CAMBRIDGE UNIVERSITY

ENGINEERING DEPARTMENT

HIERARCHICAL BAYESIAN-KALMAN MODELS FOR REGULARISATION AND ARD IN SEQUENTIAL LEARNING

JFG de Freitas, M Niranjan and AH Gee

CUED/F-INFENG/TR 307

November 10, 1998

Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
England

E-mail: `jfgf@eng.cam.ac.uk`
URL: `http://www-svr.eng.cam.ac.uk/~jfgf`

Abstract

In this paper, we show that a hierarchical Bayesian modelling approach to sequential learning leads to many interesting attributes such as regularisation and automatic relevance determination. We identify three inference levels within this hierarchy, namely model selection, parameter estimation and noise estimation. In environments where data arrives sequentially, techniques such as cross-validation to achieve regularisation or model selection are not possible. The Bayesian approach, with extended Kalman filtering at the parameter estimation level, allows for regularisation within a minimum variance framework. A multi-layer perceptron is used to generate the extended Kalman filter nonlinear measurements mapping. We describe several algorithms at the noise estimation level, which allow us to implement adaptive regularisation and automatic relevance determination of model inputs and basis functions. An important contribution of this paper is to show the theoretical links between adaptive noise estimation in extended Kalman filtering, multiple adaptive learning rates and multiple smoothing regularisation coefficients.

Contents

1 Introduction **1**

2 State Space Models, Regularisation and Bayesian Inference **2**

3 Hierarchical Bayesian Sequential Modelling **4**

4 Parameter Estimation **5**
4.1 Linear-Gaussian Estimation 6
4.2 The Extended Kalman Filter 7
4.3 Training MLPs with the EKF 8

5 Noise Estimation and Regularisation **8**
5.1 Adaptive Distributed Learning Rates and Kalman Filtering 9
5.2 Sequential Bayesian Regularisation with Weight Decay Priors 11
5.3 Sequential Evidence Maximisation with Sequentially Updated Priors 13
5.3.1 Scalar process noise estimation 14
5.3.2 Scalar measurement noise estimation 16
5.3.3 Multiple noise hyper-parameters estimation 16

6 Automatic Relevance Determination **17**

7 Experiments **18**
7.1 Experiment 1: Comparison Between the Various Noise Estimation Methods . . 18
7.2 Experiment 2: Sequential Evidence Maximisation with Sequentially Updated Priors 20
7.3 Experiment 3: Automatic Relevance Determination 21
7.4 Application: Pricing Financial Options 22

8 Conclusions **24**

9 Acknowledgements **25**

A Bayesian Derivation of the Kalman Filter **25**
A.1 Prior Gaussian Density Function 25
A.2 Evidence Gaussian Density Function 26
A.3 Likelihood Gaussian Density Function 26
A.4 Posterior Gaussian Density Function 26

B Computing Derivatives for the Jacobian Matrix **27**

1 Introduction

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

Although “different” regularisation techniques are being spawned at an almost alarming rate in the machine learning literature, they are all based on the same assumption. That is, they all assume that the process generating the data obeys certain smoothness constraints. In other words, for small changes in the model input data, we expect small variations in the outputs. Regularisation is the simplest and yet one of the most useful forms of incorporating *a priori* knowledge (smoothness) in model selection.

In addition, most of the proposed regularisation techniques have been targeted at scenarios where the data can be processed in batches. Regularisation in sequential environments has not enjoyed much attention. When data arrives sequentially, conventional techniques such as cross validation for regularisation or model selection, and bootstrapping to deal with model uncertainty, are inapplicable. In this paper, we show how, starting from a Bayesian derivation of the extended Kalman Filter (EKF), several ideas can be developed to deal with regularisation, model selection and automatic relevance determination (ARD) in sequential environments.

One of the main purposes of this report is to show several interesting mathematical correlations between the problems of adaptive filtering, regularised error functions and adaptive learning rates. In the late sixties, Jazwinski (Jazwinski 1969, Jazwinski and Bailie 1967) proposed an algorithm for adaptive Kalman filtering based on the maximisation of the probability density function of the new data given all the past data (evidence probability density function). His algorithm employed adaptive noise parameters. In the early nineties, Sutton (Sutton 1992a, Sutton 1992b) showed for linear networks that distributed adaptive learning rates can be used to improve conventional error back-propagation. We extend Sutton’s ideas to nonlinear neural networks and relate them to other learning paradigms. Also in the early nineties, Mackay (Mackay 1992) introduced a method for estimating multiple regularisation coefficients, previously known in the Bayesian literature, to the neural network field. His method was also based on maximising the evidence density function. In this work, we show that multiple adaptive regularisers, adaptive process noise parameters and adaptive learning rates are mathematically equivalent.

Intuitively, imagine we are trying to descend on a landscape with numerous peaks and troughs. If we want to reach a low trough in an efficient manner, we have to avoid the upper troughs without having to spend too much energy in doing so. Three options arise: we can descend efficiently by varying our speed (adaptive learning rates), by jumping while we descend (adaptive noise estimation) or by smoothing the whole landscape before we attempt to descend (smoothing error functions).

In this paper, we focus on regression tasks. Nonetheless, the results may be easily extended to online classification. In Section 2, we present an overview of the sequential learning problem. In particular, we discuss state space modelling, optimal Bayesian inference and the minimum variance estimation framework as a regularisation approach to sequential learning. Section 3 describes the sequential learning task within a three-level hierarchical Bayesian structure. The three levels of inference adopted correspond to a noise estimation level, a parameter estimation level and a model selection level. In Section 4, we propose a solution to the parameter estimation level based on the application of the extended Kalman filter to neural networks. Section 5 is devoted to the noise estimation level and the regularisation/tracking dilemma. It describes several algorithms for noise estimation and regularisation, including adaptive distributed learning rates, adaptive smoothing regularisers and adaptive noise estimation. Section 6 discusses the topic of automatic relevance determination of inputs and basis functions. Finally, we present our results in Section 7 and point out several areas for further research in Section 8.

2 State Space Models, Regularisation and Bayesian Inference

To study the many sequential processes manifested in the real world, we need to create abstractions or models that capture the essence of these processes. State space models provide a suitable representation:

$$\mathbf{w}_{k+1} = \mathbf{f}_k(\mathbf{w}_k) + \mathbf{d}_k \tag{1}$$

$$\mathbf{y}_k = \mathbf{g}_k(\mathbf{w}_k, \mathbf{x}_k) + \mathbf{v}_k \tag{2}$$

where k denotes the discrete time index. The output measurements of the system ($\mathbf{y}_k \in \Re^m$) depend on a nonlinear, multivariate, time-varying function of the system inputs ($\mathbf{x}_k \in \Re^d$) and a set of states ($\mathbf{w}_k \in \Re^q$). In this work we assume that the states correspond to the model parameters. However, it is possible to incorporate other variables, for example the model outputs, into the state vector.

The measurements nonlinear mapping $\mathbf{g}_k(.)$ is approximated by a multi-layer perceptron (MLP) whose weights are the model parameters $\mathbf{w}$. Nonetheless, 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 in our case we model as zero mean, uncorrelated noise with covariance R_k. It is possible to extend the work to other noise models such as correlated and coloured noise.

We model the evolution of the model parameters by assuming that they depend on a deterministic component $\mathbf{f}_k(\mathbf{w}_k)$ and a stochastic component $\mathbf{d}_k$. The process noise $\mathbf{d}_k$ may represent our uncertainty on how the parameters evolve, modelling errors or unknown inputs such as target manoeuvres. We assume the process noise to be zero mean with adaptive covariance Q_k. In addition, we assume no knowledge of the drift function $\mathbf{f}_k(.)$, that is the parameters are generated by a first order Markov process $\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{d}_k$. In certain applications, such as image tracking (Reynard, Wildenberg, Blake and Marchant 1996) and speech enhancement (Niranjan, Cox and Hingorani 1994), we may know the equations governing the evolution of a subset of the states. In such scenarios, the drift function may be modelled for that particular subset of states, while the remaining states are assumed to obey a first order Markov process.

Note that because the covariance of the process noise $\mathbf{d}_k$ is estimated adaptively, the evolution of the states may be described by a nonlinear trajectory. This way, we avoid the problem of having to estimate $\mathbf{f}_k(\mathbf{w}_k)$. We have proven that the drift function may be estimated via extended Kalman smoothing and the EM algorithm (de Freitas, Niranjan and Gee 1998). However, the approach is only applicable to stationary environments. In this paper, we favour the approach of estimating the noise covariances because it will lead us to an elegant framework for regularisation and automatic relevance determination in sequential learning.

