Nonlinear Dynamical Factor Analysis for State Change Detection

Alexander Ilin, Harri Valpola, and Erkki Oja, *Fellow, IEEE*

***Abstract*—Changes in a dynamical process are often detected by monitoring selected indicators directly obtained from the process observations, such as the mean values or variances. Standard change detection algorithms such as the Shewhart control charts or the cumulative sum (CUSUM) algorithm are often based on such first- and second-order statistics. Much better results can be obtained if the dynamical process is properly modeled, for example by a nonlinear state-space model, and then the accuracy of the model is monitored over time. The success of the latter approach depends largely on the quality of the model. In practical applications like industrial processes, the state variables, dynamics, and observation mapping are rarely known accurately. Learning from data must be used; however, methods for the simultaneous estimation of the state and the unknown nonlinear mappings are very limited. We use a novel method of learning a nonlinear state-space model, the nonlinear dynamical factor analysis (NDFA) algorithm. It takes a set of multivariate observations over time and fits blindly a generative dynamical latent variable model, resembling nonlinear independent component analysis. We compare the performance of the model in process change detection to various traditional methods. It is shown that NDFA outperforms the classical methods by a wide margin in a variety of cases where the underlying process dynamics changes.**

***Index Terms*—Change detection, independent component analysis, nonlinear state-space model, variational Bayesian learning.**

I. Introduction

PROCESS change detection is an important problem in many fields of engineering. An abrupt change in the process usually indicates a fault, and the goal of change detection is to pinpoint the exact occurrence of the fault and to give an alarm. It would also be very desirable to be able to analyze exactly where in the process the original reason for the fault is. This may be quite difficult because a fault in some underlying subsystems or parameters may manifest itself in complicated ways in the observables, or sometimes be hardly observable at all.

Many current methods monitor some direct indicators of the process observables and respond to changes in the indicators, such as the mean or variance of a process measurement [7]. For example, the Shewhart control charts or the geometric moving average charts smooth a simple statistic over a sliding window and compare it to a given threshold [4]. Another example is the cumulative sum (CUSUM) method [4], [15] which is related to the traditional sequential probability ratio test. As applied to monitoring the process mean or variance, all these algorithms are based on the first- and second-order statistics of the observations. Such indicator-based approaches do not take all the relevant information about the process into account which usually means a delayed response to a change or neglect of the change in the worst case.

A better solution to the change detection problem is to model the process and then use the goodness-of-fit of the new observations to the previously established model as the change indicator. A popular modeling tool is the autoregressive (AR) model in which traditional supervised learning algorithms can be used to learn the required linear or nonlinear mapping from past observations $\mathbf{x}(t-1),\ldots$ to the current observation vector $\mathbf{x}(t)$:

$$\mathbf{x}(t) = \mathbf{f}[\mathbf{x}(t-1),\ldots,\mathbf{x}(t-d)] + \mathbf{n}(t). \tag{1}$$

For simplicity we omit here and in the following the external inputs to the process, using an AR model instead of ARMAX. The inputs can easily be added and will not change the picture [15]. For learning nonlinear mappings $\mathbf{f}$, neural networks have been widely used, for example, feedforward multilayer perceptron (MLP) networks, radial basis function (RBF) networks, and recurrent MLP networks [16], [10].

A much more powerful process model can be built using physical knowledge, however. Then the process is usually described by a state-space model. Especially, the nonlinear state-space model (NSSM) is a very natural, general, and flexible model for multivariate time series data. The observation vectors $\mathbf{x}(t)$ are assumed to be generated from hidden multivariate sources, or states $\mathbf{s}(t)$ of the dynamical system, through a nonlinear mapping $\mathbf{f}$ according to (2)

$$\mathbf{x}(t) = \mathbf{f}(\mathbf{s}(t)) + \mathbf{n}(t) \tag{2}$$

$$\mathbf{s}(t) = \mathbf{g}(\mathbf{s}(t-1)) + \mathbf{m}(t). \tag{3}$$

The states follow the nonlinear dynamics $\mathbf{g}$ defined by (3). The terms $\mathbf{n}(t)$ and $\mathbf{m}(t)$ account for modeling errors and noise. The model assumes an underlying state which has dynamics and is reflected in the observations through an observation mapping. Note that one step delay suffices in (3), since the state variable $\mathbf{s}(t)$ is a vector that can include velocities, accelerations and other quantities which summarize the physical state of the system. For continuous time systems, differential equations replace the difference equation of the dynamics used here.

The problem is considerably simplified if we can assume that the system is deterministic, with unknown disturbances. Then the noise terms are not included, and the faults are modeled as extra terms in the dynamic (3). The general methods are based

Manuscript received April 11, 2003; revised December 16, 2003. This work was supported by the European Commission Project BLISS, and the Finnish Center of Excellence Programme (2000–2005) under the Project New Information Processing Principles.

The authors are with the Neural Networks Research Centre, Helsinki University of Technology, 02015 HUT, Finland (e-mail: erkki.oja@hut.fi).

Digital Object Identifier 10.1109/TNN.2004.826129

on decoupling the faults from the unknown disturbances and monitoring the residuals of a fully deterministic dynamic system [6]. The residuals should be close to zero in a fault-free state and depart from zero when a change in the process occurs. A survey on the fault detection methods for nonlinear deterministic systems can be found in [12].

Usually, however, we have to assume that modeling errors or external noise occur, and then the noise terms are included in the NSSM. In this case, the change detection problem has been well studied for linear systems, in which the functions $\mathbf{f}$ and $\mathbf{g}$ are assumed linear. The most common technique is testing statistical properties of the innovations generated by a Kalman filter [4], [15]. The case of extra unknown disturbances has also been studied for linear systems [6].

The results on detecting changes in nonlinear stochastic systems have been quite limited. The main tool for this problem is linearization like in the extended Kalman filter, and change detection methods for linear systems. Nevertheless some nonlinear techniques for static and dynamic stochastic systems have been presented recently. See for example [45], [3].

Although state-space models are often able to capture the essential properties of a complex dynamical system, and despite their theoretical appeal, they are not in extensive use, as it is usually difficult to find a sufficiently accurate model. In real industrial processes, the state variables, dynamics and observation mapping are rarely known accurately enough to allow model-based approaches without estimating the process from the data.

Estimating an NSSM-like (2), (3) with *unknown* nonlinearities $\mathbf{f}$and$\mathbf{g}$ from the observations is much more difficult than the estimation of an AR type model (1). Dual estimation techniques for simultaneous learning of the unknown mappings and the state have been limited. Methods have been proposed for learning linear state-space models [42], but estimating a nonlinear state-space model is inherently more difficult.

First, there are far more unknown parameters than in AR models, since the state of the system is usually unknown and cannot be completely determined from the observations in a simple way. Thus the main obstacle is overfitting, and some regularization is necessary. Second, there are infinitely many solutions. Any invertible nonlinear transformation of the state-space can be compensated by a suitable transformation of the dynamics and the observation mapping. Note that this does not pose a problem for change detection, though, since all these models give similar predictions of the observations and thus perform identically in change detection.

In the following, we apply a recently developed unsupervised method called nonlinear dynamical factor analysis (NDFA) [37] for off-line learning of the NSSM (2)–(3) of the process. The learned model can then be used efficiently for on-line change detection. With linear but unknown $\mathbf{f}$, unknown Gaussian $\mathbf{s}$, and without $\mathbf{g}$, the model would be the linear factor analysis model (see for example [22, Section 6.3]). The model (2)–(3) can thus be seen as a nonlinear dynamical extension of factor analysis.

NDFA is based on variational Bayesian learning, which imposes a natural regularization on the estimation problem. It can also be phrased in information theoretic terms as finding the model with minimum description length. Benefits are robustness to overfitting and ability to do model selection. MLP networks [16] are used to model the unknown nonlinear mappings $\mathbf{f}$ and $\mathbf{g}$, and the noise terms are assumed to be Gaussian. The MLP network provides an efficient parameterization for mappings in high dimensional spaces, and it is a universal approximator for smooth functions. Similar models using a radial-basis function network [16] as nonlinearity have been proposed in [13], [31] and using an MLP network in [5]. Some preliminary results on applying NDFA to change detection were given by us in [24], [38].

As aforementioned, the nonlinear dynamical reconstruction problem addressed in this paper is severely ill-posed [18]. In this paper, we apply variational Bayesian learning to estimate the parameters and hidden sources or states of the nonlinear state-space model. Variational Bayesian learning is a recently developed practical method for fitting a parametric approximation to the exact posterior probability density function [19], [26]. We describe how it can be used to regularize the dynamical reconstruction problem by restricting the complexity of the posterior structure of the solution.

The proposed method can also be seen as a nonlinear dynamical generalization of standard linear blind source separation (BSS) and independent component analysis (ICA) [22]. Several authors have recently applied variational Bayesian learning or closely related Bayesian methods to the linear ICA problem [2], [29], [8], [20]. One of the present authors has previously also used variational Bayesian learning for nonlinear ICA [25], and shown how the approach can be extended for nonlinear dynamical models using nonlinear state-space models [34], [36]. A general discussion of nonlinear ICA and BSS with many references can be found in [22, Ch. 17].

Even though the NDFA method presented in this paper can be regarded as a generalization of ICA, the recovered sources need not be independent. Our method tries to find the simplest possible explanation for the data, and hence avoids unnecessary dependencies between the recovered sources. If the process being studied cannot be described as a composition of one-dimensional (1-D) independent processes, the method tries to split it to as small pieces as possible, as will be seen in the experimental results.

The NDFA model and its off-line learning were covered in detail in the recent article [37]. Here we present a modification which results in an efficient on-line filtering algorithm for state estimation. We demonstrate that NDFA has been able to learn a model which performs very well in change detection.

The rest of this paper is organized as follows. In Section II, our NDFA model and the variational Bayesian estimation principle are reviewed, largely based on [37]. It is described how the parameters of the model as well as the hidden source signals can be learned from observation data in an unsupervised manner. Section III introduces the basics of change detection, details how NDFA is used for change detection and discusses the relation of the approach to Kalman filtering and smoothing. Section IV presents an extensive set of experiments of change detection using a fairly demanding simulated setting for a multivariate process involving both oscillatory and chaotic sources. Varying degrees of abrupt changes in the process dynamics are simulated and the abilities of several change detection methods are compared. Very good results for the NDFA method in detecting changes of the difficult nonlinear process are shown. The method clearly outperforms even such indicator based methods which are taylormade to the change. Finally, in Section V some conclusions are drawn.

II. Variational Bayesian Learning of Nonlinear State-Space Models

This section briefly outlines the NDFA model and learning algorithm. A thorough presentation can be found in [37].

The NDFA model is a dynamical extension of the static nonlinear model used in the recently developed nonlinear factor analysis (NFA) [25]. It assumes the nonlinear observation model given by (2) and Gaussianity of the hidden sources (or factors) $\mathbf{s}(t)$, but does not include the dynamics (3) as NDFA. The NFA algorithm estimates both the nonlinear mapping $\mathbf{f}$ and the factors $\mathbf{S} = \{\mathbf{s}(t) \,|\, t = 1, \ldots, T\}$ from the set of observations $\mathbf{X} = \{\mathbf{x}(t) \,|\, t = 1, \ldots, T\}$ only. By looking for a good *nonlinear* representation of the data (usually of a smaller dimension), NFA represents a nonlinear counterpart of the well-known principal component analysis (PCA) [22].

The NDFA method makes a further assumption that the hidden factors have a certain dynamical model that is well described by (3). In this way, NDFA is an algorithm for finding *dynamic* factors which give a good nonlinear explanation for the data. Combining the basic assumptions of NFA and the factor dynamics assumption (3), NDFA represents a new method for NSSM identification.

