Variational Approach to Factor Analysis and Related Models

Frederik Brink Nielsen

May 6, 2004

Preface

This master thesis serves as documentation for the final assignment in the requirements to achieve the degree Master of Science in Engineering. The work has been carried out in the period from the 1st of September 2003 to the 6th of May 2004 at the Intelligent Signal Processing group at the Institute of Informatics and Mathematical Modelling, Technical University of Denmark. The work has been supervised by Professor Lars Kai Hansen and Associate Professor Ole Winther. I wish to thank both my supervisors for great enthusiasm and guidance during the project.

Kgs. Lyngby, May 6, 2004

Frederik Brink Nielsen, s973991

Abstract

This thesis focuses on factor analysis, extended factor analysis and parallel factor analysis. For these models Expectation maximization and variational Bayes algorithms are derived. The variational algorithms are implemented with hierarchical automatic relevance determination priors in order to automatically infer the model order. The algorithms are tested on synthetic data and used for analyzing amino acids and fMRI data. In a model selection task the variational lower bound on the marginal likelihood inferred, in a reliable manner, the correct model type. On the other hand, it was found that the lower bound under estimated the model order compared to the Bayesian information criterion. The effect of hyperparameter optimization is discussed in relation to the generalization error. Furthermore, it is found that the variational algorithms have properties similar to independent component analysis. A model called probabilistic partial least squares is proposed and used in a NPAIRS analysis of the fMRI data. The NPAIRS analysis and the parallel factor model were found to give very similar results.

Keywords: *variational Bayes, model selection, hyperparameter optimization, factor analysis, parallel factor analysis, ARD, fMRI*

Resumé

Denne afhandling omhandler faktor analyse, udviddet faktor analyse og parallel faktor analyse. Forventnings maksimering og variationel Bayes algoritmer er udledt for samtlige modeller. Hirarkisk automatisk relevans priori fordelinger er anvendt i de variationelle algoritmer til automatisk at selektere model ordenen. Algoritmerne er testet på syntetisk data og anvendt til at analysere amino syre og fMRI data. Den variationelle nedre grænse på det marginale likelihood var robust i at selektere mellem forskellige model typer. I sammenligning med Bayes informations kriterie underestimerede den nedre grænse på det marginale likelihood model ordenen. Effekten af hyperparameter optimering er sammenlignet med generalisations fejlen. Endvidere har det vist sig, at de variationelle algoritmer har samme egenskaber som uafhængig komponent analyse. En model kaldet probabilistic partial least squares er foreslået og brugt til en NPAIRS analyse af fMRI data. NPAIRS analysen og parallel faktor analyse gav meget identiske resultater.

Nøgleord: *variationel Bayes, model selektering, hyperparameter optimering, faktor analyse, parallel faktor analyse, ARD, fMRI*

Contents

Nomenclature

Below follow the most used symbols and abbreviations. Scalars, vectors and matrices appear as: x, $\mathbf{x}$ and $\mathbf{X}$. Unless otherwise stated all vectors are column vectors.

$\circ$	Hadamard product (element wise product of two matrices)
$\langle\cdot\rangle$	Expectation operator also denoted by $\mathcal{E}[\cdot]$
$\mathcal{F}(\cdot)$	Lower bound on the log marginal likelihood
$\mathcal{L}(\cdot)$	Log likelihood function or marginal likelihood
diag$[\cdot]$	Sets off diagonal terms of a matrix to zero. When used on a vector the operator returns a diagonal matrix
tr$[\cdot]$	Returns the sum of the diagonal of a matrix
vec$[\cdot]$	Returns the diagonal of a matrix as a column vector
$\mathbf{A}$	Factor loading matrix of dimension $(p \times k)$
$\mathbf{S}$	Factor data matrix $(k \times N)$
$\mathbf{X}$	Data matrix of dimension $(p \times N)$
k	Number of factors
K_T^m	Number of individual factors for data set m
N	Number of samples
p	Data dimension
$p(\cdot)$	Probability distribution
P_m	Data dimension for the m^{th} data set
$q(\cdot)$	Variational posterior distribution
AIC	Akaike's information criterion

ALS Alternate least squares

ARD Automatic relevance determination

BIC Bayes information criterion

EFA Extended factor analysis

EM Expectation maximization

FA Factor analysis

fMRI Functional magnetic resonance imaging

ICA Independent component analysis

KL Kullback-Leibler divergence

MLEFA Maximum likelihood extended factor analysis

MLFA Maximum likelihood factor analysis

MLPARAFAC Maximum likelihood parallel factor analysis

NPAIRS Nonparametric prediction, activation, influence, and reproducibility resampling

PARAFAC parallel factor analysis

PCA Principal component analysis

PLS Partial least squares

pPCA Probabilistic principal component analysis

pPLS Probabilistic partial least squares

VBEFA Variational Bayesian extended factor analysis

VBEM Variational Bayesian expectation maximization

VBFA Variational Bayesian factor analysis

VBPARAFAC Variational Bayesian parallel factor analysis

Chapter 1

Introduction

In data modelling the goal is to find models which explain the structure in the data at hand and generalizes well to new data. In most cases, the data is corrupted by noise, thus a general problem in data modelling is to separate the noise from the interesting features in the data. The structure in the data is typically the interesting part and model estimation can become erroneous if the noise is not taking into account. Formulating a model in a probabilistic sense one incorporates the noise when estimating the model. In this way, the presence of noise is not neglected. Generative models containing hidden (latent) variables often provides a very appealing way to formulate probabilistic models. The factor analysis model which is the main model investigated in this thesis is a generative model. The model assumes that some underlying factors, corrupted by additive Gaussian noise, are responsible for generating the structure in the data. Since the factors are unobserved, estimating the model is in general not possible without resorting to iterative schemes.

Bayesian methods provide a very appealing framework for parameter learning and model selection. In a Bayesian model all uncertainties (e.g., the parameters and hidden variables) are represented by probability distributions. When predicting a new data item, the prediction is based on an average setting of parameters and hidden variables. Thus, uncertainty is handled in an intuitive way - by averages. Averaging out all uncertainties in a model, one gets a probability for the model, which can be used for model comparison.

Those averages are in most cases intractable to compute. Bayesian averaging typically involves complicated and high dimensional integrals, which are not analytical tractable. The variational Bayesian approach which is taken in this project is a method used to approximate these complicated integrals. By transforming the integrals into an optimization problem one can resort to iterative schemes. The variational Bayesian method is an approximation to the ideal Bayesian analysis since the averages are not done with respect to the true distributions. Nevertheless, we are still doing well because we are making averages over all possible settings of parameters and hidden variables, which is an improvement compared to a best fit point estimate of parameters and hidden variables.

In this thesis, the data analyzed are amino acids fluorescence data and functional magnetic resonance imaging (fMRI) data. The amino acids data are analyzed with the variational parallel factor analysis model. fMRI is a noninvasive technique that measures blood oxygen level dependent (BOLD) signals of neuronal activity in the brain. fMRI provides a time series of 3D-images over time. During two scanning session a subject was exposed for a visual stimuli and two fMRI data sets were made available. Most of the algorithms proposed are used for analyzing the fMRI.

The thesis is organized in the following way,

Chapter 2 has three sections. The first section introduces general topics in Bayesian inference. The second section describes maximum likelihood and variational Bayesian learning. The latter section deals with factor analysis, extended factor analysis and parallel factor analysis.

Chapter 3 describes the algorithms derived for the three factor analysis models. Each model is treated separately in each section and expectation maximization and variational Bayes algorithms are provided. For each model, the variational algorithms are used on a small example on synthetic data.

Chapter 4 is divided in two sections. The first section contain experiments on synthetic data aimed for an understanding of the algorithms. Four topics are investigated: the relation between independent component analysis and variational Bayesian factor analysis, model order selection by Bayesian information criterion compared to variational Bayes, model selection using variational Bayes and finally, hyperparameter optimization is compared to the generalization error. In the second section the algorithms are tested on amino acids data and fMRI data.

Chapter 5 summarizes and concludes on the results of this thesis.

Appendix A contains an example of the Kullback-Leibler divergence between two Gaussians.

Chapter 2

Theory

This chapter presents the majority of the theory and models considered throughout this thesis. The first section deals with general topics in Bayesian inference including; parameter learning, model selection, Occam's razor, automatic relevance determination, etc. The second section describes maximum likelihood and variational Bayesian learning. Factor analysis, extended factor analysis and parallel factor analysis are explained in the last section.

2.1 Bayesian Inference

Inference: "The act or process of deriving logical conclusions from premises known or assumed to be true" [1]. Consider a court room example where a prosecuted is convicted based on some evidence. The evidence can be questioned, but at the end of the day when inferring the judgement, the evidence is usually assumed true or false. As MacKay points out: "you cannot do inference without making assumptions" [27]. In Bayesian inference all beliefs are represented by probability distributions and the laws of probability theory are used to manipulate the probabilities and infer the quantities of interest. A good poker player instinctively knows Bayes and probability theory. The cards at hand and previous experience can be mapped to probabilities and manipulated by Bayes rule to produce an optimal decision. Bayesian inference is simply common sense expressed mathematically by probability theory. Furthermore, if all assumptions/prior beliefs are true it can be proven that the Bayesian solution is optimal.

In probabilistic data modelling the goal is to develop models that explains the data at hand and generalizes well on new data. The Bayesian approach to data modelling is to treat all uncertain parameters in a model as random variables described by probability distributions. Prior knowledge can be included into the model by specifying a *prior* probability distribution over the parameters. Estimation of the parameters in a Bayesian paradigm leads to a *posterior* probability distribution of the parameters. Usually the parameters are estimated by the mean value of the posterior. Furthermore, using the posterior, it is easy to calculate errorbars which reflects the uncertainty on the values that the parameters can take. Obtaining a distribution over parameters and not just a single point estimate is one of the main strengths of Bayesian modelling. Mathematically speaking Bayes' theorem states

$$p(\boldsymbol{\theta}|\mathcal{D}, m) = \frac{p(\mathcal{D}|\boldsymbol{\theta}, m)p(\boldsymbol{\theta}|m)}{p(\mathcal{D}|m)}\,, \tag{2.1}$$

where $\boldsymbol{\theta} = \{\theta_1 \dots \theta_k\}$ denotes the unknown parameters and $\mathcal{D}$ denotes the data. $p(m)$ is a prior over model class and $p(\boldsymbol{\theta}|m)$ is the parameter prior. $p(\mathcal{D}|\boldsymbol{\theta}, m)$ is the *likelihood* of the parameters also termed the *likelihood function*. $p(\boldsymbol{\theta}|\mathcal{D}, m)$ is the posterior distribution for the parameters. $p(\mathcal{D}|m)$ is a normalizing constant called the *marginal likelihood* or *evidence*. The marginal likelihood is found by marginalizing the likelihood function with respect to the parameter prior

$$p(\mathcal{D}|m) = \int p(\mathcal{D}|\boldsymbol{\theta}, m)p(\boldsymbol{\theta}|m)d\boldsymbol{\theta}\,. \tag{2.2}$$

In many cases a single specific model is assumed and the dependencies on m can be left out of the equation. When inferring the parameters one can usually discard the normalization constant as it is independent of the parameter values [25]. The posterior distribution given by equation (2.1) is the quantity we're after in Bayesian parameter learning, but we can take the inference problem one step further. Defining a prior distribution over model structures $p(m)$ and prior distributions over parameters for each model structure $p(\boldsymbol{\theta}|m)$ yields a posterior distribution over models given by Bayes' rule

$$p(m|\mathcal{D}) \propto p(\mathcal{D}|m)p(m) \tag{2.3}$$

In many cases there is no reason to prefer one model over another and $p(m)$ is given a flat prior. In this case $p(m|\mathcal{D})$ is directly proportional to the marginal likelihood and the most probable model or model structure is the one that maximizes $p(\mathcal{D}|m)$. The marginal likelihood plays an important role in Bayesian inference. In the denominator of equation (2.1) it serves as a normalization constant and in equation (2.3) it is used to perform model selection. Integrating out the parameters penalizes models with more degrees of freedom, since these models can model a larger range of data sets. This property of Bayesian integration has been called *Occam's razor*, since it favors simpler explanations (models) for the data over complex ones [26]. Occam's razor an old scientific principle states that, when trying to explain some phenomenon (i.e the observed data), the simplest model that can adequately explain it should be chosen. There is no point in choosing an complex model when a much simpler would do. A very complex model will be able

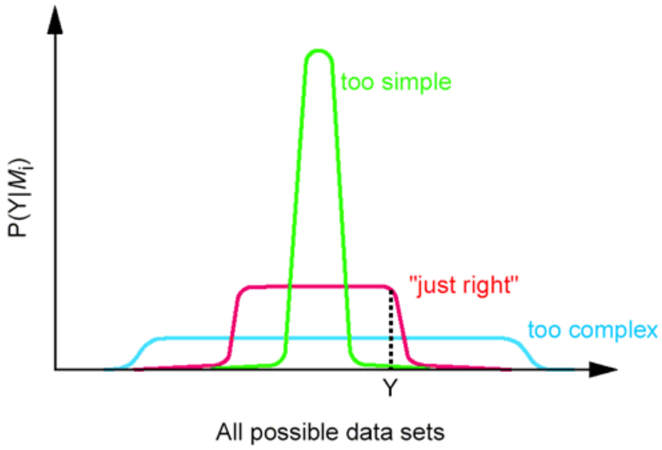


Figure 2.1: Caricature depicting Occam's razor (taken from lecture slides by Zoubin Gharahmani, originally adapted from MacKay [26]). The horizontal axis denotes all possible data sets of a particular size and the vertical axis is the marginal likelihood for three different models M_i of differing complexity. Given a particular data set Y (previously called $\mathcal{D}$), model selection is possible because model structures that are too simple are unlikely to generate the data set in question, while model structures that are to complex can generate many possible data sets, but again, are unlikely to generate that particular data set at random. Remember that $p(Y \mid M_i)$ is a probability distribution and has to sum to one.

to fit the given data almost perfectly but it will not be able to generalize very well. On the other hand, very simple models will not be able to capture the essential structure in the data. A caricature of Occam's razor is given in figure 2.1, where the horizontal axis denotes all possible data sets to be modelled, and the vertical axis is the marginal probability under each of the three models of increasing complexity. The complexity of a model is related to the range of data sets it can capture. Thus for a simple model the probability is concentrated over a small range of data sets, and conversely a complex model has the ability to model a wide range of data sets. Anyhow, when inferring a model based on the marginal likelihood, one should not forget that Bayesian predictions are stochastic just like any other inference scheme that generalize from a finite sample [16]. The data set Y on figure 2.1 could fall in regions favoring a wrong model being either too simple or too complex. For further illustration, figure 2.2 shows Dilbert faced with a very stochastic model/hypothesis selection problem.

When performing Bayesian model selection the marginal likelihood is the key quantity to work with, but unfortunately it is intractable for all most all models of interest, and thus needs to be approximated. The intractability arises from the fact that the integral over parameters and possibly hidden

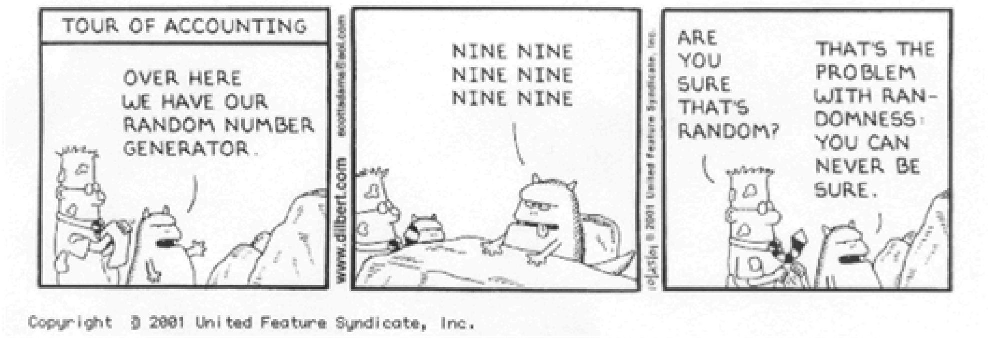


Figure 2.2: Dilbert's tour of accounting.

variables can be a complicated and very high dimensional integral with couplings between parameters and hidden variables. There are many different ways of approximating the marginal likelihood. The simplest methods are analytical approximations such as the Bayesian Information Criteria (BIC), the Laplace approximation or the Cheeseman-Stutz approximation. These methods are attractive due to their simplicity, especially BIC, but can lead to inaccurate results. Computational intensive methods such as Monte Carlo integration and its' many extensions rely on sampling and are accurate in the limit of an infinite number of samples. Another approach is the variational Bayesian method which optimizes a lower bound on the marginal likelihood. The thesis by Beal [5] provides both theoretical and experimental results regarding the accuracy of many of the above mentioned approximation schemes. His general take-home message is that variational lower bounds has superior performance over BIC and Cheeseman-Stutz and tend to produce more reliable results than the sampling method, annealed importance sampling [31]. Moreover the computational burden is only about 1% for the variational method compared to annealed importance sampling.

An important topic in Bayesian inference is the problem of choosing priors. In general there are three different schools of thought when it comes to assigning priors; these can be loosely categorized into *subjective*, *objective* and *hierarchical* approaches [5]. Although one should not forget that priors, whatever type, are still subjective in some sense.

The subjective approach is to encapsulate as much prior knowledge as possible into the priors. The prior knowledge may be due to previous experiments or expert knowledge. A subjective and analytic convenient class of priors are *conjugate* priors. A prior is conjugate if the posterior distribution resulting from multiplying the likelihood function by the prior is of the same form as the prior. Conjugate priors only exist for models in the *exponential family* [5].

The definition of an exponential family model is one that has a likelihood function of the form

$$p(\mathbf{x}_i|\boldsymbol{\theta}) = g(\boldsymbol{\theta})f(\mathbf{x}_i)e^{\boldsymbol{\phi}(\boldsymbol{\theta})^\top \mathbf{u}(\mathbf{x}_i)} \tag{2.4}$$

where $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ is the observed data, $g(\boldsymbol{\theta})$ is a normalization constant and $\boldsymbol{\phi}(\boldsymbol{\theta})$ is a vector holding the so-called *natural parameters.* $\mathbf{u}(\mathbf{x}_i)$ is the *sufficient statistics* and together with $f(\mathbf{x}_i)$ they define the exponential family. If we assume that the observed data is independent and identical distributed (iid) then the probability of the data under the model is $p(\mathbf{X}|\boldsymbol{\theta}) = \prod_{i=1}^N p(\mathbf{x}_i|\boldsymbol{\theta})$. Now consider the conjugate prior

$$p(\boldsymbol{\theta}|\eta, \boldsymbol{\nu}) = h(\eta, \boldsymbol{\nu})g(\boldsymbol{\theta})^\eta e^{\boldsymbol{\phi}(\boldsymbol{\theta})^\top \boldsymbol{\nu}} \tag{2.5}$$

where η and $\boldsymbol{\nu}$ are parameters of the prior, and $h(\eta, \boldsymbol{\nu})$ is an appropriate normalization constant. Note that $g(\boldsymbol{\theta})$ and $\boldsymbol{\phi}(\boldsymbol{\theta})$ are the same functions as in equation (2.4). The posterior arising from multiplying the likelihood by the prior results in

$$p(\boldsymbol{\theta}|\mathbf{X}) \propto p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\eta, \boldsymbol{\nu}) \propto p(\boldsymbol{\theta}|\tilde{\eta}, \tilde{\boldsymbol{\nu}}) \tag{2.6}$$

where $\tilde{\eta} = \eta + N$ and $\tilde{\boldsymbol{\nu}} = \boldsymbol{\nu} + \sum_{i=1}^N \mathbf{u}(\mathbf{x}_i)$ are the posterior parameters. Note that the posterior has the same form as the conjugate prior.

Objective priors also called *non-informative* priors can be used when the modeller has no available prior knowledge. This approach also termed 'prior ignorance' is the practice of setting priors so that the resulting posterior is determined by the likelihood alone. The often encountered Jeffreys' priors and reference priors are non-informative, but conjugate priors can also be non-informative under certain conditions. An example of a non-informative conjugate prior is the Gamma prior for an unknown precision parameter α (inverse variance $\alpha = \frac{1}{\sigma^2}$) of a Gaussian distribution. A Gamma distribution has shape parameter a and inverse scale parameter b and is defined as

$$p(\alpha|a, b) = \frac{b^a}{\Gamma(a)}\alpha^{a-1}e^{-b\alpha}, \qquad 0 \leq \alpha < \infty \tag{2.7}$$

in which $\Gamma(a) = \int_0^\infty t^{a-1}e^{-t}dt$, is the Gamma function. The Gamma distribution is a single mode distribution having mean $\frac{a}{b}$ and variance $\frac{a}{b^2}$. When $a \to 0$ and $b \to 0$ we obtain a non-informative prior. In this limit the distribution becomes uniform over a logarithmic scale ($p(\ln \alpha) = c$), but the prior is also *improper*, meaning that it is not normalizable [27]. However, in practice we can set the parameter values: e.g. $a = b = 10^{-3}$ resulting in a broad normalizable prior reflecting our lack of knowledge.

Hierarchical priors can be used when the parameters of a prior can be assumed to be drawn from another higher level prior distribution. A parameter controlling another parameter is called a *hyperparameter* which can be defined by a *hyperprior* distribution. A hyperparameter controlling another hyperparameter is called a *hyperhyperparameter* which can be defined by a *hyperhyperprior* distribution and so on. That should specify the terminology. As an example, consider a Gaussian parameter prior $p(\theta|\mu,\alpha)$ having a known mean μ and an unknown precision α. Now, let the precision hyperparameter α follow a conjugate Gamma hyperprior $p(\alpha|a,b)$ with the hyperhyperparameters a and b. The resulting parameter prior is found by integrating out α

$$\begin{aligned}
p(\theta|\mu,a,b) &= \int_0^\infty p(\theta|\mu,\alpha)p(\alpha|a,b)d\alpha \\
&= \int_0^\infty \sqrt{\frac{\alpha}{2\pi}} e^{-\frac{\alpha^2}{2}(\theta-\mu)^2} \frac{b^a}{\Gamma(a)} \alpha^{a-1} e^{-b\alpha} d\alpha \\
&= \frac{b^a}{\Gamma(a)\sqrt{2\pi}} \int_0^\infty \alpha^{a-\frac{1}{2}} e^{-\alpha(b+\frac{1}{2}(\theta-\mu)^2)} d\alpha \\
&= \frac{b^a}{\Gamma(a)\sqrt{2\pi}} \frac{\Gamma(a+\frac{1}{2})}{(b+(\theta-\mu)^2)^{a+\frac{1}{2}}} \\
&= \frac{\Gamma(a+\frac{1}{2})}{\Gamma(a)\sqrt{2\pi b}} \left(1+\frac{(\theta-\mu)^2}{2b}\right)^{-(a+\frac{1}{2})} \qquad (2.8)
\end{aligned}$$

where the 'new' parameter prior now depend on the hyperhyperparameters a and b. Marginalizing out α yields a Student-t distribution with $\nu = 2b$ degrees of freedom displaced around μ. Just as for the Gamma distribution a and b are respectively shape and scale parameters. Interpreting the parameter distribution as a hierarchical prior is often more intuitive than simply enforcing a Student-t prior.

Hierarchical priors are very useful in model selection tasks where the number of parameters increase with the model complexity. Often we have no clue concerning what values the parameters might take and therefore the parameters need to be estimated in some way. Following an estimation scheme which is not strictly Bayesian, more parameters will introduce a bias favoring models with high complexity. In practice, very few estimation methods are strictly Bayesian and at some point we have to introduce approximations into the Bayesian analysis. Constructing hierarchical priors so that the number of parameters or hyperparameters to be estimated does not scale with model complexity makes model selection less prone to error. As we will see later in chapter 4, we can correctly infer the model generating the data, without considering the complexity for different types of hierarchical models.

The complexity is in this case related to the number of parameters in each type of model.

Hierarchical priors can be constructed to automatically determine a certain model structure without the need for testing all possible model structures, or to determine the relevance of inputs to a system. These specific types of hierarchical priors are motivated by the framework of *Automatic relevance determination* (ARD) introduced in the context of neural networks by Neal and MacKay [26]. In a feed forward neural network, individual hyperparameters would control groups of weights/parameters, in particular those associated with a certain input variable. Forcing a Gaussian prior with mean value $\mu = 0$ and precision α on a group of weights used for a certain input, the posterior for those weights can only become more concentrated around zero. Thus the adjustable hyperparameter α has the same functionality as a weight decay regularizer. No matter whether these hyperparameters are optimized or integrated out, they can only regularize the model and therefore they are incapable of causing over-fitting. A finite data set will show random correlations between inputs of which some might be irrelevant. A conventional neural network, even with weight decay, can fail in setting the irrelevant inputs to zero, by overfitting to the random correlations.
One might think that ARD is just another pruning technique, but a clear advantage is that we can just leave the extra irrelevant inputs in the model without harm. When an input is totally irrelevant in predicting the data, the respective precision hyperparameter will tend towards infinity and effectively shutting off that input. The conceptually difference between ARD and pruning is that ARD allows all inputs to interfere to some degree in the model, but by analyzing the hyperparameters one can remove the irrelevant inputs. The ARD prior really demonstrates that hierarchical priors can lead to very convenient additional information. Had we used the Student-t prior we would not be able determine the relevance of the inputs.

In chapter 3 we will see examples on the use of conjugate, non-informative and hierarchical ARD priors when deriving variational Bayesian algorithms. In chapter 4 we will see that ARD priors can be used to avoid model structure search, and how the variational lower bound on the marginal likelihood can be used as a guide for model selection.

2.2 Learning Algorithms

In this section, maximum likelihood and variational learning will be described with focus on the Expectation Maximization (EM) algorithm (Dempster et al. [9]) and the Variational Bayesian EM (VBEM) algorithm by Attias [2] which has been further formalized by Beal and Ghahramani [4]. Both algorithms are attractive because they are assured to converge monotonically to a local maxima and no step sizes or other convergence parameters needs to be tuned. EM maximizes the likelihood of the parameters $p(\mathbf{X}|\boldsymbol{\theta})$ whereas VBEM maximizes a lower bound on the marginal likelihood $p(\mathbf{X}|m)$.

2.2.1 Maximum Likelihood learning

Maximum likelihood is the principle of finding the most probable parameter setting in a model given some data

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\mathbf{X}|\boldsymbol{\theta}) \tag{2.9}$$

where $p(\mathbf{X}|\boldsymbol{\theta})$ is the likelihood function. No priors have influence and the parameters $\boldsymbol{\theta}^*$ are set to maximize the likelihood, which is the probability of the observed data given the parameters. Often it is more convenient to work with the logarithm of the likelihood which is defined as

$$\mathcal{L}(\boldsymbol{\theta}) \equiv \ln p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{i=1}^{N} \ln p(\mathbf{x}_i|\boldsymbol{\theta}) \tag{2.10}$$

The model may also include *latent* or *hidden* variables $\mathbf{S}$, which are unobserved yet interact through the parameters to generate the data. With hidden variables we can formulate the *complete likelihood*

$$\mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \equiv \ln p(\mathbf{X}, \mathbf{S}|\boldsymbol{\theta}) = \sum_{i=1}^{N} \ln p(\mathbf{x}_i, \mathbf{s}_i|\boldsymbol{\theta}) \tag{2.11}$$

where $\{\mathbf{X}, \mathbf{S}\}$ is the *complete* data. In this case $\{\mathbf{X}\}$ would be the *incomplete* data. The interpretation is that with hidden variables in the model, the observed data is an incomplete account for all the factors in the model. The log likelihood is obtained from the complete log likelihood by integrating out the hidden variables

$$\mathcal{L}(\boldsymbol{\theta}) = \ln \int p(\mathbf{X}, \mathbf{S}|\boldsymbol{\theta}) d\mathbf{S} \tag{2.12}$$

In many cases, finding the parameters that maximizes the log likelihood can be difficult, especially when the model incorporates hidden variables. In probabilistic models where some variables, $\mathbf{X}$, are observed and other variables, $\mathbf{S}$, are hidden, learning of the model parameters, $\boldsymbol{\theta}$, can be achieved by the EM algorithm.

The EM algorithm

The EM algorithm is an algorithm that iteratively finds a maximum likelihood estimate of the parameters. There are two main applications of the EM algorithm. The first occurs when the data has missing values, due to limitations of the observation process. The second occurs when optimizing the likelihood function is intractable, but the likelihood function can be simplified by assuming the existence of hidden variables. An example for this is the mixture of Gaussian model. The maximum likelihood solution for this model can not be found on closed form, but assuming that the responsibilities are hidden variables, one can derive an EM algorithm. The EM algorithm alternates between an E-step, which infers posterior distributions over hidden variables given a current parameter setting, and a M-step, which maximizes $\mathcal{L}(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ given the statistics gathered from the E-step. EM can be viewed as a bound maximization algorithm. Introducing any arbitrary set of distributions $q(\mathbf{S}) = \{q(\mathbf{s}_i)\}_{i=1}^N$ over hidden variables and applying Jensen's inequality we obtain a lower bound on the log likelihood

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= \ln \int q(\mathbf{S}) \frac{p(\mathbf{X}, \mathbf{S}|\boldsymbol{\theta})}{q(\mathbf{S})} d\mathbf{S} &\quad (2.13)\\ &\geq \int q(\mathbf{S}) \ln \frac{p(\mathbf{X}, \mathbf{S}|\boldsymbol{\theta})}{q(\mathbf{S})} d\mathbf{S} &\quad (2.14)\\ &\equiv \mathcal{F}(q(\mathbf{S}), \boldsymbol{\theta}) &\quad (2.15)\end{aligned}$$

The inequality is possible because the logarithm is a concave function and because a distribution have to sum to one. The inequality turns into an equality in the case where $q(\mathbf{S})$ is equal to the true posterior $p(\mathbf{S}|\mathbf{X}, \boldsymbol{\theta})$. This can be seen by inserting the true posterior into the equation

$$\begin{aligned}\mathcal{F}(p(\mathbf{S}|\mathbf{X}, \boldsymbol{\theta}), \boldsymbol{\theta}) &= \int p(\mathbf{S}|\mathbf{X}, \boldsymbol{\theta}) \ln \frac{p(\mathbf{X}, \mathbf{S}|\boldsymbol{\theta})}{p(\mathbf{S}|\mathbf{X}, \boldsymbol{\theta})} d\mathbf{S} &\quad (2.16)\\ &= \int p(\mathbf{S}|\mathbf{X}, \boldsymbol{\theta}) \ln p(\mathbf{X}|\boldsymbol{\theta}) d\mathbf{S} &\quad (2.17)\\ &= \ln p(\mathbf{X}|\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) &\quad (2.18)\end{aligned}$$

Thus, using the true posterior the bound is tight. Neal and Hinton [30] has showed that if the functional $\mathcal{F}(q(\mathbf{S}), \boldsymbol{\theta})$ has a local or global maximum at $q(\mathbf{S})^*$ and $\boldsymbol{\theta}^*$, then $\mathcal{L}(\boldsymbol{\theta}^*)$ is at a corresponding local or global maximum. Denoting iteration number by superscript (t), the EM algorithm is given by the iterations

$$\begin{aligned}q(\mathbf{s}_i)^{(t+1)} &= \arg\max_{q(\mathbf{s}_i)} \mathcal{F}(q(\mathbf{S}), \boldsymbol{\theta}^{(t)}) &\quad (2.19)\\ \boldsymbol{\theta}^{(t+1)} &= \arg\max_{\boldsymbol{\theta}} \mathcal{F}(q(\mathbf{S})^{(t+1)}, \boldsymbol{\theta}) &\quad (2.20)\end{aligned}$$

Starting from an initial guess of the parameters, the E-step sets the hidden variable posterior $q(\mathbf{S})$ equal to the exact posterior $p(\mathbf{S}|\mathbf{X}, \boldsymbol{\theta})$. Since $\mathcal{F}(q(\mathbf{S})^{(t+1)}, \boldsymbol{\theta}^{(t)}) = \mathcal{L}(\boldsymbol{\theta}^{(t)})$ at the beginning of each M-step, and since the E-step does not change the parameters, the likelihood is guaranteed not to decrease after each combined EM-step. In practice what to do when deriving an EM algorithm is summarized below.

