PAPER

Nonlinear Blind Source Separation by Variational Bayesian Learning

Harri VALPOLA[†], **Erkki OJA**[†], **Alexander ILIN**[†], **Antti HONKELA**[†], *and* **Juha KARHUNEN**[†], ***Nonmembers***

SUMMARY

Blind separation of sources from their linear mixtures is a well understood problem. However, if the mixtures are nonlinear, this problem becomes generally very difficult. This is because both the nonlinear mapping and the underlying sources must be learned from the data in a blind manner, and the problem is highly ill-posed without a suitable regularization. In our approach, multilayer perceptrons are used as nonlinear generative models for the data, and variational Bayesian (ensemble) learning is applied for finding the sources. The variational Bayesian technique automatically provides a reasonable regularization of the nonlinear blind separation problem. In this paper, we first consider a static nonlinear mixing model, with a successful application to real-world speech data compression. Then we discuss extraction of sources from nonlinear dynamic processes, and detection of abrupt changes in the process dynamics. In a difficult test problem with chaotic data, our approach clearly outperforms currently available nonlinear prediction and change detection techniques. The proposed methods are computationally demanding, but they can be applied to blind nonlinear problems of higher dimensions than other existing approaches.

1. Nonlinear blind source separation

Blind separation of sources (BSS) from their linear mixtures is a well-established problem with many suggested solutions; for a review, see [12]. One of the natural extensions of linear BSS is nonlinear blind source separation, in which a collection of nonlinear combinations of unknown source signals, possibly with additive noise, are observed and the problem is to estimate the sources from them. Such a problem setting has important applications in many cases where it is unreasonable to assume that the mixtures are linear, because the underlying natural processes are inherently nonlinear. Examples can be found in speech, biomedical, industrial, or financial time series processing. Many kinds of nonlinearities may turn these into nonlinear mixing problems, and any linear separation technique such as linear independent component analysis (ICA) then fails to find the sources.

As an example, consider a speech signal, which can be characterized by a source emitted by vocal chords and a transformation due to the vocal tract. A natural representation of the speech signal consists of parameters describing the states of the vocal chords on one hand and the vocal tract on the other hand. To some extent, these parameters are independent of each other. In the common preprosessing for speech recognition, a spectrogram is computed from the observed speech signal. The observed spectrum depends nonlinearly on the parameters of vocal chords and the vocal tract, and it might be possible to find the sources from it using nonlinear BSS.

During the past few years, several authors have tried to generalize linear ICA and BSS for nonlinear data models. For details and references on nonlinear BSS, see Chapter 17 of the book [12]. It turns out that nonlinear BSS is inherently much more difficult than the linear case. In particular, the nonlinear ICA problem is highly non-unique [12], [13], [19]. Contrary to linear ICA, it is always possible to find statistically independent factors from any multivariate density, if the mixing model is allowed to be a nonlinear function. Thus, suitable regularizing constraints must be used to make the problem well-posed. Other problems appearing in most of the current nonlinear BSS methods are that their computational load often grows exponentially with the dimensionality, limiting their use to small scale problems, and the mixture model is often restricted to some special case such as post-nonlinear mixtures [20], in which only the sensor nonlinearities are taken into account but the mixture itself is linear.

In the following, we review our approach of unsupervised variational Bayesian (ensemble) learning in nonlinear blind source separation of static and dynamic mixtures. This approach alleviates many of the problems of nonlinear BSS: the fully Bayesian treatment guarantees a natural regularization by controlling the complexity of both the sources and the nonlinear mixing mapping. The computational load does not grow strongly with the size of the problem, making it possible to solve larger models.

To phrase the nonlinear BSS problem, let us denote by $\mathbf{x}(t)$ the vector whose elements are the observed signals $x_i(t)$, $i = 1, \ldots, n$, by $\mathbf{s}(t)$ the vector whose elements are the source signals $s_j(t)$, $j = 1, \ldots, m$, and by $\mathbf{n}(t)$ the additive noise vector. The general data model assumed in nonlinear BSS is then

$$\mathbf{x}(t) = \mathbf{f}(\mathbf{s}(t)) + \mathbf{n}(t) \tag{1}$$

[†]Helsinki University of Technology, Neural Networks Research Centre, P.O. Box 5400, FIN-02015 HUT, Espoo, Finland. Email: {firstname.lastname}@hut.fi
URL: http://www.cis.hut.fi/

where $\mathbf{f}$ is a nonlinear mapping from the source signals to the observed signals. The dimensions of the vectors $\mathbf{s}(t)$ and $\mathbf{x}(t)$ are generally different, and often the dimensionality of $\mathbf{s}(t)$ is smaller. The term "blind" refers to the fact that both the source signals, the nonlinear mapping, and the noise are unknown.

The nonlinear function $\mathbf{f}$ can in principle be arbitrary. A relevant question is the identifiability of the model (1), that is, under what conditions is it possible to do the separation, knowing only the outputs $\mathbf{x}(t)$. This question cannot be answered in the general case [19]. For post-nonlinear mixtures, which are a rather mild extension of the linear model, separability can be analyzed theoretically [20]. In the general case, we have to resort to approximations. Instead of solving the "true" nonlinearity, our variational Bayesian learning principle attempts at finding a compact nonlinear representation for the observations. The function $\mathbf{f}$ is then modelled as an MLP network.

The nonlinear BSS problem cannot be solved unless some assumptions are made on the sources $\mathbf{s}(t)$. In this paper, we shall consider mainly two cases. In the static case, the source signals $s_i(t)$ are assumed to be Gaussian, and the noise vector $\mathbf{n}(t)$ corrupting the observations or representing the modeling errors is assumed to be zero mean Gaussian. The model (1) is then a nonlinear generalization of standard linear factor analysis [12]. Therefore, the Bayesian method based on it is called Nonlinear Factor Analysis (NFA).

The components of the source vectors $\mathbf{s}(t)$ found by the NFA method are generally statistically dependent because their distributions are taken to be Gaussian. In fact, the Nonlinear Factor Analysis method provides a useful nonlinear generalization of linear principal component analysis (PCA). For finding roughly statistically independent source signals, the source vectors $\mathbf{s}(t)$ are rotated using an efficient linear ICA algorithm such as FastICA [12]. This usually already provides reasonably good and physically meaningful nonlinear independent components.