A. Model Structure

In the NDFA method, the unknown nonlinear mappings $\mathbf{f}$ and $\mathbf{g}$ in (2) and (3) are modeled by MLP networks having one hidden layer of sigmoidal $\tanh$ nonlinearities. The function realized by the network for $\mathbf{s}$ can be written in vector notation as

$$\mathbf{f}(\mathbf{s}) = \mathbf{B} \tanh(\mathbf{A}\mathbf{s} + \mathbf{a}) + \mathbf{b} \tag{4}$$

where the $\tanh$ nonlinearity is applied componentwise. $\mathbf{A}$ and $\mathbf{B}$ are the weight matrices of the hidden and output layers, respectively, and $\mathbf{a}$ and $\mathbf{b}$ are the bias vectors. They are unknown parameters, to be determined. The function $\mathbf{g}$ has a similar structure except that the MLP network is used to model only the change in the state values

$$\mathbf{g}(\mathbf{s}) = \mathbf{s} + \mathbf{D} \tanh(\mathbf{C}\mathbf{s} + \mathbf{c}) + \mathbf{d}. \tag{5}$$

When implementing the Bayesian approach, all the assumptions made in the model must be expressed in the form of the joint distribution of the observations $\mathbf{X}$, states $\mathbf{S}$ and parameters $\boldsymbol{\theta}$ of the model

$$p(\mathbf{X}, \mathbf{S}, \boldsymbol{\theta}) = p(\mathbf{X} \,|\, \mathbf{S}, \boldsymbol{\theta}) p(\mathbf{S} \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta}). \tag{6}$$

The parameters $\boldsymbol{\theta}$ include the parameters of the MLPs (4)–(5), the variance parameters of the noise terms $\mathbf{n}(t)$ and $\mathbf{m}(t)$ as well as hyperparameters. Due to the definition of the NSSM model, (6) can be written as a product of univariate Gaussian densities. First, the components of the noise vector $\mathbf{n}(t)$ are assumed to be independent and Gaussian; then the NSSM observation (2) yields the likelihood

$$\begin{aligned} p(\mathbf{X} \,|\mathbf{S}, \boldsymbol{\theta}) &= \prod_{i,t} p(x_i(t) \,|\, \mathbf{s}(t), \boldsymbol{\theta}) \\ &= \prod_{i,t} N(x_i(t); f_i(\mathbf{s}(t)), \exp(2v_i)) \end{aligned} \tag{7}$$

where $N(x; \mu, \sigma^2)$ denotes a Gaussian distribution over x with mean μ and variance σ^2, $f_i(\mathbf{s}(t))$ denotes the ith component of the output of $\mathbf{f}$ computable from (4), and v_i is a hyperparameter specifying the noise variance. The means $f_i(\mathbf{s}(t))$ depend on the parameters of the MLP (4). Second, the assumption of independent and Gaussian components of $\mathbf{m}(t)$ and the state update (3) give the prior model of the states

$$\begin{aligned} p(\mathbf{S} \,|\, \theta) &= p(\mathbf{s}(1)) \prod_{t=2}^{T} p(\mathbf{s}(t) \,|\, \mathbf{s}(t-1), \theta) \\ p(\mathbf{s}(t) \,|\, \mathbf{s}(t-1), \theta) &= N\left(\mathbf{s}(t); \mathbf{g}(\mathbf{s}(t-1)), \operatorname{diag}\left[\exp\left(2v_i^g\right)\right]\right) \\ p(\mathbf{s}(1)) &= N\left(\mathbf{s}(1); 0, \operatorname{diag}\left[\exp\left(2v_i^0\right)\right]\right). \end{aligned} \tag{8}$$

The parameters of the prior distributions such as the variance parameters v_i, v_i^g, v_i^0 are further assigned Gaussian priors making the prior $p(\boldsymbol{\theta})$ of the parameters hierarchical. For example, the noise parameters v_i of different components of the data share a common prior [25], [37].

B. Posterior Approximation and Regularization

Following the Bayesian approach, once the joint distribution (6) is defined and the set of observations $\mathbf{X}$ is obtained, all the relevant information about the unknown parameters $\mathbf{S}, \boldsymbol{\theta}$ is contained in the posterior

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

The conventional approach would be to find point estimates of the unknowns, for example, by maximizing the posterior (9), which yields the well-known maximum *a posteriori* (MAP) estimate. However, this strategy would easily lead to overfitting problems especially in this highly ill-posed estimation problem [26].

Instead of using point estimates, NDFA is based on *variational Bayesian learning* whose goal is to approximate the actual posterior (9) by an approximating distribution over $\mathbf{S}, \boldsymbol{\theta}$: $q(\mathbf{S}, \boldsymbol{\theta}) = q(\mathbf{S}, \boldsymbol{\theta}; \boldsymbol{\theta}_q)$ where $\boldsymbol{\theta}_q$ are the parameters of q. The advantage of this strategy is that more information about the full posterior can be preserved in its estimate $q(\mathbf{S}, \boldsymbol{\theta})$ than in some point estimates only. Also, selecting a suitable form for the approximation corresponds to a regularization of the estimation problem and helps avoid overfitting.

By estimating the density q, we reduce all the posterior information about the states $\mathbf{S}$ and parameters $\boldsymbol{\theta}$ to the pdf $q(\mathbf{S}, \boldsymbol{\theta}; \boldsymbol{\theta}_q)$. Then, different kinds of parameter and state estimates can be obtained from the estimated pdf $q(\mathbf{S}, \boldsymbol{\theta})$. For example, the mean $\hat{\mathbf{s}}(t) = \mathrm{E}_{q(\mathbf{S}, \boldsymbol{\theta})}[\mathbf{s}(t)]$ can be taken as a point estimate of the state $\mathbf{s}(t)$. Similarly, the variance $\tilde{s}(t) = \operatorname{var}_{q(\mathbf{S}, \boldsymbol{\theta})}[s(t)]$ can define a confidence region for the point estimate $\hat{\mathbf{s}}(t)$. The complexity of these calculations largely depends on the chosen parametric form of q.

In the type of variational learning which is called *ensemble learning*, the goodness of fit between the two probability density functions $p(\mathbf{S},\boldsymbol{\theta}\,|\,\mathbf{X})$ and $q(\mathbf{S},\boldsymbol{\theta})$ is measured by the Kullback-Leibler divergence

$$D(q(\mathbf{S},\boldsymbol{\theta})\,\|\,p(\mathbf{S},\boldsymbol{\theta}\,|\,\mathbf{X})) = \mathrm{E}_{q(\mathbf{S},\boldsymbol{\theta})}\left[\log\frac{q(\mathbf{S},\boldsymbol{\theta})}{p(\mathbf{S},\boldsymbol{\theta}\,|\,\mathbf{X})}\right] \qquad (10)$$

where the expectation is taken over the distribution $q(\mathbf{S},\boldsymbol{\theta})$. The Kullback-Leibler divergence is a standard dissimilarity measure for probability densities. It is always nonnegative and attains the value zero if and only if the two distributions are equal. Therefore, the parameters $\boldsymbol{\theta}_q$ of the pdf $q(\mathbf{S},\boldsymbol{\theta})$ are optimized to get the approximation as close to the true posterior as possible. Interpreted in information-geometric terms [1], minimizing the KL divergence means finding the projection of the true pdf $p(\mathbf{S},\boldsymbol{\theta}\,|\,\mathbf{X}))$ on the manifold of the approximating densities $q(\mathbf{S},\boldsymbol{\theta})$.

The posterior $p(\mathbf{S},\boldsymbol{\theta}\,|\,\mathbf{X})$ in (10) includes the term $p(\mathbf{X})$ which cannot be evaluated. However, as it is constant w.r.t. the parameters of the model, it can be removed and the actual cost function minimized in ensemble learning is

$$\begin{aligned} C &= D(q(\mathbf{S},\boldsymbol{\theta})\,\|\,p(\mathbf{S},\boldsymbol{\theta}\,|\,\mathbf{X})) - \log p(\mathbf{X}) \\ &= \mathrm{E}_q\left[\log\frac{q(\mathbf{S},\boldsymbol{\theta})}{p(\mathbf{S},\boldsymbol{\theta}\,|\,\mathbf{X})p(\mathbf{X})}\right] \\ &= \mathrm{E}_q\left[\log\frac{q(\mathbf{S},\boldsymbol{\theta})}{p(\mathbf{S},\boldsymbol{\theta},\mathbf{X})}\right] \geq -\log p(\mathbf{X}). \end{aligned} \qquad (11)$$

As can be seen from (11), the cost function C yields a lower bound for the model evidence.[1]

As follows from (7)–(8), the joint probability density $p(\mathbf{S},\boldsymbol{\theta},\mathbf{X})$ is a product of simple terms and the cost function (11) can be minimized efficiently if a suitably simple factorial form for the approximation is chosen. We use $q(\boldsymbol{\theta},\mathbf{S}) = q(\boldsymbol{\theta})q(\mathbf{S})$, where

$$q(\boldsymbol{\theta}) = \prod_j q(\theta_j) = \prod_j N(\theta_j;\bar{\theta}_j,\tilde{\theta}_j)$$

is a product of univariate Gaussian distributions. Hence the distribution for each parameter θ_j is parameterized by its mean $\bar{\theta}_j$ and variance $\tilde{\theta}_j$. These are part of the variational parameters $\boldsymbol{\theta}_q$ to be optimized.

The posterior approximation $q(\mathbf{S})$ is somewhat more complex. It takes into account the posterior dependences between the values of states at consecutive time instants imposed by the state equation (3)

$$q(\mathbf{S}) = \prod_i \left[q(s_i(1))\prod_t q(s_i(t)\,|\,s_i(t-1))\right].$$

The distribution $q(s_i(t)\,|\,s_i(t-1))$ is a Gaussian with mean that depends linearly on the previous value as in $\mu_i(t) = \bar{s}_i(t) + \breve{s}_i(t-1,t)(s_i(t-1) - \bar{s}_i(t-1))$, and variance $\dot{s}_i(t)$. Thus all the variational parameters $\boldsymbol{\theta}_q$ of the distribution $q(\mathbf{S},\boldsymbol{\theta})$ to be optimized are

$$\boldsymbol{\theta}_q = \{\bar{\theta}_j, \tilde{\theta}_j, \bar{s}_i(t), \breve{s}_i(t-1,t), \dot{s}_i(t), \forall j,i,t\}$$

A positive side effect of the restrictions on the approximating distribution $q(\mathbf{S},\boldsymbol{\theta})$ is that the nonlinear dynamical reconstruction problem is regularized and becomes well posed. With linear $\mathbf{f}$ and $\mathbf{g}$, the true posterior distribution of the states $\mathbf{S}$ would be Gaussian, while nonlinear $\mathbf{f}$ and $\mathbf{g}$ result in a non-Gaussian posterior distribution. Restricting the approximation $q(\mathbf{S})$ to be Gaussian even in the nonlinear model therefore favors smooth mappings and regularizes the problem. This still leaves a rotational ambiguity which is resolved by discouraging the posterior dependences between $s_i(t)$ and $s_j(t-1)$ with $j \neq i$.

This kind of posterior regularization has not been emphasized in the earlier work on Bayesian variational learning, probably because the method has not been previously applied to problems which would be as severely ill-posed as the nonlinear dynamic reconstruction problem that we are discussing here.

C. Evaluating the Cost Function

Using the above form and parameterizations for the approximating pdf, the cost function (11) becomes a function of all the parameters $\boldsymbol{\theta}_q$, as well as all the observation vectors in $\mathbf{X}$. Due to the simple form of the approximating pdf, the cost function splits into a sum of simple terms. Most of the terms can be evaluated analytically. Only the terms involving the outputs of the MLP networks cannot be computed exactly because of the nonlinearity involved. To evaluate those terms, the distributions of the outputs of the MLPs are calculated using a truncated Taylor series approximation for the MLPs. This procedure is explained in detail in [25] and [37].

Let us denote the two parts of the cost function (11) $C = C_p + C_q$ arising from the denominator and numerator of the logarithm, respectively, by $C_p = \mathrm{E}_q[-\log p(\mathbf{S},\boldsymbol{\theta},\mathbf{X})]$, and $C_q = \mathrm{E}_q[\log q(\mathbf{S},\boldsymbol{\theta})]$. The term C_q is simply a sum of negative entropies of Gaussians, and has the form

$$C_q = \sum_i -\frac{1}{2}[1+\log(2\pi\tilde{\theta}_i)] + \sum_{t,i} -\frac{1}{2}[1+\log(2\pi\dot{s}_i(t))].$$

The terms in the corresponding sum for C_p are somewhat more complicated but they are also relatively simple expectations over Gaussian distributions [25], [34]. For a detailed exposition, see [37].

Finally, collecting all terms, the cost function of the model with T observations can be presented in the following form

$$C = C(T) = \sum_{t=1}^{T}[C_{\mathbf{s}}(t) + C_{\mathbf{x}}(t)] + C_{\boldsymbol{\theta}} \qquad (12)$$

where $C_{\mathbf{s}}(t)$ includes the terms originating from $q(s_i(t)\,|\,s_i(t-1))$ and $p(s_i(t)\,|\,\mathbf{s}(t-1),\boldsymbol{\theta}), C_{\mathbf{x}}(t)$ includes the terms from $p(x_j(t)\,|\,\mathbf{S},\boldsymbol{\theta})$, and $C_{\boldsymbol{\theta}}$ includes the rest of the terms originating from the parameters. This decomposition will be used later when an on-line filtering version of the method is presented. Note that we will refer to the quantities $C(T)$ and $C_{\mathbf{s}}(t)+C_{\mathbf{x}}(t)$ as the cost of the model and the cost of the observation at time t, respectively.

