# Statistical mechanics of EKF learning in neural networks

J. Phys. A: Math. Gen. **32** (1999) 1605–1621. Printed in the UK PII: S0305-4470(99)96046-8

# Statistical mechanics of EKF learning in neural networks

Bernhard Schottky† and David Saad‡

Neural Computing Research Group, Aston University, Birmingham B4 7ET, UK

Received 17 July 1998, in final form 30 November 1998

**Abstract.** We formulate a learning algorithm for online learning in neural networks using the extended Kalman filter approach, providing a principled and practicable approximation to the full Bayesian treatment. The latter, which constitutes optimal learning, does not require artificial setting of training parameters and allows for the estimation of a wide range of quantities of interest. We analyse the performance of the algorithm using tools of statistical physics in several scenarios: we look at drifting rules represented by linear and nonlinear perceptrons and investigate how different prior settings affect the generalization performance as well as learnability itself. We investigate the learning behaviour of stationary two-layer network, where the algorithm seems to avoid the, otherwise common, problem of long symmetric plateaus.

## 1. Introduction

Online learning is an important learning paradigm in the context of neural networks, particularly for nonstationary tasks. A continuous stream of training examples is used for adapting sequentially a set of parameters, gradually improving the approximation to the underlying rule realized by the system. This rule is often referred to as the ‘teacher’, whereas the approximating system is termed ‘student’.

A widely used approach for regression problems is to define an objective measure for the discrepancy between the desired and the actual output produced by the current estimate of the rule. The parameters of the student are then updated by gradient descent on this error measure; the update mechanism is controlled by learning parameters in general and the learning rate $\eta$ in particular.

One problem of this ‘ad hoc’ approach is the arbitrary choice of the learning parameters and learning rules. For example, the optimal learning rate schedule, depends on the characteristics of the system [1, 2] and is generally not known, so that heuristic estimates have to be used. For instance, in a noisy but learnable stationary scenario, one has to decay the learning rate to zero asymptotically, inversely to the number of training examples with a specific prefactor [1]. If, however, the underlying rule is drifting, the learning rate has to stay finite, keeping track of the changing rule.

Similarly, several advanced and principled training rules have been suggested over the years (e.g., Newton’s method [3] and natural gradient descent [4]) but there is no clear understanding as to what is the best method to use in different cases, especially when taking into account the different computational costs involved.

† E-mail address: Bernhard.Schottky@sdm.de

‡ E-mail address: saadd@aston.ac.uk

Lots of work has been done, using methods of statistical mechanics (for an overview on theoretical methods in online learning see [5]), on determining locally [6] and globally [2] optimal learning rate schedule and learning rules [8, 7] and on analysing the properties of practicable second-order methods [4, 9]. However, these methods are mainly of theoretical relevance as they require quantities which are unknown during the learning progress.

Another deficiency of current online training methods is that they represent a single evolution path and they may therefore critically depend on the choice of initial conditions and will not be able to provide information on statistical properties of the evolving solution, such as error bars and parameter relevance.

These deficiencies may be handled by the Bayesian approach, which offers a principled method for following the evolving posterior with respect to the training examples presented, with no need for artificially choosing learning parameters and rules. Moreover, it constitutes the optimal learning procedure, if the prior is set correctly, and facilitates the calculation of a wide range of quantities of interest, such as the posterior mean, error-bar estimation parameter relevance and more.

Unfortunately, obtaining exact analytical expressions for Bayesian online learning is unfeasible, and even obtaining approximated expressions via numerical methods such as Markov chain Monte Carlo [10] is usually impractical. A principled and practicable alternative is the application of the extended Kalman filter (EKF) to online learning in neural networks. The EKF has been used to speed up batch learning in [11] and has been introduced in the online learning of neural nets in [12] and independently in [13] and [14]. The method has some of the advantages of a Bayesian method: scheduling the learning rate is not required after the initial setting, it is self-controlled and some statistical properties of the solution may be calculated. However, as the EKF merely approximates the posterior we cannot expect it to perform optimally, as one would expect from the exact Bayesian solution, and one should therefore carefully assess the impact of the approximations on the performance.

We should also point out that online Bayesian approaches have been presented for classification in [15] and were subsequently studied within the statistical mechanics framework in [16]. This approach also relies on a Gaussian approximation to the posterior but does not require the use of EKF techniques.

The scope of this paper is to present the algorithm derived from the EKF approach and to analyse the learning behaviour in several scenarios using methods of statistical physics. We will compare its performance to that of non-Bayesian approaches and discuss the effects of the approximation used.

The algorithm is applicable and efficient in the case of smooth networks. However, it is of larger time and space complexity in comparison to conventional approaches ($\mathcal{O}(N^2)$ instead of $\mathcal{O}(N)$). We will also formulate and examine a simplified version using isotropic posterior distributions (and quasi-isotropic in complicated cases), which brings the algorithm's complexity back to $\mathcal{O}(N)$; this ansatz can be justified asymptotically. The simplified version will be analysed exactly while the algorithm with the more general posterior will be studied by numerical simulations.

The paper is organized as follows. Section 2 outlines the EKF approach which is applied in section 3 to nonstationary feed-forward neural networks. We introduce the explicit update equations for the cases of linear and nonlinear perceptron and of the soft committee machine (SCM) [17] as well as a more economical version of the algorithm using quasi-isotropic posteriors. In section 4 we present the statistical mechanics framework for analysing dynamical properties of the algorithm and in section 5 we evaluate its performance in three cases: in the case of the linear drifting perceptron we look at the influence of mismatched prior choices. In the nonlinear drifting perceptron case we investigate, in particular, how the nonlinearity influences the performance and finally we examine the SCM via numerical simulations.

Before moving on to describing the EKF, we would like to explain our notation, as it comprises conventions used in both neural networks and EKF literature. We need to distinguish between scalars, column vectors and matrices. For scalars we use lower case letters (e.g. $h$) except for $N$ and $M$ which are special scalars denoting the input dimensionality and the number of hidden nodes respectively. Column vectors are denoted by bold lower case letters (e.g. $\boldsymbol{\xi}$). Finally, matrices are denoted by upper case roman letters (e.g. C). Where necessary we state the dimensionality of quantities explicitly.

We will denote the discrete time index $t$ in two different ways: either as subscript (as in $\boldsymbol{\xi}_t$) or in parentheses (as in $\boldsymbol{w}(t)$). These notations should be considered identical; the former will be used most of the time for brevity while the latter will be used for main dynamical variables that will be converted later to variables which depend on a continuous time.

## 2. The extended Kalman filter approach

As the first step we sketch the general EKF approach. This is meant only as a short summary, details and explicit derivations of the formulae can be found in the literature, e.g. in [18, 19]. The relevance of the EKF approach to nonstationary neural networks training will be explained in section 3.

The typical task in the EKF approach [18, 19] is to keep track of a vector $\boldsymbol{w}^0$ which is evolving in time due to the general dynamics

$$\boldsymbol{w}^0(t+1) = \boldsymbol{v}(\boldsymbol{w}^0(t), \boldsymbol{\rho}_t). \tag{1}$$

The dimension of $\boldsymbol{w}^0$ (and thus of $\boldsymbol{v}$) is $N$ which also generally is an characterizes the system size. The function $\boldsymbol{v}$ is known whereas the vector $\boldsymbol{\rho}_t$ (of fixed dimensionality depending on the model) representing some noise is unknown. This means that the dynamics of $\boldsymbol{w}^0$ has both deterministic and a stochastic components.

At each time step we get some information about the state of the system given by the measurement equation

$$z_t = h_t(\boldsymbol{w}^0(t), \boldsymbol{\zeta}_t). \tag{2}$$

Here $h_t$ is time-dependent and known (or assumed to be known) whereas the measurement noise vector $\boldsymbol{\zeta}_t$ (of some fixed dimension, depending on the model) is unknown.

The idea is now to represent our knowledge (or belief) about the vector $\boldsymbol{w}^0(t)$ by a probability distribution and update this distribution at each time step in a Bayesian manner.

