HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Computer Science
and Engineering

Markus Harva

Hierarchical Variance Models of Image Sequences

Master's thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology

Espoo, March 10, 2004

Supervisor: Professor Juha Karhunen
Instructor: Tapani Raiko, M.Sc. (Tech.)

Teknillinen korkeakoulu	Diplomityön tiivistelmä
Tietotekniikan osasto	

Tekijä:	Markus Harva
Osasto:	Tietotekniikan osasto
Pääaine:	Informaatiotekniikka
Sivuaine:	Perustieteiden sivuaine
Työn nimi:	Kuvasekvenssien hierarkkiset varianssimallit
Title in English:	Hierarchical Variance Models of Image Sequences
Professuurin koodi ja nimi:	T-61 Informaatiotekniikka
Työn valvoja:	Prof. Juha Karhunen
Työn ohjaaja:	DI Tapani Raiko

Tiivistelmä:

Ohjaamattomaan oppimiseen perustuvat kuvasekvenssien mallit tuottavat yleensä yksinkertaisia piirteitä kuten reunasuotimia. Nämä yksinkertaiset piirteet eivät tarjoa kovinkaan korkean tason informaatiota kuvasekvenssistä. Yhdistämällä näiden tuottamaa informaatiota on kuitenkin mahdollista irrottaa mielekkäämpiä piirteitä datasta.

Tilastollisten mallien ennustamat arvot ovat yleensä taustalla olevien todennäköisyysjakaumien odotusarvoja. Korkeamman kertaluvun statistiikat jätetään huomiotta. Varianssi kuvaa todennäköisyysjakauman hajontaa sen keskiarvosta. Varianssien estimointi yhdessä odotusarvojen kanssa on hankalaa ja yleensä sitä ei juurikaan tehdä. Kuitenkin on hyvin tiedossa, että monissa datajoukoissa varianssi sisältää paljon informaatiota, jota ei saada irrotettua pelkkiä keskiarvoja mallintamalla.

Tässä työssä oleellinen kysymys on, saavutetaanko varianssien mallintamisella kuvasekvensseissä jotain hyödyllistä tavallisiin malleihin verrattuna. Työssä näytetään, että näin todellakin on ja rakennetaan eräs variansseja hyödyntävä hierarkkinen malli. Myös opetusalgoritmi, mukaanlukien lokaalit päivityssäännöt ja globaalit alustusskeemat, esitellään. Perusmenetelmänä sovelletaan bayesiläistä variaatio-oppimista, joka on osoittautunut luotettavaksi menetelmäksi vaikeidenkin ongelmien ratkaisemiseen. Mallia kokeillaan keinotekoisella datalla, millä pyritään osoittamaan, että opetusalgoritmi toimii. Simulaatiot luonnollisesta näkymästä tuotetulla kuvasekvenssillä osoittavat, että algoritmi toimii myös realistisemmalla datalla.

Sivumäärä: 82	Avainsanat:	varianssimallinnus, hierarkkiset mallit, kuvasekvenssit, bayesiläinen variaatio-oppiminen, Harvakoodaus

Täytetään osastolla

Hyväksytty:	Kirjasto:

Helsinki University of Technology
Department of Computer Science and Engineering

Abstract of Master's thesis

Author:	Markus Harva
Department:	Department of Computer Science and Engineering
Major subject:	Computer and Information Science
Minor subject:	Mathematics
Title:	Hierarchical Variance Models of Image Sequences
Title in Finnish:	Kuvasekvenssien hierarkkiset varianssimallit
Chair:	T-61 Computer and Information Science
Supervisor:	Prof. Juha Karhunen
Instructor:	Tapani Raiko, M.Sc. (Tech.)

Abstract:

Unsupervised learning applied to image sequences ordinarily yields simple features such as edge detectors. These simple features can't provide, as such, very high level information about an image sequence. It is by combining the information they provide that something more meaningful can be extracted from the data.

The values statistical models usually predict are the means of the underlying probability distributions. All the higher order statistics are ignored. The variance describes the deviation of a probability distribution from the mean value. The estimation of variances jointly with means is difficult and consequently not much emphasis is put on it. However, it is very well known that in several datasets the variance conveys a lot of information that is not extractable if one only models means.

The basic question of the thesis is, whether the modelling of variances in image sequences is useful and if something can be gained by doing it. It is shown that this indeed is the case and a specific hierarchical model utilizing variances is constructed step by step. The learning algorithm, including the local update rules and global initialization schemes, is also derived and presented. The basic approach taken is variational Bayesian learning which has proven to be a robust method even in rather hard problems. The model is put to test by performing simulations with artificial data which shows that the learning algorithm works. Simulations with an image sequence from a natural scene show that the algorithm works also with realistic data.

Number of pages: 82 Keywords: variance modelling, hierarchical models, image sequences, variational Bayesian learning, sparse coding

Department fills
Approved: Library code:

Preface

This work has been conducted in the Laboratory of Computer and Information Science at Helsinki University of Technology. It has been funded by the Finnish Center of Excellence Programme under the project New Information Processing Principles.

I would like to express my gratitude to my instructor Tapani Raiko and to my supervisor Professor Juha Karhunen. Also Antti Honkela's technical advice have been most helpful. Additionally, I would like to thank Doctors Jarmo Hurri and Hans van Hateren — the former for providing advice and code for accessing the video database of the latter [56]. Finally I wish to acknowledge Dr. Harri Valpola, whose pioneering work on applying variational Bayesian learning to non-linear and non-Gaussian models has been the essential basis of this thesis.

Otaniemi, March, 2004

Markus Harva

Contents

Abbreviations

cpf	Conditional probability (distribution) function
FA	Factor analysis
HNFA	Hierarchical nonlinear factor analysis
ICA	Independent component analysis
KL	Kullback-Leibler (divergence)
MAP	Maximum a posteriori (solution)
ML	Maximum likelihood (solution)
MLP	Multi-layer perceptron (network)
PCA	Principal component analysis
pdf	Probability density function

Notation

$\mathbf{A}$, $\mathbf{B}$	Matrices
$\mathbf{a}$, $\mathbf{b}$	Vectors
a, b	Scalars
C, C', C''	Generic constants in derivations
$p(x \mid y)$	Probability distribution of x given y
I	The prior beliefs
$\mathcal{C}$	The cost function
$\mathcal{C}_{\mathrm{KL}}$	Kullback-Leibler divergence
$D(q \,\Vert p)$	Kullback-Leibler divergence between distributions q and p
$\langle\cdot\rangle_{q(\theta)}$	Expectation of $\cdot$ over the distribution $q(\theta)$. If the distibution is omitted, it should be clear from the context over which distribution the expectation is taken
$\mathrm{Var}\{\cdot\}$	Variance of $\cdot$
$\mathcal{N}(s \mid m, v)$	Gaussian distribution of s with mean m and variance v
$\mathcal{N}(\mathbf{s} \mid \mathbf{m}, \mathbf{v})$	Multivariate Gaussian distribution of vector $\mathbf{s}$ with mean vector $\mathbf{m}$ and diagonal covariance matrix $\mathrm{diag}(\mathbf{v})$
$\mathcal{N}^R(s \mid m, v)$	Rectified Gaussian distribution of s with parameters m and v
$\mathcal{D}(\mathbf{c} \mid \mathbf{u})$	Dirichlet distribution of $\mathbf{c}$ with prior sample counts $\mathbf{u}$
$\mathcal{E}(s \mid \lambda)$	Exponential distribution of s with the decay parameter λ
z^{-1}	Delay operator

Chapter 1

Introduction

Image sequences are very interesting data to be considered from the unsupervised learning perspective. From the practical point of view the learning of a compact representation of an image sequence can be useful for the purpose of data compression or extraction of features. The features that can be considered to be of interest in the context of image sequences are for example description of movement, presence of a certain kind of object etc.

One characteristic of image sequences is that the relations between consecutive frames are important. Usually the redundancy in two adjacent frames is great, at least on a higher level of abstraction. However, elements such as noise and small camera movements can make the pixel level correlations negligible. Hence, a hierarchical model, where the level of abstraction increases upwards in the hierarchy, could have potential to capture more relevant things from image sequences.

In modelling images and image sequences there are several different possibilities when it comes to choosing the actual data that is fed to the learning algorithm. For some methods the raw pixel data is of too high dimensionality to be used directly. Instead some fixed feature extraction procedure that reduces the dimensionality has to be applied beforehand. In this work, the data is taken to be the raw pixel representation of the image sequence.

1.1 Aim of the thesis

Variance is a concept in statistics. It describes the deviation of a probability distribution from its mean value. It is known, that several phenomena in natural signals can be explained by a Gaussian process whose scale, i.e. the variance, is varied [45].

The aim of this thesis is to study the modelling of variances in image sequences. The basic question is whether variance modelling has some benefits over conventional modelling of means.

Modelling of variances, without any reference to image sequences, is an interesting topic in its own right. Ordinary estimation methods, based on maximizing the likelihood function or posterior density function, can have serious difficulties with models that have free variance parameters, resulting in overlearning or even severe numerical problems. The learning approach taken in this thesis is the variational Bayes, which has proven to be a very robust method for learning many kinds of difficult models including nonlinear factor analysis and nonlinear state-space models.

1.2 Contributions

The work is mainly based on the framework considered in papers of Valpola et al. [55, 47] where a variational Bayesian learning approach based on combining simple blocks have been introduced and applied to some complex problems. The block library has been extended by the author with derivations and implementations of some new blocks. The new blocks are rectified Gaussian variable, mixture of Gaussians variable, discrete variable with Dirichlet and Markov priors, and Dirichlet variable. The learning algorithms, initialization schemes and such and the actual work of constructing the models and doing the simulations were also done by the author.

1.3 Structure

The structure of this thesis is the following.

In Chapter 2, the background of Bayesian inference is covered. The fundamental theory of probabilities and its justification are reviewed and some practical points of constructing probabilistic models are considered. Before introducing the variational Bayesian learning, a brief overview of standard estimation methods is given.

The central part of the thesis is Chapter 3 where both variance modelling and image sequence models are considered first separately and then in conjunction. Some of the earlier work on both subjects is reviewed and then some general motivation of the methods and models of this thesis are given.

Bayes Blocks, the theoretical framework and the software library used in this thesis, is the subject of Chapter 4. There the key principles and practical issued are discussed.

The concrete model serving as an example of modelling variances in image sequences is constructed step by step in Chapter 5. The experimental results guide the development which ends in a rather complex model. The learning algorithms, which have proven to be essential and continue do so, are also considered.

There are two appendices. The first, Appendix A, contains a summary of the probability distributions needed in this thesis. The derivations of the cost functions and update rules for the new blocks are given in Appendix B.

Chapter 2

Bayesian Inference

Bayesian inference is a methodology based on Bayesian probability theory. Although Bayes' rule, the most essential part of Bayesian inference, has been known for several centuries, the firm theoretical ground was developed only in the 1940s. Practical usage of Bayesian methods has gained more popularity recently due to better computational resources and development of better approximating methods making Bayesian inference tractable even in very complex problems.

2.1 Bayesian probability theory

The entire Bayesian probability theory can be derived from a set of simple intuitive axioms concerning rational and consistent reasoning. These axioms were introduced by Cox [9, 10].

One characteristic of Bayesian probability theory is that all probabilities are conditional or subjective. Hence the probability of a proposition A, $p(A)$, without reference to any context, is not meaningful. Only with a given *prior information* I, is the probability $p(A \mid I)$ reasonable.

The fundamental rules of probability theory derivable from Cox's axioms are the product rule and the sum rule. Given three propositions A, B, I the product rule states that

$$p(AB \mid I) = p(A \mid BI)p(B \mid I)$$

where AB means the logical and of propositions A and B. The sum rule says that

$$p(A \mid I) + p(\neg A \mid I) = 1$$

where $\neg A$ is the logical negation of proposition A. These rules extend straightforwardly to continuous variables. From the product rule one can derive the most import tool of Bayesian inference, the Bayes' rule:

$$p(A \mid BI) = \frac{p(B \mid AI)p(A \mid I)}{p(B \mid I)} \tag{2.1}$$

If we have n propositions, $A_1, A_2, \ldots, A_n$, which are exclusive and exhaustive meaning that $p(A_i A_j \mid I) = 0$, $\forall i \neq j$ and $\sum_{i=1}^{n} p(A_i \mid I) = 1$, we can apply another important tool, the marginalization principle:

$$p(B \mid I) = \sum_{i=1}^{n} p(B \mid A_i\, I)p(A_i \mid I) \tag{2.2}$$

Now, considering the proposition B as the data that we observe, and the probabilities $p(B \mid A_i, I)$ as the model for the data, we can immediately see the significance of (2.1) and (2.2). Using them, we can turn our prior beliefs, $p(A_i \mid I)$, to posterior probabilities $p(A_i \mid B, I)$ where the information that we got from the data B has been included.

Apart from Cox's book [10], a good account on fundamental issues of Bayesian inference can be found in [31]. More practically oriented approach on applying Bayesian theory can be found in [14, 6].

2.2 Constructing probabilistic models

To make inferences in the Bayesian framework about some parameters $\boldsymbol{\theta}$ based on observations $\mathbf{X}$, we need a model for the data, $p(\mathbf{X} \mid \boldsymbol{\theta}, I)$, and a prior for the parameters, $p(\boldsymbol{\theta} \mid I)$. Then we can obtain the posterior distribution of the parameters by an application of Bayes' theorem. In the following subsections the construction of models and choosing of priors are considered.

2.2.1 Hierarchical models

Complex models are usually constructed hierarchically specifying the model as a product of many simpler models. Consider a model where the data depends only on a subset of variables $\boldsymbol{\theta}$, say $\boldsymbol{\theta}_1$, and these variables in turn depend on another subset of variables, say $\boldsymbol{\theta}_2$. We can continue this construction until finally the pdf of variables $\boldsymbol{\theta}_n$ depends only on the model I. Then the joint distribution $p(\mathbf{X}, \boldsymbol{\theta} \mid I)$, needed in the computation of the posterior probability distribution of $\boldsymbol{\theta}$, factors into $n+1$ terms. Denoting $\boldsymbol{\theta}_{-k} = \boldsymbol{\theta} \setminus \bigcup_{i=1}^{k} \boldsymbol{\theta}_i$

$$\begin{aligned}
p(\mathbf{X}, \boldsymbol{\theta} \mid I) &= p(\mathbf{X} \mid \boldsymbol{\theta}, I) p(\boldsymbol{\theta} \mid I) \\
&= p(\mathbf{X} \mid \boldsymbol{\theta}_1, I) p(\boldsymbol{\theta}_1 \mid \boldsymbol{\theta}_{-1}, I) p(\boldsymbol{\theta}_{-1} \mid I) \\
&= p(\mathbf{X} \mid \boldsymbol{\theta}_1, I) p(\boldsymbol{\theta}_1 \mid \boldsymbol{\theta}_2, I) p(\boldsymbol{\theta}_2 \mid \boldsymbol{\theta}_{-2}, I) p(\boldsymbol{\theta}_{-2} \mid I) \\
&\;\;\vdots \\
&= p(\mathbf{X} \mid \boldsymbol{\theta}_1, I) p(\boldsymbol{\theta}_1 \mid \boldsymbol{\theta}_2, I) p(\boldsymbol{\theta}_2 \mid \boldsymbol{\theta}_3, I) \cdots p(\boldsymbol{\theta}_{n-1} \mid \boldsymbol{\theta}_n, I) p(\boldsymbol{\theta}_n \mid I)
\end{aligned}$$

It will be seen in Section 2.4 that this factoring of the joint pdf plays a major role in connection with variational Bayesian learning.

2.2.2 Conjugate priors

In conjunction with hierarchical models, it is typical to use *conjugate priors* [14]. Looking at the Bayes rule,

$$p(\theta \mid x, I) \propto p(x \mid \theta, I) p(\theta \mid I) \tag{2.3}$$

it can be seen that the posterior is essentially the product of the likelihood function and the prior of θ. When this product, i.e. the posterior of θ, has the same functional form as the prior, it said that prior is conjugate to the likelihood. For example the likelihood function of the mean parameter of a normal distribution has the normal distribution as its conjugate prior. Consider the following example

$$\begin{aligned}
p(x \mid \theta, I) &= \mathcal{N}\left(x \mid \theta, \sigma_x^2\right) \\
p(\theta \mid I) &= \mathcal{N}\left(\theta \mid \mu_\theta, \sigma_\theta^2\right)
\end{aligned}$$

Here the posterior is $p(\theta \mid x, I) = \mathcal{N}(\theta \mid \mu, \sigma^2)$, with

$$\sigma^2 = \left(1/\sigma_x^2 + 1/\sigma_\theta^2\right)^{-1} \qquad \text{and} \qquad \mu = \sigma^2 \left(x/\sigma_x^2 + \mu_\theta/\sigma_\theta^2\right) \tag{2.4}$$

The result is obtained by directly expanding the product in Eq. (2.3) and combining the quadratic forms inside the exponentials. Conjugate priors play a major role also in variational learning, see Appendix B for several examples.

2.3 Standard methods for posterior inference

Computing density values from the unnormalized posterior distribution is easy. What is difficult, is summarizing the posterior. As an example we can consider the problem of finding the predictive distribution of a new observation $\mathbf{x}$ given the observed data $\mathbf{X}$. This is done by integrating over the posterior distribution

$$p(\mathbf{x} \mid \mathbf{X}, I) = \int p(\mathbf{x}, \boldsymbol{\theta} \mid \mathbf{X}, I)\, d\boldsymbol{\theta} = \int p(\mathbf{x} \mid \boldsymbol{\theta}, I) p(\boldsymbol{\theta} \mid \mathbf{X}, I)\, d\boldsymbol{\theta} \tag{2.5}$$

Calculating this integral is intractable in all but the simplest cases. In fact we don't usually even have the evidence term $p(\mathbf{X} \mid I)$ which is required for the evaluation of $p(\boldsymbol{\theta} \mid \mathbf{X}, I)$. The thing that we learn from (2.5) is that most emphasis is put on the values of $\boldsymbol{\theta}$ which are in the region of high probability *mass*. Hence, if the distribution $p(\boldsymbol{\theta} \mid \mathbf{X}, I)$ is approximated in one way or another, by replacing it with a single point for example, the approximation must be such, that it concentrates on the areas of high probability mass.

In the following subsections, the standard methods for posterior inference are reviewed.

2.3.1 Methods based on point estimates

A point estimate is, as the name implies, one point from the posterior distribution (or from the likelihood) function which should preferably be a good summary of the whole distribution. The methods belonging to this category include maximum likelihood, maximum a posterior and Laplace's method.

ML

The maximum likelihood estimation aims at finding a point $\boldsymbol{\theta}$ which maximizes the likelihood function $p(\mathbf{X} \mid \boldsymbol{\theta}, I)$. Instead of maximizing the likelihood itself, the usual approach is to find the maximum of the logarithmic likelihood $\log p(\mathbf{X} \mid \boldsymbol{\theta}, I)$. If the likelihood has a factorial form, as it usually does, taking the logarithm splits it into a sum of simple terms.

MAP

The maximum a posterior estimation is essentially the same as ML estimation with the exception that prior pdf, $p(\boldsymbol{\theta} \mid I)$, is now also included. If we have only very vague prior information about $\boldsymbol{\theta}$, meaning that the prior pdf is essentially a constant in the region of interest, the MAP and ML yield the same results.

Laplace

Laplace's method is based on the idea of fitting a simple distribution to the mode of the posterior pdf. If we perform a second order Taylor series expansion of the log posterior at the mode, it turns out that the optimal approximating distribution is a Gaussian, with the mode as its mean and the variance computed from the second derivative of the posterior.[1] The variance can then be interpreted as the inaccuracy of the point estimate. Also the normalizing constant of the true posterior can be approximated by using the normalizing constant of the Gaussian approximation [40].

Remarks about point estimation

Although successful in many simple models, point estimates can be troublesome in more complex models. This is due to the fact that high probability density doesn't necessarily imply high probability mass. The distribution $p(x, y \mid I)$ in Figure 2.1 serves as a schematic illustration. There there are two spikes, a narrow and a wide one. The narrow one is somewhat higher than the wide one and hence point estimation method would yield an estimate from that spike. However, the wider spike (or more appropriately bump) contains approximately ten times more probability mass than the narrow one. Hence, if we were to draw samples from that distribution only one tenth of them would be from the narrow one, although our point estimation method prefers it.

This problem is considered in Section 3.1.1 in a concrete situation where it arises when one is trying to jointly model both means and variances.

2.3.2 Sampling methods

A way to approximate an integral of the form

$$\langle f(\boldsymbol{\theta}) \rangle = \int f(\boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathbf{X}, I)\, d\boldsymbol{\theta}$$

[1]The mode is the MAP-solution.

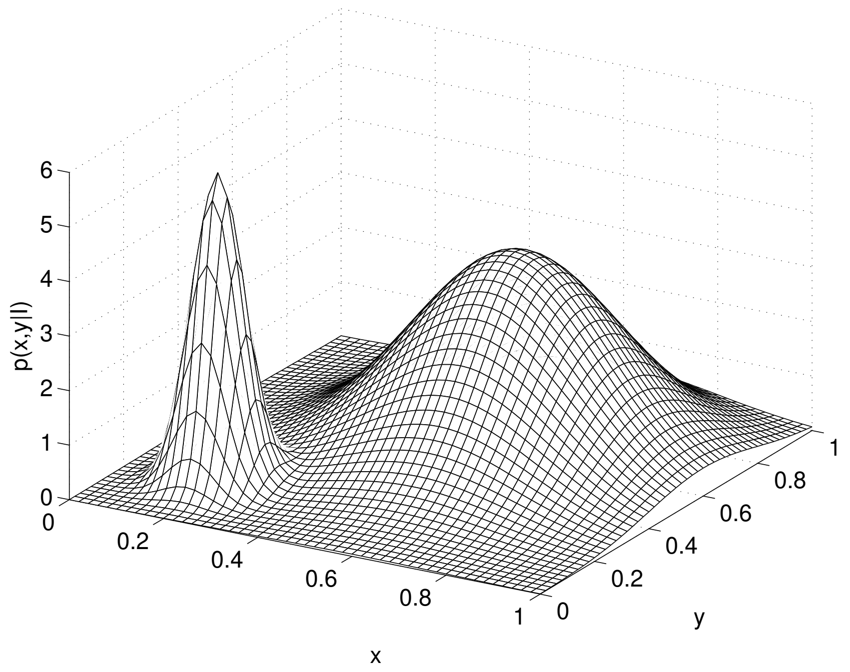


Figure 2.1: The probability distribution $p(x, y \mid I)$ containing two peaks.

is to generate independent samples, $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_N$, from the distribution $p(\boldsymbol{\theta} \mid \mathbf{X}, I)$ and to compute the average

$$\langle f(\boldsymbol{\theta}) \rangle \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{\theta}_i)$$

We would often like to compute these kinds of integrals to summarize our posterior distribution. Hence, the problem of generating independent samples from a complex distribution is a very relevant problem to estimation.

For the actual generation of the samples, there exists numerous ways. Elementary distributions such as uniform and Gaussian are easily sampled but arbitrary distributions require special means. Most popular approaches are the Markov chain Monte Carlo (MCMC) methods, originally due to Metropolis [41] and Hastings [16]. Using these methods one generates a chain of samples starting from some random point. The next sample candidate is chosen according to a jumping distribution. The candidate is selected or rejected based on the true (unnormalized) posterior in a way that favors samples from the area of high probability density. It can be shown, that with rather general assumptions, the distribution of the samples will eventually converge to the true posterior.

