Nonlinear Static and Dynamic Blind Source Separation Using Ensemble Learning

Harri Valpola, Antti Honkela, and Juha Karhunen

Helsinki University of Technology, Neural Networks Research Centre

P.O. Box 5400, FIN-02015 HUT, Espoo, Finland.

Email: {Harri.Valpola, Antti.Honkela, Juha.Karhunen}@hut.fi URL: http://www.cis.hut.fi/

Abstract

Blind separation of sources from their nonlinear mixtures is generally a very difficult problem. This is because both the nonlinear mapping and the underlying sources are unknown, and must be learned in a completely unsupervised manner from the data. We use multilayer perceptrons as nonlinear generative models for the data, and apply Bayesian ensemble learning for finding the sources. In this paper, we first consider a static nonlinear mixture model, with a successful application to real-world speech data. Then we briefly discuss extraction of sources from nonlinear dynamic processes. In a difficult test problem with chaotic data, our approach clearly outperforms currently available nonlinear prediction techniques. The proposed methods are computationally demanding especially in the dynamic case, but they allow the use of higher dimensional nonlinear latent variable models than other existing approaches.

1 Introduction

A flexible model family provides infinitely many possible explanations of different complexity for the observed data. In the Bayesian learning formalism all the possible explanations are taken into account and weighted according to their posterior probabilities. This approach optimally solves the tradeoff between under- and overfitting. All the relevant information needed in choosing an appropriate model is contained in the posterior probability density functions (pdf's) of different model structures. In practice, exact treatment of the posterior pdf's of the models is not possible. Therefore, some suitable approximation method must be used. Ensemble learning [2, 9], which is a special case of variational learning, is a recently developed method for parametric approximation of posterior pdf's where the search takes into account the probability *mass* of the models. Therefore, it is fairly insensitive to overfitting. The basic idea in ensemble learning is to minimize the misfit between the posterior pdf and its parametric approximation.

In this work, we assume that the observed data vectors $\mathbf{x}(t)$ at time (or index value) t have been generated by respective unknown source vectors $\mathbf{s}(t)$ through an unknown nonlinear mapping $\mathbf{f}(\cdot)$. Mathematically, the data model is

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

where $\mathbf{n}(t)$ denotes additive zero mean Gaussian noise corrupting the observations or representing the modeling errors. The components $s_i(t)$ of the source vector $\mathbf{s}(t)$ are assumed to be Gaussian, too. We call them the source signals or sources. In other contexts, different names such as latent variables, factors, or hidden causes have been used for them. 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.

Denote by $X = \{\mathbf{x}(t)|t\}$ the set of available data (mixture) vectors, by $S = \{\mathbf{s}(t)|t\}$ the respective source vectors, and by $\boldsymbol{\theta}$ all the unknown parameters of the data model. Furthermore, $P(S, \boldsymbol{\theta}|X)$ denotes the exact posterior pdf and $Q(S, \boldsymbol{\theta}|X)$ its parametric approximation. The cost function used in ensemble learning is the Kullback-Leibler (KL) divergence C_{KL} between P and Q, defined by

$$C_{\mathrm{KL}} = \int_S \int_{\boldsymbol{\theta}} Q(S, \boldsymbol{\theta}|X) \log \frac{Q(S, \boldsymbol{\theta}|X)}{P(S, \boldsymbol{\theta}|X)} d\boldsymbol{\theta} dS \tag{2}$$

This information-theoretic criterion measures the difference in the probability mass between the densities P and Q. Its minimum value 0 is achieved when the two densities are the same.

In this work, we apply unsupervised ensemble learning to nonlinear blind source separation (BSS) of static and dynamic mixtures. Many well performing approaches already exist for the basic linear BSS and for the closely related independent component analysis (ICA) [6]. During the last years, several authors have tried to generalize linear ICA and BSS for nonlinear data models. Nonlinear BSS is much more difficult compared with the linear case. In particular, the problem is highly non-unique without suitable regularizing constraints. See [6, 7] for more information and references on nonlinear BSS.