The sequential learning problem involves estimating the model parameters $\hat{\mathbf{w}}_k$, estimating noise models and selecting the right model on the basis of a set of past measurements $Y_k = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_k\}$. The problem of estimating $\mathbf{w}_k$ given Y_τ is called the smoothing problem if $k < \tau$; the filtering problem if $k = \tau$; or the prediction problem if $k > \tau$ (Gelb 1974, Jazwinski

1970). In the filtering problem, the estimate $\hat{\mathbf{w}}_k$ can be used to predict future values of the output.

In non-stationary environments we should favour the filtering approach to generate models of the sequential process being studied. In stationary environments, however, we only require to train the model until its performance is acceptable and, subsequently, we can let the model operate without further training. In the latter case, it needs to be emphasised that models for describing dynamical systems should be structured so as to account for time-dependent behaviour. Examples of models that conform to this requirement include ARMA models, recurrent networks and networks with tapped delay lines. In stationary environments, the smoothing approach, with forward and backward filtering, produces better estimates than plain forward filtering.

For optimality reasons, we want $\hat{\mathbf{w}}_k$ to be an unbiased, minimum variance and consistent estimate (Gelb 1974), where:

- An unbiased estimate is one whose expected value is equal to the quantity being estimated.
- A minimum variance (unbiased) estimate is one that has its variance less than or equal to that of any other unbiased estimator.
- A consistent estimate is one that converges to the true value of the quantity being estimated as the number of measurements increases.

The minimum variance estimation framework leads to smooth estimates for the parameters and model outputs. Accordingly, it constitutes a regularisation scheme for sequential learning.

The uncertainty in the model parameters and measurements leads naturally to a probabilistic treatment of the problem. In addition, the sequential learning nature of the problem motivates a Bayesian framework whereby our prior belief about the unknown quantities is improved each time new data arrives. That is, the unknown quantities are described by probability density functions, whose widths indicate the range of values that are consistent with the prior information and the new data (Jaynes 1986).

The conditional probability density function of $\mathbf{w}_k$ given Y_k ($p(\mathbf{w}_k|Y_k)$) constitutes the complete solution of the estimation problem (Bar-Shalom and Li 1993, Ho and Lee 1964, Jazwinski 1970). This is simply because $p(\mathbf{w}_k|Y_k)$ embodies all the statistical information about $\mathbf{w}_k$ given the measurements Y_k and the initial condition $\mathbf{w}_0$. The estimate $\hat{\mathbf{w}}_k$ can be computed from $p(\mathbf{w}_k|Y_k)$ according to any of the following criteria:

MAP estimation : Maximise the probability such that the solution is the largest mode (peak) of $p(\mathbf{w}_k|Y_k)$. For uniform fixed priors, the resulting solution is the maximum likelihood estimate.

Minimum variance estimation : Minimise the integral error $\int \|\mathbf{w}_k - \hat{\mathbf{w}}_k\|^2 p(\mathbf{w}_k|Y_k) d\mathbf{w}_k$ so that the estimate corresponds to the expected value or conditional mean $\mathbf{E}[\mathbf{w}_k|Y_k]$.

Minimax estimation : Minimise the maximum of $|\mathbf{w} - \hat{\mathbf{w}}|$ so that the estimate is the median of $p(\mathbf{w}_k|Y_k)$.

The criteria are illustrated in Figure 7. As discussed above, the minimum variance estimate is the quantity of interest in our approach.

The Bayesian solution to the optimal estimation problem is given by (Ho and Lee 1964):

$$\begin{aligned} p(\mathbf{w}_{k+1}|Y_{k+1}) &= \frac{p(\mathbf{w}_{k+1}, \mathbf{y}_{k+1}|Y_k)}{p(\mathbf{y}_{k+1}|Y_k)} \\ &= \frac{\int p(\mathbf{y}_{k+1}|Y_k, \mathbf{w}_{k+1}) p(\mathbf{w}_{k+1}|\mathbf{w}_k) p(\mathbf{w}_k|Y_k) d\mathbf{w}_k}{\int\int p(\mathbf{y}_{k+1}|Y_k, \mathbf{w}_{k+1}) p(\mathbf{w}_{k+1}|\mathbf{w}_k) p(\mathbf{w}_k|Y_k) d\mathbf{w}_{k+1} d\mathbf{w}_k} \end{aligned} \quad (3)$$

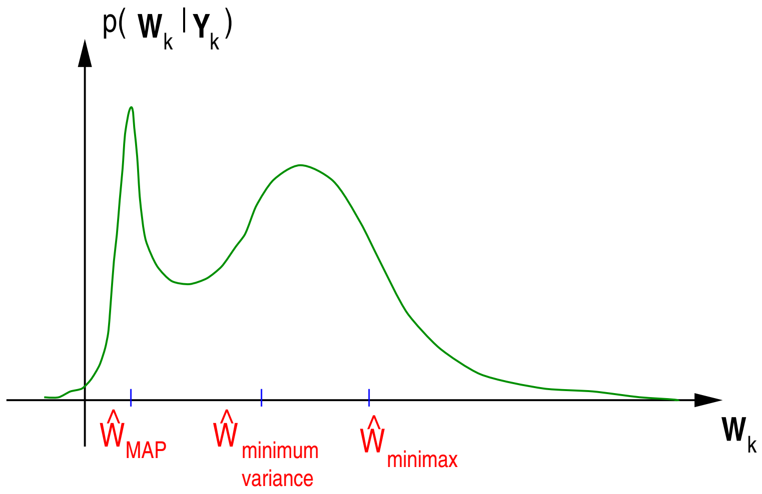

Figure 1: Estimation criteria based on the conditional density function of the model parameters.

where the integrals run over the parameter space.

The functional integral difference equation governing the evolution of the posterior density function (equation (3)) is not suitable for practical implementation (Bar-Shalom and Li 1993, Jazwinski 1970). It involves propagating a quantity (the posterior density function) that cannot be described by a finite number of parameters.

Two alternative routes have been proposed to surmount this problem, namely sampling techniques (Black and Reed 1996, Carter and Kohn 1994, Isard and Blake 1996) and Gaussian approximations (Bar-Shalom and Li 1993, Ghahramani and Jordan 1995). Sampling methods require extensive computational resources. Gaussian approximations are less computationally demanding, but suffer from several drawbacks, especially when the probability density function being approximated has a large number of modes.

In this particular work we adopt the use of Gaussian approximations because they provide an elegant unified treatment of the problems of regularisation, sequential learning and model selection. In addition, several successful applications of Gaussian approximations for neural networks in the automotive industry have been registered (Puskorius and Feldkamp 1991, Puskorius, Feldkamp and Davis 1996).

As mentioned above, the implementation of the optimal estimator involves a functional recursion and is therefore of limited feasibility. The situation for linear-Gaussian state space models is vastly simpler. There the mean and covariance are sufficient statistics for describing the Gaussian posterior density function. In addition, the state can be computed in an optimal fashion with the Kalman filter (Anderson and Moore 1979, Bar-Shalom and Li 1993, Gelb 1974, Jazwinski 1970). This motivates an approach based on Gaussian approximations that employs linear approximations about the current estimates. These two approximations lead to the formulation of the well known extended Kalman filter (EKF).

We show in this paper that, within a hierarchical Bayesian inference framework, the EKF constitutes the solution to the parameter estimation level. It, however, does not constitute the solution to the entire inference process.

3 Hierarchical Bayesian Sequential Modelling

To represent a particular sequential process, we need to infer the model parameters, noise covariances and the likelihood of a particular model. This inference 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 adopt the philosophical stance of representing the state of knowledge about the parameters, noise covariances and model choice within a hierarchical Bayesian framework. The notion of hierarchical probabilistic modelling and inference permeates throughout most areas of human endeavour, including physics, philosophy and mathematics.