EM recipe

1. **E-step:** Multiply the likelihood by the prior to obtain the hidden variable posterior and neglect constant terms $p(\mathbf{s}_i|\mathbf{x}_i, \boldsymbol{\theta}) \propto p(\mathbf{x}_i|\mathbf{s}_i, \boldsymbol{\theta})p(\mathbf{s}_i)$. Now, identify the posterior and find the moments required in the M-step.

2. **M-step:** Write down the expected complete log likelihood $\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle$ with respect to the hidden variable posterior found in the E-step. Take partial derivatives of $\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle$ with respect to the different parameters. Equate the derivatives to zero and rearrange to find the update equations.

2.2.2 Variational learning

The log marginal likelihood for model m is

$$\mathcal{L}_m \equiv \ln p(\mathbf{X}|m) = \ln \int p(\mathbf{X}, \mathbf{S}, \boldsymbol{\theta}|m)\, d\mathbf{S}d\boldsymbol{\theta} \tag{2.21}$$

Introducing any arbitrary distribution $q(\mathbf{S}, \boldsymbol{\theta})$ over hidden variables and parameters and applying Jensen's inequality we obtain a lower bound on the log marginal likelihood

$$\begin{aligned} \ln p(\mathbf{X}|m) &= \ln \int q(\mathbf{S}, \boldsymbol{\theta}) \frac{p(\mathbf{X}, \mathbf{S}, \boldsymbol{\theta}|m)}{q(\mathbf{S}, \boldsymbol{\theta})}\, d\mathbf{S}d\boldsymbol{\theta} && (2.22) \\ &\geq \int q(\mathbf{S}, \boldsymbol{\theta}) \ln \frac{p(\mathbf{X}, \mathbf{S}, \boldsymbol{\theta}|m)}{q(\mathbf{S}, \boldsymbol{\theta})}\, d\mathbf{S}d\boldsymbol{\theta} && (2.23) \\ &\equiv \mathcal{F}_m(q(\mathbf{S}, \boldsymbol{\theta})) && (2.24) \end{aligned}$$

The inequality turns into an equality in the case where $q(\mathbf{S}, \boldsymbol{\theta})$ is equal to the true posterior $p(\mathbf{S}, \boldsymbol{\theta}|\mathbf{X})$. This easily seen by inserting the true posterior into the equation. The goal in variational learning is to make the bound as tight as possible. The closer $q(\mathbf{S}, \boldsymbol{\theta})$ is to the true posterior the better is the estimate of the marginal likelihood. Lets rewrite $\mathcal{F}_m$ as

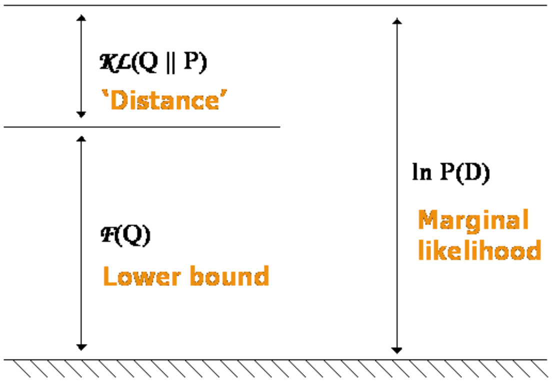


Figure 2.3: The log marginal likelihood is equal to the sum of the functional $\mathcal{F}_m$ and the KL-divergence between the true posterior and the approximate posterior ('approximation error').

$$\begin{aligned}\mathcal{F}_m(q(\mathbf{S},\boldsymbol{\theta})) &= \int q(\mathbf{S},\boldsymbol{\theta})\left(\ln\frac{p(\mathbf{S},\boldsymbol{\theta}|\mathbf{X},m)}{q(\mathbf{S},\boldsymbol{\theta})} + \ln p(\mathbf{X}|m)\right) d\mathbf{S}d\boldsymbol{\theta} &(2.25)\\ &= \ln p(\mathbf{X}|m) - \mathrm{KL}(q(\mathbf{S},\boldsymbol{\theta})||p(\mathbf{S},\boldsymbol{\theta}|\mathbf{X},m)) &(2.26)\end{aligned}$$

where $\mathrm{KL}(q(\theta)||p(\theta)) = \int q(\theta)\ln\frac{q(\theta)}{p(\theta)}$ is the Kullback-Leibler divergence. The KL-divergence is always positive and only zero when $q(\theta) = p(\theta)$. The KL-divergence can be thought of being a distance measure between the approximate posterior and the true posterior. Mathematically speaking the KL-divergence is not a distance, because it is not symmetric, and it does not satisfy the triangular inequality. Anyhow, minimizing the KL-divergence in equation (2.26) one gets an improved estimate of the marginal likelihood. Figure 2.3 shows a caricature of equation (2.26). See also appendix A for an example considering the KL-divergence of two Gaussians.

Another useful rewriting of $\mathcal{F}_m$ is

$$\mathcal{F}_m(q(\mathbf{S},\boldsymbol{\theta})) = \int q(\mathbf{S},\boldsymbol{\theta})\ln p(\mathbf{X}|\mathbf{S},\boldsymbol{\theta},m)\, d\mathbf{S}d\boldsymbol{\theta} - \mathrm{KL}(q(\mathbf{S},\boldsymbol{\theta})||p(\mathbf{S},\boldsymbol{\theta})) \quad (2.27)$$

On this form it is easy to observe the Occam effect that arises when averaging over uncertain quantities. The left term is the likelihood of parameters and hidden variables averaged with respect to the approximating posterior. The right term is a KL-penalty that assures that the approximate posterior stays close to the prior.

The VBEM algorithm

Variational learning involves optimizing the functional $\mathcal{F}_m$ with respect to the distribution $q(\mathbf{S}, \boldsymbol{\theta})$ by the use calculus of variations. To able to carry out the maximization of the lower bound an approximating posterior $q(\mathbf{S}, \boldsymbol{\theta})$ needs to be proposed. The approximation used in the VBEM algorithm is that hidden variables and parameters are independent $q(\mathbf{S}, \boldsymbol{\theta}) \approx q(\mathbf{S})q(\boldsymbol{\theta})$. Combined with the usual assumption that the observed data $\mathbf{X}$ is iid, this leads to a tractable way of inference. While the factorization of the posterior distribution over hidden variables and parameters may seem drastic, one can think of it as replacing *stochastic* dependencies between $\mathbf{S}$ and $\boldsymbol{\theta}$ with *deterministic* dependencies between relevant moments of the two sets of variables. Just like EM the VBEM algorithm alternates between a VBE-step and a VBM-step. In the VBE-step the functional is maximized with respect to $q(\mathbf{S})$ and in the VBM-step it is maximized with respect to $q(\boldsymbol{\theta})$. Using the factorized posterior the functional becomes

$$\mathcal{F}_m(q(\mathbf{S}), q(\boldsymbol{\theta})) = \int q(\mathbf{S})q(\boldsymbol{\theta}) \ln \frac{p(\mathbf{X}, \mathbf{S}, \boldsymbol{\theta}|m)}{q(\mathbf{S})q(\boldsymbol{\theta})} \, d\mathbf{S}d\boldsymbol{\theta} \tag{2.28}$$

To derive the VBE-step we add a Lagrange multiplier to assure that we find a probability distribution when optimizing for $q(\mathbf{S})$. This leads to

$$\begin{aligned}\widetilde{\mathcal{F}}_m = &\int q(\mathbf{S})q(\boldsymbol{\theta}) \ln \frac{p(\mathbf{X}, \mathbf{S}|\boldsymbol{\theta}, m)}{q(\mathbf{S})} \, d\mathbf{S}d\boldsymbol{\theta} + \int q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{\theta}|m)}{q(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \\ &+ \lambda \left(\int q(\mathbf{S}) \, d\mathbf{S} - 1 \right)\end{aligned} \tag{2.29}$$

Then taking the functional derivative of $\widetilde{\mathcal{F}}_m$ with respect to $q(\mathbf{S})$ we get

$$\begin{aligned}\frac{\delta \widetilde{\mathcal{F}}_m}{\delta q(\mathbf{S}')} &= \int \delta(\mathbf{S} - \mathbf{S}')q(\boldsymbol{\theta}) \ln \frac{p(\mathbf{X}, \mathbf{S}|\boldsymbol{\theta}, m)}{q(\mathbf{S})} \, d\mathbf{S}d\boldsymbol{\theta} \\ &\quad - \int \delta(\mathbf{S} - \mathbf{S}')q(\mathbf{S})q(\boldsymbol{\theta}) \frac{1}{q(\mathbf{S})} \, d\mathbf{S}d\boldsymbol{\theta} + \lambda \\ &= -\int q(\boldsymbol{\theta}) \ln q(\mathbf{S}') \, d\boldsymbol{\theta} + \int q(\boldsymbol{\theta}) \ln p(\mathbf{X}, \mathbf{S}'|\boldsymbol{\theta}, m) \, d\boldsymbol{\theta} - 1 + \lambda \\ &= -\ln q(\mathbf{S}') + \int q(\boldsymbol{\theta}) \ln p(\mathbf{X}, \mathbf{S}'|\boldsymbol{\theta}, m) \, d\boldsymbol{\theta} - 1 + \lambda\end{aligned} \tag{2.30}$$

equating to zero and taking the exponential we obtain

$$q(\mathbf{S}') = \exp(-1+\lambda)\exp\left(\langle \ln p(\mathbf{X},\mathbf{S}'|\boldsymbol{\theta},m)\rangle_{q(\boldsymbol{\theta})}\right) \qquad (2.31)$$

where the normalization is assured by setting

$$\lambda = 1 - \ln \int p(\mathbf{X},\mathbf{S}'|\boldsymbol{\theta},m)\, d\mathbf{S} \qquad (2.32)$$

In a similar manner one can derive the functional derivative of $\widetilde{\mathcal{F}}_m$ with respect to $q(\boldsymbol{\theta})$. Now skipping the normalization constants and denoting iteration number by superscript (t) we arrive at the VBEM algorithm

$$q^{(t+1)}(\mathbf{S}) \propto \exp\left[\langle \ln p(\mathbf{X},\mathbf{S}|\boldsymbol{\theta},m)\rangle_{q^{(t)}(\boldsymbol{\theta})}\right] \qquad (2.33)$$

$$q^{(t+1)}(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta}|m)\exp\left[\langle \ln p(\mathbf{X},\mathbf{S}|\boldsymbol{\theta},m)\rangle_{q^{(t+1)}(\mathbf{S})}\right] \qquad (2.34)$$

Iterating these equations from some initial guess is assured to monotonically increase the bound $\mathcal{F}_m$ until a local maxima is reached. Equation (2.33) is the VBE-step and equation (2.34) is the VBM-step. A caricature of the steps involved are shown on figure 2.4. The VBEM algorithm reduces to the ordinary EM algorithm if we restrict the parameter distribution to a point estimate, i.e. a Dirac delta function, $q(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$ in which case the M-step involves re-estimating $\boldsymbol{\theta}^*$.

Usually the parameter priors are functions of hyperparameters, $\mathbf{a}$, so we can write $p(\boldsymbol{\theta}|\mathbf{a},m)$. After each VBEM-step we can optimize the priors with respect to these hyperparameters

$$\mathbf{a}^{(t+1)} = \arg\max_{\mathbf{a}} \mathcal{F}_m(q(\mathbf{S}),q(\boldsymbol{\theta}),\mathbf{a}) \qquad (2.35)$$

In doing so, the log marginal likelihood increases after each combined VBEM-step and hyperparameter update. Whereas EM maximizes the log likelihood, VBEM with hyperparameter optimization maximizes the log marginal likelihood. If we do not employ hyperparameter optimization (fixed priors) there exist an exact value of the log marginal likelihood and VBEM only maximizes the bound to this quantity.

In practice, what to do when deriving an VBEM algorithm is summarized below.

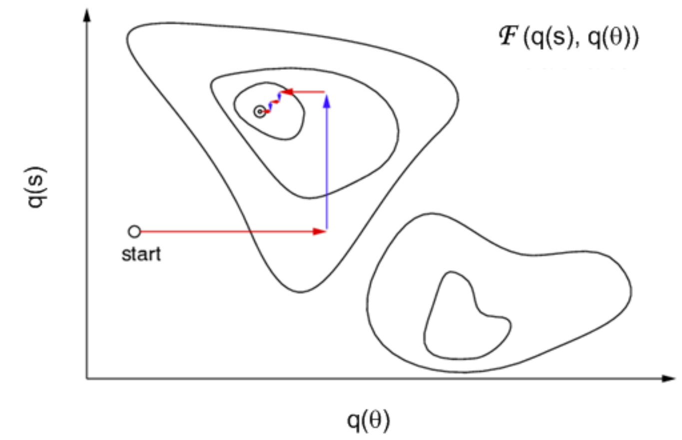


Figure 2.4: VBEM alternates by maximizing either $q(\mathbf{S})$ or $q(\boldsymbol{\theta})$ while holding the other fixed. The effect being that VBEM is coordinate ascent in $(q(\mathbf{S}), q(\boldsymbol{\theta}))$ space.

VBEM recipe

1. **VBE-step:** Write down the expected complete log likelihood $\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S})\rangle_{q(\boldsymbol{\theta})}$ with respect to the parameter posteriors. Lump all terms not dependent on the hidden variables into a constant. Now, identify the posterior for the hidden variables and find the required moments.

2. **VBM-step:** For a certain $q(\boldsymbol{\theta}_i)$, write down the expected complete log likelihood $\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S})\rangle_{q(\boldsymbol{\theta}_{j\neq i})q(\mathbf{S})}$ with respect to all other variational posteriors. Lump all terms not dependent on $\boldsymbol{\theta}_i$ into a constant and identify the required moments for $q(\boldsymbol{\theta}_i)$.

3. **Hyperparameter optimization:** This part is optional and only required if one wants to do hyperparameter optimization. Write the expected complete log likelihood $\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S})\rangle_{q(\boldsymbol{\theta})q(\mathbf{S})}$ as function of the hyperparameters $\mathbf{a}$ and neglect all other terms. Then differentiate with respect to the hyperparameters and equate to zero to find the update equations.

In general, the difference between EM and VBEM, is that VBEM has a build-in model complexity penalty since we integrate over parameters. The computational complexity is almost the same, its only the VBM-step that is more demanding. The VBEM algorithm can be applied to large variety of models, namely those for which the complete likelihood is in the exponential family and has conjugate priors. In the thesis by Beal [5], he call this type of models for *conjugate-exponential*. This includes e.g probabilistic PCA, factor analysis, mixtures of Gaussians, ICA, State-Space models etc. For conjugate-exponential models, the VBE-step computes the expected sufficient statistics $\langle \mathbf{u}(\mathbf{S}, \mathbf{X}) \rangle$ under the hidden variable distribution $q(\mathbf{S})$. The VBM-step computes the expected natural parameters $\langle \boldsymbol{\phi}(\boldsymbol{\theta}) \rangle$ under the parameter distribution $q(\boldsymbol{\theta})$. Using the expected natural parameters, one can use existing algorithms like the junction tree, belief propagation, the Kalman filter etc. in the VBE-step of VBEM.

2.3 Factor Analysis models

In this section factor analysis (FA), extended factor analysis (EFA) and parallel factor analysis (PARAFAC) are described. These are generative models in the sense that they involve unobserved latent variables/factors that generate the data. Factor analysis is a linear Gaussian model and therefore related to many other models including probabilistic principal component analysis (pPCA), noisy independent component analysis (ICA), Kalman filter models etc. (see Roweis for a review on linear Gaussian models [35]). The general idea in factor analysis is to model the correlations amongst a large set of observed variables in terms of a few underlying factors. In addition the observed variables are subject to additive Gaussian noise. Factor analysis is used in many different areas but originally developed by psychologists. As an example, suppose that a stock portfolio would follow a factor analysis model. In this case a common factor could be the Japanese Yen exchange rate or the NASDAQ index. The small fluctuations specific for a single stock would be modelled as noise. A simple extension to factor analysis, I call extended factor analysis (EFA), belongs to the same class of linear Gaussian models. EFA can be useful when dealing with multiple data sets believed to have both individual factors and some common factors. Parallel factor analysis is a multi-way decomposition method frequently used in chemometrics. For an introduction to PARAFAC and much more see the thesis by Bro [7]. PARAFAC and multi-way analysis in general can be used when the data can be described by multiple indices e.g: (batch $\times$ time $\times$ variable). A two-way (time $\times$ variable) PARAFAC model is equivalent to ordinary factor analysis, but differs in the way that FA assumes Gaussian factors and noise. In the general unconstrained PARAFAC model no assumptions on the factors are made and the model is typically estimated by a least squares technique called alternating least squares (ALS).

2.3.1 Factor Analysis

Factor analysis is a method for modelling the correlations in multidimensional data, by low-dimensional latent variables typically called factors or sources. The generative model for factor analysis is similar to models such as pPCA and noisy ICA but they differ in the assumptions on the noise and factor distributions. Denote the observed p-dimensional data set by $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$, and the k-dimensional factors by $\mathbf{S} = \{\mathbf{s}_i\}_{i=1}^N$, where typically $k << p$. The factors are then linearly transform by a $(p \times k)$ factor loading matrix $\mathbf{A}$ (the factor loadings), translated by a fixed amount $\boldsymbol{\mu}$ and added a noise term $\mathbf{e}_i$. Expressed mathematically

$$\mathbf{x}_i = \mathbf{A}\mathbf{s}_i + \boldsymbol{\mu} + \mathbf{e}_i \tag{2.36}$$

The factors and noise are assumed uncorrelated and Gaussian distributed as $\mathcal{N}(\mathbf{s}_i \,|\, \mathbf{0}, \mathbf{I})$ and $\mathcal{N}(\mathbf{e}_i \,|\, \mathbf{0}, \boldsymbol{\Psi})$. The noise covariance $\boldsymbol{\Psi}$ is assumed diagonal and can be thought of as being sensor noise. The mean $\boldsymbol{\mu}$ of the factor analyzer is from now on dropped for simplicity and for not to obscure the equations. If necessary, it can be estimated by adding the factors $\mathbf{s}_i$ with a constant bias dimension of 1 and adding a corresponding column $\boldsymbol{\mu}$ to the matrix $\mathbf{A}$. In this way, learning the $\mathbf{A}$ matrix incorporates learning the mean.

Since the factors and the noise are uncorrelated Gaussian distributed and the model is linear, $\mathbf{x}_i$ is also Gaussian distributed. Skipping $\boldsymbol{\mu}$, the mean of $\mathbf{x}_i$ is zero and the covariance of $\mathbf{x}_i$ is

$$\mathrm{cov}(\mathbf{x}_i) = \langle \mathbf{x}_i \mathbf{x}_i^\top \rangle = \langle (\mathbf{A}\mathbf{s}_i + \mathbf{e}_i)(\mathbf{A}\mathbf{s}_i + \mathbf{e}_i)^\top \rangle = \mathbf{A}\mathbf{A}^\top + \boldsymbol{\Psi} \tag{2.37}$$

One can simply think of the factor analysis model as a reduced parameter Gaussian. A full $(p \times p)$ covariance matrix has $\frac{1}{2}p(p+1)$ *well-determined* parameters (no degeneracies in the parameterization). The number of well-determined parameters in a factor analyzer with latent dimensionality k is

$$d(k) = p(k+1) - \frac{1}{2}k(k-1) \tag{2.38}$$

The first term are the number of parameters in $\mathbf{A}$ an $\boldsymbol{\Psi}$ respectively, and the second term is the number of well-determined parameters in a $(k \times k)$ orthonormal matrix. This last term needs to be subtracted because of a redundancy in the factor analysis parameterization. To see why, lets substitute $\mathbf{A}$ by its rotated version $\mathbf{AU}$. The covariance of $\mathbf{x}_i$ then becomes

$$\langle (\mathbf{AU}\mathbf{s}_i + \mathbf{e}_i)(\mathbf{AU}\mathbf{s}_i + \mathbf{e}_i)^\top \rangle = \mathbf{AU}\mathbf{U}^\top\mathbf{A}^\top + \boldsymbol{\Psi} = \mathbf{A}\mathbf{A}^\top + \boldsymbol{\Psi} \tag{2.39}$$

Thus any orthonormal matrix rotating the factor loadings would produce an equivalent covariance structure. This unfortunate property of non-uniqueness has lead people to the creation of a diversity of rotation criteria. One of the most prominent is the varimax method which iteratively maximizes a quadratic function Φ of the loadings given by

$$\Phi(\mathbf{A}) = \sum_k \left(\frac{1}{p} \sum_j a_{jk}^4 - \left(\frac{1}{p} \sum_j a_{jk}^2 \right)^2 \right) \tag{2.40}$$

Its' rationale is to provide axes with a few large loadings and as many near-zero loadings as possible (see e.g Mardia [28]). Anyhow, it is still suboptimal as we would like the loadings to be uniquely determined. Indeed we will see later that the PARAFAC model is uniquely determined.

Another problem of great concern is how to determine the latent dimensionality k. Cross validation techniques provide an estimate of the model order k by testing the learned FA model on an independent test set for different k. However they are computational costly and we also want to be able to use all the data available to train the model. If maximum likelihood is the method used for learning the factor analysis model, then one can use Akaike's information criterion (AIC) or the Bayesian information criterion (BIC) to estimate the model order. BIC provides a consistent estimate of the model order and is often preferred compared to AIC. The BIC score for factor analysis is given by

$$\ln p(\mathbf{X}|k)_{\mathrm{BIC}} = \sum_{i=1}^{N} \ln p(\mathbf{x}_i|\mathbf{A}, \boldsymbol{\Psi}, k) - \frac{d(k)}{2} \ln N \tag{2.41}$$

where $d(k)$ is given in equation (2.38), N is the sample size and $p(\mathbf{x}_i|\mathbf{A}, \boldsymbol{\Psi}, k)$ is the likelihood function. In chapter 4 we investigate the accuracy of the BIC score when determining the model order of a factor analysis model.

2.3.2 Extended Factor Analysis

When having M data sets available and we believe that these share the same factors, we can write the factor analysis model in stacked form

$$\begin{bmatrix} \mathbf{X}_1 \\ \vdots \\ \mathbf{X}_M \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_M \end{bmatrix} \mathbf{S} + \begin{bmatrix} \mathbf{E}_1 \\ \vdots \\ \mathbf{E}_M \end{bmatrix} \tag{2.42}$$

where $\mathbf{X}_m = \{\mathbf{x}_i^m\}_{i=1}^N$, $\mathbf{S} = \{\mathbf{s}_i\}_{i=1}^N$ and $\mathbf{E}_m = \{\mathbf{e}_i^m\}_{i=1}^N$ are transposed data matrices. Conventionally, the data matrix is $(N \times p)$, but here we consider the transpose $(p \times N)$. If we assume that each data set also have some individual factors and factor loadings, we arrive at the extended factor analysis model

$$\begin{bmatrix} \mathbf{X}_1 \\ \vdots \\ \mathbf{X}_M \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_M \end{bmatrix} \mathbf{S} + \begin{bmatrix} \mathbf{B}_1\mathbf{T}_1 \\ \vdots \\ \mathbf{B}_M\mathbf{T}_M \end{bmatrix} + \begin{bmatrix} \mathbf{E}_1 \\ \vdots \\ \mathbf{E}_M \end{bmatrix} \tag{2.43}$$

where $\mathbf{T}_m$ and $\mathbf{B}_m$ are the individual factors and factor loadings. The individual factors $\mathbf{T}_m$ are assumed Gaussian $\mathcal{N}(\mathbf{t}_i^m \,|\, \mathbf{0}, \mathbf{I})$. The reason for proposing this model is due to the available fMRI data which is analyzed in chapter 4. In $M = 2$ scanning sessions fMRI data was collected on the same subject. Typical artifacts in fMRI are: heart beat, respiration and movement. Since these artifacts are not related to the stimuli and are probably different in the two scanning sessions, individual factors are introduced in the model. The individual factors are then meant to model the correlations introduced by heart beat, breath and movement. The covariance for $M = 2$ is

$$\left\langle \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix}^\top \right\rangle = \begin{bmatrix} \mathbf{A}_1\mathbf{A}_1^\top + \mathbf{B}_1\mathbf{B}_1^\top + \mathbf{\Psi}_1 & \mathbf{A}_1\mathbf{A}_2^\top \\ \mathbf{A}_2\mathbf{A}_1^\top & \mathbf{A}_2\mathbf{A}_2^\top + \mathbf{B}_2\mathbf{B}_2^\top + \mathbf{\Psi}_2 \end{bmatrix} \tag{2.44}$$

Just as for ordinary factor analysis there will be degeneracies in the parameterization that need to be subtracted when calculating the number of well-determined parameters. We denote the latent dimensionality of $\mathbf{T}_m$ by k_T^m and obtain

$$d(k) = Mp(k+1) - \frac{1}{2}k(k-1) + \sum_{m=1}^{M} p(k_T^m + 1) - \frac{1}{2}k_T^m(k_T^m - 1) \tag{2.45}$$

Thus we can again appeal to BIC to give an estimate of the marginal likelihood for different settings of k and k_T^m. When M is large finding k and k_T^m becomes quite cumbersome since we have to run through all combinations of the latent dimensionalities and calculate the log likelihood for all models. As we shall see later when proposing a variational Bayesian algorithm for EFA, the problem of model order selection is solved by placing ARD priors on all factor loading matrices.

2.3.3 Parallel Factor Analysis

The PARAFAC model was proposed in 1970 by Harshman [33] and can be used when the data can be interpreted as being multi-way e.g., (batch $\times$ time $\times$ variable). The model is very constrained because it assumes that the factor loading matrix and factors are the same for all batches, but the factors are allowed to interact by different proportions in each batch. In relation to the fMRI data, this model is only relevant when considering a single scanning session, but for each session we have $M = 10$ batches (repeated experiments). The model is described as

$$\begin{bmatrix} \mathbf{X}_1 \\ \vdots \\ \mathbf{X}_M \end{bmatrix} = \begin{bmatrix} \mathbf{AC}_1 \\ \vdots \\ \mathbf{AC}_M \end{bmatrix} \mathbf{S} + \begin{bmatrix} \mathbf{E}_1 \\ \vdots \\ \mathbf{E}_M \end{bmatrix} \tag{2.46}$$

where $\mathbf{C}_m$ is a diagonal matrix that scales the factors for each batch and $\mathbf{E}_m$ is noise. Yes, this is very constrained, but the model has a nice property - it is unique up to a permutation and scaling of the factors. Permutation and scaling indeterminacy is fundamental for all linear mixture generative models and usually not a matter of great concern. Suppose that $\mathbf{A}$ and $\mathbf{S}$ were rotated by orthonormal matrices $\mathbf{U}$ and $\mathbf{V}$

$$\mathbf{X}_m = \mathbf{A}\mathbf{U}\mathbf{U}^\top\mathbf{C}_m\mathbf{V}\mathbf{V}^\top\mathbf{S} + \mathbf{E}_m \tag{2.47}$$

This would imply that the model could just as well be expressed by the factor loadings $\mathbf{AU}$, the scaling matrix $\mathbf{U}^\top\mathbf{C}_m\mathbf{V}$ and factors $\mathbf{V}^\top\mathbf{S}$. However, for the PARAFAC model to hold, the scaling matrix $\mathbf{U}^\top\mathbf{C}_m\mathbf{V}$ must be diagonal. Because of this requirement, the only $\mathbf{U}$ and $\mathbf{V}$ matrices that are valid are those that preserve the diagonality. In practice, this means that $\mathbf{U}$ and $\mathbf{V}$ has to be permutation or scaling matrices.

The model is usually estimated by a least squares technique called alternate least squares (ALS). This is an iterative algorithm that successively finds a least squares solution for a subset of parameters $\{\mathbf{A}\}$, $\{\mathbf{C}_m\}$ or $\{\mathbf{S}\}$ and holding the others fixed. $\mathbf{E}_m$ is regarded as noise residuals.

In the literature, there has been no attempts, as far as I know, to put the estimation of the model in a probabilistic framework by assuming a distribution for the data. Estimation by least squares is indirectly the same as assuming Gaussian noise. It is well known that the maximum likelihood solution to some deterministic function of the data corrupted by Gaussian noise is equivalent to the least squares solution (see e.g., Bishop [6]). In accordance with the assumptions made in ordinary factor analysis, I assume the factors to be Gaussian distributed as $\mathcal{N}(\mathbf{s}_i \,|\, \mathbf{0}, \mathbf{I})$ and the noise being distributed as $\mathcal{N}(\mathbf{e}_i \,|\, \mathbf{0}, \mathbf{\Psi})$. Again $\mathbf{\Psi}$ is assumed diagonal. With these assumptions the covariance for $M = 2$ is

$$\left\langle \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix}^\top \right\rangle = \begin{bmatrix} \mathbf{A}\mathbf{C}_1^2\mathbf{A}^\top + \mathbf{\Psi}_1 & \mathbf{A}\mathbf{C}_1\mathbf{C}_2\mathbf{A}^\top \\ \mathbf{A}\mathbf{C}_1\mathbf{C}_2\mathbf{A}^\top & \mathbf{A}\mathbf{C}_2^2\mathbf{A}^\top + \mathbf{\Psi}_2 \end{bmatrix} \tag{2.48}$$

The number of well-determined parameters is given by

$$d(k) = pk + M(k + p) \tag{2.49}$$

The first term is the number of parameters in $\mathbf{A}$ and the second term is M times the number of parameters in $\mathbf{C}_m$ and $\mathbf{\Psi}_m$. As an example suppose that $M = 2$ data sets were available and the dimension was $p = 10$. Assuming $k = 3$ factors, the number of well-determined parameters are; stacked factor analysis: $d(k) = Mp(k+1) - \frac{1}{2}k(k-1) = 77$, PARAFAC: $d(k) = pk + M(k+p) = 56$. Thus PARAFAC is obviously less flexible.

In chapter 3 an EM algorithm and a variational Bayesian algorithm are derived.

Chapter 3

Algorithms

This chapter describes the learning algorithms derived for FA, EFA and PARAFAC. The first section describes the EM and VBEM algorithms for FA. The second and third section describes in a similar manner the EM and VBEM algorithms for EFA and PARAFAC. All the models are visualized as directed acyclic graphs (DAG) also called graphical models. A graphical model show the dependencies between variables in a model. Circles are used to represent stochastic variables and square boxes denotes deterministic variables. The plates (rounded boxes) are used for representing repetitions of a variable.