For improving the quality of the nonlinear independent components further, the Nonlinear Independent Factor Analysis (NIFA) method can be used, which employs the mixture-of-Gaussians model for the sources [15]. Using NIFA, it is possible to approximate sufficiently well any well-behaving non-Gaussian source distributions. However, the NIFA method improves in practice the achieved separation results only a little, while making the learning process more complicated [12], [15]. For this reason, we adopt the simpler Gaussian source model in this paper.

Another, more complex source model which we consider in more detail in this paper is the Nonlinear Dynamic Factor Analysis (NDFA) model, introduced by one of the authors in [21]. There the source signals in (1) are further assumed to follow nonlinear dynamics according to the state-space model

$$\mathbf{s}(t) = \mathbf{g}(\mathbf{s}(t-1)) + \mathbf{m}(t) \tag{2}$$

Here $\mathbf{g}$ is another unknown nonlinear function controlling the state dynamics. The Gaussian error or noise vector $\mathbf{m}(t)$ has zero mean and it is independent of $\mathbf{s}(t)$. Due to the recursive dynamic model (2), the source signals $s_j(t)$ in NDFA are generally not independent. This is a reasonable assumption if the sources are the state space coordinates of some physical multivariate dynamical state.

The nonlinear dynamic state-space model described by Eqs. (1)-(2) is a very general and flexible model for time series data. The price that one has to pay for this flexibility is that the nonlinear dynamic BSS problem for this model is severely ill-posed [9]. A wide variety of nonlinear transformations can be applied to the sources and then embedded in the functions $\mathbf{f}$ and $\mathbf{g}$, keeping the predictions unchanged. However, variational Bayesian learning used in the NDFA method provides a useful regularization of the problem by restricting the complexity of separated sources and estimated mappings. General information on this kind of dynamic models and their estimation methods can be found in the references [3], [8], [9].

The remainder of this paper is organized as follows. In the next section, we first briefly introduce variational Bayesian learning, and then overview the theory of the NFA method. Experimental results for nonlinear factor analysis of speech data are given in Section 3. The extension to dynamic state-space model is discussed briefly in Section 4, with some experimental results on predicting chaotic Lorenz processes. In the last Section 4.4, we explain how the NDFA method for dynamic nonlinear BSS can be applied to detection of abrupt changes in process dynamics, comparing it with existing standard techniques.

Due to the space limitations, we must content ourselves in this paper with an overall description of the developed methods. More information on the details and refinements of the learning procedure, on potentially appearing problems, and experimental results can be found in [15], [21] for the static case, and in [21], [23] for the dynamic case. First results on detection of process state changes using the NDFA method have been presented in a recent conference paper [14].

2. Variational Bayesian learning for nonlinear BSS: the Nonlinear Factor Analysis method

2.1 Variational Bayesian learning

According to the Bayesian estimation and modeling philosophy, the posterior probability density function (pdf) of the unknown variables in a given model contains all the relevant information needed for further inference. The available prior information can be easily incorporated by modeling it using suitable prior distributions, and one can also select the most probable

model for the data from the chosen model family. If the prior information and the model family describe appropriately the problem under study, excellent results can be obtained.

In practical problems, exact treatment of the posterior pdf's is not possible. Therefore, some suitable approximation method must be used. Variational Bayesian learning [10], [16] is a recently developed method for parametric approximation of posterior pdf's. The basic idea in variational Bayesian learning is to approximate the true posterior pdf by a simpler function with restricted form. The method is fairly insensitive to overfitting, because the approximation is fitted to the probability mass of the true posterior pdf instead of finding some point estimate such as the peak of the posterior pdf.

Consider now application of variational Bayesian learning to the nonlinear BSS model (1). Denote by $\mathbf{X} = \{\mathbf{x}(t)|t\}$ the set of available data (mixture) vectors, by $\mathbf{S} = \{\mathbf{s}(t)|t\}$ the respective source vectors, and by $\boldsymbol{\theta}$ all the unknown parameters of the data model. Furthermore, $p(\mathbf{S}, \boldsymbol{\theta}|\mathbf{X})$ denotes the exact posterior pdf of the unknown sources and parameters, and $q(\mathbf{S}, \boldsymbol{\theta})$ its parametric approximation. In the version of variational Bayesian learning called ensemble learning, the misfit between the approximation and the true posterior is given by the Kullback-Leibler divergence

$$C_{\mathrm{KL}} = \int_{\mathbf{S}} \int_{\boldsymbol{\theta}} q(\mathbf{S}, \boldsymbol{\theta}) \ln \frac{q(\mathbf{S}, \boldsymbol{\theta})}{p(\mathbf{S}, \boldsymbol{\theta}|\mathbf{X})} d\boldsymbol{\theta} d\mathbf{S} \tag{3}$$

which measures the difference in the probability mass between the pdf's p and q. The minimum value 0 of the Kullback-Leibler divergence is achieved if and only if the two densities are the same.

The posterior distribution can be written as $p(\mathbf{S}, \boldsymbol{\theta}|\mathbf{X}) = p(\mathbf{S}, \boldsymbol{\theta}, \mathbf{X})/p(\mathbf{X})$. The normalizing term $p(\mathbf{X})$ cannot usually be evaluated, and therefore the actual cost function is

$$\begin{aligned} C &= C_{\mathrm{KL}} - \ln p(\mathbf{X}) \\ &= \int_{\mathbf{S}} \int_{\boldsymbol{\theta}} q(\mathbf{S}, \boldsymbol{\theta}) \ln \frac{q(\mathbf{S}, \boldsymbol{\theta})}{p(\mathbf{S}, \boldsymbol{\theta}, \mathbf{X})} d\boldsymbol{\theta} d\mathbf{S} \end{aligned} \tag{4}$$

The cost function (4) can also be used for model selection as explained in [16]. In the nonlinear BSS problem, it provides the necessary regularization. For each source signal and parameter, its posterior pdf is estimated instead of some point estimate. In many cases, an appropriate point estimate is given by the mean of the posterior pdf of the desired quantity, and the respective variance measures the confidence of this estimate.