We propose the following inference levels:

Level 1: Parameter estimation

$$\mathrm{p}(\mathbf{w}_{\mathrm{k+1}}|\mathrm{Y}_{\mathrm{k+1}}, \mathrm{M_j}, \mathrm{R_k}, \mathrm{Q_k}) = \frac{\mathrm{p}(\mathbf{y}_{\mathrm{k+1}}|\mathbf{w}_{\mathrm{k+1}}, \mathrm{M_j}, \mathrm{R_k}, \mathrm{Q_k})}{\mathrm{p}(\mathbf{y}_{\mathrm{k+1}}|\mathrm{Y_k}, \mathrm{M_j}, \mathrm{R_k}, \mathrm{Q_k})}\mathrm{p}(\mathbf{w}_{\mathrm{k+1}}|\mathrm{Y_k}, \mathrm{M_j}, \mathrm{R_k}, \mathrm{Q_k}) \tag{4}$$

Level 2: Noise estimation

$$\mathrm{p}(R_k, Q_k|Y_{k+1}) = \frac{\mathrm{p}(\mathbf{y}_{\mathrm{k+1}}|\mathrm{Y_k}, \mathrm{M_j}, \mathrm{R_k}, \mathrm{Q_k})}{\mathrm{p}(\mathbf{y}_{\mathrm{k+1}}|\mathrm{Y_k}, \mathrm{M_j})}\mathrm{p}(R_k, Q_k|Y_k, M_j) \tag{5}$$

Level 3: Model selection

$$\mathrm{p}(M_j|Y_{k+1}) = \frac{\mathrm{p}(\mathbf{y}_{\mathrm{k+1}}|\mathrm{Y_k}, \mathrm{M_j})}{\mathrm{p}(\mathbf{y}_{\mathrm{k+1}}|\mathrm{Y_k})}\mathrm{p}(M_j|Y_k) \tag{6}$$

where M_j represents the j-th model. It should be noticed that the likelihood function at a particular level constitutes the evidence function at the next higher level. Therefore, by maximising the evidence function in the parameter estimation level, we are, in fact, maximising the likelihood of the noise covariances R_k and Q_k as the new data arrives. This result shall play an important role when we devise methods for estimating the noise covariances.

At the parameter estimation level, we shall apply the EKF algorithm to estimate the weights of a multi-layer perceptron. The EKF, however, requires knowledge of the noise covariances. To overcome this difficulty, in Section 5, we present techniques for estimating these covariances in slowly changing non-stationary environments. In environments where the noise statistics change rapidly, we shall favour the implementation of dynamic mixtures of models with different noise covariances (Li and Bar-Shalom 1994). This remark brings us to the topic of model selection. Model selection can be formulated in two ways; dynamic model selection and static model selection.

In static model selection, the model assumed to be valid throughout the entire process is one of r hypothesised models. That is, we start with r models and compute which model describes the sequential process most accurately. The remaining models are, subsequently, discarded. In dynamic model selection, one particular model out of a set of r operating models is selected during each estimation step. Dynamic mixtures of models are far more general than static mixtures of models. However, in stationary environments, static mixtures are obviously more adequate. Dynamic mixtures of models correspond to a generalised version of hidden Markov models (Li and Bar-Shalom 1994). Mixtures of models are not covered here.

4 Parameter Estimation

In this section, we tackle the problem of estimating the model parameters. We start with a Bayesian derivation of the Kalman filter in the linear-Gaussian case and extend the approach to the nonlinear-Gaussian scenario. Subsequently, the application of the EKF algorithm to train multi-layer perceptrons (MLPs) is expounded.

4.1 Linear-Gaussian Estimation

If we simplify our state space representation (equations (1) and (2)) to the following linear Gauss-Markov process

$$
\begin{aligned}
\mathbf{w}_{k+1} &= \mathbf{w}_k + \mathbf{d}_k & (7)\\
\mathbf{y}_k &= H_k \mathbf{w}_k + \mathbf{v}_k, & (8)
\end{aligned}
$$

it is possible to apply Bayes rule to estimate the posterior density function for the model parameters after the new data arrives.

Since the prior, evidence and likelihood are Gaussian and the system is linear, the posterior will also be Gaussian. This implies that in order to update the estimator we only need to know two quantities, namely the mean and the covariance of the conditional density function. This follows from the fact that the mean and covariance are sufficient statistics to describe a Gaussian process. An additional consequence of having a Gaussian posterior, is that the minimax, minimum variance and maximum *a posteriori* estimates will be identical.

Under the assumptions that the state space model noise processes 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 (Anderson and Moore 1979, Candy 1986, Ho and Lee 1964) (see also Appendix A):

$$\text{Prior} = \mathrm{p}(\mathbf{w}_{k+1}|\mathrm{Y}_k, \mathrm{M}_j, \mathrm{R}_k, \mathrm{Q}_k) \sim \mathcal{N}(\hat{\mathbf{w}}_k, \mathrm{P}_k + \mathrm{Q}_k) \tag{9}$$

$$\text{Evidence} = \mathrm{p}(\mathbf{y}_{k+1}|\mathrm{Y}_k, \mathrm{M}_j, \mathrm{R}_k, \mathrm{Q}_k) \sim \mathcal{N}(\mathrm{H}_{k+1}\hat{\mathbf{w}}_k, \mathrm{H}_{k+1}(\mathrm{P}_k + \mathrm{Q}_k)\mathrm{H}_{k+1}^{\mathrm{T}} + \mathrm{R}_{k+1}) \tag{10}$$

$$\text{Likelihood} = \mathrm{p}(\mathbf{y}_{k+1}|\mathbf{w}_{k+1}, \mathrm{M}_j, \mathrm{R}_k, \mathrm{Q}_k) \sim \mathcal{N}(\mathrm{H}_{k+1}\mathbf{w}_{k+1}, \mathrm{R}_{k+1}) \tag{11}$$

where P corresponds to the covariance of the model parameters and the symbol T denotes the transpose of a matrix.

Substituting equations (9), (10) and (11) into (4), yields the optimal Bayes estimate:

$$\mathrm{p}(\mathbf{w}_{k+1}|\mathrm{Y}_{k+1}, \mathrm{M}_j, \mathrm{R}_k, \mathrm{Q}_k) = \mathrm{A}_{k+1} \exp\big(-\frac{1}{2}(\mathbf{w}_{k+1} - \hat{\mathbf{w}}_{k+1})\mathrm{P}_{k+1}^{-1}(\mathbf{w}_{k+1} - \hat{\mathbf{w}}_{k+1})\big) \tag{12}$$

where the coefficients A_{k+1} are represented by the following expression:

$$A_{k+1} = \frac{|H_{k+1}(P_k + Q_k)H_{k+1}^T + R_{k+1}|^{1/2}}{(2\pi)^{q/2}|R_{k+1}|^{1/2}|P_k + Q_k|^{1/2}}$$

and $\hat{\mathbf{w}}_{k+1}$ and P_{k+1} are given by:

$$
\begin{aligned}
\hat{\mathbf{w}}_{k+1} &= \hat{\mathbf{w}}_k + K_{k+1}(\mathbf{y}_{k+1} - \mathrm{H}_{k+1}\hat{\mathbf{w}}_k) & (13)\\
P_{k+1} &= P_k + Q_k - K_{k+1}H_{k+1}(P_k + Q_k) & (14)
\end{aligned}
$$

where K_k is known as the Kalman gain:

$$K_{k+1} = (P_k + Q_k)H_{k+1}^T[R_{k+1} + H_{k+1}(P_k + Q_k)H_{k+1}^T]^{-1} \tag{15}$$

Equations (13), (14) and (15) correspond exactly to the Kalman filter equations (Bar-Shalom and Li 1993, Gelb 1974, Ho and Lee 1964). Alternatively, the Kalman filter equations may be derived by adopting the minimum variance approach. That is, by minimising the following cost functional (Gelb 1974):

$$J_k = \text{trace } (P_k) \tag{16}$$

The Kalman filter algorithm may be easily implemented by computing K, $\hat{\mathbf{w}}$ and P recursively. This is shown in the predictor-corrector form in Figure 2.

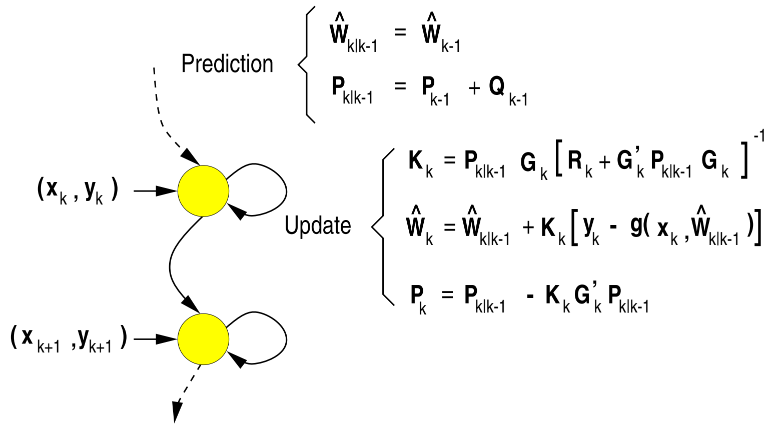

Figure 2: Extended Kalman filter predictor-corrector representation.