To model this probability distribution we will use a unimodal Gaussian posterior distribution with mean $\hat{\boldsymbol{w}}(t)$ and covariance matrix $\mathrm{C}(t)$. The distribution updates can be separated into two steps. Given the estimates $\hat{\boldsymbol{w}}(t), \mathrm{C}(t)$ at time $t$ we first have to take into account the movement of the rule vector $\boldsymbol{w}^0(t)$ by defining $\hat{\boldsymbol{w}}^-(t+1)$ and $\mathrm{C}^-(t+1)$ to be our estimates at time $t+1$ *before* receiving the measurement. The second step is then to incorporate the information obtained from the measurement to get $\hat{\boldsymbol{w}}(t+1)$ and $\mathrm{C}(t+1)$ as the distribution parameters at time $t+1$ *after* incorporating information obtained from the measurement.

It is now important to notice that under certain conditions this program can be performed exactly. For this it is sufficient that the functions $\boldsymbol{v}$ and $h_t$ are linear in the weights and the noise variables (the components of the noise vectors $\boldsymbol{\rho}$ and $\boldsymbol{\zeta}$, respectively) and that the distribution of this noise variable is a (multivariate) Gaussian. Starting at $t = 0$ with a unimodal Gaussian distribution as the weights prior probability distribution, the exact update of this distribution leads to a modified unimodal Gaussian again. This corresponds to the Kalman filter estimator.

In the following we will assume the noise variables to be Gaussian and represent the probability distribution of the weights by a multivariate unimodal Gaussian. The functions $\boldsymbol{v}$

and $h$ we are interested in are, however, in general, nonlinear. The trick is now to linearize these functions and perform updates using the linearized versions. The framework is then called the *extended* Kalman filter and is only an approximation. The quality of this approximation under certain conditions will be addressed later on in this paper.

Let us now derive the program outlined so far explicitly. First we linearize $\boldsymbol{v}$ about the current vector $\boldsymbol{w}$ and the noise variable $\rho$

$$\boldsymbol{v}(\boldsymbol{w} + \Delta\boldsymbol{w}, \boldsymbol{\rho}) \approx \boldsymbol{v}(\boldsymbol{w}, 0) + \mathrm{V}_t\Delta\boldsymbol{w} + \mathrm{P}\rho \tag{3}$$

with

$$\begin{aligned} \mathrm{V}_t &= \nabla_{\boldsymbol{w}}\boldsymbol{v}|_{w=\hat{w}(t)} \\ \mathrm{P} &= \nabla_{\rho}\boldsymbol{v}|_{\rho=\mathbf{0}}. \end{aligned} \tag{4}$$

Here, $\mathrm{V}_t$ and P are $N \times N$ and $N \times$ [the dimension of $\boldsymbol{\rho}$] matrices. The noise variables in $\boldsymbol{\rho}$ are assumed to be Gaussian with zero mean (as nonzero average could be absorbed into the deterministic part of the dynamics) and covariance

$$(\Sigma_\rho)_{kl} = \langle\rho_k\rho_l\rangle. \tag{5}$$

The so-called 'time update' due to the evolution of the vector $\boldsymbol{w}^0$ is then given by

$$\begin{aligned} \hat{\boldsymbol{w}}^-(t+1) &= \boldsymbol{v}(\hat{\boldsymbol{w}}(t), \mathbf{0}) \\ \mathrm{C}^-(t+1) &= \mathrm{V}_t\mathrm{C}(t)\mathrm{V}_t^T + \mathrm{P}\Sigma_\rho\mathrm{P}^T. \end{aligned} \tag{6}$$

Note that the linearization is carried out at the mean value $\hat{\boldsymbol{w}}(t)$ of our current posterior.

The next step is to incorporate the information provided by the measurement. As outlined before we have to linearize $h$:

$$h_t(\boldsymbol{w} + \Delta\boldsymbol{w}, \boldsymbol{\zeta}) \approx h_t(\boldsymbol{w}, \mathbf{0}) + \mathrm{H}_t\Delta\boldsymbol{w} + \mathrm{Z}\boldsymbol{\zeta} \tag{7}$$

with

$$\begin{aligned} \mathrm{H}_t &= \nabla_{\boldsymbol{x}}h_t|_{x=\hat{w}(t)} \\ \mathrm{Z} &= \nabla_\zeta h_t|_{\zeta=\mathbf{0}} \end{aligned} \tag{8}$$

and the noise variance

$$(\Sigma_\zeta)_{kl} = \langle\zeta_k\zeta_l\rangle. \tag{9}$$

The dimension of $\mathrm{H}_t$ is $1 \times N$ (thus a row vector) and of Z is $1\times$ (the dimension of $\boldsymbol{\zeta}$). With the linearization and the assumption of Gaussian noise we get a Gaussian likelihood term due to the new example. This is incorporated with the Gaussian prior, based on our current estimate of the posterior parametrized by $\boldsymbol{w}^-(t+1), \mathrm{C}^-(t+1)$, through multiplication and subsequent normalization of the two Gaussian distributions. As all distributions are Gaussian, this straightforward procedure results in a Gaussian distribution with new parameters, corresponding to the 'measurement update' in the EKF literature:

$$\begin{aligned} \hat{\boldsymbol{w}}(t+1) &= \hat{\boldsymbol{w}}^-(t+1) + \mathrm{K}_{t+1}[z_{t+1} - h_{t+1}(\hat{\boldsymbol{w}}^-(t+1))] \\ \mathrm{C}(t+1) &= (I - \mathrm{K}_{t+1}\mathrm{H}_{t+1})\mathrm{C}^-(t+1) \end{aligned} \tag{10}$$

where the Kalman gain is defined by

$$\mathrm{K}_{t+1} = \mathrm{C}^-(t+1)\ \mathrm{H}_{t+1}^T[\mathrm{H}_{t+1}\mathrm{C}^-(t+1)\mathrm{H}_{t+1}^T + \mathrm{Z}\Sigma_\rho\mathrm{Z}^T]^{-1}. \tag{11}$$

The dimension of K is $N \times 1$. Combining the 'time update' (6) and the 'measurement update' (10) provides the complete update equations from time $t$ to time $t+1$.

For a more detailed derivation of the EKF equations we refer the interested reader to, e.g., [18, 19].

## 3. Application to neural networks

### *3.1. The model*

The task is to utilize the general framework introduced in the previous section for online learning of an evolving neural network. Although it is certainly not immediately obvious how to carry this through, it turns out to be straightforward as only reinterpretation of some quantities is necessary.

In the learning scenario considered here we have an underlying rule (teacher) specified by an unknown parameter vector $\boldsymbol{w}^0$, which is to be learned by our model (student). Depending on the values of these parameters, a known function $f$, realized by a neural network, maps an $N$-dimensional input pattern $\boldsymbol{\xi}_t \in \mathbb{R}^N$ given at time $t$ to a scalar output value $z_t \in \mathbb{R}$

$$z_t = f(\boldsymbol{w}^0, \boldsymbol{\xi}_t) + \zeta_t \tag{12}$$

where $\zeta_t$ is an additive noise drawn from a Gaussian distribution with zero mean and variance $\sigma_T^2$ representing the corruption process. This is termed a measurement in the EKF literature; in online learning, a single pair $(\boldsymbol{\xi}_t, z_t)$ of pattern and corrupted teacher response, given at each time step $t$, constitutes a training example, and is used to improve our estimation of the teacher couplings.

In a nonstationary scenario one also has to take into account that the teacher couplings $\boldsymbol{w}^0$ are time-dependent with

$$\boldsymbol{w}^0(t+1) = \boldsymbol{v}(\boldsymbol{w}^0(t), \boldsymbol{\rho}_t) \tag{13}$$

where $\boldsymbol{v}$ is some function which is assumed to be known and $\boldsymbol{\rho}_t$ represents a set of random variables which drive the nondeterministic part of the evolution.