Several authors have recently applied variational or closely related Bayesian methods to linear BSS and ICA models, see for example [1], [4], [6], [11], [17]. Our work differs from these approaches in that a more general nonlinear data model is assumed, and it is further extended to nonstationary dynamic state-space models. Bayesian techniques for learning the state-space model (1)-(2) have recently been introduced in [2], [5].

2.2 MLP network model

We use the well-known multilayer perceptron (MLP) network [8] for parameterizing the nonlinear mapping $\mathbf{f}$ in (1). MLP suits well to modeling both strong and mild nonlinearities. The MLP network model for $\mathbf{f}$ is

$$\mathbf{f}(\mathbf{s}(t)) = \mathbf{B}\tanh[\mathbf{A}\mathbf{s}(t) + \mathbf{a}] + \mathbf{b} \tag{5}$$

where $\mathbf{A}$ and $\mathbf{B}$ are the weight matrices of the hidden and output layers, and $\mathbf{a}$ and $\mathbf{b}$ the corresponding bias vectors. The sigmoidal tanh nonlinearity is applied componentwise to its argument vector.

The parameters of the model, or elements of $\boldsymbol{\theta}$ are: (1) the weight matrices $\mathbf{A}$ and $\mathbf{B}$ and the bias vectors $\mathbf{a}$ and $\mathbf{b}$; (2) the variances of the observation noise, source signals and column vectors of the weight matrices; and (3) the hyperparameters used for defining the distributions of the biases and the parameters in the group (2).

Since noise terms $\mathbf{n}(t)$ and $\mathbf{m}(t)$ are assumed to be Gaussian and white, their values at different time instants and different components at the same time instant are independent. The likelihood of the observations $\mathbf{x}(t)$ defined by the model (1) can then be written as

$$\begin{aligned} p(\mathbf{X}|\mathbf{S}, \boldsymbol{\theta}) &= \prod_{i,t} p(x_i(t)|\mathbf{s}(t), \boldsymbol{\theta}) \\ &= \prod_{i,t} N(x_i(t);\ f_i(\mathbf{s}(t)), \exp(2v_i)) \end{aligned} \tag{6}$$

where $N(x;\ \mu, \sigma^2)$ denotes a Gaussian distribution over x with mean μ and variance σ^2, $f_i(\mathbf{s}(t))$ denotes the ith component of the output of the nonlinearity $\mathbf{f}$, and v_i is a parameter specifying the noise variance. The variances are parameterized using the exponential function $\exp(2v)$ where $v \sim N(\alpha, \beta)$, because this makes it easier to build hierarchical models for the prior distributions.

All the parameters of the model have hierarchical Gaussian priors. For example the noise parameters v_i of different components of the data share a common prior. The pdf's of the parameters of the model (5) are all Gaussian, and they have been specified in [15], [23].

2.3 Computation of the cost function

Denote the two parts of the cost function (4) arising from the denominator and numerator of the logarithm by

$$C_q = \int_{\mathbf{S}} \int_{\boldsymbol{\theta}} q(\mathbf{S}, \boldsymbol{\theta}) \ln q(\mathbf{S}, \boldsymbol{\theta}) d\boldsymbol{\theta} d\mathbf{S} \tag{7}$$

$$C_p = -\int_{\mathbf{S}} \int_{\boldsymbol{\theta}} q(\mathbf{S}, \boldsymbol{\theta}) \ln p(\mathbf{S}, \boldsymbol{\theta}, \mathbf{X}) d\boldsymbol{\theta} d\mathbf{S} \tag{8}$$

For evaluating the cost function $C = C_q + C_p$, we need two things: the exact formulation of the joint probability density $p(\mathbf{S}, \boldsymbol{\theta}, \mathbf{X})$, and the functional form of the approximation $q(\mathbf{S}, \boldsymbol{\theta})$.

Usually the joint pdf $p(\mathbf{S}, \boldsymbol{\theta}, \mathbf{X})$ is a product of simple terms due to the definition of the model. The joint pdf can be written

$$p(\mathbf{S}, \boldsymbol{\theta}, \mathbf{X}) = p(\mathbf{X}|\mathbf{S}, \boldsymbol{\theta}) p(\mathbf{S}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \tag{9}$$

The pdf $p(\mathbf{X}|\mathbf{S}, \boldsymbol{\theta})$ has already been evaluated in (6), and the pdf's $p(\mathbf{S}|\boldsymbol{\theta})$ and $p(\boldsymbol{\theta})$ are also products of univariate Gaussian distributions. They can be obtained directly from the model structure [15], [23]. Note that $\mathbf{S}$ is not independent of $\boldsymbol{\theta}$, because $\boldsymbol{\theta}$ also contains the parameters of the source distributions.

The cost function can be minimized efficiently if a suitably simple factorial form for the approximation is chosen. We use $q(\boldsymbol{\theta}, \mathbf{S}) = q(\boldsymbol{\theta}) q(\mathbf{S})$, where

$$q(\boldsymbol{\theta}) = \prod_i q(\theta_i) = \prod_i N(\theta_i; \overline{\theta}_i, \widetilde{\theta}_i) \tag{10}$$

is the product of Gaussian posterior pdf approximations $N(\theta_i; \overline{\theta}_i, \widetilde{\theta}_i)$ of each parameter θ_i. The distribution $q(\mathbf{S})$ is a similar product of Gaussian terms $N(s_i(t); \bar{s}_i(t), \widetilde{s}_i(t))$. The total cost C is then a function of all these means and variances $\overline{\theta}_i, \widetilde{\theta}_i, \bar{s}_i(t), \widetilde{s}_i(t)$.