D. Learning the Parameters

The parameters of the approximating distribution $q(\mathbf{S},\boldsymbol{\theta})$ are now optimized by minimizing (11), or in practice all the terms in

[1] The model evidence $p(\mathbf{X}\,|\,\mathrm{model})$ is here denoted by $p(\mathbf{X})$ for simplicity.

(12) with gradient based iterative algorithms. The exact update equations for the off-line learning algorithm are given in [37, Appendices A.5 and A.6.],

During one sweep of the algorithm all the parameters are updated once, using all the available data vectors $\mathbf{x}(1), \ldots, \mathbf{x}(T)$. One sweep consists of two different phases. The order of the computations in these two phases is the same as in standard supervised backpropagation [16] but otherwise the algorithm is different. In the forward phase, the distributions of the outputs of the MLP networks are computed from the current values of the inputs, and the value of the cost function is evaluated. In the backward phase, the partial derivatives of the cost function with respect to all the parameters are fed back through the MLPs and the parameters are updated using this information. At the beginning, the posterior means of most of the parameters are initialized to random values. The posterior variances are initialized to small constant values.

A more detailed description of the algorithm is given in [37], and a software implementation for the method is available at [35].

III. Change Detection

The general problem faced in process fault detection is that faults are rare and it is usually difficult to enumerate all possible faults that might occur. The standard approach is therefore to record data from normal operation of the process, build a model based on this data and then monitor the accuracy of the model in order to detect changes. Deviations from the behavior during the recorded normal operation are candidate faulty conditions.

Here, we explain how the NDFA approach can be applied to the problem of detecting changes in a nonlinear stochastic system. We start by reviewing the basics of change detection in stochastic systems. Then, the NDFA-based technique is proposed. Finally, some alternative nonlinear techniques based on estimation from data are discussed.

A. Basic Principles of Change Detection

The on-line detection of abrupt changes is a well studied problem for linear stochastic systems [4], [15]. Conventional change detection algorithms implement a sequential test by calculating at each time instant a test statistic $g(t)$ and raising alarms when $g(t)$ exceeds a chosen threshold h:

$$\text{decision at time}\, t = \begin{cases} \text{change}, & g(t) \geq h \\ \text{no change}, & g(t) < h. \end{cases} \quad (13)$$

Thus $g(t)$ is an indicator of the model consistency; it should be close to zero in fault-free conditions and depart from zero when a change in the system occurs.

A classical example of such a statistic is the well known CUSUM algorithm applied to detecting increase of the mean of a random independent sequence [15]. For scalar samples $s(t)$, the CUSUM one-sided test recursively calculates $g(t)$ as

$$g(t) = [g(t-1) + s(t) - m - \nu]^{+} \quad (14)$$

where m is the mean of $s(t)$ before change, ν the parameter preventing a drift yielding false alarms and $[x]^{+} = \max(0, x)$. Although the statistic (14) is justified as a sequential probability ratio test for an independent Gaussian sequence [4], it can be used as a general algorithm for detecting changes in the mean of a random scalar variable $s(t)$.

Detecting changes in the mean value of a *multivariate* independent Gaussian sequence is the basic problem considered for linear systems. The main techniques are the standard CUSUM and the generalized likelihood ratio (GLR) algorithms, respectively, for the case of known and unknown parameters after change.

The basic principle of detecting additive changes in more complex linear models such as AR or state-space models is transformation from observations to innovations

$$\boldsymbol{\varepsilon}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t) \quad (15)$$

where $\mathbf{x}(t)$ is the system output (observation) and $\hat{\mathbf{x}}(t)$ is the output estimate given by the AR-model or the Kalman filter. Then the relevant basic problem is solved [4], [15].

The case of nonadditive changes (for example, changes in the covariance matrix of a Gaussian sequence) is more complex. However as stated in [4], the same approach can be tried for detecting this type of changes as well. Among other proposed methods are monitoring shifted log-likelihood function [33], the GLR algorithm and the local approach [4].

B. Change Detection With NDFA

In change detection based on models estimated from data, there is always a tradeoff between overfitting and flexibility of the process model. Test statistics based on simple underlying models are insensitive to a large number of changes which affect those aspects of the process that are not captured by the model. Complex models easily overfit to the learning data which leads to false alarms for new data from normal operation of the process. To avoid a large number of false alarms, the threshold h needs to be set much higher than would be expected based on the performance of the model for the learning data. An acceptable false-alarm rate can be obtained by tuning h based on test data but the high value of h will then result in missed detections of real faults.

NDFA is particularly well suited to model-based change detection because the underlying nonlinear dynamic model is very flexible, making the model sensitive to changes, and the learning algorithm based on variational Bayesian learning is very robust against overfitting.

1) Estimation of States for New Observations: The model learned off-line by the method presented in Section II can be used for change detection by estimating the states $\mathbf{S} = \{\mathbf{s}(1), \ldots, \mathbf{s}(T)\}$ on-line and monitoring the prediction accuracy. The underlying assumption is that as long as the process characteristics stay constant, prediction accuracy is good and it deteriorates when a change occurs. The constant characteristics of the process are represented in the nonlinear mappings $\mathbf{f}$ and $\mathbf{g}$ and the statistics of the additive noise terms $\mathbf{n}$ and $\mathbf{m}$.

On-line state estimation for NSSM could be implemented by standard extended Kalman filters (EKF) [14], [17], but here we present a method based on the same variational Bayesian approach as was used for model estimation. The main benefits are that very little extra coding is needed and the cost function used in NDFA can be directly used for devising the test statistic monitored for change detection.

The idea of the NDFA filtering lies in the augmentation of sets $\mathbf{X}$ and $\mathbf{S}$ with the new measurement $\mathbf{x}(t)$ and the predicted state value $\mathbf{s}(t) = \mathbf{g}(\mathbf{s}(t-1))$ and performing a few iterations of the NDFA algorithm to estimate the states $\mathbf{S}$. The new measurement at time t will have an effect on all the estimates of the states $\mathbf{s}(\tau < t)$ but in practice the effect is usually restricted to the immediate past. Therefore, only a small subset of $\mathbf{S}$ corresponding to the recent history of size L is updated. The algorithm is as follows:

1) Begin the state estimation at time t. Assume that observations $\mathbf{x}$ and state estimates $\mathbf{s}$ already exist for time $t-L+1, \ldots, t-1$.
2) When a new measurement $\mathbf{x}(t)$ is obtained, new vector sets $\mathbf{X}$ and $\mathbf{S}$ are constructed from the last L data points:
$$\mathbf{X} = \mathbf{X}_{t-L+1}^{t} = \{\mathbf{x}(t-L+1), \ldots, \mathbf{x}(t)\},$$
$$\mathbf{S} = \mathbf{S}_{t-L+1}^{t} = \{\mathbf{s}(t-L+1), \ldots, \mathbf{s}(t)\}.$$
Here, the mean of the last state $\mathbf{s}(t)$ is initialized with the predicted value $\bar{\mathbf{s}}(t) = \mathbf{g}(\bar{\mathbf{s}}(t-1))$ while the state variances $\tilde{s}_i(t)$ are set to some small values.
3) Run the NDFA iterations for the last L states $\mathbf{S}$ at most N times, updating the values $\bar{s}_i(t)$ and $\tilde{s}_i(t)$. Note that only the last L terms of $C_{\mathbf{s}}(t)$ and $C_{\mathbf{x}}(t)$ need to be taken into account in the cost function (12).
4) Increment t by one and return to step 2) for a new observation $\mathbf{x}(t)$.

The parameter N in step 3) defines how persistent we are in optimizing the cost function and hence in searching for good estimates of new states. A larger number of iterations always provides better estimates and the main restriction on the choice of N is the allotted time for calculations.

In practice we have seen that the number of iterations needed for arriving at a good estimate can vary greatly. In nonlinear models the true posterior distribution of the state can be multimodal but in practice we only represent one mode. There seem to be some cases where the estimate is trapped in a mode which later turns out to be the wrong one in light of the new observations. Usually the initial guess $\mathbf{s}(t) = \mathbf{g}(\mathbf{s}(t-1))$ is very good and a couple of iterations suffice but in the cases where the mode needs to be switched for past state estimates, far more iterations may be needed to minimize the cost function and to obtain a consistent estimate of the past states.

Thus, we need a rule which assesses the convergence of the NDFA procedure and stops the optimization process when an accurate enough estimate is found. Such a heuristic rule is to stop the iterations when the difference in the value of the cost function $C_{\text{old}}(t-1)$ before and $C(t)$ after adding the new observation $\mathbf{x}(t)$ and running new iterations decreases below a given threshold Δ

$$C(t) - C_{\text{old}}(t-1) < \Delta. \tag{16}$$

It would also be possible to couple the stopping criterion with change detection algorithm. The state estimates would then be iterated until the threshold for alarm is no longer exceeded or the predefined maximum number of iterations N would be reached. We decided to decouple state estimation from change detection in order to be able to run a large number of change detection simulations without the need to reestimate the states. The present implementation of on-line state estimation could also be used for other applications besides change detection.

2) NDFA-Based Test Statistic for Change Detection: We now present the on-line NDFA-based approach for change detection. First we note that the value of the cost function (11) can be used as an estimate of the probability of the observed sequence $\mathbf{X}_1^t = \{\mathbf{x}(1), \ldots, \mathbf{x}(t)\}$: From the assumption that the approximation is close to the true posterior and therefore $D(q(\mathbf{S}, \boldsymbol{\theta}) \,\|\, p(\mathbf{S}, \boldsymbol{\theta} \,|\, \mathbf{X}_1^t)) \approx 0$, we get $C(t) \approx -\log p(\mathbf{X}_1^t)$.

Then, the difference of two values of $C(t)$ is

$$\begin{aligned} C(t) - C(t-M) &\approx -\log p\left(\mathbf{X}_1^t\right) + \log p\left(\mathbf{X}_1^{t-M}\right) \\ &= -\log \frac{p\left(\mathbf{X}_1^t\right)}{p\left(\mathbf{X}_1^{t-M}\right)} \\ &= -\log p\left(\mathbf{X}_{t-M+1}^t \middle| \mathbf{X}_1^{t-M}\right) \end{aligned} \tag{17}$$

which is the conditional log-likelihood function of the last M observations. In practice the difference can be computed by summing the cost for the last M observations:

$$C(t) - C(t-M) = \sum_{\tau=0}^{M-1} C_{\mathbf{s}}(t-\tau) + C_{\mathbf{x}}(t-\tau). \tag{18}$$

Dividing (18) by M gives the estimate of the conditional expectation of the log-likelihood or the entropy rate [9]:

$$\begin{aligned} s(t) &= \frac{1}{M}[C(t) - C(t-M)] \\ &\approx \mathrm{E}\left\{-\log p\left(\mathbf{x}(\tau) \middle| \mathbf{X}_1^{\tau-1}\right) \,|\, t-M+1 \leq \tau \leq t\right\} \end{aligned} \tag{19}$$

Following the idea of monitoring the shifted log-likelihood [33], we propose to monitor the deviation of the quantity (19) from its expected value which can be calculated from the training sequence of size T:

$$\mathrm{E}\{s(t) \,|\, \text{no change}\} \approx \frac{1}{T}[C(T) - C_{\boldsymbol{\theta}}]$$

Using the standard one-sided CUSUM test (14), we get the test statistic:

$$\begin{aligned} g(t) = \Bigg\{ g(t-1) &+ \frac{1}{M}[C(t) - C(t-M)] \\ &- \frac{1}{T}[C(T) - C_{\boldsymbol{\theta}}] - \nu \Bigg\}^{+} \end{aligned} \tag{20}$$

This is the statistic on which the standard decision rule (13) is applied in NDFA-based change detection. Similarly, a complementary one-sided CUSUM test can be constructed to detect the decrease of $\mathrm{E}\{s(t)\}$.

The proposed statistic (20) is similar to the one proposed in [33] where the cumulative sum is constructed over the shifted log-likelihood function. However, it is not the same: Firstly, calculation of the test statistic is done over a sliding window of size M. Secondly, a two-sided CUSUM test in the form (20) makes it possible to detect changes which lead both to decrease and increase of the entropy (19) (see discussion in [4, p. 311]).

