DISSERTATION

Bayesian Inference for Reliable Biomedical Signal Processing

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines
Doktors der technischen Wissenschaften unter der Leitung von

A.o. Univ Prof. Dr. Georg Dorffner
am Institut für medizinische Kybernetik und Artificial Intelligence

eingereicht an der Technischen Universität Wien
technisch naturwissenschaftliche Fakultät

von

Dipl. Ing. Peter Sykacek
Matr. Nr.: 8426669
Franz Josefgasse 11 A-3423 St. Andrä Wördern

Wien, am 23.5.2000

Kurzfassung der Dissertation

Diese Dissertation beschäftigt sich mit der Frage inwieweit Bayes'sche Inferenz geeignet ist, die Zuverläßigkeit von biomedizinischen Diagnosanwendungen zu verbessern. Die Arbeit geht im wesentlichen davon aus, daß Biosignale, die in Form einer oder mehrerer parallel aufgezeichnter Zeitreihen vorliegen, klassifiziert werden sollen. Als Beispielanwendung wird auf das Problem einer Schlafanalyse eingegangen, bei der EEG-Signale klassifiziert werden. Die Arbeit untersucht 4 Teilaspekte einer derartigen Diagnoseanwendung.

Zunaechst wird der Bayes'sche Ansatz verwendet um Modellkoeffizienten eines autoregressiven Modells zu schätzen. Dabei wird auch auf die Schätzung der Modellordnung eingegengen. In einem Experiment wird untersucht, ob sich die Modellwahrscheindlichkeiten als Maß zur Erkennung von Artefakten im EEG eignen.

Im darauf folgenden Kapitel wird das Problem der Merkmalsauswahl vom Bayes'schen Standpunkt aus betrachtet. Es wird ein Algorithmus vorgestellt, der in der Lage ist verschiedene Input-Kombinationen mit Wahrscheindlichkeiten zu versehen, die aussagen, wie viel die unterschiedlichen Input-Kombinationen zur Lösung des Problems beitrgen. Zur Modellschaetzung werden Markov chain Monte Carlo (MCMC) Techniken eingesetzt. Der Algorithmus wird im Zusammenhang mit der erwähnten Schlafanalyse, zur Bewertung verschiedener Vorvererbeitungsmethoden eingesetzt.

In Kapitel 6 wird zur Klassifikation ein generatives Modell eingesetzt. Das Modell wird mittels Bayes'scher Inferenz geschätzt, wobei eine analytische Approximation mittels Mean-Field-Techniken berechnet wird. Der Ansatz wird eingehend untersucht, wobei insbesondere auf die richtig Bestimmung der Modellordnung und die Generalisierungs-Performanz eingegengen wird.

Kapitel 7 erweitert die klassische Vorgangsweise aus Kapitel 6. Der in Kapitel 7 vorgestellte Ansatz strebt eine optimale Integration von Vorverarbeitungsmethoden und Klassifikation an. Diese optimale Integration wird erreicht, indem Vorverarbeitung und Klassifizierung zu *einem gemeinsamen* Wahrscheindlichkeitsmodell zusammengefaßt werden. Der resultierende Algorithmus bewirkt, daß Informationen entsprechend ihrer Zuverlässigkeit zu einer Entscheidung zusammengefaßt werden ("sensor fusion"). Das dabei verwendete Modell hat Ähnlichkeit mit dem bekannten Hidden-Markov-Modell. Die Inferenz erfolgt wiederum mittels MCMC Techniken. Auch diese Methode wird an Hand der Schlafanalyse evaluiert.

Abstract

This thesis investigates whether Bayesian inference can improve the reliability of biomedical diagnosis. In particular we discuss time series classification as is for example needed for an analysis of all-night sleep EEG recordings. Such an attempt needs 4 steps that are further analyzed.

First we must preprocess the raw data. In this thesis we suggest for that step a Bayesian analysis of an autoregressive lattice filter model. We are especially interested in model selection, and, whether the model probabilities are viable means for artifact detection in EEG signals.

The subsequent chapter treats feature subset selection within the Bayesian framework. We infer the probabilities of differet feature subsets, which tell us in how far each of the subsets contributes to the classification. Inference of the classifier is done with Markov chain Monte Carlo (MCMC) techniques. Changing feature subsets requires to sample accross models with different numbers of inputs, which is done with a reversible jump MCMC sampler. This approach is used to assess the importance of different preprocessing techniques as inputs for sleep analysis.

The last topic that is treated in this thesis is the classification stage. In chapter 6 we use variational approximations to derive a Bayesian posterior over model coefficients and different model orders. The classifier is used to build a sleep analyzer, where a lot of emphasis was put on model selection.

The final chapter proposes an improvement of the classical approach that was followed so far. By using *one* probabilistic model that unifies preprocessing and classification, we obtain a classifier that, in the sense of relying more on reliable information, performs optimal sensor fusion. The proposed classifier is similar to the well known hidden Markov model. Inference is performed with MCMC techniques. The proposed architecture is again applied to sleep analysis.

Acknowledgements

The work for this thesis was done at the Austrian Research Institute for Artificial Intelligence (ÖFAI), which is sponsored by the Federal Ministry of Education, Science and Culture. In particular I was employed for the European Biomed project SIESTA, funded by the EC. Dg. XII grant BMH4-CT97-2040.

I want to take the oportunity to thank all people who helped to make this thesis possible. In particular I am grateful to my supervisor, Georg Dorffner from the Departement of Medical Cybernetics and Artificial Intelligence at the University Vienna. He always took care that I move into the right direction, however without preventing me from making my own experiences as well. Georg Dorffner and my external examinor Horst Bischof from the Departement of Pattern Recognition and Image Processing at the Technical University Vienna helped me by proof reading a draft version of this thesis.

Robert Trappl gave me the opportunity to pursue my research at the Austrian Research Institute for Artificial Intelligence.

My colleagues Arthur Flexer, Johannes Fürnkranz, Bernhard Pfaringer, Johann Petrak and Peter Tino were often involved in fruitful discussions.

I am also grateful to Peter Rappelsberger from the Brain Research Institute at the University of Vienna and to Stephen Roberts, Will Penny and Iead Rezek from the Departement of Engineering Science at the University of Oxford for their help and many inspiring discussions.

Last but not least I want to thank Simona for her moral support and for allowing me to spend a lot of our spare time with working for this thesis.

Contents

Chapter 1

Introduction

The background for the ideas of this thesis is biomedical signal analysis. In such domains, especially when the methods are applied to diagnosis or monitoring of humans, wrong decisions are often associated with extremely high cost. In the worst case we might even risk someone's life. On the other hand it is *impossible* that decisions are always correct. Hence there is a large interest in understanding the models that are used in such decision processes. Furthermore we aim at techniques that allow separating cases where decisions can be made with high confidence from those where we expect high risk of misclassification. The methods discussed in this thesis aim at optimal inference in an uncertain scenario. We are interested to apply techniques that are based on a framework that is believed to render optimal behaviour. Unfortunately there are competing "theories" that might guide our analysis and we have to decide for one of them. In this work all methods have been developed under the Bayesian paradigm. We provide a Bayesian analysis of 4 different methods that are usually applied during classifying of large time series. We develop algorithms for preprocessing, feature subset selection[1], static classification and sensor fusion. All techniques are treated within the Bayesian framework. The algorithms are applied to various synthetic and benchmark datasets. However, the major application treated throughout this thesis is an analysis of all night sleep EEG recordings. These sleep data were recorded within the EC funded project SIESTA. The code developed for preprocessing (chapter 4) and for classification (chapter 6), is a major part of the SIESTA analyzer.

The Bayesian choice is motivated by many successful applications of Bayesian inference in the biomedical domain (see e.g. [GJRL98], [SBGI96] and [SDRZ98]). Although Bayesian methods have often been applied to

[1]Feature subset selection is the usual terminology. Correctly speaking we sum the contributions of different subsets, which are weighted by the subset probabilities.

biomedical problems, there is one aspect we would like to investigate further: how can Bayesian methods help to obtain reliable decisions? In order to achieve reliable predictions, we must consider the following issues:

- During inference we must choose model orders (model complexity) appropriately. An appropriatly chosen model complexitly is necessary for high generalization accuracy. Furthermore any information that is explicitly or implicitly based on the model order is only meaningful if the model order is reasonably chosen. An example is reporting which kernel in a mixture density model has most probably generated a particular sample.

- The inference process should be insensitive to suboptimal settings of all parameters that have to be tuned by the user. This allows inference without having to fine tune the algorithms for each problem separately. Such insensitivity is achieved by specifying priors hierarchically. Parameter tuning is often a major problem that causes suboptimal results. The usual approach is to use a separate validation set and cross validation. This has several major dissadvantages: a) In cases where data are rare paremeter inference would profit from additional samples; b) The optimal parameters for the problem of interest are often determined in a trial and error fashion; c) Last but not least we *must* view this parameter tuning as a higher level learning task. As such classifier selection is also subject to learning theoretical bounds [DGL96].

- Using an inferred model for predictions, we face different sources of uncertainty. As is reviewed in [DR00], we must consider model uncertainty, parameter uncertainty and sometimes also input uncertainty. Predictions must incorporate all these uncertainties into the resultig decisions. In the Bayesian paradigm all unobserved variables are considered to be uncertain. Technically this is done by treating unobserved quantities as random variables[2]. The uncertainties about unobserved quantities will be incorporated into predictions by integrating over the corresponding distributions.

One of the reasons for the success of the Bayesian methodology is that it provides means to address all three issues. A point often criticized about the Bayesian methodology is what [BS94] call the *subjectivist* view of probabilities. The solution we get (a posterior over some hypothesis space) depends

[2]Model coefficients are considered to be continuous random variables. Model selection is treated by allowing for a discrete randoim variable as model indicator. Finally if necessary inputs can be regarded as random variables as well.

on the prior that was chosen initially. This criticism however misses the point in more than one perspective:

- Every method uses some prior. The Bayesian methodology, however, is the only one that makes the prior explicit. An example is model selection: whatever method we look at, there is always a "best fit" versus "complexity" trade off. That is, Akaike's information criterion (AIC) (see [Aka74]), minimum description length (MDL) (see [Ris78]), minimum message length (MML) (see [WF87]) or likelihood ratio tests as used in frequentist statistics (see [SS71]) all use a prior that prefers simple models. The same is true for statistical learning theory summarized in [Vap95], where model complexity is constrained by minimizing the VC dimension of the model. Penalized likelihood, as is used e.g. in neural network learning - either explicitly as *weight decay* or implicitly as *early stopping* - can also be understood as priors for "small weights" (e.g. [Bis95]).

- A lot of emphasis can be put on setting up hierarchical priors that avoid solutions that are sensitive to the prior settings. Excellent examples are the work in [RG97], who use a hierarchical prior over component variances in one dimensional Gaussian mixture models and the work in [AFD00], who make *all* priors hierarchical.

- In recent years a lot of research has dealt with the issue of setting up uninformative priors. Rather exciting developments have been made in developing reference priors (see e.g [BB92]). Loosely speaking, the idea is to set up a prior such that the information contained in the posterior distribution about the data is maximized.

- Finally it is quite easy to prove that under some constraints priors are asymptotically efficient [BS94]. This is meant in the sense that the influence of the prior vanishes provided the amount of data is large enough.

However this should not be taken as an argument that setting up priors is an easy task. A lot of emphasis must be put on analyzing whether the results are sensitive to the priors chosen. In particular, we always have to bear in mind that priors, which are uninformative with respect to parameters, will be informative with respect to model orders. Hence being interested in model selection, we must keep balance between being uninformative on the parameter level and being uninformative on the level of alternative models. Such kind of analysis has been carried out by [RG97] and [Ste97].

At the end of this introduction we will briefly summarize the contributions found in subsequent chapters. The next chapter provides an example of time series classification. The application discussed there is sleep analysis as is approached in the SIESTA project. We used sleep analysis as a prototypical problem of biomedical signal analysis to provide a unifying application for the different methods discussed in the subsequent chapters. Chapter 3 gives an overview of Bayesian theory and Bayesian methods to an extent that is necessary to motivate and ground the subsequent analysis.

Chapter 4 will discuss Bayesian preprocessing. We will give an example of time series analysis with autoregressive (AR)-lattice-filters and prove that Bayes optimal classifications[3] can only be obtained by using a probabilistic link between preprocessing and classification. We will also derive the posterior probability of a particular lattice filter stage. This probability is useful for model selection, but we may also interpret it as a reliability measure of each lattice filter stage, when white noise is considered to be an artefact. A Bayesian analysis of lattice filter coefficients has not been published before.

The problem of model selection is further dealt with in chapters 5 and 6. In chapter 5 we propose an approach to input subset selection for classification problems, formulated as Bayesian model selection. The classifier used in chapter 5 is a generative model. Inference is done with Gibbs updates for within dimensional moves and with reversible jump Markov chain Monte Carlo updates for dimension changing moves. Together these updates approximate the posterior over model parameters and feature subsets. At least asymptotically, the technique will approximate the true posterior. The algorithm proposed in chapter 5 has been tested using some datasets that are publicly available. We will also show a result where we explore the posterior distribution over feature subsets using features that were extracted from sleep EEG data. The material in chapter 5 is an extended version of [Syk00]. In chapter 6 we discuss model selection for a similar generative classifier as was used in chapter 5. For reasons of efficiency, the analysis is based on variational inference. The exact posterior over latent variables and model coefficients is approximated by a mean field expansion. Although only approximately correct, we determine probabilities over models by using the lower bound of the true log evidence provided by such techniques. The method is applied both to synthetic problems and the mentioned real world data set, which is concerned with analysis of sleep recordings. It should be mentioned that the algorithms derived in chapter 6 are the core of the sleep analyzer developed for the European project SIESTA. Although similar to

[3]Bayes optimal is meant in terms of minimizing some cost functional, e.g. the misclassification rate.

the variational analysis of Gaussian mixture models reported in [Att99], a generative classifier has so far not been analysed by variational approximations. In chapter 5 we propose a more exact solution. Input subset selection for classification problems is formulated as Bayesian model selection. The classifier used in chapter 5 is again a generative model. Inference is done with Gibbs updates for within dimensional moves and with reversible jump Markov chain Monte Carlo updates for dimension changing moves. Together these updates approximate the posterior over model parameters and feature subsets. At least asymptotically, the technique will approximate the true posterior. The algorithm proposed in chapter 5 has been tested using some datasets that are publicly available. We will also show a result where we explore the posterior distribution over feature subsets using features that were extracted from sleep EEG data. The material in chapter 5 is an extended version of [Syk00].

Chapter 7 follows up on the ideas laid out in chapter 4 for the Bayesian lattice filter. Following the requirement that preprocessing *must be Bayesian* in order to allow *optimal* decisions, we derive a classifier that implements such a strategy. The method allows for spatial and temporal fusion of the *uncertain* information obtained from preprocessing. The model is a classifier where class labels exhibit a first order Markov dependency. The observation model assumes class conditional independence of different inputs. This is what is often referred to as a naïve Bayes model [Rip96]. For each input the class conditional densities are modeled with a mixture of Gaussians. We infer the posterior over latent variables and model coefficients. The key idea to allow for uncertainty of preprocessed features is to treat them as latent variables, as well. The resulting inference strategy improves classification accuracy, when the time series to be classified is contaminated with noise. However, the method can do more: when applied to a contaminated time series, we observe empirically that the expectations in the latent feature space are better estimates of the *true* features[4] as are the estimates obtained from preprocessing alone. An integrated treatment of preprocessing and classification, as is done in section 7, has not been proposed before.

Chapter 8 provides a summary and a final discussion of the material presented in this thesis.

[4]True features refers to estimates obtained from the uncontaminated time series.

Chapter 2

All night sleep analysis

As already mentioned in the introduction, the background for this thesis is biomedical signal analysis. We are interested in an investigation of different techniques that are necessary to perform time series classification reliably. Besides some publicly available benchmark problems and synthetic problems, the major application that is treated thoroughout this thesis is an automated analysis of all night sleep EEG recordings. Such an automated sleep analysis is the aim of the EC funded Biomed project SIESTA[1].

The conventional way of analyzing all night sleep is according to the so-called Rechtschaffen and Kales (R & K) scoring rules that were formulated in [RK68]. The (R & K) rules define 4 sleep stages (stage 1 to 4) as well as states for wake and rapid eye movement (REM) sleep. Sleep stage 1 is light sleep and sleep stage 4 corresponds to deep sleep. Hence there is an ordering among these 4 sleep stages. REM sleep is an important dream related event. An example of a R & K hypnogram is found in figure 2.1. Although many attempts have been made to develop an automatic R & K sleep stager (e.g. [SDRZ98]), so far all these attempts have not lead to satisfying performance. Especially when they are applied to subjects with sleep disturbances, autmated R & K sleep stager will fail. Hence until now sleep analysis according to R & K scoring rules is still performed manually. However, even among human experts the inter rater agreement can be very low.

Rechtschaffen and Kales scoring rules have two major problems:

- The rules allow for subjective interpretations that cause the low inter rater agreement.
- The rules are defined on short events that are found in the biosignals. An example are sleep spindles, K-complexes or REM activity. All these

[1]Project details can e found in the acknowledgements.

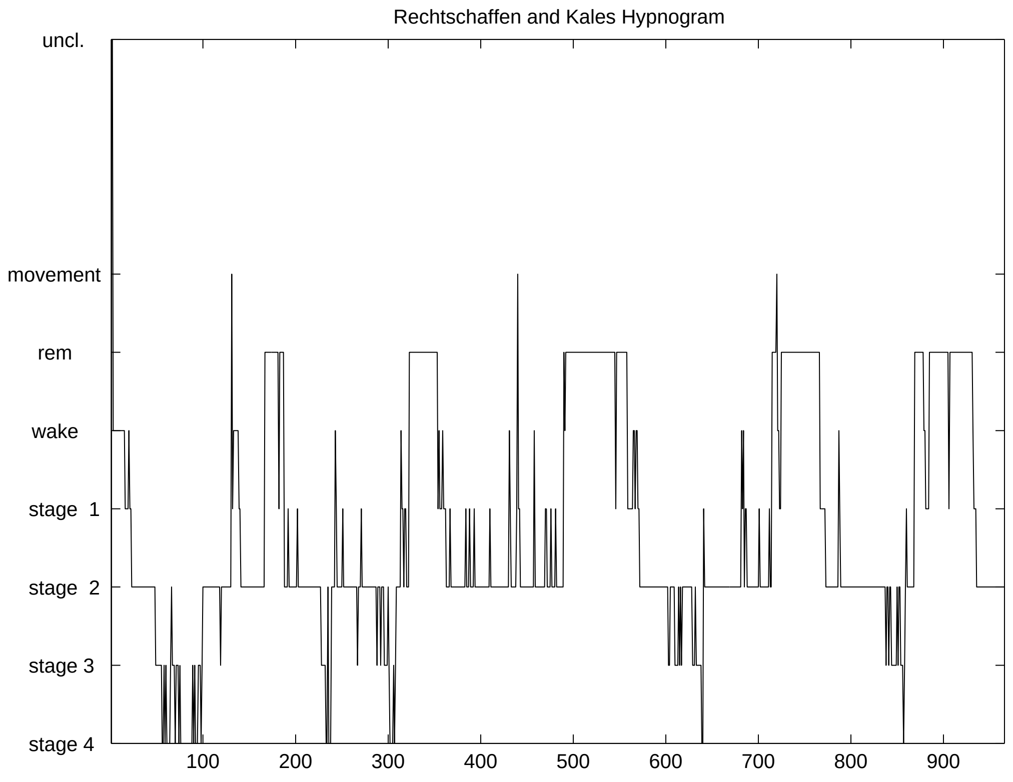

Figure 2.1: This figure shows an example of a Rechtschaffen and Kales hypnogram. The rythmicity of this hypnogram is typical for good and relaxing sleep.

events occur on a 1 second scale, whereas the sleep scorings are based on segments of 20 or 30 seconds duration. Hence the R & K rules rely heavily on smoothing. An example is classification of REM sleep, where the indicators of REM events may be separated by several minutes of stage 1 sleep. Although this stage 1 sleep does not show any REM patterns, it will be added to the REM period. Labels that are smoothed in such a way will be very difficult to model for any automated method. Unfortunately these smoothing rules were also applied when classifying the SIESTA recordings.

- The 20 or 30 seconds resolution is too low to pick up short time events such as micro arousals, that are in the range of one second.
- The ordering among the sleep stages (stage 2 represents sleep that is closer to deep sleep as is stage 1 etc.), makes correct labeling difficult. Hence comparing R & K labels from different human experts, we find

many confusions of “neighbouring” stages.

Both the necessity for smoothing and the low resolution should be eliminated if the analysis is based on 1 second epochs. The latter problem is removed if we give up on predicting all labels and concentrate on wake, deep sleep (often called delta sleep) and REM sleep. However, we are not interested in the labels themselves, but in the corresponding probabilities. That is, the aim of the SIESTA project is to interpret the entire night in terms of probabilities for (R & K) labels wake, REM sleep and delta sleep. According to [PRTJ97] this suffices to describe human sleep. This way of approaching the problem has also the advantage of being automatic and therefore not allowing for any subjective interpretations.

From the point of view of classification, we have a partially supervised problem. The labels for the supervised part are taken from segments that were unanimously classified by three human experts as either wake, REM or stage 4. Each recording was scored by two independent experts and a third, who performed the consensus scoring.

The data that were recorded within the SIESTA project are electroencephalogram (EEG), electromyogram (EMG), electrocardiogram (ECG) and other biosignals. EEG is definitely the main source of information about the state of a sleeping subject. Therefore the methods presented in this thesis were only applied to EEG, which was recorded according to the international 10-20 system at electrode positions Fp1, C3, O1, Fp2, C4 and O2.

Chapter 3

Bayesian inference

This chapter reviews important aspects of Bayesian inference. The first section provides a motivation for using the Bayesian paradigm to solve decision problems in uncertain environments. The five axioms and some important propositions presented there are known material, as e.g. provided by [BS94]. Although known, it is of particular importance to include this fundamental material: It provides us with a motivation for both the representation of beliefs (or uncertainties) by probabilities and the Bayesian inference principle used throughout this thesis. Hence all decisions taken in subsequent sections are justified theoretically.

The second section in this chapter reviews important practical concepts for a Bayesian analysis. These are techniques used for inference of model coefficients, predictions and model selection.

3.1 Bayesian foundations

3.1.1 Decision problems and consistent behaving

As summarized in [BS94], Bayesian inference results from an axiomatic framework that prescribes *one* principled way how individuals should act if they want to make decisions in uncertain environments. Behaving in accordance with these axioms, the decision maker is guaranteed to avoid any logical inconsistencies. Before we enumerate these axioms, we need to specify what we consider to be a decision problem.

Definition 3.1.1 (Decision problem) *We define a decision problem to consist of the following elements ($\mathcal{E}$, $\mathcal{C}$, $\mathcal{A}$, $\leq$), with:*

(i) $\mathcal{E} = \{\Omega, \emptyset, E_i, i \in I\}$ *denoting an algebra of relevant events.*

(ii) $\mathcal{C} = \{C_j, j \in I\}$ *is the set of possible consequences.*

(iii) $\mathcal{A}$ *denotes the set of available actions,* a_j*, where each* a_j *is a functional mapping from partitions of* Ω*, the certain event in* $\mathcal{E}$*, to some consequences* c_j*.*

(iv) $\leq$ *is a preference relation between actions.*

Discussion of definition 3.1.1

- Opting for an action a_j, the decision maker opts for an *uncertain* scenario. The uncertainty comes in, since opting for an action is equivalent to opting for a set of uncertain events that lead to corresponding consequences (c_1, ..., c_n).
- The preference relation among different actions, $\leq$, depends on the *current state of information* of the decision maker. Hence all decisions are subjective. Two different subjects can act differently without violating any consistency requirements. Obviously an individual might change his or her behaviour once additional information is available. This change of preferences will further be denoted with the *conditional preference symbol* $\leq_G$, where G is some additional information.

Apart from the upper definition, we need some additional specifications: We have to be able to formulate preference statements between consequences. The statement $c_1 \leq c_2$ means that we do not prefer consequence c_1 over consequence c_2. These preference relations among consequences *do not depend on the current state of information.* Using both preference statements defined so far, we can formulate an additional preference statement among *uncertain events.* If for all consequences c_1 and c_2, where $c_1 \leq c_2$ we *do not prefer* action a_1 to action a_2 defined as $\{c_1|E, c|\bar{E}\} \leq \{c_2|F, c|\bar{F}\}$, then we do not prefer the uncertain event E to the uncertain event F. Hence we have introduced a new preference statement between uncertain scenarios: $E \leq F$. The preference statement $\leq$ can be used to define other binary relations among consequences, options (or actions) and uncertain events. Such operations will be used below. However, we will not include these definitions here. Instead the interested reader is referred to [BS94].

Based on definition 3.1.1 of a decision problem, we can state five axioms[1] that prescribe a way that allows individuals to take their actions in a

[1]An interesting consequence of this axiomatic framework is that Bayesian's cannot argue that other subjects who do not use this inference procedure act in an inconsistent way. Their actions might be inconsistent with the axioms formulated below. However, there might be a different axiomatic framework that motivates their behaviour.

consistent manner when faced with an uncertain scenario.

Axiom 1 (Comparability of consequences and options) *:*

(i) There exist some consequences c_1, c_2 and a preference statement, $c_1 < c_2$ that allows us to choose among them.

(ii) There exist some comparable options: Given consequences c_1, c_2 and c_3 and uncertain events E and F: $\{c_1|E, c_2|\bar{E}\} \leq \{c_1|F, c_2|\bar{F}\}$ or $\{c_1|E, c_2|\bar{E}\} \geq \{c_1|F, c_2|\bar{F}\}$.

Discussion of axiom 1

The first part in axiom 1 requires that there are some consequences that are strictly preferred to others. Loosely speaking, the decision maker is willing to put some efforts into his actions in order to reach consequence c_2 instead of c_1. This argument will be used in motivating axiom 2. Axiom 1 assures that we indeed have a decision problem: given that all consequences are equal this were not the case. The second part of axiom 1 assures that there are *some* options that can be compared directly. However we do *not* require that *all* pairs of options are directly comparable using the qualitative preference statement $\leq$.

Axiom 2 (Preferences are transitive) *:*

(i) An option can be compared to itself, $a \leq a$.

(ii) Given $a_1 \leq a_2$, $a_2 \leq a_3$ then $a_1 \leq a_3$.

Discussion of axiom 2

The first part in axiom 2 is self evident: it would not make any sense to favor an option to itself or to prohibit a comparison of an option to itself. Obviously this definition leads to $a \sim a$ with $\sim$ denoting equality. The second part of axiom 2 assures that a sequence of preferred options has to be *transitive*. Allowing for *intransitivity* (allowing for $a_1 \geq a_3$ above) would render strange behaviour of the decision maker: As noted in the discussion of axiom 1, he is willing to *pay a price* to avoid the less favored situation. Hence intransitivity would permit to put efforts into moving in circles and finally reaching the state we started off. Note that axiom 2 is also valid for uncertain events and that it can be extended to other binary preference relations.

Axiom 3 (Preferences are consistent) *:*

(i) If $c_1 \leq c_2$ then given any $G > \emptyset$ we have $c_1 \leq_G c_2$.

(ii) If, for some $c_1 < c_2$, $\{c_2|E, c_1|\bar{E}\} \leq \{c_2|F, c_1|\bar{F}\}$ then $E \leq F$.

(iii) If for some c and $G > \emptyset$, $\{a_1|G, c|\bar{G}\} \leq \{a_2|G, c|\bar{G}\}$, then $a_1 \leq_G a_2$.

Discussion of axiom 3

Axiom 3, part (i) states that preferences between consequences must not depend on the current state of information. The second part states that if there are *some* $c_1 < c_2$ that lead to the preference relation $\{c_2|E, c_1|\bar{E}\} \leq \{c_2|F, c_1|\bar{F}\}$, then this preference must be assigned to *any* $c_1 < c_2$. The third part of axiom 3 requires that if the preference $\{a_1|G, c|\bar{G}\} \leq \{a_2|G, c|\bar{G}\}$ is valid for some consequence c it must be valid for *any* option a. At this point it seems advisable to add two further definitions that will be used below.

Definition 3.1.2 (Significant event) *Given $G > \emptyset$, an event E is considered significant, if $c_1 <_G c_2$ implies $c_1 <_G c_2|E, c_1|\bar{E} <_G c_2$.*

We can understand significant events as events that are plausible but not certain given information G. If c_2 is preferred to c_1, the significant event E increases the probability to arrive at the favored event c_2. Using the previous axioms we can show that for E to be significant the constraint $\emptyset < G \cap E < G$ has to be met.

Definition 3.1.3 (Pairwise independence of events) *Two events E and F are said to be independent, denoted by $E \perp F$, if $\forall c, c_1, c_2$ and any of the binary relations $<$, $>$ or $\sim$, further denoted as $*$, we find:*

(i) $c * \{c_2|E, c_1|\bar{E}\} \Rightarrow c *_F \{c_2|E, c_1|\bar{E}\}$
and

(ii) $c * \{c_2|F, c_1|\bar{F}\} \Rightarrow c *_E \{c_2|F, c_1|\bar{F}\}$.

Two events are said to be independent, if the occurrence of one does not change preference relations obtained by the occurrence of the other.

We will now formulate two quantitative axioms that make the axiomatic framework complete. These axioms allow us to give precise statements about uncertainties of events and enable us to compare *all* different pairs of options - which was not assumed in the definition of the qualitative preference relations. In order to prepare this quantification, we should remember that definition 3.1.1 casts the uncertainty attached to our decision problem in the existence of an algebra $\mathcal{E}$ of *relevant events.*

Axiom 4 (Existence of standard events) *There exists a sub algebra $\mathcal{S}$ of $\mathcal{E}$ and a function $\mu : \mathcal{S} \mapsto [0, 1]$ such that:*

(i) $S_1 \leq S_2 \Leftrightarrow \mu(S_1) \leq \mu(S_2)$.

(ii) $S_1 \cap S_2 = \emptyset \Leftrightarrow \mu(S_1 \cup S_2) = \mu(S_1) + \mu(S_2)$.

(iii) for any $\alpha \in [0, 1]$ there is a std. event S such that $\mu(S) = \alpha$, $S \perp E$ and $S \perp F$, where E and F are any other events.

(iv) $S_1 \perp S_2 \Leftrightarrow \mu(S_1 \cap S_2) = \mu(S_1)\mu(S_2)$.

(v) if $E \perp S$, $F \perp S$ and $E \perp F$ then $E \sim S \Rightarrow E \sim_F S$.

Discussion of axiom 4

Axiom 4 defines the existence of *standard events* to be part of the algebra of relevant events $\mathcal{E}$. Furthermore it defines a measure that allows to quantify the qualitative preference relations among them. In Bayesian decision theory, these *standard events* play essentially the same role as do measurements in physical sciences. Part (i) in axiom 4 establishes *compatibility* of qualitative preference relations among standard events and the relation between the numerical values obtained with function $\mu()$. Given two *standard events* S_1 and S_2 that cannot occur together, part (ii) of the axiom asserts that the mapping of the event of observing *at least one of the standard events* is equivalent to the sum of the mappings of the events. Part (iii) asserts that we can generate *independent standard events* with any $\alpha \in [0, 1]$. Having two *independent standard events* S_1 and S_2, part (iv) says that the mapping of the event that *both occur together* is equivalent to the product of the mappings of the two events. The consequence of part (v) of axiom 4 is that given pairwise independence of three events S, F and E an order relation between S and E is not affected by the occurrence of the independent event F. Finally axiom 5 requests that using *standard events*, we can measure both preferences and uncertainties with arbitrary precision.

Axiom 5 (Precise measurement of preference and uncertainty) *:*

(i) if $c_1 \leq c \leq c_2$ there exists a standard event S such that $c \sim \{c_1|S, c_2|\bar{S}\}$.

(ii) For any event E there exists a standard event *S such that $E \sim S$.*

Discussion of axiom 5

Part (i) in axiom 5 compares a consequence with an option. We can always do so as the consequence can be interpreted as an option conditioned on the certain event Ω. The main idea behind axiom 5 is that we can define two numbers $x \in [0,1]$ and $y = 1 - x$ and re map $S = x \mapsto \mathcal{S}$ as well as $\bar{S} = y \mapsto \mathcal{S}$. Increasing x will decrease the preference for the right side and we will finally reach the equivalence relation between c and the option. The second part of axiom 5 guarantees that any event E can be measured by a *standard event*.

3.1.2 Probability as measure of belief and belief updates

Equipped with the five axioms reviewed in the last subsection, we are able to formulate two propositions important for any analysis in uncertain environments. The first proposition states that a measure of uncertainty that is consistent with the five axioms *must be a probability measure*. The second proposition states that there is *only one* way for revising beliefs in the light of new information that is consistent with the axiomatic framework provided above: We *must* revise our beliefs using *Bayes' theorem*. We will only provide the theorems, for the proofs we refer to [BS94].

Definition 3.1.4 (Measure of degree of belief) *Given an uncertainty relation $\leq$, the probability $P(E)$ is the measure $\mu(S)$ associated with any* standard event *S such that $S \sim E$.*

Remarks to definition 3.1.4

In definition 3.1.4 we define the probability of an arbitrary event E by comparing to a standard event S. This comparison uses the qualitative preference relation $\leq$. As mentioned in the discussion of definition 3.1.1, these preference relations depend on the state of information of the decision maker. Hence all probabilities are personal or subjective and different decision makers can assert different probabilities to the same event. Note that [BS94] justify this definition by proving the proposition 3.1.1

Proposition 3.1.1 (Probability structure of degrees of belief) *:*

(i) $P(\emptyset) = 0$ *and* $P(\Omega) = 1$.

(ii) if $E \cap F = \emptyset$ *then* $P(E \cup F) = P(E) + P(F)$.

(iii) E is significant if, and only if $0 < P(E) < 1$.

Proof. The proof is found in [BS94], page 36. Proposition 3.1.1 tells us that under the axiomatic framework summarized above, *coherent degrees of belief must take the form of a probability measure.* Thus whenever reporting about uncertain states, we are forced to report probabilities. Another consequence of the five axioms concerns *revision of beliefs*, which must be done using Bayes theorem.

Proposition 3.1.2 (Revision of belief by Bayes theorem) *:*
For any $G > \emptyset$ *we get:*

$$P(E|G) = \frac{P(E \cap G)}{P(G)}$$

Proof. For a proof, we refer to [BS94], page 39. In [BS94] there is also a more complete formulation of Bayes theorem, valid for any partitions of the certain event Ω.

3.1.3 A decision criterion

The final proposition we will quote here will provide us a quantitative measure for comparing different actions. Again we consider the decision problem $(\mathcal{E}, \mathcal{C}, \mathcal{A}, \leq)$. For simplicity we assume here that the set of consequences $\mathcal{C}$ is bounded, that is we have two extreme consequences $c_* < c^*$ with $c_* \leq c \leq c^* \forall c \in \mathcal{C}$. A generalization that allows for unbounded consequences can be found in [BS94].

Definition 3.1.5 (Utility function for consequences) *Given a preference relation* $\leq$*, the utility of a consequence* c*,* $u(c) = u(c|c_*, c^*)$*, relative to the extreme consequences* c_* *and* c^* *is the real number* $\mu(S)$ *associated with the standard event* S *such that* $c \sim \{c_*|S, c^*|\bar{S}\}$ *with* $\bar{S}$ *denoting* not S. *The mapping* $u : \mathcal{C} \mapsto \Re$ *is called the utility function.*

A proposition about the existence and uniqueness of bounded utilities is formulated in [BS94]. The utility function $u(c)$ can now be used to attach an overall numerical value to an option a.

Definition 3.1.6 (Conditional expected utility) *Given a decision problem with bounded consequences, some relevant event* $G > \emptyset$ *and the option* $a \equiv \{c_j|E_j, j \in J\}$,

$$\bar{u}(a|c_*, c^*, G) = \sum_{j \in J} u(c_j|c_*, c^*) P(E_j|G),$$

defines the expected utility of option a given G with respect to extreme consequences c_* and c^*.

The utility of an option a is a random quantity, and $\bar{u}$ is just its expectation when conditioning on G.

Finally we will state a proposition regarding decision criteria for bounded decision problems.

Proposition 3.1.3 (Decision criterion for bounded consequences) *For any decision problem with bounded consequences $c_* < c^*$ and $G > \emptyset$,*

$$a_1 \leq_G a_2 \Leftrightarrow \bar{u}(a_1|c_*, c^*, G) \leq \bar{u}(a_2|c_*, c^*, G).$$

Proof. For a proof, we refer to [BS94], page 52. Expressed in words, proposition 3.1.3 tells us that any option we decide for has to have the largest expected utility among all possible options.

3.1.4 Summary

As already stated at the beginning of this section, this is only a fraction of the material found in [BS94]. The importance of this section lies in the fact that it *justifies* a method for taking actions consistently when faced with uncertain scenarios. Whenever we report about uncertain discrete or continuous states, proposition 3.1.1 tells us that we *must* use probabilities or probability densities. Furthermore in proposition 3.1.2 we see that once accepting the axioms *Bayesian inference* is the *only* way to update initial beliefs in the light of new information. The framework presented here assumed a finite number of possible events. However an extension to infinite space of events is possible. Details are again provided by [BS94].

Another important consequence of this section is that all probabilities are subjective. Different decision makers may have different initial states of information. Hence they will come to different conclusions.

3.2 Bayesian methods

The last section showed that probabilities as belief measures and Bayesian inference are no *ad hoc* definitions. Both are provable consequences from five axioms. Hence using a probabilistic framework and Bayesian inference are justified theoretically once accepting these axioms. However this motivation for using Bayesian methods is not constructive in the sense that we do not have any tools for inference yet. This section provides an overview of inference techniques used in later chapters.

3.2.1 Priors and Likelihoods

Applying Bayesian techniques requires a stochastic model where we know how to formulate some prior and a normalized likelihood function. We will assume here that our model has more than one parameter, some we are interested in and others, called nuisance- or hyper-parameters, we are not. Bayesian analysis is carried out by applying proposition 3.1.2 (although in a generalized version) to the problem of interest. Assuming that we want to infer the posterior distribution over model coefficients $\underline{\varphi}$, we get:

$$p(\underline{\varphi}|\mathcal{D}) \propto p(\mathcal{D}|\underline{\varphi})p(\underline{\varphi}), \tag{3.1}$$

with $p(\mathcal{D}|\underline{\varphi})$ denoting the normalized likelihood function and $p(\underline{\varphi})$ representing prior beliefs. In order to get a proper density, (3.1) still needs to be normalized.

In this thesis we will deal with three different kinds of problems, where such normalized likelihood functions exist:

- Regression problems with the likelihood: $p(\mathcal{D}|\underline{\varphi}) = \frac{1}{Z_{\mathcal{D}}} \prod_n p(y_n|\underline{\varphi}, \underline{x}_n)$, and normalization constant $Z_{\mathcal{D}} = \prod_n \left(\frac{2\pi}{\beta}\right)^{0.5}$, where $\underline{\varphi}$ are the model coefficients, y_n are the regression values ("targets") and $\underline{x}_n$ are the regressors. The normalization constant emerges from assuming a Gaussian noise model, where β denotes the precision (inverse variance) of the Gaussian.

- Dichotomous or polychotomous classification problems with the likelihood $p(\mathcal{D}|\underline{\varphi}) = \prod_n P(t_n|\underline{\varphi}, \underline{x}_n)$, where t_n are the class labels and $\underline{x}_n$ are the model inputs. Note that the likelihood has normalization constant 1.

- Density estimation with the likelihood $p(\mathcal{D}|\underline{\varphi}) = \prod_n p(\underline{x}_n|\underline{\varphi})$, where $\underline{x}_n$ are some observations of the distribution of interest. As the model predicts density estimates from a normalized probability density function, the likelihood has again proper normalization.

The second requirement to allow for a Bayesian analysis is that we must be able to formulate our prior assumptions in terms of prior distributions over model coefficients. Often we have some idea how to do this. However in other cases it is difficult and we might want to use an "uninformative prior". At this point it is important to point out that a prior cannot be wrong. In a Bayesian understanding a prior reflects the initial knowledge of the subject that is about to apply Bayesian inference to a particular problem.

This initial information can differ from subject to subject and justifies every prior. However in practice a little more care is necessary - especially when we want to formulate ignorance. One way of formulating an uninformative prior is due to [Jef61] who proposed several so called Jeffreys' priors. The Jeffreys' prior has a nice interpretation in terms of the information geometry of the model class. In [Bal97] the author shows that "ignorance" in the space of realized functions leads to Jeffreys' priors over model parameters. In order to arrive at a prior that reflects prior ignorance, [BS94] suggest to carry out a reference analysis, which has been successfully applied to several problems.

In general the priors will have an effect on the result and it is always advisable to check this effect by testing for sensitivity of the results to small changes of the prior settings. The concept of such a sensitivity analysis is again treated in [BS94]. The real problem with priors is that the less informative over parameters they are, the more strongly they force small model complexities. This suggests that we should also test for the sensitivity of the selected model complexity on the prior. Such kind of sensitivity analysis is discussed in [RG97] in the context of model selection in Gaussian mixture density models.

Another approach in the specification of priors is the so called conjugate analysis. According to [BS94], we can specify *conjugate priors* for all distributions from the exponential family. A pragmatic point of view of conjugate priors is that if we give a model coefficient a conjugate prior, multiplication with the likelihood will not change the functional form of the resulting distribution. Often this is restricted to the so called *full conditional*, which is the posterior of a particular model coefficient when conditioning on all other parameters.

3.2.2 Directed acyclic graphs, conditioning and marginalization

After having summarized the basic methodology of a Bayesian analysis, we will now give a short enumeration of various methods used to represent posterior distributions or probabilities depending on whether we infer about continuous or discrete quantities. A tool that is often useful for understanding complex stochastic models and for deriving inference algorithms is the directed acyclic graph (DAG).

The illustration provided in figure 3.1 shows such a DAG for a hypothetic model with 3 model coefficients (φ_1, φ_2 and φ_3), latent variables t_n, we assume to be discrete, and the corresponding observations x_n. The DAG illustrates the relations for 2 successive observations with all model coeffi-

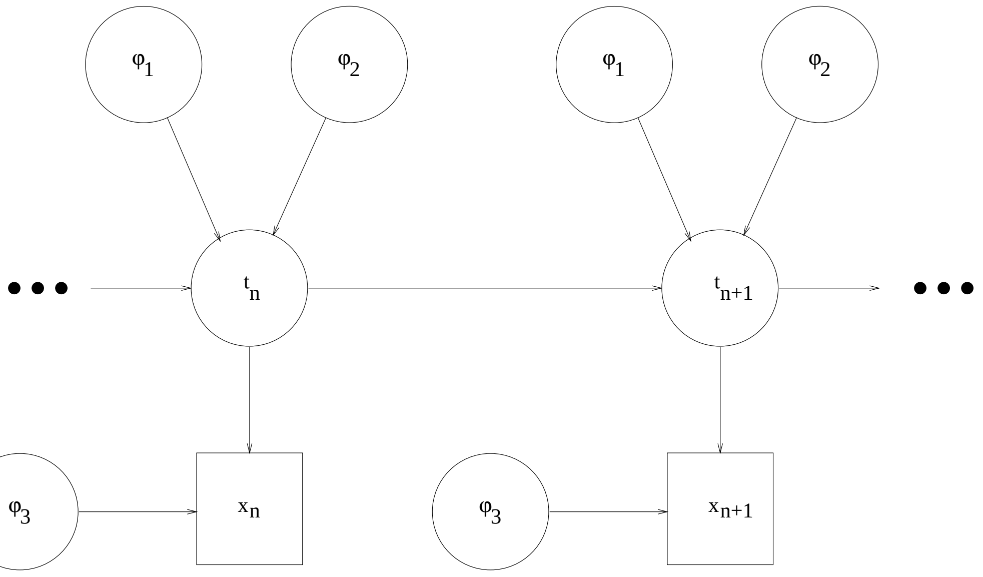

Figure 3.1: A sample directed acyclic graph. We used t_n and t_{n+1} to denote two latent variables, φ_1 to φ_3 denote the model coefficients. Finally we use x_n to denote the observations. The model is hypothetical, however, closely related to hidden Markov models.

cients being replicated. We say that node t_n is conditionally dependent on t_{n-1}, φ_1 and φ_2. Assuming we know the value of t_n we can formulate e.g. the distribution over φ_1. When observing[2] all nodes in the graph, the distribution of each variable depends on the values of all nodes in its Markov blanket. The Markov blanket are all parents, all children and the parents of the children. Thus in our case, when we observe all nodes in the DAG, the distribution $p(\varphi_1|t_n, \varphi_2)$ is conditionally dependent on the discrete latent variable t_n and the other model parameter φ_2. The DAG in figure 3.1 encodes that φ_1 and φ_2 are marginally independent. However conditioning on t_n introduces a dependeny among them. (that is why the Markov blanket contains the parents of the children). Another assumption implied by this DAG is that x_n and t_{n+1} are, conditional on t_n, independent. Note that similar relations are found in hidden Markov models.

