbol{z}$ is drawn from an isotropic Gaussian distribution. The additional random variable l, drawn from a categorical distribution (see appendix A.3), determines which component is used to transform $\boldsymbol{z}$:

$$\boldsymbol{y} = W_l \boldsymbol{z} + \boldsymbol{\mu}_l + \epsilon_l. \tag{3.17}$$

Figure 3.8 shows the graphical model of the mixture of PPCA with J components and I observations $\boldsymbol{y}_1$ to $\boldsymbol{y}_I$. There is one latent vector $\boldsymbol{z}_i$, as well as one l_i for each observation $\boldsymbol{y}_i$. The vector $\boldsymbol{\pi}$ contains the weights of the PPCA components, which correspond to the probabilities that the component is chosen by l. Also note the similarity between this model and the mixture of Gaussians model, which will be introduced in chapter 4.6.

In a practical setting, one would receive the I observations, which could e.g. correspond to the series of delay vectors obtained from embedding a scalar time series. These observations will be used to estimate the set of parameters of the mixture of PPCA model $\boldsymbol{\pi}$, W_j, $\boldsymbol{\mu}_j$ and σ_j $(j = 1, \ldots, J)$ with the iterative EM algorithm proposed in [81]. The EM algorithm will also provide soft estimates for the cluster labels l_i, which for each cluster approximate the probability that the observation $\boldsymbol{y}_i$ has been generated by that cluster. These results can then be used to estimate a noise reduced version of the observation.

Comparing the mixture of PPCA model to LPNR, one can draw several parallels. The delay vectors in LPNR correspond to the observation $\boldsymbol{y}$ in the mixture of PPCA model. The noise free delay vectors $\boldsymbol{x}$ in LPNR would correspond to the vector $W\boldsymbol{z}+\boldsymbol{\mu}$, which is the vector obtained by transforming $\boldsymbol{z}$ but without adding noise. However, the vector $\boldsymbol{z}$ itself has no counterpart in LPNR. The noise reduction step in LPNR can be reproduced in the mixture of PPCA model by projecting $\boldsymbol{y}_i - \boldsymbol{\mu}_j$ onto the subspace spanned by W_j for all clusters from 1 to J and averaging the projections according to the probability that $\boldsymbol{y}_i$ belongs to cluster number j. This approach has the advantage that vectors at the boundary between two clusters are not forced to one of the two clusters, which can lead to suboptimal results.

In this section, we have shown that LPNR is a non-parametric method, while online LPNR and related methods based on mixtures of regular PCA models can be viewed as a parametric variant of LPNR. We have also demonstrated how closely signal processing is related to modelling by drawing a connection between manifold learning and LPNR, which was designed without having mixture models in mind. Furthermore, the mixture of PPCA model has been established as a parametric and probabilistic version of LPNR. The relationship between all these methods are illustrated in table 3.1. An interesting topic for further research is the question of how to combine the advantage of the non-parametric LPNR with the probabilistic framework provided by the mixture of PPCA model. Such a model would occupy the lower right corner of table 3.1.

3.4 Nonlinear Source Separation

In this section, we will introduce a scheme for signal separation based on LPNR. This separation scheme inherits some of the properties of LPNR, which enables it to separate signal components with overlapping spectra if certain conditions are met. The effectiveness of this signal separation scheme is demonstrated on the problem of separating cardiac and respiratory BCG

	parametric	**non-parametric**
non-probabilistic	online LPNR mixture of regular PCA	LPNR
probabilistic	mixture of PPCA	open research topic

Table 3.1: Categorization of LPNR and related methods.

components.

LPNR-based source separation has been first developed for extracting the fetal ECG component from ECG signals recorded by placing the electrodes on the abdomen of the mother [14, 45, 82]. These abdominal ECG recordings are dominated by the maternal ECG component, while the fetal ECG component is faster and has a much smaller amplitude. In addition, the components have a very similar spectrum, since they are both ECG signals, which is the motivation for using the LPNR-based signal separation.

The problem of overlapping spectra between different signal components is shared by the BCG, where respiratory and cardiac components have partly overlapping spectra. This is due to subharmonics in the cardiac component, but also because the respiratory component has a broadband spectrum. Although, the base frequency of respiration is normally below the heart rate, the waveform of the respiratory component often contains sharp flanks, e.g. in the case of ragged breathing. This broadens the respiratory spectrum, causing an overlap with the spectrum of the cardiac component. Therefore, signal separation with traditional methods like linear filtering does not work well for BCG. For this reason, we have adapted the LPNR-based signal separation scheme for the separation of BCG components [83, 84].

The key idea of LPNR-based signal separation is to apply LPNR iteratively with varying parameters. Figure 3.9 shows the flow diagram of the nonlinear signal separation process for BCG components. The raw BCG first enters the left branch in figure 3.9, where it is low-pass filtered with a cutoff frequency of 3.5 Hz to remove the high frequency noise, as well as the high frequency components of the cardiac component. The output of the low-pass filter contains the respiratory component and the low frequency parts of the cardiac component, which have overlapping spectra in frequency domain.

However, inspecting the signal in time domain reveals that the cardiac component is faster and has a smaller amplitude than the respiratory component. Therefore, we apply LPNR to the output of the low-pass filter in order to re-

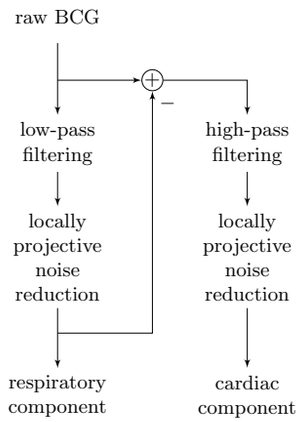


Figure 3.9: Flow diagram of the separation algorithm. The raw signal is first filtered to yield the respiratory component, which is then subtracted from the raw signal. This difference is filtered again to give the cleaned cardiac component.

move the remaining cardiac component from the signal. To achieve this, the parameters are chosen such that ε is slightly larger than the peak-to-peak amplitude of the cardiac component, while M corresponds to the number of samples which cover the length of the typical deflections in the cardiac component. Choosing the parameters this way causes LPNR to treat the cardiac component as noise. Thus, the final output of the left branch is the clean respiratory BCG component.

Before the raw BCG enters the right branch in figure 3.9, the respiratory component is removed by subtracting the output of the left branch. Therefore, the right branch receives a noisy version of the cardiac component as input, which is first high-pass filtered with a cutoff frequency of 0.6 Hz. The high-pass filter serves to remove any residual respiratory influence remaining in the signal. After high-pass filtering, the signal is filtered again with LPNR. This time however, the parameters are chosen such that ε is slightly larger than the peak-to-peak amplitude of the noise process, but smaller than the amplitude of the cardiac component. M is chosen to cover the length of typical deflections caused by noise. Thus, the operations in the right branch in figure 3.9 remove only the noise and output a clean cardiac component.

The overall strategy of this scheme is to use fast linear filtering to separate the non-overlapping parts of the spectrum and to assist the slower LPNR,

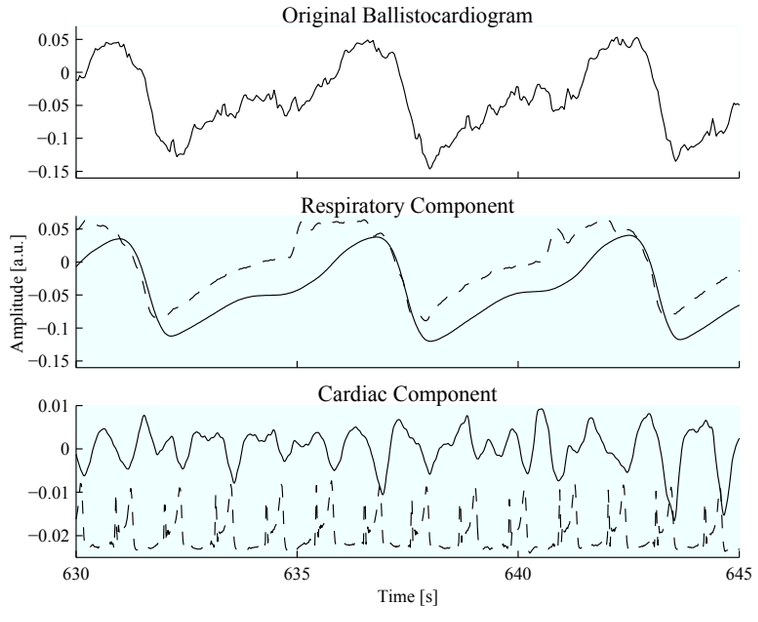


Figure 3.10: A section of a real BCG (top) with the respiratory (middle) and cardiac (bottom) components extracted via LPNR-based signal separation. The dashed lines represent the reference signals respiratory effort and ECG.

which is responsible for removing the critical parts with spectral overlap. The difference between LPNR and other approaches based on linear filters is that LPNR identifies signal components via typical time and amplitude scales of the deflections in the signal. This enables LPNR to distinguish the cardiac from the respiratory component in the BCG, despite the overlap between their spectra. Thus, LPNR offers a complementary approach to linear filtering [84].

Figure 3.10 shows a section from a real BCG, together with the cardiac and respiratory components extracted via LPNR-based signal separation. There is a clear correlation between the extracted BCG components and the reference signals. The power spectral density of the extracted BCG components are shown in figure 3.11, which illustrates that the broadband characteristic of the respiratory component's spectrum has been preserved by the LPNR-based separation process.

Figure 3.12 shows a section from the delay space trajectory of the raw BCG and the respiratory component obtained after the first LPNR filtering step in

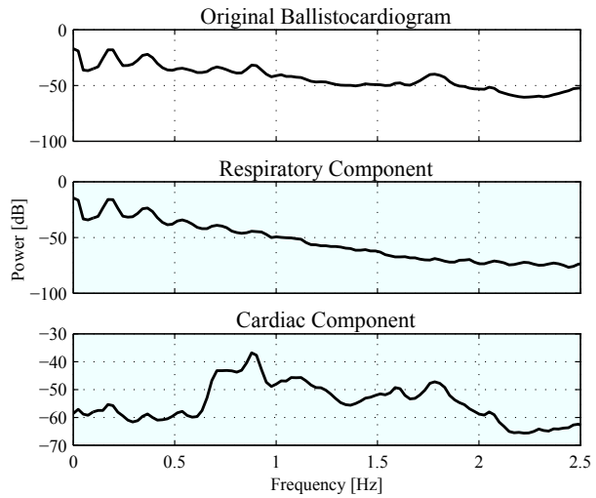


Figure 3.11: The power spectral density of the signals shown in figure 3.10.

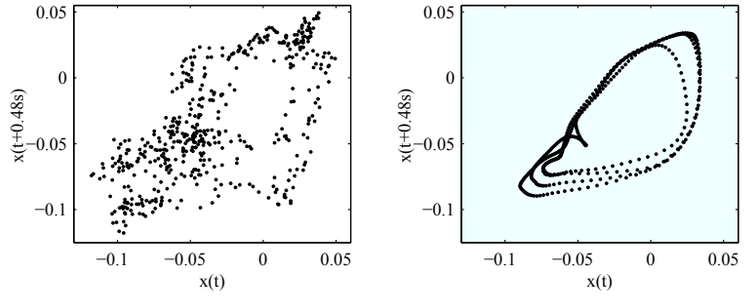


Figure 3.12: A section from the delay space trajectory of the BCG before (left) and after (right) the first LPNR filtering step to extract the respiratory component.

the left branch in figure 3.9. The noise process and the cardiac component cause a strong perturbation of the delay vectors away from the attractor shown in the left panel of figure 3.12. However, after LPNR, the trajectory is much cleaner and resembles a limit cycle attractor.

One important condition for the application of LPNR-based separation is that the amplitude scale and time scale of the BCG components must be known in advance and time invariant (or changing very slightly). In princi-

ple, this condition applies to every algorithm. E.g. when using linear filters, the cutoff frequencies of the filter which lead to optimal performance have to be known in advance. However, some quantities are more invariant than others. The typical human heart rate is around 1 Hz and although some variations are to be expected, this still limits the range of sensible cutoff frequencies for linear filters. On the other hand, the amplitude of the BCG and its components are influenced by many factors, including the gain of amplifiers in the recording equipment, the strength of the heart or the size and position of the patient, which makes it difficult to choose fixed parameters for LPNR-based separation in advance. In order to solve this problem, we have developed a parameter estimation scheme which attempts to estimate the amplitude scales of respiratory and cardiac components [85]. The estimates are based on the assumption that the time scale of the heartbeat activity is relatively invariant and known.

Although, the results in figure 3.10 are visually pleasing, a more objective assessment of the performance of LPNR-based separation is needed. In the next section, we will present an approach for verifying the BCG separation scheme presented here, which is based on modelling the BCG. This has the advantage that we can generate arbitrary amounts of BCG data, leading to a more accurate assessment. More importantly, simulation is the only way to obtain ground truth for the different BCG components, which cannot be measured directly.

3.5 Modelling BCG

In the last section, we have seen how modelling plays an important role in algorithm design. Even if we do not explicitly include a model during the design process, the assumptions made while deriving the algorithm often implicitly define a model. In the case of LPNR, we showed that the online version is closely related to mixture modelling. Furthermore, models are useful even after completing the design process, e.g. for improving or verifying an algorithm. If real data is difficult to obtain in sufficient quantity or quality, model generated data can be used for verification.

Thus, modelling is important for algorithm design, but on the other hand, algorithms can contribute to improvements in models. Advances in signal processing can lead to more advanced models, e.g. by helping to verify model assumptions, improving the choice of parameters or contributing new evidence to model selection problems.

In this section, we further investigate the two sides of this relationship. First, we take a closer look at BCG separation. Since real BCG recordings do not

provide acceptable means to obtain ground truth for the isolated respiratory and cardiac components, the performance of the LPNR-based separation can be verified only with artificial data. In section 3.5.1, we present a model for generating BCG which offers the possibility to freely select model parameters such as heart rate, respiration rate or the ratio between the amplitude of both components. This model relies on templates which are fitted to the cardiac and respiratory BCG components, in order to simulate their morphology. Using the ground truth for respiratory and cardiac BCG component from this model, it is possible to quantify the performance of the LPNR-based separation [84].

In the second part, we show how the LPNR-based BCG separation leads to a new improved model of the BCG cardiac component. The new model is probabilistic, which opens up the possibility to apply powerful methods from statistics and machine learning to higher level signal processing tasks like heartbeat detection. In addition, this model is non-parametric allowing it to adapt itself to changes in morphology of the BCG [86]. We have published part of the content from this section in [84] and [86].

3.5.1 Dynamic Nonlinear Model

One obstacle, which prevents us from quantifying the accuracy of the LPNR-based signal separation, is the difficulty of obtaining ground truth for the respiratory and cardiac BCG component in an experimental way. The cardiac component can be measured individually by asking the subject to hold his or hers breath. However, this requires the co-operation of the subject and only works for about a minute for healthy subjects. In addition, for patients with reduced lung capacity, it might be very difficult to voluntarily hold the breath. An even more serious problem with this method is that holding the breath for extended periods of time will increase the carbon dioxide concentration in the blood, which will affect heartbeat dynamics [87]. These changes in heartbeat dynamics are based on the physiological properties of the human body and cannot be avoided.

Recording a clean respiratory BCG component without cardiac influence is even more challenging and we do not know of any attempts to achieve this. Thus, we adopt a different strategy: We find short segments of clean respiratory and cardiac components in BCG recordings or substitute signals. These short segments are then used to fit models and generate more data in quantities large enough for verifying the LPNR-based separation.

This section is divided into two parts. First, we present our models for the respiratory and cardiac BCG components and describe how they are combined into the final BCG. Then, we present the performance results of

the LPNR-based BCG separation scheme, which we compare against the performance of FIR filters for reference.

BCG Component Models

The models for both respiratory and cardiac components are deterministic models based on simulating motion on a suitable limit cycle attractor in phase space [84]. These kinds of models have been successfully applied to modelling [70] and filtering [71, 72] ECG signals. Interestingly, the applications described in [71] and [72] are very similar to the problems where LPNR has been applied with great success.

Mathematically, our models consists of sets of coupled differential equations $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$, which are integrated to obtain the BCG. A similar concept called dynamical causal modelling (DCM) is used in computational neuroscience [88] to model interactions between brain regions. An in depth treatment of DCM is provided in chapter 4.2.

From a modelling perspective, the difference between modelling ECG and BCG is that for ECG, given a certain electrode placement, the signal morphology is essentially fixed. In fact, the ECG morphology is so consistent and reproducible that the series of recurrent peaks in ECG are designated capital letters from P to T [87]. These peaks have been linked to certain electrophysiological events in the cardiac cycle, e.g. the P-peak is associated with the electrical excitation of the atrium, while the R-peak is linked to ventricular depolarization [87].

To a certain extent, these characteristics are shared by BCG recorded using calibrated BCG tables. That is, for a given type of BCG table, the signal morphology is consistent and reproducible [47]. The difference is that BCG is linked to the mechanical cardiac activity instead of the electrical activity. Peaks in the BCG are designated capital letter from H to K, and some are associated with mechanical events in the cardiac cycle, e.g. the group of peaks I to K has been linked to ventricular ejection and aortic blood flow [46, 47].

However, modern BCG acquisition systems are optimized for home monitoring, which means that they are not calibrated and the position of the subject relative to the sensor is not well defined. This means that it is not possible to observe a consistent and reproducible signal morphology like in ECG or with calibrated BCG systems. The morphology can even change abruptly during a session, e.g. after a change of posture.

Given these challenges, our approach is to build a custom template from

samples of BCG with simultaneously recorded reference for ECG and respiration. The template is then used to build a model which is limited to generating BCG having the same morphology as the original sample, but with freely variable parameters like heart and respiration rate, or amplitude ratio between the components. This model provides ground truth for the cardiac and respiratory components by generating them separately. In the following, we first describe the model for the cardiac component and then the model for the respiratory component.

The Cardiac Component

In order to extract a template for the cardiac BCG component, we look for periods in the BCG, where the subject held his or her breath. Holding ones breath involuntarily during sleep is a natural process [89]. However, if the breathless period lasts longer than 10 s, it is called a sleep apnoea. The increased occurrence of apnoea is an indicator for sleep apnoea syndrome [89]. For creating the templates, we choose BCG samples from periods without breathing which last for several seconds and contain up to 20 heartbeats. These samples were chosen to be not too long, in order to minimize the risk of changes in heart dynamics within the duration of the sample.

Having found a BCG section which satisfies these conditions, we begin by segmenting the BCG section into segments each containing a single heartbeat, using the ECG reference. Then, we calculate a BCG heartbeat template by ensemble averaging these segments. Next, we fit a Gaussian bell curve to each major peak in the template, while we ignore minor peaks with amplitudes which do not range more than one standard deviation beyond the samples surrounding it. In addition, we use a sine wave to simulate the baseline.

To quickly obtain a solution, we have done the fitting manually by reading of the amplitude, width and relative position within the template of each peak. The result is a list of values for amplitudes, width and positions. This procedure is also used in [70] and it has the additional advantage of avoiding overfitting the model. In chapter 4, we will see how a similar model can be fitted by optimizing a suitably chosen criterion.

Given the information extracted from the template, we would like to define a set of differential equations which, after integration, provide us with the desired cardiac BCG component c. For this purpose, we will define the cardiac phase θ_c driven by the cardiac angular frequency ω_c, as well as the system equation $f_c(\theta_c)$ driven by the cardiac phase:

$$\dot{\theta}_c = \omega_c(t) \tag{3.18}$$

$$\dot{c} = f_c(\theta_c). \tag{3.19}$$

Since the cardiac template is parameterized by amplitude, width and position of several peaks, we would like to define equation (3.19) in such a way that the result after integration is a sum of Gaussian shaped bell curves. The Gaussian bell curve parameterized by phase and its derivative with respect to time are given by:

$$g(\theta) = a \exp\Big(- \frac{(\theta(t) - \theta_0)^2}{2\sigma^2} \Big) \tag{3.20}$$

$$\dot{g}(\theta) = - \frac{a\dot{\theta}(t)(\theta(t) - \theta_0)}{\sigma^2} \exp\Big(- \frac{(\theta(t) - \theta_0)^2}{2\sigma^2} \Big), \tag{3.21}$$

where the dot denotes derivation with respect to time. a is the peak amplitude, σ the peak width and θ_0 the position within the cardiac cycle. Thus, the system equation is given by a sum over terms, each representing one peak:

$$f_c(\theta_c) = - \frac{\omega_c^2}{\omega_{c,0}} \Big(a_0 \cos(\theta_c) + \sum_{i=1}^{I} \frac{a_i \Delta\theta_i}{\sigma_i^2} \exp\big(- \frac{\Delta\theta_i^2}{2\sigma^2} \big) \Big). \tag{3.22}$$

$\omega_{c,0}$ is the mean cardiac frequency and $\Delta\theta_i$ is the phase difference between the current cardiac phase θ_c and the phase position θ_i of the i-th peak, which is defined in a circular way as:

$$\Delta\theta_i = \begin{cases} \theta_c - \theta_i \mod 2\pi & \text{if } \theta_c - \theta_i \mod 2\pi \leq \pi, \\ \theta_c - \theta_i \mod 2\pi - 2\pi & \text{else.} \end{cases} \tag{3.23}$$

The cardiac frequency ω_c can be varied continuously to determine the instantaneous heart rate. In principle, the model allows us to choose any definition for ω_c. However, physiological studies indicate that there is a coupling between the respiratory phase and heart rate [9, 10, 90] called respiratory sinus arrhythmia (RSA). In addition, it has been known since the early days of BCG research that the amplitude of the cardiac BCG component is modulated by the respiratory phase [46, 91]. Although, recent results suggest that the respiratory influence on the cardiac BCG component includes a modification of the morphology of the cardiac signal which goes beyond amplitude modulation [92].

To keep our model simple, we do not consider the effect described and modelled in [92]. However, we do model the modulation of heart rate and amplitude of the cardiac component by coupling the cardiac frequency ω_c to the base oscillation of the respiratory component. For this to work, we need to define the model for the respiratory component.

The Respiratory Component

We have pointed out that recording a clean respiratory BCG component

without cardiac influence is difficult. Therefore, for training a respiration model, we turn to a substitute signal: the respiratory effort recorded with a respiration belt. In BCG studies, the respiratory effort is often recorded as a reference for the respiratory activity by means of a belt around thorax or abdomen, which contains a pressure sensor. From the way respiratory effort is recorded, one can imagine that it is very similar in nature to the respiratory component of the BCG recorded by a bed mounted sensor or by a sensor mounted in the back rest of a chair. This impression is also reinforced by the samples shown in figure 3.2. The sensors built into respiratory belts have a lower sample rate and are less sensitive compared to BCG sensors, since the respiratory activity is slower and has a larger amplitude than the cardiac BCG activity. Thus, the cardiac influence in the respiratory effort signal is much smaller than in the BCG.

Similar to the cardiac model, we extract a respiratory template from the respiration belt signal by ensemble averaging the signal over several breaths. The boundaries between breaths are segmented using an automated algorithm for minima detection and visually checked for errors. The template obtained using this procedure can contain some low amplitude fluctuations which could be due to remaining cardiac influence or reflect noise and artefacts present in the original sample. We suppress these fluctuations by calculating the Fourier coefficients of the template and discard coefficients with low amplitude, while at the same time we ensure that the remaining coefficients account for 99 % of the signal energy.

The list of the indices k_j of the non-zero Fourier coefficients along with the complex values of the coefficients $c_{k_j} = c_{r,k_j} + ic_{i,k_j}$ are the parameters of our respiration model, which is characterized by the following equations:

$$\dot{\theta}_r = \omega_r(t) \tag{3.24}$$
$$\dot{r}_1 = f_{r_1}(\theta_r) \tag{3.25}$$
$$\dot{r}_2 = f_{r_1}(\theta_r). \tag{3.26}$$

θ_r is the respiratory phase, r_1 is the base oscillation of the respiratory component used to define the cardiac frequency and r_2 is the respiratory BCG component. Since the signal after integration should be a sum of sine and cosine waves weighted by the coefficients, the system equations are defined as a sum over the derivatives of sine and cosine functions with the respective weights. Thus, f_{r_1} is defined as:

$$f_{r_1} = -\frac{\omega_r k_1}{|c_{k_1}|}\big(c_{r,k_1}\sin(k_1\theta_r) + c_{i,k_1}\cos(k_1\theta_r)\big) \tag{3.27}$$

and f_{r_2} as:

$$f_{r_2} = -\frac{2\omega_r a_r}{N_t} \sum_{j=1}^{J} k_j \big(c_{r,k_j} \sin(k_j\theta_r) + c_{i,k_j} \cos(k_j\theta_r)\big). \tag{3.28}$$

The instantaneous amplitude $a_r(t)$ of the respiratory component r_2 is defined as a sine shaped modulation with period T_a and phase offset $\phi_{a,0}$:

$$a_r(t) = 1 + \Delta a_r \sin\Big(\frac{2\pi}{T_a} t + \phi_{a,0}\Big), \tag{3.29}$$

where Δa_r is the maximum relative amplitude variation. N_t denotes the length of the respiratory template in samples.

To complete the model description, we have to define the angular frequencies for the respiratory and cardiac components, ω_r and ω_c. Since respiration rate depends on many factors, we decided to keep our model simple by modelling ω_r as a slow oscillation around a base frequency $\omega_{r,0}$:

$$\omega_r(t) = 2\pi\omega_{r,0}\Big(1 + \Delta\omega_r \sin\big(\frac{2\pi}{T_{\Delta r}} t\big)\Big) \tag{3.30}$$

with period $T_{\Delta r}$ and maximum relative amplitude variation $\Delta\omega_r$. We think that this is an appropriate model for breathing during resting state. In addition, we also experimented with non-periodic oscillating respiratory frequencies by replacing the sine in equation (3.30) with one of the components from the Rössler equations [93]. Alternatively, one could also attempt to extract the information on the respiratory frequency from experimental data, e.g. respiration belt recordings.

Earlier, we pointed out that for physiological reasons, the cardiac frequency is coupled to the respiratory phase [90] due to the RSA. In order to model this aspect, we define the cardiac frequency ω_c as:

$$\omega_c(t) = \omega_{c,0}\big(1 + \Delta\omega_c r_1(t)\big), \tag{3.31}$$

with $\omega_{c,0}$ being the mean cardiac frequency and $\Delta\omega_c$ the maximum relative deviation from the mean frequency. This definition ensures that the heart rate increases while breathing in and decreases while breathing out. At the same time, the amplitude modulation of the cardiac component with respiratory phase is realized by the multiplicative factor of $\omega_c^2/\omega_{c,0}$ in equation (3.22). An interesting problem for future research would be the inclusion of the direct coupling between the cardiac and respiratory phases reported in [9, 10] into the model.

The advantage of specifying our model as a dynamical system consisting of sets of coupled differential equations lies in the smoothness of the model

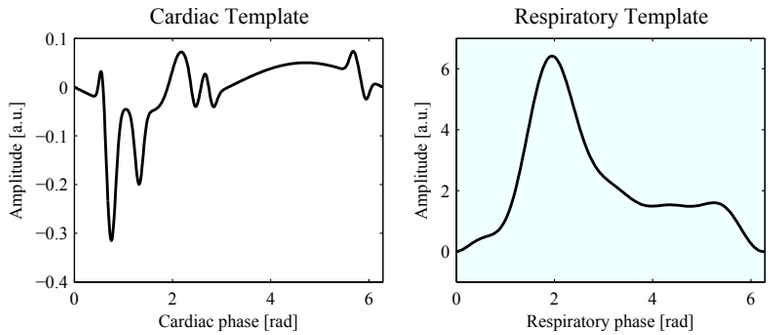


Figure 3.13: The templates of the cardiac (left) and respiratory (right) BCG components plotted against the cardiac and respiratory phase, respectively.

output. The resulting BCG components show no discontinuities at the template boundaries and are continuously differentiable. The reason is that the information on the shape of the template is decoupled from the calculation of the cardiac and respiratory frequencies and amplitudes. Therefore, the model enables us to vary amplitude and frequency of both components in a continuous fashion, which allows us to define instantaneous heart rates and respiration rates. In contrast, models which model heartbeat by generating heartbeat instances have to define the heart rate in a piecewise fashion, e.g. as a step function, or perform interpolation [86]. Although this is possible (we will introduce such a model in section 3.5.2), it is not as convenient as having access to a well-defined instantaneous heart rate. In figure 3.13, the respiratory and cardiac BCG templates are plotted against the cardiac and respiratory phase, respectively.

Verifying the BCG Components Separation

As mentioned, the primary application of our BCG model is to generate synthetic BCG with available ground truth for the cardiac and respiratory components, in quantities sufficient for the evaluation of the LPNR-based signal separation. After obtaining the respiratory and cardiac templates and fitting the model, we have to choose physiological plausible values for a number of parameters described in the previous section, before we can generate data. These parameters include the mean respiration rate $\omega_{r,0}$, the maximum relative deviation from this mean $\Delta\omega_r$ or the time scale $T_{\Delta r}$ on which the fluctuation of the respiration rate takes place. Table 3.2 contains a list of all these parameters and their values used for generating the data for our simulation in [84].

	Respiratory Parameters						Cardiac Parameters	
Name	$\omega_{r,0}$	$\Delta\omega_r$	$T_{\Delta r}$	$\phi_{a,0}$	Δa_r	T_a	$\omega_{c,0}$	$\Delta\omega_c$
Value	$0.2\,\mathrm{s}^{-1}$	0.2	30 s	1 rad	0.3	20 s	$5.7\,\mathrm{s}^{-1}$	0.2

Table 3.2: Numerical values of the respiratory and cardiac model parameters used for the simulation in [84].

In addition to the respiratory and cardiac components, the simulated BCG should also contain a noise component. For the experiments in [84], we used a noise component with low-pass characteristics. Since the mechanical coupling between subject and BCG sensor, as well as the electronic recording equipment have low-pass characteristic, one cannot assume that the noise component in real BCG is white. Therefore, we generated the noise component by filtering white noise with a 2^{nd} order infinite impulse response filter with coefficients 1 and -0.9. The experiments are repeated for eleven different noise levels, which range linearly between no noise and noise with a standard deviation of $-8\,\mathrm{dB}$ relative to the cardiac component [84]. The simulated BCG segments were 70 s long. But to average out random effects, we generated 100 realizations for each noise level. Thus, the results of the experiments are based on more than 21 h of simulated data.

Figure 3.14 shows a 9 s sample from the artificial BCG together with the ground truth for the cardiac and respiratory components. For visual reference, a section from a real BCG recording is shown in the bottom panel. The power spectral density for the artificial BCG and its components is shown in figure 3.15. One can notice how the spectrum of the respiratory component falls off sharply at 1.5 Hz. This is due to cutting off the Fourier coefficients while creating the respiratory template. However, the cardiac component already has significant spectral components starting at 0.5 Hz, which means that there is an overlap between the spectra of the respiratory and cardiac components in the range from 0.5 Hz to 1.5 Hz. As a reference, the bottom panel in figure 3.15 shows the power spectral density of a segment from a true BCG, which displays more pronounced peaks, but otherwise resembles the spectrum of the artificial signal.

Using the generated BCG, we can assess the separation performance of the LPNR-based BCG separation algorithm by comparing the components after separation $\tilde{c}$ and $\tilde{r_2}$ to their respective ground truth values c and r_2. For this purpose, we defined the normalized mean square error as a performance measure in [84]:

$$nmse_c = \frac{\langle(\tilde{c}-c)^2\rangle}{\langle c^2\rangle} \quad \text{and} \tag{3.32}$$

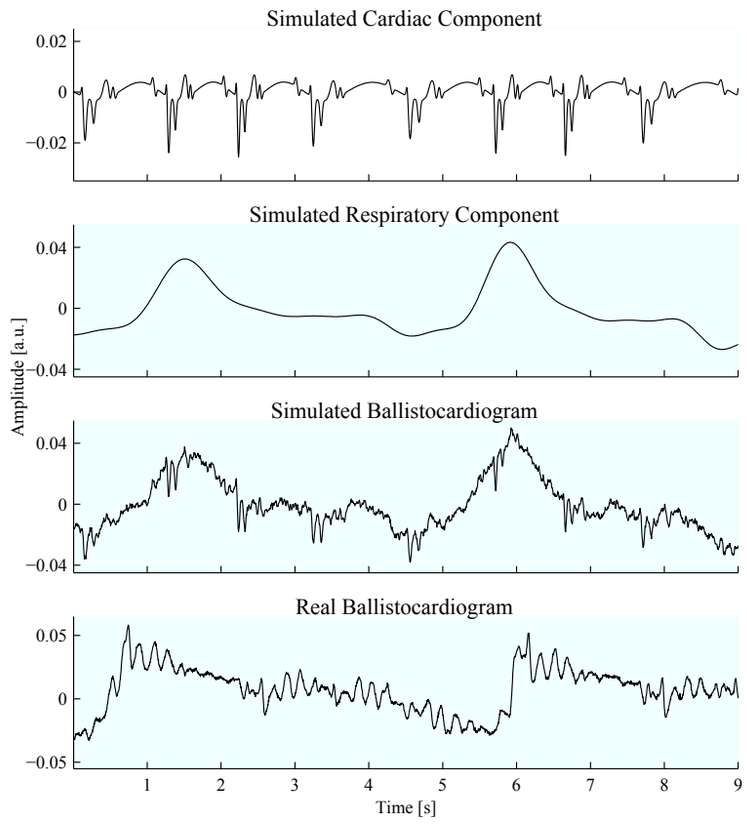


Figure 3.14: Sample of an artificial BCG segment with the corresponding cardiac and respiratory components and a section from a real BCG recording for visual reference.

$$nmse_r \quad = \quad \frac{\langle (\tilde{r_2} - r_2)^2 \rangle}{\langle r_2^2 \rangle}. \tag{3.33}$$

The angular brackets denote taking the expectation of the terms inside the brackets.