3) Relation to Kalman Filtering and Smoothing: The standard technique for on-line state estimation in linear state-space models is Kalman filtering. This procedure recursively calculates the posterior of the states $p(\mathbf{s}(t)\,|\,\mathbf{X}_1^t)$ based on the information from the past only. Kalman smoothing extends the filtering procedure by the backward phase on which the information from future observations is propagated backward in time to give the updated posterior $p(\mathbf{s}(t)\,|\,\mathbf{X}_1^t, \mathbf{X}_{t+1}^T)$.

These calculations can be done in two phases because of the essential properties of linear systems:

- The posteriors of the states are all Gaussian distributions and can be modeled as $N(\mathbf{s}(t); \bar{\mathbf{s}}(t), \mathbf{\Sigma}_{\mathbf{s}(t)})$.
- The effect of an observation $\mathbf{x}(\tau)$ on the mean $\bar{\mathbf{s}}(t)$ of any state posterior $p(\mathbf{s}(t)\,|\,\mathbf{X}_1^t), p(\mathbf{s}(t)\,|\,\mathbf{X}_1^t, \mathbf{X}_{t+1}^T)$ is linear.
- The covariance matrices $\mathbf{\Sigma}_{\mathbf{s}(t)}$ do not depend on observations and can be calculated beforehand.

For nonlinear state-space models, the standard on-line state estimation is done by the extended Kalman filter (EKF). Since none of the above properties hold for nonlinear systems, EKF uses the Gaussian *approximation* $N(\mathbf{s}(t); \bar{\mathbf{s}}(t), \mathbf{\Sigma}_{\mathbf{s}(t)})$ for the state posterior. The parameters $\bar{\mathbf{s}}(t), \mathbf{\Sigma}_{\mathbf{s}(t)}$ are estimated based on the first-order linearization of the mappings $\mathbf{f}$ and $\mathbf{g}$ at the current state estimates $\bar{\mathbf{s}}(\tau < t)$. In the EKF, these nonlinearities are assumed to be known, contrary to NDFA, which provides a disciplined method to estimate the nonlinearities from the data.

In EKF, the smoothed posterior $p(\mathbf{s}(t)\,|\,\mathbf{X}_1^t, \mathbf{X}_{t+1}^T)$ can be estimated by propagating the information from future observation backward in time using linearized Kalman smoothing. This gives new state estimates $\bar{\mathbf{s}}(t)$ and therefore a new linearized version of the original system.

Therefore, several extended Kalman filtering-smoothing iterations are required to find a good approximation of the state posterior. It is also possible that the iterations become unstable.

The NDFA on-line state estimation we proposed resembles extended Kalman smoothing. The posterior state distribution is approximated by a Gaussian distribution and several iterations may be needed. The main difference is in the criterion for obtaining the Gaussian posterior approximation of the states. In NDFA, the true posterior is implicitly estimated based on a second-order Taylor series approximation of the nonlinear functions as opposed to the first-order approximation typically used in EKF. The Gaussian approximation is reached by minimizing the Kullback-Leibler divergence between the approximation and the implicitly estimated true non-Gaussian posterior.

Unlike EKF, NDFA does not need matrix inversions. This is possible since the Gaussian approximation uses a diagonal covariance matrix for each state. This makes the computations simpler but can deteriorate the estimation accuracy as explained in [23]. On the other hand, the NSSM has an infinite number of alternative state representations and learning the model by NDFA automatically selects the representation which minimizes the posterior dependences of states. The posterior dependencies between $s_i(t)$ and $s_i(t-1)$ are taken into account.

This having been said, it should be stressed that the important feature which makes NDFA useful for change detection is robustness against overfitting during learning. The method used for on-line estimation is probably not crucial and EKF or advanced methods such as particle filters could be used instead of the NDFA-based method we used here.

C. Alternative Change Detection Methods Based on Learning From Data

The change detection methods based on model estimation have not yet entered the mainstream of change detection; a few different approaches are discussed in [6], [7]. Here, we describe several possible alternative change detection techniques which can be used when the process model is estimated from data. We emphasize the nonlinear multivariate estimation methods starting however from the simplest Gaussian model.

1) Monitoring the Mean of a Multivariate Gaussian: As we discussed in the introduction, the simplest approach to modeling a multivariate sequence is to use some simple indicators such as the first moment $\boldsymbol{\mu}$ and the second moment $\mathbf{\Sigma}$ and monitor changes in these parameters.

Shewhart control chart is one of the simplest algorithms for detecting changes in the mean $\boldsymbol{\mu}$ of a multivariate Gaussian. Using a sliding window of L samples, it calculates the sample mean $\bar{\mathbf{x}}_L(t) = 1/L\sum_{\tau=0}^{L-1}\mathbf{x}(t-\tau)$ which is distributed according to $N(\boldsymbol{\mu}, \mathbf{\Sigma}/L)$ in a change-free situation if $\boldsymbol{\mu}, \mathbf{\Sigma}$ are known exactly. Therefore the T^2-statistic [7] can be calculated

$$g(t) = L(\bar{\mathbf{x}}_L(t) - \boldsymbol{\mu})^{\mathrm{T}}\mathbf{\Sigma}^{-1}(\bar{\mathbf{x}}_L(t) - \boldsymbol{\mu}) \tag{21}$$

and then thresholded by (13). The mean and covariance matrix before change can easily be estimated from training data.

2) Monitoring the Mean and Covariance of a Multivariate Gaussian: For $n_{\mathbf{x}}$-dimensional Gaussian observations $\mathbf{x}$, detecting changes both in the mean $\boldsymbol{\mu}$ and the covariance matrix $\mathbf{\Sigma}$ is possible by considering the squared normalized innovations ([15], p. 324):

$$s(t) = (\mathbf{x}(t) - \boldsymbol{\mu})^{\mathrm{T}}\mathbf{\Sigma}^{-1}(\mathbf{x}(t) - \boldsymbol{\mu}). \tag{22}$$

In the case of known $\boldsymbol{\mu}, \mathbf{\Sigma}$, the quantity (22) is $\chi^2(n_{\mathbf{x}})$ distributed and, therefore, the standard CUSUM algorithm (14) can be used to monitor its mean $m = n_{\mathbf{x}}$ (the approximation $m \approx n_{\mathbf{x}}$ can be used if $\boldsymbol{\mu}, \mathbf{\Sigma}$ are estimated from data). Note that this approach uses the same idea of monitoring the expectation of the log-likelihood proposed to use with NDFA-based change detection in Section III-B.

3) Nonlinear AR-Models: The nonlinear function $\mathbf{f}$ in the AR-model (1) can be learned from data using MLP or RBF networks [16], [10]. Another popular tool for modeling dynamics is recurrent neural networks (RNN). For example, the Elman network [10] assumes the following dynamical model of the observations $\mathbf{x}(t)$ using the hidden states $\mathbf{s}(t)$

$$\mathbf{x}(t) = \mathbf{C}\mathbf{s}(t) + \mathbf{c} + \mathbf{n}(t) \tag{23}$$

$$\mathbf{s}(t) = \sigma(\mathbf{A}\mathbf{s}(t-1) + \mathbf{D}\mathbf{x}(t-1) + \mathbf{a}) \tag{24}$$

where the logistic function $\sigma(x) = 1/(1+\exp(-x))$ is a scaled version of the $\tanh$ nonlinearity used in MLPs (4)–(5). This model can be trained using the standard learning techniques such as the backpropagation algorithm. Then, the parameters of the observation noise $\mathbf{n}(t)$ can be estimated from the prediction error calculated on the training data.

After training, the neural network is used to estimate the system output and the difference between actual and estimated output gives the innovations (15). In order to detect changes, the statistical properties of innovations can be tested for example by the Shewhart test (21). Using the neural network innovations for change detection has also been considered in [30] and [6].

MLP, RBF, and recurrent MLP networks are the state-of-the-art techniques for nonlinear dynamical modeling, and the optimal change detection techniques are based on the conditional probability of the observations [4]. Therefore, if point estimates of the network parameters are used and the driving noise in the AR-model (1) and the RNN-model (23)–(24) is assumed Gaussian, the optimal detection of additive changes is based on the transformation from observations to innovations. Moreover, as we already mentioned, the same approach can be tried for detecting other types of changes as well.

4) Nonlinear State-Space Approaches: Recently, Bayesian techniques have been introduced for the identification of nonlinear state-space models (2)–(3). However, they are not applicable to very complex problems due to the drawbacks mentioned in [37]: In [13] and [31] the nonlinear mappings are modeled by RBF networks and the required number of RBFs increases exponentially with the dimension of the internal state $\mathbf{s}(t)$. Briegel and Tresp [5] model the nonlinearities using MLP networks with sampling. In [43], [44], the unknown state and dynamics are estimated using the extended Kalman filtering while the known mapping $\mathbf{f}$ is assumed to be the identity mapping. It is unclear whether the approach is suitable for the general NSSM model.

IV. Experiments

To test the properties of the NDFA change detection algorithm, we used the artificial data from [37]. The process states were simulated using eight time series representing three independent dynamical processes, two of which were Lorenz processes and one a simple harmonic. A Lorenz process with three state variables s_1, s_2, s_3 is described by the nonlinear dynamic equation

$$\begin{aligned}\Delta s_1 &= \sigma(s_2 - s_1)\\ \Delta s_2 &= \rho s_1 - s_2 - s_1 s_3\\ \Delta s_3 &= s_1 s_2 - \beta s_3\end{aligned}$$

where (σ, ρ, β) are three parameters. This system has three-dimensional chaotic dynamics. The three-dimensional vector-valued function from (s_1, s_2, s_3) to the updates $(\Delta s_1, \Delta s_2, \Delta s_3)$ is now the nonlinear part of the mapping $\mathbf{g}$ in (3). The additive noise $\mathbf{m}(t)$ in (3) was omitted. In our simulated data, the source processes were:

- three time series describing a Lorenz process with parameters [3 26.5 1];
- three time series describing another Lorenz process with parameters [4 30 1];
- two time series of a harmonic oscillator with angular velocity $1/3$.

Similarly to [37], one dimension of each underlying process was hidden and therefore only five linear projections of the eight states were present in the observed nonlinear mixtures. An arbitrary nonlinear mapping $\mathbf{f}$ producing ten observed signals from the five states was implemented using a random MLP network with the inverse hyperbolic sine activation function. Finally, the observations were corrupted by a zero-mean Gaussian noise $\mathbf{n}(t)$ with the standard deviation 0.1 while the standard deviation of the signal was normalized to unity.

First, the training sequence of $T = 1000$ samples was generated and the NDFA off-line procedure described in Section II was applied to estimate the NSSM model (2)–(3). The structure of the model and the learning scheme were optimized based on the value of the cost function. All data were used for learning and there was no validation set. The prediction performance was observed to correlate well with the attained value of the cost function. The MLP networks (4), (5) in the optimal NDFA model had 30 neurons in the hidden layer and the number of states was 9. The cost function continued to improve very long, and the learning was stopped after half a million iterations when the cost no longer seemed to decrease. See [37] for details.

We also trained the nonlinear AR (NAR) model (1) using an MLP network and the standard backpropagation algorithm. The data set was split to training and validation set. The criterion for optimizing the model structure and learning scheme was prediction accuracy for the validation set. The best model had 20 inputs and one hidden layer of 30 neurons. The number of delays was $d = 10$, and the dimension of the inputs to the MLP was compressed from 100 to 20 using standard PCA. These results have also been presented in [37].

In the present paper, we also tested the RNN approach discussed in Section III-C. The Elman network (23)–(24) with 30 neurons in the hidden layer was trained with the standard backpropagation algorithm. The structure of the model was again optimized to minimize the prediction error for the validation set. The best RNN used delayed inputs $\mathbf{x}(t), \ldots, \mathbf{x}(t-d)$ with $d = 10$ in the state (24).

Finally, we tested a Bayesian learning method [27], [28], [11] for learning the NAR model (1). The structure of the MLP which modeled the nonlinearity $\mathbf{f}$ was optimized to achieve the best prediction accuracy on the validation set. The best model had 20 inputs and one hidden layer of 20 neurons. The number of delays was $d = 10$, and the dimension of the inputs to the MLP was compressed from 100 to 20 using standard PCA.

After training, the fitted NDFA model was tested on new simulated measurements by applying the NDFA on-line state estimation and change detection procedures described in Section III-B. The NAR and RNN-based change detection discussed in Section III-C were also tested on the same change detection problems. In the following, we present the results of these experiments.