This scenario is often called a student–teacher scenario where the model (student) is trained on basis of the given examples to be as close as possible to the underlying rule (teacher). It clearly corresponds to the EKF scenario outlined earlier by comparing equation (12) with equation (2) and equation (13) with equation (1). We see that these scenarios can be mapped onto one another if we

- identify the EKF parameter vector $\boldsymbol{w}^0$ (section 2) with the teacher vector $\boldsymbol{w}^0$ of section 3.
- identify the function $h_t$ in equation (2) with the function $f$ (with the argument $\boldsymbol{\xi}_t$ at time $t$) in equation (12).
- identify the EKF nonstationarity function $\boldsymbol{v}$ in equation (13) with the teacher-nonstationarity function $\boldsymbol{v}$ in equation (1).
- use for estimating of the student $\boldsymbol{w}$ a Gaussian distribution specified by a mean $\hat{\boldsymbol{w}}(t)$ and a variance $\mathrm{C}(t)$ as in the EKF approach.

Having done this, the EKF-based learning algorithm can be applied to our problem directly: we just have to specify our choices for $\boldsymbol{v}$, $h_t$ (given by $f$ and $\boldsymbol{\xi}_t$) and the initial conditions for $\hat{\boldsymbol{w}}$ and C. The only restriction is that $\boldsymbol{v}$ and $f$ are smooth functions. We then get update equations for the online learning scenario directly from equations (6) and (10).

As the weight dynamics for the teacher we will use throughout this paper a random drift with a constant teacher vector length. So we choose the teacher vector $\boldsymbol{w}^0$ to be normalized to one, $|\boldsymbol{w}^0| = 1$ and the nonstationarity to be as in [21]

$$\boldsymbol{w}^0(t+1) \cdot \boldsymbol{w}^0(t) = 1 - \frac{\delta_T}{N} \tag{14}$$

where the coefficient $\delta_T$ controls the drift 'speed'.

The main quantity of interest is the Bayesian generalization error given by

$$\epsilon_g(t) = \langle (f(\boldsymbol{w}^0, \boldsymbol{\xi}) - \langle f(\boldsymbol{w}, \boldsymbol{\xi})\rangle_{p_t(\boldsymbol{w})})^2 \rangle_\xi \tag{15}$$

with $p_t(\boldsymbol{w})$ being the current estimate of the posterior. The outer average is over the set of all possible patterns $\boldsymbol{\xi}$ sampled from some probability distribution, which should be specified for calculating quantities of interest (e.g., within the statistical mechanics framework).

We have used the same function $f$ for both teacher and student, reflecting the fact that we consider them to have the same architecture. It is possible to use different functions, $f_T$ and $f_S$, to reflect the fact that we do not know the teacher architecture $f_T$ but assume it to be described by $f_S$. This enables one to investigate unrealizable and over-realizable scenarios. In this paper we will restrict ourselves to realizable scenarios.

### 3.2. Update equations for a single node

We first derive the update equations for the nonlinear perceptron

$$f(\boldsymbol{w}, \boldsymbol{\xi}) = \phi(\boldsymbol{w} \cdot \boldsymbol{\xi}) \tag{16}$$

where the activation function $\phi$ will be specified later. As described in the previous section one obtains update equations for each new example $(\boldsymbol{\xi}_{t+1}, z_{t+1})$—equations (10). Replacing the general functions in equations (10) and (6) by the specific network (16) and teacher nonstationarity (14) considered here one obtains

$$\hat{\boldsymbol{w}}(t+1) = \hat{\boldsymbol{w}}(t) + \frac{\phi'(z_{t+1} - \phi)}{\sigma_S^2 + (\phi')^2 \boldsymbol{\xi}_{t+1}^T \mathrm{C}(t) \boldsymbol{\xi}_{t+1}} \mathrm{C}(t) \boldsymbol{\xi}_{t+1} \tag{17}$$

$$\mathrm{C}(t+1) = \mathrm{C}(t) - \frac{(\phi')^2 \mathrm{C}(t) \boldsymbol{\xi}_{t+1} \boldsymbol{\xi}_{t+1}^T \mathrm{C}(t)}{\sigma_S^2 + (\phi')^2 \boldsymbol{\xi}_{t+1}^T \mathrm{C}(t) \boldsymbol{\xi}_{t+1}} + \frac{2\delta_S}{N^2} I \tag{18}$$

where $\phi$ and $\phi'$ denote $\phi(\hat{\boldsymbol{w}}(t) \cdot \boldsymbol{\xi}_{t+1})$ and its derivative with respect to its argument, respectively. We have introduced two new variables: $\sigma_S$ and $\delta_S$. Ideally these should be chosen to correspond to the true noise variances, i.e. $\sigma_S = \sigma_T$ and $\delta_S = \delta_T$. However, if the true noise variances are not known (as is generally the case) one has to use estimated values and for updating the posterior.

The matrix C in equations (17) and (18) serves, in conjunction with the noise-dependent denominator, as an effective non-isotropic learning rate (in the eigen system of C one would have different values for the different eigen directions). This reflects the fact that we might be pretty certain about the correct value of the weight vector in some directions (narrow posterior) whereas other directions are still fairly undetermined allowing for larger learning steps in these directions.

The scaling in this scenario is $\mathcal{O}(\hat{w}_i) \sim 1/\sqrt{N}$, $\mathcal{O}(\mathrm{C}_{ij}) \sim 1/N$ and $\mathcal{O}(\xi_i) \sim 1$; keeping the correct scaling is significant as we will take the limit $N \to \infty$ later on.

### 3.3. Update for the soft committee machine

In this architecture the coupling vector $\boldsymbol{w}$ is fragmented into $M$ parts $\boldsymbol{w}^l$, $l = 1 \ldots M$, with the definition of the rule being

$$f(\boldsymbol{w}, \boldsymbol{\xi}) = \sum_{l=1}^{M} \phi(\boldsymbol{w}_l \cdot \boldsymbol{\xi}). \tag{19}$$

The posterior covariance is

$$\mathrm{C} = \begin{pmatrix} \mathrm{C}_{11} & \mathrm{C}_{12} & \mathrm{C}_{13} & \ldots \\ \mathrm{C}_{21} & \mathrm{C}_{22} & \ldots & \ldots \\ \mathrm{C}_{31} & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots \end{pmatrix} \tag{20}$$

with $\mathrm{C}_{kl}$ being the $N \times N$ covariance matrix for $\boldsymbol{w}_l$ and $\boldsymbol{w}_k$. The update rules become:

$$\hat{\boldsymbol{w}}_l(t+1) = \hat{\boldsymbol{w}}_l(t) + \mathrm{K}_l(t+1)[z_{t+1} - f(\{\hat{\boldsymbol{w}}_l(t)\}, \boldsymbol{\xi}_{t+1})] \tag{21}$$

$$\mathrm{C}_{lk}(t+1) = \mathrm{C}_{lk}(t) + \delta_{lk}\frac{1}{N^2}2\delta_S I - \frac{1}{\sigma_S^2 + \sigma_c^2}\sum_m \left\{\left[\phi_m' \sum_n \phi_n' \mathrm{C}_{ln}(t)(\boldsymbol{\xi}_{t+1}\boldsymbol{\xi}_{t+1}^T)\right]\mathrm{C}_{mk}(t)\right\} \tag{22}$$

with the Kalman gain

$$\mathrm{K}_l(t+1) = \frac{1}{\sigma_S^2 + \sigma_c^2}\sum_n \phi_n' \mathrm{C}_{ln}(t)\boldsymbol{\xi}_{t+1} \tag{23}$$

the variance

$$\sigma_c^2 = \sum_{kn} \phi_k' \phi_n' \boldsymbol{\xi}_{t+1}^T \mathrm{C}_{kn}(t)\boldsymbol{\xi}_{t+1} \tag{24}$$

and using the abbreviation

$$\phi_n' = \phi'(\hat{\boldsymbol{w}}_n(t+1) \cdot \boldsymbol{\xi}_{t+1}). \tag{25}$$

### *3.4. Restricted posterior and the isotropic covariance*

The algorithm as formulated so far requires the general covariance matrix for calculating the posterior and is therefore of considerable higher time and space complexity to that of conventional online learning algorithms like gradient descent. One simplification is to restrict the space of possible posterior distributions, projecting the actual posterior onto a restricted space after each update using some suitable distance measure.