We use the familiar multilayer perceptron (MLP) network [5, 11] for modeling the nonlinear mapping (1) from the unknown sources to the observations. MLP has the universal approximation property for smooth continuous mappings, and suits well to modeling both strong and mild nonlinearities. However, the learning procedure is clearly different from standard back-propagation, and is based on unsupervised ensemble learning. The Bayesian philosophy and the information-theoretic KL cost measure (2) provide the necessary regularization for nonlinear BSS. For each source signal and parameter, its posterior pdf is estimated instead of some point estimate. If necessary, a specific estimate is given by the mean of the posterior pdf of the desired quantity, and the respective variance provides a rough estimate of its reliability.

Recently, ensemble learning and other closely related Bayesian methods have been applied to various linear BSS and ICA problems in [1, 10]. Our work differs from these approaches in that a more general nonlinear data model is assumed, and it is further extended to non-stationary dynamic state-space models. Due to the space limitations, we must content ourselves with a coarse overall description in this paper. More information on the details and refinements of the learning procedure, on potentially appearing problems, and experimental results can be found in [8, 12] for the static case, and in [12, 13] for the dynamic case.

2 Ensemble learning for nonlinear BSS

2.1 Model structure

The nonlinear mapping $\mathbf{f}$ in (1) is modeled by the MLP network

$$\mathbf{f}(\mathbf{s}(t)) = \mathbf{B}\tanh[\mathbf{A}\mathbf{s}(t) + \mathbf{a}] + \mathbf{b} \tag{3}$$

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 nonlinearity tanh is applied separately to each component of its argument vector. We have used a mixture of Gaussians model for the sources in [6, 8]. Then it is possible to approximate sufficiently well any non-Gaussian source distribution. However, this refinement provides in practice a small improvement only in the results while complicating the learning process. Hence the simpler Gaussian model is adopted here, followed by linear ICA in the last stage to find the independent sources.

The parameters of the network are: (1) the weight matrices $\mathbf{A}$ and $\mathbf{B}$ and the bias vectors $\mathbf{a}$ and $\mathbf{b}$; (2) the parameters of the distributions of the noise, source signals and column vectors of the weight matrices; (3) the hyperparameters used for defining the distributions of the biases and the parameters in the group (2). All the parameterized distributions are assumed to be Gaussian. This does not limit severely the generality of the approach, but makes computational implementation simpler and much more efficient. The hierarchical description of the distributions of the parameters of the model used here is a standard procedure in probabilistic Bayesian modeling. Its strength lies in that knowledge about equivalent status of different parameters can be easily incorporated. For example all the variances of the noise components have a similar status in the model. This is reflected by the fact that their distributions are assumed to be governed by common hyperparameters.

2.2 Overall learning procedure

Usually MLP networks learn the nonlinear input-output mapping in a supervised manner using known input-output pairs, for which the mean-square mapping error is minimized using the back-propagation algorithm [5, 11]. In our case, the inputs are the *unknown* source signals $\mathbf{s}(t)$, and only the outputs of the MLP network, namely the observed data vectors $\mathbf{x}(t)$, are known. Hence, unsupervised learning must be applied. The practical learning procedure is as follows.

First, linear PCA (principal component analysis) is applied to find sensible initial values for the posterior means of the sources. Even though PCA is a linear method, it yields clearly better initial values than a random choice. The posterior variances of the sources are initialized to small values. Good initial values are important for the method because the network can effectively prune away unused parts. Initially the weights of the network have random values, and the network has quite a bad representation for the data. In this initial stage, one must prevent the method from pruning prematurely away sources which contribute only a little to the representation of the data. This is achieved by fixing the sources at the values given by linear PCA for the first 50 sweeps through the entire data set. This allows the network to find a meaningful mapping from sources to the observations, thereby justifying the use of the sources for the representation. For the same reason, the parameters controlling the distributions of the sources, weights, noise and the hyperparameters are not adapted during the first 100 sweeps. They are adapted only after the network has found sensible values for the variables whose distributions these parameters control.