A. Experiments Without Changes

In the first experiment, we apply the proposed NDFA on-line state estimation technique from Section III-B1 to the long run continuation of the training data without simulating any changes in the data model. The sequence of 10 000 new observations was tested. The parameters of the NDFA state estimation procedure were: the size of the sliding window $L = 20$, the different values of the maximum number of NDFA iterations were

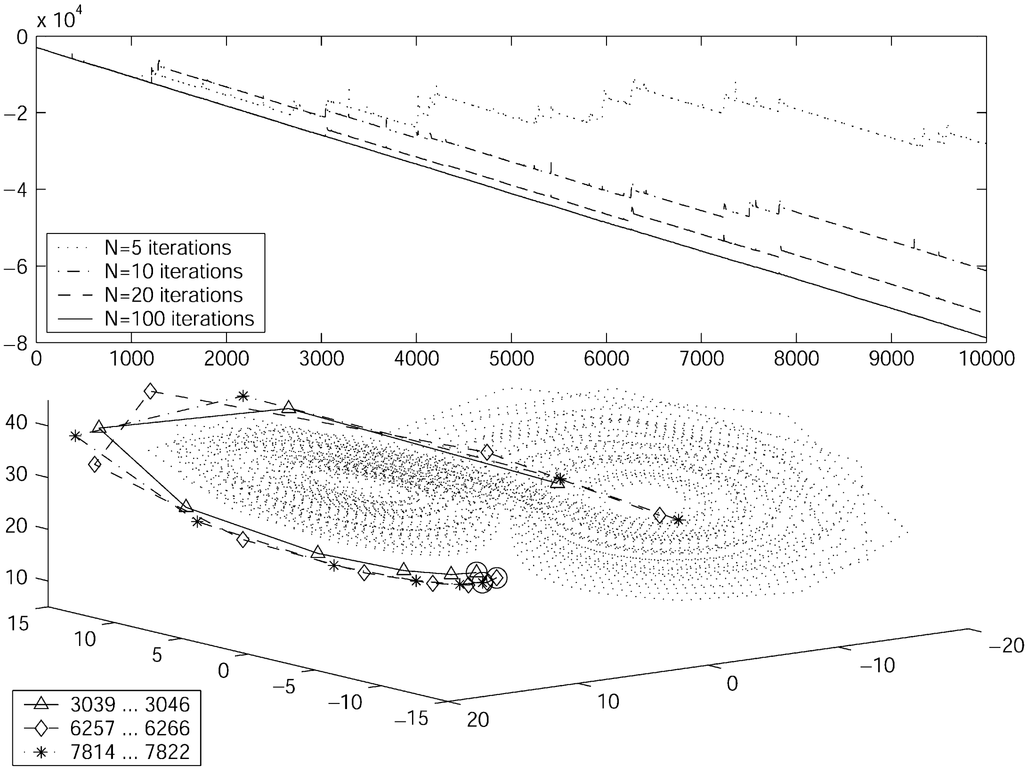

Fig. 1. Above: The cost function $C(t)$ estimated for the new 10 000 measurements without changes in the model. The jumps in the cost function correspond to false alarms. Below: The dynamics of an underlying Lorenz process on three intervals difficult for state estimation: $3039\ldots3046$, $6257\ldots6266$, and $7814\ldots7822$. The dotted line shows the training sequence. The starting points of the trajectories are circled.

$N = 5, 10, 20, 100$, and the heuristic NDFA convergence rule (16) with $\Delta = 0$ was used. The value of Δ should be compared with the average cost of the observations for training data which was -7 per sample.

The sequence of the cost function values $C(t)$ obtained for the new 10 000 observations is presented in Fig. 1. The figure shows that the cost function is mostly decreasing with a constant slope except for the points where abrupt jumps occur. This behavior means that the conditional probability of new data is mostly constant and abruptly decreases at some points when the false alarms are raised.

As Fig. 1 indicates, many of the false alarms can be avoided by performing a larger number of NDFA iterations, which means that we spend more time for calculations at each time instant, improving the accuracy of the state estimation. Some of the jumps can be eliminated by a slight increase of N. However on some intervals, significantly more iterations are required.

Let us take a closer look at the dynamics of an underlying Lorenz process on the difficult intervals (see Fig. 1). One can see that the difficulty of the state estimation on these intervals is caused by the fact that the Lorenz process enters a state space region that was not presented in the training data. The algorithm cannot find an accurate estimate for the new states if a small number of iterations is accomplished, which causes the growth of the cost function and raising the false alarm about the model change.

Thus, raising the false alarms on the difficult intervals is somewhat reasonable as, even though the data model stays constant, the algorithm reacts to some new features of the monitored process: it enters a new region in the state space. However, these alarms are still undesirable and should be avoided if possible.

TABLE I
THE NUMBER OF NDFA ITERATIONS ACCOMPLISHED FOR THE 10 000 NEW OBSERVATIONS

N	5	10	20	100
Iterations	20304	22183	23127	24074

Increasing the number of NDFA iterations solves this problem, which means that the NDFA on-line algorithm has been able to recognize the data even from the regions that have not been presented in the training sequence. It should also be noted that using a larger N does not dramatically increase the average number of NDFA iterations if a stopping rule like (16) is used: Only difficult intervals require a large number of iterations whereas most observations are processed with a few iterations only (see Table I).

B. Experiments With Pronounced Changes in the Process Dynamics

A change in a real process can take place in a variety of ways. In the NSSM model, it is reflected in a change either in the mapping $\mathbf{f}$ from the states to the observations, in the underlying state dynamics determined by the mapping $\mathbf{g}$, or in the noise levels. Different types of changes can be detected by monitoring the cost function. In this paper, we concentrate on the most demanding case where the nonlinearity $\mathbf{g}$ undergoes some change. The nonlinear mapping $\mathbf{f}$ can make this change

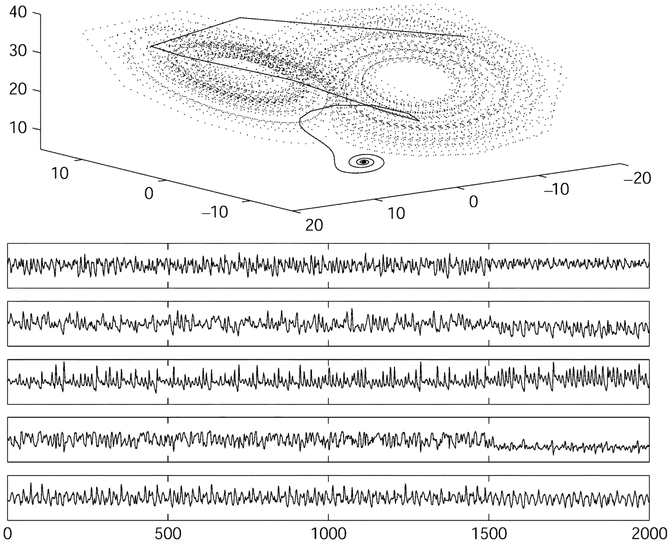

Fig. 2. Above: The Lorenz process with a pronounced change in the dynamics. The dotted line shows the 1000 points of the training sequence and the subsequent 500 points with the same model. The 500 points after the change are plotted with the solid line. Below: Five out of ten observations are presented. The simulated change at $t = 1500$ clearly affects the mean and covariance of the observed process.

hardly discernible in the observations, making the change detection problem very challenging.

In the following experiment, we simulate a pronounced abrupt change in the process dynamics: As Fig. 2 shows, changing the parameters of the first Lorenz process from $[3\ 26.5\ 1]$ to $[3\ 5\ 1]$ significantly affects the character of the underlying process and the changes are clearly visible from the observations.

The different techniques discussed in Section III were applied to this change detection problem. We test the NDFA algorithm proposed in Section III-B as well as other nonlinear methods and simple algorithms based on monitoring process indicators. The set of the compared methods is as follows:

1) *The NDFA algorithm* with the following parameters: $M = 20$ last points accounted in the statistic calculations (20), the drift parameter $\nu = 0$. The parameters of the NDFA on-line state estimation were the same as in the previous experiment.
2) *Nonlinear autoregression (NAR) approach*. The prediction error of the trained NAR-model was tested with the Shewhart chart test (21). Both the standard backpropagation and the Bayesian approach were tested.
3) *Recurrent neural network (RNN) approach*. The prediction error of the trained RNN-model was tested with the Shewhart chart test (21).
4) *CUSUM algorithm* which monitors the mean and the covariance matrix of the observations using the squared innovation approach described is Section III-C-2.
5) *Shewhart charts* (21) for monitoring changes in the mean.

We first tested all the alternative methods on the change detection problem shown in Fig. 2. All the methods successfully detected the change simulated at $t = 1500$ (see Fig. 3).

Then, the comparison of the NDFA change detection procedure with the alternative methods was performed by assessing two performance measures of the algorithms: the probability of false alarms P_f and the average time to detection D. The performance was assessed by simulating 100 changes at different time instances and estimating the two mentioned measures. The parameters of the all algorithms were optimized to achieve the best change detection performance.

The P_f-measure indicates how often an algorithm produces alarms when a monitored process does not undergo any changes. Denoting the time instant of the change by t_c and the time instant of the alarm as t_a, the probability of false alarms can be defined as

$$P_f = P(t_a < t_c). \tag{25}$$

In practice, this can be estimated from the trials by counting the relative frequency of false detections.

The other chosen performance measure D shows how long time we have to wait after a change until we get the alarm:

$$D = E(t_a - t_c \,|\, t_a \geq t_c). \tag{26}$$

It can practically be estimated by taking the average time between the true change t_c and the detected change t_a over all the 100 simulated changes.

Both measures (25) and (26) of course depend on the decision threshold h, which was varied in the simulations over a suitable range. Note that taking too high a threshold gives rise to situations where some changes are not detected at all and so D for these cases would be infinitely large. Such large values of h leading to missed detections are not shown in the experiments.

Fig. 4 shows the simulation results. The D-measure (26) is plotted against the false alarm probability P_f of (25). Each indicated point on the curves gives one (P_f, D) pair for a given value of the decision threshold h in the rule (13). Thus, one curve corresponds to one of the algorithms with different values of threshold h. The closer a curve is to the origin, the faster the algorithm can detect the change with low false alarm rate.

The NDFA method with the maximum of $N = 100$ iterations clearly outperforms all the other algorithms. It detected the simulated changes very fast with a low rate of false alarms. The NDFA with $N = 20$ performed very well too but only for high values of the P_f-measure corresponding to small thresholds h. In this case, increasing the threshold, as usual, provided smaller P_f values which however could not be made smaller than a certain limit: After this limit, larger values of h caused situations with missed detections.

This result can be explained based on the experiments in Section IV.A. As Fig. 1 shows, a small maximum number of NDFA iterations does not permit to get rid of many unjustified jumps in the cost function. These jumps have the same influence to the NDFA test statistic as the jumps caused by real changes in the process dynamics. Thus, taking a larger threshold leads to ignoring both the difficult points and the points of real changes, which causes the missed detection situations.

One can see from Fig. 4 that the NAR and RNN-based approaches learned by standard backpropagation work in this problem even worse than simpler indicator-based algorithms. To understand this, consider two very simple change detection algorithms. One monitors the mean of the observations and the other monitors the mean of the innovation process

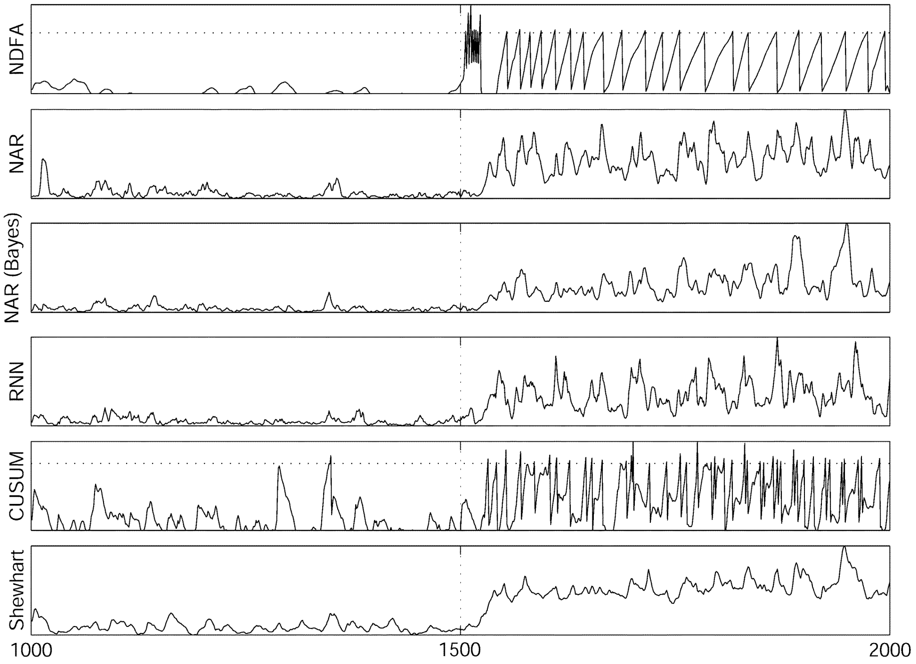

Fig. 3. The test statistics $g(t)$ of six alternative methods calculated for the 1000 new observations with the pronounced change of the dynamical model at $t = 1500$ (see Fig. 2). Using the decision rule (13), all the methods successfully detect the change.