One of the problems with MCMC methods is that there is no fool proof way to know whether the sequence has converged or not. It may require a vast amount of samples before it happens. Hence MCMC methods and sampling methods in general are not suitable for very large models containing even millions of parameters to be estimated.

Another problem of sampling is related to symmetries in models. Considering the ICA model [26], $\mathbf{x} = \mathbf{A}\mathbf{s}$, we see that by changing the signs of $\mathbf{A}$ and $\mathbf{s}$ we should have an equally good solution. Now, if our sampling method has worked and produced samples from both of these solutions the mean is zero for both $\mathbf{A}$ and $\mathbf{s}$.

2.4 Variational learning

The key idea in variational Bayesian learning is to approximate the true posterior, $p(\boldsymbol{\theta} \mid \mathbf{X}, I)$, with another distribution, $q(\boldsymbol{\theta})$, having simpler functional form. One of the variants of variational learning, *ensemble learning*, was introduced in [18]. It can be seen as a special case of variational free energy minimization of Feynman and Bogoliubov [12]. In ensemble learning, the misfit between the approximating pdf and the true posterior pdf is measured using Kullback-Leibler divergence

$$\mathcal{C}_{\mathrm{KL}} = D\left(q(\boldsymbol{\theta}) \,\|p(\boldsymbol{\theta} \mid \mathbf{X}, I)\right) = \left\langle \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta} \mid \mathbf{X}, I)} \right\rangle_{q(\boldsymbol{\theta})} \tag{2.6}$$

By defining the actual cost function to be used in the learning as

$$\mathcal{C} = \mathcal{C}_{\mathrm{KL}} - \log p(\mathbf{X} \mid I) = \left\langle \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta}, \mathbf{X} \mid I)} \right\rangle_{q(\boldsymbol{\theta})} \tag{2.7}$$

we get rid of the intractable evidence term $p(\mathbf{X} \mid I)$, which is a constant w.r.t $q(\boldsymbol{\theta})$ anyway. The KL divergence is always non-negative, which enables us to compute a lower bound for the evidence term based on cost function:

$$\mathcal{C} = \mathcal{C}_{\mathrm{KL}} - \log p(\mathbf{X} \mid I) \geq -\log p(\mathbf{X} \mid I) \implies p(\mathbf{X} \mid I) \geq \exp(-\mathcal{C})$$

Hence, by minimizing the cost function, we maximize a lower bound for the evidence.

Looking at the definition of the cost function, one might wonder how tractable it is after all. The tractability comes from the form of the approximating distribution $q(\boldsymbol{\theta})$ which is usually assumed to be a product of the form

$$q(\boldsymbol{\theta}) = \prod_{i=1}^{N} q(\boldsymbol{\theta}_i) \tag{2.8}$$

In other words, we are assuming that the subsets of variables $\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ are posteriorly independent. The most extreme case of this, is the assumption that all the variables are posteriorly independent of each other. In conjunction with a suitably factorial model $p(\boldsymbol{\theta}, \mathbf{X} \mid I)$, the cost function splits to a set of separate terms which can be optimized one at a time using a cyclic update scheme. The straightforward approach is to just update each variable in its turn. But it is possible to greatly accelerate the learning by observing the individual updates and doing line search in that direction in the parameter space [20, 23].

We can further narrow the set of admissible approximating distributions by constraining the form of the factors $q(\boldsymbol{\theta}_i)$ to be for example Gaussians. When a restriction like this is applied, we are searching a fixed form solution. Without any restriction for the form of the factors the solution is said to be free form. These two different alternatives are considered in the following two subsections. For accounts on ensemble learning in general, see e.g. [38, 36].

The variational learning has several good aspects. Firstly, it is sensitive to probability mass instead of probability density. Therefore most problems that arise when using point estimation are avoided in the variational framework. Secondly, the method is not considerably computationally more demanding than point estimation and it is certainly much more efficient than sampling and hence applicable to problems of rather large cardinality. Accordingly ensemble learning has been successfully applied to large variety of different models, including linear ICA models [1, 34, 2, 42, 46], nonlinear ICA [35], nonlinear dynamic state-space models [53], switching state-space models [15] and multi-layer networks [3] among others.

Variational learning has also some drawbacks. As any complex learning method, it is susceptible to local minima. The update is usually done for the distributions of a small subset of variables at time keeping the distributions of the other variables fixed. Hence it is guaranteed that the cost function decreases at every step, and if the cost function for the given model and data is bounded below (it doesn't unfortunately have to be), the learning converges at some stable point. In complex models it is almost certain that it is a non-global minimum — whether it is a good or a bad one, is mostly up to the learning algorithm. In addition to local minima, sometimes even the global optimum represents a solution that is not appropriate. For example Ilin and Valpola [30] showed that when a fully factorial approximating distribution is used in learning an ICA model, the cost function favors PCA type solutions.

2.4.1 Fixed form solutions

Although usually the free form solution is easy to derive, the form of it might not be desirable. One example of this is the posterior of a variance parameter of a Gaussian variable. Parameterizing the variance on the log scale means that likelihood function of the variance is of the form

$$\exp\left\{\tfrac{1}{2}v - C\exp(v)\right\}$$

where C is a positive constant. Combining this with a Gaussian prior for v yields a posterior which is not of any convenient form. It can still be rather well approximated with a Gaussian so we restrict the form of the approximation $q(v)$ to be Gaussian.

2.4.2 Free form solutions

As opposed to fixed form solutions, now the posterior factors, $q(\boldsymbol{\theta}_i)$, in (2.8), are not assumed to have any fixed form such as Gaussian. Instead they are optimized to fit the real posterior using variational methods, whence the name variational Bayesian learning.

Calculus of variations

The optimization problems to be solved in the variational framework are of the form:

$$\begin{cases} \text{minimize } C(q) = \displaystyle\int f(q(\boldsymbol{\theta}), \boldsymbol{\theta}) d\boldsymbol{\theta} \\ \text{subject to } \displaystyle\int q(\boldsymbol{\theta}) d\boldsymbol{\theta} = 1 \text{ and } q(\boldsymbol{\theta}) \geq 0 \, \forall \boldsymbol{\theta} \end{cases}$$

Taking the equality constraint, $\int q(\boldsymbol{\theta}) d\boldsymbol{\theta} = 1$, into account leads to the following Lagrangian

$$L(q, \lambda) = \int [f(q(\boldsymbol{\theta}), \boldsymbol{\theta}) + \lambda q(\boldsymbol{\theta})] d\boldsymbol{\theta} - \lambda$$

We can neglect the inequality constraint and find the optimum in the larger set. If the optimum of the larger set falls to the admissible set, it is of course the optimum of the admissible set too.

The stable points and concurrently the solution to the problem can be found by setting the Fréchet differential [37], $\delta L(q, \lambda; h)$, to zero $\forall h$.[2]

For L the Fréchet differential is

$$
\begin{aligned}
\delta L(q, \lambda; h) &= \left\{ \frac{d}{d\alpha} L(q + \alpha h, \lambda) \right\}_{\alpha=0} \\
&= \left\{ \frac{d}{d\alpha} \left[\int [f(q(\boldsymbol{\theta}) + \alpha h(\boldsymbol{\theta}), \boldsymbol{\theta}) + \lambda(q(\boldsymbol{\theta}) + \alpha h(\boldsymbol{\theta}))] d\boldsymbol{\theta} - \lambda \right] \right\}_{\alpha=0} \\
&= \int \left\{ \frac{d}{d\alpha} [f(q(\boldsymbol{\theta}) + \alpha h(\boldsymbol{\theta}), \boldsymbol{\theta}) + \lambda(q(\boldsymbol{\theta}) + \alpha h(\boldsymbol{\theta})] \right\}_{\alpha=0} d\boldsymbol{\theta} \\
&= \int \left\{ [f_q(q(\boldsymbol{\theta}) + \alpha h(\boldsymbol{\theta}), \boldsymbol{\theta}) h(\boldsymbol{\theta}) + \lambda h(\boldsymbol{\theta})] \right\}_{\alpha=0} d\boldsymbol{\theta} \\
&= \int [f_q(q(\boldsymbol{\theta}), \boldsymbol{\theta}) h(\boldsymbol{\theta}) + \lambda h(\boldsymbol{\theta})] d\boldsymbol{\theta} \\
&= \int [f_q(q(\boldsymbol{\theta}), \boldsymbol{\theta}) + \lambda] h(\boldsymbol{\theta}) d\boldsymbol{\theta}
\end{aligned}
$$

where f_q denotes the derivative w.r.t. the first variable of f i.e. $f_q = D_1 f$. From the requirement that $\delta L(q, \lambda; h)$ should be zero for all h it follows that

$$
f_q(q(\boldsymbol{\theta}), \boldsymbol{\theta}) + \lambda \equiv 0 \tag{2.9}
$$

Gibbs inequality

Gibbs inequality is an assertion about Kullback-Leibler divergence. It states that for any distributions p and q the KL-divergence $D(p \| q) \geq 0$ and the equality is obtained if and only if $p = q$. This provides a powerful way for deriving update rules.

Application

As a demonstration we solve a problem using both the solution derived using variational arguments (Eq. 2.9) and the Gibbs inequality. The solutions are the same, of course.

The model to be considered is shown graphically in Figure 2.2. The exact model equations do not matter at this point. When concerned about updating

[2]Here h is a function belonging to the same vector space as the function q.

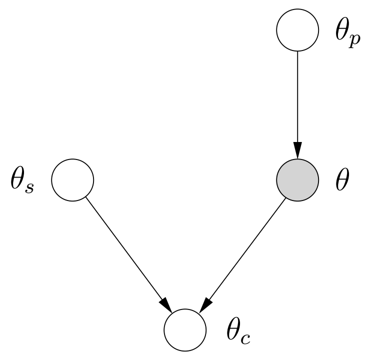


Figure 2.2: The example model. We are considering the updating of the variable θ which is conditioned on θ_p. The variable θ_c is conditioned on the variables θ_s and θ.

the variable θ in the model, the relevant part of the cost function (2.7) is

$$\mathcal{C}_\theta = \left\langle \log \frac{q(\theta)}{p(\theta_c \mid \theta, \theta_s) p(\theta \mid \theta_p)} \right\rangle_{q(\theta, \theta_c, \theta_s, \theta_p)} \tag{2.10}$$

Let us assume a factorial posterior approximation such that

$$q(\theta, \theta_c, \theta_s, \theta_p) = q(\theta) q(\theta_c, \theta_s, \theta_p)$$

While we are updating $q(\theta)$, the distribution over other variables, in this case $q(\theta_c, \theta_s, \theta_p)$, is kept fixed. We can manipulate (2.10)

$$\begin{aligned} \mathcal{C}_\theta &= \left\langle \log q(\theta) - \langle \log[p(\theta_c \mid \theta, \theta_s) p(\theta \mid \theta_p)] \rangle_{q(\theta_c, \theta_s, \theta_p)} \right\rangle_{q(\theta)} \\ &= \left\langle \log \frac{q(\theta)}{\exp \langle \log[p(\theta_c \mid \theta, \theta_s) p(\theta \mid \theta_p)] \rangle_{q(\theta_c, \theta_s, \theta_p)}} \right\rangle_{q(\theta)} = D\left(q \,\|\, h\right) \end{aligned}$$

where it is denoted

$$h(\theta) = \exp \langle \log[p(\theta_c \mid \theta, \theta_s) p(\theta \mid \theta_p)] \rangle_{q(\theta_c, \theta_s, \theta_p)}$$

Now, proceeding with the variational approach we see that

$$D\left(q \,\|\, h\right) = \int q(\theta) \log \frac{q(\theta)}{h(\theta)} d\theta$$

where $f(q(\theta), \theta)$ of (2.9) can be identified as

$$f(q(\theta), \theta) = q(\theta) \log \frac{q(\theta)}{h(\theta)}$$

Now substituting f to (2.9) we get

$$\begin{aligned} &\log \frac{q(\theta)}{h(\theta)} + q(\theta) \frac{h(\theta)}{q(\theta)} \frac{1}{h(\theta)} + \lambda \equiv 0 \implies \log \frac{q(\theta)}{h(\theta)} + 1 + \lambda \equiv 0 \\ &\qquad \implies \log q(\theta) \equiv -\lambda - 1 + \log h(\theta) \implies q(\theta) = \exp(-\lambda - 1) h(\theta) \end{aligned}$$

The value of the Lagrange coefficient doesn't matter, it just scales the distribution properly. Hence, the solution is

$$q(\theta) \propto h(\theta)$$

On the other hand, we could have directly invoked Gibbs' inequality for $D\left(q \,\|\, h\right)$ and obtained exactly the same result.

Chapter 3

Variance Modelling in Image Sequences

3.1 Modelling variances

The modelling of variances can be understood to mean several things. One common problem is to estimate some single variance parameter which controls the noise level of the data. The more interesting case, and the one relevant to this thesis, is when the variance is allowed to vary over the time course. For example, if we assume a generative model for the observations, there is usually a time independent parameter that controls the amount of noise. This assumption of constant noise level might be wrong, however. It might be the case that there is more noise at certain instances of time than at others. One very good example, where the variance itself is of interest, is financial data. The economist even have their own term for the variance, *volatility*, which in essence describes how stable the stock price, or another similar quantity, is.

Estimation of variances, when done jointly with estimation of means, can be difficult. In the next sections, an example problem is considered and it is tried to be solved using first standard methods and then applying a variational method which is used later on in this thesis.

3.1.1 The basic problem

The basic problem with standard estimation methods when it comes to variances, is overlearning. Those methods usually give overconfident results about noise levels or at worst, fail completely. This problem is demonstrated below.

Let us consider the simple example of estimating a one dimensional simplified factor analysis model summarized by the following set of equations

$$p(x \mid s, v, I) = \mathcal{N}\left(x \mid s, \exp(-v)\right) \tag{3.1}$$

$$p(s \mid I) = \mathcal{N}\left(s \mid 0, 1\right) \tag{3.2}$$

$$p(v \mid I) = \mathcal{N}\left(v \mid 0, \sigma_v^2\right) \tag{3.3}$$

The two parameters to be estimated here are the signal s and the noise (negative) log-variance v.

As a first approach let us ignore the priors and just try maximizing the likelihood, $p(x \mid s, v, I)$. Writing it explicitly out, we get

$$\begin{aligned} p(x \mid s, v, I) &= \frac{1}{\sqrt{2\pi \exp(-v)}} \exp\left\{-\frac{1}{2\exp(-v)}(x-s)^2\right\} \\ &\propto \exp\left\{-\tfrac{1}{2}(x-s)^2 \exp(v) + \tfrac{1}{2}v\right\} \end{aligned}$$

Now setting $s := x$ this simplifies to $\exp(\frac{1}{2}v)$ which approaches infinity as we let $v \to \infty$. From these results given by ML estimation we conclude that the noise variance is zero and the observation is generated by replicating the source, no matter what the value of the observed variable x is. This is clearly a really bad case of overlearning given that we had only one observation to make these conclusions from. Another problem here is that, if we had written a numerical optimization algorithm for this estimation procedure, it would have crashed due to the infinite v.

As a second attempt we can include our prior probability distributions to our estimation. The joint unnormalized posterior of s and v is readily obtained by an application of Bayes theorem

$$\begin{aligned} p(s, v \mid x, I) &\propto p(x \mid s, v, I) p(s \mid I) p(v \mid I) \\ &= \mathcal{N}\left(x \mid s, \exp(-v)\right) \mathcal{N}\left(s \mid 0, 1\right) \mathcal{N}\left(v \mid 0, \sigma_v^2\right) \\ &\propto \exp\left\{-\tfrac{1}{2}(x-s)^2 \exp(v) + \tfrac{1}{2}v - \tfrac{1}{2}s^2 - v^2/2\sigma_v^2\right\} \end{aligned} \tag{3.4}$$

Having observed $x = 1$ and chosen the variance for the prior of v to be $\sigma_v^2 = 5^2$, we can plot the contours of this function as in Figure 3.1. Looking at the figure, it is clear that the optimum s is very near 1. Substituting $s := 1 = x$ to Eq. (3.4) we are left with the expression $\exp\{\frac{1}{2}v - v^2/2\sigma_v^2\}$, which is optimized with $v = \sigma_v^2/2$. With $\sigma_v^2 = 25$ it follows that the MAP solution for v is $v^* = 12.5$. This results is certainly better than the one obtained with ML, but it is still rather overconfident. It is also clear that the point $(s^*, v^*) = (1, 12\frac{1}{2})$ is not a very good summary of the whole distribution as there is very little probability mass in the neighborhood of that point.

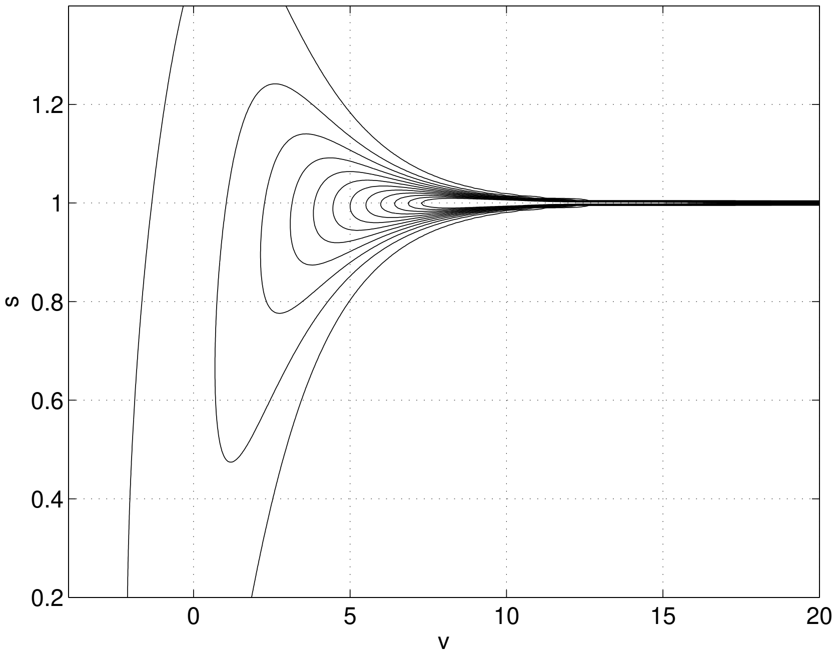


Figure 3.1: The joint posterior probability distribution $p(s, v \mid x, I)$.

3.1.2 The variational approach

It was demonstrated in the previous section that point estimation does not give satisfactory results (if any at all) when jointly estimating both means and variances. The basic problem is that high probability density does not imply high probability mass. Hence a method that is sensitive to probability mass could be hoped to produce better results.

Let us still consider the model specified by Equations (3.1), (3.2) and (3.3) and let us try to estimate the parameters using variational Bayesian learning. First we have to choose the form of our approximating posterior distribution q. We can start with a factorial approximation $q(s, v) = q(s)q(v)$ where the form of the factors is not restricted. The relevant part of the cost function (2.7) can be written as

$$\int q(s, v) \log \frac{q(v)}{p(x \mid s, v, I) p(v \mid I)} \, ds \, dv = \int q(v) \log \frac{q(v)}{p(v \mid I)} \int q(s) (-\log p(x \mid s, v, I)) \, ds \, dv \quad (3.5)$$

The expectation of $-\log p(x \mid s, v, I)$ over $q(s)$ can be evaluated to yield

$$
\begin{aligned}
\int q(s)(-\log p(x \mid s, v, I))\, ds &= -\int q(s)[\tfrac{1}{2}v - \tfrac{1}{2}\exp(v)(s-x)^2]\, ds + C \\
&= -\tfrac{1}{2}v + \exp(v) \int q(s)(s-x)^2\, ds + C \\
&= -\tfrac{1}{2}v + \exp(v)\big[(\langle s \rangle - x)^2 + \mathrm{Var}\{s\}\big] + C
\end{aligned}
$$

Defining

$$h(v) = \exp\left\{\tfrac{1}{2}v - \exp(v)\big[(\langle s \rangle - x)^2 + \mathrm{Var}\{s\}\big]\right\}$$

we see that Eq. (3.5) yields

$$\int q(v) \log \frac{q(v)}{h(v)\, p(v \mid I)}\, dv = D\left(q(v)\, \| h(v)\, p(v \mid I)\right)$$

Now, by Gibbs inequality, the cost function is optimized by setting $q(v) \propto h(v)p(v \mid I)$. Unfortunately the optimal free form unnormalized $q(v)$ is of such an inconvenient form that we cannot compute any expectations over it or even normalize it for that matter. In this case the only viable solution is to use a fixed form approximation of some convenient form. Looking at the Figure 3.2, it can be seen that Gaussian fixed form solution is an acceptable compromise. The more concentrated the prior for v is, the closer the fixed form solution will be to the free form one.

By fixing the form $q(v)$ to be Gaussian, $q(v) = \mathcal{N}(v \mid \bar{v}, \tilde{v})$, the variational minimization problem of Eq. (3.5) is converted into an optimization problem of two real variables, the mean $\bar{v}$ and the variance $\tilde{v}$ of $q(v)$. Although the cost function can be written in closed form in the terms of $\bar{v}$ and $\tilde{v}$, the optimal values cannot be solved analytically. However a fast algorithm for numerical optimization, based on Newtons method and fixed point iteration, has been derived by Valpola [50, 51].

Using this approach we can finally solve the example problem. The optimal approximating distribution is

$$q(s, v) = q(s)q(v) = \mathcal{N}(s \mid 0.80, 0.20)\, \mathcal{N}(v \mid 0.41, 1.90)$$

In Figure 3.3, this approximating posterior is plotted over the true one.

3.1.3 The variance neuron

It might seem that the example model of the previous sections and the variational solution to it have very little relevance to real world problems, where

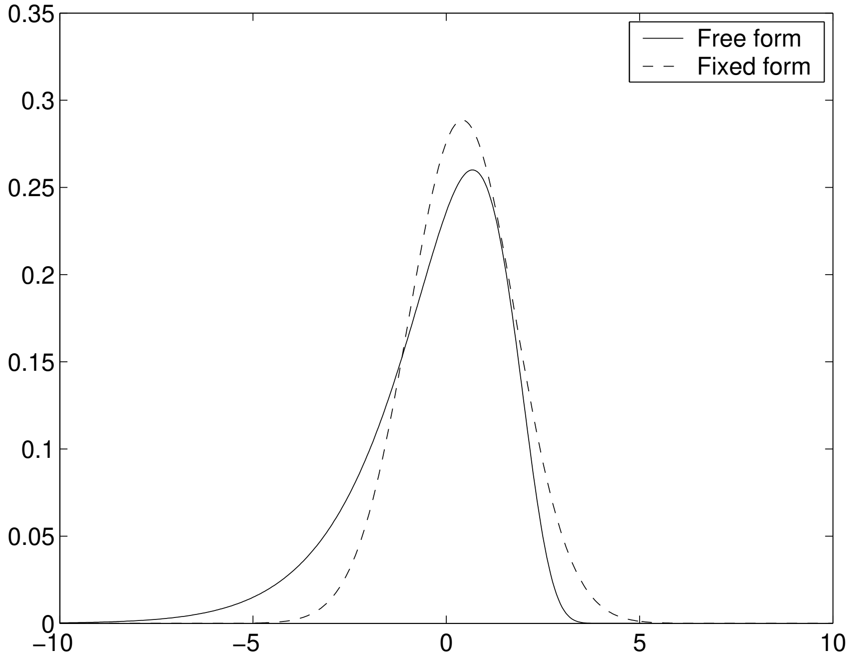


Figure 3.2: The free form approximating posterior probability distribution and the corresponding Gaussian fixed form solution.

we would like to have a much more complex model for the variance. Indeed, if we were to build a different model directly for the variance we would have to tune our learning algorithm appropriately. However, this can actually be circumvented by the introduction of so called *variance neuron* [51]. It is a time dependent Gaussian variable, $u(t)$, controlling the variance of another time dependent Gaussian variable $\xi(t)$

$$p(\xi(t) \mid \mu_\xi(t), u(t), I) = \mathcal{N}\left(\xi(t) \mid \mu_\xi(t), \exp\left[-u(t)\right]\right) \tag{3.6}$$

Now, instead of directly constructing a model for the variance of $\xi(t)$ we can build the model on top of the variance neuron $u(t)$. This in effect changes the problem of estimation of variance into estimation of mean. We still need to estimate $u(t)$ but that problem is exactly the same as the problem of estimating v in the example model of Section 3.1.1.

