Available online at www.sciencedirect.com
ScienceDirect
Digital Signal Processing 17 (2007) 914–934
Digital Signal Processing
www.elsevier.com/locate/dsp

Blind separation of nonlinear mixtures by variational Bayesian learning

Antti Honkela [a,*], Harri Valpola [b], Alexander Ilin [a], Juha Karhunen [a]

[a] *Adaptive Informatics Research Centre, Helsinki University of Technology, P.O. Box 5400, FI-02015 TKK, Finland*
[b] *Laboratory of Computational Engineering, Helsinki University of Technology, P.O. Box 9203, FI-02015 TKK, Finland*

Available online 20 March 2007

Abstract

Blind separation of sources from nonlinear mixtures is a challenging and often ill-posed problem. We present three methods for solving this problem: an improved nonlinear factor analysis (NFA) method using a multilayer perceptron (MLP) network to model the nonlinearity, a hierarchical NFA (HNFA) method suitable for larger problems and a post-nonlinear NFA (PNFA) method for more restricted post-nonlinear mixtures. The methods are based on variational Bayesian learning, which provides the needed regularisation and allows for easy handling of missing data. While the basic methods are incapable of recovering the correct rotation of the source space, they can discover the underlying nonlinear manifold and allow reconstruction of the original sources using standard linear independent component analysis (ICA) techniques.

Keywords: Bayesian learning; Blind source separation; Nonlinear mixtures; Post-nonlinear mixtures; Variational Bayes

1. Introduction

Independent component analysis (ICA) [1], that is blind separation of independent sources from their linear mixtures, forms the basis for most source separation work today. Introduction of Bayesian methods based on the variational approximation for models capable of solving the ICA problem in 1999 independently by Attias [2] and Valpola[1] [3] was an important step for Bayesian source separation. Later, slightly different methods based on the same principles have been proposed by a number of authors [4–7].

The Bayesian modelling approach is very useful in source separation, as in addition to providing tools for model order selection, it is open to all kinds of extensions simply by extending the model [6–8]. This was illustrated, for instance, by the nonlinear factor analysis (NFA) and nonlinear independent factor analysis (NIFA) models for blind separation of sources from nonlinear mixtures by Valpola and Honkela [9]. While the original NFA/NIFA model could be used to solve the difficult nonlinear separation problem in many cases, we have developed new variants of the model to better deal with certain specific instances.

* Corresponding author.
E-mail address: antti.honkela@tkk.fi (A. Honkela).
URL: http://www.cis.hut.fi/projects/bayes/.
[1] At the time his surname was Lappalainen.

1051-2004/$ – see front matter
doi:10.1016/j.dsp.2007.02.009

The hierarchical nonlinear factor analysis (HNFA) method [10] was introduced to overcome the quadratic computational complexity with respect to the number of the sources in original NFA/NIFA, thus allowing solving larger problems. The special case of post-nonlinear mixtures was addressed by post-nonlinear factor analysis (PNFA) [11]. Other important general extensions include ability to handle missing data [12,13], improved initialization strategies [14] as well as more accurate approximation of the nonlinearity in NFA/NIFA leading to increased stability [15]. In this paper we summarise these new results presented in scattered conference papers, and present the improved methods in full detail together with new experimental comparisons.

The paper is organised as follows. In Section 2, the nonlinear blind source separation problem and some basic theory behind it are introduced, followed by a review of other existing methods in Section 3. This is followed by presentation of our models to solve the problem in Section 4. Learning the models is discussed in Section 5. Some experimental results on the methods are presented in Section 6 with discussion and conclusion in Sections 7 and 8.

2. Nonlinear blind source separation

Blind separation of sources from their nonlinear mixtures—known as nonlinear blind source separation (BSS)—is generally a difficult problem, both from a theoretical and a practical point of view [1,16,17]. The task is to extract the sources $\mathbf{s}(t)$ that have generated the observations $\mathbf{x}(t)$ through a nonlinear mapping $\mathbf{f}(\cdot)$:

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

Here $\mathbf{x}(t)$ is the observed N-dimensional data (mixture) vector at time instant or index value t, $\mathbf{f} : \mathbb{R}^M \to \mathbb{R}^N$ is an unknown mixing function, $\mathbf{s}(t)$ is an M-vector whose elements are the M unknown source signals $s_i(t), i = 1, 2, \ldots, M$, and $\mathbf{n}(t)$ is an N-dimensional vector containing the additive noise terms.

In the basic linear case

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t), \tag{2}$$

where $\mathbf{A}$ is an $N \times M$ mixing matrix, the BSS problem can be solved using ICA [1] under certain conditions. Linear ICA methods utilise the strong but often realistic assumption that the source signals are statistically independent. Furthermore, all but possibly one of the source signals are assumed to be non-Gaussian, there is no or only a little noise and the number of sources M equals the number N of mixtures [1]. Under these conditions, the sources in (2) can be estimated using ICA up to permutation, scaling, and sign.

Linear ICA and other BSS methods have been extended for the cases where the number of the sources is different from the number of mixtures, and for the noisy case. But the extension to the nonlinear mixture model (1) is difficult since 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 suitable regularisation [1,16,18]. The independence assumption used in ICA is no longer sufficient for achieving separation. It can be shown that in the nonlinear case x and y can be mixed and still be statistically independent [18]. Additionally if x and y are two independent random variables, any of their functions $f(x)$ and $g(y)$ are also independent. These uniqueness issues have been considered in more detail in [16,18].

A related problem is that it is often quite difficult to infer the number of sources and the structure of the mapping $\mathbf{f}(\cdot)$. From a practical point of view, efficiency and reliability of nonlinear BSS algorithms are critical issues. They have restricted the number of sources that can be separated in practice to be quite small in many instances. An important reason for this is that the computational load of many nonlinear BSS methods increases quite rapidly with the dimensionality of the problem, preventing in practice their application to high-dimensional data sets [1].

An important special case of the general nonlinear mixing model (1) consists of so-called post-nonlinear mixtures. There each mixture has the form

$$x_i(t) = f_i\left(\sum_{j=1}^{M} A_{ij} s_j(t)\right), \quad i = 1, \ldots, N. \tag{3}$$

Thus the sources $s_j(t)$, $j = 1, \ldots, M$, are first mixed linearly according to the basic linear ICA/BSS model (2), but after that nonlinear functions f_i are applied to them to get the final observations x_i. It can be shown [16,19] that for the noiseless post-nonlinear mixtures, the indeterminacies are usually the same as for the basic linear instantaneous ICA model (2). That is, the sources can be separated or the independent components estimated up to the scaling,

permutation, and sign indeterminacies assuming the post-nonlinearities f_i are invertible and under weak conditions on the mixing matrix $\mathbf{A}$ and source distributions. The post-nonlinearity assumption is useful and reasonable in many signal processing applications, because it can be thought of as a model for a nonlinear sensor distortion. In more general situations, it is a restrictive and somewhat arbitrary constraint.

3. Existing methods

Existing nonlinear BSS methods have been reviewed in [16,17], and earlier in Chapter 17 of the book [1]. The paper [16] reviews especially uniqueness results on nonlinear ICA and BSS, separation methods for post-nonlinear mixtures, and our variational Bayesian estimation methods, referring to papers in which the detailed results have been presented. In this section, we will briefly review methods for BSS in nonlinear mixtures with i.i.d. samples.

The nonlinear blind source separation problem has been approached using auto-associative multilayer perceptron (MLP) networks, that is MLP networks trained with input–output pairs $(\mathbf{x}, \mathbf{x})$ [20]. The number of nodes in a hidden layer is restricted to be smaller than the number of inputs and outputs, thus creating a bottleneck. The extracted nonlinear features can be retrieved from the values of the hidden nodes. With standard back-propagation this approach is very prone to overfitting and local minima, but more advanced learning methods such as flat minimum search can provide a method for nonlinear BSS [21]. The computational complexity from explicitly learning both mixing and demixing models can still be high.

The MISEP method [22] is a generalisation of the infomax method of linear ICA for nonlinear mixtures using an MLP network to model the nonlinearity. The method provides good separation results in several low-dimensional artificial examples as well as in a real nonlinear image mixture problem [23], but it has not been demonstrated for more than 4 mixtures of 4 sources. Nonlinear denoising source separation [24] can also be applied to separate low-dimensional nonlinear mixtures.

Kernel methods have recently become popular in producing nonlinear counterparts for many linear statistical methods based on second-order statistics. Kernel principal component analysis (KPCA) [25] is an extension of linear principal component analysis (PCA) that operates on a high-dimensional feature space formed by a nonlinear transformation of the data space. The inner product features required for PCA can be evaluated efficiently in data space using the kernel trick. The method is computationally very efficient, but the results are highly dependent on the choice of the kernel. Additional post-processing is naturally also required to recover independent sources.

The Bayesian kernel method of Gaussian process latent variable model (GPLVM) [26] offers a probabilistic alternative to KPCA. The presentation of the method seems to focus on use in visualisation and it has not been demonstrated for more than three-dimensional latent spaces.

Post-nonlinear ICA has been approached, for instance, by direct minimisation of mutual information [19] as well as by heuristic methods. These are based on compensating for the invertible post-nonlinearities by first transforming the marginal densities of the data to be as Gaussian as possible and then applying standard ICA [27]. The downside of all of these methods is that they only work if the post-nonlinearities are invertible.

4. Models

In this section, the different models used in this work are presented. First, the multilayer perceptron (MLP) network based NFA model for general nonlinear mixtures from [9] is presented. Then, the hierarchical HNFA model based on the Bayes Blocks framework [28–30] and the PNFA model for post-nonlinear mixtures are introduced.

4.1. Multilayer perceptron (MLP) nonlinearity

Let us assume the observed data $\boldsymbol{X}$ follows the nonlinear mixing model of Eq. (1). The nonlinearity $\mathbf{f}$ is parametrised with a multilayer perceptron (MLP) network with single hidden layer with H hidden nodes [20]. This allows for writing a generative model for a data vector $\mathbf{x}(t)$ as [9]

$$\mathbf{x}(t) = \mathbf{f}\big(\mathbf{s}(t), \boldsymbol{\theta}_{\mathbf{f}}\big) + \mathbf{n}(t) = \mathbf{B}\boldsymbol{\phi}\big(\mathbf{A}\mathbf{s}(t) + \mathbf{a}\big) + \mathbf{b} + \mathbf{n}(t), \tag{4}$$

where $\boldsymbol{\theta}_{\mathbf{f}} = (\mathbf{A}, \mathbf{B}, \mathbf{a}, \mathbf{b})$, $\mathbf{A} \in \mathbb{R}^{H\times M}$, and $\mathbf{B} \in \mathbb{R}^{N\times H}$ are the weight matrices and $\mathbf{a} \in \mathbb{R}^{H}$ and $\mathbf{b} \in \mathbb{R}^{N}$ are the bias vectors of the first and second layer of the MLP network, respectively. The weight matrices and bias vectors will

hence be collectively called the weights of the MLP. The activation function ϕ is the standard hyperbolic tangent. It is applied component-wise to its argument vector. This approach is perfectly general because MLP networks are universal function approximators that can model any nonlinear mapping to arbitrary accuracy [31].