After this, learning is continued by using the nonlinear model where the sources have Gaussian distributions. This is called nonlinear factor analysis (NFA) model in [8]. Finally, the found sources were rotated using an efficient linear ICA algorithm called FastICA [6]. As mentioned earlier, the learning could have still continued using now the mixture-of-Gaussians model for the sources. In [8], that representation is called nonlinear independent factor analysis (NIFA).

2.3 Computation of the KL cost function

Consider then in more detail the Kullback-Leibler cost function C_{KL} defined in Eq. (2). For approximating and then minimizing it, we need two things: the exact formulation of the posterior density $P(S, \boldsymbol{\theta}|X)$ and its parametric approximation $Q(S, \boldsymbol{\theta}|X)$.

According to the Bayes' rule, the posterior pdf of the unknown variables S and $\boldsymbol{\theta}$ is

$$P(S, \boldsymbol{\theta}|X) = \frac{P(X|S, \boldsymbol{\theta})P(S|\boldsymbol{\theta})P(\boldsymbol{\theta})}{P(X)} \tag{4}$$

The term $P(X|S, \boldsymbol{\theta})$ is obtained from equations (1) and (3). Clearly, the distribution of $\mathbf{x}(t)$ is the same as the distribution of the noise vector $\mathbf{n}(t)$, but with the mean vector $\mathbf{f}(\mathbf{s}(t))$. Denoting the variance of the ith component of $\mathbf{n}(t)$ by σ_i^2, the distribution $P(x_i(t)|\mathbf{s}(t), \boldsymbol{\theta})$ is thus Gaussian with mean $\mathbf{b}_i^T \mathbf{f}(\mathbf{As} + \mathbf{a}) + b_i$ and variance σ_i^2. Here $x_i(t)$ is the ith component of $\mathbf{x}(t)$, $\mathbf{b}_i^T$ denotes the ith row vector of $\mathbf{B}$, and b_i is the ith component of the bias vector $\mathbf{b}$. As usually, the noise components $n_i(t)$ are assumed to be independent, and therefore $P(X|S, \boldsymbol{\theta}) = \prod_{t,i} P(x_i(t)|\mathbf{s}(t), \boldsymbol{\theta})$.

The terms $P(S|\boldsymbol{\theta})$ and $P(\boldsymbol{\theta})$ in (4) are also products of simple Gaussian distributions, and they are obtained directly from the model structure [8, 13]. The term $P(X)$ does not depend on the model parameters and can be neglected.

The approximation $Q(S, \boldsymbol{\theta}|X)$ must be simple for mathematical tractability and computational efficiency. First, we assume that the source signals S are independent of the other parameters $\boldsymbol{\theta}$, so that $Q(S, \boldsymbol{\theta}|X)$ decouples into $Q(S, \boldsymbol{\theta}|X) = Q(S|X)Q(\boldsymbol{\theta}|X)$. For the parameters $\boldsymbol{\theta}$, a Gaussian density with a diagonal covariance matrix is used. This implies that the approximation is a product of independent distributions: $Q(\boldsymbol{\theta}|X) = \prod_i Q_i(\theta_i|X)$. The parameters of each Gaussian component density $Q_i(\theta_i|X)$ are its mean $\bar{\theta}_i$ and variance $\tilde{\theta}_i$. The pdf $Q(S|X)$ is computed in the same way.

Both the posterior density $P(S, \boldsymbol{\theta}|X)$ and its approximation $Q(S, \boldsymbol{\theta}|X)$ are products of simple Gaussian terms, which simplifies the cost function (2) considerably: it splits into expectations of many simple terms. The terms of the form $E_Q\{\log Q_i(\theta_i|X)\}$ are negative entropies of Gaussians, having the exact values $-(1 + \log 2\pi\tilde{\theta}_i)/2$. The most difficult terms are of the form $-E_Q\{\log P(x_i(t)|\mathbf{s}(t), \boldsymbol{\theta})\}$. They are approximated by applying second order Taylor's series expansions of the nonlinear activation functions [8, 13]. The remaining terms are expectations of simple Gaussian terms which can be computed as explained in [9, 13].