3.1.4 Other approaches

Variance modelling has been mostly of interest in the field of financial time series modelling where the estimation of volatility is an important application.

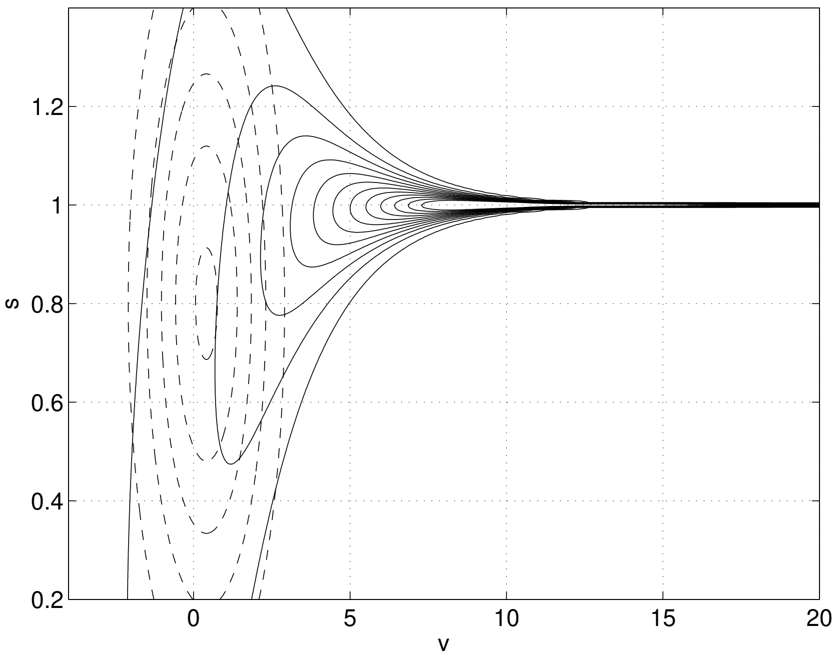


Figure 3.3: The true and approximating posterior distribution.

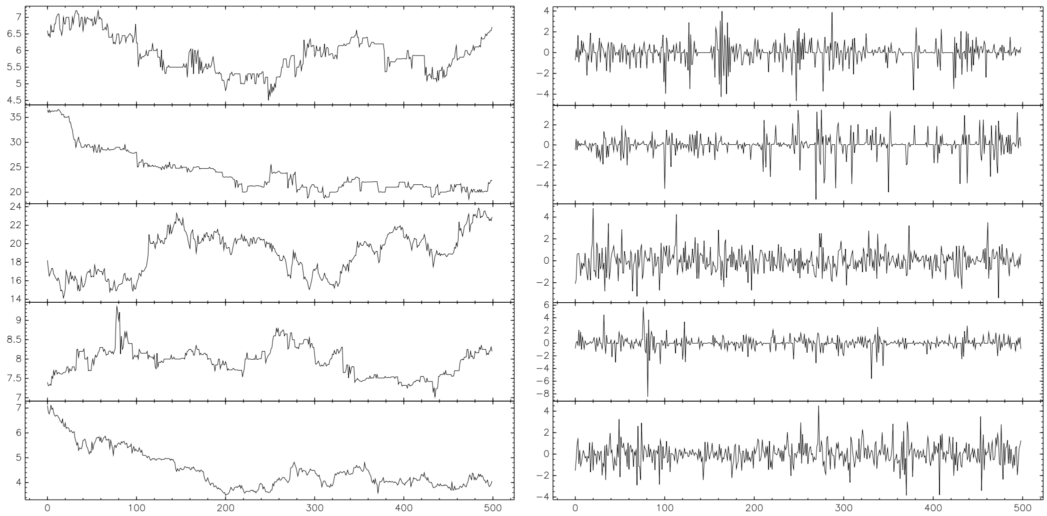
To give a hands-on feeling of volatility, typical financial time series are shown in Figure 3.4(a). The series are a set of daily stock price values in Helsinki Stock Exchange during years 2000 and 2001. The Figure 3.4(b) shows the corresponding log-returns. There the fluctuations of volatility (variance) are quite clearly visible.

The most widely discussed model is the autoregressive heteroscedasticity (ARCH) model, introduced in [11] and its extension, the Generalized ARCH (GARCH) model [7]. One simplified version of these models is the following. Using our notation, the model equations are

$$\begin{aligned} p(x(t) \mid u(t), I_1) &= \mathcal{N}\left(x(t) \mid 0, u(t)\right) \\ u(t+1) &= \alpha x^2(t) + \beta u(t) \end{aligned}$$

which state that the variance at any given instance of time depends on the past variances and on the squares of observations. Here the estimation of $u(t)$, α and β can be done recursively using an equivalent approach as in Kalman filters. A survey of ARCH models can be found in [8].

Other more recent methods are the stochastic volatility (SV) models. A good example is the model considered in [33]. In our notation again, the model is



(a) (b)

Figure 3.4: Daily stock prices of a set of securities (from Helsinki Stock Exchange) and the corresponding log-returns.

summarized by the following equations

$$p(x(t) \mid \beta, u(t), I_2) = \mathcal{N}\left(x(t) \mid 0, \exp\left[u(t)/2\right]\right)$$
$$p(u(t+1) \mid u(t), \mu, \phi, \sigma_u^2, I_2) = \mathcal{N}\left(u(t+1) \mid \mu + \phi(u(t) - \mu), \sigma_u^2\right)$$

where the observed variable $x(t)$ is the mean corrected return on holding an asset at time t. The model is learnt using a MCMC method based on Gibbs sampler.

3.2 Statistical image sequence models

Let us first examine the form of an image sequence dataset. An image sequence can be expressed as function of three variables $f(i, j, t)$. Here i and j denote discrete spatial coordinates, i corresponding to the columns from 1 to N and j to the rows from 1 to M. The time index t is also discrete, running from 1 to T. As usual, for notational and practical convenience, the individual frames are arranged into vectors $\mathbf{x}(t)$ of size $N \times M$. The exact arrangement of pixels in the vector does not make any difference. One possibility is to scan the pixels of a frame row-wise starting from the upper left corner such that the correspondence between f and $\mathbf{x}$ is $x_{jM+i}(t) = f(i, j, t)$.

The statistical models that have been of most interest have been the linear ones. This is due to the fact that their learning is usually relatively easy and the interpretation of results is straightforward. A linear model for image sequences can be formulated in several ways. We can either take an approach where we see the image sequence as an input to a linear system as in

$$\mathbf{y}(t) = \mathbf{W}\mathbf{x}(t) \tag{3.7}$$

Another approach is to see the image sequence as a product of some generative process. This can be formulated as

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \tag{3.8}$$

These two different approaches are equivalent only when $\mathbf{A}$ and $\mathbf{W}$ are of full rank. In that case the one can be derived from the other simply by matrix inversion, $\mathbf{A} = \mathbf{W}^{-1}$.

In both of the above models, the factors $\mathbf{s}(t)$ or the outputs $\mathbf{y}(t)$ have one to one correspondence with the data $\mathbf{x}(t)$ without any reference to the order of the frames. This means that we could permute the frames of the image sequence and still obtain exactly the same results. If the time dependencies are not taken into account, we will certainly miss something, since there exist strong correlations between consecutive frames in any natural image sequence. In (3.7) one can take time dependencies into account by replacing the matrix $\mathbf{W}$ with a three-dimensional tensor that operates on a set of frames at any given instance of time. This leads to a spatio-temporal model for the outputs $\mathbf{y}(t)$. In (3.8) one possibility is to add dynamics to the sources $\mathbf{s}(t)$ by making the model of $\mathbf{s}(t)$ depend on the past values $\mathbf{s}(t - \tau)$ in some way.

There have been several different criteria for learning such models as the ones in Equations (3.7) and (3.8). These include independence [4], sparseness [44] and temporal coherence [25]. The central result of all these methods has been that the obtained basis is rather sparse and the individual vectors resemble Gabor filters [13]. For the researchers in the field of computational neuroscience these have been interesting findings. This is so due to the fact that the simple cells in the mammal visual cortex have activation patterns also similar to Gabor filters, as was first shown by Hubel and Wiesel [24] by experiments with cats.

Nonlinear methods for image sequences include methods such as the slow feature analysis [5, 57]. There the key hypothesis is that the most interesting features are relatively slowly changing in time. Hence, slowness is used as the criterion for learning. A bit similar approach to the one taken in this thesis, has been used by Hyvärinen et al. They have used the correlations of energies to find independent subspaces of simple cells [27] and to form a topographic ordering for the sources [28]. In [29] Hyvärinen et al. combined several different criteria of learning. Those were temporal coherence, sparseness and topography. They estimated the variances of the simple cells of their generative model and showed that those have both spatial and temporal structure which can be seen in the time-location plane as spatio-temporal 'bubbles.'

3.3 Modelling variances in image sequences

The central goal of this thesis is to make use of variance in image sequence modelling. In this section this is motivated.

A typical linear generative model for an image sequence, such as the one in Eq. (3.8), will usually yield a sparse basis for the frames. The respective sources correspond to a certain area in the frame and in the case of Gabor filter type of basis also to some specific orientation and frequency. These simple features are not very interesting by themselves, but it can be postulated that by constructing a model over these features, we could obtain more interesting representation for the data. In Figure 3.5 there are typical signals that arise in image sequence modelling. With visual inspection, it is clear that something interesting is happening starting approximately from time index 125 and continuing until almost to time index 150. These activation patterns are a result

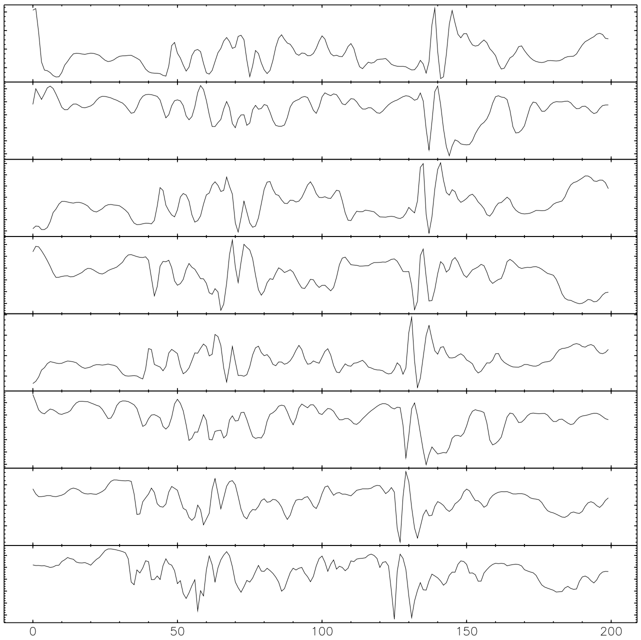


Figure 3.5: Typical sources

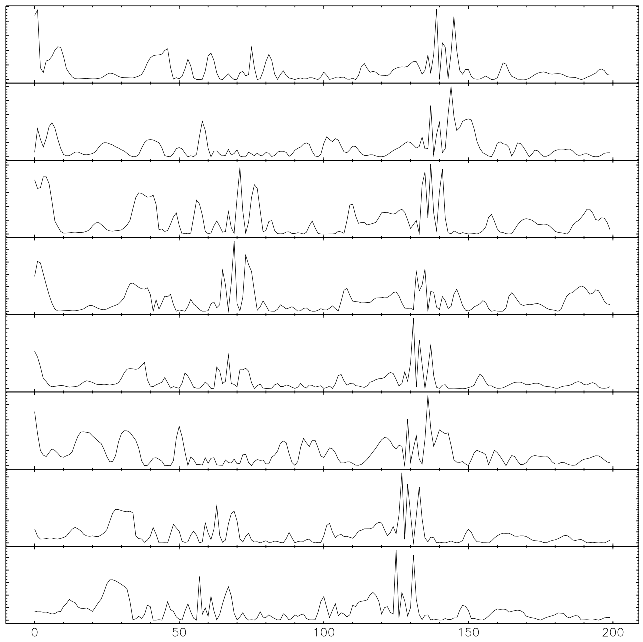


Figure 3.6: Energies of the typical sources

of an object moving in the image and activating the sources one at a time. To capture the movement, we could add dynamics to our model, meaning that the sources would be dependent on the their past values. In the later chapters, it is experimentally shown that this kind of approach fails. The reason should be quite evident, since although the eye captures the "correlations" between the sources, there actually exists none due to their different phases. If there were a way to capture the overall activity of a signal instead of its exact amplitude at any given moment, we could probably be more lucky. A quite often used approach is to model the energies of the signals i.e. use $s^2(t)$ instead of $s(t)$. Performing this transform for the sources results in the signals in Figure 3.6. Now the correlation between different signals is a bit stronger but we still do not have a very good description of the activity.

To model the variances of the sources, we can take the variational approach with the usage of variance neurons. The exact model equations are given later on. Now we are mainly interested in taking a quick peek of what kind of results are obtainable with variance modelling. Figure 3.7 shows the log-variances of the corresponding sources of Figure 3.5. The variance provides exactly the kind of feature we were hoping for in the sense that it describes the activity of the sources and does it in a much more invariant way than the energy approach.

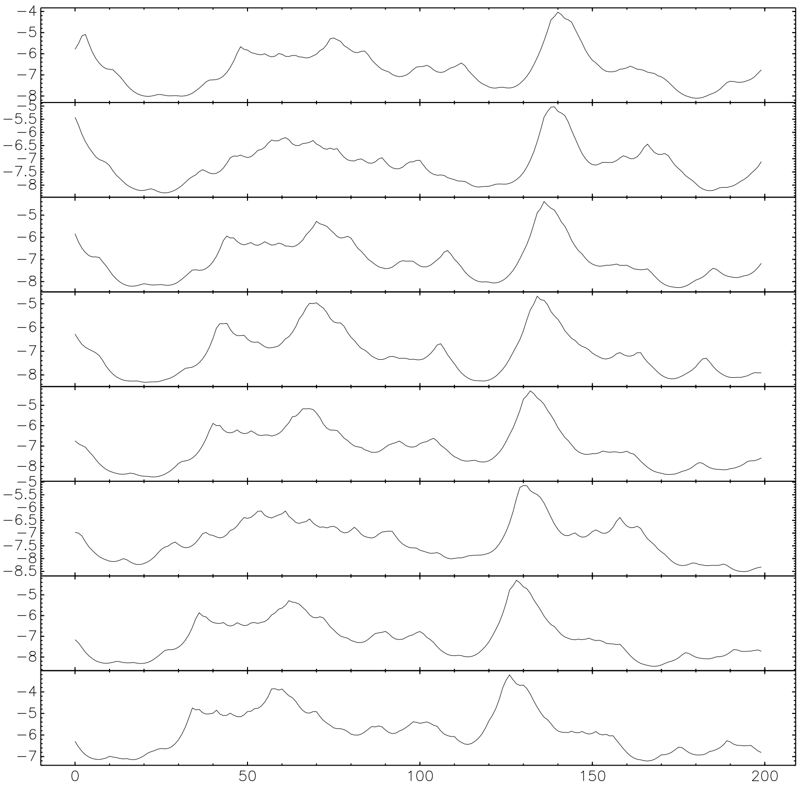


Figure 3.7: Log-variances of the typical sources

Chapter 4

Bayes Blocks

Bayes Blocks is a framework developed by Valpola et al. [55, 47]. Its theoretical foundation is in variational Bayesian learning. The software library [52] implementing the framework provides continuous and discrete variables and computational nodes. By combining them, one can construct a rich variety of diverse models. This chapter discusses the theoretical background and the available blocks of the library. Also some practical issues concerning learning of models built using the library are discussed.

4.1 Key principles

To illustrate what kind of computations are taking place in Bayes Blocks framework, let us consider a simple model where there are only Gaussian variables connected directly to each other. Let us formulate the model as follows

$$p(x \mid s, v_x) = \mathcal{N}\left(x \mid s, \exp[-v_x]\right)$$
$$p(s \mid m_s, v_s) = \mathcal{N}\left(s \mid m_s, \exp[-v_s]\right)$$

These equations can be seen as a part of a larger model. To be able to do inference using variational Bayesian learning we need to choose a simpler distribution for the approximation of the true posterior. Let us assume a fully factorial posterior approximation

$$q(x, v_x, s, m_s, v_s) = q(x)q(v_x)q(s)q(m_s)q(v_s)$$

This is the kind of assumption that is made throughout the Bayes Blocks framework. Now we are interested in what type of computations are required to update the variable s, or more precisely its posterior approximation $q(s)$, given that the distributions of the rest of the variables are assumed fixed.

Writing down the cost function relevant to updating the variable s

$$\begin{aligned}
\mathcal{C}_s &= \left\langle \log \frac{q(s)}{p(x \mid s, v_x)p(s \mid m_s, v_s)} \right\rangle_{q(x,v_x,s,m_s,v_s)} \\
&= \left\langle \log q(s) - \langle \log p(x \mid s, v_x) \rangle_{q(x,v_x)} - \langle \log p(s \mid m_s, v_s) \rangle_{q(m_s,v_s)} \right\rangle_{q(s)} \qquad (4.1)
\end{aligned}$$

The expectations over $\log p(s \mid m_s, v_s)$ and $\log p(x \mid s, v_x)$ in (4.1) can be evaluated to yield simpler expressions.

$$\begin{aligned}
\langle \log p(s \mid m_s, v_s) \rangle &= \langle \log \mathcal{N}\left(s \mid m_s, \exp -v_s\right) \rangle \\
&= \left\langle -\tfrac{1}{2}(\exp v_s)(s - m_s)^2 \right\rangle + C \\
&= -\tfrac{1}{2} \langle \exp v_s \rangle \left\langle s^2 - 2sm_s + m_s^2 \right\rangle + C \\
&= -\tfrac{1}{2} \langle \exp v_s \rangle \left(s^2 - 2s \langle m_s \rangle + \langle m_s \rangle^2\right) + C' \\
&= -\tfrac{1}{2} \langle \exp v_s \rangle \left(s - \langle m_s \rangle\right)^2 + C' \\
&= \log \exp\left(-\tfrac{1}{2} \langle \exp v_s \rangle \left(s - \langle m_s \rangle\right)^2\right) + C' \\
&= \log \mathcal{N}\left(s \mid \langle m_s \rangle, \langle \exp v_s \rangle^{-1}\right) + C'' \qquad (4.2)
\end{aligned}$$

Similarly

$$\langle \log p(x \mid s, v_x) \rangle = \log \mathcal{N}\left(\langle x \rangle \mid s, \langle \exp v_x \rangle^{-1}\right) + C \qquad (4.3)$$

Substituting the terms (4.2) and (4.3) back to (4.1) we get

$$\mathcal{C}_s = \left\langle \log \frac{q(s)}{\mathcal{N}\left(\langle x \rangle \mid s, \langle \exp v_x \rangle^{-1}\right) \mathcal{N}\left(s \mid \langle m_s \rangle, \langle \exp v_s \rangle^{-1}\right)} \right\rangle_{q(s)} + C$$

The product of the two normal distributions is proportional to another normal distribution. Hence the optimal approximation is a normal distribution $q(s) = \mathcal{N}\left(s \mid \bar{s}, \tilde{s}\right)$, with parameters

$$\begin{aligned}
\tilde{s} &= \left(\langle \exp v_x \rangle + \langle \exp v_s \rangle\right)^{-1} \qquad \text{and} \\
\bar{s} &= \tilde{s}\left(\langle \exp v_x \rangle \langle x \rangle + \langle \exp v_s \rangle \langle m_s \rangle\right).
\end{aligned}$$

We see that the update of the distribution of s can be done by propagating certain expected values from its parents, child and co-parent. This is illustrated in Figure 4.1. Actually the propagation is done in such a way that the variable s needs to communicate only with its most immediate neighbors i.e. with m_s, v_s and x. When s is updated, it asks expectations from its parents m_s and v_s.

From x it asks a gradient, which is, as its name implies, a gradient computed from the cost function w.r.t. the expectations $\langle \cdot \rangle$ and $\langle \exp \cdot \rangle$. However, the gradients are not used in any ordinary manner such as in back propagation in MLP. The propagation of gradients is just a clever trick to make x encode both $\langle x \rangle$ and $\langle \exp v_x \rangle$ so that s need not to communicate with v_x.

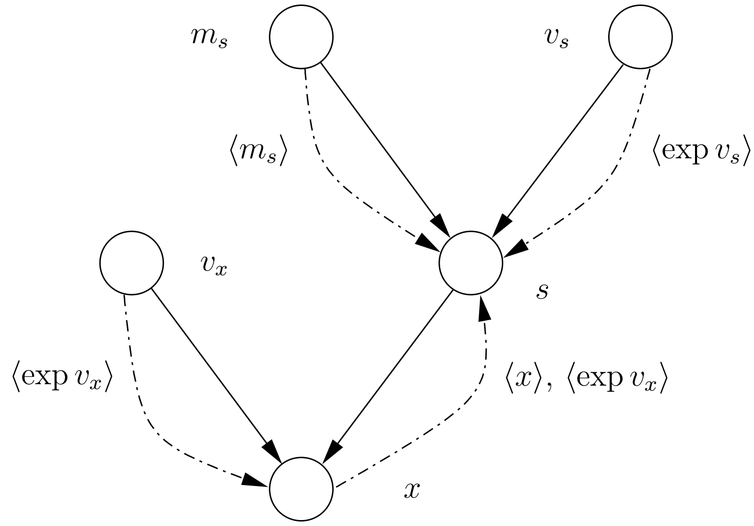


Figure 4.1: Updating the variable s. The solid lines show the actual logical dependencies and dashed lines show the propagation of the required expected values to update $q(s)$.

Here we considered only Gaussian variables, but also the other nodes of the Bayes Blocks library have been designed so that the update of the variables can be done similarly using only local information obtainable from the immediate neighbors of the variable. This has the positive effect that the computational complexity of any given model is linear w.r.t. the amount of variables in the model given that the number of connections from any variable is fixed. The local update rules come at the price that we have to use maximally factorial approximating posterior distribution. This means that we neglect all the posterior dependencies, which can result to ill behavior of our models in some cases. For example, Ilin and Valpola [30] showed that in ICA the fully factorial approximation can result in favoring a more PCA type of solution than would be appropriate.

4.2 Available blocks

The following list summarizes the blocks of the library which are relevant to this thesis.

Gaussian variable A variable s with Gaussian cpf parametrized with its mean m and negative log-variance v such that

$$p(s \mid m, v) = \mathcal{N}(s \mid m, \exp[-v])$$

The variance is parametrized this way for several technical reasons.

Rectified Gaussian variable A variable s with rectified Gaussian cpf

$$p(s \mid m, v) = \mathcal{N}^R(s \mid m, \exp[-v])$$

This is a Gaussian whose tail in the negative axis has been rectified and the right tail scaled appropriately.