We will use here the most simple form for the restricted posterior: an isotropic Gaussian with the following covariance for the one-node case:

$$\mathrm{C}(t) = \frac{\eta(t)}{N} I. \tag{26}$$

Here $I$ is the $N \times N$-identity matrix. For the SCM the corresponding ansatz is

$$\mathrm{C}_{kl} = \frac{\eta_{kl}}{N} I \tag{27}$$

so each segment of the complete covariance (see equation (20)) is restricted to be isotropic.

There are several reasons for taking this simplified ansatz: the first is that the complexity is reduced from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$ with $N$ being the number of free parameters. On top of that, if the system is noisy (which is always the case in practice) this ansatz is well justified asymptotically as the weight error decays with training time with a prefactor proportional to the Fisher information matrix. Thus, in the cases considered here, the covariance matrix structure reduces asymptotically to a (quasi) isotropic structure for stationary tasks.

A technical reason for looking at the simplified version is that it is amenable to theoretical analysis. For the general covariance matrix only the linear stationary case have been solved so far where batch (e.g., in [20]) and the proposed online scheme are equivalent.

There are many ways to redefine the update rule for this type of restricted posterior. We choose a straightforward approach: we employ the ansätze (26), (27) in the general update equations (equations (17) and (18) for the one-node case and equations (21) and (22) for the SCM). For obtaining consistent updates we have to replace the term $\boldsymbol{\xi}\boldsymbol{\xi}^T$ in equations (18) and (22) by an appropriate isotropic matrix, which is carried out by replacing the actual $\boldsymbol{\xi}\boldsymbol{\xi}^T$ by its average $\langle \boldsymbol{\xi}\boldsymbol{\xi}^T \rangle_\xi$ (meaning that we treat this matrix as self-averaging, which is clearly an approximation). For preprocessed input data, and in particular for the scenario examined later on, the general matrix C will then reduce to a constant $\eta$ (multiplied by the identity matrix $I$) for which the update equations can be obtained straightforwardly.

## 4. The order parameter dynamics

In the statistical mechanics approach we are interested in the system's behaviour for large system size $N \to \infty$. In this so called 'thermodynamic limit' one can capture the system's behaviour by a small set of macroscopic quantities [17, 22] which are sufficient for calculating the main quantities of interest. These are, for the one-node case in our model, the vector overlaps $Q = \hat{\boldsymbol{w}} \cdot \hat{\boldsymbol{w}}$, $R = \hat{\boldsymbol{w}} \cdot \boldsymbol{w}^0$ and $\eta$.

Moreover the order parameters evolve deterministically due to their self-averaging properties in the thermodynamic limit [23]. The corresponding equations are easily obtained by constructing the order parameter updates on the basis of the update equations for $\hat{\boldsymbol{w}}$ and C, applying the limit $N \to \infty$ and averaging over the the pattern and noise distributions.

We will assume a pattern distribution (from which the training patterns are drawn) where each site $\xi_i$ is chosen randomly from $\mathcal{N}(0, 1)$ (a Gaussian with zero mean and unit variance). The noise variable is, as specified earlier, a Gaussian variable with variance $\sigma_T^2$ and zero mean.

By then introducing the continuous time $\alpha = t/N$ the evolution of the order parameters is given by

$$\frac{\mathrm{d}Q}{\mathrm{d}\alpha} = 2\eta \left\langle \hat{\boldsymbol{w}} \cdot \boldsymbol{\xi} \frac{\Delta\phi'}{\sigma_S^2 + \eta(\phi')^2} \right\rangle + \eta^2 \left\langle \frac{\Delta^2(\phi')^2}{(\sigma_S^2 + \eta(\phi')^2)^2} \right\rangle \tag{28}$$

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \eta \left\langle \boldsymbol{w}^0 \cdot \boldsymbol{\xi} \frac{\Delta\phi'}{\sigma_S^2 + \eta(\phi')^2} \right\rangle - \delta_T R \tag{29}$$

$$\frac{\mathrm{d}\eta}{\mathrm{d}\alpha} = -\eta^2 \left\langle \frac{(\phi')^2}{\sigma_S^2 + \eta(\phi')^2} \right\rangle + 2\delta_S \tag{30}$$

where $\Delta$ is the difference between the student's (with weight vector $\hat{\boldsymbol{w}}$) and the noise-corrupted teacher's response, $\Delta = \phi(\boldsymbol{w}^0 \cdot \boldsymbol{\xi}) + \zeta - \phi(\hat{\boldsymbol{w}} \cdot \boldsymbol{\xi})$; the derivative $\phi'$ is to be taken at $\hat{\boldsymbol{w}} \cdot \boldsymbol{\xi}$.

The average $\langle \cdots \rangle$ is with respect to the pattern and noise distribution. The joint distribution of $\boldsymbol{w}^0 \cdot \boldsymbol{\xi}$ and $\hat{\boldsymbol{w}} \cdot \boldsymbol{\xi}$ is entirely determined by their covariance represented by the parameters $Q$ and $R$ (being a two-dimensional Gaussian), so that the above equations represent a closed system which can be solved numerically given some initial conditions.

For the SCM we introduce the order parameters $Q_{kl} = \hat{\boldsymbol{w}}_k \cdot \hat{\boldsymbol{w}}_l$ and $R_{kl} = \hat{\boldsymbol{w}}_k \cdot \boldsymbol{w}_l^0$; it is straightforward to obtain the following system of ordinary differential equations

$$\frac{\mathrm{d}\eta_{kl}}{\mathrm{d}\alpha} = 2\delta_{kl}\delta_S - \left\langle \frac{\sum_{nm} \phi'_n \phi'_m \eta_{kn} \eta_{ml}}{\sigma_S^2 + \sum_{nm} \phi'_n \phi'_m \eta_{nm}} \right\rangle \tag{31}$$

$$\frac{\mathrm{d}R_{kl}}{\mathrm{d}\alpha} = \left\langle \boldsymbol{w}_l^0 \cdot \boldsymbol{\xi} \frac{\Delta \sum_n \phi'_n \eta_{kn}}{\sigma_S^2 + \sum_{nm} \phi'_n \phi'_m \eta_{nm}} \right\rangle - \delta_T R_{kl} \tag{32}$$

$$\begin{aligned} \frac{\mathrm{d}Q_{kl}}{\mathrm{d}\alpha} = {} & \left\langle \hat{\boldsymbol{w}}_k \cdot \boldsymbol{\xi} \frac{\Delta \sum_n \phi'_n \eta_{kn}}{\sigma_S^2 + \sum_{nm} \phi'_n \phi'_m \eta_{nm}} \right\rangle + \left\langle \hat{\boldsymbol{w}}_l \cdot \boldsymbol{\xi} \frac{\Delta \sum_n \phi'_n \eta_{ln}}{\sigma_S^2 + \sum_{nm} \phi'_n \phi'_m \eta_{nm}} \right\rangle \\ & + \left\langle \frac{\Delta^2 (\sum_n \phi'_n \eta_{kn})(\sum_n \phi'_n \eta_{ln})}{(\sigma_S^2 + \sum_{nm} \phi'_n \phi'_m \eta_{nm})^2} \right\rangle \end{aligned} \tag{33}$$

where $\Delta$, also here, is the difference between the noise-corrupted teacher response and the student response.