Assuming N samples in the training data the joint probability density

[2]Allthough circles are unobserved, we may assume that their values are known. This "trick" is the basis of the Gibbs sampler, a sampling thechnique described below which is frequently used to approximate the joint density in such DAG's.

function (pdf) implied by the DAG in figure 3.1 is:

$$\begin{aligned} p(\varphi_1, \varphi_2, \varphi_3, t_1, ..., t_{n+1}, x_1, ..., x_{n+1}) &= p(\varphi_1)p(\varphi_2)p(\varphi_3) \\ &\prod_{n=1}^{N} p(t_n|t_{n-1}, \varphi_1, \varphi_2)p(x_n|t_n, \varphi_3). \end{aligned} \tag{3.2}$$

Our interest is not the joint probability density but the posterior, $p(\varphi_1, \varphi_2, \varphi_3|x_1, ..., x_N)$. We obtain $p(\varphi_1, \varphi_2, \varphi_3, t_1, ..., t_N|x_1, ..., x_N)$ from (3.2) by conditioning on observed data $(x_1, ..., x_N)$. Conditioning is *one* way to get rid of unwanted quantities. However we may only condition on *known* quantities such as observations. The latent variables $(t_1, ..., t_N)$ that occur in (3.2) have to be removed as well. As they are not observed, we must do this by integrating them out. In the Bayesian literature this operation is referred to as marginalization. As pointed out in [Siv96], the distributions over some quantity obtained by conditioning will always have less variance compared to the same distribution obtained by marginalization.

3.2.3 Different levels of inference

Bayesian inference is more than just finding posterior distributions over model coefficients. It allows treating other parameters that occur in stochastic models like noise levels or regularization constants. Furthermore we can apply the Bayesian framework to allow for model selection. This subsection is meant as an illustration of how these problems are approached theoretically.

We assume a regression model with coefficients $\underline{\varphi}$ that predicts targets y_n. The noise is assumed Gaussian with precision (inverse variance) β. The prior over model coefficients is also assumed to be Gaussian with precision α. We call α and β hyper-parameters or nuisance parameters. The posterior over model coefficients and hyper-parameters is:

$$p(\underline{\varphi}, \alpha, \beta|\mathcal{D}) = \frac{p(\mathcal{D}|\underline{\varphi}, \beta)p(\underline{\varphi}|\alpha)p(\alpha)p(\beta)}{p(\mathcal{D})} \tag{3.3}$$

We obtain $p(\underline{\varphi}|\mathcal{D})$ by integrating (3.3) over α and β. However in many cases these integrals cannot be solved analytically.

The normalization constant is obtained by integrating (3.3) over $\underline{\varphi}$ as well. Usually the normalization constant $p(\mathcal{D})$ is referred to as *model evidence*. Its importance emerges if we make the model, I, explicit. The normalization constant, $p(\mathcal{D}|I)$ is the likelihood that model I generated the data set $\mathcal{D}$. We can use this to determine the posterior probability of model I:

$$P(I|\mathcal{D}) = \frac{p(\mathcal{D}|I)P(I)}{p(\mathcal{D})}. \tag{3.4}$$

Note that $P(\mathcal{D})$ in (3.4) is the normalization constant $\sum_I P(\mathcal{D}|I)P(I)$, which differs from $P(\mathcal{D})$ in (3.3). Prior ignorance about model I is easily reflected by assigning equal prior probability to each model. We have to be aware that the prior over model coefficients has an influence on $p(I|\mathcal{D})$: the higher our prior ignorance over model coefficients, the larger the probability of less complex models.

Once we have found the posterior distributions over model coefficients and the posterior probabilities over models, we are ready for predicting. Predictions are obtained by evaluating Stieltjes integrals over the posterior probability density $p(\underline{\varphi}|\mathcal{D})$:

$$p(y_n|\mathcal{D}) = \int_{\underline{\varphi}=-\infty}^{\infty} p(y_n|\underline{\varphi})p(\underline{\varphi}|\mathcal{D})d\underline{\varphi}. \tag{3.5}$$

3.2.4 Representing posterior densities

We will now give a brief overview over those techniques used in later chapters to obtain representations of posterior distributions. We saw in the last subsection, e.g. in equation (3.5), that many apects of Bayesian inference require to solve integrals over complex distributions. The resulting integrals are hardly analytically feasible and we have to resort to approximations. There are two entirely different approaches that are commonly used:

- We can approximate the posterior by some parametric distribution(s). In subsequent chapters we will apply two of the methods: the Laplace approximation, where a multi-variate Gaussian is fit into one mode of the posterior and a technique called ensemble learning, or variational approximation. In ensemble learning, a mean field approximation is fit into one mode of the posterior using free form optimization.

- The other way of representing posterior densities is by drawing random samples according to their distribution. As one can imagine, the posterior densities will have rather complex structure with most of the probability mass found in small regions of the parameter space. Therefore only such sampling techniques are well suited that can deal with this kind of problems. Among the methods that are usually applied, Markov chain Monte Carlo (MCMC) methods are the most prominent ones.

Laplace approximation

The Laplace approximation is reviewed in [BS94]. Recently, [RHRP98] found the Laplace approximation useful for an inference of Gaussian mixture models. In principle, one mode of the posterior is found by numerical optimization. As is easily seen in (3.7), the covariance matrix of the Gaussian is given by the inverse Hessian of the regularized error function in that mode. In order to obtain this Gaussian, we approximate the penalized error function $E(\underline{\varphi})$ by a truncated Taylor series expansion around the mode[3], $\hat{\underline{\varphi}}$:

$$E(\underline{\varphi}) \quad = \quad (E(\underline{\varphi}) + (\underline{\varphi} - \hat{\underline{\varphi}})^T \underline{\Delta\varphi\Delta\varphi^T} E(\underline{\varphi})(\underline{\varphi} - \hat{\underline{\varphi}}))|_{\underline{\varphi}=\hat{\underline{\varphi}}}. \tag{3.6}$$

Exponentiation of the second order Taylor series expansion of $-E(\underline{\varphi})$[4] and setting $\underline{H} = 2\underline{\Delta\varphi\Delta\varphi^T} E(\underline{\varphi})|_{\underline{\varphi}=\hat{\underline{\varphi}}}$, we get the corresponding Laplace approximation:

$$p(\underline{\varphi}|\mathcal{D}) = \left(\frac{1}{2\pi}\right)^{d/2} |\underline{H}| \exp(-0.5(\underline{\varphi} - \hat{\underline{\varphi}})^T \underline{H}(\underline{\varphi} - \hat{\underline{\varphi}})). \tag{3.7}$$

As a final remark, we want to mention that the normalization constant of the Gaussian distribution also provides an estimate for the model evidence in (3.3).

Ensemble learning

Another method for approximating posterior distributions that will be used in a subsequent chapter of this thesis is a technique called *ensemble learning* or *variational approximation* of the posterior. As was introduced by [HvC93], this technique approximates the posterior over $\underline{\varphi}$ by a parameterized *ensemble* $Q(\underline{\varphi}; \underline{\theta})$. The optimal approximating *ensemble* of the posterior is found by minimizing the *variational free energy* which is well established technique in statistical physics (see e.g. [Fey72]):

$$F(\underline{\theta}) = -\int d\underline{\varphi} Q(\underline{\varphi}; \underline{\theta}) \log(\frac{p(\mathcal{D}|\underline{\varphi}, I)p(\underline{\varphi}|I)}{Q(\underline{\varphi}; \underline{\theta})}). \tag{3.8}$$

In terms of Bayesian statistics, minimizing the *variational free energy* (3.8) corresponds to maximizing a lower bound of the log evidence of model class I. *Variational approximations* are popular tools, especially for approximating distributions in DAG's (see [Jor99] and [JJ00]) and for Bayesian inference of various models (e.g. [Mac97] and [GB00]). Compared to Laplace approximations, *ensemble learning* has the advantage that the approximating

[3]The mode, $\hat{\underline{\varphi}}$, is found by nonlinear optimization techniques like quasi Newton methods applied to the negative log of the regularized likelihood function.

[4]The error function corresponds to the *negative* log likelihood plus a regularization term.

distribution is not restricted to multivariate Gaussians. Finding the best parametric form of the *ensemble* is part of the optimization. On the other hand, there is also the problem that (3.8) gives only a lower bound of the model evidence. Therefore, all evidence based activities such as selecting the most appropriate model class must be done with extreme care.

Markov chain Monte Carlo methods

Markov chain Monte Carlo (MCMC) methods are a special kinds of sampling techniques. In sampling, one approximates the posterior by samples drawn from it. The special property of MCMC methods is that a new sample is drawn conditional on the current sample. The power of MCMC methods is that they are useful for sampling from very narrow distributions. Hence they are widely used in statistics (see [GRe96]). We will review three different approaches here:

- the Metropolis Hastings algorithm,
- Gibbs sampling and
- the reversible jump MCMC sampler.

Metropolis Hastings updates

A common principle of all three algorithms is that detailed balance between any two successive samples of the Markov chain is a sufficient condition for leaving the limiting distribution invariant. The detailed balance condition is expressed as:

$$p(\underline{\varphi}_{t+1})p(\underline{\varphi}_t|\underline{\varphi}_{t+1}) = p(\underline{\varphi}_t)p(\underline{\varphi}_{t+1}|\underline{\varphi}_t). \tag{3.9}$$

Expressed in words, the probability of being in state $\underline{\varphi}_t$ times the probability of moving into state $\underline{\varphi}_{t+1}$ must be equal to the probability beeing in state $\underline{\varphi}_{t+1}$ and moving into state $\underline{\varphi}_t$. The Metropolis Hastings algorithm has been proposed by [MRR+53]. In principle the proposal distribution $p(\underline{\varphi}_{t+1}|\underline{\varphi}_t)$ is an arbitrary random variate that *may* depend on the current state of the Markov chain, $\underline{\varphi}_t$. After a move has been proposed, we calculate the acceptance ratio:

$$\alpha = \min\left(1, \frac{p(\underline{\varphi}_{t+1})p(\underline{\varphi}_t|\underline{\varphi}_{t+1})}{p(\underline{\varphi}_t)p(\underline{\varphi}_{t+1}|\underline{\varphi}_t)}\right). \tag{3.10}$$

It is easy to show that this acceptance ratio indeed guarantees detailed balance (see e.g. [GRe96]). If the proposal distribution is independent of the current state[5], $\underline{\varphi}_t$, we get as a special case the Metropolis algorithm ([Has70]). Another generalization is known as *single component* Metropolis Hastings algorithm: It can be shown (e.g. [GRe96]) that a split of the parameter vector and component wise updating does not violate detailed balance.

Gibbs sampler

The Gibbs sampler can be seen as a special case of a single component Metropolis Hastings algorithm. Originating from the *heat bath* algorithm known in statistical physics, the Gibbs sampler was proposed to the statistics community by [GG84]. The key idea is to propose a new state for the n-th component of the parameter vector $\underline{\varphi}$ from the so called *full conditional*, $p(\varphi^n|\varphi^1_{t+1}, ..., \varphi^{n-1}_{t+1}, \varphi^{n+1}_{t+1}, ..., \varphi^N_{t+1})$, where N denotes the number of coefficients. Using this as a proposal, it is easy to show (e.g. [GRe96]) that the acceptance rate (3.10) is always 1. The Gibbs sampler is therefore formulated as shown in rogram 3.1. As soon as we have drawn a new parameter value, it is used in the full conditional to replace the old one.

Program 3.1 The Gibbs sampler.

```
initialize($\underline{\varphi}$)
REPEAT
    $\varphi^1_{t+1} \sim p(\varphi^1|\varphi^2_t, ..., \varphi^N_t)$;
    ...
    $\varphi^n_{t+1} \sim p(\varphi^n_{t+1}|\varphi^1_{t+1}, ...\varphi^{n-1}_{t+1}, \varphi^{n+1}_t, ..., \varphi^N_t)$;
    ...
    $\varphi^N_{t+1} \sim p(\varphi^N|\varphi^1_{t+1}, ..., \varphi^{N-1}_{t+1})$;
UNTIL (convergence)
```

Reversible jump MCMC

The last technique we want to mention here was recently proposed in [Gre95]. *Reversible jump MCMC* generalizes the Metropolis Hastings algorithm in a way that permits the formulation of dimension changing moves. This allows sampling across model classes. Implementing a hybrid sampler, we get samples from each within dimension posterior *and* estimates of the posterior probabilities of each model class I. These probabilities, $P(I|\mathcal{D})$, are not provided by the other techniques described above.

[5]E.g. using a Gaussian centered in the current state.

The key idea behind *reversible jump MCMC* is to formulate the sampling algorithm in the combined parameter space $\mathcal{C} = \bigcup_k \mathcal{C}_k$. The formulation of transition kernels and detailed balance involves non-trivial measures that live in $\mathcal{C}$ (see [Gre95] for details). Metropolis Hastings-like proposals have to carry out dimension switching moves. Usually one restricts the updates to moves between neighboring spaces. That is from $\mathcal{C}_k$ we can only move to spaces $\mathcal{C}_{k+1}$ and $\mathcal{C}_{k-1}$. Technically, each move is a pair of dimension switching steps which form a bijection: $(\underline{\varphi}_k, u_k) \leftrightarrow (\underline{\varphi}'_k, u'_k)$, where $\underline{\varphi}_k$ and $\underline{\varphi}'_k$ denote the parameter values. The randomly chosen quantities u_k and u'_k are chosen such that the dimensions in this bijection match. A sufficient condition for the *reversible jump MCMC* to leave the distribution over the joint space $\mathcal{C}$ invariant is detailed balance within each type of move. Detailed balance is guaranteed when the acceptance probability of the move from space $\mathcal{C}_k$ to space $\mathcal{C}_{k'}$ takes the following form:

$$\alpha = \min\left(1, \text{lh. ratio} \times \text{prior ratio} \times \frac{p(m, u'_k|\underline{\varphi}'_k)}{p(m, u_k|\underline{\varphi}_k)} \times \left|\frac{\partial(\underline{\varphi}'_k, u'_k)}{\partial(\underline{\varphi}_k, u_k)}\right|\right) \tag{3.11}$$

The third term is the proposal ratio which is the joint probability of proposing move m and the parameter values, u_k and u'_k, that are used in constructing the new model coefficients. The last is the Jacobian arising from a change of variables.

3.2.5 Summary

This section on Bayesian methods was meant to provide an overview of the inference aim in a Bayesian analysis and to review some techniques how these aims can be obtained technically. We saw that Bayesian inference allows more than just parameter inference and predictions. Elaborated models will take some nuisance parameters like noise levels or regularization constants into account. Serious applications of Bayesian inference will also attempt model selection or even better model veraging. In the following chapters of we will apply these inference techniques to various problems.

Chapter 4

Bayesian Preprocessing

In chapter 3 we motivated a probabilistic approach and Bayesian inference whenever we deal with uncertain scenarios. We also reviewed some tools that will be useful for problem solving. This chapter will investigate an *optimal* Bayesian setup for reporting some unknown states given a segment of a time series. Such problems are found in physical or natural sciences. Especially biomedical problems are often structured like that. Examples of time series are bio-signals like the electroencephalogram (EEG), the electrocardiogram (ECG) or the electromyogram (EMG). In physical sciences we find recordings of temperature, pressure or other measurements. The unknown quantity of interest is some unobserved state, usually not directly available from the time series. In biomedical applications the state is usually given by an expert's opinion, in physical sciences we might get the state by inspecting the outcome of the physical process. The novel part in this chapter are a proof that renders the classical approach of treating preprocessing separately to be wrong. Furthermore, we analyze a lattice filter structure of an AR process, which has so far not been done within a Bayesian framework.

4.1 A Bayesian view of preprocessing

Usually recordings provide much less information as is given by the amount of sampled data. The basic idea is that the unknown state of interest causes some unobserved coefficients in a stochastic process which is responsible for the observed time series. The problem is classically solved by partitioning the analysis in a preprocessing and a post processing stage. Preprocessing obtains some features which are a compressed representation of the time series. The classification stage uses some function that maps these features to the unknown state of interest. The uncertainty associated with this state is evident and usually taken it into account by modeling the probability of the state of interest. The classical approach is to condition the state probabilities on the preprocessed features. This means to condition on latent variables which can be shown to be suboptimal. The extracted features are latent (unobserved) variables - hence conditioning on them is equivalent to neglecting the uncertainty associated with their values.

Considering the problem from a Bayesian point of view, we have two sets of unobserved variables. Obviously the first one is the unknown state we want to infer. Being unknown, the Bayesian framework tells us that we should report probabilities to express our beliefs about this state. So far this is still identical to the classical approach. Contrary to the classical approach, we use a hierarchical model with a second unobserved variable. This second latent variable are the coefficients of the dynamic process that generates the time series. Hence we have another unknown quantity with some uncertainty attached to its value. The Bayesian framework tells us that we have to take this uncertainty into account. Figure 4.1 provides an illustration of this hierarchical model in terms of a directed acyclic graph (DAG).

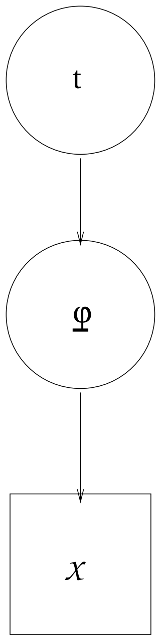

Figure 4.1: A directed acyclic graph for the hierarchical model. The variable t denotes the unknown state variable of interest, $\underline{\varphi}$ are latent (unobserved) variables representing features from preprocessing and $\mathcal{X}$ denotes a segment of a time series. Circles indicate latent variables, whereas the square indicates that $\mathcal{X}$ is an observed quantity.

The probabilistic graph in figure 4.1 is just a graphical representation of conditional independence underlying the hierarchical approaches considered in this section. We can state this independence formally as:

$$p(t = t_j, \underline{\varphi}, \mathcal{X}) = P(t = t_j)p(\underline{\varphi}|t = t_j)p(\mathcal{X}|\underline{\varphi}). \tag{4.1}$$

The probabilistic dependency between features $\underline{\varphi}$ and a segment of the observed time series $\mathcal{X}$ is motivated by the stochastic model underlying all preprocessing techniques. This assumption is true for AR-processes, but it is also true for "non-parametric" techniques like periodogram (fast Fourier based) estimates of the power spectral density. According to [RF95], we can view periodogram estimation as a generalized linear model under a Gaussian noise assumption.

The only known quantity in the DAG 4.1 is the observed segment of the time series $\mathcal{X}$. Hence there is no uncertainty associated with $\mathcal{X}$[1], conditioning on it, we get the best estimate of the probability of $t = t_j$, $P(t = t_j|\mathcal{X})$. An important remark is that conditioning on $\mathcal{X}$ is just another notion of using information to change beliefs. Applying Bayes theorem, we transform the joint probability density in (4.1) and get the conditional density:

$$p(t = t_j, \underline{\varphi}|\mathcal{X}) = P(t = t_j|\underline{\varphi})p(\underline{\varphi}|\mathcal{X}). \tag{4.2}$$

We get $P(t = t_j|\mathcal{X})$ by removing $\underline{\varphi}$, i.e. solving the *marginalization* integral:

$$P(t = t_j|\mathcal{X}) = \int_{\underline{\varphi}=-\infty}^{\infty} P(t = t_j|\underline{\varphi})p(\underline{\varphi}|\mathcal{X})d\underline{\varphi}. \tag{4.3}$$

Contrary to (4.3), the classical approach to predicting the probability of the unknown state t would be to use a point estimate of the feature vector, $\hat{\underline{\varphi}}$, and approximate the integral by $P(t = t_j|\hat{\underline{\varphi}})$. This is indeed an approximation of the integral in (4.3), where the posterior distribution $p(\underline{\varphi}|\mathcal{X})$ is replaced by a degenerate posterior that puts all its probability mass to the point estimate $\hat{\underline{\varphi}}$. Intuitively this approach is suboptimal because it neglects the uncertainty we have about $\underline{\varphi}$.

We will now prove that solving the marginalization integral (4.3) is the *only* "Bayesian" way of dealing with models represented by a DAG like the one in figure 4.1. This DAG incorporates the classical approach, thus conditioning on some best estimate of a feature, $\hat{\underline{\varphi}}$, is not consistent with the

[1]That is, we neglect any uncertainties about the observed time series such as measurement noise. Such uncertanties can be taken into account by using similar ideas as are fomulated below and in chapter 7.

Bayesian framework. More precisely, the classical approach means using the wrong preference relation between options as it appears in definition 3.1.1.

First we formulate reporting the unknown state of $t \in \{t_1, ..., t_k\}$ as a generalized decision problem. We allow to report class label $k \in \{t_1, ..., t_j, u\}$, with u being the label for unknown (doubt case). A decision d is a mapping of an arbitrary $(t_j, \underline{\varphi}) \in \Omega$ to a consequence c_j. For simplicity, we assume that there are 3 consequences: $c = c_w$ if we report the wrong state, $c = c_u$ if we refuse to report any state (reporting doubt case u) and $c = c_c$ if we report the correct state. Assuming that reporting t correctly is of most value, we require that $c_w < c_c$ and $c_u < c_c$. Hence we have a generalized decision problem with bounded consequences. The worst consequence is $c_* = c_w$ or $c_* = c_u$, whichever is smaller, and the best consequence is $c^* = c_c$. This definition of possible consequences depends only on the state variable t. The *expected* utility however will also depend on $\underline{\varphi}$.

Proposition 4.1.1 (Consistency of marginalization) *Given the DAG in 4.1, the posteriori belief $P(t|\mathcal{X})$ about a state variable t, obtained by solving the marginalization integral (4.3), is the only belief consistent with the Bayesian framework.*

Proof. We will proof this by deriving the expected utility of reporting state $t = k$ when observing data $\mathcal{X}$:

$$\bar{u}(d(\underline{\varphi}, t_j), c_*, c^*, \mathcal{X}) = \int_{\underline{\varphi}} \sum_j u(d(\underline{\varphi}, t_j), c_*, c^*) p(t, \underline{\varphi}|\mathcal{X}) d\underline{\varphi}$$

Using (4.2) we get:

$$\bar{u}(d(\underline{\varphi}, t_j), c_*, c^*, \mathcal{D}) = \int_{\underline{\varphi}} \sum_j u(d(\underline{\varphi}, t_j), c_*, c^*) P(t|\underline{\varphi}) p(\underline{\varphi}|\mathcal{X}) d\underline{\varphi}. \qquad (4.4)$$

Since $d(t_j, \underline{\varphi})$ is just a function of t_j we can solve the integral over $\underline{\varphi}$ in(4.4) and finally arrive at:

$$\bar{u}(d(t_j), c_*, c^*, \mathcal{D}) = \sum_j u(d(t_j), c_*, c^*) P(t|\mathcal{X}) \qquad (4.5)$$

The integral solved in order to get (4.5) from (4.4) is exactly the marginalization integral (4.3). From the step that took us from (4.4) to (4.5), we see that the belief about t in the definition of the utility function is obtained by solving (4.3). Hence the proof follows from proposition 3.1.3, which tells us that the expected utility, $\bar{u}(d(t_j), c_*, c^*, \mathcal{X})$, is the *only* consistent measure of the conditional preference relation $\leq_{\mathcal{X}}$ between options. $\diamond$

The consequence of proposition 4.1.1 is that any other approaches of inferring beliefs than the one stated in (4.3), *are not consistent with the information provided by $\mathcal{X}$.* This includes any technique based on conditioning on a best estimate of a feature variable $\underline{\varphi}$.

4.2 A Bayesian lattice filter

We may summarize the last section in one sentence: As soon as we report uncertainties about some state within a Bayesian setting, we have to do this throughout the entire system. The question remains: Where is the point when we start to think "Bayesian"? There might be different opinions but according to the above principles, it starts as soon as we report uncertainties about classes by using posterior probabilities for classes. Therefore it seems imperative that preprocessing should be done within a Bayesian setting. In this chapter we infer reflection coefficients, the parameters of the lattice filter structure of an AR process, within a Bayesian framework. We use a non-informative prior and derive an expression of the posterior distribution of the m-th reflection coefficient given the distributions of all lower order coefficients. Based on this posterior distribution, we derive an analytic expression of the posterior volume (the Bayesian model evidence) and show how this can be used for model order estimation. The posterior probability of a model *must* be based on the Bayesian model evidence. Following the axioms and propositions summarized in the chapter 3, the posterior probability of a model is *the only* consistent measure of the degree of belief given our prior assumptions and the data. Having more than one choice for a particular dataset, we can use the posterior probability of the model to find the "best" model. Alternatively, and consistently with the last section, we must use them all, weighted by the posterior probabilities. However, we can also compare the degree of beliefs of models inferred on *different* data sets. This requires that we have common reference model - which is easily obtained by the proposed approach.

4.2.1 A lattice filter representation of AR-processes

Autoregressive (AR) processes are often used in signal processing applications and the variety of parameter estimation methods is vast. A Bayesian analysis of static AR-models is well known. Among several researchers, it was done by [KG85] and by [BJ76]. More recently a Bayesian analysis of AR model coefficients appeared in [RF95] in several different sections. All this literature has in common that the researchers have calculated predictive distributions over AR-model coefficients.

We aim at a different solution that uses an order recursive implementation of Bayesian inference which is fast and nevertheless provides the benefits of a Bayesian solution: we get a distribution over parameters and the possibility to calculate the model evidence. The latter allows us to determine the appropriate model order and it serves as a reliability measure. The approach taken here is similar to the Laplace method described in section 3.2.4.

The AR-process is a time series model that is often used in practice. Its popularity is motivated by the linearity of the model, which makes parameter estimation simple. Equation 4.6 shows the generator transfer funtion of an m-th order model

$$y_t = \epsilon - \sum_{k=1}^{m} y_{t-k} a_k, \tag{4.6}$$

where we used ϵ as one sample of zero mean Gaussian noise, a_k, as AR-model coefficients and y_t as observation of the time series at time t. The coefficients of the order recursive representation of AR processes are the so called reflection coefficients and the corresponding model is called *lattice filter*. As reviewed in [Lju99] a certain setup in calculating the AR model coefficients, a_k, leads to the so called Levinson algorithm (4.7). The Levinson algorithm shows us the connection between AR-process and reflection coefficients, ρ_m:

$$\begin{aligned} a_k^{m+1} &= a_k^m + \rho_m a_{m-k+1}^m \\ a_{m+1}^{m+1} &= \rho_m \end{aligned} \tag{4.7}$$

The superscripts used in (4.7) denote the model order, the subscripts are the indices of the AR-coefficients and $\hat{\rho}_m$ is the m-th order reflection coefficient.

4.2.2 The likelihood of reflection- and AR-coefficients

In order to be able to express the posterior distribution over m-th order reflection coefficients, we need an expression for their likelihood. As usual for AR-processes, we assume that the process is driven by an i.i.d. Gaussian noise process. This simple noise model enables us to carry out all calculations analytically. Gaussian noise results in the likelihood, (4.8), of the $m+1$-th order AR-coefficients and the noise level, β.

$$p(\mathcal{X}|\underline{\theta}^{m+1}, \beta) = \frac{1}{Z_{\mathcal{X}}(\beta)} \exp\left(-\frac{1}{2}\beta(\underline{y} - \underline{Y}^{m+1}\underline{\theta}^{m+1})^T(\underline{y} - \underline{Y}^{m+1}\underline{\theta}^{m+1})\right) \tag{4.8}$$

The vector $\underline{y}$ denotes the observed samples from the AR-process, $\underline{Y}$ denotes the embedding matrix and $\mathcal{X}$ denotes the data[2]. Plugging (4.7) into (4.8)

[2]In a Bayesian setting it is advisable to express the likelihood as a distribution of the data conditioned on model coefficients.

we get the likelihood of the m-th order reflection coefficient, ρ_m, which still depends on the m-th order AR-coefficients, $\underline{\theta}^m$, and the noise level β. When we express $\underline{\theta}^m = \hat{\underline{\theta}}^m + \underline{\Delta\theta}^m$, where $\underline{\Delta\theta}^m$ is a multivariate zero mean Gaussian, then we arrive at the following likelihood expression:

$$\begin{aligned} p(\mathcal{X}|\underline{\Delta\theta}^m, \rho_m, \beta) &= \frac{1}{Z_{\mathcal{X}}(\beta)} \exp(-\frac{1}{2}\beta((\underline{\epsilon}^m + \rho_m \underline{r}^m)^T(\underline{\epsilon}^m + \rho_m \underline{r}^m) \\ &+ 2(\underline{\epsilon}^m + \rho_m \underline{r}^m)(\underline{Y}^m + \rho_m \underline{Y}^{mflr})\underline{\Delta\theta}^m \\ &+ \underline{\Delta\theta}^{mT}(\underline{Y}^m + \rho_m \underline{Y}^{mflr})^T(\underline{Y}^m + \rho_m \underline{Y}^{mflr})\underline{\Delta\theta}^m)) \end{aligned} \quad (4.9)$$

We use $\underline{\epsilon}^m$ to denote the forward prediction error of the m-th order model and $\underline{r}^m$ for the reverse time prediction error at lag -1. The quantity $Z_{\mathcal{X}}(\beta)$ is the normalization constant which we calculate as follows:

$$Z_{\mathcal{X}}(\beta) = \underbrace{\int_{-\infty}^{\infty} ... \int_{-\infty}^{\infty}}_{N} exp\left(-\frac{\beta}{2}(\epsilon_n^m + \rho_m r_n^m)^2\right) d\epsilon_n^m = (\frac{2\pi}{\beta})^{\frac{N}{2}}. \quad (4.10)$$

We will now express the posterior distribution of the m-th order reflection coefficient. We do this by using appropriate priors and applying Bayes' theorem. The resulting expression is a joint probability density function (pdf) of the reflection coefficient, ρ_m, the *variation* of the m-th order AR coefficients, $\underline{\Delta\theta}^m$, and the noise level β.

4.2.3 Priors for noise levels and reflection coefficients

The priors required by any Bayesian analysis should reflect our prior beliefs about the model coefficients. In general we do not have much prior information about the functions we want to model and even when we have, transforming them into priors over model coefficients is still not easy. Hence we will usually resort to using some uninformative normalizeable priors.

Fortunately this is not too difficult in the case of the lattice filter model: In biomedical systems, we usually expect to find *stable* models. This allows us to constrain the reflection coefficients within the closed interval $[-1, 1]$ and the uninformative prior is $p(\rho_m) = 0.5$.

The situation for the noise level β is somewhat less convenient: Usually we do not know how to constrain the noise level. We know however that the noise level β is a scale parameter. Hence we will follow [Jef61] and use a Jeffreys' prior, $p(\beta) = 1/\beta$.

4.2.4 A posterior distribution over reflection coefficients

In this section we will derive a posterior distribution over reflection coefficients. We combine the prior distributions over the reflection coefficient, the noise level and the Gaussian over the *variation* of the m-th order AR coefficient with the likelihood expression (4.9). Applying Bayes' theorem, we get the posterior distribution $p(\rho_m, \beta, \underline{\Delta\theta}^m|\mathcal{X})$. A final marginalization step, where we integrate over β and $\underline{\Delta\theta}^m$ leads us to the final goal, the posterior distribution over the m-th order reflection coefficient, $p(\rho_m|\mathcal{X})$.

Multiplying (4.9) with the prior distributions $p(\beta) = 1/\beta$, $p(\rho_m) = 0.5$ and $p(\underline{\Delta\theta}^m) = 1/C \exp(-0.5\underline{\Delta\theta}^{mT}\underline{H}\underline{\Delta\theta}^m)$ gives us:

$$\begin{aligned} p(\rho_m, \beta, \underline{\Delta\theta}^m|\mathcal{X}) \propto & \frac{1}{2Z_{\mathcal{X}}(\beta)C\beta} \exp(-\frac{1}{2}\beta((\underline{\epsilon}^m + \rho_m\underline{r}^m)^T(\underline{\epsilon}^m + \rho_m\underline{r}^m) \\ + & 2(\underline{\epsilon}^m + \rho_m\underline{r}^m)(\underline{Y}^m + \rho_m\underline{Y}^{mflr})\underline{\Delta\theta}^m \\ + & \underline{\Delta\theta}^{mT}(\underline{Y}^m + \rho_m\underline{Y}^{mflr})^T(\underline{Y}^m + \rho_m\underline{Y}^{mflr})\underline{\Delta\theta}^m) \\ + & \underline{\Delta\theta}^{mT}\underline{H}\underline{\Delta\theta}^m). \end{aligned} \tag{4.11}$$

The remaining steps of this section are concerned with finding the marginal distribution over ρ_m. First we will integrate over $\underline{\Delta\theta}^m$. Exact marginalization renders the subsequent analysis intractable and we have to introduce the following approximation. Assuming that $p(\underline{\Delta\theta}^m)$ is sharply peaked around zero, we get

$$\int_{-\infty}^{\infty} p(\rho_m, \beta, \underline{\Delta\theta}^m|\mathcal{X})d\underline{\Delta\theta}^m \approx p(\rho_m, \beta|\mathcal{D}; \hat{\underline{\Delta\theta}}^m).$$

Note that $\hat{\underline{\Delta\theta}}^m \equiv 0$. This seems a rather rough approximation. However, similar arguments are commonly used within the Bayesian community. An example is [Bre90], who uses the same approximation to derive an analytic expression for the Bayesian evidence of generalized linear models. Using the above approximation, we get

$$p(\rho_m, \beta|\mathcal{X}) \propto \frac{1}{2Z_{\mathcal{X}}(\beta)\beta} \exp(-\frac{1}{2}\beta((\underline{\epsilon}^m + \rho_m\underline{r}^m)^T(\underline{\epsilon}^m + \rho_m\underline{r}^m))) \tag{4.12}$$

as an expression of the joint posterior distribution over the m-th order reflection coefficient and the noise level.

The marginalization integral that takes us from (4.12) to the posterior distribution over the m-th order reflection is:

$$\int_0^{\infty} p(\rho_m, \beta|\mathcal{X})d\beta.$$

After some calculations (details are found in appendix A) we get as a final result:

$$\begin{aligned} p(\rho_m|\mathcal{X}) \quad &\propto \quad 0.5\pi^{-\frac{N}{2}}\Gamma(\frac{N}{2})\left((\underline{\epsilon}^m + \hat{\rho}_m\underline{r}^m)^T(\underline{\epsilon}^m + \hat{\rho}_m\underline{r}^m)\right)^{-\frac{N}{2}}\sqrt{2\pi}s \\ & \underbrace{\frac{1}{\sqrt{2\pi}s}\exp\left(-\frac{1}{2s^2}(\rho_m - \hat{\rho}_m)^2\right)}_{\text{normalized Gaussian}}, \end{aligned} \tag{4.13}$$

as expression of the posterior distribution of the m-th order reflection coefficient. The posterior in 4.13 is a normalized Gaussian and a multiplicative factor, which is the Bayesian model evidence that will be used below for model selection. Furthermore we find:

$$\hat{\rho}^m = -\frac{\underline{r}^{mT}\underline{\epsilon}^m}{\underline{r}^{mT}\underline{r}^m} \tag{4.14}$$

as expression of the most probable value of the reflection coefficient and

$$s^2 = \frac{1-(\hat{\rho}^m)^2}{(N-1)} \tag{4.15}$$

as corresponding variance. The variance of the reflection coefficient depends on the estimate of the most probable value. It is interesting to see that the variance tends to zero as soon as the reflection coefficient approaches one. Although it is already evident from our derivation of the posterior probability over reflection coefficients, we want to add as a final remark that if m reflection coefficients are extracted from a time series, the reflection coefficients exhibit conditional independence $p(\rho_1, ...\rho_m|\mathcal{X}) = \prod_m p(\rho_m|\mathcal{X})$.

4.2.5 Bayesian evidence and model selection

In this subsection we use an interesting aspect of Bayesian learning: its capability to perform model selection. In general, Bayesian inference reports conditional beliefs of different models, where we condition on the data segment used for inference. This requires that there must be more than one possible model (or explanation) of a data segment. In the context of estimating the appropriate model order of an AR lattice filter structure, we have to compare against a “pure noise explanation”. If the probability of “reflection coefficient” is larger than the probability of “noise only”, we should opt for using the coefficient. We calculate the probability for the k-th model by proceeding within the Bayesian framework in a hierarchical manner. The posterior probability for the k'th model is expressed as:

$$p(I_k|\mathcal{X}) = \frac{p(\mathcal{X}|I_k)p(I_k)}{p(\mathcal{X})}.$$

Assuming an uninformative prior for the k'th model of $p(I_k) = 1/K$ with K denoting the total number of models, we see that model selection should be done according to the likelihood function of the k-th model $p(\mathcal{X}|I_k)$. This quantity is the so called Bayesian model evidence. It is exactly the normalization constant of the posterior over all model coefficients. As such, we have already derived the model evidence of the m-th reflection coefficient in the last section: it is the factor the normalized Gaussian in (4.13) is multiplied with.

However, the usual approach to Bayesian model order selection requires to compare different models on the *same* data. In the context of lattice filters each step estimates only one coefficient. Hence we have to compare the Bayesian evidence of a filter using the m-th reflection coefficient to the Bayesian evidence of a lattice filter that does not use the m-th coefficient. This is equivalent to the question whether the evidence using the m-th coefficient is larger than the evidence assuming a pure noise process for the m-th stage of the lattice filter.

As already mentioned, the Bayesian model evidence of the m-th reflection coefficient is the multiplicative factor from (4.13):

$$p(\mathcal{X}|I_m) = 0.5\pi^{-\frac{N}{2}}\Gamma(\frac{N}{2})\left((\underline{\epsilon}^m + \hat{\rho}_m\underline{r}^m)^T(\underline{\epsilon}^m + \hat{\rho}_m\underline{r}^m)\right)^{-\frac{N}{2}}\sqrt{2\pi}s. \tag{4.16}$$

The Bayesian evidence of not using the m-th reflection coefficient is derived in a similar way as we do for the reflection coefficient. The likelihood is just a function of the m-th order AR coefficients and the noise level β. Some calculations finally lead to the evidence expression for *not* using the m-th order coefficient. We use $\bar{I_m}$ to denote the corresponding model.

$$p(\mathcal{X}|\bar{I_m}) = \pi^{-\frac{N}{2}}\Gamma(\frac{N}{2})\underline{\epsilon}^{mT}\underline{\epsilon}^m \tag{4.17}$$

Using (4.16) and (4.17), the probability of the m-th reflection coefficient is:

$$P(I_m|\mathcal{X}) = \frac{p(\mathcal{X}|I_m)}{p(\mathcal{X}|I_m) + p(\mathcal{X}|\bar{I_m})}. \tag{4.18}$$

Contrary to the common use of the posterior probability of models, where the m probabilities sum up to 1, we have a slightly different situation here: Each of the probabilities $P(I_m|\mathcal{X})$ is between 0 and 1 and the optimal model order is the largest index m, where $P(I_m|\mathcal{X})$ is above 0.5.

4.3 Experiments

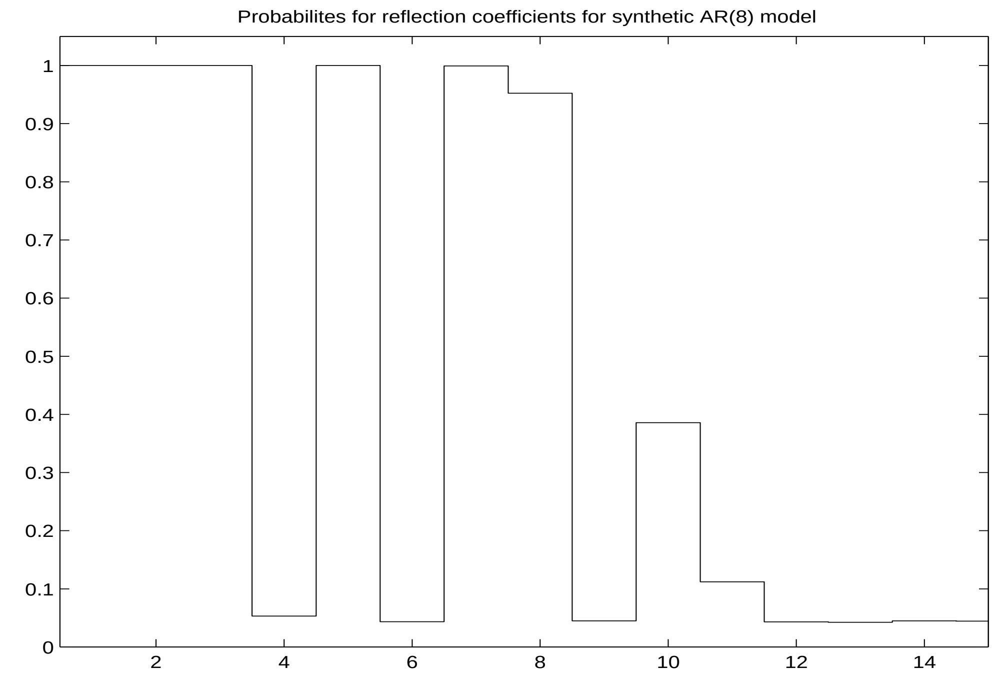

Figure 4.2: Probabilities for reflection coefficients for a synthetic 8-th order AR model (see the text for details of the model). The model was estimated from a time series with 400 samples. The 8-th reflection coefficient is the last stage that got a probability larger than 0.5. Hence the model order was estimated correctly.

The experiments performed with the Bayesian lattice filter proposed in section 4.2 serve two purposes. First of all we need to check whether the approach is correct. These checks are best performed using data from synthetic AR-models. In the context of the sleep analysis project that was described in section 2, we are interested in applying the lattice filter as preprocessing technique. Our main interest is in assessing whether small model probabilities correspond to segments of EEG that human experts consider to be contaminated by some artefact. As white noise leads to small model probability, this is definitely true if an artifact corresponds to contamination with white noise. Our interest in investigating the behaviour of the algorithm in connection with artefacts is motivated by our aim to avoid giving wrong decisions which might be due to such artifacts. An example might illustrate this: the estimates obtained from sleep EEG during wake will be similar to those obtained from white noise. However an EEG signal that looks like white noise has most probaly no physiological origin. It is usualy due to some technical problem. Hence a subsequent analysis would decide "wake" where the true decision should be "artefact". The experiments concerned with artefacts will be done using data from all night recordings.

Correctness of the most probable coefficient as defined in equation (4.14) is easy to verify. A simple comparison of the estimates obtained from (4.14) with the results obtained by the ar-function provided in the MatLab systems identification toolbox reveals that both methods are identical. The situation is slightly more difficult in case of the variance estimates (4.15) and model probabilities (4.18). Since the model probabilities depend on the variance estimates, we may assume that both are correct as soon as we have verified (4.18). Hence we decided to use two synthetic AR-models with known dimension and estimate the most probable model orders implied by (4.18). The optimal model order is the index of the last lattice filter stage whos model probability (4.18) is larger 0.5.

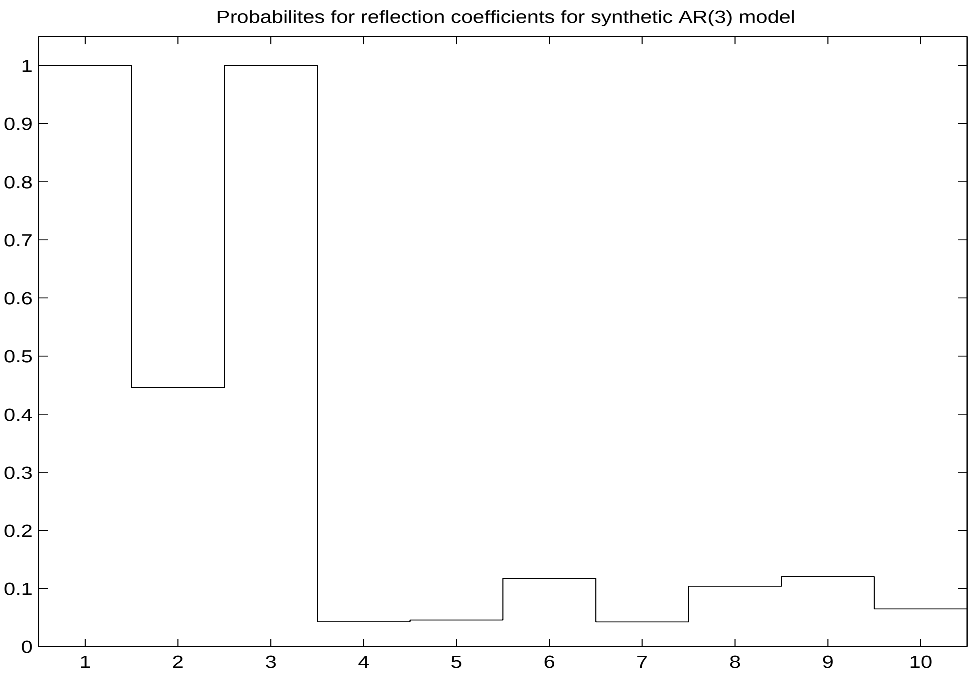

Figure 4.3: Probabilities for reflection coefficients for a synthetic 3-rd order AR model (see the text for details of the model). The model was estimated from a time series with 400 samples. The 3-rd reflection coefficient is the last lattice filter stage that got a probability larger than 0.5. This implies that the model order estimation was performed correctly.