In ensemble learning each variable has its own associated posterior pdf. In our case, the variables consist of all the source signals $s_i(t)$ for different t and i as well as of the other parameters θ_j of the network. These posterior pdf's are in turn characterized by their means $\bar{s}_i(t)$, $\bar{\theta}_i$ and variances $\tilde{s}_i(t)$, $\tilde{\theta}_i$. The cost function C_{KL} is a function of all these means and variances. This is because instead of finding a point estimate, the joint posterior pdf of the sources and parameters is estimated.

2.4 Updating the parameters and sources

After the cost function (2) has been calculated by propagating the posterior means and variances of the sources and other parameters through the MLP network, the values of these quantities are updated. These forward and backward stages are then repeated until convergence. In this respect, the learning process resembles batch type back-propagation [5].

Let us denote the two parts of the cost function (2) arising from the denominator and numerator of the logarithm respectively by $C_p = -E_Q\{\log P\}$ and $C_q = E_Q\{\log Q\}$. The variances $\tilde{\theta}_i$ are obtained by differentiating (2) with respect to $\tilde{\theta}_i$ [8]:

$$\frac{\partial C}{\partial \tilde{\theta}_i} = \frac{\partial C_p}{\partial \tilde{\theta}_i} + \frac{\partial C_q}{\partial \tilde{\theta}_i} = \frac{\partial C_p}{\partial \tilde{\theta}_i} - \frac{1}{2\tilde{\theta}_i} \tag{5}$$

Equating this to zero yields a fixed-point iteration for updating the variances:

$$\tilde{\theta}_i = \left[2 \frac{\partial C_p}{\partial \tilde{\theta}_i} \right]^{-1} \tag{6}$$

The means $\bar{\theta}_i$ can be estimated from the approximate Newton iteration [8]

$$\bar{\theta}_i \leftarrow \bar{\theta}_i - \frac{\partial C_p}{\partial \bar{\theta}_i} \left[\frac{\partial^2 C}{\partial \bar{\theta}_i^{\,2}} \right]^{-1} \approx \bar{\theta}_i - \frac{\partial C_p}{\partial \bar{\theta}_i} \tilde{\theta}_i \qquad (7)$$

The posterior means $\bar{s}_i(t)$ and variances $\tilde{s}_i(t)$ of the sources are updated quite similarly.

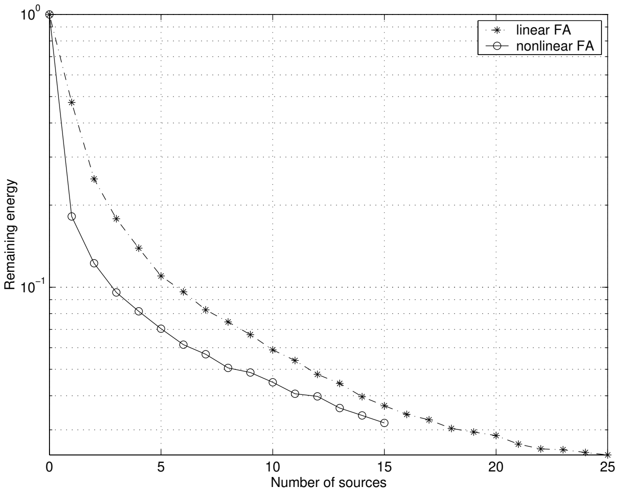

Figure 1: The remaining energy of the speech data as a function of the number of extracted components using linear and nonlinear factor analysis.

3 Experimental results for speech data

In all our experiments, the total number of sweeps was 7500. One sweep means going through all the observations, evaluating the cost function, and updating the variables once. A NFA representation for the sources was estimated first, and the final results were then obtained by applying linear ICA to the found sources $\mathbf{s}(t)$.

The data set consisted of spectrograms of 24 individual words of Finnish speech, spoken by 20 different speakers. The spectrum was 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 was applied to the data. The results are shown in Fig. 1. The figure shows also the results with nonlinear factor analysis. All the results were obtained by using an MLP network with 30 hidden neurons. The data are clearly nonlinear, because nonlinear factor analysis is able to explain them equally well with fewer components than linear factor analysis. The difference is clear when the number of components is small.