In order to provide a reference, we evaluate this performance measure not only for the LPNR-based separation, but also for separation based on linear filters. For the separation with linear filters, we used zero-phase finite impulse response (FIR) filters of order 1024 with a sampling frequency of 200 Hz. The respiratory component was extracted using a low-pass filter and the cardiac component was extracted with a band-pass filter. The cut-

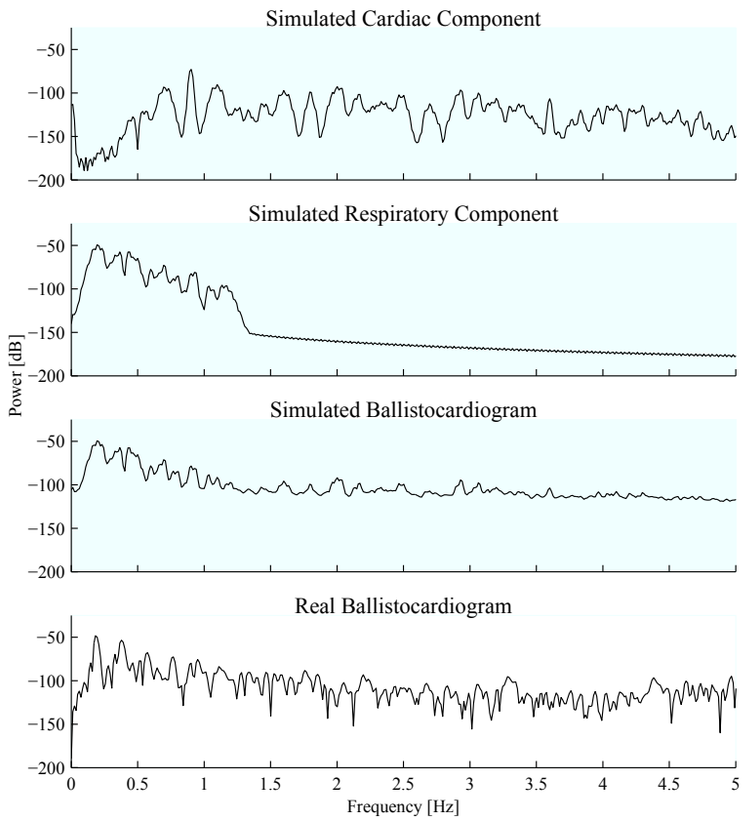


Figure 3.15: The power spectral density of the artificial BCG components shown in figure 3.14.

off frequencies of both filters were optimized individually, i.e. for each noise level and each component we chose the cutoff frequencies that yielded the lowest error.

The parameters for the LPNR-based separation were set manually according to the criteria given in section 3.4. After visual inspection of the BCG, ε is chosen slightly larger than the peak-to-peak amplitude of the cardiac component in order to extract the respiratory component. In the second step, ε is chosen larger than the noise level but smaller than the cardiac component to extract the cardiac component and reduce noise. To assess the sensitivity of LPNR-based separation scheme to misspecification of ε, we varied ε by $\pm 20\,\%$ relative to the chosen value. The results of this

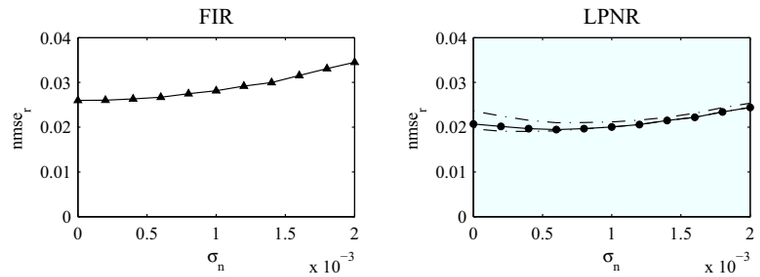


Figure 3.16: Normalized mean squared error obtained from the simulation in [84] plotted against the noise level. Here, shown for the respiratory component extracted with linear filters (left) and the LPNR-based separation (right).

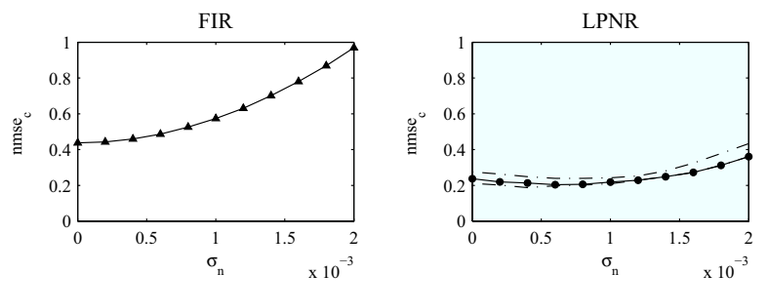


Figure 3.17: Normalized mean squared error obtained from the simulation in [84] plotted against the noise level. This time for the cardiac component extracted with linear filters (left) and the LPNR-based separation (right).

analysis are summarized in figure 3.16 for the respiratory component and in figure 3.17 for the cardiac component.

The graphs show that for both components and at all noise levels the LPNR-based separation achieves superior results than the FIR filters. This holds true even in the case where ε is perturbed from the manually chosen value. An interesting observation is that in general the manually chosen values of ε lead to optimum results for the LPNR-based separation. However, for low noise levels perturbing the values sometimes results in a slightly lower error.

Also note that the error increases more rapidly with the noise level for the FIR filter based separation than for the LPNR-based version; especially for the cardiac component shown in figure 3.17. This means that the nonlinear noise reduction is more effective at reducing the low frequency noise compo-

nent in the artificial BCG. On the other hand, the fact that the LPNR-based separation achieves a lower error, even in the noise free case, indicates that the LPNR-based separation is inherently better at preserving the signal morphology of the BCG components in the presence of a spectral overlap between the components.

The BCG model presented in this section models both the cardiac and respiratory components using sets of differential equations. It is a parametric model and except for the additive measurement noise, it is also purely deterministic. The data generated with this model is used to validate the LPNR-based BCG separation presented in section 3.4. In the next section, we will present a model for the cardiac BCG component only, which is non-parametric and probabilistic. The new model will allow us to apply powerful, statistical methods for model inversion

3.5.2 Probabilistic Model

Throughout this thesis, much emphasis has been put on the close relationship between modelling and signal processing. In the previous section, we have shown how model generated data contributes to the verification process of our signal separation scheme. Now, we are changing the perspective by showing how the LPNR-based separation of BCG enables us to build an improved model of the cardiac BCG component. Part of the content in this section has been presented in our previous publication [86].

The dynamic nonlinear model from the previous section is deterministic, i.e. the morphology of the generated BCG is fixed and predetermined by the template. Some parameters like heart and respiration rate or the amplitudes of the components can be varied. But once the parameters governing the signal shape are fitted to a given template, the model cannot be adapted to changes in the signal morphology. In addition, the model is essentially a parametric model, since all information is contained in a fixed number of parameters, which are fitted to the template or set by hand.

In this section, we present a non-parametric, probabilistic model of the cardiac BCG component, which adapts itself to the current morphology of the BCG. The model is governed by the central assumption that the BCG is quasi-periodic, i.e. the signal follows a pattern consisting of several subsequent peaks, which repeats itself for each heartbeat. The peak pattern of adjacent heartbeats are very similar, but the morphology can change with time and becomes more dissimilar with growing temporal distance. The model has hyper-parameters controlling general properties of the BCG like amplitude range and time scale of the peaks, but the exact morphology of

the signal is learned from the data. In addition, the model is probabilistic, which means that when using this model, we have very powerful methods from statistics and machine learning at our disposal to solve inference tasks like heartbeat detection.

The model presented in this section consists of two parts. The first part is a model of the heartbeat sequence underlying the BCG. For this part, we used an inverse Gaussian (IG) renewal process [86]. The second part of the model is tasked with modelling the morphology of the BCG, given a certain heartbeat sequence. This is done with a Gaussian process (GP) model. In the following, we first present the heartbeat sequence model. Then, we describe the signal morphology model and how the two parts are combined to yield the complete BCG model. Finally, we present an approach to heartbeat detection in BCG, as an example of how this model can be applied to a practical problem.

Modelling the Heartbeat Sequence

The task of the beat sequence part is to model the time intervals between subsequent heartbeats. In principle, any model predicting heartbeat interval times can be used for this part. However, we prefer a probabilistic model, which means that the model output is not a single estimate of the time to the next heartbeat. Instead, the model should provide a distribution over possible values of heartbeat time intervals, with values having higher probability when they are more likely to occur. For this reason, we chose the inverse Gaussian (IG) distribution (see appendix A.3) to model heartbeat intervals in our framework.

The IG distribution is a probability distribution with support over the non-negative real numbers. It is the distribution of the first crossing time of a Gaussian random walk with drift [94] and has been used primarily as a model for failure time. However, Barbieri *et al.* demonstrated that it is also a good model for heartbeat intervalls [95, 96], since the physiology of heartbeat generation can be modelled as a Gaussian random walk with drift of the action potential in the sinoatrial node towards a threshold [87].

It is interesting to note that the same model is known in computational psychology as the drift diffusion model, where it is used model decision making in uncertain environments [97].

The model proposed in [95] consists of an IG process with a mean parameter which depends on the history of past heartbeats. For our purpose, we have simplified the model to an IG distribution with static parameters μ and λ,

which can be learned e.g. from ECG data:

$$\mathcal{IG}(\Delta t|\mu,\lambda) = \sqrt{\frac{\lambda}{2\pi\Delta t^3}}\exp\big(-\frac{\lambda}{2\mu^2\Delta t}(\Delta t-\mu)^2\big). \tag{3.34}$$

Equation (3.34) defines the distribution over a single beat interval Δt with $0 \leq \Delta t < \infty$. If we allow a whole sequence of heartbeats $\{t_b\}_{b=1\ldots B}$ to be contained in the observation interval, the distribution is given by:

$$\begin{aligned}\log(p(\{t_b\}|\mu,\lambda)) = \log(1-\mathcal{IG}'(t_1-t_{min}|\mu,\lambda)) \\ + \sum_{b=1}^{B-1}\log(\mathcal{IG}(t_{b+1}-t_b|\mu,\lambda)) \\ + \log(1-\mathcal{IG}'(t_{max}-t_B|\mu,\lambda)),\end{aligned} \tag{3.35}$$

where the $\mathcal{IG}'$ symbol in the first and last term denotes the cumulative distribution function of the IG distribution (eq (A.30)). The first term describes the probability that the first heartbeat occurs at t_1, while the last term is the probability that the last beat occurs at t_B. The sum is the probability of the B heartbeat intervals in between. Note that in our notation t_b is the time when the b-th heartbeat occurs, while $\Delta t_b = t_{b+1} - t_b$ is the time interval between the b-th heartbeat and the next beat.

The model described by equation (3.35) does not depend on the measured BCG, but assumes that heartbeat intervals are drawn independently from a static IG distribution. Thus, equation (3.35) can be viewed as a prior distribution on the heartbeat sequence in the observation interval (t_{min}, t_{max}).

Modelling the BCG Morphology

The task of the second part of the model is to model the morphology of the BCG, given a possible heartbeat sequence. Since the morphology of the BCG acquired with modern uncalibrated devices can vary substantially, we would like the model to adapt itself to the waveform. For this purpose, we choose a Gaussian process (GP) model for this part.

The Gaussian process is a non-parametric model for regression and classification [44]. Informally, it can be viewed as a generalization of the Gaussian distribution over finite dimensional vectors to a probability distribution over functions. Given an underlying function $f(\boldsymbol{x})$, which is sampled at points $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N$, the Gaussian processes provides a way to specify a Gaussian distribution over the set of function values $\{f(\boldsymbol{x}_n)\}_{n=1,\ldots,N}$. If a new sample $f(\boldsymbol{x}_{N+1})$ is added, the Gaussian process allows the distribution to be extended to the new sample in a consistent way.

The key component, which provides the GP with this flexibility is the covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$. It has the same structure as a kernel function, which means it accepts two inputs and outputs a scalar value. The matrix formed by plugging pairs of function arguments $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ into the covariance function forms the covariance matrix for a Gaussian distribution over the vector of (noise free) function values $\boldsymbol{f} = \big(f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)\big)^T$, i.e.:

$$\boldsymbol{f} \sim \mathcal{N}(0, K) \quad \text{and} \tag{3.36}$$

$$K = \begin{pmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix}. \tag{3.37}$$

When adding a new sample $f(\boldsymbol{x}_{N+1})$, K is simply extended by one row and one column using the covariance function. The data variable, which in our case is the recorded BCG $\boldsymbol{y}$, is modelled as a noisy observation of $\boldsymbol{f}$ with variance σ:

$$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{f}, \sigma I). \tag{3.38}$$

However, the covariance function has to guarantee that the matrix K is always positive definite, irrespective of the exact values of the function arguments $\boldsymbol{x}_n$. Thus, not any function of two arguments can be used as a covariance function. In addition, the form of the covariance function determines which properties the function $f(\boldsymbol{x})$ needs to have in order to receive a high probability under the Gaussian process model. In this respect, the covariance function in a Gaussian process is analogous to the covariance matrix in a Gaussian distribution, since in a Gaussian distribution the covariance matrix determines which vectors receive a high probability value. [44] provides a comprehensive overview on Gaussian processes, including a list of functions that can be used as covariance functions and their properties.

For the BCG model, we need a Gaussian process with a covariance function which prefers signals that are periodic with respect to the underlying beat sequence. However, we must also take into account that the true BCG is not exactly periodic and that the shape of the signal can change with time. To achieve this, we define a compound covariance function consisting of several components [86].

The most important part of the compound covariance function is the periodic component:

$$k_p(t', t) = \sigma_p^2 \exp\Big(- \frac{2\sin^2(\phi(t') - \phi(t))}{\tau_p^2} - \frac{(t' - t)^2}{2\tau_c^2} \Big), \tag{3.39}$$

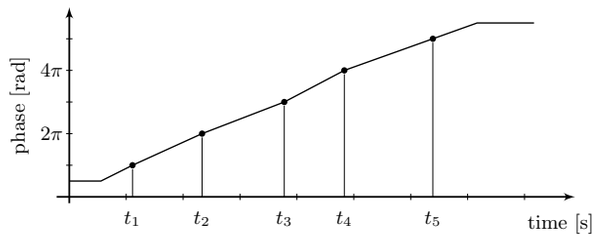


Figure 3.18: Illustrating the phase calculation for a sequence of five heartbeats t_1 to t_5. The phase is fixed to multiples of π at the beat times marked by dots, and linearly interpolated or extrapolated elsewhere. Phase extrapolation is capped at $\pi/2$ before the first beat and $11/2\pi$ after the last beat.

which depends on the cardiac phase function $\phi(t)$, defined as:

$$\phi(t) = \begin{cases} \frac{\pi}{2} & \text{if} \quad t < t_1 - \frac{\Delta t_1}{2} \\ \pi\left(1 - \frac{t_1 - t}{\Delta t_1}\right) & \text{if} \quad t_1 - \frac{\Delta t_1}{2} \leq t < t_1 \\ \pi\left(b + \frac{t - t_b}{\Delta t_b}\right) & \text{if} \quad t_b \leq t < t_{b+1} \\ \pi\left(B + \frac{t - t_B}{\Delta t_{B-1}}\right) & \text{if} \quad t_B \leq t < t_B + \frac{\Delta t_{B-1}}{2} \\ \pi\left(B + \frac{1}{2}\right) & \text{else.} \end{cases} \tag{3.40}$$

t_b and Δt_b are the heartbeat times and heartbeat intervals of the underlying beat sequence, which for this part of the model can be treated as known inputs. The intuitive meaning of equation (3.40) is that the phase $\phi(t)$ is set to multiples of π at the heartbeat times, i.e.: $\phi(t_b) = b\pi$. For time instances between two heartbeats, the phase is linearly interpolated and for times shortly before the first and after the last beat, the phase is linearly extrapolated. However, the extrapolation is capped beyond $\pi/2$ and $\pi(B + 1/2)$. Figure 3.18 visualizes the cardiac phase function for an example sequence containing five beats t_1 to t_5. This piecewise definition of the cardiac phase is not as convenient as the continuously defined phase of the nonlinear dynamic model introduced in section 3.5.1, but our experience with the Gaussian process model showed us that it works well in practice.

With this, we can now give an interpretation of equation (3.39). The sine term in the argument of the exponential is causing the preference for periodic behaviour. This is because, when t and t' happen to be at the same phase within the cardiac cycle, the phase difference in the argument of the sine will be a multiple of π and the sine will be zero. Thus, the exponential, and with it the covariance, will be maximized for combinations of inputs which are at the same phase within the cardiac cycle. The hyper-parameter τ_p can be used to express the time scale we expect the variations in the signal to

be on.

The second term in the argument of the exponential serves to decrease the covariance with increasing temporal distance between the input times. This means that if we sample the BCG at time instances which are on the order of τ_c seconds apart, we expect a difference in the signal amplitude, even if the time instances happen to be at the same phase within the cardiac cycle.

Aside from the periodic component, we also defined two contributions to the covariance function which model noise components in the signal. Both components are described by the same squared exponential [44] type equation:

$$k_{SE,i}(t',t) = \sigma_{SE,i}^2 \exp\Big(-\frac{(t'-t)^2}{2\tau_{SE,i}^2}\Big) \quad (i=1,2),$$

while the difference lies in the values of the $\tau_{SE,i}$ parameter. In [86], we choose $\tau_{SE,1}$ on the order of 1 s to model irregular variations on a beat-to-beat level like baseline wander. $\tau_{SE,2}$ is set to a value on the order of 0.1 s to model intra-beat variations.

The covariance function used in the GP model of BCG is the sum of these three components:

$$k_B(t',t) = k_p(t',t) + k_{SE,1}(t',t) + k_{SE,2}(t',t). \tag{3.41}$$

Given a section from a BCG recording $\boldsymbol{y} = (y_1, y_2, \ldots, y_N)^T$ sampled at time instances $t_{s,1}, t_{s,2}, \ldots, t_{s,N}$, the covariance matrix K is obtained by plugging the sampling time pairs into the covariance function from equation (3.41):

$$K = \begin{pmatrix} k_B(t_{s,1},t_{s,1}) & k_B(t_{s,1},t_{s,2}) & \ldots & k_B(t_{s,1},t_{s,N}) \\ k_B(t_{s,2},t_{s,1}) & k_B(t_{s,2},t_{s,2}) & \ldots & k_B(t_{s,2},t_{s,N}) \\ \vdots & \vdots & \ddots & \vdots \\ k_B(t_{s,N},t_{s,1}) & k_B(t_{s,N},t_{s,2}) & \ldots & k_B(t_{s,N},t_{s,N}) \end{pmatrix}. \tag{3.42}$$

The log-probability of the BCG $\boldsymbol{y}$ conditioned on a beat sequence $\{t_b\}_{b=1,\ldots,B}$ is given by plugging in $\boldsymbol{y}$ and the covariance matrix K from equation (3.42) into the general formula for the log-marginal probability of a Gaussian process derived in [44]:

$$\begin{aligned} \log\big(p(\boldsymbol{y}|\{t_b\}_{b=1,\ldots,B})\big) = &-\frac{1}{2}\boldsymbol{y}^T(K+\sigma_n^2 I)^{-1}\boldsymbol{y} \\ &-\frac{1}{2}\log|K+\sigma_n^2 I| - \frac{N}{2}\log(2\pi). \end{aligned} \tag{3.43}$$

Here, I is an identity matrix of size N, σ_n denotes the standard deviation of the measurement noise and N is the number of samples in the BCG $\boldsymbol{y}$.

Equation (3.43) is the so called log-likelihood function of our model. It tells us how probable the observed BCG $\boldsymbol{y}$ is if we knew the underlying heartbeat sequence $\{t_b\}_{b=1,\ldots,B}$, which in our case is the latent variable we would like to infer. Together with equation (3.35), it forms the two parts of our BCG model, which we can combine using the logarithmic form of the chain rule (eq. (2.9)):

$$\begin{aligned}\log\big(p(\boldsymbol{y},\{t_b\})\big) &= \log\big(p(\{t_b\})\big) + \log\big(p(\boldsymbol{y}|\{t_b\})\big) \qquad (3.44)\\ &= \sum_{b=1}^{B-1} \log\big(\mathcal{IG}(t_{b+1}-t_b|\mu,\lambda)\big)\\ &\quad + \log\big(1-\mathcal{IG}'(t_1-t_{min}|\mu,\lambda)\big)\\ &\quad + \log\big(1-\mathcal{IG}'(t_{max}-t_B|\mu,\lambda)\big)\\ &\quad - \frac{1}{2}\boldsymbol{y}^T(K+\sigma_n^2 I)^{-1}\boldsymbol{y}\\ &\qquad - \frac{1}{2}\log|K+\sigma_n^2 I| - \frac{N}{2}\log(2\pi). \qquad (3.45)\end{aligned}$$

This equation provides us with the logarithm of the joint probability density over the data $\boldsymbol{y}$ and the latent variable $\{t_b\}_{b=1,\ldots,B}$ and forms the key component of our model. In addition, it is one of the terms in Bayes law (eq. 2.10), which provides us with the posterior density

$$p(\{t_b\}_{b=1,\ldots,B}|\boldsymbol{y}) = \frac{p(\boldsymbol{y},\{t_b\}_{b=1,\ldots,B})}{p(\boldsymbol{y})} \qquad (3.46)$$

used for inverting the model. Note that the denominator $p(\boldsymbol{y})$ in the above equation, which is called the model evidence, depends only on the data $\boldsymbol{y}$ and is a constant with respect to the beat sequence. Thus, the joint probability from equation (3.45) is proportional to the posterior density:

$$\log\big(p(\boldsymbol{y},\{t_b\})\big) = \log\big(p(\{t_b\}_{b=1,\ldots,B}|\boldsymbol{y})\big) + const. \qquad (3.47)$$

This observation will be important for the application of the model to heartbeat detection, which we introduce in the next section.

Application to Heartbeat Detection in BCG Signals

The list of potential applications for the BCG model introduced in the last two sections includes many problems. Similar to the nonlinear dynamic model introduced in section 3.5.1, we can use the GP model to generate data for verifying other algorithms. The difference to the nonlinear dynamic model, which except for the measurement noise was deterministic, is that data generated with the GP model will be inherently random, even if we do not add measurement noise.

The generation of data for testing algorithms is of special relevance to BCG since there are no freely accessible data bases for BCG, like there are for ECG or EEG [98]. A reason that contributed to this situation in the past is certainly that BCG has never been established as a widespread method. However, with modern BCG devices for home monitoring another contributing factor is the variety of acquisition systems being developed, often for commercial purposes. These systems are based on different working principles and have different characteristics, which makes it difficult to find a common standard. On the other hand, the comparability of results across publications would certainly benefit from a common database to compare the algorithms on. Until such a database is established, model generated data might be a solution to this problem.

Another possible application, which we explored in [86], is the reconstruction of missing data from damaged BCG recordings if a simultaneously recorded ECG reference is available. The advantage of the GP model in this setting is that it does not only provide a single estimate of the missing BCG in the damaged part of the recording, but also provides the variance of the estimate with little additional computational effort.

However in this section, we will discuss the application of the Gaussian process BCG model to the problem of detecting heartbeats in the BCG. We have chosen heartbeat detection as an example application for several reasons. First of all, heartbeat detection in BCG is a problem with high practical relevance, which is evident from the multitude of proposed methods in the BCG literature including but not limited to [99, 100, 101, 102, 103, 104, 105, 106]. Furthermore, it seems that so far, the approach of treating the heartbeat detection problem as a probabilistic inference task has been neglected in the BCG literature. This could be due to the sparseness of probabilistic BCG models which support statistical inference.

The BCG model introduced in equation (3.45) provides a description of how probable a certain BCG $\boldsymbol{y}$ is, in combination with a certain heartbeat sequence. In practice, one knows the BCG $\boldsymbol{y}$ and the aim is to infer the heartbeat sequence based on the observed information. The approach advertised in Bayesian statistics is to treat the heartbeat sequence as a random variable and calculate its probability distribution conditioned on the observation $\boldsymbol{y}$. This conditional distribution, which is called the posterior distribution, is proportional to the joint-distribution (eq. (3.47)) and expresses how probable a certain beat sequence is, given the information from the observed BCG.

In order to obtain an approximation to the posterior distribution, we turn to the method of Markov chain Monte Carlo (MCMC), which has been introduced in section 2.4.1. Applied to the heartbeat detection problem,

MCMC draws a set of representative samples from the space of possible beat sequences, which form an approximation to the posterior distribution that becomes more and more accurate with increasing number of samples. Thus, a sample consists of a heartbeat sequence and the output of the MCMC scheme is a set of plausible beat sequences that could have generated the observed BCG.

As described in section 2.4.1, MCMC obtains the set of representative samples by starting with an initial sample and continuing with proposing a small change to the current sample. The modified sample is accepted or rejected based on its and the current sample's probabilities. If accepted, the modified sample becomes the new current sample, otherwise the old sample stays the current sample. This way, a chain of samples is formed which is the set of representative samples we want. The distribution over the samples in this chain is guaranteed to converge to the target distribution, which in our case is the posterior distribution over heartbeat sequences, as long as the detailed balance condition is satisfied [28].

One problem with this scheme is that, since our samples consists of heartbeat sequences of potentially different length, the standard methods of proposing modified samples do not work [28]. In [86], we have solved this problem by introducing a custom proposal scheme that was specifically developed for the inference of heartbeat sequences with the Gaussian process BCG model. The details of the proposal scheme are given in appendix B.

In figure 3.19, a typical result from a run of the MCMC scheme is illustrated. The top panel shows the BCG for which the heartbeat positions should be detected. The second panel shows the beginning of the chain of samples generated by the MCMC scheme including the first 23 samples, while the third panel shows the last 23 samples in the Monte Carlo chain.

Each line corresponds to one sample consisting of a potential heartbeat sequence that could have generated the BCG in the top panel and the first line marked by *init* corresponds to the initial sample. Note how the heartbeats, marked by black dots, in the samples at the beginning of the chain change their position rapidly and sometimes even appear at very unlikely positions. On the other hand, the samples at the end of the chain are more stable, since the chain has reached convergence.

Upon completion, the MCMC scheme returns the chain of samples as the output, which consists of a set of potential heartbeat sequences. The beat sequences in this chain represent the approximation to the posterior distribution and should in principle be treated as the result of the inference process. However, the user of a heartbeat detection algorithm will typically request an answer to the question: At which time instances did the heartbeats oc-

cur? In order to make the result of the MCMC scheme more interpretable, we apply a kernel density estimator to the samples in the chain by placing a small Gaussian curve on the position of each heartbeat in each sequence of the chain and summing over all Gaussian curves.

The fourth panel in figure 3.19 illustrates the output of the kernel density estimator, which is a smooth curve approximating the marginal beat probability density. Roughly speaking, it approximates the probability for each time instance t that there is a heartbeat in an infinitesimal interval around t. It is evident from figure 3.19 that the probability at t is high whenever there are many sequences with a heartbeat near t. In addition, the peaks of the marginal beat probability density correlate well with the r-peaks of the ECG reference in the last panel of figure 3.19.

In order to provide a quantitative error estimate, we calculate the time difference between the peaks of the marginal beat probability density and compare them with the r-peak intervals of the ECG reference. The MCMC heartbeat detector achieved a mean error of 24 ms and a median error of 10 ms on a test dataset of 150 segments of BCG recordings with 10 s length [86]. Compared to the mean estimation error reported for state-of-the-art BCG heartbeat and beat interval detectors of 7 ms [104] and 13 ms [106], the error of the MCMC detection scheme are higher, but still in the same magnitude.

On the other hand, the MCMC based heartbeat detection introduced here is not intended to compete with state-of-the art beat detection methods. Instead, it is designed as a proof of concept for inference based beat detection. Thus, considering that the MCMC beat detection scheme mainly consists of general purpose software, with only the implementation of the specialized proposal function being specific to the heartbeat detection problem [86], the estimation error achieved by the MCMC scheme is a promising result. It indicates that with further specialization to the specific task at hand, methods based on statistical inference can be considered as an interesting alternative for BCG beat detection.

Additionally, caution is advised when comparing results across publications, since the detection error is subject-dependent and can vary substantially even between subjects within the same study. For example, the estimation error between the subjects reported in [104] differ by a factor of four to five and those reported in [106] differ by a factor of up to seven.

The reason for choosing MCMC as the inference method is based on the generality of the method and the simplicity of its implementation. MCMC can be used to invert any statistical model, as long as the posterior probability density (eq. (3.47)) can be evaluated up to a constant factor for any given point. This makes MCMC a very powerful and popular method, which can

even be applied to models where the posterior probability density cannot be written down in closed form. However, this flexibility comes at the cost of speed. In our example application, the MCMC scheme needs on average more than 100 s for detecting the heartbeats in a 10 s BCG segment. For an application like heartbeat detection, a method capable of online processing would be desirable. In the next chapter, we will encounter an alternative method for approximate inference called variational Bayes. In contrast to Monte Carlo, variational Bayes is an analytic approximation method which can be significantly faster than sampling from a distribution.

3.6 Summary

In this chapter, we have discussed a number of interrelated topics on modelling and signal processing for BCG data. Starting with a BCG separation scheme based on LPNR, we have developed a deterministic BCG model in order to generate artificial BCG data for validating the separation scheme. The availability of ground truth is one of the strongest advantages of synthetic data.

Using the result from the validated BCG separation, we were able to create a new, non-parametric and probabilistic model for the cardiac BCG component. This model allows us to apply powerful inference schemes, which offer the potential for finding solutions to more sophisticated signal processing tasks. As a proof of concept, we have designed a special MCMC scheme that allowed us to perform statistical heartbeat detection in the BCG, closing the cycle between modelling and signal processing.

In the course of the discussion, we have also seen that a modelling viewpoint can help to identify common concepts behind seemingly unrelated topics. This has lead us to the discovery that LPNR, which was developed in a nonlinear dynamics setting, can be reinterpreted as a non-parametric and non-probabilistic version of a mixture of PPCA model.

Mixture models are also an important topic in the next chapter, which introduces the embedded clustering framework. Although embedded clustering is applied to data from a different modality, i.e. functional magnetic resonance imaging, we will see that from a modelling perspective, there are many common threads and connections to the ideas presented in this chapter, including system description via differential equations, mixture modelling and statistical inference.

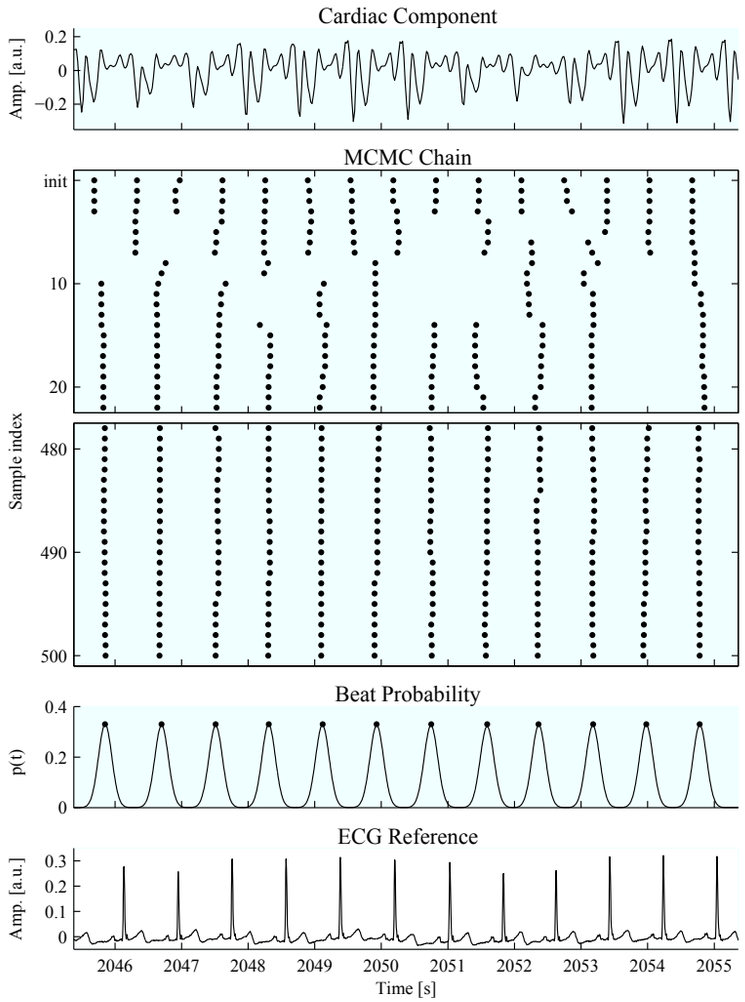


Figure 3.19: The MCMC scheme applied to the BCG (top panel) produces a chain of samples, of which the first 23 (second panel) and the last 23 (third panel) are shown here. Each sample, which consists of one heartbeat sequence, corresponds to one line, where the heartbeat positions are marked with black dots. The marginal beat probability density (fourth panel) is obtained via kernel density estimator. Black dots mark the peaks in the marginal beat probability density, which correlate well with the r-peaks of the ECG reference (bottom panel).