Mixture of Gaussians (MoG) variable A variable s whose cpf is controlled by a discrete variable λ which selects one of the K different Gaussian distributions:

$$p(s \mid \{m\}_{i=1}^K, \{v\}_{i=1}^K, \lambda = k) = \mathcal{N}(s \mid m_k, \exp[-v_k])$$

Discrete variable A variable λ which has a discrete distribution over integers $1 \ldots K$. The cpf of the discrete variable can either be static

$$p(\lambda(t) = i \mid \mathbf{c}) = c_i$$

or it can depend on the previous value as in Markov chains

$$p(\lambda(t) = i \mid \lambda(t-1) = j, \mathbf{C}) = c_{ij}$$

Here the parents $\mathbf{c}$ (or the parents $\mathbf{c}_j$ in Markov prior) can be constant or Dirichlet variables. The main usage of the discrete variable is to serve as a parent to the MoG variable.

Dirichlet variable A variable $\mathbf{c}$ with Dirichlet prior

$$p(\mathbf{c} \mid \mathbf{u}) = \mathcal{D}(\mathbf{c} \mid \mathbf{u})$$

It can be used as the parent of a discrete variable or to provide the weights of the transition probability matrix when the discrete variable has Markov prior.

Summation node A computational node that computes the sum of its two parents.

Product node A computational node that computes the product of its two parents. Can be used to construct linear mappings in conjunction with the summation node.

The cost function and the update rules for the Gaussian variable (both when acting as a mean or as a variance parent) in connection with computational nodes can be found from e.g. [47]. For the other variables listed above, the update rules and cost functions are derived in Appendix B. There also exists a bunch of other nodes, some of which were not used in this work, and some which only have relevance when one is concretely implementing a model using the library.

The block framework has been successfully used for several models. It has been applied to hierarchical nonlinear factor analysis [54] and variance modelling [50, 51]. It can be relatively easily used in on-line fashion [21]. Also handling of missing and partially observed values is possible [49, 48].

4.3 Learning

The variational Bayesian learning provides a cost function which can be analytically evaluated for the blocks. This can be minimized by minimizing the contribution of one variable at a time. By doing this, it is clear that the cost has to decrease at every step. Cost function being (usually) bounded from below with some constant, the iteration finally ends at some stationary point. However, this is all that can be guaranteed. Whether the stationary point is even near any good solution, is totally up to the learning algorithm. By learning algorithm it is meant the initializations, gradual building of the model, pruning of some parameters etc. Another problem is that although we were to reach a good stationary point finally, it might take so many iterations that we couldn't possibly wait for the simulation to reach that point. These phenomena are illustrated below with linear positive ICA applied to separation of images.

The data consists of ten mixtures of tree original images. Ignoring the details of the model, we can compare the learning with different initializations. We try three approaches: non-negative PCA [43], ordinary PCA [32] followed by taking the absolute values (remember that we are doing positive ICA) and random initialization. The first one of these, the non-negative PCA, should give an initialization that is very near the actual sources and there should not be much to learn.

The model is able to reconstruct the sources very well with all these initializations. The only difference is the amount of iterations that is needed. Figure 4.2 shows the cost function as the function of the number of iteration for the different methods.

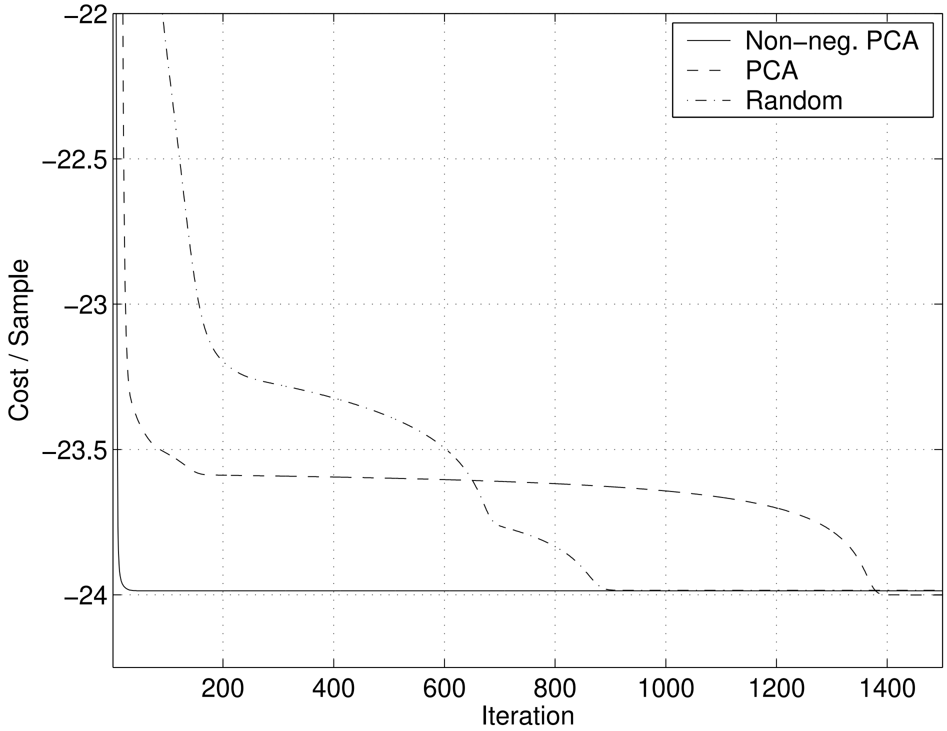


Figure 4.2: Cost function values

As expected, the Non-negative PCA initialization makes the model converge almost immediately. With the other two methods, the right subspace is found in a couple of hundred iterations, but finding the correct rotations takes approximately 900 iterations for the model with random initializations and 1400 iterations for the model with standard PCA initializations. It is worth noting that the cost function of the PCA initialized model doesn't change much from iteration 200 to iteration 800. One might stop the simulation in the belief that it has converged although there's still a long way to the optimum.

Chapter 5

Experiments

In this chapter, a chain of experiments with image sequence models is reported. We start with an ordinary linear model and continue to build a higher level model on that. The development ends at a model, which is aimed to be able to extract motion features from image sequence data.

5.1 The first layer model

Remembering the notation from Section 3.2, we will construct a linear model for the frames of the image sequence $\mathbf{x}(t)$ such that

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}_x(t) \tag{5.1}$$

where $\mathbf{n}_x(t)$ denotes Gaussian noise. The source model will be non-Gaussian with possible dynamics or positivity constraint.

First we write the equivalent of (5.1) in probabilistic notation

$$p(\mathbf{x}(t) \mid \mathbf{A}, \mathbf{s}(t), \mathbf{v}_x) = \mathcal{N}\left(\mathbf{x}(t) \mid \mathbf{A}\mathbf{s}(t), \exp[-\mathbf{v}_x]\right) \tag{5.2}$$

There's a choice to be made which is related to the scale of the different parts of the model. We can either fix the scale of the weights a_{ij} in the linear mapping $\mathbf{A}$ and allow the scale of the source vary or vice versa. We choose to fix the scale of the mixing matrix by fixing the prior variance for the weights to be equal to one

$$p(\mathbf{A}) = \prod_{i,j} \mathcal{N}\left(a_{ij} \mid 0, 1\right) \tag{5.3}$$

We do not have much knowledge about the observation noise variance. It is of course dependent on the amount of sources used. The more sources, the better we can reconstruct the data and the less there is to be accounted as noise. At some point, when all the significant correlations have been modelled in data by the sources, increasing the number of the sources does not help any more. We can express this ignorance by using a hierarchical prior which is essentially uninformative

$$
\begin{aligned}
p(\mathbf{v}_x \mid m_{vx}, v_{vx}) &= \prod_i \mathcal{N}\left(v_{xi} \mid m_{vx}, \exp[-v_{vx}]\right) \\
p(m_{vx}) &= \mathcal{N}\left(m_{vx} \mid 0, e^7\right) \\
p(v_{vx}) &= \mathcal{N}\left(v_{vx} \mid 0, e^7\right)
\end{aligned}
$$

The sources can have several different models. The most obvious one is the Gaussian model

$$
p(\mathbf{s}(t) \mid \mathbf{v}_s) = \mathcal{N}\left(\mathbf{s}(t) \mid 0, \exp[-\mathbf{v}_s]\right) \tag{5.4}
$$

Using Gaussian sources will yield PCA type solutions. However, we would like to find a more ICA type solution and therefore want our sources to be non-Gaussian. In addition to that, we are interested in modelling the variances of the sources, so the natural approach is to change the static variance parameters $\mathbf{v}_s$ in (5.4) to variance neurons $\mathbf{u}(t)$:

$$
p(\mathbf{s}(t) \mid \mathbf{u}(t)) = \mathcal{N}\left(\mathbf{s}(t) \mid 0, \exp[-\mathbf{u}(t)]\right) \tag{5.5}
$$

Consecutive frames in an image sequence usually have strong temporal correlations so we can expect our sources also to have such quality. Consequently, we can add simple dynamics to our sources by making the mean of sources at a moment t to depend straightforwardly on the previous values at the moment of time $t-1$

$$
p(\mathbf{s}(t) \mid \mathbf{s}(t-1), \mathbf{u}(t)) = \mathcal{N}\left(\mathbf{s}(t) \mid \mathbf{s}(t-1), \exp[-\mathbf{u}(t)]\right) \tag{5.6}
$$

Now the variance $\mathbf{u}(t)$ has a bit different interpretation. It characterizes the innovation process, in effect telling us something about the changes of the signal amplitude rather than about the signal amplitude itself.

The model built so far is graphically summarized in Figure 5.1. Now we are ready to take a look at some experimental results obtainable using this model. The data used was a real gray scale video sequence with 16×16 spatial dimensions and 4000 samples. The most interesting part of the current model is the linear mapping $\mathbf{A}$. Its column vectors $\mathbf{a}_i$ correspond to the different sources $s_i(t)$. Consequently the behavior of any given source $s_i(t)$ is characterized

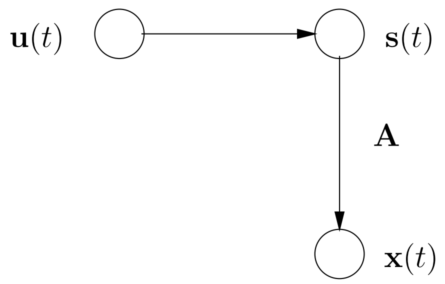


Figure 5.1: The first layer model. Observations $\mathbf{x}(t)$ are generated from the sources $\mathbf{s}(t)$ through a linear mapping $\mathbf{A}$. The variance of the sources is controlled by variance neurons $\mathbf{u}(t)$.

completely by the corresponding basis vector $\mathbf{a}_i$. These are shown in Figure 5.2. The basis is relatively sparse in the sense that most of the features are localized to a certain relatively small area of the patch. One thing to be noted is that there exists one component whose weights are constant throughout the patch. This is called the dc-component. We could remove it from the dataset altogether by removing the sample mean, but instead we choose to handle it in our model. However, we do not want to estimate the weights corresponding to the dc-component, since we know they will have essentially constant values overall the patch. Instead, we define a constant vector $\mathbf{c} = [1\,1\,\ldots 1]^T$, of same dimension as $\mathbf{x}$, which is modulated by the scalar dc-component $s_{\mathrm{dc}}(t)$. This changes (5.2) to

$$p(\mathbf{x}(t) \mid \mathbf{A}, \mathbf{s}(t), s_{\mathrm{dc}}(t), \mathbf{v}_x) = \mathcal{N}\left(\mathbf{x}(t) \mid \mathbf{A}\mathbf{s}(t) + s_{\mathrm{dc}}(t)\,\mathbf{c}, \exp[-\mathbf{v}_x]\right)$$

In principle, gray scale images are measurements of light intensity and hence image sequence data is naturally positive, black pixels corresponding to zero intensity. Accordingly, we might consider a positive model. This can be achieved by using rectified Gaussian priors both for the sources and for the linear mapping, changing (5.3) and (5.5) to

$$p(\mathbf{A}) = \prod_{i,j} \mathcal{N}^R\left(a_{ij} \mid 0, 1\right) \qquad \text{and}$$
$$p(\mathbf{s}(t) \mid \mathbf{u}(t)) = \mathcal{N}^R\left(\mathbf{s}(t) \mid 0, \exp[-\mathbf{u}(t)]\right)$$

The basis changes quite a lot when using this positive model, as can be seen in Figure 5.3. The features of the ordinary model do overlap quite a lot, whereas the basis vectors of the positive model are almost completely disjoint. This is of course what should be expected because now there is no way to compensate the effect of one feature by subtracting another feature from it.

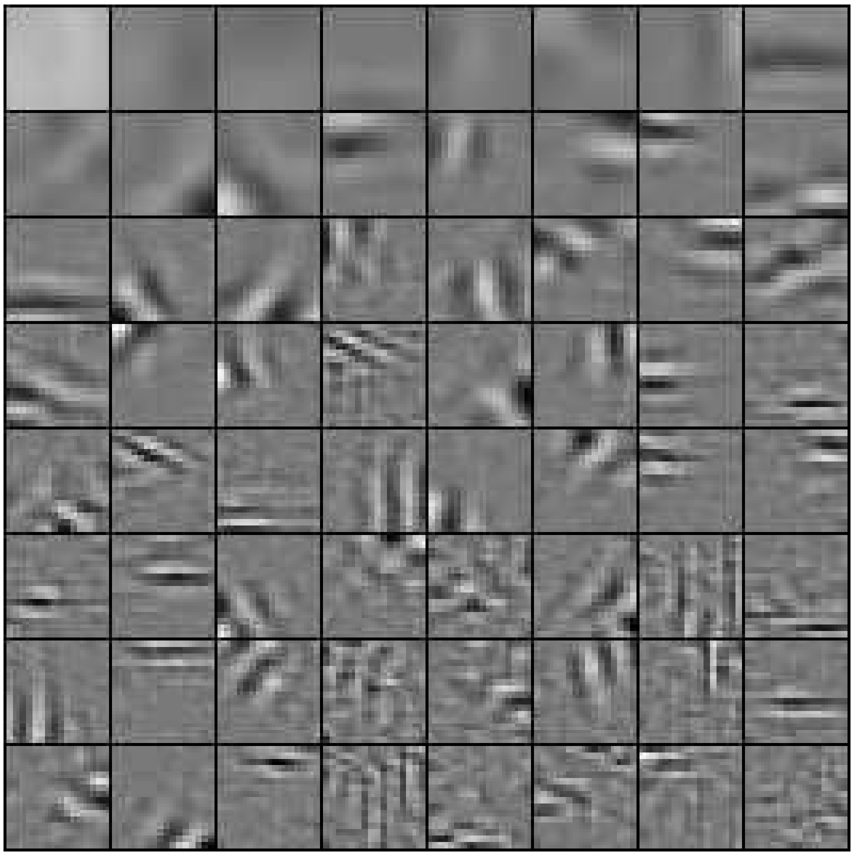


Figure 5.2: Ordinary basis

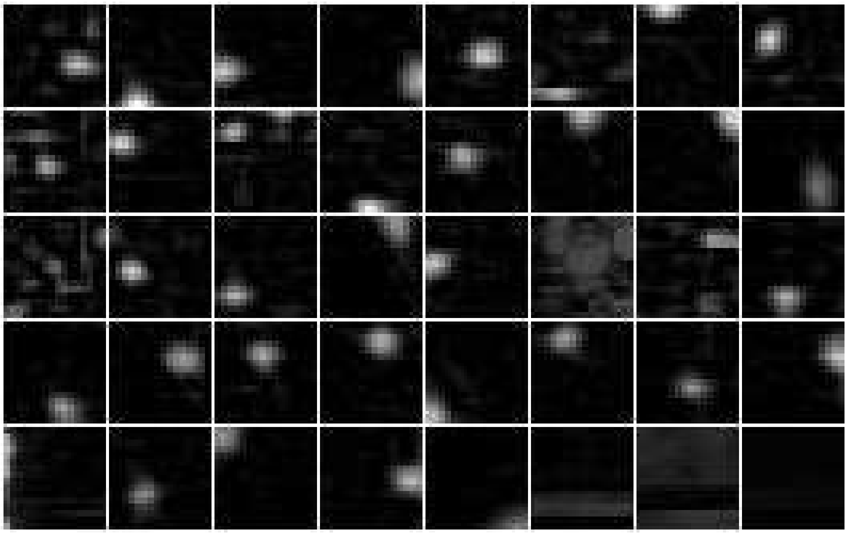
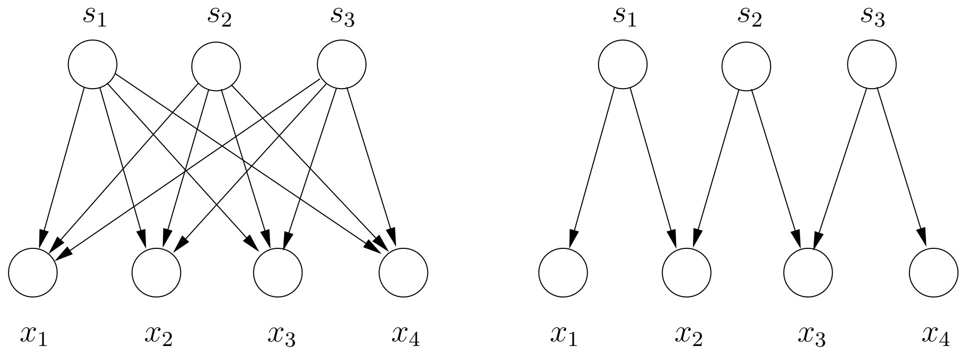


Figure 5.3: Positive basis

5.2 Restriction to localized sparse codes

As was suspected, the basis of the first layer model in the previous section became relatively sparse. We can make this sparsity work for us in several ways. Let us consider a model where there are three sources and four observation. If we have a full linear mapping, the connections from the sources to observations correspond to those in Figure 5.4(a). In a full linear mapping, the number of



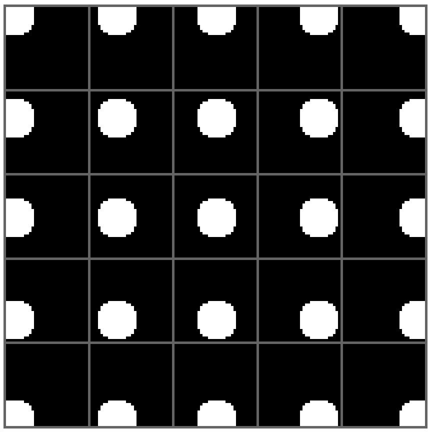
(a) Dense coding (b) Sparse coding

Figure 5.4: Dense vs. sparse coding

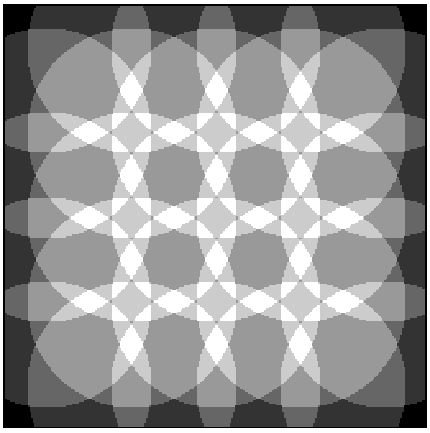
weights is of course the number of columns (sources) times the number of rows (observations). In this thesis typical values for these might be 64 and 256 respectively summing up to total of 16384 weights. It goes without saying that the learning will be quite slow. And as it turned out, most of the weights will be near zero anyway, corresponding to the situation in Figure 5.4(b), meaning that we have done a lot of redundant computations.

The cost function of variational Bayesian learning allows us to compare different models. For example, we can remove some weights from the linear mapping and see if the cost decreases. If it does decrease, it means that the weights were useless. This is called *pruning*. By pruning we can get rid of unnecessary parts of the model, but one has to be careful at which stage of learning one uses pruning. On one hand, if it is done too early, some parts of the model that might become useful can be removed. On the other hand, if it is done too late, the redundant parts of the model might interfere with the other parts, and again a suboptimal solution is found.

Instead of using a full basis and pruning it at some point of the learning, we can restrict our linear mapping to be suitably sparse from the very beginning. While we do this, we can also add some structure to it to make the interpretation of results easier and the construction of the upper layer model simpler. For



(a) Connection mask



(b) Overlap pattern

Figure 5.5: Mask and overlap

example, we can organize the spatially neighboring features near each other. Figure 5.5(a) shows a connection matrix where such an approach has been taken. However, we want to be careful not to make the basis too sparse such that there is no overlap in the features. The overlap pattern of the connection matrix in Figure 5.5(a) is shown in Figure 5.5(b). In that example every pixel has at most six sources connected to it and on average approximately 3.8 sources.

A sparse basis that has been estimated from an image sequence of dimensions $16 \times 16 \times 1200$ is shown in Figure 5.6. There the number of sources is 64 and they are organized in a grid of spatially neighboring features.

5.3 Adding dynamics

With the first layer model we can find features similar to those in Figure 5.6. The sources $\mathbf{s}(t)$ of the model do not give us very high level information about the image sequence. It is by combining these simple features that we try to extract some useful representations of the data.

Let us for a while consider the example image sequence of Figure 5.7. There an ice hockey player skates horizontally through our field of vision. We are interested in the activation pattern of the sources that this sequence produces. The basis $\mathbf{A}$ in this case is similar to that in Figure 5.6. We can inspect a subset of sources whose basis vectors correspond to the features of the second row in the figure. The values of these sources are plotted in Figure 5.8(a).

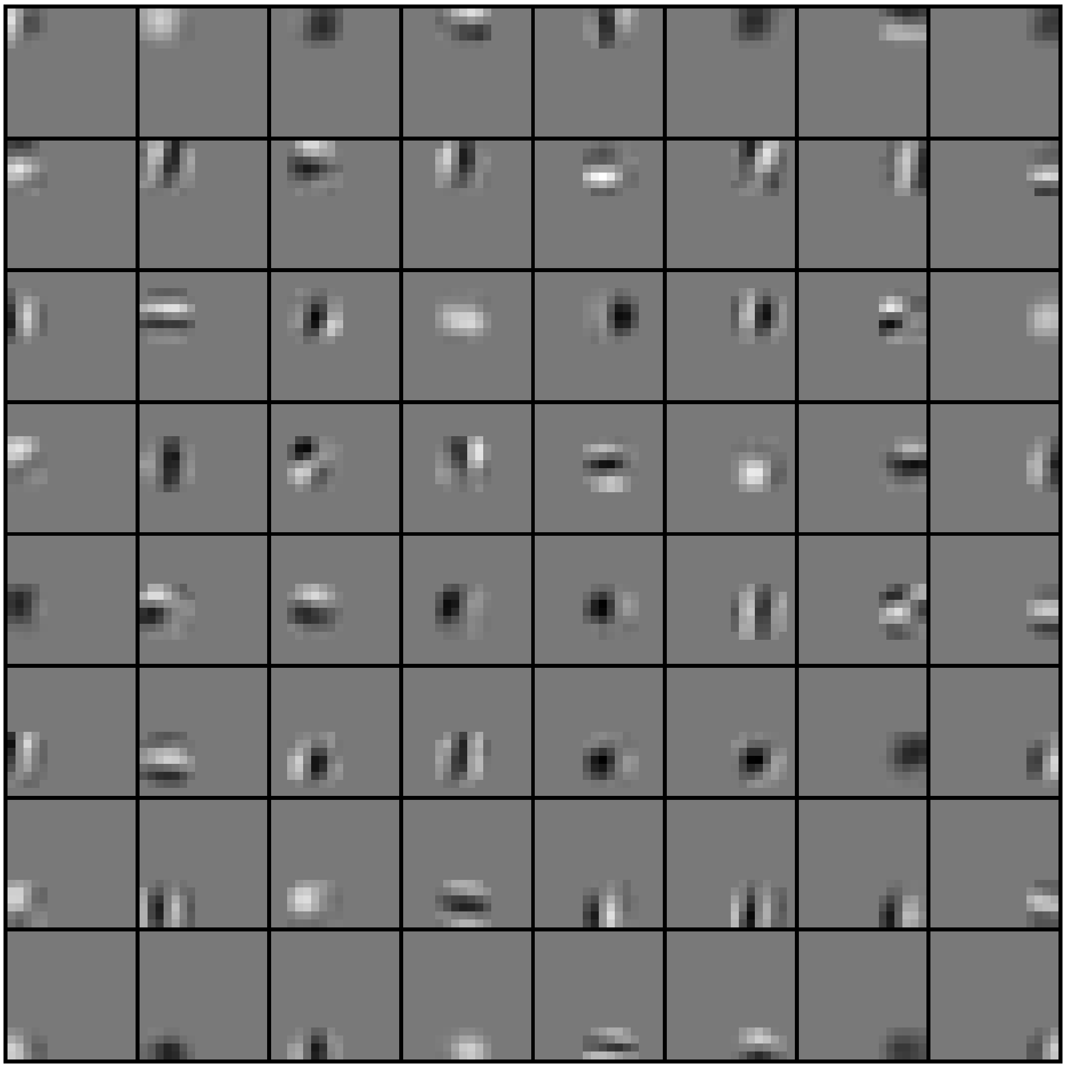


Figure 5.6: A sparse basis.