This empirical test has been done using an 8-order AR model that has been used in [RF95] for exactly the same purpose. The poles of the generator transfer function are located at $(-0.4572 \pm \jmath 0.7765)$, $(-0.8284 \pm \jmath 0.1079)$, $(-0.5979 \pm \jmath 0.6138)$ and $(-0.5976 \pm \jmath 0.5364)$. We also performed a second test with a third order model whos transfer function poles are located at (0.9) and $(-0.5 \pm \jmath 0.5)$. The probabilities for reflection coefficients obtained for one simulation run have been plotted in figure 4.2 (for the 8-th order model) and in figure 4.3 (for the 3-rd order model).

Having shown that the algorithms work correctly on synthetic data, we will now apply the Bayesian lattice filter to sleep EEG. The emphasis of these experiments is to show that Bayesian preprocessing provides useful inputs for an analysis of all night sleep as was proposed in chapter 2. The data used in the experiments reported below has been taken from 6 EEG channels that had been recorded at electrodes Fp1, C3, O1, Fp2, C4 and O2 placed according to the international 10-20 system. The question we try to answer with the experiments described below is whether, apart from the theoretical necessity formulated in proposition 4.1.1, we have some practical advantage from using a Bayesian technique. Of course, we have already shown the first advantage: we can use (4.18) to perform model selection. It would be of considerable interest to show that low probability for reflection coefficients corresponds to segments that are judged as suspicious by human experts. In order to test that, we used a database consisting of segments of 15 different subjects that have been specifically marked for artifacts.

The experiments aim to show that the posterior probability $P(I_m|\mathcal{D})$ is indeed a measure that captures the reliability of coefficients extracted from some data. We will try to detect segments marked as artifacts by human experts. However, we do not argue that the proposed measure mimics the expert opinion about artifacts. It only works if the data contaminated by artifacts contain almost no information. Some artifacts, e.g. ECG, and small contaminations in general will not lead to lower reliability. Hence this method will not recognize such artifacts.

The data were scored by human experts on a one second basis as either contaminated by some artifact - where different artifacts have been considered separately - or as being clean. All together we have 213664 clean segments and 115736 segments marked as artifacts. Since our recordings are sampled at different rates, we had to resample[3] to a common frequency of 100 Hz. After resampling, we used two seconds windows and an offset of one second to estimate 3 reflection coefficients and the corresponding model probabilities.

In order to assess the correlation of our reliability measure with the ex-

[3]In these experiments we used the MATLAB signal processing function “resample”.

perts' opinion on artifacts, we performed two tests. In a first experiment we tried to detect so called movement artifacts by applying a rejection threshold to the probability of the first lattice filter stage. In general we want to have only few false positives, therefore we aim at a specificity of 0.99. The data used to set the threshold were excluded from any further study. Accumulating results from 7 different subjects, we get a sensitivity of 0.298 and a specificity of 0.974. In table 4.1 we see that there is some inter-subject variation.

Table 4.1: Sensitivities and specificities for 7 subjects

sens.	0.26	0.13	0.37	0.59	0.21	0.27	0.27
spec.	0.95	0.99	0.99	0.99	0.90	0.99	0.99

The setup of the first experiment is suboptimal because the reliability measure can only indicate a general problem with the reliability of a feature. It is not designed to separate different artifacts. Hence in a second experiment we looked whether we can find a correlation between low reliability and segments contaminated with *some* artifact. The critical point, where we would better suggest *not* to use a particular model, is at probability 0.5. Using this threshold, we find 2321 segments marked as contaminated and only 504 clean segments. Hence both experiments suggest that there is a correlation between artifacts and our reliability measure. However, there are some differences between expert opinions and the indications based on (4.18): Even if we are restrictive and request that *all three* lattice stages have to be extremely implausible (we request that $\forall m$ of a segment $P(I_m|\mathcal{D}) < 0.07$) [4], we still get 3 samples marked as "clean" by experts.

Despite this disagreement, we still believe that the estimates obtained by (4.18) is a useful quantity to assess the reliability of features. In order to show that, we have extracted 3 reflection coefficients[5] from all 6 EEG channels. The data was taken from healthy subjects of different gender and age. We extracted one coefficient for each second of the entire night. The coefficients were estimated using a two seconds window and one seconds

[4]Note that this is equivalent to suggesting that the data are pure noise.

[5]Extracting 15 reflection coefficients together with variances and model probabilities revealed that more than 80% of all epochs of one second length require model orders less than 5. The "magic number" 3 came from investigating how many reflection coefficients are necessary for supervised classification of Rechtschaffen & Kales (R & K) sleep stages wake, rapid eye movement (REM) and delta (combined R & K stages 3 and 4). Details are reported in [SRR+99] and in chapter 5.

overlap. Figure 4.4 provides an illustration for one all night recording. It shows the most probable value of reflection coefficient 1, the corresponding variances and model probabilities extracted at electrode C3. The plots show the entire night. For comparison we also see the corresponding Rechtschaffen and Kales (R & K) scoring. It is clearly visible that there is a correlation between the coefficient and the sleep stages.

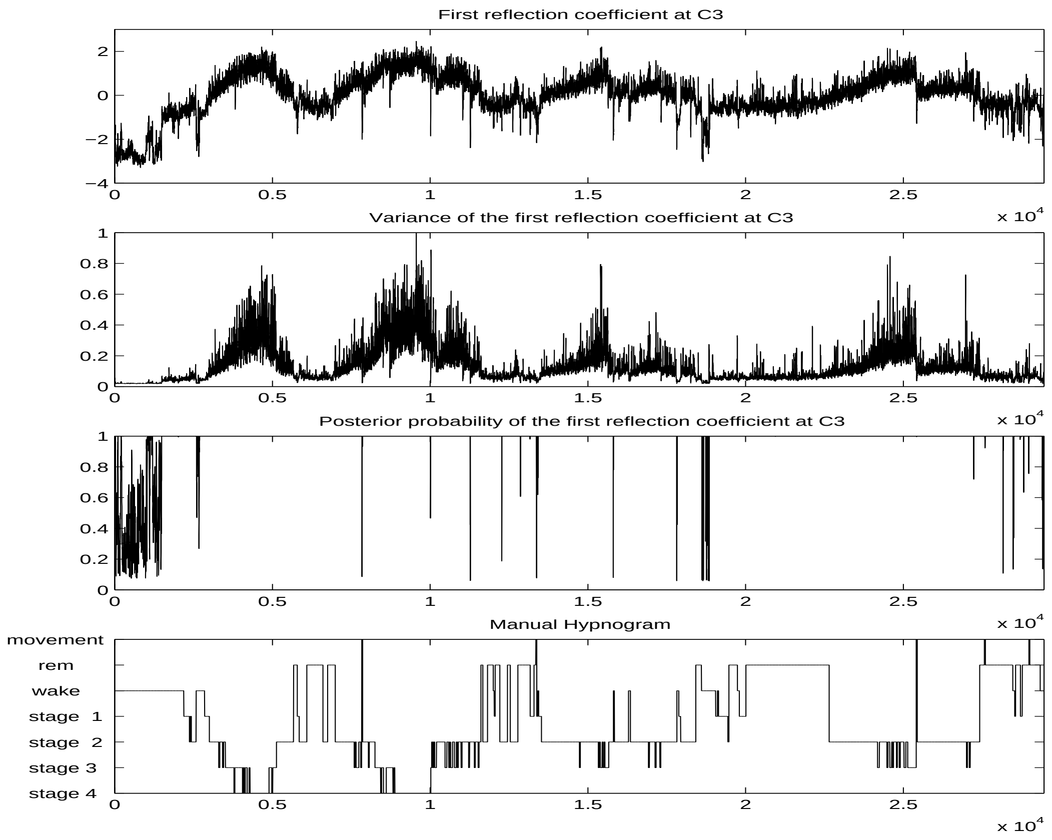

Figure 4.4: Reflection coefficient 1, the corresponding variance and model probabilities extracted for an entire night. We can clearly identify a correlation with the corresponding R & K scoring shown below.

In order to assess whether the model probability (4.18) is a useful indicator of unreliable data, we asked clinical experts for their opinion about such segments where the first lattice filter stage had a low probability. In almost every case the opinion was that the segment was contaminated by a signal that is physiologically implausible. Figure 4.5 gives an impression of the global situation for a recording with many segments of low probability. It shows the probability for the first reflection coefficient for all 6 EEG channels for the first 1000 seconds of the night. The experts were provided

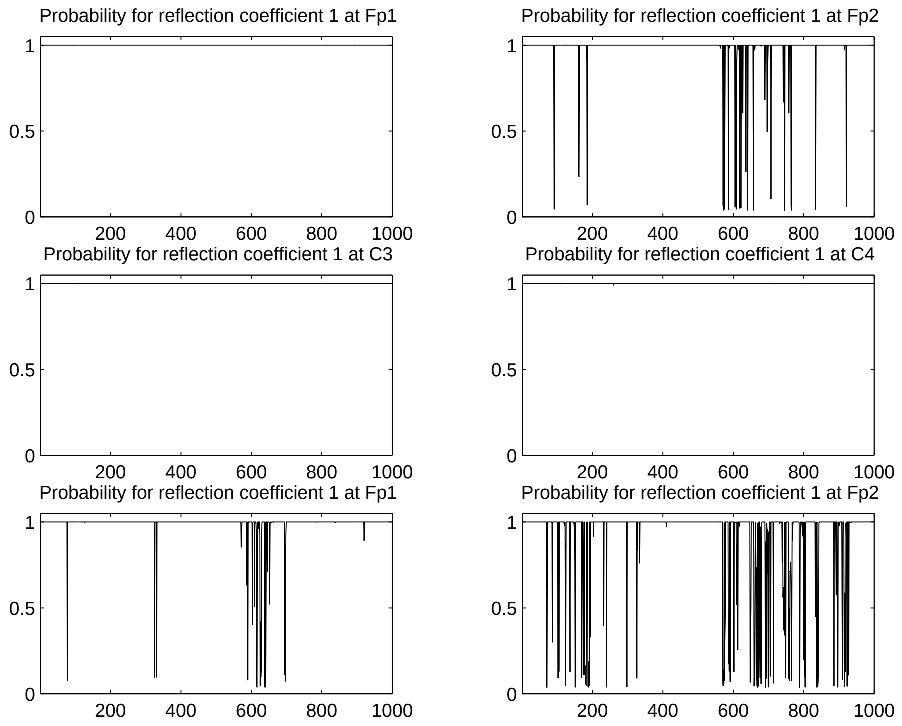

Figure 4.5: Probabilities for reflection coefficient 1 shown for the first 1000 seconds of one of the recordings. The data is representative for the entire night. This recording contains many segments where the model is rather improbable. For every such segment the corresponding estimate of the reflection coefficient is similar to estimates obtained during wake. Consultations with clinical experts ruled out that there might be a physiological origin of this effect. The effect is most probably caused by some technical problem in the recording equipment.

with plots that showed both the probability for the coefficient and the corresponding data segment. An example of such artifacts is provided in figure 4.6. It shows 10 seconds of data recorded at electrode Fp2 together with the probability of the first reflection coefficient. We can clearly identify bursts of noise that cause the low probabilities of the coefficients. Detecting these events is of vital importance, since the corresponding feature values are close to the estimates obtained from segments classified as wake. Hence if such physiologically implausible noise bursts occur, the corresponding prediction will missinterpret this artefact as an arousal.

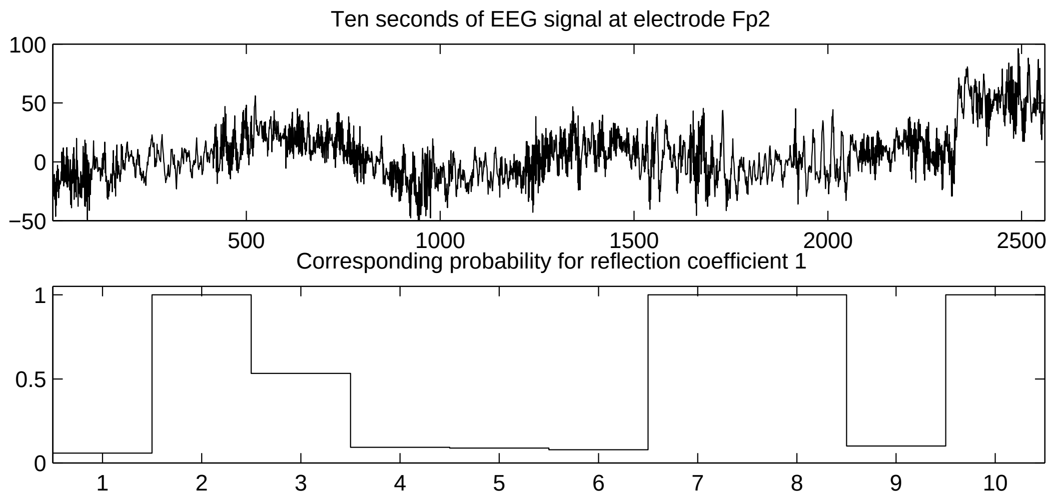

Figure 4.6: These plots show 10 seconds of data from electrode Fp2 and the corresponding probabilities for the first reflection coefficient. We see that low probabilities are caused by bursts of noise. Consultations with clinical experts did not explain the cause of these bursts, however according to their opinion they have no physiological origin.

4.4 Summary

The important contributions in this chapter are proposition 4.1.1, where we prove that preprocessing *must* be Bayesian in order to allow optimal decisions to be taken in a later classification stage. Furthermore we derived a Bayesian analysis for a lattice filter representation of an AR-process. As a result we obtain a posterior distribution over reflection coefficients *and* a posterior probability for each lattice filter stage. The proposed signal processing technique was then applied to sleep EEG. We were especially interested in an assessment whether the posterior probability of the lattice filter stage is useful to determine artifacts in the signal. In an experiment we tried to detect artifacts in a manually marked database. This experiment revealed that only a small percentage of all artifacts could be detected. In fact the human opinion on artifacts and the white noise hypothesis underlying the approach taken here are quite different. However, when presenting the flagged segments to human experts they agreed in most cases that the data contained some signal that had no physiological origin. Low probability of a lattice filter stage is coupled with feature estimates that are similar to the estimates observed during wake. Thus conditioning on these estimates, a classifier will predict high probability for stage wake, which is *not* the cause of the data.

In this perspective the model probabilities of the lattice filter stage can be seen as a promising means to avoid such failure. This observation motivates the approach to time series classification laid out in chapter 7. There we will treat both feature and model uncertainties in the correct Bayesian way: by marginalizing them out.

Appendix A

In this appendix we show how to derive (4.13) from (4.12). We start solving the marginalization integral

$$\int_0^\infty p(\rho_m, \beta|\mathcal{X})d\beta$$

by substituting $u = \frac{1}{2}\beta((\underline{\epsilon}^m + \rho_m\underline{r}^m)^T(\underline{\epsilon}^m + \rho_m\underline{r}^m))$. This leads to:

$$p(\rho_m|\mathcal{X}) \propto \pi^{-\frac{N}{2}}0.5\left((\underline{\epsilon}^m + \rho_m\underline{r}^m)^T(\underline{\epsilon}^m + \rho_m\underline{r}^m)\right)^{-\frac{N}{2}} \int_0^\infty u^{\frac{N}{2}-1}\exp(-u)du.$$

Using the definition of the Gamma function:

$$\Gamma(\frac{N}{2}) = \int_0^\infty u^{\frac{N}{2}-1}\exp(-u)du,$$

we get:

$$p(\rho_m|\mathcal{X}) \propto \pi^{-\frac{N}{2}}0.5\Gamma(\frac{N}{2})\left((\underline{\epsilon}^m + \rho_m\underline{r}^m)^T(\underline{\epsilon}^m + \rho_m\underline{r}^m)\right)^{-\frac{N}{2}}.$$

This is a t-distribution with $N-1$ degrees of freedom, which is more easily seen by developing a Taylor series expansion around the most probable value for ρ_m. The most probable value, $\hat{\rho}_m$, is found by solving a least squares problem and reads as:

$$\hat{\rho}^m = -\frac{\underline{r}^{mT}\underline{\epsilon}^m}{\underline{r}^{mT}\underline{r}^m}.$$

As the gradient at $\hat{\rho}_m$ vanishes, we get the following exact Taylor series expansion:

$$(\underline{\epsilon}^m + \rho_m\underline{r}^m)^T(\underline{\epsilon}^m + \rho_m\underline{r}^m) = (\underline{\epsilon}^m + \hat{\rho}_m\underline{r}^m)^T(\underline{\epsilon}^m + \hat{\rho}_m\underline{r}^m) + \underline{r}^{mT}\underline{r}^m(\rho_m - \hat{\rho}_m)^2.$$

Plugging this expansion into $p(\rho_m|\mathcal{X})$, we get:

$$\begin{aligned} p(\rho_m|\mathcal{X}) \quad &\propto \quad 0.5\pi^{-\frac{N}{2}}\Gamma(\frac{N}{2})\left((\underline{\epsilon}^m + \hat{\rho}_m\underline{r}^m)^T(\underline{\epsilon}^m + \hat{\rho}_m\underline{r}^m)\right)^{-\frac{N}{2}} \qquad (4.19) \\ &\underbrace{\left(1 + \frac{(\rho_m - \hat{\rho}_m)^2}{(N-1)s^2}\right)^{-\frac{N}{2}}}_{\propto\text{t-distribution}}, \end{aligned}$$

with

$$s^2 = \frac{(\underline{\epsilon}^m + \hat{\rho}_m \underline{r}^m)^T (\underline{\epsilon}^m + \hat{\rho}_m \underline{r}^m)}{(N-1)\underline{r}^{mT}\underline{r}^m} \tag{4.20}$$

denoting the variance of the t-distribution. The normalization constant of the unnormalized t-distribution in (4.19) is given as:

$$\int_{-\infty}^{\infty} \left(1 + \frac{(\rho_m - \hat{\rho}_m)^2}{(N-1)s^2}\right)^{-\frac{N}{2}} d\rho_m = \sqrt{2\pi} s (N-1)^{0.5} \frac{\Gamma(\frac{N-1}{2})}{\Gamma(\frac{N}{2})}.$$

Given a sufficiently large number of data points, e.g. $N > 50$, we are allowed to use the normal distribution instead of the t-distribution. As the limit

$$\lim_{N\to\infty} (N-1)^{0.5} \frac{\Gamma(\frac{N-1}{2})}{\Gamma(\frac{N}{2})} = 1,$$

we get as a final expression for the posterior distribution over the m-th order reflection coefficient:

$$\begin{aligned} p(\rho_m|\mathcal{X}) \quad \propto \quad & 0.5\pi^{-\frac{N}{2}} \Gamma(\frac{N}{2}) \left((\underline{\epsilon}^m + \hat{\rho}_m \underline{r}^m)^T (\underline{\epsilon}^m + \hat{\rho}_m \underline{r}^m)\right)^{-\frac{N}{2}} \sqrt{2\pi} s \\ & \underbrace{\frac{1}{\sqrt{2\pi} s} \exp\left(-\frac{1}{2s^2}(\rho_m - \hat{\rho}_m)^2\right)}_{\text{normalized Gaussian}}, \end{aligned}$$

which is our final result (4.13) from the methods section. A simplification of the expression of the variance, s^2, from (4.20) is also possible. Making the most probable reflection coefficient explicit and using the identity (see [Lju99]): $\underline{\epsilon}^{mT}\underline{\epsilon}^m \equiv \underline{r}^{mT}\underline{r}^m$, we get:

$$s^2 = \frac{1 - (\hat{\rho}^m)^2}{2(N-1)},$$

which is the result reported in (4.15).

Chapter 5

A Bayesian wrapper

In this chapter we will treat input selection for a radial basis function (RBF)-like classifier within a Bayesian framework. We approximate the posterior distribution over both model coefficients and input subsets by samples drawn with Gibbs updates and reversible jump moves. According to the terminology used in [KJ97], where an algorithm that uses the final classifier for feature subset selection is refered to as *wrapper*, we call the approach the *Bayesian wrapper*.

Using some public datasets, we compare the classification accuracy of the method with a conventional automatic relevance determination (ARD) scheme. These datasets are also used to infer the posterior probabilities of different input subsets. A final experiment reveals that a rather small number of input features suffices to build the sleep analyzer developed in the EC funded Biomed project SIESTA[1]. It should be mentioned that this chapter is an extended version of [Syk00]. Parts of the results have also been published as extended abstract in [SRR$^+$99].

5.1 Introduction

Methods that aim at determining relevance of inputs have always interested researchers in various communities. Classical feature subset selection techniques, as reviewed in [DK82], use search algorithms and evaluation criteria to determine *one* optimal subset. Although these approaches can improve classification accuracy, they do not explore different equally probable subsets. Automatic relevance determination (ARD) is another approach which determines relevance of inputs. ARD is due to [Nea96] who uses Bayesian

[1]For a detailed reference to the project see the Acknowledgements at the end of the thesis.

techniques, where hierarchical priors penalize irrelevant inputs.

Our approach is also "Bayesian". Relevance of inputs is measured by a probability distribution over all possible feature subsets. This probability measure is determined by the Bayesian evidence of the corresponding models. The general idea was already used in [PS96] for variable selection in linear regression models. Our interest is different since we select inputs for a nonlinear classification model. We aim at an approximation of the true distribution over *all* different subsets. As the number of subsets grows exponentially with the total number of inputs, we can not calculate Bayesian model evidence directly. We need a method that samples efficiently across different dimensional parameter spaces. Besides the jump diffusion sampler from [GM94] and the product space approach from [CC95], the most general method that achieve this is the reversible jump Markov chain Monte Carlo sampler (reversible jump MC) recently proposed in [Gre95]. The approach was successfully applied by [RG97] to determine a probability distribution in a mixture density model with a variable number of kernels and in [HM98] to sample from the posterior of RBF regression networks with a variable number of kernels. A Markov chain that switches between different input subsets is useful for two tasks; counting how often a particular subset was visited gives us a relevance measure of the corresponding inputs; for classification, we approximate the integral over input sets and coefficients by summation over samples from the Markov chain.

The next sections will show how to implement such a reversible jump MC and apply the proposed algorithm to classification and input evaluation using some public datasets. Though the approach could not improve the MLP-ARD scheme from [Nea96] in terms of classification accuracy, we still think that it is interesting. We can assess the importance of different *feature subsets* which is different than importance of *single features* as estimated by ARD.

5.2 Methods

The classifier used in this chapter is a RBF like model. Inference is performed within a Bayesian framework. When conditioning on *one* set of inputs, the posterior over model parameters is already multimodal. Therefore we resort to Markov chain Monte Carlo (MCMC) sampling techniques to approximate the desired posterior over both model coefficients and feature subsets. In the following subsections we will propose an appropriate architecture for the classifier and a *hybrid sampler* for model inference. This hybrid sampler consists of two parts. We use (a) Gibbs updates ([GG84]) to sample when conditioning on a particular set of inputs, and, (b) reversible jump moves that carry out dimension switching updates.

5.2.1 The classifier

In order to allow input relevance determination by Bayesian model selection, the classifier needs at least one coefficient that is associated with each input. Roughly speaking, the probability of each model is proportional to the likelihood of the most probable coefficients, weighted by their posterior width and divided by their prior width. The first factor always increases when using more coefficients (or input features). The second will decrease the more inputs we use. Together this gives a peak for the most probable model. A classifier that satisfies these constraints is the so called *classification in the sampling paradigm.* We model class conditional densities and, together with class priors, express posterior probabilities for classes. In neural network literature this approach was first proposed in [Trå91]. We use a model that allows for overlapping class conditional densities:

$$p(\underline{x}|k) = \sum_{d=1}^{D} w_{kd} p(\underline{x}|\underline{\Phi}_d) \ , \ p(\underline{x}) = \sum_{k=1}^{K} P_k p(\underline{x}|k) \tag{5.1}$$

Using P_k for the K class priors and $p(\underline{x}|k)$ for the class conditional densities, (5.1) can be used to calculate posterior probabilities for the classes as $P(k|\underline{x}) = P_k p(\underline{x}|k)/p(\underline{x})$. We choose the component densities, $p(\underline{x}|\underline{\Phi}_d)$, to be Gaussian with restricted parametrisation. Each kernel is a multivariate normal distribution with a mean and a diagonal covariance matrix. For all Gaussian kernels together, we get $2 * D * I$ parameters, with I denoting the current input dimension and D denoting the number of kernels. Apart from kernel coefficients, Φ_d, (5.1) has D coefficients per class, w_{kd}, indicating the prior kernel allocation probabilities and K class priors. Model (5.1) allows to treat labels of patterns as *missing data.* That is we can use both labeled and unlabeled data for model inference. In this case training is carried out using the likelihood of observing inputs *and* targets:

$$p(\mathcal{T}, \mathcal{X}|\underline{\Theta}) = \Pi_{k=1}^{K} \Pi_{n_k=1}^{N_k} P_k p_k(\underline{x}_{n_k}|\underline{\Theta}_k) \Pi_{m=1}^{M} p(\underline{x}_m|\underline{\Theta}), \tag{5.2}$$

where $\mathcal{T}$ denotes labeled and $\mathcal{X}$ unlabeled training data. In (5.2) $\underline{\Theta}_k$ denotes all coefficients the k-th class conditional density depends on. We further use $\underline{\Theta}$ for all model coefficients, n_k as number of samples belonging to class k and m as index for unlabeled samples. To make Gibbs updates possible, we further introduce two latent allocation variables. The first one, d, indicates the kernel number each sample was generated from; the second one is the unobserved class label c, introduced for unlabeled data. A typical approach for training models like (5.1), e.g. [GJ94] and [SS95], is the EM algorithm, which is closely related to the Gibbs sampler introduced in the next subsection. One disadvantage of mixture models like (5.1) is that these models are not *identified*: The likelihood function,(5.2), is invariant to permutations of component indices. In any setting, where only one set of coefficients is used for predictions, this need not necessarily bother us. Howeve when applying sampling techniques, we face a completely different situation: Different sampled instances of one component index may contain parameters belonging to different components, a property which is called label switching. Label switching really makes problems: First, we are in trouble when we want to look at the component densities. But even if we are not interested in the properties of one component, label switching prevents any reasonable checks[2] of the Markov chain. For mixture models, which are up to missing probabilities for classes (and some target labels) identical to (5.1), the problem is either solved by parameter constraints (see e.g. [RG97] and [Rip96]) or by reparametrisation (see [RM99]). Another approach is due to [Ste00], who proposed a method how to obtain appropriate permutations of the sampled coefficients.

5.2.2 Fixed dimension sampling

In this subsection we will formulate Gibbs updates for sampling from the posterior when conditioning on a fixed set of inputs. Figure 5.1 shows the directed acyclic graph (DAG) that illustrates the conditional independence structure introduced by these latent variables. It also contains prior specifications that follow largely the settings of [RG97].

For all but the kernel variances, we know how to specify reasonable priors. Hence a detail in figure 5.1 worth mentioning is that we use a hierarchical prior specification for the inverse kernel variances. In order to allow the hyper parameter β to evolve, it has a hyper prior, controlled by the two hyper hyper parameters g and h.

According to [GRe96], Gibbs sampling is a special case of single component Metropolis Hastings updates, where the proposal distribution is the *full conditional*[3] of the updated parameter. Hence invariance of the target distribution is guaranteed by the detailed balance condition met by all Metropolis

[2]We are interested in an assessment of convergence and whether the Markov chain visits different modes of the posterior distribution, which is called *good mixing*.

[3]We condition on the current values of all other coefficients occurring in the model.

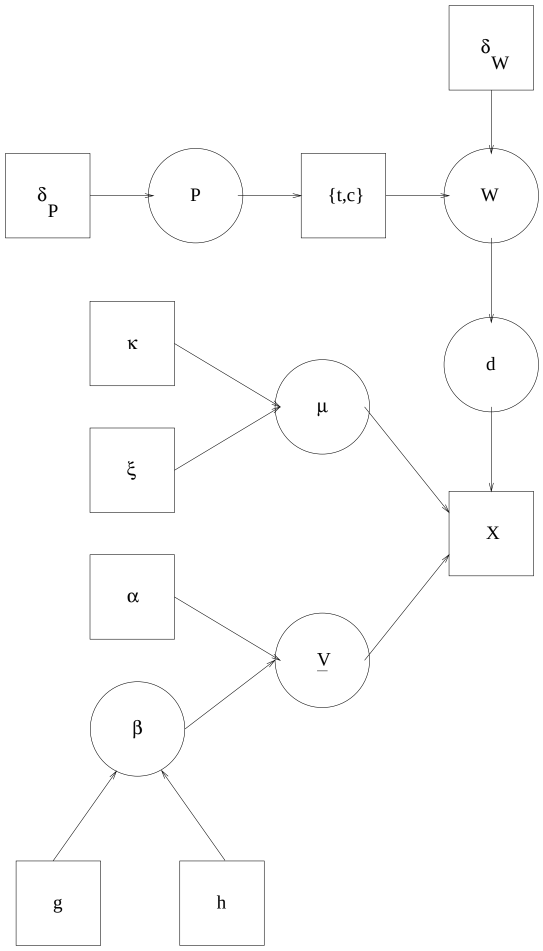

Figure 5.1: This is the directed acyclic graph of the model including all hyper parameters. Latent (unobserved) variables are shown as circles, observed variables are shown as squares. The latent allocation variable d makes the observations conditionally independent from the allocation probabilities. This "trick" allows Gibbs sampling. Without this variable, we would have to use Metropolis-Hastings-like sampling for the entire model.

Hastings samplers. In order to allow sampling from the full conditional, we have to choose priors over coefficients from their *conjugate family*:

- Each component mean, $\underline{m}_d$, is given a Gaussian prior: $\underline{m}_d \sim \mathcal{N}_d(\underline{\xi}, \underline{\kappa})$.
- The inverse variance of input i and kernel d gets a Gamma prior: $\sigma_{id}^{-2} \sim \Gamma(\alpha, \beta_i)$.
- All d variances of input i have a common hyperparameter, β_i, that has itself a Gamma hyperprior: $\beta_i \sim \Gamma(g, h_i)$.
- The mixing coefficients, $\underline{w}_k$, get a Dirichlet prior: $\underline{w}_k \sim \mathcal{D}(\delta_w, ..., \delta_w)$.
- Class priors, $\underline{P}$, also get a Dirichlet prior: $\underline{P} \sim \mathcal{D}(\delta_P, ..., \delta_P)$.

The quantitative settings are similar to those used in [RG97]: Values for α are between 1 and 2, g is usually between 0.2 and 1 and h_i is typically between $1/R_i^2$ and $10/R_i^2$, with R_i denoting the i'th input range. The mean gets a Gaussian prior centered at the midpoint, $\underline{\xi}$, with diagonal inverse covariance matrix $\underline{\kappa}$, with $\kappa_{ii} = 1/R_i^2$. The prior counts δ_w and δ_P are set to 1 to give the corresponding probabilities non-informative proper Dirichlet priors.

The Gibbs sampler uses updates from the full conditional distributions in (5.3). For notational convenience we use $\underline{\Theta}_k$ for the parameters that determine class conditional densities. We use m as index over unlabeled data and c_m as latent class label. The index for all data is n, d_n are the latent kernel allocations and n_d is the number of samples allocated by the d-th component. One distribution does not occur in the prior specification, namely $\mathcal{M}n(1, ...)$ which is a multinomial - 1 distribution. Finally we need some counters: m_1 ... m_K are the counts per class and m_{1k} .. m_{Dk} count kernel allocations of class-k patterns. The full conditional of the d-th kernel variances and the hyper parameter β_i contain i as index of the input dimension. In these full conditionals we express each $\sigma_{i,d}^{-2}$ separately. In the expression of the d-th kernel mean, $\underline{m}_d$, we use $\underline{V}_d$ to denote the entire covariance matrix.

$$\begin{aligned}
p(c_m|...) &= \mathcal{M}n\left(1, \left\{\frac{P_k p(\underline{x}_m|\underline{\Theta}_k)}{\sum_k P_k p(\underline{x}_m|\underline{\Theta}_k)}, k = 1..K\right\}\right) \qquad (5.3)\\
p(d_n|...) &= \mathcal{M}n\left(1, \left\{\frac{w_{t_n d} p(\underline{x}_n|\underline{\Phi}_d)}{\sum_l w_{t_n d} p(\underline{x}_n|\underline{\Phi}_d)}, d = 1..D\right\}\right)\\
p(\underline{\beta}_i|...) &= \Gamma\left(g + D\alpha, h_i + \sum_d \sigma_{d,i}^{-2}\right)\\
p(\underline{w}_k|...) &= \mathcal{D}\left(\delta_w + m_{1k}, ..., \delta_w + m_{Dk}\right)\\
p(\underline{P}|...) &= \mathcal{D}\left(\delta_P + m_1, ..., \delta_P + m_K\right)\\
p(\underline{m}_d|...) &= \mathcal{N}\left((n_d \underline{V}_d^{-1} + \underline{\kappa})^{-1}(n_d \underline{V}_d^{-1}\bar{\underline{x}}_d + \underline{\kappa}\underline{\xi}), (n_d \underline{V}_d^{-1} + \underline{\kappa})^{-1}\right)\\
p(\sigma_{i,d}^{-2}|...) &= \Gamma\left(\alpha + \frac{n_d}{2}, \beta_i + \frac{1}{2}\sum_{\underline{x}_n \forall n|d_n = d} (\underline{x}_{n,i} - \underline{m}_{d,i})^2\right)
\end{aligned}$$

5.2.3 Moving between different input subsets

The core part of this sampler are reversible jump updates, where we move between different feature subsets. The probability of a feature subset will be determined by the corresponding Bayesian model evidence and by an additional prior over number of inputs. In accordance with [PS96], we use the truncated Poisson prior

$$p(I) = 1/\begin{pmatrix} I_{max} \\ I \end{pmatrix} c\frac{\lambda^I}{I!},$$

where c is a constant and I_{max} is the total number of inputs. Reversible jump updates are generalizations of conventional Metropolis-Hastings updates, where moves are bijections $(x, u) \leftrightarrow (x', u')$. For a thorough treatment we refer to [Gre95]. In order to switch subsets efficiently, we will use two different types of moves. The first move consists of a step where we add one input chosen at random and a matching step that removes one randomly chosen input. A second move exchanges two inputs which allows “tunneling” through low likelihood areas.

Adding an input, we have to increase the dimension of all kernel means and diagonal covariances. These coefficients are drawn from their priors. In addition, the move proposes new allocation probabilities in a semi-deterministic way. Assuming the ordering, $w_{k,d} \leq w_{k,d+1}$, we propose

$$\begin{aligned} \delta_p &= \text{Beta}(b_a, b_b + I) \\ \forall d \leq D/2 \qquad & \begin{cases} w'_{k,D+1-d} = w_{k,D+1-d} + w_{k,d}\delta_p \\ w'_{k,d} = w_{k,d}(1-\delta_p). \end{cases} \end{aligned} \tag{5.4}$$

The matching step proposes the removal of a randomly chosen input. Removing corresponding kernel coefficients is again combined with a semi-deterministic proposal of new allocation probabilities, which is exactly symmetric to the proposal in (5.4). We accept births with probability:

$$\begin{aligned} \alpha_b &= \min\Bigg(1, \text{lh. rt.} \times \frac{p(I+1)}{p(I)} \left(\frac{1}{R'}\sqrt{2\pi}\right)^D \prod_D \exp\left(-0.5\frac{1}{R'^2}(\mu'_d - \xi'_d)^2\right) \\ &\quad \times \left(\frac{\beta'^\alpha}{\Gamma(\alpha)}\right)^D \prod_D (\sigma'^{-2}_d)^{\alpha-1} \exp(-\beta' \sigma'^{-2}_d) \\ &\quad \times \frac{d_m/(I+1)}{b_m/(I_{max}-I)} \times \frac{1}{\left(\frac{1}{R'}\sqrt{2\pi}\right)^D \prod_D \exp\left(-0.5\frac{1}{R'^2}(\mu'_d - \xi'_d)^2\right)} \\ &\quad \times \frac{1}{\left(\frac{\beta'^\alpha}{\Gamma(\alpha)}\right)^D \prod_D (\sigma'^{-2}_d)^{\alpha-1} \exp(-\beta' \sigma'^{-2}_d)}\Bigg). \end{aligned} \tag{5.5}$$

The first line in (5.5) are the likelihood and prior ratios. The prior ratio results from the difference in input dimension, which affects the kernel means and the prior over number of inputs. The first term of the proposal ratio is from proposing to add or remove one input. The second term is the proposal density of the additional kernel components, which cancels with the corresponding term in the prior ratio. Due to symmetry of the proposal (5.4) and its reverse in a death move, there is no contribution from changing allocation probabilities. Death moves are accepted with probability $\alpha_d = 1/\alpha_b$.

The second type of move is an exchange move. We select a new input and one from the model inputs and propose new mean coefficients. This gives the following acceptance probability

$$\begin{aligned} \alpha_c &= min\Bigg(1, \text{lh. ratio} \times \frac{\left(\frac{1}{R'}\sqrt{2\pi}\right)^D \prod_D \exp\left(-0.5\frac{1}{R'^2}(\mu'_d - \xi'_d)^2\right)}{\left(\frac{1}{R'}\sqrt{2\pi}\right)^D \prod_D \exp\left(-0.5\frac{1}{R'^2}(\mu_d - \xi_d)^2\right)} \\ &\quad \times \frac{c_m/I}{c_m/(I_{max}-I)} \times \frac{\prod_D \mathcal{N}(\mu_d|...)}{\prod_D \mathcal{N}(\mu'_d|...)}\Bigg). \end{aligned} \tag{5.6}$$

Table 5.1: Summary of experiments

Data	**avg(#)**	**max(#)**	**RBF (%,n_a)**	**MLP (%,n_b)**
Ionosphere	4.3	9	(91.5,11)	(95.5,4)
Pima	4	7	(78.9,11)	(79.8,8)
Wine	4.4	8	(100, 0)	(96.8,2)

The first line of (5.6) are again likelihood and prior ratios. For exchange moves, the prior ratio is just the ratio from different values in the kernel means. The first term in the proposal ratio is from proposing the exchange of an input. The second term is the proposal density of new kernel mean components. The last part stems from proposing new allocation probabilities.

5.3 Experiments

Although the method can be used with labeled and unlabeled data, the following experiments were performed using only labeled data. For all experiments we set $\alpha = 2$ and $g = 0.2$. The first two data sets are from the UCI repository[4]. We use the Ionosphere data, which has 33 inputs, 175 training and 176 test samples. For this experiment, we use 6 kernels and set $h = 0.5$. The second data is the wine recognition data which provides 13 inputs, 62 training and 63 test samples. For this data, we use 3 kernels and set $h = 0.28$. The third experiment is performed with the Pima data provided by B. D. Ripley[5]. For this one, we use 3 kernels and set $h = 0.16$.

For all experiments we draw 15000 samples from the posterior over coefficients and input subsets. We discard the first 5000 samples as burn in and use the rest for predictions. Classification accuracy is compared with an MLP classifier using R. Neal's hybrid Monte Carlo sampling with ARD priors on inputs. These experiments use 25 hidden units. Table 5.1 contains further details: avg(#) is the average and max(#) is the maximal number of inputs used by the hybrid sampler; RBF (%, n_a) is the classification accuracy of the hybrid sampler and the number of errors it made that were not made by the ARD-MLP; MLP(%, n_b) is the same for the ARD-MLP. We compare classifiers by testing (n_a, n_b) against the null hypothesis that this is an observation from a Binomial $\mathcal{B}n(n_a + n_b, 0.5)$ distribution. This reveals

[4]Available at http://www.ics.uci.edu/ mlearn/MLRepository.html.

[5]Available at http://www.stats.ox.ac.uk

that neither difference is significant. Hence we could compete with Neal's ARD-MLP, a method that usually obtains high generalization accuracies.

The real benefit from using the hybrid sampler is that the inferred probabilities tell us which *feature subsets* contribute to an explanation of the target variables. Figure 5.3 shows the occurrence probabilities of feature subsets and features. Note that table 5.1 also shows the details about how many features were used in these problems. Especially the results from Ionosphere data are interesting since on average we use only 4.3 out of 33 input features. For ionosphere and wine data the Markov chain visits about 500 different input subsets within 10000 samples. For the Pima data the number is about 60 and an order of magnitude smaller.

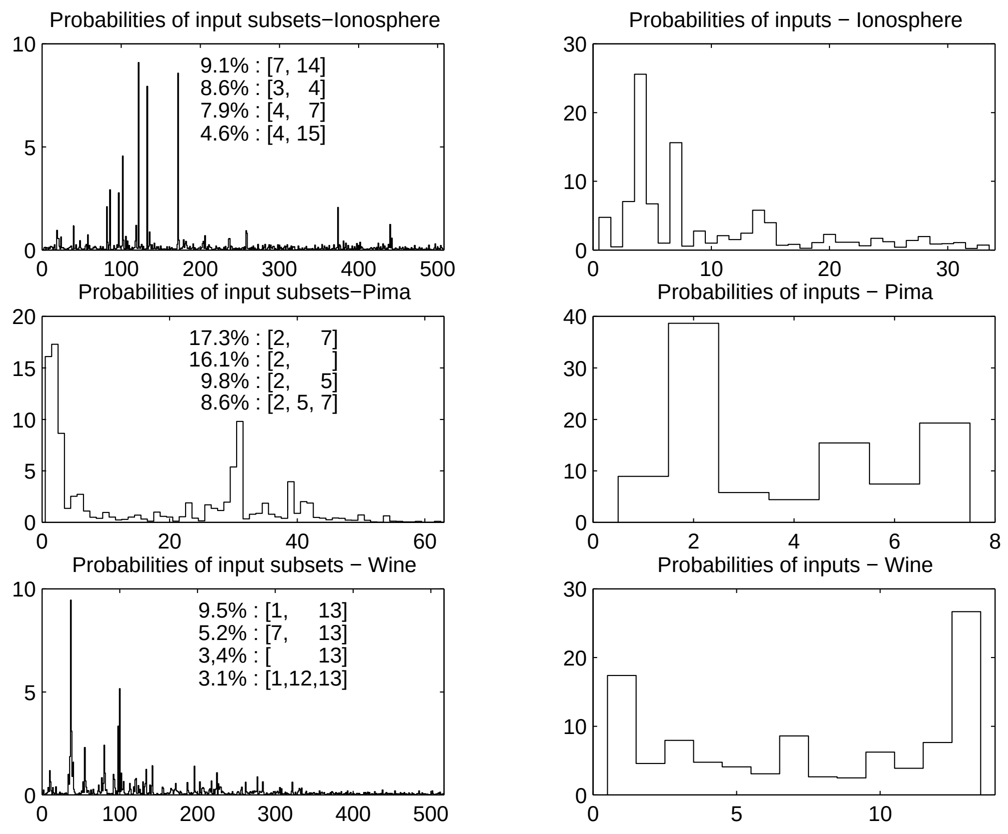

Figure 5.2: Probabilities of inputs and input subsets measuring their relevance.

In the next experiment we apply the Bayesian wrapper to the problem of sleep EEG classification that was introduced in chapter 2. We are interested in assessing the discriminative power of features obtained from different preprocesing techniques. The sleep analyzer motivated in chapter 2 will predict probabilities for the states wake, REM and delta sleep. Hence we aim at such feature subsets that are usefulfor separating these states. Therefore the data used in this experiment contains only samples that were labeled wake, REM or stage 4. We used data from 4 selected recordings.

Preprocessing was done using the following algorithms[6], providing one estimate for each second of the all night recording:

- Power spectral density and coherence function estimates calculated by using a fast Fourier transform [FRKZ97]. Both methods provide 9 estimates obtained by smoothing over 9 different frequency bands.
- Auto regressive filter (AR)-coefficients estimated via Kalman filtering. The Kalman filter was implemented for a 10-th order AR-model [SP99].
- Three Hjorth [Hjo70] complexity measures and a nonlinear combination of them.
- An embedding dimension complexity measure [RR98].
- Coefficients obtained from a 10-th order lattice filter estimated by the technique reported in chapter 4.

The difficulty of this FSS was that the algorithms used different window lengths. Together with the 30 seconds based R&K scorings, this means that features cannot be compared in the first place. Features with longer windows will be preferred. In order to avoid that longer windows are an advantage, we decided to run the FSS with the median segment of each 30 seconds epoch. Resampling to equal priors, we get 546 samples.

The Bayesian wrapper was used with data from electrode C3 only. This reduces the total number of available features to 43. After drawing 10000 samples from the posterior distribution of model coefficients and different dimensions, we discarded the first 5000 samples as burn in. The probabilities of feature subsets observed in the remaining samples are plotted in figure 5.3. The most probable feature subset found in this evaluation has a probability of $P_{mp} = 0.698$. The feature variables contained in the subset are listed in table 5.2.

[6]I would like to express gratitude to all who provided the feature estimates. The FFT based features have been calculated by P. Rappelsberger and O. Filz. The Kalman filter AR-coefficients have been provided by A. Schloegl. The Hjorth coefficients have been obtained from A. Varri and M. Koivuluoma. Finally the stochastic complexity has been provided by I. Rezek and S. Roberts.

