Chapter 3

Variational Bayesian learning of generative models

Harri Valpola, Antti Honkela, Alexander Ilin, Tapani Raiko, Markus Harva, Tomas Östman, Juha Karhunen, Erkki Oja

3.1 Bayesian modeling and variational learning

Unsupervised learning methods are often based on a generative approach where the goal is to find a model which explains how the observations were generated. It is assumed that there exist certain source signals (also called factors, latent or hidden variables, or hidden causes) which have generated the observed data through an unknown mapping. The goal of generative learning is to identify both the source signals and the unknown generative mapping.

The success of a specific model depends on how well it captures the structure of the phenomena underlying the observations. Various linear models have been popular, because their mathematical treatment is fairly easy. However, in many realistic cases the observations have been generated by a nonlinear process. Unsupervised learning of a nonlinear model is a challenging task, because it is typically computationally much more demanding than for linear models, and flexible models require strong regularization.

In Bayesian data analysis and estimation methods, all the uncertain quantities are modeled in terms of their joint probability distribution. The key principle is to construct the joint posterior distribution for all the unknown quantities in a model, given the data sample. This posterior distribution contains all the relevant information on the parameters to be estimated in parametric models, or the predictions in non-parametric prediction or classification tasks [1].

Denote by $\mathcal{H}$ the particular model under consideration, and by $\boldsymbol{\theta}$ the set of model parameters that we wish to infer from a given data set X. The posterior probability density $p(\boldsymbol{\theta}|X, \mathcal{H})$ of the parameters given the data X and the model $\mathcal{H}$ can be computed from the Bayes' rule

$$p(\boldsymbol{\theta}|X, \mathcal{H}) = \frac{p(X|\boldsymbol{\theta}, \mathcal{H})p(\boldsymbol{\theta}|\mathcal{H})}{p(X|\mathcal{H})} \tag{3.1}$$

Here $p(X|\boldsymbol{\theta}, \mathcal{H})$ is the likelihood of the parameters $\boldsymbol{\theta}$, $p(\boldsymbol{\theta}|\mathcal{H})$ is the prior pdf of the parameters, and $p(X|\mathcal{H})$ is a normalizing constant. The term $\mathcal{H}$ denotes all the assumptions made in defining the model, such as choice of a multilayer perceptron (MLP) network, specific noise model, etc.

The parameters $\boldsymbol{\theta}$ of a particular model $\mathcal{H}_i$ are often estimated by seeking the peak value of a probability distribution. The non-Bayesian maximum likelihood (ML) method uses to this end the distribution $p(X|\boldsymbol{\theta}, \mathcal{H})$ of the data, and the Bayesian maximum a posteriori (MAP) method finds the parameter values that maximize the posterior probability density $p(\boldsymbol{\theta}|X, \mathcal{H})$. However, using point estimates provided by the ML or MAP methods is often problematic, because the model order estimation and overfitting (choosing too complicated a model for the given data) are severe problems [1].

Instead of searching for some point estimates, the correct Bayesian procedure is to use all possible models to evaluate predictions and weight them by the respective posterior probabilities of the models. This means that the predictions will be sensitive to regions where the probability mass is large instead of being sensitive to high values of the probability density [2]. This procedure solves optimally the issues related to the model complexity and choice of a specific model $\mathcal{H}_i$ among several candidates. In practice, however, the differences between the probabilities of candidate model structures are often very large, and hence it is sufficient to select the most probable model and use the estimates or predictions given by it.

A problem with fully Bayesian estimation is that the posterior distribution (3.1) has a highly complicated form except for in the simplest problems. Therefore it is too difficult to handle exactly, and some approximative method must be used. Variational methods form a class of approximations where the exact posterior is approximated with a simpler

distribution. We use a particular variational method known as ensemble learning [3, 4] that has recently become very popular because of its good properties. In ensemble learning, the misfit of the approximation is measured by the Kullback-Leibler (KL) divergence between two probability distributions $q(v)$ and $p(v)$. The KL divergence is defined by

$$D(q \parallel p) = \int q(v) \ln \frac{q(v)}{p(v)} dv \tag{3.2}$$

which measures the difference in the probability mass between the densities $q(v)$ and $p(v)$.

A key idea in ensemble learning is to minimize the misfit between the actual posterior pdf and its parametric approximation using the KL divergence. The approximating density is often taken a diagonal multivariate Gaussian density, because the computations become then tractable. Even this crude approximation is adequate for finding the region where the mass of the actual posterior density is concentrated. The mean values of the Gaussian approximation provide reasonably good point estimates of the unknown parameters, and the respective variances measure the reliability of these estimates.

A main motivation of using ensemble learning is that it avoids overfitting which would be a difficult problem if ML or MAP estimates were used. Ensemble learning allows one to select a model having appropriate complexity, making often possible to infer the correct number of sources or latent variables. It has provided good estimation results in the very difficult unsupervised (blind) learning problems that we have considered.

Ensemble learning is closely related to information theoretic approaches which minimize the description length of the data, because the description length is defined to be the negative logarithm of the probability. Minimal description length thus means maximal probability. In the probabilistic framework, we try to find the sources or factors and the nonlinear mapping which most probably correspond to the observed data. In the information theoretic framework, this corresponds to finding the sources and the mapping that can generate the observed data and have the minimum total complexity. Ensemble learning was originally derived from information theoretic point of view in [3]. The information theoretic view also provides insights to many aspects of learning and helps explain several common problems [5].

In the following subsections, we first present some recent theoretical improvements to ensemble learning methods and a practical building block framework that can be used to easily construct new models. After this we discuss practical models for nonlinear static and dynamic blind source separation as well as hierarchical modeling of variances. Finally we present applications of the developed Bayesian methods to inferring missing values from data and to detection of changes in process states.

References

[1] C. Bishop, *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

[2] H. Lappalainen and J. Miskin. Ensemble Learning. In M. Girolami, editor, *Advances in Independent Component Analysis*, Springer, 2000, pages 75–92.

