LETTER Communicated by C. Lee Giles

Hierarchical Bayesian Models for Regularization in Sequential Learning

J. F. G. de Freitas
M. Niranjan
A. H. Gee
Cambridge University Engineering Department, Cambridge CB2 1PZ, U.K.

We show that a hierarchical Bayesian modeling approach allows us to perform regularization in sequential learning. We identify three inference levels within this hierarchy: model selection, parameter estimation, and noise estimation. In environments where data arrive sequentially, techniques such as cross validation to achieve regularization or model selection are not possible. The Bayesian approach, with extended Kalman filtering at the parameter estimation level, allows for regularization within a minimum variance framework. A multilayer perceptron is used to generate the extended Kalman filter nonlinear measurements mapping. We describe several algorithms at the noise estimation level that allow us to implement on-line regularization. We also show the theoretical links between adaptive noise estimation in extended Kalman filtering, multiple adaptive learning rates, and multiple smoothing regularization coefficients.

1 Introduction

Sequential training of neural networks is important in applications where data sequences either exhibit nonstationary behavior or are difficult and expensive to obtain before the training process. Scenarios where this type of sequence arise include tracking and surveillance, control systems, fault detection, signal processing, communications, econometric systems, demographic systems, geophysical problems, operations research, and automatic navigation.

Although there has been great interest on the topic of regularization in batch learning tasks, this topic has not received much attention in sequential learning tasks. In this article, we adopt a hierarchical Bayesian framework in conjunction with dynamical state-space models to derive regularization algorithms for sequential estimation. These algorithms are based on estimating the noise processes on-line, while estimating the network weights with the extended Kalman filter (EKF). In addition, we show that this methodology provides a unifying theoretical framework for many sequential esti-

mation algorithms that attempt to avoid local minima in the error function. In particular, we show that adaptive noise covariances in extended Kalman filtering, multiple adaptive learning rates in on-line backpropagation, and multiple smoothing regularization coefficients are mathematically equivalent.

Section 2 describes the sequential learning task using state-space models and a three-level hierarchical Bayesian structure. The three levels of inference correspond to noise estimation, parameter estimation, and model selection. In section 3, we propose a solution to the parameter estimation level based on the application of the EKF to neural networks. Section 4 is devoted to the noise estimation level and regularization. Finally, we present our experiments in section 5 and point out several areas for further research in section 6.

2 Dynamical Hierarchical Bayesian Models

We address the problem of training neural networks sequentially within the following dynamical state-space framework:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{d}_k \tag{2.1}$$

$$\mathbf{y}_k = \mathbf{g}_k(\mathbf{w}_k, \mathbf{x}_k) + \mathbf{v}_k, \tag{2.2}$$

where $k (k = 1, \ldots, L)$ denotes the discrete time index. The output measurements of the system ($\mathbf{y}_k \in \Re^o$) depend on a nonlinear, multivariate, time-varying function of the system inputs ($\mathbf{x}_k \in \Re^d$) and a set of states ($\mathbf{w}_k \in \Re^m$). The measurements nonlinear mapping $\mathbf{g}_k(.)$ is approximated by a multilayer perceptron (MLP) whose weights are the model states $\mathbf{w}$. The work may be easily extended to encompass recurrent networks, radial basis networks, and many other approximation techniques. The measurements are assumed to be corrupted by noise $\mathbf{v}_k$, which we model as a zero mean, uncorrelated gaussian process with adaptive covariance R_k. We model the evolution of the model parameters by assuming that they depend on a deterministic component $\mathbf{w}_k$ and a stochastic component $\mathbf{d}_k$. The process noise $\mathbf{d}_k$ may represent our uncertainty on how the parameters evolve, modeling errors, or unknown inputs. We assume the process noise to be zero mean with adaptive covariance Q_k.

The sequential learning problem involves estimating the model parameters $\hat{\mathbf{w}}_k$, estimating noise models, and selecting the right model $\{M_j \mid j = 1, \ldots, r\}$ on the basis of a set of past measurements $Y_k = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_k\}$. This problem may be formulated in terms of three hierarchical hypothesis spaces. In each space, probabilities for each quantity are defined in terms of Bayes' rule:

$$\text{Posterior} = \frac{\text{Likelihood}}{\text{Evidence}} \text{Prior}.$$

We propose the following inference levels:

Level 1: Parameter estimation

$$\begin{aligned} &\mathrm{p}(\mathbf{w}_{k+1} \mid \mathrm{Y}_{k+1}, \mathrm{M}_j, \mathrm{R}_{k+1}, \mathrm{Q}_k) \\ &\quad = \frac{\mathrm{p}(\mathbf{y}_{k+1} \mid \mathbf{w}_{k+1}, \mathrm{M}_j, \mathrm{R}_{k+1}, \mathrm{Q}_k)}{\mathrm{p}(\mathbf{y}_{k+1} \mid \mathrm{Y}_k, \mathrm{M}_j, \mathrm{R}_{k+1}, \mathrm{Q}_k)} \mathrm{p}(\mathbf{w}_{k+1} \mid \mathrm{Y}_k, \mathrm{M}_j, \mathrm{R}_{k+1}, \mathrm{Q}_k). \end{aligned} \tag{2.3}$$

Level 2: Noise estimation

$$\begin{aligned} &\mathrm{p}(R_{k+1}, Q_k \mid Y_{k+1}) \\ &\quad = \frac{\mathrm{p}(\mathbf{y}_{k+1} \mid \mathrm{Y}_k, \mathrm{M}_j, \mathrm{R}_{k+1}, \mathrm{Q}_k)}{\mathrm{p}(\mathbf{y}_{k+1} \mid \mathrm{Y}_k, \mathrm{M}_j)} \mathrm{p}(R_{k+1}, Q_k \mid Y_k, M_j). \end{aligned} \tag{2.4}$$

Level 3: Model selection

$$\mathrm{p}(M_j \mid Y_{k+1}) = \frac{\mathrm{p}(\mathbf{y}_{k+1} \mid \mathrm{Y}_k, \mathrm{M}_j)}{\mathrm{p}(\mathbf{y}_{k+1} \mid \mathrm{Y}_k)} \mathrm{p}(M_j \mid Y_k). \tag{2.5}$$

The likelihood function at a particular level constitutes the evidence function at the next higher level. Therefore, by maximizing the evidence function in the parameter estimation level, we are, in fact, maximizing the likelihood of the noise covariances R_k and Q_k as the new data arrive. This result plays an important role when we devise methods for estimating the noise covariances.

At the parameter estimation level, we apply the EKF algorithm to estimate the weights of an MLP. The EKF, however, requires knowledge of the noise covariances. To overcome this difficulty, in section 4 we present techniques for estimating these covariances in slowly changing non-stationary environments. There, we show that algorithms for estimating the noise covariance allow us to perform regularization in a sequential framework. Model selection is not covered in this article.