The part C_q of the cost function in (7) is now a sum of negative entropies of Gaussians, and it has the exact form [15], [23]

$$C_q = \sum_i -\frac{1}{2}[1 + \ln(2\pi\widetilde{\theta}_i)] + \sum_{t,i} -\frac{1}{2}[1 + \ln(2\pi\widetilde{s}_i(t))] \tag{11}$$

Evaluation of the cost C_p arising from the data is somewhat more complicated. However, the cost C_p also splits into a sum of simple terms, and most of them can be evaluated analytically. Only the terms involving the outputs $\mathbf{f}(\mathbf{s}(t))$ of the MLP network (5) cannot be computed exactly. These terms can be approximated by using a truncated Taylor series for $\mathbf{f}(\mathbf{s}(t))$. This is explained in detail in [15], [23].

2.4 Updating the parameters and sources

The parameters of the approximating distribution are optimized with gradient based iterative algorithms. During one epoch or sweep of the algorithm all the parameters are updated once, using all the available data. One sweep consists of two different phases. The order of the computations in these two phases is the same as in one epoch of the standard back-propagation algorithm for MLP networks [8], but otherwise the learning procedure is quite different. The most important differences are that in the NFA method learning is unsupervised, the cost function is different, and unknown variables are characterized by distributions instead of point estimates.

In the forward phase, the distributions of the outputs of the MLP networks are computed from the current values of the inputs. The value of the cost function is also evaluated as explained in the previous subsection. In the backward phase, the partial derivatives of the cost function with respect to all the parameters are fed back through the MLP and the parameters are updated using this information.

An update rule for the posterior variances $\widetilde{\theta}_i$ is obtained by differentiating (4) with respect to $\widetilde{\theta}_i$, yielding [15], [23]

$$\frac{\partial C}{\partial \widetilde{\theta}_i} = \frac{\partial C_p}{\partial \widetilde{\theta}_i} + \frac{\partial C_q}{\partial \widetilde{\theta}_i} = \frac{\partial C_p}{\partial \widetilde{\theta}_i} - \frac{1}{2\widetilde{\theta}_i} \tag{12}$$

Equating this to zero yields a fixed-point iteration:

$$\widetilde{\theta}_i = \left[2 \frac{\partial C_p}{\partial \widetilde{\theta}_i} \right]^{-1} \tag{13}$$

The posterior means $\overline{\theta}_i$ can be estimated from the approximate Newton iteration [15], [23]

$$\overline{\theta}_i \leftarrow \overline{\theta}_i - \frac{\partial C_p}{\partial \overline{\theta}_i} \left[\frac{\partial^2 C}{\partial \overline{\theta}_i^2} \right]^{-1} \approx \overline{\theta}_i - \frac{\partial C_p}{\partial \overline{\theta}_i} \widetilde{\theta}_i \tag{14}$$

After each sweep through the data, the posterior pdf approximation $q(\boldsymbol{\theta})$ is updated. The pdf $q(\mathbf{S})$ is updated in a similar fashion.

2.5 Overall learning procedure

The learning procedure is somewhat different in the beginning of training. The hyperparameters governing the distributions of other parameters are not updated, to avoid pruning away parts of the model that do not seem useful at the moment. The posterior means of most of the parameters are initialized to random values. The posterior variances are initialized to small constant values. The posterior means of the sources $\mathbf{s}(t)$ are initialized using a suitable number of principal components of the entire data set $\mathbf{X}$. They are kept fixed to these values for the first 50 sweeps when only the MLP network $\mathbf{f}$ is updated. Updates of the hyperparameters begin after the first 100 sweeps.

The learning is continued until convergence. In variational Bayesian learning, there is no danger of over-learning, so in principle the "optimal" number of learning steps is infinite. In some cases, like the example in Section 4.2, the actual number of learning sweeps can be quite large.

It is not possible to give all the details of the NFA algorithm here. A more detailed description is given in [23], and a software implementation for the method is available at [22].

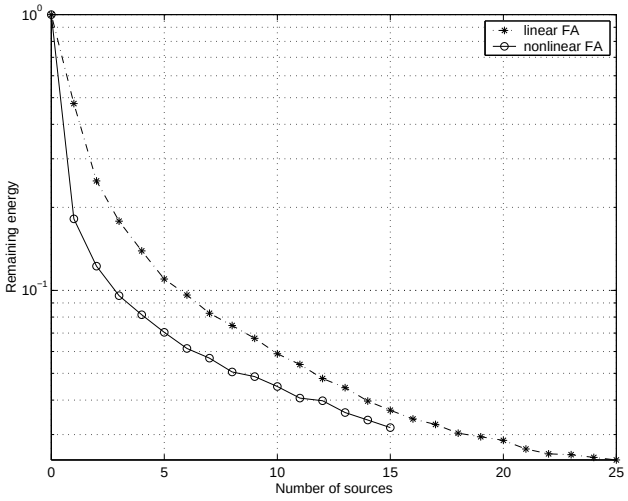

Fig. 1 The remaining (residual) energy of the speech data as a function of the number of extracted components using linear and nonlinear factor analysis.

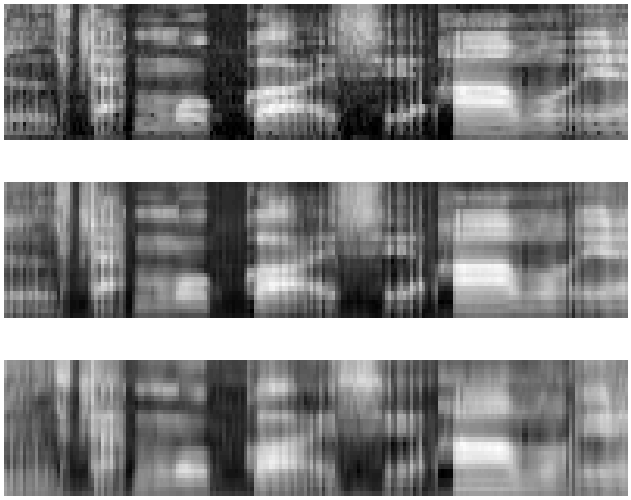

Fig. 2 A short fragment of the data used in the speech modeling experiment. The subfigures show the original data (top), and the reconstructions from 8 nonlinear components (middle) and from 8 linear components (bottom).