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

[4] D. 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, Nijmegen, Netherlands, 1995.

[5] 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.

3.2 Theoretical improvements

Using pattern searches to speed up learning

The parameters of a latent variable model used in unsupervised learning can usually be divided into two sets: the latent variables or sources $\boldsymbol{S}$, and other model parameters $\boldsymbol{\theta}$. The learning algorithms used in variational Bayesian learning of these models have traditionally been variants of the expectation maximization (EM) algorithm, which is based on alternatively estimating $\boldsymbol{S}$ and $\boldsymbol{\theta}$ given the present estimate of the other.

The standard update algorithm can be very slow in case of low noise, because the updates needed for $\boldsymbol{S}$ and $\boldsymbol{\theta}$ are strongly correlated. Therefore one set can be changed very little only while the other is kept fixed in order to preserve the reconstruction of the data. So-called pattern searches, which use the combined direction of a round of standard updates and then perform a line search in this direction, can help to avoid this problem [1].

The effect of using pattern searches is demonstrated in Figure 3.1 which shows the speedups attained in experiments with hierarchical nonlinear factor analysis (HNFA) (see Sec. 3.4) in different phases of learning. As the nonlinear model is susceptible to local minima, the different algorithms do not always converge to the same point. So the comparison was made by looking at the times required by the methods to reach a certain level of the cost function value above the worst local minimum found by the two algorithms.

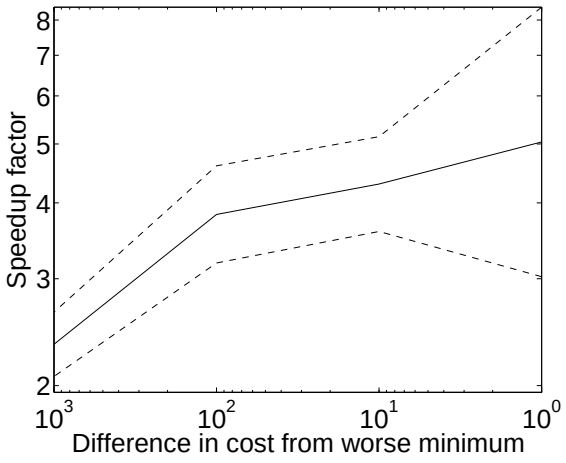

Figure 3.1: The average speedup obtained by pattern searches in different phases of learning. The speedup is measured by the ratio of times required by the basic algorithm and pattern search method to reach certain level of cost function value. The solid line shows the mean of the speedups over 20 simulation with different initializations and the dashed lines show 99 % confidence intervals for the mean.

Effect of posterior approximation

Most applications of ensemble learning to ICA models reported in the literature assume a fully factorized posterior approximation $q(v)$, because this usually results in a computationally efficient learning algorithm. However, the simplicity of the posterior approximation does not allow to represent all different solutions, which may greatly affect the found solution.

Our recent paper [2] shows that neglecting the posterior correlations of the sources in $q(\boldsymbol{S})$ introduces a bias in favor of principal component analysis (PCA) solution. By the

PCA solution we mean the solution which has an orthogonal mixing matrix. Nevertheless, if the true mixing matrix is close to orthogonal and the source model is strongly in favor of the desirable ICA solution, the optimal solution can be expected to be close to the ICA solution.

Figure 3.2 illustrates this general trade-off of variational Bayesian learning between the misfit of the posterior approximation and the accuracy of the model. According to our assumption, the sources can be accurately modeled in the ICA solution. Therefore, the cost of inaccurate assumption would increase towards the ICA solution as shown with the dashed line on the second plot of Fig. 3.2. On the other hand, if the true mixing matrix is not orthogonal, the optimal posterior covariance of the sources could look like the one in the upper plot of Fig. 3.2. Then, the misfit of the posterior approximation of the sources is minimized in the PCA solution where the true posterior covariance would be diagonal.

In [2], we considered a linear dynamic ICA model but the analysis extends to nonlinear mixtures and non-Gaussian source models as well.

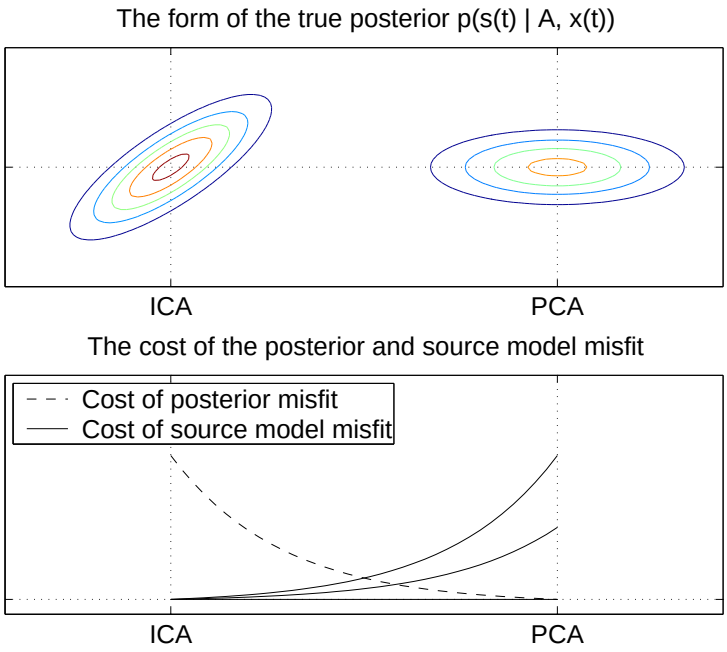

Figure 3.2: Schematic illustration of the trade-offs between the ICA and PCA solutions. In the PCA solution, the posterior covariance of the sources is diagonal. This minimizes the misfit between the optimal posterior and its approximation. However, the sources are explained better in the ICA solution.

References

[1] 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.

[2] 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.