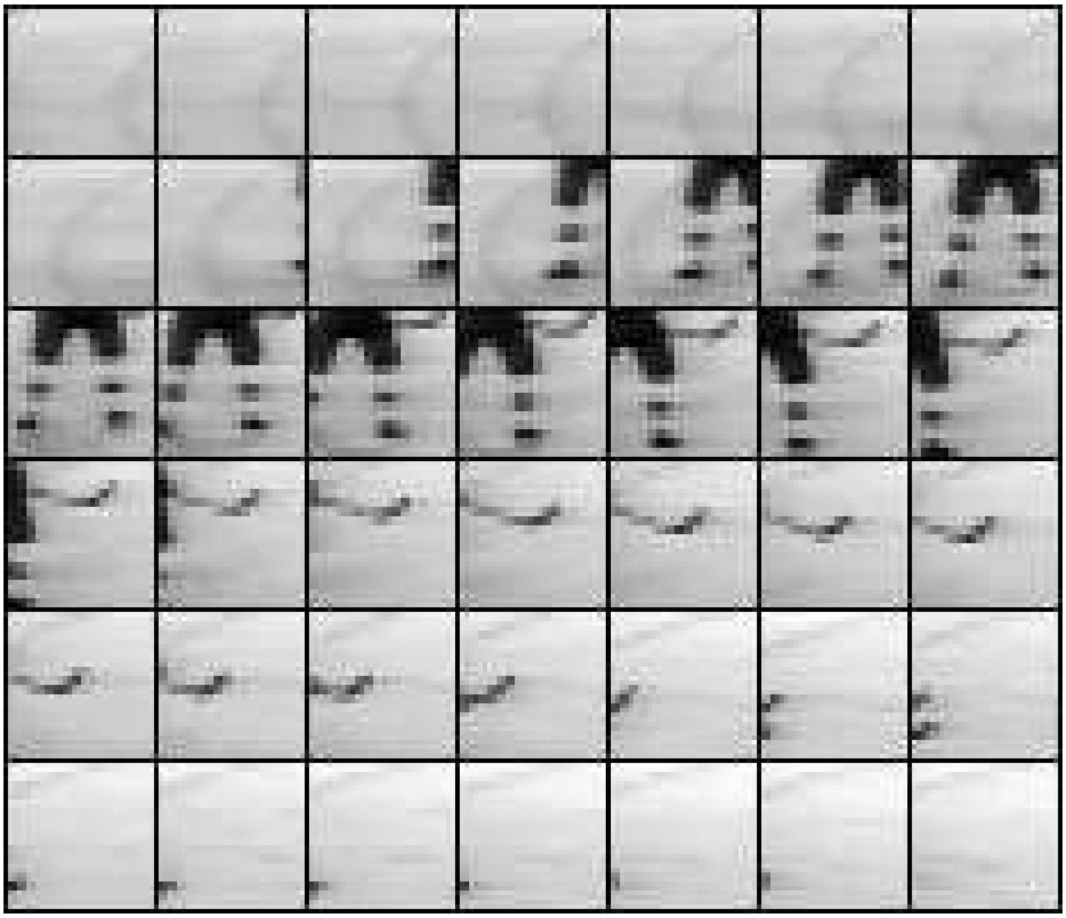
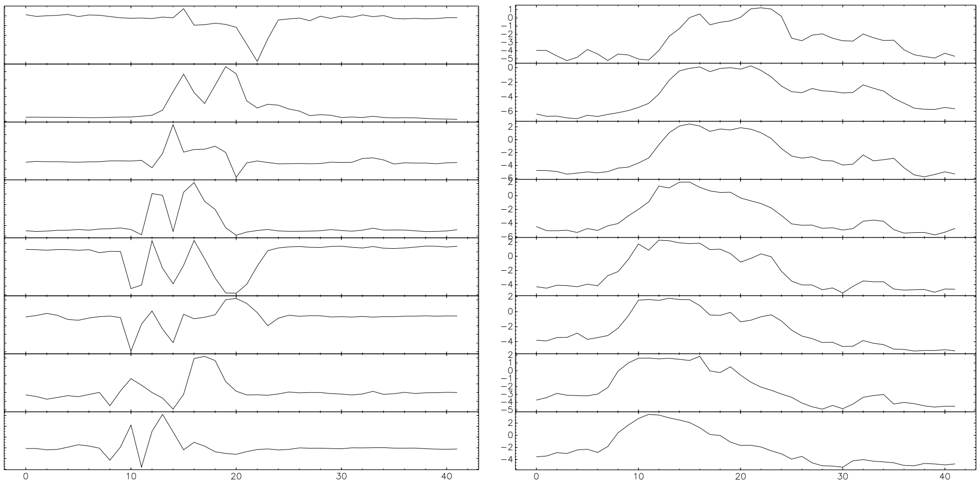


Figure 5.7: A sequence of 42 frames from a video. The first frame is in the top left corner and time flows row-vice. The frames are 16×16 pixels in size. The sequence is taken from a video of an ice hockey game. Here a player enters the frame from the right side end exits from the left side.



(a) Sources (b) Variances

Figure 5.8: Sources and their variances

Qualitatively they seem to have very similar behavior. It seems that the other signals are in their basic nature delayed copies of the bottom most signal. This of course is natural given the data and the basis vectors. Signals like these suggest the modelling of their dynamics. If the rest of the signals are copies of the bottom most signal, then there should be some way to express this in our model. A simple linear dynamics characterized by the linear mapping $\mathbf{B}$ can be added to the model by changing Equation (5.6) to

$$p(\mathbf{s}(t) \mid \mathbf{B}, \mathbf{s}(t-1), \mathbf{u}(t)) = \mathcal{N}\left(\mathbf{s}(t) \mid \mathbf{B}\mathbf{s}(t-1), \exp[-\mathbf{u}(t)]\right) \tag{5.7}$$

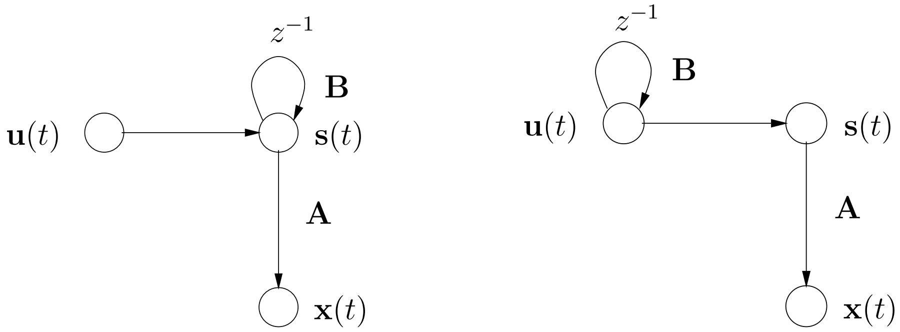
Keeping in mind the discussion of Section 3.3 this approach can be suspected to fail. Although it was said that the signals seem qualitatively similar, quantitatively they are not. Figure 5.8(b) shows the variances that correspond to the sources. There wee see much better behaving signals which could actually have some kind of dynamic dependencies. As an comparative approach we keep the Equation (5.6) as it was, but instead add a linear dynamics for the variance neurons $\mathbf{u}(t)$

$$p(\mathbf{u}(t) \mid \mathbf{B}, \mathbf{u}(t-1), \mathbf{v}_u) = \mathcal{N}\left(\mathbf{u}(t) \mid \mathbf{B}\mathbf{u}(t-1), \exp[-\mathbf{v}_u]\right) \tag{5.8}$$

These two different attempts to add dynamics to our model, recapitulated by Equations (5.7) and (5.8) are graphically illustrated in Figure 5.9. Using the whole image sequence, part of which the frames of Figure 5.7 were, we can test our hypothesis about the performance of the two different models.

Now the interesting part of the model is the linear mapping $\mathbf{B}$ describing the dynamical relations between different sources / variance neurons. A column vector $\mathbf{b}_j$ tells us the extent to which the signal j can predict the other signals. It can be expected that at least the diagonal elements b_{jj} of $\mathbf{B}$ will have some positive values. Figures 5.10 and 5.11 show the column vectors of the dynamics mapping for both of the cases considered. We can see that when the dynamics is in the sources, no significant dynamic relations are found between different sources. The only strongly positive elements are the diagonal ones, which was expected, as the sources are temporally coherent.

The situation is rather different in the case where the dynamics is used in the variance neurons. There exists strong dependencies with spatially neighboring sources. This is intuitively satisfying since the objects in an usual image sequence tend to move continuously from one point to another hence exciting neighboring features at consecutive instances of time.



(a) Dynamics in sources

(b) Dynamics in variance neurons

Figure 5.9: The model structure after having added linear dynamics either to sources (a) or to variance neurons (b).

5.4 What can linear dynamics model?

The addition of linear dynamics to our model, especially when it was used in the variance neurons, was beneficial. It revealed some interesting relations between different sources. It should of course be clear that if we are to find interesting features from an image sequence, the system doing the extraction has to have some kind of memory. Otherwise it only sees a collection of independent frames which have no relation to each other and which could be randomly permuted without any effect for that matter.

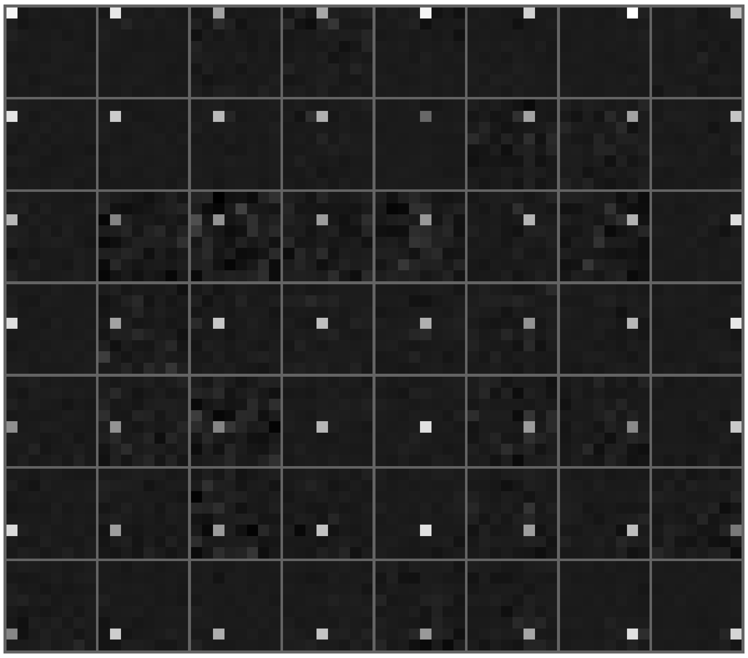


Figure 5.10: The column vectors of the linear mapping **B** when the dynamics is in the sources.

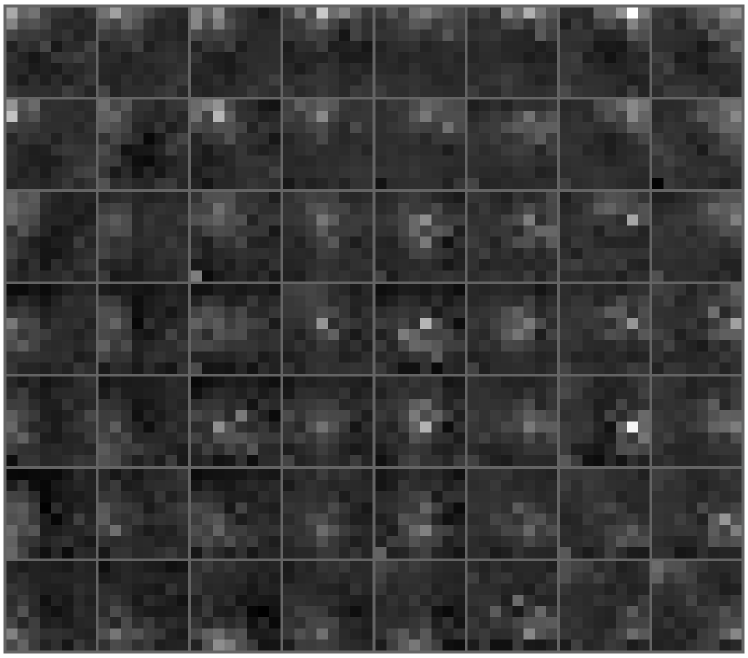


Figure 5.11: The column vectors of the linear mapping **B** when the dynamics is in the variance neurons.

There are many ways to make our model to have a memory. One way would be to allow our sources $\mathbf{s}(t)$ to have connections not only to the current observations $\mathbf{x}(t)$ but also to some set of past observations $\mathbf{x}(t-\tau)$, $\tau = 1 \dots K$. Another possibility is to add dynamics to the sources, as was done in the previous section.

It is a well known fact that linear systems can be divided in a half a dozen categories based on the eigenvalues of the linear mapping. Consequently the number of phenomena linear dynamics can describe is relatively limited. Keeping our focus on image sequences, we can study this matter in a bit simplified setting.

Assume that we have a model with four sources and a basis consisting of features spanning the frame horizontally from left to right as in Figure 5.12. Now, if we have an object in the image sequence moving over the frame from

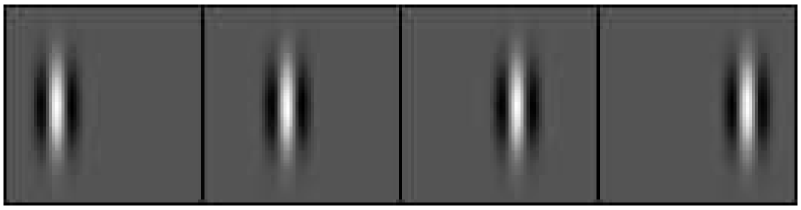


Figure 5.12: A set of features spanning the frame from left to right.

left to right in such a speed that it happens to excite the sources one at a time, then the variance of the first source predicts quite perfectly the variance of the second source and so on. Thus the situation is relatively well described by the dynamics mapping

$$\mathbf{B} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{5.9}$$

Similarly if we have an object moving from left to right over the frame, then dynamic relation goes the other way around and is perfectly well described by the mapping

$$\mathbf{B} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix} \tag{5.10}$$

With a different set of features, $\mathbf{A}$, and some linear mapping $\mathbf{B}$ relating them to each other, we can describe any kind of elementary movement such as the left to right or right to left above. However, if we for example have an object that moves over the frame to both directions, it is no longer perfectly caught by linear dynamics. Since we have a statistical model, we will still get some estimate of the dynamics, possibly a combination of (5.9) and (5.10), such as that in Figure 5.11. There we have nonzero values for all the neighbouring features, although it is certain that there is no such motion where all the neighboring sources at once would predict the given source. It is just the combination of all such elementary movements that produce that result.

5.5 Switching dynamics

It would be useful to be able to segment the different elementary movements described by the dynamics by some means. This would mean that we should have several different linear mappings which would take charge when data supporting their predictions would appear. Such a model can be formulated using mixture of Gaussians (MoG) variables controlled by a shared discrete variable.

Now, our linear mapping is no longer time-independent but varies over the time course. Assume that we have K different dynamics mappings $\mathbf{B}_1 \dots \mathbf{B}_K$ from which the active one is selected by a discrete variable $\lambda(t)$:

$$p(\mathbf{B}(t) \mid \{\mathbf{B}_i\}_{i=1}^{K}, \lambda(t) = k) = \mathcal{N}\left(\mathbf{B}(t) \mid \mathbf{B}_k, \exp[-\mathbf{v}_B]\right)$$

Now $\mathbf{B}_i$ only specify the mean for $\mathbf{B}(t)$. If the variance $\exp[-\mathbf{v}_B]$ is large, $\mathbf{B}(\mathrm{t})$ can have arbitrary values. We can limit this kind of behavior by making the prior of $\mathbf{v}_B$ favor big values, which means that the variance of $\mathbf{B}(t)$ is small. We want $\mathbf{B}(t)$ effectively to work as a switch between the different $\mathbf{B}_i$. This resembles the ideology which is used in HNFA for the hidden neurons [54].

The cpfs for the different means $\mathbf{B}_i$ of $\mathbf{B}(t)$ are

$$p(\mathbf{B}_i \mid \bar{\mathbf{B}}, \tilde{\mathbf{B}}) = \mathcal{N}\left(\mathbf{B}_i \mid \bar{\mathbf{B}}_i, \tilde{\mathbf{B}}_i\right)$$

Here the $\bar{\mathbf{B}}_i$ are chosen to represent some prototypical movements such as (5.9) and (5.10). To allow deviation from these prototypes the variances in $\tilde{\mathbf{B}}_i$ are set to equal one.

The states of the discrete variable have probabilities $\mathbf{c} = [c_1 \dots c_K]$. The probability vector $\mathbf{c}$ has a Dirichlet prior

$$p(\lambda(t) = k \mid \mathbf{c}) = c_k$$
$$p(\mathbf{c}) = \mathcal{D}\left(\mathbf{c} \mid \mathbf{u}\right)$$

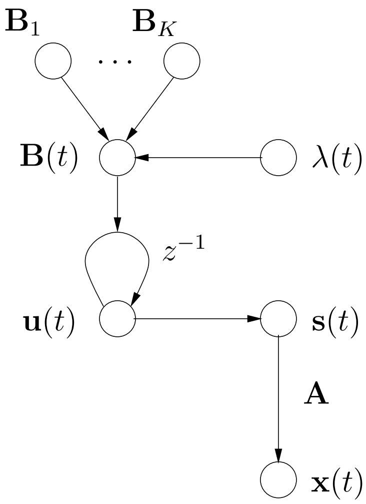


Figure 5.13: The static dynamics mapping $\mathbf{B}$ has been changed to a time-dependent one $\mathbf{B}(t)$ which effectively reflects the values of $\mathbf{B}_1 \dots \mathbf{B}_K$. The discrete variable $\lambda(t)$ controls which one of these is active at any given moment.

To keep the big picture in mind, we can again graphically summarize our model structure. See Figure 5.13. To test the proposed model structure we use a simple dataset containing bars moving first from right to left, then staying put, and then moving from left to right. Part of that dataset is shown in Figure 5.14. The number of possible dynamics is set to three, and the priors for the $\mathbf{B}_1 \dots \mathbf{B}_3$ are chosen to reflect the dataset. There are some technical issues concerning the learning of the current model. These are dealt with in detail in the next section. Now we can first test our model by initializing the discrete variable to the correct values and see if the model works with them. A part of the initial values are shown in Figure 5.15(a) and the values after learning are shown in Figure 5.15(b). The learnt values are similar in spirit to the initial and correct values, but the state of the discrete variable seems to change inappropriately every now and then. The reason for this is probably that there is not constantly enough evidence to choose the correct value for the discrete variable. When there is a lack of evidence, the system should be able to use past information to make the decision about the correct state. This suggests that we should make the cpf of $\lambda(t)$ dependent on the past values.

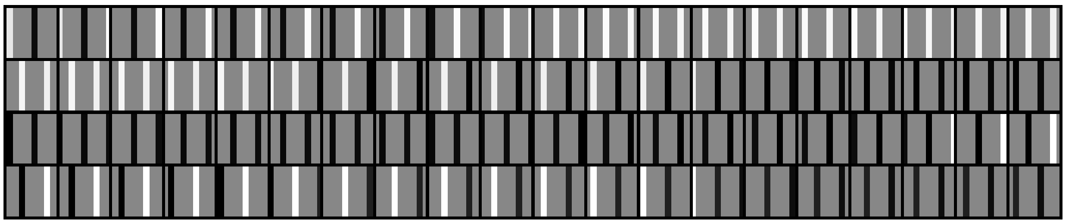


Figure 5.14: A part from the bars dataset. The sequence shown here contains bars moving from right to left.

5.6 Adding Markov prior for the discrete variable

We can improve the model of our discrete variable by making it dependent on its value at the previous time instance

$$p(\lambda(t) = i \mid \lambda(t-1) = j, \mathbf{C}) = c_{ij}$$

Here $\mathbf{C}$ is the transition probability matrix whose element c_{ij} gives us the probability that the next state will be i given that we are in state j. The column vectors $\{\mathbf{c}_j\}_{j=1}^{K}$ of the matrix $\mathbf{C}$ have to sum up to one. For them, we can use Dirichlet priors

$$p(\mathbf{c}_j \mid \mathbf{u}_j) = \mathcal{D}\left(\mathbf{c}_j \mid \mathbf{u}_j\right) \qquad \forall j = 1, \ldots, K$$

Let us try how this modification to our model affects the state change problem. We use the same bars dataset as before and initialize the discrete variable to the correct values. Looking at the values of the discrete variable after learning, Figure 5.16, we see that there are no inappropriate state changes. The mean of the transition probability matrix is

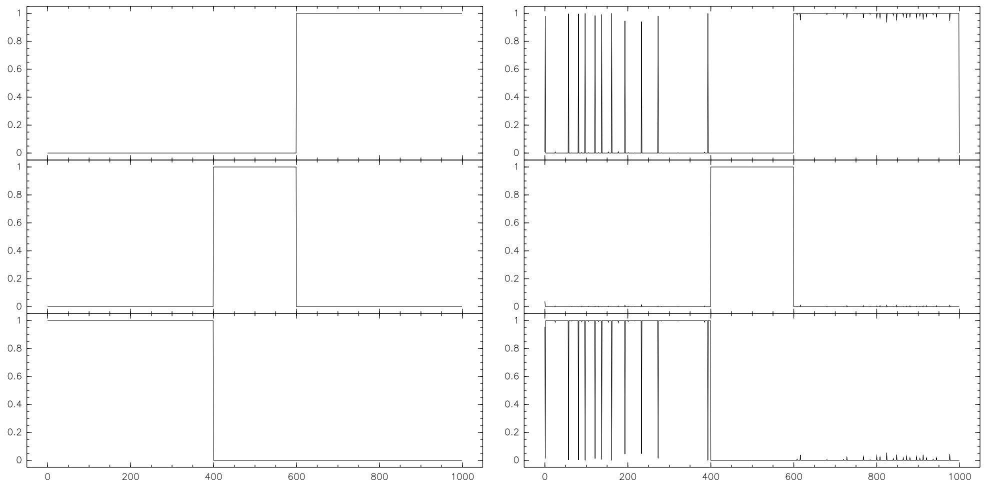
$$\langle \mathbf{C} \rangle = \begin{pmatrix} 0.9950 & 0.0099 & 0.0025 \\ 0.0025 & 0.9848 & 0.0050 \\ 0.0025 & 0.0053 & 0.9926 \end{pmatrix}$$

The probabilities for staying in the same state are very near to one. Now the model has the belief that changing the dynamics is rarely appropriate, so it doesn't switch the state unless there is a really good reason to do so.