3.1 Maximum Likelihood Factor Analysis

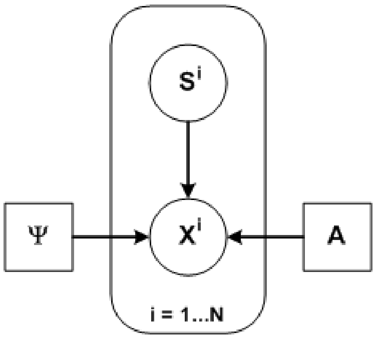


Figure 3.1: Graphical model for maximum likelihood factor analysis

The generative model for maximum likelihood factor analysis is given by

$$\mathbf{X} = \mathbf{AS} + \mathbf{E} \tag{3.1}$$

where the model assumptions in factor analysis implies that $\mathbf{x}_i$ is Gaussian distributed as

$$\begin{aligned}
p(\mathbf{x}_i|\mathbf{s}_i, \mathbf{A}, \boldsymbol{\Psi}) &= (2\pi)^{-\frac{p}{2}} |\boldsymbol{\Psi}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x}_i - \mathbf{A}\mathbf{s}_i)^\top \boldsymbol{\Psi}^{-1} (\mathbf{x}_i - \mathbf{A}\mathbf{s}_i)\right) \\
&\propto \exp\left(\mathbf{s}_i^\top \mathbf{A}^\top \boldsymbol{\Psi}^{-1}\mathbf{x}_i - \frac{1}{2}\mathbf{s}_i^\top \mathbf{A}^\top \boldsymbol{\Psi}^{-1}\mathbf{A}\mathbf{s}_i\right) \qquad (3.2)
\end{aligned}$$

The posterior for the factors can be found by multiplying the prior

$$\begin{aligned}
p(\mathbf{s}_i|\mathbf{x}_i, \mathbf{A}, \boldsymbol{\Psi}) &\propto p(\mathbf{x}_i|\mathbf{s}_i, \mathbf{A}, \boldsymbol{\Psi})p(\mathbf{s}_i) \\
&\propto \exp\left(\mathbf{s}_i^\top \mathbf{A}^\top \boldsymbol{\Psi}^{-1}\mathbf{x}_i - \frac{1}{2}\mathbf{s}_i^\top \left(\mathbf{I} + \mathbf{A}^\top \boldsymbol{\Psi}^{-1}\mathbf{A}\right)\mathbf{s}_i\right)
\end{aligned} \qquad (3.3)$$

Thus, due to conjugacy the posterior is again Gaussian. A Gaussian distribution can be written as $p(\mathbf{y}_i|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \propto \exp(\mathbf{y}_i^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} - \frac{1}{2}\mathbf{y}_i^\top \boldsymbol{\Sigma}^{-1}\mathbf{y}_i)$. Using this and equation (3.3) we can easily identify the posterior and its' moments. Finding the moments of $p(\mathbf{s}_i|\mathbf{x}_i, \mathbf{A}, \boldsymbol{\Psi})$ maintain the E-step of the EM algorithm and are shown in equation (3.10) - (3.12).

The M-step involves maximizing the expected complete log likelihood which can be written

$$\begin{aligned}
\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S})\rangle = -\frac{N}{2}\ln|\boldsymbol{\Psi}| - \frac{1}{2}\sum_i^N \Big(&\mathbf{x}_i^\top \boldsymbol{\Psi}^{-1}\mathbf{x}_i - 2\mathbf{x}_i^\top \boldsymbol{\Psi}^{-1}\mathbf{A}\langle \mathbf{s}_i\rangle \\
&+ \mathrm{tr}\left[\mathbf{A}^\top \boldsymbol{\Psi}^{-1}\mathbf{A}\langle \mathbf{s}_i\mathbf{s}_i^\top\rangle\right]\Big) + \mathrm{const}
\end{aligned} \qquad (3.4)$$

Now, taking partial derivatives with respect to the parameters we get

$$\frac{\partial\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S})\rangle}{\partial \mathbf{A}} = \sum_i^N \boldsymbol{\Psi}^{-1}\mathbf{x}_i\langle \mathbf{s}_i\rangle^\top - \boldsymbol{\Psi}^{-1}\mathbf{A}\langle \mathbf{s}_i\mathbf{s}_i^\top\rangle \qquad (3.5)$$

$$\frac{\partial\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S})\rangle}{\partial \boldsymbol{\Psi}^{-1}} = \frac{N}{2}\boldsymbol{\Psi} - \sum_i^N \left(\frac{1}{2}\mathbf{x}_i\mathbf{x}_i^\top - \mathbf{A}\langle \mathbf{s}_i\rangle\mathbf{x}_i^\top + \frac{1}{2}\mathbf{A}\langle \mathbf{s}_i\mathbf{s}_i^\top\rangle\mathbf{A}^\top\right) \qquad (3.6)$$

Setting the above equations to zero, rearranging and substituting equation (3.5) in (3.6) we get the update equations for the M-step which are shown in equation (3.13) and (3.14). To ensure that $\boldsymbol{\Psi}$ is diagonal we set off diagonal terms to zero by the diag[$\cdot$] operator as seen in equation (3.14) of the M-step.

The log likelihood can be written

$$\mathcal{L}(\boldsymbol{\theta}) = -\frac{Np}{2}\ln(2\pi) - \frac{N}{2}\ln\left|\mathbf{A}\mathbf{A}^\top + \boldsymbol{\Psi}\right| - \frac{1}{2}\operatorname{tr}\left[\left(\mathbf{A}\mathbf{A}^\top + \boldsymbol{\Psi}\right)^{-1}\mathbf{X}\mathbf{X}^\top\right] \tag{3.7}$$

which can be efficiently computed by appealing to the Sherman-Morrison-Woodbury formula

$$\left(\mathbf{A}\mathbf{A}^\top + \boldsymbol{\Psi}\right)^{-1} = \boldsymbol{\Psi}^{-1} - \boldsymbol{\Psi}^{-1}\mathbf{A}\left(\mathbf{I} + \mathbf{A}^\top\boldsymbol{\Psi}^{-1}\mathbf{A}\right)^{-1}\mathbf{A}^\top\boldsymbol{\Psi}^{-1} \tag{3.8}$$

and the Cholesky decomposition $\mathbf{U}\mathbf{U}^\top = \left(\mathbf{A}\mathbf{A}^\top + \boldsymbol{\Psi}\right)$ which leads to

$$\ln\left|\mathbf{A}\mathbf{A}^\top + \boldsymbol{\Psi}\right| = \sum_{j=1}^{p}\ln u_{jj} \tag{3.9}$$

EM Algorithm

E-step:

$$\boldsymbol{\Sigma}_\mathbf{S} = \left(\mathbf{I} + \mathbf{A}^\top\boldsymbol{\Psi}^{-1}\mathbf{A}\right)^{-1} \tag{3.10}$$

$$\langle\mathbf{S}\rangle = \boldsymbol{\Sigma}_\mathbf{S}\mathbf{A}^\top\boldsymbol{\Psi}^{-1}\mathbf{X} \tag{3.11}$$

$$\langle\mathbf{S}\mathbf{S}^\top\rangle = N\boldsymbol{\Sigma}_\mathbf{S} + \langle\mathbf{S}\rangle\langle\mathbf{S}\rangle^\top \tag{3.12}$$

M-step:

$$\mathbf{A} = \mathbf{X}\langle\mathbf{S}\rangle^\top\langle\mathbf{S}\mathbf{S}^\top\rangle^{-1} \tag{3.13}$$

$$\boldsymbol{\Psi} = \frac{1}{N}\operatorname{diag}\left[\mathbf{X}\mathbf{X}^\top - \mathbf{A}\langle\mathbf{S}\rangle\mathbf{X}^\top\right] \tag{3.14}$$

3.2 Variational Bayesian Factor Analysis

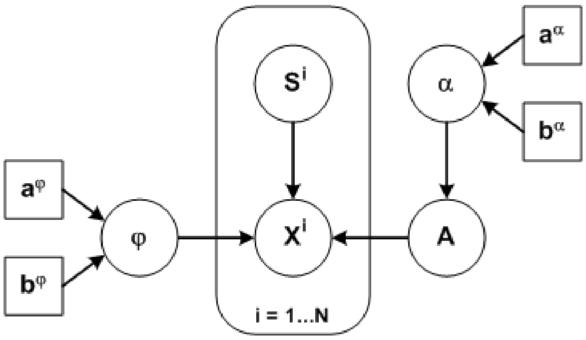


Figure 3.2: Graphical model for variational Bayesian factor analysis

The model proposed for variational Bayesian factor analysis (VBFA) can be seen on figure 3.2. Compared to MLFA we have now placed distributions on the factor loading matrix and the noise covariance. The noise covariance matrix $\mathbf{\Psi}$ in MLFA is replaced by its inverse, the noise precision matrix $\boldsymbol{\varphi} = \mathbf{\Psi}^{-1}$. In order to do automatic latent dimensionality determination a hierarchical prior is used on the factor loading matrix $\mathbf{A}$. The $\boldsymbol{\alpha}$ node serves as regularization parameter that effectively can shut off unneeded factors. The priors in the model are defined as

$$p(\mathbf{A} \,|\, \boldsymbol{\alpha}) = \prod_{k=1}^{K} \mathcal{N}\big(\mathbf{a}_k \,|\, \mathbf{0}, \tfrac{1}{\alpha_k}\mathbf{I}\big) \tag{3.15}$$

$$p(\boldsymbol{\alpha} \,|\, \mathbf{a}^\alpha, \mathbf{b}^\alpha) = \prod_{k=1}^{K} \mathcal{G}\big(\alpha_k \,|\, a_k^\alpha, b_k^\alpha\big) \tag{3.16}$$

$$p(\boldsymbol{\varphi} \,|\, \mathbf{a}^\varphi, \mathbf{b}^\varphi) = \prod_{j=1}^{p} \mathcal{G}\big(\varphi_j \,|\, a_j^\varphi, b_j^\varphi\big) \tag{3.17}$$

where $\mathbf{a}_k$ denotes a column vector that corresponds to the k^{th} column in $\mathbf{A}$. We have an isotropic Gaussian on each column in $\mathbf{A}$, where the hyper-parameter α_k controls the precision. Since each column in $\mathbf{A}$ is given zero mean, what will happen if α_k goes to infinity is that the variance for the k^{th} column will go to zero. When the variance for the k^{th} column goes to zero we effectively shut down the k^{th} factor. One can think of this as an ARD prior that can switch off unneeded factors, where the factors are the inputs to the system. Both $\boldsymbol{\alpha}$ and $\boldsymbol{\varphi}$ are precisions, so they are given factorized Gamma priors. Remember that $\boldsymbol{\varphi}$ is a diagonal matrix and the prior is therefore only specified for the diagonal elements.

The log marginal likelihood for the model we lower bound as

$$\begin{aligned}\mathcal{L} &= \ln \int p(\mathbf{X}, \mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi}) d\mathbf{S} d\mathbf{A} d\boldsymbol{\alpha} d\boldsymbol{\varphi} \\ &\geq \int q(\mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi}) \ln \frac{p(\mathbf{X}, \mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi})}{q(\mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi})} d\mathbf{S} d\mathbf{A} d\boldsymbol{\alpha} d\boldsymbol{\varphi} \\ &= \mathcal{F}_m(q) \end{aligned} \tag{3.18}$$

where $q(\mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi})$ is our approximation to the true posterior $p(\mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi}|\mathbf{X})$ which is not analytically tractable. To derive an VBEM algorithm we need to the assume that the hidden variables and the parameters are independent $q(\mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi}) \approx q(\mathbf{S}) q(\mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi})$. Furthermore we approximate $q(\mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi}) \approx q(\mathbf{A}) q(\boldsymbol{\alpha}) q(\boldsymbol{\varphi})$ so the resulting variational posterior is fully factorized. This further approximation significantly eases the derivation since now all variational posteriors are on the same form as the conjugate priors.

The complete log likelihood for the model is given by

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) = \frac{N}{2} \ln |\boldsymbol{\varphi}| &- \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{A}\mathbf{s}_i)^\top \boldsymbol{\varphi} (\mathbf{x}_i - \mathbf{A}\mathbf{s}_i) - \frac{1}{2} \sum_{i=1}^{N} \mathbf{s}_i^\top \mathbf{s}_i \\ &+ \frac{p}{2} \sum_{k=1}^{K} \ln \alpha_k - \frac{1}{2} \sum_{k=1}^{K} \alpha_k \mathbf{a}_k^\top \mathbf{a}_k \\ &+ \sum_{k=1}^{K} (a_k^\alpha - 1) \ln \alpha_k - \sum_{k=1}^{K} b_k^\alpha \alpha_k \\ &+ \sum_{j=1}^{p} (a_j^\varphi - 1) \ln \varphi_j - \sum_{j=1}^{p} b_j^\varphi \varphi_j + \text{const} \end{aligned} \tag{3.19}$$

The VBE-step

To derive the VBE-step we proceed by writing up the expected complete log likelihood $\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle_{q(\boldsymbol{\theta})}$ with respect to the parameter posteriors and neglect terms not depending on the hidden variables

$$\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle_{q(\boldsymbol{\theta})} = \sum_{i=1}^{N} \mathbf{s}_i^\top \langle \mathbf{A} \rangle^\top \langle \boldsymbol{\varphi} \rangle \mathbf{x}_i - \frac{1}{2} \mathbf{s}_i^\top \left(\mathbf{I} + \langle \mathbf{A}^\top \boldsymbol{\varphi} \mathbf{A} \rangle \right) \mathbf{s}_i + \text{const} \tag{3.20}$$

from which we can infer that $q(\mathbf{S})$ is of the form

$$q(\mathbf{S}) = \prod_{i=1}^{N} q(\mathbf{s}_i) = \prod_{i=1}^{N} \mathcal{N} \left(\mathbf{s}_i \,|\, \mathbf{m}_{\mathbf{s}}^{(i)}, \boldsymbol{\Sigma}_{\mathbf{s}} \right) \tag{3.21}$$

where

$$\begin{aligned} \mathbf{\Sigma_s} &= \left(\mathbf{I} + \langle \mathbf{A}^\top \boldsymbol{\varphi} \mathbf{A} \rangle\right)^{-1} && (3.22) \\ \mathbf{m}_\mathbf{s}^{(i)} &= \mathbf{\Sigma_s} \langle \mathbf{A} \rangle^\top \langle \boldsymbol{\varphi} \rangle \mathbf{x}_i && (3.23) \end{aligned}$$

To compute the expectation $\langle \mathbf{A}^\top \boldsymbol{\varphi} \mathbf{A} \rangle$ we use the fact that $\boldsymbol{\varphi}$ is diagonal

$$\langle \mathbf{A}^\top \boldsymbol{\varphi} \mathbf{A} \rangle = \sum_{j=1}^{p} \langle \varphi_j \rangle \langle \mathbf{a}_j \mathbf{a}_j^\top \rangle \tag{3.24}$$

where $\mathbf{a}_j$ is a column vector corresponding to the j^{th} row of $\mathbf{A}$. Note that the VBE-step is equivalent to the E-step in MLFA, but with expected values for the parameters. This concludes the VBE-step.

The VBM-step

To derive the VBM-step we need to write the expected complete log likelihood $\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle_{q(\boldsymbol{\theta}_{j \neq i}) q(\mathbf{S})}$ with respect to all other variational posteriors. The derivations follow below.

The $q(\mathbf{A})$ distribution
In order to estimate $q(\mathbf{A})$ we start from equation (3.19) and retain only the terms that contain $\mathbf{A}$. To infer the posterior we furthermore have to write the expected complete likelihood as a sum over the rows of $\mathbf{A}$. Let $\mathbf{a}_j$ denote a column vector that corresponds to the j^{th} row in $\mathbf{A}$ and let $\overline{\overline{\boldsymbol{\alpha}}} = \operatorname{diag}[\boldsymbol{\alpha}]$ we get

$$\begin{aligned} \langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle_{q(\boldsymbol{\alpha}) q(\boldsymbol{\varphi}) q(\mathbf{S})} = \sum_{j=1}^{p} \Bigg[& \mathbf{a}_j^\top \langle \varphi_j \rangle \left(\sum_{i=1}^{N} x_{ji} \langle \mathbf{s}_i \rangle \right) \\ & - \frac{1}{2} \mathbf{a}_j^\top \left(\langle \varphi_j \rangle \sum_{i=1}^{N} \langle \mathbf{s}_i \mathbf{s}_i^\top \rangle + \langle \overline{\overline{\boldsymbol{\alpha}}} \rangle \right) \mathbf{a}_j \Bigg] + \text{const} \end{aligned} \tag{3.25}$$

from which we can infer that $q(\mathbf{A})$ is of the form

$$q(\mathbf{A}) = \prod_{j=1}^{p} q(\mathbf{a}_j) = \prod_{j=1}^{p} \mathcal{N}\left(\mathbf{a}_j \,|\, \mathbf{m}_\mathbf{a}^{(j)}, \mathbf{\Sigma}_\mathbf{a}^{(j)}\right) \tag{3.26}$$

where

$$\begin{aligned}\boldsymbol{\Sigma}_{\mathbf{a}}^{(j)} &= \left(\langle\varphi_j\rangle\sum_{i=1}^{N}\langle\mathbf{s}_i\mathbf{s}_i^\top\rangle + \langle\,\overline{\overline{\boldsymbol{\alpha}}}\,\rangle\right)^{-1} && (3.27)\\ \mathbf{m}_{\mathbf{a}}^{(j)} &= \boldsymbol{\Sigma}_{\mathbf{a}}^{(j)}\langle\varphi_j\rangle\left(\sum_{i=1}^{N}x_{ji}\langle\mathbf{s}_i\rangle\right) && (3.28)\end{aligned}$$

The $q(\boldsymbol{\alpha})$ distribution

In order to estimate $q(\boldsymbol{\alpha})$ we start again from equation (3.19) and retain only the terms that contain $\boldsymbol{\alpha}$. The k^{th} column of $\mathbf{A}$ is denoted by $\mathbf{a}_k$

$$\langle\mathcal{L}(\boldsymbol{\theta},\mathbf{S})\rangle_{q(\mathbf{A})q(\boldsymbol{\varphi})q(\mathbf{S})} = \sum_{k=1}^{K}(a_k^\alpha+\frac{p}{2}-1)\ln\alpha_k - \sum_{k=1}^{K}\left(b_k^\alpha+\frac{\langle\mathbf{a}_k^\top\mathbf{a}_k\rangle}{2}\right)\alpha_k + \text{const} \tag{3.29}$$

from which we can infer that $q(\boldsymbol{\alpha})$ is of the form

$$q(\boldsymbol{\alpha}) = \prod_{k=1}^{K}q(\alpha_k) = \prod_{k=1}^{K}\mathcal{G}\left(\alpha_k\,|\,\widehat{a}_k^\alpha,\widehat{b}_k^\alpha\right) \tag{3.30}$$

where

$$\begin{aligned}\widehat{a}_k^\alpha &= a_k^\alpha + \frac{p}{2} && (3.31)\\ \widehat{b}_k^\alpha &= b_k^\alpha + \frac{\langle\mathbf{a}_k^\top\mathbf{a}_k\rangle}{2} && (3.32)\end{aligned}$$

We can compute the expectation $\langle\mathbf{a}_k^\top\mathbf{a}_k\rangle$ by

$$\langle\mathbf{a}_k^\top\mathbf{a}_k\rangle = \left(\sum_{j=1}^{p}\boldsymbol{\Sigma}_{\mathbf{a}}^{(j)} + \mathbf{m}_{\mathbf{a}}^{(j)}\mathbf{m}_{\mathbf{a}}^{(j)\top}\right)_{(k,k)} \tag{3.33}$$

The $q(\boldsymbol{\varphi})$ distribution

In order to estimate $q(\boldsymbol{\varphi})$ we start again from equation (3.19) and retain only the terms that contain $\boldsymbol{\varphi}$.

$$\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle_{q(\mathbf{A})q(\boldsymbol{\alpha})q(\mathbf{S})} = \sum_{j=1}^{p} (a_j^{\varphi} + \frac{N}{2} - 1) \ln \varphi_j - \sum_{j=1}^{p} \left(b_j^{\varphi} + \frac{1}{2} \sum_{i=1}^{N} \left\langle (x_{ji} - \mathbf{a}_j^{\top} \mathbf{s}_i)^2 \right\rangle \right) \varphi_j + \text{const} \tag{3.34}$$

from which we can infer that $q(\boldsymbol{\varphi})$ is of the form

$$q(\boldsymbol{\varphi}) = \prod_{j=1}^{p} q(\varphi_j) = \prod_{j=1}^{p} \mathcal{G} \left(\varphi_j \,|\, \widehat{a}_k^{\varphi}, \widehat{b}_k^{\varphi} \right) \tag{3.35}$$

where

$$\widehat{a}_j^{\varphi} = a_j^{\varphi} + \frac{N}{2} \tag{3.36}$$

$$\widehat{b}_j^{\varphi} = b_j^{\varphi} + \frac{1}{2} \sum_{i=1}^{N} \left\langle (x_{ji} - \mathbf{a}_j^{\top} \mathbf{s}_i)^2 \right\rangle \tag{3.37}$$

We can compute the expectation $\left\langle (x_{ji} - \mathbf{a}_j^{\top} \mathbf{s}_i)^2 \right\rangle$ by

$$\left\langle (x_{ji} - \mathbf{a}_j^{\top} \mathbf{s}_i)^2 \right\rangle = x_{ji}^2 - 2x_{ji} \langle \mathbf{a}_j \rangle^{\top} \langle \mathbf{s}_i \rangle + \operatorname{tr} \left[\langle \mathbf{a}_j \mathbf{a}_j^{\top} \rangle \langle \mathbf{s}_i \mathbf{s}_i^{\top} \rangle \right] \tag{3.38}$$

This concludes the VBM-step.

Hyperparameter Optimization

The model has hyperparameters $\{a_j^{\varphi}, b_j^{\varphi}\}$ and hyperhyperparameters $\{a_k^{\alpha}, b_k^{\alpha}\}$. We can optimize these by writing the expected complete log likelihood as a function of only these parameters, differentiate and set to zero. Writing the expected complete log likelihood as a function of $\{a_j^{\varphi}, b_j^{\varphi}\}$ and neglecting constant terms we get the fixed point equations

$$\langle \mathcal{L}(a_j^{\varphi}, b_j^{\varphi}) \rangle_{q(\boldsymbol{\theta})q(\mathbf{S})} = a_j^{\varphi} \ln b_j^{\varphi} - \ln \Gamma(a_j^{\varphi}) + (a_j^{\varphi} - 1) \langle \ln \varphi_j \rangle - b_j^{\varphi} \langle \varphi_j \rangle \tag{3.39}$$

$$\frac{\partial \langle \mathcal{L}(a_j^{\varphi}, b_j^{\varphi}) \rangle_{q(\boldsymbol{\theta})q(\mathbf{S})}}{\partial a_j^{\varphi}} = 0 \quad \Rightarrow \quad \psi(a_j^{\varphi}) = \ln b_j^{\varphi} + \langle \ln \varphi_j \rangle \tag{3.40}$$

$$\frac{\partial \langle \mathcal{L}(a_j^{\varphi}, b_j^{\varphi}) \rangle_{q(\boldsymbol{\theta})q(\mathbf{S})}}{\partial a_j^{\varphi}} = 0 \quad \Rightarrow \quad \frac{1}{b_j^{\varphi}} = \frac{\langle \varphi_j \rangle}{a_j^{\varphi}} \tag{3.41}$$

where $\psi(x) = \frac{\partial}{\partial x} \ln \Gamma(x)$ is the digamma function and $\langle \ln \varphi_j \rangle = \psi(\widehat{a}^{\varphi}_j) - \ln \widehat{b}^{\varphi}_j$. Since the priors for $\boldsymbol{\alpha}$ and $\boldsymbol{\varphi}$ are of same type, the fixed point equations for $\{a^{\alpha}_k, b^{\alpha}_k\}$ are identical.

We can solve the fixed point equations by Newton-Raphson, but we must assure that the Newton-Raphson iterations yield a solution where both a^{φ}_j and b^{φ}_j remains positive. This we can overcome by performing the iterations in the exponential domain. In this way subtraction becomes multiplication and we are assured that we find a solution with positive values. The iterations are given by

$$a^{\varphi}_{j,\text{new}} \leftarrow a^{\varphi}_j \exp\left[-\frac{\psi(a^{\varphi}_j) - \ln a^{\varphi}_j + \ln \langle \varphi_j \rangle - \langle \ln \varphi_j \rangle}{a^{\varphi}_j \psi'(a^{\varphi}_j) - 1}\right] \tag{3.42}$$

where $\psi'(\cdot)$ is the derivative of the digamma function. When $a^{\varphi}_{j,\text{new}}$ is estimated we can simply insert in equation (3.41) to find the update for $b^{\varphi}_{j,\text{new}}$. If we propose that the model should have equivalent priors, meaning that all $a^{\varphi}_j = a^{\varphi}_{j'}$ and $b^{\varphi}_j = b^{\varphi}_{j'}$ we simply just replace $\langle \ln \varphi_j \rangle$ by $\frac{1}{p}\sum_{j=1}^{p} \langle \ln \varphi_j \rangle$ and $\langle \varphi_j \rangle$ by $\frac{1}{p}\sum_{j=1}^{p} \langle \varphi_j \rangle$ in equation (3.42). In chapter 4, the effect of hyperparameter optimization is investigated for both type of priors.

Calculation of $\mathcal{F}_m$

We need to calculate the lower bound $\mathcal{F}_m$ if we want to use it as guide for model selection and to monitor convergence. The functional can be written

$$\begin{aligned}\mathcal{F}_m = &\langle \ln p(\mathbf{X}|\mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi}) \rangle_{q(\mathbf{S})q(\mathbf{A})q(\boldsymbol{\alpha})q(\boldsymbol{\varphi})} \\ &- \text{KL}(q(\mathbf{S})||p(\mathbf{S})) - \langle \text{KL}(q(\mathbf{A})||p(\mathbf{A}|\boldsymbol{\alpha})) \rangle_{q(\boldsymbol{\alpha})} \\ &- \text{KL}(q(\boldsymbol{\alpha})||p(\boldsymbol{\alpha} \,|\, \mathbf{a}^{\alpha}, \mathbf{b}^{\alpha})) - \text{KL}(q(\boldsymbol{\varphi})||p(\boldsymbol{\varphi} \,|\, \mathbf{a}^{\varphi}, \mathbf{b}^{\varphi}))\end{aligned} \tag{3.43}$$

where the individual terms can be calculated by

$$\begin{aligned}&\langle \ln p(\mathbf{X}|\mathbf{S}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi}) \rangle_{q(\mathbf{S})q(\mathbf{A})q(\boldsymbol{\alpha})q(\boldsymbol{\varphi})} = \\ &\qquad -\frac{Np}{2}\ln(2\pi) + \frac{N}{2}\sum_{j=1}^{p} \langle \ln \varphi_j \rangle + \sum_{j=1}^{p} \left(\langle \varphi_j \rangle b^{\varphi}_j - \widehat{a}^{\varphi}_j \right)\end{aligned} \tag{3.44}$$

$$\text{KL}(q(\mathbf{S})||p(\mathbf{S})) = -\frac{N}{2} \ln |\boldsymbol{\Sigma}_{\mathbf{s}}| - \frac{N}{2}\text{tr}\left[\mathbf{I} - \boldsymbol{\Sigma}_{\mathbf{s}}\right] + \frac{1}{2}\sum_{i=1}^{N} \text{tr}\left[\mathbf{m}^{(i)}_{\mathbf{s}} \mathbf{m}^{(i)\top}_{\mathbf{s}}\right] \tag{3.45}$$

$$\begin{aligned}\langle \mathrm{KL}(q(\mathbf{A})||p(\mathbf{A}|\boldsymbol{\alpha}))\rangle_{q(\boldsymbol{\alpha})} = &-\frac{p}{2}\sum_{k=1}^{K}\left(\psi(\widehat{a}_k^{\alpha}) - \ln \widehat{b}_k^{\alpha}\right)\\ &-\frac{1}{2}\sum_{j=1}^{p}\left(\ln|\boldsymbol{\Sigma}_{\mathbf{a}}^{(j)}| + \mathrm{tr}\left[\mathbf{I} - \left(\boldsymbol{\Sigma}_{\mathbf{a}}^{(j)} + \mathbf{m}_{\mathbf{a}}^{(j)}\mathbf{m}_{\mathbf{a}}^{(j)\top}\right)\langle\overline{\overline{\boldsymbol{\alpha}}}\rangle\right]\right)\end{aligned} \tag{3.46}$$

$$\begin{aligned}\mathrm{KL}(q(\boldsymbol{\alpha})||p(\boldsymbol{\alpha}\,|\,\mathbf{a}^{\alpha},\mathbf{b}^{\alpha}) = \sum_{k=1}^{K}\Bigg(&\widehat{a}_k^{\alpha}\ln\widehat{b}_k^{\alpha} - a_k^{\alpha}\ln b_k^{\alpha} - \ln\frac{\Gamma(\widehat{a}_k^{\alpha})}{\Gamma(a_k^{\alpha})}\\ &+ b_k^{\alpha}\langle\alpha_k\rangle - \widehat{a}_k^{\alpha} + (\widehat{a}_k^{\alpha} - a_k^{\alpha})(\psi(\widehat{a}_k^{\alpha}) - \ln\widehat{b}_k^{\alpha})\Bigg)\end{aligned} \tag{3.47}$$

$$\begin{aligned}\mathrm{KL}(q(\boldsymbol{\varphi})||p(\boldsymbol{\varphi}\,|\,\mathbf{a}^{\varphi},\mathbf{b}^{\varphi}) = \sum_{j=1}^{p}\Bigg(&\widehat{a}_j^{\varphi}\ln\widehat{b}_j^{\varphi} - a_j^{\varphi}\ln b_j^{\varphi} - \ln\frac{\Gamma(\widehat{a}_j^{\varphi})}{\Gamma(a_j^{\varphi})}\\ &+ b_j^{\varphi}\langle\varphi_j\rangle - \widehat{a}_j^{\varphi} + (\widehat{a}_j^{\varphi} - a_j^{\varphi})(\psi(\widehat{a}_j^{\varphi}) - \ln\widehat{b}_j^{\varphi})\Bigg)\end{aligned} \tag{3.48}$$

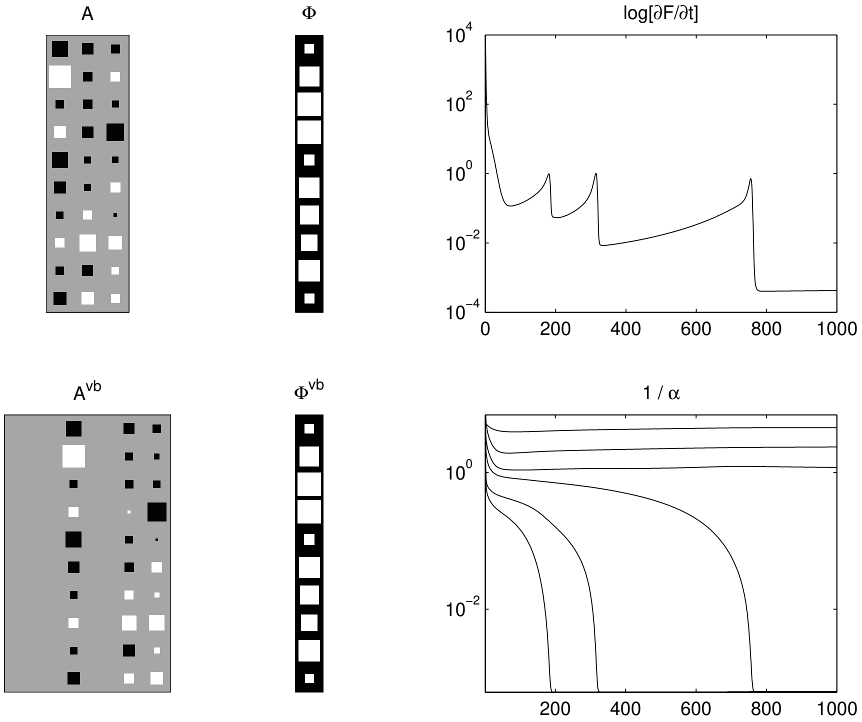


Figure 3.3: Upper left show the reference factor loading matrix and noise precisions. Bottom left show the learnt factor loading matrix and noise precisions. Upper right show the log to the difference of $\mathcal{F}_m$ during 10^3 iterations. Plotting $\mathcal{F}_m$ in this way its more clear when the bound increases. Bottom right show the inverse precisions (variances) on the columns of $\mathbf{A}$. Notice that $\mathcal{F}_m$ increases faster when a column is on its way to be shut down.