3.3 Building blocks for variational Bayesian learning

In graphical models, there are lots of possibilities to build the model structure that defines the dependencies between the parameters and the data. To be able to manage the variety, we have designed a modular software package using C++/Python called the Bayes Blocks [1]. The theoretical background on which it is based on, was published in [2].

The design principles for Bayes Blocks have been the following. Firstly, we use standardized building blocks that can be connected rather freely and can be learned with local learning rules, i.e. each block only needs to communicate with its neighbors. Secondly, the system should work with very large scale models. We have made the computational complexity linear with respect to the number of data samples and connections in the model.

The building blocks include Gaussian variables, summation, multiplication, and nonlinearity. Each of them can be a scalar or a vector. Variational Bayesian learning provides a cost function which can be used for updating the variables as well as optimizing the model structure. The derivation of the cost function and learning rules is automatic which means that the user only needs to define the connections between the blocks.

Figure 3.3 shows an example of a structure which can be built using the Bayes Blocks library. More structures can be found in [2, 3], and their application to hierarchical modeling of variances is described in Sec. 3.5.

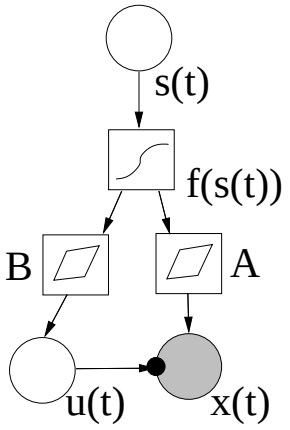

Figure 3.3: An example of a structure that can be built using Bayes Blocks. It includes latent variables $s(t)$, $u(t)$ (and some parameters), observed variables $x(t)$, a nonlinearity $f(\cdot)$, and affine transformations A and B. The variables $u(t)$ model the variances of $x(t)$.

References

[1] H. Valpola, A. Honkela, M. Harva, A. Ilin, T. Raiko, T. Östman. Bayes Blocks software library. `http://www.cis.hut.fi/projects/bayes/software/`, 2003.

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

[3] Harri Valpola, Markus Harva, and Juha Karhunen. Hierarchical models of variance sources. *Signal Processing*, 84(2):267–282, 2004.

3.4 Nonlinear static and dynamic blind source separation

The linear principal and independent component analysis (PCA and ICA, respectively) [1] model the data so that it has been generated by sources through a linear mapping. PCA looks for uncorrelated sources, restricting the directions of the sources to be mutually orthogonal. On the other hand, ICA requires that the sources are statistically independent which is a stronger assumption than uncorrelatedness, but there is no orthogonality restriction. In general, PCA is sufficient for Gaussian sources only, because it does not exploit higher than second-order statistics in any way [1].

We have applied variational Bayesian learning to nonlinear counterparts of PCA and ICA where the generative mapping from sources to data is not restricted to be linear. The general form of the model is

$$\mathbf{x}(t) = \mathbf{f}(\mathbf{s}(t), \boldsymbol{\theta}_f) + \mathbf{n}(t)\,. \tag{3.3}$$

This can be viewed as a model about how the observations were generated from the sources. The vectors $\mathbf{x}(t)$ are observations at time t, $\mathbf{s}(t)$ are the sources, and $\mathbf{n}(t)$ the noise. The function $\mathbf{f}(\cdot)$ is a mapping from source space to observation space parametrized by $\boldsymbol{\theta}_f$.

Nonlinear ICA by multi-layer perceptrons

In an earlier work [2, 3] we have used multi-layer perceptron (MLP) network with tanh-nonlinearities to model the mapping $\mathbf{f}$:

$$\mathbf{f}(\mathbf{s}; \mathbf{A}, \mathbf{B}, \mathbf{a}, \mathbf{b}) = \mathbf{B}\tanh(\mathbf{A}\mathbf{s} + \mathbf{a}) + \mathbf{b}\,. \tag{3.4}$$

The mapping $\mathbf{f}$ is thus parametrized by the matrices $\mathbf{A}$ and $\mathbf{B}$ and bias vectors $\mathbf{a}$ and $\mathbf{b}$. MLP networks are well suited for nonlinear PCA and ICA. First, they are universal function approximators which means that any type of nonlinearity can be modeled by them in principle. Second, it is easy to model smooth, close to linear mappings with them. This makes it possible to learn high dimensional nonlinear representations in practice.

Traditionally MLP networks have been used for supervised learning where both the inputs and the desired outputs are known. Here sources correspond to inputs and observations correspond to desired outputs. The sources are unknown and therefore learning is unsupervised.

Usually the linear PCA and ICA models do not have an explicit noise term $\mathbf{n}(t)$ and the model is thus simply $\mathbf{x}(t) = \mathbf{f}(\mathbf{s}(t)) = \mathbf{A}\mathbf{s}(t) + \mathbf{a}$, where $\mathbf{A}$ is a mixing matrix and $\mathbf{a}$ is a bias vector. The corresponding PCA and ICA models which include the noise term are often called factor analysis and independent factor analysis (FA and IFA) models. The nonlinear models discussed here can therefore also be called nonlinear factor analysis and nonlinear independent factor analysis models.