The noise $\mathbf{n}(t)$ and all the weight matrices and bias vectors of the MLP are a priori assumed to be Gaussian and independent of each other. The noise is assumed to have a general diagonal covariance and a hierarchical log-normal variance prior of the form

$$p\big(\mathbf{n}(t)|\mathbf{v}_n\big) = N\big(\mathbf{n}(t); \mathbf{0}, \operatorname{diag}\big(\exp(2\mathbf{v}_n)\big)\big), \tag{5}$$

$$p(v_{n_i}|m_{v_n}, v_{v_n}) = N\big(v_{n_i}; m_{v_n}, \exp(2v_{v_n})\big), \tag{6}$$

where $\mathbf{v}_n = (v_{n_1}, \ldots, v_{n_N})$ and $\operatorname{diag}(\mathbf{v})$ is a diagonal matrix with the elements of vector $\mathbf{v}$ on the main diagonal. The log-normal parametrisation is used because it is easy to define a hierarchical prior. As the variance model is not that important in this application, a conjugate inverse Gamma variance prior could be used here as well. Restriction to an isotropic noise model with the noise covariance of the form $\lambda\mathbf{I}$ is also possible. The noise model and the generative model (4) imply the likelihood

$$p\big(\mathbf{x}(t)|\boldsymbol{\theta}_\mathbf{f}, \mathbf{s}(t), \mathbf{v}_n\big) = N\big(\mathbf{x}(t); \mathbf{f}\big(\mathbf{s}(t), \boldsymbol{\theta}_\mathbf{f}\big), \operatorname{diag}\big(\exp(2\mathbf{v}_n)\big)\big). \tag{7}$$

The hierarchical priors of the MLP weights are similar to those of the noise except that the prior of $\mathbf{A}$ is fixed to unit variance to resolve the scaling ambiguity between $\mathbf{A}$ and $\mathbf{s}$, and the different columns of $\mathbf{B}$ have their own priors:

$$p(A_{ij}) = N(A_{ij}; 0, 1), \tag{8}$$

$$p(B_{ij}|v_{B_j}) = N\big(B_{ij}; 0, \exp(2v_{B_j})\big), \tag{9}$$

$$p(a_i|m_a, v_a) = N\big(a_i; m_a, \exp(2v_a)\big), \tag{10}$$

$$p(b_i|m_b, v_b) = N\big(b_i; m_b, \exp(2v_b)\big), \tag{11}$$

$$p(v_{B_j}|m_{v_B}, v_{v_B}) = N\big(v_{B_j}; m_{v_B}, \exp(2v_{v_B})\big). \tag{12}$$

The highest-level hyperparameters m_{v_n}, v_{v_n}, m_a, v_a, m_b, v_b, m_{v_B} and v_{v_B} have vague Gaussian priors $N(0, 100^2)$.[2]

To complete the definition of the model, the prior of the sources $\mathbf{s}$ must still be determined. In this case there are several possibilities, leading to models with different benefits and drawbacks. The differences between the alternatives are discussed in more detail in Section 4.4 below.

The simplest source model is the Gaussian model used in nonlinear factor analysis (NFA). This is achieved by

$$p\big(\mathbf{s}(t)|\mathbf{v}_s\big) = N\big(\mathbf{s}(t); \mathbf{0}, \operatorname{diag}\big(\exp(2\mathbf{v}_s)\big)\big), \tag{13}$$

$$p(v_{s_i}|m_{v_s}, v_{v_s}) = N\big(v_{s_i}; m_{v_s}, \exp(2v_{v_s})\big), \tag{14}$$

where the hyperparameters m_{v_s} and v_{v_s} again have noninformative prior $N(0, 100^2)$. With these definitions, the sets of observations $\boldsymbol{X} = \{\mathbf{x}(t)|t\}$ and sources $\boldsymbol{S} = \{\mathbf{s}(t)|t\}$ are defined as usual. The parameter vector $\boldsymbol{\theta}$ contains everything described above, that is $\boldsymbol{\theta} = (\boldsymbol{\theta}_\mathbf{f}, \mathbf{v}_n, m_{v_n}, v_{v_n}, (v_{B_j}), m_a, v_a, m_b, v_b, m_{v_B}, v_{v_B}, \mathbf{v}_s, m_{v_s}, v_{v_s})$ including all j in v_{B_j}.

The Gaussian model is computationally simple, but it suffers from the same rotation indeterminacy as the linear factor analysis (FA) model. This can be corrected in the same way linear independent factor analysis [2] extends linear FA: by using a mixture of Gaussians as the source prior. Introducing a new latent variable $M_i(t)$ to denote the active mixture component for source i at sample t leads to a nonlinear independent factor analysis (NIFA) model with a prior of the form

$$p(s_i(t)|M_i(t) = l, m_{\mathrm{sil}}, v_{\mathrm{sil}}) = N\big(s_i(t); m_{\mathrm{sil}}, \exp(2v_{\mathrm{sil}})\big), \tag{15}$$

$$p(m_{\mathrm{sil}}|v_{m_s}) = N\big(m_{\mathrm{sil}}; 0, \exp(2v_{m_s})\big), \tag{16}$$

$$p(v_{\mathrm{sil}}|m_{v_s}, v_{v_s}) = N\big(v_{\mathrm{sil}}; m_{v_s}, \exp(2v_{v_s})\big). \tag{17}$$

The mixing proportions have a logistic normal prior given by the softmax function

[2] The data is usually preprocessed by scaling to approximately unit variance to ensure proper scaling of the weights so that the priors really are vague.

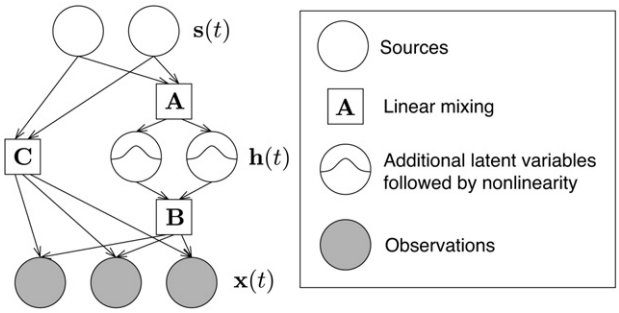

Fig. 1. The HNFA model is illustrated. Square nodes correspond to weight matrices and round nodes to the variables with shaded nodes being observed and unshaded latent.

$$p(M_i(t) = l|c_i) = \exp(c_{il}) \Big/ \sum_{l'} \exp(c_{il'}), \tag{18}$$

$$p(c_{il}|v_c) = N\big(c_{il}; 0, \exp(2v_c)\big). \tag{19}$$

All the highest level parameters v_{m_s}, m_{v_s}, v_{v_s}, and v_c again have a noninformative prior $N(0, 100^2)$. The parameters $\boldsymbol{\theta}$ can be defined similarly as above.

In the following, the simpler NFA model is used instead of the more complex NIFA model. More detailed justification for this is presented in Section 4.4. Most of the discussion generalises fairly easily to NIFA as well, as is shown in [9,32].

4.2. Hierarchical nonlinearity

The computational complexity of learning the NFA model is quadratic with respect to the number of the sources. This arises from the need to evaluate the Jacobian matrix of the outputs of the MLP network with respect to the inputs for each sample. The Jacobian is needed to reliably approximate the nonlinearity [15].

One way of defining a more efficiently learnable new model is to introduce additional latent variables to the hidden nodes of the MLP-like network as is done in the hierarchical nonlinear factor analysis (HNFA) model [10]. The HNFA model is defined by the equations

$$\mathbf{s}(t) \sim N(\mathbf{0}, \mathbf{I}), \tag{20}$$

$$\mathbf{h}(t) \sim N\big(\mathbf{A}\mathbf{s}(t) + \mathbf{a}, \operatorname{diag}\big(\exp(-\mathbf{v}_h)\big)\big), \tag{21}$$

$$\mathbf{x}(t) \sim N\big(\mathbf{C}\mathbf{s}(t) + \mathbf{B}\boldsymbol{\phi}\big(\mathbf{h}(t)\big) + \mathbf{b}, \operatorname{diag}\big(\exp(-\mathbf{v}_n)\big)\big), \tag{22}$$

where $\mathbf{h}(t)$ are the latent variables modelling the values of the hidden nodes, the vectors $\mathbf{v}_h$ and $\mathbf{v}_n$ parametrise the variances of the noise or innovation of the hidden nodes and the observations, respectively, $\boldsymbol{\phi}$ is a vector of activation functions and $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{a}$, $\mathbf{b}$ are the weights of the mapping. The activation function is chosen to be $\phi(y) = \exp(-y^2)$, for computational simplicity. The structure of the model is illustrated in Fig. 1.

When assigning hierarchical priors to the parameters of the mapping, we have taken into account the role of different parameters in the model. The weights between different layers, that is $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$, can be expected to scale according to the scales of inputs and outputs. Since all $s_i(t)$ are assumed normally distributed, there should be no significant differences in the scales of the weights leaving different nodes but the hidden nodes and observations can have different scales. This is reflected in the following parametrisation:

$$p(A_{ij}|v_{I,h_i}) = N\big(A_{ij}; 0, \exp(-v_{I,h_i})\big), \tag{23}$$

$$p(B_{ij}|v_{I,x_i}, v_{O,h_j}) = N\big(B_{ij}; 0, \exp(-v_{I,x_i} - v_{O,h_j})\big), \tag{24}$$

$$p(C_{ij}|v_{I,x_i}) = N\big(C_{ij}; 0, \exp(-v_{I,x_i})\big), \tag{25}$$

$$p(a_i) = N(a_i; 0, 1), \tag{26}$$

$$p(b_i) = N(b_i; 0, 1). \tag{27}$$

The parameters v_{O,h_j} reflect the different scales of weights leaving the hidden nodes and v_{I,h_i} and v_{I,x_i} correspond to the different scales of weights entering the hidden nodes and observations, respectively.

Finally, the parameters which model variances on logarithmic scale, that is $\mathbf{v}_h$, $\mathbf{v}_n$, v_{I,h_i}, v_{I,x_i}, and v_{O,h_j}, each have hyperparameters for mean and log-variance. Their priors are $N(0, \exp(7))$ and $N(0, \exp(4))$, respectively.

MLP networks with sigmoid activation function are known to be universal approximators [31]. Similarly, the set of deterministic HNFA mappings without the linear shortcut

$$f(\mathbf{s}) = \mathbf{B}\boldsymbol{\phi}\big(\mathbf{A}\mathbf{s}(t) + \mathbf{a}\big) + \mathbf{b}, \tag{28}$$

where $\phi(y) = \exp(-y^2)$, has the universal approximation property exactly analogous to that of MLP networks [33].