Chapter 4

Embedded Clustering

In this chapter, we focus on the second methodological aspect in this thesis, which is embedded clustering. In contrast to the previous chapter, we focus on the task of clustering instead of separation. Additionally, the applications we demonstrate the methods on shift from BCG based home monitoring to functional magnetic resonance imaging (fMRI) in the context of computational psychology. However, we will see that models still play a vital role in this chapter. Since clustering is an unsupervised learning technique, it lacks the user input that helps to interpret the data. This role has to be filled by a model, which determines on what criteria similarity between data points should be based.

The example modality in this chapter will be fMRI, which is analysed with dynamic causal modelling (DCM), a method for estimating the effective coupling strength between brain regions in the context of psychological experiments. The DCM framework includes a model comparison aspects, since different connectivity estimates correspond to different models of how brain regions communicate and interact with each other. This model selection problem lends itself to a treatment via the Bayesian framework described in section 2.5.1.

The embedded clustering problem, which is the main focus of this chapter, goes one step further. The aim is to cluster unknown brain region connectivity parameters of a group of subjects into subgroups. These connectivity parameters have to be estimated via DCM first. This introduces a second modelling aspect, since we have to build a clustering model, in addition to the DCM. DCM helps us to solve the inverse problem of estimating the connectivity parameters, which we then cluster with the help of the clustering model. In the embedded clustering concept, a unified model is build consisting of a DCM part modelling brain connectivity and another part

modelling the clusters. The two tasks of inverting the system model and perform the clustering are then solved jointly by inverting the embedded clustering model.

The clustering part in the embedded clustering framework introduces a second model selection problem which pertains the question of how many subgroups are present. As we will see, this problem is solved automatically when using the variational Bayes scheme to invert the embedded clustering model.

The structure of this chapter is as follows. Since the primary modality in this chapter is fMRI, we will begin with an introduction to the working principle of fMRI with focus on the blood-oxygenation-level dependent (BOLD) contrast. We will then introduce the two components of the embedded clustering framework in separate sections, beginning with DCM for fMRI and continuing with the mixture of Gaussians model, which is a popular choice for clustering problems. Lastly, we combine these two models in the embedded clustering framework and demonstrate its application in the context of multi-subject fMRI studies. Throughout this chapter, we will make heavy use of graphical models, which were introduced in section 2.3, to illustrate the structure of the models we use.

4.1 Functional Magnetic Resonance Imaging

In this section, we will provide a short introduction on functional magnetic resonance imaging (fMRI) via the blood-oxygenation-level dependent (BOLD) contrast. The fMRI BOLD contrast is the main modality to which the embedded clustering model discussed in this chapter is applied. Due to the vastness of this topic, we will only provide an overview of fMRI. For a comprehensive review, we refer the interested reader to text books such as [107] or [108].

Magnetic resonance imaging (MRI) is a very versatile modality, which can be used for both anatomical and functional imaging [108]. It is based on recording the signal of the precessing magnetization of certain atomic nuclei (mostly protons) in a static magnetic field, initialized by a radio frequency excitation. This effect of magnetic resonance has been known since the 1930s due to the work of Rabi *et al.* [109]. However, it was not until the 1970s before first ideas for using the magnetic resonance signal for imaging purpose were presented by Lauterbur and Mansfield [108].

Figure 4.1 show a schematic illustration of the basic principle of MRI. The main components of MRI are the static magnetic field ($\vec{B}_0$) used to align the

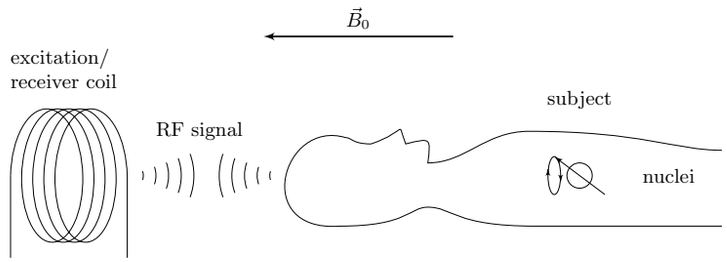


Figure 4.1: Basic principle of magnetic resonance imaging (MRI). The subject lies in a strong, static magnetic field ($\vec{B}_0$), which aligns the spins of protons in the subject's body. A radio frequency (RF) signal from the excitation coil is used to induce precession of the proton spins (here illustrated symbolically). The RF response caused by the decaying spins is measured with the receiver coil, which can be identical to the excitation coil

spins, as well as the radio frequency coil used to excite the spin precession and receive the response. In order to target a specific region of the subject, a so called slice, gradient coils are needed, which are omitted in figure 4.1 to keep the image uncluttered.

The first human MRI was demonstrated by Damadian *et al.* in 1977 [108] and subsequent improvements in imaging techniques, including the development of the echo-planar imaging technique, led to significant speed up of the image acquisition process. While the techniques used by Damadian *et al.* to demonstrate the feasibility of human MRI required imaging times of hours for a single relatively coarse image, modern scanners can acquire images at a speed of 20 slices per second [108]. As we will see, this is one of the prerequisites for functional imaging.

Modern MRI systems can be used to record the spatial density of atomic nuclei, e.g. protons, in the human body. Since proton density varies across tissue types, this can be used to obtain anatomical images of the subject. Alternatively, it is possible to record MRI images sensitive to relaxation properties of the signal. These images are called T_1, T_2 or T_2^* weighted images, where T_1 is the time constant for the recovery of the longitudinal magnetization and T_2 and T_2^* are time constants for the relaxation of the transversal magnetization [108]. These time constants are also tissue dependent, which allows recording of anatomical images.

Figure 4.2 shows two T_1-weighted, anatomical images acquired with MRI. The left image is acquired in the so called coronal view, which shows the brain from the front. The right image shows a sagittal slice showing the

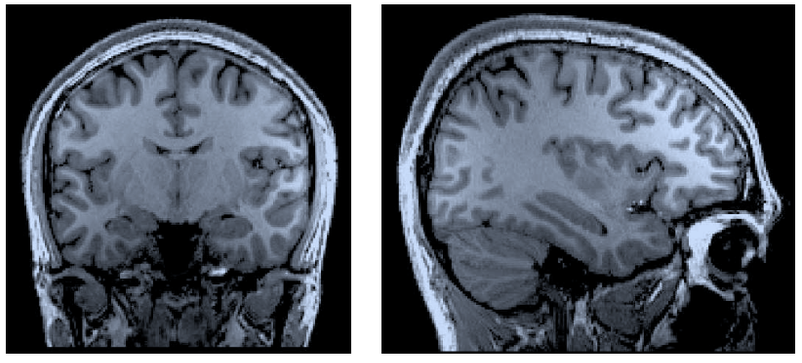


Figure 4.2: Two slices from an anatomical MRI scan. The image on the left side shows a coronal slice, i.e. the view from the front of the head. The image on the right shows a sagittal slice, i.e. the view from the side of the head. (Image courtesy of Johannes Lindemeyer, INM-4.)

brain from the side. As is typical for T_1-weighted MRI scans, white matter appears brighter than grey matter and the cerebrospinal fluid appears almost completely black.

The door to creating images of brain function was opened by the discovery of Ogawa *et al.* that the T_2^*-weighted MRI intensity of oxygenated and deoxygenated blood is different [110]. Since neuronal activity raises energy demand and the brain has no local energy storage [108], the increased rate of metabolism following neuronal activation causes a change in the concentration of oxygenated and deoxygenated haemoglobin in the capillaries surrounding the activated brain region. Deoxygenated haemoglobin has a smaller T_2^* time constant, which leads to a faster relaxation of the MR signal and thus to a darker image compared to oxygenated haemoglobin. Therefore, T_2^*-weighted MRI can be used to detect the change in deoxygenated haemoglobin content following neuronal activity. This type of functional imaging is called blood-oxygenation-level dependent (BOLD) contrast.

However, the exact change in MRI signal depend not only on the increased oxygen consumption induced by neuronal activity, but is also influenced by the dynamics of blood flow, which will increase oxygen supply as a reaction to increased activity. Thus, the time course of the MR signal follows complicated dynamics, where the increase in oxygen supply after neuronal activity lead to a delayed peak in MR signal intensity followed by an undershot. To allow for an interpretation of the BOLD signal, hemodynamic models are required and for this purpose, different theories have been proposed, among them the so called balloon model [111]. In this chapter, we will mainly rely

on the models presented in [112] and [113], which are derived from the balloon model. These models help us to resolve the relationship between the observed MRI signal, which is a function of the amount of oxygenated and deoxygenated haemoglobin, and the underlying brain activity.

The balloon model and related models describe hemodynamics by modelling the interactions among four hemodynamic state variables and the impact of the neuronal brain activity on these variables. The hemodynamic state variables include the vasodilatory signal, blood flow, blood volume and deoxyhaemoglobin content which are used to predict the fMRI signal. In simplified terms, the idea behind the balloon model is that the increase in blood flow triggered by neuronal activation causes a temporary increase in blood volume, since the increase in outflow lags behind the increase in inflow of oxygenated blood [111]. Thus, the venous compartment behind the activated region expands like a balloon filled with oxygenated blood, which accounts for the increase in MRI signal. After neuronal activity stops, the influx of oxygenated blood reverts to baseline. However, the excess blood volume, which now contains more deoxygenated blood, takes some time to reduce. This causes the undershot in fMRI intensity.

Mathematically, the interaction between the hemodynamic state variables is governed by a set of differential equations depending on so called hemodynamic parameters. Some of these parameters depend on the physiology of the subject, while others depend on the fMRI system, e.g. the field strength of the scanner. Empirically validated estimates of these parameters can be found in the fMRI literature; and in addition, these parameters can also be estimated from the fMRI data [36, 113]. Since the mathematical details of this model are discussed in depth in [112] and [113], we will omit the details of the model here and refer the reader to these two publications.

Since its inception, fMRI BOLD has become one of the standard modalities in functional MRI studies with many applications in psychology and neuroscience [108]. In the context of embedded clustering and DCM, the aim is to study brain region connectivity and to cluster subjects with respect to connectivity. The information supplied by fMRI BOLD serves as the data basis which, together with an algorithm for inverting the DCM model, allows us to obtain estimates for the effective connectivity between the brain regions of interest. Figure 4.3 shows samples from a simulated fMRI BOLD time series for two brain regions, which is used to verify the solution to the embedded clustering problem presented in the subsequent sections.

As embedded clustering requires a clustering step in addition to the DCM inversion, we will continue in the next sections by first discussing the basic ideas behind DCM and then introducing a class of models known as mixture models, which are used to solve the clustering problem.

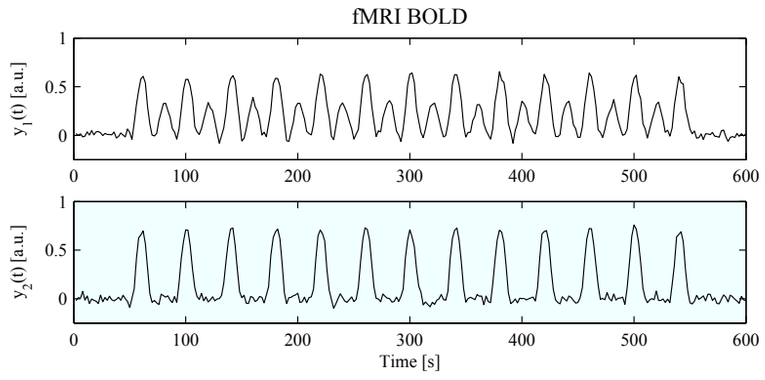


Figure 4.3: Samples of simulated fMRI BOLD time series for two brain regions of an imaginary subject.

4.2 Dynamic Causal Modelling

Dynamic causal modelling (DCM) is a modelling technique developed for inferring the effective coupling strength between brain regions and their modulation by external factors in the context of psychological experiments. Initially, it was designed to be applied to fMRI BOLD time series [88], which is also the version of DCM we will focus on in this chapter. Subsequently, it was also adapted for evoked responses in EEG and MEG [114].

The DCM model consists of two parts. The first part is a model of the brain regions involved in the experiment and the connections between those regions. The brain regions are modelled as the nodes of a graph connected by directed edges with weights representing the connections between the regions. In addition, the model contains inputs, which can be used to model experimental variables. These experimental variables can either represent stimuli presented to the subject, which have a direct influence on the activity in the brain regions or they can represent effects like attention, which influence the dynamics by changing the connectivity between brain regions [88].

Mathematically, each brain region is described by a variable x_r representing the neuronal activity in that region. The dynamics of the vector containing the neuronal activity of all regions $\boldsymbol{x} = (x_1, x_2, \ldots, x_R)^T$ is described by a differential equation, which depends on the inputs $\boldsymbol{u}$, as well as parameters $\boldsymbol{\theta}_c$ that describe the connection strength between the regions:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\theta}_c). \tag{4.1}$$

This equation means that the change in brain activity $\dot{\boldsymbol{x}}$ is given by a fixed

function $\boldsymbol{f}$ which depends on the current activity $\boldsymbol{x}$ and the time varying but known input $\boldsymbol{u}$. In addition, a set of fixed but unknown parameters $\boldsymbol{\theta}_c$ governs the behaviour of $\boldsymbol{f}$ and determines how exactly current state $\boldsymbol{x}$ and input $\boldsymbol{u}$ will influence the state evolution through $\boldsymbol{f}$ [88]. Note the similarity to equations (3.18) to (3.19), as well as equations (3.24) to (3.26) from section 3.5.1, which describe the dynamics of the nonlinear deterministic BCG model. Thus, both the nonlinear deterministic BCG model as well as DCM models, which come from entirely different areas of medical research, are based on the same structure, namely sets of differential equations that describe the dynamics of the systems.

The relationship in equation (4.1) has the most general form possible. However, in practice most applications use the so called bilinear approximation [88] to the function $\boldsymbol{f}$ given by:

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\theta}_c) \approx A\boldsymbol{x} + \sum_{j=1}^{J} u_j B^{(j)} \boldsymbol{x} + C\boldsymbol{u} \tag{4.2}$$

$$= \big(A + \sum_{j=1}^{J} u_j B^{(j)}\big)\boldsymbol{x} + C\boldsymbol{u}, \tag{4.3}$$

where the set of connection parameters $\boldsymbol{\theta}_c$ corresponds to the three matrices: $\boldsymbol{\theta}_c = \{A, B, C\}$. Here, the matrix C determines how the inputs $\boldsymbol{u}$ influence the state variable $\boldsymbol{x}$, which is also called the extrinsic influence on $\boldsymbol{x}$ [88].

The roles of A and B are more complex. Intuitively, A describes the intrinsic or resting state connectivity, i.e. how the states $\boldsymbol{x}$ influence themselves in the absence of inputs ($\boldsymbol{u} = 0$) [88]. B is a set of matrices containing one matrix per input: $B = \{B^{(1)}, B^{(2)}, \ldots, B^{(J)}\}$, with J being the number of inputs. Each of the $B^{(j)}$ can be interpreted as the differential connectivity with respect to input u_j [88]. Thus, $B^{(j)}$ encodes how the connectivity between regions change when input u_j is present. This interpretation of A and B is most evident from equation (4.3), which shows that the influence of $\boldsymbol{x}$ on $\dot{\boldsymbol{x}}$ is given by A in the absence of inputs. If inputs are present, the relation between $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ is modified by $u_j B^{(j)}$.

The second part of the DCM is the observation model, which models the modality used to observe the hidden neuronal brain activity. In the case of DCM for fMRI studies, this part is a hemodynamic model, which describes how the hidden neuronal activities $\boldsymbol{x}$ invoke the measured fMRI BOLD responses $\boldsymbol{y} = (y_1, \ldots, y_R)^T$, consisting of one time series per brain region. However, for EEG or MEG studies this part of the model can be easily swapped for an appropriate model of the EEG or MEG signals, since it is essentially independent from the first part. The standard observation model for fMRI BOLD, called the balloon model, was first proposed in [111] and

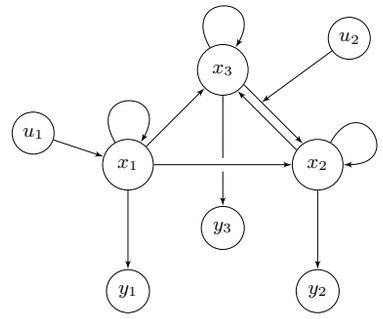


Figure 4.4: Example of a DCM with three interacting brain regions x_1 to x_3. y_1 to y_3 represent the fMRI time series of each region, while u_1 and u_2 are inputs that influence the regions directly or modify the coupling between regions.

subsequently refined and extended in [112] and [113].

The DCM model is often illustrated graphically as a network of nodes connected with edges. Figure 4.4 shows such a network which represents a simple example of a DCM with three interacting brain regions x_1 to x_3, which are observed through fMRI BOLD signals y_1 to y_3.

The edges from the x-nodes to the y-nodes represent the hemodynamic model [113]. On the other hand, the edges between the x-nodes represent elements of A and the edges from the u-nodes to the x-nodes represent elements of C. The elements of the $B^{(j)}$ correspond to the edges which points from the u-nodes to the edges between the x-nodes. Thus, for the model shown in figure 4.4 the connectivity parameters are:

$$A = \begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, \tag{4.4}$$

$$B^{(1)} = 0, \tag{4.5}$$

$$B^{(2)} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & b^{(2)}_{23} \\ 0 & 0 & 0 \end{pmatrix} \quad \text{and} \tag{4.6}$$

$$C = \begin{pmatrix} c_{11} & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}. \tag{4.7}$$

At this stage, we would like to point out two things. First, the kind of

graphical representation of DCMs shown in figure 4.4 is not a Bayesian network. Although the edges in the graph are directed, cycles are allowed in these kinds of graphs, i.e. one can move along the direction of the arrows and return to the node one started at. This makes sense, since the graph should reflect effective connectivity between brain regions and these can include feedback loops. However, attention should be paid not to confuse the graphical representation of DCMs with Bayesian networks, which we will use in the next sections.

Second, since brain activity cannot diverge to infinity, one would like to restrict the DCM model such that the intrinsic neuronal activity in the absence of inputs is guaranteed to stay finite. In a dynamical system, this corresponds to requiring the principal Lyapunov exponent of the system to be negative [8]. And for systems described by equation (4.2), it means that the largest real eigenvalue of A has to be negative. Unfortunately, there is no simple way to parameterize a matrix, such that its real eigenvalues are negative. When paying close attention to the graph in figure 4.4, one can notice that all x-nodes have self-connections. Such self-connections which are fixed to have negative values, are included in all DCM models, since Friston *et al.* showed that including negative self-connections, combined with restricting the amplitude of the remaining connections in relation to the self-connections, will limit the probability of getting a matrix A with positive eigenvalues [88]. Restricting the amplitude of the cross-connections can be done by using appropriate prior distributions [88].

Combining equation (4.1) with the hemodynamic model provides us with a deterministic forward model of neuronal activity and the related fMRI BOLD responses. The system described by DCM receives inputs $\boldsymbol{u}$, which encode experimental conditions or stimuli presented to the subject, and is governed by connectivity parameters $\boldsymbol{\theta}_c$ describing effective brain connectivity, as well as hemodynamic parameters $\boldsymbol{\theta}_h$ describing hemodynamic properties [88]. With these parameters, the model predicts, from the known inputs $\boldsymbol{u}$, the neuronal activity of the brain regions involved in the experiment and the corresponding fMRI bold response.

However, from the point of the neuroscientist, the neuronal activity $\boldsymbol{x}$ of the brain regions are not of primary interest. Instead, most experiments aim at estimating the unknown parameters $\boldsymbol{\theta}_c$ encoding the effective connectivity between regions, with the hemodynamic parameters $\boldsymbol{\theta}_h$ also being of interest. Thus, the form of the DCM model which we will encounter most often is the following:

$$\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{u}) + \boldsymbol{\eta}, \tag{4.8}$$

with $\boldsymbol{\theta} = (\boldsymbol{\theta}_c^T, \boldsymbol{\theta}_h^T)^T$ defined as the concatenation of $\boldsymbol{\theta}_c$ and $\boldsymbol{\theta}_h$. The function $\boldsymbol{g}$ denotes the entirety of the operations involved in evaluating the DCM

model to obtain the fMRI BOLD response. This starts with taking the input $\boldsymbol{u}$ and the connectivity parameters $\boldsymbol{\theta}_c$ and integrating equation (4.2) to obtain the neuronal activity $\boldsymbol{x}$. And it continues with integrating the hemodynamic equations that map the neuronal activity $\boldsymbol{x}$ onto the fMRI BOLD response $\boldsymbol{y}$. Since $\boldsymbol{x}$ is not of primary interest, it is not included in the output of $\boldsymbol{g}$. The variable $\boldsymbol{\eta}$ represents the measurement noise, which is assumed to be mean free and normal-distributed: $\boldsymbol{\eta} \sim \mathcal{N}(0, \Lambda^{-1})$ [88].

The precision matrix Λ is often assumed to be a diagonal matrix [115], with a main diagonal of the form $\lambda_1, \ldots, \lambda_1, \lambda_2, \ldots, \lambda_2, \ldots, \lambda_R, \ldots, \lambda_R$ and zeros elsewhere. Thus, Λ is parameterized by a set of real positive entries $\{\lambda_r\}_{r=1,\ldots,R}$, which denote the scalar noise precisions for each of the brain regions. Each of the entries λ_r appears multiple times in the main diagonal and we denote the number of times entry number r appears by q_r. If we assume $q_r = 2$ for all r, the noise precision matrix for the DCM shown in figure 4.4 is given by:

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \lambda_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & \lambda_3 \end{pmatrix}. \tag{4.9}$$

The number q_r corresponds to the number of samples in the fMRI time series for each region and for real datasets it is typically in the range of 1000. For use in later sections, it will be convenient to define Q_r as a matrix with the same structure as Λ, but with ones on the positions of λ_r in Λ and zeros elsewhere. With this definition, we can express the precision matrix compactly as:

$$\Lambda = \sum_{r=1}^{R} \lambda_r Q_r. \tag{4.10}$$

For the example in equation (4.9), Q_1 would be given by:

$$Q_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{4.11}$$

Thus, the inverse problem for DCM lies in reversing the $\boldsymbol{g}$-function to estimate the unknown connectivity and hemodynamic parameters $\boldsymbol{\theta}$ from the

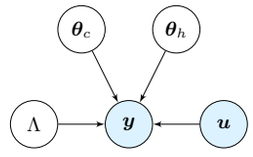


Figure 4.5: Graphical model showing the Bayesian network for the fMRI DCM model. For consistency with the embedded clustering model introduced in section 4.4, we have split the parameter vector $\boldsymbol{\theta}$ across two nodes: one for the connectivity parameters $\boldsymbol{\theta}_c$ and one for the hemodynamic parameters $\boldsymbol{\theta}_h$. The node representing the input $\boldsymbol{u}$ is optional.

observed fMRI BOLD time series $\boldsymbol{y}$. The parameter estimates answer questions about the connectivity among brain regions and their modulation by certain conditions in the experiment. Solutions to this inverse problem for single subject DCM models using expectation maximization [88] and variational Bayes [36] have already been proposed. These solutions have been adapted by several groups in the neuroscience community and are included in an open source software package [116].

Figure 4.5 shows the graphical model for DCM. Note that this is the Bayesian network representing DCM, while the graph in figure 4.4 is a graphical illustration of the connectivity between the brain regions and the influence of the inputs. In the Bayesian network shown here, the signal generation process, starting from the inputs $\boldsymbol{u}$ and ending with the fMRI BOLD response $\boldsymbol{y}$, is summarized in the edges pointing from $\boldsymbol{\theta}$ and $\boldsymbol{u}$ towards $\boldsymbol{y}$, which represent the $\boldsymbol{g}$-function. Since the input $\boldsymbol{u}$ is part of the experiment design, it is an observed variable, which is represented in figure 4.5 by shading the respective node. In addition, the node representing the input $\boldsymbol{u}$ in this graphical model is considered to be optional and is often left out.

It might be worth noting that the structure of the DCM model consisting of a combination of a state evolution model (eq (4.2)) with an observation model (hemodynamic model) is a very common theme in statistical modelling. Other examples for well-known models with this structure include Markov random fields and the Markov chain which was introduced in section 2.4.1 in the context of Markov chain Monte Carlo.

In addition, estimating the values of the A, B and C matrices can be viewed as a model selection problem, since different values for the connectivity between the regions correspond to different models of how the brain functions in the given experiment. Thus, inverting a DCM is closely related to the model comparison problem introduced in section 2.5.1.

4.3 Mixture Models for Clustering

In section 3.3, we introduced the mixture of PPCA as a generalization of the online LPNR algorithm. The mixture of PPCA is a member of a class of models called mixture models. In general, a model defines a probability density function over the space of objects we want to model. In doing so, it has to achieve a trade-off between accuracy and model complexity. Typically, we would like to be able to describe the probability density with simple analytic functions. On other hand, the model should be able to reproduce the real probability density over the objects of interest with sufficient accuracy.

One strategy, to achieve this trade-off is to build a complex model by superposing a number of simple models. This strategy, called mixture modelling, is characterized by probability density functions of the form

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k p_k(\boldsymbol{x}|\boldsymbol{\theta}_k), \tag{4.12}$$

where $\boldsymbol{x}$ is the variable we would like to model, π_k is the weight of the k-th component, $p_k(\cdot)$ is the probability density defined by the k-th component and K is the number of components. Typically, the mathematical form of the components is identical and the distinguishing factors are the different parameters sets $\boldsymbol{\theta}_k$.

There are different motivations for using mixture models. One reason is that the probability density we would like to model is too complex to be described by a single analytic function. By using a mixture model, we can attempt to break up the $\boldsymbol{x}$-space over which $p(\boldsymbol{x})$ is defined into several parts, and use simple analytic component to model each part. This approach is often encountered in density modelling applications and the mixture of PPCA model, in its role as a generalization of the online LPNR scheme (section 3.3), falls into this category. Another example is the application of mixture of PPCA to modelling points distributed on a sphere [81].

In the case described above, using a mixture model is only a convenient way of specifying a complex model from simple building blocks and the components do not have a sensible interpretation or carry any abstract meaning. However, in some applications dividing the data space does reflect an underlying structure of the problem. This is often the case for clustering problems, where the assumption is made that data points belong to different categories, with each category being represented by one model component.

In this context, mixture models are often interpreted as latent variable models: Each data point is associated with a discrete component label which is unobserved, i.e. it is a latent variable drawn from a distribution

$p(d=k|\boldsymbol{\pi}) = \pi_k$. Given the component label d, the observed data point is generated from the simple model specified by the respective component: $p(\boldsymbol{x}|d=k) = p_k(\boldsymbol{x}|\boldsymbol{\theta}_k)$. And since the component label is unobserved, the marginal distribution over $\boldsymbol{x}$ is given by the superposition of the probability density of all the components:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} p(d=k|\boldsymbol{\pi}) p(\boldsymbol{x}|d=k) \tag{4.13}$$

$$= \sum_{k=1}^{K} \pi_k p_k(\boldsymbol{x}|\boldsymbol{\theta}_k). \tag{4.14}$$

Note that to derive the above relation, we have used the sum (eq (2.7)) and product rules (eq (2.9)) introduced in section 2.1. In this context, the model weights π_k in equation (4.12) are interpreted as the probability of the data being generated from the k-th component.

For the components in mixture modelling, different models can be used. If $\boldsymbol{x}$ is a counting variable defined over the space of integers or as a vector of integers, the components $p_k(\boldsymbol{x})$ are typically chosen to be multinomial distributions. If $\boldsymbol{x}$ is continuous, a popular choice for $p_k(\boldsymbol{x})$ is the Gaussian distribution, which leads to the well-known mixture of Gaussians model. The Gaussian mixture model is also our choice for the clustering part in the embedded clustering framework for DCM parameters. Therefore, we will dedicate the next section to an in depth discussion of the mixture of Gaussians model in the context of clustering applications, as well as inference schemes which are often used for learning the parameters of Gaussian mixtures.

As a side note, we would like to point to an interesting connection between mixture models and the single PPCA model, which both belong to the same model family. Equation (4.14) shows us that mixture models can be interpreted as latent variable models where the latent variable is discrete. However, in section 3.3, we pointed out that the PPCA model can also be viewed as a latent variable model [17, 81], with the difference that the latent variable in PPCA is continuous. Thus, mixture models and PPCA are both latent variable models, with the difference that a mixture model is a discrete latent variable model and PPCA is a continuous latent variable model. On the other hand, this also means that the mixture of PPCA model is a two-fold latent variable model: On the mixture level it has a discrete latent variable, while on the component level it consists of continuous latent variable models.

4.3.1 Mixture of Gaussians

This section will introduce the mixture of Gaussians model with a focus on applications to clustering problems. We will begin with an introduction to the general structure of the mixture of Gaussians model, followed by a discussion of how to learn the parameters of the model from measurements using the variational Bayes scheme introduced in section 2.4.2. In the course of this discussion, we cannot avoid coming across a few mathematical details of the variational approximation scheme for mixtures of Gaussians. However, we deem this to be a necessary evil for the understanding of the overall topic, since the mixture of Gaussians is an important component of the embedded clustering scheme.

The clustering problem belongs to the category of unsupervised learning, which deals with the task of discovering structure in unlabelled data. Other unsupervised learning problems include density estimation, which we encountered in the discussion on mixture of PPCA in section 3.3.

In clustering, the assumption is that there are unobserved categories or clusters underlying the data generation process. The aim is to assign the observed data points to the clusters and at the same time estimate the cluster parameters. For data points $\boldsymbol{x}$ defined in a continuous space, the mixture of Gaussians is a popular model, which is often used for the clustering problem [17, 37].

Structure of Gaussian Mixture Models

In a mixture of Gaussians model, the components of the mixture are Gaussian distributions, with probability distribution given in equation (A.26):

$$p_k(\boldsymbol{x}) \quad = \quad \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \Sigma_k). \tag{4.15}$$

The Gaussian distribution is a popular choice for modelling continuous variables because it has several nice properties. It has a well-studied analytical form, and the central limit theorem states that the sum of a set of independent random variables with finite variance tend towards a Gaussian distribution with increasing number of summands [23, 24].

The generative process of the mixture of Gaussians model is as follows. First, a value for the component label d is drawn according to a categorical distribution: $p(d = k|\boldsymbol{\pi}) = \pi_k$. Here, $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)^T$ is the vector of component probabilities. Then, the value of d is used to pick from one of K different Gaussian distributions. The chosen component is used to generate the data point $\boldsymbol{x}$. Thus, the model equation is given by plugging

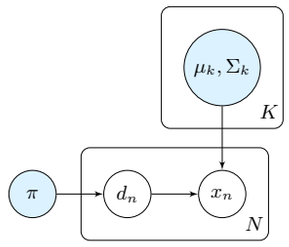


Figure 4.6: A graphical model illustrating the generative process of the mixture of Gaussians model. Note that for the generative process, the parameters are assumed to be fixed to known or chosen values, indicated by shading.

equation (4.15) into the general mixture model distribution given by equation (4.14):

$$p(\boldsymbol{x}|\boldsymbol{\pi}, \{\mu_k, \Sigma_k\}_{k=1,\ldots,K}) = \sum_{k=1}^{K} p(d{=}k|\boldsymbol{\pi}) p_k(\boldsymbol{x}|\boldsymbol{\mu}_k \Sigma_k) \qquad (4.16)$$

$$= \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \Sigma_k). \qquad (4.17)$$

Viewed from the point of the whole mixture model, the component density $p_k(\boldsymbol{x})$ from equation (4.15) is equivalent to the conditional distribution of the data given the cluster label: $p_k(\boldsymbol{x}) = p(\boldsymbol{x}|d{=}k)$. And as already mentioned in the beginning of this section, the cluster weight π_k is equivalent to the prior probability of the respective component: $p(d{=}k) = \pi_k$.

Equation (4.17) corresponds to the likelihood function with cluster label d marginalized out. This function describes the probability of the data point $\boldsymbol{x}$ given the parameters of the model, but without knowledge of the labels. Here, the parameters are the mean and covariance matrix for each Gaussian component $\{\boldsymbol{\mu}_k, \Sigma_k\}_{k=1,\ldots,K}$, as well as the cluster weights $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)^T$, which are constrained to sum to one: $\sum_{k=1}^{K} \pi_k = 1$.

Figure 4.6 illustrates the generative process of the mixture of Gaussians model with a graphical model. The parameters are shaded, since for the generative process they are considered to be fixed to known values. Here, we have used the plate notation as a compact way to denote that there are multiple components (K) and multiple realizations of data points (N). Note the similarity to the graphical model of the mixture of PPCA in figure 3.8.