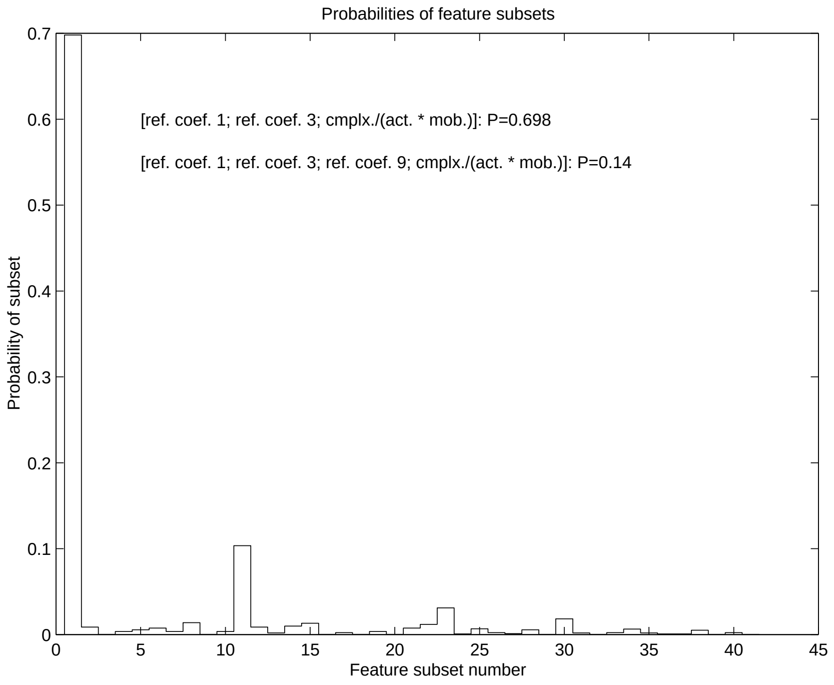

Figure 5.3: Probabilities of input subsets measuring their relevance.

Table 5.2: Most probable feature subset at electrode C3

Bayesian reflection coefficient 1
Bayesian reflection coefficient 3
Hjorth coefficient, cmplx./(act. * mob.)

As a final remark, it seems important to mention that the result suggested by the Bayesian technique is comparable to the result reported in [SRR+99]. In [SRR+99], conventional search algorithms and feature evaluation criteria have been used together with a statistical hypothesis test to determine the optimal subset. The results reported there suggest that between 2 and 3 features suffice. Both the complexity measures and the reflection coefficients are reported to be the most useful ones for classifying sleep.

5.4 Summary

In this chapter we have evaluated a hybrid sampler that uses Gibbs updates and reversible jump moves to approximate the posterior distribution over parameters and input subsets in nonlinear classification problems. The classification accuracy of the method could compete with R. Neals' MLP-ARD-implementation. However, the real advantage of the Bayesian wrapper is that it provides us with a relevance measure of feature subsets. This allows to infer the optimal number of inputs and how many different explanations the data provides.

The algorithm was applied to feature subset selection in a sleep classification task. We found the Bayesian reflection coefficients as the most useful ones. The result suggests that instead of extracting 10 coefficients it suffices to extract the first 3 coefficiets.

Chapter 6

A Bayes' inferred generative classifier

In this chapter we propose a Bayesian treatement of a generative classifier. We approximate the posterior over model parameters by an *ensemble* technique. This learning scheme, also known as *variational* approximation of the posterior, allows us to perform model selection. Variational inference provides us with an approximation of the log evidence of the model that can be used to find the posterior probabilities of different model classes. In order to assess the properties of the proposed approach, we perform experiments with various synthetic problems. Practical relevance of the method is demonstrated implementing a sleep analyzer for the problem formulated in chapter 2.

6.1 Introduction

In this chapter we propose a Bayesian learning scheme for a generative classifier. In order to allow both efficient inference and predictions, we aim at an analytic solution. That is, we try to get a parameterized representation of the posterior. The nonlinearity of the model does not permit an exact solution as would be possible for generalized linear models. Thus we are faced with two choices: we can fit the posterior by using a Laplace approximation or a variational approximation. Especially the latter has recently been used as tool for inference in graphical models (see e.g. [JGJS99] and [Fre98]) and for Bayesian inference of various models (see e.g. [Mac97], [GB00]). In particular, [Att99] proposes a variational Bayesian approach for Gaussian mixture models, which show some similarity with the generative classifier used in this chapter.

Our preference for a Bayesian treatment of a latent variable model is motivated by several requirements to the inference process and the model:

- We are interested in reliably classifying some data as belonging to one of k classes.
- During learning, we would like to be able to use unlabeled data. As long as unlabeled data was obtained from the same distribution as the labeled samples, their use will improve inference of supervised models. In many domains labeled data is expensive to get, whereas unlabeled data are obtained easily. Hence there is a considerable interest in techniques that can use unlabeled data to improve supervised learning.
- The trained model should provide some deeper insight in the way a particular classification result was achieved. That is we would like to get more detailed information than just probabilities for class labels.

The requirement of being able to use training data without labels can be approached by a generative model. The third requirement is met by generative models as well: they provide us with probabilities of the states of latent variables. This information, however, is only meaningful if we have means to choose appropriate model orders. In order to do model inference in a Bayesian setting, we extend the DAG associated with the generative classifier by adding model and hyperparameters. Due to efficiency constraints, both during inference and predictions, we use a variational approximation of the true posterior. This provides predictive distributions over model parameters and an approximative estimate of the log-evidence. Despite all motivations to use a generative model for classification, we should also remember that such models have a disadvantage: we can not use these models in situations where large input dimensions are paired with few training samples. One of the experiments is dedicated to investigate this problem. In particular we will show that if a model can not be inferred due to lack of data, the model probability will be largest for the smallest model under consideration. Furthermore we will also show that in such cases it suffices to add *unlabeled* data to improve the model.

6.2 A generative model and its likelihood function

In this chapter we will consider a variational Bayesian approach to obtain a posterior distribution over latent variables and model coefficients for classifiers that are constructed from class conditional density estimates. This approach is known as *classification in the sampling paradigm* (e.g. [Rip96]) and goes back to [Daw76]. The classifier discussed here will use a semi-parametric (mixture) representations of the class conditional densities. The architecture is almost identical to the classifier that was proposed in chapter 5. As we will see, the only difference is that we allow for one diagonal covariance matrix for each Gaussian.

Recently a rising interest in generative models can be observed - even if classification is the only interest. This interest is mainly motivated by the probabilistic nature of the models. A directed acyclic graph (DAG) that corresponds to this type of classifier is shown in figure 6.1. This DAG illustrates the probabilistic dependencies among target labels, k, the latent kernel indicator[1], d, and input variables denoted as x. Furthermore, all model parameters are linked probabilistically with these variables. These are the prior probabilities for a class, P, the class conditional allocation probabilities, W, and the kernel coefficients, μ and σ. Last but not least the probability densities over model coefficients depend on several prior probabilities. We use δ_P and δ_W as prior counts in the Dirichlet distributions over P and W. Both ξ and κ determine the Gaussian prior over μ. The Gamma prior over σ is determined by the parameters α and β. For β we used a hierarchical specification. In order to avoid that kernels collapse[2], we must use *informative* priors that avoid zero variances. In such cases we better uses a hierarchical setting which allows for some influence of the data on the value of such hyperparameters. This is a trick adopted from [RG97]. Consequently, we need two parameters, g and h, that define this hyper-prior over β.

Compared with non-probabilistic models, the generative classifier has two advantages:

- Generative models give us the possibility to solve missing data problems: we may infer the conditional probability density over missing inputs during training and predictions and the conditional probability of missing target labels during training. Both types of problems occur frequently in practical settings.

- The model provides deeper insight into the problem structure: We do not only get probabilities for classes, but probabilities for generating kernels as well. With appropriate inference procedures such as the

[1] The kernel indicator is an unobserved variable that tells us which kernel generated the variable x.

[2] A problem often encountered with mixture density models, as the likelihood approaches ∞ when one of the kernel means is equivalent to a data point and the variance is set to 0.

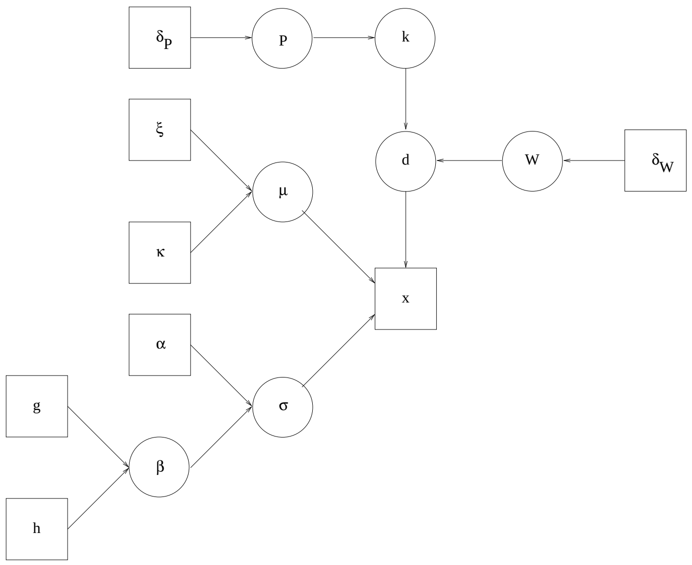

Figure 6.1: A directed acyclic graph (DAG) that shows the generative classifier used in this chapter. The DAG illustrates the probabilistic relations between latent variables, observed variables and model coefficients. All observed variables are drawn as squares whereas all latent quantities are represented with circles. This DAG anticipates that model inference is also carried out in a probabilistic setting.

variational Bayesian scheme proposed in this chapter, we can guarantee that the model order (number of kernels) is chosen appropriately. This seems imperative to give these kernel probabilities some deeper meaning. By using the appropriate model complexity we can avoid that two identical kernels lead to an identifiability problem. (see chapter 5 for details of the identifiability problem)

Of course we have to pay a price for these advantages: As is pointed out in [Rip96], modeling class conditional densities is much more difficult than modeling posterior probabilities directly[3].

Using mixture models as density estimators of class conditional densities, [Trå91] proposes two different models that can be used to build the classifier. One model assumes that the data are hard partitioned: Each mixture component is entirely allocated to one class. The other model proposed there is more appropriate for our purpouse as it allows for overlapping class

[3]As is for example mentioned in [Vap95], modeling a-posteriori probabilities is in turn more complex then modeling class labels. However, having the class labels only, we do not know how reliable they are.

conditional densities

$$\begin{aligned} p(\underline{x}|k) &= \sum_{d=1}^{D} w_{kd} p(\underline{x}|\underline{\Theta}_d) \\ p(\underline{x}) &= \sum_{k=1}^{K} P_k p(\underline{x}|k). \end{aligned} \quad (6.1)$$

Using P_k for the K class priors and $p(\underline{x}|k)$ for the class conditional densities, we may express the posterior probabilities for classes as $P(k|\underline{x}) = P_k p(\underline{x}|k)/p(\underline{x})$. The D component densities $p(\underline{x}|\underline{\Theta}_d)$ can be any parameterized density function. For sake of convenience we will use normal densities with diagonal covariance matrices.

In order to parameterize the model, we will use the probability density form of the likelihood function. According to the notation introduced in (6.1), we use t to denote given class labels and $\underline{x}$ to denote inputs. Using $\underline{\varphi}$ as model coefficients, the likelihood observing labeled data $\mathcal{T}$ and unlabeled data $\mathcal{X}$ is

$$p(\mathcal{T}, \mathcal{X}|\underline{\varphi}) = \prod_{m=1}^{M} p(t_m, \underline{x}_m) \prod_{n=1}^{N} p(\underline{x}_n). \quad (6.2)$$

In order to allow inference, we have to link (6.2) with our model (6.1) to get

$$\begin{aligned} p(\mathcal{T}, \mathcal{X}|\underline{\varphi}) &= \prod_{m=1}^{M} \left(P(t_m) \sum_{d=1}^{D} w_{d,t_m} p(\underline{x}_m|d) \right) \\ & \prod_{n=1}^{N} \left(\sum_{k=1}^{K} P(k) \sum_{d=1}^{D} w_{d,k} p(\underline{x}_n|d) \right) \end{aligned} \quad (6.3)$$

as the corresponding expression for the likelihood.

6.3 Variational approximation of the posterior

In this section we derive a simple variational approximation to the posterior over latent variables and model coefficients for the classifier introduced above. We use here an approximation of the posterior by a *mean field* representation. This approach has been used for a Bayesian treatment of various models. The method is also known as *ensemble learning*. Such an approach was used in [Mac97] for inference of a hidden Markov model with a discrete observation sequence. Recently [GB00] proposed such a mean field approximation for a Bayesian treatement of a mixture of factor analyzers.

The key idea of variational techniques is to fit the approximating ensemble to the Bayesian posterior by maximizing a lower bound of the logarithmic partition function introduced by $\int_{\underline{\varphi}} \log(p(\underline{\varphi})p(\mathcal{T}, \mathcal{X}|\underline{\varphi}))d\underline{\varphi}$. In a Bayesian understanding the partition function corresponds to the normalization constant or *model evidence*.

The variational Bayesian approach as applied here to the posterior over model coefficients and latent variables will use the following simplifying assumptions:

- The posterior is modeled by a mean field expansion. That is:

$$p(\underline{\varphi}|\mathcal{T}, \mathcal{X}) \approx \prod_{\forall \underline{\varphi}_l} Q(\underline{\varphi}_l),$$

  where $\underline{\varphi}_l$ are all parameters that are modeled by the same approximating distribution.

- The logarithm of the partition function is bounded below by the following type of functional:

$$F(\underline{\varphi}) = \int_{\underline{\varphi}} \log(\frac{p(\underline{\varphi})p(\mathcal{T}, \mathcal{X}|\underline{\varphi})}{Q(\underline{\varphi})})Q(\underline{\varphi})d\underline{\varphi}, \tag{6.4}$$

  where due to the mean field assumption, $Q(\underline{\varphi})$ will have a factorial representation. It should be mentioned that maximizing (6.4) is equivalent to minimizing the *variational free energy*, $-F(\underline{\varphi})$, which is a well-known tool in statistical physics (see e.g. [Fey72]). By using Jensens inequality it is easy to show (see e.g. [JGJS99]) that (6.4) is indeed a lower bound of the log evidence.

- During maximization of the functional (6.4), we will face the problem that the log of the marginalization operation[4] over latent variables is not analytically tractable. Hence we will also bound this arithmetic mean below by a geometric mean. This kind of lower bound is suggested by [JJ00]:

$$\sum_{d_m=1}^{D} w_{d,t_m} p(\underline{x}_m|d_m) \leq \prod_{d_m=1}^{D} \left(\frac{w_{d_m,t_m} p(\underline{x}_m|d_m)}{Q(d_m)}\right)^{Q(d_m)}, \tag{6.5}$$

[4]This marginalization operation is part of the likelihood function (6.3).

where, apart from a minor difference, we have used the notation introduced in (6.1). The difference is that we must fit one such $Q(d_m)$ distribution for each training pattern. The same type of lower bound will be necessary *twice* for all unlabeled data[5].

Using these ideas, we are able to formulate a functional over Q-distributions that has to be maximized to obtain a variational approximation of the posterior over model coefficients and *latent* variables:

$$\begin{aligned}
\mathcal{F}(Q) &= \int_{\mu,\sigma,P,W,\beta} Q(\mu)Q(\sigma)Q(P)Q(W)Q(\beta) \qquad (6.6)\\
& \Bigg(\log\left[\frac{P(\mu)P(\sigma)P(P)P(W)P(\beta)}{Q(\mu)Q(\sigma)Q(P)Q(W)Q(\beta)}\right]\\
&+ \sum_n \Bigg(\log(P(t_n)) + \sum_{d_n}\Bigg(Q(d_n)\Bigg[\log(W_{t_n,d_n}) - \frac{I}{2}\log(2\pi)\\
&- 0.5\sum_i \log(\sigma^2_{d_n,i}) - 0.5(\underline{x}_n - \underline{\mu}_{d_n})^T \underline{\Sigma}_{d_n}^{-1}(\underline{x}_n - \underline{\mu}_{d_n}) - \log(Q(d_n))\Bigg]\Bigg)\Bigg)\\
&+ \sum_m\sum_{k_m} Q(k_m)\Bigg[\log(P_{k_m}) + \sum_{d_m}\Bigg(Q(d_m)\Bigg(\log(W_{k_m,d_m}) - \frac{I}{2}\log(2\pi)\\
&- 0.5\sum_i \log(\sigma^2_{d_m,i}) - 0.5(\underline{x}_m - \underline{\mu}_{d_m})^T \underline{\Sigma}_{d_m}^{-1}(\underline{x}_m - \underline{\mu}_{d_m})\\
&- \log(Q(d_m))\Bigg)\Bigg) - \log(Q(k_m))\Bigg]\Bigg) d\mu d\sigma d\beta dP dW
\end{aligned}$$

In (6.6) we largely use the same symbols as introduced above. In addition, I, denotes the input dimension, and i serves as iterator over different inputs. Before we maximize the functional (6.6) w.r.t the different Q-functions, it seems appropriate to give two simple qualitative arguments that such iterative procedure will indeed lead to the desired approximation of the posterior: On one hand, it is easy to see that a (local) maximum of (6.6) corresponds to a minimum of the Kullback-Leibler (KL) divergence between the product of Q-distributions and the true posterior (see e.g. [JGJS99]); on the other hand, the results obtained for the EM-algorithm, in their most general form provided by [NH99], guarantee that maximizing (6.6) iteratively w.r.t. each Q-function in turn will indeed reach such a (local) maximum. Hence we may optimize (6.6) for each Q-function separately and formulate an algorithm that will iterate over these maximization steps.

[5]We have to bound both the marginal operation over latent kernel indicators and unknown class labels.

6.4 Prior specification

The model and the priors that will be introduced below are almost identical with chapter 5. The only difference is that the DAG in figure 6.1 allows for one covariance matrix for each Gaussian component, whereas in chapter 5 we used a common covariance matrix for all kernels. Apart from model coefficients the DAG in figure 6.1 also shows nodes that specify the parameter priors. In this section, we will have a closer look at these priors. We would like that our variational approximation leads to such Q-functions that have known functional form. In order to obtain these simple Q-functions (in terms of *known* distributions), we have to choose so called *conjugate* priors, as are discussed in [BS94]:

- Each component mean, $\mu_{d,i}$, is given a Gaussian prior: $\mu_{d,i} \sim \mathcal{N}_1(\xi_i, \kappa_{ii}^{-1})$.
- The inverse variances are given a Gamma prior: $\sigma_{d,i}^{-2} \sim \Gamma(\alpha, \beta_i)$.
- The hyper-parameter, β_i, gets a Gamma hyper-prior: $\beta_i \sim \Gamma(g, h_i)$.
- The class conditional mixing coefficients, $\underline{W}_k$, get a Dirichlet prior: $\underline{W}_k \sim \mathcal{D}(\delta_W, ..., \delta_W)$.
- Class priors, $\underline{P}$, also get a Dirichlet prior: $\underline{P} \sim \mathcal{D}(\delta_P, ..., \delta_P)$.

The quantitative settings are similar to those used in [RG97]: Values for α are between 1 and 2, g is usually between 0.2 and 1 and h_i is typically between $1/R_i^2$ and $10/R_i^2$, with R_i denoting the i-th dimensional input range. The mean, μ_i, gets a Gaussian prior centered at the midpoint, ξ_i, with inverse variance $\kappa_{ii} = 1/R_i^2$. Both the prior counts δ_P and δ_W are set to 1 to give the corresponding probabilities the most uninformative proper Dirichlet prior.

6.5 Maximizing with respect to Q-functions

After having formulated conjugate priors, maximization of the functional (6.6) will lead to Q-functions with identical functional form as have the corresponding priors. In this section, we will consider maximizing each of these Q-functions in turn. This has to be done by taking expectations with respect to all other Q-distributions. The resulting expressions can be used in an algorithm that will iteratively maximize (6.6) until a local maximum has been reached. Measured in terms of a KL-distance, the Q-functions obtained by this algorithm give the factorized distribution closest to the true posterior.

Maximizing with respect to $Q(d_n)$

Dropping all additive constants that have no effect on $Q(d_n)$, functional (6.6) leads to

$$\mathcal{F}(Q(d_n)) = \int_{\mu,\sigma,W} Q(\mu)Q(\sigma)Q(W) \sum_{d_n} Q(d_n)\Big(\log(W_{t_n,d_n}) - \frac{I}{2}\log(2\pi)$$
$$-0.5\sum_i \left(\log(\sigma^2_{d_n,i}) + (x_{n,i}-\mu_{d_n,i})^2\sigma^{-2}_{d_n,i}\right) - \log(Q(d_n))\Big) d\mu d\sigma dW. \quad (6.7)$$

Maximizing the functional (6.7) with respect to $Q(d_n)$ is now a purely technical problem: First we have to take the expectations w.r.t each Q-functions involved. We will consider each integral in turn and start by integrating out $Q(W)$

$$\begin{aligned} & \int_{W_{t_n}} \log(W_{t_n,d_n}) Q(W_{t_n}) dW_{t_n} && (6.8)\\ = & \int_{W_{t_n,d_n}=0}^{1} \Bigg(\log(W_{t_n,d_n}) \frac{\Gamma(\sum_{d=1}^{D} c^W_{t_n,d})}{\Gamma(c^W_{t_n,d_n})\Gamma(\sum_{d\neq d_n} c^W_{t_n,d})} \\ & W_{t_n,d_n}^{(c^W_{t_n,d_n}-1)}(1-W_{t_n,d_n})^{(-1+\sum_{d\neq d_n} c^W_{t_n,d})}\Bigg) dW_{t_n,d_n} \\ = & \Psi(c^W_{t_n,d_n}) - \Psi(\sum_{d=1}^{D} c^W_{t_n,d}), \end{aligned}$$

which gives as a result an expression in terms of the posterior class conditional allocation counts $c^W_{t_n,d}$ and the Digamma function Ψ. The expectation $< \log(\sigma^2_{d_n,i}) >_{Q(\sigma^{-2}_{d_n,i})}$ gives

$$\int_0^\infty \log(\sigma^2_{d_n,i}) Q(\sigma^{-2}_{d_n,i}) d\sigma^{-2}_{d_n,i} = \log(\beta_{d_n,i}) - \Psi(\alpha_{d_n,i}), \quad (6.9)$$

where $Q(\sigma^{-2}_{d_n,i}) = \Gamma(\alpha_{d_n,i}, \beta_{d_n,i})$, and Ψ again denotes the Digamma function. Finally, we have as the input data dependent part

$$\begin{aligned} & \int_0^\infty Q(\sigma^{-2}_{d_n,i}) d\sigma^{-2}_{d_n,i} \int_{-\infty}^{\infty} Q(\mu_{d_n,i}) d\mu_{d_n,i} && (6.10)\\ - & \ 0.5\sum_i \left((x_{n,i}-\mu_{d_n,i})^2 \sigma^{-2}_{d_n,i}\right) \\ = & \ (x_{n,i}-\hat{\mu}_{d_n,i})^2 + \sigma^2_{\mu_{d_n,i}} \frac{\alpha_{d_n,i}}{\beta_{d_n,i}}, \end{aligned}$$

where $\hat{\mu}_{d_n,i}$ denotes the most probable value of the i-th dimensional component of the d_n-th kernel mean, $\sigma^2_{\mu_{d_n,i}}$ is the corresponding variance and $\alpha_{d_n,i}$ as well as $\beta_{d_n,i}$ are again the parameters of the Q-function over the corresponding inverse kernel variance, $\sigma^{-2}_{d_n,i}$. The maximizing Q-function $Q(d_n)$ is given by:

$$Q(d_n) = \frac{\exp(\log(Q(x_n, d_n)))}{\sum_{d_n} \exp(\log(Q(x_n, d_n)))} \tag{6.11}$$

with

$$\begin{aligned} \log(Q(x_n, d_n)) &= \Psi(c^W_{t_n,d_n}) - \Psi(\sum_{d=1}^{D} c^W_{t_n,d}) - \frac{I}{2}\log(2\pi) \\ - \quad & 0.5 \sum_i \left(\log(\beta_{d_n,i}) - \Psi(\alpha_{d_n,i}) + (x_{n,i} - \hat{\mu}_{d_n,i})^2 + \sigma^2_{\mu_{d_n,i}} \frac{\alpha_{d_n,i}}{\beta_{d_n,i}} \right) \end{aligned}$$

Maximizing with respect to $Q(d_m)$

Maximizing (6.6) with respect to $Q(d_m)$ is done in a similar way as with respect to $Q(d_n)$. First we drop all additive constants and rearrange summation to get

$$\begin{aligned} \mathcal{F}(Q(d_m)) &= \int_{\mu,\sigma,W} Q(\mu)Q(\sigma)Q(W) \sum_{d_m} Q(d_m) \Bigg(\sum_{k_m} Q(k_m) \Bigg(\log(W_{k_m,d_m}) \\ &- \frac{I}{2}\log(2\pi) - 0.5 \sum_i \left(\log(\sigma^2_{d_m,i}) + (x_{m,i} - \mu_{d_m,i})^2 \sigma^{-2}_{d_m,i} \right) \\ &- \log(Q(k_m)) \Bigg) - \log(Q(d_m)) \Bigg) d\mu d\sigma dW, \end{aligned} \tag{6.12}$$

with $Q(k_m)$ approximating the posterior probabilities for class k_m for all unlabeled training data. Maximizing (6.12) with respect to $Q(d_m)$ requires solving similar integrals as was the case during maximization of (6.7). Hence we may use the results obtained above to get the maximizing Q-function $Q(d_m)$ as

$$Q(d_m) = \frac{\exp(\log(Q(x_m, d_m)))}{\sum_{d_m} \exp(\log(Q(x_m, d_m)))} \tag{6.13}$$

with

$$\begin{aligned} \log(Q(x_m, d_m)) &= \sum_{k_m} Q(k_m) \left(\Psi(c^W_{k_m,d_m}) - \Psi(\sum_{d=1}^{D} c^W_{k_m,d}) \right) - \frac{I}{2}\log(2\pi) \\ -0.5 \sum_i & \left(\log(\beta_{d_m,i}) - \Psi(\alpha_{d_m,i}) + (x_{m,i} - \hat{\mu}_{dm,i})^2 + \sigma^2_{\mu_{dm,i}} \frac{\alpha_{dm,i}}{\beta_{d_m,i}} \right). \end{aligned}$$

Maximizing with respect to $Q(k_m)$

In oder to maximize (6.6) with respect to $Q(k_m)$, we again drop all additive constants to get the simplified relations

$$\begin{aligned}\mathcal{F}(Q(k_m)) &= \int_{W,P} Q(W)Q(P)\sum_{k_m} Q(k_m)\Big(\log(P_{k_m}) \\ &+ \sum_{d_m} Q(d_m)\log(W_{k_m,d_m}) - \log(Q(k_m))\Big)dWdP.\end{aligned} \tag{6.14}$$

Maximizing the functional (6.14) involves integrating out prior probabilities for classes and the class conditional allocation probabilities. The maximizing Q-function is obtained as

$$\begin{aligned}Q(k_m) &= \exp\Big(\Psi(c^P_{k_m}) - \Psi(\sum_k c^P_k) \\ &\quad +\sum_{d_m} Q(d_m)\Big(\Psi(c^W_{k_m,d_m}) - \Psi(\sum_k c^W_{k,d_m})\Big)\Big) \\ / &\quad \sum_{k_m}\exp\Big(\Psi(c^P_{k_m}) - \Psi(\sum_k c^P_k) \\ &\quad +\sum_{d_m} Q(d_m)\Big(\Psi(c^W_{k_m,d_m}) - \Psi(\sum_k c^W_{k,d_m})\Big)\Big).\end{aligned} \tag{6.15}$$

Maximizing with respect to $Q(\underline{\mu}_d)$

Restricting the kernel parameters to diagonal covariance matrices, the Q-distribution over kernel mean coefficients factorizes into $D\cdot I$ univariate Gaussians. For notational reasons it is more convenient to maximize (6.6) with respect to $Q(\underline{\mu}_d)$, the Q-distribution over the d-th kernel mean. Ignoring all additive constants, we get the functional

$$\begin{aligned}\mathcal{F}(Q(\underline{\mu}_d)) &= \int_{\underline{\Sigma}_d^{-1}} Q(\underline{\Sigma}_d^{-1})Q(\underline{\mu}_d)\Big[-\frac{I}{2}\log(2\pi) \qquad (6.16)\\ &- 0.5\sum_i \log(\kappa_{i,i}^{-1})\\ &\quad -0.5(\underline{\mu}_d - \underline{\xi})^T\underline{\kappa}^{-1}(\underline{\mu}_d - \underline{\xi})\\ &+ \sum_n Q(d_n)(\underline{x}_n - \underline{\mu}_d)^T\underline{\Sigma}_d^{-1}(\underline{x}_n - \underline{\mu}_d)\\ &+ \sum_m Q(d_m)(\underline{x}_m - \underline{\mu}_d)^T\underline{\Sigma}_d^{-1}(\underline{x}_m - \underline{\mu}_d)\\ &- \log(Q(\underline{\mu}_d))\Big] d\underline{\Sigma}_d^{-1} d\underline{\mu}_d.\end{aligned}$$

The expression $-0.5\sum_i \log(\kappa_{i,i}^{-1}) - 0.5(\underline{\mu}_d - \underline{\xi})^T\underline{\kappa}^{-1}(\underline{\mu}_d - \underline{\xi})$ is the contribution from the Gaussian prior over the kernel mean $\underline{\mu}_d$. In order to obtain the $Q(\underline{\mu}_d)$ that maximizes the functional (6.16), we have to integrate out the inverse kernel variance $\underline{\Sigma}_d^{-1}$. Thus we have to replace the inverse kernel covariance matrix $\underline{\Sigma}_d^{-1}$ by its expectation $< \underline{\Sigma}_d^{-1} >_{Q(\underline{\Sigma}_d^{-1})}$. We must consider an EM-like marginalization with respect to $Q(d_n)$ for all labeled data and with respect to $Q(d_m)$ for all unlabeled data. Some manipulations reveal that the optimal $Q(\underline{\mu}_d)$ is Gaussian:

$$\begin{aligned}Q(\underline{\mu}_d) &= \mathcal{N}(\hat{\underline{\mu}}_d, \underline{\Sigma}_{\mu_d}^{-1}) \qquad (6.17)\\ &\quad \text{with}\\ \underline{\Sigma}_{\mu_d}^{-1} &= \underline{\kappa}^{-1} + \left(\sum_n Q(d_n = d) + \sum_m Q(d_m = d)\right) < \underline{\Sigma}_d^{-1} >_{Q(\underline{\Sigma}_d^{-2})}\\ &\quad \text{and}\\ \hat{\underline{\mu}}_d^T &= \Bigg(\underline{\xi}^T\underline{\kappa}^{-1} + \left(\sum_n Q(d_n = d)\underline{x}_n^T + \sum_m Q(d_m = d)\underline{x}_m^T\right)\\ &\quad < \underline{\Sigma}_d^{-1} >_{Q(\underline{\Sigma}_d^{-2})}\Bigg)\underline{\Sigma}_{\mu_d},\end{aligned}$$

where $\hat{\underline{\mu}}_d$ is the best estimate for the d-th kernel mean and $\underline{\Sigma}_{\mu_d}$ is the diagonal covariance matrix of the Gaussian distribution. At this point it might be useful to mention that $< \underline{\Sigma}_d^{-1} >_{Q(\underline{\Sigma}_d^{-1})} = \mathrm{diag}(\{\alpha_d/\beta_{d,i}, \forall i\})$.

Maximizing with respect to $Q(\underline{\Sigma}_d^{-1})$

Assuming diagonal kernel covariance matrices, maximization of (6.6) with respect to $Q(\underline{\Sigma}_d^{-1})$ is best expressed for the i-th inverse variance, $\sigma_{d,i}^{-2}$. Again, several additive constants in (6.6) can be dropped. We finally get a functional that depends on $Q(\beta_i)$, $Q(\sigma_{d,i}^{-2})$ and $Q(\mu_{d,i})$:

$$\begin{aligned}
\mathcal{F}(Q(\sigma_{d,i}^{-2})) &= \int_{\beta_i,\mu_{d,i},\sigma_{d,i}^{-2}} Q(\beta_i)Q(\sigma_{d,i}^{-2})Q(\mu_{d,i})\Bigg(\log(p(\sigma_{d,i}^{-2})) \qquad (6.18)\\
&- 0.5\sum_n Q(d_n = d)\Big[\log(\sigma_{d,i}^2)\sigma_{d,i}^{-2}\left((x_{n,i}-\mu_{d,i})^2\right)\Big]\\
&- 0.5\sum_m Q(d_m = d)\Big[\log(\sigma_{d,i}^2)\sigma_{d,i}^{-2}\left((x_{m,i}-\mu_{d,i})^2\right)\Big]\\
&- \log(Q(\sigma_{d,i}^{-2}))\Bigg)d\beta_i d\mu_{d,i} d\sigma_{d,i}^{-2},\\
\text{with}\\
\log(p(\sigma_{d,i}^{-2})) &= \text{const.} + (\alpha_i - 1)\log(\sigma_{d,i}^{-2}) - \sigma_{d,i}^{-2}\beta_i.
\end{aligned}$$

In order to be able to maximize (6.18) with respect to $Q(\sigma_{d,i}^{-2})$, we have to marginalize out both β_i and $\mu_{d,i}$. The Q-distribution $Q(\beta_i)$ is a Gamma distribution with parameters g_i^q and h_i^q. Therefore, the expected value of β_i is $< \beta_i >_{Q(\beta_i)} = \frac{g_i^q}{h_i^q}$. Furthermore we need the expectation of $\mu_{d,i}$ and $\mu_{d,i}^2$ with respect to a normal distribution. We get $< \mu_{d,i} >_{Q(\mu_{d,i})} = \hat{\mu}_{d,i}$ and $< \mu_{d,i}^2 >_{Q(\mu_{d,i})} = \hat{\mu}_{d,i}^2 + \sigma_{\mu_{d,i}}^2$. This allows to formulate the functional that we need to maximize with respect to $Q(\sigma_{d,i}^{-2}))$ as

$$\begin{aligned}
\mathcal{F}(Q(\sigma_{d,i}^{-2})) &= \int_{\sigma_{d,i}^{-2}} Q(\sigma_{d,i}^{-2})\Bigg((\alpha_i - 1)\log(\sigma_{d,i}^{-2}) - \sigma_{d,i}^{-2}\frac{g_i^q}{h_i^q} \qquad (6.19)\\
&- 0.5\sum_n Q(d_n = d)\Big[\log(\sigma_{d,i}^2)\sigma_{d,i}^{-2}\left((x_{n,i}-\hat{\mu}_{d,i})^2 + \sigma_{\mu_{d,i}}^2\right)\Big]\\
&- 0.5\sum_m Q(d_m = d)\Big[\log(\sigma_{d,i}^2)\sigma_{d,i}^{-2}\left((x_{m,i}-\hat{\mu}_{d,i})^2 + \sigma_{\mu_{d,i}}^2\right)\Big]\\
&- \log(Q(\sigma_{d,i}^{-2}))\Bigg)d\sigma_{d,i}^{-2}.
\end{aligned}$$

We identify the expression within the brackets as a logarithm of a Gamma distribution, $\Gamma(\alpha_{\sigma_{d,i}}, \beta_{\sigma_{d,i}})$, with

$$\begin{aligned} \alpha_{\sigma_{d,i}} &= \alpha_i + 0.5 \left(\sum_n Q(d_n = d) + \sum_m Q(d_m = d) \right) \\ \beta_{\sigma_{d,i}} &= \frac{g_i^q}{h_i^q} + 0.5 \Bigg(\sum_n Q(d_n = d) \left((x_{n,i} - \hat{\mu}_{d,i})^2 + \sigma^2_{\mu_{d,i}} \right) \\ &\quad + \sum_n Q(d_m = d) \left((x_{m,i} - \hat{\mu}_{d,i})^2 + \sigma^2_{\mu_{d,i}} \right) \Bigg). \end{aligned} \tag{6.20}$$

Maximizing with respect to $Q(\beta_i)$

In order to make the Gamma prior over inverse kernel variance less informative, we use a hierarchical setting. This hierarchical setting requires also to determine a posterior over the hyper parameter β. This is again easier to express in one dimension for β_i. As usual, we drop all irrelevant additive constants from (6.6) to obtain

$$\begin{aligned} \mathcal{F}(Q(\beta_i)) &= \int_{\beta_i, \sigma_{d,i}^{-2}} Q(\beta_i) Q(\sigma_{d,i}^{-2}) \Bigg((g_i - 1) \log(\beta_i) - h_i \beta_i \\ &+ \sum_d \left(\alpha_i \log(\beta_i) - \beta_i \sigma_{d,i}^{-2} \right) - \log(Q(\beta_i)) \Bigg) d\beta_i d\sigma_{d,i}^{-2} \end{aligned} \tag{6.21}$$

as the functional to be maximized with respect to $Q(\beta_i)$. As was already obtained above for β_i, the expectation of $\sigma_{d,i}^{-2}$ with respect to the Gamma distribution $Q(\sigma_{d,i}^{-2})$ is $\frac{\alpha_{\sigma_{d,i}}}{\beta_{\sigma_{d,i}}}$. Plugging this expectation into (6.21), we immediately recognize that the maximizing Q-distribution is again a Gamma distribution:

$$\begin{aligned} Q(\beta_i) &= \Gamma(g_i^q, h_i^q) \\ \text{with} & \\ g_i^q &= g_i + D\alpha_i \\ h_i^q &= h_i + \sum_d \frac{\alpha_{\sigma_{d,i}}}{\beta_{\sigma_{d,i}}}. \end{aligned} \tag{6.22}$$

Maximizing with respect to $Q(\underline{P})$

Maximizing (6.6) with respect to $Q(\underline{P})$ leads to the following expression:

$$\begin{aligned} Q(\underline{P}) &= \int_{\underline{P}} Q(\underline{P}) \Big[\sum_k \log(P_k) \Big((\delta_p - 1) + n_k + \sum_m Q(k_m = k) \Big) \\ &\quad - \log(Q(\underline{P})) d\underline{P}. \end{aligned} \tag{6.23}$$

This is the logarithm of a Dirichlet distribution. That is, the optimal Q-function is Dirichlet:

$$\begin{aligned} Q(\underline{P}) &= \mathcal{D}(\alpha_1^P, ..., \alpha_K^P) \\ \text{with} & \\ \alpha_k^P &= \delta_p + n_k + \sum_m Q(k_m = k). \end{aligned} \tag{6.24}$$

Maximizing with respect to $Q(\underline{W})$

Maximizing (6.6) with respect to $Q(\underline{W})$ must be done for each class conditional allocation parameter $\underline{W}_k$ separately

$$\begin{aligned} Q(\underline{W}_k) &= \int_{\underline{W}_k} Q(\underline{W}_k) \Big[\sum_d \log(W_{k,d})(\delta_W - 1) \\ &\quad + \sum_n \sum_{d_n} \log(W_{k,d_n}) \delta(t_n = k) Q(d_n = d) \\ &\quad + \sum_m \sum_{d_m} \log(W_{k,d_m}) Q(k_m = k) Q(d_m = d) \\ &\quad - \log(Q(\underline{W}_k)) \Big] d\underline{W}_k. \end{aligned} \tag{6.25}$$

Hence, the optimal Q-function, $Q(\underline{W}_k)$, is again a Dirichlet distribution:

$$\begin{aligned} Q(\underline{W}_k) &= \mathcal{D}(\alpha_{k,1}^W, ..., \alpha_{k,D}^W) \\ \text{with} & \\ \alpha_{k,d}^W &= \delta_W + \sum_n \delta(t_n = k) Q(d_n = d) + \sum_m Q(k_m = k) Q(d_m = d), \end{aligned} \tag{6.26}$$

where $\delta(t_n = k)$ denotes the Kronecker delta.

With (6.25) we have completed all Q-functions. As already mentioned in section 6.3, we find the optimal Q-distributions by iteratively maximizing each of the Q-functions in turn.

6.6 The approximate log evidence

In order to be able to implement an EM-like algorithm, as we suggested in section 6.3, we will successively apply the updates for the Q-functions derived in the last section. Apart from these updates we also need some measure to check whether the algorithm has converged to some local maximum. The best measure to check for convergence is to evaluate (6.6) and monitor the difference after having cycled once through all Q-function updates.

Having reached a local optimum, the value of (6.6) can serve a second purpose as well: as mentioned in section 6.3 it is a lower bound of the logarithm of the true model evidence corresponding to the particular mode found during optimization. Although we have no guarantee that the bound is equally tight for different models or modes, we may nevertheless use the approximation of the log-evidence for model selection. This approach was successfully applied in [Att99] to determine the optimal number of kernels for a Gaussian mixture density estimation.

So far, the expression in (6.6) can not be evaluated directly. In order to evaluate (6.6) quantitatively, we must solve all integrals over Q-functions. In principle, these integrals *have* already been solved during the variational optimizations with respect to different Q-functions in the last section. Hence we are just left with the problem of collecting these results. However an *evaluation* of (6.6) must not drop any of the constants that were irrelevant during the variational optimization.

In order to make it easier for the reader to follow the calculations involved, we split the expression of the approximated log-evidence. First we consider the expectations of the contributions from different model parameters which are of the form $< \log(p(\varphi)) - \log(Q(\varphi)) >_{Q(\varphi)}$:

$$\begin{aligned}
< \log(p(\underline{P})) - \log(Q(\underline{P})) >_{Q(\underline{P})} &= \log(\Gamma(K\delta_P)) - \log(\Gamma(\sum_k \alpha_k^P)) \qquad (6.27)\\
&+ \sum_k \Big(- \log(\Gamma(\delta_p)) + \log(\Gamma(\alpha_k^P))\\
&+ (\delta_P - \alpha_k^P) \left(\Psi(\alpha_k^P) - \Psi(\sum_l \alpha_l^P) \right) \Big)\\
\sum_k < \log(p(\underline{W}_k)) - \log(Q(\underline{W}_k)) >_{Q(\underline{W}_k)} &= \sum_k \Big[\log(\Gamma(D\delta_W)) - \log(\Gamma(\sum_d \alpha_{k,d}^W))\\
&+ \sum_d \Big(\log(\Gamma(\alpha_{k,d}^W)) - \log(\Gamma(\delta_W))\\
&+ \left(\Psi(\alpha_{k,d}^W) - \Psi(\sum_l \alpha_{k,l}^W) \right) \left(\delta_W - \alpha_{k,d}^W \right) \Big) \Big]\\
\sum_d < \log(p(\underline{\mu}_d)) - \log(Q(\underline{\mu}_d)) >_{Q(\underline{\mu}_d)} &= \sum_d -0.5 \Big[-I + \sum_i \Big(\log(\kappa_{i,i}) - \log(\sigma^2_{\mu_{d,i}})\\
&+ \left[(\xi_i - \hat{\mu}_{d,i})^2 + \sigma^2_{\mu_{d,i}} \right] \kappa_{i,i}^{-1} \Big) \Big]\\
\sum_d < \log(p(\underline{\Sigma}_d^{-1})) - \log(Q(\underline{\Sigma}_d^{-1})) >_{Q(\underline{\Sigma}_d^{-1})} &= \sum_d \sum_i \Big[\alpha_i \left(\log(h_i^q) - \Psi(g_i^q) \right) - \log(\Gamma(\alpha_i))\\
&+ \alpha_i \left(\log(\beta_{\sigma_{d,i}}) - \Psi(\alpha_{\sigma_{d,i}}) \right) - \frac{g_i^q \alpha_{\sigma_{d,i}}}{h_i^q \beta_{\sigma_{d,i}}}\\
&- \alpha_{\sigma_{d,i}} \Big(2\log(\beta_{\sigma_{d,i}}) - \Psi(\alpha_{\sigma_{d,i}}) - 1 \Big)\\
&+ \log(\Gamma(\alpha_{\sigma_{d,i}})) \Big]\\
< \log(p(\underline{\beta})) - \log(Q(\underline{\beta})) >_{Q(\underline{\beta})} &= \sum_i \Big[g_i \log(h_i) - \log(\Gamma(g_i))\\
&+ g_i \left(\log(h_i^q) - \Psi(g_i^q) \right) - h_i \frac{g_i^q}{h_i^q}\\
&- g_i^q \left(2\log(h_i^q) - \Psi(g_i^q) - 1 \right) + \log(\Gamma(g_i^q)) \Big]
\end{aligned}$$

Furthermore, we get contributions from both labeled and unlabeled data. These contributions are integrated log-likelihoods of the form $< \log(p(\mathcal{D}|\varphi)) >_{Q(\varphi)}$:

$$
\begin{aligned}
&< \log(p(\mathcal{T}|\underline{P},\underline{W},\underline{\mu},\underline{\Sigma})) >_{Q(\underline{P}),Q(\underline{W}),Q(\underline{\mu}),Q(\underline{\Sigma}),Q(d_n)}= \\
&\sum_n \Big[(\Psi(\alpha^P_{t_n}) - \Psi(\sum_k \alpha^P_k)) + \sum_{d_n} Q(d_n) \Big[\Psi(\alpha^W_{d_n}) - \Psi(\sum_{d=1}^{D} \alpha^W_d) \\
&-\frac{I}{2}\log(2\pi) - 0.5 \sum_i \Big(\log(\beta_{d_n,i}) - \Psi(\alpha_{d_n,i}) + \Big((x_{n,i} - \hat{\mu}_{d_n,i})^2 \\
&+\sigma^2_{\mu_{d_n,i}} \Big) \frac{\alpha_{d_n,i}}{\beta_{d_n,i}} \Big) - \log(Q(d_n)) \Big] \Big] \\
&< \log(p(\mathcal{X}|\underline{P},\underline{W},\underline{\mu},\underline{\Sigma})) >_{Q(\underline{P}),Q(\underline{W}),Q(\underline{\mu}),Q(\underline{\Sigma}),Q(k_m),Q(d_m)}= \\
&\sum_m \sum_{k_m} Q(k_m) \Big[(\Psi(\alpha^P_{k_m}) - \Psi(\sum_k \alpha^P_k)) + \sum_{d_m} Q(d_m) \Big[\Psi(\alpha^W_{d_m}) - \Psi(\sum_{d=1}^{D} \alpha^W_d) \\
&-\frac{I}{2}\log(2\pi) - 0.5 \sum_i \Big(\log(\beta_{d_m,i}) - \Psi(\alpha_{d_m,i}) + \Big((x_{m,i} - \hat{\mu}_{d_m,i})^2 \\
&+\sigma^2_{\mu_{d_m,i}} \Big) \frac{\alpha_{d_m,i}}{\beta_{d_m,i}} \Big) - \log(Q(d_m)) \Big] - \log(Q(k_m)) \Big]
\end{aligned}
\tag{6.28}
$$

The approximated log evidence obtained by integrating (6.6) with respect to all Q-functions is just the sum of all expressions listed in (6.27) and (6.28):

$$
\begin{aligned}
\mathcal{F}(Q) &= < \log(p(\underline{P})) - \log(Q(\underline{P})) >_{Q(\underline{P})} \\
&+ \textstyle\sum_k < \log(p(\underline{W}_k)) - \log(Q(\underline{W}_k)) >_{Q(\underline{W}_k)} \\
&+ \textstyle\sum_d < \log(p(\underline{\mu}_d)) - \log(Q(\underline{\mu}_d)) >_{Q(\underline{\mu}_d)} \\
&+ \textstyle\sum_d < \log(p(\underline{\Sigma}_d)) - \log(Q(\underline{\Sigma}_d)) >_{Q(\underline{\Sigma}_d)} \\
&+ < \log(p(\underline{\beta})) - \log(Q(\underline{\beta})) >_{Q(\underline{\beta})} \\
&+ < \log(p(\mathcal{T}|\underline{P},\underline{W},\underline{\mu},\underline{\Sigma})) >_{Q(\underline{P}),Q(\underline{W}),Q(\underline{\mu}),Q(\underline{\Sigma}),Q(d_n)} \\
&+ < \log(p(\mathcal{X}|\underline{P},\underline{W},\underline{\mu},\underline{\Sigma})) >_{Q(\underline{P}),Q(\underline{W}),Q(\underline{\mu}),Q(\underline{\Sigma}),Q(k_m),Q(d_m)}
\end{aligned}
\tag{6.29}
$$

6.7 An algorithm for variational Bayes'

This section is meant to provide an implementation of a complete Bayesian learning scheme using the updates for all Q-functions derived in section 6.5 and the approximation to the log-evidence obtained in (6.29). *'Complete'* refers to the idea that we are interested in an approximation of the posterior probability over model coefficients *and* model classes. The latter considers the problem that the *correct* model class is unknown. However the expectations of the reader should be moderate: We will only consider models with different numbers of kernels, D. Model selection requires that the algorithm will contain two nested loops:

- The inner loop cycles through the updates of all Q-functions. The termination criterion is a test whether the variational functional (6.29) has converged.