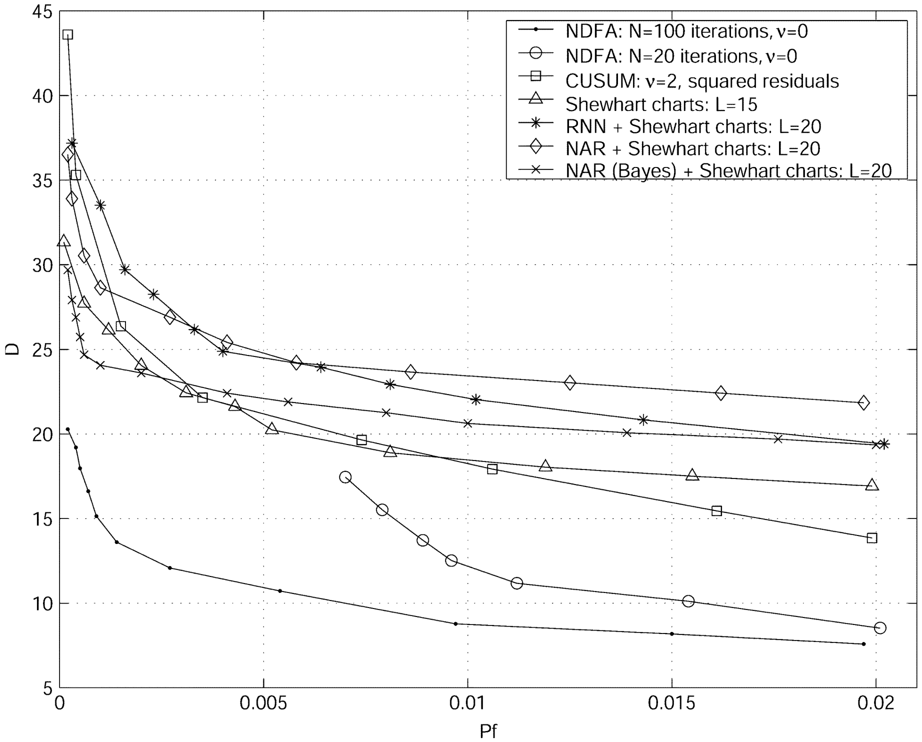

Fig. 4. Performance comparison of various change detection methods for the problem with the pronounced changes in the Lorenz process dynamics.

$\varepsilon(t) = \mathbf{x}(t) - \mathbf{x}(t-1)$. The problem of the latter approach is that the mean of the innovation process $\varepsilon(t)$ remains zero even if the mean of the observations $\mathbf{x}(t)$ changes. Adding a model of the dynamics thus first makes change detection more difficult. Only after the model of the dynamics is sufficiently good, monitoring the innovation process performs better. The NAR and RNN models started suffering from overfitting before they were able to find sufficiently good models for the dynamics.

C. Experiments With Slight Changes in the Process Dynamics

We now test the NDFA change detection algorithm in a case when the model change is less severe than above. The change of the dynamical model is simulated by changing the parameters of the first Lorenz process from $[3\ 26.5\ 1]$ to $[3\ 20\ 1]$ at time instant $t = 1500$. This change is quite clear when looking at the underlying Lorenz process dynamics but hardly visible from the

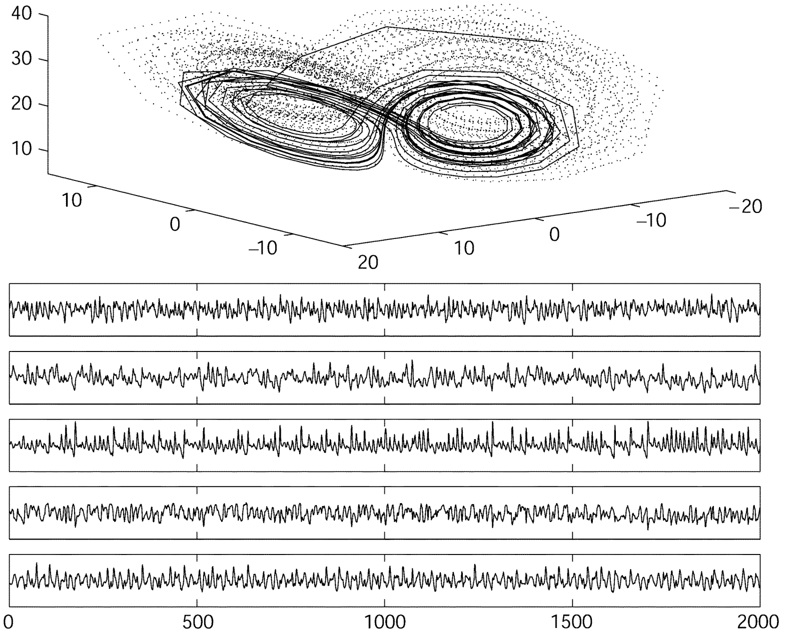

Fig. 5. Above: The Lorenz process with slight changes in the dynamics. The dotted line shows the 1000 points of the training sequence and the subsequent 500 points with the same model. The 500 points after the change are plotted with the solid line. Below: Five out of ten observations are presented. The simulated change at $t = 1500$ is hardly visible.

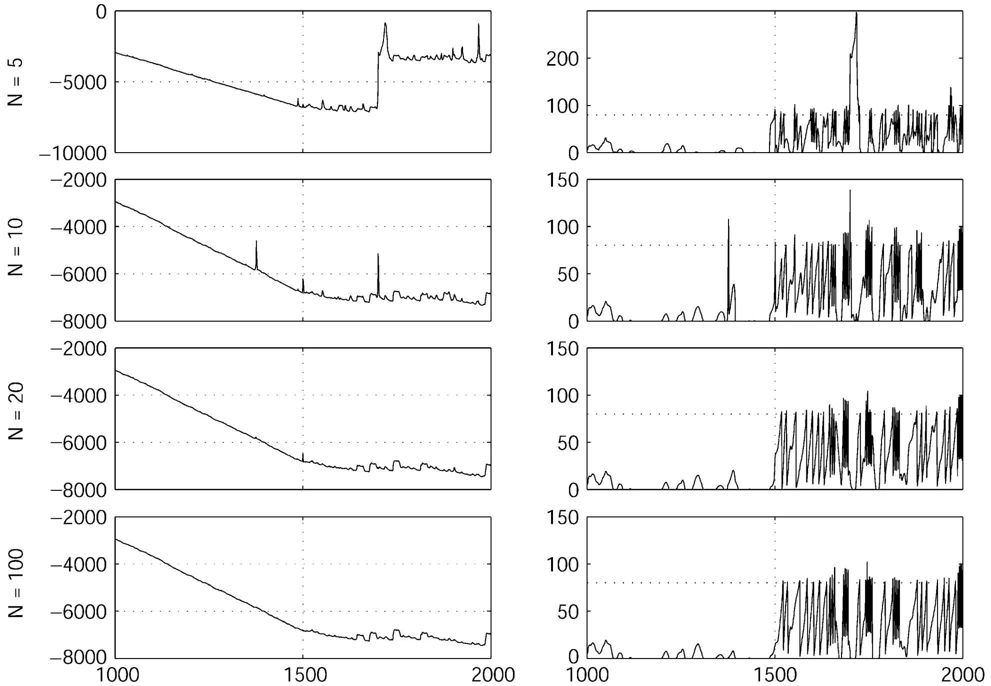

Fig. 6. Left: The cost function $C(t)$ calculated for the 1000 new observations with the slight change of the dynamical model at $t = 1500$ (see Fig. 5). The slope of the curve changes after the time instant of change. Right: The corresponding NDFA statistic.

observations (see Fig. 5) as it has been hidden by the nonlinear mixing model and the additive noise.

Let us apply the NDFA state estimation procedure to the described data. Fig. 6 presents the cost function calculated for the new 1000 measurements consisted of 500 points with the unchanged model $(1001 \leq t \leq 1500)$ and 500 points with the changed dynamical parameters $(1501 \leq t \leq 2000)$. One can easily see that the slope of the cost function curve decreases after the time instant of change, which is identified as a change in the process model.

The NDFA statistic (20) applied to the cost function in Fig. 6 is shown in the same figure. The following parameters were used here: $M = 20$ last points accounted in the statistic calculations, the drift parameter $\nu = 0$ and the threshold $h = 80$.

In spite of the fact that the NDFA method can readily detect the simulated change, the described change detection problem

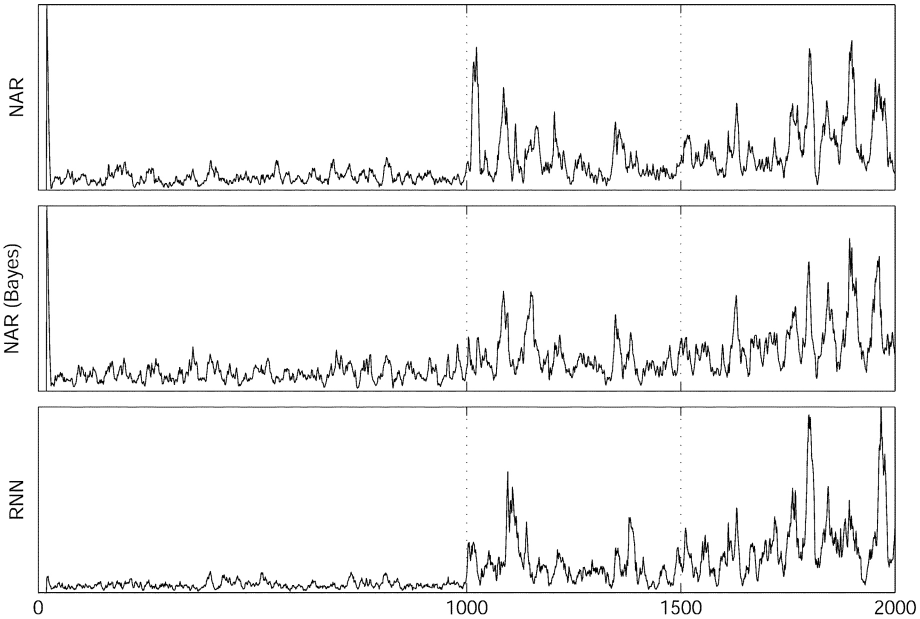

Fig. 7. The test statistics $g(t)$ of three alternative nonlinear methods calculated for the training data (first 1000 samples) and the 1000 new observations with the slight change of the dynamical model at $t = 1500$ (see Fig. 5).

turned out to be very hard for the alternative methods considered in the work such as the NAR and RNN-based approaches. Neither of these algorithms gave any significant rise of their test statistic after the time instant of change $t = 1500$, which means that they failed in this particular change detection problem. Fig. 7 shows the test statistics for NAR and RNN-based methods for training data and test data before and after the change. It is obvious that the prediction errors are significantly higher for the test data than for the training data. This seems to be partly due to overfitting and the Bayesian learning technique [27], [28], [11] alleviates the problem.

The model estimated by NDFA is much more accurate than the models estimated with the alternative approaches. Although the simulated changes are hardly visible to the eye directly from the observations, they are demonstrated well enough for the NDFA model to detect them. We believe that this is because NDFA has been able to find a state representation which makes the prediction task easier. As Fig. 8 shows, the states estimated by NDFA closely correspond to the states of the underlying process. The physically meaningful state representation leads to good prediction accuracy and allows one to analyze in which states the changes took place. Fig. 8 also presents the contributions $C_{\mathbf{s}}(t)$ of different states to the cost function (12). The plot shows that the estimated time series reproduce well the character of the original underlying processes. Only eight out of nine estimated states are actually used: Two states model the harmonic oscillator dynamics, three states describe the Lorenz process with constant parameters, and the other three states correspond to the Lorenz process with changing parameters. Notice the increasing cost contributions for the states with changing dynamics: analyzing the structure of the cost function helps in localizing detected changes.