3 Parameter Estimation

For optimality reasons, we want $\hat{\mathbf{w}}_k$ to be an unbiased, minimum variance, and consistent estimate (Gelb, 1974). The minimum variance estimation framework is based on minimizing the variance of the neural network weights, thereby leading to smooth estimates for the network weights and outputs. A popular and suboptimal strategy for obtaining estimates of this type is to employ the EKF for parameter estimation. The EKF is a minimum variance estimator based on a Taylor series expansion of the nonlinear function $\mathbf{g}_k(\mathbf{w}_k, \mathbf{x}_k)$ around the previous estimate. Using this expansion and under the assumptions that the state-space model noise pro-

cesses are uncorrelated with each other and the initial estimates of the parameters $\mathbf{w}_k$ and their covariance matrix P_k, we can model the prior, evidence, and likelihood functions as follows (de Freitas, Niranjan, & Gee 1997):

$$\begin{aligned}\text{Prior} &= p(\mathbf{w}_{k+1} \mid Y_k, M_j, R_{k+1}, Q_k) \\ &\approx \mathcal{N}(\hat{\mathbf{w}}_k, P_k + Q_k)\end{aligned}$$

$$\begin{aligned}\text{Evidence} &= p(\mathbf{y}_{k+1} \mid Y_k, M_j, R_{k+1}, Q_k) \\ &\approx \mathcal{N}(\mathbf{g}(\hat{\mathbf{w}}_k, \mathbf{x}_{k+1}), G_{k+1}(P_k + Q_k)G_{k+1}^T + R_{k+1})\end{aligned}$$

$$\begin{aligned}\text{Likelihood} &= p(\mathbf{y}_{k+1} \mid \mathbf{w}_{k+1}, M_j, R_{k+1}, Q_k) \\ &\approx \mathcal{N}(\mathbf{g}(\mathbf{w}_{k+1}, \mathbf{x}_{k+1}), R_{k+1}),\end{aligned}$$

where G denotes the Jacobian $\frac{\partial \mathbf{g}}{\partial \mathbf{w}}|_{(\mathbf{w}=\hat{\mathbf{w}})}$ and the symbol T denotes the transpose of a matrix. Since the EKF is a suboptimal estimator based on linearization of a nonlinear mapping, $\hat{\mathbf{w}}$ is only an approximation to the expected value, and, strictly speaking, P_k is an approximation to the covariance matrix. It is also important to point out that the EKF may diverge as a result of its inherent approximations. The consistency of the EKF may be evaluated by means of extensive Monte Carlo simulations (Bar-Shalom & Li 1993). Substituting the expressions for the prior, likelihood, and evidence into equation (2.3), yields the posterior density function:

$$\text{Posterior} = p(\mathbf{w}_{k+1} \mid Y_{k+1}, M_j, R_{k+1}, Q_k) \approx \mathcal{N}(\hat{\mathbf{w}}_{k+1}, P_{k+1}),$$

where the updated weights, covariance and Kalman gain (K_{k+1}) are given by:

$$\hat{\mathbf{w}}_{k+1} = \hat{\mathbf{w}}_k + K_{k+1}(\mathbf{y}_{k+1} - \mathbf{g}(\hat{\mathbf{w}}_k, \mathbf{x}_{k+1})) \tag{3.1}$$

$$P_{k+1} = P_k + Q_k - K_{k+1}G_{k+1}(P_k + Q_k) \tag{3.2}$$

$$K_{k+1} = (P_k + Q_k)G_{k+1}^T[R_{k+1} + G_{k+1}(P_k + Q_k)G_{k+1}^T]^{-1}. \tag{3.3}$$

By grouping the MLP weights into a single vector $\mathbf{w}$, we can use the EKF equations (3.1–3.3) to compute new estimates of the weights recursively. The entries of the Jacobian matrix are calculated by backpropagating the o output values $\{y_1(t), y_2(t), \cdots, y_o(t)\}$ through the network. An example of how to do this for a simple MLP is presented in appendix B of de Freitas et al. (1997).

One of the earliest implementations of EKF-trained MLPs is due to Singhal and Wu (1988). The algorithm's computational complexity is of the order om^2 multiplications per time step. Shah, Palmieri, and Datum (1992) and Puskorius and Feldkamp (1991) have proposed various approximations to the weights covariance so as to simplify this problem. The EKF is an improvement over conventional MLP estimation techniques, such as online backpropagation, in that it makes use of second-order statistics (Ruck, Rogers, Kabrisky, Maybeck, & Oxley, 1992; Schotky & Saad, 1999). These statistics are essential for placing error bars on the predictions and for combining separate networks into committees of networks when $p(\mathbf{w}_k \mid Y_k)$ has multiple modes (Bar-Shalom & Li, 1993; Blom & Bar-Shalom, 1988; Kadirkamanathan & Kadirkamanathan, 1995).

4 Noise Estimation and Regularization

A well-known limitation of the EKF is the assumption of known a priori statistics to describe the measurement and process noise. Setting these noise levels appropriately often makes the difference between success and failure in the use of the EKF (Candy, 1986). In many applications, it is not straightforward to choose the noise covariances (Jazwinski, 1970). In addition, in environments where the noise statistics change with time, such an approach can lead to large estimation errors and even to a divergence of errors. Several researchers in the estimation, filtering and control fields have attempted to solve this problem (Jazwinski, 1969; Mehra, 1970, 1971; Myers & Tapley, 1976; Tenney, Hebbert, & Sandall, 1977). Mehra (1972) and Li and Bar-Shalom (1994) give brief surveys on this topic.

It is important to note that algorithms for estimating the noise covariances within the EKF framework can lead to a degradation of the performance of the EKF. By increasing the process noise covariance Q_k, the Kalman gain also increases, thereby producing bigger changes in the weight updates (refer to equations 3.1 and 3.3). That is, more importance is placed on the most recent measurements. Consequently, it may be asserted that filters with adaptive process noise covariances exhibit adaptive memory. Additionally, as the Kalman gain increases, the bandwidth of the filter also increases (Bar-Shalom & Li, 1993). Therefore, the filter becomes less immune to noise and outliers.

The amount of oscillation in the model prediction clearly depends on the value of the process noise covariance. As a result, this covariance can be used as a regularization mechanism to control the smoothness of the prediction.

In the following subsections, we derive three algorithms to estimate the noise covariances. The first two derivations serve to illustrate the fact that algorithms for adapting distributed learning rates, smoothing regularizers, or stochastic noise in gradient descent methods are equivalent (see Figure 1).