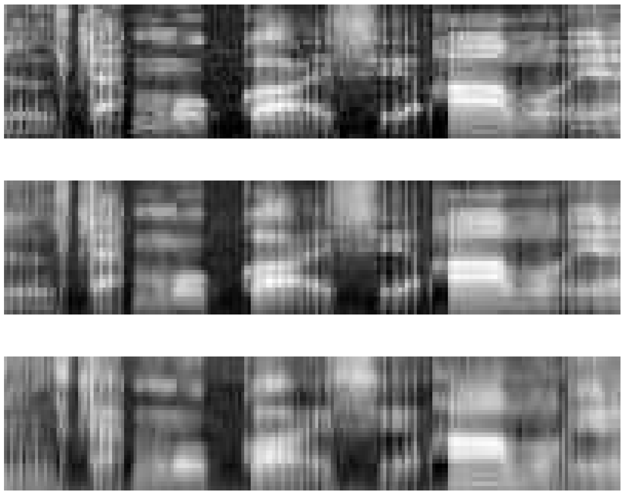

Figure 2: A short fragment of the data used in the speech modeling experiment. The subfigures show the original data (top), and the reconstructions from 8 nonlinear components (middle) and from 8 linear components (bottom).

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 can 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 improved by taking into account also temporal information in the data, leading probably to even better results.

The NFA based method allows nonlinear BSS and ICA for larger-scale problems than previous methods [6], for which the computational load typically grows exponentially with the dimension. It can be extended in various directions. In [6, 8], we have applied it to industrial pulp process data and to artificially generated data with good results. There we improved the quality of estimated sources somewhat further by using a mixture-of-Gaussians model for them.

4 Extension to a dynamic nonlinear model

4.1 The model and method

Recently, we have extended the NFA method described above for dynamic processes. This extension is called here nonlinear dynamic factor analysis (NDFA). For the data vectors $\mathbf{x}(t)$, the same generative model (1) and MLP network structure (3) as before are still employed. The new part in NDFA consists of a state-space model for the dynamics of the source vector $\mathbf{s}(t)$:

$$\mathbf{s}(t) = \mathbf{g}(\mathbf{s}(t-1)) + \mathbf{m}(t) \tag{8}$$

The components of $\mathbf{s}(t)$ are generated through another nonlinear tranformation $\mathbf{g}$ from the previous source vector $\mathbf{s}(t-1)$. The Gaussian error term $\mathbf{m}(t)$ has zero mean and it is independent of $\mathbf{s}(t)$.

Neural networks approaches for this type of dynamic models are discussed in [11, 5]. Bayesian techniques for learning the mappings (1) and (8) have been introduced recently in [3, 4].

The dynamic mapping $\mathbf{g}$ in (8) is modeled using another MLP network as follows:

$$\mathbf{g}(\mathbf{s}(t-1)) = \mathbf{s}(t-1) + \mathbf{D}\tanh[\mathbf{C}\mathbf{s}(t-1) + \mathbf{c}] + \mathbf{d} \tag{9}$$

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 factors $\mathbf{s}(t)$ do usually not change much from their previous values $\mathbf{s}(t-1)$ during a single time step, we have used the MLP network for modeling only the change in factors in (9).

The NDFA method can now be developed in a similar manner as the NFA method before. This is described in detail in [12, 13]. A special problem in NDFA is that the components of subsequent source vectors $\mathbf{s}(t)$ and $\mathbf{s}(t-1)$ are no longer completely independent due to the model (8). This is handled by using a linear dependency model for their same components $s_i(t)$ and $s_i(t-1)$, 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 Predicting a Lorenz process

The dynamic process used to test the NDFA method was a combination of 3 independent dynamic processes. The first one was a harmonic oscillator which has a 2-dimensional state representation with linear dynamics. The 2 other processes were Lorenz processes which have

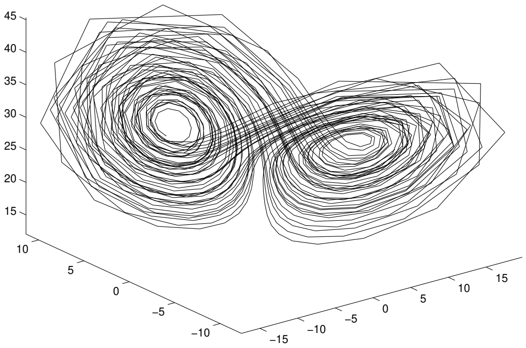

Figure 3: The original sampled Lorenz process.