D. Experiments With Oscillator Changes

In the following experiments, we test the universality of the NDFA method by simulating another type of changes affecting other features of the observed mixtures. For instance, eliminating the harmonic oscillator in the state space (see Fig. 9) changes the frequency content of the observed signals, which can be used to design a specific change detector monitoring these particular observation features. Such a taylored change detector can be compared with the generic NDFA.

Let us test how the power spectrum of the observations changes after eliminating the two sinusoidal signals. This will be done in a practical way leading to the change detector based on the frequency features we are going to estimate. Taking a sliding window of 64 last observations we perform the discrete cosine transform calculating the 64 components of the power spectrum. Summing the corresponding components of all the observed signals we get 64 time series comprised of the spectral components estimated at different time instants. Taking the mean of the 64 signals, we can see how the power spectrum changes after the oscillator change takes place (see Fig. 10).

Fig. 10 indicates that in spite of the high nonlinearity of the mixing function $\mathbf{f}$, the power spectrum of the original signal contains a large peak corresponding to the harmonic oscillator frequency. As expected, the peak disappears after eliminating the oscillator from the mixture and this can be used to construct a change detection algorithm.

Thus, the alternative change detector designed for this particular type of changes can perform as follows:

1) Take 64 last observations at each time instant and calculate the discrete cosine transform (DCT);
2) Summing the spectral components corresponding to the peak of the oscillator frequency (we take the seventh and eighth components that undergo the most significant changes), one obtains the test sequence $s(t)$;
3) Apply the standard one-sided CUSUM test like (17) for detecting decrease of the mean of the test sequence $s(t)$.

The described DCT-based algorithm was first tested on the single change detection problem and it gave a significant rise

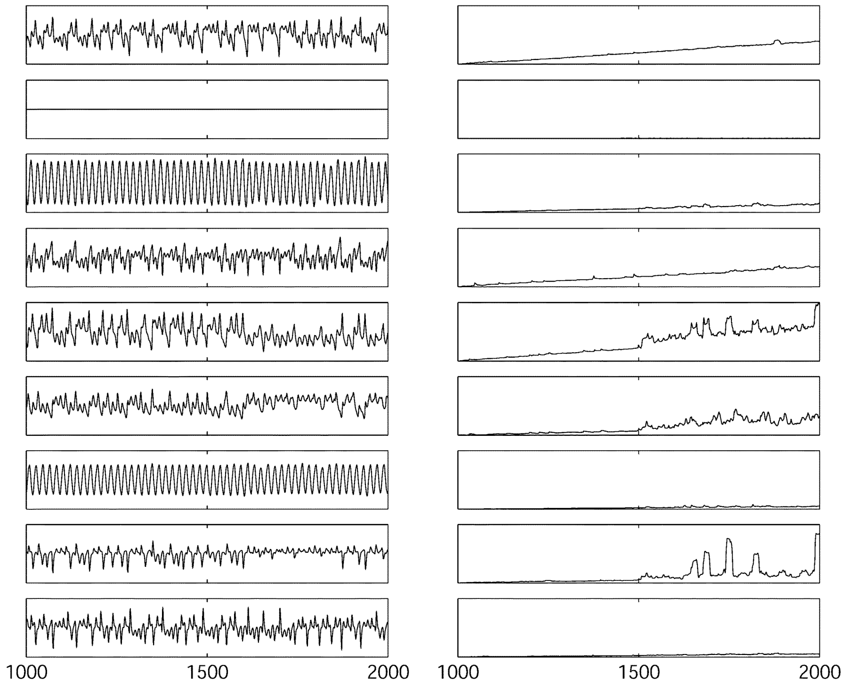

Fig. 8. The states estimated on-line by NDFA with $N = 100$ (left) and their contribution to the cost function (right) in the experiment with slight changes (see Fig. 5). The states which undergo changes increase their contribution.

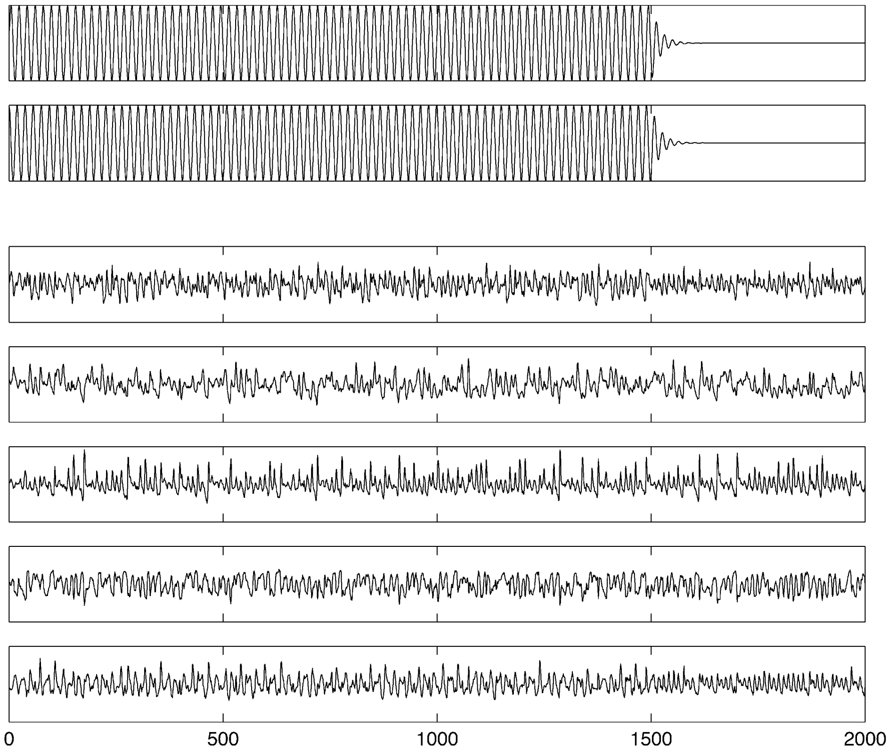

Fig. 9. The changes in the hidden oscillator dynamics at $t = 1500$ (two time series above) are not clearly visible in the observations (five time series below). Only five out of ten observations are presented here.

of its test statistic after the time instant of change. The NDFA method was applied to the same data and it also produced a short-term jump in the cost function after the change.

The test with repeatedly simulated changes assessing the performance of the two alternative methods was carried out as well. The results are presented in Fig. 11. One can easily see that the NDFA in general outperforms the DCT-approach but it again encounters the problems discussed in Section IV-B: Increasing the threshold h does not give very small values of the P_f-measure before the problem of missed detections is faced.

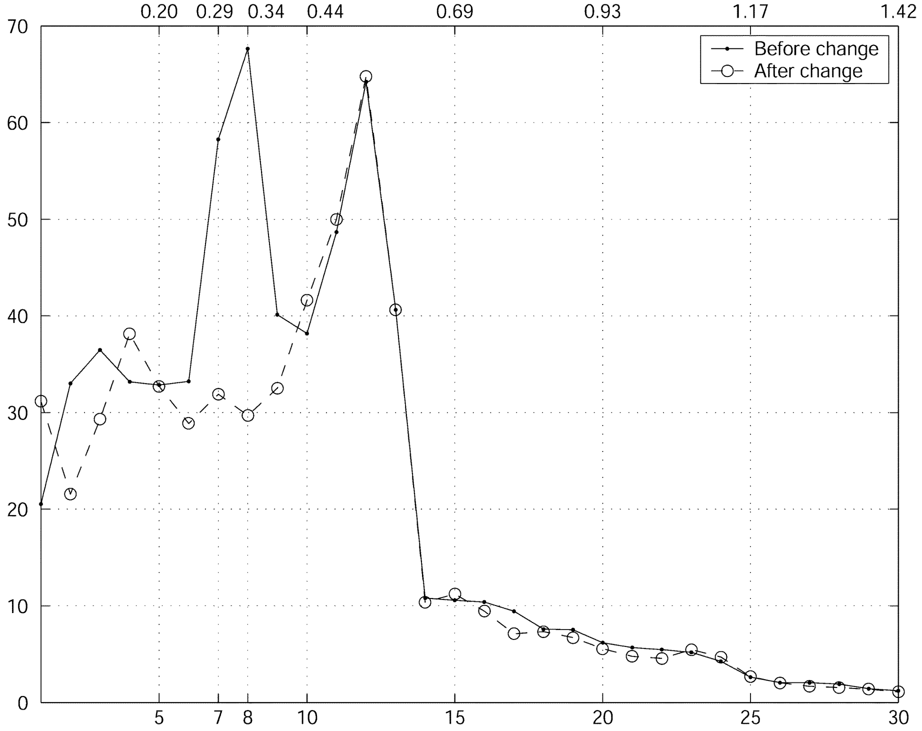

Fig. 10. The power spectrum of the observations explained in Fig. 9. The plot shows the mean of the first 30 (out of 64) time series obtained by calculating the discrete cosine transform on the sliding window of 64 last observations at each time instant. The corresponding angular velocities are shown at the top.

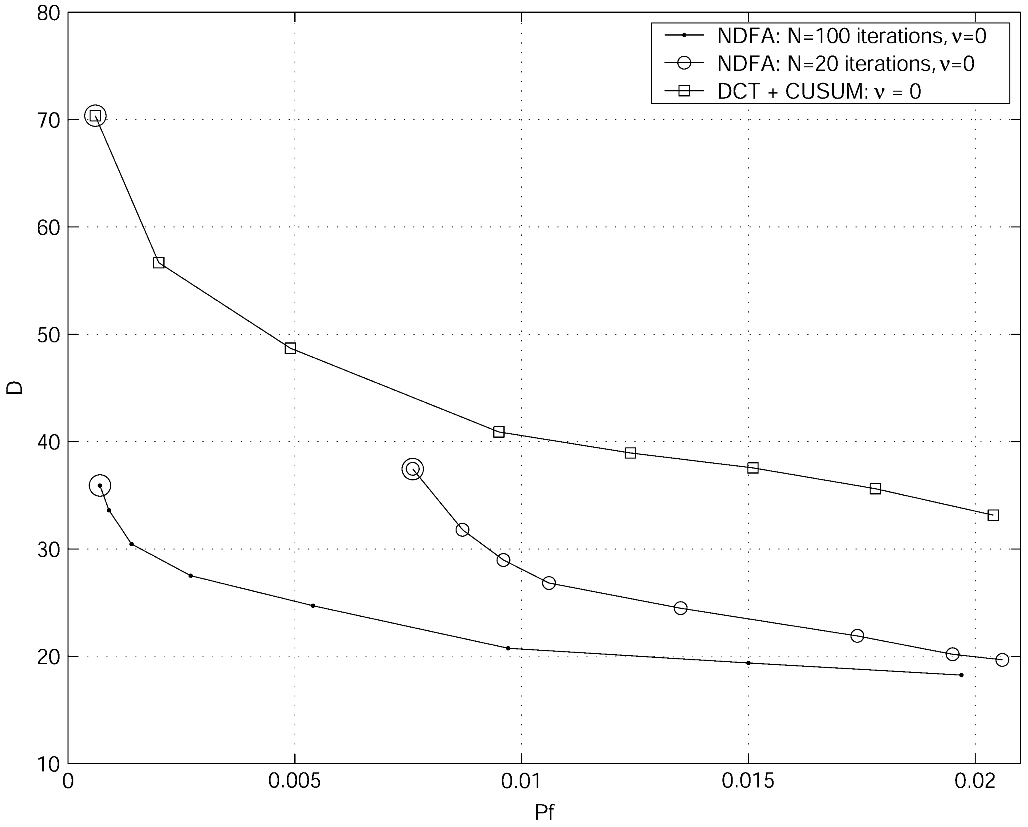

Fig. 11. Performance comparison of various change detection methods for the problem with the oscillator changes. The circled points correspond to the threshold values for which there were missed detection situations.

V. Discussion

We applied here the recently introduced nonlinear dynamical factor analysis algorithm for the problem of state change detection. The algorithm was reviewed and then extended to the change detection problem by introducing an on-line version, computing the cost of a fixed window of the most recent observations. When this cost is compared to the average model cost, departures from the learned model can be detected. We showed that with NDFA, change detection is considerably more accurate than with standard CUSUM and Shewhart charts. The main reason is that NDFA is building a full nonlinear state-space model. It can therefore detect a large variety of changes in a process, while most ad hoc methods have been developed for and are sensitive to a particular type of change only. Yet, NDFA seems to be able to outperform even such taylor-made algorithms. Another advantage of the model-based approach is that NDFA is able not only to pinpoint the time of the change, but also to show which of the underlying states were the reason for the change.