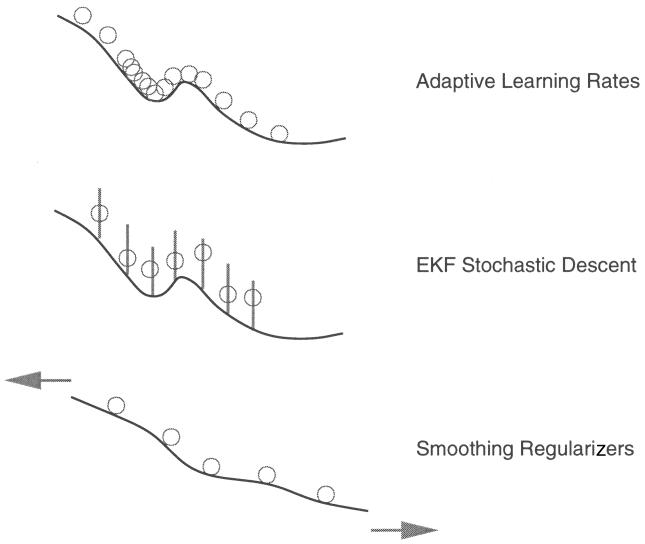

Figure 1: One-dimensional example of several adaptive gradient descent methods. To escape from local minima, we can increase the momentum of the ball as it approaches a particular local minimum, allow the ball to bounce randomly within a given vertical interval (stochastic descent), or stretch the surface with smoothing regularizers.

In the derivation of the third algorithm, we address the trade-off between regularization and tracking performance in sequential learning.

4.1 Algorithm 1: Adaptive Distributed Learning Rates. Sutton (1992b) proposed an optimization approach for linear networks, using the Kalman filter equations with P_k updated by a variation of the least-mean-square rule (Jacobs, 1988; Sutton, 1992a). The main purpose of the method was to reduce the computational time at the expense of a small deterioration in the performance of the estimator. Another important aspect of the algorithm is that it circumvents the problem of choosing the process noise covariance Q. The technique involves approximating P with a diagonal matrix, whose ith diagonal entry is given by:

$$p_{mm} = \exp(\beta_m),$$

where β_m is updated by the least-mean-square rule modified such that the learning rates for each parameter are updated sequentially. The diagonal matrix approximation to P implies that the model parameters are uncorrelated. This assumption may, of course, degrade the performance of the EKF estimator.

To circumvent the problem of choosing the process noise covariance Q when training nonlinear neural networks, while at the same time increasing computational efficiency, we have extended Sutton's algorithm to the nonlinear case. In particular, we make use of MLPs with sigmoidal basis functions in the hidden layer and linear basis functions in the output layer. The network weights and Kalman gain are updated using the EKF, while the weights covariance P is updated by backpropagating the squared output errors, with β as follows:

$$\beta_{k+1} = \begin{cases} \beta_k + \eta \delta_{ik} o_{jk} & \text{output layer} \\ \beta_k + \eta w_{ijk} \delta_{ik} o_{jk} (1 - o_{jk}) x_{dk} & \text{hidden layer} \end{cases}$$

where the index i corresponds to the ith neuron in the output layer, j to the jth neuron in the hidden layer, d to the dth input variable, and k to the estimation step. δ_{ik} represents the k-output error for neuron i. The symbols o_i and η denote the output of the ith neuron in layer i and the learning rate, respectively. This learning rate is a parameter that quantifies another parameter P_k. We shall refer to it as a hyperparameter. We have found in practice that choosing this hyperparameter is easier than choosing the process noise covariance matrix.

The EKF equation used to update the weights is similar to the update equations typically used to compute the weights of neural networks by error backpropagation. The only difference is that it assumes that there is a different adaptive learning-rate parameter for each weight. The mathematical relation between adaptive learning rates ($\mathcal{L}$) in on-line backpropagation and Kalman filtering parameters is given by de Freitas et al. (1997):

$$\mathcal{L} = (P_k + Q_k)(R_{k+1} + G_{k+1}(P_k + Q_k)G_{k+1}^T)^{-1}.$$

Thus, adapting the process noise is equivalent to adapting the learning rates.

4.2 Algorithm 2: Evidence Maximization with Weight Decay Priors. We derive a sequential method for updating R and Q, based on the evidence approximation framework with weight decay priors for batch learning (see Bishop, 1995, chap. 10). In so doing, we show that algorithms for adapting the noise covariances are equivalent to algorithms that make use of smoothing regularizers.

In the evidence approximation framework for batch learning, the prior and likelihood are expressed as follows:

$$\mathrm{p}(\mathbf{w}) = \frac{1}{(2\pi)^{m/2}\alpha^{-m/2}} \exp\left(-\frac{\alpha}{2}\|\mathbf{w}\|^2\right) \tag{4.1}$$

$$\mathrm{p}(Y_k \mid \mathbf{w}) = \frac{1}{(2\pi)^{L/2}\beta^{-L/2}} \exp\left(-\frac{\beta}{2}\sum_{k=1}^{L}(\mathbf{y}_\mathrm{k} - \hat{\mathbf{g}}_{\mathrm{L,m}}(\mathbf{w}, \mathbf{x}_\mathrm{k}))^2\right), \tag{4.2}$$

where $\hat{\mathbf{g}}_{L,m}(\mathbf{w}, \mathbf{x}_k)$ corresponds to the model prediction. The hyperparameters α and β control the variance of the prior distribution of the weights and the variance of the measurement noise. α also plays the role of the regularization coefficient. Using Bayes' rule (see equation 2.3) and taking into account that the evidence does not depend on the weights, the following posterior density function may be obtained:

$$p(\mathbf{w} \mid Y_k) = \frac{1}{Z_s(\alpha, \beta)} \exp(-S(\mathbf{w})),$$

where $Z_s(\alpha, \beta)$ is a normalizing factor. For the prior and the likelihood of equations 4.1 and 4.2, $S(\mathbf{w})$ is given by

$$S(\mathbf{w}) = \frac{\alpha}{2}\|\mathbf{w}\|^2 + \frac{\beta}{2}\sum_{k=1}^{L}(\mathbf{y}_k - \hat{\mathbf{g}}_{L,m}(\mathbf{w}, \mathbf{x}_k))^2. \tag{4.3}$$

The posterior density may be approximated by applying a Taylor series expansion of $S(\mathbf{w})$ around a local minimum ($\mathbf{w}_{MP}$) and retaining the series terms up to second order:

$$S(\mathbf{w}) = S(\mathbf{w}_{MP}) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_{MP})^T A(\mathbf{w} - \mathbf{w}_{MP}).$$

Hence, the gaussian approximation to the posterior density function becomes:

$$p(\mathbf{w} \mid Y_k) = \frac{1}{(2\pi)^{m/2}|A|^{-1/2}} \exp\left(-\frac{1}{2}(\mathbf{w} - \mathbf{w}_{MP})^T A(\mathbf{w} - \mathbf{w}_{MP})\right). \tag{4.4}$$

Maximizing the posterior probability density function involves minimizing the error function given by equation 4.3. Equation 4.3 is a particular case of a regularized error function. More generally, this error function is given by:

$$S(\mathbf{w}) = \sum_{k=1}^{L}(\mathbf{y}_k - \hat{\mathbf{g}}_{L,m}(\mathbf{w}, \mathbf{x}_k))^2 + \nu\Omega,$$

where ν is a positive parameter that serves to balance the trade-off between smoothness and data approximation. A large value of ν places more importance on the smoothness of the model; a small value of ν places more emphasis on fitting the data. The functional Ω penalizes for excessive model complexity (Girosi, Jones, & Poggio, 1995).