4.2 The Extended Kalman Filter

In view of the simplicity and versatility of the Kalman filter, it would be desirable to have a framework for nonlinear estimation similar to the one for linear-Gaussian estimation. The extended Kalman filter (EKF) constitutes an attempt in this direction (Bar-Shalom and Li 1993, Gelb 1974). The EKF is a minimum variance estimator based on a Taylor series expansion of the nonlinear function $\mathbf{g}(\mathbf{w})$ around the previous estimate. That is,

$$\mathbf{g}(\mathbf{w}) = \mathbf{g}(\hat{\mathbf{w}}) + \frac{\partial \mathbf{g}}{\partial \mathbf{w}} \Big|_{(\mathbf{w}=\hat{\mathbf{w}})} (\mathbf{w} - \hat{\mathbf{w}}) + \cdots$$

The EKF equations for a linear expansion are given by:

$$\begin{aligned} K_{k+1} &= (P_k + Q_k) G_{k+1} [R_k + G_{k+1}^T (P_k + Q_k) G_{k+1}]^{-1} && (17) \\ \hat{\mathbf{w}}_{k+1} &= \hat{\mathbf{w}}_k + K_{k+1} (\mathbf{y}_{\mathrm{k+1}} - \mathbf{g}_{\mathrm{k+1}} (\hat{\mathbf{w}}_{\mathrm{k}}, \mathbf{x}_{\mathrm{k}})) && (18) \\ P_{k+1} &= P_k + Q_k - K_{k+1} G_{k+1}^T (P_k + Q_k) && (19) \end{aligned}$$

In the general multiple input, multiple output (MIMO) case, $\mathbf{g} \in \Re^{\mathrm{m}}$ is a vector function and G represents the Jacobian matrix:

$$G = \frac{\partial \mathbf{g}}{\partial \mathbf{w}} \Big|_{(\mathbf{w}=\hat{\mathbf{w}})} = \begin{bmatrix} \frac{\partial \mathbf{g}_1}{\partial \mathbf{w}_1} & \frac{\partial \mathbf{g}_2}{\partial \mathbf{w}_1} & \cdots & \frac{\partial \mathbf{g}_{\mathrm{m}}}{\partial \mathbf{w}_1} \\ \frac{\partial \mathbf{g}_1}{\partial \mathbf{w}_2} & & & \\ \vdots & & & \vdots \\ \frac{\partial \mathbf{g}_1}{\partial \mathbf{w}_{\mathrm{q}}} & \cdots & & \frac{\partial \mathbf{g}_{\mathrm{m}}}{\partial \mathbf{w}_{\mathrm{q}}} \end{bmatrix}^T$$

Since the EKF is a suboptimal estimator based on linearisation 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 and Li 1993).

The EKF provides a minimum variance Gaussian approximation to the posterior probability density function. In many cases, $\mathrm{p}(\mathbf{w}_{\mathrm{k}}|\mathrm{Y}_{\mathrm{k}})$ is a multi-modal (several peaks) function. In this scenario, it is possible to use a committee of Kalman filters, where each individual filter approximates a particular mode, to produce a more accurate approximation (Bar-Shalom and Li 1993, Blom and Bar-Shalom 1988, Kadirkamanathan and Kadirkamanathan 1995, Kadirkamanathan and Kadirkamanathan 1996). The parameter covariances of the individual estimators may be used to determine the contribution of each estimator to the committee. In addition, the parameter covariances serve the purpose of placing confidence intervals on the output prediction.

The immediate availability of confidence intervals and of mixing coefficients, required to generate mixtures of models, has motivated us to train neural networks with the EKF algorithm.

4.3 Training MLPs with the EKF

One of the earliest implementations of EKF trained MLPs is due to Singhal and Wu (Singhal and Wu 1988). In their method, the network weights are grouped into a single vector $\mathbf{w}$ that is updated in accordance with the EKF equations. The entries of the Jacobian matrix are calculated by back-propagating the m output values $\{y_1(t), y_2(t), \cdots, y_m(t)\}$ through the network. An example of how to do this for a simple MLP is presented in Appendix B.

The algorithm proposed by Singhal and Wu requires a considerable computational effort. The complexity is of the order mq^2 multiplications per estimation step. Shah, Palmieri and Datum (Shah, Palmieri and Datum 1992) and Puskorius and Feldkamp (Puskorius and Feldkamp 1991) have proposed strategies for decoupling the global EKF estimation algorithm into local EKF estimation sub-problems. For example, they suggest that the weights of each neuron could be updated independently. The assumption in the local updating strategies is that the weights are decoupled and, consequently, P is a block-diagonal matrix.

The EKF is an improvement over conventional MLP estimation techniques, such as online back-propagation, in that it makes use of second order statistics (covariances). These statistics are essential for placing error bars on the predictions and for combining separate networks into committees of networks. As a matter of interest, it has been proven elsewhere that the back-propagation algorithm is simply a degenerate of the EKF algorithm (Ruck, Rogers, Kabrisky, Maybeck and Oxley 1992).

However, the EKF algorithm for training MLPs suffers from serious difficulties, namely choosing the initial conditions $(\mathbf{w}_0, \mathrm{P}_0)$ and the noise covariance matrices R and Q. In our work we place more emphasis on the problem of automatically estimating the noise covariances. To initialise the weights and their covariances we make use of a maximum likelihood method (Levenberg-Marquardt optimisation (More 1977)). This prior is subsequently improved with the EKF recurrent algorithm.

5 Noise Estimation and Regularisation

A well known limitation of the Kalman-Bucy filter (Kalman and Bucy 1961) and the extended Kalman filter, is the assumption of known *a priori* statistics to describe the measurement and process noise. 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, Li and Bar-Shalom 1994, Mehra 1970, Mehra 1971, Mehra 1972, Myers and Tapley 1976, Tenney, Hebbert and Sandell 1977). Mehra (Mehra 1972) and Li and Bar-Shalom (Li and Bar-Shalom 1994) have written brief surveys on this topic. In our work we make use of these results, from the adaptive estimation field, to improve the existing algorithms for training neural networks with the EKF algorithm (Kadirkamanathan and Niranjan 1992, Kadirkamanathan and Niranjan 1993, Puskorius and Feldkamp 1991, Puskorius and Feldkamp 1994, Puskorius et al. 1996, Shah et al. 1992, Singhal and Wu 1988, Williams 1992). We achieve this in a principled manner by adhering to a hierarchical Bayesian methodology. In doing so, we are able to place some of the heuristic algorithms in the estimation field within a proper theoretical framework. Furthermore, this framework serves to unify important results concerning regularisation and the training of neural networks.

It is important to note that algorithms for estimating the noise covariances within the EKF algorithm 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 (13) and (15)). 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 and 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 regularisation mechanism to control the smoothness of the prediction.

It is important to keep in mind that when designing algorithms for updating the noise covariances, we should beware of not degrading the performance of the parameter estimation algorithm. This problem is also encountered in Bayesian methods for inverse problems, for example image reconstruction (Sibisi 1989) and neural networks (Mackay 1992, Mackay 1994b), where the regularisation coefficients are computed automatically from batches of data. An underlying requirement for updating the noise covariances without degrading the performance of the parameter estimation algorithm, is that the convergence of the parameter estimator to an acceptable solution should be faster than the variation of the noise statistics (Jazwinski 1969, Li and Bar-Shalom 1994). As mentioned in Section 3, for rapidly changing noise statistics mixtures of models with different noise statistics should be employed.

Subsequently, we present three alternative methods for updating the noise covariances, namely multiple back-propagation, sequential evidence maximisation with weight decay priors and sequential evidence maximisation with updated priors. The multiple back-propagation method allows us to establish a theoretical connection between back-propagation with adaptive distributed learning rates and Kalman filtering. It has the advantage of being a computationally efficient algorithm for improving the computational speed of Kalman filtering.

The evidence maximisation technique with weight decay priors will serve to illuminate the relationship between smoothing regularisers and adaptive noise covariances. Finally, the sequential evidence maximisation with updated priors will be appropriate for illustrating various issues including the regularisation/tracking dilemma and automatic relevance determination of network inputs and basis functions.

5.1 Adaptive Distributed Learning Rates and Kalman Filtering

The work of Richard Sutton with linear networks (Sutton 1992a, Sutton 1992b) sheds light on the relationship between adaptive distributed learning rates in gradient descent algorithms and adaptive Kalman filtering. To merely simplify the exposition, without loss of generality, we shall restrict our explanation of this topic to a linear network with a single neuron. Later, we extend the ideas to nonlinear networks. Consider the following linear mapping:

$$y_k = \sum_i x_k^{(i)} w_k^{(i)} + v_k$$

where $x_k^{(i)}$ indicates the i-th input to the network at the k-th estimation step, $w_k^{(i)}$ the network weights and v_k a white noise sequence.