The HNFA model can be implemented efficiently using the building-blocks framework [28–30]. As the additional latent variables $\mathbf{h}(t)$ are independent by definition of the factorial posterior approximation, there is no need to evaluate any nontrivial Jacobians. The disadvantage of HNFA is that the approximation of the posterior density is farther away from the true posterior density. This may occasionally lead to inferior performance [34]. The linear shortcut is included in the model to partially help this, because representing nearly linear mappings would otherwise be significantly more difficult than with the MLP model.

4.3. Post-nonlinear model

As discussed in Section 2, post-nonlinear (PNL) mixtures are restricted nonlinear mixtures of the form

$$x_i(t) = f_i\left(\sum_{j=1}^{M} A_{ij}s_j(t)\right) + n_i(t), \quad i = 1, \ldots, N, \tag{29}$$

where the functions $f_i : \mathbb{R} \to \mathbb{R}$ are called the post-nonlinearities and n_i is additive Gaussian noise. PNL mixtures are a theoretically important special case of nonlinear ICA, because they can be proven to be separable in the noiseless case if the post-nonlinearities are invertible and the mixing matrix satisfies certain regularity conditions [19].

A variational Bayesian approach to post-nonlinear mixtures was first presented in [11]. The post-nonlinear factor analysis (PNFA) model is based on the generative model

$$x_i(t) = f_i\left(\sum_{j=1}^{M} A_{ij}s_j(t); \boldsymbol{\theta}_{f_i}\right) + n_i(t), \tag{30}$$

where the post-nonlinearities f_i are modelled with MLP networks with weights $\boldsymbol{\theta}_{f_i}$ as

$$f_i(y, \boldsymbol{\theta}_{f_i}) = \mathbf{D}_i\boldsymbol{\phi}(\mathbf{C}_i y + \mathbf{c}_i) + d_i, \tag{31}$$

where $\boldsymbol{\theta}_{f_i} = (\mathbf{C}_i, \mathbf{c}_i, \mathbf{D}_i, d_i)$ includes the column weight and bias vectors $\mathbf{C}_i, \mathbf{c}_i$, row weight vector $\mathbf{D}_i$ and bias scalar d_i. This approach also allows for noninvertible post-nonlinearities.

Most of the weights have priors governed by higher level hyperparameters

$$p(A_{ij}) = N(A_{ij}; 0, 1), \tag{32}$$

$$p(C_{ij}|v_{C_i}) = N\big(C_{ij}; 0, \exp(2v_{C_i})\big), \tag{33}$$

$$p(c_{ij}|m_{c_i}, v_{c_i}) = N\big(c_{ij}; m_{c_i}, \exp(2v_{c_i})\big), \tag{34}$$

$$p(D_{ij}|v_{D_i}) = N\big(D_{ij}; 0, \exp(2v_{D_i})\big), \tag{35}$$

$$p(d_i|m_d, v_d) = N\big(d_i; m_d, \exp(2v_d)\big), \tag{36}$$

with hierarchical priors for $v_{C_i}, m_{c_i}, v_{c_i}, v_{D_i}$ similar to those used in the NFA model in Eq. (12). The highest level parameters again have vague priors $N(0, 100^2)$.

The source model used in PNFA is Gaussian. It could in principle be rather easily replaced with mixtures-of-Gaussians, but simple post-processing of the extracted sources with linear ICA is often sufficient, as discussed below in Section 4.4.

4.4. On different source models

Two different source models were presented above: a Gaussian nonlinear factor analysis (NFA) model in Eq. (13) and a mixture-of-Gaussians based nonlinear *independent* factor analysis (NIFA) model in Eq. (15). From purely theoretical perspective, the NIFA model is preferable as its non-Gaussian model is able to resolve the rotation indeterminacy inherent to the Gaussian model.

In practice, things are not quite that simple. The NIFA model with its fully factorial posterior approximation seems to suffer from similar problems of not being able to extract the true independent sources as linear algorithms based on similar principles [34].

Using the more complicated NIFA model slows down the computation and requires careful initialisation of the mixture components. As this seems to cause more trouble than offer benefits, we have usually chosen not to do it and only use plain NFA. In order to achieve BSS, standard linear ICA can be applied as post-processing to the extracted sources. This approach resembles the one used in [35] for linear separation, where linear factor analysis is used as pre-processing for ICA instead of PCA. Use of factor analysis pre-processing has been shown to lead to good results in case of noisy linear mixtures. The nonlinear case is, however, more difficult as the model may try to nonlinearly transform the sources to be more Gaussian than they should be. Some evidence of this behaviour can be seen in the experiments in Section 6.2. Even with a Gaussian source model the sources are, however, not fully Gaussian and using linear ICA to determine the rotation is thus possible.

5. Learning

In this section, methods for learning the models presented in Section 4 are presented. The learning method for NFA is again heavily based on earlier work [9] but uses a new method of linearising the MLP [15] that leads to significantly more stable and reliable algorithm.

All the models presented in Section 4 are learned using variational Bayesian (VB) learning [36]. Given the data $\boldsymbol{X}$ and a model with parameters $\boldsymbol{\theta}$ and latent variables or sources $\boldsymbol{S}$, the method is based on approximating the posterior distribution $p(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{X}, \mathcal{H})$ with a factorial approximation $q(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{\xi}) = q(\boldsymbol{S}|\boldsymbol{\xi}_{\boldsymbol{S}})q(\boldsymbol{\theta}|\boldsymbol{\xi}_{\boldsymbol{\theta}})$, where $\boldsymbol{\xi}$ are the variational parameters and $\mathcal{H}$ represents the background assumptions of a particular model. The approximation is fitted by minimising the Kullback–Leibler divergence between q and p, or equivalently maximising a lower bound on marginal log-likelihood of the data. By changing the sign, this can be formulated as minimisation of the cost

$$\mathcal{C} = \left\langle \log \frac{q(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{\xi})}{p(\boldsymbol{S}, \boldsymbol{\theta}, \boldsymbol{X}|\mathcal{H})} \right\rangle = D_{\mathrm{KL}}\big(q(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{\xi})||p(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{X}, \mathcal{H})\big) - \log p(\boldsymbol{X}|\mathcal{H}) \geqslant -\log p(\boldsymbol{X}|\mathcal{H}), \tag{37}$$

where $\langle\cdot\rangle$ denotes the expectation over q and $D_{\mathrm{KL}}(q||p)$ is the Kullback–Leibler divergence between q and p.

Most applications of variational Bayesian learning [37] involve only models in the conjugate exponential family. The models described in Section 4 are clearly not in the exponential family, so the learning methods are somewhat different and require numerical minimisation of the cost or even only its approximation.

In almost all the models, we use the naïve mean field approximation, that is a fully factorial Gaussian distribution

$$q(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{\xi}) = q(\boldsymbol{S}|\boldsymbol{\xi}_{\boldsymbol{S}})q(\boldsymbol{\theta}|\boldsymbol{\xi}_{\boldsymbol{\theta}}) = \prod_{i,t} q\big(s_i(t)|\boldsymbol{\xi}_{s_i(t)}\big) \prod_j q(\theta_j|\boldsymbol{\xi}_{\theta_j}). \tag{38}$$

The individual factors are parametrised with variational parameters corresponding to the posterior mean and variance of the variable as

$$q\big(s_i(t)|\boldsymbol{\xi}_{s_i(t)}\big) = N\big(s_i(t); \bar{s}_i(t), \tilde{s}_i(t)\big), \tag{39}$$

$$q(\theta_j|\boldsymbol{\xi}_{\theta_j}) = N(\theta_j; \bar{\theta}_j, \tilde{\theta}_j). \tag{40}$$

For parameters modelling the mean of a Gaussian, the Gaussian distribution is a conjugate prior and the optimal free-form approximation is of this form, assuming the factorisation. For parameters modelling the log-variance of a Gaussian, the Gaussian prior is not conjugate and the posterior approximation is only an approximation.

The definition of the model and the approximating distribution allow evaluating the cost (37) as a function of the variational parameters $\boldsymbol{\xi}$. The cost can be split into two parts as the negative entropy of the approximation and an expected energy

$$\mathcal{C} = \mathcal{C}_q + \mathcal{C}_p = \big\langle \log q(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{\xi}) \big\rangle + \big\langle -\log p(\boldsymbol{X}, \boldsymbol{S}, \boldsymbol{\theta}|\mathcal{H}) \big\rangle. \tag{41}$$

Using the factorisation of q, the negative entropy term $\mathcal{C}_q$ splits into

$$\begin{aligned}\mathcal{C}_q &= \big\langle \log q(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{\xi}) \big\rangle = \left\langle \log \prod_{i,t} q\big(s_i(t)|\boldsymbol{\xi}_{s_i(t)}\big) \prod_j q(\theta_j|\boldsymbol{\xi}_{\theta_j}) \right\rangle \\ &= \sum_{i,t} \big\langle \log q\big(s_i(t)|\boldsymbol{\xi}_{s_i(t)}\big) \big\rangle + \sum_j \big\langle \log q(\theta_j|\boldsymbol{\xi}_{\theta_j}) \big\rangle. \end{aligned} \tag{42}$$

The remaining terms are negative entropies of univariate Gaussians having values depending only on the variance

$$\big\langle \log q(\theta_j|\boldsymbol{\xi}_{\theta_j}) \big\rangle = -\frac{1}{2} - \frac{1}{2} \log\big(2\pi\tilde{\theta}_j\big). \tag{43}$$

The expected energy term $\mathcal{C}_p$ is more model specific and is presented separately for the different models.

5.1. MLP nonlinearity

While basic NFA uses a fully factorial posterior approximation, this is not the case for NIFA. Instead, the dependences between $M_i(t)$ and $s_i(t)$ are modelled so that the approximation for them is of the form

$$q\big(M_i(t), s_i(t)\big) = q\big(s_i(t)|M_i(t)\big) q\big(M_i(t)\big) \tag{44}$$

which yields a Gaussian mixture approximation for $s_i(t)$

$$q\big(s_i(t)\big) = \sum_l q\big(s_i(t)|M_i(t) = l\big) q\big(M_i(t) = l\big). \tag{45}$$

Otherwise the approximation is similar to NFA.