In the evidence framework, the parameters $\mathbf{w}$ are obtained by minimizing equation 4.3, while the hyperparameters α and β are obtained by maximizing the evidence $p(Y_k \mid \alpha, \beta)$ after approximating the posterior density function by a gaussian function centered at $\mathbf{w}_{MP}$. In doing so, the following recursive formulas for α and β are obtained:

$$\alpha_{k+1} = \frac{\gamma}{\sum_{i=1}^{m} w_i^2} \quad \text{and} \quad \beta_{k+1} = \frac{L - \gamma}{\sum_{k=1}^{L}(\mathbf{y}_k - \hat{\mathbf{g}}_{L,m}(\mathbf{w}_k, \mathbf{x}_k))^2}. \tag{4.5}$$

The quantity $\gamma = \sum_{i=1}^{m} \frac{\lambda_i}{\lambda_i + \alpha}$, represents the effective number of parameters, where the λ_i are the eigenvalues of the Hessian of the error function E_D. The effective number of parameters, as the name implies, is the number of parameters that effectively contributes to the neural network mapping. The remaining weights have no contribution because their magnitudes are forced to zero by the weight decay prior.

It is possible to maximize the posterior density function by performing integrations over the hyperparameters analytically (Buntine & Weigend, 1991; Mackay, 1996; Williams, 1995; Wolpert, 1993). The latter approach is known as the MAP framework for α and β. The hyperparameters computed by the MAP framework differ from the ones computed by the evidence framework in that the former makes use of the total number of parameters and not only the effective number of parameters. That is, α and β are updated according to:

$$\alpha_{k+1} = \frac{m}{\sum_{i=1}^{m} w_i^2} \quad \text{and} \quad \beta_{k+1} = \frac{L}{\sum_{k=1}^{L} (\mathbf{y}_\mathrm{k} - \hat{\mathbf{g}}_{\mathrm{L,m}}(\mathbf{w}_\mathrm{k}, \mathbf{x}_\mathrm{k}))^2}. \tag{4.6}$$

By comparing the expressions for the prior, likelihood, and evidence in the EKF framework (see section 3) with equations 4.1, 4.2, and 4.4, we can establish the following relations:

$$P = A^{-1}, \quad Q = \alpha^{-1} I_m - A^{-1} \quad \text{and} \quad R = \beta^{-1} I_o, \tag{4.7}$$

where I_m and I_o represent identity matrices of sizes m and o, respectively. Therefore, it is possible to update Q and R sequentially by expressing them in terms of the sequential updates of α and β. That is, adapting the noise processes is equivalent to adapting the regularization coefficients. A moving window may be implemented to estimate β. The size of the window is a parameter that requires tuning.

4.3 Algorithm 3: Evidence Maximization with Sequentially Updated Priors. In EKF, we have knowledge of the equation describing the evidence function in terms of the noise covariances. Consequently, we can compute R_k and Q_k automatically by maximizing the evidence density:

$$\mathrm{p}(y_{k+1} \mid Y_k, M_j, R_{k+1}, Q_k) \sim \mathcal{N}(\mathbf{g}_{\mathrm{k+1}}(\hat{\mathbf{w}}_\mathrm{k}, \mathbf{x}_{\mathrm{k+1}}), \\ G_{k+1}(P_k + Q_k)G_{k+1}^T + R_{k+1}).$$

Strictly speaking, this is not a full Bayesian solution. We are computing solely the likelihood of the noise covariances. That is, we are assuming no knowledge of the prior at the noise estimation level. For simplicity, we have restricted our analysis in this section to a single output. Let us now define the model residuals:

$$r_{k+1} = y_{k+1} - \mathbf{E}[y_{k+1} \mid Y_k, M_j, R_{k+1}, Q_k] = y_{k+1} - \mathbf{g}_{\mathrm{k+1}}(\hat{\mathbf{w}}_\mathrm{k}, \mathbf{x}_{\mathrm{k+1}}). \tag{4.8}$$

It follows that the probability of the residuals is equivalent to the evidence function at the parameter estimation level, that is:

$$\mathrm{p}(r_{k+1}) = \mathrm{p}(y_{k+1} \mid Y_k, M_j, R_{k+1}, Q_k).$$

Let us assume initially that the process noise covariance may be described by a single parameter q—more specifically:

$$Q = qI_m.$$

The maxima of the evidence function with respect to q may be calculated by differentiating the evidence function as follows:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}q}\mathrm{p}(r_{k+1}) = \frac{1}{(2\pi)^{1/2}} \exp\left(-\frac{1}{2}\frac{r_{k+1}^2}{G_{k+1}(P_k+Q_k)G_{k+1}^T + R_{k+1}}\right) \\ \left[-\frac{1}{2}G_{k+1}G_{k+1}^T(G_{k+1}(P_k+Q_k)G_{k+1}^T + R_{k+1})^{-3/2}\right. \\ \left. + \frac{1}{2}r_{k+1}^2 G_{k+1}G_{k+1}^T(G_{k+1}(P_k+Q_k)G_{k+1}^T + R_{k+1})^{-5/2}\right].\end{aligned}$$

Equating the derivative to zero yields:

$$r_{k+1}^2 = \mathbf{E}[r_{k+1}^2]. \tag{4.9}$$

It is straightforward to prove that this singularity corresponds to a global maximum on $[0, \infty)$ by computing the second derivative. This result reveals that maximizing the evidence function corresponds to equating the covariance over time r_{k+1}^2 to the ensemble covariance $\mathbf{E}[r_{k+1}^2]$. That is, maximizing the evidence leads to a covariance matching method.

Jazwinski (1969; Jazwinski & Bailie, 1967) devised an algorithm for updating q according to equation 4.9. Since

$$r_{k+1}^2 = G_{k+1}P_kG_{k+1}^T + qG_{k+1}G_{k+1}^T + R_{k+1},$$

it follows that q may be recursively computed according to:

$$q = \begin{cases} \frac{r_{k+1}^2 - \mathbf{E}[r_{k+1}^2 \mid q=0]}{G_{k+1}G_{k+1}^T} & \text{if } q \geq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{4.10}$$

This estimator increases q each time the variance over time of the model residuals exceeds the ensemble variance. When q increases, the Kalman gain also increases, and consequently the model parameters update also increases (see equations 3.1 and 3.3). That is, the estimator places more

emphasis on the incoming data. As long as the variance over time of the residuals remains smaller than the ensemble covariance, the process noise input is zero and the filter carries on minimizing the variance of the parameters (i.e., tending to a regularized solution). Section 5.2 discusses an experiment where this behavior is illustrated.