## 5. Numerical results

In this section we employ the statistical mechanics framework for the single node case and for the SCM (equations (28)–(30) and (31)–(33), respectively) to study analytically and

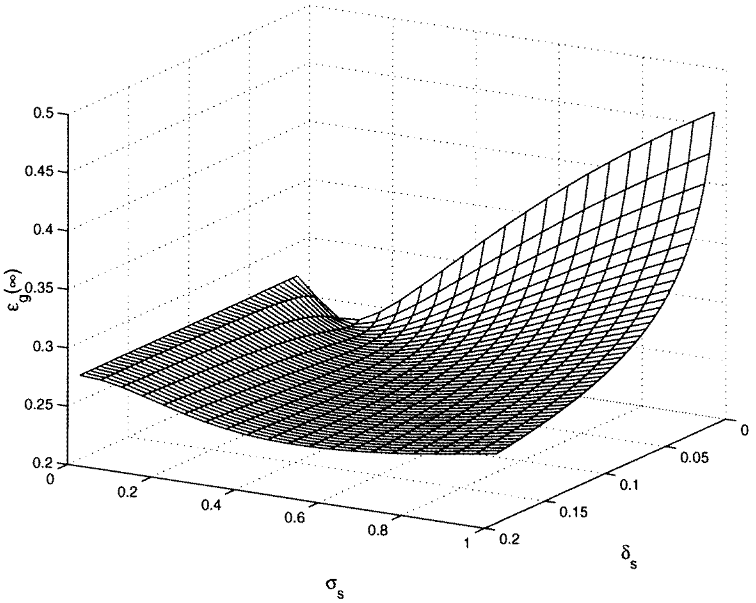

**Figure 1.** For fixed $\sigma_T = 0.3$ and $\delta_T = 0.1$ the dependency of the asymptotically achieved generalization error on $\sigma_S$ and $\delta_S$ is shown. The combination of large $\sigma_S$ and small $\delta_S$ proves most disadvantageous. On the other hand there is a wide range of choices that lead to reasonable asymptotic values.

numerically the performance of the EKF approach in several scenarios with emphasis on different aspects.

### 5.1. The drifting linear rule

The linear case can be analysed exactly; here we assess the algorithm's asymptotic performance where sub-optimal prior parameters are selected: more specifically we investigate qualitatively the influence of the parameter choices for $\sigma_T$, $\delta_T$, $\sigma_S$, $\delta_S$.

Using $\bar{Q}$, $\bar{R}$ and $\bar{\eta}$ for the asymptotic values we obtain

$$\bar{\eta} = \delta_S + \sqrt{\delta_S^2 + 2\delta_S\sigma_S^2} \tag{34}$$

and

$$\begin{pmatrix} \bar{Q} \\ \bar{R} \end{pmatrix} = -\bar{\eta} \begin{pmatrix} \frac{\bar{\eta}}{\sigma_S^2+\bar{\eta}} - 2 & 2 - \frac{2\bar{\eta}}{\sigma_S^2+\bar{\eta}} \\ 0 & -\bar{\eta}\frac{1}{\sigma_S^2+\bar{\eta}} - \delta_T \end{pmatrix}^{-1} \begin{pmatrix} \frac{\sigma_T^2}{\sigma_S^2+\bar{\eta}} \\ \frac{\sigma_T}{\sigma_S^2+\bar{\eta}} \end{pmatrix}. \tag{35}$$

The generalization error is given by $\epsilon_g = 1 + \bar{Q} - 2\bar{R}$. Figure 1 shows the asymptotic generalization error dependence on the choice of $\sigma_S$ and $\delta_S$ for the specific choice $\sigma_T = 0.3$, $\delta_T = 0.1$. One can identify areas which are rather insensitive to the parameter choice as well as areas where poor parameter assignments lead to bad generalization. The latter can be easily identified as areas where the model undervalues the drift rate and overestimates the noise variance. The student performs then much too small updates and is lagging behind the actual state of the drifting teacher.

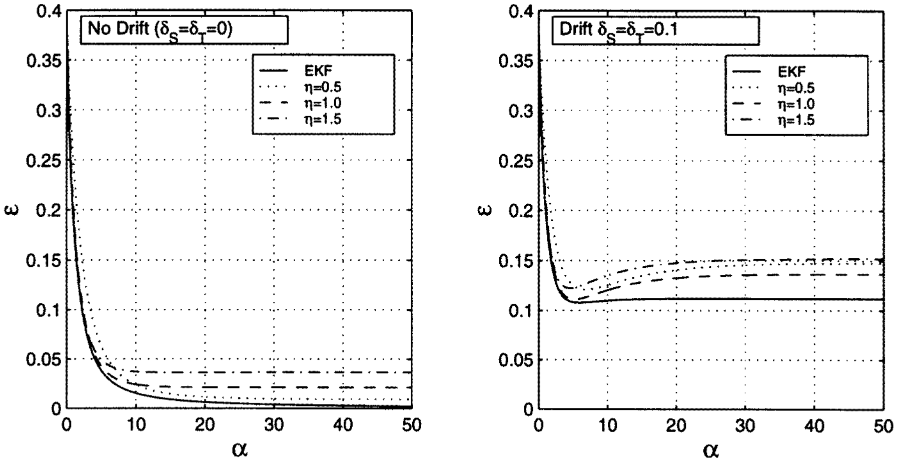

**Figure 2.** The generalization error for EKF- and gradient descent learning (for several values of the learning rate $\eta$) is compared for stationary (left) and nonstationary tasks. The Bayesian approach shows superior performance while determining the learning rate schedule automatically.

### *5.2. The nonlinear drifting perceptron*

As the EKF approach is based on linearizing the dynamics about the current estimates, its performance may depend on the accumulated errors due to linearization. These will presumably depend on the system's nonlinearity and the drift speed of the underlying rule. Both aspects will be examined in the current example and compared with the performance of gradient descent learning.

We focus on the case were $\sigma_T$ and $\delta_T$ are known (setting $\sigma_S = \sigma_T$ and $\delta_S = \delta_T$) fixing the noise rates to $\sigma_S = \sigma_T = 0.3$. As the activation function we introduce $\phi(x) = \mathrm{erf}(ax/\sqrt{2})$, where the parameter $a$ controls the system's nonlinearity.

In figure 2 we compare EKF- and gradient descent learning for three fixed learning rates $\eta$, setting the nonlinearity parameter $a = 1$. We see that for a stationary task (left figure) the learning rate has to be small for good asymptotic results (it actually has to be annealed to zero as $\alpha \to \infty$), which deteriorates the performance at the beginning of the learning process. Optimal results may be obtained by imposing an explicit and non-trivial learning rate schedule [2]. For learning the drifting rule there is an optimal asymptotic nonzero learning rate which is, however, not known. In contrast, EKF learning yields superior results and the choice of the effective learning rate is done automatically.

We now turn to the adequacy of EKF in the case of nonlinear and drifting rules. As mentioned earlier, the update equations use a linearization around the actual mean, an approximation which becomes more inaccurate as the nonlinearity increases and as the posterior distribution tails become more significant. Therefore, cases with high nonlinearity and large drifting speed would lead to a bad performance of the algorithm. Varying the nonlinearity ($a$) and drift ($\delta_T$) parameters allows one to investigate these effects.

Figure 3 shows the learning curves for several values of the nonlinearity parameter values $a = 1, 2, 3$ for stationary (left) and nonstationary tasks. The theoretical results show, in the region investigated, that for drifting rules the asymptotic performance deteriorates with increasing nonlinearity; for the case $a = 3$ the generalization error even diverges (simulations we carried out confirm the theoretical results). This means that there is a transition to a non-converging phase, depending on the specific system parameters, where the EKF algorithm fails

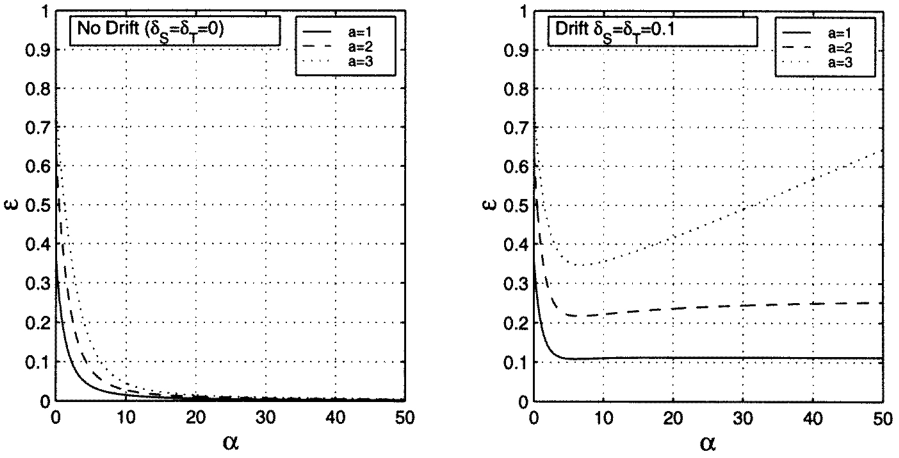

**Figure 3.** The Bayesian generalization error for stationary (left) and drifting tasks (right). Whereas there is always convergence for $\delta_S = \delta_T = 0$ this is not the case for drifting concepts: when the nonlinearity is sufficiently large (here $a = 3$) the system diverges.