5.1.1. Evaluating the cost

Evaluating the expected energy term $\mathcal{C}_p$ is slightly more difficult than term $\mathcal{C}_q$. Using the definition of the model, it can for NFA and PNFA models be factored into terms

$$\begin{aligned}\mathcal{C}_p &= \big\langle -\log p(\boldsymbol{X}|\boldsymbol{S}, \boldsymbol{\theta}, \mathcal{H}) \big\rangle + \big\langle -\log p(\boldsymbol{S}|\boldsymbol{\theta}, \mathcal{H}) \big\rangle + \big\langle -\log p(\boldsymbol{\theta}|\mathcal{H}) \big\rangle \\ &= \sum_{i,t} \big\langle -\log p\big(x_i(t)|\mathbf{s}(t), \boldsymbol{\theta}, \mathcal{H}\big) \big\rangle + \sum_{i,t} \big\langle -\log p\big(s_i(t)|\boldsymbol{\theta}, \mathcal{H}\big) \big\rangle + \big\langle -\log p(\boldsymbol{\theta}|\mathcal{H}) \big\rangle. \end{aligned} \tag{46}$$

The terms in the second and third summand are expectations of negative logarithm of Gaussian pdf over Gaussian mean and log-variance parameters. These can be evaluated for example parameter $\theta \sim N(m, \exp(2v))$ through integrals of the form [38]

$$\begin{aligned}\big\langle -\log p(\theta|m, v, \mathcal{H}) \big\rangle &= \iiint -\log N\big(\theta; m, \exp(2v)\big) q(\theta) q(m) q(v)\, d\theta\, dm\, dv \\ &= \frac{1}{2} \log(2\pi) + \bar{v} + \frac{1}{2} \big[(\bar{\theta} - \bar{m})^2 + \tilde{\theta} + \tilde{m} \big] \exp(2\tilde{v} - 2\bar{v}). \end{aligned} \tag{47}$$

The terms of the first summand of Eq. (46),

$$\begin{aligned}\big\langle -\log p\big(x_i(t)|\mathbf{s}(t), \boldsymbol{\theta}, \mathcal{H}\big) \big\rangle &= \big\langle -\log N\big(x_i(t);\, f_i\big(\mathbf{s}(t), \boldsymbol{\theta}_\mathbf{f}\big), \exp(v_{n_i})\big) \big\rangle \\ &= \frac{1}{2} \log(2\pi) + \bar{v}_{n_i} + \frac{1}{2} \big[\big(x_i(t) - \bar{f}_i(t)\big)^2 + \tilde{f}_i(t) \big] \exp\big(2\tilde{v}_{n_i} - 2\bar{v}_{n_i}\big), \end{aligned} \tag{48}$$

are more difficult as they depend on the mean $\bar{f}_i(t)$ and variance $\tilde{f}_i(t)$ of the outputs of the MLP network. Techniques for approximating these are presented in Appendix A.

5.1.2. Update algorithm

The posterior approximations of the parameters of the hierarchical model, that is those of the type m_θ and v_θ, can be updated using a standard VB EM algorithm [9,38]. The update rules for the sources $\boldsymbol{S}$ as well as the weights $\boldsymbol{\theta}_\mathbf{f}$ of the MLP network are more difficult and they are therefore presented in more detail.[3]

The variational parameters corresponding to the means of $\boldsymbol{S}$ and $\boldsymbol{\theta}_\mathbf{f}$ are updated with a conjugate gradient algorithm. The required derivatives can be computed from the expression of the cost function. In case of inputs and weights of the MLP, this leads to similar computation of partial derivatives as in back-propagation. The parameters corresponding to the variances are updated using a fixed point algorithm that can be derived by differentiating the split cost (41) with respect to $\tilde{\theta}_j$ and using the evaluated result (43). This yields

$$\frac{\partial \mathcal{C}}{\partial \tilde{\theta}_j} = \frac{\partial \mathcal{C}_q}{\partial \tilde{\theta}_j} + \frac{\partial \mathcal{C}_p}{\partial \tilde{\theta}_j} = -\frac{1}{2\tilde{\theta}_j} + \frac{\partial \mathcal{C}_p}{\partial \tilde{\theta}_j}. \tag{49}$$

Setting this to zero leads to a fixed point update rule for the variances of the sources and MLP network weights

$$\tilde{\theta}_j = \frac{1}{2}\left(\frac{\partial \mathcal{C}_p}{\partial \tilde{\theta}_j}\right)^{-1}. \tag{50}$$

Blindly applying this rule may in some cases lead to instability. This can be corrected by some form of dampening, such as halving the step length on logarithmic scale until the update does not increase the value of the cost function. The variance must also not be set to a negative value, even if the derivative is negative.

5.1.3. Initialisation

The MLP network and the gradient-based learning algorithms are notoriously prone to local minima [20,40]. In order to achieve good results, the methods require a reasonable initialisation.

The NFA method is typically initialised by setting the means of the sources to values given by suitable number of principal components of the data. The means of the MLP weights are initialised to random values while all the variances are initialised to small constant values. After this, only the MLP weights are updated during the first 20 iterations of the update algorithm so that the model can learn a mapping from the PCA sources to the observations. The hyperparameters of the model are only updated after the first 100 iterations.

The PCA initialisation is easy to compute and sufficient for many purposes, but its linearity may sometimes lead to suboptimal results. To resolve this, kernel PCA (KPCA) was used in the initialisation in [14]. The KPCA initialisations are sensitive to the choice of the kernel and its parameters, but with suitable choices they may help NFA yield significantly better results while using less time. The kernel can also be selected with the variational Bayesian criterion by running the learning algorithm for a few iterations with different initialisations and comparing the cost function values.

5.2. Hierarchical nonlinearity

The HNFA model is defined by a graphical model of Gaussian variables combined using addition, multiplication and nonlinearities directly following the variables. Such models can be implemented using the building-blocks framework introduced in [28–30]. The learning scheme is designed to minimise the cost function

$$\mathcal{C} = \left\langle \log \frac{q(\boldsymbol{S}, \boldsymbol{H}, \boldsymbol{\theta}|\boldsymbol{\xi})}{p(\boldsymbol{S}, \boldsymbol{H}, \boldsymbol{\theta}, \boldsymbol{X}|\mathcal{H})} \right\rangle, \tag{51}$$

where $\boldsymbol{H} = \{\mathbf{h}(t)|t\}$ is the set of all hidden nodes. The cost function is otherwise the same as Eq. (37), except that the hidden nodes $\boldsymbol{H}$ have been included as unknown variables.

The basic operation during learning is an iteration where all the terms $q(\theta_i|\boldsymbol{\xi}_{\theta_i})$, $q(s_i(t)|\boldsymbol{\xi}_{s_i(t)})$ and $q(h_i(t)|\boldsymbol{\xi}_{h_i(t)})$ of the approximation

$$\begin{aligned} q(\boldsymbol{S}, \boldsymbol{\theta}, \boldsymbol{H}|\boldsymbol{\xi}) &= q(\boldsymbol{S}|\boldsymbol{\xi}_{\boldsymbol{S}}) q(\boldsymbol{\theta}|\boldsymbol{\xi}_{\boldsymbol{\theta}}) q(\boldsymbol{H}|\boldsymbol{\xi}_{\boldsymbol{H}}) \\ &= \prod_{i,t} q\big(s_i(t)|\boldsymbol{\xi}_{s_i(t)}\big) \prod_j q(\theta_j|\boldsymbol{\xi}_{\theta_j}) \prod_{i,t} q\big(h_i(t)|\boldsymbol{\xi}_{h_i(t)}\big) \end{aligned} \tag{52}$$

[3] A Matlab toolbox [39] implementing the NFA method is available at http://www.cis.hut.fi/projects/bayes/software/.

are updated one at a time by minimising (51) with respect to the corresponding variational parameters $\boldsymbol{\xi}$. The approximation is again otherwise the same as Eq. (38), except for the inclusion of the hidden nodes $\boldsymbol{H}$. Details of the minimisation of the q-terms are presented in [30, Appendices A and C].[4]

In addition to the basic updates, several other operations are performed in order to help avoid local optima and speed up learning:

- addition of hidden nodes;
- addition of weights;
- pruning of weights;
- line search.

Line search has been explained in [42]. The idea is to monitor the individual updates during one iteration and then perform a line search simultaneously for all terms of the approximation. We applied the line search after every tenth iteration.

The addition and pruning operations aim at optimising the model structure. Incremental addition of hidden nodes to the HNFA model is necessary because the hidden nodes are "expensive" and thus easily pruned out by making their outgoing weights approach to zero. The cost function (37) relates to the marginal likelihood $p(\boldsymbol{X}|\mathcal{H})$ which can be used to find the most likely model structure.

In general, addition takes place randomly and pruning is based on estimating whether the cost function can be decreased by removing a weight. The motivation for this is that variational Bayesian learning can effectively prune out parts of the model which are not needed. The weights in the matrix $\mathbf{B}$ corresponding to one hidden node can for instance approach zero. The cost function can usually be decreased by removing such weights. If all outgoing weights of a hidden node have been removed, the hidden node becomes useless and can be removed from the model. Variational Bayesian learning cannot, however, actively make room for a part of the model which may be added in the future. It usually takes some time for the rest of the model to accommodate to additions.

5.2.1. Evidence node

In order to avoid local minima in variational Bayesian learning, it is necessary to initialise some variables and keep them fixed for a while until other parts of the model have accommodated appropriately. In the building-blocks framework, we have adopted the virtual-evidence approach [43]. This is implemented by attaching evidence nodes [44], as we call them, to a variable θ_i, whose value we want to set. The node provides a term for the likelihood $p(\text{children}|\theta_i)$. When $q(\theta_i|\boldsymbol{\xi}_{\theta_i})$ is updated, the distribution of θ_i will be close to the value set by evidence node if the likelihood term has a narrow peak but θ_i can accommodate to other parts of the model if the likelihood term is wide. Since all the prior distributions in the present model are Gaussian, it is convenient to use the same functional form for the evidence nodes:

$$-\log p\big(\text{virtual child}|\theta_i;\alpha_i(\tau_i),\mu_i,\sigma_i^2\big)=\alpha_i(\tau_i)\frac{(\theta_i-\mu_i)^2}{2\sigma_i^2}, \tag{53}$$

where μ_i and σ_i^2 are the mean and variance induced by the evidence node. The term $\alpha(\tau_i)$ controls the decay of the virtual evidence. It decreases linearly from $\alpha(0)=1$ to $\alpha(T)=0$ after which the evidence node is removed. The persistence of the initialisation can thus be controlled by the life-span T (iterations) of the evidence node.

5.2.2. Phases of learning

The model is built in stages. First, only the linear mapping $\mathbf{C}\mathbf{s}(t)$ is used, so that there are no hidden nodes. The sources $\mathbf{s}(t)$ are initialised by principal component analysis (PCA) using evidence nodes with a life-span of 40 iterations and variance $\sigma^2=10^{-3}$ in order to estimate a reasonable $\mathbf{C}$. The linear model is learned for 100 iterations.

After that, 50 randomly initialised hidden nodes are added to the model and estimation of the model structure begins. That is, weights are added and pruned and hidden nodes are added every now and then. Every time new hidden nodes are added, five of them are selected from a pool of 1000 random candidates. After each addition of

[4] The Bayes Blocks toolbox [41] and the code for the HNFA method are available at http://www.cis.hut.fi/projects/bayes/software/.