The estimator of equation 4.10 is based on a single residual and is therefore of little statistical significance. This difficulty is overcome by employing a sliding window to compute the sample mean for N predicted residuals instead of a single residual. Jazwinski (1969) shows that for the following sample mean,

$$m_r = \frac{1}{N}\sum_{l=1}^{N}\frac{r_{k+l}}{R_{k+l}^{1/2}},$$

we may proceed as above, by maximizing $\mathrm{p}(m_r)$, to obtain the following estimator:

$$q = \begin{cases} \frac{m_r^2 - \mathbf{E}[m_r^2 | q=0]}{S} & \text{if } q \geq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{4.11}$$

where

$$\mathbf{E}[m_r^2 \mid q = 0] = S_N P_k S_N^T + 1/N,$$

$$S = S_N S_N^T + S_{N-1} S_{N-1}^T +, \cdots, + S_1 S_1^T$$

and

$$S_N = \frac{1}{N}\sum_{l=1}^{N}\frac{1}{R_{k+l}^{1/2}} G_{k+l}, \quad S_{N-1} = \frac{1}{N}\sum_{l=2}^{N}\frac{1}{R_{k+l}^{1/2}} G_{k+l}, \quad \ldots,$$

$$S_1 = \frac{1}{N}\frac{1}{R_{k+N}^{1/2}} G_{k+N}. \tag{4.12}$$

With this estimator, one has to choose the length N of the moving window used to update q. If the window size is too small, the algorithm places more emphasis on fitting the incoming data than fitting the previous data. As a result, it might never converge. On the other hand, if the window size is too large, the algorithm will fail to adapt quickly to new environments. We refer to the problem of choosing the right window length as the regularization/tracking dilemma. It is a dilemma because we cannot ascertain, without a priori knowledge, whether the fluctuations in the data correspond to time-varying components of the data or to noise.

It is possible to extend the derivation to a more general noise model by adopting the following covariance:

$$Q = \begin{bmatrix} q_1 & 0 & \cdots & 0 \\ 0 & q_2 & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & q_m \end{bmatrix}. \tag{4.13}$$

By calculating the derivative of the evidence function with respect to a generic diagonal entry of Q and then equating to zero, we obtain an estimator involving the following system of equations:

$$\begin{bmatrix} \left(\frac{\partial y_{k+1}}{\partial \mathbf{w}_1}\right)^2 & \left(\frac{\partial y_{k+1}}{\partial \mathbf{w}_2}\right)^2 & \cdots & \left(\frac{\partial y_{k+1}}{\partial \mathbf{w}_m}\right)^2 \\ \left(\frac{\partial y_k}{\partial \mathbf{w}_1}\right)^2 & \left(\frac{\partial y_k}{\partial \mathbf{w}_2}\right)^2 & & \left(\frac{\partial y_k}{\partial \mathbf{w}_m}\right)^2 \\ \vdots & & \ddots & \\ \left(\frac{\partial y_{k-N}}{\partial \mathbf{w}_1}\right)^2 & \left(\frac{\partial y_{k-N}}{\partial \mathbf{w}_2}\right)^2 & & \left(\frac{\partial y_{k-N}}{\partial \mathbf{w}_m}\right)^2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_m \end{bmatrix} = \begin{bmatrix} \varepsilon_{k+1} \\ \varepsilon_k \\ \vdots \\ \varepsilon_{k-N} \end{bmatrix}. \tag{4.14}$$

Multiple hyperparameters are very handy when one considers distributed priors for automatic relevance determination (input and basis functions selection) (de Freitas et al., 1997; Mackay 1994, 1995). Estimating Q in equation 4.14 is, however, not very reliable (de Freitas et al., 1997; Mackay 1994, 1995) because it involves estimating a large number of noise parameters and, in addition, requires a long moving window of size N to avoid illconditioning.

We can also maximize the evidence function with respect to $R_k = rI_o$ and obtain the following estimator for r:

$$r = \begin{cases} r_k^2 - G_k(P_k + Q_k)G_k^T & \text{if } r \geq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{4.15}$$

The hyperparameter r is not as useful as q in controlling filter divergence. This is because r can slow the rate of decrease of the covariance matrix P_k, but cannot cause it to increase (see equations 3.1–3.3).

5 Experiments

5.1 Experiment 1: Comparison Between the Noise Estimation Methods. To compare the performance of the various EKF training algorithms, 100 input-output data vectors were generated from the following nonlinear,

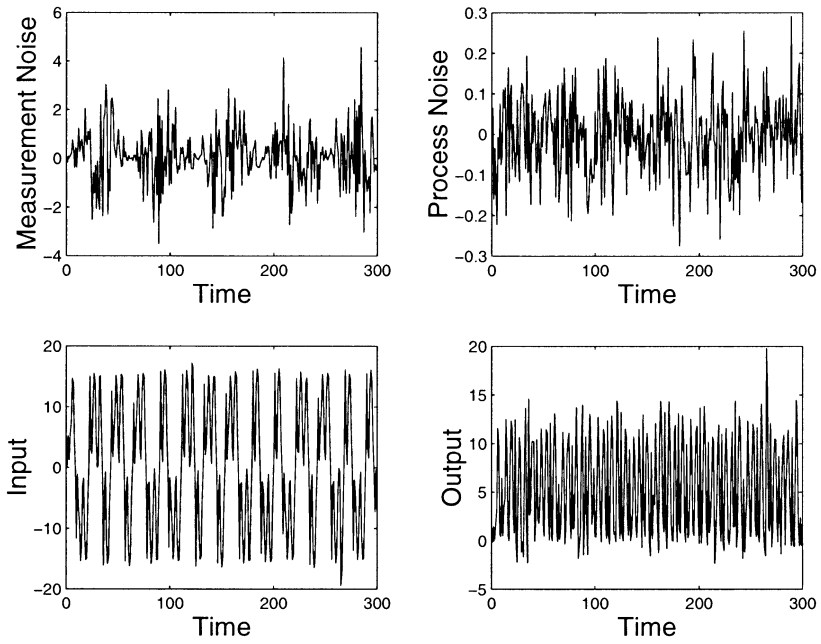

Figure 2: Data generated to train MLP.

nonstationary process:

$$x_k = 0.5x_{k-1} + \frac{25x_{k-1}}{1 + x_{k-1}^2} + 8\cos(1.2(k-1)) + d_k$$

$$y_k = \frac{x_k^2}{20} + v_k,$$

where x_k denotes the input vectors and y_k the output vectors of 300 time samples each. The gaussian process noise standard deviation was set to 0.1; the measurement noise standard deviation was set to $3\sin(0.05k)$. The initial state x_0 was 0.1. Figure 2 shows the data generated with this model. The changes of the measurement noise variance are similar to the ones typically observed in financial returns data (Shephard, 1996).