- The outer loop starts off with one kernel for each class. During inference we increase the number of kernels as long as the final value of (6.29) obtained in the mode increases. Thus the outer loop compares the value of the approximate log-evidence in the local maximum reached in the inner loop with the corresponding value obtained with one fewer kernel. As long as the more complex model has a higher log-evidence, we increase the number of kernels by one. If the model is too complex, we will observe a decrease of the log evidence. Without any additional a-priori preference for a particular model, the most probable model is the one that reached the largest log evidence.

Pseudo-code of this algorithm together with references to the equations that contain the numerical details is shown in program 6.1.

We should remember that algorithm 6.1 minimizes the KL-divergence between the true posterior and the Q-distributions in a deterministic way. That is, each update will increase the value of functional (6.6) and we converge to a *local* optimum. Hence it is advisable to perform several runs of the algorithm and select the result that reached the largest value of (6.29).

A similar, although simpler, algorithm is also necessary for predictions. During predictions, the Q-distributions over parameters are fixed and inference is restricted to estimating the Q-distributions over latent kernel indicators, $Q(d_m)$, and unknown class labels, $Q(k_m)$. The resulting algorithm is shown in program 6.2. Note that during predictions we reach convergence rather quickly. In the experiments reported below, between 2 and 5 cycles were sufficient.

6.8 Experiments

The classifier used in this chapter is known to be difficult to infer if the number of training samples is small compared to the number of input variables. Since the Bayesian learning scheme used for inference is based on a

Program 6.1 Variational Bayes' for a generative classifier

```
initialize($Q(\underline{\mu})$, $Q(\underline{\Sigma})$, $Q(\underline{\beta})$, $Q(\underline{P})$,$Q(\underline{W})$, $Q(k_m)$)
$D = K$                              % set nr. kernels equal to nr. classes
$\mathcal{F}_{max} = -\infty$
REPEAT
    $\forall Q$: $Q_{result} = Q$
    REPEAT
        $\forall d_n$: update($Q(d_n)$)          % according to (6.11)
        $\forall k_m$: update($Q(d_m)$)          % according to (6.13)
        $\forall k_m$: update($Q(k_m)$)          % according to (6.15)
        update($Q(\underline{\mu})$)                % according to (6.17)
        update($Q(\underline{\Sigma})$)                % according to (6.20)
        update($Q(\underline{\beta})$)                % according to (6.22)
        update($Q(\underline{P})$)                % according to (6.24)
        update($Q(\underline{W})$)                % according to (6.26)
        update($\mathcal{F}(Q)$)               % according to (6.29)
    UNTIL (convergence($\mathcal{F}(Q)$))
    IF $\mathcal{F}(Q) > \mathcal{F}_{max}$ D=D+1         % then increase number of kernels
    initialize($Q(\underline{\mu})$, $Q(\underline{\Sigma})$, $Q(\underline{\beta})$, $Q(\underline{P})$, $Q(\underline{W})$, $Q(k_m)$)
UNTIL ($\mathcal{F}_{max} \geq \mathcal{F}(Q)$)
```

Program 6.2 Variational inference during predictions

```
initialize($Q(k_m)$)
REPEAT
    $\forall k_m$: update($Q(d_m)$)                  % according to (6.13)
    $\forall k_m$: update($Q(k_m)$)                  % according to (6.15)
UNTIL (convergence($\mathcal{F}(Q)$))
```

variational technique, which uses several approximations, it is advisable to check the validity of the resulting approach by some empirical tests. The experiments reported below serve four purposes:

- We will assess empirically which number of inputs we can deal with given a particular training size.
- We show that adding unlabeled samples, obtained from the same distribution as the labeled samples, *can* improve classification accuracy. We will only get an improvement if there are not sufficient labeled samples to fit a generative model.
- We will also provide empirical evidence that model selection is possible despite the approximative nature of the log evidence derived above.
- A comparison of the variational approximation to a sampled solution obtained via Gibbs updates[6] shows that the classification accuracy does not suffer - despite an increased computational efficiency both during model inference and prediction.

Practical relevance of the approach is demonstrated by applying the method to the problem of sleep analysis that was introduced in chapter 2.

6.8.1 Small data limits and model selection

One of the problems of generative classification is that, compared with predictive classifiers, the models are much more complicated to train. The larger the input dimensionality, the higher the chance that model inference fails. This is the *curse of dimensionality*, such as mentioned in [Bis95]. Hence, when proposing a generative model, we should assess the behaviour with respect to data sets with high input dimensionality. These aspects are best shown on synthetic data with known properties. A problem that can not be solved in a subspace spanned by the inputs is an aritificial XOR-problem that can be generated in various dimensions. For an I-dimensional problem we need 2^I kernels, each positioned in a vertex of a hyper cube. The vertices are obtained by generating all possible permutations of 1 and -1 within this I dimensional space. The labels for all samples drawn from one of these kernels are obtained to give an XOR relationship between the kernels and the label. (e.g. set the label to 1 if the number of positive 1's in the kernel vector

[6]It can be shown that the Gibbs sampler will - at least theoretically - find the correct posterior. As already mentioned in section 6.3, this is not true for the variational approximation.

is even and to 2 otherwise.) The experiments mentioned below have been performed with Gaussian kernels using diagonal covariance with standard deviation 1. This problem is referred to as *XOR*.

The disadvantage of the XOR data is that the underlying model is contained in the model class represented by the generative classifier. Therefore, we consider an alternative problem as well, where data are uniformly distributed within an I-dimensional hyper cube. Labeling is obtained by assigning all samples within a sphere to class 1 and all samples outside to class 2. In order to make this problem stochastic, we added a small amount of Gaussian noise with std. deviation of 0.1 to each dimension. These data will be referred to as *circle*.

In order to assess the performance with respect to different ratios of input dimensions and number of samples, we draw 500 training samples for both synthetic problems at different input dimensions. The smallest data set has 2 inputs. The generative model is fit into these data sets and its performance is evaluated using a test set that contains 500 samples as well. As pointed out in section 6.7 model fitting must be done with care. Especially for complex problems such as XOR in high dimensions the inference algorithm must be run more than once. The results obtained in this section were achieved by allowing for 5 repetitions in each training run.

The behaviour of the proposed algorithm is illustrated by plots of the approximated log evidence and by a table that lists the achieved generalization accuracies. Figures 6.2 and 6.3 show the log evidences for the XOR problem. We see that the model is estimated correctly for the 2- and 3-dimensional problem. However inference of the 4-dimensional problem breaks down completely. Table 6.1 shows a decay of the generalization accuracy with increasing input dimension.

Table 6.1: Generalization accuracies at different input dimensions

	2 dim.	3 dim.	4 dim.	5 dim.
circle data	94 %	83 %	79 %	50 %
XOR data	87 %	78 %	50 %	

The approximated log evidence using different numbers of kernels for the *circle* problem is shown in figures 6.4 and 6.5. What we clearly see is that the algorithm does not find a proper solution for the 5-dimensional problem. The generalization accuracies reported in table 6.1 confirm this impression. We see that the performance gets worse with increased number of inputs and breaks down for the 5 dimensional problem.

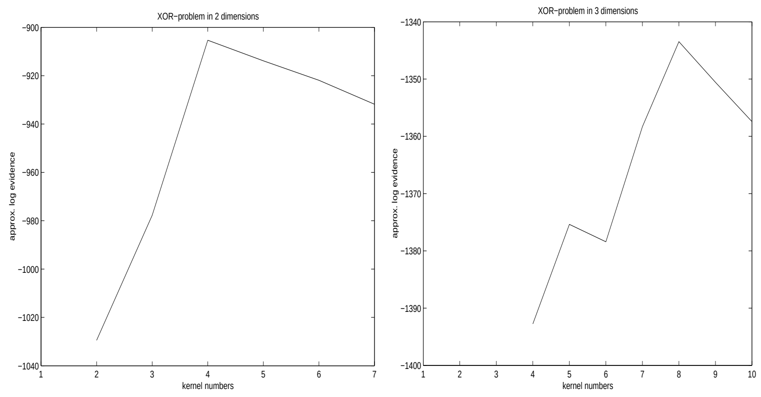

Figure 6.2: Approximated log evidence for the XOR problem in 2 and 3 dimensions. Predictions have been carried out with the model that got highest log evidence.

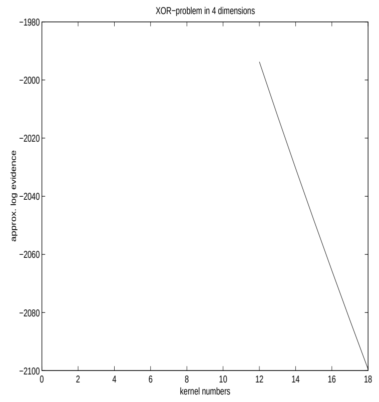

Figure 6.3: Approximated log evidence for the XOR problem in 4 dimensions. Inference of this model fails already. Hence we did not worry about testing the 5 dimensional problem. At least the inference is honest in the sense that the smallest model we tried got the highest log evidence.

In the case of the circle data, the 5-dimensional problem could be solved by adding the 500 test samples as unlabeled data to the training samples. The generalization accuracy increases to 70%. In order to assess significance, we count the samples that were classified differently by the two classifiers. In particular we are interested in the errors made by each of the classifiers that were not made by the other (n_a and n_b). The classifier that was trained using labeled and unlabeled got $n_a = 129$, the original classifier $n_b = 229$. Referring this to a Binomial $\mathcal{B}n(n_a + n_b, 0.5)$ distribution reveals that this difference is highly significant. Figure 6.6 shows the log evidences obtained in the corresponding training run.

The conclusion from these experiments is that we can not easily determine

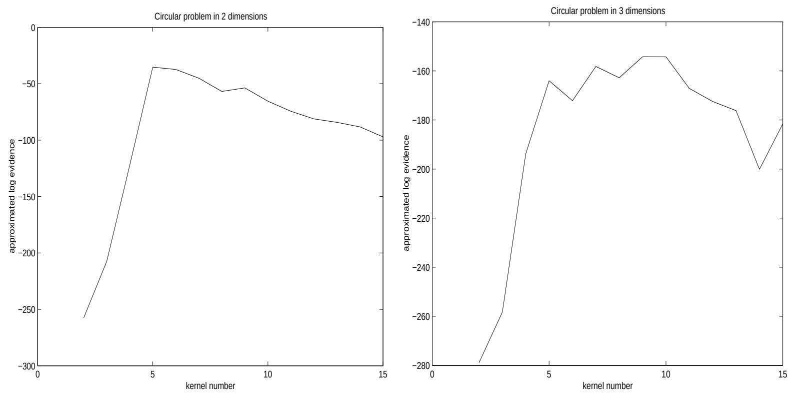

Figure 6.4: Approximated log evidence for the uniform circular problem in 2 and 3 dimensions. Predictions have been carried out with the model that got largest log evidence.

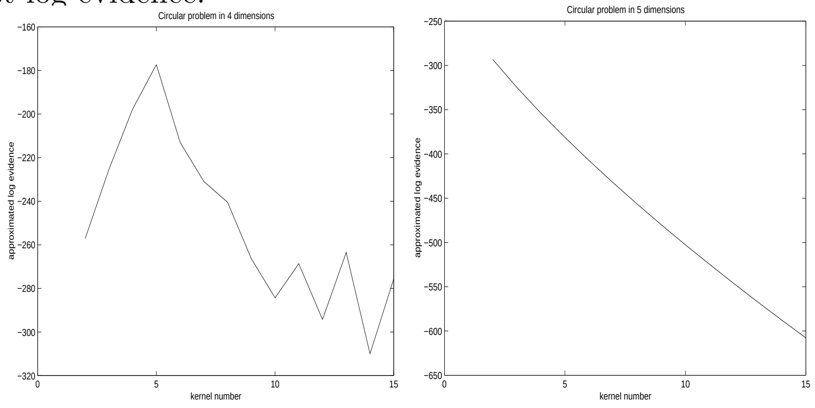

Figure 6.5: Approximated log evidence for the uniform circular problem in 4 and 5 dimensions. In the last plot we can see that the method suggests a model with 2 kernels, which was the simplest model we tried.

how many samples we need for a particular number of inputs. The number of inputs we need will also depend on the complexity of the problem. However the good news is that the method will propose the smallest model complexity that was tried when the number of inputs does not suffice to determine a solution.

These results have also shown that although (6.29) is only a lower bound of the log evidence, the results we obtain from model selection are promising. We may conclude that the optimal number of kernels suggested by the algorithm illustrated in program 6.1 is a reasonable choice of model complexity. This is important in situations where we want to use the generative nature of the model to obtain further insight into how a particular classification was reached. That is, these results allow us to be confident that the information

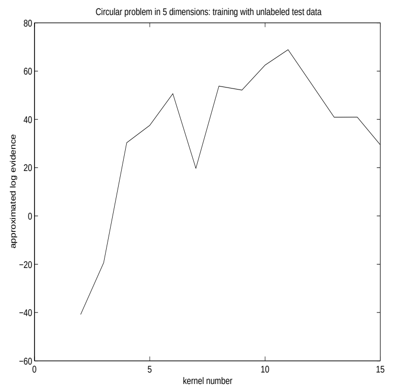

Figure 6.6: Approximated log evidence for the uniform circular problem in 5 dimensions when using labeled and unlabeled data for inference. Predictions have been carried out with the model that got largest log evidence.

about the most probable kernel a sample was generated from is useful if such analysis is of interest.

6.8.2 Comparison with a Gibbs sampler

Comparing the variational solution with inference via Gibbs updates, we are mainly interested in issues of computational efficiency and generalization accuracy. We use two data sets to assess that. One test is performed with the 3 dimensional XOR problem. We use again 500 samples as training data and 1000 samples as test data. The other data we use in thic comparison is from the sleep analysis task discribed in chapter 2. The training data contain 5929 samples extracted from 6 different recordings. The generalization accuracy was estimated with 17070 samples from 3 independent recordings. The classification inputs are the first 3 reflection coefficients extracted from electrode C3. Preprocessing was performed with the methods described in chapter 4. According to the motivations in chapter 2, we predict probabilities for wake, rapid eye movement (REM) sleep and deep sleep. In both experiments we fix the number of kernels; For the XOR problem we know the true number, which is 8; for the EEG data we set the number of kernels to 9. As is shown in the next subsection, this is at the lower end of probable models.

Both algorithms are assessed in terms of time needed until convergence. For the Gibbs sampler convergence is hard to define. In principle, this is done by checking the sampled Markov chains. As this is quite an involved procedure, we decided to draw 1000 samples from the posterior and use the second half for prediction. Apart from the time needed for inference we are also interested in the time the method needs during predictions and, of course, in generalization accuracy.

Inference (that is training) of the XOR model took about 3700 seconds with the Gibbs sampler and about 531 seconds with variational inference. Inference of the classifier for classification of sleep data took $1.8636 \cdot 10^4$ seconds, which is slightly more than 5 hours. With the same data variational inference took about 2994 seconds. Hence in both experiments variational inference was about 5 to 6 times faster than Gibbs sampling.

Table 6.2: Comparison variational inference versus Gibbs sampling

	XOR	Sleep 1	Sleep 2	Sleep 3
nr. test cases	1000	1140	7530	8400
	predictions from variational inference			
time (s)	3.86	12.3	12.96	12.27
gen. acc. (%)	78.5	97.8	46.7	74.4
n_a	7	19	319	348
	predictions from Gibbs sampling			
time (s)	4.45	58.37	58.07	55.61
gen. acc. (%)	77.9	98.2	39.8	66.4
n_b	13	11	737	1200

The resulting predictions together with the time we need for predicting are reported in table 6.2. As already mentioned, the XOR test data consists of 1000 samples. The results obtained on the 3 sleep recordings have been evaluated using such epochs only that were equally classified as wake, REM or deep sleep by 2 independent and one consensus human experts. Hence the number of samples in the corresponding test sets differs. We conclude from these experiments that we obtain the variational results faster without losing performance. In fact, for 2 of the sleep experiments we even get a significantly better result when predicting with the variational classifier. Note, however, that this rather points to the fact that we should have drawn more samples during inference than to a shortcoming of the Gibbs sampler.

We want to finish this comparison of variational inference and Gibbs sampling with two figures that show the probability traces obtained for 2 sleep recordings. We see three traces that predict probabilities for wake, REM sleep and deep sleep. The probabilities have been "smoothed" by assuming conditional independence of the observations contained within a window of 20 seconds. That is, we perform some kind of naïve Bayes sensor fusion[7].

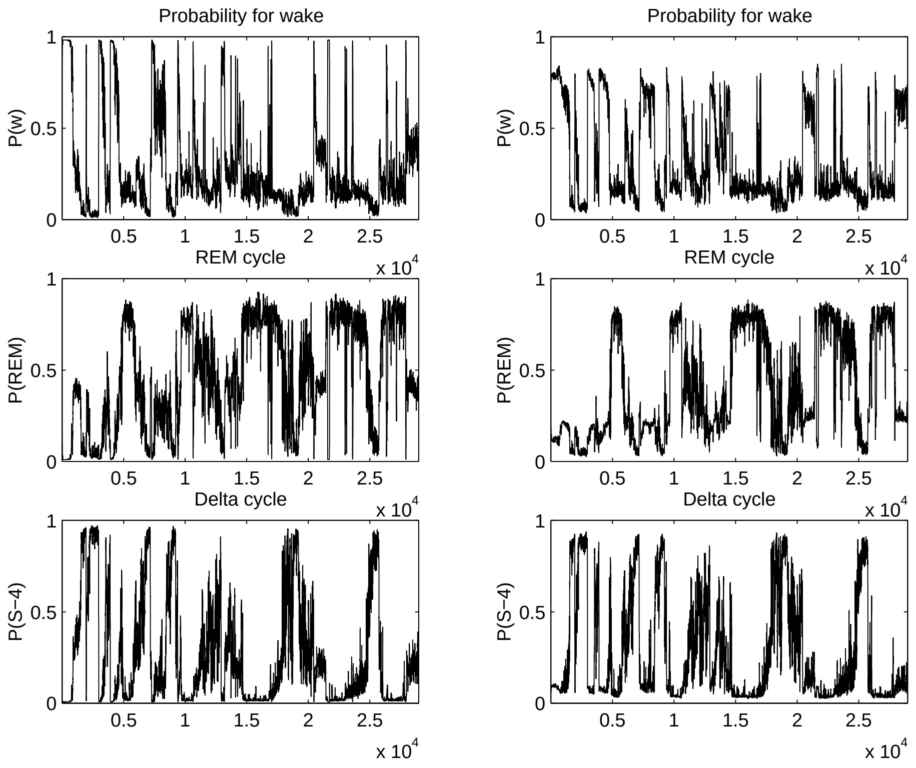

Figure 6.7: Probability traces obtained for an all night recording. The entire night is interpreted via probabilities for wake, REM and deep sleep. The left plots show the probabilities predicted by a classifier obtained via variational inference. The right plots show the probabilities predicted with a classifier parameterized via Gibbs sampling. The left plots have a larger range which is due to an underestimation of the parameter variance caused by the mean field approximation. We see that the entire night is segmented in epochs of deep sleep and REM sleep that is occasionally interrupted by wake.

Figure 6.7 shows a recording where we clearly identify some "structure" of sleep. The entire night is segmented in epochs of high probability for deep sleep which alternate with epochs of high REM probability. This cycle is occasionally interrupted by segments with high probability for wake.

A possible explanation why we can not see such structure in figure 6.8, is the poor quality of the second recording. The probabilities are closer to equal probability, which indicates some uncertainty in classifying this recording.

[7]The code used for this sensor fusion has been provided by I. Rezek, S. Roberts and W. Penny from the Robotics research group at the University of Oxford, UK.

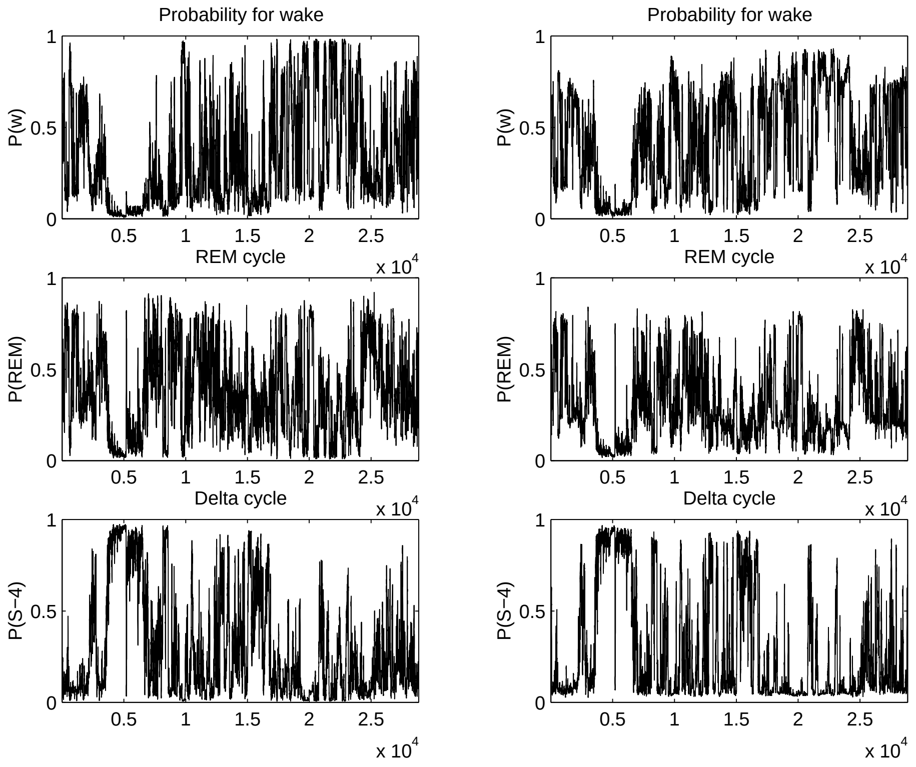

Figure 6.8: Probability traces obtained for an all night reording. The entire night is interpreted via probabilities for wake, REM and deep sleep. The left plots show the probabilities predicted by the variational Bayesian classifier. The probabilities in the right plots are predictions from a classifier sampled via Gibbs updates. The left plots have larger range which is due to an under estimation of the parameter variance caused by the mean field approximation. A nice segmentation of the night is not possible, which is most probably caused by the poor quality of the recording.

The left traces in figures 6.7 and 6.8 have been obtained with the variational classifier. The right plots are obtained by predicting with the classifier inferred with Gibbs sampling.

It is remarkable that the range of the traces obtained by variational inference is larger. The reason for the different ranges are larger moderation effects[8] due to higher variance of the parameter distributions as estimated by the Gibbs sampler. This is confirmed through observations by T. Jaakkola[9]. The variance of the Q-distributions obtained by variational inference is smaller because of simplifications introduced by using a mean field approximation. Assuming a factorizing distribution of the parameters neglects dependencies among them. Hence the variances of the true parameter distributions are underestimated.

6.8.3 A thorough analysis of all night sleep EEG

The experiments presented here are, so-to-speak, the driving force behind all the efforts reported in this chapter. We performed an experiment, where we once more use sleep recordings. As motivated in chapter 2, we interpret the entire night in terms of probabilities for Rechtschaffen and Kales (R & K) labels wake, REM sleep and deep sleep (stage 4). The labels taken for the supervised part were taken from segments that have been unanimously been classified as either wake, REM or stage 4 by 3 human experts. All other epochs were added to the training data without labels. That is we have a partially supervised problem. We used raw EEG from electrodes C3, O1, C4 and O2. The raw EEG was preprocessed using the lattice filter described in chapter 4. We extracted 3 coefficients for each second of each all night recording. Training was done using data from 6 recordings, where we fitted one model for each electrode separately. The idea behind this redundancy is that a failing electrode will not invalidate the entire analysis. Sensor fusion will fuse the probabilities obtained for each electrode. The expert labels have been provided for each 30 seconds epoch of the entire night. Hence we would get 30 successive samples labeled equally. The resulting number of training samples is much larger than required and we should down-sample such that only one sample remains for each 30 seconds epoch[10]. Instead of random sub-sampling, we extracted the median sample, measured in terms of Euclidean distance from the origin. This procedure gave a training set

[8]Probabilities obtained by marginalizing out parameter uncertainties will always be less certain about the state of interest. This effect is referred to as moderation (e.g. [Bis95]).

[9]Personal communication at the NIPS conference, Denver Colorado, in December 1999.

[10]Using all data is no problem except for an unnecessary computational burden we want to avoid.

with 1835 labeled and 3962 unlabeled samples.

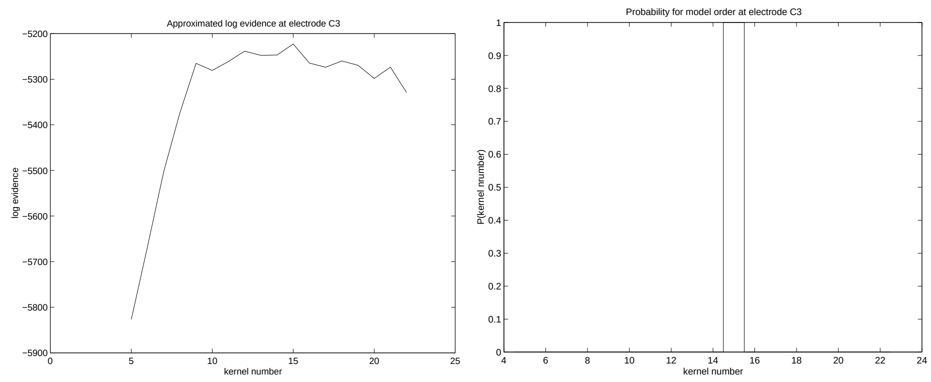

Figure 6.9: These plots show the approximated log evidence and model probabilities for different numbers of kernels, resulting from inference at electrode C3. We see that the model with 15 kernels dominates by far, it has almost the entire probability mass.

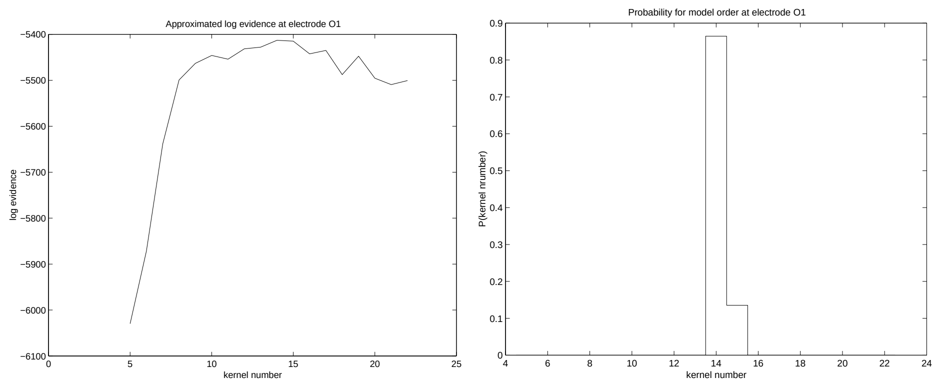

Figure 6.10: These plots show the approximated log evidence and model probabilities for different numbers of kernels, resulting from inference at electrode O1. The model with 14 kernels has a probability of about 0.85. The remaining mass is on the 15 kernel model.

In this experiment a lot of care has been taken to perform training optimal. That is we perform model selection based on the approximated log evidence, (6.29). Due to prior expirience with the data, we expected the optimal number of kernels to be between 4 and 22. Inference was repeated 5 times for each kernel number. For each kernel we selected the result with highest log evidence. The resulting plots of the log evidence and model probabilities for electrode C3 are shown in figure 6.9, for electrode O1 in figure

6.10, for electrode C4 in figure 6.11 and for electrode O2 in figure 6.12. We get rather different answers about the optimal kernel numbers for the different electrodes. The best interpretation of that is that it hardly matters which model to chose in the range of 10 to 19 kernels. The differences in log evidences are most probably due to different modes found during inference.

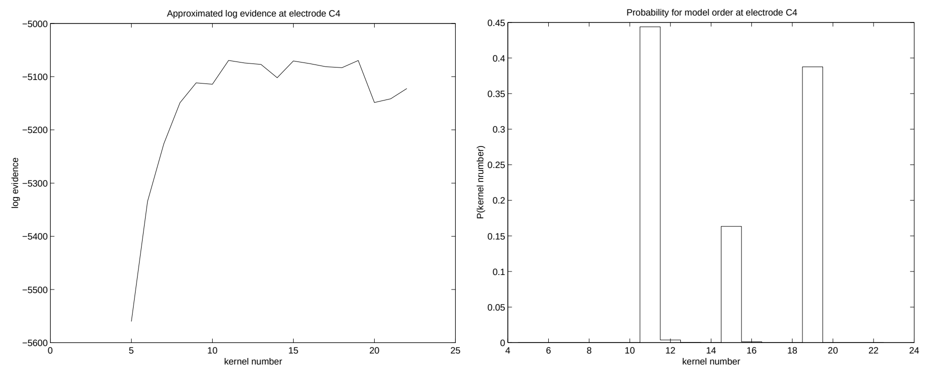

Figure 6.11: These plots show the approximated log evidence and model probabilities for different numbers of kernels, resulting from inference at electrode C4. Several models share the probability mass in this experiment: the model with 11 kernels has a probability of 0.45, the model with 15 kernels has probability 0.15 and the rest is allocated to the model with 19 kernels.

Predictions were done applying the most probable model to each electrode and using a naïve Bayes conditional independence assumption for sensor fusion. The test data are 6 all-night recordings. The plots in figures 6.17, 6.18, 6.19, 6.20, 6.21 and 6.22 were obtained by sensor fusion of all 4 channels and in order to smooth with a window of 20 seconds length. In order to allow a comparison with the classical Rechtschaffen and Kales scorings, the consensus scoring is depicted as well. Since visual inspectio alone is not sufficient, we also aimed at a numerical quantification.

In order to allow such numerical quantification of the plots, we compared the generalization accuracies obtained from predicting labels wake, REM and delta sleep (the combined R & K stages 3 and 4) when comparing with the corresponding R & K labels of the consensus scorings. The difficulty with this attempt is that we have to determine 3 thresholds for each probability trace, in order to predict the corresponding labels. To avoid a too strong dependence on the estimates of the class priors for these 3 events, we use a variable cost approach and plug in priors known from literature (e.g. [Kub95]), instead of empirical estimates from the recordings.

Despite the fact that we deal with a 3 class problem, we can use a sensitivity - specificity analysis as known from receiver operating characteristic

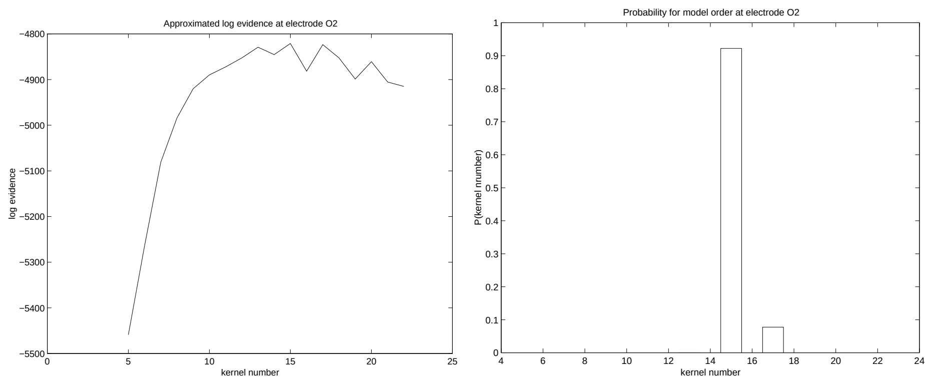

Figure 6.12: These plots show the approximated log evidence and model probabilities for different numbers of kernels, resulting from inference at electrode O2. We see that the model with 15 kernels has a probability of 0.9 and the model with 17 kernels has a probability of 0.1.

(ROC) plots (see e.g. [ZC93]), originally defined only for 2-class problems. We may do so because at the threshold values of interest (rather high thresholds) the problem separates nicely into 3 independent dichotomous classification problems. That is, at those thresholds there will not be any interference among the 3 different classes.

The ROC-like plots obtained for this problem are shown in figure 6.13. It is worth mentioning the difference to conventional ROC plots. We see that neither of the 3 classes reaches specificity 0. This is because at low threshold values we will predict the "winner", which is the class with highest probability.

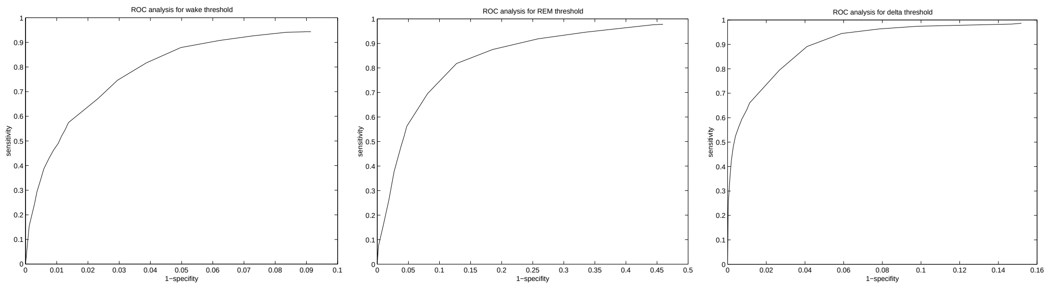

Figure 6.13: Three ROC plots obtained for R & K classes wake (left), REM (center) and delta sleep (right). The analysis was performed using the 6 recordings that have been used for training and labels from consensus hypnograms. As opposed to classical ROC curves these plots do not reach specificity 0, as for small thresholds we predict the class with largest probability.

The optimal thresholds for classifying wake, REM or delta sleep depend on the prior probabilities of those classes. According to [Kub95], these are different depending on the length of a subjects sleep. The prior probability for wake is between 0.036 and 0.086. For REM we observe prior probabilities between 0.198 and 0.236. Finally, delta sleep (combined stages 3 and 4) has a prior probability between 0.136 and 0.238. Assuming equal cost for misclassifications, the threshold value must minimize the expected generalization error. Figure 6.14 shows the expected generalization error over threshold values for class wake. In figures 6.15 and 6.16 we see these expected generalization errors for class REM and delta. The minimizing threshold depends on the assumed class priors. They are summarized in table 6.3.

Table 6.3: Optimal threshold values from a sensitivity - specificity analysis

	P(class)	P_{thr}	P(class)	P_{thr}	P_{thr} chosen
wake	0.036	0.86	0.086	0.8	0.83
REM	0.198	0.8	0.238	0.75	0.775
delta	0.138	0.65	0.238	0.6	0.625

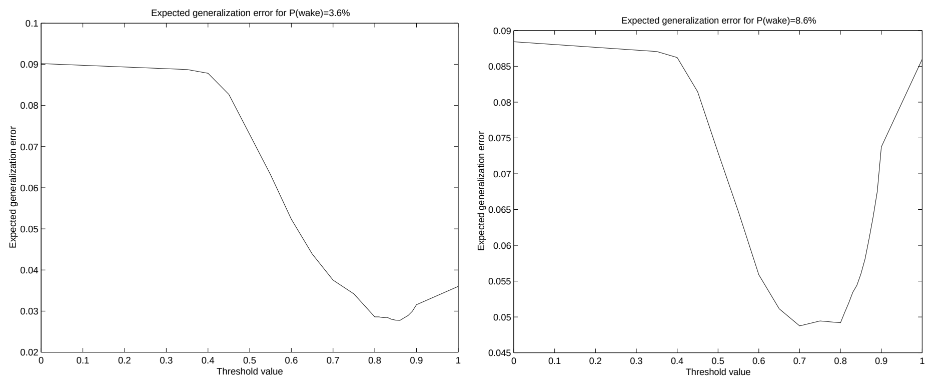

Figure 6.14: The plots show the expected generalization error when classifying class wake over different threshold values. The left plot is obtained assuming $P(wake) = 0.036$. The results of the right plot assume $P(wake) = 0.086$.

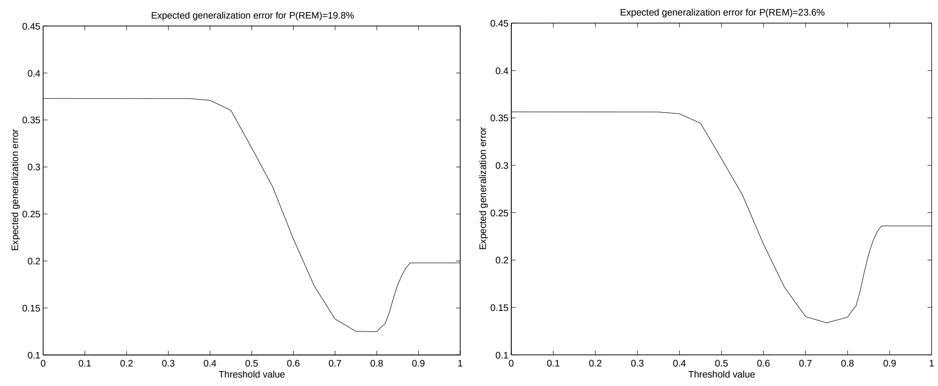

Figure 6.15: The plots show the expected generalization error when classifying class REM over different threshold values. The left plot is obtained assuming $P(REM) = 0.198$. The results of the right plot assume $P(REM) = 0.236$.

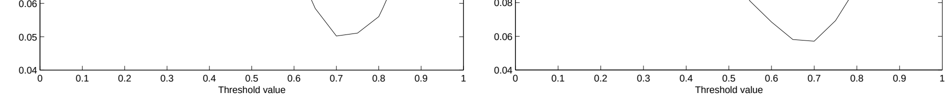

Figure 6.16: The plots show the expected generalization error when classifying combined R & K stages 3 and 4 over different threshold values. The left plot is obtained assuming $P(delta) = 0.136$. The results of the right plot assume $P(delta) = 0.238$.

Having determined the thresholds summarized in table 6.3, we are now ready to predict the R & K stages wake, REM and delta (combined stages 3 and 4) from the continuous hypnograms. The results summarized in table 6.4 were obtained for the 6 independent test recordings already mentioned above. Note that table 6.4 also contains the agreement of the 2 independent scorings with the consensus score. We see that these results agree with the impression we get from visually inspecting the corresponding continuous hypnograms in figures 6.17, 6.18, 6.19, 6.20, 6.21 and 6.22. That is, for some recordings it is possible to predict the R & K stages reliably. For others this is impossible.

It is worth mentioning to that also a separation of REM non-REM is possible on *some* recordings by using only information from EEG. This is contrary to the results e.g. reported in [FSRD00], where on the same data such a separation was not possible - even when adding EMG[11]. The major difference between [FSRD00] and our experiment is that [FSRD00] has used a hidden Markov model, which is an unsupervised technique and hence not an optimal choice for a supervised problem (e.g. [Rip96]).

Table 6.4: Comparison with expert scorings

	subj. 1	subj. 2	subj. 3	subj. 4	subj. 5	subj. 6
spc. wake (%)	81.62	59.8	58.68	49.02	52.32	20.5
spc. REM (%)	94.93	79.37	96	47.24	77.72	19.7
spc. S3 & S4 (%)	79.6	93.73	49.81	38.6	51.11	67.21
sns. wake (%)	94.4	97.03	97.86	99.91	99.96	92.02
sns. REM (%)	99.19	89.17	88.15	69	79.8	78.7
sns. S3 & S4 (%)	98.07	89.42	90.55	100	100	92.66
acc. var. inf. (%)	86.58	69.96	71.75	46.99	62.32	50.44
acc. rater 1 (%)	44.82	93.16	95.38	92.48	92.46	92.2
acc rater 2 (%)	91.82	94.1	92	92.92	90.99	96.26

6.9 Summary

The method proposed in this chapter has been extensively studied using several synthetic and one real world problems. The major findings of these experiments are:

[11]This is due to the poor quality of the EMG signal

- Despite that variational inference maximizes a lower bound of the log evidence, model selection is possible. Hence the number of kernels used in the classifier is a well determined quantity that allows to use the posterior probabilities of each kernel to obtain deeper insight into the problem. We may for example identify a more fine grained structure of the problem as is provided by the number of predicted classes.

- We can diagnose a break down of inference when the number of training samples is insufficient by monitoring the approximated log evidence. In such cases where we have not enough labeled examples adding unlabeled data will improve inference. In these cases we observe a significantly higher generalization accuracy.

- The approximation obtained by variational inference allows significantly faster inference than sampling techniques. Due to the mean field expansion of the estimated posterior, we get a result that underestimates the variance of the parameter distributions. This manifests in over confident probability estimates. However despite that we did not observe significantly worse generalization accuracies.

- Comparing the generalization accuracies obtained on sleep recordings with human labels of stages wake, REM and delta sleep (combined stages 3 and 4) we find less problems with predicting REM as was previously reported in [FSRD00].

We may therefore suggest that the inference technique proposed in this chapter is a viable way to perform predictions in the biomedical domain. At present the method undergoes an extensive evaluation within the EC biomed project SIESTA[12]. This evaluation uses about 600 all night recordings and investigates whether these recordings can be quantified in terms of a measure ranging from good sleep to bad sleep and sleep disorders. The first step in this analysis has been done in subsection 6.8.3: We could show that for some recordings a REM - non-REM separation was possible solely based on EEG information. This observation is of particular interest because it could help to overcome the problems with EMG data as are in [FSRD00].

There is also some room left for improving the inference algorithm. Since program 6.1 will only find a *local* optimum of (6.29), it might be advisable to perform the search for optimal models on a kernel basis similar to the approach taken in [GB00].