3. Experimental results for speech data compression

Both nonlinear and linear factor analysis can be seen as data compression methods, as they find a more compact representation for data in the form of the hidden factors. A relevant question is whether NFA is better than linear FA for compression: given an equal number of factors in both models, how do the residual errors compare?

To test this on natural data, we used a speech data set consisting of spectrograms of 24 individual words of Finnish speech, spoken by 20 different speakers. The spectra were modified to mimic the reception abilities of the human ear. This is a standard preprocessing procedure for speech recognition. The preprocessed data consisted of 2547 30-dimensional spectrogram vectors.

For studying the dimensionality of the data, linear factor analysis as well as NFA were applied to the data. 7500 sweeps were used to find a NFA representation for the sources. The final results were then obtained by applying linear ICA to the found sources $\mathbf{s}(t)$. The nonlinearity in NFA was an MLP network with 30 hidden neurons. For a given number of found factors, the residual error was computed for both nonlinear and linear FA. The results are shown in Fig. 1. Nonlinear factor analysis is able to explain the observations equally well with fewer components than linear factor analysis, showing that the data clearly follow a nonlinear mixing model. The difference is especially clear when the number of components is small.

A small segment of the original data and its reconstructions with eight nonlinear and linear components are shown in Fig. 2. The reconstructed spectrograms are somewhat smoother than the original ones. Still, all the discriminative features of the original spectrum are well preserved in the nonlinear reconstruction. The linear reconstruction is not as good, especially in the beginning. The sources found by the algorithm could be used as features for a speech recognition system. Since the essential contents of the data can be represented with fewer components than using linear methods, nonlinear BSS should provide better performance for feature extraction. The method could be further improved by also taking into account the temporal information in the data.

In [12], [15], we have applied the NFA method to 30-dimensional industrial pulp process data and to artificially generated data with good results. In these experiments, we improved the quality of estimated sources somewhat further by using the NIFA method.

4. Extension to a dynamic nonlinear model: the NDFA method

4.1 The model and method

In nonlinear dynamical factor analysis (NDFA), the same generative model (1) and MLP network structure (5) as before are still employed for the data vectors $\mathbf{x}(t)$. The nonlinear mapping $\mathbf{g}$ in the additional state-space model (2) for the dynamics of the sources $\mathbf{s}(t)$ is modeled using another MLP network as follows:

$$\mathbf{g}(\mathbf{s}(t-1)) = \mathbf{s}(t-1) + \mathbf{D}\tanh[\mathbf{C}\mathbf{s}(t-1) + \mathbf{c}] + \mathbf{d} \quad (15)$$

The matrices $\mathbf{C}$ and $\mathbf{D}$ contain the weights of the hidden and output layers, respectively, and the vectors $\mathbf{c}$ and $\mathbf{d}$ their biases. Because the sources $\mathbf{s}(t)$ do usually not change much from their previous values $\mathbf{s}(t-1)$ during a single time step, we have used the MLP network

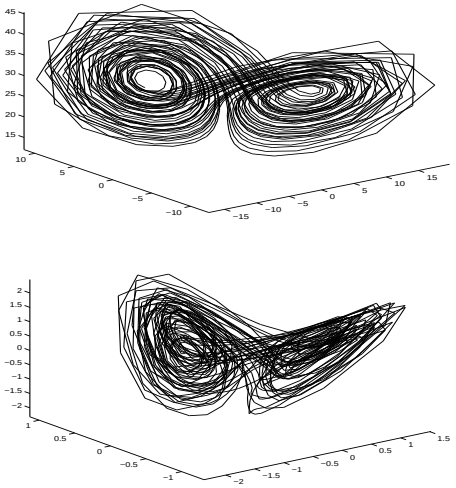

Fig. 3 The original sampled Lorenz process and the corresponding 3 components of the estimated process.

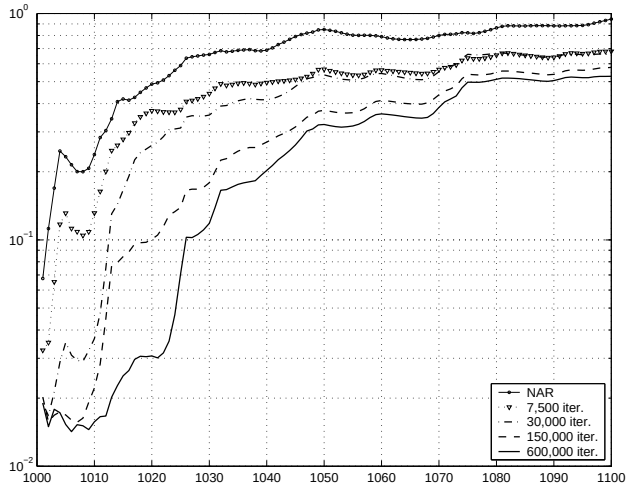

Fig. 4 The average cumulative squared prediction error for the nonlinear autoregressive (NAR) model (solid with dots) and for our dynamic algorithm with different iteration numbers.

to model only their change in (15).

The NDFA method can now be developed in a similar manner as the NFA method before. This is described in detail in [21], [23]. A special problem in NDFA is that the components of subsequent source vectors $\mathbf{s}(t)$ and $\mathbf{s}(t-1)$ are strongly dependent due to the model (2). This dependency is handled by using a linear model for their corresponding components $s_i(t)$ and $s_i(t-1)$ in the posterior approximation $q(\mathbf{S})$, while different components are still assumed to be independent. This does not increase the computational load too much. The NDFA method is initialized using PCA of concatenated $2d+1$ subsequent data vectors $\mathbf{z}^T(t) = [\mathbf{x}^T(t+d), \ldots, \mathbf{x}^T(t-d)]$. Such embedding methods are standard tools in the analysis of nonlinear dynamic systems.

4.2 Estimation of states for artificial data

The dynamic process used to test the NDFA method was a combination of three independent dynamic processes. The first one was a harmonic oscillator having a two-dimensional state representation with linear dynamics. The two other processes were Lorenz processes with three-dimensional chaotic nonlinear dynamics. Figure 3 shows one of the sampled Lorenz processes. To make the BSS and state prediction tasks more challenging, one dimension of each of the three processes was hidden. This left five states from which the 10-dimensional data vectors were generated nonlinearly using a randomly chosen MLP network with $\sinh^{-1}$ nonlinearity. The number of data vectors $\mathbf{x}(t)$ was 1000. The standard deviations of the observation noise and the signal were normalized to 0.1 and 1.