We then proceeded to train an MLP with 10 sigmoidal neurons in the hidden layer and 1 linear output neuron with the following methods: the standard EKF algorithm, the EKF algorithm with P_k updated by error backpropagation (EKFBP), with evidence maximization and weight decay priors (EKFEV), with MAP noise adaptation (EKFMAP), and with evidence maximization and sequentially updated priors (EKFQ). The initial variance of the weights, initial weights covariance matrix entries, initial R, and

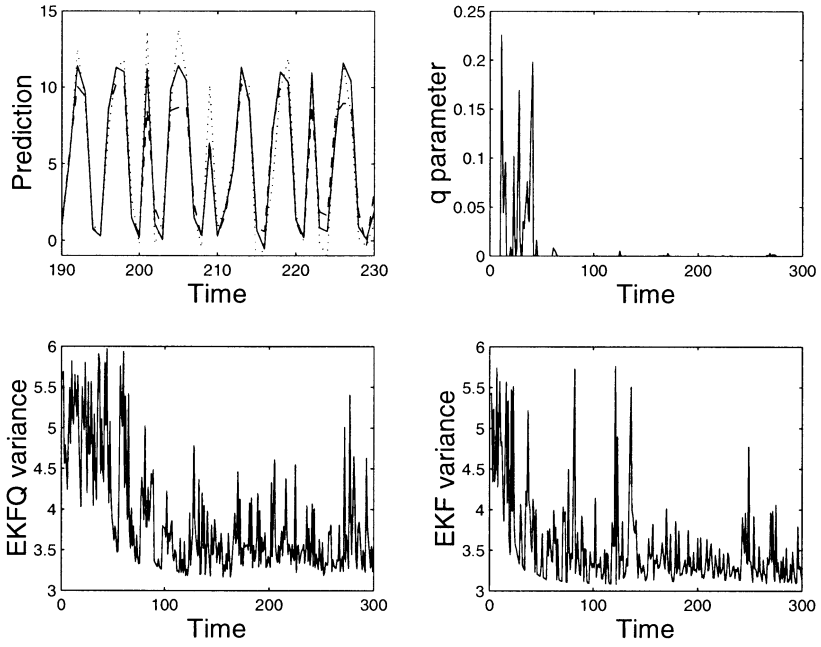

Figure 3: Simulation results for the EKF and EKFQ algorithms. The top left plot shows the actual data [$\cdots$] and the EKF [- -] and EKFQ [—] one-step-ahead predictions. The top right plot shows the estimated process noise parameter. The bottom plots show the innovations covariance for both methods.

initial Q were set to 1, 10, 3, and $1e-5$, respectively. The length of the sliding window of the adaptive noise algorithms was set to 10 time samples.

The simulation results for the EKF and EKFQ algorithms are shown in Figure 3. Note that the EKFQ algorithm slows the convergence of the EKF parameter estimates so as to be able to track the changing measurement variance. In Table 1, we compare the one-step-ahead normalized square errors (NSE) obtained with each method. The NSE are defined as follows:

$$\text{NSE} = \sqrt{\sum_{k=1}^{300} (\mathbf{y}_k - \hat{\mathbf{g}}(\mathbf{w}_k, \mathbf{x}_k))^2} \ .$$

According to Table 1, it is clear that the only algorithm that provides a clear prediction improvement over the standard EKF algorithm is the evidence maximization algorithm with sequentially updated priors. In terms of computational time, the EKF algorithm with P_k updated by backpropagation is

Table 1: Simulation Results for 100 Runs in Experiment 1.

	NSE	Mega Floating-Point operations
EKF	25.95	21.9
EKFQ	23.01	24.1
EKFMAP	61.06	22.6
EKFEV	73.94	22.6
EKFBP	58.87	2.2

faster, but its prediction is worse than the one for the standard EKF. This is not a surprising result considering the assumption of uncorrelated weights. The EKFEV and EKFMAP performed poorly because they require the network weights to converge to a good solution before the noise covariances can be updated. That is, the noise estimation algorithm does not facilitate the estimation of the weights, as it happens in the case of the EKFQ algorithm. The EKFEV and EKFMAP therefore appear to be unsuitable for sequential learning tasks.

5.2 Experiment 2: Sequential Evidence Maximization with Sequentially Updated Priors. This experiment aims to describe the behavior of the evidence-maximization algorithm (EKFQ) of equation 4.11 in a time-varying, noisy, and chaotic scenario. The problem tackled is a more difficult variation of the chaotic quadratic or logistic map. One hundred input (y_k) and output (y_{k+1}) data vectors were generated according to the following equation:

$$y_{k+1} = \begin{cases} 3.5y_k(1-y_k) + v_k & 1 \le k \le 150 \\ 3.7y_k(1-y_k) + v_k & 150 < k \le 225 \\ 3.1y_k(1-y_k) + v_k & 225 < k \le 300 \end{cases}$$

where v_k denotes gaussian noise with a standard deviation of 0.01. In the interval $150 < k \le 225$, the series exhibits chaotic behavior. A single hidden-layer MLP with 10 sigmoidal neurons in the hidden layer and a single output linear neuron were trained to approximate the mapping between (y_k) and (y_{k+1}). The initial weights, weights covariance matrix diagonal entries, R, and Q were set to 1, 100, 1×10^{-4}, and 0, respectively. The sliding window to estimate Q was set to three time samples.

As shown in Figure 4, during the initialization and after each change of behavior (samples 150 and 225), the estimator for the process noise covariance Q becomes active. That is, each time the environment undergoes a severe change, more importance is given to the new data. As the environment stabilizes, the minimum variance minimization criterion of the Kalman filter leads to a decrease in the variance of the output. Therefore, it is possible to design an estimator that balances the trade-off between regularization

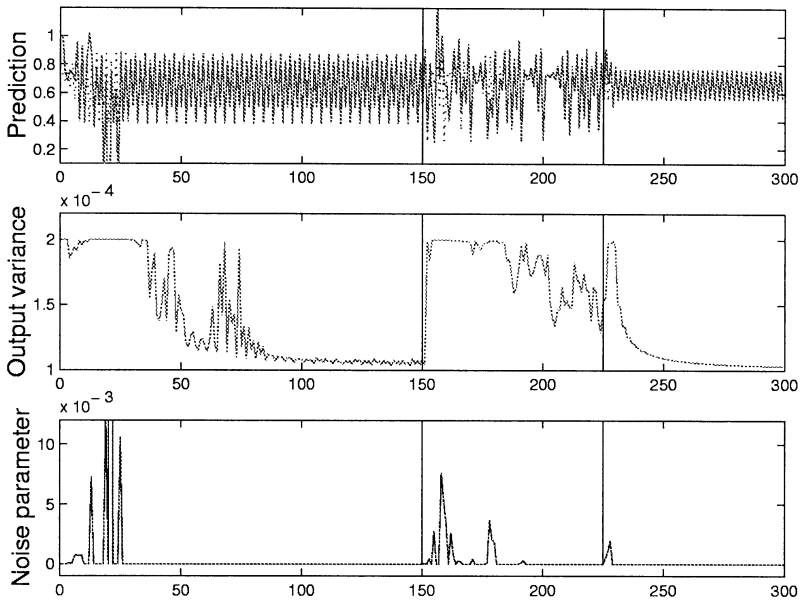

Figure 4: Performance of the evidence maximization for a nonstationary chaotic quadratic map. (Top) True data [· · ·] and the prediction [—]. (Middle) Output confidence intervals. (Bottom) Value of the adaptive noise parameter.

and tracking. The results obtained with the EKF and EKFQ algorithms are summarized in Table 2.