5.7 The final model

Up to this point, in this chapter, only the model equations have been given. Nothing has been said about how the model is actually learnt. The basic learning approach has of course been already discussed in Chapter 4. This means that we have local update rules for all the variables and we update them cyclically. However, as it was discussed in Section 4.3 these local update rules might get us only to a very poor stationary point and the solution might be quite useless.

Now, having reached the topmost level of the model hierarchy, we can again take a look at our model in Figure 5.17, and consider what kind of things we



(a) Initial values (b) Learnt values

Figure 5.15: Initial values vs. values after learning.

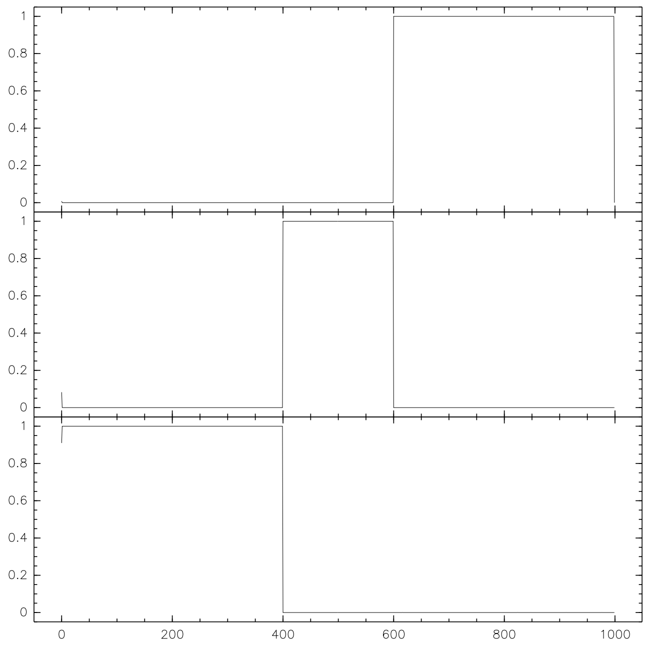


Figure 5.16: Learnt values when the discrete has Markov prior.

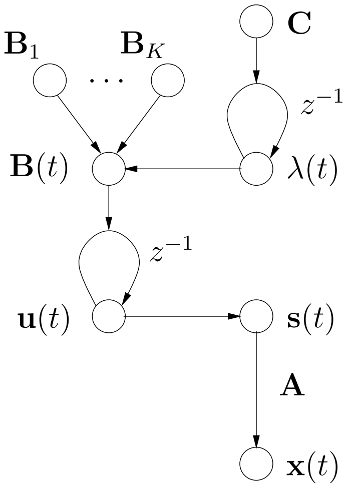


Figure 5.17: The final model structure. Starting from the bottom: we have a linear sparse model for the observations $\mathbf{x}(t)$. The variances $\mathbf{u}(t)$ of the sources $\mathbf{s}(t)$ have switching linear dynamics $\mathbf{B}(t)$ which is controlled by the discrete variable $\lambda(t)$. The discrete variable has Markov prior with the transition probabilities specified by the matrix $\mathbf{C}$.

need in addition to the local update rules to get our model to learn useful things. First of all, we need an initialization scheme for the sources. When a full linear mapping is used, the customary thing to do is to compute PCA sources from the data and use that as the starting point for learning. Now that the basis $\mathbf{A}$ is initially sparse, it does not make sense to compute PCA. Instead the instantaneous spatial average from the connection matrix area for all the sources is computed. To make this notion more clear, let us define the set of indicator variables $\{w_{ij}\}_{ij}$ such that $w_{ij} = 1$ if there is a connection from the source j to the observation i, and zero otherwise. Accordingly, the initialization for the source s_j can be computed as

$$s_j(t) = \frac{\sum_i w_{ij} x_i(t)}{\sum_i w_{ij}}$$

This initialization is a good starting point for learning, better than the PCA initialization at least.

Another critical point in the learning algorithm is the initialization of the discrete variable $\lambda(t)$. Without reasonable initializations, usually only one of the K linear mappings is used, which effectively means that the switching dynamics is pruned away altogether. In some cases this might be appropriate, the switching dynamics does not affect significantly enough to the representation of the data, that keeping it in the model would be worthwhile. In other cases, it is intuitively (and experimentally as it is shown later) clear that using several different dynamics mappings should really make the description of the data better.

In the following, an initialization scheme is developed for the discrete variable $\lambda(t)$. It is based on estimating $\mathbf{B}(t)$ and then fitting prototypical dynamics candidates to it. If the discrete variable in the model has strange values, also $\mathbf{B}(t)$ will be weird and hence we cannot base the estimation of $\lambda(t)$ on it. Instead, we go down to the variance neurons $\mathbf{u}(t)$ and use them to construct an estimate for $\mathbf{B}(t)$. Denote

$$\mathbf{U}(t) = \begin{bmatrix} u_1(t) & u_1(t+1) & \dots & u_1(t+L-1) \\ u_2(t) & u_2(t+1) & \dots & u_2(t+L-1) \\ \vdots & \vdots & \ddots & \vdots \\ u_n(t) & u_n(t+1) & \dots & u_n(t+L-1) \end{bmatrix}$$

where L is the length of the window. Assuming that the dynamical relations in $\mathbf{u}(t)$ can be characterized with some linear mapping $\mathbf{B}$, the identity $\mathbf{U}(t) = \mathbf{B}\mathbf{U}(t-1)$ holds. With known values for $\mathbf{U}(t)$, the dynamics can be estimated for each time instance t using the minimum mean square error solution

$$\mathbf{B}(t) = \mathbf{U}(t)\mathbf{U}^T(t-1)\left(\mathbf{U}(t-1)\mathbf{U}^T(t-1)\right)^{-1}$$

Now, if we have some candidates $\mathbf{C}_k$, $k = 1, \dots, K$ for the possible dynamics, we estimate the distribution of $\lambda(t)$ by fitting these candidates to the estimated $\mathbf{B}(t)$ and seeing which one is most appropriate at any given moment of time. The candidate mappings can be for example prototypes of movement from left to right, top to bottom etc.

Selecting a time window of right size L is critical. With L being too small the estimates for the discrete variable will be essentially noise. Choosing too big a L, on the other hand, can make shorter lasting states of dynamics disappear. Figure 5.18 shows an example of $q(\lambda(t))$ estimated with different values for L.

Using a heuristic initialization does not make sense unless there is some reliable way to compare the different initializations. The cost function in variational learning provides us with a tool to choose from different initializations. For example, when trying to choose the most suitable value of L, it turns out that the cost attains its minimum at a point which gives the best initial values in the sense that it corresponds to the true underlying motion. In Figure 5.19 the costs obtained after 5000 iterations with different initialization parameters L are plotted.

Now, having constructed a model and considered an appropriate learning algorithm for it, we can try applying it on a more realistic data. The data to be considered is generated from a static image of a forest scenery by sliding a 16×16 size window over it. To some extent, this corresponds to a camera movement in a video. The whole dataset consists of 1200 frames. Part of it is shown in Figure 5.20. There, also some context outside the actual data

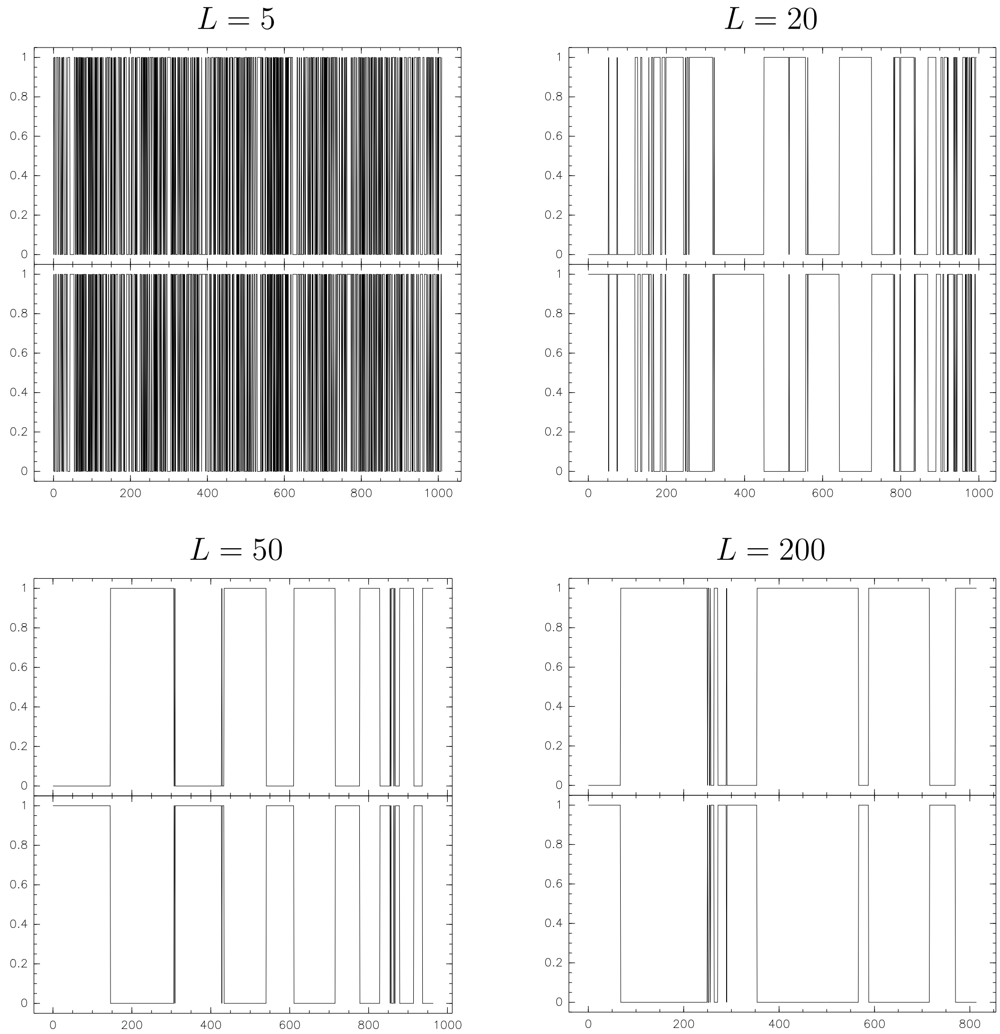


Figure 5.18: Estimates for the distribution $q(\lambda(t))$. The image sequence used as the data contained alternating camera movement from left to right and from right to left on a static scene. The initialization with $L = 50$ is nearest to truth. With $L = 20$ the initialization becomes noisier and with $L = 5$ it is completely useless. With $L = 200$ some of the shorter lasting motions are missed completely.

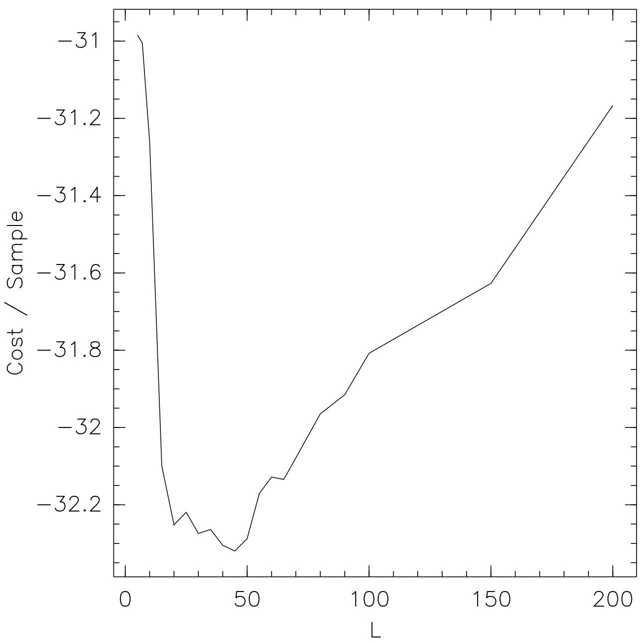


Figure 5.19: Cost per sample as a function of the initialization parameter L.

is shown to make it more clear what is happening. The real data is inside the white rectangle. To emphasize what the model actually sees, the same sequence of the real data is shown in Figure 5.21.

The priors for the five different dynamics mappings represented prototypical movement vertically and horizontally. One of these prototypes was the identity mapping for the situation where there is no motion. The initialization scheme for the discrete variable was similar to that described earlier. Again, careful tuning of the parameters of the initialization procedure was required to get reasonable results.

Let us finally look some parts of our model after learning. In Figure 5.22 there is a set of horizontally and vertically aligned sources and their respective variances (or more exactly the variances of their innovation processes). The top figures correspond to the horizontally aligned features. There we clearly see how the excitement of sources happens one after another when the tree goes through the frame. It is worth noting that there is not such behavior all the time. Looking at the discrete variable in Figure 5.23, however, we see that its state fairly well describes what is actually happening. This is due to the Markov prior which allows the model to predict the next value based on the earlier information whenever there is a lack of real evidence from the data. The left to right motion changes to top to bottom motion exactly at time

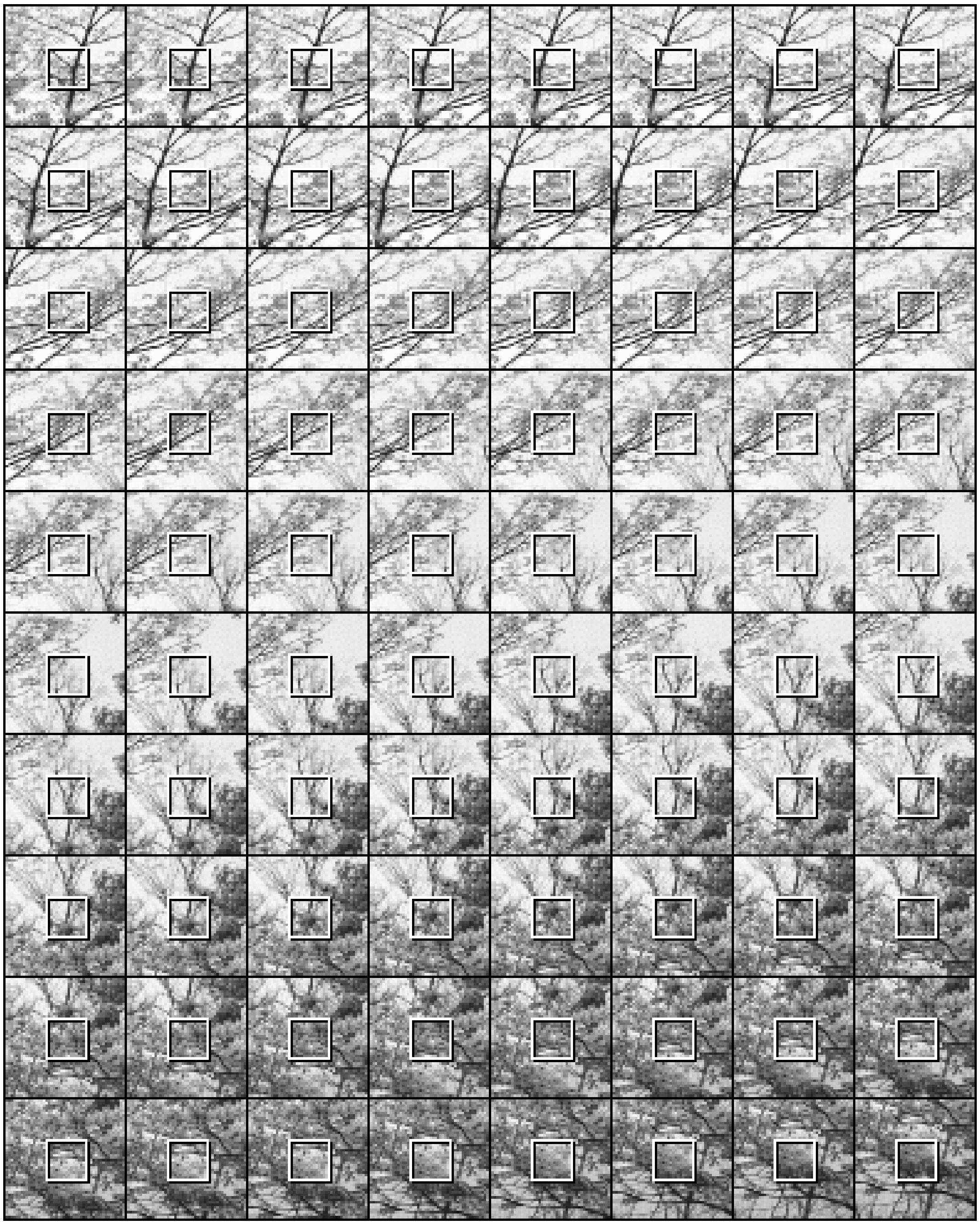


Figure 5.20: A subset of frames from the image sequence of a natural scene. Again, time flows row-wise starting from the top left corner. The white box shows the part of the frame that is actually used as the data. The direction of the motion changes in the middle of the sequence.

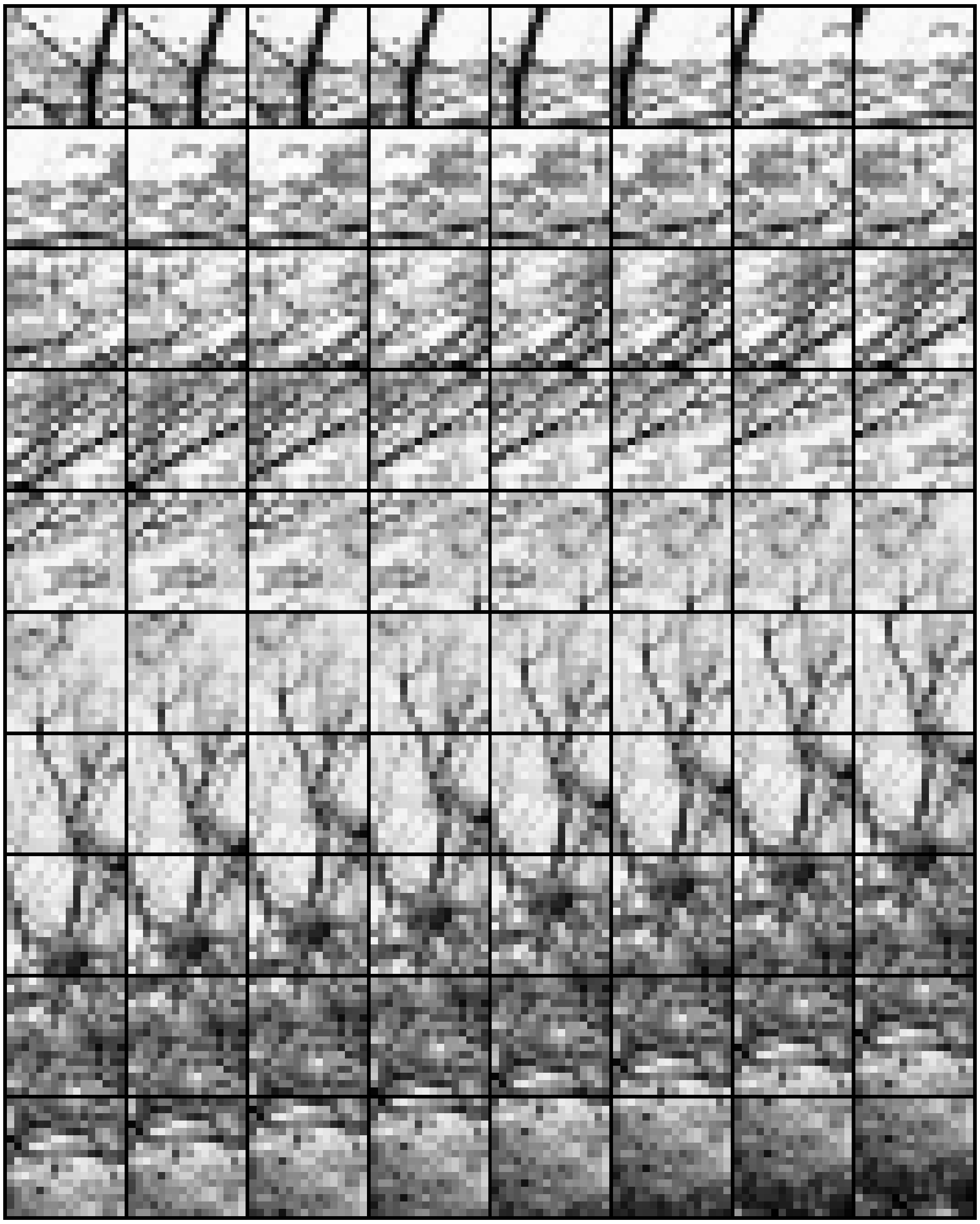


Figure 5.21: The same data as in Figure 5.20 but now without the context.

index 800 (the 40th frame in figures 5.20 and 5.21). The state of the discrete variable changes only after some twenty samples, near time index 820. This is understandable by looking at Figure 5.22(d) which shows the variances of a set of vertically aligned sources. There is not much happening until the index 820.

Finally, a brief summary of the development is appropriate. There are three fundamental points in the model that are now emphasized. Firstly, natural signals, such as image sequences, are well characterized by varying variance. In basic ICA this is seen as components, which are uncorrelated but by no means independent. As there are still some dependencies between those components,

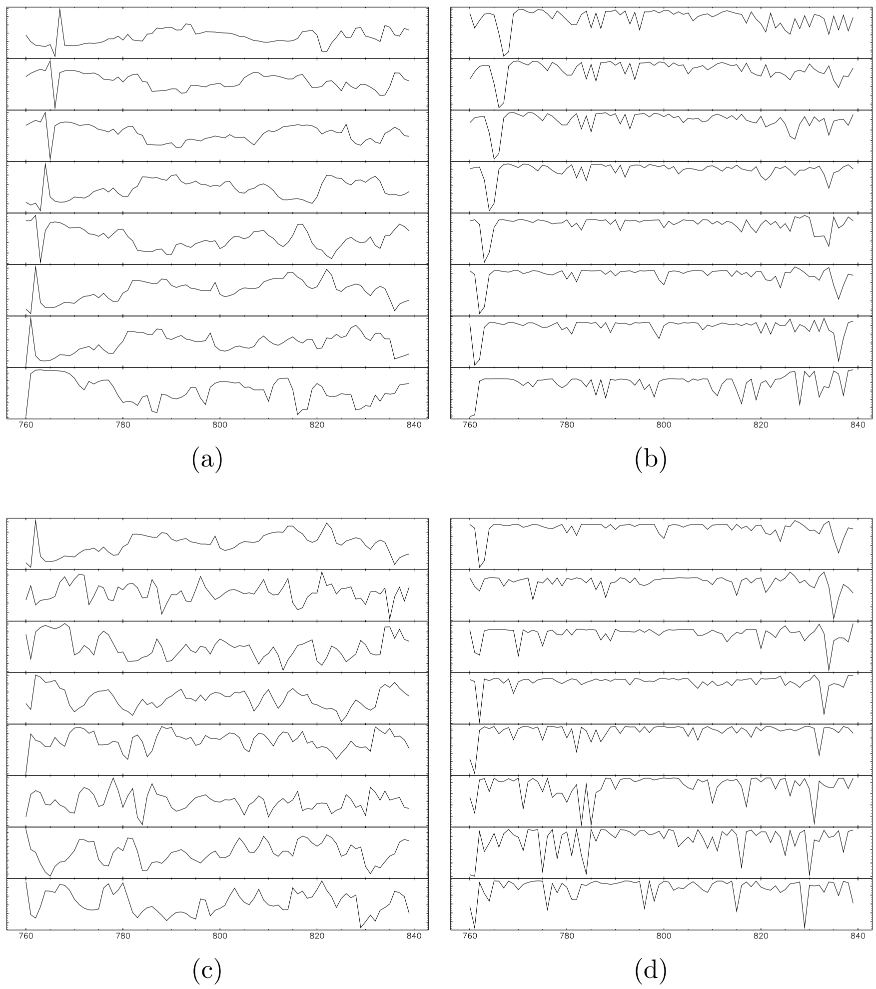


Figure 5.22: A set of horizontally aligned sources (a) and their variances (b) and a set of vertically aligned sources (c) and their variances (d).