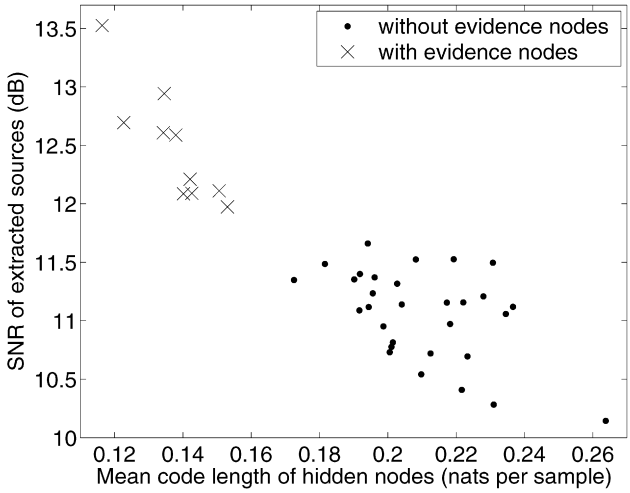

Fig. 2. The attained signal-to-noise ratio of the sources separated using HNFA with and without evidence nodes [45].

hidden nodes, there is a period of 30 iterations during which no pruning is applied. This gives the new hidden nodes enough time to fit themselves into the model.

Hidden nodes are added a limited number of times. After that, learning continues with pruning and random additions of weights. The number of weights to be added decreases with time. Finally, only line searches are applied for the last 1000 iterations. The total number of iterations in the HNFA simulations is 10,000 unless otherwise stated.

5.2.3. Addition of hidden nodes

The hidden nodes are latent variables which can independently represent some aspects of the observations. Due to our model structure, this usually corresponds to a local minimum of the cost function. It is better that the sources $\mathbf{s}(t)$ represent the data since they can share their information with all hidden nodes. Local minima can be avoided by evidence nodes which keep the variance of the Gaussian noise, innovation, associated to each newly added hidden node $h_i(t)$ through the corresponding parameter v_{h_i} low. Once the sources take the responsibility for the representation, the variances of hidden nodes no longer grow significantly. In our experiments, the life-span of these evidence nodes was 500 iterations and their mean and variance were set to 9 and unity, respectively. Since the parameter controls the logarithm of the inverse variance, value of 9 corresponds to standard deviation of approximately $1/90$.

Figure 2 shows a clear correlation of the quality of the results for the experimental setting described in Section 6.1, and a measure of the amount of independent innovation in the hidden nodes, the latter of which can be influenced by introducing the evidence nodes. More detailed discussion is presented in [45].

In addition to restricting the innovations, the incoming weights $\mathbf{A}$ of the hidden nodes are initialised to random values by evidence nodes with variance $\sigma^2 = 10^{-2}$ and life time of 40 iterations, when new nodes are added. However, after the first addition, the added hidden nodes are selected from a large pool of candidate initialisations. The hidden nodes which correlate best with the remaining modelling error of the observations are selected. The first hidden nodes are able to model many of the large scale nonlinearities in the data. It is much more difficult to find useful hidden nodes by random initialisations later on since the new nodes tend to be quickly pruned away.

5.3. Post-nonlinear model

The evaluation of the cost function and the learning process of the PNFA model are similar to those of the general NFA. Approximating the nonlinearity is a little easier, as there is only one "source" input with larger posterior variance. The posterior variances of the MLP weights are typically smaller as a single weight affects and thus gains evidence from several observations. This allows for using a hybrid of a Gaussian quadrature with respect to the MLP inputs $y_i(t) = \sum_{j=1}^{M} A_{ij} s_j$ and a Taylor approximation with respect to the weights $\boldsymbol{\theta}_{f_i}$, as presented in Appendix A. The more general approach used with NFA could of course be used as well.

Using the approximation of the nonlinearity presented in Appendix A, the PNFA cost function can be evaluated using Eqs. (41)–(43) and (46)–(48). The sources $\mathbf{s}$ and all the weights $\mathbf{A}$ and $\boldsymbol{\theta}_{f_i}$ are updated by conjugate gradient and all the other parameters by VB EM, as described for the NFA model in Section 5.1.2.

The initialisation of the PNFA method is similar to NFA: the sources are initialised with PCA. It would be possible to use other simple post-nonlinear methods such as Gaussianisation [27]. After the initialisation, the sources and the hyperparameters are kept fixed for the first 100 and 150 iterations, respectively, while the linear mapping $\mathbf{A}$ and the post-nonlinearities f_i are updated.

5.4. Missing values

The Bayesian framework facilitates principled handling of missing or partially missing elements in the data matrix [44]. Assuming the elements are missing at random, they provide no evidence and can hence be simply ignored in learning. After learning, the missing values can be reconstructed by using the posterior predictive distribution.

Missing values in nonlinear FA have been studied using NFA and HNFA models in [12,13].

6. Experiments

In this section, we present the results of two comparisons of the methods using artificial data sets. The artificial data sets are necessary for comparing different methods as the true sources are not known in most real applications and comparison of different methods is thus practically impossible.

6.1. General nonlinear mixtures

The NFA and HNFA methods were tested with an artificial example. The data set consisted of 1000 samples from nonlinear mixtures of eight sources. Four of the sources were super-Gaussian and the remaining four were sub-Gaussian. The nonlinear mixing was a randomly initialised MLP network with $\sinh^{-1}$ as the activation function for the 30 hidden nodes. The standard deviation of the additive Gaussian noise was one tenth of that of the signal. This corresponds to a signal-to-noise ratio (SNR) of 20 dB. The same data set was also used in [10] and a noiseless version of it in [9].

Linear models (implemented as HNFA models without any hidden nodes) as well as NFA and HNFA models were tested with varying number of sources. The values of the cost function attained after learning were compared. The best linear model had 14 sources while the best NFA and HNFA models had eight sources. Hence both NFA and HNFA methods are able to infer the correct subspace dimension, while the linear model tries to model nonlinear effects using extra dimensions.

In addition to the proposed variational methods, the experiment was repeated using maximum a posteriori (MAP) estimation for the sources and MLP weights of the NFA model (MAP-NFA). In this case all the priors and the model structure were fixed to the their generative values.

Unfortunately most other nonlinear BSS methods do not scale to problems of this size. For Kernel PCA [25] the size is not a problem, but out of several Gaussian and polynomial kernels tested, the best results were attained with a linear kernel which is equivalent to linear PCA. Finding a better kernel may certainly be possible, but at least very difficult. We also tried the MISEP method [22], but were unable to attain better results than using linear methods. Again, this may be caused by the authors' lack of experience with MISEP.

The experiment was repeated a number of times with different random initialisations. Because of the flexibility of the nonlinear models, each of these ended in a different local optimum. The variation among the results is illustrated in Fig. 3 in terms of signal-to-noise ratio (SNR) with respect to the true sources after rotation of the extracted sources by symmetric FastICA [1]. The results shown are from 10 experiments with VB-NFA, 42 experiments with HNFA and 20 experiments with MAP-NFA. The results show that VB-NFA consistently finds very good solutions while there is lot more variation in the results of HNFA and especially MAP-NFA.

The figure also shows VB-NFA to be clearly superior to MAP-NFA, which suffers from serious overfitting. Additionally, there is relatively strong correlation between the VB cost function value and SNR, making it easy to find the best solutions among the alternatives. For MAP-NFA, the correlation between posterior density value and SNR is significantly weaker.

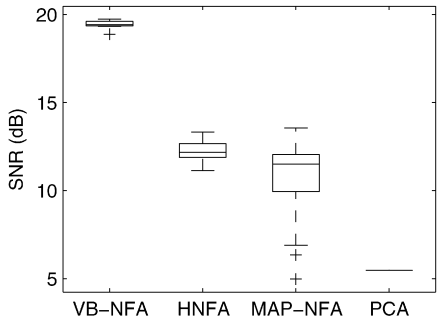

Fig. 3. Boxplot showing variation in the results of different algorithms in the general artificial mixture example.

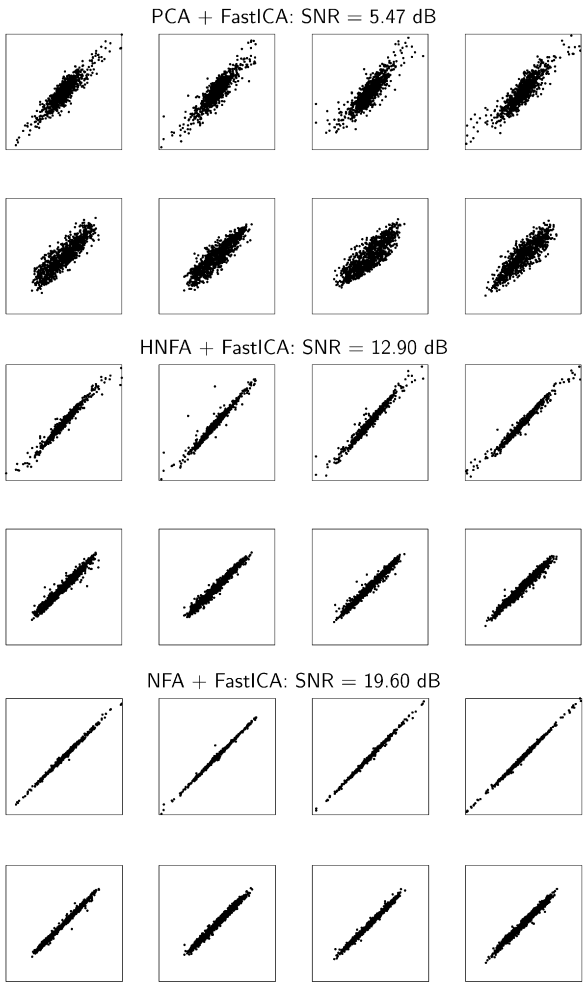

Fig. 4. Each scatter plot shows the values of one original source signal plotted against the best corresponding estimated source signal after a rotation with FastICA. The perfect result would be a straight line.

After a nonlinear subspace has been estimated by NFA or HNFA, standard linear ICA algorithms [1] can be used for rotating the subspace to obtain independent source signals. Figure 4 shows scatter plots of the original sources and the sources obtained after a rotation by symmetric FastICA. The plots are from the best results in Fig. 3 according to the VB cost function, hence they are not necessarily the ones with the best overall SNR. On the first two rows,

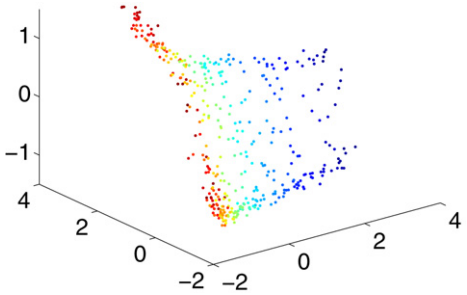

Fig. 5. The data set from the PNL mixture experiment.

standard linear PCA has been used to estimate the subspace. The middle two rows show the results when the subspace was estimated using the HNFA method and the two bottom rows when the NFA method was used.

The results show that linear methods alone are not sufficient to separate the sources. The results attained by HNFA are reasonably good while those of NFA are already very good. The difference in performance between the methods may be partly due to the fact that the generative nonlinearity more closely resembles that used in the NFA model, even though it is not the same. The computation times required by NFA and HNFA in this experiment are comparable.