5.3 Example with Real-World Data. The mathematical modeling of financial derivatives has become increasingly popular for two reasons. First, financial institutions have much interest in developing more sophisticated pricing models for options contracts (Hull, 1997). Second, options data offer an excellent source of difficult and challenging problems to the statistical and neural computing communities (Hutchinson, Lo, & Poggio, 1994; Ingber &

Table 2: Simulation Results for 100 Runs of the Quadratic Chaotic Map.

	NSE	Mega Floating-Point Operations
EKF	31.76	21.8
EKFQ	1.37	23.0

Note: The EKFQ algorithm provides a large improvement over the EKF at a very small computational cost.

Wilson, 1999; Niranjan, 1996). So far, the research results seem to provide clear evidence that there is a nonlinear and nonstationary relation between the options' price and the cash products' price, maturity time, strike price, interest rates, and variance of the returns on the cash product (volatility). The standard model used to describe this relation is the Black-Scholes (1973) model.

Hutchinson et. al. (1994) and Niranjan (1996) have focused on the options pricing problem from a neural computing perspective. The former showed that good approximations to the widely used Black-Scholes formula may be obtained with neural networks, while the latter looked at the nonstationary aspects of the problem. Our work follows from Niranjan (1996), with the aim of showing that more accurate tracking of the options prices can be achieved by adapting the noise covariances. We train MLPs to generate one-step-ahead predictions of the options prices. These predictions for a group of options on the same cash product, but with different strike prices and/or time to maturity, can be used to determine whether one of the options is being mispriced.

We treat the cash product's value normalized by the strike price and time to maturity as the network inputs. The network's output consists of the call and put option prices normalized by the strike price. We used five pairs of call and put option contracts on the FTSE100 index (February 1994–December 1994) to evaluate our pricing algorithms. The parameters were estimated by the following methods:

Trivial: Involves using the current value of the option as the next prediction.

RBF-EKF: Represents a regularized radial basis function network with four hidden neurons, which was originally proposed in Hutchinson et al. (1994). The output weights are estimated with a Kalman filter, while the means of the radial functions correspond to random subsets of the data and their covariance is set to the identity matrix as in Niranjan (1996).

BS: Corresponds to a conventional Black-Scholes model with two outputs (normalized call and put prices) and two parameters (risk-free interest rate and volatility). The risk-free interest rate was set to 0.06, while the volatility was estimated over a moving window (of 50 time steps) as described in Hull (1997).

MLP-EKF: Stands for an MLP, with six sigmoidal hidden units and a linear output neuron, trained with the EKF algorithm. The noise covariances R and Q and the states covariance P were set to diagonal matrices with entries equal to 10^{-6}, 10^{-7}, and 10, respectively. The weights prior corresponded to a zero mean gaussian density with covariance equal to 1.

MLP-EKFQ: Represents an MLP, with six sigmoidal hidden units and a linear output neuron, trained with the EKF with evidence maximization and sequentially updated priors. The states covariance P was given by a

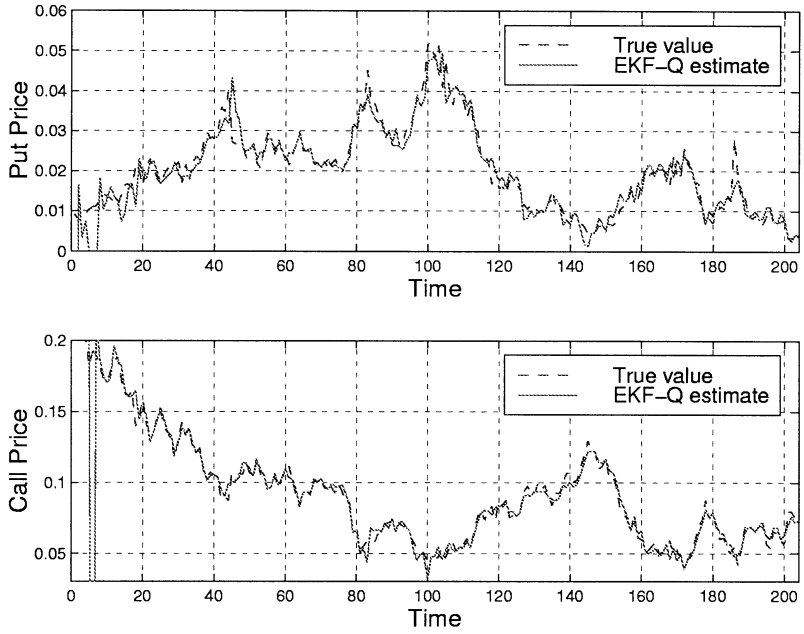

Figure 5: Tracking put and call option prices with an MLP trained with the EKFQ algorithm.

diagonal matrix with entries equal to 10. The weights prior corresponded to a zero mean gaussian density with covariance equal to 1.

Figure 5 shows the one-step-ahead predictions obtained with the EKFQ algorithm. In Table 3, we compare the one-step-ahead normalized square errors obtained with each method. The square errors were measured over only the last 100 days of trading, so as to allow the algorithms to converge. It is clear from the results that adapting the process noise covariance sequentially leads to improved predictions with the options data.

Table 3: One-Step-Ahead Prediction Errors on Call Options.

Strike Price	2925	3025	3125	3225	3325
Trivial	0.0783	0.0611	0.0524	0.0339	0.0205
RBF-EKF	0.0538	0.0445	0.0546	0.0360	0.0206
BS	0.0761	0.0598	0.0534	0.0377	0.0262
MLP-EKF	0.0414	0.0384	0.0427	0.0285	0.0145
MLP-EKFQ	0.0404	0.0366	0.0394	0.0283	0.0150

6 Conclusions

We have presented several algorithms to perform regularization in sequential learning tasks. The algorithms are based on adapting the noise processes sequentially. The experiments2[1] indicated that one of these algorithms, EKFQ, may lead to improved prediction results when either the data source is time varying or there is little a priori knowledge about how to tune the noise processes.

We showed that the hierarchical Bayesian inference methodology provides an elegant, unifying treatment of the sequential learning problem. We showed that distributed learning rates, adaptive noise parameters, and adaptive smoothing regularizers are mathematically equivalent. This result sheds light on many areas of the machine learning field. It places many diverse approaches to estimation and regularization within a unified framework.

There are many avenues for further research. These include deriving convergence bounds, implementing static and dynamic mixtures of models, implementing other model structures such as recurrent networks, and, testing the algorithms on additional problems.