Learning the Parameters

In practice, the parameters are unknown, just like the cluster labels, and the aim is to estimate them from a number of observed data points $\{\boldsymbol{x}_n\}_{n=1,\dots,N}$. With the Bayesian approach to statistical inference, learning the parameters is equivalent to inverting the model in order to obtain the posterior probability distribution, i.e. the conditional probability of the parameters given the data: $p(\boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}_{k=1,\dots,K} | \{\boldsymbol{x}_n\}_{n=1,\dots,N})$.

As introduced in section 2.2, the posterior distribution can be obtained using Bayes rule (eq (2.10)), which for the mixture of Gaussians model is given by:

$$p(\boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\} | \{\boldsymbol{x}_n\}) = \frac{p(\{\boldsymbol{x}_n\} | \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) p(\boldsymbol{\pi}) p(\{\boldsymbol{\mu}_k, \Sigma_k\})}{p(\{\boldsymbol{x}_n\})}. \tag{4.18}$$

Note that in the above equation, we have dropped the subscript by using the shortcut $\{\boldsymbol{x}_n\}$ to denote the set of all data points $\{\boldsymbol{x}_n\}_{n=1,\dots,N}$, in order to avoid cluttering the notation. In the following, we will always use this notation, whenever it is clear from the context that we mean a set of variables.

The term $p(\{\boldsymbol{x}_n\} | \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\})$ is the likelihood function, which is given in equation (4.17) for a single data point. For multiple data points the likelihood function is given by the product of the likelihood of each of the data points, under the assumption that the data points have been drawn independently from the same distribution. This so called iid. assumption (independent and identically distributed) is a very common assumption in statistical inference, which leads to the following form for the likelihood of a set of observations:

$$p(\{\boldsymbol{x}_n\} | \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \Sigma_k). \tag{4.19}$$

A common practice in statistics is to work with the logarithm of the likelihood, the so called log-likelihood:

$$\begin{aligned} \log p(\{\boldsymbol{x}_n\} | \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) &= \log \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \Sigma_k) \\ &= \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \Sigma_k). \end{aligned} \tag{4.20}$$

The other terms in the numerator of equation (4.18) are the prior distributions of the parameters. These distributions encode model assumptions and

enforce constraints on the parameters. For example, the values in $\boldsymbol{\pi}$ have to be positive and sum to one, since they encode cluster probabilities. Therefore, it is common practice to define $p(\boldsymbol{\pi})$ as a Dirichlet distribution [17, 37], which is a distribution defined over the space of probability vectors (see appendix A.3).

The prior distribution over cluster parameters $p(\{\boldsymbol{\mu}_k, \Sigma_k\}_{k=1,\dots,K})$ is a distribution over $\boldsymbol{\mu}_k$, which is a vector, and Σ_k, which is a covariance matrix. For convenience, one typically picks a prior distribution, which factors across clusters:

$$p(\{\boldsymbol{\mu}_k, \Sigma_k\}_{k=1,\dots,K}) = \prod_{k=1}^{K} p(\boldsymbol{\mu}_k, \Sigma_k). \tag{4.21}$$

This way the variational distribution $q(\{\boldsymbol{\mu}_k, \Sigma_k\}_{k=1,\dots,K})$ will also factor across clusters [17, 37].

For the multiplicands in the product $p(\boldsymbol{\mu}_k, \Sigma_k)$, which are the prior distributions of the parameters of each cluster, the standard choice is the normal-inverse Wishart distribution:

$$p(\boldsymbol{\mu}_k, \Sigma_k) = \mathcal{N}(\boldsymbol{\mu}_k | \boldsymbol{\mu}_0, \Sigma_k/\tau_0)\, \mathcal{W}^{-1}(\Sigma_k | \Sigma_0, \nu_0). \tag{4.22}$$

This distribution consists of a Gaussian distribution over $\boldsymbol{\mu}_k$, which depends on Σ_k and an inverse Wishart distribution over Σ_k, which is a distribution defined over symmetric, positive definite matrices. Thus, the inverse Wishart distribution helps to enforce that Σ_k is a valid covariance matrix.

The parameters of the prior distributions are called hyper-parameters. In this case, they are $\boldsymbol{\alpha}_0$ for the Dirichlet distribution and $\boldsymbol{\mu}_0, \tau_0, \Sigma_0$ and ν_0 for the normal-inverse Wishart distribution. Through the value of the hyper-parameters, one can encode certain model assumptions. E.g. when defining $\boldsymbol{\alpha}_0$ as a vector with large ($\alpha_{0,k} \gg 1$) but equal elements, the model will favour an outcome where the cluster weights tend to be similar for all clusters in the mixture. On the other hand, one can encode the assumption that one cluster might dominate the others by setting the elements of $\boldsymbol{\alpha}_0$ to values much smaller than one ($0 < \alpha_{0,k} \ll 1$). Alternatively, one can also specify a flat prior ($\alpha_{0,k} = 1$), which does not favour any of the extreme cases described before [17].

In addition to helping us enforce constrains and encode model assumptions, the above choices for the prior distributions have the special property of being so called conjugate priors. These are prior distributions which have a convenient mathematical form allowing the posterior distribution to be described by the same functional form as the prior distribution [17, 117]. We will see what this means in detail when we discuss the variational Bayes scheme for embedded clustering in section 4.5.

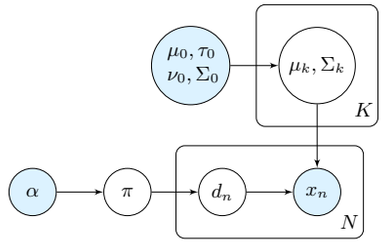


Figure 4.7: Graphical model for the inference for mixture of Gaussians models. In contrast to figure 4.6, the graphical model shown here contains parent nodes for the parameter variables $\boldsymbol{\pi}$, $\boldsymbol{\mu}_k$ and Σ_k, representing the prior distributions. Additionally, the data variable $\boldsymbol{x}_n$ is shaded to indicate that they are observed, while the parameters are unobserved.

Figure 4.7 illustrates the graphical model of the inference process for the mixture of Gaussians model. Again, we have used plate notation to denote the existence of multiple components (K) and multiple observations (N). In contrast to the graphical model shown in figure 4.6, which describes the process of generating data, this graphical model illustrates the inference of the model parameters from observed data. Thus, the data variables $\boldsymbol{x}_n$ are now shaded to indicate that they are observed and the parameters $\boldsymbol{\pi}$, $\boldsymbol{\mu}_k$ and Σ_k are unknown. In addition, this graphical model includes the prior distributions indicated by the parent nodes of the parameter variables, which represent the hyper-parameters $\boldsymbol{\mu}_0$, τ_0, ν_0, Σ_0 and $\boldsymbol{\alpha}_0$. These hyper-parameters are known, since they are chosen prior to inference, in order to encode model assumptions and enforce constraints on the parameters.

The only term remaining is the evidence in the denominator of equation (4.18), which according to the sum and product rules (eq (2.7) and (2.9)) can be obtained by integrating over the product of likelihood and prior:

$$p(\{\boldsymbol{x}_n\}) = \int_{\boldsymbol{\pi},\{\boldsymbol{\mu}_k,\Sigma_k\}} p(\{\boldsymbol{x}_n\}|\boldsymbol{\pi},\{\boldsymbol{\mu}_k,\Sigma_k\})p(\boldsymbol{\pi},\{\boldsymbol{\mu}_k,\Sigma_k\})d\boldsymbol{\pi}d\{\boldsymbol{\mu}_k,\Sigma_k\} \quad (4.23)$$

$$= \int_{\boldsymbol{\pi},\{\boldsymbol{\mu}_k,\Sigma_k\}} \prod_{n=1}^{N} \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\Sigma_k) \right) \times \mathcal{D}(\boldsymbol{\pi}) \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{\mu}_k)\, \mathcal{W}^{-1}(\Sigma_k) d\boldsymbol{\pi}d\{\boldsymbol{\mu}_k,\Sigma_k\}. \quad (4.24)$$

The evidence corresponds to the probability of the observed data given the current model, with the parameters marginalized out. As mentioned in section 2.5.1, this term is named evidence or model evidence because it is

used for model comparison and model selection. In the context of mixture modelling, the model selection problem is related to the question of how to choose K, i.e. how many clusters the mixture needs to have in order to best explain the observed data. As we will see later, the advantage of the Bayesian approach to model fitting is that it will automatically achieve a trade-off between fitting the data and limiting the complexity of the model by optimizing the model evidence with respect to K [17, 117].

Approximate Inference for Gaussian Mixtures

The problem with equation (4.23) is that the integration needed for evaluating the model evidence is analytically intractable. For the mixture of Gaussians, the problem lies in the summation inside the product under the integral in equation (4.24), which prevents us from factoring the integrand and integrating over each factor separately. In fact, this problem applies to all mixture models and is not limited to mixture of Gaussians [37]. Thus, in order to obtain estimates of the model evidence, we need to resort to approximation schemes, which can be divided into two categories. On the one hand, we have Monte Carlo methods, which were introduced in section 2.4.1 and which we have already encountered in section 3.5.2 in the context of inference in the Gaussian process BCG model. Alternatively, there is the variational Bayes approximation introduced in section 2.4.2, which we will focus on in the following sections.

As introduced in section 2.4, the difference between Monte Carlo and variational Bayes is that Monte Carlo is a numerical approximation method, which works by drawing a number of samples from a target distribution and using these samples to calculate estimates. Variational Bayes, on the other hand, relies on optimizing an analytic bound to the evidence called free energy. In general, the free energy approach is much faster, but it lacks the asymptotic exactness, which distinguishes Monte Carlo methods.

The details of the variational Bayes scheme, which aims at deriving a lower bound to the model evidence, are described in section 2.4.2. The first step in this process is to distinguish between the set of observed variables X, which in the case of the mixture of Gaussians are the set of data points $\{\boldsymbol{x}_n\}$, and the set of unobserved variables Z, which consists of the cluster labels d_n and the model parameters $\boldsymbol{\pi}$ and $\{\boldsymbol{\mu}_k, \Sigma_k\}_{k=1,\ldots,K}$ [17].

In the next step, we have to choose a factorization by dividing the set Z into subsets $Z_1, Z_2, \ldots$ and approximating each subset with its own variational distribution $q(Z_1), q(Z_2), \ldots$. This will cause the variational distribution over Z to factor into several components $q(Z) = q(Z_1)q(Z_2)\cdots$, which sim-

plifies the evaluation of the free energy bound (eq (2.28)). For the mixture of Gaussian model, one typically assumes a factorization between the set of cluster labels $Z_1 = \{d_n\}$ and the model parameters $Z_2 = \{\boldsymbol{\pi}, \{\mu_k, \Sigma_k, \}\}$ [17].

To optimize the free energy bound, one can in principle use any optimization algorithm. However, for the mixture of Gaussians model, a popular choice is to use the iterative update scheme proposed in [33], which is closely related to the expectation maximization (EM) algorithm [118]. The EM algorithm is a method for estimating model parameters by maximizing the likelihood function (eq (4.19)) instead of the free energy. This will provide point estimates of the model parameters. In contrast, the variational Bayesian approach provides an estimate for the posterior distribution over the parameters, as well as a lower bound on the model evidence, which allows us to determine the optimum number of mixture components by optimizing the lower bound with respect to K. Details on the variational update equations for the mixture of Gaussians model, which can be viewed as an extension of the EM update equations, are given in [17].

Summary

In this section, we have discussed how to learn the parameters of a Gaussian mixture model from observed data using the variational Bayes approach. However, in the embedded clustering problem the data corresponds to the DCM parameters, which cannot be observed directly. This means that in order to solve the embedded clustering problem, we have to estimate the DCM parameters while performing the clustering. In the next section, we will show how to define a unified model for DCM and clustering in the embedded clustering framework, and invert the model using a variational Bayes approach. In this discussion, the reader will notice that much of the embedded clustering framework can be viewed as an extension of the mixture of Gaussians model.

Analogous to the discussion from section 3.3 on non-parametric and parametric models in the context of the LPNR algorithm, we should point out that the mixture of Gaussians model discussed in this section is a parametric method, just like the mixture of PPCA. This is because the number of Gaussian components in the mixture is fixed in advance. However, there exists non-parametric extensions of mixture modelling to infinite mixtures, i.e. mixture models where the number of components is not limited in advance. These non-parametric mixture models are based on the Dirichlet process [119], and although we will not deal with them in this thesis, we would still like to point out that these kinds of models would occupy the lower right quarter of the diagram in table 3.1.

4.4 Embedded Clustering for DCM

In this section, we will introduce the embedded clustering framework for dynamic causal modelling (DCM). This framework combines the DCM described in section 4.2 with the mixture of Gaussians model for clustering from section 4.3. In the following, we will first explain the motivation for embedded clustering and then derive the model equations.

Throughout the section, we will continue following the simplified notation introduced in the last section, which involves dropping the subscript when it is clear from the context that we are referring to a set. Thus, $\{\boldsymbol{y}_n\}$ denotes the set of fMRI measurements from all subjects $\{\boldsymbol{y}_n\}_{n=1,\dots,N}$ and similarly $\{\boldsymbol{\mu}_k, \Sigma_k\}$ is the set of cluster parameters for all clusters from 1 to K. $\boldsymbol{\mu}_k$, on the other hand, denotes the mean of cluster number k, only.

4.4.1 Motivation

The motivation for embedded clustering arises from the need to extend the DCM framework to multi-subject studies. The task of embedded clustering within the context of such a study is to identify subgroups within the subject population. In the simplest case, this corresponds to distinguishing between patients and healthy control subjects. However, in more sophisticated scenarios this task can also include distinguishing between different diseases or between subtypes of the same disease [115].

The straight forward way of solving this problem consists of obtaining DCM parameter estimates for each subject separately. Methods for inverting single subject DCM [36, 88] are well established and available as open source software [116]. If the inversion scheme for the single subject DCM provides a posterior distribution over the parameter space, point estimates for the parameters $\hat{\boldsymbol{\theta}}_n$ can be obtained by choosing the maximum a posteriori (MAP) solution $\boldsymbol{\theta}_{n,MAP}$, i.e. the parameter that maximizes the posterior probability:

$$\hat{\boldsymbol{\theta}}_{n,MAP} = \arg\max_{\boldsymbol{\theta}_n} p(\boldsymbol{\theta}_n|\boldsymbol{y}_n). \tag{4.25}$$

Alternatively, one can also obtain point estimates by maximizing the likelihood function with respect to the parameters, which yields the so called maximum likelihood (ML) solution:

$$\boldsymbol{\theta}_{n,ML} = \arg\max_{\boldsymbol{\theta}_n} p(\boldsymbol{y}_n|\boldsymbol{\theta}_n). \tag{4.26}$$

These point estimates for the DCM parameters of all subjects $\{\hat{\boldsymbol{\theta}}_n\}_{n=1,\dots,N}$ can be treated as the observation $\boldsymbol{x}_n$ in a clustering model like mixture of

Gaussians, or as inputs to a classification algorithm, irrespective of the way they are obtained. Thus, the subjects are clustered or classified based on the point estimates of the DCM parameters.

This approach has already been tested with success in [120] with data from an fMRI study, where DCM parameter estimates where classified using an algorithm called support vector machine. However, there are also a few disadvantages associated with this approach. Firstly, by using a point estimate of the DCM parameters, one disregards the information about the reliability of the parameter estimate, which is encoded in the posterior distribution. Thus, the clustering or classification algorithm in the second step cannot take the uncertainty of the parameter estimates from the first step into account. In addition, the parameter estimation will not benefit from the results of the clustering or classification process.

The alternative to the stepwise approach of first obtaining point estimates and then performing clustering or classification on the point estimates is to build a unified model for joint DCM inversion and clustering, where the clustering process is integrated into a multi-subject DCM model. By inverting the unified model, which will be described in the following, one performs the DCM parameter estimation and the clustering simultaneously. In addition to avoiding the drawbacks mentioned above, this method holds the advantage that applied to the single subject scenario, it can be considered as an empirical Bayes method, since it integrates out the parameters of the prior distribution on the DCM parameters [115]. In the following, we will derive the joint probability distribution of the embedded clustering model.

4.4.2 Model Equations

The unified model for DCM inversion and clustering is obtained by placing the mixture of Gaussians model over the connectivity parameters of the DCM model, with separate prior distributions over the hemodynamic parameters and the precision matrix of the measurement noise. The model can be best explained with a graphical representation, which is shown in figure 4.8.

It is easily noticed that the Bayesian network representation of the embedded clustering model contains both the graphical model for DCM from figure 4.5, as well as the graphical model for the mixture of Gaussians from figure 4.7. Here, the connectivity parameter $\boldsymbol{\theta}_c$ has taken the place of the data variable $\boldsymbol{x}$ in the Gaussian mixture part and the entire DCM part of the model is placed inside the N-plate, in order to indicate that there is one DCM for each subject.

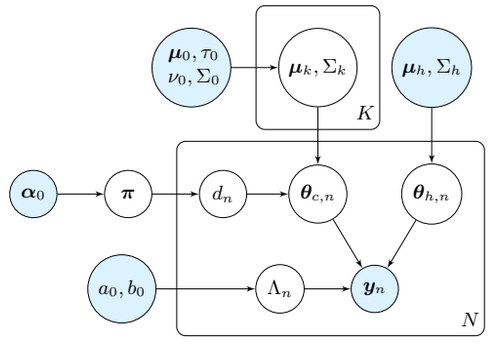


Figure 4.8: Graphical model of the embedded clustering model. This model contains both the graphical model for DCM (fig 4.5) and the graphical model for the mixture of Gaussians (fig 4.8).

One of the advantages of Bayesian networks is that the joint distribution can be easily read off from the graphical model. Thus, based on figure 4.8, the joint probability distribution over the observed data variable and the unobserved parameters and latent variables of the embedded clustering model is given by:

$$p(\{\boldsymbol{y}_n, d_n, \boldsymbol{\theta}_n, \Lambda_n\}, \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) = \prod_{n=1}^{N} \Big(p(\boldsymbol{y}_n|\boldsymbol{\theta}_n, \Lambda_n) \times p(\boldsymbol{\theta}_{c,n}|d_n, \{\boldsymbol{\mu}_k, \Sigma_k\}) p(d_n|\boldsymbol{\pi}) p(\boldsymbol{\theta}_{h,n}|\boldsymbol{\mu}_h, \Sigma_h) p(\Lambda_n|a_0, b_0) \Big) \times \prod_{k=1}^{K} \Big(p(\boldsymbol{\mu}_k, \Sigma_k|\boldsymbol{\mu}_0, \tau_0, \nu_0, \Sigma_0) \Big) p(\boldsymbol{\pi}|\alpha_0). \quad (4.27)$$

Similar to the mixture of Gaussians model, we have made the assumption that the data variable $\boldsymbol{y}_n$ is independent and identically distributed (iid.) for each of the subjects. Additionally, we also use the same prior for the cluster mean $\boldsymbol{\mu}_k$ and covariance Σ_k as in the mixture of Gaussian case, which factors across clusters. This is reflected in the plate notation in the graphical model, where the data and the cluster parameters are contained in their own plate, as well as in equation (4.27), where the distribution over the data and the cluster parameters factors into their own product over n and k, respectively. The plate notation has been introduced in section 2.3.

The graphical model shown in figure 4.8 already includes the prior distributions and their functional form is chosen analogous to the regular mixture of Gaussians model, as far as the mixture modelling part is concerned. Thus, the prior for the mixture weights is given by a Dirichlet distribution with

hyper-parameter $\boldsymbol{\alpha}_0$,

$$p(\boldsymbol{\pi}|\boldsymbol{\alpha}_0) = \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{\alpha}_0), \tag{4.28}$$

while the prior for the mixture parameters $\boldsymbol{\mu}_k$ and Σ_k is given by a normal-inverse Wishart distribution with hyper-parameters $\boldsymbol{\mu}_0$, τ_0, ν_0 and Σ_0 (see equation (4.22)).

The hemodynamic parameter vector $\boldsymbol{\theta}_h$, on the other hand, inherits its Gaussian prior distribution from the original formulation of the DCM model [36]:

$$p(\boldsymbol{\theta}_{h,n}|\boldsymbol{\mu}_h, \Sigma_h) = \mathcal{N}(\boldsymbol{\theta}_{h,n}|\boldsymbol{\mu}_h, \Sigma_h). \tag{4.29}$$

In the single subject DCM, the model would have an additional Gaussian prior on the connection parameters $\boldsymbol{\theta}_c$, just like the one for $\boldsymbol{\theta}_h$. In the embedded clustering model however, this role is taken by the mixture of Gaussians part of the model, which has become the parent node of $\boldsymbol{\theta}_c$ in figure 4.8:

$$p(\boldsymbol{\theta}_{c,n}|d_n, \{\boldsymbol{\mu}_k, \Sigma_k\}) = \mathcal{N}(\boldsymbol{\theta}_{c,n}|\boldsymbol{\mu}_{d_n}, \Sigma_{d_n}). \tag{4.30}$$

Due to the parameterization of the precision matrix Λ_n introduced in section 4.2 (eq (4.9)), it is sufficient to work with distributions over the set of diagonal elements $\{\lambda_{r,n}\}_{r=1,\ldots,R}$. As with the other variables, the prior distribution for the noise precisions also factors across subjects. However, we will introduce an addition factorization across brain regions r:

$$p(\{\Lambda_n\}) = \prod_{n=1}^{N} p(\Lambda_n) \tag{4.31}$$

$$= \prod_{n=1}^{N} \prod_{r=1}^{R} p(\lambda_{r,n}), \quad \text{with} \tag{4.32}$$

$$p(\lambda_{r,n}) = \text{Gam}(\lambda_{r,n}|a_{r,0}, b_{r,0}). \tag{4.33}$$

The distribution over each of the factors $p(\lambda_{r,n})$ is chosen as a gamma distribution (see appendix A.3). In this respect, we deviate from the original formulation of the single subject DCM model, which uses a log-normal prior distribution. This decision is based on the observation that for $\lambda_{r,n}$, the Gamma distribution is a conjugate prior, while the log-normal distribution is not. As we will see in section 4.5, choosing the conjugate prior will lead to a very simple form of the update equations, as the variational distribution $q(\lambda_{r,n})$ will also be a Gamma distribution.

Just like in the case of the mixture of Gaussians model, the inversion of the embedded clustering model is analytically intractable. And just like in section 4.3.1, we have the choice between two fundamentally different solutions to this problem: numerical approximations using Monte Carlo based

methods and analytic approximations using variational methods. A solution based on Markov chain Monte Carlo (MCMC), which we already encountered in section 3.5.2 in the context of BCG heartbeat detection, is presented in [115]. The advantage of this approach is that it is a flexible and very powerful method, which at the same time is easy to implement. Additionally, like all Monte Carlo methods, MCMC possesses the asymptotical exactness property, making it an attractive choice if high accuracy is needed. However, the major disadvantage of MCMC is its slow speed [28].

On the other hand, the variational Bayes approximation lacks the asymptotical exactness property of Monte Carlo methods, which means that even after reaching the maximum of the free energy function the approximation error will not vanish. However, variational methods are typically much faster than numerical approximations based on Monte Carlo and the difference in running time can span several magnitudes. Possible application scenarios for a variational solution to the embedded clustering problem could include doing quick preliminary analysis on data from fMRI studies with moderate computational resources. The results of the preliminary analysis can be used to suggest which options are most promising for further analysis. Among other possibilities, this can include picking better starting points for the Markov chain to speed up the MCMC method or to conduct a preselection among possible models, which are then investigated in more detail using MCMC.

In the following section, we will derive the variational update equations for the embedded clustering model, which we introduced above. This includes deriving expressions for updating the parameters of the variational distributions $q(Z)$, as well as an expression for the free energy of the model, which is a lower bound on the model evidence. Some of these update equations are very similar to their respective counterparts from the mixture of Gaussians model, which shows that these two model are closely related to each other.

4.5 Variational Bayes for Embedded Clustering

For the single subject version of DCM, a variational approximation has already been derived [36, 88]. This solution is part of an open source software suite called statistical parametric mapping (SPM), which is actively used in the computational psychiatry community [116]. Similarly, the variational Bayes approximation for the mixture of Gaussians models has been derived in [33] and is considered to be a very well-studied application of the variational Bayes scheme [17, 37].

Embedded clustering for DCM is a combination of these two models, DCM

and mixture of Gaussians, into a single unified model. However, due to the interaction between the DCM part and Gaussian mixture part, we cannot simply reuse the variational update equations given in [33] and [36] for the embedded clustering model. Thus, our task in this section is to derive new update equations for the variational Bayes approximation to the embedded clustering model for DCM with fMRI observation. In the course of this derivation, we will notice that some of the update equations are similar to those of the variational approximation to the mixture of Gaussians model. Indeed, many of the techniques used for the derivation of the approximation for Gaussian mixtures work analogously for the embedded clustering model.

4.5.1 Factorization of the Variational Distribution

A detailed introduction to the variational Bayes approximation methods has been provided in section 2.4.2. The first steps in the approximation scheme are always to distinguish between the observed variables and the latent variables, as well as to divide the set of latent variables into subgroups, which defines a factorization over the variational distribution.

In the case of the embedded clustering model for DCM with fMRI observation, the data consists of the measured fMRI BOLD response $\boldsymbol{y}_n$ of each of the N subjects. On the other hand, the set of unobserved variables consists of all variables which are represented in figure 4.8 with unshaded nodes. This includes one cluster label d_n for each subject, as well as the cluster probabilities or weights $\boldsymbol{\pi}$ from which the labels are drawn. Furthermore, this also includes the parameters of the Gaussian mixture components $\boldsymbol{\mu}_k$ and Σ_k, as well as the parameters of the DCM model $\boldsymbol{\theta}_n$ and Λ_n. Note that since we have one DCM model for each subject, we also have one set of DCM parameters per subject.

The second step in the variational Bayes scheme is to divide the latent variables into subgroups. The variables in each subgroup are approximated with a variational distribution, which is independent from the distribution over the other subgroups. This causes the variational distribution over all latent variables to factor across several terms, one for each subgroup. For the embedded clustering model, it is a natural choice to factor between the variables of the DCM part and the variables pertaining to the mixture model. In addition, we also have to define factorizations within these subgroups. For the DCM part, we factor between the measurement noise precision Λ_n and the DCM parameters $\boldsymbol{\theta}_n$. For the part of the mixture model, we apply the factorization between the cluster labels d_n and the model parameters $\boldsymbol{\pi}$, $\boldsymbol{\mu}_k$ and Σ_k, which was already used for the regular mixture of Gaussians model.

Thus, the variational distribution factors as follows:

$$q(\{\boldsymbol{\theta}_n, \Lambda_n, d_n\}, \{\boldsymbol{\mu}_k, \Sigma_k\}, \boldsymbol{\pi}) \;=\; q(\{\boldsymbol{\theta}_n\})q(\{\Lambda_n\})q(\{d_n\})q(\{\boldsymbol{\mu}_k, \Sigma_k\})q(\boldsymbol{\pi}) \tag{4.34}$$

In addition to defining a factorization over the variational distribution q, one can also restrict the factors of q on the right hand side of the equation above to have certain functional forms. This is discussed in more detail in the next section.

4.5.2 Functional Form of the Variational Factors

In the following, we discuss the functional form of the factors in the variational distribution given in equation (4.34). For convenience, the discussion will be split into two parts. We first discuss the factors for the mixture model part and then turn towards the factors concerning the DCM part of the embedded clustering model.

In the mixture of Gaussians example from section 4.3.1, restricting the functional form of the factors in the variational distribution was not necessary, since the functional form of the likelihood and the conjugacy of the prior distributions already suggested a convenient form for the factors of q. This argument extends to the Gaussian mixtures part of the embedded clustering model as well, which means that the factors $q(\boldsymbol{\pi})$, $q(\{d_n\})$ and $q(\{\boldsymbol{\mu}_k, \Sigma_k\})$ are determined in their functional form by the likelihood and the choice of the priors. And as the priors are conjugate, the factors of q have the same functional form as their respective prior distributions.

For the Gaussian mixture part of the model, this means that $q(\boldsymbol{\pi})$ is a Dirichlet distribution governed by a variational parameter $\boldsymbol{\alpha}$:

$$q(\boldsymbol{\pi}) = \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{\alpha}), \tag{4.35}$$

because the prior on $\boldsymbol{\pi}$ (eq (4.28)) is given by a Dirichlet distribution.

Analogously, $q(\{\boldsymbol{\mu}_k, \Sigma_k\})$ is a product of independent distributions for each cluster k, which are given by normal-inverse Wishart distributions with variational parameters $\bar{\boldsymbol{\mu}}_k$, τ_k, ν_k and $\bar{\Sigma}_k$:

$$q(\{\boldsymbol{\mu}_k, \Sigma_k\}_{k=1,\ldots,K}) = \prod_{k=1}^{K} q(\boldsymbol{\mu}_k, \Sigma_k), \tag{4.36}$$

$$q(\boldsymbol{\mu}_k, \Sigma_k) = \mathcal{N}(\boldsymbol{\mu}_k|\bar{\boldsymbol{\mu}}_k, \Sigma_k/\tau_k)\, \mathcal{W}^{-1}(\Sigma_k|\bar{\Sigma}_k, \nu_k). \tag{4.37}$$

This is because the conjugate prior (eq (4.22)) has been defined as a normal-inverse Wishart distribution; and it follows the same pattern as the regular mixture of Gaussians model.

Finally, the variational distribution over the set of cluster labels $q(\{d_n\})$ factors into a product containing one categorical distribution per subject n. Each of these distributions has variational parameters q_{nk} which have to be positive and sum to one when summed over k:

$$q(d_n{=}k) = q_{nk} \tag{4.38}$$

$$\sum_{k=1}^{K} q_{nk} = 1. \tag{4.39}$$

In contrast to mixture modelling, where a restriction of the functional form of the variational distribution is unnecessary, the variational approximation to the single subject DCM model contains what is called the Laplace approximation in [36]. This means that all factors of the variational distribution q for the DCM model are restricted to be Gaussian. Variables which have to be positive are transformed to the log domain first, which corresponds to placing a log-normal distribution over the original variable. In [36], this step is what makes the variational approximation to the DCM model tractable. However, we would like to point out that throughout the machine learning community, the term Laplace approximation is used in a slightly different way than in [36].

For the embedded clustering model, we have decided to deviate slightly from this approach, by only restricting the variational distribution over the DCM parameters to be Gaussian. As will be shown later, this distribution will factor across subjects:

$$q(\{\boldsymbol{\theta}_n\}) = \prod_{n=1}^{N} q(\boldsymbol{\theta}_n) \tag{4.40}$$

$$q(\boldsymbol{\theta}_n) = \mathcal{N}(\boldsymbol{\theta}_n | \boldsymbol{\mu}_\theta, \Sigma_\theta). \tag{4.41}$$

The distribution over the diagonal elements $\lambda_{r,n}$ of the noise precision Λ_n will be a gamma distribution $q(\lambda_{r,n}) = \mathrm{Gam}(\lambda_{r,n} | a_{r,n}, b_{r,n})$ instead of a log-normal distribution. This is not due to any assumptions, but a direct consequence of choosing the gamma distribution as a conjugate prior for the elements of Λ_n. We will see later that this leads to a very simple form for the update equations for the variational parameters $a_{r,n}$ and $b_{r,n}$.

The next step in the approximation scheme is to derive an expression for the free energy and a solution for the parameters of the variational distributions q which maximizes the free energy.

4.5.3 The Variational Update Equations

Above, we have decided on a factorization of the variational distribution q (eq (4.34)), as well as the functional form of some of its factors. As mentioned in section 2.4.2, the aim of these assumptions is to make the integral in the expression for the free energy (eq (2.28)) tractable. In order to complete the variational approximation, two steps remain. One of them is to plug in the factors of the variational distribution q derived above into equation (2.30) and evaluate the integrals in order to derive the expression for the free energy of the embedded clustering model.

On the other hand, the factors of the variational distribution q depend on parameters, which we called variational parameters and which, for the embedded clustering model, were identified above as $\boldsymbol{\alpha}$, q_{nk}, $\bar{\boldsymbol{\mu}}_k$, τ_k, ν_k, $\bar{\Sigma}_k$, $\boldsymbol{\mu}_\theta$, Σ_θ, $a_{r,n}$ and $b_{r,n}$. Thus, the other step is to obtain a good approximation by maximizing the free energy with respect to these variational parameters.

As already mentioned, this can be done with any optimization method. One possible way, which was proposed in [36], would be to first derive the expression for the free energy and take the derivative of the free energy with respect to the variational parameters. The maximization of the free energy can then be done using a gradient based optimizer. However, for the embedded clustering model, we will use the alternative optimization strategy introduced in section 2.4.2.

This optimization method, which is based on minimizing a Kullback-Leibler divergence [17], can be viewed as an extension of the EM algorithm and has been applied to the mixture of Gaussians model with good success [17, 37]. The advantage of using this scheme is that we can reuse some of the results from the variational approximation to the Gaussian mixture model. Due to the similarity between the mixture of Gaussians model and the mixture modelling part in the embedded clustering model, some of the update equations for the embedded clustering model will have a similar functional form as their respective counterparts from the mixture of Gaussians.