[12]Project details may be found in the acknowledgments chapter.

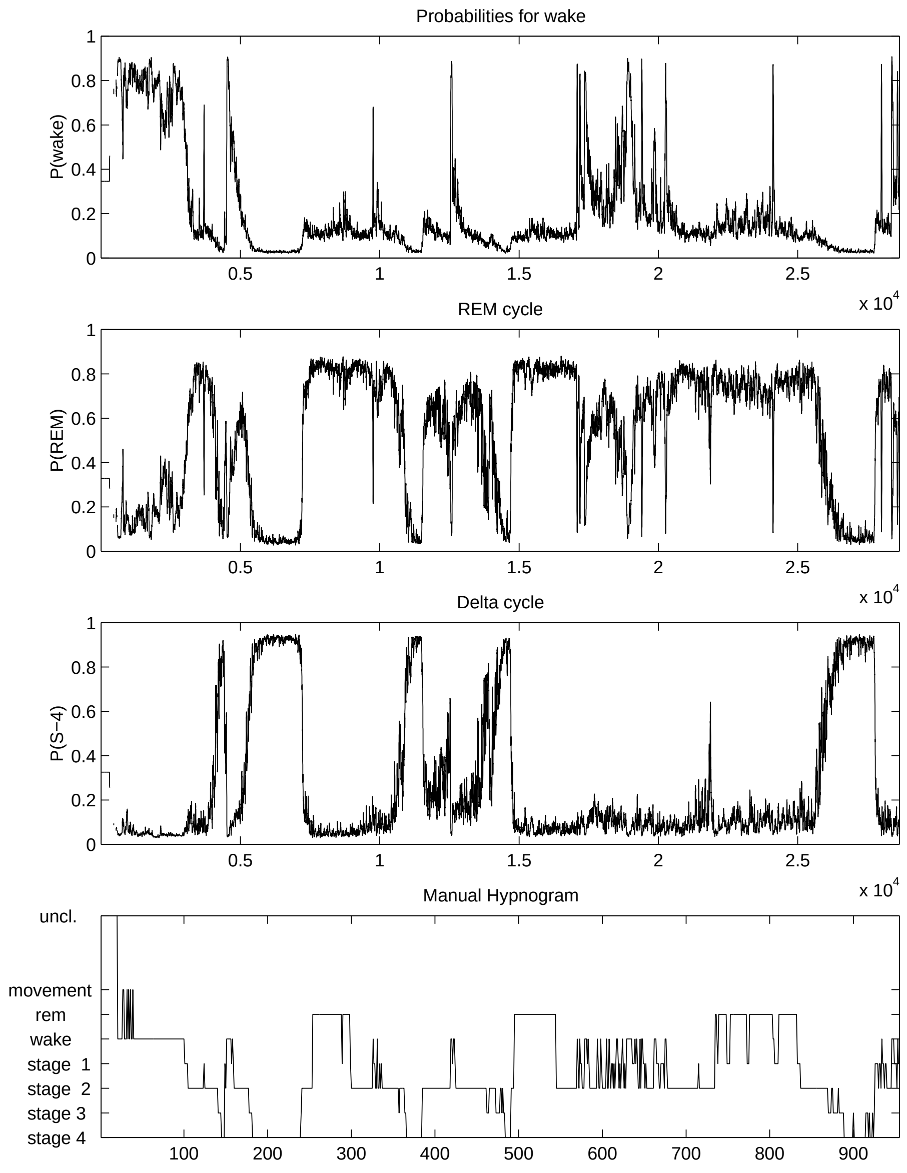

Figure 6.17: Probability traces obtained for an all-night recording. The entire night is interpreted via probabilities for wake, REM and deep sleep. The probabilities have been obtained by sensor fusion of channels C3, O1, C4 and O2. An R & K scoring is added to allow for a comparison with the classical scoring technique. It shows that at the information contained in the delta cycle and in the probabilities for wake conform with the R & K scoring. The REM cycle looks plausible in the first half of the night. However, during the second half there is some overlap with the R & K stage 2.

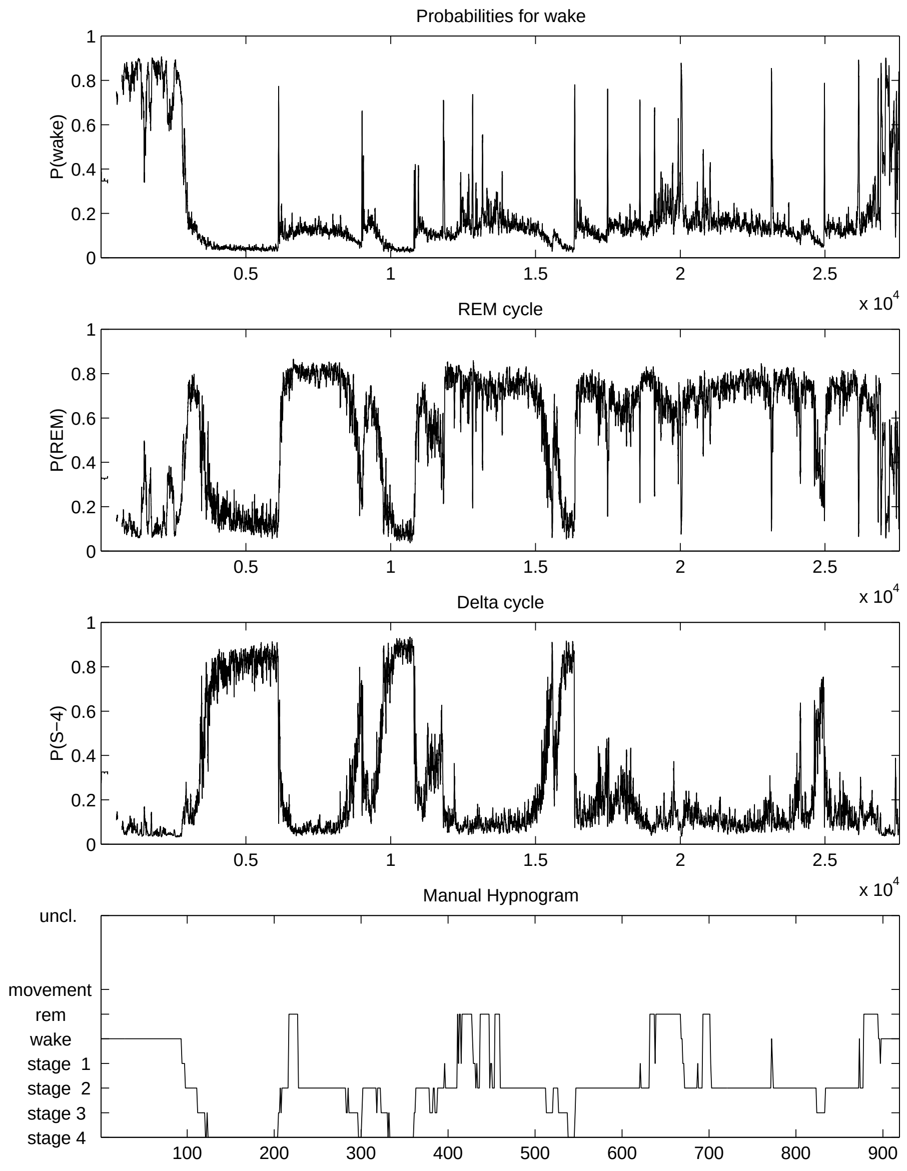

Figure 6.18: Probability traces obtained for an all-night recording. The entire night is interpreted via probabilities for wake, REM and deep sleep. The probabilities have been obtained by sensor fusion of channels C3, O1, C4 and O2. An R & K scoring is added to allow for a comparison with the classical scoring technique. We see that the delta cycle and the probabilities for wake show similar information as the R & K scoring. Again there are some problems with the REM cycle.

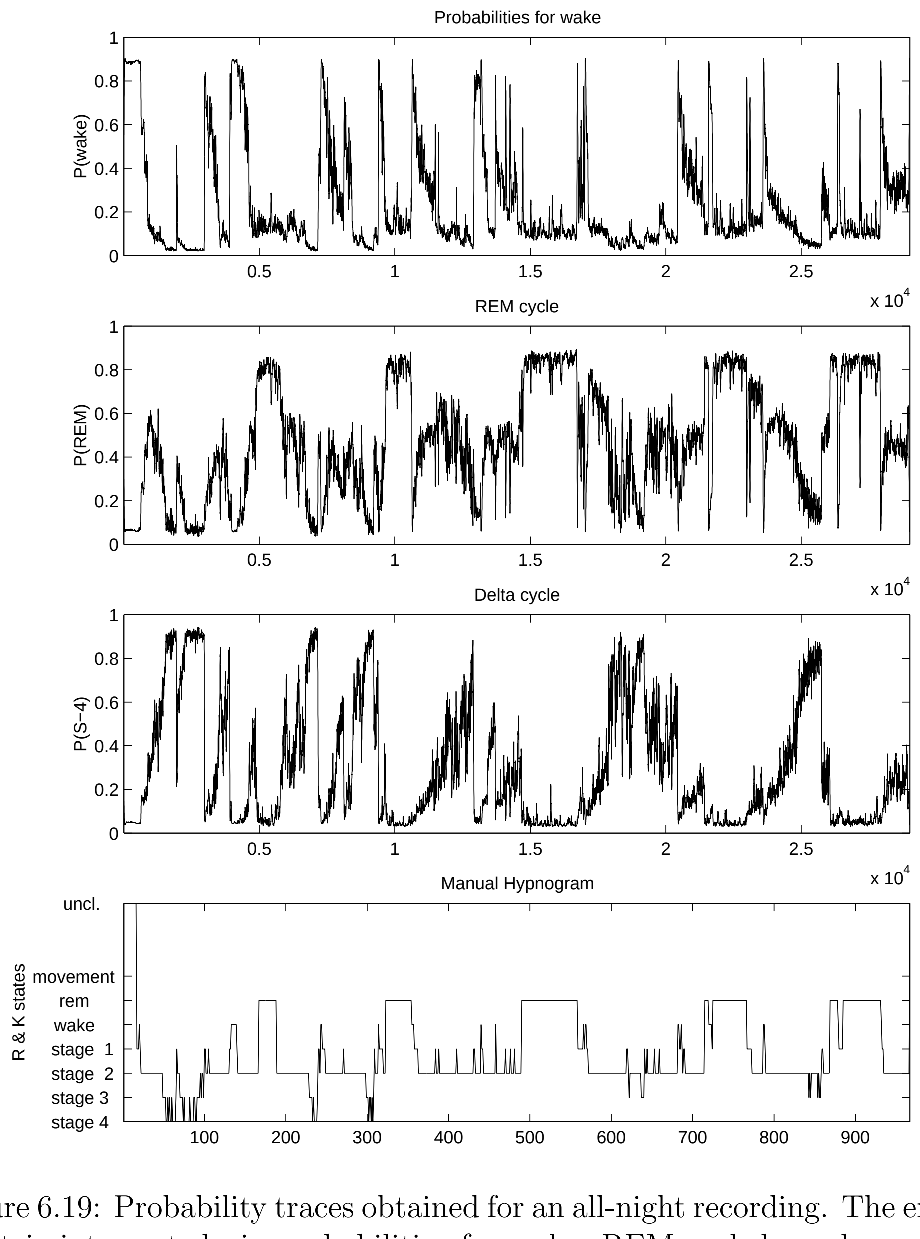

Figure 6.19: Probability traces obtained for an all-night recording. The entire night is interpreted via probabilities for wake, REM and deep sleep. The probabilities have been obtained by sensor fusion of channels C3, O1, C4 and O2. An R & K scoring is added to allow for a comparison with the classical scoring technique. In this plot we see an almost perfect correspondence of the plots with R& K labels. Also in the sense that there will not bee too many false positives when thresholding the REM probability.

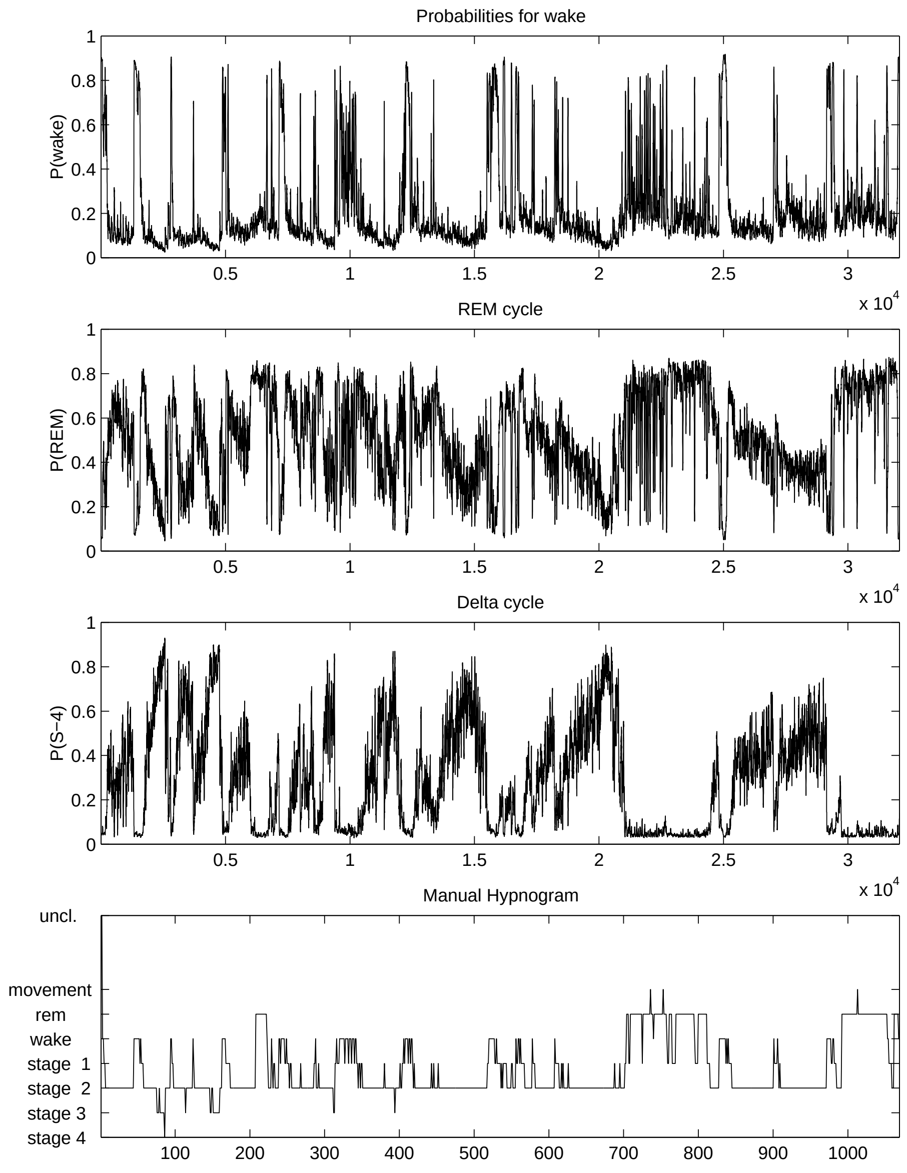

Figure 6.20: Probability traces obtained for an all-night recording. The entire night is interpreted via probabilities for wake, REM and deep sleep. The probabilities have been obtained by sensor fusion of channels C3, O1, C4 and O2. An R & K scoring is added to allow for a comparison with the classical scoring technique. The probability traces shown in this figure allow also to extract R & K stages. However, there will also be some false positives for REM, especially in the first half of the night.

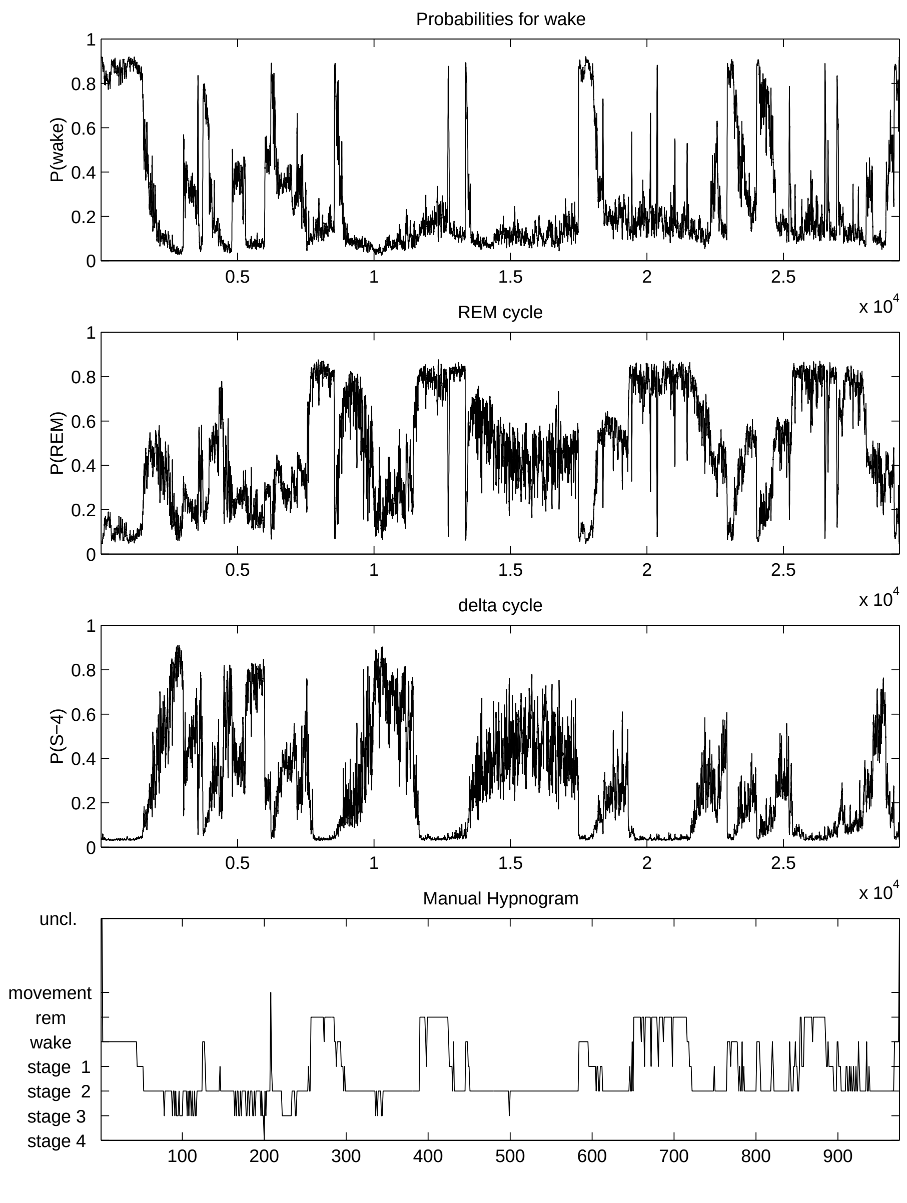

Figure 6.21: Probability traces obtained for an all-night recording. The entire night is interpreted via probabilities for wake, REM and deep sleep. The probabilities have been obtained by sensor fusion of channels C3, O1, C4 and O2. An R & K scoring is added to allow for a comparison with the classical scoring technique. These plots allow again a rather good determination of the extreme sleep events wake REM and deep sleep without too many false positives.

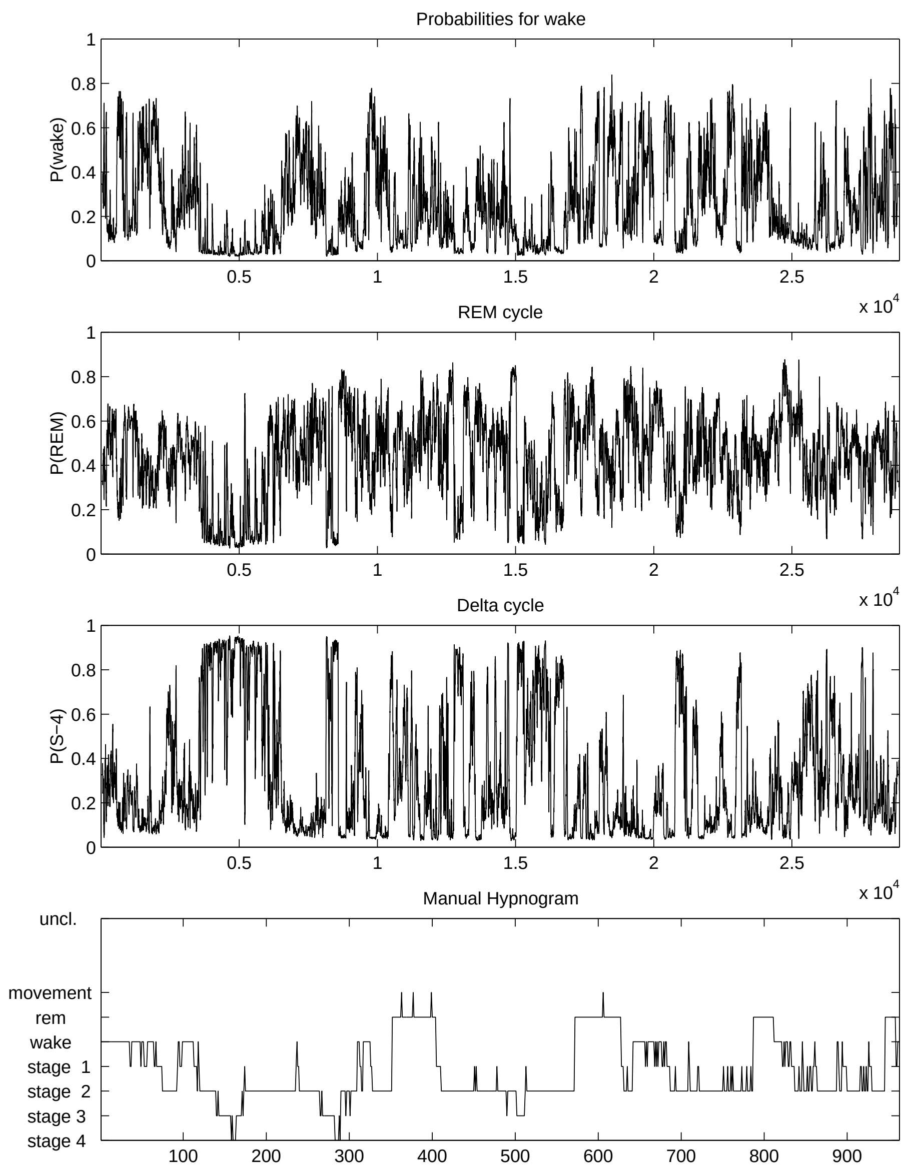

Figure 6.22: Probability traces obtained for an all-night recording. The entire night is interpreted via probabilities for wake, REM and deep sleep. The probabilities have been obtained by sensor fusion of channels C3, O1, C4 and O2. An R & K scoring is added to allow for a comparison with the classical scoring technique. In this figure the plots obtained by the variational classifier have almost no correspondence with the R & K stages. The reason is probably the extreme low quality of the recording.

Chapter 7

Bayesian sensor fusion

Sensor fusion denotes an approach that combines information obtained from different sources. In time series classification of EEG signals we obtain information extracted from different electrodes and at different time instances. This information should be combined in a way that the resulting classifications are optimal. An intuitive idea is to downweight such information that was assessed to be unreliable. The method proposed in this chapter will assess the reliability of sensors by using the Bayesian preprocessing technique described in chapter 4. We will show that in order to perform optimal sensor fusion, it is sufficient to use a probabilistic link between preprocessing results obtained from different sensors and the classification stage.

The approach developed in this chapter is also interesting from another point of view. The proposed model and the inference scheme result when we follow the requirements that are necessary to comply with the Bayesian framework. These requirments were formulated in chapter 4 to motivate that preprocessing must be done within the Bayesian framework. We concluded that predictions must be obtained from marginal distributions after integrating out all uncertainties. Hence the key idea of what we call *Bayesian sensor fusion* is to treat the results obtained from preprocessing as latent variables and consider two types of uncertainties. We assess the reliability of each lattice filter stage by its probability as opposed to a white noise explanation of the corresponding data. The uncertainties about the lattice filter coefficients are modeled by a posterior distribution. Both the probabilities of the lattice filter stages and the posterior distributions were obtained in chapter 4.

In this chapter we propose a generative model and a Bayesian learning scheme that builds on the findings from chapter 4. The proposed algorithms for learning and prediction fuse information from different sensors spatially and across time. Thus our models as well as predictions will depend more on accurate information. Furthermore the proposed approach will also estimate the expected values of the latent feature variables. Our experiments show that in such situations where either the feature variables or the preprocessing method are associated with high uncertainty these expected feature values differ a lot from the best estimates obtained from preprocessing alone. The proposed algorithm deals with all kind of missing values. We may use unlabeled data for learning as well as missing inputs during learning and predictions. In order to get a computationally tractable method, both feature and model uncertainty of the preprocessing stage are obtained in closed form. In the classification part both parameter inference and predictions are performed with Markov chain Monte Carlo (MCMC) methods.

We conclude this chapter with an evaluation of the method that uses two problems. A synthetic experiment is used to provide some insight into the method. Finally we apply the method to a classification of all night sleep EEG recordings.

7.1 Introduction

We have argued in chapter 4 that once we opt for an analysis within a Bayesian framework, we must do this throughout the entire decision process. The approach studied in this section assumes that we are given some segments of time series, each labeled as being in one of K possible states. Such a setting is typically found in biomedical diagnosis, where successive segments of bio-signals are to be classified. Usually, the number of samples within the segments is large. However, the information contained in a segment is often represented by a much smaller number of features. Such problems are typically solved by splitting the whole problem into two parts: a preprocessing method that extracts some features and a classifier.

So far we have used Bayesian inference for both of the tasks separately.

By using the best estimates from preprocessing as inputs for the classification task, we have so far used a non probabilistic link between preprocessing and classification. Such a link does not allow feature or model uncertainty, as is found by Bayesian preprocessing, to have an influence on the beliefs reported by the classifier. In a Bayesian sense this is equivalent to approximating the posterior probability density over feature variables by a delta peak and ignoring the uncertainty in the stochastic model used for preprocessing.

As was proposed in chapter 4, a model that allows for a probabilistic link between preprocessing and classification must have a structure similar to the directed acyclic graph (DAG) shown in figure 7.1. The key idea is to treat the features obtained by preprocessing as latent variables. The link between ρ and $\mathcal{X}$ represents preprocessing, which must be carried out in a Bayesian

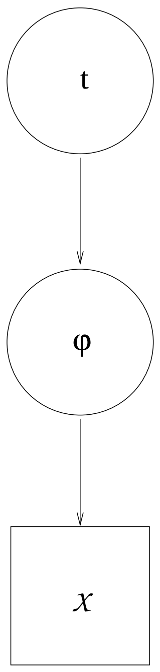

Figure 7.1: A directed acyclic graph for the hierarchical model. We use t to denote the unknown state variable of interest. The node ρ is a latent (unobserved) variable representing features from preprocessing and $\mathcal{X}$ denotes a segment of a time series. Circles indicate latent variables, whereas the square indicates that $\mathcal{X}$ is an observed quantity.

setting. The model used in this section for preprocessing is the lattice filter representation of an auto regressive (AR) model that was derived in chapter 4.

So far approaches that used a probabilistic structure similar to the DAG in figure 7.1 have focused on marginalizing out input uncertainties. The main emphasis in [Wri98] and [DS95], for instance, is to derive a predictive distribution for regression models where input uncertainty is taken into account. In a Bayesian sense, this approach is the only way of consistent reasoning if perfect knowledge of inputs is not available. We provide a similar analysis for classification problems. However our approach goes further than that.

- We allow for different uncertainties at different inputs. This leads to predictions that are dominated by inputs that are more certain. Using generative models to link t and ρ, the model provides information about the *true* input values of less certain sensors. Using several sensors will lead to some kind of "Bayesian sensor fusion". As has been shown in chapter 6, the use of a generative model gives us the additional benefit that we can use unlabeled data for parameter inference as well.

- Another extension of the work reported in [Wri98] and [DS95] is that we take the model uncertainty, inherent in all preprocessing methods, into account. Model uncertainty is represented by the posterior probability of the model conditional on the corresponding data segment, $\mathcal{X}$. Consideration of model uncertainties also allows a treatment of *missing inputs*[1].

We will show in the following sections that marginalizing out latent variables in a static DAG indeed performs sensor fusion such that predictions depend more on certain information. Equipped with this theoretical insight, we propose a DAG and inference scheme that is well suited for time series classification. We assume a first order Markov dependency among class labels of interest and derive MCMC updates that sample from the joint probability distribution over latent variables and model coefficients. In the experimental section we apply these techniques to a synthetic problem and to the problem described in chapter 2. We show that probability estimates for sleep stages obtained from EEG signals with the proposed method are less affected by artifacts contaminating sleep EEG.

7.2 Spatial fusion

In our case the input uncertainty is a result of the limited accuracy of feature estimates as obtained by preprocessing. If we know that the model used in preprocessing is the *true* model that generated the time series $\mathcal{X}$, the approach taken by [Wri98] and [DS95] would be sufficient. However, as was argued in chapter 4, we do not know whether the model used during preprocessing is the *true* one and we must consider feature as well as model uncertainty. Thus inference must be carried out with a DAG that allows for both feature *and* model uncertainty. Sensor fusion will only take place if different sensors are conditionally dependent on the state of interest. In order to take this into account, we have to extend the DAG structure[2] as shown in figure 7.2. We introduce binary indicator variables, I_a and I_b, that control

[1]It suffices to set the posterior probability of all lattice filter stages for *missing inputs* to zero.

[2]Another way of respecting both uncertainties would be to use a hierarchical sampling scheme that generates the *marginal* uncertainty of feature values. This could be achieved by first sampling I_a and I_b from their posterior probabilities. Given the indicator is 1, we would then sample from the posterior densities $p(\rho_a|\mathcal{X}_a)$ or $p(\rho_b|\mathcal{X}_b)$, respectively. Given the indicator is 0, we would sample from the prior $p(\rho_a)$ or $p(\rho_b)$, respectively. However, this approach is suboptimal, because it neglects information about the latent

the conditional dependency of the observed data segments, $\mathcal{X}_a$ and $\mathcal{X}_b$, on the latent variables ρ_a and ρ_b, respectively. The DAG in figure 7.2 illustrates

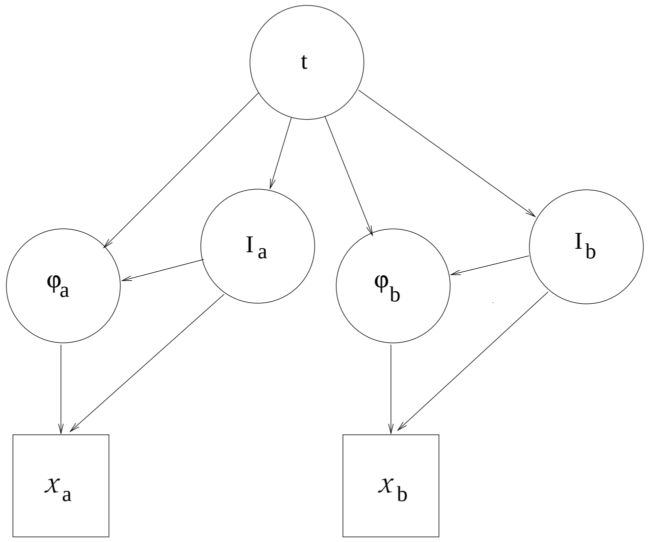

Figure 7.2: A directed acyclic graph that captures both parameter uncertainty as well as model uncertainty in preprocessing. We use t to denote the unknown state variable of interest. Both ρ_a and ρ_b denote latent variables representing the unknown values of the features that are estimated by preprocessing. The corresponding segments of the time series are denoted by $\mathcal{X}_a$ and $\mathcal{X}_b$. The latent binary indicator variables I_a and I_b control the dependency of the data segments on the latent variables.

the dependency between a state variable t, latent variables, ρ_a and ρ_b, and the corresponding segments of a time series, $\mathcal{X}_a$ and $\mathcal{X}_b$. The dependency is controlled by two model indicator variables, I_a and I_b, one for each sensor. Both models, I_a and I_b, representing particular stages of a lattice filter, are *probable* explanations of the corresponding time series $\mathcal{X}_a$ and $\mathcal{X}_b$.

During preprocessing we allow for two possible explanations of each segment of the time series. With probability $P(I_a|\mathcal{X}_a)$, the latent variable ρ_a is conditional on both the time series[3] $\mathcal{X}_a$ *and* the state variable t. However, with probability $1-P(I_a|\mathcal{X}_a)$, $\mathcal{X}_a$ *is pure white noise and does not provide any information about* ρ_a. In this case, we have to condition on t only. Loosely

feature variables provided via the latent state variable t. The approach proposed here does not have this problem.

[3]The DAG in figure 7.2 shows $\mathcal{X}_a$ conditionally dependent on ρ_a. But directions of arrows can always be inverted via Bayes theorem and that is what we do when we are interested in the value of ρ_a.

speaking, we deal with a problem of "probably missing values". This interpretation of figure 7.2 tells us that we need to use the following definition of the conditional probability density of $\mathcal{X}_a$:

$$p(\mathcal{X}_a|\rho_a, I_a) = \begin{cases} p(\mathcal{X}_a|\rho_a, I_a \equiv 1) \\ p(\mathcal{X}_a|I_a \equiv 0) \end{cases} \tag{7.1}$$

Depending on the value of I_a, we introduce a conditional independence between the data, $\mathcal{X}_a$, and the true feature value, ρ_a. Obviously this independence is not seen in the DAG in figure 7.2. When changing indices, we get the same statements for the latent variable ρ_b.

As already mentioned, we want to predict the belief of state t conditional on *all* available information. This is the observed time series $\mathcal{X}_a$ and $\mathcal{X}_b$ as well as all training data $\mathcal{D}$ observed so far. Although we will not state this explicitly, *all* beliefs are also conditional on training data $\mathcal{D}$. This is a requirement that humans apply intuitively. Whenever a clinical expert wants to monitor a new recording of some biological time series he has some prior expectations about the range they should use.

In order to provide deeper insight into the model illustrated in figure 7.2, we will infer both the posterior probability $P(t|\mathcal{X}_a, \mathcal{X}_b)$ and the posterior probability density $p(\rho_a|\mathcal{X}_a, \mathcal{X}_b)$. Using Bayes theorem, we express $P(t)p(\rho_a|t)$ as $p(\rho_a)P(t|\rho_a)$ and $P(t)p(\rho_b|t)$ as $p(\rho_b)P(t|\rho_b)$. Conditioning on $\mathcal{X}_a$ and $\mathcal{X}_b$, the DAG in figure 7.2 implies:

$$\begin{gathered} p(t, \rho_a, I_a, \rho_b, I_b|\mathcal{X}_a, \mathcal{X}_b) = \\ \frac{P(t|\rho_a)P(t|\rho_b)p(\rho_a|I_a, \mathcal{X}_a)p(I_a|\mathcal{X}_a)p(\rho_b|I_b, \mathcal{X}_b)p(I_b|\mathcal{X}_b)p(\mathcal{X}_a)p(\mathcal{X}_b)}{P(t)p(\mathcal{X}_a, \mathcal{X}_b)}. \end{gathered} \tag{7.2}$$

As neither variable on the left side of the conditioning bar in (7.2) is observed, we use marginal inference. We obtain $P(t|\mathcal{X}_a, \mathcal{X}_b)$ by plugging (7.1) into (7.2) and integrating out ρ_a, I_a, ρ_b and I_b and get

$$\begin{aligned}
P(t|\mathcal{X}_a, \mathcal{X}_b) = \frac{1}{C}\frac{1}{P(t)}\Bigg(& P(I_a \equiv 1|\mathcal{X}_a)\Bigg[P(I_b \equiv 1|\mathcal{X}_b) \qquad (7.3)\\
& \int_{\rho_a=-\infty}^{\infty} P(t|\rho_a)p(\rho_a|\mathcal{X}_a)d\rho_a \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b)p(\rho_b|\mathcal{X}_b)d\rho_b \\
+ \;& P(I_b \equiv 0|\mathcal{X}_b) \int_{\rho_a=-\infty}^{\infty} P(t|\rho_a)p(\rho_a|\mathcal{X}_a)d\rho_a \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b)p(\rho_b)d\rho_b\Bigg] \\
+ \;& P(I_a \equiv 0|\mathcal{X}_a)\Bigg[P(I_b \equiv 1|\mathcal{X}_b) \\
& \int_{\rho_a=-\infty}^{\infty} P(t|\rho_a)p(\rho_a)d\rho_a \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b)p(\rho_b|\mathcal{X}_b)d\rho_b \\
+ \;& P(I_b \equiv 0|\mathcal{X}_b) \int_{\rho_a=-\infty}^{\infty} P(t|\rho_a)p(\rho_a)d\rho_a \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b)p(\rho_b)d\rho_b\Bigg]\Bigg),
\end{aligned}$$

where $p(\rho_a)$ and $p(\rho_b)$ are the unconditional priors observed on the training data and C is a normalization constant. The influence of feature uncertainty on the probability of the state t is most easily seen if we assume perfect knowledge of the preprocessing model. In this case equation (7.3) gives

$$P(t|\mathcal{X}_a, \mathcal{X}_b) \propto \int_{\rho_a=-\infty}^{\infty} P(t|\rho_a)p(\rho_a|\mathcal{X}_a)d\rho_a \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b)p(\rho_b|\mathcal{X}_b)d\rho_b.$$

We may interpret this expression such that the probability $P(t|\mathcal{X}_a, \mathcal{X}_b)$ is dominated by those variables whose posterior allows to distinguish between different class labels of t. At the first glimpse this expression seems counter-intuitive, since we expect those variables to dominate that are known with higher accuracy. However, perfect knowledge of a variable does not help, if its value suggests equal probability for different classes. Thus, although known with less precision, a variable might dominate, if it allows better discrimination on average. Another question is, how a variable contributes, given that we know that the corresponding model has *not* generated the time series. This renders either $I_a \equiv 0$ or, for the other variable, $I_b \equiv 0$. The corresponding contribution in (7.3) is

$$\int_{\rho_a=-\infty}^{\infty} P(t|\rho_a)p(\rho_a)d\rho_a,$$

which is equivalent to the prior probabilities for class $P(t)$. This is an intuitive result, since without any knowledge of the variables we can still predict that each class t occurs according to its prior probability.

Similar manipulations lead to the posterior density of the latent variable ρ_a.

$$
\begin{aligned}
p(\rho_a|\mathcal{X}_a, \mathcal{X}_b) = \frac{1}{C}\Bigg(& P(I_a \equiv 1|\mathcal{X}_a) \qquad (7.4)\\
& \Bigg(P(I_b \equiv |\mathcal{X}_b) \sum_t \left[\frac{P(t|\rho_a)}{P(t)} p(\rho_a|\mathcal{X}_a) \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b) p(\rho_b|\mathcal{X}_b) d\rho_b \right] \\
+ \quad & P(I_b \equiv 0|\mathcal{X}_b) \sum_t \left[\frac{P(t|\rho_a)}{P(t)} p(\rho_a|\mathcal{X}_a) \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b) p(\rho_b) d\rho_b \right] \Bigg) \\
+ \quad & P(I_a \equiv 0|\mathcal{X}_a) \\
& \Bigg(P(I_b \equiv |\mathcal{X}_b) \sum_t \left[\frac{P(t|\rho_a)}{P(t)} p(\rho_a) \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b) p(\rho_b|\mathcal{X}_b) d\rho_b \right] \\
+ \quad & P(I_b \equiv 0|\mathcal{X}_b) \sum_t \left[\frac{P(t|\rho_a)}{P(t)} p(\rho_a) \int_{\rho_b=-\infty}^{\infty} P(t|\rho_b) p(\rho_b) d\rho_b \right] \Bigg) \Bigg)
\end{aligned}
$$

If we replace marginalization over ρ_b with marginalization over ρ_a, equation (7.4) immediately gives the corresponding expression for the posterior density over ρ_b.

Although using *all* information is the only consistent way of reasoning, there is a point where our analysis will not follow this requirement. In preprocessing, when we infer the posterior distribution over reflection coefficients and the posterior probability of the model, we cannot afford to do so. If we know t, then $p(\rho_a|t)$ and $p(\rho_b|t)$ provide some information about ρ_a and ρ_b, respectively. But even if t is unknown, the links $p(\rho_a|t)$ and $p(\rho_b|t)$ allow for information flow between ρ_a and ρ_b. Hence, in both situations we have information about ρ_a and ρ_b that goes beyond the "stability prior" adopted in chapter 4. However, the problem is that using this information, we cannot derive any expressions analytically, as was done in preprocessing. Instead, we would have to resort to MCMC techniques for the entire analysis *including preprocessing*. Even with modern computers, this is still an intractable procedure. Although this is an approximation, we will not loose much when using the uniform "stability prior". The usual settings in preprocessing will lead to posterior densities over coefficients that, compared with the priors, are sharply peaked around the most probable value[4]. In this case, using either a uniform prior in the range $[-1, 1]$, or, the more informative prior provided via t, does not make a big difference.

[4]As can be seen from equation (4.15) in chapter 4, this is just a matter of the number of samples used to estimate the feature values.

As a last step, it remains to provide an abstract formulation of an inference procedure for the model coefficients. In order to make life easier, we assume that we are given only labeled samples. A generalization to include also unlabeled samples is provided in the next section. Conventional model inference would condition on $\mathcal{A} = \{\rho_{a,i} \forall i\}$, $\mathcal{B} = \{\rho_{b,i} \forall i\}$ and $\mathcal{T} = \{t_i \forall i\}$ provided as "training samples". Denoting all model coefficients together by $\underline{\varphi}$, inference would lead to $p(\underline{\varphi}|\mathcal{A}, \mathcal{B}, \mathcal{T})$. In our setting, the $\rho_{a,i}$'s and the $\rho_{b,i}$'s are latent variables, and we cannot condition on them. Instead we would like to condition on the corresponding $\mathcal{X}_{a,i}$'s and $\mathcal{X}_{b,i}$'s. Such conditioning cannot be done directly. Assuming independence of observations allows us to formulate a likelihood function as the product of joint densities as are imposed by the DAG in figure 7.2. We get

$$\begin{aligned}
& p(\mathcal{T}, \mathcal{A}, I_A, \mathcal{X}_A, \mathcal{B}, I_B, \mathcal{X}_B|\underline{\varphi}) = \qquad (7.5)\\
= & \prod_i p(t_i, \rho_{a,i}, I_{a,i}, \mathcal{X}_{a,i}, \rho_{b,i}, I_{b,i}\mathcal{X}_{b,i}|\underline{\varphi})\\
= & \prod_i \frac{P(t_i|\rho_{a,i})P(t_i|\rho_{b,i})}{P(t_i)} p(\rho_{a,i}|I_{a,i,}\mathcal{X}_{a,i})p(I_{a,i}|\mathcal{X}_{a,i})p(\rho_{b,i}|I_{b,i}, \mathcal{X}_{b,i})p(I_{b,i}|\mathcal{X}_{b,i})
\end{aligned}$$

where we have used $I_A = \{I_{a,i} \forall i\}$, $\mathcal{X}_A = \{\mathcal{X}_{a,i} \forall i\}$, $I_B = \{I_{b,i} \forall i\}$ and $\mathcal{X}_B = \{\mathcal{X}_{b,i} \forall i\}$. We get the posterior over model coefficients $\underline{\varphi}$ by multiplication with the prior $p(\underline{\varphi})$ and marginalizing out all $\rho_{a,i}$, $I_{a,i}$, $\rho_{b,i}$ and $I_{b,i}$. This gives

$$\begin{aligned}
& p(\underline{\varphi}|\mathcal{T}, \mathcal{X}_A, \mathcal{X}_B) \propto \qquad (7.6)\\
\propto & \; p(\underline{\varphi}) \prod_i \frac{1}{P(t_i)} \Bigg(P(I_{a,i} \equiv 1|\mathcal{X}_{a,i}) \Bigg[P(I_{b,i} \equiv 1|\mathcal{X}_{b,i})\\
& \int_{\rho_{a,i}=-\infty}^{\infty} P(t_i|\rho_{a,i})p(\rho_{a,i}|\mathcal{X}_{a,i}) \int_{\rho_{b,i}=-\infty}^{\infty} P(t_i|\rho_{b,i})p(\rho_{b,i}|\mathcal{X}_{b,i})\\
+ & \; P(I_{b,i} \equiv 0|\mathcal{X}_{b,i}) \int_{\rho_{a,i}=-\infty}^{\infty} P(t_i|\rho_{a,i})p(\rho_{a,i}|\mathcal{X}_{a,i}) \int_{\rho_{b,i}=-\infty}^{\infty} P(t_i|\rho_{b,i})p(\rho_{b,i}) \Bigg]\\
+ & \; P(I_{a,i} \equiv 0|\mathcal{X}_{a,i}) \Bigg[P(I_{b,i} \equiv 1|\mathcal{X}_{b,i})\\
& \int_{\rho_{a,i}=-\infty}^{\infty} P(t_i|\rho_{a,i})p(\rho_{a,i}) \int_{\rho_{b,i}=-\infty}^{\infty} P(t_i|\rho_{b,i})p(\rho_{b,i}|\mathcal{X}_{b,i})\\
+ & \; P(I_{b,i} \equiv 0|\mathcal{X}_{b,i}) \int_{\rho_{a,i}=-\infty}^{\infty} P(t_i|\rho_{a,i})p(\rho_{a,i}) \int_{\rho_{b,i}=-\infty}^{\infty} P(t_i|\rho_{b,i})p(\rho_{b,i}) \Bigg] \Bigg),
\end{aligned}$$

where we have also made the marginal sums over $I_{a,i}$ and $I_{b,i}$ explicit. As we still need to normalize, expression (7.6) is just proportional to the posterior. In general, neither the integrals nor the normalization constants in (7.3), (7.4) or (7.6) will be analytically tractable. In all practical problems we will have to apply MCMC methods.