6.2. *Post-nonlinear (PNL) mixtures*

The proposed NFA and PNFA methods were also tested with a three-dimensional PNL mixture of two independent sources. The sources were a sine wave and uniformly distributed white noise. The PNL transformation used for generating the data contained two noninvertible post-nonlinear distortions:

$$\mathbf{y} = \begin{bmatrix} 1.2 & 0.2 \\ 1 & 0.7 \\ 0.2 & 0.8 \end{bmatrix} \mathbf{s}; \qquad \mathbf{x} = \begin{bmatrix} (y_1 - 0.5)^2 \\ (y_2 + 0.4)^2 \\ \tanh(2y_3) \end{bmatrix}. \tag{54}$$

The observations were centred and normalised to unit variance and observation noise with variance 0.01 was added. The number of samples was 400. The data set is illustrated in Fig. 5.

The NFA and PNFA models were trained by trying different model structures, i.e., different numbers of hidden nodes in the MLPs, and several random initialisations of weights of the linear and nonlinear mappings. The PNFA model with the lowest cost had 5 nodes in the hidden layers of all MLPs. The corresponding NFA model with the lowest cost had 10 hidden nodes in the single MLP. The PNFA model was trained for 10,000 iterations and NFA for 5000 iterations.

The sources found by NFA and PNFA were further rotated by the FastICA algorithm to obtain independent signals (see Fig. 6). The scatter plots in Fig. 6a show how well the original sources were reconstructed. Each point corresponds to one source $s_i(t)$. The abscissa of a point is the original source which was used for generating the data and the ordinate is the estimated source. The optimal result would be a straight line which would mean that the estimated values of the sources coincide with the true values. For PNFA, the sources were estimated quite well except for some points at the edges, while the sources recovered by NFA are not as good.

The PNL distortions learned by the best PNFA model are presented in Fig. 7. The post-nonlinearities f_i are estimated quite well except for some points at the edges. The difficulties mostly affect the two quadratic functions which are difficult to model with such small MLP networks and relatively few observations, especially at those edges.

As expected, regular PNL ICA methods using invertible post-nonlinearities cannot recover the true sources at all. This is illustrated in Fig. 8, which shows a relatively clean separation by PNFA compared to severely tangled solution by Taleb–Jutten algorithm [19].

The mediocre overall quality of the results is understandable due to the great difficulty of the test problem: There are only two bounded sub-Gaussian sources in the mixture and their linear combinations are quite far from Gaussianity assumed by NFA and PNFA. Another difficulty is the complex PNL mapping with a small number of observations and several noninvertible post-nonlinear distortions. Removing any of the observations from the mixture would make the mixing process noninjective and the separation problem unsolvable. The results of PNFA are slightly better than those of regular NFA, which is also natural as the data follows the more restricted model.

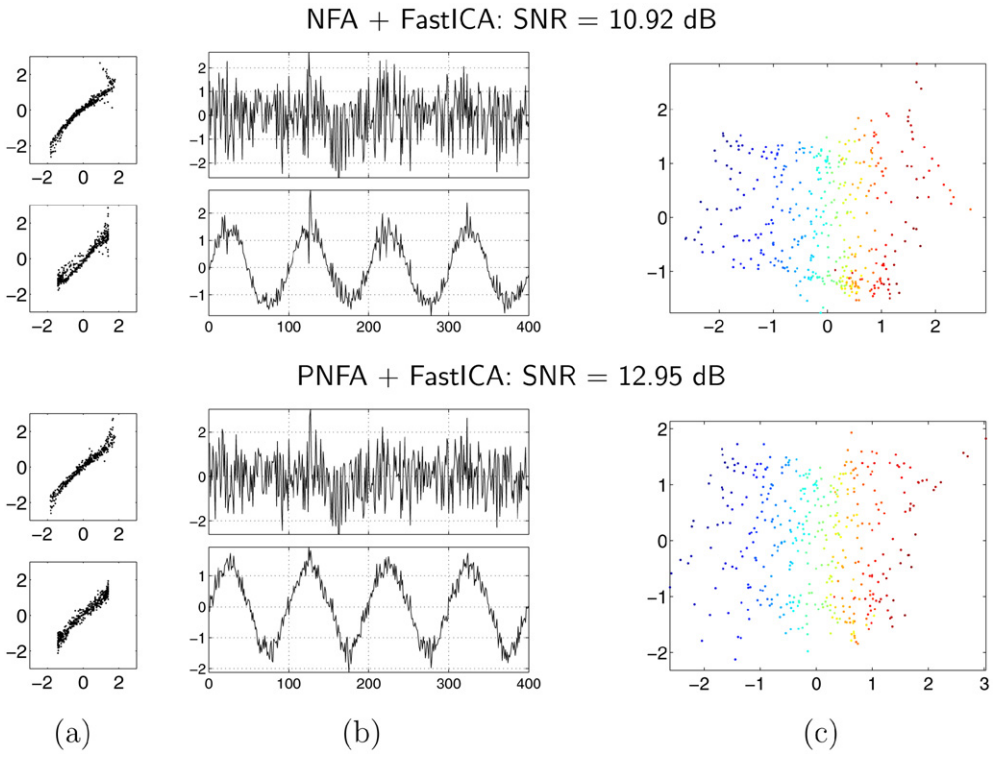

Fig. 6. The sources found by the NFA (top) as well as PNFA (bottom) and further rotated with the FastICA algorithm. (a) The scatter plots; (b) the estimated time series; (c) the distribution of the sources. The signal-to-noise ratio is 10.92 dB for NFA and 12.95 dB for PNFA.

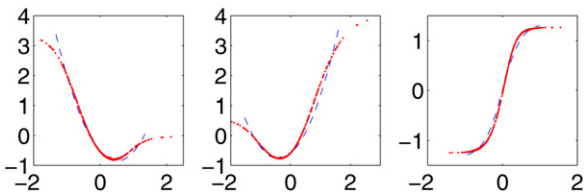

Fig. 7. The estimated post-nonlinear distortions f_i against the functions used for generating the data (the dashed line). Each point in the figure corresponds to a single observation.

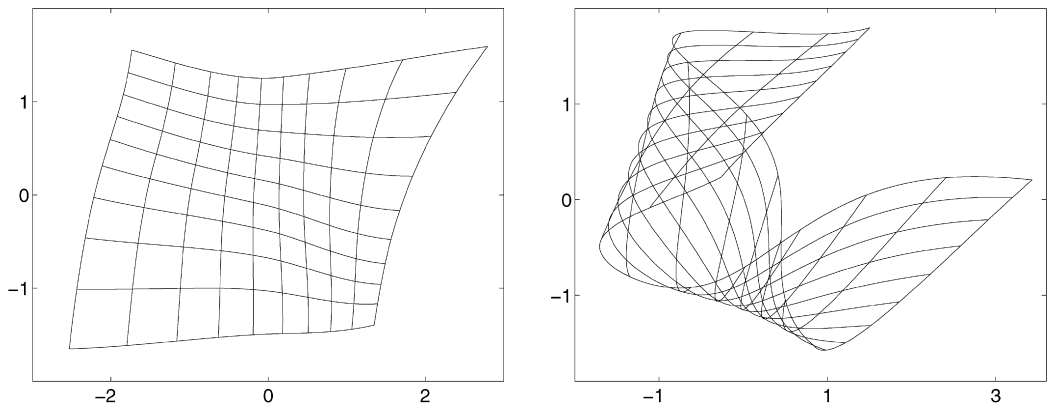

Fig. 8. A grid of original sources $\mathbf{s}$ as mapped through the mixing to $\mathbf{x}$ and demixing to their reconstructions $\hat{\mathbf{s}}$: the perfect solution would be a regular square. Because of the noise, the mappings are only approximate. The results are shown for the PNFA method (left) and Taleb–Jutten PNL ICA algorithm [19] based on invertible post-nonlinearities f_i (right).

7. Discussion

Nonlinear BSS is a very difficult problem, as illustrated by the nonuniqueness results for nonlinear FA and ICA. From a Bayesian perspective, having a multitude of potential solutions that all should somehow be taken into account does not sound so unreasonable. The Bayesian averaging procedure together with reasonable priors on the mapping will help avoid most overly complicated solutions. This was also illustrated experimentally in Section 6.1. Using a slightly simpler EM-like approach with point estimates for the parameters of the nonlinear mapping would probably be enough to avoid most overfitting problems. However, as integration over the sources is the most complicated operation, the computational savings would be limited and the ability to infer the structure of the nonlinearity would be lost.

The variational approach used in this work has also other properties that further help regularise the problem. The flexible nonlinear model has many internal symmetries that are reflected in the true posterior. For purposes of the source separation problem, these symmetries are, however, not interesting. The variational approximation breaks the symmetry by ignoring the dependencies between different groups of parameters and the sources. The unimodal posterior approximation will find a broad region of potential solutions, thus returning essentially a single most plausible solution.

The regularisation power of the variational Bayesian approach is nicely demonstrated in [46], where HNFA is shown to prefer a linear model for a magnetoencephalographic (MEG) brain imaging data set where there are theoretical grounds to assume a linear mixing process. The example with approximately 50 sources extracted from 122 channels also demonstrates the scalability of HNFA, using NFA in a problem of this size would be infeasible.

Instead of a nonlinear model with a non-Gaussian prior capable of separating independent sources, simpler Gaussian source prior and post-processing by linear ICA has been used in this work to separate the independent sources. The main reason for using this suboptimal process is practical: even with a non-Gaussian source prior the factorial source posterior approximation would not allow determining the proper rotation of the independent sources [34]. This could be corrected by more advanced approximation techniques [5]. Application of such techniques for nonlinear models is an interesting topic of future research.

In this work layered mappings have been used to model the nonlinearities. This gives a quite strong prior that allows us to solve the difficult example cases demonstrated in Section 6. The broad literature on applications of MLP networks on a wide variety of real world problems provides empirical evidence that such a prior may also be useful in real world applications. Still, having a more easily interpretable prior for the nonlinearities through a Gaussian process [26], for instance, would be interesting.

The presented HNFA and NFA methods have also been compared in reconstructing missing values in speech spectrograms in [30]. According to those experiments, the performance of HNFA was between linear FA and NFA when there were major differences. This seems to indicate that HNFA may emphasise less nonlinear solutions than NFA through stronger regularisation. Thus HNFA is likely better suited for mildly nonlinear problems.

8. Conclusion

We have presented a number of methods for BSS of nonlinear mixtures based on variational Bayesian learning. The basic NFA method uses a MLP network to model the nonlinearity. With a suitable linearisation technique, the variational Bayesian methodology can be easily applied to perform learning and inference on the model much more reliably than before [9]. The computational complexity of the resulting algorithm is quadratic in the number of the sources. To avoid this, an alternative HNFA method based on a hierarchical nonlinearity was proposed. The attained speedup comes at the expense of less accurate results as there are more latent variables in the model and the variational approximate posterior provides a poorer fit to the true posterior. The speeds of NFA and HNFA are comparable in the examples with approximately 10 sources presented in Section 6, but better scalability allows applying HNFA to larger problems, where no other alternatives are available. Additionally, a variant of NFA specifically designed for more restricted post-nonlinear mixtures was proposed.