Hierarchical nonlinear factor analysis

The computational complexity of the variational Bayesian learning algorithm for the MLP network model is quadratic with respect to the number of sources in the model. To avoid this problem, an alternative hierarchical structure based on the build block approach presented in Section 3.3 was studied in [4]. One of the building blocks is a Gaussian variable ξ followed by a nonlinearity ϕ:

$$\phi(\xi) = \exp(-\xi^2)\,. \tag{3.5}$$

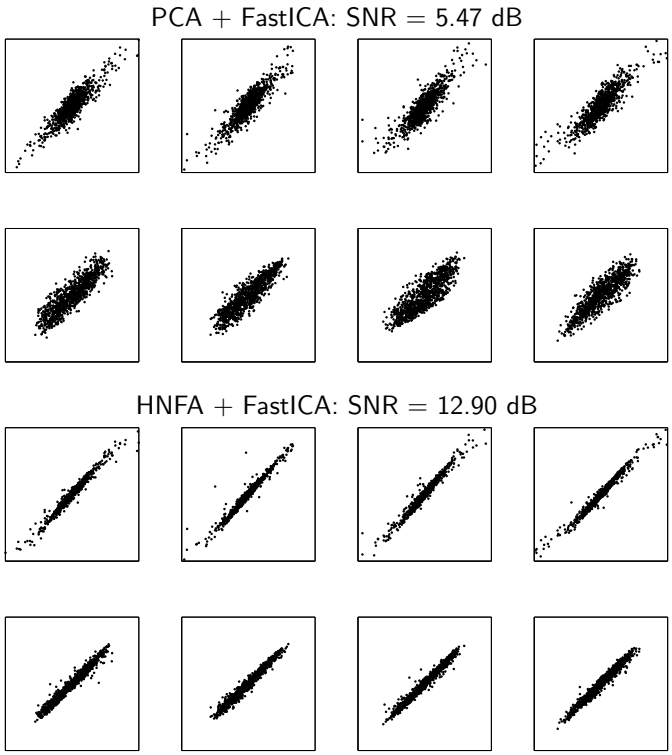

Figure 3.4: Each scatter plot shows the values of one original source signal plotted against the best corresponding estimated source signal after a rotation with FastICA.

The motivation for choosing this particular nonlinearity is that for Gaussian posterior approximation $q_\xi(\xi)$, the posterior mean and variance and consequently the cost function can be evaluated analytically.

Using this construction—Gaussian variables followed by nonlinearity—it is possible to build nonlinear mappings for which the learning time is linear with respect to the size of the model. The key idea is to introduce latent variables $\mathbf{h}(t)$ before the nonlinearities and thus split the mapping Eq. (3.4) into two parts in the hierarchical nonlinear factor analysis (HNFA) model:

$$\mathbf{h}(t) = \mathbf{B}\mathbf{s}(t) + \mathbf{b} + \mathbf{n}_h(t) \tag{3.6}$$

$$\mathbf{x}(t) = \mathbf{A}\phi[\mathbf{h}(t)] + \mathbf{C}\mathbf{s}(t) + \mathbf{a} + \mathbf{n}_x(t)\,, \tag{3.7}$$

where $\mathbf{n}_h(t)$ and $\mathbf{n}_x(t)$ are Gaussian noise terms. Note that we have included a short-cut mapping $\mathbf{C}$ from sources to observations. This means that hidden nodes only need to model the deviations from linearity.

The source model in HNFA is Gaussian so the model cannot be used for ICA. It can, however, be used as a nonlinear preprocessing method that extracts the correct subspace within which the correct rotation is then recovered using a standard linear ICA algorithm. Figure 3.4 illustrates the results of using HNFA together with the FastICA algorithm [1] for standard linear ICA to extract the sources from an artificial noisy nonlinear mixture. The data set used consisted of 1000 20-dimensional vectors which were created by nonlinearly mixing eight non-Gaussian independent random sources. A nonlinear model is clearly required in order to capture to underlying nonlinear manifold.

Nonlinear state-space models

In many cases, measurements originate from a dynamic system and form time series. In such cases, it is often useful to model the dynamics in addition to the instantaneous observations. We have extended the nonlinear factor analysis model by adding a nonlinear model for the dynamics of the sources $\mathbf{s}(t)$ [5]. This results in a state-space model where the sources can be interpreted as the internal state of the underlying generative process.

The nonlinear static model of Eq. (3.3) can easily be extended to a dynamic one by adding another nonlinear mapping to model the dynamics. This leads to source model

$$\mathbf{s}(t) = \mathbf{g}(\mathbf{s}(t-1), \boldsymbol{\theta}_g) + \mathbf{m}(t)\,, \tag{3.8}$$

where $\mathbf{s}(t)$ are the sources (states), $\mathbf{m}$ is the Gaussian process noise, and $\mathbf{g}(\cdot)$ is a vector containing as its elements the nonlinear functions modeling the dynamics.

As in nonlinear factor analysis, the nonlinear functions are modeled by MLP networks. The mapping $\mathbf{f}$ has the same functional form (3.4). Since the states in dynamical systems are often slowly changing, the MLP network for mapping $\mathbf{g}$ models the change in the value of the source:

$$\mathbf{g}(\mathbf{s}(\mathbf{t}-\mathbf{1})) = \mathbf{s}(t-1) + \mathbf{D}\tanh[\mathbf{C}\mathbf{s}(t-1) + \mathbf{c}] + \mathbf{d}\,. \tag{3.9}$$

An important advantage of the proposed new method is its ability to learn a high-dimensional latent source space. We have also reasonably solved computational and overfitting problems which have been major obstacles in developing this kind of unsupervised methods thus far. Potential applications for our method include prediction and process monitoring, control and identification. A process monitoring application is discussed in Section 3.6 in more detail.

Postnonlinear factor analysis

Our recent work restricts the general nonlinear mapping in (3.3) to the important case of post-nonlinear (PNL) mixtures. The PNL model consists of a linear mixture followed by componentwise nonlinearities acting on each output independently from the others:

$$x_i(t) = f_i\left[\mathbf{A}_{i,:}\mathbf{s}(t)\right] + n_i(t) \qquad i = 1, \ldots, n \tag{3.10}$$

The notation $\mathbf{A}_{i,:}$ in this equation means the i:th row of the mixing matrix $\mathbf{A}$. Preliminary results from the model (3.10) are encouraging. The results will be published in forthcoming papers.

References

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

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

[3] H. Valpola, E. Oja, A. Ilin, A. Honkela, and J. Karhunen. Nonlinear blind source separation by variational Bayesian learning. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E86-A(3):532–541, 2003.

[4] 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.