The DAG used in this section was used to illustrate that marginalizing out variable and model uncertainties leads to predictions that are dominated by such information that allows for reliable discrimination. As we are interested in classification of adjacent segments of a time series, we can improve on the DAG shown in figure 7.2, if we also allow for temporal dependencies among class labels. In the following section we propose the simplest such model, which exhibits a first order Markov dependency among class labels. We propose Bayesian inference of such a model, which is done by sampling from the joint probability density over latent variables and model coefficients.

7.3 Spatio-temporal fusion

A DAG structure that allows spatio-temporal sensor fusion is obtained by imposing conditional dependencies among adjacent state variables and to spatially different sensors. Sensor fusion is then achieved very naturally by treating this DAG within the Bayesian framework. Bayesian preprocessing will assess both model and coefficient uncertainties. Marginalizing out these uncertainties will lead to beliefs about states that are less affected by less reliable information. Such an approach can go even further; it allows one to infer the expected feature values. The obtained expectations will differ from the values obtained from preprocessing alone if the correspoinding segment was unreliable.

7.3.1 A simple DAG structure

This subsection proposes a DAG structure that allows spatio-temporal sensor fusion. Measured in terms of model complexity (number of free parameters), we aim at a simple solution. Assuming that we want to solve a classification task, we regard the class labels as the state of interest. The simplest case will assume a first order Markov dependency among these state variables. Furthermore, we will also assume conditional independence between all latent variables depending on each of the state variables. Figure 7.3 shows a DAG structure imposed by these assumptions. Training data can contain labeled as well as unlabeled segments of a time series. We represent unobserved states with circles and the observed states with squares. Such relations will be found during model inference. The structure of the DAG in figure 7.3 is similar to hidden Markov models, however, the states are only partially hidden. For our purpose, such a partial HMM is more appropriate than an HMM, because HMM's are fully unsupervised. However, we have partial knowledge about class labels and consistency arguments tell us that we must use the labels. During prediction, all states are unknown and the model operates similar to an ordinary HMM.

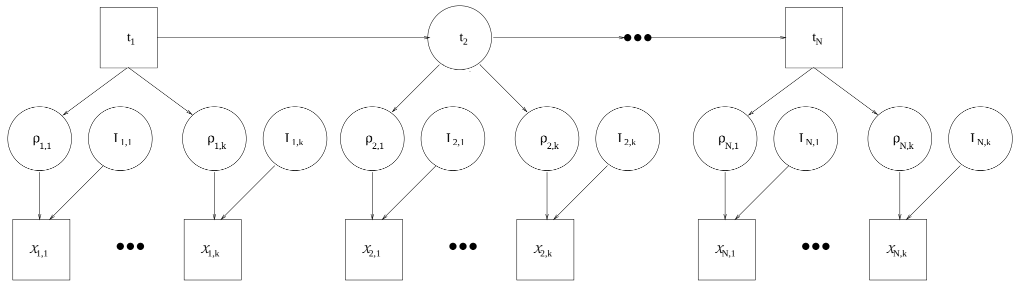

Figure 7.3: A directed acyclic graph for spatio-temporal sensor fusion. The DAG assumes a first order Markov dependency among states and, conditional on these, independence of the latent variables $\rho_{i,j}$. We use t_i to denote the unknown state variables of interest. The $\rho_{i,j}$ are latent variables, each representing one feature from preprocessing. Finally $\mathcal{X}_{i,j}$ denote the corresponding segments of a time series, and $I_{i,j}$ denote latent indicator variables that allow us to consider model uncertainties.

We have already decided about the latent variables $\rho_{i,j}$ and indicator variables $I_{i,j}$. They are stages in an AR-lattice filter, and $\mathcal{X}_{i,j}$ are the corresponding segments of a time series. It remains to decide about the model details between the state variables t_i and the latent variables $\rho_{i,j}$. This part of the model has to meet two requirements.

- The model should allow inference of model coefficients from such segments of time series $\mathcal{X}_{i,j}$, where the corresponding state t_i is unknown.
- Furthermore, model inference as well as predictions require the estimation of the conditional probability density function, $p(\rho_{i,j}|\{\mathcal{X}_{\iota,\zeta}\forall\iota,\zeta \text{ being relevant }\}, \underline{t}_i, \overline{t}_i)$, over latent variables, $\rho_{i,j}$. We will condition on all relevant information. During prediction, where all state variables are unknown, all segments, $\mathcal{X}_{\iota,\zeta}$, of the entire time series at all sensors are relevant. In this case $\underline{t}_i$ and $\overline{t}_i$ are the (certain) start and end states. During model inference, relevant information is provided by all segments $\mathcal{X}_{\iota,\zeta}$ between the observed states that form the DAG and the states at these time instances.

Both requirements are met if we use a *generative* model (i.e. we model the probability density functions $p(\rho_{i,j}|t_i)$.). In our case the $\rho_{i,j}$'s are continuous variables and the generative model will be implemented as a mixture of normals

$$p(\rho_{i,j}|t_i) = \sum_{d=1}^{D} P(d|t_i) p(\rho_{i,j}|\underline{\Theta}_d), \tag{7.7}$$

where $P(d|t_i)$ are the prior allocation probabilities of mixture component d conditional on state t_i and $\underline{\Theta}_d$ are all coefficients of the d-th component. As the model is fixed, we carry on with the usual approach in a Bayesian analysis. We use an MCMC method which requires us to formulate a likelihood function, design a DAG that shows the relations during inference, specify convenient priors and, as a final step, formulate updates for an MCMC scheme.

7.3.2 A likelihood function for sequence models

As already mentioned, we are interested in a fully Bayesian treatment of the model. This can be done as soon as we are able to formulate a normalized likelihood function and priors. We are dealing with a sequence model, where the likelihood is usually (see [BB98], pp. 150) formulated via paths that are possible sequences of latent states

$$P(\mathcal{D}, \Pi|\underline{\varphi}) = P(t_1) \prod_j p(\rho_{1,j}|t_1) \prod_{i=2}^{N} \left(P(t_i|t_{i-1}) \prod_j p(\rho_{i,j}|t_i) \right). \tag{7.8}$$

In (7.8) we use $\mathcal{D}$ to denote a realization of all latent variables $\rho_{i,j}$ and Π to denote a state sequence (path) that could have generated $\mathcal{D}$. Contrary to conventional HMM's, where all states are hidden, in our case not all paths are possible. Each path in (7.8) has to visit all observed states and the likelihood function that applies in the case of a partial HMM has to marginalize over all these possible paths Π only. We formulate

$$P(\mathcal{D}, \mathcal{T}|\underline{\varphi}) = \sum_{\Pi} P(t_1) \prod_j p(\rho_{1,j}|t_1) \prod_{i=2^N} \left(P(t_i|t_{i-1}) \prod_j p(\rho_{i,j}|t_i) \right), \tag{7.9}$$

where $\mathcal{T}$ denotes the observed states. If several independent sequences are used to infer model coefficients, the overall likelihood is the product of several expressions like (7.9). In order to obtain a final expression of the likelihood of model coefficients we have to plug (7.7) into (7.9). It is evident that the resulting likelihood is highly nonlinear and parameter inference had to be done by carrying out Metropolis-Hastings updates. The conventional way to maximize likelihood functions like (7.9) is to apply the expectation maximization (EM) algorithm [DLR77], which was introduced for HMMs in [BPSW70]. The sampling algorithm proposed in the next subsection uses similar ideas. We use both Gibbs updates and Metropolis-Hastings updates to draw samples from the posterior distribution over model coefficients and latent variables.

7.3.3 An augmented DAG for MCMC sampling

The likelihood function associated with the probabilistic model in figure 7.3 is non-quadratic in the model coefficients and we have to use MCMC methods to infer them. As already indicated, we want to use Gibbs updates wherever possible. Only such variables that do not allow Gibbs moves will be updated with Metropolis-Hastings steps. Following the ideas of the EM procedure, we introduce *latent* variables, $d_{i,j}$, which indicate the kernel number the latent variables $\rho_{i,j}$ were generated from.

Figure 7.4 shows a DAG that results from the DAG in figure 7.3 when augmenting it with these latent allocation variables. Furthermore, the DAG in figure 7.4 contains all coefficients of the probabilistic model and the hyper-parameters of the corresponding priors. In order to keep the graph simple, figure 7.4 displays only one state variable. We also restrict the DAG to one latent variable, the others are indicated by dots.

The state variable t_i is either conditionally dependent on the prior probabilities of states, $\underline{P}$, or it depends on the transition probability $\underline{T}_{t_{i-1}}$. The former is true for state t_1 only, the latter is true for all other states. For both the transition probabilities, $\underline{T}_{t_{i-1}}$, and prior allocation probabilities, $\underline{W}_{t_i,j}$, we use a hierarchical Dirichlet-multinomial prior. This has the advantage of allowing for informative priors without introducing too much sensitivity. The price we pay is that both $\underline{\delta}_T$ and $\underline{\delta}_W$ must be sampled with single component Metropolis-Hastings updates. Both $\underline{T}_{t_{i-1}}$ and $\underline{W}_{t_i,j}$ are conditional on the state at particular time instances. In order to emphasize that inference of the coefficient values is independent of these time instances, we will use k to represent a state whenever inferring one of these variables.

The next part of the DAG, that is, the observation model of the latent variables $\rho_{i,j}$, is largely identical to the model we used in chapters 5 and 6. A similar DAG was also used by [RG97] for their one dimensional mixture of Gaussians analysis with varying number of kernels. We use an a-priori fixed number of kernels here. The component means, $\mu_{j,d}$, get a normal prior with mean ξ_j and variance κ_j^{-1}. Each component has its own precision (inverse variance) $\sigma_{j,d}^{-2}$. In order to avoid problems with singular solutions the variances are coupled with hyper-parameters α and β_j. The latter, β_j, has

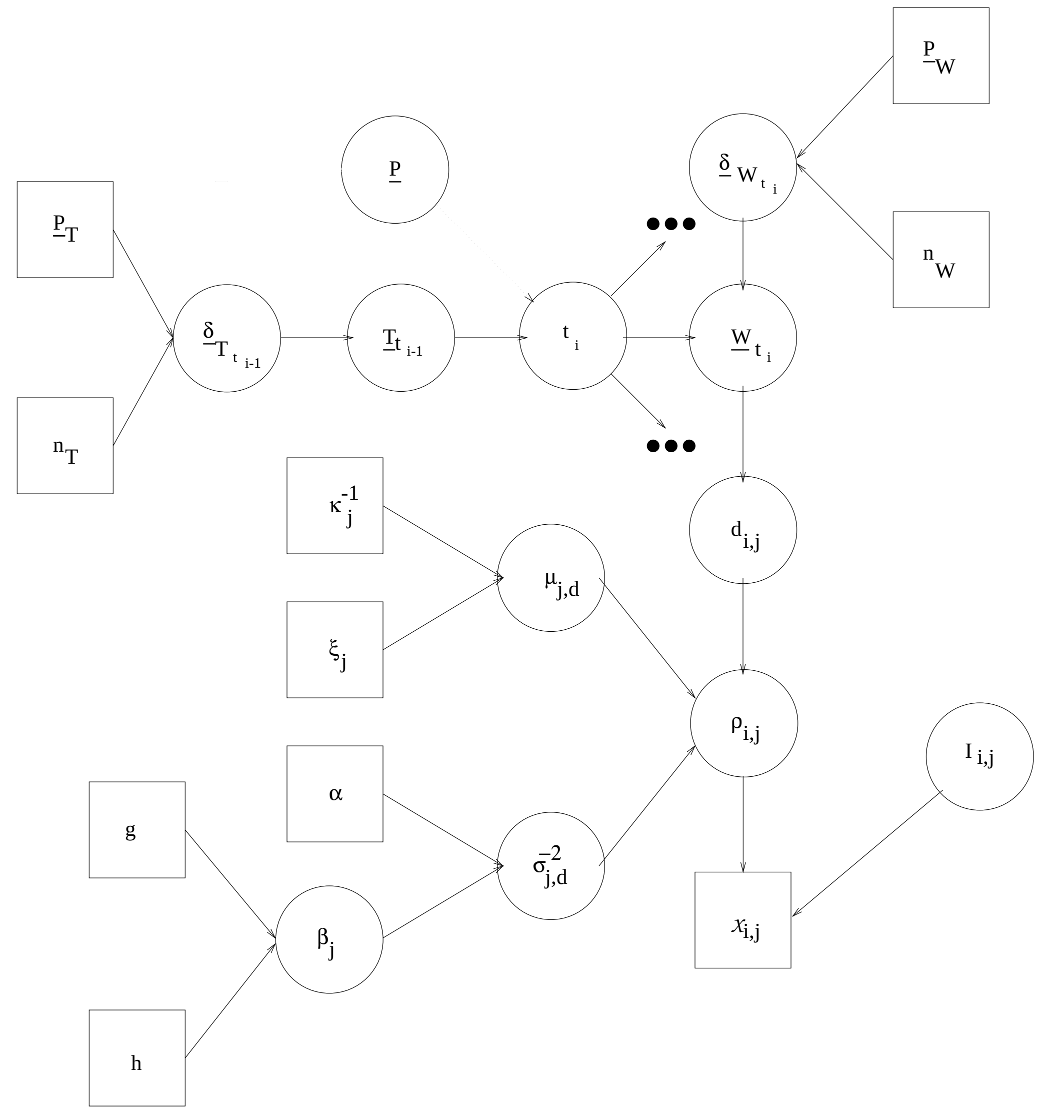

Figure 7.4: This figure shows the DAG underlying Gibbs sampling. In order to keep it simple, the DAG shows only one state variable t_i and only one of the latent variables $\rho_{i,j}$. The other variables are indicated by three dots. The dotted connection between the prior probability of states, $\underline{P}$, and t_i and the connection between the transition probability $\underline{T}_{t_{i-1}}$ and t_i are mutually exclusive. For the first state, t_1, the dotted connection is present, for all other states it is absent. As usual, square nodes denote observed quantities and circles are latent variables. However there are two exceptions: some of the t_i's as well as the $\mathcal{X}_{i,j}$'s are observed. A more detailed description can be found in the text.

itself a Gamma prior. This hierarchical prior specification allows again for informative priors without introducing large dependencies on the values of the hyper-parameters. The difference of our observation model to the one used in [RG97] is that the $\rho_{i,j}$ are latent variables. Both the model indicator $I_{i,j}$ and the corresponding segments of time series $\mathcal{X}_{i,j}$ are conditionally dependent on $\rho_{i,j}$.

7.3.4 A specification of priors

In order to be able to derive an MCMC scheme for sampling from the posterior distributions of model coefficients and latent variables, we need to specify the functional form, as well as parameters, of all priors from the DAG in figure 7.4. Gibbs sampling requires *full conditional distributions* that can be sampled from. The *full conditional distributions* are the distributions of model coefficients when conditioning on all other model coefficients, latent variables and data. Apart from the EM-like idea to introduce latent variables, tractable distributions will be obtained when using so-called *conjugate* priors, as discussed in [BS94].

In order to allow Gibbs updates for most of the parameters, we use the following prior specification:

- Each component mean, $\mu_{j,d}$, is given a Gaussian prior: $\mu_{j,d} \sim \mathcal{N}_1(\xi_j, \kappa_j^{-1})$.
- The inverse variance is given a Gamma prior: $\sigma_{j,d}^{-2} \sim \Gamma(\alpha, \beta_j)$.
- The hyper-parameter, β_j, gets a Gamma hyper-prior: $\beta_j \sim \Gamma(g, h)$.
- The mixing coefficients, $\underline{W}_k$, get a Dirichlet prior: $\underline{W}_k \sim \mathcal{D}(\delta_{W_k}^1, ..., \delta_{W_{t_i}}^D)$, with D denoting the number of kernels.
- The prior allocation counts, $\underline{\delta}_{W_k}$, get a multinomial prior: $\underline{\delta}_{W_k} \sim \mathcal{M}n(n_W, \underline{P}_W)$, with $k \in \{1, ..., K\}$ denoting a possible state between 1 and the number of states, K.
- The transition probabilities, $\underline{T}_k$, get a Dirichlet prior as well: $\underline{T}_k \sim \mathcal{D}(\delta_{T_k}^1, ..., \delta_{T_k}^K)$
- The prior transition counts, $\underline{\delta}_{T_k}$, get a multinomial prior: $\underline{\delta}_{T_k} \sim \mathcal{M}n(n_T, \underline{P}_T)$.

The quantitative settings are similar to those used in [RG97]. Values for α are between 1 and 2, g is usually between 0.2 and 1 and h is typically between $1/R_{max}^2$ and $10/R_{max}^2$, with R_{max} denoting the largest input range. The mean, μ_j, gets a Gaussian prior centered at the midpoint, ξ_j, with inverse variance $\kappa_j = 1/R_j^2$, where R_j is the range of the j-th input. The multinomial priors of the prior allocation counts and the prior transition counts are set up with equal probabilities for all counters. We set the number of preallocated samples, n_W, between 10 and 50 and n_T between 50 and 100. The larger n_W, the more certain we are that the samples from a particular class are distributed equally among the different Gaussian kernels. Increasing n_T leads to smaller differences among transition probabilities. This corresponds to a temporal smoothness assumption. The attentive reader might be surprised that neither the DAG in figure 7.4 nor the prior specification in this section contain any prior counts for the prior state probabilities $\underline{P}$. The reason why we do not need any prior counts for these probabilities is because we do not infer them directly. As is shown below, assuming ergodicity of the Markov chain[5], $\underline{P}$ is directly specified by the estimates of the transition probabilities $\underline{T}$.

7.3.5 MCMC updates of coefficients and latent variables

The prior specification proposed in the last subsection enables us to use mainly Gibbs updates. However, some of the variables need to be sampled via Metropolis-Hastings updates. The resulting sampling scheme is a single component MCMC sampler. The main difficulty is that we regard "input" variables as being *latent*. In other words, conventional approaches condition on the $\rho_{i,j}$'s, whereas our approach regards them as random variables which have to be updated as well.

The single component sampler uses updates from the full conditional distributions in (7.10). We use $\underline{\Theta}_{j,d}$ to denote the parameters of the d-th Gaussian and j-th latent variable. The sample index is i and t_i are the (partially) unobserved states. Of course we update only unobserved state variables. The full conditional distribution of the (latent) kernel allocations, $d_{i,j}$, depends on the prior allocation probabilities, $\underline{W}_{t_i}$, and on the latent variables $\rho_{i,j}$. After having obtained latent states and kernel allocations we are able to update the model coefficients. Several counters are needed: $l_{k,j}^d$ is the number of $\rho_{i,j}$'s allocated by the d-th component and state k; m_k are the number of transitions from state k to state i; finally we use $n_{d,j}$ as counter for all $\rho_{i,j}$'s allocated to the d-th kernel and $\bar{a}_{d,j}$ as the corresponding sample

[5]The terminology at this stage is somewhat confusing: here we refer to the Markov chain in the model.

mean. During updates of the $\rho_{i,j}$'s, we use $\hat{\rho}_{i,j}^{\mathcal{X}_{i,j}}$ to denote the most probable value of $\rho_{i,j}$ conditional on $\mathcal{X}_{i,j}$ and $(\sigma_{i,j}^{\mathcal{X}_{i,j}})^2$ as corresponding variance. These estimates, as well as $P(I_{i,j}|\mathcal{X}_{i,j})$, are obtained from Bayesian preprocessing.

$$\begin{array}{rcl}
P(t_1|...) & = & \dfrac{\underline{P}_{t_1}\underline{T}_{t_2,t_1}\prod_j p(\rho_{1,j}|t_1)}{\sum_{t_1}\underline{P}_{t_1}\underline{T}_{t_2,t_1}\prod_j p(\rho_{1,j}|t_1)} \\
P(t_{i\neq 1}|...) & = & \dfrac{\underline{T}_{t_i,t_{i-1}}\underline{T}_{t_{i+1},t_i}\prod_j p(\rho_{i,j}|t_1)}{\sum_{t_i}\underline{T}_{t_i,t_{i-1}}\underline{T}_{t_{i+1},t_i}\prod_j p(\rho_{i,j}|t_1)} \\
t_i & \sim & \mathcal{M}n(1,\{P(t_i|...)\forall t_i\}) \\
P(d_{i,j}|...) & = & \dfrac{\underline{W}_{t_i,j,d}\,p(\rho_{i,j}|\underline{\Theta}_{j,d})}{\sum_d \underline{W}_{t_i,j,d}\,p(\rho_{i,j}|\underline{\Theta}_{j,d})} \\
d_{i,j} & \sim & \mathcal{M}n(1,\{P(d_{i,j}|...)\forall d_{i,j}\}) \\
\underline{W}_{k,j} & \sim & \mathcal{D}(\delta^1_{W_{k,j}}+l^1_{k,j},...,\delta^K_{W_{k,j}}+l^D_{k,j}) \\
\underline{\delta}_{W_{k,j}} & \sim & \displaystyle\prod_{d=1}^{D}\frac{\left(\underline{P}^d_W\underline{W}^d_{k,j}\right)^{\delta^d_{W_{k,j}}}}{\delta^d_{W_{k,j}}!(\delta^d_{W_{k,j}}-1)!} \\
\underline{T}_k & \sim & \mathcal{D}(\delta^1_{T_k}+m^1_k,...,\delta^1_{T_k}+m^K_k) \\
\underline{\delta}_{T_k} & \sim & \displaystyle\prod_{i=1}^{K}\frac{(P^i_T T^i_k)^{\delta^i_{t_k}}}{\delta^i_{t_k}!(\delta^i_{t_k}-1)!} \\
\mu_{d,j} & \sim & \mathcal{N}((n_{d,j}\sigma^{-2}_{d,j}+\kappa_j)^{-1}(n_{d,j}\sigma^{-2}_{d,j}\bar{a}_{d,j}+\kappa_j\xi_j),(n_{d,j}\sigma^{-2}_{d,j}+\kappa_j)^{-1}) \\
\sigma^{-2}_{d,j} & \sim & \Gamma(\alpha+\dfrac{n_{d,j}}{2},\beta_j+\dfrac{1}{2}\displaystyle\sum_{\rho_{i,j}\forall i d_i=d}(\rho_{i,j}-\mu_{d,j})^2) \\
\beta_j & \sim & \Gamma(g+D\alpha,h+\displaystyle\sum_d\sigma^{-2}_{d,j}) \\
I_{i,j} & \sim & P(I_{i,j}|\mathcal{X}_{i,j}) \\
\rho_{i,j} & \sim & \begin{cases}\forall I_{i,j}\equiv 1: P(t_i|\rho_{i,j})\mathcal{N}(\rho_{i,j};\hat{\rho}^{\mathcal{X}_{i,j}}_{i,j},(\sigma^{\mathcal{X}_{i,j}}_{i,j})^2) \\ \forall I_{i,j}\equiv 0: p(\rho_{i,j}|t_i)\end{cases}
\end{array} \tag{7.10}$$

Except for $\underline{\delta}_{W_{k,j}}$, $\underline{\delta}_{T_k}$ and $\rho_{i,j}$ all variables in (7.10) can be sampled from. Details on how to generate random numbers from all necessary distributions can be found in [Rip87]. The exceptions can be updated using single component Metropolis-Hastings steps. For $\underline{\delta}_{W_{k,j}}$ and $\underline{\delta}_{T_k}$, we use the same Metropolis-Hastings scheme, which needs to be discussed only for $\underline{\delta}_{W_{k,j}}$. We propose a modified $\underline{\delta}'_{W_{k,j}} = \underline{\delta}_{W_{k,j}} + \underline{\Delta}$ with $\underline{\Delta}$ constructed in a way that the sum over all prior counts remains equal to n_W. The move is accepted according to the ratio $r = p(\underline{\delta}'_{W_{k,j}}|...)/p(\underline{\delta}_{W_{k,j}}|...)$, with the full conditional distributions being equal to the corresponding right side in (7.10). The situation with $\rho_{i,j}$ is slightly more difficult as it requires a hierarchical scheme with two independent sets of observations of $\rho_{i,j}$. The first set is updated by drawing from $P(t_i|\rho_{i,j})\mathcal{N}(\rho_{i,j}; \hat{\rho}_{i,j}^{\mathcal{X}_{i,j}}, (\sigma_{i,j}^{\mathcal{X}_{i,j}})^2)$. We draw new samples from the Gaussian and compare the acceptance ratios $r = P(t_i|\rho'_{i,j})/P(t_i|\rho_{i,j})$. In a second step, we generate the observations of the $\rho_{i,j}$ that will be used in the next round of the sampler. All observations where $I_{i,j} \equiv 1$ are taken from the first set. If $I_{i,j} \equiv 0$, the $\rho_{i,j}$'s are conditional on t_i and $I_{i,j}$ and due to the uniform prior used in preprocessing, we have to sample from $p(\rho_{i,j}|t_i)$.

As already mentioned in the last section, under some conditions, the prior probabilities of states, $\underline{P}$, can be obtained directly from the transition probabilities, $\underline{T}$. Assuming a stationary Markov chain, we get:

$$\underline{P}_{i+1} = \underline{T}\underline{P}_i. \tag{7.11}$$

Linear algebra tells us that (7.11) is true if and only if $\underline{P}$ is equal to the eigenvector of $\underline{T}$ corresponding to the eigenvalue 1 that must exist. Hence we obtain $\underline{P}$ from $\underline{T}$ as (normalized) eigenvector corresponding to the eigenvalue 1.

The entire MCMC scheme starts by drawing initial model parameters from their priors. We initialize the $\rho_{i,j}$'s with samples drawn from the posterior found in preprocessing, $\mathcal{N}(\rho_{i,j}; \hat{\rho}_{i,j}^{\mathcal{X}_{i,j}}, (\sigma_{i,j}^{\mathcal{X}_{i,j}})^2)$, and all unobserved states t_i with samples drawn from the prior probabilities $P(t_i)$. In order to allow the Markov chain to converge, we draw several thousand samples from the posterior. The first half is usually regarded as burn in and not used for predictions.

7.3.6 Sensor fusion during predictions

In order to get consistent estimates of beliefs of states, predictions have to be marginalized over $\rho_{i,j}$ and $I_{i,j}$, as well. The integrals need to be solved numerically. We use all samples drawn from the posterior in order to allow the $\rho_{i,j}$'s of the test data to converge. All sample expectations are then taken after allowing for a burn in period. Apart from beliefs about states, we can also obtain expectations from all latent variables most interestingly from the $\rho_{i,j}$'s.

Predictions are based on an approximation of the posterior distribution of states, t_i, latent allocations, $d_{i,j}$, and latent variables, $\rho_{i,j}$. We initialize the states t_i by a sample drawn from the first estimate of the prior probabilities $\underline{P}$. The initial $\rho_{i,j}$'s are drawn from the normal distribution $\mathcal{N}(\rho_{i,j}; \hat{\rho}_{i,j}^{\mathcal{X}_{i,j}}, (\sigma_{i,j}^{\mathcal{X}_{i,j}})^2)$, the coefficients of which were obtained by preprocessing. The coefficients are then updated using full conditional distributions for t_i, $d_{i,j}$ and $\rho_{i,j}$, which are identical to those formulated in (7.10). During each round of the sampler, we use the next sample from the Markov chain obtained during parameter inference. We are interested in obtaining the expected values of state probabilities and latent variables $\rho_{i,j}$. These are estimated as sample averages by summing over all values obtained after having allowed for a burn in period.

7.4 Experiments

The algorithm proposed in this section should improve classification if the time series to be classified is contaminated with white noise. In order to demonstrate that the proposed algorithm will improve results in such situations, we performed an experiment where a synthetic time series had to be classified. We also apply the algorithm to a real time series classification problem, where we interpret all night EEG recordings in the light of a 3-process model. We predict probabilities for 3 Rechtschaffen and Kales stages wake, REM and delta sleep (combined stages 3 and 4). Some of the recordings used as independent test cases contain artifact segments of different durations ranging form 1 up to 10 seconds length. Although not marked as artifacts in the routine scorings, consultations with clinical experts lead to the conclusion that these observations are clearly not physiologically plausible. Comparing the probabilities assigned to the lattice filter stages, when compared against a white noise explanation, reveals that the segments are white noise sequences. In particular we are interested in the behaviour of our algorithm when such segments appear within epochs of deep sleep. Classical approaches that condition on best estimates will predict stage wake with high probability. On the contrary we expect that the probabilities predicted with the suggested approach will ignore these artifact segments and instead rely on the information obtained from other sensors and from different time instances.

7.4.1 Classification of synthetic time series

In order to see whether integrating out all preprocessing uncertainty improves results of time series classification, we prepared a data set generated with two synthetic third-order AR models. The poles of the transfer function of the process generator have been set to $(0.9; -0.5 + j0.5; -0.5 - j0.5)$ and $(-0.9; 0.5 + j0.5; 0.5 - j0.5)$. As long as the time series was generated according to the first model, we give label 0. As soon as the time series is generated from the second model it is labeled 1.

The label sequence was obtained from a first order Markov model with the following transition probabilities:

- $P(t_{i+1} = 0|t_i = 0) = 0.95$.
- $P(t_{i+1} = 1|t_i = 0) = 0.05$.
- $P(t_{i+1} = 0|t_i = 1) = 0.05$.
- $P(t_{i+1} = 1|t_i = 1) = 0.95$.

According to these transition probabilities, we generated a sequence of length 1000 that is used for model inference. Each label corresponds to a segment of 100 samples of a time series generated from the corresponding AR-process. A short sequence of the time series and labels is shown in figure 7.5.

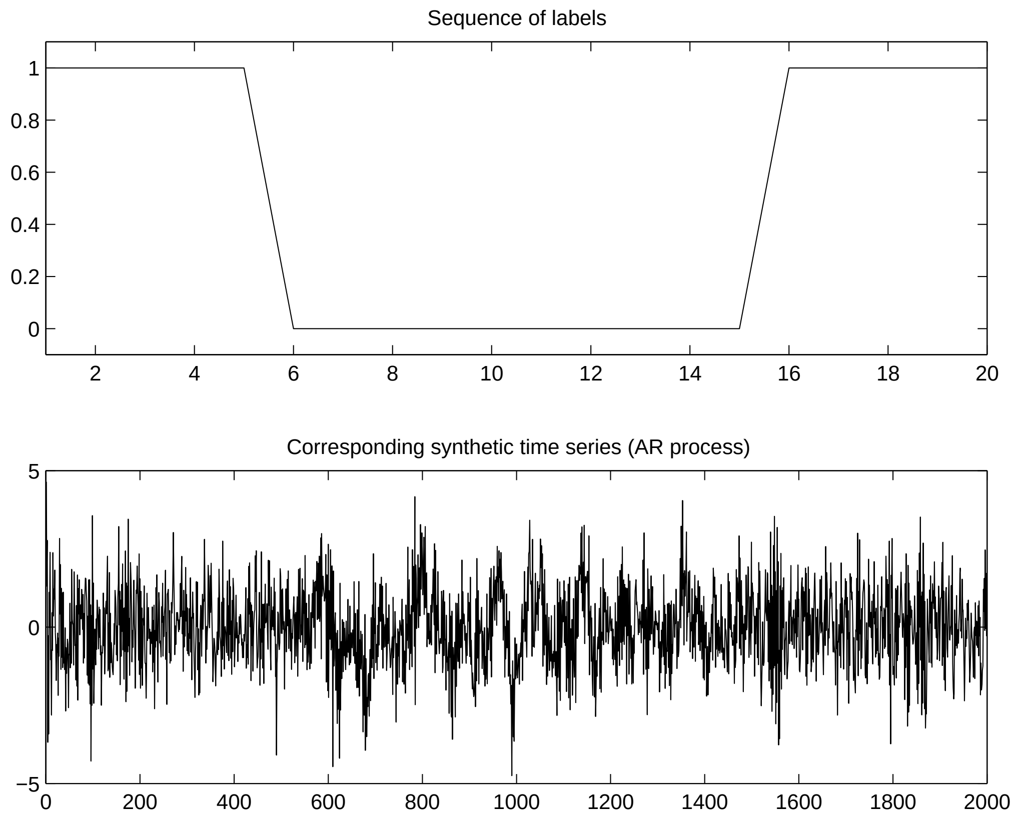

Figure 7.5: Labels and time series used in the synthetic experiment.

Using equations (4.14), (4.15) and (4.18) defined in chapter 4 , we extracted 3 reflection coefficients, the corresponding variances and model probabilities. Preprocessing was done with a 100-sample non-overlapping window.

The training sequence used to parameterize the model proposed above contained 1000 such inputs and the corresponding labels. Model inference is based on the MCMC updates formulated in (7.10) in subsection 7.3.5. We use only labeled data and draw 2000 samples from the posterior. The same data was used to train a generative classifier as proposed in chapter 6. This algorithm uses the most probable feature values only and serves as reference method.

In order to test the hypothesis that integrating out feature and model uncertainty will lead to better predictions, we took a test set with 4000 samples that was generated as described above. A fraction of segments was replaced with white noise. We used a firs- order Markov sequence with a probability for replacement of $P_r = 0.15$ to decide whether or not a particular segment of the clean time series is replaced with white noise. This ensures that we will have situations of different complexity. The length of the white noise segment as well as the position with respect to the switching times in the clean time series will vary. A short segment of the resulting time series is shown in figure 7.6, together with the original labels and an indicator where the contamination took place.

As was done for the training data, we extracted 3 reflection coefficients with the method proposed in chapter 4. The plots in figure 7.7 show the model probabilities in the third lattice filter stage before and after contamination.

A sequence of 500 predictions obtained from the generative classifier that uses most probable feature estimates is shown in figure 7.8. The trace of the probabilities obtained from noisy data contains many deviations from the probabilities obtained with clean data. It is interesting to see that the wrong predictions are obtained with rather high probabilities.

Predicting with the integrating approach proposed in this section requires to sample from the posterior distribution over latent feature variables and unobserved class labels. As described in subsection 7.3.6, we must allow for convergence of the Markov chain *also during predictions*. We start therefore simulating with the first sample obtained during model fitting. However, all expectations are only taken after allowing for the usual burn in period (1000 samples in this experiment). The probabilities predicted by our approach are shown in figure 7.9. Many of the misclassifications made by the the conventional approach can be corrected. Furthermore we see that the probabilities for classes are *moderated* when predicted from contaminated segments (e.g. in figure 7.9 close to sample 250).

In order to make the comparison of the classifications obtained from both methods for the contaminated time series easier, we show both probabilities together with the true labels in figure 7.10. We can see that integrating out

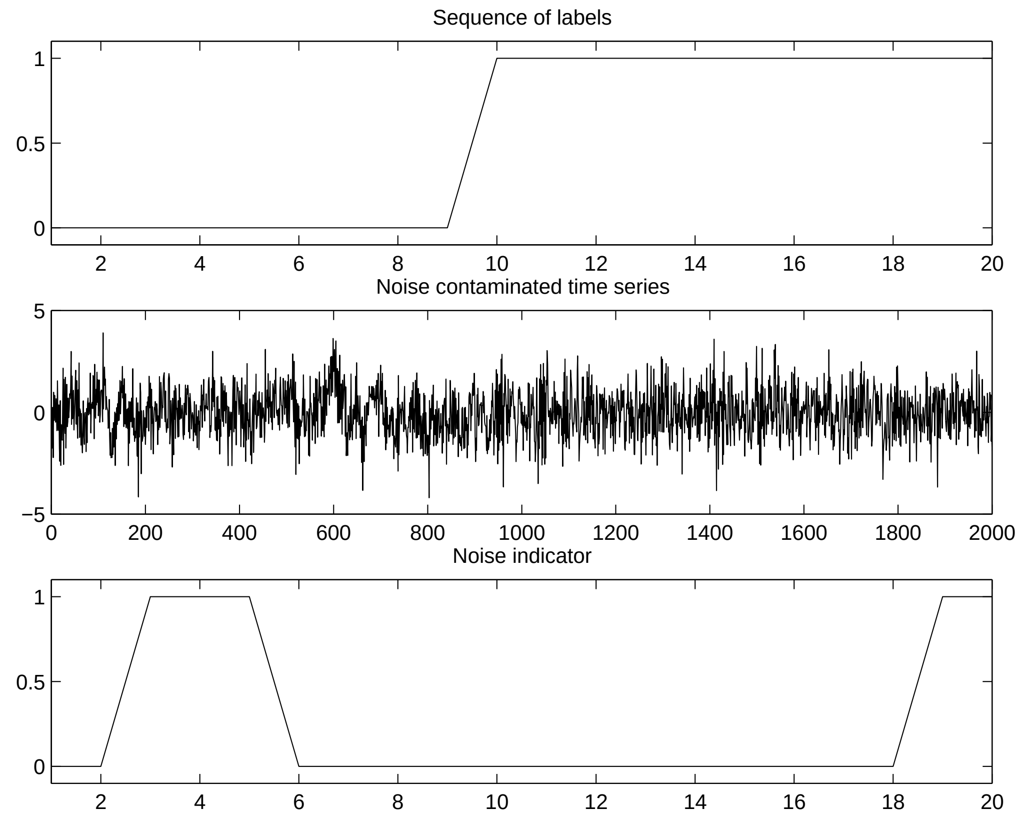

Figure 7.6: "True" labels and contaminated time series used as test data. The third trace in this figure shows where the original data have been replaced with white noise.

uncertainties leads to probabilities that are much less certain about class label when compared with the probabilities obtained from a conventional classifier. As soon as we condition on the feature values, we get wrong classifications that are made with rather high confidence.

The results of this experiment have also been analyzed quantitatively. For this purpose we have used all 4000 test cases. As can be seen in figure 7.9 there are some samples of clean data that are misclassified. Hence it seems advisable to investigate the behaviour of the proposed method using both clean and contaminated data.

We find that the proposed method misclassifies 8 of 4000 samples, which corresponds to a generalization performance of 99.8%. At the same time the alternative method has no problem in classifying *all* samples correctly. Hence the differences between both classifiers are determined by: $n_a = 8$, where n_a is the number of errors made by the integrating HMM classifier that were

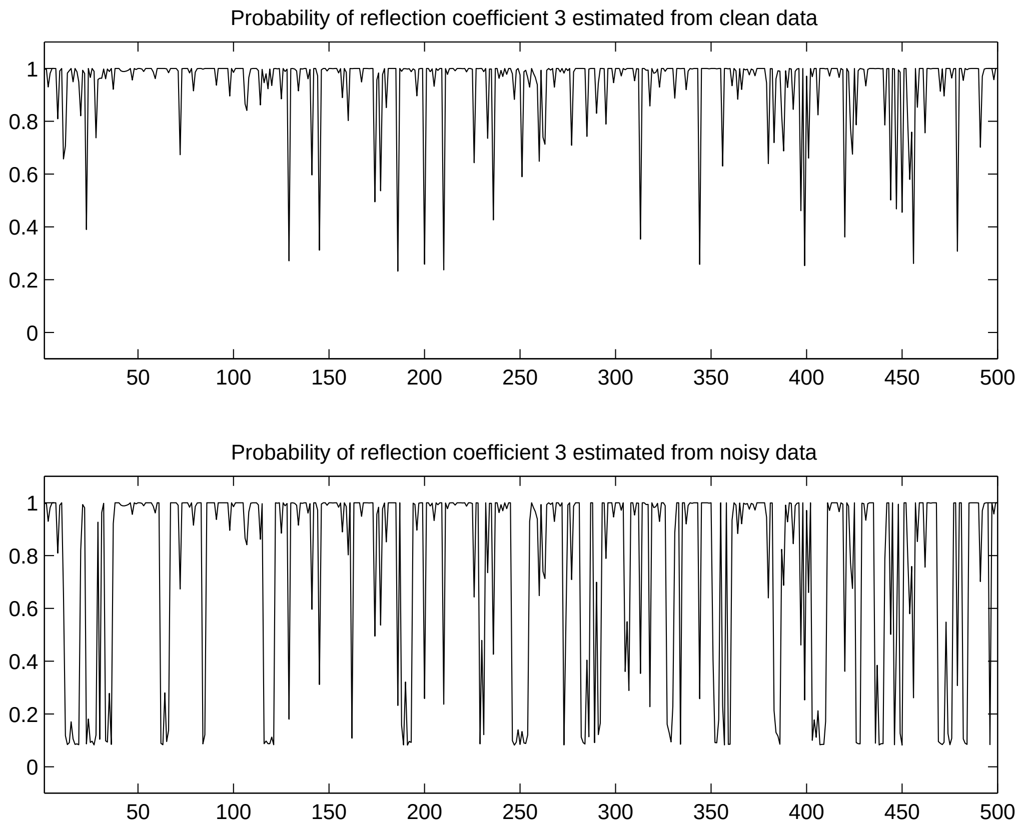

Figure 7.7: Bayesian model probabilities for the third lattice filter stage obtained from clean (upper plot) and contaminated (lower plot) input data.

not made by the conventional classifier; and $n_b = 0$, where n_b denotes the number of mistakes made by the conventional classifier that were not made by the integrating HMM. Under a Binomial distribution, this difference is significant at a threshold of 0.01. The probability observing the result by chance if both methods are equal is $P = 0.0039$. Although we do no think that a difference in generalization accuracy of 0.2% is interesting, we have further investigated where the integrating HMM had problems classifying the label correctly. We realized that this may happen when a state appears in the the test sequence for only one time instance. Note that such a state has rather low prior probability under the transition probabilities used to generate the time series.

When classifying the contaminated time series the integrating HMM achieves a generalization accuracy of 96.3%, which compares to 92.4% reached by the conventional classifier. This equals to $n_a = 36$ and $n_b = 192$, which is highly significant. The probability of observing this result if both

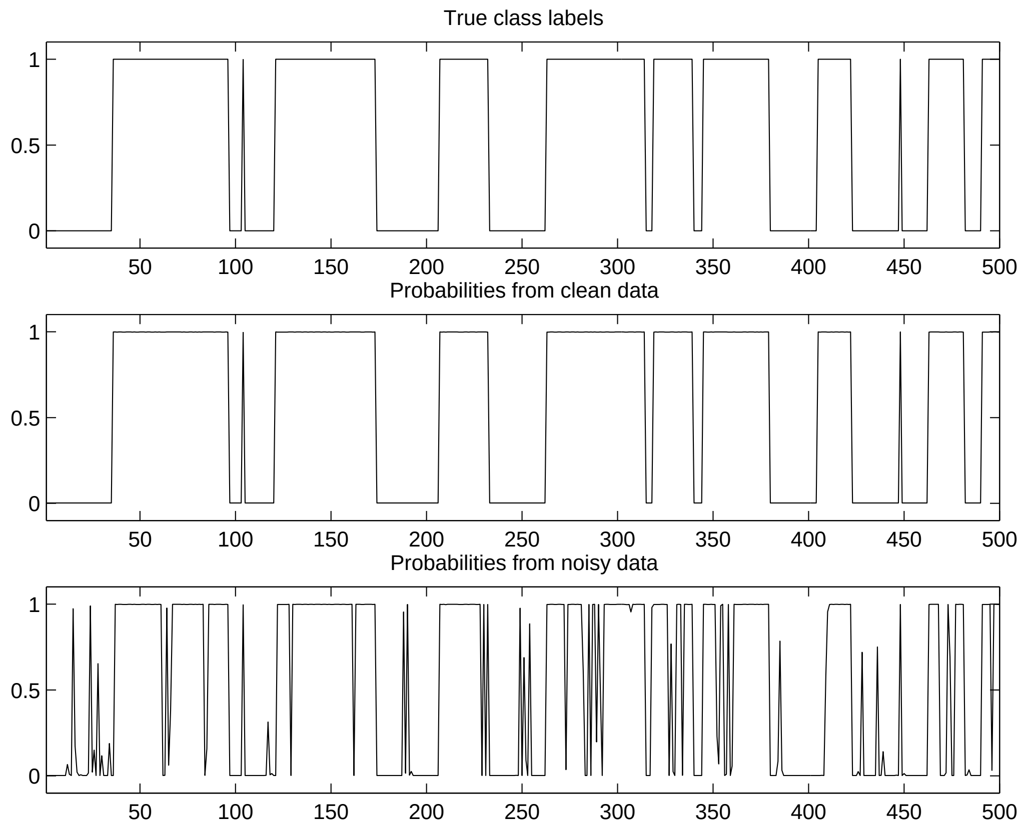

Figure 7.8: This figure shows traces of the probabilities for class 2 predicted by a conventional method using clean and noisy data as well as the true labels.

methods are actually equal is $P = 3.1710^{-23}$.