VBFA in Action

To demonstrate the model I created a *reference* factor analysis model with precisions drawn from Gamma distributions and a factor loading matrix drawn from a Gaussian distribution controlled by the precisions. From this model I generated a $p = 10$ dimensional training set of size $N = 10^3$. The reference latent dimensionality was $K = 3$. No hyperparameter optimization was invoked. The hyperparameters were set as $a_k^\alpha = b_k^\alpha = b_j^\varphi = b_j^\varphi = 10^{-3}$ corresponding to non-informative priors for all practical purposes. The algorithm was run for 10^3 iterations with a maximum dimensionality of $k_{max} = 6$ and a random initialization. Figure 3.3 summarize the results. The learnt latent dimensionality is $K = 3$ since only three columns are left back. Notice also that the columns in the learnt factor loading matrix resemble the reference model (no rotation indeterminacy). In chapter 4 we will discuss this further.

3.3 Maximum Likelihood Extended Factor Analysis

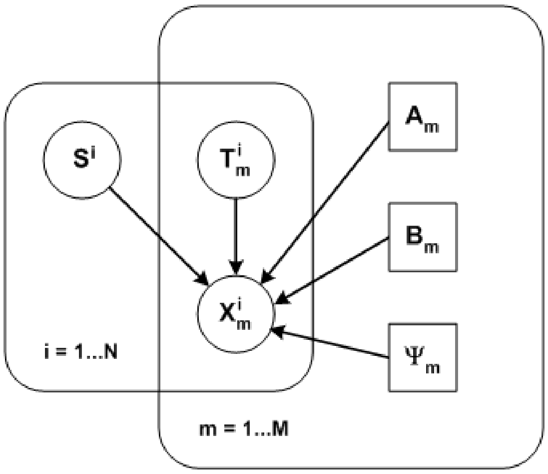


Figure 3.4: Graphical model for maximum likelihood extended factor analysis

The generative model for maximum likelihood extended factor analysis is given by

$$\mathbf{X}_m = \mathbf{A}_m \mathbf{S} + \mathbf{B}_m \mathbf{T}_m + \mathbf{E}_m \tag{3.49}$$

where the model assumptions in extended factor analysis implies that $\{\mathbf{x}_m^i\}_{m=1}^M$ is Gaussian distributed as

$$p(\{\mathbf{x}_m^i\}_{m=1}^M | \, \mathbf{s}^i, \{\mathbf{t}_m^i, \mathbf{A}_m, \mathbf{B}_m, \boldsymbol{\Psi}_m\}_{m=1}^M) \propto \\ \exp\left(-\frac{1}{2} \sum_{m=1}^M \Big(\mathbf{x}_m^i - \mathbf{A}_m \mathbf{s}^i - \mathbf{B}_m \mathbf{t}_m^i \Big)^\top \boldsymbol{\Psi}_m^{-1} \Big(\mathbf{x}_m^i - \mathbf{A}_m \mathbf{s}^i - \mathbf{B}_m \mathbf{t}_m^i \Big) \right) \tag{3.50}$$

To identify the E-step we write down the posterior for the factors $\mathbf{s}^i$ by multiplying the likelihood by the prior and neglect constant terms

$$p(\mathbf{s}^i | \, \{\mathbf{x}_m^i, \mathbf{t}_m^i, \mathbf{A}_m, \mathbf{B}_m, \boldsymbol{\Psi}_m\}_{m=1}^M) \propto \\ \exp\left(\mathbf{s}^{i\top} \left(\sum_{m=1}^M \mathbf{A}_m^\top \boldsymbol{\Psi}_m^{-1} \left(\mathbf{x}_m^i - \mathbf{B}_m \mathbf{t}_m^i \right) \right) - \frac{1}{2} \mathbf{s}^{i\top} \left(\mathbf{I} + \sum_{m=1}^M \mathbf{A}_m^\top \boldsymbol{\Psi}_m^{-1} \mathbf{A}_m \right) \mathbf{s}^i \right) \tag{3.51}$$

The posterior for the individual factors $\mathbf{t}_m^i$ can be written

$$p(\mathbf{t}_m^i | \mathbf{s}^i, \{\mathbf{x}_m^i, \mathbf{A}_m, \mathbf{B}_m, \mathbf{\Psi}_m\}_{m=1}^M) \propto \\ \exp\left(\mathbf{t}_m^{i\top}\mathbf{B}_m^\top\mathbf{\Psi}_m^{-1}\left(\mathbf{x}_m^i - \mathbf{A}_m\mathbf{s}^i\right) - \frac{1}{2}\mathbf{t}_m^{i\top}\left(\mathbf{I} + \mathbf{B}_m^\top\mathbf{\Psi}_m^{-1}\mathbf{B}_m\right)\mathbf{t}_m^i\right) \tag{3.52}$$

For both posteriors its easy to identify the covariance which are given in equation (3.57) and (3.58) of the EM algorithm. To identify the mean of the posteriors is a bit more demanding since these are coupled equations. By inserting the equation for the mean of the posterior for $\mathbf{t}_m^i$ into the equation for the mean of $\mathbf{s}^i$ and doing some algebraic gymnastics we arrive at the update equations shown in equation (3.59) and (3.61). Notice that the EM algorithm of EFA reduces to the EM algorithm of FA if there are no individual factors.

The M-step involves maximizing the expected complete log likelihood which can be written

$$\begin{aligned}\langle\mathcal{L}(\boldsymbol{\theta}, \mathbf{S}, \{\mathbf{T}_m\}_{m=1}^M)\rangle = -\frac{N}{2}\sum_{m=1}^M \ln|\mathbf{\Psi}_m| - \frac{1}{2}\sum_{i=1}^N\sum_{m=1}^M\Big(\mathbf{x}_m^{i\top}\mathbf{\Psi}_m^{-1}\mathbf{x}_m^i \\ + \operatorname{tr}\left[\mathbf{A}_m^\top\mathbf{\Psi}_m^{-1}\mathbf{A}_m\langle\mathbf{s}^i\mathbf{s}^{i\top}\rangle\right] + \operatorname{tr}\left[\mathbf{B}_m^\top\mathbf{\Psi}_m^{-1}\mathbf{B}_m\langle\mathbf{t}_m^i\mathbf{t}_m^{i\top}\rangle\right] \\ + 2\,\mathbf{s}^{i\top}\mathbf{A}_m^\top\mathbf{\Psi}_m^{-1}\mathbf{B}_m\langle\mathbf{t}_m^i\rangle - 2\,\mathbf{x}_m^{i\top}\mathbf{\Psi}_m^{-1}\left(\mathbf{A}_m\langle\mathbf{s}^i\rangle + \mathbf{B}_m\langle\mathbf{t}_m^i\rangle\right)\Big) + \text{const}\end{aligned} \tag{3.53}$$

Now, taking partial derivatives with respect to the parameters we get

$$\frac{\partial\langle\mathcal{L}(\boldsymbol{\theta}, \mathbf{S}, \{\mathbf{T}_m\}_{m=1}^M)\rangle}{\partial\mathbf{A}_m} = \sum_{i=1}^N \mathbf{\Psi}_m^{-1}(\mathbf{x}_m^i - \mathbf{B}_m\langle\mathbf{t}_m^i\rangle)\langle\mathbf{s}^i\rangle^\top - \mathbf{\Psi}_m^{-1}\mathbf{A}_m\langle\mathbf{s}^i\mathbf{s}^{i\top}\rangle \tag{3.54}$$

$$\frac{\partial\langle\mathcal{L}(\boldsymbol{\theta}, \mathbf{S}, \{\mathbf{T}_m\}_{m=1}^M)\rangle}{\partial\mathbf{B}_m} = \sum_{i=1}^N \mathbf{\Psi}_m^{-1}(\mathbf{x}_m^i - \mathbf{A}_m\langle\mathbf{s}^i\rangle)\langle\mathbf{t}_m^i\rangle^\top - \mathbf{\Psi}_m^{-1}\mathbf{B}_m\langle\mathbf{t}_m^i\mathbf{t}_m^{i\top}\rangle \tag{3.55}$$

$$\begin{aligned}\frac{\partial\langle\mathcal{L}(\boldsymbol{\theta}, \mathbf{S}, \{\mathbf{T}_m\}_{m=1}^M)\rangle}{\partial\mathbf{\Psi}_m^{-1}} &= \frac{N}{2}\mathbf{\Psi}_m - \sum_i^N\Big(\frac{1}{2}\mathbf{x}_m^i\mathbf{x}_m^{i\top} - (\mathbf{A}_m\langle\mathbf{s}^i\rangle + \mathbf{B}_m\langle\mathbf{t}_m^i\rangle)\mathbf{x}_m^{i\top} \\ &+ \frac{1}{2}\mathbf{A}_m\langle\mathbf{s}^i\mathbf{s}^{i\top}\rangle\mathbf{A}_m^\top + \frac{1}{2}\mathbf{B}_m\langle\mathbf{t}_m^i\mathbf{t}_m^{i\top}\rangle\mathbf{B}_m^\top + \mathbf{B}_m\langle\mathbf{t}_m^i\rangle\langle\mathbf{s}^i\rangle^\top\mathbf{A}_m^\top\Big)\end{aligned} \tag{3.56}$$

Setting the above equations to zero, rearranging and substituting equation (3.54) and (3.55) in (3.56) the update for $\mathbf{\Psi}_m$ simplifies which is shown in equation (3.65) of the M-step. To ensure that $\mathbf{\Psi}$ is diagonal we set off diagonal terms to zero by the diag$[\cdot]$ operator as seen in equation (3.65) of the M-step. The updates for $\mathbf{A}_m$ and $\mathbf{B}_m$ are coupled equations thus we have to substitute the update equation for $\mathbf{B}_m$ in the update equation for $\mathbf{A}_m$. This lead to the result given in equation (3.63) of the M-step. This concludes the derivation of the EM algorithm for EFA.

EM Algorithm

E-step:

$$\mathbf{\Sigma_S} = \left(\mathbf{I} + \sum_{m=1}^{M} \mathbf{A}_m^\top \mathbf{\Psi}_m^{-1} \mathbf{A}_m\right)^{-1} \tag{3.57}$$

$$\mathbf{\Sigma}_{\mathbf{T}_m} = \left(\mathbf{I} + \mathbf{B}_m^\top \mathbf{\Psi}_m^{-1} \mathbf{B}_m\right)^{-1} \tag{3.58}$$

$$\begin{aligned}\langle \mathbf{S} \rangle = & \left(\mathbf{\Sigma_S}^{-1} - \sum_{m=1}^{M} \mathbf{A}_m^\top \mathbf{\Psi}_m^{-1} \mathbf{B}_m \mathbf{\Sigma}_{\mathbf{T}_m} \mathbf{B}_m^\top \mathbf{\Psi}_m^{-1} \mathbf{A}_m\right)^{-1} \times \\ & \left(\sum_{m=1}^{M} \left(\mathbf{A}_m^\top \mathbf{\Psi}_m^{-1} - \mathbf{A}_m^\top \mathbf{\Psi}_m^{-1} \mathbf{B}_m \mathbf{\Sigma}_{\mathbf{T}_m} \mathbf{B}_m^\top \mathbf{\Psi}_m^{-1}\right) \mathbf{X}_m\right)\end{aligned} \tag{3.59}$$

$$\langle \mathbf{S}\mathbf{S}^\top \rangle = N\mathbf{\Sigma_S} + \langle \mathbf{S} \rangle \langle \mathbf{S} \rangle^\top \tag{3.60}$$

$$\langle \mathbf{T}_m \rangle = \mathbf{\Sigma}_{\mathbf{T}_m} \left(\mathbf{B}_m^\top \mathbf{\Psi}_m^{-1} \left(\mathbf{X}_m - \mathbf{A}_m \langle \mathbf{S} \rangle\right)\right) \tag{3.61}$$

$$\langle \mathbf{T}_m \rangle \dagger \equiv \left(\langle \mathbf{T}_m \rangle \langle \mathbf{T}_m \rangle^\top\right)^{-1} \langle \mathbf{T}_m \rangle \tag{3.62}$$

M-step:

$$\begin{aligned}\mathbf{A}_m = & \mathbf{X}_m \left(\langle \mathbf{S} \rangle^\top - \langle \mathbf{T}_m \rangle^\top \langle \mathbf{T}_m \rangle \dagger \langle \mathbf{S} \rangle^\top\right) \times \\ & \left(\langle \mathbf{S}\mathbf{S}^\top \rangle - \langle \mathbf{S} \rangle \langle \mathbf{T}_m \rangle^\top \langle \mathbf{T}_m \rangle \dagger \langle \mathbf{S} \rangle^\top\right)^{-1}\end{aligned} \tag{3.63}$$

$$\mathbf{B}_m = \left(\mathbf{X}_m - \mathbf{A}_m \langle \mathbf{S} \rangle\right) \langle \mathbf{T}_m \rangle \dagger^\top \tag{3.64}$$

$$\mathbf{\Psi}_m = \frac{1}{N} \mathrm{diag}\left[\mathbf{X}_m \mathbf{X}_m^\top - \left(\mathbf{A}_m \langle \mathbf{S} \rangle + \mathbf{B}_m \langle \mathbf{T}_m \rangle\right) \mathbf{X}_m^\top\right] \tag{3.65}$$

3.4 Variational Bayesian Extended Factor Analysis

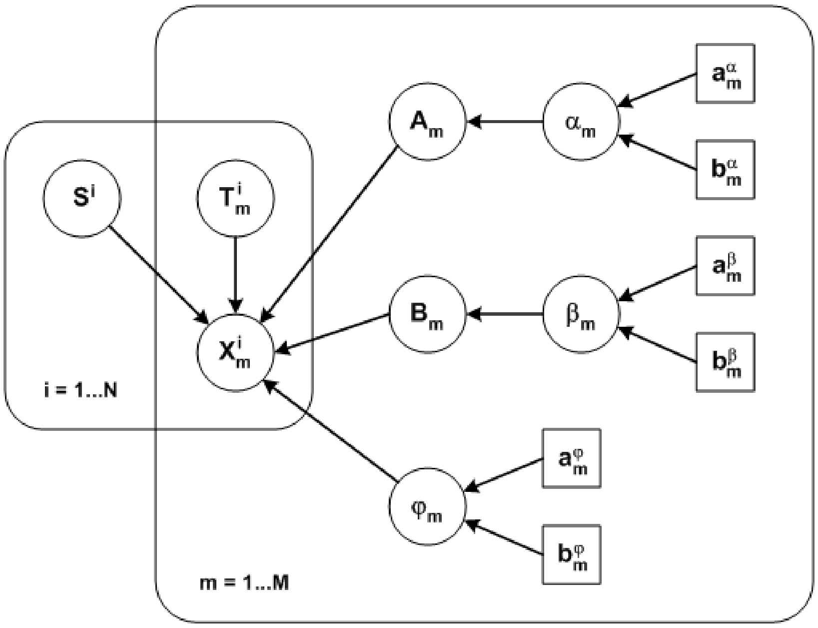


Figure 3.5: Graphical model for variational Bayesian extended factor analysis

The model proposed for variational Bayesian extended factor analysis (VBEFA) can be seen on figure 3.5. Compared to MLEFA we have now placed distributions on all factor loading matrices and noise covariances. In order to do automatic latent dimensionality determination a hierarchical prior is used on each factor loading matrix. In this way we can automatically infer the latent dimensions in the model without doing a complete search for every combination. The priors in the model are similar to those defined in VBFA, but each factor loading matrix $\mathbf{A}_m, \mathbf{B}_m$ and noise covariance $\boldsymbol{\varphi}_m$ have separate priors. The data sets $\mathbf{X}_m$ are allowed to have different dimensionality P_m and different latent dimensionality K_T^m for the individual factors. The derivations are very similar to those described for VBFA, thus some of the details are omitted. The priors are defined as

$$p(\mathbf{A}_m \,|\, \boldsymbol{\alpha}_m) = \prod_{k=1}^{K} \mathcal{N}\big(\mathbf{a}_{mk} \,|\, \mathbf{0}, \tfrac{1}{\alpha_{mk}}\mathbf{I}\big) \tag{3.66}$$

$$p(\boldsymbol{\alpha}_m \,|\, \mathbf{a}_m^{\alpha}, \mathbf{b}_m^{\alpha}) = \prod_{k=1}^{K} \mathcal{G}\big(\alpha_{mk} \,|\, a_{mk}^{\alpha}, b_{mk}^{\alpha}\big) \tag{3.67}$$

$$p(\mathbf{B}_m \,|\, \boldsymbol{\beta}_m) = \prod_{k=1}^{K_T^m} \mathcal{N}\big(\mathbf{b}_{mk} \,|\, \mathbf{0}, \tfrac{1}{\beta_{mk}}\mathbf{I}\big) \tag{3.68}$$

$$p(\boldsymbol{\beta}_m \,|\, \mathbf{a}_m^{\beta}, \mathbf{b}_m^{\beta}) = \prod_{k=1}^{K_T^m} \mathcal{G}\big(\beta_{mk} \,|\, a_{mk}^{\beta}, b_{mk}^{\beta}\big) \tag{3.69}$$

$$p(\boldsymbol{\varphi}_m \,|\, \mathbf{a}_m^{\varphi}, \mathbf{b}_m^{\varphi}) = \prod_{j=1}^{P_m} \mathcal{G}\big(\varphi_{mj} \,|\, a_{mj}^{\varphi}, b_{mj}^{\varphi}\big) \tag{3.70}$$

where $\mathbf{a}_{mk}$ denotes a column vector that corresponds to the k^{th} column in $\mathbf{A}_m$. Just like VBFA there is now an ARD prior on each factor loading matrix. The joint variational posterior for all hidden variables and parameters we approximate by one which is fully factorized. The complete log likelihood for the model is given by

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}, \mathbf{S}, \{\mathbf{T}_m\}_{m=1}^{M}) = &-\frac{1}{2}\sum_{i=1}^{N} \mathbf{s}^{i\top}\mathbf{s}^{i} + \sum_{m=1}^{M}\Bigg[\frac{N}{2}\ln|\boldsymbol{\varphi}_m| - \frac{1}{2}\sum_{i=1}^{N}\mathbf{t}_m^{i\top}\mathbf{t}_m^{i} \\
&- \frac{1}{2}\sum_{i=1}^{N}\left(\mathbf{x}_m^i - \mathbf{A}_m\mathbf{s}^i - \mathbf{B}_m\mathbf{t}_m^i\right)^{\top}\boldsymbol{\varphi}_m\left(\mathbf{x}_m^i - \mathbf{A}_m\mathbf{s}^i - \mathbf{B}_m\mathbf{t}_m^i\right) \\
&+ \frac{P_m}{2}\sum_{k=1}^{K}\ln\alpha_{mk} - \frac{1}{2}\sum_{k=1}^{K}\alpha_{mk}\mathbf{a}_{mk}^{\top}\mathbf{a}_{mk} + \sum_{k=1}^{K}(a_{mk}^{\alpha}-1)\ln\alpha_{mk} - \sum_{k=1}^{K} b_{mk}^{\alpha}\alpha_{mk} \\
&+ \frac{P_m}{2}\sum_{k=1}^{K_T^m}\ln\beta_{mk} - \frac{1}{2}\sum_{k=1}^{K_T^m}\beta_{mk}\mathbf{b}_{mk}^{\top}\mathbf{b}_{mk} + \sum_{k=1}^{K_T^m}(a_{mk}^{\beta}-1)\ln\beta_{mk} - \sum_{k=1}^{K_T^m} b_{mk}^{\beta}\beta_{mk} \\
&+ \sum_{j=1}^{P_m}(a_{mj}^{\varphi}-1)\ln\varphi_{mj} - \sum_{j=1}^{P_m} b_{mj}^{\varphi}\varphi_{mj}\Bigg] + \text{const}
\end{aligned} \tag{3.71}$$

The VBE-step

The posterior for $q(\mathbf{T}_m)$ is of the form

$$q(\mathbf{T}_m) = \prod_{i=1}^{N} q(\mathbf{t}_m^i) = \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{t}_m^i \,|\, \mathbf{m}_{\mathbf{t}_m}^{(i)}, \boldsymbol{\Sigma}_{\mathbf{t}_m}\right) \tag{3.72}$$

where

$$\Sigma_{\mathbf{t}_m} = \left(\mathbf{I} + \langle \mathbf{B}_m^\top \boldsymbol{\varphi}_m \mathbf{B}_m \rangle\right)^{-1} \tag{3.73}$$

$$\mathbf{m}_{\mathbf{t}_m}^{(i)} = \Sigma_{\mathbf{t}_m} \langle \mathbf{B}_m \rangle^\top \langle \boldsymbol{\varphi}_m \rangle \left(\mathbf{x}_m^i - \langle \mathbf{A}_m \rangle \langle \mathbf{s}^i \rangle\right) \tag{3.74}$$

The posterior for $q(\mathbf{S})$ is of the form

$$q(\mathbf{S}) = \prod_{i=1}^{N} q(\mathbf{s}^i) = \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{s}^i \,|\, \mathbf{m}_{\mathbf{s}}^{(i)}, \Sigma_{\mathbf{s}}\right) \tag{3.75}$$

where

$$\Sigma_{\mathbf{s}} = \left(\mathbf{I} + \sum_{m=1}^{M} \langle \mathbf{A}_m^\top \boldsymbol{\varphi}_m \mathbf{A}_m \rangle\right)^{-1} \tag{3.76}$$

$$\begin{aligned} \mathbf{m}_{\mathbf{s}}^{(i)} = & \left(\Sigma_{\mathbf{S}}^{-1} - \sum_{m=1}^{M} \langle \mathbf{A}_m \rangle^\top \langle \boldsymbol{\varphi}_m \rangle \langle \mathbf{B}_m \rangle \Sigma_{\mathbf{T}_m} \langle \mathbf{B}_m \rangle^\top \langle \boldsymbol{\varphi}_m \rangle \langle \mathbf{A}_m \rangle\right)^{-1} \times \\ & \left(\sum_{m=1}^{M} \left(\langle \mathbf{A}_m \rangle^\top \langle \boldsymbol{\varphi}_m \rangle - \langle \mathbf{A}_m \rangle^\top \langle \boldsymbol{\varphi}_m \rangle \langle \mathbf{B}_m \rangle \Sigma_{\mathbf{T}_m} \langle \mathbf{B}_m \rangle^\top \langle \boldsymbol{\varphi}_m \rangle\right) \mathbf{x}_m^i\right) \end{aligned} \tag{3.77}$$

This rather complicated form is the result of insertion of the update equations for $\mathbf{T}_m$. In this way the update equations for the VBE-step are uncoupled if we start by computing the update for $q(\mathbf{S})$ and then compute the update for $q(\mathbf{T}_m)$. Notice that the VBE-step is the same as the E-step in MLEFA, but with expected values of the parameters.

The VBM-step

The $q(\mathbf{A}_m)$ distribution

The posterior for $q(\mathbf{A}_m)$ is of the form

$$q(\mathbf{A}_m) = \prod_{j=1}^{P_m} q(\mathbf{a}_{mj}) = \prod_{j=1}^{P_m} \mathcal{N}\left(\mathbf{a}_{mj} \,|\, \mathbf{m}_{\mathbf{a}_m}^{(j)}, \Sigma_{\mathbf{a}_m}^{(j)}\right) \tag{3.78}$$

where

$$\Sigma_{\mathbf{a}_m}^{(j)} = \left(\langle \varphi_{mj} \rangle \left(\sum_{i=1}^{N} \langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle \right) + \langle \overline{\overline{\boldsymbol{\alpha}}}_m \rangle \right)^{-1} \tag{3.79}$$

$$\begin{aligned} \mathbf{m}_{\mathbf{a}_m}^{(j)} = & \left(\mathbf{I} - \langle \varphi_{mj} \rangle^2 \Sigma_{\mathbf{a}_m}^{(j)} \left(\sum_{i=1}^{N} \langle \mathbf{s}^i \rangle \langle \mathbf{t}_m^i \rangle^\top \right) \Sigma_{\mathbf{b}_m}^{(j)} \left(\sum_{i=1}^{N} \langle \mathbf{t}_m^i \rangle \langle \mathbf{s}^i \rangle^\top \right) \right)^{-1} \times \\ & \left[\langle \varphi_{mj} \rangle \Sigma_{\mathbf{a}_m}^{(j)} \left(\sum_{i=1}^{N} \langle \mathbf{s}^i \rangle x_{mji} \right) \right. \\ & \left. - \langle \varphi_{mj} \rangle \left(\sum_{i=1}^{N} \langle \mathbf{s}^i \rangle \langle \mathbf{t}_m^i \rangle^\top \right) \Sigma_{\mathbf{b}_m}^{(j)} \left(\sum_{i=1}^{N} \langle \mathbf{t}_m^i \rangle x_{mji} \right) \right] \end{aligned} \tag{3.80}$$

The $q(\mathbf{B}_m)$ distribution
The posterior for $q(\mathbf{B}_m)$ is of the form

$$q(\mathbf{B}_m) = \prod_{j=1}^{P_m} q(\mathbf{b}_{mj}) = \prod_{j=1}^{P_m} \mathcal{N} \left(\mathbf{b}_{mj} \,|\, \mathbf{m}_{\mathbf{b}_m}^{(j)}, \Sigma_{\mathbf{b}_m}^{(j)} \right) \tag{3.81}$$

where

$$\Sigma_{\mathbf{b}_m}^{(j)} = \left(\langle \varphi_{mj} \rangle \left(\sum_{i=1}^{N} \langle \mathbf{t}_m^i \mathbf{t}_m^{i\top} \rangle \right) + \langle \overline{\overline{\boldsymbol{\beta}}}_m \rangle \right)^{-1} \tag{3.82}$$

$$\begin{aligned} \mathbf{m}_{\mathbf{b}_m}^{(j)} = & \left(\mathbf{I} - \langle \varphi_{mj} \rangle^2 \Sigma_{\mathbf{b}_m}^{(j)} \left(\sum_{i=1}^{N} \langle \mathbf{t}_m^i \rangle \langle \mathbf{s}^i \rangle^\top \right) \Sigma_{\mathbf{a}_m}^{(j)} \left(\sum_{i=1}^{N} \langle \mathbf{s}^i \rangle \langle \mathbf{t}_m^i \rangle^\top \right) \right)^{-1} \times \\ & \left[\langle \varphi_{mj} \rangle \Sigma_{\mathbf{b}_m}^{(j)} \left(\sum_{i=1}^{N} \langle \mathbf{t}_m^i \rangle x_{mji} \right) \right. \\ & \left. - \langle \varphi_{mj} \rangle \left(\sum_{i=1}^{N} \langle \mathbf{t}_m^i \rangle \langle \mathbf{s}^i \rangle^\top \right) \Sigma_{\mathbf{a}_m}^{(j)} \left(\sum_{i=1}^{N} \langle \mathbf{s}^i \rangle x_{mji} \right) \right] \end{aligned} \tag{3.83}$$

Again these complicated update equations stems from the fact that we have substituted the equation for the mean for $q(\mathbf{A}_m)$ in $q(\mathbf{B}_m)$ and vice versa. If we did not do this, the update equations would have been coupled. In this way the updates only depend on the statistics gathered in the VBE-step.

The $q(\boldsymbol{\alpha}_m)$ distribution
The posterior for $q(\boldsymbol{\alpha}_m)$ is of the form

$$q(\boldsymbol{\alpha}_m) = \prod_{k=1}^{K} q(\alpha_{mk}) = \prod_{k=1}^{K} \mathcal{G}\left(\alpha_{mk} \,|\, \widehat{a}^{\alpha}_{mk}, \widehat{b}^{\alpha}_{mk}\right) \tag{3.84}$$

where

$$\boxed{\begin{aligned} \widehat{a}^{\alpha}_{mk} &= a^{\alpha}_{mk} + \frac{P_m}{2} && (3.85)\\ \widehat{b}^{\alpha}_{mk} &= b^{\alpha}_{mk} + \frac{\langle \mathbf{a}^{\top}_{mk}\mathbf{a}_{mk}\rangle}{2} && (3.86)\end{aligned}}$$

The $q(\boldsymbol{\beta}_m)$ distribution
The posterior for $q(\boldsymbol{\beta}_m)$ is of the form

$$q(\boldsymbol{\beta}_m) = \prod_{k=1}^{K_T^m} q(\alpha_{mk}) = \prod_{k=1}^{K_T^m} \mathcal{G}\left(\beta_{mk} \,|\, \widehat{a}^{\beta}_{mk}, \widehat{b}^{\beta}_{mk}\right) \tag{3.87}$$

where

$$\boxed{\begin{aligned} \widehat{a}^{\beta}_{mk} &= a^{\beta}_{mk} + \frac{P_m}{2} && (3.88)\\ \widehat{b}^{\beta}_{mk} &= b^{\beta}_{mk} + \frac{\langle \mathbf{b}^{\top}_{mk}\mathbf{b}_{mk}\rangle}{2} && (3.89)\end{aligned}}$$

The $q(\boldsymbol{\varphi}_m)$ distribution
The posterior for $q(\boldsymbol{\varphi}_m)$ is of the form

$$q(\boldsymbol{\varphi}_m) = \prod_{j=1}^{P_m} q(\varphi_{mj}) = \prod_{j=1}^{P_m} \mathcal{G}\left(\varphi_{mj} \,|\, \widehat{a}^{\varphi}_{mj}, \widehat{b}^{\varphi}_{mj}\right) \tag{3.90}$$

where

$$\widehat{a}^{\varphi}_{mj} = a^{\varphi}_{mj} + \frac{N}{2} \tag{3.91}$$

$$\widehat{b}^{\varphi}_{mj} = b^{\varphi}_{mj} + \frac{1}{2}\sum_{i=1}^{N} \left\langle (x_{mji} - \mathbf{a}^{\top}_{mj}\mathbf{s}^{i} - \mathbf{b}^{\top}_{mj}\mathbf{t}^{i}_{m})^2 \right\rangle \tag{3.92}$$

This concludes the VBM-step. Hyperparameter optimization and computation of the lower bound is done as described in the VBFA section.

Optimization Heuristics

A problem discovered when testing the algorithm is that the learnt latent dimensions are not ordered correctly. A typical scenario is that too many common factors are found. These are then only shared by some of the data sets and not common to all. To fix this I introduced a heuristic in the optimization by observing the ARD parameters. When an ARD parameter is going to infinity for a certain factor loading matrix, the corresponding column is removed. To my experience this always increases the lower bound when the removed column represents an individual factor. On the other hand, if it is a common factor we remove this might not always be the case. If only a part of the data sets share a certain common factor, the data sets not using it will have ARD parameters that tend to infinity. Thus, a removal will give a drop in the lower bound. We would like the algorithm to only find common factors that are common to all data sets. Therefore a common factor is removed when it is not shared by all data sets. The data sets that use the common factor, which is being removed, then gets the common factor rearranged as an individual factor. To do this rearrangement we need to change the statistics obtained in the E-step. In many cases this gives a short drop in the lower bound, but it is usually recovered within a few iterations. Experimentally, this heuristic works without problems and the algorithm usually recover the true latent dimensions.