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

3.5 Hierarchical modeling of variances

In many models, variances are assumed to be constant although this assumption is often unrealistic in practice. Joint modeling of means and variances is difficult in many learning approaches, because it can give rise to infinite probability densities. In Bayesian methods using sampling the difficulties with infinite probability densities are avoided, but these methods are not efficient enough for very large datasets. Our variational Bayesian method [1, 2], which is based on our building blocks framework (see Sec. 3.3), is able to jointly model both variances and means efficiently.

The basic building block in our models is the variance neuron, which is a time-dependent Gaussian variable $u(t)$ controlling the variance of another time-dependent Gaussian variable $\xi(t)$

$$\xi(t) \sim \mathcal{N}\big(\mu_\xi(t), \exp[-u(t)]\big)$$

Here $\mathcal{N}(\mu, \sigma^2)$ is the Gaussian distribution with mean μ and variance σ^2, and $\mu_\xi(t)$ is the mean of $\xi(t)$ given by other parts of the model.

Figure 3.5 shows three examples of usage of variance neurons. The first model does not have any upper layer model for the variances. There the variance neurons are useful as such for generating super-Gaussian distributions for $\mathbf{s}$, enabling in effect us to find independent components. In the second model the sources can model concurrent changes in both the observations $\mathbf{x}$ and the modeling error of the observations through variance neurons $\mathbf{u}_x$. The third model is a hierarchical extension of the linear ICA model. The correlations and concurrent changes in the variances $\mathbf{u}_s(t)$ of conventional sources $\mathbf{s}(t)$ are modeled by higher-order variance sources $\mathbf{r}(t)$.

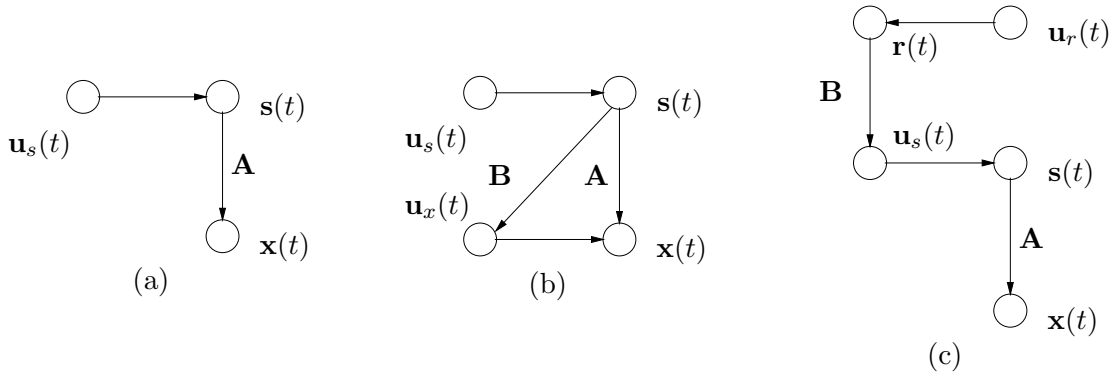

Figure 3.5: Various model structures utilizing variance neurons. Observations are denoted by $\mathbf{x}$, linear mappings by $\mathbf{A}$ and $\mathbf{B}$, sources by $\mathbf{s}$ and $\mathbf{r}$, and variance neurons by $\mathbf{u}$.

We have used the model of Fig. 3.5(c) for finding variance sources from biomedical data containing MEG measurements from a human brain. Part of that dataset is shown in Figure 3.6(a). The signals are contaminated by external artefacts such as digital watch, heart beat as well as eye movements and blinks. The most prominent feature in the area we used from the dataset is the biting artefact. There the muscle activity contaminates many of the channels starting after 1600 samples.

Some of the estimated ordinary sources $\mathbf{s}(t)$ and their variance neurons $\mathbf{u}_s(t)$ are shown in Figures 3.6(b) and 3.6(c). The variance sources $\mathbf{r}(t)$ that were discovered are shown in Figure 3.6(d). The first variance source clearly models the biting artefact. This variance source integrates information from several conventional sources and its activity varies very little over time. The second variance appears to represent increased activity during the onset of the biting, and the third variance source seems to be related to the amount of rhythmic activity on the sources.