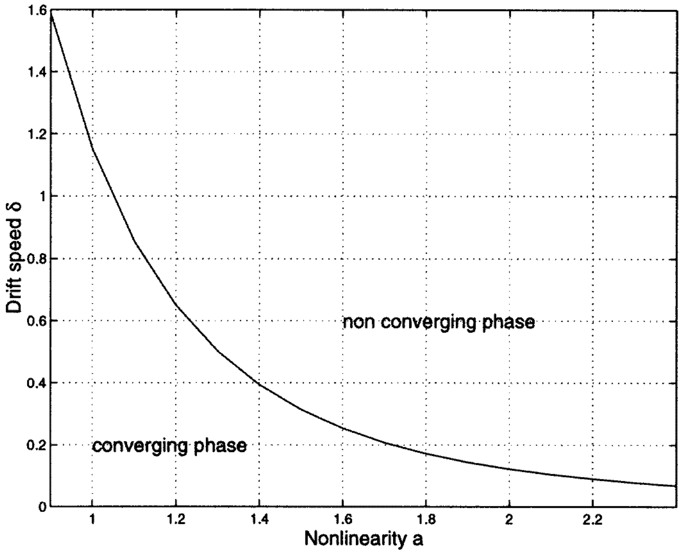

**Figure 4.** For a fixed noise rate $\sigma_S = \sigma_T = 0.3$ there are two phases in the $a - \delta$ plane: For large $a$ and $\delta$ the system does not converge to a stationary solution, so the task is unlearnable via the EKF-algorithm. This is quantified by the phase border shown in the graph.

completely. In figure 4 we depict the phase diagram showing regions in the $a$–$\delta$ plane where stationary solutions (however bad) are reached (the noise rates are here $\sigma_T = \sigma_S = 0.3$ as mentioned above).

### *5.3. Results for a two-node SCM*

We now turn to the question how the algorithm works for more complicated networks. The SCM is a model often looked at in this context.

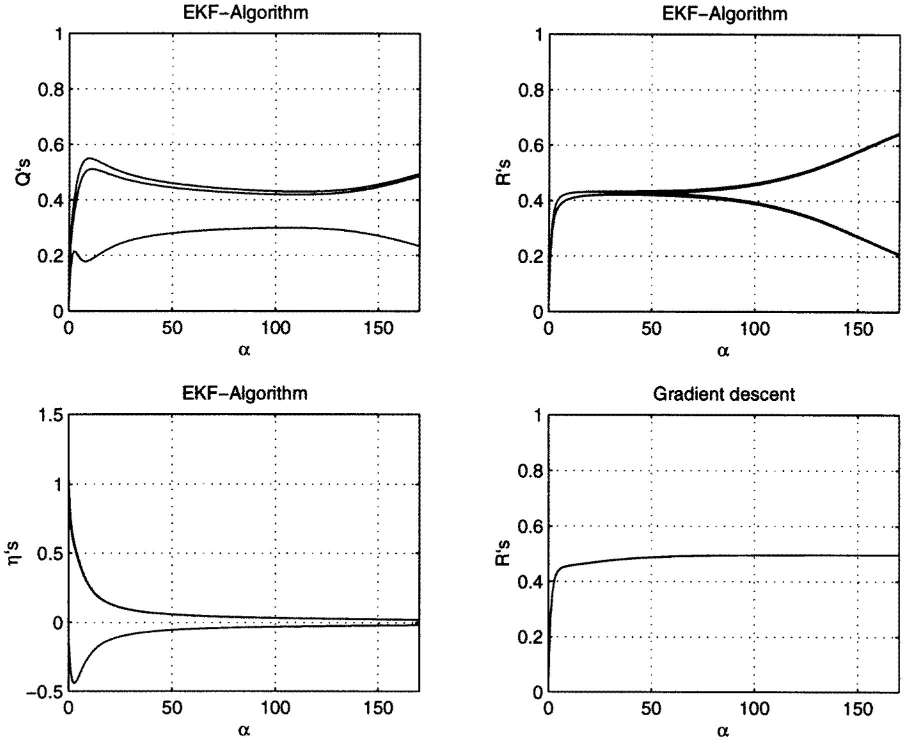

**Figure 5.** We compare EKF and gradient descent learning in the case of a two-node SCM, focusing on the symmetric phase.

One characteristic effect in such learning machines with inherent symmetries (the role of the two sub-perceptrons can be swapped) is the occurrence of a symmetric phase [22]: the student perceptrons do not specialize on the different teacher couplings acquiring a similar symmetric state. This results in a prolonged phase (showing as a plateau in the evolution of the generalization error) where only slow learning progress is made until the system escape the unstable fixed point. There are methods to shorten the plateau: by local [6] optimization of the learning rule (where, however, unknown quantities are referred to), using natural gradient descent [4, 8, 9] or just by a heuristic change of the objective function [24].

We will first assess the performance of the quasi-isotropic approach. Figure 5 shows the evolution of the order parameter for the $M = 2$ case when the quasi-isotropic covariance is used, focusing on the symmetric phase and the onset of specialization. The system is stationary ($\delta_S = \delta_T = 0$) with noise rates $\sigma_S = \sigma_T = 0.3$ and the teacher perceptrons are orthogonal and normalized, so $T_{lk} = \boldsymbol{w}_l^0 \cdot \boldsymbol{w}_k^0 = \delta_{lk}$. The symmetry-breaking happens quite early, in comparison with the evolution of the $R$ for simple gradient descent where the system is trapped in the symmetric phase until $\alpha \approx 200$.

However, although the algorithm manages to break the symmetric phase quite early it fails in another way: the $\eta$ collapse much too fast, giving an unreasonably narrow posterior which slows down the asymptotic convergence.

We also investigated the benefit of using a general covariance matrix via numerical simulations. Figure 6 shows the evolution of the order parameter for the same case as in figure 5, this time using the general covariance matrix. Surprisingly, there is hardly any symmetric

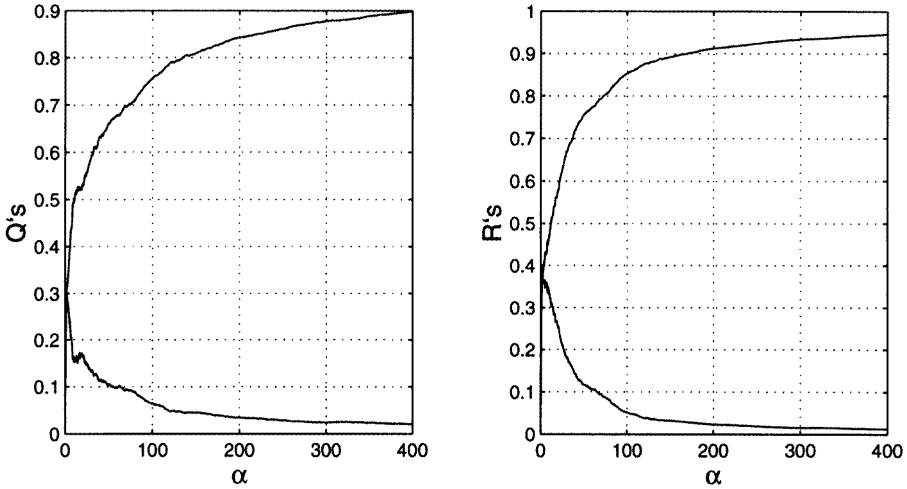

**Figure 6.** Order parameter evolution for the two-node SCM and a general covariance matrix. The curves are averages over five runs and different curves of same type (e.g. $R_{11}$, $R_{22}$) are averaged over.