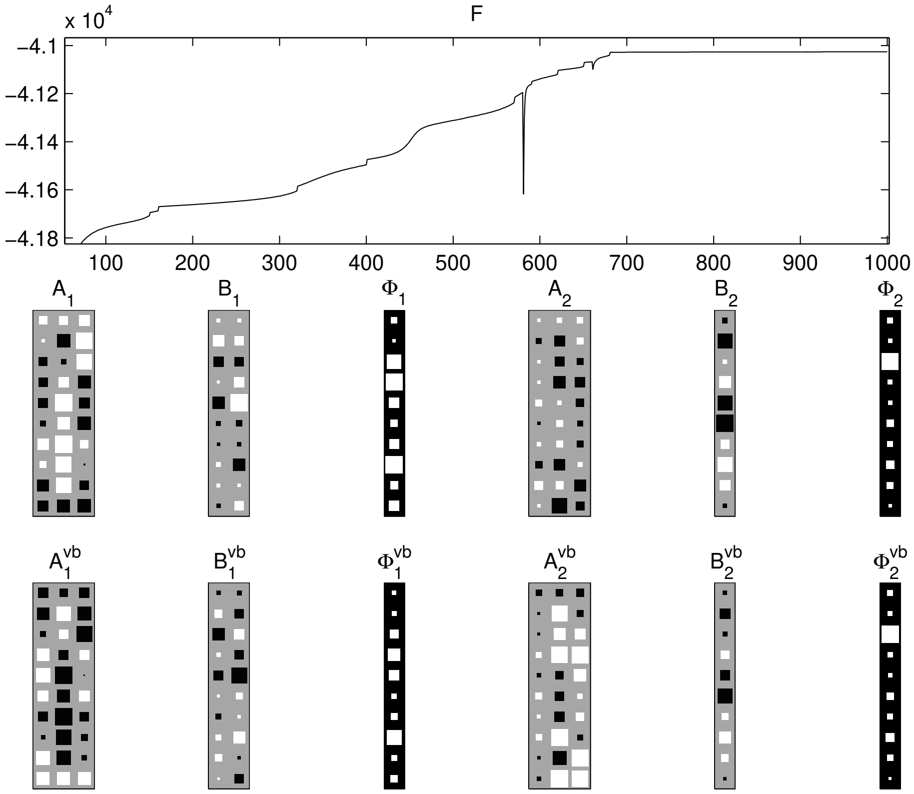


Figure 3.6: Top row shows the bound during 10^3 iterations of VBEFA. Second row shows the factor loading matrices and the noise precisions of the reference model. Bottom row shows the estimated matrices.

VBEFA in Action

To demonstrate the model I created a *reference* extended factor analysis model with precisions drawn from Gamma distributions and factor loading matrices drawn from Gaussian distributions controlled by the precisions. From this model I generated two $P_1 = P_2 = 10$ dimensional training sets of size $N = 10^3$. The reference latent dimensions were $K = 3$, $K_T^1 = 2$ and $K_T^2 = 1$. No hyperparameter optimization was invoked. All priors were set to be non-informative. The algorithm was run for 10^3 iterations with a maximum dimensionality of $k_{max} = 5$ for all factor loading matrices. The optimization heuristic was invoked to guide the learning process. Figure 3.6 summarize the results. The learnt latent dimensions corresponds to the reference latent dimensions. Notice the drop in $\mathcal{F}$ and how it recovers within a few iterations.

3.5 Maximum Likelihood Parallel Factor Analysis

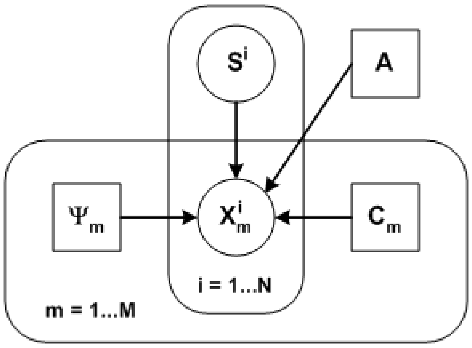


Figure 3.7: Graphical model for maximum likelihood parallel factor analysis

The generative model for maximum likelihood parallel factor analysis is given by

$$\mathbf{X}_m = \mathbf{A}\mathbf{C}_m\mathbf{S} + \mathbf{E}_m \tag{3.93}$$

where the model assumptions in parallel factor analysis implies that $\{\mathbf{x}_m^i\}_{m=1}^M$ is Gaussian distributed as

$$p(\{\mathbf{x}_m^i\}_{m=1}^M | \, \mathbf{s}^i, \mathbf{A}, \{\mathbf{C}_m, \boldsymbol{\Psi}_m\}_{m=1}^M) \propto \exp\left(-\frac{1}{2}\sum_{m=1}^M \Big(\mathbf{x}_m^i - \mathbf{A}\mathbf{C}_m\mathbf{s}^i\Big)^\top \boldsymbol{\Psi}_m^{-1}\Big(\mathbf{x}_m^i - \mathbf{A}\mathbf{C}_m\mathbf{s}^i\Big)\right) \tag{3.94}$$

To identify the E-step we write down the posterior for the factors $\mathbf{s}^i$ by multiplying the likelihood by the prior and neglect constant terms

$$p(\mathbf{s}^i | \, \mathbf{A}, \{\mathbf{x}_m^i, \mathbf{C}_m, \boldsymbol{\Psi}_m\}_{m=1}^M) \propto \exp\left(\mathbf{s}^{i\top}\left(\sum_{m=1}^M \mathbf{C}_m\mathbf{A}^\top\boldsymbol{\Psi}_m^{-1}\mathbf{x}_m^i\right) - \frac{1}{2}\mathbf{s}^{i\top}\left(\mathbf{I} + \sum_{m=1}^M \mathbf{C}_m\mathbf{A}^\top\boldsymbol{\Psi}_m^{-1}\mathbf{A}\mathbf{C}_m\right)\mathbf{s}^i\right) \tag{3.95}$$

Though the E-step is easy to identify and is given in equation (3.100) and (3.101).

The M-step involves maximizing the expected complete log likelihood which can be written

$$\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle = -\frac{N}{2} \sum_{m=1}^{M} \ln |\boldsymbol{\Psi}_m| - \frac{1}{2} \sum_{i=1}^{N} \sum_{m=1}^{M} \Big(\mathbf{x}_m^{i\top} \boldsymbol{\Psi}_m^{-1} \mathbf{x}_m^i + \operatorname{tr} \Big[\mathbf{C}_m \mathbf{A}^\top \boldsymbol{\Psi}_m^{-1} \mathbf{A} \mathbf{C}_m \langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle \Big] - 2\, \mathbf{x}_m^{i\top} \boldsymbol{\Psi}_m^{-1} \mathbf{A} \mathbf{C}_m \langle \mathbf{s}^i \rangle \Big) + \text{const} \tag{3.96}$$

Now, taking partial derivatives with respect to the parameters we get

$$\frac{\partial \langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle}{\partial \mathbf{A}} = \sum_{i=1}^{N} \sum_{m=1}^{M} \boldsymbol{\Psi}_m^{-1} \mathbf{x}_m^i \langle \mathbf{s}^i \rangle^\top \mathbf{C}_m - \boldsymbol{\Psi}_m^{-1} \mathbf{A} \mathbf{C}_m \langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle \mathbf{C}_m \tag{3.97}$$

$$\frac{\partial \langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle}{\partial \mathbf{C}_m} = \sum_{i=1}^{N} \Big(\mathbf{A}^\top \boldsymbol{\Psi}_m^{-1} \mathbf{x}_m^i \langle \mathbf{s}^i \rangle^\top \Big) \circ \mathbf{I} - \Big(\Big(\langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle \circ (\mathbf{A}^\top \boldsymbol{\Psi}_m^{-1} \mathbf{A}) \Big) \mathbf{C}_m \Big) \circ \mathbf{I} \tag{3.98}$$

$$\frac{\partial \langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle}{\partial \boldsymbol{\Psi}_m^{-1}} = \frac{N}{2} \boldsymbol{\Psi}_m - \sum_{i}^{N} \Big(\frac{1}{2} \mathbf{x}_m^i \mathbf{x}_m^{i\top} - \mathbf{A} \mathbf{C}_m \langle \mathbf{s}^i \rangle \mathbf{x}_m^{i\top} + \frac{1}{2} \mathbf{A} \mathbf{C}_m \langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle \mathbf{C}_m \mathbf{A}^\top \Big) \tag{3.99}$$

where the derivative in equation (3.98) is a result of $\langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle$ and $\mathbf{A}^\top \boldsymbol{\Psi}_m^{-1} \mathbf{A}$ being symmetric and $\mathbf{C}_m$ diagonal. To ensure that $\boldsymbol{\Psi}$ is diagonal we set off diagonal terms to zero by the diag[$\cdot$] operator as seen in equation (3.105) of the M-step. Isolating $\mathbf{A}$ in equation (3.97) is not that straight forward because of the sum over m. Thus we skip the details. The update equations for the M-step are shown in equations (3.103) - (3.105). Notice that the equations for the M-step are coupled. One could iterate these, but to my experience this is not necessary.

EM Algorithm

E-step:

$$\mathbf{\Sigma_S} = \left(\mathbf{I} + \sum_{m=1}^{M} \mathbf{C}_m \mathbf{A}^\top \mathbf{\Psi}_m^{-1} \mathbf{A} \mathbf{C}_m\right)^{-1} \tag{3.100}$$

$$\langle \mathbf{S} \rangle = \mathbf{\Sigma_S} \left(\sum_{m=1}^{M} \mathbf{C}_m \mathbf{A}^\top \mathbf{\Psi}_m^{-1} \mathbf{X}_m\right) \tag{3.101}$$

$$\langle \mathbf{S}\mathbf{S}^\top \rangle = N\mathbf{\Sigma_S} + \langle \mathbf{S} \rangle \langle \mathbf{S} \rangle^\top \tag{3.102}$$

M-step:

$$\mathbf{a}_k = \left(\sum_{m=1}^{M} \mathbf{\Psi}_m^{-1} \mathbf{X}_m \langle \mathbf{S} \rangle^\top \mathbf{C}_m\right)_{(k)} \left(\sum_{m=1}^{M} (\mathbf{\Psi}_m^{-1})_{kk} \mathbf{C}_m \langle \mathbf{S}\mathbf{S}^\top \rangle \mathbf{C}_m\right)^{-1} \tag{3.103}$$

$$\mathbf{c}_m = \left(\langle \mathbf{S}\mathbf{S}^\top \rangle \circ (\mathbf{A}^\top \mathbf{\Psi}_m^{-1} \mathbf{A})\right)^{-1} \mathrm{vec}\left[\mathrm{diag}\left[\mathbf{A}^\top \mathbf{\Psi}_m^{-1} \mathbf{X}_m \langle \mathbf{S} \rangle^\top\right]\right] \tag{3.104}$$

$$\mathbf{\Psi}_m = \frac{1}{N} \mathrm{diag}\left[\mathbf{X}_m \mathbf{X}_m^\top + \mathbf{A}\mathbf{C}_m \langle \mathbf{S}\mathbf{S}^\top \rangle \mathbf{C}_m \mathbf{A}^\top - 2\mathbf{A}\mathbf{C}_m \langle \mathbf{S} \rangle \mathbf{X}_m^\top\right] \tag{3.105}$$

$\mathbf{c}_m$ is column vector holding the diagonal of $\mathbf{C}_m$. $\mathbf{a}_k$ is a row vector corresponding to the k^{th} row of $\mathbf{A}$. $(\mathbf{Z})_{(k)}$ denotes a row vector corresponding to the k^{th} row of $\mathbf{Z}$. The elements of $\mathbf{\Psi}_m^{-1}$ are denoted by $(\mathbf{\Psi}_m^{-1})_{ij}$.

3.6 Variational Bayesian Parallel Factor Analysis

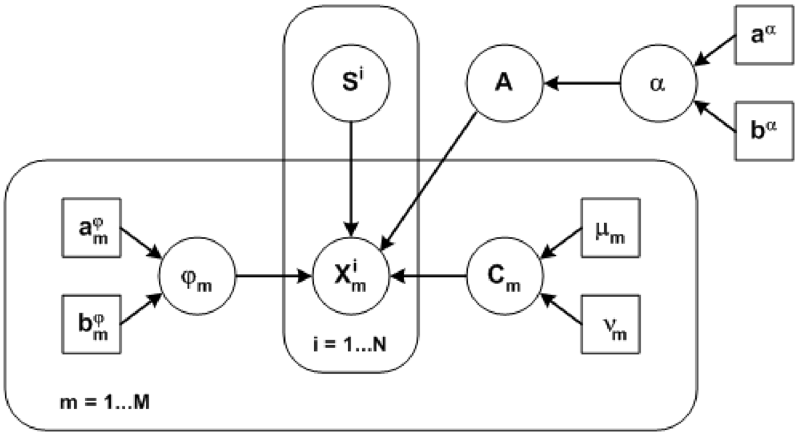


Figure 3.8: Graphical model for variational Bayesian parallel factor analysis

The model proposed for variational Bayesian parallel factor analysis (VB-PARAFAC) can be seen on figure 3.8. Compared to MLPARAFAC we have now placed distributions on the factor loading matrix, the noise covariance and the proportion matrix $\mathbf{C}_m$. We have placed an ARD prior on the factor loading matrix in order to do automatic latent dimensionality determination. The priors are defined as

$$p(\mathbf{A}\,|\,\boldsymbol{\alpha}) = \prod_{k=1}^{K} \mathcal{N}\big(\mathbf{a}_k\,|\,\mathbf{0}, \tfrac{1}{\alpha_k}\mathbf{I}\big) \tag{3.106}$$

$$p(\boldsymbol{\alpha}\,|\,\mathbf{a}^{\alpha}, \mathbf{b}^{\alpha}) = \prod_{k=1}^{K} \mathcal{G}\big(\alpha_k\,|\,a_k^{\alpha}, b_k^{\alpha}\big) \tag{3.107}$$

$$p(\mathbf{c}_m\,|\,\boldsymbol{\mu}_m, \nu_m) = \mathcal{N}\big(\mathbf{c}_m\,|\,\boldsymbol{\mu}_m, \tfrac{1}{\nu_m}\mathbf{I}\big) \tag{3.108}$$

$$p(\boldsymbol{\varphi}_m\,|\,\mathbf{a}_m^{\varphi}, \mathbf{b}_m^{\varphi}) = \prod_{j=1}^{p} \mathcal{G}\big(\varphi_{mj}\,|\,a_{mj}^{\varphi}, b_{mj}^{\varphi}\big) \tag{3.109}$$

where $\mathbf{c}_m$ is a column vector holding the diagonal of $\mathbf{C}_m$. The priors on the noise precisions and the factor loading matrix are equivalent to those used for VBEFA. A simple isotropic Gaussian prior is used for the proportion matrix $\mathbf{C}_m$. We could have given this prior a full covariance, but we keep it simple. If appropriate we could just as well enforce a Gamma prior on the proportions and thereby obtain a non-negative constrain on $\mathbf{c}_m$. Again we approximate the joint posterior with a fully factorized one. The derivations of the learning rules are given below.

The VBE-step

Writing up the expected complete log-likelihood with respect to the parameter posteriors we get a posterior for the factors on the form

$$q(\mathbf{S}) = \prod_{i=1}^{N} q(\mathbf{s}^i) = \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{s}^i \,|\, \mathbf{m}_{\mathbf{s}}^{(i)}, \boldsymbol{\Sigma}_{\mathbf{s}}\right) \tag{3.110}$$

where

$$\boxed{\begin{aligned} \boldsymbol{\Sigma}_{\mathbf{s}} &= \left(\mathbf{I} + \sum_{m=1}^{M} \left\langle \mathbf{C}_m \mathbf{A}^\top \boldsymbol{\varphi}_m \mathbf{A} \mathbf{C}_m \right\rangle \right)^{-1} && (3.111) \\ \mathbf{m}_{\mathbf{s}}^{(i)} &= \boldsymbol{\Sigma}_{\mathbf{s}} \sum_{m=1}^{M} \langle \mathbf{C}_m \rangle \langle \mathbf{A} \rangle^\top \langle \boldsymbol{\varphi}_m \rangle \mathbf{x}_m^i && (3.112) \end{aligned}}$$

The expectation in equation (3.111) can be computed by

$$\left\langle \mathbf{C}_m \mathbf{A}^\top \boldsymbol{\varphi}_m \mathbf{A} \mathbf{C}_m \right\rangle = \langle \mathbf{c}_m \mathbf{c}_m^\top \rangle \circ \left(\sum_{j=1}^{p} \langle \varphi_{mj} \rangle \langle \mathbf{a}_j \mathbf{a}_j^\top \rangle \right) \tag{3.113}$$

where $\mathbf{c}_m$ is the diagonal of $\mathbf{C}_m$ and $\mathbf{a}_j$ is column vector corresponding to the j^{th} row of $\mathbf{A}$.

The VBM-step

To derive the VBM-step we need to write the expected complete log likelihood $\langle \mathcal{L}(\boldsymbol{\theta}, \mathbf{S}) \rangle_{q(\boldsymbol{\theta}_{j \neq i}) q(\mathbf{S})}$ with respect to all other variational posteriors. The derivations are skipped but the update equations follow below.

The $q(\mathbf{A})$ distribution

The posterior for $\mathbf{A}$ is of the form

$$q(\mathbf{A}) = \prod_{j=1}^{p} q(\mathbf{a}_j) = \prod_{j=1}^{p} \mathcal{N}\left(\mathbf{a}_j \,|\, \mathbf{m}_{\mathbf{a}}^{(j)}, \boldsymbol{\Sigma}_{\mathbf{a}}^{(j)}\right) \tag{3.114}$$

where

$$\boldsymbol{\Sigma}_{\mathbf{a}}^{(j)} = \left(\sum_{m=1}^{M} \langle \varphi_{mj} \rangle \langle \mathbf{c}_m \mathbf{c}_m^\top \rangle \circ \left(\sum_{i=1}^{N} \langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle\right) + \langle \overline{\overline{\boldsymbol{\alpha}}} \rangle\right)^{-1} \tag{3.115}$$

$$\mathbf{m}_{\mathbf{a}}^{(j)} = \boldsymbol{\Sigma}_{\mathbf{a}}^{(j)} \left[\sum_{m=1}^{M} \langle \varphi_{mj} \rangle \langle \mathbf{C}_m \rangle \left(\sum_{i=1}^{N} x_{mji} \langle \mathbf{s}^i \rangle\right)\right] \tag{3.116}$$

The $q(\boldsymbol{\alpha})$ distribution
The posterior for $q(\boldsymbol{\alpha})$ is of the form

$$q(\boldsymbol{\alpha}) = \prod_{k=1}^{K} q(\alpha_k) = \prod_{k=1}^{K} \mathcal{G}\left(\alpha_k \,|\, \widehat{a}_k^\alpha, \widehat{b}_k^\alpha\right) \tag{3.117}$$

where

$$\widehat{a}_k^\alpha = a_k^\alpha + \frac{p}{2} \tag{3.118}$$

$$\widehat{b}_k^\alpha = b_k^\alpha + \frac{\langle \mathbf{a}_k^\top \mathbf{a}_k \rangle}{2} \tag{3.119}$$

The $q(\boldsymbol{\varphi}_m)$ distribution
The posterior for $q(\boldsymbol{\varphi}_m)$ is of the form

$$q(\boldsymbol{\varphi}_m) = \prod_{j=1}^{p} q(\varphi_{mj}) = \prod_{j=1}^{p} \mathcal{G}\left(\varphi_{mj} \,|\, \widehat{a}_{mj}^\varphi, \widehat{b}_{mj}^\varphi\right) \tag{3.120}$$

where

$$\widehat{a}_{mj}^\varphi = a_{mj}^\varphi + \frac{N}{2} \tag{3.121}$$

$$\widehat{b}_{mj}^\varphi = b_{mj}^\varphi + \frac{1}{2} \sum_{i=1}^{N} \left\langle (x_{mji} - \mathbf{a}_j^\top \mathbf{C}_m \mathbf{s}^i)^2 \right\rangle \tag{3.122}$$

The expectation in equation (3.122) can be computed by

$$\begin{aligned}\left\langle (x_{mji} - \mathbf{a}_j^\top \mathbf{C}_m \mathbf{s}^i)^2 \right\rangle = x_{mji}^2 - 2x_{mji} \langle \mathbf{a}_j \rangle^\top \langle \mathbf{C}_m \rangle \langle \mathbf{s}^i \rangle \\ + \operatorname{tr}\left[\langle \mathbf{a}_j \mathbf{a}_j^\top \rangle \left(\langle \mathbf{c}_m \mathbf{c}_m^\top \rangle \circ \langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle \right) \right]\end{aligned} \tag{3.123}$$

The $q(\mathbf{C}_m)$ distribution
The posterior for $q(\mathbf{C}_m)$ is of the form

$$q(\mathbf{c}_m) = \mathcal{N}\left(\mathbf{c}_m \,|\, \mathbf{m}_{\mathbf{c}_m}, \mathbf{\Sigma}_{\mathbf{c}_m}\right) \tag{3.124}$$

where $\mathbf{c}_m$ is the diagonal of $\mathbf{C}_m$. The required statistics for this posterior is given by

$$\mathbf{\Sigma}_{\mathbf{c}_m} = \left(\nu_m \mathbf{I} + \left(\sum_{i=1}^{N} \langle \mathbf{s}^i \mathbf{s}^{i\top} \rangle \right) \circ \left(\sum_{j=1}^{p} \langle \varphi_{mj} \rangle \langle \mathbf{a}_j \mathbf{a}_j^\top \rangle \right) \right)^{-1} \tag{3.125}$$

$$\mathbf{m}_{\mathbf{c}_m} = \mathbf{\Sigma}_{\mathbf{c}_m} \left[\nu_m \boldsymbol{\mu}_m + \sum_{j=1}^{p} \sum_{i=1}^{N} x_{mji} \langle \varphi_{mj} \rangle \langle \mathbf{a}_j \rangle \circ \langle \mathbf{s}^i \rangle \right] \tag{3.126}$$

Hyperparameter optimization and computation of the lower bound is quite similar to what is described in the VBFA section.

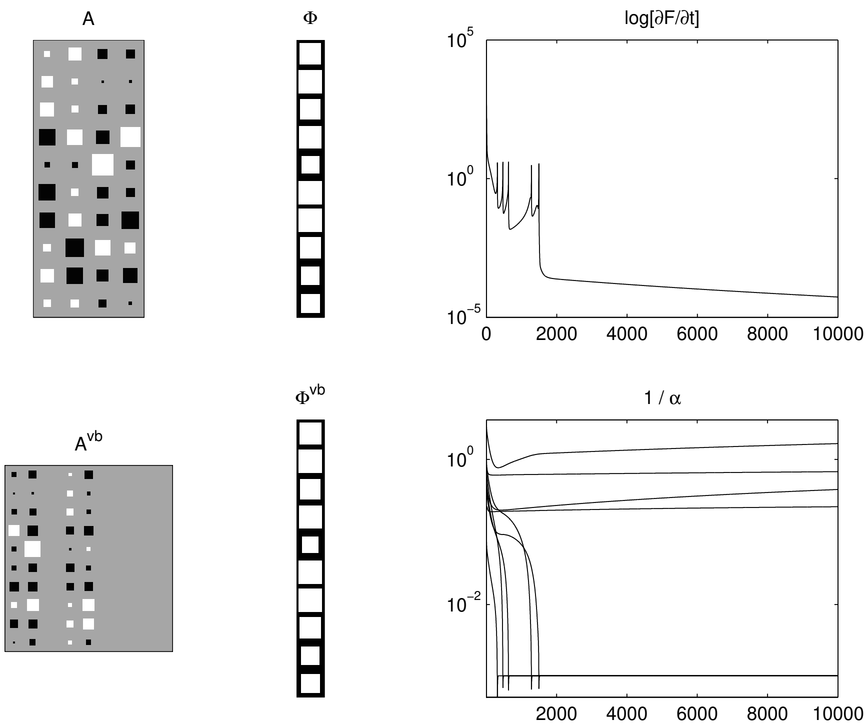


Figure 3.9: Upper left show the reference factor loading matrix and noise precisions. Bottom left show the learnt factor loading matrix and noise precisions. Upper right show the log to the difference of $\mathcal{F}$ during 10^4 iterations. Bottom right show the inverse precisions (variances) on the columns of $\mathbf{A}$.

VBPARAFAC in Action

To demonstrate the model I created a *reference* parallel factor analysis model with all parameters drawn from Gaussian and uniform distributions. From this model I generated two $p = 10$ dimensional training sets of size $N = 10^2$. The reference latent dimensionality was $k = 4$. The 4 factors generated were respectively Gaussian, Sinusoid, Super Gaussian and Sub Gaussian. No hyperparameter optimization was invoked. All priors were set to be non-informative. The algorithm was run for 10^4 iterations with a maximum dimensionality of $k_{max} = 9$ and a random initialization. Figure 3.9 shows the results. The learnt latent dimensionality is $k = 4$ since only four columns are left back. The ARD mechanism has correctly inferred the model order and the factors found are unique. Figure 3.10 shows the estimated factors which were not switched off. The strength of this model is definitely that the solution is unique. Thus, if the model hold and noise assumptions are correct we are ensured to find the true factors.

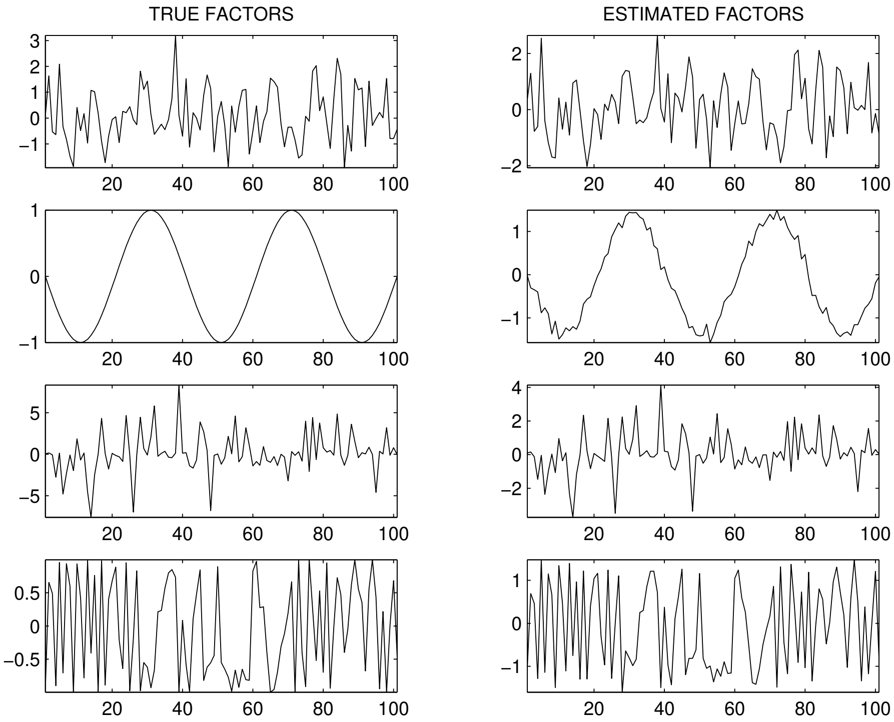


Figure 3.10: The true factors are shown in the left column. The estimated factors in the the right column. From top to bottom the true factors are of type: Gaussian, Sinusoid, Super Gaussian and Sub Gaussian.

Chapter 4

Experiments

This chapter is split in two sections. The first section covers experiments on synthetic data pointed towards an understanding of the proposed algorithms. In the second section, the algorithms are tested on fluorescence and fMRI data.

4.1 Synthetic Data

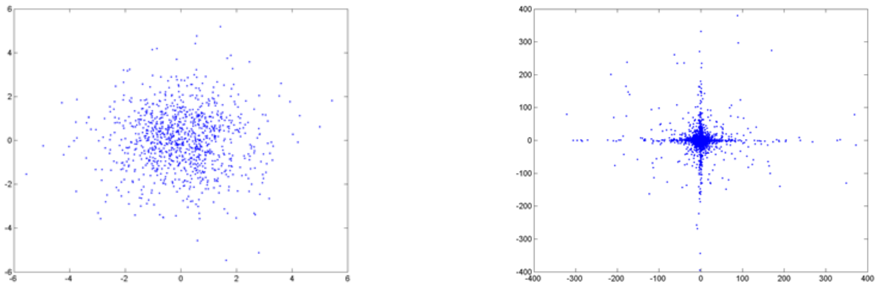
Four topics are covered in the following subsections. First, the ICA-like properties of VBFA are investigated. Second, model order selection is tested using BIC and the lower bound. Third, the marginal likelihood is tested as a guide for model selection. Finally, the effect of hyperparameter optimization is analyzed and compared to the generalization error.

4.1.1 VBFA $\approx$ ICA

The variational Bayesian factor analysis model is non-Gaussian because of the conjugate non-Gaussian Gamma priors. To see this, let's write down the posterior for the factors

$$\begin{aligned} p(\mathbf{S}|\mathbf{X}) &\propto \int p(\mathbf{X}, \mathbf{S}|\boldsymbol{\theta}) q(\boldsymbol{\theta}) d\boldsymbol{\theta} \\ &\Rightarrow \text{super Gaussian} \end{aligned} \tag{4.1}$$

where the parameters of VBFA are denoted by $\boldsymbol{\theta} = \{\mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\varphi}\}$ and the variational parameter posterior is denoted by $q(\boldsymbol{\theta}) = q(\mathbf{A})q(\boldsymbol{\alpha})q(\boldsymbol{\varphi})$. The variational posteriors $q(\boldsymbol{\alpha})$ and $q(\boldsymbol{\varphi})$ are factorized Gamma distributions. Performing the marginalization over these, the resulting posterior $p(\mathbf{S}|\mathbf{X})$ is a super Gaussian t-like distribution. Figure 4.1 shows data drawn the VBFA model. Setting the Gamma priors have big influence on the type of data which the model can capture. If the Gamma priors on the precisions on the



(a) (b)

Figure 4.1: Setting the priors in VBFA controls whether the model is Gaussian or not. The two subfigures show data generated from models with different values of the hyperparameters. Setting $\mathbf{a}^{\varphi} = \mathbf{b}^{\varphi} = \mathbf{a}^{\alpha} = \mathbf{b}^{\alpha} = 10^2$, we obtain a Gaussian model (a). Setting $\mathbf{a}^{\varphi} = \mathbf{b}^{\varphi} = \mathbf{a}^{\alpha} = \mathbf{b}^{\alpha} = 1$ we obtain a super Gaussian model (b).

factor loading matrix are set to be very narrow (delta-like), the effective prior becomes a t-distribution with many degrees of freedom. A t-distribution with degrees of freedom $\nu > 30$ is approximately a Gaussian distribution.
Experimentally, it is found that variational Bayesian factor analysis often finds the interesting directions in the data. The true factor loading matrix is recovered in many cases. In maximum likelihood factor analysis, this is not possible due to the rotation ambiguity, because it is a Gaussian model. In independent component analysis, one assumes that the factors (sources) follows a non-Gaussian distribution, implying that the model is no longer Gaussian. Kurtosis, $\langle (x-\mu)^4 \rangle - 3\langle (x-\mu)^2 \rangle^2$, is the quantity that many ICA algorithms maximize. The kurtosis is zero for Gaussian signals, positive for super Gaussian signals and negative for sub Gaussian signals. A super Gaussian distribution is a sparse distribution that has heavy tails. Speech signals are typically super Gaussian. Figure 4.2 shows the data types and models for which ICA is possible.
I tested a $k = 2$ order VBFA model when the factors, generating the data were distributed according to Gaussian, sub Gaussian and super Gaussian. A true factor loading matrix of dimension (6×2) was generated with the main directions in the first two dimensions. Figure 4.3 shows the estimated $\mathbf{A}$ matrix and figure 4.4 shows the corresponding estimated factors in the three scenarios. The VBFA model finds the correct directions in all cases. It is clear that VBFA has similar properties to those of ICA even though the model assumptions in ICA and VBFA are far from each other. ICA assumes non-Gaussian factors whereas VBFA assumes Gaussian factors. Anyway, they have similar properties because both models are non-Gaussian and have posterior distributions for the factors that are non-Gaussian.