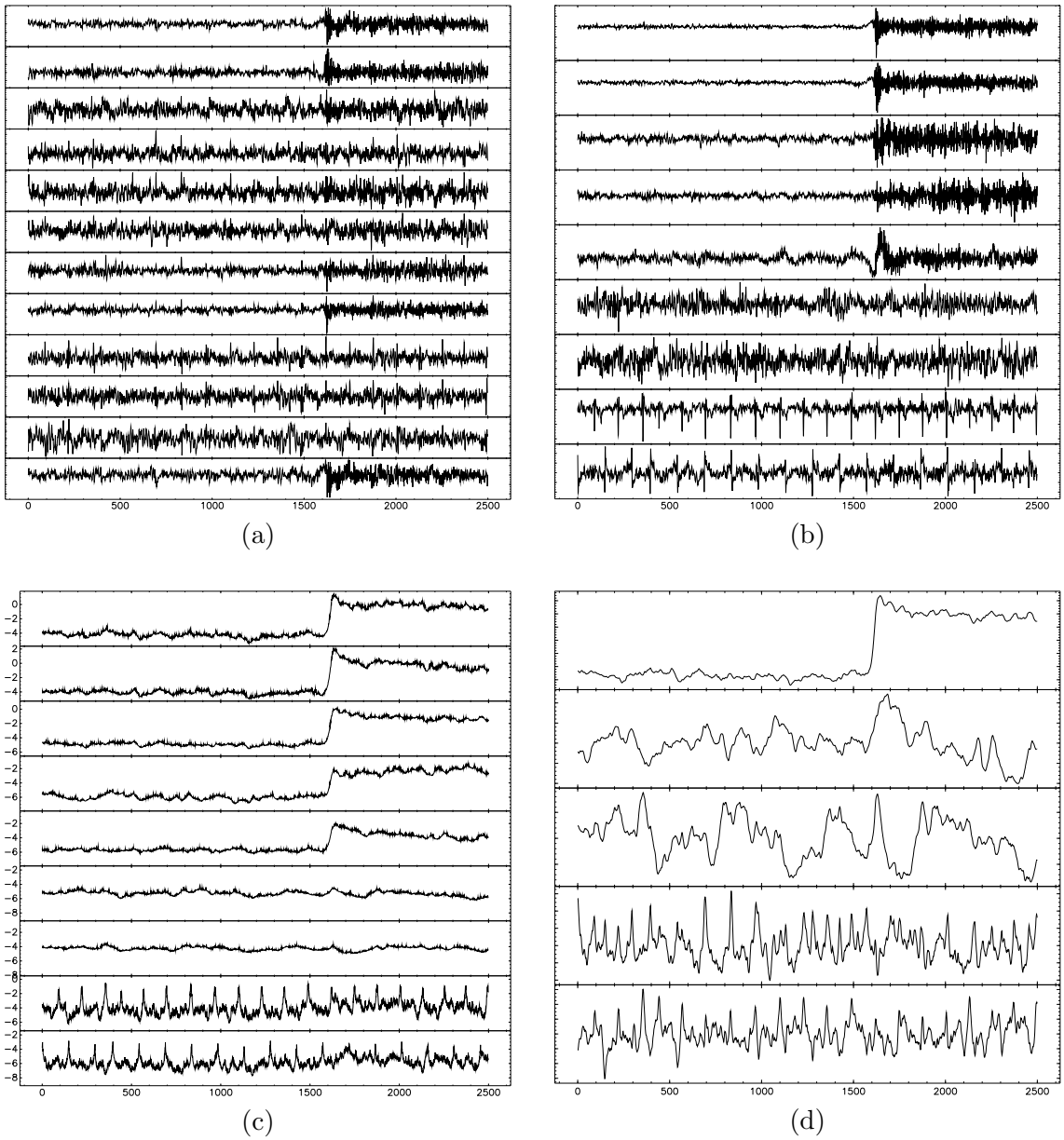

Figure 3.6: (a) MEG recordings (12 out of 122 time series). (b) Sources $\mathbf{s}(t)$ (nine out of 50) estimated from the data. (c) Variance neurons $\mathbf{u}_s(t)$ corresponding to the sources. (d) Variance sources $\mathbf{r}(t)$ which model the regularities found from the variance neurons.

References

[1] Harri Valpola, Markus Harva, and Juha Karhunen. Hierarchical models of variance sources. *Signal Processing*, 84(2):267–282, 2004.

[2] Harri Valpola, Markus Harva, and Juha 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.

3.6 Applications

In this section, applications of hierarchical nonlinear factor analysis and nonlinear state-space models discussed earlier in Section 3.4 are presented.

Missing values

Generative models can usually easily deal with missing observations. For instance in self-organizing maps (SOM) the winning neuron can be found based on those observations that are available. The generative model can also be used to fill in the missing values. This way unsupervised learning can be used for a similar task as supervised learning. Both the inputs and desired outputs of the learning data are treated equally. When a generative model for the combined data is learned, it can be used to reconstruct the missing outputs for the test data. The scheme used in unsupervised learning is more flexible because any part of the data can act as the cue which is used to complete the rest of the data. In supervised learning, the inputs always act as the cue.

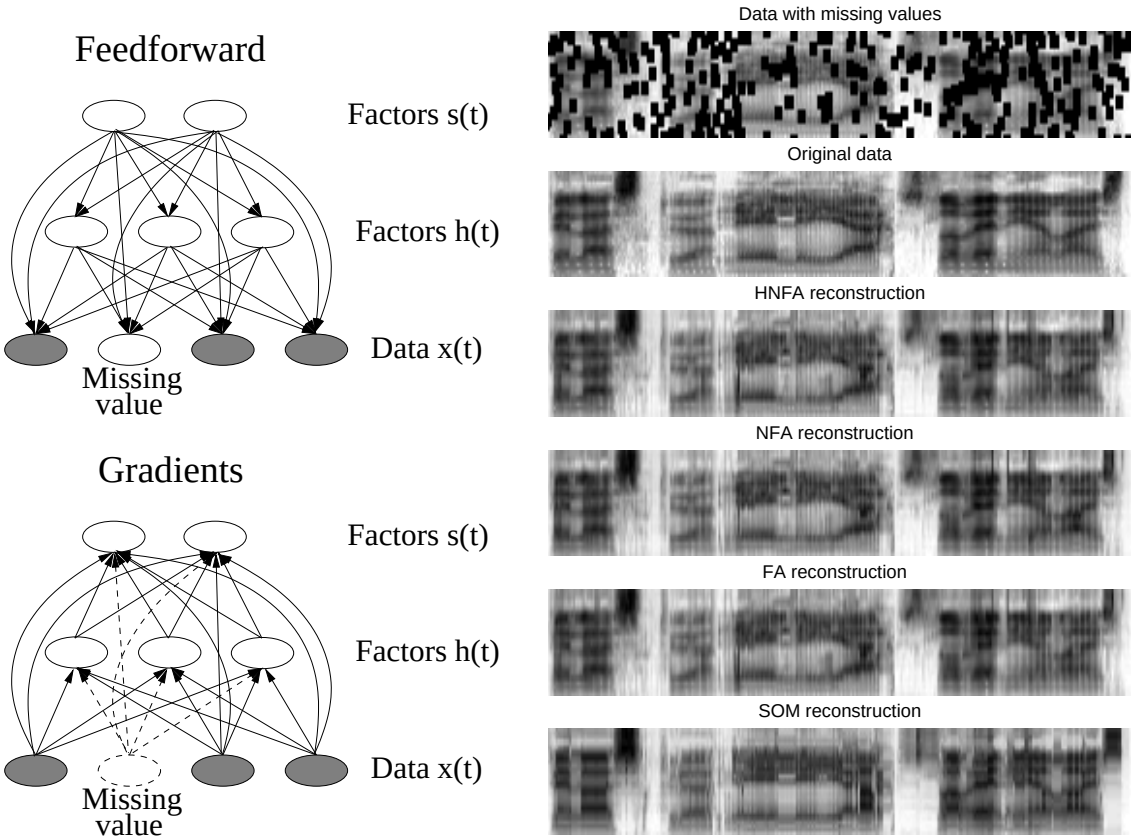

Figure 3.7: Left: The reconstructions of missing values using HNFA are produced in the feedforward direction. In the gradient direction, the components with missing values do not affect the factors, which are thus inferred using only the observed data. Right: Speech data reconstruction example with best parameters of each algorithm.