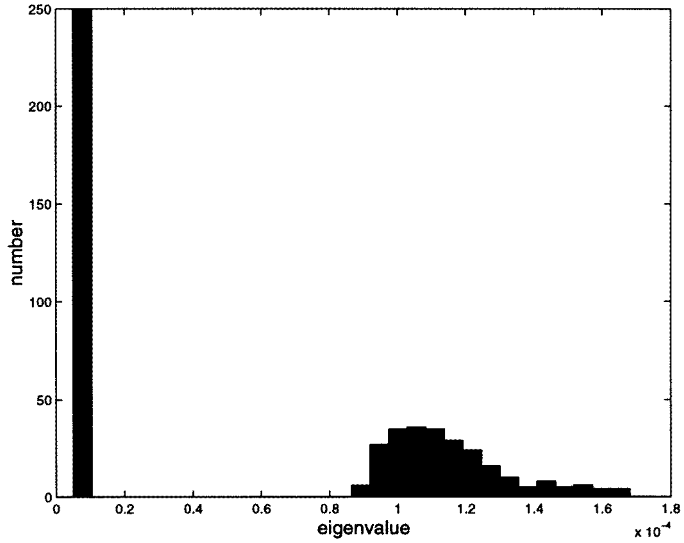

**Figure 7.** Eigenvalue spectrum for learning the SCM ($a = 1$) with the general covariance matrix.

phase; in the first stage of the learning process the solution heads towards a symmetric fixed point which is then immediately escaped and specialization begins.

The results in figure 6 were obtained using an input size of $N = 50$. However, simulations with $N = 100$ and $N = 200$ did not change the picture significantly, indicating that this behaviour, of an extremely short symmetric plateau, is not due to finite size effects but is probably genuine. However, by decreasing the activation function nonlinearity one can see the emergence of a symmetric phase, as for a linear mapping there is always an optimal symmetric student (both sub-perceptrons are equal).

To study the properties of the approximated posterior we show in figure 7 a histogram

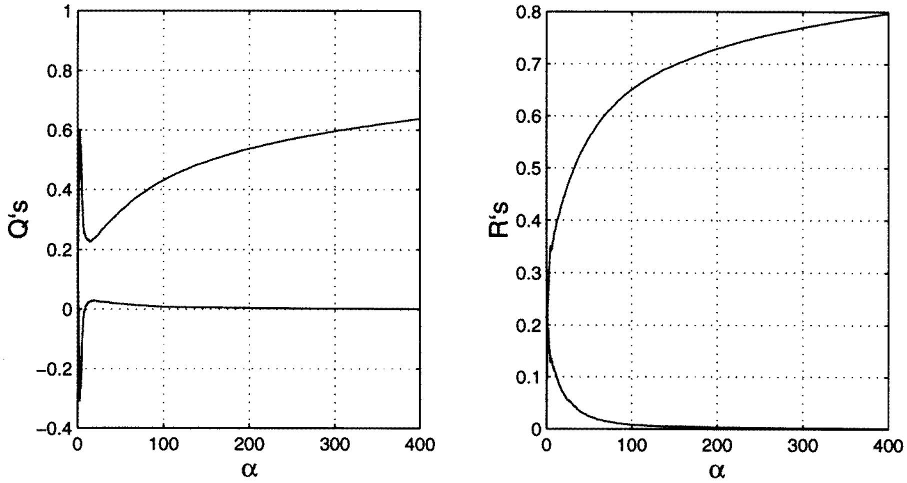

**Figure 8.** Order parameter evolution for the two-node SCM with increased nonlinearity $a = 3$.

of the covariance matrix eigenvalues at $\alpha = 400$ obtained from five separate runs (over 500 values). The covariance matrix structure mirrors that of the quasi-isotropic ansatz (27), with two different $\eta$, $\eta_{11} = \eta_{22} = \eta^s$ and $\eta_{12} = \eta_{21} = \eta^a$, which give rise to two eigenvalue types, $\eta^s - \eta^a$ (bigger) and $\eta^s + \eta^a$. Both groups are smeared out slightly due to the stochasticity of the examples, which is less emphasized in the smaller eigenvalue group due to their small absolute value.

The typical size of $\eta_{11}$ observed is about $6 \times 10^{-5}$ giving a weight uncertainty of $\pm 0.055/\sqrt{N}$ which is consistent with the numerical results obtained at $\alpha = 400$. However, in the case of high nonlinearity the algorithm still suffers from a fast-shrinking posterior distribution, similar to that observed in the isotropic algorithm. In figures 8 and 9 we show the order parameter evolution and the covariance matrix eigenvalues for the case of $a = 3$; the weight uncertainty here is $\pm 0.016/\sqrt{N}$, which is far too narrow compared with the overlap reached.

An intuitive explanation for the failure to capture the uncertainty in this case is due to the approximation used in the EKF approach. The linearization process, on which the method is based, fails to incorporate all the information provided by new examples, and the student parameter improvement is smaller than it could have been without the approximation. This is, however, not registered by the posterior, which narrows down due to the expected (higher) improvement. The stronger the nonlinearity, the stronger the effect. For the general covariance matrix the effect occurs later than for the isotropic algorithm; both are affected by this problem, which is exacerbated by the isotropic ansatz.

In order to compensate for this behaviour one can increase the noise rate $\sigma_S$ of the student. Results are shown in figures 10 and 11 where $\sigma_S = 5$ is chosen. The narrowing down is prevented in the expense of some slowing down at the beginning of the process. So there is a price to be paid for the approximation made within the EKF approach, and not surprisingly things become more problematic with increasing nonlinearities.

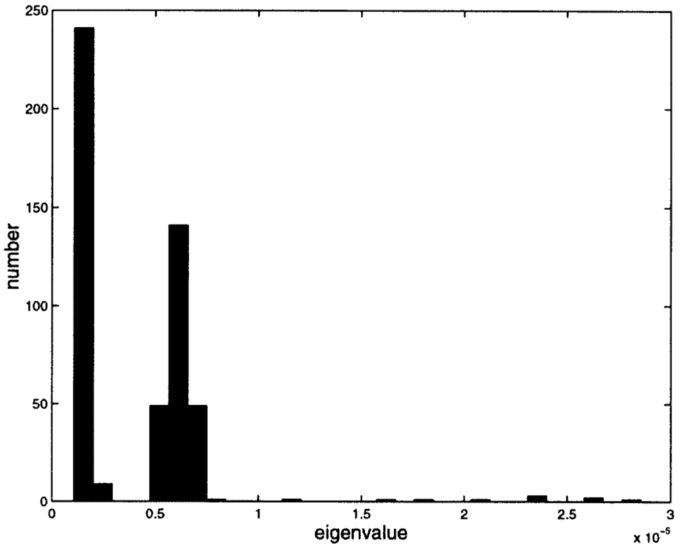

**Figure 9.** Eigenvalue spectrum for learning the SCM with increased nonlinearity $a = 3$.

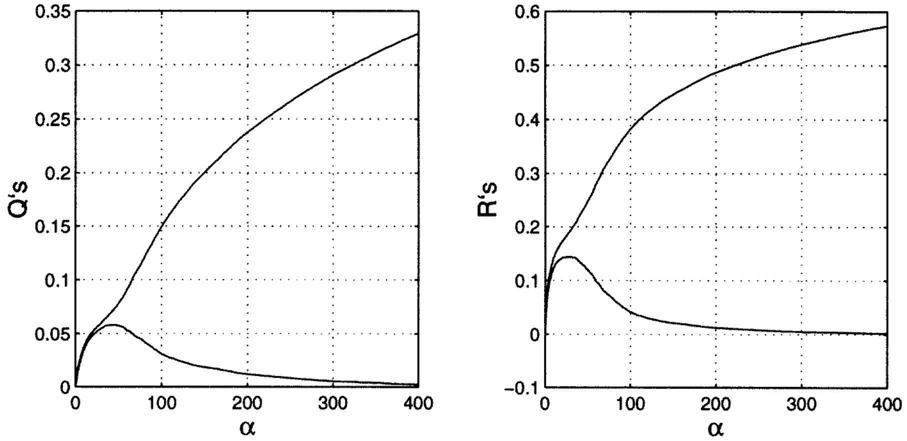

**Figure 10.** We have increased the student noise rate from $\sigma_S = 0.3$ (figure 8) to $\sigma_S = 5$ to compensate for the unwanted narrowing down of the posterior. The teacher noise rate is in both cases $\sigma_T = 0.3$.