In its simplest form, gradient descent updates are computed according to the following equation:

$$\hat{w}_{k+1}^{(i)} = \hat{w}_k^{(i)} + \eta(y_k - \hat{y}_k)x_k^{(i)}$$

where η represents a learning rate parameter. An obvious improvement on this algorithm is to adapt the learning rates for each weight (Jacobs 1988, Sutton 1992a). That is,

$$\hat{w}_{k+1}^{(i)} = \hat{w}_k^{(i)} + \eta_k^{(i)}(y_k - \hat{y}_k)x_k^{(i)}$$

If we compare the above equation with the Kalman filter equation for updating the same network

$$\hat{\mathbf{w}}_{k+1} = \hat{\mathbf{w}}_k + \frac{(P_k + Q_k)\mathbf{x}_k^T}{R_{k+1} + \mathbf{x}_k(P_k + Q_k)\mathbf{x}_k^T}(y_k - \hat{y}_k),$$

we find that the adaptive distributed learning rate corresponds to the following adaptive distributed term in the Kalman filtering framework:

$$\eta = \frac{(\mathrm{P_k} + \mathrm{Q_k})}{\mathrm{R_{k+1}} + \mathbf{x}_\mathrm{k}(\mathrm{P_k} + \mathrm{Q_k})\mathbf{x}_\mathrm{k}^\mathrm{T}}$$

The above result indicates that the problem of sequentially updating the learning rates in gradient descent algorithms and the problem of updating the process noise covariances in Kalman filtering are equivalent.

Sutton (Sutton 1992b) has proposed a gradient descent approach for updating the Kalman filter equations. The aim of his method was to reduce the computational time at the expense of a small deterioration in the performance of the estimator. Another important aspect of Sutton's algorithm is that it circumvents the problem of choosing the process noise covariance Q. To understand how this is done, we need to write the Kalman filter equations in the following format (Gelb 1974):

$$\begin{aligned} K_{k+1} &= P_k H_{k+1}^T [R_{k+1} + H_{k+1} P_k H_{k+1}^T]^{-1} \\ \hat{\mathbf{w}}_{k+1} &= \hat{\mathbf{w}}_k + K_{k+1}(y_{k+1} - H_{k+1}\hat{\mathbf{w}}_k) \\ P_{k+1} &= P_k + Q_k - K_{k+1} H_{k+1} P_k \end{aligned}$$

In this format, only the parameters' covariance update equation depends on Q. Sutton eliminates the problem of choosing Q by updating P with a variation of the least-mean-square rule (Jacobs 1988, Sutton 1992a). More specifically, Sutton's technique involves approximating P with a diagonal matrix, whose i-th diagonal entry is given by:

$$p_{ii} = \exp(\beta_i)$$

where β_i 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, lead to poor results.

To circumvent the problem of choosing the process noise covariance Q when training nonlinear neural networks, while at the same time increasing computational efficiency, it is possible to extend Sutton's algorithm to the nonlinear case. In doing so, the weights covariance matrix is approximated by a diagonal matrix with entries given by:

$$p_{qq} = \exp(\beta_q)$$

where β is updated by error back-propagation. That is

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

where the index i corresponds to the i-th neuron in the output layer, j to the j-th neuron in the hidden layer, d to the d-th 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 i-th 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 hyper-parameter. We have found, in practice, that choosing this hyper-parameter is easier than choosing the process noise covariance matrix.

In a similar fashion to the linear case, the Kalman gain K_k and the weights estimate $\hat{\mathbf{w}}_k$ are updated using the following extended Kalman filter equations:

$$\begin{aligned} K_{k+1} &= P_k G_{k+1}[R_{k+1} + G_{k+1}^T P_k G_{k+1}]^{-1} \\ \hat{\mathbf{w}}_{k+1} &= \hat{\mathbf{w}}_k + K_{k+1}(\mathbf{y}_{\mathrm{k+1}} - \mathbf{g}_{\mathrm{k+1}}(\hat{\mathbf{w}}_{\mathrm{k}}, \mathbf{x}_{\mathrm{k+1}})), \end{aligned}$$

while the weights covariance P is updated by back-propagation.

5.2 Sequential Bayesian Regularisation with Weight Decay Priors

A further improvement on the EKF algorithm for training MLPs would be to update R and Q automatically at each estimation step. This can be done by borrowing some ideas from the Bayesian estimation field. In particular, we shall attempt to link the work on Bayesian estimation for neural networks (Mackay 1992, Mackay 1994b) and inverse problems (Sibisi 1989) with the EKF estimation framework. This theoretical link should serve to enhance both methods. To pave the way for linking these methods, we shall briefly discuss David Mackay's Bayesian estimation approach to neural networks.

We begin by considering Bayes theorem at the parameter estimation level:

$$\mathrm{p}(\mathbf{w}|\mathrm{Y_k}) = \frac{\mathrm{p}(\mathrm{Y_k}|\mathbf{w})}{\mathrm{p}(\mathrm{Y_k})}\mathrm{p}(\mathbf{w})$$

Mackay expresses the prior and likelihood density functions in terms of the following Gaussian functions:

$$\mathrm{p}(\mathbf{w}) = \frac{1}{Z_w(\alpha)}\exp(-\alpha E_w) \tag{20}$$

$$\mathrm{p}(Y_k|\mathbf{w}) = \frac{1}{Z_D(\beta)}\exp(-\beta E_D) \tag{21}$$

where E_w represents a regularisation prior, E_D is an error function and Z_w and Z_D are normalisation factors. The hyper-parameters α and β control the variance of the prior distribution of weights and the variance of the measurement noise. α also plays the role of the regularisation coefficient as will be shown soon.

By choosing a weight decay prior given by

$$E_w = \frac{1}{2}||\mathbf{w}||^2 = \frac{1}{2}\sum_{\mathrm{i}=1}^{\mathrm{q}} \mathrm{w_i^2}$$

the prior density function becomes:

$$\mathrm{p}(\mathbf{w}) = \frac{1}{(2\pi)^{\mathrm{q}/2}\alpha^{-\mathrm{q}/2}}\exp\Big(-\frac{\alpha}{2}||\mathbf{w}||^2\Big) \tag{22}$$

Similarly, by choosing a Gaussian noise model, the likelihood function becomes:

$$\mathrm{p}(Y_k|\mathbf{w}) = \frac{1}{(2\pi)^{\mathrm{n}/2}\beta^{-\mathrm{n}/2}}\exp\Big(-\frac{\beta}{2}\sum_{\mathrm{k}=1}^{\mathrm{n}}(\mathbf{y}_\mathrm{k} - \hat{\mathbf{g}}_{\mathrm{n,q}}(\mathbf{w},\mathbf{x}_\mathrm{k}))^2\Big) \tag{23}$$

where $\hat{\mathbf{g}}_{n,q}(\mathbf{w}, \mathbf{x}_\mathrm{k})$ corresponds to the model prediction. We emphasise that this prediction depends on the number of samples and the model complexity.

Using equations (22) and (23) and taking into account that the evidence does not depend on the weights, the following posterior density function may be obtained:

$$\mathrm{p}(\mathbf{w}|\mathrm{Y_k}) = \frac{1}{\mathrm{Z_s}(\alpha,\beta)}\exp(-\alpha\mathrm{E_w} - \beta\mathrm{E_D}) = \frac{1}{\mathrm{Z_s}(\alpha,\beta)}\exp(-\mathrm{S}(\mathbf{w})) \tag{24}$$

For the prior and the likelihood of equations (23) and (24), $S(\mathbf{w})$ is given by

$$S(\mathbf{w}) = \frac{\alpha}{2}\|\mathbf{w}\|^2 + \frac{\beta}{2}\sum_{\mathrm{k}=1}^{\mathrm{n}}(\mathbf{y}_{\mathrm{k}} - \hat{\mathbf{g}}_{\mathrm{n,q}}(\mathbf{w},\mathbf{x}_{\mathrm{k}}))^2 \tag{25}$$

Mackay approximates the posterior density function with a Gaussian density function. This is done by applying a Taylor series expansion of $S(\mathbf{w})$ around a local minimum ($\mathbf{w}_{\mathrm{MP}}$) and retaining the series terms up to second order

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

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

$$\mathrm{p}(\mathbf{w}|\mathrm{Y}_{\mathrm{k}}) = \frac{1}{(2\pi)^{\mathrm{q}/2}|\mathrm{A}|^{-1/2}}\exp\Big(-\frac{1}{2}(\mathbf{w} - \mathbf{w}_{\mathrm{MP}})^{\mathrm{T}}\mathrm{A}(\mathbf{w} - \mathbf{w}_{\mathrm{MP}})\Big) \tag{26}$$

Maximising the posterior probability density function involves minimising the error function given by equation (25). Equation (25) is a particular case of a regularised error function. More generally

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

where ν is a positive parameter that serves to balance the tradeoff between smoothness and data approximation. A large value of ν places more importance on the smoothness of the model, while a small value of ν places more emphasis on fitting the data. The functional Ω penalises for excessive model complexity. In batch learning, the regularisation parameter is often obtained by cross-validation (Stone 1974, Stone 1978, Wahba and Wold 1969).