The key component of the EM-style free energy optimization introduced in section 2.4.2 is the general formula for the optimum expression for one factor of the variational distribution $q(Z_j)$, under the assumption that the other factors $q(Z_{i\neq j})$ are kept constant. This formula is given by equation (2.35) and involves averaging over the logarithm of the joint distribution with respect to the other factors of q. Equation (2.35) tells us how to update the parameters of $q(Z_j)$ in order to increase the free energy, while keeping the parameters of the other factors $q(Z_{i\neq j})$ constant. Applying this equation to each of the factors in turn over several iterations will allow us to converge

to a local maximum of the free energy, while restarting the whole process with different initial conditions will increase the chance of finding the global maximum.

In the case of the embedded clustering model, the factors $q(Z_j)$ of the variational distribution correspond to the terms on the right hand side of equation (4.34). Thus, our task in the next step is to evaluate equation (2.35) for each of these factors. We will begin with a more detailed discussion of the derivation for the factor $q(\boldsymbol{\pi})$ and then move on to the other factors. Since the derivation always follows a similar scheme, we will limit the discussion to the initial steps and the result, while the details are provided in the appendix.

Cluster Weights: $q(\boldsymbol{\pi})$
As mentioned above, the functional form of the variational distribution over the cluster weights is given by a Dirichlet distribution $q(\boldsymbol{\pi}) = \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{\alpha})$, which depends on the variational parameter $\boldsymbol{\alpha}$. This functional form is a consequence of the structure of the model and the conjugacy of the prior distribution.

When manipulating the update equation, it is more convenient to use the logarithmic form given in equation (2.36), which for the cluster weights reads:

$$\log q(\boldsymbol{\pi}) = \sum_{\{d_n\}} \int q(\{\boldsymbol{\theta}_n\}) q(\{\Lambda_n\}) q(\{d_n\}) q(\{\boldsymbol{\mu}_k, \Sigma_k\}) \times \log p(\{\boldsymbol{y}_n, d_n, \boldsymbol{\theta}_n, \Lambda_n\}, \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) \times d\{\boldsymbol{\theta}_n\} d\{\Lambda_n\} d\{\boldsymbol{\mu}_k, \Sigma_k\} + const, \quad (4.42)$$

where $\sum_{\{d_n\}}$ means summing over all possible configurations of the set of cluster labels $\{d_n\}_{n=1,\ldots,N}$. The *const* term contains terms which do not depend on $\boldsymbol{\pi}$, such as the logarithm of the normalization factor, which ensures that the distribution integrates to one. The probability distribution p inside the logarithm under the integral is the joint distribution of the embedded clustering model. When plugging in the expression for this distribution, which is given in equation (4.27), the integral splits up into several parts:

$$\log q(\boldsymbol{\pi}) = \log p(\boldsymbol{\pi}|\alpha_0) + \sum_{\{d_n\}} q(\{d_n\}) \sum_{n=1}^{N} \log p(d_n|\boldsymbol{\pi}) + \sum_{\{d_n\}} \int q(\{\boldsymbol{\theta}_n\}) q(\{\Lambda_n\}) q(\{d_n\}) q(\{\boldsymbol{\mu}_k, \Sigma_k\}) \times$$

$$\Big(\sum_{k=1}^{K} \Big(\log p(\boldsymbol{\mu}_k, \Sigma_k | \boldsymbol{\mu}_0, \tau_0, \nu_0, \Sigma_0) \Big) + \sum_{n=1}^{N} \Big(\log p(\boldsymbol{y}_n | \boldsymbol{\theta}_n, \Lambda_n) + \log p(\boldsymbol{\theta}_{c,n} | d_n, \{\boldsymbol{\mu}_k, \Sigma_k\}) + \log p(\boldsymbol{\theta}_{h,n} | \boldsymbol{\mu}_h, \Sigma_h) + \log p(\Lambda_n | a_0, b_0) \Big) \Big) \times d\{\boldsymbol{\theta}_n\} d\{\Lambda_n\} d\{\boldsymbol{\mu}_k, \Sigma_k\} + const \quad (4.43)$$

However, in the equation above, all terms below the first line do not depend on $\boldsymbol{\pi}$, which means that they can be absorbed into the *const* term. This reduces the equation to:

$$\begin{aligned} \log q(\boldsymbol{\pi}) &= \log p(\boldsymbol{\pi}|\boldsymbol{\alpha}_0) + \sum_{\{d_n\}} q(\{d_n\}) \sum_{n=1}^{N} \log p(d_n|\boldsymbol{\pi}) + const \\ &= \log p(\boldsymbol{\pi}|\boldsymbol{\alpha}_0) + \sum_{k=1}^{K} \sum_{n=1}^{N} q_{nk} \log \pi_k + const \\ &= \sum_{k=1}^{K} \Big((\alpha_{0,k} - 1) \log \pi_k + \sum_{n=1}^{N} q_{nk} \log \pi_k \Big) + const. \end{aligned} \quad (4.44)$$

Above, we have plugged in the definition for the variational distribution of the cluster labels $q(d_n = k) = q_{nk}$ in the second line and the definition of the prior $p(\boldsymbol{\pi}|\boldsymbol{\alpha}_0)$ in the third line. $\alpha_{0,k}$ denotes the k-th element of the vectors $\boldsymbol{\alpha}_0$.

Comparing the above equation to the general form of a Dirichlet distribution, which is given in the appendix A.3, it becomes clear that $q(\boldsymbol{\pi})$ is described by a Dirichlet distribution with parameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)^T$ given by:

$$\alpha_k = \alpha_{0,k} + \sum_{n=1}^{N} q_{nk}. \quad (4.45)$$

This is the update equation for the parameter of the variational distribution over the cluster weights $\boldsymbol{\pi}$. Notice how the α_k depend on q_{nk}, which are the parameters of the variational distribution over the cluster labels d_n. This definition is circular, since we will see in a moment that the update equations for q_{nk} also depend on α_k. Therefore, the update equations have to be evaluated iteratively in order to converge to a stable solution.

In addition, one should note that q_{nk} can be interpreted as approximation to the posterior probability that $\boldsymbol{\theta}_{c,n}$ was generated by cluster k. This variable is also called the responsibility of cluster k for generating observation n [17].

Thus, the sum in equation (4.45) over all responsibilities of cluster k can be interpreted as the effective number of observations associated with cluster k. The variational parameter α_k is the sum of this effective number of observations and the parameter of the prior distribution α_0. This in turn, means that $\alpha_{0,k}$ can be interpreted as the number of pseudo observations, which are assigned to cluster number k before data collection begins.

In this observation lies the answer to the question raised in sections 2.2 and 4.3.1 on how to choose a weak or strong prior. The prior for the cluster weights $p(\boldsymbol{\pi}|\boldsymbol{\alpha}_0)$ is weak if the value for $\alpha_{0,k}$ is small compared to the overall number of subjects N. This interpretation of the prior distribution parameters as pseudo observations is very common in Bayesian statistics and can be applied to many models [17].

Cluster Mean and Covariance: $q(\{\boldsymbol{\mu}_k, \Sigma_k\})$
We start again with equation (2.36), which for the cluster parameters is given by:

$$\log q(\{\boldsymbol{\mu}_k, \Sigma_k\}) = \sum_{\{d_n\}} \int q(\{\boldsymbol{\theta}_n\}) q(\{\Lambda_n\}) q(\{d_n\}) q(\boldsymbol{\pi}) \times \log p(\{\boldsymbol{y}_n, d_n, \boldsymbol{\theta}_n, \Lambda_n\}, \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) \times d\{\boldsymbol{\theta}_n\} d\{\Lambda_n\} d\boldsymbol{\pi} + const, \quad (4.46)$$

The difference to equation (4.42) is that we now integrate over $\boldsymbol{\pi}$ instead of $\{\boldsymbol{\mu}_k, \Sigma_k\}$. After plugging in the logarithm of the joint distribution, we arrive at:

$$\log q(\{\boldsymbol{\mu}_k, \Sigma_k\}) = \sum_{k=1}^{K} \sum_{n=1}^{N} q_{nk} \int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n}) \log p(\boldsymbol{\theta}_{c,n}|d_n = k, \boldsymbol{\mu}_k, \Sigma_k) d\boldsymbol{\theta}_{c,n} + \sum_{k=1}^{K} \log p(\boldsymbol{\mu}_k, \Sigma_k) + const, \quad (4.47)$$

where the *const*-term contains all terms that do not depend on the set of cluster parameters $\{\boldsymbol{\mu}_k, \Sigma_k\}$. Equation (4.47) consists of a sum over the clusters, which is the reason why the variational distribution over the set of cluster parameters $q(\{\boldsymbol{\mu}_k, \Sigma_k\})$ factors across clusters (eq (4.37)). This is possible due to the structure of the likelihood, but also because we defined the prior distribution over the set of cluster parameters as a product over the clusters (eq (4.21)). Thus, the logarithm of the factors in $q(\{\boldsymbol{\mu}_k, \Sigma_k\})$

equal the summands in the above equation, up to a constant term:

$$\log q(\boldsymbol{\mu}_k, \Sigma_k) = \sum_{n=1}^{N} q_{nk} \int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n}) \log p(\boldsymbol{\theta}_{c,n} | d_n = k, \boldsymbol{\mu}_k, \Sigma_k) d\boldsymbol{\theta}_{c,n} + \log p(\boldsymbol{\mu}_k, \Sigma_k) + const. \quad (4.48)$$

After plugging in the logarithm of the prior $\log p(\boldsymbol{\mu}_k, \Sigma_k)$, as well as the solution to the integral, which is derived in appendix C.1, the variational distribution over the parameters of each cluster is given by a normal-inverse Wishart distribution:

$$\log q(\boldsymbol{\mu}_k, \Sigma_k) = \mathcal{N}(\boldsymbol{\mu}_k | \bar{\boldsymbol{\mu}}_k, \Sigma_k / \tau_k) \, \mathcal{W}^{-1}(\Sigma_k | \bar{\Sigma}_k, \nu_k), \quad (4.49)$$

with parameters given by:

$$\bar{\boldsymbol{\mu}}_k = \frac{q_k \boldsymbol{\mu}_{ck} + \tau_0 \bar{\boldsymbol{\mu}}_0}{q_k + \tau_0}, \quad (4.50)$$

$$\tau_k = q_k + \tau_0, \quad (4.51)$$

$$\nu_k = q_k + \nu_0 \quad \text{and} \quad (4.52)$$

$$\bar{\Sigma}_k = \Sigma_{ck} + \sum_{n=1}^{N} q_{nk} (\boldsymbol{\mu}_{c,n} - \boldsymbol{\mu}_{ck}) (\boldsymbol{\mu}_{c,n} - \boldsymbol{\mu}_{ck})^T + \frac{q_k \tau_0}{q_k + \tau_0} (\boldsymbol{\mu}_{ck} - \bar{\boldsymbol{\mu}}_0)(\boldsymbol{\mu}_{ck} - \bar{\boldsymbol{\mu}}_0)^T + \bar{\Sigma}_0. \quad (4.53)$$

In order to simplify the notation for these update equations, we have defined the following shortcuts:

$$q_k = \sum_{n=1}^{N} q_{nk}, \quad (4.54)$$

$$\boldsymbol{\mu}_{ck} = \frac{1}{q_k} \sum_{n=1}^{N} q_{nk} \boldsymbol{\mu}_{c,n}, \quad (4.55)$$

$$\Sigma_{ck} = \sum_{n=1}^{N} q_{nk} \Sigma_{c,n}, \quad (4.56)$$

where $\boldsymbol{\mu}_{c,n}$ is the mean of the variational distribution over the DCM connectivity parameters $q(\boldsymbol{\theta}_{c,n})$ of subject number n and $\Sigma_{c,n}$ is the covariance matrix of $q(\boldsymbol{\theta}_{c,n})$, derived below. Thus, we see that also the update equations for the variational distribution over the cluster parameters depend on q_{nk}.

Cluster Labels: $q(\{d_n\})$
Starting from equation (2.36), we have the following ansatz for the variational distribution over the cluster labels:

$$\log q(\{d_n\}) = \int q(\{\boldsymbol{\theta}_n\})q(\{\Lambda_n\})q(\boldsymbol{\pi})q(\{\boldsymbol{\mu}_k, \Sigma_k\}) \times \log p(\{\boldsymbol{y}_n, d_n, \boldsymbol{\theta}_n, \Lambda_n\}, \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) \times d\{\boldsymbol{\theta}_n\}d\{\Lambda_n\}d\boldsymbol{\pi}d\{\boldsymbol{\mu}_k, \Sigma_k\} + const \tag{4.57}$$

$$= \sum_{n=1}^{N} \Big(\int q(\boldsymbol{\pi}) \log p(d_n|\boldsymbol{\pi})d\boldsymbol{\pi} + \int q(\boldsymbol{\theta}_{c,n})q(\{\boldsymbol{\mu}_k, \Sigma_k\}) \times \log p(\boldsymbol{\theta}_{c,n}|d_n, \{\boldsymbol{\mu}_k, \Sigma_k\})d\boldsymbol{\theta}_{c,n}d\{\boldsymbol{\mu}_k, \Sigma_k\} \Big) + const, \tag{4.58}$$

which consists of a sum over n. This means that $q(\{d_n\})$ separates into independent distributions across subjects. Thus, for the probability of a single label $q(d_n = k) = q_{nk}$, we have the following relationship:

$$\log q_{nk} = \int q(\boldsymbol{\pi}) \log \pi_k d\boldsymbol{\pi} + \int q(\boldsymbol{\theta}_{c,n})q(\boldsymbol{\mu}_k, \Sigma_k) \times \log p(\boldsymbol{\theta}_{c,n}|d_n = k, \boldsymbol{\mu}_k, \Sigma_k)d\boldsymbol{\theta}_{c,n}d\boldsymbol{\mu}_k d\Sigma_k + const \tag{4.59}$$

$$= -\frac{1}{2}\log|\bar{\Sigma}_k| + \frac{1}{2}\Psi_{p_c}(\nu_k) - \frac{p_c}{2\tau_k} - \frac{\nu_k}{2}\operatorname{tr}(\bar{\Sigma}_k^{-1}\Sigma_{c,n}) - \frac{\nu_k}{2}(\boldsymbol{\mu}_{c,n} - \bar{\boldsymbol{\mu}}_k)^T\bar{\Sigma}_k^{-1}(\boldsymbol{\mu}_{c;n} - \bar{\boldsymbol{\mu}}_k) + \Psi(\alpha_k) + const. \tag{4.60}$$

Here, we have used $p(d_n = k|\boldsymbol{\pi}) = \pi_k$. The symbol Ψ_{p_c} denotes a sum of digamma functions defined in equation (A.18) and $\operatorname{tr}(\cdot)$ denotes the trace operator, which sums the diagonal elements of a matrix. In addition, we have defined p_c as the dimensionality of the vector of DCM connectivity parameters $\boldsymbol{\theta}_c$.

The solution of the integrals in the above equation are derived in appendix C.1 and the value of the *const*-term can be obtained via the normalization constraint $\sum_{k=1}^{K} q_{nk} = 1$.

DCM Parameters: $q(\boldsymbol{\theta}_n)$
The version of equation (2.36) for the DCM parameters

$$\log q(\{\boldsymbol{\theta}_n\}) = \sum_{\{d_n\}} \int q(\boldsymbol{\pi})q(\{\Lambda_n\})q(\{d_n\})q(\{\boldsymbol{\mu}_k, \Sigma_k\}) \times \log p(\{\boldsymbol{y}_n, d_n, \boldsymbol{\theta}_n, \Lambda_n\}, \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) \times d\boldsymbol{\pi}d\{\Lambda_n\}d\{\boldsymbol{\mu}_k, \Sigma_k\} + const, \tag{4.61}$$

$$= \sum_{n=1}^{N} \Big(\int q(\Lambda_n) \log p(\boldsymbol{y}_n|\boldsymbol{\theta}_n, \Lambda_n) d\Lambda_n + \sum_{k=1}^{K} q_{nk} \int q(\boldsymbol{\mu}_k, \Sigma_k) \log p(\boldsymbol{\theta}_{c,n}|d_n = k, \boldsymbol{\mu}_k, \Sigma_k) d\boldsymbol{\mu}_k d\Sigma_k + \log p(\boldsymbol{\theta}_{h,n}|\boldsymbol{\mu}_h, \Sigma_h) \Big) + const \tag{4.62}$$

also separates across subjects. Again, we have absorbed all terms which do not depend on $\boldsymbol{\theta}_n$ into the constant. Thus, we have for each subject:

$$\log q(\boldsymbol{\theta}_n) = \int q(\Lambda_n) \log p(\boldsymbol{y}_n|\boldsymbol{\theta}_n, \Lambda_n) d\Lambda_n + \sum_{k=1}^{K} q_{nk} \int q(\boldsymbol{\mu}_k, \Sigma_k) \log p(\boldsymbol{\theta}_{c,n}|d_n = k, \boldsymbol{\mu}_k, \Sigma_k) d\boldsymbol{\mu}_k d\Sigma_k + \log p(\boldsymbol{\theta}_{h,n}|\boldsymbol{\mu}_h, \Sigma_h) + const \tag{4.63}$$

$$= -\frac{1}{2} \int q(\Lambda_n) \big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)^T \Lambda_n \big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big) d\Lambda_n + \sum_{k=1}^{K} q_{nk} \int q(\boldsymbol{\mu}_k, \Sigma_k) \log p(\boldsymbol{\theta}_{c,n}|d_n = k, \boldsymbol{\mu}_k, \Sigma_k) d\boldsymbol{\mu}_k d\Sigma_k + \log p(\boldsymbol{\theta}_{h,n}|\boldsymbol{\mu}_h, \Sigma_h) + const, \tag{4.64}$$

where we have plugged in the definition for $\log p(\boldsymbol{y}_n|\boldsymbol{\theta}_n, \Lambda_n)$ in the second line. The function $\boldsymbol{g}$ is the DCM system equation introduced in equation (4.8) in the beginning of this chapter. Here, we have dropped the input $\boldsymbol{u}$ from the function arguments to simplify the notation.

The problem with this equation is that the integral over Λ_n is not tractable, since we cannot write down the function $\boldsymbol{g}$ in closed form. However, we can evaluate $\boldsymbol{g}$ for any given argument, which allows us to solve the problem by applying a first order Taylor approximation to $\boldsymbol{g}$ around a suitable chosen expansion point $\boldsymbol{m}_n$. The first order derivative, which in the case of $\boldsymbol{g}$ is given by its Jacobian matrix G, can be obtained using numerical methods.

Applying the Taylor approximation and solving the integrals leads to a functional form for $\log q(\boldsymbol{\theta}_n)$ which consists of a quadratic form in $\boldsymbol{\theta}_n$. This corresponds, up to a constant, to the logarithm of a Gaussian distribution and shows that applying the first order Taylor approximation is equivalent to restricting the variational distribution $q(\boldsymbol{\theta}_n)$ to be Gaussian, mentioned at the end of the previous section.

The detailed derivation is given in appendix C.1. Here, we only provide the

result for the update equations for the mean $\boldsymbol{\mu}_n$ and covariance Σ_n of $q(\boldsymbol{\theta}_n)$:

$$\Sigma_n^{-1} = G_n^T \bar{\Lambda}_n G_n + \Lambda'_n \tag{4.65}$$

$$\boldsymbol{\mu}_n = \Sigma_n \big(G_n^T \bar{\Lambda}_n (\boldsymbol{\varepsilon}_n + G_n \boldsymbol{m}_n) + \boldsymbol{\mu}'_n \big). \tag{4.66}$$

The variable $\boldsymbol{\varepsilon}_n$ is the residual, i.e. the difference between the observation $\boldsymbol{y}_n$ and the function value of $\boldsymbol{g}(\boldsymbol{m}_n)$ at the expansion point:

$$\boldsymbol{\varepsilon}_n = \boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{m}_n), \tag{4.67}$$

while G_n is the Jacobian matrix of $\boldsymbol{g}$ at the expansion point:

$$G_n = \left. \frac{\partial \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{u})}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta} = \boldsymbol{m}_n}. \tag{4.68}$$

Additionally, we have defined the following shortcuts:

$$\Lambda'_n = \begin{pmatrix} \sum_{k=1}^K q_{nk} \nu_k \bar{\Sigma}_k^{-1} & 0 \\ 0 & \Sigma_h^{-1} \end{pmatrix}, \tag{4.69}$$

$$\boldsymbol{\mu}'_n = \Big(\sum_{k=1}^K q_{nk} \nu_k \bar{\Sigma}_k^{-1} \bar{\boldsymbol{\mu}}_k,\, \Sigma_h^{-1} \boldsymbol{\mu}_h \Big). \tag{4.70}$$

The zeros in the first line denote matrices of appropriate size and with all zero entries, while $\bar{\Lambda}_n$ in equation (4.65) denotes the mean noise precision matrix defined in equation (4.81) in the next paragraph.

Noise Precision: $q(\Lambda_n)$

The precision matrix of the measurement noise Λ_n is a diagonal matrix parameterized by a set of real positive entries $\{\lambda_{r,n}\}_{r=1,\dots,R}$, which denote the noise precisions for the fMRI BOLD time series from each of the brain regions. For the variational distribution over $\{\lambda_{r,n}\}_{r=1,\dots,R,\, n=1,\dots,N}$, equation (2.36) is given by:

$$\begin{aligned} \log q(\{\lambda_{r,n}\}) = \sum_{\{d_n\}} \int q(\boldsymbol{\pi}) q(\{\boldsymbol{\theta}_n\}) q(\{d_n\}) q(\{\boldsymbol{\mu}_k, \Sigma_k\}) \times \\ \log p(\{\boldsymbol{y}_n, d_n, \boldsymbol{\theta}_n, \Lambda_n\}, \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}) \times \\ d\boldsymbol{\pi} d\{\boldsymbol{\theta}_n\} d\{\boldsymbol{\mu}_k, \Sigma_k\} + const. \end{aligned} \tag{4.71}$$

After plugging in the expression for the log-joint distribution, as well as the definition of Λ_n (eq (4.10)), the equation reduces to a double sum:

$$\begin{aligned} \log q(\{\lambda_{r,n}\}) = \sum_{n=1}^N \sum_{r=1}^R \Big(-\frac{\lambda_{r,n}}{2} \int_{\boldsymbol{\theta}_n} q(\boldsymbol{\theta}_n) \big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)^T Q_r \big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big) d\boldsymbol{\theta}_n \\ + \frac{q_r}{2} \log \lambda_{r,n} + \log p(\lambda_{r,n} | a_{r,0}, b_{r,0}) \Big) + const. \end{aligned} \tag{4.72}$$

This means that the distribution over the precision matrices will factor into a product of independent distributions over single subjects and regions:

$$q(\{\Lambda_n\}) = \prod_{r,n} q(\lambda_{r,n}), \tag{4.73}$$

in the same way as the prior distribution introduced in equation (4.33). The logarithm of each factor in the product is then given by:

$$\log q(\lambda_{r,n}) = -\frac{\lambda_{r,n}}{2} \int_{\boldsymbol{\theta}_n} q(\boldsymbol{\theta}_n)\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)^T Q_r \big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big) d\boldsymbol{\theta}_n + \frac{q_r}{2} \log \lambda_{r,n} + \log p(\lambda_{r,n}|a_{r,0}, b_{r,0}) + const. \tag{4.74}$$

Solving the integral in the equation above requires applying the same Taylor approximation to $\boldsymbol{g}$ as in the previous paragraph. And due to the conjugacy of the prior distribution, $q(\lambda_{r,n})$ is also given by a gamma distribution $q(\lambda_{r,n}) = \text{Gam}(\lambda_{r,n}|a_{r,n}, b_{r,n})$ with parameters $a_{r,n}$ and $b_{r,n}$. Here, we omit the details, which are provided in appendix C.1, and only present the update equations:

$$a_{r,n} = a_{r,0} + \frac{q_r}{2} \tag{4.75}$$

$$b_{r,n} = b_{r,0} + \frac{b'_{r,n}}{2}, \tag{4.76}$$

with $b'_{r,n}$ given by:

$$b'_{r,n} = \boldsymbol{\varepsilon}_n^T Q_r \boldsymbol{\varepsilon}_n + \text{tr}\left(G_n^T Q_r G_n \Sigma_n\right). \tag{4.77}$$

$\boldsymbol{\varepsilon}_n$ and G_n are defined just like in the previous paragraph and Q_r has been introduced in equation (4.10), as a matrix with the same structure as Λ, but with ones on the positions of λ_r and zeros elsewhere. With the expression for the mean of the gamma distribution (eq (A.25)), we can calculate the mean value of $\lambda_{r,n}$ as:

$$\langle \lambda_{r,n} \rangle = \frac{a_{r,n}}{b_{r,n}} \tag{4.78}$$

$$= \frac{a_{r,0} + q_r/2}{b_{r,0} + b'_{r,n}/2} \tag{4.79}$$

$$=: \bar{\lambda}_{r,n}. \tag{4.80}$$

With this, we can define the mean precision matrix from the previous paragraph as:

$$\bar{\Lambda}_n = \sum_{r=1}^{R} \bar{\lambda}_{r,n} Q_r. \tag{4.81}$$

It is worth taking a moment to inspect equation (4.79) in more detail. For a weak prior distribution, i.e. for values of $a_{r,0}$ and $b_{r,0}$ which are negligible compared to q_r and $b'_{r,n}$, equation (4.79) can be approximated by:

$$\bar{\lambda}_{r,n} \approx \frac{q_r}{b'_{r,n}}. \tag{4.82}$$

The term $\boldsymbol{\varepsilon}_n^T Q_r \boldsymbol{\varepsilon}_n$ in b'_{nr} is the sum of squares of the residual and can be interpreted as a sum of squared (prediction) error terms. The second term in b'_{nr} can be interpreted as a correction term taking into account the uncertainty about $\boldsymbol{\mu}_n$. On the other hand, q_r is the number of samples in the fMRI time series of region r, which also corresponds to the number of ones in Q_r. This expression is very similar to the unbiased estimate of the variance of a Gaussian from N iid. observations, which is also given by a sum of squared error term divided by the effective number of samples:

$$\sigma^2 = \frac{1}{N-1}\sum_{n=1}^{N}(x_n - \mu)^2.$$

Note that the above considerations would apply equally to the single subject DCM without clustering if the authors of [36] had chosen a gamma prior on the noise precision. With this interpretation, we can also see that the prior parameter $a_{r,0}$ corresponds to the number of pseudo observations, while $b_{r,0}$ corresponds to the sum of squared error induced by the pseudo observations.

Free Energy: $\mathcal{F}$

In section 2.4.2, we have shown that generally, the variational free energy can be expressed as a sum of the expectation of the log-joint distribution and the entropy of the variational distribution (eq (2.39)). Plugging in the joint and variational distributions for the embedded clustering model, the free energy splits into a sum of several expressions, due to the variational distribution being a product of independent factors:

$$\begin{aligned}
\mathcal{F} &= \Big\langle \log p\big(\{\boldsymbol{y}_n, d_n, \boldsymbol{\theta}_n, \Lambda_n\}, \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \Sigma_k\}\big)\Big\rangle_q \\
&\qquad + \mathcal{H}\left(q(\boldsymbol{\pi})q(\{\boldsymbol{\theta}_n\})q(\{\Lambda_n\})q(\{d_n\})q(\{\boldsymbol{\mu}_k, \Sigma_k\})\right) \qquad (4.83)\\
&= \big\langle \log p(\boldsymbol{\pi}|\alpha_0)\big\rangle_q + \sum_{k=1}^{K}\big\langle \log p(\boldsymbol{\mu}_k, \Sigma_k|\boldsymbol{\mu}_0, \tau_0, \nu_0, \Sigma_0)\big\rangle_q \\
&\quad + \sum_{n=1}^{N}\big\langle \log p(\boldsymbol{y}_n|\boldsymbol{\theta}_n, \Lambda_n)\big\rangle_q + \sum_{n=1}^{N}\big\langle \log p(\boldsymbol{\theta}_{c,n}|d_n, \{\boldsymbol{\mu}_k, \Sigma_k\})\big\rangle_q \\
&\quad + \sum_{n=1}^{N}\big\langle \log p(d_n|\boldsymbol{\pi})\big\rangle_q + \sum_{n=1}^{N}\big\langle \log p(\boldsymbol{\theta}_{h,n}|\boldsymbol{\mu}_h, \Sigma_h)\big\rangle_q
\end{aligned}$$

$$+\sum_{n=1}^{N}\left\langle \log p(\Lambda_n|a_0,b_0)\right\rangle_q + \mathcal{H}\left(q(\boldsymbol{\pi})\right) + \mathcal{H}\left(q(\{\boldsymbol{\theta}_n\})\right) + \mathcal{H}\left(q(\{\Lambda_n\})\right) + \mathcal{H}\left(q(\{d_n\})\right) + \mathcal{H}\left(q(\{\boldsymbol{\mu}_k,\Sigma_k\})\right), \quad (4.84)$$

However, most of the integrals contained in these terms have already been solved during the derivation of the update equations. Since the final expression for the free energy is rather cumbersome and does not contribute to the understanding of the topic, we will omit it here. Instead, we will close this section with a summary of the variational update scheme.

Summary:
In this section, we have derived update equations for the parameters of the variational distribution q and the free energy of the embedded clustering model, with the aim to obtain an approximation to the model evidence. The model evidence, which is the denominator in Bayes law (eq (2.10)), plays an important role for model comparison; and the free energy as defined in equation (2.28) is a lower bound of the model evidence. Thus, by maximizing the free energy with respect to the variational parameters, we reduce the difference between this lower bound and the evidence and obtain an approximation to the evidence upon convergence. The list of variational parameters include: q_{nk}, $\boldsymbol{\alpha}$, $\boldsymbol{\mu}_n$, Σ_n, $\bar{\boldsymbol{\mu}}_k$, τ_k, ν_k, Σ_k, $a_{r,n}$ and $b_{r,n}$. And as stated in section 2.4.2, the variational distribution q will be an approximation to the posterior distribution when the free energy is maximized.

For the purpose of maximizing the free energy, we need to evaluate the update equations given above in turn. Each of these equations updates one of the factors of the variational distribution q in such a way that the free energy will always increase. However, since the update equations for the different factors of q depend on each other, we have to iterate this process to reach a fix point. A nice property of this scheme is that the update equations derived from equation (2.35) are guaranteed to converge, albeit not necessarily to the global optimum [17]. In order to increase the chance of finding the global maximum, it is generally recommended to restart the iteration from a set of randomized initial conditions. The whole update process is summarized as pseudo-code listing in algorithm 2 and illustrated graphically as a flow chart in figure 4.9.

In the following, we will present a validation of the variational Bayes approximation to the embedded clustering model derived in this section on artificial datasets. Just like in section 3.5.1, the motivation for using artificial data lies in the availability of ground truth data, which for the embedded clustering model has to include both the cluster labels and the DCM parameters.

Algorithm 2 Variational Update Scheme for the Embedded Clustering Model

function VBEC($\boldsymbol{y}, noIterations, noRestarts, \Delta\mathcal{F}$)
  for $idxRestart \leftarrow 1, \ldots, noRestarts$ **do**
    $\boldsymbol{\pi}, q_{nk}, \ldots, a_{r,n}, b_{r,n} \leftarrow$ randInit() ▷ randomize initial conditions
    for $idxIteration \leftarrow 1, \ldots, noIterations$ **do**
      $q_{nk} \leftarrow \text{update}(\boldsymbol{\alpha}, \boldsymbol{\mu}_n, \Sigma_n, \bar{\boldsymbol{\mu}}_k, \tau_k, \nu_k, \bar{\Sigma}_k, a_{r,n}, b_{r,n})$
      $\boldsymbol{\alpha} \leftarrow \text{update}(q_{nk}, \boldsymbol{\mu}_n, \Sigma_n, \bar{\boldsymbol{\mu}}_k, \tau_k, \nu_k, \bar{\Sigma}_k, a_{r,n}, b_{r,n})$
      $\boldsymbol{\mu}_n, \Sigma_n \leftarrow \text{update}(q_{nk}, \boldsymbol{\alpha}, \bar{\boldsymbol{\mu}}_k, \tau_k, \nu_k, \bar{\Sigma}_k, a_{r,n}, b_{r,n})$
      $\bar{\boldsymbol{\mu}}_k, \tau_k, \nu_k, \bar{\Sigma}_k \leftarrow \text{update}(q_{nk}, \boldsymbol{\alpha}, \boldsymbol{\mu}_n, \Sigma_n, a_{r,n}, b_{r,n})$
      $a_{r,n}, b_{r,n} \leftarrow \text{update}(q_{nk}, \boldsymbol{\alpha}, \boldsymbol{\mu}_n, \Sigma_n, \bar{\boldsymbol{\mu}}_k, \tau_k, \nu_k, \bar{\Sigma}_k)$
      $\mathcal{F}_{old} \leftarrow \mathcal{F}$
      $\mathcal{F} \leftarrow \text{update}(q_{nk}, \boldsymbol{\alpha}, \boldsymbol{\mu}_n, \Sigma_n, \bar{\boldsymbol{\mu}}_k, \tau_k, \nu_k, \bar{\Sigma}_k, a_{r,n}, b_{r,n})$
      if $\Delta\mathcal{F} > \mathcal{F} - \mathcal{F}_{old}$ **then**
        break
    save($\mathcal{F}, q_{nk}, \boldsymbol{\alpha}, \boldsymbol{\mu}_n, \Sigma_n, \bar{\boldsymbol{\mu}}_k, \tau_k, \nu_k, \bar{\Sigma}_k, a_{r,n}, b_{r,n}$)
  return

4.6 Validation with Artificial Data

In order to test the update scheme introduced in the last section, we have implemented the variational Bayes scheme for embedded clustering in Matlab. The first step in the validation process is to test the implementation with artificial data, before moving on to more realistic tests with real data. Similar to the BCG separation scheme from chapter 3, the motivation is to have ground truth available, which can be used to compare the results against. In the case of the embedded clustering model for DCM, the ground truth has to include the values for the DCM parameters, both connectivity parameters and hemodynamic parameters, as well as the correct values for the cluster labels.