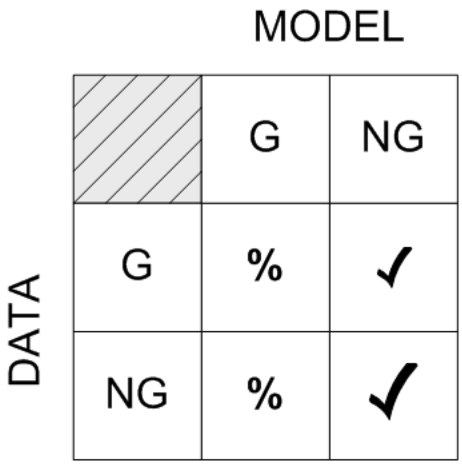
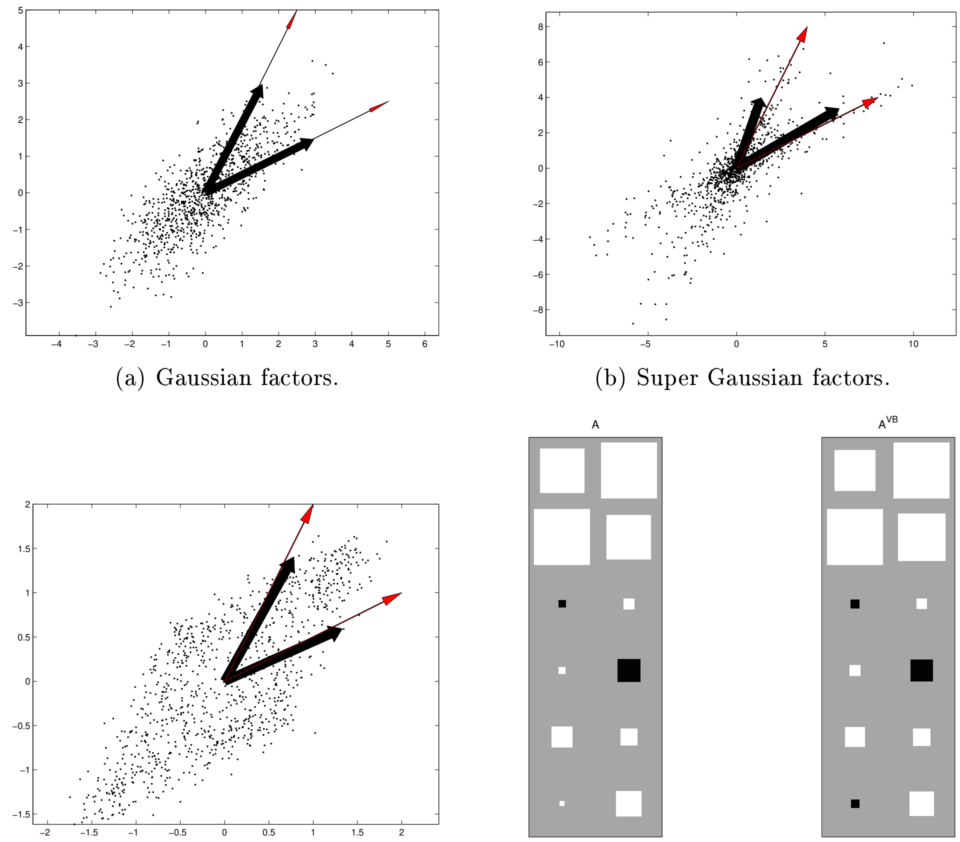


Figure 4.2: The figure shows in which cases ICA is possible. **G** and **NG** refer to Gaussian and non-Gaussian. For a Gaussian model, ICA is not possible. For a non-Gaussian model ICA is possible even in the case when the data is Gaussian distributed. In practice, this is because only a finite data set is available and therefore the data will only be approximately Gaussian distributed. The empirical kurtosis for Gaussian factors of limit length can be negative or positive and therefore it is possible to separate them.



(a) Gaussian factors.

(b) Super Gaussian factors.

(c) Sub Gaussian factors.

(d) True (left) and estimated (right) factor loading matrix.

Figure 4.3: A reference factor loading matrix of dimension (6×2) was generated, see (d) left. The two first entries in the columns of $\mathbf{A}$ are plotted as thin arrows. The corresponding VBFA estimates are plotted as thick arrows for: (a) Gaussian factors, (b) super Gaussian factors and (c) sub Gaussian factors. The estimated factor loading matrix was almost equivalent in all scenarios, see (d) right.

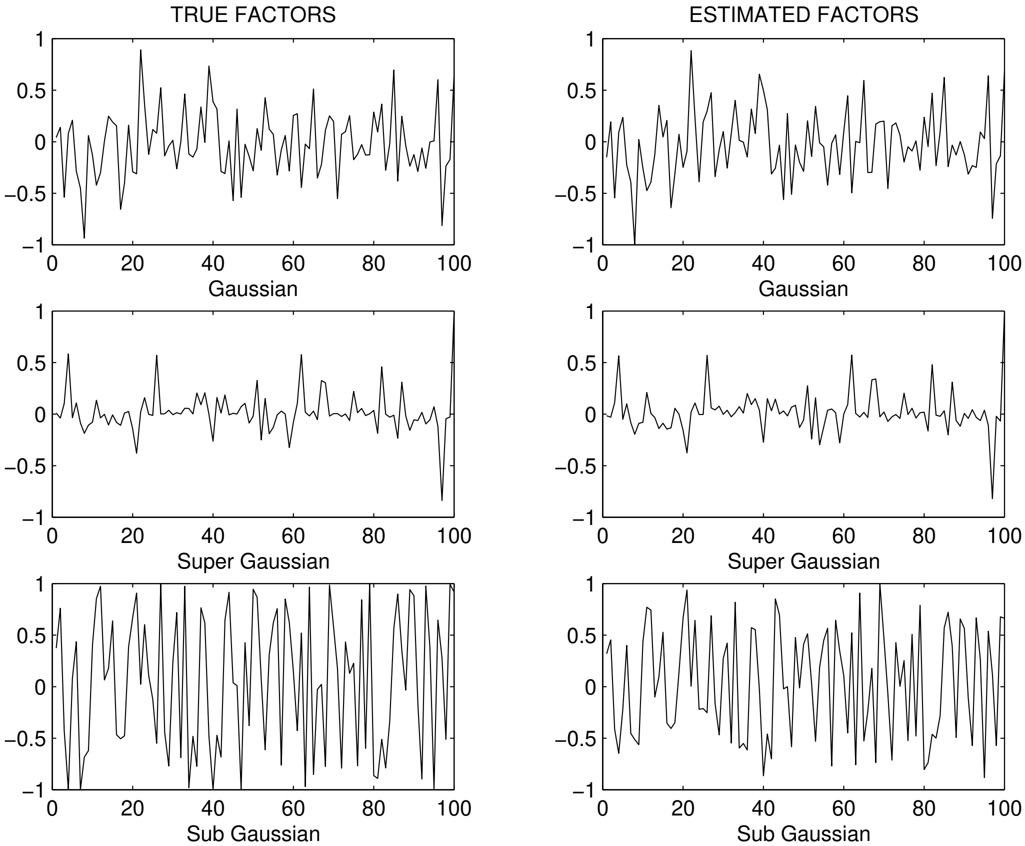


Figure 4.4: The true factors are shown in left column. The estimated factors in the right column. From top to bottom the true factor distributions are of type: Gaussian, super Gaussian and sub Gaussian.

4.1.2 Model Order Selection by BIC and VB

In this experiment I investigated the capability of scoring the right model order of a factor analysis model with BIC and the variational lower bound. A factor analysis model with $K = 5$ factors and a factor loading matrix of size (10×5) was used as reference. The BIC score and the variational lower bound were calculated for the MLFA and VBFA models having a latent dimensionality in the range $k = \{1 \ldots 9\}$, for varying sample sizes N. To avoid local maxima, every BIC and VB score was calculated 10 times and the best solution chosen. The results are shown in figure 4.5. It is obvious that the VB bound has some under-fitting tendencies. This bias towards simpler models is of course undesirable. We can roughly explain this phenomena by considering the difference in the exact log marginal likelihoods $\mathcal{L}(k{+}1){-}\mathcal{L}(k)$ and assuming that the KL-penalty is the same for each column of $\mathbf{A}$

$$\begin{aligned}
\mathcal{L}(k+1) - \mathcal{L}(k) &= [\mathcal{F}(k+1) + (k+1)\mathrm{KL}] - [\mathcal{F}(k) + (k)\mathrm{KL}] \\
&= \mathcal{F}(k+1) - \mathcal{F}(k) + \mathrm{KL} \\
&\neq \mathcal{F}(k+1) - \mathcal{F}(k)
\end{aligned} \tag{4.2}$$

where the last line is the basis of our decision. If each column in $\mathbf{A}$ contributes an independent KL-penalty, a systematic error exist when comparing model orders. According to Attias [2], BIC is recovered in the large sample limit of the variational Bayesian lower bound. Referring to the bottom right subplot of figure 4.5 it is seen that they indeed are coinciding.

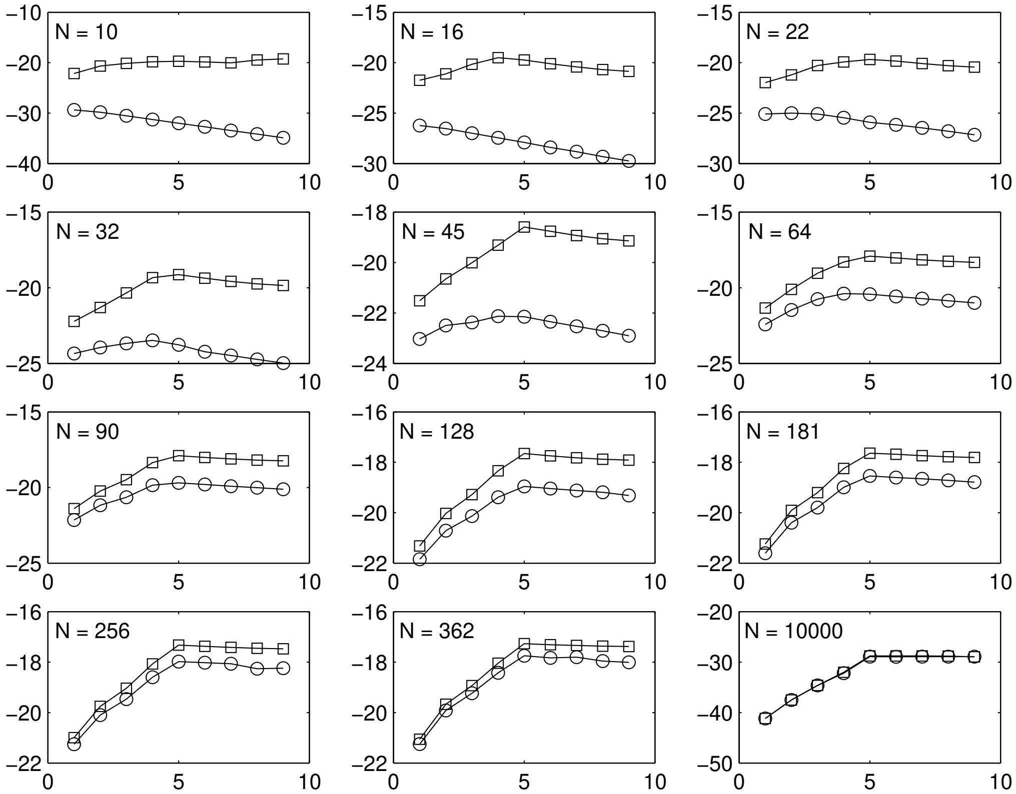
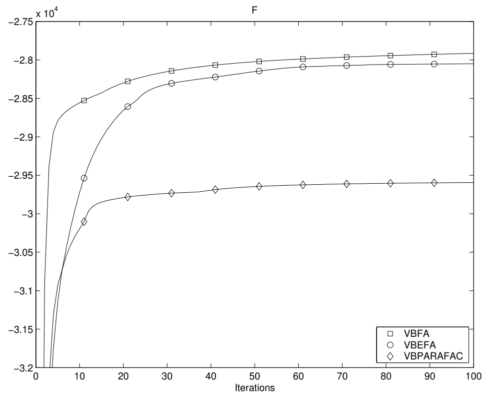


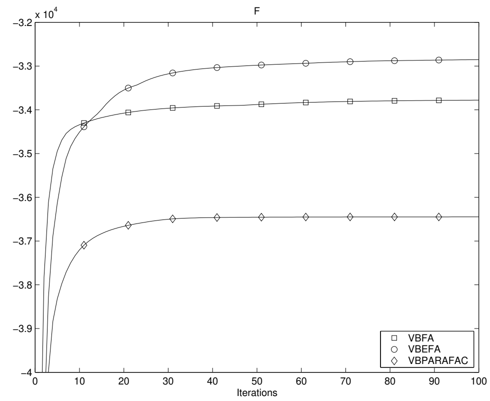
Figure 4.5: The subplots show the MLFA BIC score ($-\Box-$) and the VBFA lower bound ($-\circ-$) for different k and sample size N. Each score is normalized by N. The true latent dimensionality is $K = 5$. Notice that BIC recovers the correct model order for $N \geq 22$. The VB lower bound finds the true model order for $N \geq 90$. For large N they approach each other, but for small sample sizes the VB lower bound suggests a too simple model order.

4.1.3 Model Selection using VB

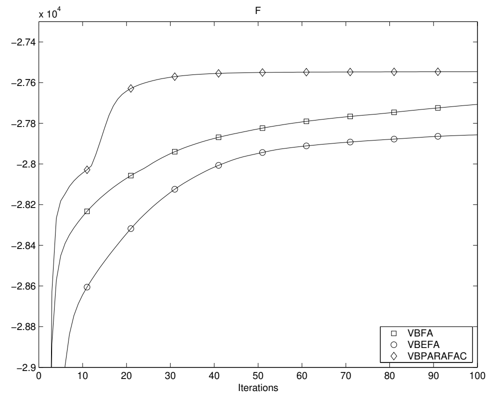
In this experiment, I tested the variational Bayesian lower bound for a model selection problem. The models tested are VBFA, VBEFA and VB-PARAFAC. Data sets following the three models were constructed and the lower bound was calculated for each model when trained on the different types of data. The number of batches was set to $M = 2$ and the data dimension was $p = 10$. All data sets were generated with a latent dimensionality of $k = 3$. The individual factor dimensionality for the EFA data was $K_T^1 = 2$ and $K_T^2 = 1$. The FA data was generated with a $Mp = 20$ factor loading matrix. The number of samples generated was $N = 10^3$. All algorithms were run for 10^2 iterations, having non-informative priors and a maximum latent dimensionality of $K = K_T^m = 5$. Local maxima were avoided by running the algorithms 10 times and choosing the solution with the maximum lower bound. The convergence curves are shown in figure 4.6. In all cases the correct model is found, thus the model having the highest value for the lower bound after 10^2 iterations corresponds to the reference model that generated the data. Even though the models were trained with the wrong latent dimensionality, this did not astray the inference. The ARD mechanism worked properly and switched off unneeded factors. It is worth noting that the lower bound obtained with the VBPARAFAC model is significant lower when trained on FA and EFA data. This model is much more restrained and not very probable for generating this type of data. Equivalent experiments for different sample sizes and data dimensionality all yielded the same result, thus the lower bound is very robust in selection amongst these models.
In summary, this leads to the conclusion that the variational Bayesian lower bound is indeed worth using as a guide for model selection.



(a) FA data.



(b) EFA data.



(c) PARAFAC data.

Figure 4.6: The lower bounds for VBFA, VBEFA and VBPARAFAC are calculated when trained on data sets generated by (a) factor analysis, (b) extended factor analysis and (c) parallel factor analysis. See text for details.

4.1.4 Hyperparameter Optimization and Generalization

In this section I investigate the generalization performance of the factor analysis model using maximum likelihood and variational Bayesian parameter learning. In the following subsections the generalization performance is mainly visualized by learning curves. A learning curve is a plot showing the predictive power of a model on an independent test set, when trained on varying training set sizes. Learning curves are typical useful when comparing models of varying complexity. The test error is measured as the negative log-likelihood and is an unbiased estimate of the generalization error. In terms of negative log-likelihood a low test error corresponds to high probability for the model. A synthetic test set $\mathcal{D}$ of size $N = 10^6$ was generated from a *reference* factor analysis model with $k = 3$ Gaussian factors of unit variance. The factors are embedded in a 10-dimensional space through a factor loading matrix and added with Gaussian noise with a diagonal covariance. The elements of the factor loading matrix are drawn from a Gaussian distribution of unit variance. The diagonal entries of the noise covariance (the sensor variance in each dimension) are sampled from a uniform distribution in the range {0-1}. Choosing the parameter distributions of the reference model this way, the variational algorithm has no advantage compared to the maximum likelihood algorithm. The same test set is used throughout the experiments and training sets are generated using the reference model.

Maximum Likelihood

An often encountered scenario, is that a simple erroneous model can outperform a realistic but complicated model, when trained on small data sets . To illustrate this behavior, three MLFA models with increasing latent dimensionality, $k = \{1, 3, 5\}$, were trained and average test errors were calculated on the test set. For each training set of a fixed size, the average test error was based on 100 separate runs with different initializations and training sets. The resulting learning curves are plotted in figure 4.7 and we note that the above mentioned scenario is indeed present. For $N \geq 16$, the average test error is minimized for models trained with the reference latent dimensionality $k = 3$. For $N < 16$ the average test error is minimized for models with $k = 1$. In this case there is simply not enough data to support a more flexible model. When N increases the average test error for the MLFA model with $k = 5$ approaches the MLFA model with the same dimensionality as the reference. This illustrates the need for parameter penalizing information criterions like e.g. BIC, when determining the model order in a maximum likelihood setting. One can decompose the generalization error into the sum of the *bias* squared and the *variance*, see e.g. Bishop [6]. A model which is too inflexible will have a high bias, while a model which is very flexible will

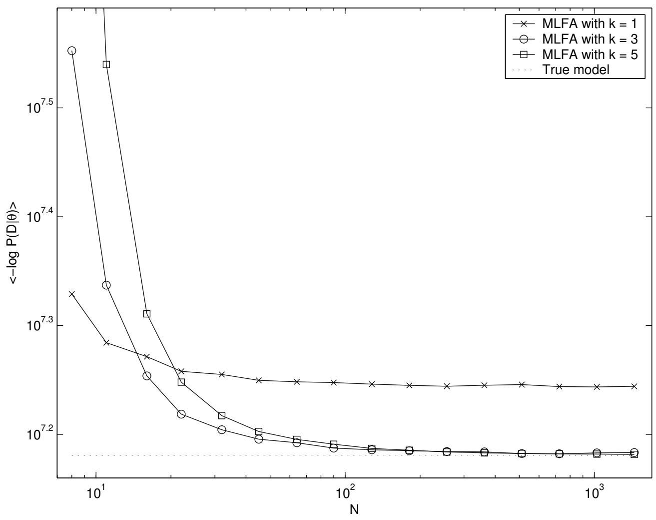
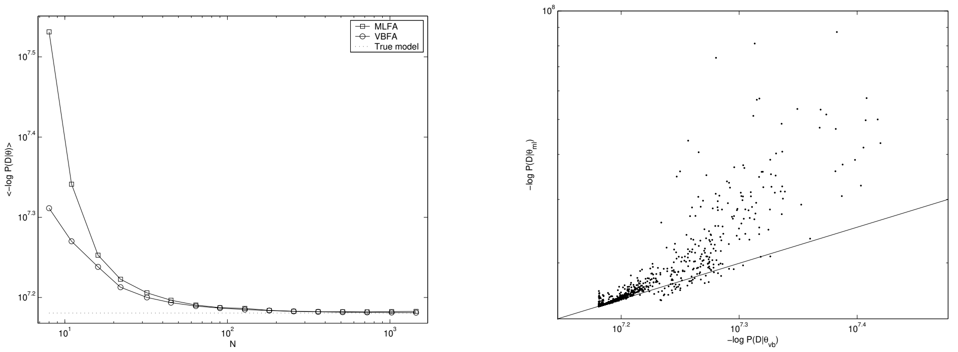


Figure 4.7: Learning curves for MLFA models of increasing latent dimensionality. Plotted as the dotted line is the test error obtained when using the reference parameters.

in relation to the particular data set have a high variance. In this context we should expect the factor analysis model with $k = 1$ to have a high bias, and the model with $k = 5$ to have a high variance. The model with $k = 3$ should give the best compromise between bias and variance.

Maximum Likelihood vs. Variational Bayes

In order to investigate the generalization performance of the variational algorithms proposed, both MLFA and VBFA models were trained for comparison. Learning curves were calculated in the same manner as described in the previous experiment. Both MLFA and VBFA models were trained with the same dimensionality as the reference. The VBFA model was not optimized with respect to the hyperparameters. Figure 4.8 shows the learning curves and a scatter plot with all test errors. It is evident, referring to figure 4.8(a), that the VB-algorithm on average has a lower test error than the ML-algorithm for small N. The variational Bayesian approach integrates over parameters and therefore does not suffer from overfitting. This is the case with the maximum likelihood approach. In addition, the ARD mechanism controls the model flexibility by damping the unneeded degrees of freedom in the model. Nevertheless, when N increases, the generalization performances of the two methods do not differ. Figure 4.8(b) shows the test errors from the algorithms plotted against each other and a line $x = y$ is added. When a point is above the line, it corresponds to a larger test error of the ML-algorithm than of the VB-algorithm for that particular training



(a) Learning curves for MLFA and VBFA.

(b) Scatter plot of test errors by MLFA (y-axis) and VBFA (x-axis).

Figure 4.8: Test errors for MLFA and VBFA visualized as learning curves (a) and scatter plot (b).

set. Analyzing these plots, one can conclude that parameter estimation by the VB-algorithm compared to the ML-algorithm is consistently better in terms of generalization, not only on average but also in the typical case.

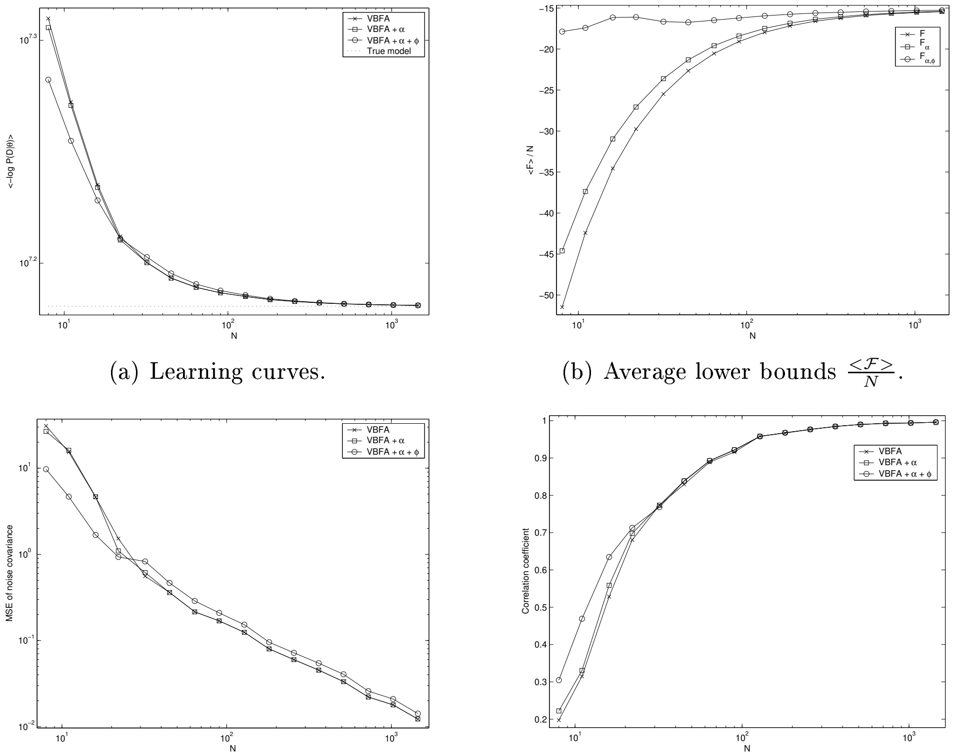
Hyperparameter Optimization

Hyperparameter optimization leads to an increase in the lower bound on the marginal likelihood, but does it have any effect on the generalization?. This and related questions are investigated in the following experiments. Hyperparameter optimization has many names: *empirical Bayes*, *the evidence framework*, *maximum likelihood II* and *maximum marginal likelihood*. Despite the confusion, they all refers to the same practice of optimizing the hyperparameters of the priors. I will test two scenarios of hyperparameter optimization. In the first experiments, I will consider a model where the priors for the α_k precisions are identical and the priors for the φ_j precisions are identical. So in this case, I denote $\mathbf{a}^\alpha$ by a^α, $\mathbf{b}^\alpha$ by b^α etc. The model is called VBFA+α+φ if both sets of hyperparameters are optimized and e.g VBFA+φ if only the noise precision prior is optimized. Later, I will investigate what happens if all priors are optimized individually. This model is denoted by VBFA+$\boldsymbol{\alpha}$+$\boldsymbol{\varphi}$. Generalization is tested for two different reference models. The reference model previously used and one having the parameters drawn from distributions assumed by the VBFA model.

The VBFA model has hyperparameters $\{a^\varphi, b^\varphi\}$ interacting in the variational posterior $q(\boldsymbol{\varphi})$ and $\{a^\alpha, b^\alpha\}$ interacting in the variational posterior $q(\boldsymbol{\alpha})$. In the following experiment three models were trained. A model with all hyperparameters and hyperhyperparameters fixed as $a^\varphi = b^\varphi = a^\alpha = b^\alpha = 10^{-3}$

corresponding to non-informative/broad priors. A model with only a^α and b^α optimized, and one where all hyperparameters and hyperhyperparameters are optimized. Figure 4.9 shows learning curves, lower bounds, noise covariance mean square errors (MSE) and factor loading matrix correlation coefficients. All quantities were averaged over 100 runs for each N. The mean square errors were measured using the reference noise covariance. The correlation coefficients were obtained using the reference factor loading matrix. The reference and the estimated factor loading matrices were rotated by singular value decomposition (SVD) and the absolute value taken prior to calculation of correlation coefficients.
The three learning curves approach the reference model in the large sample limit. When N increases the mean square errors of the noise covariance decreases almost linearly in a log-log plot and the correlation coefficients approach 1. This leads to the conclusion that the reproducibility of the parameters and the generalization are strongly correlated and that the general performance depends crucially on the training set size. As expected, the lower bounds obtained when optimizing with respect to hyperparameters are at all times higher, especially for the VBFA+α+φ model. Some interesting observations are obtained when using the fully optimized model. The generalization is improved to high a degree for small N, but for $N \geq 32$, the generalization is slightly better for the two other models. A high value of the lower bound does not imply better generalization in this case. From figures 4.9(c) and 4.9(d), we note that both the correlation coefficient and the noise MSE are better for small sample sizes for the VBFA+α+φ model. For larger sample sizes the noise covariance MSE is higher, so apparently the generalization error is connected to the MSE in this case. Considering the VBFA+α model compared to the not optimized model, we note that optimizing the α precision hyperparameters has a small positive effect on the generalization, but does not have as much impact as optimizing the noise precision hyperparameters. The reason why VBFA+α+φ leads to better generalization for small N and worse generalization for moderate N is simply because it is a more biased model. The prior common to all the noise precisions is optimized to a certain mean and variance which is common to all φ_j. In this way the model is restricted in the sense that φ_j should not be to far from another $\varphi_{j'}$. When N increases, the prior has less influence and the generalization is identical to the generalizations of the other models.
Figure 4.10 shows what happens during a single run of VBFA+α+φ. The bound increases and the log-likelihood drops when hyperparameter optimization is invoked. It is obvious from the trajectories of the hyperparameters that the optimization restricts the space of values which they can take.

The parameters in the reference model used in the previous experiment were sampled from distributions not assumed by the VBFA model. A reference model with parameters following the VBFA model assumptions was



(a) Learning curves.

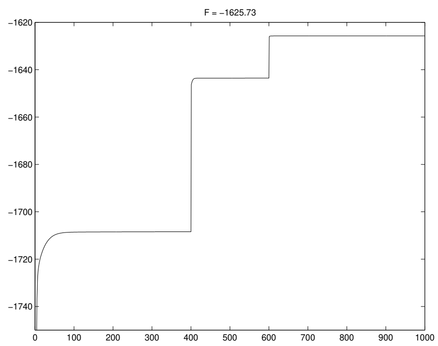
(b) Average lower bounds $\frac{<\mathcal{F}>}{N}$.

(c) Mean square errors.

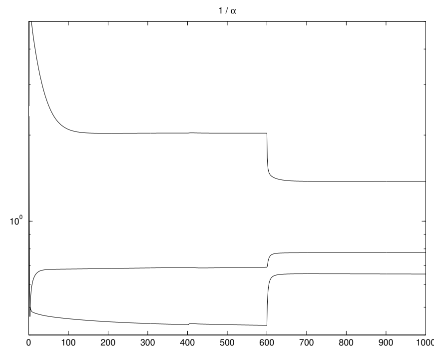
(d) Correlation coefficients

Figure 4.9: Three models with different hyperparameter optimization schemes are compared: VBFA, VBFA+α and VBFA+α+φ. The reference model is used for generating the training sets. The subfigures show learning curves (a), average lower bounds (b), mean square errors with respect to the reference noise covariance (c) and correlation coefficients with respect to the reference factor loading matrix (d).

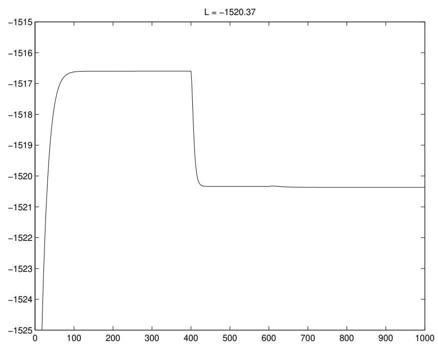
constructed in order to investigate any possible differences. The reference Gamma priors were set as $a^\varphi = b^\varphi = a^\alpha = b^\alpha = 1$ corresponding to a mean value and variance of 1. Proceeding as in the previous experiment the results are visualized in figure 4.11. In addition, a fourth model having the same hyperparameter and hyperhyperparameter values as the reference priors was trained. We note that the model having the true priors also has the lowest test error and highest correlation coefficients. The worst performing model is the one with fixed non-informative priors. The non-informative prior is far from the truth and optimizing therefore is a better alternative. When the parameters in the reference model are sampled from the same type of distributions assumed by the VBFA model, the VBFA+α+φ model is more well behaved compared to last experiment. For all N, the test error for the



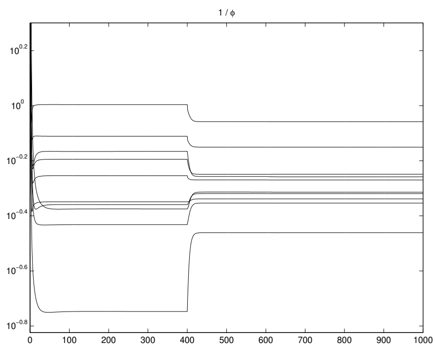
(a) Lower bound.



(b) $\frac{1}{\alpha_k}$.



(c) Log likelihood.