7 Acknowledgments

We thank Analytical Mechanics Associates (Virginia, U.S.A.) for helpful assistance in providing some of the references. We are also very grateful to the referees for their valuable comments. J.F.G.F. is financially supported by two University of the Witwatersrand Merit Scholarships, a Foundation for Research Development Scholarship (South Africa), an ORS award (UK), and a Trinity College External Research Studentship (Cambridge).

References

Bar-Shalom, Y., & Li, X. R. (1993). *Estimation and tracking: Principles, techniques and software*. Boston: Artech House.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford: Clarendon Press.

Black, F., & Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of Political Economy*, *81*, 637–659.

Blom, H. A. P., & Bar-Shalom, Y. (1988). The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, *33*(8), 780–783.

Buntine, W. L., & Weigend, A. S. (1991). Bayesian back-propagation. *Complex Systems*, *5*, 603–643.

[1] Matlab demos for each experiment are available online at: http://www.cs.berkeley.edu/~jfgf/.

Candy, J. V. (1986). *Signal processing: The model based approach*. New York: McGraw-Hill.

de Freitas, J. F. G., & Niranjan, M., & Gee, A. H. (1997). *Hierarchichal Bayesian-Kalman models for regularization and ARD in sequential learning* (Tech. Rep. No. CUED/F-INFENG/TR 307). Cambridge: Cambridge University. Available online at: http://svr-www.eng.cam.ac.uk/~jfgf,

Gelb, A. (ed.). (1974). *Applied optimal estimation*. Cambridge, MA: MIT Press.

Girosi, F., Jones, M., & Poggio, T. (1995). Regularization theory and neural networks architectures. *Neural Computation*, *7*, 219–269.

Hull, J. C. (1997). *Options, futures, and other derivative* (3rd ed.). Englewood Cliffs, NJ: Prentice Hall.

Hutchinson, J. M., & Lo, A W., & Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. *Journal of Finance*, *49*(3), 851–889.

Ingber, L., & Wilson, J. K. (1999). Volatility of volatility of financial markets. *Journal of Mathematical and Computer Modeling*, *99*, 39–57.

Jacobs, R. A. (1988). Increased rates of convergence through learning rates adaptation. *Neural Networks*, *1*, 295–307.

Jazwinski, A. H. (1969). Adaptive filtering. *Automatica*, *5*, 475–485.

Jazwinski, A. H. (1970). *Stochastic processes and filtering theory*, Orlando, FL: Academic Press.

Jazwinski, A. H., & Bailie, A. E. (1967). *Adaptive filtering*, Tech. Rep. No. 67–6. Hampton, VA: Analytical Mechanics Associates.

Kadirkamanathan, V., & Kadirkamanathan, M. (1995). Recursive estimation of dynamic modular RBF networks. In D. S. Touretzky, M. C. Mozer, & M. E. Hasselmo (Eds.), *Advances in neural information processing systems, 8* (pp. 239–245). Cambridge, MA: MIT Press.

Li., X. R. & Bar-Shalom, Y. (1994). A recursive multiple model approach to noise identification. *IEEE Transactions on Aerospace and Electronic Systems*, *30*(3), 671–684.

Mackay, D. J. C. (1994). Bayesian nonlinear modeling for the prediction competition. *ASHRAE Transactions Pt. 2* (Vol. 100). Atlanta, GA.

Mackay, D. J. C. (1995). Probable networks and plausible predictions—a review of practical Bayesian methods for supervized neural networks. *Network-Computation in Neural Systems*, *6*, 469–505.

Mackay, D. J. C. (1996). Hyperparameters: Optimize or integrate out? In G. Heidbreder (Ed.), *Fundamental theories of physics* (pp. 43–59). Norwell, MA: Kluwer.

Mehra, R. K. (1970). On the identification of variances and adaptive Kalman filtering. *IEEE Transactions on Automatic Control*, *AC-15*(2), 175–184.

Mehra, R. K. (1971). On-line identification of linear dynamic systems with applications to Kalman filtering. *IEEE Transactions on Automatic Control*, *AC-16*(1), 12–21.

Mehra, R. K. (1972). Approaches to adaptive filtering. *IEEE Transactions on Automatic Control*, *AC-17*, 693–698.

Myers, K. A., & Tapley, B. D. (1976). Adaptive sequential estimation of unknown noise statistics. *IEEE Transactions on Automatic Control*, *AC-21*, 520–523.

Niranjan, M. (1996). Sequential tracking in pricing financial options using model based and neural network approaches. In M. C. Mozer, M. I. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems* (pp. 960–966). Cambridge, MA: MIT Press.

Puskorius, G. V., & Feldkamp, L. A. (1991). Decoupled extended Kalman filter training of feedforward Layered networks. In *International Joint Conference on Neural Networks*, Seattle (pp. 307–312).

Ruck, D. W., Rogers, S. K., Kabrisky, M., Maybeck, P. S., & Oxley, M. E. (1992). Comparative analysis of backpropagation and the extended Kalman filter for training multilayer perceptrons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *14*(6), 686–690.

Schottky, B., & Saad, D. (1999). Statistical mechanics of EKF learning in neural networks. *Journal of Physics A*, *32*, 1605–1621.

Shah, S., Palmieri, F., & Datum, M. (1992). Optimal filtering algorithms for fast learning in feedforward neural networks. *Neural Networks*, *5*, 779–787.

Shephard, N. (1996). Statistical aspects of ARCH and stochastic volatility. In D. P. Cox, O. E. Barndorff-Nielson, & D. V. Hinkley (Eds), *Time series models in econometrics, finance and other fields*. London: Chapman and Hall (pp. 1–67).

Singhal, S., & Wu, L. (1988). Training multilayer perceptrons with the extended Kalman algorithm. In D. S. Touretzky (Ed.), *Advances in neural information processing systems*, *1* (pp. 133–140), San Mateo, CA: Morgan Kaufmann.

Sutton, R. S. (1992). Adapting bias by gradient descent: An incremental version of delta-bar-delta. *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp. 171–176). Cambridge, MA: MIT Press.

Sutton, R. S. (1992). Gain adaptation beats least squares? *Proceedings of the Seventh Yale Workshop on Adaptive Learning Systems* (pp. 161–166).

Tenney, R. R., Hebbert, R. S., & Sandell, N. S. (1977). A tracking filter for maneuvering sources. *IEEE Transactions on Automatic Control*, *22*, 246–251.

Williams, P. M. (1995). Bayesian regularization and pruning using a Laplace prior. *Neural Computation*, *7*(1), 117–143.

Wolpert, D. H. (1993). On the use of evidence in neural networks. In S. J. Hanson, J. D. Cowan, & C. L. Giles (Eds.), *Advances in neural information processing systems*, *5* (pp. 539–546). San Mateo, CA: Morgan Kaufmann.

Received March 31, 1998; accepted March 12, 1999.