For the tests, we used a very simple configuration with 40 simulated subjects in two well distinguishable groups with 20 subjects per group. The DCM contains two interacting brain regions and two inputs, such that the corresponding bilinear DCM equations are given by:

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\theta}_c) = (A + u_1 B^{(1)} + u_2 B^{(2)})\boldsymbol{x} + C\boldsymbol{u}, \tag{4.85}$$

with all matrices being 2×2. The connection parameter vector $\boldsymbol{\theta}$ consists of the concatenation of the elements of all matrices.

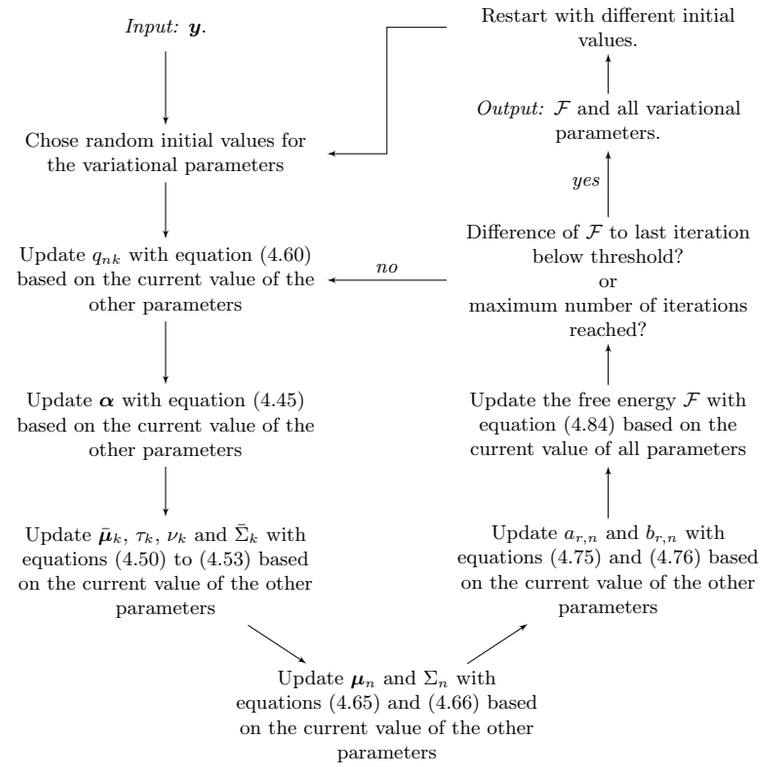


Figure 4.9: Flowchart of the variational update process for the embedded clustering model. The list of variational parameters include: q_{nk}, $\boldsymbol{\alpha}$, $\boldsymbol{\mu}_n$, Σ_n, $\bar{\boldsymbol{\mu}}_k$, τ_k, ν_k, $\bar{\Sigma}_k$, $a_{r,n}$ and $b_{r,n}$.

Before starting the variational Bayes scheme, we have to first choose values for the parameters of the prior distributions. As we have shown in the previous section, the values for $\alpha_{0,k}$ can be interpreted as the number of pseudo observations. Thus, with 40 subjects, any choice for $\alpha_{0,k}$ below one should provide a sufficiently weak prior.

A more critical choice is the value for $\bar{\Sigma}_0$, which, based on our tests, seems to be the only prior parameter that has a noticeable impact on the convergence of the update scheme. Since $\bar{\Sigma}_0/(\nu_0 - p_c - 1)$ is the mean of the inverse Wishart distribution [117], with p_c being the dimension of the matrix, choosing very large values for the elements of $\bar{\Sigma}_0$ means that the prior distribution will favour clusters with large covariance matrices. In extreme cases, this can lead to the variational scheme converging to a suboptimal

solution, where all data points are erroneously assigned to one cluster. This is not surprising, when considering that by combining all data points into one cluster, one obtains the maximum cluster covariance.

On the other hand, if one chooses very small values for the elements of $\bar{\Sigma}_0$, the prior distribution will favour clusters with small covariance matrices and additionally, it will slow down the convergence of the update process. To understand this, one can either inspect the update equation for the cluster labels (eq (4.60)), which shows that q_{nk} depends on $\bar{\Sigma}_k^{-1}$. Because $\bar{\Sigma}_k$ is close to $\bar{\Sigma}_0$ at the beginning of the iteration process, a small value for $\bar{\Sigma}_0$ will cause the cluster which is nearest to the data point to dominate equation (4.60). This means that the cluster assignments q_{nk} for a subject will be dominated by one cluster and change only very slowly, and since all other update equations depend on q_{nk}, they will also change very slowly. Alternatively, this effect can also be understood intuitively by noting that for small elements of $\bar{\Sigma}_0$, the prior distribution will favour clusters with small covariance matrices. And this in turn means that data points will favour the nearest cluster and stick to this assignment.

To find a suitable prior, one should note that the parameter ν_0 in the inverse Wishart distribution can be interpreted as the number of pseudo observations [117]. Therefore, in the case of the inverse Wishart distribution, choosing a weak prior corresponds to setting ν_0 to a small value.

In our tests with artificial data, we tried out different settings for the prior distribution and found that a typical successful run of the variational Bayes scheme takes about 10 min on a computer with a 2.5 GHz CPU and 8 GB of memory. The entire code was implemented in Matlab with the exception of the code which evaluates the $\boldsymbol{g}$-function from the DCM model, which was implemented using the C-language to increase efficiency. In comparison, the Monte Carlo version of the embedded clustering model inversion has a typical running time of about 500 min on a cluster of computers [115]. Although, the duration of each individual run will vary, this still shows that the variational Bayes scheme is significantly faster. In addition, analysis of the variational Bayes implementation showed that the potential speed-up gained by parallelizing the code could be as high as 90 % of the current running time.

Thus, the tests on artificial data showed that the computational complexity of the variational Bayes scheme for embedded clustering compares quite favourably to its Monte Carlo counterpart, which is not surprising since speed is one of the advantages of variational approximations. On the other hand, the advantage of Monte Carlo methods lies in the accuracy of the results. In this respect, the test have shown a few interesting results for the variational approximation.

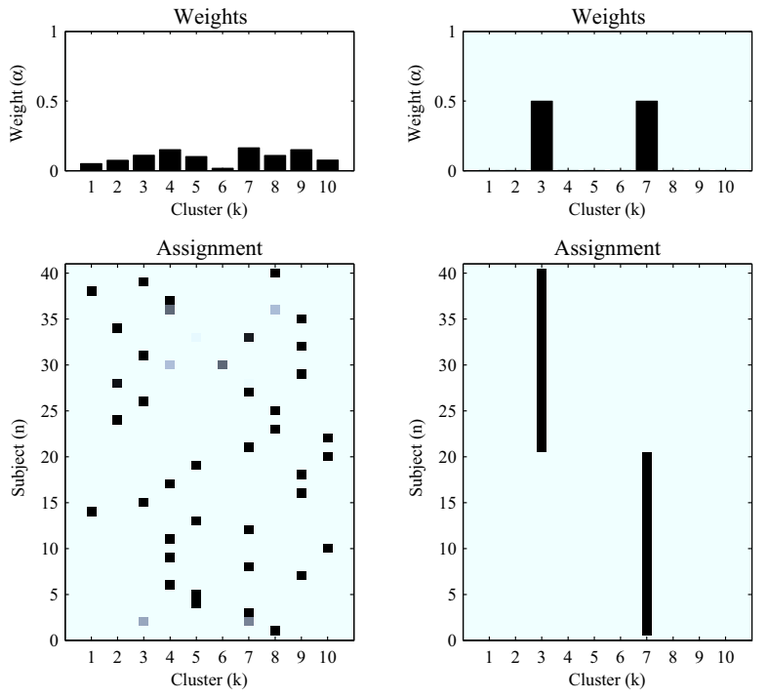


Figure 4.10: Cluster weights $\boldsymbol{\pi}$ (top) and assignments q_{nk} (bottom) after initialization (left column) and at convergence (right column). A grey-scale is used to encode the value of q_{nk}, which ranges from $q_{nk} = 0$ (white) to $q_{nk} = 1$ (black).

Figure 4.11 illustrates the cluster weights $\boldsymbol{\pi}$ as bar graphs in the upper row and the assignments q_{nk} as a matrix of rectangles in the lower row. The value of the q_{nk} is expressed on a grey-scale, where darker colours mean that the value of q_{nk} is closer to one and lighter colours mean that the value is closer to zero. The left column shows the cluster assignment and weights in their initial condition, which is random and the right column shows the values at convergence.

As shown in the graph, we have chosen $K = 10$. However, the variational scheme correctly identifies that the subjects belong to only two clusters and effectively switches off the remaining clusters, by assigning them zero weight. This is a very typical behaviour of variational approximations to clustering models, which can also be observed for the variational approximation to the mixture of Gaussians model [17]. And at the same time, if offers an elegant

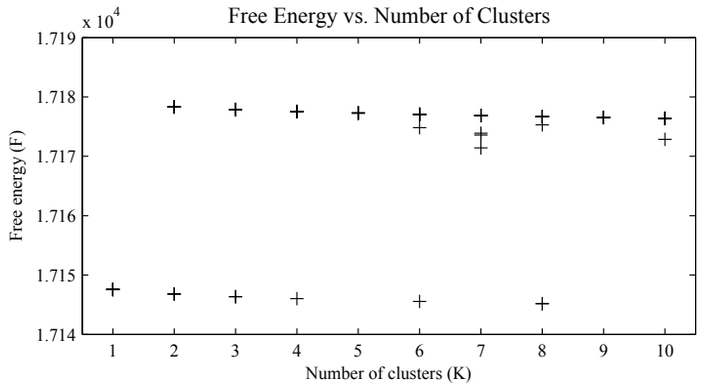


Figure 4.11: Free energy from the variational Bayes scheme for embedded clustering plotted versus the number of clusters.

solution to the model comparison problem of selecting the correct number of clusters mentioned in section 4.3.1.

Another way of estimating the optimal number of clusters K, is to repeat the variational Bayes scheme for different numbers of K and plotting the free energy $\mathcal{F}$, which is an approximation of the model evidence, against K. Figure 4.11 shows such a plot, and it can be seen that the free energy for $K = 1$ is considerably lower than for higher values of K, since the correct number of clusters is two. We also see that some of the trials of the variational scheme get stuck in local optima, which is why we repeat the variational Bayes scheme several times with random initial conditions for each value of K.

On the other hand, the embedded clustering model consists not only of the clustering part, but also includes the DCM part. Figure 4.12 shows exemplary DCM connectivity parameter estimates $\boldsymbol{\mu}_{c,n}$ for selected subjects of both groups, along with the ground truth. DCM parameter estimates of all subjects are shown in figures C.1 and C.2 in the appendix.

The error bars in figure 4.12 indicate two times the marginal standard deviation, which is obtained by extracting the diagonal elements of Σ_n and taking the square root. For each subject, we have subtracted the cluster mean $\bar{\boldsymbol{\mu}}_k$ from both the DCM parameter estimate $\boldsymbol{\mu}_{c,n}$ and the ground truth, which means that in the panels showing the DCM estimates, the baseline corresponds to the curve shown in the bottom panel of each column. This is done because, from the point of the DCM connectivity parameter $\boldsymbol{\theta}_{c,n}$, the clustering model acts like a prior, since it is the parent node of $\boldsymbol{\theta}_{c,n}$ in the

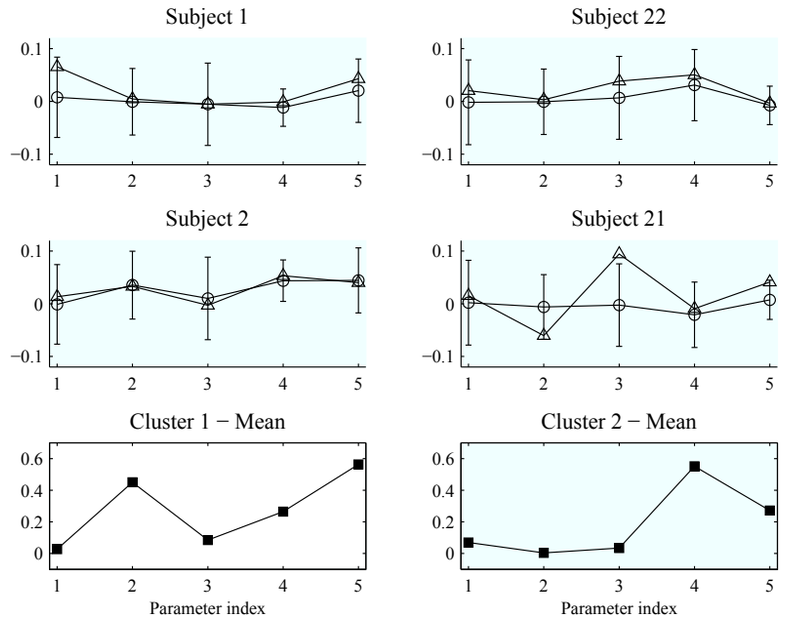


Figure 4.12: DCM connectivity parameters estimates $\boldsymbol{\mu}_{c,n}$ (○) and ground truth (△) for selected subjects from group one (left column) and group two (right column). Error bars correspond to two standard deviations. Baseline in the upper panels correspond to the cluster mean $\bar{\boldsymbol{\mu}}_k$ shown at the bottom of the respective column. Detailed explanation see text.

graphical model (fig 4.8). And indeed, we can observe that the value of most DCM parameter estimates is between the true value and the pseudo-prior given by the baseline, which corresponds to the cluster mean. This is a well-known behaviour of Bayesian inference techniques [17, 117].

However, the degree to which the estimates of the subjects are influenced by the cluster mean varies substantially. For some subjects, the estimates are almost in perfect agreement with the ground truth (e.g. subject number 37), while for other subjects the estimates are drawn entirely to the cluster mean (e.g. subject number 25). The reason for this variance is currently unknown and will be subject to further investigation. On the other hand, the results show that for most estimates, even those that are strongly influenced by the cluster mean, the true value lies within two standard deviations of the estimate.

Additionally, we have observed that the result of the variational approxima-

tion to the embedded clustering model is relatively independent between the two parts of the model. In particular, the clustering part can fail to identify the two clusters as separate clusters when choosing extremely large values for the prior parameter $\bar{\Sigma}_k$, which are more than two magnitudes larger than the true cluster covariances. However, the DCM parameter estimates $\boldsymbol{\mu}_{c,n}$ are almost unaffected.

In addition to the variational distribution q, the variational Bayes scheme also provides a value for the free energy $\mathcal{F}$, which is a lower bound on the model evidence, and which can be viewed as an approximation to the evidence when maximized. Figure 4.13 shows the development of free energy plotted against the iteration count over the course of a typical run of the variational Bayes scheme.

A property of the update scheme introduced in section 2.4.2, which we applied to the embedded clustering model in section 4.5, is that with each update step, the free energy is guaranteed to increase. And since $\mathcal{F}$ is a lower bound, this is also the reason why the scheme is guaranteed to converge. However, since we had to apply the Taylor approximation to the $\boldsymbol{g}$-function in the derivation of the update equations for the DCM parameters (eq (4.65) to (4.66)) and the noise precision matrix (eq (4.75) to (4.76)), these guarantees are not valid any more. This means that when evaluating the update equations from section 4.5, the free energy can also decrease, which we do observe in practice.

One possible reason for the occurrence of a decrease in $\mathcal{F}$ is related to the stability of the bilinear DCM system equations (4.2). As noted in section 4.2, every plausible configuration of the DCM connectivity parameters $\boldsymbol{\theta}_{c,n}$ must ensure that equation (4.2) is stable. However, if the value of $\boldsymbol{\theta}_{c,n}$ causes equation (4.2) to become unstable, the brain activity $\boldsymbol{x}$ and subsequently the output of $\boldsymbol{g}$ will diverge. From a modelling point, the observation produced by a diverging system should be very implausible and receive a low probability under the model. Thus, when the maximization algorithm for $\mathcal{F}$, which is a lower bound on the probability of the data, is working perfectly, it will automatically steer away from the unstable regions of $\boldsymbol{\theta}_{c,n}$. If we remember that some of the update equations contain a first order approximation for $\boldsymbol{g}$, what could be happening is that in places where $\boldsymbol{g}$ is very nonlinear, the approximation is so inaccurate that the new values of the parameters calculated from the update equations lead to an unstable system with low probability, and consequently with low $\mathcal{F}$.

In practice, we have observed that cases of decreasing free energy occur more frequently with certain settings of the prior distribution. However, we do not think that the prior itself is causing the decrease. Rather, we believe that the prior is causing a preference for regions of parameter space

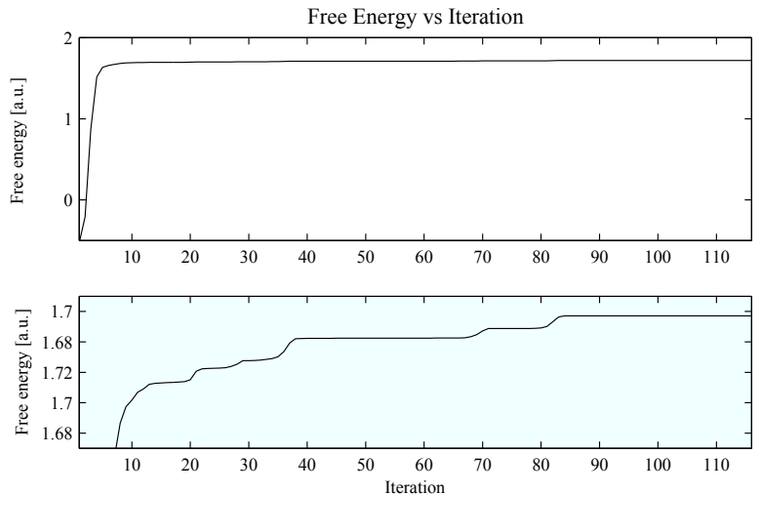


Figure 4.13: Free energy versus iteration count. Top panel shows the entire range of the free energy during the complete optimization process. Bottom panel shows a close up of the free energy close to convergence.

which are closer to the unstable configurations of the system, such that the phenomena described above would happen more frequently. Further tests have to be conducted to confirm these assumptions and to understand how to avoid the decrease in $\mathcal{F}$ during the update scheme.

4.7 Summary

In this chapter, we have discussed the embedded clustering problem along with its solution via the variational Bayes scheme. Embedded clustering provides a framework for unifying the estimation of effective brain connectivity from fMRI data and the clustering of brain connectivity estimates for group studies.

For the estimation of effective brain connectivity from fMRI data, we use DCM, which has already been established as a reliable method for single subject analysis [88]. In the embedded clustering framework, DCM based estimation is combined with the mixture of Gaussians model for joint DCM inversion and clustering via the variational Bayes approximation.

The main contribution of this chapter is the derivation of the variational

update equations, which can be used to achieve an approximate inversion of the embedded clustering model. These equations not only provide an iterative approximation to the posterior density, but also provide a solution for the maximization of the free energy of the system, which itself is a lower bound of the model evidence. Another advantage of the variational Bayes approximation is that the number of clusters is determined automatically, which solves the model selection problem introduced in the clustering part of the embedded clustering framework.

The solution is tested on artificial fMRI data, which provides ground truth for both the brain connectivity parameters, as well as the cluster labels. Preliminary analysis indicate that the variational approximation is able to find the correct solution under most circumstances and that the connectivity estimates are relatively independent from the solution to the clustering problem. This means that connectivity estimates can be correct, even when the clustering fails. Extreme choices for the prior distributions can have an impact on the convergence of the variational scheme, which can cause the algorithm to either converge to a local maximum or to not converge at all. However, in case of successful convergence, the variational Bayes scheme presented here is several magnitudes faster than the Monte Carlo scheme proposed for embedded clustering.

Chapter 5

Conclusion

In this thesis, we have discussed aspects of algorithm design, modelling and the interaction between these two topics, with focus on applications in the field of biomedical signal processing. We demonstrated these concepts on two example modalities ballistocardiography (BCG) and functional magnetic resonance imaging (fMRI).

BCG is a method for recording mechanical heartbeat signals, which is currently a promising candidate modality for home monitoring scenarios. The algorithms and modelling aspects discussed in chapter 3 include separation of cardiac and respiratory BCG components, as well as deterministic modelling of BCG for verifying the separation algorithm. From the verified result of the BCG separation, we derived an improved BCG model, which is suitable for statistical inference. As an example, we demonstrated model-based heartbeat detection in BCG via Markov chain Monte Carlo.

In chapter 4, we discussed fMRI applications in the context of embedded clustering. The most common form of fMRI is the blood-oxygenation-level dependent (BOLD) contrast, which relies on the dependency of the MR signal on the concentration of deoxyhaemoglobin. Thus, the fMRI BOLD signal is indirectly related to neuronal brain activity and is one the most common non-invasive modality for functional brain studies.

In our discussion, fMRI BOLD provides the data basis for embedded clustering, which unifies two concepts. On the one hand, we have dynamic causal modelling (DCM), which provides a model for effective brain region connectivity in the context of a psychological experiment. On the other hand, we have mixture models, which can be used for unsupervised learning applications, e.g. clustering a set of data points into clusters containing similar data points. These two concepts are unified in the embedded clustering

framework, where the effective brain region connectivity estimated via DCM forms the input of a clustering model. The novel aspect is that the inversion of the embedded clustering model will jointly invert the DCM and perform the clustering in one step. Our contribution is the solution of the embedded clustering problem via variational Bayes.

From an application centric viewpoint, we have discussed two different modalities with very different applications in their respective fields of research. BCG recording devices are mostly developed for an intended application in home monitoring settings, which are characterized by an extremely noisy and uncertain environment. This requires simple and robust sensors leading to system which are very simple to describe but have limited sensing capability. On the other hand, fMRI BOLD is a modality used in neuro-scientific or psychological studies. They consist of highly sophisticated imaging devices operated in a controlled environment, where the uncertainty originates from the complex and indirect measurement principle. However, the discussion in this thesis has shown that in both cases the uncertainty in the system requires a model-based approach, irrespective of its origin.

When viewed from a modelling perspective many common aspects between our treatment of the two modalities BCG and fMRI become clear. Both the nonlinear dynamic BCG model and the DCM are based on system equations of the same mathematical form, where the latent state variables evolve according to first order differential equations, while the observable, or its derivative, is a function of the state variable:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{\theta}) \tag{5.1}$$

$$\boldsymbol{y} = g(\boldsymbol{x}). \tag{5.2}$$

This is one of the most fundamental models, which can be used to describe any nonlinear dynamic systems [8] with static observation function, and its versatility is proven through the fact that it can model both the BCG and the DCM, which are fundamentally different in nature.

A second common methodological topic in the treatment of both modalities is mixture modelling. We have seen that the mixture of Gaussians model plays an important role in the embedded clustering framework, where it is combined with DCM. In this context, the task of the mixture model is to solve the unsupervised learning problem of clustering. However, mixture models can also be applied to other problems, including density estimation. A model developed for this purpose is the mixture of probabilistic principal component analysers (PPCA).

Interestingly, the discussion in chapter 3 reveals a connection between the mixture of PPCA and the locally projective noise reduction (LPNR) algorithm used for BCG components separation: LPNR can be viewed as the

non-parametric and non-probabilistic form of a mixture of PCA model. This is remarkable, since no explicit modelling was involved in the development of the LPNR algorithms, which comes from a purely nonlinear dynamics background, without any machine learning influence.

This observation also serves as an example for one of the aspects mentioned in the introduction of this thesis, namely that re-examining a method in the light of a model-based viewpoint sometimes leads to reinterpretations, which can suggest potential extensions of the method. In the case of LPNR, one possibility is the extension to a non-parametric and probabilistic method, indicated in the lower right corner of table 3.1, via the introduction of the Dirichlet process [119]. At the same time, the Dirichlet process would also lead to a non-parametric version of the embedded clustering model. Thus, we see that also on the mixture modelling level, a model-based discussion leads to an unified view on the two different modalities.

In addition to possible reinterpretations, other aspects of the interaction between modelling and algorithm design mentioned in the introduction are the verification of both algorithms and models, as well as the improvement of models. The cycle from algorithm design to model-based verification to model improvement was demonstrated for the BCG separation in chapter 3, where the LPNR-based signal separation was validated using data generated from the nonlinear dynamic BCG model. Based on the cardiac BCG component extracted via LPNR, an improved, probabilistic and non-parametric BCG model was developed, which was capable of supporting statistical inference on BCG, thus closing the cycle. Figure 5.1 summarizes the relations between all the methods and algorithms discussed in the context of the two modalities BCG and fMRI.

In view of the numerous connections between BCG and fMRI which arise from the model-oriented discussion in this thesis, it becomes clear that this model-based approach provides a powerful, unifying viewpoint on aspects of biosignal processing, which are seemingly unrelated. The potential behind model-based analysis techniques has already been recognized in the field of computational psychiatry, where it is seen as a new inter-disciplinary approach, which could potentially lead to new diagnostic tools [18]. However, the results in this thesis indicate that it might sometimes even be worth to step across the boundaries between different fields of research, similar to how contributions from the nonlinear dynamics community demonstrated the application of many modelling concepts from physics to important problems in biology and medicine [8].

Aside from the topics mentioned above, we have also discussed a technical, but nonetheless important aspect which is approximate inference. Since model-based signal processing is often aimed at deriving estimates of hid-

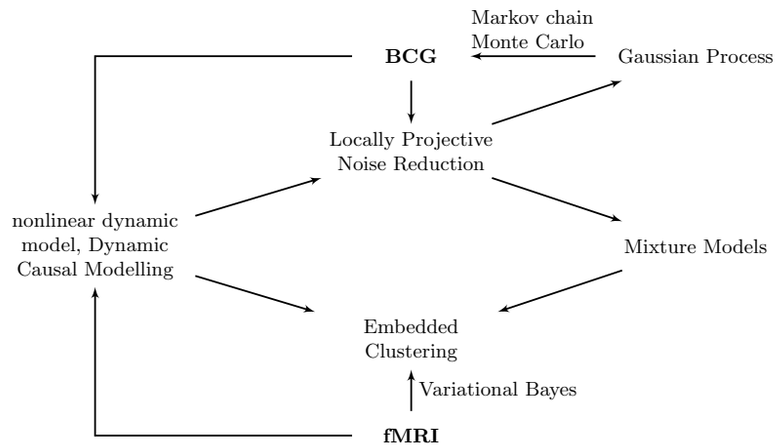


Figure 5.1: Relationship between the models, algorithms and the two modalities BCG and fMRI as discussed in this thesis.

den variables, i.e. variables which cannot be observed directly, from related observable variables, also called data, it is necessary to invert the model. For most models, inversion is not tractable, which means that approximate solutions have to be found.

Within this thesis, we introduced and applied two methods for approximate inference: Markov chain Monte Carlo and variational Bayes. Both methods have their root in statistical physics [16], with variational Bayes being closely related to the mean field approximation [17]. Indeed, many concepts and notions in statistical inference were initially borrowed from statistical physics. This includes the concept of entropy in information theory [41], or the idea of maximizing the variational free energy in the variational Bayes framework, where the variational free energy as given in equation (2.28) actually corresponds to the negative free energy in statistical physics [17].

In conclusion, we can state that the main contribution of this work is to show that the model-based approach to algorithm design provides a powerful framework for improving and extending methods in biosignal processing and related fields. The interaction between algorithm design and modelling offers benefits for both the signal processing community, as well as the modelling community, by pointing out common aspects of algorithms and models on many different levels.

On the application level, a model-oriented viewpoint can help to discover that algorithms for different, seemingly unrelated modalities have a common,

unifying basis. On a methodological level, we have shown how advances in algorithm design can lead to new or improved models, while a model-based view can lead to new insights into existing algorithms. Furthermore, on a technical level, we see that the exchange of ideas and concepts between the two unrelated fields of statistical physics and biosignal processing can lead to solutions of previously intractable inference problems in biosignal analysis.

Looking forward, the contributions from this thesis form the basis for a systematic, model-based approach to new and existing topics in biosignal processing. As mentioned several times throughout the thesis, the extension of the embedded clustering model and the LPNR algorithm to their respective non-parametric versions via the Dirichlet process is a promising first step. However, the improvements offered by a consistent model-based approach to biosignal analysis also opens the door to new applications, which have not been considered up until now. This, of course, also requires that powerful inference methods can be brought to bear on the problem.

A good example for such an application is the estimation of cardio-respiratory synchronization from BCG data. As introduced in chapter 1, cardio-respiratory synchronization is a potential indicator for cardio-vascular health and has been shown to be a very reliable predictor for sleep apnoea in infants [9, 10]. However, the assessment and quantification of cardio-respiratory synchronization requires access to the cardiac and respiratory phase, which normally necessitates the measurement of both ECG and respiratory effort. In this context BCG offers a simple alternative, since it contains information on both the cardiac and respiratory activity. On the other hand, even after separation of the BCG components, extracting the cardiac and respiratory phase is not a trivial task.

A potential solution to this problem could lie in a model-oriented approach based on the nonlinear dynamic BCG model presented in chapter 3, which contains cardiac and respiratory phase as hidden variables. Just as the embedded clustering model combined DCM with a mixture model, combining the BCG model with a model of synchronization between dynamic oscillators and applying a powerful inference scheme can provide a way for estimating cardio-respiratory synchronization from non-invasive, unobtrusive BCG recordings.

Although a detailed analysis of the framework described in the previous paragraph, or similar methods, is outside the scope of this thesis, these examples still serves as an insight into the potential that lies in the model-based approach to biosignal analysis, for which the basis has been laid out in this work.

In addition, the challenges presented by biosignals are also encountered in

many other areas of research and application. These challenges include low signal to noise ratio, high uncertainty of system or environment or the presence of hidden variables. Thus, the methods and models developed in this thesis are not restricted to biosignals, but can be applied to any field which has to deal with similar problems.

Appendix A

Mathematical Basics

A.1 Introduction to Lagrange Multiplier

Lagrange multiplier is a method used to solve constraint optimization problems. This is best explained using a simple example: Given two functions $f(x_1, x_2)$ and $g(x_1, x_2)$, the task is to maximize or minimize f with respect to the two arguments x_1 and x_2 under the constraint $g(x_1, x_2) = 0$.

It can be shown that for points (x_1, x_2) which are solutions to the above problem, there must exist a factor λ such that the following conditions are satisfied [17]:

$$\frac{\partial f(x_1, x_2)}{\partial x_1} + \lambda \frac{\partial g(x_1, x_2)}{\partial x_1} = 0, \tag{A.1}$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} + \lambda \frac{\partial g(x_1, x_2)}{\partial x_2} = 0 \quad \text{and} \tag{A.2}$$

$$g(x_1, x_2) = 0. \tag{A.3}$$

We now introduce a function called the Lagrangian, which depends on the eponymous Lagrange multiplier λ:

$$\mathcal{L}(x_1, x_2, \lambda) = f(x_1, x_2) + \lambda g(x_1, x_2). \tag{A.4}$$

By setting the gradient of the Lagrangian to zero, we recover the above conditions:

$$\frac{\partial \mathcal{L}(x_1, x_2, \lambda)}{\partial x_1} = \frac{\partial f(x_1, x_2)}{\partial x_1} + \lambda \frac{\partial g(x_1, x_2)}{\partial x_1}, \tag{A.5}$$

$$\frac{\partial \mathcal{L}(x_1, x_2, \lambda)}{\partial x_2} = \frac{\partial f(x_1, x_2)}{\partial x_2} + \lambda \frac{\partial g(x_1, x_2)}{\partial x_2} \quad \text{and} \tag{A.6}$$

$$\frac{\partial \mathcal{L}(x_1, x_2, \lambda)}{\partial \lambda} = g(x_1, x_2). \tag{A.7}$$

Thus, we see that by finding the stationary points of the Lagrangian, we have transformed the constrained optimization problem in two dimensions into a regular unconstrained optimization problem in three dimensions, which can be easily solved. In general, the Lagrange multiplier approach transforms a p-dimensional optimization problem with q constrains into a $p+q$-dimensional unconstrained optimization problem [17].