Figure 3 also shows the NDFA estimate of the original Lorenz process after 600,000 iterations. The quality of the estimated process is good, taking into account the very difficult problem and the indeterminacies in this type of nonlinear dynamic estimation.

According to our experiments, the NDFA method is able to separate blindly at least 15 source processes. This is clearly more than many other methods can in practice handle. The price that one has to pay for that is that learning requires a lot of computer time, taking easily days or even more.

4.3 Predicting the Lorenz data

We tested the quality of the NDFA model also by studying its short-term prediction ability for new samples. It should be noted that since the Lorenz processes are chaotic, exact numerical long-term prediction is impossible. However, the NDFA method is able to capture the general long-term behavior as shown in [23].

We compared NDFA in short-time prediction of future observations with the nonlinear autoregressive (NAR) model

$$\mathbf{x}(t) = \mathbf{h}(\mathbf{x}(t-1), \ldots, \mathbf{x}(t-d)) + \mathbf{n}(t) \qquad (16)$$

The nonlinear mapping $\mathbf{h}(\cdot)$ was again modeled by an MLP network, but now standard back-propagation was applied in learning. Before that, the dimension of the data vectors $\mathbf{x}(t)$ was compressed from 100 to 20 using standard PCA. The best performance was given by an MLP network with 20 inputs and one hidden layer of 30 neurons, and the number of delays was $d = 10$. Figure 4 shows the results, averaged over 100 realizations of the data set. Our NDFA method is already after 7500 sweeps clearly better than the NAR-based method in predicting the process $\mathbf{x}(t)$. Its performance improves further greatly when learning is continued. The final predictions given by NDFA after 600,000 sweeps are excellent up to the time $t = 1010$ and good up to $t = 1022$,

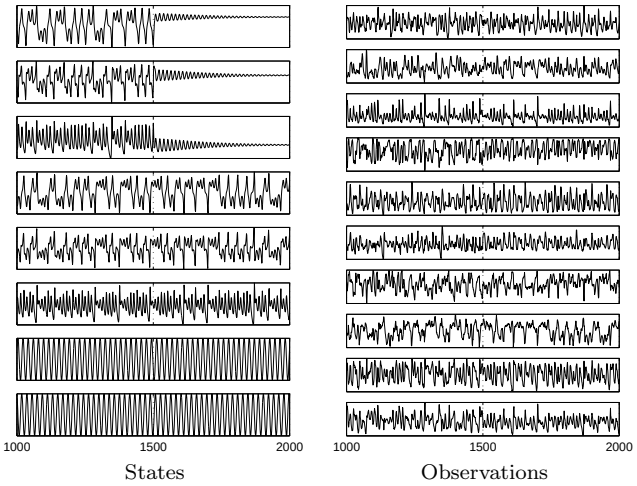

Fig. 5 The simulated changes in the process states and the corresponding observed mixture.

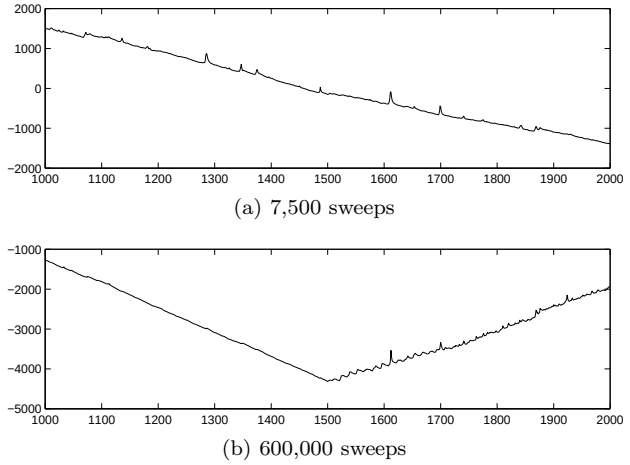

Fig. 6 The cost function calculated for the new observations.

while the NAR method is quite inaccurate already after $t \geq 1003$. Here the prediction started at time $t = 1000$.

We also tried a recurrent network for this test problem, and it performed somewhat better than the NAR model. The results for the recurrent net are roughly equivalent to those given by NDFA after 7500 sweeps, and clearly inferior to the fully converged NDFA model.

4.4 Detecting process state changes

We also examined the ability of the NDFA algorithm to detect changes in the process properties. The training sequence from Section 4.2 was continued with 1000 new samples and the changes in the process model were achieved by changing the parameters of one of the Lorenz processes at time instant $t_c = 1500$. Fig. 5 (left panel) clearly shows the simulated changes in the process states. However, the complex nonlinear mixing model and the additive noise have hidden the state changes which are now hardly discernible from the observed signals (see Fig. 5, right panel).

The NDFA change detection procedure was based on the sequential estimation of states $\mathbf{s}(t)$ corresponding to new obtained measurements $\mathbf{x}(t)$. At each time instant t the new state values were found by taking the initial guess $\mathbf{s}(t) = \mathbf{g}(\mathbf{s}(t-1))$ and then updating the posterior means and variances of $\mathbf{s}(t)$ with a few iterations of the NDFA algorithm. Mappings $\mathbf{f}$ and $\mathbf{g}$ were fixed and not adjusted to the new observations.

The cost function $C(t)$ produced by NDFA for each new observation $\mathbf{x}(t)$ carries valuable information which can be used to detect changes in the data model. Recall that according to (4), the cost function implicitly includes the term $-\ln p(\mathbf{X}_t)$, where $\mathbf{X}_t$ is the set of data vectors $\mathbf{x}(\tau)$ for $1 \leq \tau \leq t$. Therefore the difference of two consecutive values of $C(t)$ can be used to estimate the probability of the new observation given the preceding ones:

$$\begin{aligned} C(t) - C(t-1) &\approx -\ln p(\mathbf{X}_t) + \ln p(\mathbf{X}_{t-1}) \;\Rightarrow \\ p(\mathbf{x}(t)|\mathbf{X}_{t-1}) &= \frac{p(\mathbf{X}_t)}{p(\mathbf{X}_{t-1})} \approx e^{-[C(t)-C(t-1)]} \end{aligned}$$

The last approximation follows from the fact that e^{-C} is roughly proportional to $p(\mathbf{X})$.