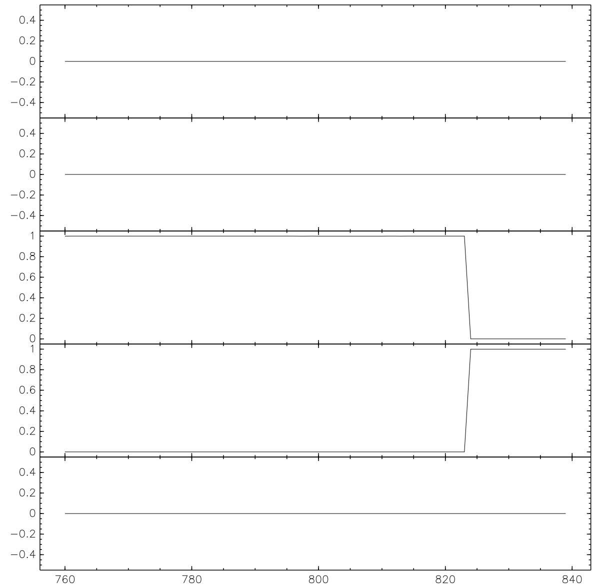


Figure 5.23: The distribution of the discrete variable. The third state correspond to horizontal right to left motion and the fourth to vertical top to bottom movement.

the question is whether modelling those could give us more information of the data. This was the motivation for concentrating on the variances of the sources instead of the sources themselves. Secondly, the variance neurons attached to the sources still do not give us very high level information. They just represent the activity of the sources, albeit in a conveniently invariant way. These simple features have to be somehow cleverly combined to obtain more abstract representations.

One of the models that was tried before the experiments reported in this thesis, contained a first layer similar to that of the model introduced here. There the dynamics of the variance neurons was not modelled, but instead another linear layer was used on top of those giving birth to another set of sources, so called *variance sources* (for details, see [51]). Although the variance sources could model the regularities in the variance neurons, the model contained no memory. This makes it impossible for example to model motion in image sequences. One can convince oneself of this by the fact that the reordering of the frames does not affect the results in any way. This gives rise to the third point: image sequence models without memory are necessarily poorer in their representation capability than models that do have memory.

Chapter 6

Discussion

In this thesis, the modelling of variances in image sequences was studied. The goal was to examine whether modelling of variances could be useful and if so, to what kind of specific model structure could this approach be applied.

It turned out that the variance of a source in the image sequence model could provide a better feature than the source itself. By better it is meant that building a model on top of the variance can be more fruitful, at least in some cases, than on top of the corresponding source.

The concrete model that was constructed in Chapter 5 was aimed to be able to extract features relating to different kind of movements in an image sequence. It was demonstrated by experiments with artificial data, that in principle the model can represent quite well the underlying movement of the objects and that it can do so in an invariant way. However, learning the model proved to be quite difficult.

The first and most obvious problem is the difficulty of local minima of the cost function. Although our cost function would support a certain model, the local update rules are usually not enough to get us there or even near any good solution. In this thesis, this problem was tried to be solved by introducing a heuristic and model specific initialization scheme. Then the cost function was used to select the best initialization meaning that the variational method was mainly used for model selection, not to learning, at least in the top of the model structure. Going through a large space of initializations might be feasible in smaller problems, but large problems, such as the ones dealing with real image sequences, cannot be solved in this manner.

One way to tackle the problem would be to develop a more global optimization approach. Although such a method cannot most probably be analytically derived for the model that was considered, it could be possible to use some kind of stochastic algorithm, which is less sensitive to local minima. Such a method could for example be simulated annealing [17]. It could be used to learn the discrete variable. After every update suggested by the simulated annealing algorithm, the local update rules could be applied a couple of times to get also the other parts of the model adjusted. By doing this sufficiently long in a relatively high pseudo temperature[1] we might find some good solution.

In addition to the local minima problem, there might be even some more fundamental problems concerning variational Bayesian learning and probabilistic hierarchical modelling in general. A phenomenon which has been encountered several times by the author while using variational Bayesian learning is automatic and sometimes counterintuitive pruning of some parts of the model. There have been some early studies by MacKay [39] where this problem is examined in an experimental setting. In the abstract of that paper MacKay writes

> Approximate inference by variational free energy minimization has maximum likelihood and maximum a posteriori methods as special cases, so we might hope that it can only work better than these standard methods. However, cases have been found in which degrees of freedom are pruned, perhaps inappropriately.

Overlearning is a serious problem in point estimation. Variational Bayesian learning solves that problem but could it be possible that in some cases it suffers from underlearning? Perhaps this is a subject that could be further studied.

One interpretation that has been offered to understanding the variational Bayesian learning is the coding interpretation [22]. There, learning is seen as minimization of the description length of the data, meaning that we are trying to find a model that explains the data as compactly as possible. This interpretation can give insight to several phenomena in learning. In this light, we can consider the kind of a very hierarchical model as the one constructed in Chapter 5, where there are several layers of variables on top of each other. Although by using the upper layer variables, we can code the lower layer variables better, the upper layer variables introduce an additional cost, because we also need to code them. Hence, it might be difficult to learn a very hierarchical model, if the lower level model already codes the data efficiently. This

[1]The higher the pseudo temperature in simulated annealing, the bigger steps upwards along the cost function are allowed, meaning that escaping from local minima is easier.

leads one to wonder whether the compact presentations are the ones we want when doing for example feature extraction. Perhaps less efficient models, in the sense of coding, can provide more useful features.

Bibliography

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

[2] H. Attias. ICA, graphical models and variational methods. In S. Roberts and R. Everson, editors, *Independent Component Analysis: Principles and Practice*, pages 95–112. Cambridge University Press, 2001.

[3] D. Barber and C. Bishop. Ensemble learning for multi-layer networks. In M. Jordan, M. Kearns, and S. Solla, editors, *Advances in Neural Information Processing Systems 10*, pages 395–401. The MIT Press, Cambridge, MA, USA, 1998.

[4] A. J. Bell and T. J. Sejnowski. The 'independent components' of natural scenes are edge filters. *Vision Research*, 37(23):3327–3338, 1997.

[5] P. Berkes and L. Wiskott. Applying slow feature analysis to image sequences yields a rich repertoire of complex cell properties. In *Proc. Int. Conf. Artifical Neural Networks (ICANN'02)*, pages 81–86, 2002.

[6] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. J. Wiley, 2000.

[7] T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31:307–327, 1986.

[8] T. Bollerslev, R. F. Engle, and D. B. Nelson. ARCH models. In R. F. Engle and D. McFadden, editors, *The handbook of econometrics, volume 4*, pages 2959–3038. North-Holland, Amsterdam, 1994.

[9] R. T. Cox. Probability, frequency and reasonable expectation. *American Journal of Physics*, 14(1):1–13, 1946.

[10] R. T. Cox. *The Algebra of probable inference*. The Johns Hopkins Press, Baltimore, Maryland, 1961.

[11] R. F. Engle. Autoregressive conditional heteroskedasticity with estimates of the variance of United Kingdom inflation. *Econometrica*, 50:987–1007, 1982.

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

[13] D. Gabor. Theory of communication. *Journal of the Institute of Electrical Engineers*, 93:429–549, 1946.

[14] A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC Press, Boca Raton, Florida, 1995.

[15] Z. Ghahramani and G. E. Hinton. Variational learning for switching state-space models. *Neural Computation*, 12(4):963–996, 2000.

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

[17] S. Haykin. *Neural Networks – A Comprehensive Foundation, 2nd ed.* Prentice-Hall, 1999.

[18] G. E. Hinton and D. van Camp. Keeping neural networks simple by minimizing the description length of the weights. In *Proc. of the 6th Ann. ACM Conf. on Computational Learning Theory*, pages 5–13, Santa Cruz, CA, USA, 1993.

[19] A. Honkela. Nonlinear switching state-space models. Master's thesis, Helsinki University of Technology, Espoo, 2001.

[20] A. Honkela. Speeding up cyclic update schemes by pattern searches. In *Proc. of the 9th Int. Conf. on Neural Information Processing (ICONIP'02)*, pages 512–516, Singapore, 2002.

[21] A. Honkela and H. Valpola. On-line variational Bayesian learning. In *Proc. of the 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 803–808, Nara, Japan, 2003.

[22] A. Honkela and H. Valpola. Variational learning and bits-back coding: an information-theoretic view to Bayesian learning. *IEEE Transactions on Neural Networks*, 2004. To appear.

[23] A. Honkela, H. Valpola, and J. Karhunen. Accelerating cyclic update algorithms for parameter estimation by pattern searches. *Neural Processing Letters*, 17(2):191–203, 2003.

[24] D. Hubel and T. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962.

[25] J. Hurri and A. Hyvärinen. Simple-cell like receptive fields maximize temporal coherence in natural video. *Neural Computation*, 15(3):663–691, 2003.

[26] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. J. Wiley, 2001.

[27] A. Hyvärinen and P. Hoyer. Emergence of phase- and shift-invariant features by decomposition of natural images into independent feature subspaces. *Neural Computation*, 12:1705–1720, 2000.

[28] A. Hyvärinen, P. Hoyer, and M. Inki. Topographic independent component analysis. *Neural Computation*, 13:1527–1558, 2001.

[29] A. Hyvärinen, J. Hurri, and J. Väyrynen. Bubbles: a unifying framework for low-level statistical properties of natural image sequences. *Journal of the optical society of America A*, 20(7):1237–1252, 2003.

[30] A. Ilin and H. Valpola. On the effect of the form of the posterior approximation in variational learning of ICA models. In *Proc. of the 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 915–920, Nara, Japan, 2003.

[31] E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, Cambridge, UK, 2003.

[32] I. T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 2nd edition, 2002.

[33] S. Kim, N. Shepard, and S. Chib. Stochastic volatility: likelihood inference and comparison with ARCH models. *The Review of Economic Studies*, 65(3):361–393, July 1998.

[34] H. Lappalainen. Ensemble learning for independent component analysis. In *Proc. Int. Workshop on Independent Component Analysis and Signal Separation (ICA'99)*, pages 7–12, Aussois, France, 1999.

[35] H. Lappalainen and A. Honkela. Bayesian nonlinear independent component analysis by multi-layer perceptrons. In M. Girolami, editor, *Advances in Independent Component Analysis*, pages 93–121. Springer-Verlag, Berlin, 2000.

[36] H. Lappalainen and J. Miskin. Ensemble learning. In M. Girolami, editor, *Advances in Independent Component Analysis*, pages 75–92. Springer-Verlag, Berlin, 2000.

[37] D. G. Luenberger. *Optimization by Vector Space Methods*. John Wiley & Sons, New York, 1969.

[38] D. J. C. MacKay. Developments in probabilistic modelling with neural networks – ensemble learning. In *Neural Networks: Artificial Intelligence and Industrial Applications. Proc. of the 3rd Annual Symposium on Neural Networks*, pages 191–198, 1995.

[39] D. J. C. MacKay. Local minima, symmetry-breaking, and model pruning in variational free energy minimization. Available online at `http://www.inference.phy.cam.ac.uk/mackay/BayesVar.html`, 2001.

[40] D. J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[41] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[42] J. Miskin and D. J. C. MacKay. Ensemble learning for blind source separation. In S. Roberts and R. Everson, editors, *Independent Component Analysis: Principles and Practice*, pages 209–233. Cambridge University Press, 2001.

[43] E. Oja and M. Plumbley. Blind separation of positive sources using non-negative PCA. In *Proc. 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 11–16, Nara, Japan, 2003.

[44] B. Olshausen and D. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[45] L. Parra, C. Spence, and P. Sajda. Higher-order statistical properties arising from the non-stationarity of natural signals. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 786–792, Cambridge, MA, USA, 2001. The MIT Press.

[46] W. Penny, R. Everson, and S. Roberts. ICA: model order selection and dynamic source models. In S. Roberts and R. Everson, editors, *Independent*

Component Analysis: Principles and Practice, pages 299–314. Cambridge University Press, 2001.

[47] T. Raiko. Hierarchical nonlinear factor analysis. Master's thesis, Helsinki University of Technology, Espoo, 2001.

[48] T. Raiko. Partially observed values. In *Proc. Int. Joint Conf. on Neural Networks (IJCNN'04)*, Budapest, 2004. Submitted.

[49] T. Raiko, H. Valpola, T. Östman, and J. Karhunen. Missing values in hierarchical nonlinear factor analysis. In *Proc. of the Int. Conf. on Artificial Neural Networks and Neural Information Processing - ICANN/ICONIP 2003*, pages 185–189, Istanbul, Turkey, 2003.

[50] H. Valpola, M. Harva, and J. Karhunen. Hierarchical models of variance sources. In *Proc. 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 83–88, Nara, Japan, 2003.

[51] H. Valpola, M. Harva, and J. Karhunen. Hierarchical models of variance sources. *Signal Processing*, 84(2):267–282, 2004.

[52] H. Valpola, A. Honkela, M. Harva, A. Ilin, T. Raiko, and T. Östman. Bayes blocks software library. *http://www.cis.hut.fi/projects/bayes/software/*, 2003.

[53] H. Valpola and J. Karhunen. An unsupervised ensemble learning method for nonlinear dynamic state-space models. *Neural Computation*, 14(11):2647–2692, 2002.

[54] H. Valpola, T. Östman, and J. Karhunen. Nonlinear independent factor analysis by hierarchical models. In *Proc. 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 257–262, Nara, Japan, 2003.

[55] H. Valpola, T. Raiko, and J. Karhunen. Building blocks for hierarchical latent variable models. In *Proc. 3rd Int. Conf. on Independent Component Analysis and Signal Separation (ICA2001)*, pages 710–715, San Diego, USA, 2001.

[56] J. H. van Hateren and D. L. Ruderman. Independent component analysis of natural image sequences yields spatio-temporal filters similar to simple cells in primary visual cortex. *Proceedings of the Royal Society of London B*, 265(1412):2315–2320, 1998.

[57] L. Wiskott and T. J. Sejnowski. Slow feature analysis: unsupervised learning of invariances. *Neural Computation*, 14:715–770, 2002.

Appendix A

Probability distributions

This appendix describes the probability distributions that are needed in the models of this thesis.

A.1 Gaussian

The Gaussian (or central, or normal) distribution is the most used distribution in statistics. The functional form of it is

$$\mathcal{N}(s \mid m, v) = \frac{1}{\sqrt{2\pi v}} \exp\left\{-\frac{1}{2v}(s-m)^2\right\}$$

where m is the mean and v the variance of the distribution. A detailed account on the ubiquitous use and equally ubiquitous success of the normal distribution is given in [31].

A.2 Rectified Gaussian

A rectified Gaussian distribution is a Gaussian distribution which is constrained to be positive. By defining the step function

$$u(s) = \begin{cases} 0 & \text{if } s < 0, \\ 1 & \text{if } s \geq 0 \end{cases}$$

the rectified Gaussian distribution can be expressed as

$$\mathcal{N}^R(s \mid m, v) = \frac{2}{\operatorname{erfc}(-m/\sqrt{2v})} \mathcal{N}(s \mid m, v)\, u(s) \tag{A.1}$$

where erfc is the complement of erf:

$$\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty \exp(-t^2)\, dt$$

Naturally, the parameters m and v are no longer the mean and variance of the distribution. The expectation have a bit more complex expressions and they are derived in Section B.1.

A.3 Mixture of Gaussians

As its name implies, the mixture of Gaussians (MoG) distribution is a composite of several Gaussian distributions. Concretely, this can be formulated by introducing a categorical variable $\lambda \in \{1, \ldots, K\}$ which selects one of the K Gaussian components $\{\mathcal{N}(s \mid m_i, v_i)\}_{i=1}^K$

$$p(s \mid \{m_i\}_{i=1}^K, \{v_i\}_{i=1}^K, \lambda = k) = \mathcal{N}(s \mid m_k, v_k)$$

This does not make (much) sense in this scalar case, but more so in the situation where s and λ are time-dependent

$$p(s(t) \mid m, v, \lambda(t) = k_t) = \mathcal{N}(s(t) \mid m_{k_t}, v_{k_t})$$

A.4 Dirichlet

The Dirichlet distribution is defined on the set $\{\mathbf{c} \in \mathbb{R}^K \mid \sum_{i=1}^n c_i = 1 \,\text{and}\, c_i \geq 0 \;\forall i\}$ as

$$\mathcal{D}(\mathbf{c} \mid \mathbf{u}) = \frac{1}{Z(\mathbf{u})} \prod_{i=1}^n c_i^{u_i - 1}$$

where u_is are the so called prior observation counts and Z is a normalization constant

$$Z(\mathbf{u}) = \frac{\prod_{i=1}^n \Gamma(u_i)}{\Gamma(\sum_{i=1}^n u_i)}$$

Appendix B

Derivations

This appendix contains the necessary derivations of cost functions and update rules for the new blocks in the Bayes Blocks library.

B.1 Rectified Gaussian Variable

Expectations

In this section expressions for the mean and the mean square of a rectified Gaussian distribution $\mathcal{N}^R(s \mid \bar{s}, \tilde{s})$ are derived.

The mean:

$$\begin{aligned}\langle s \rangle = \langle s - \bar{s} + \bar{s} \rangle &= \bar{s} + \int_{-\infty}^{\infty} (s - \bar{s}) \mathcal{N}^R(s \mid \bar{s}, \tilde{s})\, ds \\ &= \bar{s} + \frac{\sqrt{2}}{\sqrt{\pi \tilde{s}}\, \mathrm{erfc}(-\bar{s}/\sqrt{2\tilde{s}})} \int_0^{\infty} (s - \bar{s}) \exp\left(-\tfrac{1}{2\tilde{s}}(s - \bar{s})^2\right) ds \end{aligned} \tag{B.1}$$

The integral in (B.1) can be evaluated by performing a change of variable $z = (s - \bar{s})/\sqrt{2\tilde{s}}$

$$\begin{aligned}&\int_0^{\infty} (s - \bar{s}) \exp\left(-\tfrac{1}{2\tilde{s}}(s - \bar{s})^2\right) ds \\ &\qquad = 2\tilde{s} \int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} z \exp(-z^2) dz = \tilde{s} \exp\left(-(\bar{s}/\sqrt{2\tilde{s}})^2\right) \end{aligned} \tag{B.2}$$

By substituting (B.2) back to (B.1) we get

$$\langle s\rangle = \bar{s} + \sqrt{\frac{2\tilde{s}}{\pi}} \frac{1}{\exp\big((\bar{s}/\sqrt{2\tilde{s}})^2\big)\operatorname{erfc}\big(-\bar{s}/\sqrt{2\tilde{s}}\big)} \tag{B.3}$$

The mean square:

$$\begin{aligned}\langle s^2\rangle &= \int_{-\infty}^{\infty} s^2 \mathcal{N}^R(s \mid \bar{s}, \tilde{s})\, ds \\ &= \frac{\sqrt{2}}{\sqrt{\pi\tilde{s}}\operatorname{erfc}(-\bar{s}/\sqrt{2\tilde{s}})} \int_0^{\infty} s^2 \exp\left(-\tfrac{1}{2\tilde{s}}(s-\bar{s})^2\right) ds \end{aligned}\tag{B.4}$$

Again, doing the change of variable $z = (s-\bar{s})/\sqrt{2\tilde{s}}$

$$\begin{aligned}\int_0^{\infty} s^2 \exp\left(-\tfrac{1}{2\tilde{s}}(s-\bar{s})^2\right) ds &= \sqrt{2\tilde{s}}\int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} \left(\sqrt{2\tilde{s}}z+\bar{s}\right)^2 \exp(-z^2)dz \\ &= (2\tilde{s})^{3/2}\int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} z^2\exp(-z^2)dz + 4\tilde{s}\bar{s}\int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} z\exp(-z^2)dz \\ &\quad + \sqrt{2\tilde{s}}\bar{s}^2\int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty}\exp(-z^2)dz. \end{aligned}\tag{B.5}$$

The second and the third integral can be directly evaluated and they yield

$$\int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} z\exp(-z^2)dz = \frac{1}{2}\exp\big(-(\bar{s}/\sqrt{2\tilde{s}})^2\big), \tag{B.6}$$

$$\int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} \exp(-z^2)dz = \frac{\sqrt{\pi}}{2}\operatorname{erfc}\big(-\bar{s}/\sqrt{2\tilde{s}}\big). \tag{B.7}$$

The first integral can be integrated by parts

$$\begin{aligned}\int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} z^2\exp(-z^2)dz &= \int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} z\left[z\exp(-z^2)\right]dz \\ &= \left[-\frac{1}{2}z\exp(-z^2)\right]_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty} + \frac{1}{2}\int_{-\bar{s}/\sqrt{2\tilde{s}}}^{\infty}\exp(-z^2)dz \\ &= -\frac{\bar{s}}{2\sqrt{2\tilde{s}}}\exp\big(-(\bar{s}/\sqrt{2\tilde{s}})^2\big) + \frac{\sqrt{\pi}}{4}\operatorname{erfc}\big(-\bar{s}/\sqrt{2\tilde{s}}\big). \end{aligned}\tag{B.8}$$

Now, substituting the terms (B.6), (B.7) and (B.8) back to (B.5), substituting the result to (B.4) and finally simplifying, we get

$$\langle s^2\rangle = \bar{s}^2 + \tilde{s} + \sqrt{\frac{2\tilde{s}}{\pi}} \frac{\bar{s}}{\exp\big((\bar{s}/\sqrt{2\tilde{s}})^2\big)\operatorname{erfc}\big(-\bar{s}/\sqrt{2\tilde{s}}\big)}. \tag{B.9}$$

When $\bar{s}/\sqrt{2\tilde{s}} \ll 0$ we get into trouble using equations (B.3) and (B.9) due to the numerical issues concerning the erfc function. The following lemma is needed for the theorem that follows.

Lemma 1. *The* erfc *function can be approximated as*

$$\operatorname{erfc}(x) = \frac{\exp(-x^2)}{\sqrt{\pi}}\Big(x^{-1} + O(x^{-3})\Big)$$

Proof. We can integrate the function $\exp(-t^2)$ by parts by expressing it as $\frac{1}{t}t\exp(-t^2)$

$$\begin{aligned}
\int_x^\infty \exp(-t^2)\,dt &= \int_x^\infty \frac{1}{t}t\exp(-t^2)\,dt \\
&= -\frac{1}{2}\left[\frac{1}{t}\exp(-t^2)\right]_x^\infty - \frac{1}{2}\int_x^\infty \frac{1}{t^2}\exp(-t^2)\,dt \\
&= \frac{1}{2x}\exp(-x^2) - \frac{1}{2}\int_x^\infty \frac{1}{t^3}t\exp(-t^2)\,dt \\
&= \frac{1}{2x}\exp(-x^2) + \frac{1}{4}\left[\frac{1}{t^3}\exp(-t^2)\right]_x^\infty + \frac{3}{4}\int_x^\infty \frac{1}{t^4}t\exp(-t^2)\,dt \\
&= \exp(-x^2)\left(\frac{1}{2x} - \frac{1}{4x^3} + \dots\right) = \exp(-x^2)\left(\frac{1}{2x} + O(x^{-3})\right)
\end{aligned}$$

The result follows by multiplying with $2/\sqrt{\pi}$. $\square$

Now we state and prove a theorem which asserts that the rectified Gaussian distribution is well approximated with the exponential distribution in the case when $m/\sqrt{v} \ll 0$.