A.1.1 Lagrange Multiplier for LPNR

The derivation of the LPNR algorithm in section 3.2.1 involves the minimization of the average squared perturbation

$$L = \sum_{k=1}^{K} \sum_{q=1}^{Q} \left(\boldsymbol{a}_q (\boldsymbol{y}_k - \boldsymbol{x}_0) \right)^2 \tag{A.8}$$

with respect to $\boldsymbol{a}_q$, under the constraints:

$$\boldsymbol{a}_q (\boldsymbol{x}_k - \boldsymbol{x}_0) = 0, \tag{A.9}$$

$$\|\boldsymbol{a}_q\| = 1 \quad \text{and} \tag{A.10}$$

$$\boldsymbol{a}_p^T \boldsymbol{a}_q = \delta_{pq} \tag{A.11}$$

for $p, q = 1, \ldots, Q$ and $k = 1, \ldots, K$. Here, δ_{pq} denotes the Kronecker delta defined as:

$$\delta_{pq} = \begin{cases} 0 & \text{if } p \neq q \\ 1 & \text{if } p = q. \end{cases} \tag{A.12}$$

The Lagrangian for this problem, with Lagrange multiplier λ_{kq} and μ_{pq}, is given by:

$$\mathcal{L} = L + \sum_{k=1}^{K} \sum_{q=1}^{Q} \lambda_{kq} \boldsymbol{a}_q (\boldsymbol{x}_k - \boldsymbol{x}_0) + \sum_{p=1}^{Q} \sum_{q=1}^{Q} \mu_{pq} (\boldsymbol{a}_p^T \boldsymbol{a}_q - \delta_{pq}) \tag{A.13}$$

and setting its gradient to zero leads to the solution that the vectors $\boldsymbol{a}_q$ are the eigenvectors of the matrix C which corresponds to the Q smallest eigenvalues [57]. C is defined as:

$$C = \frac{1}{K} \sum_{k=1}^{K} (\boldsymbol{y}_k - \boldsymbol{x}_0)(\boldsymbol{y}_k - \boldsymbol{x}_0)^T, \tag{A.14}$$

which we can identify as the covariance matrix of the delay vectors $\boldsymbol{y}_k$. Note that the notation used here differs slightly from the notation in [57]. However, the two formulations are mathematically equivalent.

A.2 The Gamma and Digamma Functions

Throughout the thesis, we have made use of the gamma function $\Gamma(x)$ defined by:

$$\Gamma(x) = \int_0^\infty u^{x-1} \exp(-u) du. \tag{A.15}$$

The gamma function can be viewed as an extension of the factorial function to real valued arguments and for integer n, we have $\Gamma(n) = (n-1)!$. In addition, it possesses the following property $\Gamma(x+1) = x\Gamma(x)$ [16].

The digamma function $\Psi(x)$ is defined as the derivative of the natural logarithm of the gamma function [73]:

$$\Psi(x) = \frac{d}{dx} \log \Gamma(x). \tag{A.16}$$

Since the variational Bayes scheme for embedded clustering requires us to work a lot with the inverse Wishart distribution, we will introduce a shortcut from [37] for the product of gamma functions in the normalization constant of the inverse Wishart distribution. This will help to significantly improve readability of some formulas.

$$\Gamma_p(\nu) = \pi^{p(p-1)/4} \prod_{i=1}^{p} \Gamma\Big(\frac{\nu+1-i}{2}\Big). \tag{A.17}$$

Furthermore, we will define a shortcut for the sum of digamma functions:

$$\Psi_p(\nu) = \sum_{i=1}^{p} \Psi\Big(\frac{\nu+1-i}{2}\Big). \tag{A.18}$$

A.3 Important Probability Distributions

In this section, we will introduce important probability distributions, which are used throughout this thesis. Each description consists of a short comment on the distribution, as well as a mathematical description of the functional form and formulas for the first and second moments.

The Categorical Distribution

The categorical distribution is a discrete distribution over a fixed number of cases. It is often used to describe the class or category label in finite

mixture models, where a data point belongs to one of K possible classes or categories. This distribution is parameterized by a single vector $\boldsymbol{\pi}$ of length K containing real positive entries, which must sum to one. Its functional form is given by [117]:

$$p(d{=}k|\boldsymbol{\pi}) = \pi_k, \quad k = 1, \ldots, K \tag{A.19}$$

$$\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)^T. \tag{A.20}$$

The Dirichlet Distribution

The Dirichlet distribution $\mathcal{D}(\boldsymbol{\pi}|\boldsymbol{\alpha})$ is a distribution over vectors of a fixed dimension K which contain elements that are constrained to be positive and sum to one. In other words, the Dirichlet distribution is a distribution over the kind of probability vectors which are used as parameters for categorical distributions. As such, Dirichlet distributions are often used as prior distributions for the parameters of categorical distributions. The parameter of the Dirichlet distribution is a vector $\boldsymbol{\alpha}$ of the same dimension as its argument $\boldsymbol{\pi}$ that contains positive, but not necessarily normalized elements α_k, which can be interpreted as number of pseudo observations for each category [117].

$$\mathcal{D}(\boldsymbol{\pi}|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_{k=1}^{K} \alpha_k))}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1}, \quad \text{for } \pi_k \geq 0 \text{ and } \sum_{k=1}^{K} \pi_k = 1. \tag{A.21}$$

First moment:

$$\langle \boldsymbol{\pi} \rangle = \frac{\boldsymbol{\alpha}}{\sum_{k=1}^{K} \alpha_k}. \tag{A.22}$$

Other moments [17]:

$$\langle \log \pi_k \rangle = \Psi\big(\alpha_k\big) - \Psi\big(\sum_{k=1}^{K} \alpha_k\big). \tag{A.23}$$

The Gamma Distribution

The gamma distribution $\mathrm{Gam}(x|a, b)$ is the equivalent of the normal distribution for positive real numbers $x > 0$. It is the conjugate prior of the precision parameter of a one-dimensional Gaussian distribution. The gamma distribution is parameterized by two positive real numbers: the shape parameter a and the inverse scale parameter b [117].

$$\mathrm{Gam}(x|a, b,) = \frac{b^a}{\Gamma(a)} x^{a-1} \exp(-bx). \tag{A.24}$$

First moment:

$$\langle x \rangle = \frac{a}{b}. \tag{A.25}$$

The Gaussian Distribution

The Gaussian distribution $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma)$, also known as the normal distribution, is possibly the most well-known probability distribution, aside from the uniform distribution. It is defined over real vectors $\boldsymbol{x}$ (multivariate case) of fixed dimension D, and is parameterized by a mean vector $\boldsymbol{\mu}$ and a positive definite covariance matrix Σ. Among many other properties, the Gaussian distribution is fully described by its first two moments [117].

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma) = \frac{1}{\sqrt{|2\pi\Sigma|}} \exp\big(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\big). \tag{A.26}$$

First moment:

$$\langle \boldsymbol{x} \rangle = \boldsymbol{\mu}. \tag{A.27}$$

Second moment:

$$\big\langle (\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^T \big\rangle = \Sigma. \tag{A.28}$$

The inverse Gaussian Distribution

The inverse Gaussian (IG) distribution $\mathcal{IG}(t|\mu, \lambda)$ is a probability distribution with support over the non-negative real numbers, similar to the gamma distribution. It is the distribution of the first crossing time of a Gaussian random walk with drift [94] and has been used primarily as a model for failure time, but also as a model for heartbeat intervals [95]. The probability density function of the IG distribution, which has two parameters: mean μ and scale λ, is given by [73]:

$$\mathcal{IG}(x|\mu, \lambda) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp\big(-\frac{\lambda}{2\mu^2 x}(x-\mu)^2\big), \tag{A.29}$$

and the cumulative distribution function is given by [73]:

$$\mathcal{IG}'(x|\mu, \lambda) = \frac{1}{2}\Big(1+\mathrm{erf}\Big(\sqrt{\frac{\lambda}{2x}}\big(\frac{x}{\mu}-1\big)\Big)\Big) + \frac{\exp\big(\frac{2\lambda}{\mu}\big)}{2}\Big(1-\mathrm{erf}\Big(\sqrt{\frac{\lambda}{2x}}\big(\frac{x}{\mu}+1\big)\Big)\Big). \tag{A.30}$$

Here, $\mathrm{erf}(\cdot)$ denotes the error function obtained by integrating the scalar normal distribution:

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2)dt. \tag{A.31}$$

The Inverse Wishart Distribution

The inverse Wishart distribution $\mathcal{W}^{-1}(X|S,\nu)$ is a distribution over positive definite $p \times p$ matrices, which is often used as a conjugate prior on the covariance of a normal distribution. The two parameters are the scale matrix S of size $p \times p$ and the degrees of freedom ν, which can be interpreted as the number of pseudo observations contributed by the prior [117].

$$\mathcal{W}^{-1}(X|S,\nu) = \frac{|S|^{\nu/2}}{2^{\nu k/2}\Gamma_p(\nu)} \frac{\exp\left(-\frac{1}{2}\operatorname{tr}(SX^{-1})\right)}{|X|^{(\nu+p+1)/2}}, \tag{A.32}$$

where the definition of the shortcut $\Gamma_p(\nu)$ is given above in equation (A.17). First moment [117]:

$$\langle X \rangle = \frac{S}{\nu - p - 1}. \tag{A.33}$$

Other moments [37]:

$$\langle X^{-1} \rangle = \nu S^{-1} \tag{A.34}$$

$$\langle \log|X| \rangle = \log|S/2| - \Psi_p(\nu), \tag{A.35}$$

with $\Psi_p(\nu)$ being defined in equation (A.18).

The Normal-Inverse Wishart Distribution

The normal-inverse Wishart distribution is a combination of a multivariate Gaussian distribution with an inverse Wishart distribution of matching dimension, which defines a joint distribution over the mean and covariance of a Gaussian distribution. It is often used as a conjugate prior over the parameters of a normal distribution [16]. The functional form of the distribution is governed by four parameters: a p-dimensional vector $\boldsymbol{m}$, two scalars τ and ν and the $p \times p$-matrix S.

$$p(\boldsymbol{\mu}, \Sigma|\boldsymbol{m}, \tau, \nu, S) = \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{m}, \Sigma/\tau)\, \mathcal{W}^{-1}(\Sigma|S,\nu). \tag{A.36}$$

First and higher order moments [37, 117]:

$$\langle \boldsymbol{\mu} \rangle = \boldsymbol{m} \tag{A.37}$$

$$\langle X^{-1}\boldsymbol{\mu} \rangle = \nu S^{-1}\boldsymbol{m} \tag{A.38}$$

$$\langle \boldsymbol{\mu}^T X^{-1}\boldsymbol{\mu} \rangle = \frac{p}{\tau} + \nu \boldsymbol{m}^T S^{-1}\boldsymbol{m}. \tag{A.39}$$

Appendix B

Details of the MCMC Based BCG Beat Detection

In this chapter, we describe the MCMC scheme used in section 3.5.2 to infer the heartbeat times from BCG recordings and derive the acceptance ratios which are needed for satisfying the detailed balance condition. The version of the algorithm, which we proposed in [86], uses the GPML toolbox for Matlab [121] for the Gaussian process related calculations and custom code, also written in Matlab, for the MCMC related operations.

The basics of MCMC are provided in section 2.4.1. The method works by proposing new samples, which depend on the old sample, based on a proposal density $q(\boldsymbol{s}'; \boldsymbol{s})$. The proposal are either accepted or declined according to the acceptance probability given in equation (2.19).

The challenge in our method lies in the structure of the samples, which are vectors of heartbeat times. The number of heartbeats can vary, which means that we need to design a proposal scheme which explores samples of different sizes. In our solution, we mix proposals of three different types, with different proposal densities, which are derived in the following.

In case of the BCG model, the current sample $\boldsymbol{s} = (t_1, \ldots, t_b, \ldots, t_B)$ and the proposal $\boldsymbol{s}' = (t'_1, \ldots, t'_b, \ldots, t'_{B'})$ consist of a vectors of B and B' heartbeats, respectively. Beat intervals in state and proposal are defined as $\Delta t_b = t_{b+1} - t_b$ and $\Delta t'_b = t'_{b+1} - t'_b$. For the purpose of difference calculation, let $t_0 = t_{min}$ and $t_{B+1} = t_{max}$ correspond to the beginning and end of the observation interval. To calculate the acceptance probability, we need the joint distribution $p(\boldsymbol{y}, \boldsymbol{s})$ over observed BCG $\boldsymbol{y}$ and the sample $\boldsymbol{s}$ or $\boldsymbol{s}'$, which is given in logarithmic form by equation (3.45).

First type: *shift*
Given the old sample $\boldsymbol{s}$, chose a beat t_b and perturb its position without changing the overall order of beats in $\boldsymbol{s}$. For this purpose we define

$$T = t_{b+1} - t_{b-1} \quad \text{and} \tag{B.1}$$

$$\boldsymbol{\tau} = \frac{1}{T}(\Delta t_{b-1}, \Delta t_b)^T. \tag{B.2}$$

We then draw a random vector from a mixture of Dirichlet distributions:

$$\boldsymbol{\tau}' \sim \pi_1 \mathcal{D}(\boldsymbol{\tau}'|\alpha_1\boldsymbol{\tau}) + \pi_2 \mathcal{D}(\boldsymbol{\tau}'|\alpha_2\boldsymbol{\tau}), \tag{B.3}$$

where π_1 and $\pi_2 = 1 - \pi_1$ are the mixture coefficients, and α_1 and α_2 are parameters controlling the inverse step size. Finally, the new beat intervals are given by scaling $\boldsymbol{\tau}'$:

$$(\Delta t'_{b-1}, \Delta t'_b)^T = \boldsymbol{\tau}' T, \tag{B.4}$$

and the proposal $\boldsymbol{s}'$ is given by $t'_i = t_i$ for $i = 1, \ldots, B$, $i \neq b$ and $t'_b = t_{b-1} + \Delta t'_{b-1}$. The proposal density is:

$$q(\boldsymbol{s}'; \boldsymbol{s}) = \frac{1}{T}(\pi_1 \mathcal{D}(\boldsymbol{\tau}'|\alpha_1\boldsymbol{\tau}) + \pi_2 \mathcal{D}(\boldsymbol{\tau}'|\alpha_2\boldsymbol{\tau})), \tag{B.5}$$

which needs to be plugged into equation (2.19) for the acceptance ratio:

$$a = \min\left(1, \frac{p(\boldsymbol{y}, \boldsymbol{s}')}{p(\boldsymbol{y}, \boldsymbol{s})} \frac{\pi_1 \mathcal{D}(\boldsymbol{\tau}|\alpha_1\boldsymbol{\tau}') + \pi_2 \mathcal{D}(\boldsymbol{\tau}|\alpha_2\boldsymbol{\tau}')}{\pi_1 \mathcal{D}(\boldsymbol{\tau}'|\alpha_1\boldsymbol{\tau}) + \pi_2 \mathcal{D}(\boldsymbol{\tau}'|\alpha_2\boldsymbol{\tau})}\right). \tag{B.6}$$

Second type: *split*
Given the old sample, choose a beat t_b and split it into two consecutive beats. T is defined as above. In addition, we define $\boldsymbol{\alpha}_s = (\alpha_s, \alpha_s, \alpha_s)^T/3$, where α_s is an inverse step size parameter. We draw $\boldsymbol{\tau}'$, which is now a three-dimensional random vector, from a Dirichlet distribution

$$\boldsymbol{\tau}' \sim \mathcal{D}(\boldsymbol{\tau}'|\boldsymbol{\alpha}_s), \tag{B.7}$$

and obtain the new beat intervals via scaling:

$$(\Delta t'_{b-1}, \Delta t'_b, \Delta t'_{b+1})^T = \boldsymbol{\tau}' T. \tag{B.8}$$

The proposal $\boldsymbol{s}' = (t'_1, \ldots, t'_{B+1})$ now contains an additional beat:

$$\begin{aligned} t'_1 &= t_1 \\ &\vdots \\ t'_{b-1} &= t_{b-1} \\ t'_b &= t_{b-1} + \Delta t_{b-1} \\ t'_{b+1} &= t_b + \Delta t_b \\ t'_{b+2} &= t_{b+1} \\ &\vdots \\ t'_{B+1} &= t_B. \end{aligned}$$

The proposal density is given:

$$q(\boldsymbol{s}'; \boldsymbol{s}) = \frac{f_{split}}{T} \mathcal{D}(\boldsymbol{\tau}'|\boldsymbol{\alpha}_s), \tag{B.9}$$

where f_{split} is the frequency of occurrence of split type proposals relative to the other proposals types. Due to mixing different proposal types, the acceptance probability for split proposals

$$a = \min\left(1, \frac{p(\boldsymbol{y}, \boldsymbol{s}')}{p(\boldsymbol{y}, \boldsymbol{s})} \frac{f_{merge}\, \mathcal{D}(\boldsymbol{\tau}|\boldsymbol{\alpha}_m)}{f_{split}\, \mathcal{D}(\boldsymbol{\tau}'|\boldsymbol{\alpha}_s)}\right) \tag{B.10}$$

depends on the proposal density of merge proposals described below. f_{merge} is the frequency of occurrence of merge type proposals. And to simplify the algorithm, f_{merge} can be chosen equal to f_{split}.

Third type: *merge*
Given the old sample $\boldsymbol{s}$, choose t_b among the first $B-1$ beats and merge it with the following beat t_{b+1}. This time we define $T = t_{b+2} - t_{b-1}$ and $\boldsymbol{\alpha}_m = (\alpha_m, \alpha_m)^T/2$. We draw $\boldsymbol{\tau}'$ from a two-dimensional Dirichlet distribution and obtain the new beat intervals via scaling:

$$\boldsymbol{\tau}' \sim \mathcal{D}(\boldsymbol{\tau}'|\boldsymbol{\alpha}_m) \tag{B.11}$$

$$(\Delta t'_{b-1}, \Delta t'_b)^T = \boldsymbol{\tau}' T. \tag{B.12}$$

The proposal contains one beat less than the old sample:

$$\begin{aligned} t'_1 &= t_1 \\ &\vdots \\ t'_{b-1} &= t_{b-1} \\ t'_b &= t_{b-1} + \Delta t_{b-1} \end{aligned}$$

$$\begin{aligned} t'_{b+1} &= t_{b+2} \\ &\vdots \\ t'_{B-1} &= t_B, \end{aligned}$$

and the proposal density is given by:

$$q(\boldsymbol{s}'; \boldsymbol{s}) = \frac{f_{merge}}{T} \mathcal{D}(\boldsymbol{\tau}'|\boldsymbol{\alpha}_m).$$

Defining $\boldsymbol{\tau} = (\Delta t_{b-1}, \Delta t_b, \Delta t_{b+1})^T/T$, the acceptance probability is:

$$a = \min\Big(1, \frac{p(\boldsymbol{y}, \boldsymbol{s}')}{p(\boldsymbol{y}, \boldsymbol{s})} \frac{f_{split}\, \mathcal{D}(\boldsymbol{\tau}|\boldsymbol{\alpha}_s)}{f_{merge}\, \mathcal{D}(\boldsymbol{\tau}'|\boldsymbol{\alpha}_m)}\Big).$$

When running the MCMC scheme, we need to cycle through the three proposal types, where the shift proposal is responsible for exploring different positions of the heartbeats, while split and merge type proposals are responsible for exploring different numbers of beats.

Appendix C

Embedded Clustering

C.1 Solution of Integrals in the Update Equations

In this section, we will solve the integrals encountered during the derivation of the variational Bayes update equations for the embedded clustering framework in section 4.5.

Cluster parameters:
The derivation of the variational distribution over the cluster parameters $q(\boldsymbol{\mu}_k, \Sigma_k)$ in equation (4.48) contains the integral:

$$\int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n}) \log p(\boldsymbol{\theta}_{c,n}|d_n = k, \boldsymbol{\mu}_k, \Sigma_k) d\boldsymbol{\theta}_{c,n}. \tag{C.1}$$

Plugging in the definition of $p(\boldsymbol{\theta}_{c,n}|d_n = k, \boldsymbol{\mu}_k, \Sigma_k)$ from section 4.5, we have:

$$\int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n})(\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_k) d\boldsymbol{\theta}_{c,n} - \frac{1}{2} \log |\Sigma_k|, \tag{C.2}$$

and after expanding and reordering, the integral splits up into three parts:

$$\int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n}) \operatorname{tr}(\Sigma_k^{-1} (\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_{c,n})(\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_{c,n})^T) d\boldsymbol{\theta}_{c,n}, \tag{C.3}$$

$$-2 \int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n})(\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_{c,n})^T \Sigma_k^{-1} (\boldsymbol{\mu}_{c,n} - \boldsymbol{\mu}_k) d\boldsymbol{\theta}_{c,n} \quad \text{and} \tag{C.4}$$

$$\int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n})(\boldsymbol{\mu}_{c,n} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\boldsymbol{\mu}_{c,n} - \boldsymbol{\mu}_k) d\boldsymbol{\theta}_{c,n}. \tag{C.5}$$

Since $q(\boldsymbol{\theta}_{c,n})$ is a Gaussian distribution with mean $\boldsymbol{\mu}_{c,n}$ and covariance $\Sigma_{c,n}$, the solution to the three parts are given by:

$$\mathrm{tr}(\Sigma_k^{-1}\Sigma_{c_n}), \tag{C.6}$$

$$0 \qquad \text{and} \tag{C.7}$$

$$(\boldsymbol{\mu}_{c_n} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\boldsymbol{\mu}_{c_n} - \boldsymbol{\mu}_k). \tag{C.8}$$

Thus, the solution to the integral in equation (C.1) is:

$$\mathrm{tr}(\Sigma_k^{-1}\Sigma_{c_n}) + (\boldsymbol{\mu}_{c_n} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\boldsymbol{\mu}_{c_n} - \boldsymbol{\mu}_k) - \frac{1}{2}\log|\Sigma_k|. \tag{C.9}$$

Cluster labels:
The formula for the variational distribution over the cluster labels $q(d_n = k) = q_{nk}$ (eq (4.59)) contains two integrals:

$$\int q(\boldsymbol{\pi}) \log \pi_k d\boldsymbol{\pi} \qquad \text{and} \tag{C.10}$$

$$\int q(\boldsymbol{\theta}_{c,n}) \int q(\boldsymbol{\mu}_k, \Sigma_k) \log p(\boldsymbol{\theta}_{c,n} | d_n = k, \boldsymbol{\mu}_k, \Sigma_k) d\boldsymbol{\mu}_k d\Sigma_k \, d\boldsymbol{\theta}_{c,n}. \tag{C.11}$$

Since $q(\boldsymbol{\pi})$ is a Dirichlet distribution, the first integral is nothing else than the expectation of $\log \pi_k$ with respect to the Dirichlet distribution, which is given in equation (A.23):

$$\int q(\boldsymbol{\pi}) \log \pi_k d\boldsymbol{\pi} = \Psi\big(\alpha_k\big) - \Psi\big(\sum_{k=1}^{K} \alpha_k\big) \tag{C.12}$$

The second integral is a double integral, consisting of an inner and an outer integral. The inner integral is given by:

$$\int q(\boldsymbol{\mu}_k, \Sigma_k) \log p(\boldsymbol{\theta}_{c,n} | d_n = k, \boldsymbol{\mu}_k, \Sigma_k) d\boldsymbol{\mu}_k d\Sigma_k \tag{C.13}$$

$$= \int q(\boldsymbol{\mu}_k, \Sigma_k) \Big(-\frac{p_c}{2}\log 2\pi - \frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_k)\Big) d\boldsymbol{\mu}_k d\Sigma_k \tag{C.14}$$

$$= -\frac{p_c}{2}\log 2\pi - \frac{1}{2}\int q(\boldsymbol{\mu}_k, \Sigma_k) \log|\Sigma_k| d\boldsymbol{\mu}_k d\Sigma_k - \frac{1}{2}\int q(\boldsymbol{\mu}_k, \Sigma_k)(\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_k) d\boldsymbol{\mu}_k d\Sigma_k, \tag{C.15}$$

where we have plugged in the definition of $\log p(\boldsymbol{\theta}_{c,n} | d_n = k, \boldsymbol{\mu}_k, \Sigma_k)$ on the second line. Since $q(\boldsymbol{\mu}_k, \Sigma_k)$ is given by a normal-inverse Wishart distribution $\mathcal{N}(\boldsymbol{\mu}_k | \bar{\boldsymbol{\mu}}_k, \Sigma_k/\tau_k)\, \mathcal{W}^{-1}(\Sigma_k | \bar{\Sigma}_k, \nu_k)$, we recognize the last two terms

in equation (C.15) as moments of the normal-inverse Wishart distribution, which are given in equations (A.33) to (A.39).

Plugging the solution of equation (C.15) back into the outer integral in equation (C.11), we have:

$$\int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n})\Big(-\frac{1}{2}\log|\bar{\Sigma}_k| - \frac{p_c}{2}\log\pi + \frac{1}{2}\Psi_{p_c}(\nu_k) - \frac{p_c}{2\tau_k} - \frac{\nu_k}{2}(\boldsymbol{\theta}_{c,n}-\bar{\boldsymbol{\mu}}_k)^T\bar{\Sigma}_k^{-1}(\boldsymbol{\theta}_{c,n}-\bar{\boldsymbol{\mu}}_k)\Big)d\boldsymbol{\theta}_{c,n} \tag{C.16}$$

$$= -\frac{1}{2}\log|\bar{\Sigma}_k| - \frac{p_c}{2}\log\pi + \frac{1}{2}\Psi_{p_c}(\nu_k) - \frac{p_c}{2\tau_k} - \frac{\nu_k}{2}\int_{\boldsymbol{\theta}_{c,n}} q(\boldsymbol{\theta}_{c,n})(\boldsymbol{\theta}_{c,n}-\bar{\boldsymbol{\mu}}_k)^T\bar{\Sigma}_k^{-1}(\boldsymbol{\theta}_{c,n}-\bar{\boldsymbol{\mu}}_k)d\boldsymbol{\theta}_{c,n}. \tag{C.17}$$

The integral in the last line is identical to the one in equation (C.2), which has already been solved. Thus, the final result for the double integral in equation (C.11) is:

$$-\frac{1}{2}\log|\bar{\Sigma}_k| + \frac{1}{2}\Psi_{p_c}(\nu_k) - \frac{p_c}{2\tau_k} - \frac{\nu_k}{2}\operatorname{tr}(\bar{\Sigma}_k^{-1}\Sigma_{c,n}) - \frac{\nu_k}{2}(\boldsymbol{\mu}_{c,n}-\bar{\boldsymbol{\mu}}_k)^T\bar{\Sigma}_k^{-1}(\boldsymbol{\mu}_{c;n}-\bar{\boldsymbol{\mu}}_k) + const. \tag{C.18}$$

DCM parameters:
The derivation for the variational distribution over the DCM parameters $q(\boldsymbol{\theta}_n)$ in equation (4.64) contains the two integrals

$$\int q(\Lambda_n)\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)^T\Lambda_n\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)d\Lambda_n \quad \text{and} \tag{C.19}$$

$$\int q(\boldsymbol{\mu}_k,\Sigma_k)\log p(\boldsymbol{\theta}_{c,n}|d_n{=}k,\boldsymbol{\mu}_k,\Sigma_k)d\boldsymbol{\mu}_k d\Sigma_k. \tag{C.20}$$

As mentioned in section 4.5, we will apply a Taylor approximation to the nonlinear function $\boldsymbol{g}$:

$$\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n) \approx \boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\mu}_n) + \frac{\partial\boldsymbol{g}(\boldsymbol{\mu}_n)}{\partial\boldsymbol{\theta}_n}(\boldsymbol{\theta}_n - \boldsymbol{\mu}_n) \tag{C.21}$$

$$\approx \boldsymbol{\varepsilon}_n + G_n(\boldsymbol{\theta}_n - \boldsymbol{\mu}_n), \tag{C.22}$$

where we have chosen the expansion point to be the mean $\boldsymbol{\mu}_n$ of $q(\boldsymbol{\theta}_n)$. Plugging in this approximation and taking advantage of the diagonal structure of Λ_n detailed in section 4.2 (eq (4.10)), the integral in equation (C.19) can

be rewritten as:

$$\sum_{r=1}^{R} \int_{\lambda_{r,n}} q(\lambda_{r,n})\lambda_{r,n}\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)^T Q_r\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)\, d\lambda_{r,n} \tag{C.23}$$

$$= \sum_{r=1}^{R} \bar{\lambda}_{r,n}\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)^T Q_r\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big) \tag{C.24}$$

$$\approx \big(\boldsymbol{\varepsilon}_n - G_n(\boldsymbol{\theta}_n - \boldsymbol{\mu}_n)\big)^T \bar{\Lambda}_n\big(\boldsymbol{\varepsilon}_n - G_n(\boldsymbol{\theta}_n - \boldsymbol{\mu}_n)\big) \tag{C.25}$$

$$\approx \boldsymbol{\theta}_n^T G_n^T \bar{\Lambda}_n G_n \boldsymbol{\theta}_n + 2\boldsymbol{\theta}_n^T G_n^T \bar{\Lambda}_n(\boldsymbol{\varepsilon}_n + G_n\boldsymbol{\mu}_n) + const, \tag{C.26}$$

where $\bar{\lambda}_{r,n}$ and $\bar{\Lambda}_n$ are the mean region noise precision and mean subject noise precision matrix, defined in equations (4.79) and (4.81), respectively.

Plugging the definition of $p(\boldsymbol{\theta}_{c,n}|d_n = k, \boldsymbol{\mu}_k, \Sigma_k)$ from equation (4.30) into the second integral in equation (C.20), we obtain:

$$\int q(\boldsymbol{\mu}_k, \Sigma_k)(\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1}(\boldsymbol{\theta}_{c,n} - \boldsymbol{\mu}_k) d\boldsymbol{\mu}_k d\Sigma_k + const. \tag{C.27}$$

The solution for this integral is given by [37] as:

$$\boldsymbol{\theta}_{c,n}^T \nu_k \bar{\Sigma}_k^{-1} \boldsymbol{\theta}_{c,n} + 2\boldsymbol{\theta}_{c,n}^T \nu_k \bar{\Sigma}_k^{-1} \bar{\boldsymbol{\mu}}_k + const. \tag{C.28}$$

The update equations (4.65) to (4.66) are obtained by plugging the solutions (C.26) and (C.28) back into equation (4.64) and comparing the resulting expression with the logarithmic form of the Gaussian distribution (eq A.26).

Noise precision:
The variational distribution over the noise precision $q(\{\lambda_{r,n}\})$ in equation (4.71) requires the solution of:

$$\int_{\boldsymbol{\theta}_n} q(\boldsymbol{\theta}_n)\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big)^T Q_r\big(\boldsymbol{y}_n - \boldsymbol{g}(\boldsymbol{\theta}_n)\big) d\boldsymbol{\theta}_n. \tag{C.29}$$

After applying the Taylor approximation from equation (C.22) and rearranging the terms, we obtain:

$$\int_{\boldsymbol{\theta}_n} q(\boldsymbol{\theta}_n)\Big(\boldsymbol{\varepsilon}_n^T Q_r \boldsymbol{\varepsilon}_n - 2G_n(\boldsymbol{\theta}_n - \boldsymbol{\mu}_n)^T Q_r \boldsymbol{\varepsilon}_n + \operatorname{tr}\big(G_n^T Q_r G_n(\boldsymbol{\theta}_n - \boldsymbol{\mu}_n)(\boldsymbol{\theta}_n - \boldsymbol{\mu}_n)^T\big)\Big) d\boldsymbol{\theta}_n \tag{C.30}$$

Considering that $q(\boldsymbol{\theta}_n)$ is given by a Gaussian distribution with mean $\boldsymbol{\mu}_n$ and covariance Σ_n, the solution to the integral above is given by:

$$\boldsymbol{\varepsilon}_n^T Q_r \boldsymbol{\varepsilon}_n + \mathrm{tr}\left(G_n^T Q_r G_n \Sigma_n\right). \tag{C.31}$$

C.2 DCM Parameter Estimates

Below, we provide figures showing the estimates of the DCM connectivity parameters $\boldsymbol{\mu}_{c,n}$ for all subjects obtained via the variational Bayes approximation to the embedded clustering framework. A detailed analysis of these results is provided in section 4.6.

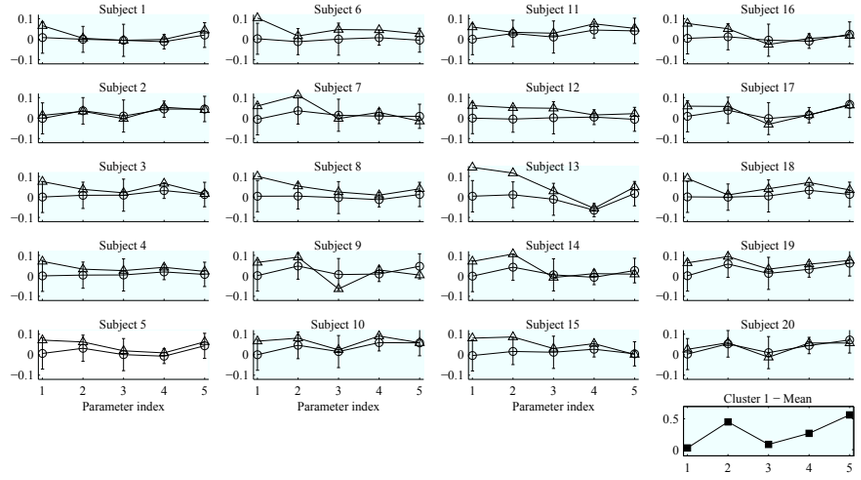


Figure C.1: DCM parameter estimates $\boldsymbol{\mu}_{c,n}$ (○) and ground truth $\boldsymbol{\theta}_c$ (△) for subjects of group one. Error bars correspond to two standard deviations. Baseline in all panels correspond to the cluster mean $\boldsymbol{\mu}_k$ shown in the lower right panel. See page 130 for detailed analysis.