Although HNFA seems less accurate in comparison to NFA in the experiment in Section 6.1, the same is not true in general. The comparison of Section 6.1 most likely favours NFA somewhat, because the structure of the generative nonlinearity is closer to that used in the NFA model, even though it is not the same. Other comparisons, such as some of the missing data experiments reported in [30] show HNFA outperform NFA.

Most of the proposed methods employ a Gaussian prior on the sources and are thus unable to perform source separation by themselves. Still, post-processing the recovered sources by standard linear ICA was enough to recover the underlying independent sources in the presented examples. Using a non-Gaussian source model does not necessarily solve the problem, as ignoring the posterior correlations between the sources in the variational approximation may cause the model to prefer more orthogonal PCA-like solutions. Extending the model by using a Gaussian mixture source model and modelling the posterior correlations of the sources would be an interesting line of future work. The computational complexity of such a model is exponential in the number of the sources, but it would probably help in low dimensional cases.

In order to allow others use the presented methods more easily, free software implementations of the NFA and HNFA methods [39,41] are available on the web at http://www.cis.hut.fi/projects/bayes/software/.

Acknowledgments

The authors wish to thank Tapani Raiko for useful comments and discussions and Tomas Östman for his help with the HNFA experiments. This work was supported in part by the IST Programme of the European Community, under the project BLISS, IST-1999-14190, and under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

Appendix A. Approximating the MLP nonlinearity

The mean $\bar{f}_i$ and variance $\tilde{f}_i$ of the outputs of the MLP used in Eq. (48) can be evaluated by multidimensional Gaussian integrals

$$\bar{f}_i(t) = \int\int f_i\big(\mathbf{s}(t), \boldsymbol{\theta}_\mathbf{f}\big) q\big(\mathbf{s}(t), \boldsymbol{\theta}_\mathbf{f}|\boldsymbol{\xi}\big)\, d\mathbf{s}(t)\, d\boldsymbol{\theta}_\mathbf{f}, \tag{A.1}$$

$$\tilde{f}_i(t) = \int\int \big(f_i\big(\mathbf{s}(t), \boldsymbol{\theta}_\mathbf{f}\big) - \bar{f}_i(t)\big)^2 q\big(\mathbf{s}(t), \boldsymbol{\theta}_\mathbf{f}|\boldsymbol{\xi}\big)\, d\mathbf{s}(t)\, d\boldsymbol{\theta}_\mathbf{f}. \tag{A.2}$$

These integrals depend on all the inputs and weights of the MLP network, thus leading to cases of the order of thousands of dimensions.

A.1. Multivariate nonlinearities

In [9], the moments were evaluated by using a first-order Taylor approximation of $\mathbf{f}$ for (A.2) and second-order approximation for (A.1). This method was later found to produce unreliable estimates in cases of large source posterior variance, which are common when trying to extract a large number of sources. This caused instability of the algorithm in such cases. This can be corrected by a more accurate approximation based on global linearisation of $\mathbf{f}$. This can be implemented by replacing the derivatives of the activation function appearing in the Taylor scheme by global linearisations evaluated using Gauss–Hermite quadratures [15].

The Gauss–Hermite quadrature is a general method for approximating integrals of the form[5]

$$I(\phi) = \int_{-\infty}^{\infty} \phi(y) N(y; 0, 1)\, dy \approx \sum_{i=1}^{n} w_i \phi(t_i), \tag{A.3}$$

where $\phi : \mathbb{R} \to \mathbb{R}$ is a scalar function. For an approximation using n points, the weights w_i and abscissas t_i can be selected so that the result is exact for all polynomials up to order $2n$. We have used a 3-point approximation as it

[5] The general Gauss–Hermite quadrature is most commonly defined with respect to the weight $\exp(-x^2)$ but the transformation to our representation is straightforward.

provides a good compromise between accuracy and efficiency. The evaluation points can be easily transformed to handle general mean and variance of the input distribution to get general expectation of $\phi(y)$

$$\bar{\phi}(y)_{\mathrm{GH}} := \sum_{i=1}^{n} w_i \phi\big(\bar{y} + t_i \sqrt{\tilde{y}}\big) \approx \big\langle \phi(y) \big\rangle = \int_{-\infty}^{\infty} \phi(y) N(y; \bar{y}, \tilde{y})\, dy. \tag{A.4}$$

Variance of $\phi(y)$ can be evaluated through

$$\tilde{\phi}(y)_{\mathrm{GH}} := \sum_{i=1}^{n} w_i \big[\phi\big(\bar{y} + t_i \sqrt{\tilde{y}}\big) - \bar{\phi}(y)_{\mathrm{GH}}\big]^2 \approx \big\langle \big[\phi(y) - \langle \phi(y) \rangle\big]^2 \big\rangle. \tag{A.5}$$

Both the evaluated mean and variance of $\phi(y)$ depend on the mean and variance of y in a nonlinear manner capable of taking into account the specific form of function ϕ.

The quadrature depends on the mean and variance of $\mathbf{y}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{a}$. These can be evaluated exactly because of the linearity of the mapping as

$$\bar{y}_i(t) = \sum_j \big(\bar{A}_{ij} \bar{s}_j(t)\big) + \bar{a}_i, \tag{A.6}$$

$$\tilde{y}_{i,\mathrm{tot}}(t) = \sum_j \big(\tilde{A}_{ij} \big(\bar{s}_j(t)^2 + \tilde{s}_j(t)\big) + \bar{A}_{ij}^2 \tilde{s}_j(t)\big) + \tilde{a}_i, \tag{A.7}$$

where $\bar{\theta}$ denotes the mean and $\tilde{\theta}$ the variance of θ. Here it is assumed that the posterior approximations $q(\boldsymbol{S}|\boldsymbol{\xi}_{\boldsymbol{S}})$ and $q(\boldsymbol{\theta}_{\mathbf{f}}|\boldsymbol{\xi}_{\boldsymbol{\theta}})$ have diagonal covariances. Full covariances can be used instead without too much difficulty, if necessary. In order to get more accurate approximation, another quadrature is evaluated using the variance of $y_i(t)$ originating mainly from $\boldsymbol{\theta}_{\mathbf{f}}$,

$$\tilde{y}_{i,\mathrm{weight}}(t) = \sum_j \tilde{A}_{ij} \big(\bar{s}_j(t)^2 + \tilde{s}_j(t)\big) + \tilde{a}_i, \tag{A.8}$$

and using the implied $\tilde{\phi}(y_j(t))_{\mathrm{GH,weight}}$ in the evaluation of the effects of these variances. The total variance (A.7) is still used in evaluation of the means and the evaluation of the effects of the variance of $\mathbf{s}(t)$.

The evaluated mean and variance can be used to define an effective linearisation of the hidden nodes by finding a corresponding linear function that would yield the same mean and variance. This yields the effective slope

$$\big\langle \phi'\big(y_i(t)\big) \big\rangle := \sqrt{\frac{\tilde{\phi}(y_i(t))_{\mathrm{GH}}}{\tilde{y}_i(t)}}. \tag{A.9}$$

The effective linearisation is able to take into account the variance of the input, thus following the form of the function more globally when the variance is large.

Thus the final approximation for the mean is

$$\big\langle \bar{f}_i(t) \big\rangle = \sum_j \bar{B}_{ij} \bar{\phi}\big(y_i(t)\big)_{\mathrm{GH}} + \bar{b}_i \tag{A.10}$$

and for the variance

$$\big\langle \big(f_i(t) - \bar{f}_i(t)\big)^2 \big\rangle = \sum_j \big(\tilde{B}_{ij} \big(\bar{\phi}(y)_{\mathrm{GH}}^2 + \tilde{\phi}\big(y_j(t)\big)_{\mathrm{GH}}\big) + \bar{B}_{ij}^2 \tilde{\phi}\big(y_j(t)\big)_{\mathrm{GH,weight}}\big) + \tilde{b}_i + \tilde{\nabla}_{\mathbf{s}} f_i \,\mathrm{diag}(\tilde{\mathbf{s}}) \tilde{\nabla}_{\mathbf{s}} f_i^T, \tag{A.11}$$

where the effective gradient is

$$\tilde{\nabla}_{s_j} f_i(t) = \sum_l \bar{B}_{il} \big\langle \phi'\big(y_l(t)\big) \big\rangle \bar{A}_{lj}. \tag{A.12}$$

A.2. Scalar post-nonlinearities

In case of the MLP networks modelling the scalar post-nonlinearities in PNFA, the approximation is slightly easier as it is possible to directly use the Gauss–Hermite quadrature with respect to the inputs of the MLP. As the variances of the weights are usually small, their effects are represented sufficiently well by using first-order Taylor approximation of the network with respect to them. Thus the mean of the output is approximated as

$$\bar{f}_i(t) = \sum_j w_j f_i\big(\hat{y}_j(t), \bar{\boldsymbol{\theta}}_{\mathbf{f}_i}\big), \tag{A.13}$$

where $\mathbf{y} = \mathbf{A}\mathbf{s}$, w_j are the weights and $\hat{y}_j(t) = \bar{y}(t) + t_j\sqrt{\tilde{y}(t)}$ are the basis points of the Gauss–Hermite quadrature corresponding to the abscissas t_j, and $\bar{\boldsymbol{\theta}}_{\mathbf{f}_i}$ denotes the mean of the weights $\boldsymbol{\theta}_{\mathbf{f}_i}$.