Model-based approaches using NAR and RNN were found to perform worse than the much simpler approaches which monitored the mean and variance of the observations. We argued that a model of dynamics can actually make detecting changes more difficult and only after the model is accurate enough, improvements over the simpler methods are possible. We used standard techniques from neural networks literature for learning the NAR and RNN models, including optimization of model structure and early stopping of learning based on prediction performance on a validation set.

The variational Bayesian learning used in NDFA can do without a validation set because it is very robust against overfitting and overlearning. Model structure can be optimized based on the cost function and there is no need for early stopping of learning. This property is shared by the Bayesian learning technique [27], [28], [11] which we tested for NAR. It suffered less from overfitting than NAR learned by standard backpropagation and yielded the best performance in change detection among the alternative methods we tested, but the performance was still far from that obtained by NDFA. This difference probably stems from the physically meaningful state representation which NDFA is able to find.

We assumed a completely unknown process without any prior information. The learning is fully unsupervised. In practice, the situation would not be as difficult. The method would readily allow to incorporate partial knowledge, as well as any external process inputs which were now omitted. Since many processes have nonstationary noise variances, it may also be useful to use the methods described in [40]. There, models of nonstationary variance that are estimated by variational learning were developed.

The main drawback in NDFA is the large number of iterations needed for off-line learning of the process model, although the actual change detection can then be done on-line. Finding ways to speed up the learning algorithm is an important next step. The building-blocks approach presented in [39] has smaller computational complexity and its application to learning NSSMs should help to reduce the learning time. An additional improvement was a different nonlinearity for hidden neurons which makes the cost function analytic, obviating the need for Taylor-series expansions. The NDFA method can in practice learn fairly large processes, up to 15-dimensional state spaces. Beyond that, learning may become unstable which seems to be due to the Taylor-series approximation. The building-blocks based approach does not suffer from this limitation. Since there exists a cost function which is guaranteed to decrease on each update, iterations are inherently stable in contrast to EKF and the present implementation of NDFA which resort to Taylor-series approximations.

A hierarchical nonlinear factor analysis model based on the building-blocks approach was presented in [41]. We are currently extending the model to include the dynamics of the states. Another speed-up which can be used in combination with both the NDFA learning algorithm and the hierarchical extension was presented in [21].

The proposed method has several potential applications. In addition to the simulated experiments reported here, essentially the same method has already been successfully applied by the authors to analysing magnetoencephalographic (MEG) signals measured from the human brain [32].

References

[1] S. Amari and H. Nagaoka, *Methods of Information Geometry*: American Mathematical Society, Providence, 2000.

[2] H. Attias, "ICA, graphical models and variational methods," in *Independent Component Analysis: Principles and Practice*, S. Roberts and R. Everson, Eds: Cambridge Univ. Press, 2001, pp. 95–112.

[3] B. Azimi-Sadjadi and P. S. Krishnaprasad, "Change detection for nonlinear systems; a particle filtering approach," in *Proc. Amer. Contr. Conf. (ACC2002)*, 2002.

[4] M. Basseville and I. Nikiforov, *Detection of Abrupt Changes: Theory and Application*, ser. Information and system science. Englewood Cliffs, NJ: Prentice-Hall, 1993.

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

[6] J. Chen and R. J. Patton, *Robust Model-Based Fault Diagnosis for Dynamic Systems*, MA: Kluwer Academic, 1999.

[7] L. H. Chiang, E. L. Russell, and R. D. Braatz, *Fault Detection and Diagnosis in Industrial Systems*, London: Springer-Verlag, 2001.

[8] R. A. Choudrey and S. J. Roberts, "Flexible Bayesian independent component analysis for blind source separation," in *Proc. Int. Conf. Independent Component Analysis and Signal Separation (ICA2001)*, San Diego, CA, 2001, pp. 90–95.

[9] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[10] G. Dorffner, "Neural networks for time series processing," *Neural Network World*, vol. 6, no. 4, pp. 447–468, 1996.

[11] F. D. Foresee and M. T. Hagan, "Gauss-Newton approximation to Bayesian regularization," in *Proc. 1997 Int. Joint Conf. Neural Networks*, 1997, pp. 1930–1935.

[12] E. A. García and P. M. Frank, "Deterministic nonlinear observer-based approaches to fault diagnosis: A survey," *Contr. Eng. Practice*, vol. 5, no. 5, pp. 663–670, 1997.

[13] Z. Ghahramani and S. Roweis, "Learning nonlinear dynamical systems using an EM algorithm," in *Advances in Neural Information Processing Systems 11*, M. Kearns, S. Solla, and D. Cohn, Eds. Cambridge, MA: The MIT Press, 1999, pp. 431–437.

[14] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice*, ser. Information and system science. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[15] F. Gustafsson, *Adaptive Filtering and Change Detection*: Wiley, 2000.

[16] S. Haykin, *Neural Networks—A Comprehensive Foundation*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[17] ——, *Adaptive Filter Theory*, 4th ed. Englewood Cliffs, NJ: Prentice-Hall, 2001.

[18] S. Haykin and J. Principe, "Making sense of a complex world," *IEEE Signal Processing Mag.*, vol. 15, pp. 66–81, May 1998.

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

[20] P. Højen-Sørensen, L. K. Hansen, and O. Winther, "Mean field implementation of Bayesian ICA," in *Proc. Int. Conf. Independent Component Analysis and Signal Separation (ICA2001)*, San Diego, CA, 2001, pp. 439–444.

[21] A. Honkela, H. Valpola, and J. Karhunen, "Accelerating cyclic update algorithms for parameter estimation by pattern searches," *Neural Processing Lett.*, vol. 17, no. 2, pp. 191–203, 2003.

[22] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. New York: Wiley, 2001.

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

[24] A. Iline, H. Valpola, and E. Oja, "Detecting process state changes by nonlinear blind source separation," in *Proc. 3rd Int. Workshop on Independent Component Analysis and Signal Separation (ICA2001)*, San Diego, CA, Dec. 2001, pp. 704–709.

[25] H. Lappalainen and A. Honkela, "Bayesian nonlinear independent component analysis by multi-layer perceptrons," in *Advances in Independent Component Analysis*, M. Girolami, Ed, Berlin: Springer-Verlag, 2000, pp. 93–121.

[26] H. Lappalainen and J. Miskin, "Ensemble learning," in *Advances in Independent Component Analysis*, M. Girolami, Ed, Berlin: Springer-Verlag, 2000, pp. 75–92.

[27] D. J. C. MacKay, "Bayesian interpolation," *Neural Comput.*, vol. 4, no. 3, pp. 415–447, 1992.

[28] ——, "A practical Bayesian framework for backpropagation networks," *Neural Comput.*, vol. 4, no. 3, pp. 448–472, 1992.

[29] J. Miskin and D. J. C. MacKay, "Ensemble learning for blind source separation," in *Independent Component Analysis: Principles and Practice*, S. Roberts and R. Everson, Eds: Cambridge Univ. Press, 2001, pp. 209–233.

[30] R. J. Patton, J. Chen, and T. M. Siew, "Fault diagnosis in nonlinear dynamic systems via neural networks," in *Proc. IEE Int. Conf. Contr. '94*, Conf. Pub. 389, Warwick, U.K., 1994, pp. 1346–1351.

[31] S. Roweis and Z. Ghahramani, "An EM algorithm for identification of nonlinear dynamical systems," in *Kalman Filtering and Neural Networks*, S. Haykin, Ed. New York: Wiley, 2001, pp. 175–220.

[32] J. Särelä, H. Valpola, R. Vigário, and E. Oja, "Dynamical factor analysis of rhythmic magnetoencephalographic activity," in *Proc. Int. Conf. Independent Component Analysis and Signal Separation (ICA2001)*, San Diego, CA, 2001, pp. 451–456.

[33] J. Segen and A. C. Sanderson, "Detecting change in a time-series," *IEEE Trans. Inform. Theory*, vol. IT-26, pp. 249–255, 1980.

[34] H. Valpola, *Unsupervised Learning of Nonlinear Dynamic State-Space Models*, Espoo, Finland: Publications in Computer and Information Science A59, Lab. Computer and Information Science, Helsinki Univ. Technol., 2000.

[35] H. Valpola, A. Honkela, and X. Giannakopoulos. (2002) *Matlab Codes for the NFA and NDFA Algorithms* [Online]. Available: http://www.cis.hut.fi/projects/bayes/

[36] H. Valpola, A. Honkela, and J. Karhunen, "Nonlinear static and dynamic blind source separation using ensemble learning," in *Proc. Int. Joint Conf. Neural Networks (IJCNN'01)*, Washington, D.C., 2001, pp. 2750–2755.

[37] H. Valpola and J. Karhunen, "An unsupervised ensemble learning method for nonlinear dynamic state-space models," *Neural Comput.*, vol. 14, no. 11, pp. 2647–2692, 2002.

[38] H. Valpola, E. Oja, A. Ilin, A. Honkela, and J. Karhunen, "Nonlinear blind source separation by variational Bayesian learning," *IEICE Trans. Fund. Electron., Commun. Comput. Sci.*, vol. E86-A, no. 3, Mar. 2003.

[39] H. Valpola, T. Raiko, and J. Karhunen. Building blocks for hierarchical latent variable models. presented at Proc. 3rd Int. Workshop on Independent Component Analysis and Signal Separation (ICA2001). [Online]. Available: http://www.cis.hut.fi/harri/

[40] H. Valpola, M. Harva, and J. Karhunen, "Hierarchical models of variance sources," *Signal Process.*, vol. 84, no. 2, pp. 267–282, 2004.

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

[42] P. van Overschee and B. de Moor, *Subspace Identification for Linear Systems*. Boston, MA: Kluwer Academic, 1996.

[43] E. Wan and A. Nelson, "Dual extended Kalman filter methods," in *Kalman Filtering and Neural Networks*, S. Haykin, Ed. New York: Wiley, 2001, pp. 123–173.

[44] E. A. Wan and R. van der Merwe, "The unscented Kalman filter," in *Kalman Filtering and Neural Networks*, S. Haykin, Ed. New York: Wiley, 2001, pp. 221–280.

[45] J. Zhang, E. B. Martin, and A. J. Morris, "Process monitoring using nonlinear statistical techniques," *Chem. Eng. J.*, vol. 67, no. 3, pp. 181–189, 1997.

Alexander N. Ilin was born in Saint Petersburg, Russia, in 1977. He received the B.S. and M.S. degrees with honors in computer science and engineering in 1998 and 2000, respectively, from the Saint Petersburg State Polytechnical University. While an undergraduate, he received the Russian Government Scholarship, the Grant of the Gagarin Foundation, and a medal at the All-Russian student work competition.

He joined the Neural Network Research Centre, Helsinki University of Technology, Finland, in 2001. He is currently working toward the Ph.D. degree by developing Bayesian methods for nonlinear process modeling.

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

He has been with the Neural Networks Research Centre, Helsinki University of Technology, since 1993. Currently he is working on a humanoid-robot project with the Artificial Intelligence Laboratory, University of Zurich.

Erkki Oja (S'75–M'78–SM'90–F'00) received the Dr. Sc. degree, in 1977.

He is currently the Director of the Neural Networks Research Centre and Professor of computer science, the Laboratory of Computer and Information Science, Helsinki University of Technology, Finland. Previously, he was a Research Associate at Brown University, Providence, RI, and Visiting Professor at the Tokyo Institute of Technology. He is a member of the editorial boards of several journals and has been in the program committees of several recent conferences including ICANN. IJCNN, and ICONIP. His research interests are in the study of principal component and independent component analysis, self-organization, statistical pattern recognition, and applying artificial neural networks to computer vision and signal processing. He is the author or coauthor of more than 260 articles and book chapters on pattern recognition, computer vision, and neural computing. He is also the author of three books: "Subspace Methods of Pattern Recognition" (New York: Wiley, 1983), which has been translated into Chinese and Japanese, "Kohonen Map" (New York: Elsevier, 1999), and "Independent Component Analysis" (New York: Wiley, 2001).

Dr. Oja is member of the Finnish Academy of Sciences, Founding Fellow of the International Association of Pattern Recognition (IAPR), and President of the European Neural Network Society (ENNS).