## 6. Conclusions

We have presented an EKF-based Bayesian learning scheme for neural networks learning regression tasks. This principled approach provides a cheap alternative to the full Bayesian treatment which is practicable and efficient but still provides some of the main benefits of the Bayesian scheme. In addition, this algorithm avoids the problem of choosing training parameters like the learning rate, adapted here automatically, as should be done heuristically in other learning schemes. Methods from statistical mechanics allow us to analyse the proposed algorithm and to obtain exact learning curves.

We analysed the performance of the algorithm in several scenarios. Looking at the drifting linear perceptron we investigated how mismatch between the priors and the true parameters

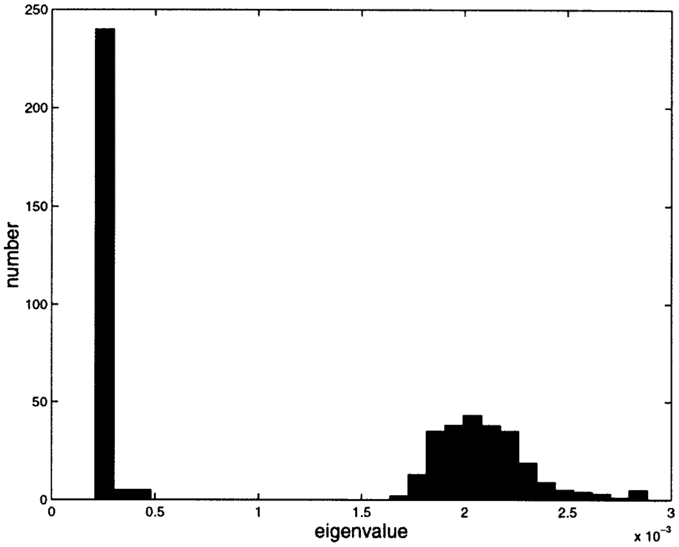

**Figure 11.** The width of the posterior as seen from the eigenvalues is now within reasonable bounds in the expense of slowing down the learning progress due to the noise rate overestimation.

affects the performance. A reasonable guess of the drift rate and noise variance usually leads to good performance while, unavoidably, low (assumed) drift speed and a high student noise variance lead to poor performance as the model associates the drift with noise.

In the case of drifting nonlinear perceptron we investigated the dependence of the performance on the nonlinearity. We have found a phase transition between learnable and unlearnable problems depending on drift speed and nonlinearity of the rule to be learned. This shows analytically the (already known) limitation of the algorithm.

Finally, we found that the symmetric plateaus, which may dominate the SCM training process, can be almost avoided in the EKF approach if the general covariance matrix is used. One has, however, to be aware of errors introduced due to the linear approximation which may result in an improperly narrow posterior. This can partly be compensated for, e.g. by a suitable increase of the student noise rate, in the expense of training speed due to the increased uncertainty.

Beside a more exhaustive investigation of models within the given framework there are several interesting questions for future research. These concentrate on improving the EKF approximation to deal with highly nonlinear and drifting concepts more efficiently, and on the question of model evaluation in an online manner [14] in analogy to that of batch learning [25].

## Acknowledgments

We would like to thank the Leverhulme Trust for their support (F/250/K). DS would also like to thank the British Council ARC programme Project 1037 for their support.

## References

[1] Leen T K, Schottky B and Saad D 1998 Two approaches to optimal annealing *Advances in Neural Information Processing Systems* vol 10, ed M I Jordan, M J Kearns and S A Solla (Cambridge, MA: MIT) p 301
Leen TK, Schottky B and Saad D 1999 Asymptotically Optimal Learning Rate Annealing *Phys. Rev.* E **59** 985

[2] Saad D and Rattray M 1997 Globally optimal parameters for online learning in multilayer networks *Phys. Rev. Lett.* **79** 2578

[3] Bishop C M 1995 *Neural Networks for Pattern Recognition* (New York: Oxford University Press)

[4] Amari S 1997 Natural learning in structured parameter spaces—natural Riemannian gradient *Advances in Neural Information Processing Systems* vol 9, ed M C Mozer, M I Jordan and T Petsche (Cambridge, MA: MIT) p 127

[5] Saad D (ed) 1998 *On-Line Learning in Neural Networks (Publications of the Newton Institute)* (Cambridge: Cambridge University Press)

[6] Kinouchi O and Caticha N 1992 Optimal generalization in perceptrons*J. Phys. A: Math. Gen.* **25** 6243

[7] Rattray M and Saad D 1997 Globally optimal rules for online learning in multilayer networks *J. Phys. A: Math. Gen.* **30** L771

[8] Rattray M, Saad D and Amari S 1998 Natural gradient descent for online learning *Phys. Rev. Lett.* **81** 5461

[9] Rattray M and Saad D 1998 Transients and asymptotics of natural gradient learning *Proc. Int. Conf. on Artificial Neural Networks* ed L Niklasson, M Bodóden and T Ziemke (London: Springer) p 165

[10] Neal R 1996 *Lecture Notes in Statistics: Bayesian Learning in Neural Networks* (London: Springer)

[11] Shah S, Palmieri F and Datum M 1992 Optimal filtering for fast learning feedforward neural networks *Neural Networks* **5** 779

[12] Sutton R S 1992 Adapting bias by gradient descent: An incremental version of the delta-bar-delta *Proc. 10th National Conf. on Artificial Intelligence* (Cambridge, MA: MIT) p 171

[13] Schottky B and Saad D 1998 Exact learning curves for EKF training *Proc. Int. Conf. on Artificial Neural Networks* ed L Niklasson, M Bodóden and T Ziemke (London: Springer) p 535

[14] deFreitas J F G, Niranjan M and Gee A H 1997 Hierarchical Bayesian–Kalman models for regularisation and ARD in sequential learning *Technical Report* CUED/F-INFENG/TR307 Cambridge University

[15] Opper M 1996 Online versus offline learning from random examples: general results *Phys. Rev. Lett.* **77** 4671

[16] Winther O and Solla S 1997 Bayesian online learning in the perceptron *European Symp. on Artificial Neural Networks (ESANN'97)* (Brussels: Facto) p 167
Winther O and Solla S 1998 Optimal Bayesian online *Theoretical Aspects of Neural Computation (TANC'97)* ed K Y M Wong, I King and D Y Yeung (Hong Kong: Springer)

[17] Biehl M and Schwarze H 1995 Learning by online gradient descent *J. Phys. A: Math. Gen.* **28** 643

[18] Brown R G and Hwang P Y C 1992 *Introduction to Random Signals and Applied Kalman Filtering* 2nd edn (New York: Wiley)

[19] Welch G and Bishop G 1995 An introduction to the Kalman filter *UNC-CH Computer Science Technical Report* 95-041

[20] Krogh A and Hertz J A 1992 Generalization in a linear perceptron in the presence of noise *J. Phys. A: Math. Gen.* **25** 1135

[21] Biehl M and Schwarze H 1993 Learning drifting concepts with neural networks *J. Phys. A: Math. Gen.* **26** 2651

[22] Saad D and Solla S 1995 On-line learning in soft committee machines *Phys. Rev.* E **52** 4225

[23] Reents G and Urbanczik R 1998 Self-averaging and online learning *Phys. Rev. Lett.* **80** 5445

[24] West A H L and Saad D 1997 On-line learning with adaptive back-propagation in two-layer networks *Phys. Rev.* E **56** 3426

[25] MacKay D J C 1992 Bayesian interpolation *Neural Comput.* **4** 415
MacKay D J C 1992 The evidence framework applied to classification networks *Neural Comput.* **4** 720