Correspondingly, the variance is approximated by a combined Gauss–Hermite and Taylor approximation

$$\big\langle \big(f_i(t) - \bar{f}_i(t)\big)^2 \big\rangle = \sum_j w_j \Big[\big(f_i\big(\hat{y}_j(t), \bar{\boldsymbol{\theta}}_{\mathbf{f}_i}\big) - \bar{f}_i(t)\big)^2 + \nabla_{\boldsymbol{\theta}_{\mathbf{f}_i}} f_i\big(\hat{y}_j(t), \bar{\boldsymbol{\theta}}_{\mathbf{f}_i}\big)\, \mathrm{diag}(\tilde{\boldsymbol{\theta}}_{\mathbf{f}_i}) \nabla_{\boldsymbol{\theta}_{\mathbf{f}_i}} f_i\big(\hat{y}_j(t), \bar{\boldsymbol{\theta}}_{\mathbf{f}_i}\big)^T \Big]. \tag{A.14}$$

References

[1] A. Hyvärinen, J. Karhunen, E. Oja, Independent Component Analysis, Wiley, New York, 2001.
[2] H. Attias, Independent factor analysis, Neural Comput. 11 (4) (1999) 803–851.
[3] H. Lappalainen, Ensemble learning for independent component analysis, in: Proc. Int. Workshop on Independent Component Analysis and Signal Separation (ICA 1999), Aussois, France, 1999, pp. 7–12.
[4] J. Miskin, Ensemble learning for independent component analysis, Ph.D. thesis, University of Cambridge, UK, 2000.
[5] P. Højen-Sørensen, O. Winther, L.K. Hansen, Mean-field approaches to independent component analysis, Neural Comput. 14 (4) (2002) 889–918.
[6] R.A. Choudrey, S.J. Roberts, Variational mixture of Bayesian independent component analyzers, Neural Comput. 15 (1) (2003) 213–252.
[7] K. Chan, T.-W. Lee, T.J. Sejnowski, Variational Bayesian learning of ICA with missing data, Neural Comput. 15 (8) (2003) 1991–2011.
[8] H. Attias, Independent factor analysis with temporally structured sources, in: S. Solla, T. Leen, K.-R. Müller (Eds.), Advances in Neural Information Processing Systems, vol. 12, MIT Press, Cambridge, MA, 2000, pp. 386–392.
[9] H. Lappalainen, A. Honkela, Bayesian nonlinear independent component analysis by multi-layer perceptrons, in: M. Girolami (Ed.), Advances in Independent Component Analysis, Springer-Verlag, Berlin, 2000, pp. 93–121.
[10] H. Valpola, T. Östman, J. Karhunen, Nonlinear independent factor analysis by hierarchical models, in: Proc. 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA 2003), Nara, Japan, 2003, pp. 257–262.
[11] A. Ilin, A. Honkela, Postnonlinear independent component analysis by variational Bayesian learning, in: C.G. Puntonet, A. Prieto (Eds.), Proc. Fifth Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA 2004), in: Lecture Notes in Computer Science, vol. 3195, Springer-Verlag, Berlin, 2004, pp. 766–773.
[12] T. Raiko, H. Valpola, Missing values in nonlinear factor analysis, in: Proc. 8th Int. Conf. on Neural Information Processing (ICONIP 2001), Shanghai, 2001, pp. 822–827.
[13] T. Raiko, H. Valpola, T. Östman, J. Karhunen, Missing values in hierarchical nonlinear factor analysis, in: Proc. Int. Conf. on Artificial Neural Networks and Neural Information Processing (ICANN/ICONIP 2003), Istanbul, Turkey, 2003, pp. 185–189.
[14] A. Honkela, S. Harmeling, L. Lundqvist, H. Valpola, Using kernel PCA for initialisation of variational Bayesian nonlinear blind source separation method, in: C.G. Puntonet, A. Prieto (Eds.), Proc. 5th Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA 2004), in: Lecture Notes in Computer Science, vol. 3195, Springer-Verlag, Berlin, Granada, Spain, 2004, pp. 790–797.
[15] A. Honkela, H. Valpola, Unsupervised variational Bayesian learning of nonlinear models, in: L. Saul, Y. Weiss, L. Bottou (Eds.), Advances in Neural Information Processing Systems, vol. 17, MIT Press, Cambridge, MA, 2005, pp. 593–600.
[16] C. Jutten, J. Karhunen, Advances in blind source separation (BSS) and independent component analysis (ICA) for nonlinear mixtures, Int. J. Neural Syst. 14 (5) (2004) 267–292.
[17] L.B. Almeida, Nonlinear Source Separation, Synthesis Lectures on Signal Processing, Morgan Kaufmann, San Mateo, CA, 2006.
[18] A. Hyvärinen, P. Pajunen, Nonlinear independent component analysis: Existence and uniqueness results, Neural Networks 12 (3) (1999) 429–439.
[19] A. Taleb, C. Jutten, Source separation in post-nonlinear mixtures, IEEE Trans. Signal Process. 47 (10) (1999) 2807–2820.
[20] S. Haykin, Neural Networks—A Comprehensive Foundation, second ed., Prentice–Hall, Englewood Cliffs, NJ, 1999.
[21] S. Hochreiter, J. Schmidhuber, LOCOCODE performs nonlinear ICA without knowing the number of sources, in: Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA 1999), Aussois, France, 1999, pp. 149–154.
[22] L.B. Almeida, MISEP—linear and nonlinear ICA based on mutual information, J. Machine Learn. Res. 4 (2003) 1297–1318.
[23] L.B. Almeida, Separating a real-life nonlinear image mixture, J. Machine Learn. Res. 6 (2005) 1199–1229.

[24] M.S.C. Almeida, H. Valpola, J. Särelä, Separation of nonlinear image mixtures by denoising source separation, in: Proc. 6th Int. Conf. on Independent Component Analysis and Blind Source Separation (ICA 2006), Charleston, South Carolina, USA, 2006, pp. 8–15.
[25] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput. 10 (5) (1998) 1299–1319.
[26] N. Lawrence, Probabilistic non-linear principal component analysis with Gaussian process latent variable models, J. Machine Learn. Res. 6 (2005) 1783–1816.
[27] A. Ziehe, M. Kawanabe, S. Harmeling, K.-R. Müller, Blind separation of post-nonlinear mixtures using linearizing transformations and temporal decorrelation, J. Machine Learn. Res. 4 (2003) 1319–1338.
[28] H. Valpola, T. Raiko, J. Karhunen, Building blocks for hierarchical latent variable models, in: Proc. 3rd Int. Conf. on Independent Component Analysis and Signal Separation (ICA 2001), San Diego, USA, 2001, pp. 710–715.
[29] M. Harva, T. Raiko, A. Honkela, H. Valpola, J. Karhunen, Bayes Blocks: An implementation of the variational Bayesian building blocks framework, in: Proc. 21st Conf. on Uncertainty in Artificial Intelligence (UAI 2005), Edinburgh, Scotland, 2005, pp. 259–266.
[30] T. Raiko, H. Valpola, M. Harva, J. Karhunen, Building blocks for variational Bayesian learning of latent variable models, J. Machine Learn. Res. 8 (2007) 155–201.
[31] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Networks 2 (5) (1989) 359–366.
[32] H. Valpola, Nonlinear independent component analysis using ensemble learning: Theory, in: Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA 2000), Helsinki, Finland, 2000, pp. 251–256.
[33] M. Stinchcombe, H. White, Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions, in: Proc. Int. Joint Conf. on Neural Networks (IJCNN 1989), 1989, pp. I–613–617.
[34] A. Ilin, H. Valpola, On the effect of the form of the posterior approximation in variational learning of ICA models, Neural Process. Lett. 22 (2) (2005) 183–204.
[35] S. Ikeda, ICA on noisy data: A factor analysis approach, in: M. Girolami (Ed.), Advances in Independent Component Analysis, Springer-Verlag, Berlin, 2000, pp. 201–215.
[36] T.S. Jaakkola, Tutorial on variational approximation methods, in: M. Opper, D. Saad (Eds.), Advanced Mean Field Methods: Theory and Practice, MIT Press, Cambridge, MA, 2001, pp. 129–159.
[37] J. Winn, C.M. Bishop, Variational message passing, J. Machine Learn. Res. 6 (2005) 661–694.
[38] H. Lappalainen, J. Miskin, Ensemble learning, in: M. Girolami (Ed.), Advances in Independent Component Analysis, Springer-Verlag, Berlin, 2000, pp. 75–92.
[39] A. Honkela, H. Valpola, X. Giannakopoulos, Nonlinear factor analysis Matlab package, 2004, http://www.cis.hut.fi/projects/bayes/software/.
[40] K. Fukumizu, S. Amari, Local minima and plateaus in hierarchical structures of multilayer perceptrons, Neural Networks 13 (3) (2000) 317–327.
[41] H. Valpola, A. Honkela, M. Harva, A. Ilin, T. Raiko, T. Östman, Bayes Blocks software library, 2003, http://www.cis.hut.fi/projects/bayes/software/.
[42] A. Honkela, H. Valpola, J. Karhunen, Accelerating cyclic update algorithms for parameter estimation by pattern searches, Neural Process. Lett. 17 (2) (2003) 191–203.
[43] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, San Mateo, CA, 1988.
[44] T. Raiko, Partially observed values, in: Proc. Int. Joint Conf. on Neural Networks (IJCNN 2004), Budapest, Hungary, 2004, pp. 2825–2830.
[45] A. Honkela, H. Valpola, Variational learning and bits-back coding: an information-theoretic view to Bayesian learning, IEEE Trans. Neural Networks 15 (4) (2004) 800–810.
[46] A. Honkela, T. Östman, R. Vigário, Empirical evidence of the linear nature of magnetoencephalograms, in: Proc. 13th European Symposium on Artificial Neural Networks (ESANN 2005), Bruges, Belgium, 2005, pp. 285–290.

Antti Honkela received his M.Sc. (Tech.) degree in mathematics in 2001 and the D.Sc. (Tech.) degree in computer science in 2005 from Helsinki University of Technology, Finland. He is currently a postdoctoral research fellow at the Adaptive Informatics Research Centre, Helsinki University of Technology. His research interests include Bayesian machine learning and approximate inference, especially unsupervised learning of nonlinear models.

Harri Valpola received his M.Sc. degree in technical physics in 1996 and the D.Sc. degree in computer science in 2000 from the Helsinki University or Technology, Finland. His Ph.D. research was on developing approximate Bayesian inference techniques suited to nonlinear factor analysis and related models.

He is currently an academy research fellow heading a computational neuroscience group at the Laboratory of Computational Engineering, Helsinki University of Technology. He has previously been a researcher at the Laboratory of Computer and Information Science, Helsinki University of Technology, and a post-doctoral researcher at the Artificial Intelligence Laboratory, University of Zurich, Switzerland.

Alexander N. Ilin was born in Saint-Petersburg, Russia, in 1977. He received his M.Sc. (honors) degree in computer science and engineering in 2000 from Saint-Petersburg State Polytechnical University, Russia, and his Dr.Sc. (honors) degree in technology in 2006 from Helsinki University of Technology, Finland. He is a postdoctoral researcher in the Adaptive Informatics Research Centre in Helsinki University of Technology, Finland. His research interests are in statistical machine learning, intelligent signal processing with applications to analysis of large-scale datasets.

Juha Karhunen received his D.Sc. (Tech.) degree from Helsinki University of Technology in 1984. Since 1976, he has been in the Laboratory of Computer and Information Science at Helsinki University of Technology, Espoo, Finland, where he became a Professor in computer science in 1999 (specialization area: neural networks and signal processing). He belongs to the Adaptive Informatics Research Centre in the laboratory, which has been selected by the Academy of Finland as one of the centres of excellence in research in Finland.

His current research interests include unsupervised variational Bayesian learning, independent component analysis, blind source separation, and their applications. Professor Karhunen has published about 100 conference and journal papers on these topics, and given invited talks in international conferences. He is a co-author of the popular textbook and monograph [A. Hyvärinen, J. Karhunen, E. Oja, Independent Component Analysis, Wiley, 2000]. He is a senior member of IEEE, and has been a member of the editorial board in the journals Neurocomputing and Neural Processing Letters.