Since we use a generative model, we can also provide an expectation of feature values. Using the third reflection coefficient as an example, the plots in figure 7.11 show the difference between estimates calculated from clean and noisy data and the difference between the expectations in latent space and the estimates obtained from clean data. Compared with the estimates from the contaminated time series, the expectations in latent space are indeed closer to the estimates obtained from the uncontaminated time series. Hence we may not only improve predictions, we may even provide better estimates for the input features. However that is not always true: the difference between the expected latent values and the estimates obtained on clean data might even increase when the contamination is close to a change-point in the time series. An example may be seen in figure 7.11 close to sample 400.

We have analyzed the relations in latent space quantitatively by consider-

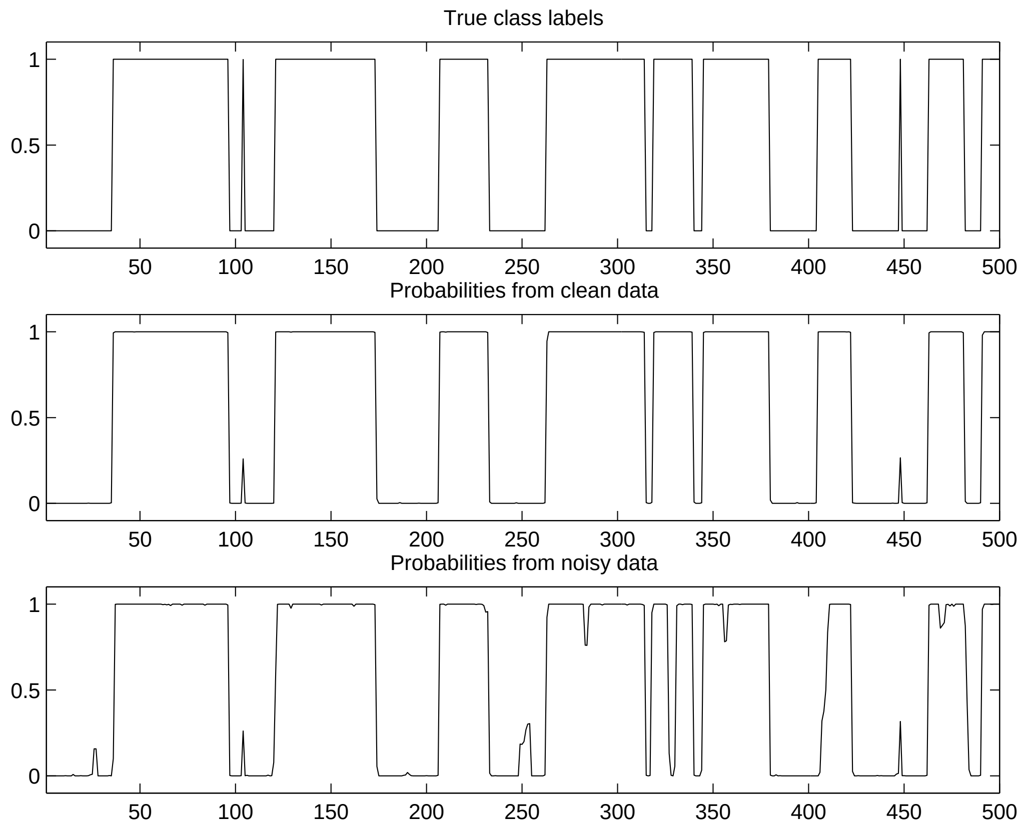

Figure 7.9: This figure shows probability plots for class 2 obtained by integrating out feature and model uncertainty for clean as well as noisy data. Note that the predictions obtained from "clean" data misclassify the 2 samples that are from a single switch to class 2.

ing the sum of squares of the observed differences. On one hand, we have the difference between the best feature estimates extracted from clean data and the expected value obtained from contaminated data. On the other hand, there is the difference between the best estimates extracted from clean and contaminated data. The results are reported in table 7.1. We see that for both the second and third reflection coefficient the expected latent value is closer than the best estimates from contaminated data. However for the first reflection coefficient, the best estimate from contaminated data is closer. The reason for this discrepancy is that when extracted from clean data, the model probability of the first reflection coefficient is often rather small whereas both the second and third coefficient usually have rather high probabilities.

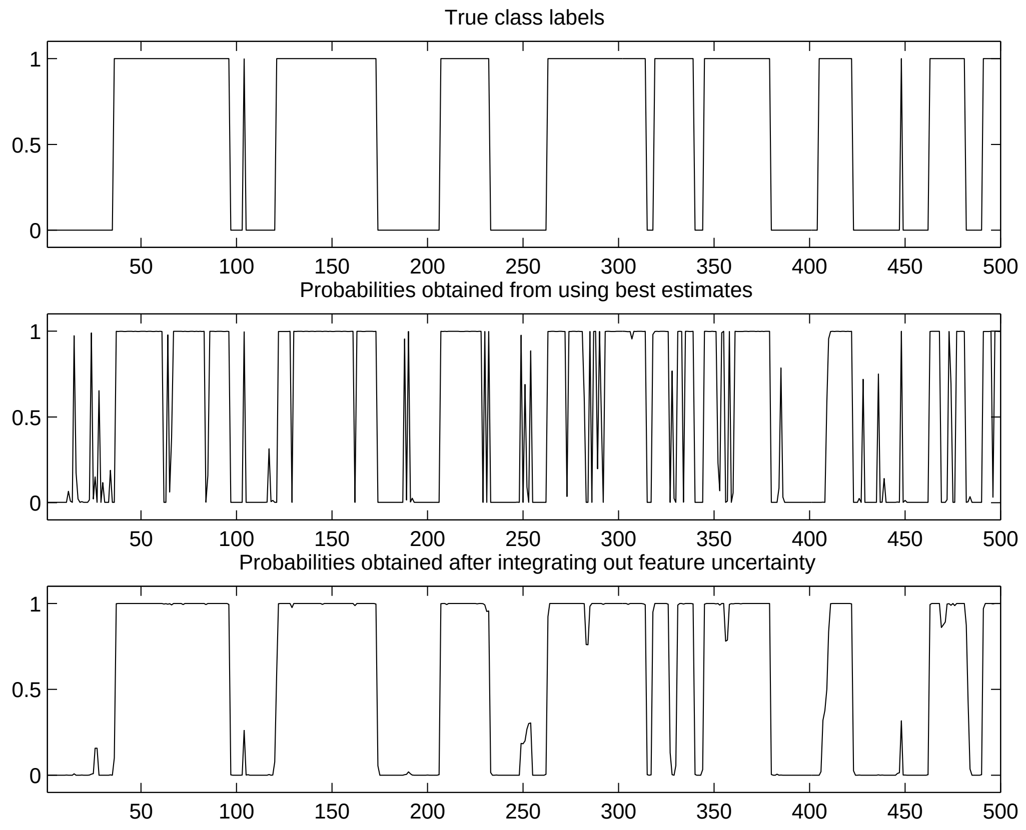

Figure 7.10: For easier comparison this figure shows the true labels and the probabilities obtained by both methods for the contaminated data. We see that the proposed method reduces the number of mistakes *and* that in general the probabilities obtained from contaminated data are less certain about the class label.

Table 7.1: Sum of squared differences at feature level

	cf. 1	cf. 2	cf. 3
$\sum_k(\hat{\rho}_{clean} - <\rho_{noise}>)^2$	1298.0	1888.6	487.9
$\sum_k(\hat{\rho}_{clean} - \hat{\rho}_{noise})^2$	923.8	7056.5	650.7

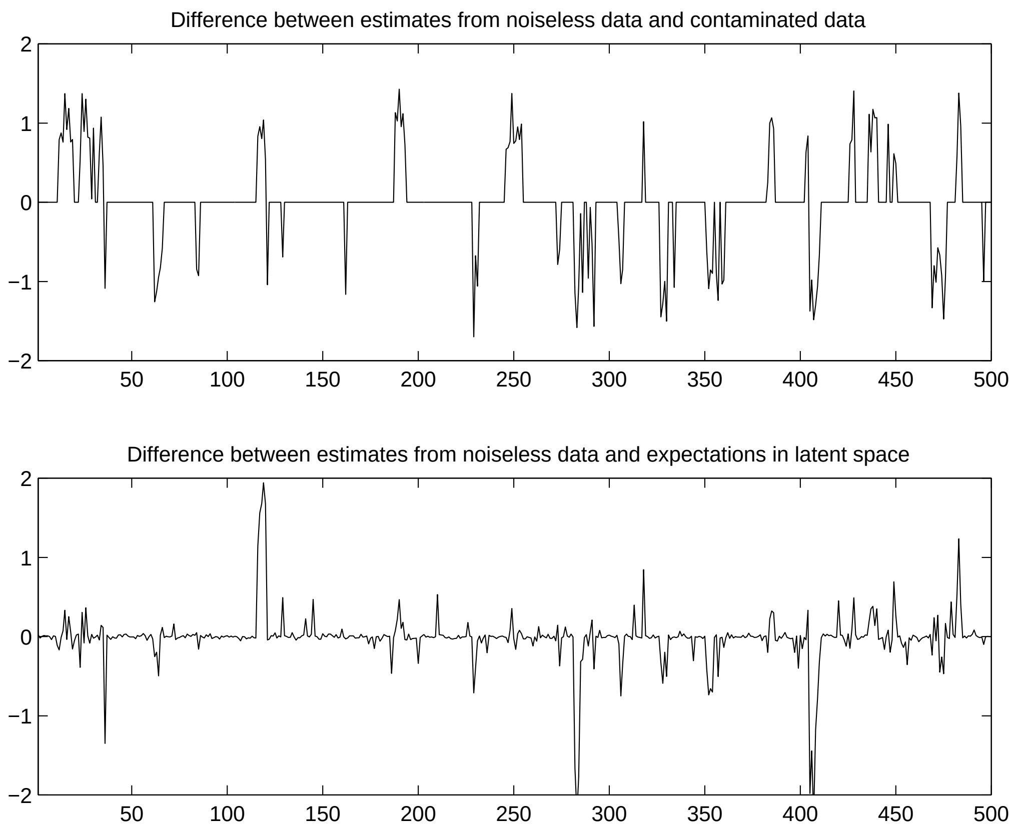

Figure 7.11: In this figure we see plots showing the difference between feature estimates from clean and noisy data and the difference between the feature estimated from clean data and the expectation obtained by marginalization.

7.4.2 Results using sleep EEG

In a second experiment, we used the same all night EEG recordings that were already used in chapter 6. That is, we used data recorded at electrodes C3, O1, C4 and O2. The raw EEG was preprocessed using the lattice filter model that was proposed in chapter 4. As we already did in chapter 6, we predict probabilities for Rechtschaffen and Kales labels wake, REM and stage 4. The remaining epochs are added to the training data without labels. Model inference was performed with the same 6 recordings that were already used in chapter 6. Following the arguments laid out in chaper 6, we reduced the number of training samples by extracting the median feature vector from each labeled (30 seconds) epoch. This reduction of training samples was again motivated to reduce the time for model inference. Model inference was done by drawing 2000 samples according to the full conditionals formulated in equation (7.10).

The inferred model was then applied to 6 independent recordings by using the scheme described in subsection 7.3.6. For predictions, we had to calculate the expectations of the latent feature estimates. These expectations were obtained from a sampled Markov Chain, which had to converge as well. Therefore we started simulating with the first sample from the Markov Chain that was obtained during model inference. All predictions of latent variables and state probabilities are then obtained by taking expectations after allowing the first 1000 samples as burn in.

In principle, we observed similar results as were reported in chapter 6. That is, for some recordings we found that the predicted probabilities correspond to the consensus scoring obtained by a human expert. However, for a majority of the 6 recordings that were used in this test we observed problems with REM non-REM separation. The results from one recording, where the probability plots give a similar impression as does the consensus hypnogram, is shown in figure 7.12. The probability traces shown in figure 7.13 are from a recording where the analysis failed. As was argued in chapter 6, this failure is most probably due to the low quality of the recording.

In order to obtain quantitative results, we compared the generalization accuracy for predicting wake, REM sleep and delta sleep. The labels obtained from thresholding the probabilities were compared against the Rechtschaffen and Kales labels wake, REM and combined stage 3 and 4 from the consensus scorings. The thresholds used for predicting these labels from the probability plots were obtined by the same ROC analysis that was used in chapter 6. The results obtained from this comparison are summarized in table 7.2. Compared with the generalization accuracies that were obtained with a conventional classifier[6], the proposed architecture performed worse. The most obvious reason for this lower generalization accuracy is the HMM-like architecture that was used. The probabilities predicted for a segment will depend on the states of neighbouring segments. That is, if a segment shows high probabilities for either wake, REM or delta sleep, the same state will have a high prior probability in both the preceeding and adjacent segment. This will lead to larger probabilites in general, and the method will be more sensitive for all three events. We see this by comparing the sensitivities and specificities reported in table 7.2 with those reported in table 6.4.

Table 7.2: Comparison with expert scorings

	subj. 1	subj. 2	subj. 3	subj. 4	subj. 5	subj. 6
spc. wake (%)	39.5	1.7	19.6	13.9	37.7	1.5
spc. REM (%)	87.4	4.2	44.5	80	52.8	0
spc. S3 & S4 (%)	78.4	70.3	24.0	64.7	46.5	11.4
sns. wake (%)	95.6	99.8	97.7	99.8	99.2	99.8
sns. REM (%)	96.2	97.2	94.5	55.2	71.9	99.3
sns. S3 & S4 (%)	91.5	91.0	91.0	99.8	99.3	97.0
acc. machine (%)	77.4	63.1	64.0	38.8	49.4	60.4
acc. rater 1 (%)	44.8	93.2	95.4	92.5	92.5	92.2
acc. rater 2 (%)	91.8	94.1	92	92.9	91.0	96.3

[6]These results are found in table 6.4

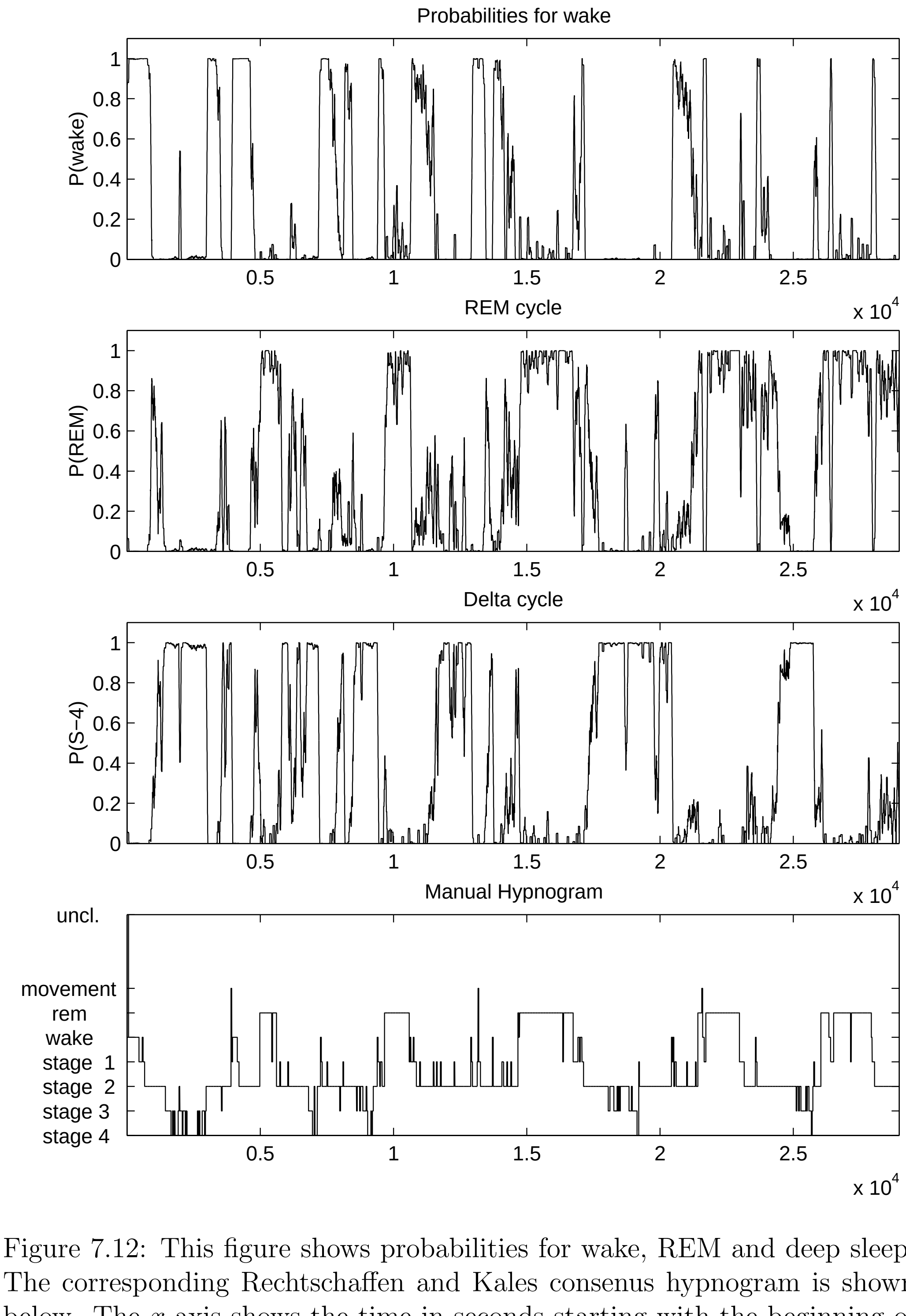

Figure 7.12: This figure shows probabilities for wake, REM and deep sleep. The corresponding Rechtschaffen and Kales consenus hypnogram is shown below. The x-axis shows the time in seconds starting with the beginning of the recording. We see that the agreement between the expert scoring and the auomatic analysis with respect to REM is reasonably good. However, the sensitivity for wake and delta sleep is too large.

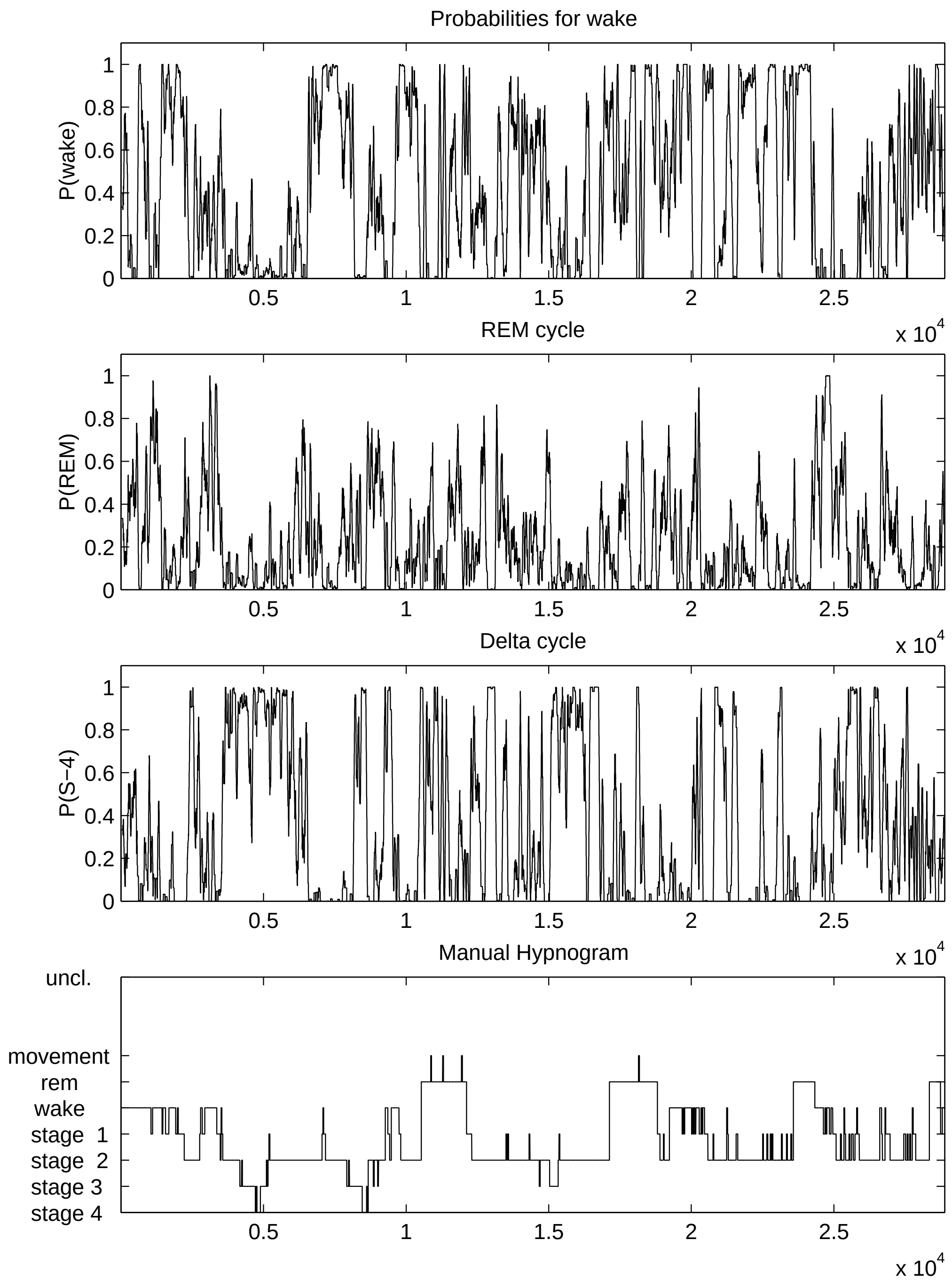

Figure 7.13: This figure shows probabilities for wake, REM and deep sleep. The corresponding Rechtschaffen and Kales consenus hypnogram is shown below. The x-axis shows the time in seconds starting with the beginning of the recording. The bad correlation between the probability plots and the Rechtschaffen and Kales stages is most probably caused by the low quality of the recording.

Our final interest is to get an impression how the method handles inputs that have been found to be unreliable. As was explained above, the reliability of an input is measured by the variance of the estimate and the probability of the corresponding lattice filter stage. In figure 7.14 we see a plot of 120 seconds length that allows to compare the probability plots for stage wake, REM and delta sleep obtained from integrating out uncertainty with plots that were obtained by conventional conditioning. The plot below shows the probability of the first reflection coefficient. We see clearly that the probabilities obtained from integrating out feature uncertainty do not depend on uncertain information. The probabilities obtained from the conventional approach rely on all features. This manifests itself in increased probabilities for wake caused by unreliable features.

The differences between the probability estimates shown in figure 7.14 are caused by differences between feature estimates obtained from preprocessing and the expectations obtained by integrating out feature and model uncertainties. Such differences between feature estimates are illustrated in figure 7.15. The plots represent a segment of 5 minutes of EEG, where a large part of the information at the beginning and at the end is missing. Such missing inputs are caused by overflows in the analog digital converters. Another possible reason for missing inputs is that the electrodes were clamped to ground when the subject had to go to the bathroom. During preprocesing, missing information is set to "not a number" (nan). In order to allow predictions, we treat such missing inputs by setting the corresponding model probabilities to 0. The upper traces in figure 7.15 show plots of the model probabilities, the variances and the feature itself. Note the missing inputs in the variance and feature estimates. The two traces below show the expectations obtained from integrating out uncertainty and the differences between the expectations and the values obtained from preprocessing. The feature values are reconstructed (or changed) using information obtained form other sensors and from past and future estimates.

7.5 Summary

We reported in this chapter about a method of optimal integration between preprocessing and estimation of probabilities. The proposed method is optimal in a Bayesian sense, because all uncertainties are treated by marginalization. An experiment with synthetic data showed that the method will improve results, when the original data is contaminated with white noise artifacts. Compared with a conventional classifier the results improved from a generalization accuracy of 92.4% to 96.3%. This difference is highly significant.

The results obtained for sleep analysis are less satisfying. Although the method improves the probability estimates of unreliable segments, the overall generalization accuracy is significantly below the accuracy reported in chapter 6. An explanation for the low generalization accuracy is the Markov dependency among class lables as is assumed by the model. Every segment that shows a high probability for wake, REM or delta sleep will introduce a high prior probability for the same event in both neighbouring segments. Thus we will in general observe large probabilities. An idea that could help in this particular case would be to infer the model using a strong prior for equal transition probabilities. We could also drop the conditional dependency accross time completely. However, this would mean to reduce to spatial sensor fusion only.

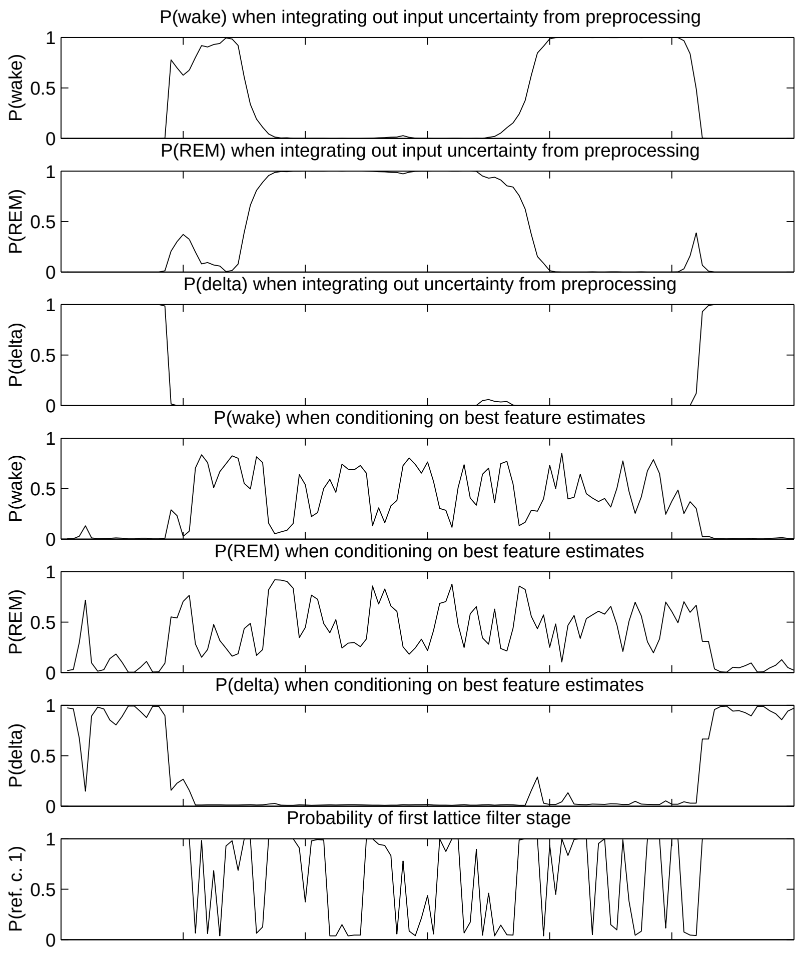

Figure 7.14: This figure shows the probabilities for stage wake, REM and delta sleep, for a segment of 120 seconds length. At the top we see plots obtained by integrating out feature and model uncertainty. Below we see the same plots obtained from conditioning on best estimates. The last trace shows the corresponding probabilities of the corresponding lattice filter stage. The probabilities obtained by conditioning are more affected by low reliable information. Low reliable features are classified as wake.

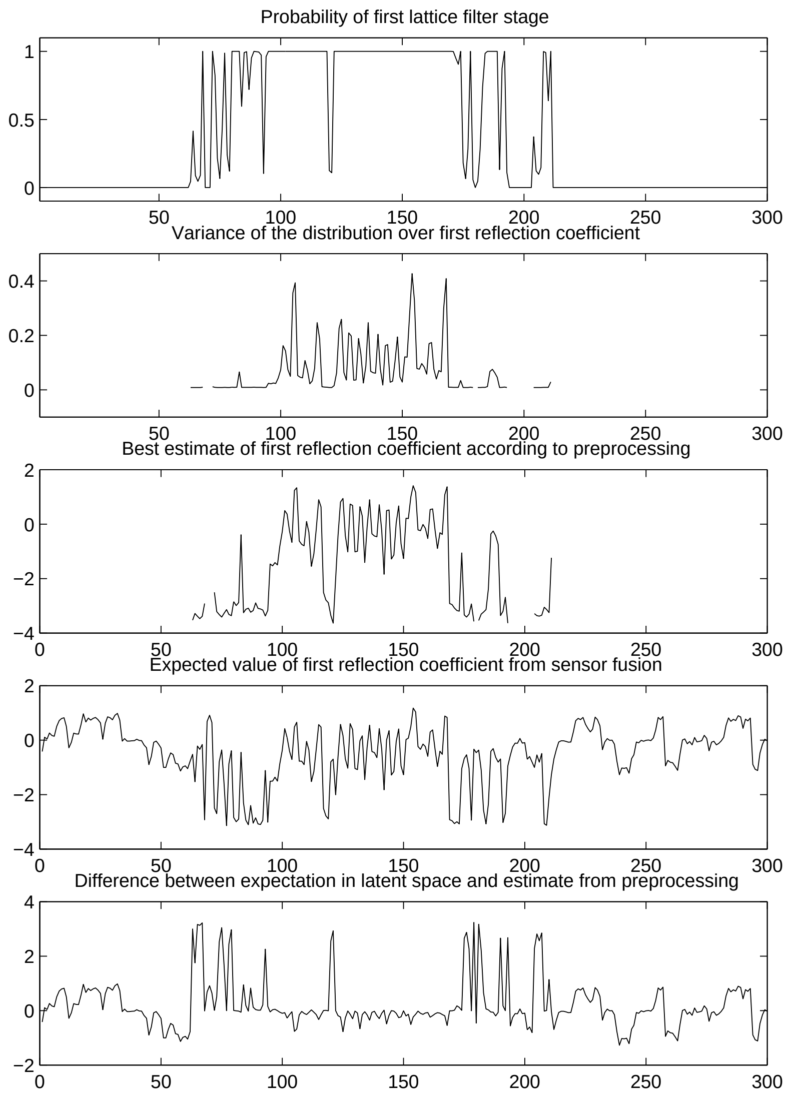

Figure 7.15: The plots in this figure represent 5 minutes of data. The plots show the estimates from preprocessing (model probabilities, variances and feature values), the expectations of the latent feature variables and the differences between the estimates from preprocessing and the expected values. A large part of information at the beginning and at the end of the plot is missing. The corresponding values are reconstructed from other sensors and from past and futre estimates.

Chapter 8

Discussion

This final chapter provides a summary of the methods developed in this thesis. The main contribution has been an investigation of how Bayesian methods can be used to increase the reliability of decisions. The application that served as a major example was classification of all night sleep EEG reordings. The Bayesian methodology was used to derive *optimal* algorithms for preprocessing, feature subset selection[1], static classification and sensor fusion.

We have argued that one requirement to obtain reliable decisions is an appropriately chosen model. In the Bayesian framework we treat model selection by calculating the probabilities of different models. In chapter 4 we derived the probabilities of lattice filter stages as opposed to a white noise explanation of the observed data. These probabilities were used to obtain the optimal order of AR-processes. If white noise is considered as an artefact, we may also use this probability as reliability measure of a lattice filter stage. Based on this measure we developed a Bayesian method for sensor fusion in chapter 7.

Model selection was also dealt with in chapter 6, where we derived the probabilities of classifiers with different model orders. Another example for model selection was presented in chapter 5, where we determined the probabilities of different feature subsets in several classification tasks. In all three cases we found models with reasonably chosen complexity. In chapter 6, this was a mandatory requirement. One of the requirements for the SIESTA analyzer was to allow for obtaining information that goes beyond probabilities for classes. The clinical experts involved in the SIESTA project wanted to have means to analyze the model structure. Obviously, this information is only meaningful for an appropriately chosen model.

[1]Feature subset selection is the usual terminology, correctly speaking the proposd technique integrates out subset uncertainty.

The seond requirement that has been found to be important to achieve reliable decisions is to integrate out all kinds of uncertainties involved. We have developed such a technique in chapter 5, where the predicted probabilities result from integrating over the parameter posteriors *and* summing over different feature subsets according to their probability. Viewed in that perspective, the predictions obtained for the generative classifier that was proposed in chapter 6 are somewhat suboptimal. We integrate over the posterior in parameter space, but we condition on the most probable model. Although conditioning is not optimal, we found only one dominating model in most experiments .

In situations with large amount of data, limited computational resources prevent us from applying interesting models directly to the data. An example of this sort are the all night sleep EEG recordings used in the thesis. Usually such problems are approached by separating the analysis in a preprocessing stage and in a classification stage. A conventional architecture would condition on the feature estimates obtained from preprocessing. This way of solving the problem neglects the uncertanty in preprocessing and it violates the assumptions underlying the Bayesian paradigm. An algorithm that avoids such suboptimality was proposed in chapter 7. The method resolves the problem by using *one* probability distribution over the *entire* model. Predictions are obtained from the margial distribution after integrating over *all* unobserved variables *including* preprocessed features. Compared with classical approaches, the proposed method has several advantages:

- From a Bayesian point of view, the method results in optimal decisions.
- We demonstrated for both a synthetic problem and the analysis of all night sleep EEG that the predicted probabilities are less affected by unreliable information.

The generalization accuracy in the sleep analysis experiment was nevertheless significantly below the accuracy reached by the static classifier that was proposed in chapter 6. The reason for this effect is the way by which we decided to approach sleep analysis. According to the decisions taken in the SIESTA project, we used labels for the extrem states wake, REM and deep sleep. The predictions for all data from between these extreme events depend on a-priori assumptions about spatial smoothness. The probability functions obtained from static classification will in general be much smoother compared with those obtained from dynamic classification as was used in chapter 7. Hence we could either use a very strong prior for equal transition probabilities, or we could drop the conditional dependency over time completely. The latter would mean to reduce to spatial sensor fusion only.

The final conclusion from the analysis reported in this thesis is that, once we decide for the Bayesian way, we should do the entire analysis within this framework. In particular, we expect that the reliability of all decisions will increase, if we put all unknown quantities into *one* probabilistic model, and predict from the marginal distributions. In terms of computational cost, this is still a quite demanding attempt. However, we should not take this as an argument for relying on suboptimal techniques. Since the computer power increases every year by at least a factor of 2, the resulting algorithms will soon be tractable on usual PC's.

Bibliography

[AFD00] C. Andrieu, J. F. G. Freitas, and A. Doucet. Robust full Bayesian methods for neural networks. In S. A. Solla, T. K. Leen, and K. R. Müller, editors, *Advances in Neural Processing Systems 12*, pages 379–385. MIT Press, 2000.

[Aka74] H. Akaike. A new look at statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

[Att99] H. Attias. Inferring parameters and structure of latent variable models by variational Bayes. In *Proc. 15th Conf. on Uncertainty in AI, 1999*, 1999.

[Bal97] V. Balasubramanian. Statistical inference, Occam's razor, and statistical mechanics on the space of probability distributions. *Neural Computation*, 9:349–368, 1997.

[BB92] J. O. Berger and J. M. Bernardo. On the development of reference priors. In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics 4*, Oxford, 1992. Oxford University Press.

[BB98] S. Brunak and P. Baldi. *Bioinformatics.* MIT Press, Cambridge, Massachusetts, 1998.

[Bis95] C. M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

[BJ76] G. E. P. Box and G. M. Jenkins. *Time Series Analysis, forecasting and control.* Holden-Day, Oakland, California, 1976.

[BPSW70] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, 41:164–171, 1970.

[Bre90] G. L. Bretthorst. Bayesian analysis iii: Applications to nmr signal detection, model selection and parameter estimation. *Journal of Magnetic Resonance*, 88:572–595, 1990.

[BS94] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory.* Wiley, Chichester, 1994.

[CC95] B. P. Charlin and S. Chib. Bayesian model choice via Markov chain Monte Carlo. *Journal of the Royal Statistical Society series B*, 57:473–484, 1995.

[Daw76] A. P. Dawid. Properties of diagnostic data distributions. *Biometrics*, 32:647–658, 1976.

[DGL96] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition.* Springer, New York, 1996.

[DK82] P. A. Devijver and J. V. Kittler. *Pattern Recognition. A Statistical Approach.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society series B*, 39:1–38, 1977.

[DR00] R. Dybowski and S. Roberts. Confidence intervals and prediction intervals for feed forward neural networks. In *Clinical Applications of Artifical Neural Networks*, page to appear. Cambridge University Press, Cambridge, UK., 2000.

[DS95] P. Dellaportas and S. A. Stephens. Bayesian analysis of errors-in-variables regression models. *Biometrics*, 51:1085–1095, 1995.

[Fey72] R. P. Feynman. *Statistical Mechanics.* W.A. Benjamin, Inc., 1972.

[Fre98] B. Frey. *Graphical models for machine learning and digital communication.* MIT Press, Cambride Massachusets, 1998.

[FRKZ97] O. Filz, P. Rappelsberger, H. Koenig, and J. Zeitlhofer. Coherence analysis in eeg sleep studies. In I. Frollo and A. Plackova, editors, *Proc. of Measurement 97*, pages 202–205, 1997.

[FSRD00] A. Flexer, P. Sykacek, I. Rezek, and G. Dorffner. Using hidden Markov models to build an automatic, continuous and probabilistic stager. In *Proceedings of the ICANN 2000*, page to appear, New York, 2000. Springer Verlag.

[GB00] Z. Ghahramani and M. J. Beal. Variational inference for Bayesian mixture of factor analysers. In *Advances in Neural Information Processing Systems 12*, pages 449–455, 2000.

[GG84] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984. [Reprinted in [SP90]].

[GJ94] Z. Ghahramani and M.I. Jordan. Supervised learning from incomplete data via an em approach. In *Advances in Neural Information Processing Systems 6*, 1994.

[GJRL98] Guihenneuc-Jouyaux, S. Richardson, and V. Lasserre. Convergence assessment in latent variable models: application to the longitudinal modelling of marker of HIV progression. In C. Robert, editor, *Discretization and Convergence Assessment for MCMC algorithms*, pages 147–160. Springer Verlag, 1998.

[GM94] U. Grenander and M. Miller. Representation of knowledge in complex sytems (with Discussion). *Journal of the Royal Statistical Society series B*, 56:549–603, 1994.

[Gre95] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[GRe96] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter (ed.). *Markov Chain Monte Carlo in Practice.* Chapman & Hall, London, 1996.

[Has70] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[Hjo70] B. Hjorth. EEG analysis based on time domain properties. *Electroencephalography and Clinical Neurophysiology*, pages 306–319, 1970.

[HM98] C. C. Holmes and B. K. Mallick. Bayesian radial basis functions of variable dimension. *Neural Computation*, 10:1217–1234, 1998.

[HvC93] G. E. Hinton and D. van Camp. Keeping neural networks simple by minimizing the description length of the weights. In *Proc. 6th Annu. Workshop on Computational Learning Theory*, pages 5–13, New York, NY, 1993. ACM Press.

[Jef61] H. Jeffreys. *Theory of Probability*. Clarendon Press, Oxford, third edition, 1961.

[JGJS99] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. In Jordan [Jor99].

[JJ00] T. S. Jaakkola and M. I. Jordan. Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10:25–37, 2000.

[Jor99] M. I. Jordan, editor. *Learning in Graphical Models*. MIT Press, Cambridge, MA, 1999.

[KG85] G. Kitagawa and W. Gersch. A smoothness priors long ar model method for spectral estimation. *IEEE Transactions on Automatic Control*, 30:57–65, 1985.

[KJ97] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence Journal*, pages 273–324, 1997.

[Kub95] S. Kubicki. Vigilanz und Schlaf. In S. Zschocke, editor, *Klinische Elektroenzephalographie*, chapter 5. Springer Verlag, Berlin, 1995. in German.

[Lju99] L. Ljung. *System Identification, Theory for the User*. Prentice-Hall, Englewood Cliffs, New Jersey, 1999.

[Mac97] D. J. MacKay. Ensmble learning for hidden Markov models. Technical report Cavendish Laboratory, University of Cambridge, 1997.

[MRR$^+$53] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1091, 1953.

[Nea96] R. M. Neal. *Bayesian Learning for Neural Networks*. Springer, New York, 1996.

[NH99] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse and other variants. In Jordan [Jor99], pages 355–368.

[PRTJ97] J. Pardey, S. J. Roberts, L. Tarassenko, and J.Stradling. A new approach to the analysis of the human sleep/wakefulness continuum. *J. of Sleep. Res.*, 5:201–210, 1997.

[PS96] D. B. Phillips and A. F. M. Smith. Bayesian model comparison via jump diffusions. In W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, editors, *Markov Chain Monte Carlo in Practice*, pages 215–239, London, 1996. Chapman & Hall.

[RF95] J. J. K. Ó Ruanaidh and W. J. Fitzgerald. *Numerical Bayesian Methods Applied to Signal Processing.* Springer-Verlag, New York, 1995.

[RG97] S. Richardson and P. J. Green. On Bayesian analysis of mixtures with an unknown number of components. *Journal Royal Stat. Soc. B*, 59:731–792, 1997.

[RHRP98] S. J. Roberts, D. Husmeier, I. Rezek, and W. Penny. Bayesian approaches to Gaussian mixture modeling. *IEEE PAMI*, 20:1133–1142, 1998.

[Rip87] B. D. Ripley. *Stochastic Simulation.* Wiley, New York, 1987.

[Rip96] B. D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge, 1996.

[Ris78] J. Rissanen. Modelling by shortest data description. *Automatica*, 14:465–471, 1978.

[RK68] A. Rechtschaffen and A. Kales. *A manual of standardized terminology, techniques and scoring system for sleep stages of human subjects.* NIH Publication No. 204, US Government Printing Office, Washington, DC., 1968.

[RM99] C. P. Robert and K. L. Mengersen. Reparametrization issues in mixture estimationand their bearings on the Gibbs sampler. *Comput. Statis. Data Ana.*, pages 325–343, 1999.

[RR98] I.A. Rezek and S.J. Roberts. Stochastic complexity measures for physiological signal analysis. *IEEE Transactions on Biomedical Engineering*, pages 1186–1191, 1998.

[SBGI96] D. J. Spiegelhalter, N. G. Best, W. R. Gilks, and H. Inskip. Hepatitis: a case study in MCMC methods. In *Markov Chain Monte Carlo in Practice* [GRe96], pages 21–44.

[SDRZ98] P. Sykacek, G. Dorffner, P. Rappelsberger, and J. Zeitlhofer. Experiences with Bayesian learning in a real world application. In M. I. Jordan, M.J. Kearns, and S. Solla, editors, *Advances in Neural Information Processing Systems 10*, pages 964–970, 1998.

[Siv96] D. S. Sivia. *Data analysis a Bayesian tutorial.* Clarendon Press, Oxford, UK, 1996.

[SP90] G. Shafer and J. Pearl, editors. *Readings in Uncertainty Reasoning.* Morgan Kaufmann, San Mateo, CA, 1990.

[SP99] A. Schlögl and G. Pfurtscheller. Estimation of adaptive autoregressive parameters using Kalman filters. Technical report, Dept. for Medical Informatics Technical University Graz, 1999.

[SRR$^+$99] P. Sykacek, S. J. Roberts, I. Rezek, A. Flexer, and G. Dorffner. Bayesian wrappers versus conventional filters: Feature subset selection in the Siesta project. In *Proceedings of the European Medical & Biomedical Engineerinmg Conference*, pages 1652–1653, 1999.

[SS71] A. J. Scott and M. J. Symons. Clustering methods based on likelihood ratio criteria. *Biometrics*, 27:387–397, 1971.

[SS95] M. Stensmo and T.J. Sejnowski. A mixture model system for medical and machine diagnosis. In G. Tesauro et. al., editor, *Advances in Neural Information Processing System 7*, pages 1077–1084, Boston MA, 1995. MIT Press.

[Ste97] M. Stephens. *Bayesian methods for mixtures of normals distributions.* PhD thesis, Magdalen Colledge University of Oxford, 1997.

[Ste00] M. Stephens. Dealing with multimodal posteriors and non-identifiability in mixture models. *Journal of the Royal Statistical Society series B*, page to appear, 2000.

[Syk00] P. Sykacek. On input selection with reversible jump Markov chain Monte Carlo sampling. In S.A. Solla, T.K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 638–644, Boston, MA, 2000. MIT Press.

[Trå91] H. G. C. Tråvén. A neural network approach to statistical pattern classification by "semiparametric" estimation of probability density functions. *IEEE Transactions on Neural Networks*, 2:366–377, 1991.

[Vap95] V. N. Vapnik. *The nature of statistical learning theory.* Springer, New York, 1995.

[WF87] C. S. Wallace and P. R. Freeman. Estimation and inference by compact encoding (with discussion). *Journal of the Royal Statistical Society series B*, 49:240–265, 1987.

[Wri98] W. A. Wright. Neural network regression with uncertain inputs. In *Proceedings of the 1994 IEEE Workshop on Neural Networks for Signal Processing VIII*, pages –, Long Beach, CA, 1998. IEEE Press.

[ZC93] M. H. Zweig and G. Campbell. Receiver-operating characteristic (ROC) plots. *Clinical Chemistry*, 29:561–577, 1993.