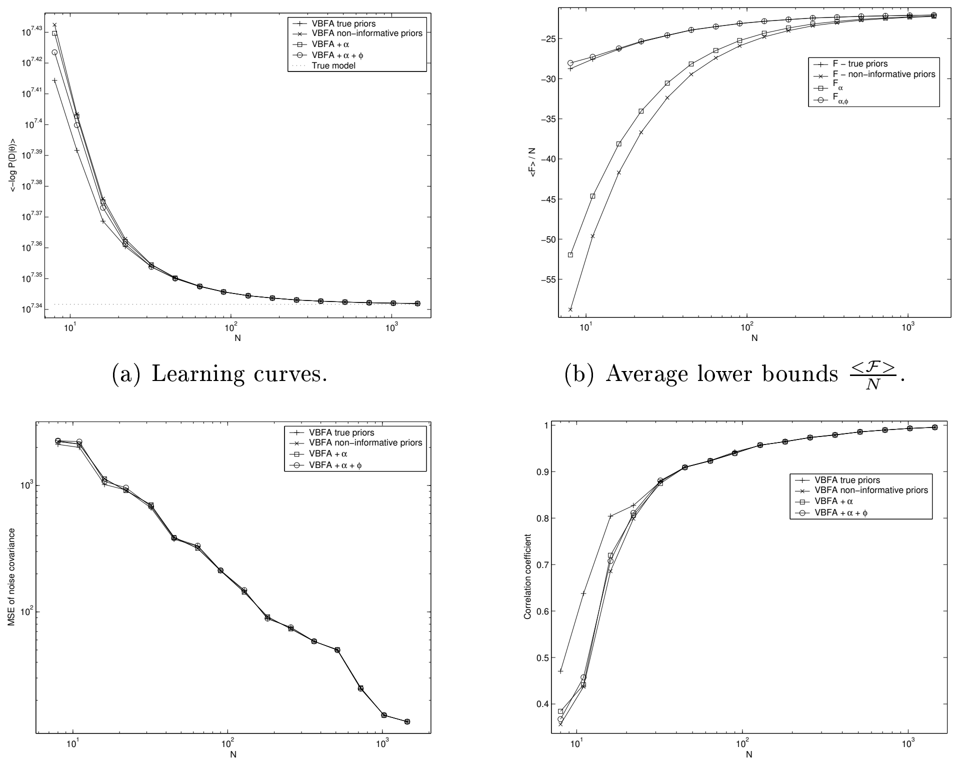


(d) $\frac{1}{\varphi_j}$.

Figure 4.10: The subfigures show: (a) lower bound, (b) $\frac{1}{\alpha_k}$, (c) log likelihood and (d) $\frac{1}{\varphi_j}$. VBFA is optimized with respect to the hyperparameters $\{a^\varphi, b^\varphi, a^\alpha, b^\alpha\}$. Note that this causes all priors $p(\alpha_k|a^\alpha, b^\alpha)$ to be identical and all priors $p(\varphi_j|a^\varphi, b^\varphi)$ to be identical. The optimization schemes are invoked after 400 and 600 iterations. The parameters in the reference model were drawn from uniform and Gaussian distributions. Notice how hyperparameter optimization in this case constrains the model by minimizing the range of values the hyperparameters can take. The log likelihood drops and the lower bound increases as an effect of hyperparameter optimization.

noise optimized model is between the two models having respectively true priors or non-informative priors.

If we set the priors to $p(\alpha_k|\mathbf{a}^\alpha, \mathbf{b}^\alpha)$ and $p(\varphi_j|\mathbf{a}^\varphi, \mathbf{b}^\varphi)$, there is a separate prior for each α_k and φ_j which can be optimized. A reference model having its parameters sampled according the VBFA assumptions was constructed and four different models were tested. The resulting learning curves are shown in figure 4.12. Note that the generalization is worse for the VBFA+$\boldsymbol{\alpha}$+$\boldsymbol{\varphi}$ model. It clearly has over-fitting tendencies. The over-fitting only arises



(a) Learning curves.

(b) Average lower bounds $\frac{<\mathcal{F}>}{N}$.

(c) Mean square errors.

(d) Correlation coefficients

Figure 4.11: Four models with different hyperparameter settings or optimization schemes are compared: VBFA with true priors, VBFA with non-informative priors, VBFA+α and VBFA+α+φ. The reference model used for these plots are constructed by sampling the noise covariance φ and precision α from Gamma distributions having a mean and variance of 1. The subfigures shows learning curves (a), average lower bounds (b), mean square errors with respect to the reference noise covariance (c) and correlation coefficients with respect to the reference factor loading matrix (d).

from optimization of the noise precision priors. The VBFA+$\boldsymbol{\alpha}$ model shows no over-fitting tendencies, in fact it shows nothing. This model simply has the same generalization error as the model with non-informative priors.

An often cited reference regarding hyperparameter optimization is MacKay [26]. He argues that hyperparameters cannot fit to the noise in the data because it is only the parameters that can do so. This, he mention when considering a regularization hyperparameter like α_k. In the worst case scenario $\alpha_k = 0$ and no regularization is present. Otherwise, the effect is a damping of the unneeded degrees of freedom in the model. Thus, the hyper-

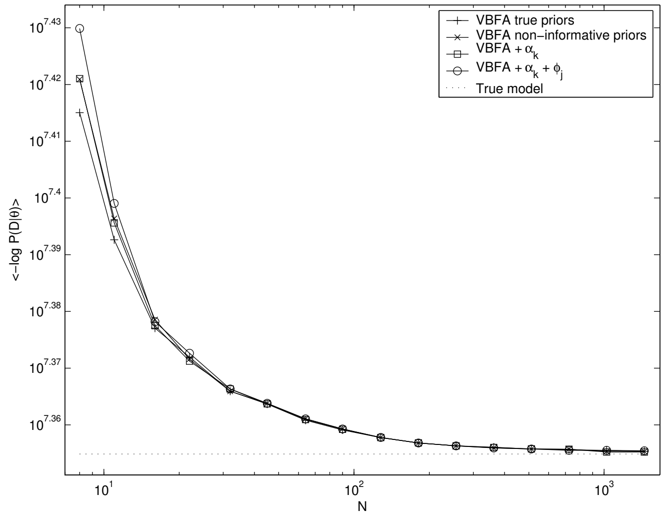


Figure 4.12: Learning curves for 4 VBFA models with different hyperparameter settings. The first model is given the true priors. The second has non-informative priors. The third model is VBFA+$\boldsymbol{\alpha}$. The fourth model is VBFA+$\boldsymbol{\alpha}$+$\boldsymbol{\varphi}$.

parameters have no effect on the worst-case capacity of the model.
When referring to the previous experiments this indeed seems to be the case. Optimizing the priors for the $\boldsymbol{\alpha}$ precisions has no significant effect on the generalization. When referring to the hierarchy of the VBFA model, this might be the result of optimization on a high level in the model hierarchy. Actually, optimizing these hyperhyperparameters does not make any sense at all. Especially for the VBFA+α model where the priors are identical for all α_k. In this case optimization only hampers the ARD process. Any prior which is not non-informative will limit the space of possible values the precisions can take. The reason for introducing the ARD prior was its ability to prune away unneeded factors. This is done by allowing some precisions to go to infinity, thereby introducing a prior which is not flat is in fact contradictory. Optimization of the noise precisions showed to have a great impact on the generalization. For the VBFA+α+φ model this lead to a more biased model in which the generalization was improved for small sample sizes. Optimization of the VBFA+$\boldsymbol{\alpha}$+$\boldsymbol{\varphi}$ model lead to worse generalization and a model which overfitted to the noise.
From this we can conclude that it is necessary to distinguish between 'noise' parameters and 'regularization' parameters when performing hyperparameter optimization. Definitely we should avoid the use of hyperparameter optimization of priors that control 'noise' parameters, unless one wants a model like VBFA+α+φ. We could loosely categorize this model as being something in between pPCA and factor analysis. Considering the experiments with the VBFA+α+φ model, the results indicates that the hyperparameter optimiza-

tion was only beneficial when the data strictly followed the assumed model. However, modelling real data, this result is really not for any practical use.

It is definitely clear that hyperparameter optimization for the VBFA model is not recommendable. There are several reasons: A high lower bound does not imply good generalization. Overfitting can occur when optimizing hyperparameters that controls a 'noise' parameter. The ARD mechanism can become restrained. Hyperparameter optimization is computationally costly. For large sample sizes the priors are less important and hyperparameter optimization has no effect.
The proper thing to do in a strict Bayesian analysis is to integrate out all uncertain variables and optimization is only suboptimal. Thus, for this model, I believe it is better to set the priors empirically or to use the non-informative approach. For other kind of models where hierarchical priors are not tractable to handle or sensible to include, hyperparameter optimization can probably be very beneficial when used on regularization parameters.

4.2 Real Data

In this section, the variational algorithms are tested on real data. The VBPARAFAC model is tested on an amino acids fluorescence data set. The remaining experiments involve two fMRI data sets which are analyzed with the proposed variational algorithms and *probabilistic partial least squares* (pPLS).

4.2.1 Analyzing Amino Acids Fluorescence Data with VBPARAFAC

In chemometrics, PARAFAC is widely used for analyzing fluorescence data. In the following experiment I test the variational PARAFAC model with this type of data. The data is due to Claus A. Andersson [8]. Five samples containing different amounts of tyrosine, tryptophane and phenylalanine were measured by fluorescence (excitation $250-300$ nm, emission $250-450$ nm, 1 nm intervals). The samples were measured on a PE LS50B spectrofluorometer with excitation slitwidth of 2.5 nm, an emission slitwidth of 10 nm and a scanspeed of 1500 nm/s. The five samples are plotted in figure 4.13. Notice the fluctuations in the upper left corner (small emission wavelength and high excitation wavelength). These fluctuations are due to Rayleigh scatter.
According to Bro [7], a PARAFAC model with $k = 3$ components should be appropriate to model the data. Therefore, I estimated the variational PARAFAC model for $k = 3$. All priors and hyperpriors were set to be non-informative. The data was not preprocessed in any way. Figure 4.14 shows the resulting unit norm factors and factor loadings when running 10^3 iterations of VBPARAFAC. The results are identical to what plotted in Bro [8]. Figure 4.15 shows the corresponding lower bound of the marginal likelihood.
A general problem with PARAFAC is the dependence on initialization. Using ALS, one often has to restart the algorithm many times to reach a good minimum of the error function. This seems also to be the case when using variational Bayesian learning. The many spikes of the log of the difference of $\mathcal{F}$ indicates that the 'error landscape' is not a smooth valley. If this was the case we should expect an exponential trajectory of the functional $\mathcal{F}$, because VBEM is coordinate ascent in function space of $q(\boldsymbol{\theta})$ and $q(\mathbf{s})$.

To test the ARD mechanism, a $k = 5$ VBPARAFAC model was trained. The resulting factors and factor loadings are shown on figure 4.16. Apparently, the ARD scheme does not work properly. The true spectra are still obtained, but it is obvious that component 4 and 5 are 'junk spectra'. The emission and excitation spectra for these components are more noisy and the spectra also take on negative values. The variances $(1/\alpha_j)$ for each column of $\mathbf{A}$ are plotted on figure 4.17. The variances are low for the junk spectra and high for the true spectra. Analyzing the spectra and the variances one can

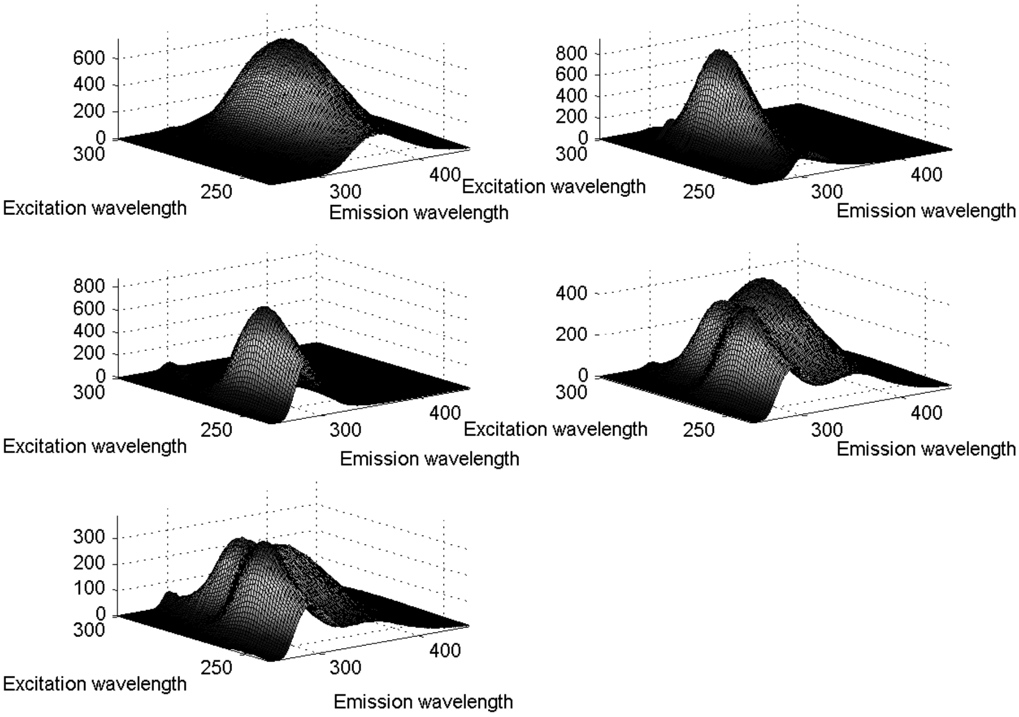
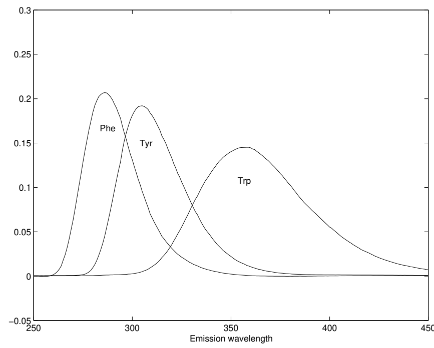


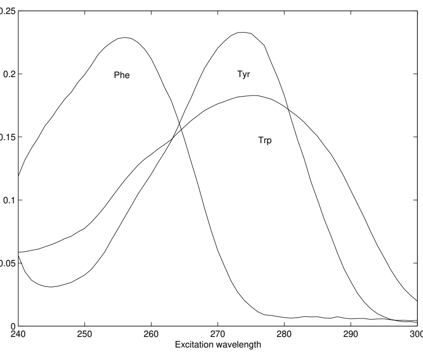
Figure 4.13: Five fluorescence samples containing Phe, Trp and Tyr.

conclude that three components is the proper choice. The ARD mechanism did not prune away the unneeded spectra. Now, can we reject the $k = 5$ model based on the lower bound. No, the lower bound for this run was calculated to $\mathcal{F} = -17.6 * 10^4$, while the lower bound for the model having $k = 3$ was calculated to $\mathcal{F} = -18 * 10^4$. To be sure that local optima was not in question, I performed several runs with different initializations. The model with $k = 5$ components simply fits the data better and according to the lower bound, it is the most probable model. I believe the reason why the lower bound does not favor the simpler model, is due to fact that VB-PARAFAC assumes Gaussian factors. The factors/spectra are clearly not Gaussian distributed and therefore the model priors are wrong. When the priors are wrong in a Bayesian model the inference step is no longer optimal.

As an experiment, I estimated two $k = 3$ component VBPARAFAC models, one with the data arranged as (excitation $\times$ emission $\times$ sample) and one as (emission $\times$ excitation $\times$ sample). In this way, an estimate of the variances in each dimension was obtained. For visualization, the variances were multiplied as an outer product to get a 3D-plot indicating where the noise is present, see figure 4.18. As expected, the estimated noise variances are highest in the areas where Rayleigh scatter might occur.



(a) Emission spectra.



(b) Excitation spectra.

Figure 4.14: A $k = 3$ component VBPARAFAC trained on raw fluorescence data. The estimated spectra: (a) Emission / factors and (b) Excitation / factor loadings. The labels: Phe, Trp and Tyr are set as in Bro [8].

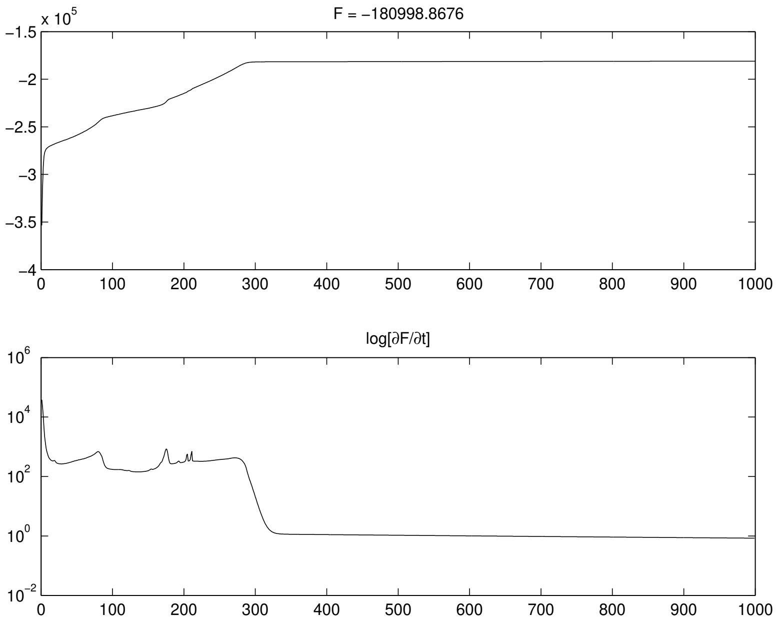
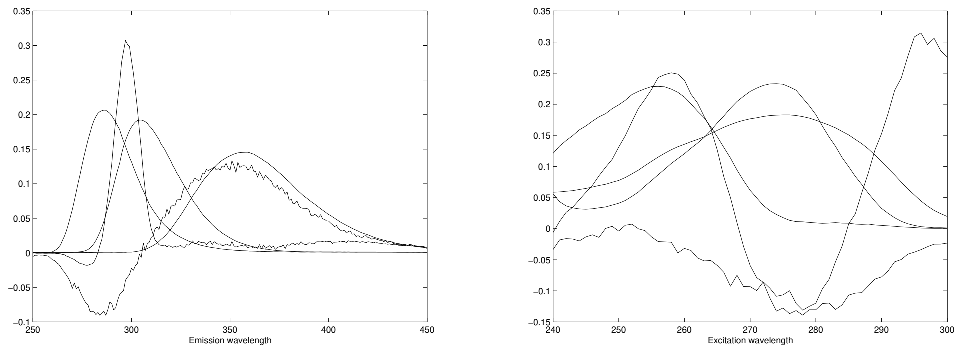


Figure 4.15: Convergence of the VBPARAFAC model when trained on the fluorescence data. Upper plot shows the lower bound $\mathcal{F}$ and the bottom plot shows $\log \frac{\partial \mathcal{F}}{\partial t}$ for 10^3 iterations.



(a) Emission spectra.

(b) Excitation spectra.

Figure 4.16: A $k = 5$ component VBPARAFAC trained on raw fluorescence data. The estimated spectra: (a) Emission / factors and (b) Excitation / factor loadings.

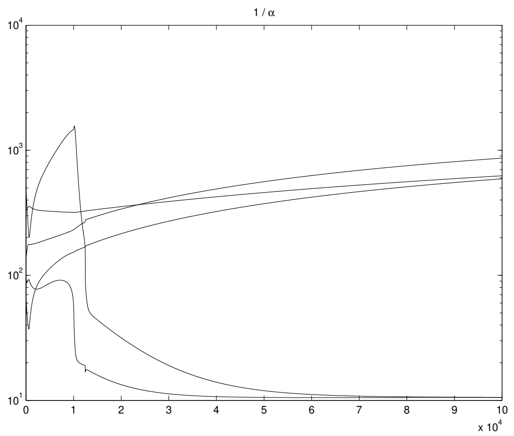


Figure 4.17: The plot shows the variance $(1/\alpha_j)$ for each column of $\mathbf{A}$. Notice that it takes approximately 10^4 iterations before the variances drop for the junk spectra.

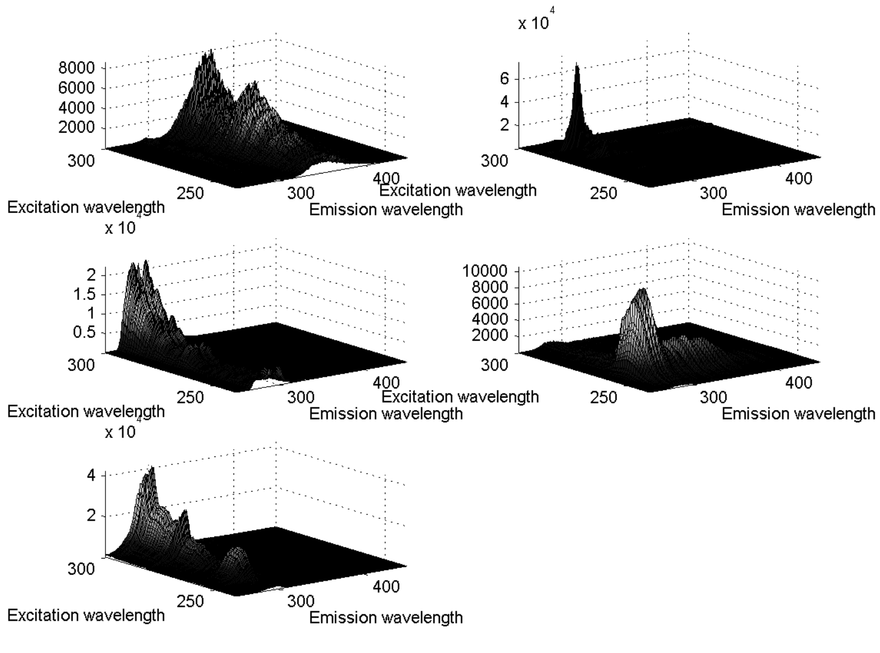


Figure 4.18: The subplots show the outer product of the estimated noise variances. Clearly VBPARAFAC estimates a high noise variance in the areas where Rayleigh scatter might be present.

4.2.2 Analyzing fMRI Data

In this section, I test the proposed variational algorithms and pPLS on a fMRI data set. A description of the data according to [15] is reproduced in the following.

The data set was acquired at Hvidovre Hospital on a 1.5-T Magnetom Vision MR scanner by Egill Rostrup. The scanning sequence was a 2-D gradient echo EPI (T2* weighted) with 66 ms echo time and 50 degrees of RF flip angle. The images were acquired with a matrix of 128×128 pixels, FOV of 230-mm, and 10-mm slice thickness, in a para-axial orientation parallel to calcarine sulcus. The region of interest (ROI) will be limited to a 68×82 two-dimensional (2-D) voxel map. The voxel dimension is $1.8 \times 1.8 \times 10$ mm. The visual paradigm consists of a rest period of 20 s of darkness using a light fixation dot, followed by 10 s of full-field checker board reversing at 8 Hz, and ending with 20 s of rest (darkness). In total, 150 images were acquired in 50 s, corresponding to a period of approximately TR = 330 ms/image. The experiment was repeated in ten separate runs containing 150 images each. in order to reduce saturation effects, the first 29 images were discarded, leaving 121 images for each run.

The data sets were acquired on the same subject, but during two separate scanning sessions, such that, e.g., the position and the shape of the slice are slightly different. In each case, the data set was built by combining the ten runs into a single sequence of 1210 images. To reduce the data even more, each sequence has been masked by a so-called omnibus F-test. This removes voxels which are not correlated to the stimuli. Furthermore, the data was preprocessed using a run-based detrending. The resulting data matrices are denoted by $\mathbf{X}$ (3891×1210) and $\mathbf{Y}$ (4372×1210).

VBFA

I tested the variational factor analysis model on the $\mathbf{X}$ data set. No hyperparameter optimization was used and different initializations with varying latent dimensions were tested. All priors were set to be non-informative. In the later section dealing with pPLS, the latent dimensionality was found to $k = 11$ by calculating test errors on an independent test set. Thus, I believe that the optimal dimensionality is in the range $k = 10 \ldots 20$ since only a part of the data was used when calculating test errors. Indeed, this seem also to be the case when observing the ARD hyperparameters for VBFA when trained with large latent dimension. I initialized a VBFA model with extreme $k = 100$ and the algorithm was set to switch off factors when the ARD hyperparameters were larger than 10^3. The run ended with $k = 22$ surviving factors. A visual inspection showed that only few of these had any significant meaning. The remaining factors were some kind of replica of other factors or factors that obviously modelled some spurious noise. If I removed these 'noise' factors or replica it caused a drop in $\mathcal{F}$ which was not recovered. It seem that the number of significant factors is around $3-5$. When running the algorithm no matter how many factors the average response is always attained. Furthermore, there is typically around $2-3$ 'high frequency' factors corresponding to e.g., heart beat or respiration. In addition, one or two interesting factors that are correlated with the stimuli are usually found. Figure 4.19 shows a typical run of VBFA. A $k = 20$ model was run for 1000 iterations. The factors were sorted according to the relevance ARD hyperparameters and the first ten are shown on the figure. As expected, the factor having the highest relevance, was very correlated with the stimuli. The second factor obviously models some high frequency content. Note that factor 6 is some kind of replica of factor number 2.

The 'noise' factors are not switched off by the ARD mechanism, thus one might think that this method is useless. The reason why ARD does not work appropriate, I believe, is because factor analysis in fact is a very simple model. It assumes linear mixing and uncorrelated Gaussian noise. None of these assumptions are very likely to be true for the fMRI data. Therefore, these extra 'noise' factors are needed to compensate for the model inaccuracy.

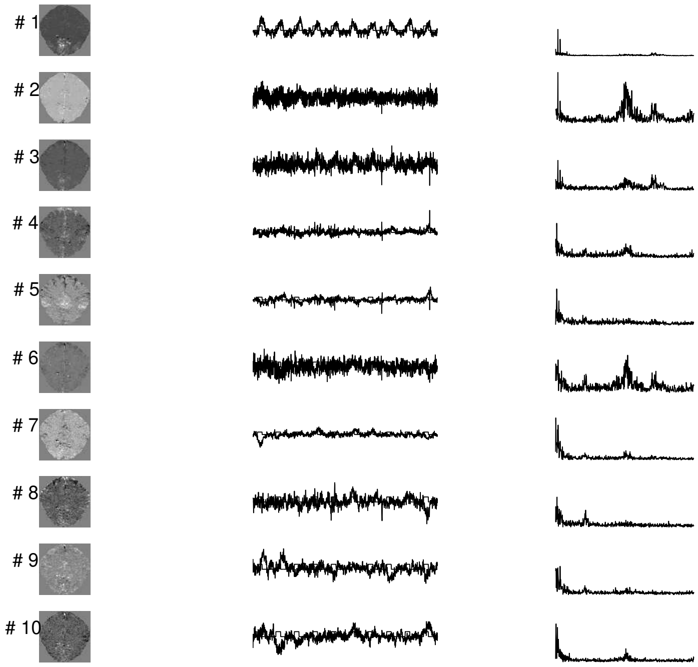


Figure 4.19: A $k = 20$ VBFA model was run for 1000 iterations. The subplots show ten of the found factors sorted according to the ARD hyperparameters. The first column shows the factor loads (columns of $\mathbf{A}$). The second column shows the factors. The third column shows the frequency spectrum of the factors.

VBEFA

I tested the variational extended factor analysis model on the data sets $\mathbf{X}$ and $\mathbf{Y}$. No hyperparameter optimization was used and different initializations with varying latent dimensions were tested. All priors were set to be non-informative. The optimization heuristic was used to focus the algorithm on the common factors. To ensure that a common factor was not interpreted as an individual factor, I initialized the algorithm with a latent dimension of zero for the individual factors. In this way the algorithm only attains individual factors when a common factor is switched off. The results from a run with $k = 10$ is shown in figures 4.20, figure 4.22 and figure 4.23. The figures respectively show: common factors, individual factors for $\mathbf{X}$ and individual factors for $\mathbf{Y}$. Four common factors are found and three individual factors for each data set. Note that the individual factors found are factors which have high frequency content, thus they are related to heart beat and breath. Figure 4.21 shows the estimated noise variances for subject $\mathbf{X}$ and $\mathbf{Y}$. As expected, the noise is concentrated in areas near vessels. This also illustrates that a factor analysis model is more appropriate than a PCA model which assumes isotropic noise.

A typical scenario which I have encountered is that the algorithm does not find any common factors. Especially for long runs of the algorithm. The obtained factors are similar to those plotted, but in many cases all common factors are rearranged as individual factors. When referring to figure 4.20 the common factor loads do not seem very coinciding. The factors being equivalent between two scanning sessions is probably a too restricted assumption. Nevertheless, the algorithm heuristic succeeds in discarding factors which are not common. Only factors that are strictly common to both data sets are retained.

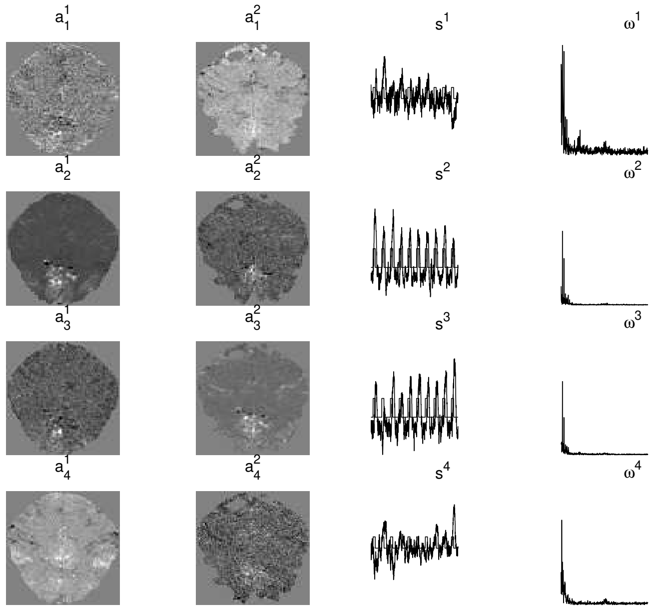


Figure 4.20: Four common factors are found from a run initialized with $k = 10$. Note that there is no high frequency content in the common factors.

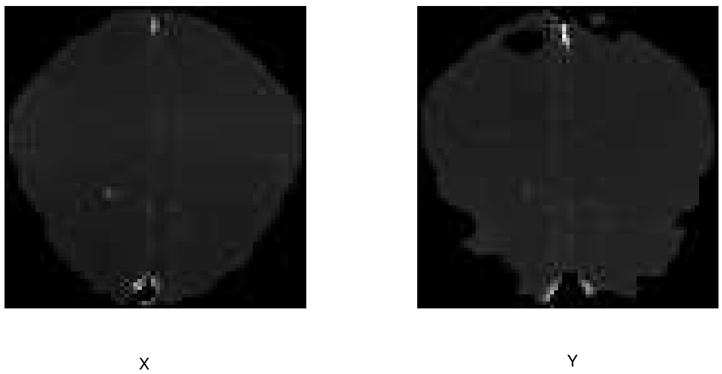


Figure 4.21: The left subplot shows the estimated noise variance of $\mathbf{X}$. The right subplot shows the estimated noise variance of $\mathbf{Y}$.