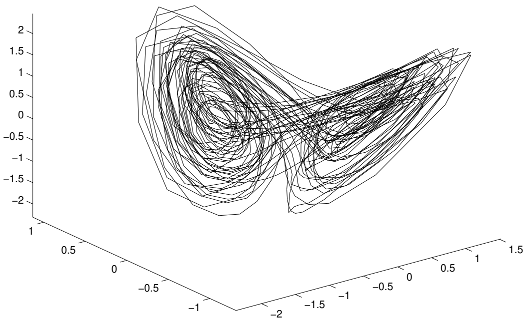

Figure 4: The corresponding 3 components of the estimated process.

3-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 3 processes was hidden. This left 5 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 4 shows the NDFA estimate of the Lorenz process of Fig. 3 after $600,000$ iterations. The quality of the estimated process is good, taking into account the the very difficult problem and the indeterminacies in this type of nonlinear dynamic estimation. We tested the quality of the NDFA estimates also by studying its prediction ability for new samples. It should be noted that since the Lorenz processes are chaotic, the best that any method can do is to capture its general long-term behavior - exact numerical prediction is impossible.

We compared NDFA in predicting future observations with the nonlinear autoregressive (NAR) model

$$\mathbf{x}(t) = \mathbf{h}(\mathbf{x}(t-1), \ldots, \mathbf{x}(t-d)) + \mathbf{n}(t) \qquad (10)$$

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 5 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 $t = 1022$, while the NAR method is quite inaccurate already after $t \geq 1003$. Here the prediction started at time $t = 1000$.

According to our experiments, the NDFA method is able to learn up to about 15-dimensional latent spaces, or to separate blindly roughly 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. Finding means to speed up learning is therefore highly desirable.

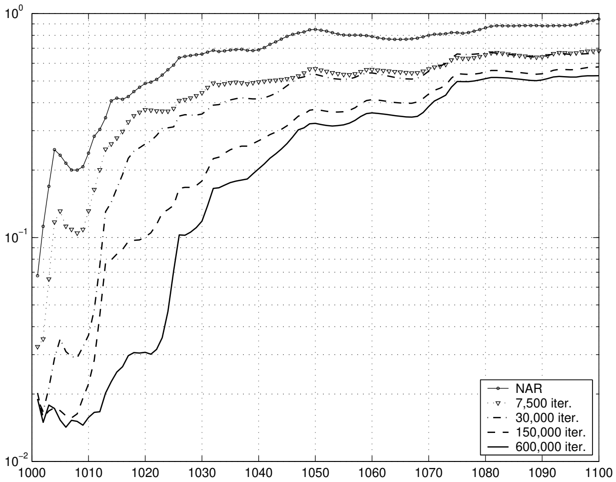

Figure 5: The average cumulative squared prediction error for the nonlinear autoregressive (NAR) model (solid with dots) and for our dynamic algorithm with different iteration numbers.

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] D. Barber and C. Bishop. Ensemble learning in Bayesian neural networks. In M. Jordan, M. Kearns, and S. Solla, editors, *Neural Networks and Machine Learning*, pages 215–237. Springer, Berlin, 1998.

[3] 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.

[4] 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.

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

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

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

[8] 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, 2000.

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

[10] 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.

[11] J. Principe, N. Euliano, and C. Lefebvre. *Neural and Adaptive Systems - Fundamentals Through Simulations.* Wiley, 2000.

[12] H. Valpola. *Bayesian Ensemble Learning for Nonlinear Factor Analysis.* PhD thesis, Helsinki University of Technology, Espoo, Finland, 2000. Published in Acta Polytechnica Scandinavica, Mathematics and Computing Series No. 108.

[13] H. Valpola and J. Karhunen. An unsupervised ensemble learning method for nonlinear dynamic state-space models. 2001. Submitted to a journal.