Several methods have been proposed for the design of the regularisation functional. In our work we shall focus on weight decay regularisers. As a matter of interest, Girosi, Jones and Poggio (Girosi, Jones and Poggio 1995) have proposed and alternative regularisation approach using a functional that clearly shows the relationship between regularisation and smoothness:

$$\Omega = \int_{\Re^d} \frac{|\tilde{\mathbf{g}}_{n,q}(\mathbf{s})|^2}{\tilde{H}(\mathbf{s})}\mathrm{d}\mathbf{s}$$

where the tildes indicate Fourier transforms and $1/\tilde{H}(\mathbf{s})$ is chosen to be a high-pass filter. In other words, the functional returns the high frequency components (oscillations) of the mapping. Therefore, a large value of ν simply indicates that any excessive oscillation will constitute a major contribution to the modelling error.

In Mackay's estimation framework, also known as the evidence framework, the parameters $\mathbf{w}$ are obtained by minimising equation (25), while the hyper-parameters α and β are obtained by maximising the evidence $\mathrm{p}(Y_k|\alpha,\beta)$ after approximating the posterior density function by a Gaussian function centred at $\mathbf{w}_{\mathrm{MP}}$. In doing so, the following recursive formulae for α and β are obtained:

$$\alpha_{k+1} = \frac{\gamma}{\sum_{i=1}^{q} w_i^2} \tag{27}$$

$$\beta_{k+1} = \frac{n-\gamma}{\sum_{k=1}^{n}(\mathbf{y}_{\mathrm{k}} - \hat{\mathbf{g}}_{\mathrm{n,q}}(\mathbf{w}_{\mathrm{k}},\mathbf{x}_{\mathrm{k}}))^2} \tag{28}$$

The quantity γ represents the effective number of parameters and is given by

$$\gamma = \sum_{i=1}^{q}\frac{\lambda_i}{\lambda_i + \alpha}$$

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.

Instead of adopting Mackay's evidence framework, it is possible to maximise the posterior density function by performing integrations over the hyper-parameters analytically (Buntine and Weigend 1991, Mackay 1994b, Williams 1995, Wolpert 1993). The latter approach is known as the MAP framework for α and β. The hyper-parameters 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:

$$\alpha_{k+1} = \frac{q}{\sum_{i=1}^{q} w_i^2} \tag{29}$$

$$\beta_{k+1} = \frac{n}{\sum_{k=1}^{n} (\mathbf{y}_\mathrm{k} - \hat{\mathbf{g}}_{\mathrm{n,q}}(\mathbf{w}_\mathrm{k}, \mathbf{x}_\mathrm{k}))^2} \tag{30}$$

Mackay (Mackay 1994b) has argued in favour of the evidence framework by stating that fitting a Gaussian around the MAP estimate, that is the peak of the posterior density function, does not represent the best approximation to the posterior density function. On the other hand, maximising the evidence after fitting a Gaussian to the posterior leads to a solution that is closer to the minimum variance estimate. In our work, we have adopted both methods for comparison purposes.

By comparing equations (9), (11) and (12) in the Kalman filtering framework with equations (22), (23) and (26), we can establish the following relations:

$$P = A^{-1} \tag{31}$$

$$Q = \alpha^{-1} I_q - A^{-1} \tag{32}$$

$$R = \beta^{-1} I_m \tag{33}$$

where I_q and I_m represent identity matrices of sizes q and m respectively.

Therefore, it is possible to update Q and R sequentially by expressing them in terms of the sequential updates of α and β. Implementing this idea is a straightforward task. It simply involves computing α and β at each estimation step using the recursive equations (27) and (28) or (29) and (30), and then substituting the answers into equations (32) and (33). As in the work of Mackay, we take care that the covariance hyperparameters remain positive by setting them to a small number each time they become negative. We believe that this thresh-holding is one of the problems with this approach. A moving window may be implemented to estimate β. The size of the window is a parameter that requires tuning.

Equations (32) and (33) are extremely important in that they reveal the relationship between adaptive noise in Kalman filtering and smoothing regularisers.

5.3 Sequential Evidence Maximisation with Sequentially Updated Priors

Within a minimum variance framework, a weight decay prior constitutes a redundant smoothing constraint. Smooth estimates may be obtained by equating the prior at the current estimation step to the posterior from the previous estimation step. In addition, in extended Kalman filtering, we have cognisance of the equation describing the evidence function in terms of the noise covariances. As mentioned in Section 3, maximising the evidence function at the parameter estimation level is analogous to maximising the likelihood of the noise covariances as new data is gathered. Consequently, we can compute R_k and Q_k automatically by maximising the evidence density:

$$\mathrm{p}(y_{k+1}|Y_k, M_j, R_k, Q_k) \sim \mathcal{N}(\mathbf{g}_{\mathrm{k+1}}(\hat{\mathbf{w}}_\mathrm{k}, \mathbf{x}_{\mathrm{k+1}}), \mathrm{G}_{\mathrm{k+1}}(\mathrm{P}_\mathrm{k} + \mathrm{Q}_\mathrm{k})\mathrm{G}_{\mathrm{k+1}}^\mathrm{T} + \mathrm{R}_{\mathrm{k+1}}) \tag{34}$$

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:

$$\begin{aligned} r_{k+1} &= y_{k+1} - \mathbf{E}[y_{k+1}|Y_k, M_j, R_k, Q_k] \\ &= y_{k+1} - \mathbf{g}_{\mathrm{k}+1}(\hat{\mathbf{w}}_{\mathrm{k}}, \mathbf{x}_{\mathrm{k}}) \end{aligned}$$

It follows from equation (34) and the definition of the model residuals that

$$\mathbf{E}[r_{k+1}] = 0$$

and

$$\mathbf{E}[r_{k+1}^2] = G_{k+1}(P_k + Q_k)G_{k+1}^T + R_{k+1}$$

In addition, it is not difficult to prove that

$$\mathbf{E}[r_k r_l] = 0$$

That is, the residuals are uncorrelated, and assuming they are Gaussian, they are also independent (Jazwinski and Bailie 1967). Consequently

$$\mathrm{p}(r_{k+1} r_{k+2} \cdots r_{k+N}) = \mathrm{p}(r_{k+1})\mathrm{p}(r_{k+2}) \cdots \mathrm{p}(r_{k+N})$$

where

$$\begin{aligned} \mathrm{p}(r_{k+j}) &= \frac{1}{(2\pi)^{1/2}(G_{k+1}(P_k + Q_k)G_{k+1}^T + R_{k+j})^{-1/2}} \\ &\quad \exp\Big(-\frac{1}{2}\frac{r_{k+j}^2}{G_{k+1}(P_k + Q_k)G_{k+1}^T + R_{k+j}}\Big) \end{aligned}$$

The probability of the residuals is thus equivalent to the evidence function at the parameter estimation level. That is

$$\mathrm{p}(r_{k+1}) = \mathrm{p}(y_{k+1}|Y_k, M_j, R_k, Q_k)$$

In the following subsections, we present three algorithms for updating the noise covariances by maximising the evidence function.

5.3.1 Scalar process noise estimation

Let us assume that the process noise covariance may be described by a single parameter q. More specifically

$$Q = qI_q$$

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

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

Equating the derivative to zero yields:

$$r_{k+1}^2 = \mathbf{E}[r_{k+1}^2] \tag{35}$$

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 maximising the evidence function corresponds to equating the covariance over time r^2_{k+1} to the ensemble covariance $\mathbf{E}[r^2_{k+1}]$. That is, maximising the evidence leads to a covariance matching method.

Jazwinski (Jazwinski 1969, Jazwinski and Bailie 1967) devised an algorithm for updating q according to equation (35). Since

$$r^2_{k+1} = G_{k+1} P_k G^T_{k+1} + q G_{k+1} G^T_{k+1} + R_{k+1},$$

it follows that q may be recursively computed according to:

$$q = \begin{cases} \frac{r^2_{k+1} - \mathbf{E}[r^2_{k+1} | q=0]}{G_{k+1} G^T_{k+1}} & \text{if } q \geq 0 \\ 0 & \text{Otherwise} \end{cases} \tag{36}$$

This estimator increases q each time the model errors (residuals) are greater than what their theoretical value (the ensemble covariance) predicts. When q increases, the Kalman gain also increases and consequently the model parameters update also increases (equations (17) and (18)). That is, the estimator places more emphasis on the incoming data. As long as the residuals remain smaller that the ensemble covariance, the process noise input is zero and the filter carries on minimising the variance of the parameters (i.e. tending to a regularised solution). Section 7.2 discusses an experiment where this behaviour is illustrated.