Fig. 6 shows the estimated cost function $C(t)$ for the observed vectors of Fig. 5. It is evident that the NDFA method is able to detect the changes in the dynamic model after 600,000 sweeps while 7500 sweeps is not enough.

Moreover, the NDFA method can find out in which states the changes took place. Fig. 7 presents the states estimated for new measurements together with their contributions to the cost function (see [14] for details). The plot shows that the estimated time series reproduce well the character of the original underlying processes. Only nine out of ten estimated states are actually used: Two states model the harmonic oscillator dynamics, four states describe the Lorenz process with constant parameters, and the other three states correspond to the Lorenz process with changing parameters. Notice the increasing cost contributions for the states with changing dynamics: analyzing the structure of the cost function helps in localizing the detected changes.

Following standard change detection techniques, the NDFA method raises alarms about detected changes according to the decision rule

$$\text{decision at time } t = \begin{cases} \text{change}, & g(t) \geq h \\ \text{no change}, & g(t) < h \end{cases} \qquad (17)$$

where $g(t)$ is a test statistic calculated by the method at each time instant and h is a chosen threshold. The NDFA test statistic proposed in [14] was

$$g(t) = C(t) - \min_{k \leq t} C(k)$$

Similarly to [14], the performance of the NDFA

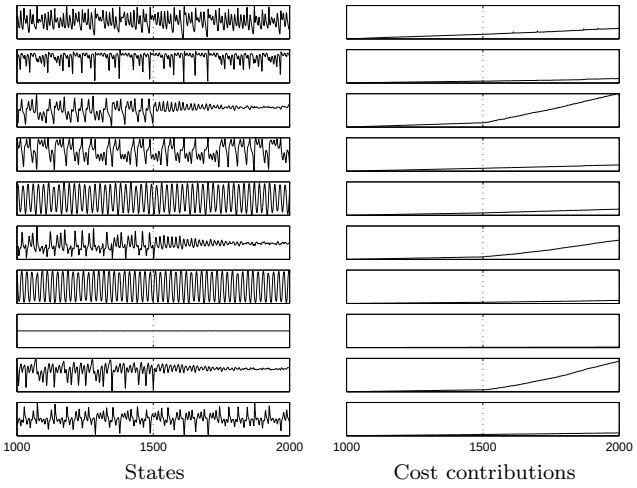

Fig. 7 The estimated states and their contribution to the cost function.

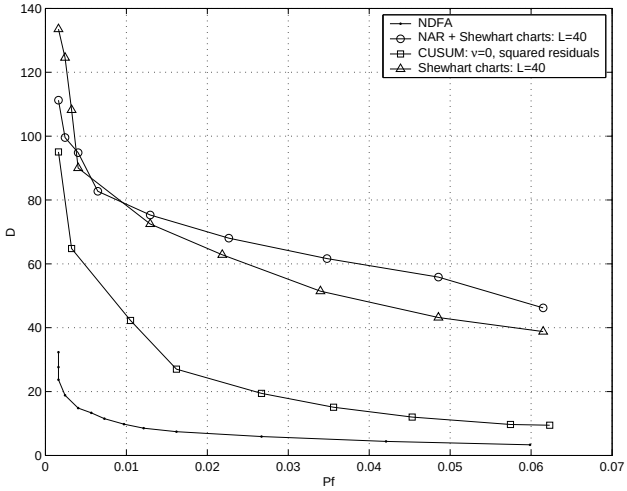

Fig. 8 Performance comparison of various change detection methods.

change detection method was assessed and compared with some classical methods employing different types of the test statistic $g(t)$ in (17). The two performance measures used in the comparison were the probability of false alarms P_f formally defined as

$$P_f = P(t_a < t_c) \tag{18}$$

and the average time to detection

$$D = E(t_a - t_c | t_a \geq t_c, t_c) \tag{19}$$

where t_c is the time of the change and t_a the time of the alarm about the detected change. In practice, measures (18), (19) were estimated by multiple simulation of changes at different time instants over 100 trials and taking the relative frequency of false alarms as P_f and the average time $t_a - t_c$ as the D-measure. Both measures of course depend on the decision threshold h which was varied in the simulations over a suitable range.

The following methods alternative to NDFA were considered in the experiments:

1. The CUSUM approach monitoring the current mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{R}$ of the observed process $\mathbf{x}(t)$ by calculating squared normalized residuals

   $$s(t) = (\mathbf{x}(t) - \boldsymbol{\mu})^T \mathbf{R}^{-1} (\mathbf{x}(t) - \boldsymbol{\mu}) - n_{\mathbf{x}}$$

   and using the two-sided CUSUM test [7]

   $$g_1(t) = [g_1(t-1) + s(t) - \nu]^+$$
   $$g_2(t) = [g_2(t-1) - s(t) - \nu]^+$$
   $$g(t) = \max(g_1(t),\, g_2(t))$$

   where $n_{\mathbf{x}}$ is the number of the observed signals, ν the drift parameter and $[x]^+ = \max(0, x)$. Vectors $\boldsymbol{\mu}$ and $\mathbf{R}$ were estimated from the training sequence.

2. The Shewhart charts generalization which detects changes in the mean $\boldsymbol{\mu}$ of a multivariate process $\mathbf{x}(t)$ by calculating the T^2-statistic [18]:

   $$g(t) = L(\overline{\mathbf{x}}_L(t) - \boldsymbol{\mu})^T \mathbf{R}^{-1} (\overline{\mathbf{x}}_L(t) - \boldsymbol{\mu})$$

   where $\overline{\mathbf{x}}_L(t)$ is the sample mean computed from last L observations.
3. The NAR-based approach using the model from Section 4.3. The mean of the NAR-prediction is monitored using the generalized Shewhart charts.