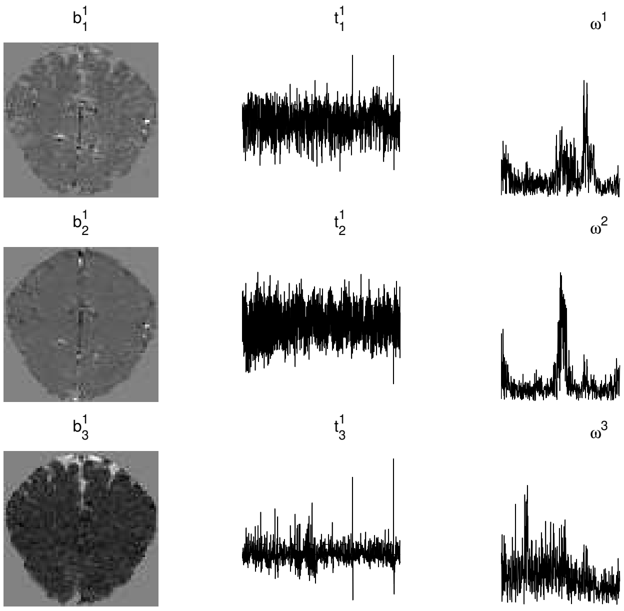


Figure 4.22: The individual factors of $\mathbf{X}$.

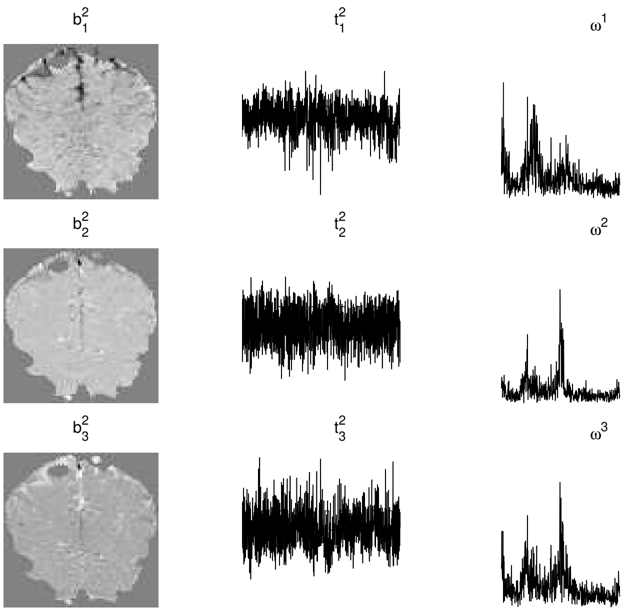


Figure 4.23: The individual factors of $\mathbf{Y}$.

VBPARAFAC

I tested the variational parafac model on the data sets $\mathbf{X}$ and $\mathbf{Y}$. No hyper-parameter optimization was used and different initializations with varying latent dimensions were tested. All priors were set to be non-informative. This model definitely has a much different performance compared to the previously tested models. The ARD mechanism is very effective when tested on the fMRI data compared to the amino acids data. Initializing with some model order k, usually half or less factors are found, the rest are automatically switched off. A model with more factors is always found to have a higher lower bound. As was the case for the amino acids data, we cannot determine the model order from $\mathcal{F}$. But by visual inspection one gets a clue of how to set the model order. A typical run of the algorithm, initialized with $k = 10$ is shown in figure 4.24. Running the algorithm for different k and initializations, it seems like there are $3 - 4$ distinct factors. One factor, which corresponds to the average response, is the 'activation' factor. Two high frequency factors and a fourth low frequency factor. The results from a run for $k = 4$ on both data sets are shown in figure 4.25 and figure 4.26. The activation factors are plotted in an enlarged format in figure 4.27. Clearly the factors and factor loads are very similar for the two data sets. How to interpret factor number two is not clear to me, but the others are obvious related the cardiac rhythm and the stimuli. In general the results are very reproducible. When running the algorithm from different initializations and for only 10 iterations, the results are in most cases identical to what is plotted in the figures.

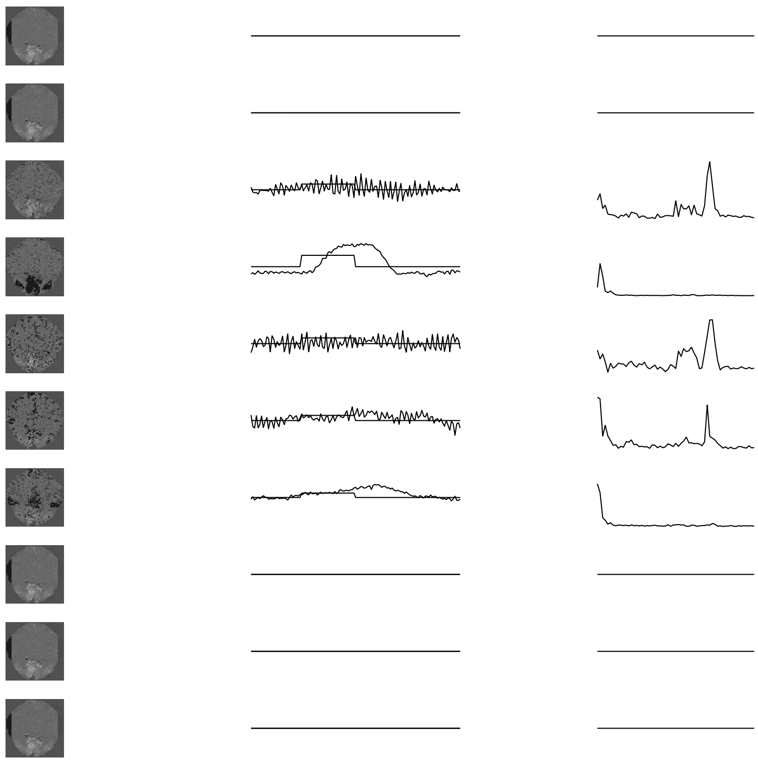


Figure 4.24: A typical run of VBPARAFAC, initialized with $k = 10$. All but five factors are switched off. Note that factors 3 and 5 have almost equivalent frequency spectra.

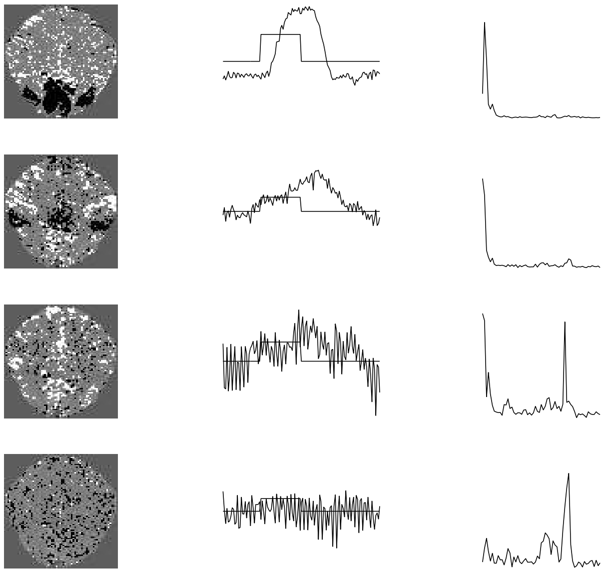


Figure 4.25: VBPARAFAC with $k = 4$ trained on $\mathbf{X}$.

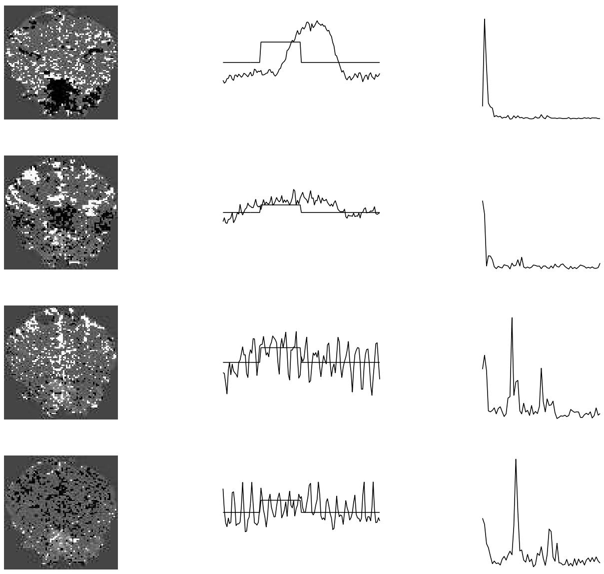


Figure 4.26: VBPARAFAC with $k = 4$ trained on $\mathbf{Y}$.

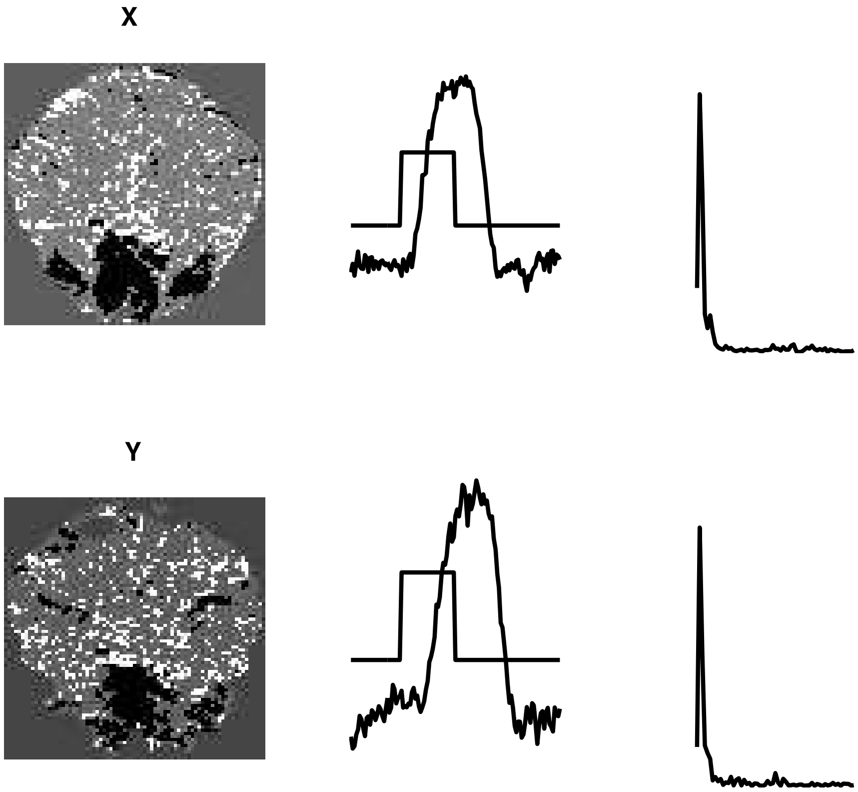


Figure 4.27: 'Activation' factors.

pPLS

Partial least squares (PLS) is a multivariate technique used in chemometrics and neuroimaging. The PLS model establishes a relation between a typical high dimensional or illposed data set and a design matrix. There are many different algorithms for fitting the PLS model. The simplest method involves computing a SVD on the cross covariance between the data set and the design matrix. PLS can be cast as a factor analysis of the concatenated set of images and design matrices

$$\begin{bmatrix} \mathbf{X} \\ \mathbf{D} \end{bmatrix} = \mathbf{AS} + \mathbf{E} \tag{4.3}$$

where the concatenated design matrix is denoted $\mathbf{D}$. Within this probabilistic framework it is possible to estimate the noise variances as well as estimating the model order that generalize best to new data. Hence the name probabilistic partial least squares. The design matrix used was set up representing the so-called 'agnostic' labelling scheme

$$\mathbf{D} = \left[\ \mathbf{I}^1_{121}, \mathbf{I}^2_{121}, \ldots \mathbf{I}^{10}_{121}\ \right] \tag{4.4}$$

where $\mathbf{I}^m_{121}$ is a 121×121 identity matrix. This type of design matrix does not provide any information on the stimuli, but it provides a link between the repeated experiments (epochs). Thus, in some sense, it is related to PARAFAC. I used the MLFA algorithm for the following experiments where the latent dimensionality was estimated by measuring the test error on an independent test set. For each model $k = 1 \ldots 20$, I calculated the training error (log-likelihood) on a training set of 5 epochs and the test error on the remaining 5 epochs. Figure 4.28 shows the log-likelihood evaluated on the training and test sets. The likelihood is maximized for a factor analysis model with $k = 11$. The results from the model having $k = 11$ are shown in figure 4.29. The resulting factor loading matrix was rotated by SVD and the maximum variance factor extracted. The subplots show the maximum variance factor, factor load, power spectrum and the 'design vector' (the factor load for the design matrix). Notice that the design vector is less noisy and can be thought of as being the prototype activation response.

An NPAIRS analysis was performed to get error bars on the design vector. The data set was randomly split into 50 different pairs. MLFA was run for 400 iterations for each pair and the mean and the squared difference of the design vector were calculated. The result is shown on figure 4.30. Note that the average design vector is almost identical to the 'activation' factor found by VBPARAFAC, i.e., the dip at scan number 100. A statistical parametric map (SPM) showing all the maximum variance factor loads can be seen on figure 4.31.

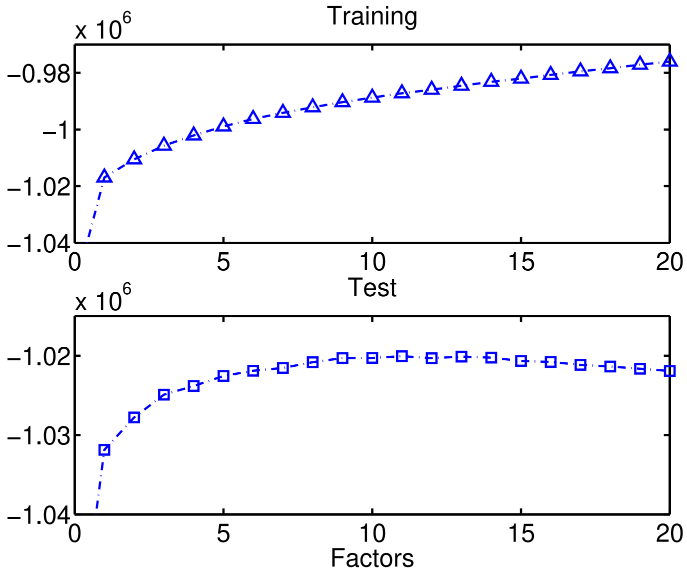


Figure 4.28: Log likelihood evaluated on training and test sets. $k = 11$ is the found latent dimensionality.

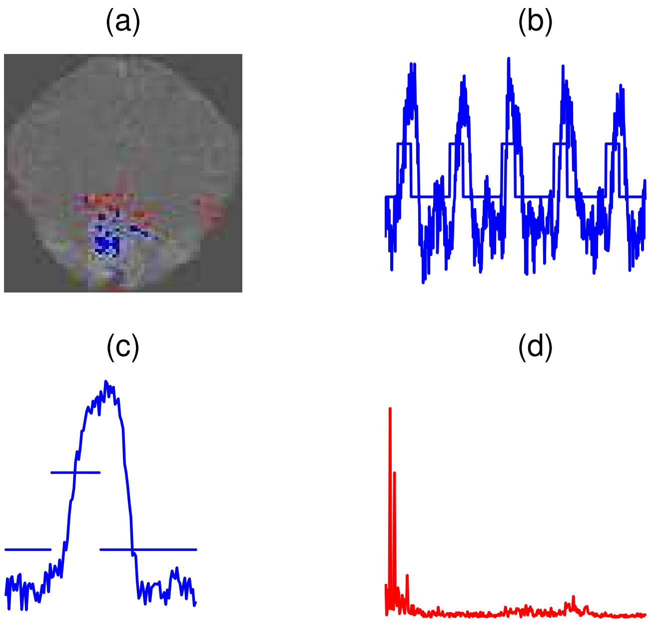


Figure 4.29: Subplots show: the maximum variance factor (b) and the corresponding factor load (spatial representation) (a). (d) is the power spectrum of (b). (c) is the 'design vector' (the factor load for the design matrix).

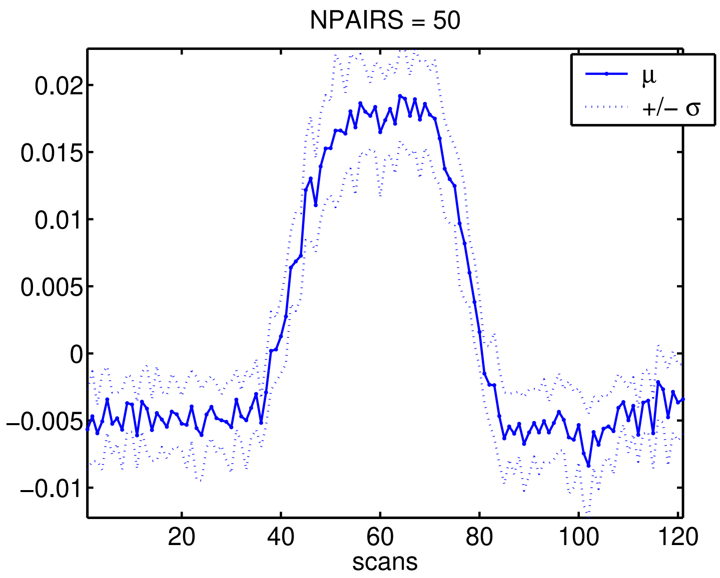


Figure 4.30: The data is split into 50 NPAIRS and for each pair the mean and the difference squared (variance) were calculated. The plot shows the average of the 50 means (μ) and standard deviations (σ). All training sessions did 400 EM-iterations.

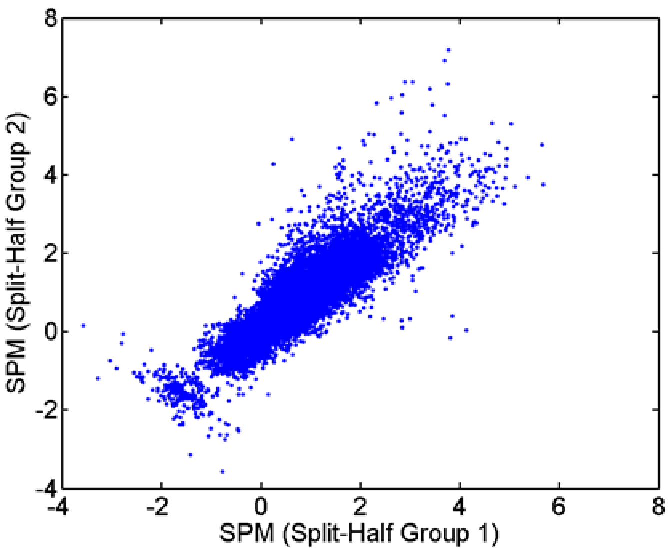


Figure 4.31: Scatter plot of the maximum variance factor loads for all the NPAIR splits.

Chapter 5

Conclusion

This work has been focused on variational Bayes and factor analysis models. Expectation maximization and variational Bayesian algorithms has been derived for factor analysis, extended factor analysis and parallel factor analysis. Furthermore, a model called pPLS was proposed. The variational Bayesian approach has several advantages which have been exploited.
ARD priors made it possible to correctly infer the model order on synthetic data. The lower bound on the marginal likelihood showed to be reliable in a model selection task, but was also found to underestimate the model order compared to BIC. It was shown that the introduction of non-Gaussian prior distributions lead to models with properties similar to ICA. Variational factor analysis provided a remarkable improvement on the generalization performance compared to maximum likelihood factor analysis.
Hyperparameter optimization was investigated for two types of priors. The first type of prior resulted in a more biased model, while the latter showed to cause overfitting. It was noted that only 'noise' parameters were capable of causing overfitting. Thus, hyperparameter optimization was in general not recommendable for the models studied.
On amino acids data the variational PARAFAC model yielded equivalent results to what obtained with ALS. The ARD mechanism did not prune away unneeded factors and it was not possible to infer the model order using the lower bound. In general, many of the problems reported with ALS, like initialization problems and slow convergence was not solved by VBPARAFAC.
A fMRI analysis was performed using the variational algorithms and pPLS. It was not possible to infer the model order with ARD precisely, but results relying on a test error measure suggested $k = 11$. The VBEFA model was capable of separating factors arising from artifacts, but it was often encountered that no common factors were retained. Both VBFA and VBEFA were able to find interesting factors such as cardiac rhythm, respiration and factors correlated with stimuli.

The VBPARAFAC model gave interesting results on the fMRI data. Almost noise free and typical less than five distinct factors were found. The ARD mechanism worked to some extent, but inferring the model order was not possible. The pPLS model was used in a NPAIRS analysis and the result was almost equivalent to what obtained by VBPARAFAC.

Suggestions for further work

The VBPARAFAC and pPLS models have not been compared to their counterparts PARAFAC and PLS. Any possible differences on convergence and model generalization would be interesting to investigate. Another topic is how to interpret the factors found by VBPARAFAC and whether the model is appropriate for analyzing fMRI data. Other kind of priors or maybe structured variational approximations could be investigated in general. One could also look for other data types for which the variational algorithms could be adequate. A preliminary experiment, combining VBPARAFAC with simulated annealing showed to avoid local maxima problems. This scheme might be worth a further study.

Appendix A

Kullback-Leibler divergence

The KL divergence between two distributions $q(x)$ and $p(x)$ is given by

$$\mathrm{KL}(q(x)\|p(x)) = \int q(x) \ln \frac{q(x)}{p(x)} dx \tag{A.1}$$

The KL divergence can be regarded as a distance measure between distributions, and is always non-negative and only zero when $q(x) = p(x)$. As an example, consider two 1-D Gaussians $q(x)$ and $p(x)$ given by the parameters $\theta_q = (\mu_q, \sigma_q^2)$ and $\theta_p = (\mu_p, \sigma_p^2)$. The Gaussian distribution with mean μ and variance σ^2 is

$$p(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) \tag{A.2}$$

The KL divergence between the Gaussians $q(x)$ and $p(x)$ is

$$\begin{aligned}
\mathrm{KL}(q(x)\|p(x)) &= \int q(x) \ln \frac{\frac{1}{\sqrt{2\pi\sigma_q^2}} \exp\left(-\frac{1}{2\sigma_q^2}(x-\mu_q)^2\right)}{\frac{1}{\sqrt{2\pi\sigma_p^2}} \exp\left(-\frac{1}{2\sigma_p^2}(x-\mu_p)^2\right)} dx \\
&= \int q(x) \left(\ln \frac{\sigma_p}{\sigma_q} - \frac{1}{2\sigma_q^2}(x-\mu_q)^2 + \frac{1}{2\sigma_p^2}(x-\mu_p)^2\right) dx \\
&= \ln \frac{\sigma_p}{\sigma_q} - \frac{1}{2} + \frac{1}{2\sigma_p^2} \int (x-\mu_p)^2 q(x) dx \\
&= \ln \frac{\sigma_p}{\sigma_q} - \frac{1}{2} + \frac{1}{2\sigma_p^2} \left[\int x^2 q(x) dx - 2\mu_p \int x q(x) dx + \mu_p^2\right] \\
&= \ln \frac{\sigma_p}{\sigma_q} - \frac{1}{2} + \frac{\sigma_q^2 + \mu_q^2}{2\sigma_p^2} - \frac{\mu_q \mu_p}{\sigma_p^2} + \frac{\mu_p^2}{2\sigma_p^2} \\
&= \ln \frac{\sigma_p}{\sigma_q} + \frac{1}{2\sigma_p^2}\left[(\mu_q - \mu_p)^2 + (\sigma_q^2 - \sigma_p^2)\right] \qquad \text{(A.3)}
\end{aligned}$$

In the above derivation we have used that

$$\mathcal{E}[x^n] = \int_{-\infty}^{\infty} x^n p(x)dx \tag{A.4}$$

$$\sigma^2 = \mathcal{E}[x^2] - (\mathcal{E}[x])^2 = \mathcal{E}[(x-\mu)^2] = \int_{-\infty}^{\infty} (x-\mu)^2 p(x)dx \tag{A.5}$$

where $\mu = \mathcal{E}[x]$. Figure A.1 shows the integrand of the KL divergence as a function of x. Note that $\mathrm{KL}(q(x)\|p(x)) \neq \mathrm{KL}(p(x)\|q(x))$ in the general case. If $\sigma_q = \sigma_p$ then $\mathrm{KL}(q(x)\|p(x)) = \mathrm{KL}(p(x)\|q(x))$ due to the symmetrical properties of the Gaussian distribution.
A specific value of the KL divergence does not tell anything about the shape of the distribution. This is illustrated on figure A.2 for two different scenarios.

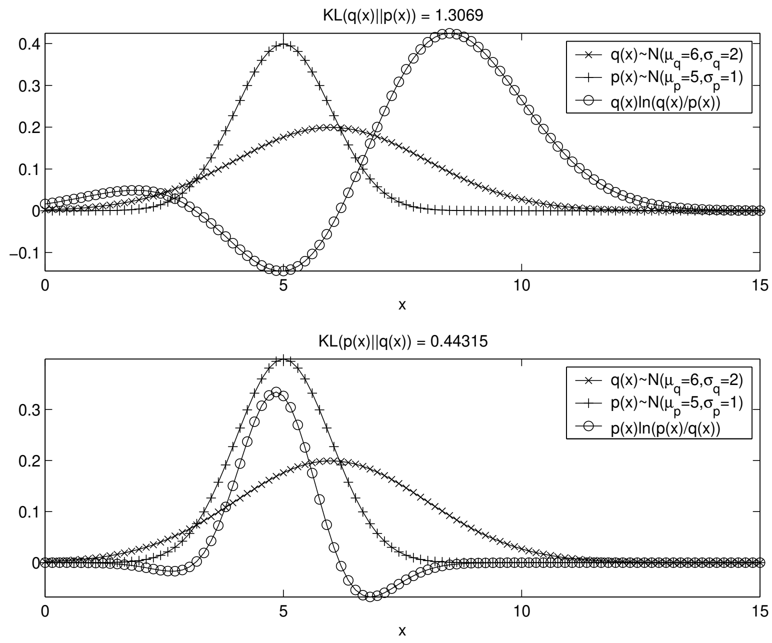


Figure A.1: Integrand $q(x)\ln\frac{q(x)}{p(x)}$ and $p(x)\ln\frac{p(x)}{q(x)}$.

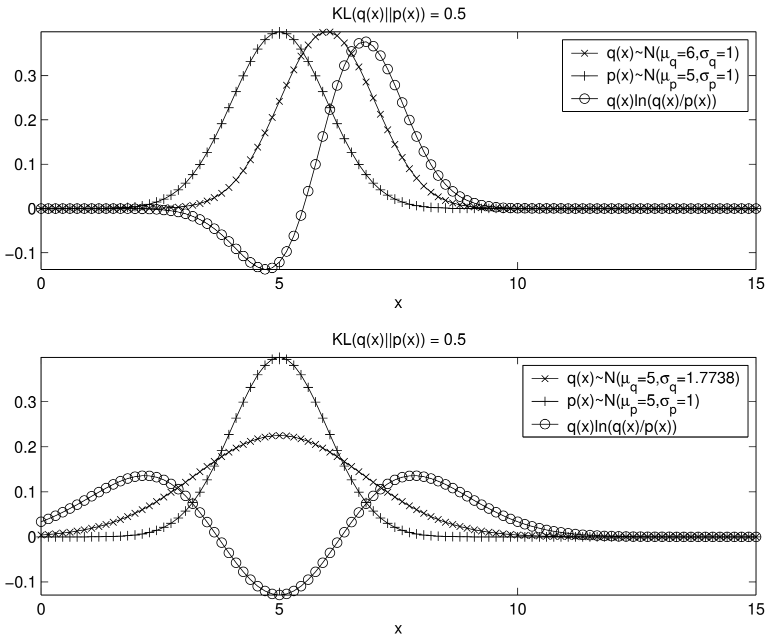


Figure A.2: Two different scenarios where $\mathrm{KL}(q(x)\|p(x)) = 0.5$.

Bibliography

[1] http://dictionary.reference.com/search?q=inference.

[2] Hagai Attias. Inferring parameters and structure of latent variable models by variational bayes. pages 21–30.

[3] Hagai Attias. Independent factor analysis. *Neural Computation*, 11(4):803–851, 1999.

[4] Matthew J. Beal and Zoubin Ghahramani. The variational bayesian em algorithm for incomplete data: with application to scoring graphical model structures. In *Bayesian Statistics 7*. Oxford University Press, 2003.

[5] M.J. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, May 2003.

[6] C. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995.

[7] Rasmus Bro. Multi-way analysis in the food industry - models, algorithms, and applications.

[8] Rasmus Bro. *PARAFAC: Tutorial and applications*. Chemometrics and Intelligent Laboratory Systems, 1997. Data available at: http://www.models.kvl.dk/research/data/.

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

[10] Z. Ghahramani and M. Beal. Variational inference for Bayesian mixture of factor analysers, 1999.

[11] Z. Ghahramani and G. E. Hinton. Parameter estimation for linear dynamical systems. Technical Report CRG-TR-96-2, Department of Computer Science, University of Toronto, 2 1996.

[12] Zoubin Ghahramani and Matthew J. Beal. Propagation algorithms for variational bayesian learning. In *NIPS 13*, pages 507–513, 2001.

[13] Zoubin Ghahramani and Geoffrey E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, 21 1996.

[14] C. Goutte, L. K. Hansen, M. G. Liptrot, and E. Rostrup. Feature-space clustering for fmri meta-analysis. *Human Brain Mapping*, 13:165–183, 2001.

[15] C. Goutte, F. A. Nielsen, and L. K. Hansen. Modelling the heamodynamic response in fmri with smooth FIR filters. *IEEE Transactions on Medical Imaging*, 19(12):1188+, 2000.

[16] L. K. Hansen. Bayesian averaging is well-temperated. In S .S. Solla et al., editor, *Proceedings of NIPS 99, Denver, November 29 - December 4, 1999*, pages 265–271, 2000.

[17] G. Hinton and D. van Camp. Keeping neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual ACM Conference on Computational Learning Theory*, pages 5–13. ACM Press, New York, NY, 1993.

[18] T. Jaakkola. Tutorial on variational approximation methods, 2000.

[19] T. Jaakkola and M. Jordan. Bayesian parameter estimation via variational methods.

[20] Michael I. Jordan, Zoubin Ghahramani, Tommi Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

[21] H. Lappalainen and J. Miskin. Ensemble learning, 2000.

[22] Bishop C. M. Variational principal components. In *ICANN 1*, page 509–514, 1999.

[23] D. MacKay. Ensemble learning and evidence maximization.

[24] D. MacKay. Ensemble learning for hidden markov models, 1997.

[25] D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992.

[26] D. J. C. MacKay. Probable networks and plausible predictions — a review of practical Bayesian methods for supervised neural networks. *Network: Computation in Neural Systems*, 6:469–505, 1995.

[27] David J.C. MacKay. *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press, 2003. available from http://www.inference.phy.cam.ac.uk/mackay/itila/.

[28] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate analysis.* Academic Press, 1979.

[29] T. Minka. Variational bayes for mixture models: Reversing em, 2000.

[30] R. Neal and G. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models.* Kluwer, 1998.

[31] R. M. Neal. Annealed importance sampling. Technical Report 9805, Dept. of Statistics, Univ. of Toronto, 1998.

[32] L. K. Hansen P.A.d.F.R. Hoejen-Soerensen, O. Winther. Mean field approaches to independent component analysis. 2001.

[33] Harshman R.A. Foundations of the parafac procedure: Models and conditions for an 'explanatory' multi-mode factor analysis, 1970.

[34] Carl Edward Rasmussen and Zoubin Ghahramani. Occam's razor.

[35] S.Roweis and Z. Ghahramani. A unifying review of linear Gaussian models. *Neural Computation*, 11:305–345, 1999.

[36] L. Stephen, D. Rottenberg, S. Strother, J. Anderson, S. Muley, J. Ashe, S. Frutiger, K. Rehm, L. K. Hansen, E. Yacoub, and X. Hu. The evaluation of preprocessing choices in single-subject BOLD fmri using NPAIRS performance metrics. *NeuroImage*, 18(1):10–27, 2003.

[37] M. Tipping and C. Bishop. Probabilistic principal component analysis, 1997.