Theorem 1. *The rectified Gaussian distribution* $\mathcal{N}^R(s \mid m, v)$ *approaches the exponential distribution* $\mathcal{E}(s \mid -m/v)$ *when* $m/\sqrt{2v} \to -\infty$ *such that*

$$D\left(\mathcal{E}(s \mid -m/v) \,\|\, \mathcal{N}^R(s \mid m, v)\right) \xrightarrow[m/\sqrt{2v}\to-\infty]{} 0$$

Proof. First we write out the KL-divergence

$$D\left(\mathcal{E}(s \mid -m/v) \,\|\, \mathcal{N}^R(s \mid m, v)\right) = \int_{-\infty}^{\infty} \mathcal{E}(s \mid -m/v)\log\frac{\mathcal{E}(s \mid -m/v)}{\mathcal{N}^R(s \mid m, v)}\,ds \tag{B.10}$$

Now, substituting the functional forms of the distributions to (B.10) yields

$$\int_0^\infty -\frac{m}{v}\exp(ms/v)\log\frac{-\frac{m}{v}\exp(ms/v)}{\frac{2}{\mathrm{erfc}(-m/\sqrt{2v})}\frac{1}{\sqrt{2\pi v}}\exp(-\frac{1}{2v}(s-m)^2)}\,ds$$
$$= \log\Big\{-\frac{m}{\sqrt{2v}}\sqrt{\pi}\,\mathrm{erfc}(-m/\sqrt{2v})\exp(m^2/2v)\Big\} + \int_0^\infty -\frac{m}{v}\exp(ms/v)\frac{s^2}{2v}\,ds$$
$$= \log\Big\{-\frac{m}{\sqrt{2v}}\sqrt{\pi}\,\mathrm{erfc}(-m/\sqrt{2v})\exp(m^2/2v)\Big\} + \frac{v}{m^2}$$

The last term, v/m^2, approaches zero when $m^2/v \to \infty$. We can examine the asymptotic behaviour of the first term by invoking Lemma 1 and replacing the erfc function with its approximation. This yields

$$-\frac{m}{\sqrt{2v}}\sqrt{\pi}\,\mathrm{erfc}(-m/\sqrt{2v})\exp(m^2/2v)$$
$$= -\frac{m}{\sqrt{2v}}\left[\left(-\frac{m}{\sqrt{2v}}\right)^{-1} + O\left(\left(-\frac{m}{\sqrt{2v}}\right)^{-3}\right)\right]$$
$$= 1 + O\left(\left(\frac{m}{\sqrt{2v}}\right)^{-2}\right)$$

which approaches one when $m/\sqrt{2v} \to -\infty$ and hence the KL-divergence approaches zero. □

Using the result of Theorem 1, we can avoid the difficulties related to erfc function by calculating the expectations from the corresponding exponential distribution.

Cost function

In this section the two parts of the variational Bayesian learning cost function, $\mathcal{C}_s = \mathcal{C}_{s,p} + \mathcal{C}_{s,q}$, are evaluated for a rectified Gaussian variable whose posterior is approximated[1] using another rectified Gaussian distribution. The first term

[1] actually no approximation is needed since the rectified prior is conjugate to Gaussian likelihood

is:

$$
\begin{aligned}
\mathcal{C}_{s,p} &= -\left\langle \log p(s \mid m, v) \right\rangle \\
&= -\left\langle \log \mathcal{N}^R\left(s \mid m, \exp(-v)\right) \right\rangle \\
&= -\left\langle \log u(s) \right\rangle - \left\langle \log \mathcal{N}\left(s \mid m, \exp(-v)\right) \right\rangle \\
&\quad - \left\langle \log \left[\tfrac{1}{2} \operatorname{erfc}\left(-m/\sqrt{2\exp(-v)} \right) \right]^{-1} \right\rangle \\
&= \tfrac{1}{2} \left\{ \langle \exp v \rangle \left[(\langle s \rangle - \langle m \rangle)^2 + \mathrm{Var}\{s\} + \mathrm{Var}\{m\} \right] - \langle v \rangle + \log 2\pi \right\} \\
&\quad - \log 2 + \left\langle \log \operatorname{erfc}\left(-m/\sqrt{2\exp(-v)} \right) \right\rangle
\end{aligned} \tag{B.11}
$$

The last term in (B.11) can be evaluated only in the case when m is a constant equaling zero, i.e. $q(m) = \delta(m)$. If, for some reason, one wants to use non-constant prior mean, an upper bound for the term can be used which is $\log 2$. In this case the two last terms can be dropped out and the remaining cost function bounds the correct one from above. However, the update rule for the mean m is in this case only approximate. The second term is:

$$
\begin{aligned}
\mathcal{C}_{s,q} &= \left\langle \log q(s) \right\rangle \\
&= \left\langle \log \frac{\sqrt{2}\exp(-\frac{1}{2\tilde{s}}(s-\bar{s})^2)}{\sqrt{\pi\tilde{s}}\operatorname{erfc}\left(-\bar{s}/\sqrt{2\tilde{s}}\right)} \right\rangle \\
&= -\tfrac{1}{2\tilde{s}} \left\langle (s-\bar{s})^2 \right\rangle + \log\sqrt{2} - \log\left[\sqrt{\pi\tilde{s}}\operatorname{erfc}\left(-\bar{s}/\sqrt{2\tilde{s}}\right) \right]
\end{aligned}
$$

The term $\langle (s-\bar{s})^2 \rangle$ yields

$$
\begin{aligned}
\left\langle (s-\bar{s})^2 \right\rangle &= \left\langle s^2 - 2s\bar{s} + \bar{s}^2 \right\rangle = \left\langle s^2 \right\rangle - 2\bar{s}\langle s \rangle + \bar{s}^2 \\
&= \left\langle s^2 \right\rangle - \langle s \rangle^2 + \langle s \rangle^2 - 2\bar{s}\langle s \rangle + \bar{s}^2 = \mathrm{Var}\{s\} + (\langle s \rangle - \bar{s})^2
\end{aligned}
$$

Now we can write $\mathcal{C}_{s,q}$ out:

$$
\mathcal{C}_{s,q} = -\tfrac{1}{2\tilde{s}}[\mathrm{Var}\{s\} + (\langle s \rangle - \bar{s})^2] + \tfrac{1}{2}\log\tfrac{2}{\pi\tilde{s}} - \log\operatorname{erfc}\left(-\bar{s}/\sqrt{2\tilde{s}}\right)
$$

Update rule

Let us denote the likelihood arising from the children of s simply as $p(x \mid s) = \mathcal{N}(x \mid s, \sigma_x^2)$. With a very similar derivation as in (4.2)

$$
\left\langle \log \mathcal{N}^R\left(s \mid m_s, \exp[-v_s]\right) \right\rangle = \log \mathcal{N}^R\left(s \mid \langle m_s \rangle, \langle \exp v_s \rangle^{-1}\right) + C
$$

Hence the relevant part of the cost function (2.10) is

$$\mathcal{C}_s = \left\langle \log \frac{q(s)}{\mathcal{N}\left(x \mid s, \sigma_x^2\right) \mathcal{N}^R\left(s \mid \langle m_s \rangle, \langle \exp v_s \rangle^{-1}\right)} \right\rangle_{q(s)}$$

It is clear from the definition of the rectified Gaussian distribution (Eq. A.1) that it is a conjugate prior for the Gaussian likelihood. Consequently the optimal posterior $q(s)$ is also a rectified Gaussian distribution with parameters computed similarly as in the Gaussian case.

B.2 Mixture of Gaussians variable

Expectations

If we use a factorial posterior approximation $q(s, \lambda) = q(s)q(\lambda)$, $q(s)$ is a Gaussian and the expectations are directly the mean and the variance parameter of the distribution. In the non-factorial case, we need to marginalize over λ

$$q(s) = \sum_{i=1}^{K} q(s, \lambda = i) = \sum_{i=1}^{K} q(s \mid \lambda = i) q(\lambda = i) = \sum_{i=1}^{K} \pi_i \mathcal{N}\left(s \mid \bar{s}_i, \tilde{s}_i\right)$$

where $q(\lambda = i)$ is denoted as π_i. Now the mean is

$$\begin{aligned}\langle s \rangle = \int_{-\infty}^{\infty} s\, q(s)\, ds = \int_{-\infty}^{\infty} s \sum_{i=1}^{K} \pi_i \mathcal{N}\left(s \mid \bar{s}_i, \tilde{s}_i\right)\, ds \\ = \sum_{i=1}^{K} \pi_i \int_{-\infty}^{\infty} s\, \mathcal{N}\left(s \mid \bar{s}_i, \tilde{s}_i\right)\, ds = \sum_{i=1}^{K} \pi_i \bar{s}_i\end{aligned}$$

The mean square can be computed similarly

$$\left\langle s^2 \right\rangle = \sum_{i=1}^{K} \pi_i (\tilde{s}_i + \bar{s}_i^2)$$

Cost function

The part of the cost function affecting the mixture components is

$$
\begin{aligned}
\mathcal{C}_{s,p} &= -\left\langle \log p(s \mid m, v, \lambda) \right\rangle_{q(s,m,v,\lambda)} \\
&= \left\langle -\left\langle \log p(s \mid m, v, \lambda) \right\rangle_{q(s|\lambda)q(m,v)} \right\rangle_{q(\lambda)} \\
&= \sum_{i=1}^{K} q(\lambda = i)\big(-\left\langle \log p(s \mid m, v, \lambda = i) \right\rangle \big) \\
&= \sum_{i=1}^{K} q(\lambda = i)\big(-\left\langle \log \mathcal{N}\left(s \mid m_i, \exp[-v_i]\right) \right\rangle \big)
\end{aligned}
$$

So the likelihood seen by the i:th component is a Gaussian weighted by the probability $q(\lambda = i)$.

Now, depending on the form of the approximating posterior distribution $q(s, \lambda)$ we get different equations for the other term of the cost function $\mathcal{C}_{s,q}$.

Assuming a factorial q such that $q(s, \lambda) = q(s)q(\lambda)$ the optimal posterior of s is a Gaussian (shown to be true later on). Hence the other part of the cost function in this case is

$$
\mathcal{C}_{s,q} = \left\langle \log q(s) \right\rangle_{q(s)} = \left\langle \log \mathcal{N}\left(s \mid \bar{s}, \tilde{s}\right) \right\rangle_{q(s)} = -\tfrac{1}{2} \log 2\pi e \tilde{s}
$$

On the other hand, assuming a non factorial q the optimal posterior is a mixture of Gaussians and the cost function is

$$
\mathcal{C}_{s,q} = \sum_{i=1}^{K} q(\lambda = i) \left\langle \log q(s \mid \lambda = i) \right\rangle_{q(s|\lambda=i)}
$$

Update rule

Again, the update rule is dependent whether we assume a factorial q or not.

The factorial case

Now we assume that $q(s, \lambda) = q(s)q(\lambda)$. The term of concern is

$$
\left\langle \log p(s \mid m, v, \lambda) \right\rangle_{q(m,v,\lambda)}
$$

If this has a convenient form in the sense that the exponential of it is conjugate to Gaussian likelihood the updating will be easy. This indeed is true:

$$\begin{aligned}
&\langle \log p(s \mid m, v, \lambda) \rangle_{q(m,v,\lambda)} = \big\langle \langle \log p(s \mid m, v, \lambda) \rangle_{q(\lambda)} \big\rangle_{q(m,v)} \\
&= \Big\langle \sum_{i=1}^{K} \pi_i \log p(s \mid m, v, \lambda = i) \Big\rangle_{q(m,v)} = \sum_{i=1}^{K} \pi_i \langle \log p(s \mid m, v, \lambda = i) \rangle_{q(m,v)} \\
&\quad = \sum_{i=1}^{K} \pi_i \left\langle \log \mathcal{N}\left(s \mid m_i, e^{-v_i}\right) \right\rangle = \sum_{i=1}^{K} \pi_i \log \mathcal{N}\left(s \mid \langle m_i \rangle, \langle e^{v_i} \rangle^{-1}\right) + C \\
&= \sum_{i=1}^{K} \log \mathcal{N}\left(s \mid \langle m_i \rangle, \langle e^{v_i} \rangle^{-1}\right)^{\pi_i} + C = \log \prod_{i=1}^{K} \mathcal{N}\left(s \mid \langle m_i \rangle, \langle e^{v_i} \rangle^{-1}\right)^{\pi_i} + C
\end{aligned}$$

The product in the last equation is proportional to a Gaussian distribution $\mathcal{N}(s \mid \mu_s, \sigma_s^2)$, with parameter values

$$\sigma_s^2 = \Big(\sum_{i=1}^{K} \pi_i \langle \exp v_i \rangle \Big)^{-1} \qquad \text{and} \qquad \mu_s = \sigma_s^2 \Big(\sum_{i=1}^{K} \pi_i \langle \exp v_i \rangle \langle m_i \rangle \Big)$$

Assuming that the likelihood arising from the children is of the form $p(x \mid s) = \mathcal{N}(x \mid s, \sigma_x^2)$, the optimal posterior is easily obtained as $q(s) = \mathcal{N}(s \mid \bar{s}, \tilde{s})$, with

$$\tilde{s} = (1/\sigma_x^2 + 1/\sigma_s^2)^{-1} \qquad \text{and} \qquad \mu_s = \tilde{s}(x/\sigma_x^2 + \mu_s/\sigma_s^2)$$

The non-factorial case

When we do not neglect the posterior dependence between s and λ we get slightly different update rules. First of all, we can write $q(s, \lambda) = q(s \mid \lambda)\, q(\lambda)$. Now, keeping $q(\lambda)$ fixed

$$\begin{aligned}
\left\langle \log \frac{q(s,\lambda)}{p(x \mid s)\, p(s \mid m, v, \lambda)} \right\rangle_{q(s,\lambda)} &= \left\langle \log \frac{q(s \mid \lambda) q(\lambda)}{p(x \mid s)\, p(s \mid m, v, \lambda)} \right\rangle_{q(s|\lambda)q(\lambda)} \\
&= \left\langle \log \frac{q(s \mid \lambda)}{p(x \mid s)\, p(s \mid m, v, \lambda)} \right\rangle_{q(s|\lambda)q(\lambda)} + C \\
&= \sum_{i=1}^{K} \pi_i \left\langle \log \frac{q(s \mid \lambda = i)}{p(x \mid s)\, p(s \mid m, v, \lambda = i)} \right\rangle_{q(s|\lambda=i)} + C
\end{aligned}$$

which is of course minimized when

$$q(s \mid \lambda = i) \propto p(x \mid s) p(s \mid m, v, \lambda = i) \qquad \forall i = 1, \ldots, K$$

Now, again $p(x \mid s)$ and $p(s \mid m, v, \lambda = i)$ are Gaussians and hence the distributions $q(s \mid \lambda = i)$ are Gaussians too.

B.3 Categorical variable

There are three different categorical variables. The one with constant prior, the one with Dirichlet cpf and the one with Markov prior. In the following the constant case is covered. The derivations for the other two possibilities are very similar.

Cost function

The prior is $p(\lambda = i) = c_i, i = 1, \dots, K$. Lets denote $\pi_i = q(\lambda = i)$. The terms of the cost function are

$$\mathcal{C}_{\lambda,p} = -\left\langle \log p(\lambda) \right\rangle_{q(\lambda)} = -\sum_{i=1}^{K} \pi_i \log c_i, \qquad \text{and}$$
$$\mathcal{C}_{\lambda,q} = \left\langle \log q(\lambda) \right\rangle_{q(\lambda)} = \sum_{i=1}^{K} \pi_i \log \pi_i.$$

Update rule

The posterior of λ is naturally a discrete distribution. The part of the cost function to be optimized is

$$\left\langle \log \frac{q(s, \lambda)}{p(x \mid s) p(s \mid \lambda)} \right\rangle_{q(s,\lambda)}.$$

The prior $p(\lambda)$ is not included here because it is constant.

Now, assuming $q(s \mid \lambda)$ fixed $\forall \lambda$

$$\begin{aligned}
\left\langle \log \frac{q(s, \lambda)}{p(x \mid s) p(s \mid \lambda)} \right\rangle_{q(s,\lambda)} &= \left\langle \log \frac{q(s, \lambda)}{p(x \mid s) p(s \mid \lambda)} \right\rangle_{q(s\mid\lambda)q(\lambda)} \\
= \Big\langle \left\langle \log q(s \mid \lambda) \right\rangle_{q(s\mid\lambda)} - \left\langle \log p(x \mid s) \right\rangle_{q(s\mid\lambda)} &- \left\langle \log p(s \mid \lambda) \right\rangle_{q(s\mid\lambda)} + \log q(\lambda) \Big\rangle_{q(\lambda)} \\
= \left\langle C(\lambda) + \log q(\lambda) \right\rangle_{q(\lambda)} &= \left\langle \log \frac{q(\lambda)}{\exp(-C(\lambda))} \right\rangle_{q(\lambda)}.
\end{aligned}$$

Hence, $q(\lambda)$ is optimized by setting $q(\lambda) \propto \exp(-C(\lambda))$, where

$$C(\lambda) = \left\langle \log q(s \mid \lambda) \right\rangle_{q(s\mid\lambda)} - \left\langle \log p(x \mid s) \right\rangle_{q(s\mid\lambda)} - \left\langle \log p(s \mid \lambda) \right\rangle_{q(s\mid\lambda)}. \tag{B.12}$$

It is worth noting that a categorical variable can be a parent to more than one mixture nodes $s_1, \ldots, s_n$ if their posteriors are assumed to be conditionally independent given λ

$$q(s_1, \ldots, s_n, \lambda) = q(s_1, \ldots, s_n \mid \lambda)q(\lambda) = \prod_{i=1}^{n} q(s_i \mid \lambda)q(\lambda).$$

It is now easily seen that $C(\lambda)$ in Eq. (B.12) becomes

$$C(\lambda) = \sum_{i=1}^{n} \left[\langle \log q(s_i \mid \lambda) \rangle_{q(s_i|\lambda)} - \langle \log p(x_i \mid s_i) \rangle_{q(s_i|\lambda)} - \langle \log p(s_i \mid \lambda) \rangle_{q(s_i|\lambda)} \right].$$

As another note, when s and λ are assumed posteriorly independent, the only relevant term in $C(\lambda)$ is $- \langle \log p(s \mid \lambda) \rangle_{q(s)}$.

B.4 Dirichlet variable

Expectations

Denoting $u_0 = \sum_{i=1}^{K} u_i$, the mean and variance can be expressed as

$$\langle c_i \rangle = \frac{u_i}{u_0} \quad \text{and} \quad \mathrm{Var}\{c_i\} = \frac{u_i(u_0 - u_i)}{u_0^2(u_0 + 1)}$$

The discrete variable, whose parent the Dirichlet variable is, also needs the expectation over the logarithm of c_i. This is

$$\langle \log c_i \rangle = \psi(u_i) - \psi(u_0)$$

where $\psi(x) = \frac{d}{dx} \log \Gamma(x)$ is the digamma function. For more detailed account on how this is derived, see e.g. [19].

Cost function

The posterior of $\mathbf{c}$ is also a Dirichlet distribution: $q(\mathbf{c}) = \mathcal{D}(\mathbf{c} \mid \mathbf{v})$. Let us again denote

$$u_0 = \sum_{i=1}^{K} u_i \quad \text{and} \quad v_0 = \sum_{i=1}^{K} v_i$$

The two terms of the cost function, $\mathcal{C}_p$ and $\mathcal{C}_q$ are rather similar

$$\mathcal{C}_p = -\left\langle \log p(\mathbf{c}) \right\rangle_{q(\mathbf{c})} = \log Z(\mathbf{u}) - \sum_{i=1}^{K} (u_i - 1)(\psi(v_i) - \psi(v_0))$$
$$\mathcal{C}_q = \left\langle \log q(\mathbf{c}) \right\rangle_{q(\mathbf{c})} = -\log Z(\mathbf{v}) + \sum_{i=1}^{K} (v_i - 1)(\psi(v_i) - \psi(v_0))$$

Update rule

First, assume that the cpf of the discrete variable is $p(\lambda = i \mid \mathbf{c}) = c_i$ Assuming $q(\lambda, \mathbf{c}) = q(\lambda)q(\mathbf{c})$ we can write

$$\left\langle \log \frac{q(\mathbf{c})}{p(\lambda \mid \mathbf{c})p(\mathbf{c})} \right\rangle_{q(\lambda)q(\mathbf{c})} = \left\langle \log q(\mathbf{c}) - \left\langle \log p(\lambda \mid \mathbf{c}) \right\rangle_{q(\lambda)} - \log p(\mathbf{c}) \right\rangle_{q(\mathbf{c})}$$

Computing the expectation over $\log p(\lambda \mid \mathbf{c})$

$$\begin{aligned}\left\langle \log p(\lambda \mid \mathbf{c}) \right\rangle_{q(\lambda)} &= \sum_{i=1}^{K} \pi_i \log p(\lambda = i \mid \mathbf{c}) = \sum_{i=1}^{K} \pi_i \log c_i \\ &= \sum_{i=1}^{K} \log c_i^{\pi_i} = \log \prod_{i=1}^{K} c_i^{\pi_i} \propto \log \mathcal{D}\left(\mathbf{c} \mid \boldsymbol{\pi}\right)\end{aligned}$$

Substituting this back we get $D\left(q(\mathbf{c}) \,\|\mathcal{D}\left(\mathbf{c} \mid \boldsymbol{\pi}\right) \mathcal{D}\left(\mathbf{c} \mid \mathbf{u}\right)\right)$ and hence the cost function is optimized by setting $q(\mathbf{c}) \propto \mathcal{D}\left(\mathbf{c} \mid \boldsymbol{\pi}\right) \mathcal{D}\left(\mathbf{c} \mid \mathbf{u}\right)$ which further simplifies to $q(\mathbf{c}) = \mathcal{D}\left(\mathbf{c} \mid \mathbf{u} + \boldsymbol{\pi}\right)$. In the concrete case where λ is time dependent we can compute similarly to obtain $q(\mathbf{c}) = \mathcal{D}\left(\mathbf{c} \mid \mathbf{u} + \sum_t \boldsymbol{\pi}(t)\right)$.

Another possibility for the cpf of the discrete variable is

$$p(\lambda(t) = i \mid \lambda(t-1) = j, \mathbf{C}) = c_{ij}$$

For each column $\mathbf{c}_j$ of the transition probability matrix $\mathbf{C}$, there is a Dirichlet prior

$$p(\mathbf{c}_j) = \mathcal{D}\left(\mathbf{c}_j \mid \mathbf{u}_j\right)$$

It can be shown, with a similar derivation as above, that the optimal $q(\mathbf{c}_j)$ is

$$q(\mathbf{c}_j) = \mathcal{D}\left(\mathbf{c}_j \mid [u_{ij} + \textstyle\sum_t \pi_i(t)\pi_j(t-1)]_{i=1}^{K}\right)$$