The quality of the reconstructions provides insight to the properties of different unsupervised models. The ability of self-organizing maps, linear principal component analysis, nonlinear factor analysis, and hierarchical nonlinear factor analysis to reconstruct the missing values of various data sets have been studied in [1]. Experiments were conducted

Missing value pattern	FA	HNFA	NFA	SOM
patches	1.87	1.80 ± 0.03	1.74 ± 0.02	1.69 ± 0.02
patches, permutated	1.85	1.78 ± 0.03	1.71 ± 0.01	1.55 ± 0.01
randomly	0.57	$0.55 \pm .005$	$0.56 \pm .002$	0.86 ± 0.01
randomly, permutated	0.58	$0.55 \pm .008$	$0.58 \pm .004$	0.87 ± 0.01

Table 3.1: Mean-square reconstruction errors and their standard deviations for speech spectra containing various types of missing values.

using four different patterns for the missing values. This way, different aspects of the algorithms could be studied.

Table 3.1 shows the mean-square reconstruction errors (and their standard deviations over different runs) of missing values in speech spectra. In Figure 3.7, the reconstructed spectra are shown for a case where data were missing in patches and the data is not permutated. In the permutated case, the learning data contained samples which were similar to the test data with missing values. This task does not require generalization but rather memorization of the learned data. SOM performs the best in this task because it has the largest amount of parameters in the model.

The task where the data was missing randomly and not in patches of several neighboring frequencies does not require a very nonlinear model but rather an accurate representation of a high-dimensional latent space. Linear and nonlinear factor analysis perform better than SOM whose parametrization is not well suited for very high-dimensional latent spaces. The conclusion of these experiments was that in many respects the properties of (hierarchical) nonlinear factor analysis are closer to linear factor analysis than highly nonlinear mappings such as SOM. The nonlinear extensions of linear factor analysis are nevertheless able to capture nonlinear structure in the data and perform as well or better than linear factor analysis in all the reconstruction tasks. The new hierarchical nonlinear factor analysis did not outperform the older nonlinear factor analysis in reconstruction accuracy, but it was more reliable and computationally lighter.

Detection of process state changes

One potential application for the nonlinear dynamic state-space model discussed in Section 3.4 is process monitoring. In [2, 3], ensemble learning was shown to be able to learn a model which is capable of detecting an abrupt change in the underlying dynamics of a fairly complex nonlinear process.

The process was artificially generated by nonlinearly mixing some of the states of three independent dynamical systems: two independent Lorenz processes and one harmonic oscillator.

The nonlinear dynamic model was first estimated off-line using 1000 samples of the observed process. The model was then fixed and applied on-line to new observations with artificially generated changes of the dynamics.

Figures 3.8 and 3.9 show an experiment with a change generated in the middle of the new data set, at time 1500, when the underlying dynamics of one of the Lorenz processes abruptly changes. Even though it is very difficult to detect this from the observed nonlinear mixtures shown in Fig. 3.8, the change detection method based on the estimated model readily detects the change raising alarms after the time of change. The method is also able to find out in which states the change occurred: Analyzing the structure of the cost function helps in localizing the detected changes, as demonstrated in Fig. 3.9.

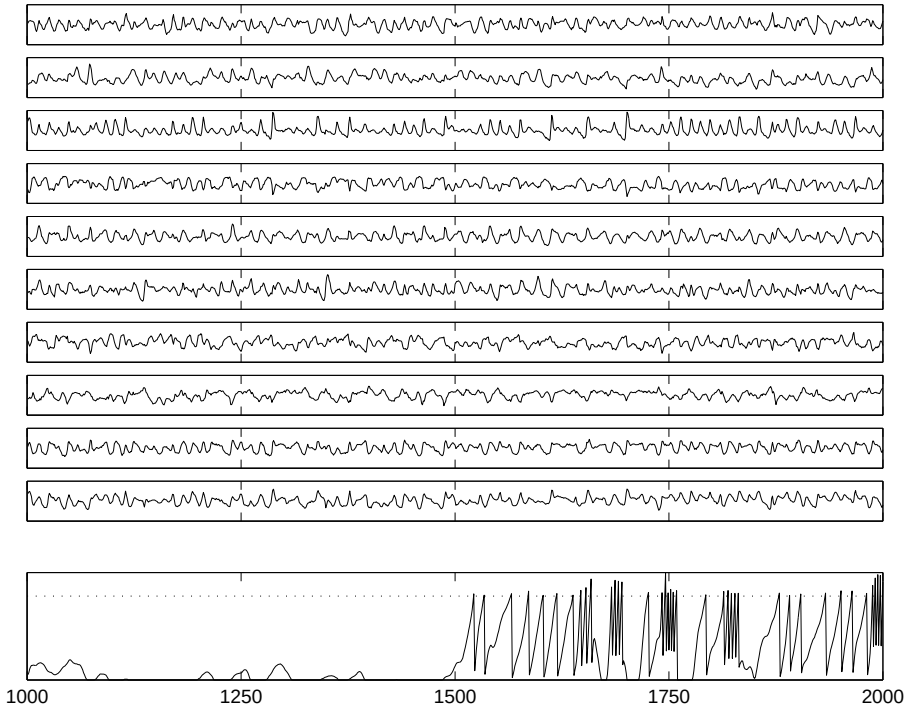

Figure 3.8: The monitored process (10 time series above) with the change simulated at $t = 500$. Even though the change is hardly visible to the eye, it has been detected with the estimated model: The test statistic of the proposed method is shown below.