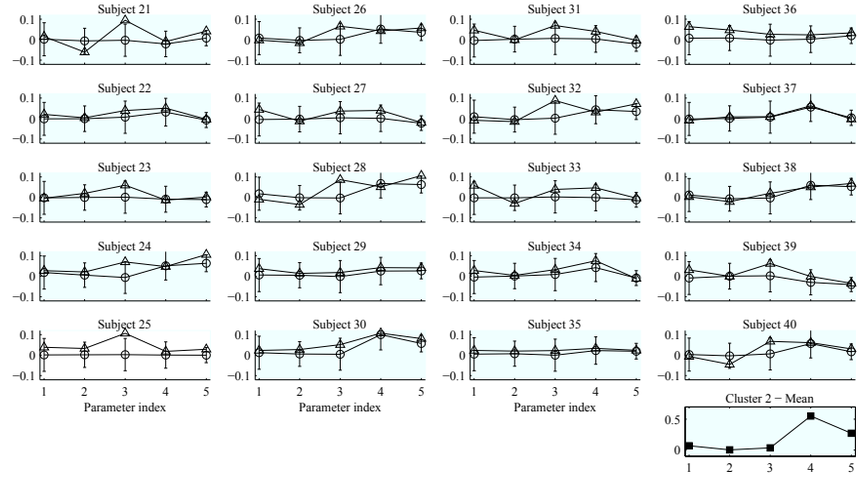


Figure C.2: DCM parameter estimates $\boldsymbol{\mu}_{c,n}$ (○) and ground truth $\boldsymbol{\theta}_c$ (△) for subjects of group two. Error bars correspond to two standard deviations. Baseline in all panels correspond to the cluster mean $\boldsymbol{\mu}_k$ shown in the lower right panel. See page 130 for detailed analysis.

Bibliography

[1] U. Pietrzyk and K. A. Bente, "Erzeugung von Modelldaten zur Prüfung von Bildregistrierungstechniken angewandt auf Daten aus der PET und MRT," in *Bildverarbeitung für die Medizin 2003*, ser. Informatik aktuell, T. Wittenberg, P. Hastreiter, U. Hoppe, H. Handels, A. Horsch, and H.-P. Meinzer, Eds. Springer Berlin Heidelberg, 2003, ch. 8, pp. 36–40.

[2] K. Amunts, C. Lepage, L. Borgeat, H. Mohlberg, T. Dickscheid, M.-É. Rousseau, S. Bludau, P.-L. Bazin, L. B. Lewis, A.-M. Oros-Peusquens, N. J. Shah, T. Lippert, K. Zilles, and A. C. Evans, "BigBrain: An ultrahigh-resolution 3D human brain model," *Science*, vol. 340, no. 6139, pp. 1472–1475, 2013.

[3] U. Pietrzyk and M. Khodaverdi, "Der Blick ins Gehirn: Bildgebende Verfahren in der Gehirnforschung," *Physik in unserer Zeit*, vol. 37, no. 5, pp. 235–240, 2006.

[4] U. Pietrzyk, K. Herholz, A. Schuster, H.-M. v. Stockhausen, H. Lucht, and W.-D. Heiss, "Clinical applications of registration and fusion of multimodality brain images from PET, SPECT, CT, and MRI," *European Journal of Radiology*, vol. 21, no. 3, pp. 174–182, 1996.

[5] U. Pietrzyk, "Does PET/CT render software registration obsolete?" *Nuklearmedizin*, vol. 44, no. 5a, pp. 13–17, 2005.

[6] U. Pietrzyk and H. Herzog, "Does PET/MR in human brain imaging provide optimal co-registration? A critical reflection," *Magnetic Resonance Materials in Physics, Biology and Medicine*, vol. 26, no. 1, pp. 137–147, 2013.

[7] Y. Kuramoto, "Self-entrainment of a population of coupled non-linear oscillators," in *International Symposium on Mathematical Problems in Theoretical Physics*, ser. Lecture Notes in Physics, H. Araki, Ed. Springer Berlin Heidelberg, 1975, vol. 39, ch. 71, pp. 420–422.

[8] S. Strogatz, *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*, ser. Addison-Wesley studies in nonlinearity. Boulder, Colorado: Westview Press, 1994.

[9] M. Schiek, F. Drepper, R. Engbert, H. Abel, and K. Suder, "Cardiorespiratory synchronization," in *Nonlinear Analysis of Physiological Data*, H. Kantz, J. Kurths, and G. Mayer-Kress, Eds. Berlin, Germany: Springer, 1998, pp. 191–209.

[10] M. Schiek, "Zeitreihenanalyse der kardiorespiratorischen Synchronisation," 1998.

[11] C. Schäfer, M. G. Rosenblum, J. Kurths, and H.-H. Abel, "Heartbeat synchronized with ventilation," *Nature*, vol. 392, no. 6673, pp. 239–240, 1998.

[12] J. Dammers, M. Schiek, F. Boers, C. Silex, M. Zvyagintsev, U. Pietrzyk, and K. Mathiak, "Integration of amplitude and phase statistics for complete artifact removal in independent components of neuromagnetic recordings," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 10, pp. 2353–2362, 2008.

[13] J. Dammers and M. Schiek, "Detection of artifacts and brain responses using instantaneous phase statistics in independent components," in *Magnetoencephalography*, E. W. Pang, Ed. InTech, 2011, pp. 131–150.

[14] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, 1st ed., ser. Cambridge Nonlinear Series. Cambridge: Cambridge University Press, 2002.

[15] T. Lippert, *Recent Developments in Supercomputing*, ser. NIC series. Jülich: John von Neumann Institute for Computing, 2008, vol. 39, ch. 1, pp. 1–8.

[16] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*, repr. with corr. ed. Cambridge: Cambridge University Press, 2004.

[17] C. Bishop, *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics. Cambridge: Springer, 2006.

[18] X.-J. Wang and J. Krystal, "Computational psychiatry," *Neuron*, vol. 84, no. 3, pp. 638–654, 2014.

[19] A. Klümper, R. Raupach, and F. Schönfeld, "Finite temperature density-matrix-renormalization-group investigation of the spin-peierls transition in $CuGeO_3$," *Physical Review B*, vol. 59, no. 5, pp. 3612–3616, 1999.

[20] A. Klümper and D. C. Johnston, "Thermodynamics of the spin-1/2 antiferromagnetic uniform Heisenberg chain," *Physical Review Letters*, vol. 84, no. 20, pp. 4701–4704, 2000.

[21] A. Klümper and O. I. Pâţu, "Efficient thermodynamic description of multicomponent one-dimensional Bose gases," *Physical Review A*, vol. 84, no. 5, p. 051604, 2011.

[22] T. Lippert, "Recent development and perspectives of machines for lattice QCD," *Nuclear Physics B - Proceedings Supplements*, vol. 129–130, no. 0, pp. 88–101, 2004.

[23] H. Weber, *Einführung in die Wahrscheinlichkeitsrechnung und Statistik für Ingenieure*. Stuttgart: Teubner, 1992.

[24] H.-O. Georgii, *Stochastik: Einführung in die Wahrscheinlichkeitstheorie und Statistik*, 4th ed. Berlin: de Gruyter, 2009.

[25] O. Cappé, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential Monte Carlo," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 899–924, 2007.

[26] F. V. Jensen, *Introduction to Bayesian Networks*, 1st ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1996.

[27] S. Lauritzen, *Graphical Models*, ser. Oxford Statistical Science Series. Clarendon Press, 1996.

[28] W. R. Gilks, S. Richardson, and D. Spiegelhalter, Eds., *Markov chain Monte Carlo in practice*, ser. Interdisciplinary statistics series. Boca Raton, Fla.: Chapman & Hall/CRC, 1996.

[29] R. Neal, "Probabilistic inference using Markov chain Monte Carlo methods," Department of Computer Science, University of Toronto, Canada, Tech. Rep. CRG-TR-93-1, 1993.

[30] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[31] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[32] P. Cheeseman and J. Stutz, "Bayesian classification (AutoClass): Theory and results," in *Advances in Knowledge Discovery and Data Mining*, U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1996, pp. 153–180.

[33] H. Attias, "Inferring parameters and structure of latent variable models by variational Bayes," in *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, ser. UAI'99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 21–30.

[34] T. S. Jaakkola and M. I. Jordan, "Bayesian parameter estimation via variational methods," *Statistics and Computing*, vol. 10, no. 1, pp. 25–37, 2000.

[35] T. S. Jaakkola, "Tutorial on variational approximation methods," in *Advanced Mean Field Methods: Theory and Practice*, M. Opper and D. Saad, Eds. MIT Press, 2001, pp. 129–159.

[36] K. Friston, J. Mattout, N. Trujillo-Barreto, J. Ashburner, and W. Penny, "Variational free energy and the Laplace approximation," *NeuroImage*, vol. 34, no. 1, pp. 220–234, 2007.

[37] T. P. Minka, "Using lower bounds to approximate integrals," 2001.

[38] K. Huang, *Introduction to Statistical Physics*. London; New York: Taylor & Francis, 2001.

[39] L. E. Reichl, *A Modern Course in Statistical Physics*, ser. Physics textbook. Weinheim WILEY-VCH, 2009. 3., rev. and updated ed., 2009.

[40] K. Friston, N. Trujillobarreto, and J. Daunizeau, "DEM: A variational treatment of dynamic systems," *NeuroImage*, vol. 41, no. 3, pp. 849–885, 2008.

[41] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[42] K. E. Stephan, W. D. Penny, J. Daunizeau, R. J. Moran, and K. J. Friston, "Bayesian model selection for group studies," *NeuroImage*, vol. 46, no. 4, pp. 1004–1017, 2009.

[43] R. E. Kass and A. E. Raftery, "Bayes factors," *Journal of the American Statistical Association*, vol. 90, no. 430, pp. 773–795, 1995.

[44] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[45] T. Schreiber and D. T. Kaplan, "Signal separation by nonlinear projections: The fetal electrocardiogram," *Physical Review E (Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics)*, vol. 53, no. 5, 1996.

[46] I. Starr and A. Noordergraaf, *Ballistocardiography in Cardiovascular Research: Physical Aspects of the Circulation in Health and Disease*. Philadelphia/ Montreal: Lippincott, 1967.

[47] L. Giovangrandi, O. T. Inan, R. M. Wiard, M. Etemadi, and G. T. A. Kovacs, "Ballistocardiography — a method worth revisiting," in *Proceedings of the 33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2011, pp. 4279–4282.

[48] O. T. Inan, M. Etemadi, R. M. Wiard, L. Giovangrandi, and G. T. A. Kovacs, "Robust ballistocardiogram acquisition for home monitoring," *Physiological Measurement*, vol. 30, no. 2, pp. 169–185, 2009.

[49] X. L. Aubert and A. Brauers, "Estimation of vital signs in bed from a single unobtrusive mechanical sensor: Algorithms and real-life evaluation," in *Proceedings of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, 2008, pp. 4744–4747.

[50] D. C. Mack, J. T. Patrie, P. M. Suratt, R. A. Felder, and M. Alwan, "Development and preliminary validation of heart rate and breathing rate detection using a passive, ballistocardiography-based sleep monitoring system," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 1, pp. 111–120, 2009.

[51] T. Koivistoinen, S. Junnila, A. Varri, and T. Koobi, "A new method for measuring the ballistocardiogram using EMFi sensors in a normal chair," in *Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, vol. 1, 2004, pp. 2026–2029.

[52] O. Postolache, P. S. Girao, and G. Postolache, "New approach on cardiac autonomic control estimation based on BCG processing," in *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, 2007, pp. 876–879.

[53] P. Castiglioni, A. Faini, G. Parati, and M. Di Rienzo, "Wearable seismocardiography," in *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, 2007, pp. 3954–3957.

[54] M. Di Rienzo, P. Meriggi, F. Rizzo, E. Vaini, A. Faini, G. Merati, G. Parati, and P. Castiglioni, "A wearable system for the seismocardiogram assessment in daily life conditions," in *Proceedings of the 33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2011, pp. 4263–4266.

[55] F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Warwick 1980*, ser. Lecture Notes in Mathematics, D. Rand and L.-S. Young, Eds. Springer Berlin Heidelberg, 1981, vol. 898, ch. 21, pp. 366–381.

[56] T. Sauer, J. A. Yorke, and M. Casdagli, "Embedology," *Journal of Statistical Physics*, vol. 65, no. 3–4, pp. 579–616, 1991.

[57] P. Grassberger, R. Hegger, H. Kantz, and C. Schaffrath, "On noise reduction methods for chaotic data," *Chaos*, vol. 3, no. 2, pp. 127–141, 1993.

[58] T. Schreiber and P. Grassberger, "A simple noise-reduction method for real data," *Physics Letters A*, vol. 160, no. 5, pp. 411–418, 1991.

[59] R. Cawley and G.-H. Hsu, "Local-geometric-projection method for noise reduction in chaotic maps and flows," *Physical Review A (Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics)*, vol. 46, no. 6, pp. 3057–3082, 1992.

[60] R. Cawley and G.-H. Hsu, "SNR perfomance of a noise reduction algorithm applied to coarsely sampled chaotic data," *Physics Letters A*, vol. 166, no. 3–4, pp. 188–196, 1992.

[61] T. Sauer, "A noise reduction method for signals from nonlinear systems," *Physica D: Nonlinear Phenomena*, vol. 58, no. 1–4, pp. 193–201, 1992.

[62] T. Schreiber and D. T. Kaplan, "Nonlinear noise reduction for electrocardiograms," *Chaos*, vol. 6, no. 1, pp. 87–92, 1996.

[63] T. Schreiber, "Interdisciplinary application of nonlinear time series methods," *Physics Reports-Review Section of Physics Letters*, vol. 308, no. 1, pp. 2–64, 1999.

[64] A. Effern, K. Lehnertz, G. Fernandez, T. Grunwald, P. David, and C. E. Elger, "Single trial analysis of event related potentials: Nonlinear de-noising with wavelets," *Clinical Neurophysiology*, vol. 111, no. 12, pp. 2255–2263, 2000.

[65] A. Effern, K. Lehnertz, T. Schreiber, T. Grunwald, P. David, and C. E. Elger, "Nonlinear denoising of transient signals with application to event-related potentials," *Physica D*, vol. 140, no. 3, pp. 257–266, 2000.

[66] R. Hegger, H. Kantz, and L. Matassini, "Denoising human speech signals using chaoslike features," *Physical Review Letters*, vol. 84, no. 14, pp. 3197–3200, 2000.

[67] R. Hegger, H. Kantz, and L. Matassini, "Noise reduction for human speech signals by local projections in embedding spaces," *IEEE Transactions on Circuits and Systems I-Fundamental Theory and Applications*, vol. 48, no. 12, pp. 1454–1461, 2001.

[68] A. I. Mees and K. Judd, "Dangers of geometric filtering," *Physica D: Nonlinear Phenomena*, vol. 68, no. 3, pp. 427–436, 1993.

[69] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of the Atmospheric Sciences*, vol. 20, no. 2, pp. 130–141, 1963.

[70] P. E. McSharry, G. D. Clifford, L. Tarassenko, and L. A. Smith, "A dynamical model for generating synthetic electrocardiogram signals," *IEEE Transactions on Biomedical Engineering*, vol. 50, pp. 289–294, 2003.

[71] R. Sameni, M. Shamsollahi, C. Jutten, and G. Clifford, "A nonlinear Bayesian filtering framework for ECG denoising," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 12, pp. 2172–2185, 2007.

[72] R. Sameni, M. B. Shamsollahi, and C. Jutten, "Model-based Bayesian filtering of cardiac contaminants from biomedical recordings," *Physiological Measurement*, vol. 29, no. 5, pp. 595–613, 2008.

[73] M. Abramowitz and I. A. Stegun, Eds., *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. New York: Dover Publications, 1972.

[74] T. Schreiber and M. Righter, "Fast nonlinear projective filtering in a data stream," *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering*, vol. 9, no. 10, pp. 2039–2045, 1999.

[75] D. S. Broomhead, R. Indik, A. C. Newell, and D. A. Rand, "Local adaptive Galerkin bases for large-dimensional dynamical systems," *Nonlinearity*, vol. 4, no. 2, pp. 159–197, 1991.

[76] G. E. Hinton, M. Revow, and P. Dayan, "Recognizing handwritten digits using mixtures of linear models," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds. Cambridge, Mass: MIT Press, 1995, pp. 1015–1022.

[77] G. E. Hinton, P. Dayan, and M. Revow, "Modeling the manifolds of images of handwritten digits," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 65–74, Jan 1997.

[78] N. Kambhatla and T. K. Leen, "Dimension reduction by local principal component analysis," *Neural Computation*, vol. 9, no. 7, pp. 1493–1516, 1997.

[79] R. Dony and S. Haykin, "Optimally adaptive transform coding," *IEEE Transactions on Image Processing*, vol. 4, no. 10, pp. 1358–1370, Oct 1995.

[80] C. Bregler and S. M. Omohundro, "Nonlinear image interpolation using manifold learning," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds. Cambridge, Mass: MIT Press, 1995, pp. 973–980.

[81] M. E. Tipping and C. M. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural Computation*, vol. 11, no. 2, pp. 443–482, 1999.

[82] M. Richter, T. Schreiber, and D. T. Kaplan, "Fetal ECG extraction with nonlinear state-space projections," *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 1, pp. 133–137, 1998.

[83] Y. Yao, C. Brüser, T. Vollmer, and M. Schiek, "Signal separation for ballistocardiography via locally projective noise reduction," in *World Congress on Medical Physics and Biomedical Engineering*, M. Long, Ed., vol. 39. Springer, 2012, pp. 501–504.

[84] Y. Yao, C. Brüser, U. Pietrzyk, S. Leonhardt, S. van Waasen, and M. Schiek, "Model-based verification of a non-linear separation scheme for ballistocardiography," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 1, pp. 174–182, 2014.

[85] Y. Yao, S. van Waasen, and M. Schiek, "Automated parameter estimation for a nonlinear signal separation scheme," in *Proceedings of the 15th International Conference on Biomedical Engineering*, J. Goh, Ed., vol. 43. Heidelberg: Springer, 2013, pp. 84–87.

[86] Y. Yao, J. Schiefer, S. van Waasen, and M. Schiek, "A non-parametric model for ballistocardiography," in *2014 IEEE Workshop on Statistical Signal Processing (SSP)*, Gold Coast, QLD, Australia, June 2014, pp. 69–72.

[87] R. F. Schmidt and G. Thews, Eds., *Physiologie des Menschen*, 24th ed. Berlin: Springer, 1990.

[88] K. J. Friston, L. Harrison, and W. Penny, "Dynamic causal modelling," *NeuroImage*, vol. 19, no. 4, pp. 1273–1302, 2003.

[89] K. Rasche, M. Konermann, T. Schäfer, M. Schläfke, A. Sturm, and G. Schultze-Werninghaus, Eds., *Schlafbezogene Atmungsstörungen im Kindes- und Erwachsenenalter*. München: MMV Medizin Verlag, 1994.

[90] U. Rajendra Acharya, K. Paul Joseph, N. Kannathal, C. Lim, and J. Suri, "Heart rate variability: A review," *Medical and Biological Engineering and Computing*, vol. 44, no. 12, pp. 1031–1051, 2006.

[91] O. T. Inan, M. Etemadi, R. M. Wiard, G. T. A. Kovacs, and L. Giovangrandi, "Non-invasive measurement of valsalva-induced hemodynamic changes on a bathroom scale ballistocardiograph," in *Proceedings of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, 2008, pp. 674–677.

[92] J. Paalasmaa, "A respiratory latent variable model for mechanically measured heartbeats," *Physiological Measurement*, vol. 31, no. 10, pp. 1331–1344, 2010.

[93] O. Rössler, "An equation for continuous chaos," *Physics Letters A*, vol. 57, no. 5, pp. 397–398, 1976.

[94] B. Betrò and R. Rotondi, "On Bayesian inference for the inverse Gaussian distribution," *Statistics & Probability Letters*, vol. 11, no. 3, pp. 219–224, 1991.

[95] R. Barbieri, E. C. Matten, A. A. Alabi, and E. N. Brown, "A point-process model of human heartbeat intervals: New definitions of heart rate and heart rate variability," *American Journal of Physiology - Heart and Circulatory Physiology*, vol. 288, no. 1, pp. H424–H435, 2004.

[96] R. Barbieri and E. N. Brown, "Analysis of heartbeat dynamics by point process adaptive filtering," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 1, pp. 4–12, 2006.

[97] P. L. Smith and R. Ratcliff, "Psychology and neurobiology of simple decisions," *Trends in Neurosciences*, vol. 27, no. 3, pp. 161–168, 2004.

[98] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000, Circulation Electronic Pages: http://circ.ahajournals.org/cgi/content/full/101/23/e215 PMID:1085218;.

[99] F. Wang, M. Tanaka, and S. Chonan, "Development of a PVDF piezopolymer sensor for unconstrained in-sleep cardiorespiratory monitoring," *Journal of Intelligent Material Systems and Structures*, vol. 14, no. 3, pp. 185–190, 2003.

[100] J. Paalasmaa and M. Ranta, "Detecting heartbeats in the ballistocardiogram with clustering," in *Proceedings of the Workshop on Machine Learning for Health-Care Applications (ICML/UAI/COLT)*, 2008.

[101] J. M. Kortelainen, M. O. Mendez, A. M. Bianchi, M. Matteucci, and S. Cerutti, "Sleep staging based on signals acquired through bed sensor," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 3, pp. 776–785, 2010.

[102] D. Friedrich, X. L. Aubert, H. Führ, and A. Brauers, "Heart rate estimation on a beat-to-beat basis via ballistocardiography - a hybrid approach," in *Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2010, pp. 4048–4051.

[103] C. Brüser, K. Stadlthanner, S. de Waele, and S. Leonhardt, "Adaptive beat-to-beat heart rate estimation in ballistocardiograms," *IEEE Transactions on Information Technology in Biomedicine*, vol. 15, no. 5, pp. 778–786, 2011.

[104] C. Brüser, S. Winter, and S. Leonhardt, "Robust inter-beat interval estimation in cardiac vibration signals," *Physiological Measurement*, vol. 34, no. 2, pp. 123–138, 2013.

[105] A. Vehkaoja, S. Rajala, P. Kumpulainen, and J. Lekkala, "Correlation approach for the detection of the heartbeat intervals using force sensors placed under the bed posts," *Journal of Medical Engineering and Technology*, vol. 37, no. 5, pp. 327–333, 2013.

[106] J. Paalasmaa, H. Toivonen, and M. Partinen, "Adaptive heartbeat modelling for beat-to-beat heart rate measurement in ballistocardiograms," *IEEE Journal of Biomedical and Health Informatics*, 2014, in press.

[107] R. B. Buxton, *Introduction to Functional Magnetic Resonance Imaging: Principles and Techniques*. Cambridge: Cambridge University Press, 2002.

[108] S. A. Huettel, A. W. Song, and G. MacCarthy, *Functional Magnetic Resonance Imaging*. Sunderland, Mass.: Sinauer, 2004.

[109] I. I. Rabi, J. R. Zacharias, S. Millman, and P. Kusch, "A new method of measuring nuclear magnetic moment," *Physical Review*, vol. 53, p. 318, 1938.

[110] S. Ogawa, T. M. Lee, A. R. Kay, and D. W. Tank, "Brain magnetic resonance imaging with contrast dependent on blood oxygenation,"

Proceedings of the National Academy of Sciences, vol. 87, pp. 9868–9872, 1990.

[111] R. B. Buxton, E. C. Wong, and L. R. Frank, "Dynamics of blood flow and oxygenation changes during brain activation: The balloon model," *Magnetic Resonance in Medicine*, vol. 39, no. 6, pp. 855–864, 1998.

[112] K. Friston, A. Mechelli, R. Turner, and C. Price, "Nonlinear responses in fMRI: The balloon model, Volterra kernels, and other hemodynamics," *NeuroImage*, vol. 12, no. 4, pp. 466–477, 2000.

[113] K. E. Stephan, N. Weiskopf, P. M. Drysdale, P. A. Robinson, and K. J. Friston, "Comparing hemodynamic models with DCM," *NeuroImage*, vol. 38, no. 3, pp. 387–401, 2007.

[114] O. David, S. J. Kiebel, L. M. Harrison, J. Mattout, J. M. Kilner, and K. J. Friston, "Dynamic causal modeling of evoked responses in EEG and MEG," *NeuroImage*, vol. 30, no. 4, pp. 1255–1272, 2006.

[115] S. Shankar Raman, L. Deserno, F. Schlagenhauf, and K. E. Stephan, "A hierarchical model for unifying unsupervised generative embedding and empirical Bayes," *NeuroImage*, 2015, submitted.

[116] W. Penny, K. Friston, J. Ashburner, S. Kiebel, and T. Nichols, Eds., *Statistical Parametric Mapping: The Analysis of Functional Brain Images*. London, UK: Academic Press, 2007.

[117] A. Gelman, *Bayesian Data Analysis*, 3rd ed., ser. Texts in statistical science series. Boca Raton, Fla.: CRC Press, 2014.

[118] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, vol. 39, no. 1, pp. 1–38, 1977.

[119] Y. W. Teh, "Dirichlet processes," in *Encyclopedia of Machine Learning*. Springer, 2010.

[120] K. H. Brodersen, T. M. Schofield, A. P. Leff, C. S. Ong, E. I. Lomakina, J. M. Buhmann, and K. E. Stephan, "Generative embedding for model-based classification of fMRI data," *PLoS Computational Biology*, vol. 7, no. 6, p. e1002079, 06 2011.

[121] C. Rasmussen and H. Nickisch, "GPML Matlab toolbox," http://www.gaussianprocess.org/gpml/code/matlab/doc/, 2006, accessed: 2012-11-05.

Band / Volume 31
Carrier mobility in advanced channel materials using alternative gate dielectrics
E. Durğun Özben (2014), 111 pp
ISBN: 978-3-89336-941-6

Band / Volume 32
Electrical characterization of manganite and titanate heterostructures
A. Herpers (2014), ix, 165 pp
ISBN: 978-3-89336-948-5

Band / Volume 33
Oxygen transport in thin oxide films at high field strength
D. Weber (2014), XII, 115 pp
ISBN: 978-3-89336-950-8

Band / Volume 34
Structure, electronic properties, and interactions of defects in epitaxial GaN layers
P. H. Weidlich (2014), 139 pp
ISBN: 978-3-89336-951-5

Band / Volume 35
Defect Engineering of $SrTiO_3$ thin films for resistive switching applications
S. Wicklein (2014), xi, 144 pp
ISBN: 978-3-89336-963-8

Band / Volume 36
Integration and Characterization of Atomic Layer Deposited TiO_2 Thin Films for Resistive Switching Applications
M. Reiners (2014), xiv, 166 pp
ISBN: 978-3-89336-970-6

Band / Volume 37
Resistive switching in ZrO_2 based metal-oxide-metal structures
I. Kärkkänen (2014), xviii, 125 pp
ISBN: 978-3-89336-971-3

Band / Volume 38
Resistive switching phenomena of extended defects in Nb-doped $SrTiO_3$ under influence of external gradients
C. Rodenbücher (2014), xiii, 200 pp
ISBN: 978-3-89336-980-5

Schriften des Forschungszentrums Jülich
Reihe Information

Band / Volume 39
Micro-spectroscopic investigation of valence change processes in resistive switching $SrTiO_3$ thin films
A. Köhl (2014), viii, 166 pp
ISBN: 978-3-89336-988-1

Band / Volume 40
Strained Silicon and Silicon-Germanium Nanowire Tunnel FETs and Inverters
S. Richter (2014), iii, 117 pp
ISBN: 978-3-95806-002-9

Band / Volume 41
Integration of Redox-Based Resistive Switching Memory Devices
F. Lentz (2014), i, 166 pp
ISBN: 978-3-95806-019-7

Band / Volume 42
Ladungstransportuntersuchungen an nanofunktionalen Bauelementen mit Diodencharakteristik basierend auf funktionalisierten Nanopartikeln
N. Babajani (2015), iv, 138, XLVII
ISBN: 978-3-95806-026-5

Band / Volume 43
Transport and Noise Properties of Nanostructure Transistors for Biosensor Applications
J. Li (2015), vii, 175 pp
ISBN: 978-3-95806-034-0

Band / Volume 44
Quantitative scanning tunneling spectroscopy of non-polar III-V compound semiconductor surfaces
M. Schnedler (2015), 122 pp
ISBN: 978-3-95806-075-3

Band / Volume 45
Model-based Algorithm Development with Focus on Biosignal Processing
Y. Yao (2015), x, 169 pp
ISBN: 978-3-95806-080-7

Weitere ***Schriften des Verlags im Forschungszentrum Jülich*** unter
http://wwwzb1.fz-juelich.de/verlagextern1/index.asp

Information

45

Model-based Algorithm Development

Yu Yao

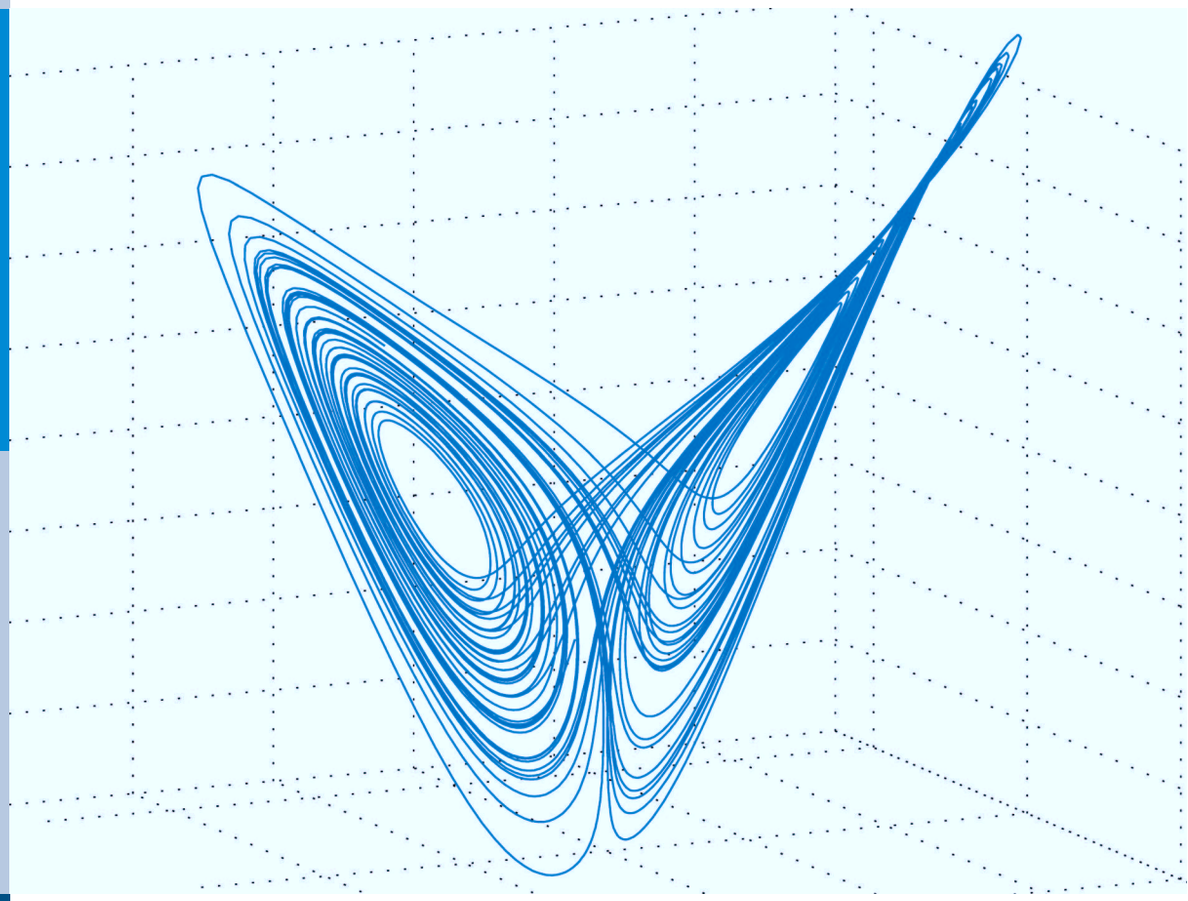
Member of the Helmholtz Association

Information
Band / Volume 45
ISBN 978-3-95806-080-7

JÜLICH
FORSCHUNGSZENTRUM

Model-based Algorithm Development with Focus on Biosignal Processing

Yu Yao



Information
Band / Volume 45
ISBN 978-3-95806-080-7

JÜLICH
FORSCHUNGSZENTRUM