Fig. 8 shows the simulation results. The D-measure of Eq. (19) is plotted against the false alarm probability P_f of Eq. (18). Each point on the curves corresponds to one (P_f, D) pair for a given value of the decision threshold h in the rule (17). Each curve shows the results given by the respective method at different values of h. The closer a curve is to the origin, the faster the respective method can detect the change with low false detection rate. The NDFA method outperforms the compared classical methods clearly. They are not able to detect the change properly because of the complexity and nonlinearity of the process, while the NDFA method is highly accurate in this problem, giving a D value close to zero with very low false alarm rates.

5. Conclusions

The variational Bayesian learning principle was applied to the problem of nonlinear blind source separation (BSS). The problem is an important and natural extension of linear BSS and ICA. It is of great practical importance in cases where linear mixing cannot be assumed. However, the problem is essentially much more difficult than linear BSS because of the many ambiguities. In theory, it is not possible to retrieve the original sources except in some highly restricted cases. More or

less heuristic algorithms have been earlier suggested for the problem, but due to the non-uniqueness, there is no guarantee that any meaningful solutions are found.

Instead of trying to invert the unknown nonlinearity, what one can do instead is to get an approximation that gives a compact nonlinear representation for the observations. We call such an approximation Nonlinear Factor Analysis (NFA), in analogy with the standard factor analysis which solves a similar linear model. A further extension of NFA to the case of dynamical systems is given by the Nonlinear Dynamical Factor Analysis (NDFA). Then also the dynamics of the sources are modelled. The NDFA model is superficially similar to a nonlinear Kalman filter, but the solution method must be totally different because neither the dynamics nor the mapping from the states to the observations are known.

The NFA and NDFA algorithms are based on the Bayesian philosophy. The solution is not only a set of point estimates for the parametric nonlinear mappings and the sources themselves, as would be the case for example in the maximum likelihood solution. Instead, the pdf is found which best approximates the true posterior in the family of approximating densities, in this case product of independent gaussians. The Bayesian treatment allows all prior knowledge about the nonlinearities or sources to be included in the model in a very natural way. Also the complexity of the model is automatically regularized. The variances of the posterior densities around the mean directly indicate the confidences of the estimates. All these advantages make the variational Bayesian approach very promising, as compared to some less disciplined attempts to solve the nonlinear BSS problem. The experimental results in this paper confirm that very useful results can be obtained in difficult nonlinear problems.

The major disadvantage of the NFA and NDFA algorithms, as well as Bayesian learning in general, is the large computational demand. A high number of learning steps or sweeps through the training data set are needed for convergence. Then it is essential to optimize each sweep. In ongoing work, the computational complexity of the algorithms has been studied and decreased, to allow larger models to be analyzed.

Acknowledgments. This research has been funded by the European Commission project BLISS, and the Finnish Center of Excellence Programme (2000 - 2005) under the project New Information Processing Principles.

References

[1] 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.

[2] T. Briegel and V. Tresp. Fisher scoring and a mixture of modes approach for approximate inference and learning in nonlinear state space models. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 403–409, Cambridge, MA, USA, 1999. The MIT Press.

[3] M. Casdagli. Nonlinear prediction of chaotic time series. *Physica D*, 35:335–356, 1989.

[4] R.A. Choudrey and S.J. Roberts. Flexible Bayesian independent component analysis for blind source separation. In *Proc. Int. Conf. on Independent Component Analysis and Signal Separation (ICA2001)*, pages 90–95, San Diego, USA, 2001.

[5] Z. Ghahramani and S. Roweis. Learning nonlinear dynamical systems using an EM algorithm. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 599–605, Cambridge, MA, USA, 1999. The MIT Press.

[6] M. Girolami. Variational method for learning sparse and overcomplete representations. *Neural Computation*, 13(11):2517–2532, 2001.

[7] F. Gustafsson. *Adaptive Filtering and Change Detection*. John Wiley & Sons, 2000.

[8] S. Haykin. *Neural Networks – A Comprehensive Foundation, 2nd ed.* Prentice-Hall, 1998.

[9] S. Haykin and J. Principe. Making sense of a complex world. *IEEE Signal Processing Magazine*, 15(3):66–81, May 1998.

[10] G. 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.

[11] P. Højen-Sørensen, L. K. Hansen, and O. Winther. Mean field implementation of Bayesian ICA. In *Proc. Int. Conf. on Independent Component Analysis and Signal Separation (ICA2001)*, pages 439–444, San Diego, USA, 2001.

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

[13] A. Hyvärinen and P. Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural Networks*, 12(3):429–439, 1999.

[14] A. Iline, H. Valpola, and E. Oja. Detecting process state changes by nonlinear blind source separation. In *Proc. Int. Conf. on Independent Component Analysis and Signal Separation (ICA2001)*, pages 704–709, San Diego, USA, 2001.

[15] 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.

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

[17] J. Miskin and D. 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.

[18] D. C. Montgomery and P. J. Klatt. Economic design of T^2-control charts to maintain current control of a process. *Management Science*, 19(1):76–89, 1972.

[19] A. Taleb. A generic framework for blind source separation in structured nonlinear models. *IEEE Trans. on Signal Processing*, 50(8):1819–1830, 2002.

[20] A. Taleb and C. Jutten. Source separation in post-nonlinear mixtures. *IEEE Trans. on Signal Processing*, 47(10):2807–2820, 1999.

[21] H. Valpola. *Bayesian Ensemble Learning for Nonlinear Factor Analysis*. PhD thesis, Helsinki University of Tech-

nology, Espoo, Finland, 2000. Published in Acta Polytechnica Scandinavica, Mathematics and Computing Series No. 108.

[22] H. Valpola, A. Honkela, and X. Giannakopoulos. Matlab codes for the NFA and NDFA algorithms. `http://www.cis.hut.fi/projects/ica/bayes/`, 2002.

[23] H. Valpola and J. Karhunen. An unsupervised ensemble learning method for nonlinear dynamic state-space models. *Neural Computation*, 2002. In press.