The estimator of equation (36) is based on a single residual and is therefore of little statistical significance. This difficulty is overcome by employing the sample mean for N predicted residuals, instead of a single residual. Jazwinski (Jazwinski 1969) shows that for the following sample mean:

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

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

$$q = \begin{cases} \frac{m^2_r - \mathbf{E}[m^2_r | q=0]}{S} & \text{if } q \geq 0 \\ 0 & \text{Otherwise} \end{cases} \tag{37}$$

where

$$\mathbf{E}[m^2_r | q=0] = S_N P_k S^T_N + 1/N,$$
$$S = S_N S^T_N + S_{N-1} S^T_{N-1} + \cdots + S_1 S^T_1$$

and

$$\begin{aligned} S_N &= \frac{1}{N} \sum_{l=1}^{N} \frac{1}{R^{1/2}_{k+l}} G_{k+l} \\ S_{N-1} &= \frac{1}{N} \sum_{l=2}^{N} \frac{1}{R^{1/2}_{k+l}} G_{k+l} \\ &\vdots \\ S_1 &= \frac{1}{N} \frac{1}{R^{1/2}_{k+N}} G_{k+N} \end{aligned}$$

With this estimator, one has to choose the length N of the moving window used to update q. As the window size increases, the estimator for q has less effect on the extended Kalman filter estimates. That is the data idiosyncrasies are interpreted more as non-stationarities than as noise. We refer to the problem of choosing the right window length as the regularisation/tracking dilemma. It is a dilemma because we cannot ascertain, without *a priori* knowledge, whether the fluctuations in the data correspond to non-stationary behaviour or noise. This problem is typical of high frequency predictions in foreign exchange markets (Moody 1997).

5.3.2 Scalar measurement noise estimation

By maximising the evidence function with respect to $R_k = rI_m$, as done in the previous section, one obtains the following estimator for r:

$$r = \begin{cases} r_k^2 - G_k(P_k + Q_k)G_k^T & \text{if } r \geq 0 \\ 0 & \text{Otherwise} \end{cases} \tag{38}$$

The hyper-parameter r is not as useful as q in controlling filter divergence. This is because r can slow down the rate of decrease of the covariance matrix P_k, but cannot cause it to increase (equations (17), (18) and (19)). Nonetheless, it constitutes an improvement over standard Kalman filtering. It is also possible to combine estimators for r and q simultaneously.

5.3.3 Multiple noise hyper-parameters estimation

It is possible to extend the derivations of the previous sections to a more general noise model. We may assume the following covariance model:

$$Q = \begin{bmatrix} q_1 & 0 & \cdots & 0 \\ 0 & q_2 & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & q_q \end{bmatrix} \tag{39}$$

Calculating the derivative of the evidence function with respect to a generic diagonal entry of Q yields:

$$\begin{aligned} \frac{\partial}{\partial q_i}\mathrm{p}(r_{k+1}) &= \frac{\mathrm{p}(r_{k+1})}{2} G_{k+1}(\frac{\partial}{\partial q_i}Q)G_{k+1}^T \left[-(G_{k+1}(P_k + Q_k)G_{k+1}^T + R_k)^{-1} + \right. \\ & \left. r_{k+1}^2 (G_{k+1}(P_k + Q_k)G_{k+1}^T + R_k)^{-2} \right] \end{aligned}$$

Under the assumption that

$$G_{k+1}(\frac{\partial}{\partial q_i}Q)G_{k+1}^T \neq 0,$$

we obtain a relation for the maxima of the evidence function when the quantity in the square brackets vanishes. This relation is given by:

$$G_{k+1}QG_{k+1}^T = r_{k+1}^2 - \mathbf{E}[r_{k+1}^2 | Q = 0] = \varepsilon_{k+1} \tag{40}$$

Equation (40) represents an under-determined system of equations for the unknowns q_i. Since Q is diagonal, it may be rewritten as:

$$\begin{bmatrix} (\frac{\partial y_{k+1}}{\partial \mathbf{w}_1})^2 & (\frac{\partial y_{k+1}}{\partial \mathbf{w}_2})^2 & \cdots & (\frac{\partial y_{k+1}}{\partial \mathbf{w}_q})^2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_q \end{bmatrix} = \varepsilon_{k+1} \tag{41}$$

To improve the statistical significance of the estimator, while at the same time conditioning it, we may use a moving window of size N to estimate the noise hyper-parameters. That is,

$$\begin{bmatrix} (\frac{\partial y_{k+1}}{\partial \mathbf{w}_1})^2 & (\frac{\partial y_{k+1}}{\partial \mathbf{w}_2})^2 & \cdots & (\frac{\partial y_{k+1}}{\partial \mathbf{w}_q})^2 \\ (\frac{\partial y_{k}}{\partial \mathbf{w}_1})^2 & (\frac{\partial y_{k}}{\partial \mathbf{w}_2})^2 & & (\frac{\partial y_{k}}{\partial \mathbf{w}_q})^2 \\ \vdots & & \ddots & \\ (\frac{\partial y_{k-N}}{\partial \mathbf{w}_1})^2 & (\frac{\partial y_{k-N}}{\partial \mathbf{w}_2})^2 & & (\frac{\partial y_{k-N}}{\partial \mathbf{w}_q})^2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_q \end{bmatrix} = \begin{bmatrix} \varepsilon_{k+1} \\ \varepsilon_k \\ \vdots \\ \varepsilon_{k-N} \end{bmatrix} \tag{42}$$

This estimator is less reliable than the estimator of Section 5.3.1. It involves estimating more parameters and it requires a long moving window to avoid ill-conditioning. Nonetheless, multiple hyper-parameters are very handy when one considers distributed priors for automatic relevance determination. This topic is covered in the next section.

6 Automatic Relevance Determination

Although most of the work on learning theory has been devoted to the study of the generalisation error in terms of model complexity and the number of samples, we believe that the issues of selecting the right input variables and basis functions are equally important. Spurious input variables will invariably lead to deterioration in model performance. For instance, in conventional neural network modelling of finite data sets, the weights in charge of scaling the irrelevant inputs do not usually tend to zero because of random correlations between the inputs and the output (Mackay 1995).

In addition, in many engineering and financial applications (de Freitas, Gaylard, Stevens, Ridley and Landy 1996, de Freitas, Macleod and Maltz 1996, Niranjan 1996), we often find that we know part of the equation governing a particular process. That is, we might have inferred that the relationship between two measurements x and y is given by:

$$y = x^2 + g(x)$$

where the function $g(x)$ is unknown. In such cases, it would be wasteful to estimate the dependence of y on x with generic basis functions, such as Gaussians or sigmoids. Instead, we should incorporate a quadratic basis function into the neural network structure and hence avoid extra model fitting.

Furthermore, we might, from the outset, include different basis functions and then use an automatic relevance mechanism to establish the significance of each basis function. When choosing the initial basis functions, preference should be given to basis functions that appear frequently in physical laws, such as polynomials, exponentials, sinusoids, logarithms, etc. This strategy has been successfully employed in computing models for pneumatic control valves and structural vibration in induction motors (de Freitas, Gaylard, Stevens, Ridley and Landy 1996, de Freitas, Macleod and Maltz 1996). It follows that this idea may also be applied to clustering algorithms to determine the relevance of each cluster.

The problems of automatic relevance determination of inputs (ARDI) and of basis functions (ARDF) can be addressed with clarity and efficiency within a Bayesian framework. In this context, distributed priors are used to determine the relevance of the various inputs and basis functions. For example, while addressing the ARDI problem, Mackay (Mackay 1994a, Mackay 1995) has proposed the following modification to the prior of equation (22):

$$\mathrm{p}(\mathbf{w}|\alpha_{\mathrm{c}}) = \frac{1}{\prod_{\mathrm{c}} \mathrm{Z_w}(\alpha_{\mathrm{c}})} \exp\Big(\sum_{\mathrm{c}} \alpha_{\mathrm{c}} \mathrm{E_{wc}} \Big)$$

where

$$E_{wc} = \sum_{i \in c} \frac{w_i^2}{2}$$

and the regularisation coefficient α_c controls the weight decay rates for the parameters linked to input c. That is, a different regularisation coefficient is assigned to each input. The decay rates for irrelevant inputs may, therefore, be automatically inferred to be large. Consequently, their harmful effect on the model is prevented.

A similar technique based on multiple hyper-parameters for ARDI has been proposed by Sutton (Sutton 1992a). Sutton implements multiple learning rates to determine the relevance of various inputs to a linear network. The theory developed in Sections 5.1, 5.2 and 5.3 established the equivalence between multiple learning rates η, regularisation coefficients α_c and process noise hyper-parameters q. By virtue of these results, the equivalence between Mackay's and Sutton's ARDI frameworks follows immediately.