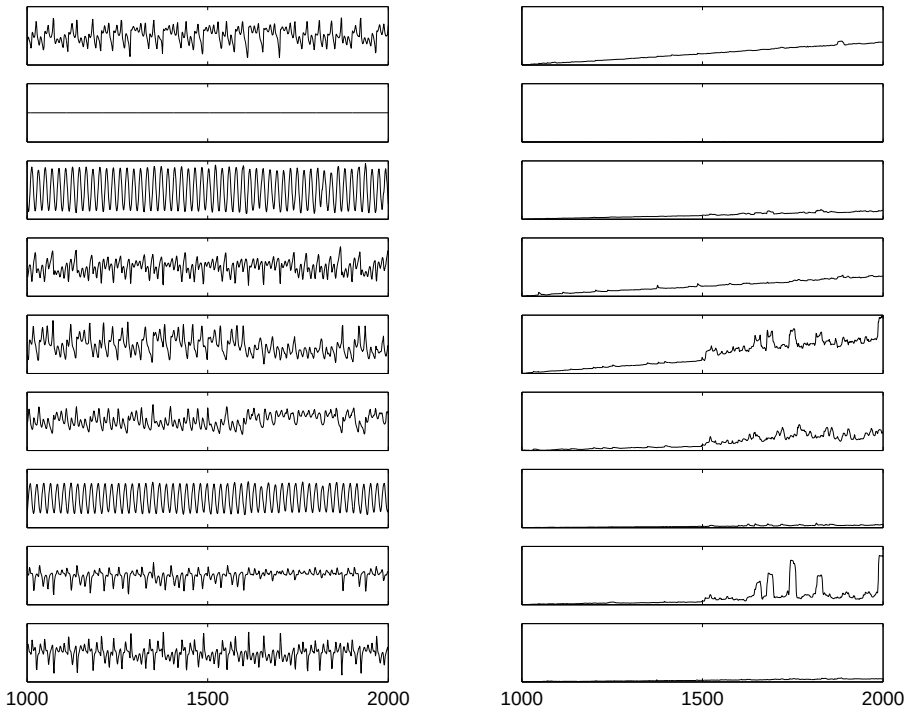

Figure 3.9: The estimated process states (left) and their contribution to the cost function (right). The cause of the change can be verified by detecting the states which increase their cost contribution.

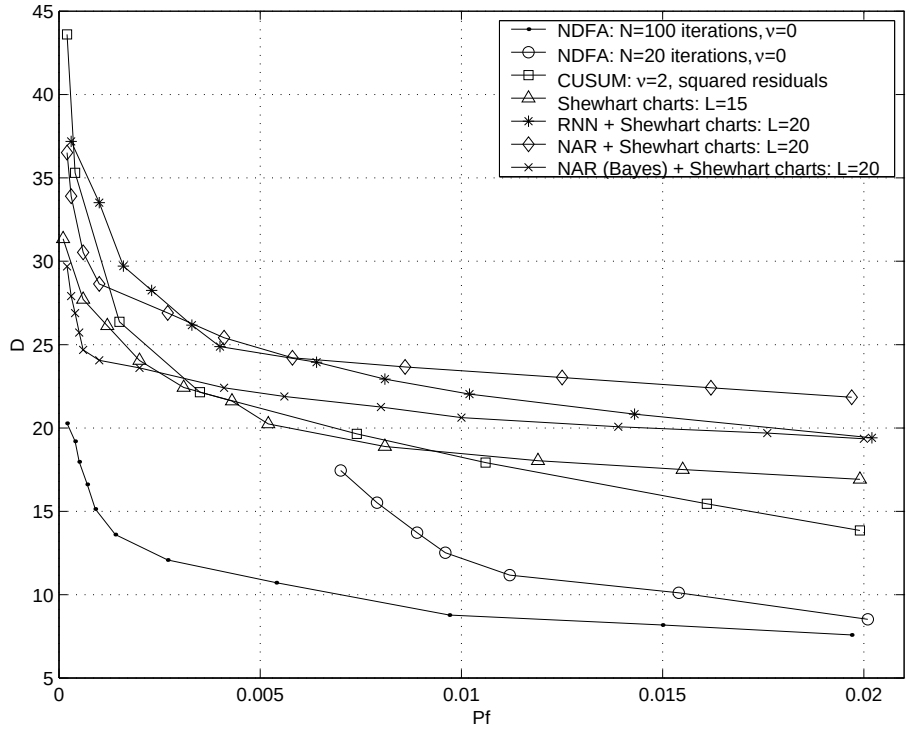

Figure 3.10: Two change detection performance measures calculated for the proposed NDFA method as well as for some alternative techniques. The probability of false alarms P_f is plotted against the average time to detection D for different values of the detection threshold. The closer a curve is to the origin, the faster the algorithm can detect the change with low false alarm rate.

The experimental results in Fig. 3.10 show that the method outperforms several other change detection techniques. Two alternative approaches compared with our method are based on other types of nonlinear dynamic models, namely nonlinear autoregressive (NAR) model and recurrent neural network (RNN) model. Other compared methods monitored simple indicators of the process such as its mean and covariance matrix (CUSUM algorithm and Shewhart control charts).

References

[1] 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, June 2003.

[2] A. Iline, H. Valpola, and E. Oja. Detecting process state changes by nonlinear blind source separation. In *Proc. of the 3rd Int. Workshop on Independent Component Analysis and Signal Separation (ICA2001)*, pages 704–709, San Diego, California, December 2001.

[3] A. Ilin, H. Valpola, and E. Oja. Nonlinear Dynamical Factor Analysis for State Change Detection. *IEEE Transaction on Neural Networks*, 2004. To appear.