In our work on sequential learning, we choose to implement ARDI and ARDF using multiple adaptive noise hyper-parameters q_i, obtained with the algorithm described in Section 5.3.3. Different priors q_i are assigned to each input for ARDI and to each basis function for ARDF. By monitoring the model prediction and fluctuations in the q_i's, it is possible to detect irrelevant inputs or basis functions. As shown in an experiment in Section 7, the q_i's corresponding to irrelevant variables tend to fluctuate excessively. To improve model performance, the harmful basis functions or inputs should be removed.

7 Experiments

7.1 Experiment 1: Comparison Between the Various Noise Estimation Methods

To compare the performance of the various EKF training algorithms discussed in this report, 100 input-output data vectors were generated from the following nonlinear, non-stationary process:

$$\begin{aligned} x_{k+1} &= 0.5x_{k-1} + \frac{25x_{k-1}}{1+x_{k-1}^2} + 8\cos(1.2(k-1)) + d_k \\ y_k &= \frac{x_k^2}{20} + v_k \end{aligned}$$

where x_k denotes the input vectors and y_k the ouput vectors of 300 time samples each. The Gaussian process noise standard deviation was set to 0.1, while the measurement noise standard deviation was set to $3\sin(0.05k)$. The initial state x_0 was 0.1. Figure 3 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 output linear neuron with the following methods: the standard EKF algorithm, the EKF algorithm with P_k updated by error back-propagation (EKFBP), the EKF algorithm with evidence maximisation and weight decay priors (EKFEV), the EKF algorithm with MAP noise adaptation (EKFMAP) and the EKF algorithm with evidence maximisation and sequentially updated priors (EKFQ). The initial variance of the weights, initial weights covariance matrix entries, initial R and initial Q were set to 1, 10, 3 and 1×10^{-5} respectively. The length of the sliding window of the adaptive noise algorithms was set to 10 time samples[1].

The simulation results for the EKF and EKFQ algorithms are shown in Figure 4. Note that the EKFQ algorithm slows down the convergence of the EKF paramater estimates so as to be able to track the changing measurement variance. In Table 1, we compare the one-step-ahead normalised square errors (NSE) obtained with each method. The NSE are defined as

[1] A matlab demo, corresponding to this example, is available at the following web site: `http://svr-www.eng.cam.ac.uk/~jfgf/software.html` .

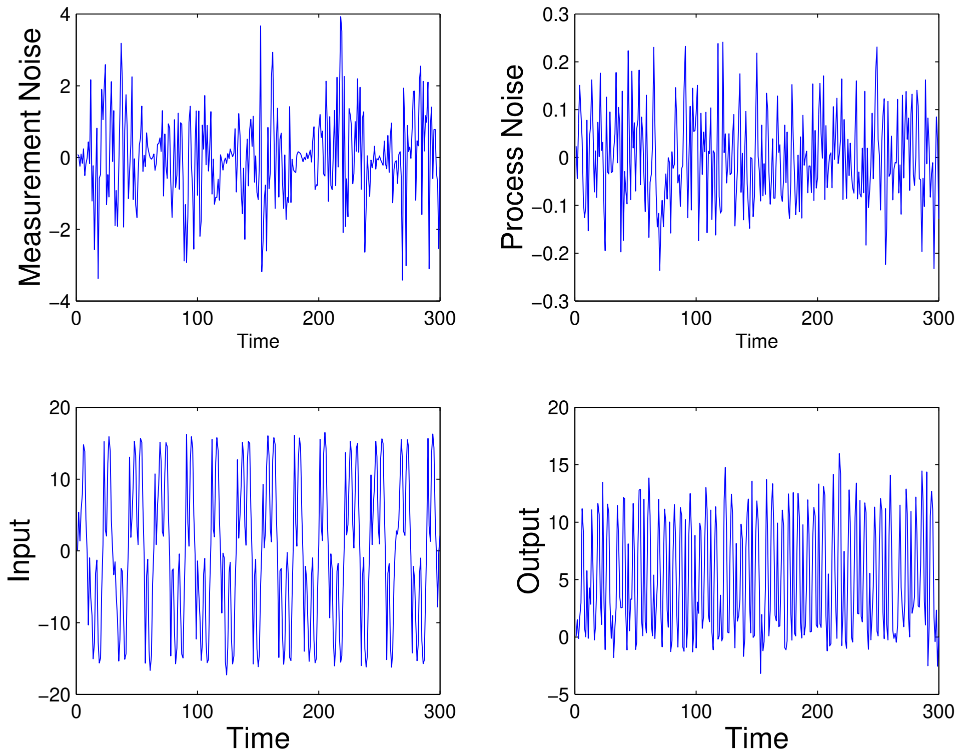

Figure 3: Data generated to train MLP.

follows:

$$\text{NSE} = \sqrt{\sum_k (\mathbf{y}_k - \hat{\mathbf{g}}(\mathbf{w}_k, \mathbf{x}_k))^2}$$

According to the table, it is clear that the only algorithm that provides a clear prediction improvement over the standard EKF algorithm is the evidence maximisation algorithm with sequential update priors. In terms of computational time, the EKF algorith with P_k updated by back-propagation is 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 are therefore unsuitable for sequential learning tasks.

	NSE	**Floating point operations**
EKF	25.95	21,886,963
EKFQ	23.01	24,106,195
EKFMAP	61.06	22,584,733
EKFEV	73.94	22,595,195
EKFBP	58.87	2,187,763

Table 1: Simulation results for 100 runs in experiment 1.

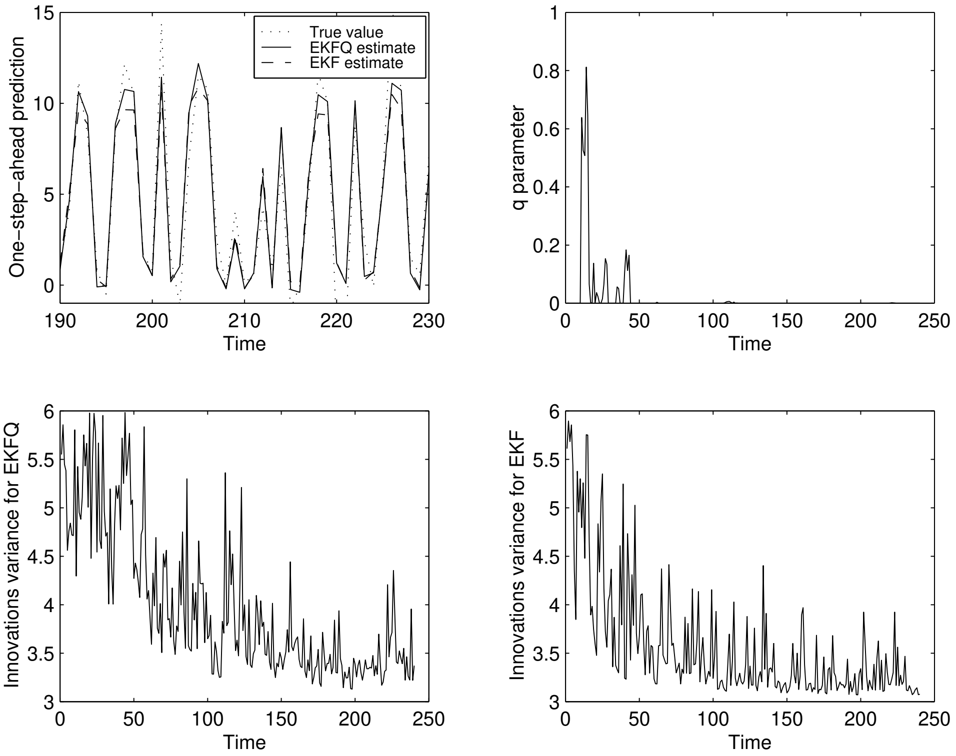

Figure 4: Simulation results for the EKF and EKFQ algorithms.

7.2 Experiment 2: Sequential Evidence Maximisation with Sequentially Updated Priors

This experiment aims at describing the behaviour of the evidence maximisation with prior updating algorithm in a time-varying, noisy and chaotic scenario. The problem tackled is a more difficult variation of the chaotic quadratic or logistic map. 100 input (y_k) and output (y_{k+1}) data vectors were generated according to the following equation:

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

where v_k denotes Gaussian noise with standard deviation equal to 0.01. In the interval $150 < k \leq 225$, the series exhibits chaotic behaviour. A 2 layer MLP with 10 sigmoidal neurons in the hidden layer and a single output linear neuron was 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, $1e-4$ and 0 respectively. The sliding window to estimate Q was set to 3 time samples.

As shown in Figure 5, during the initialisation and after each change of behaviour (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 stabilises, the minimum variance minimisation 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 tradeoff between regularisation and tracking. The results obtained with the EKF and EKFQ algorithms are summarised in table 2[2].

[2] A matlab demo, corresponding to this example, is available at the following web site:

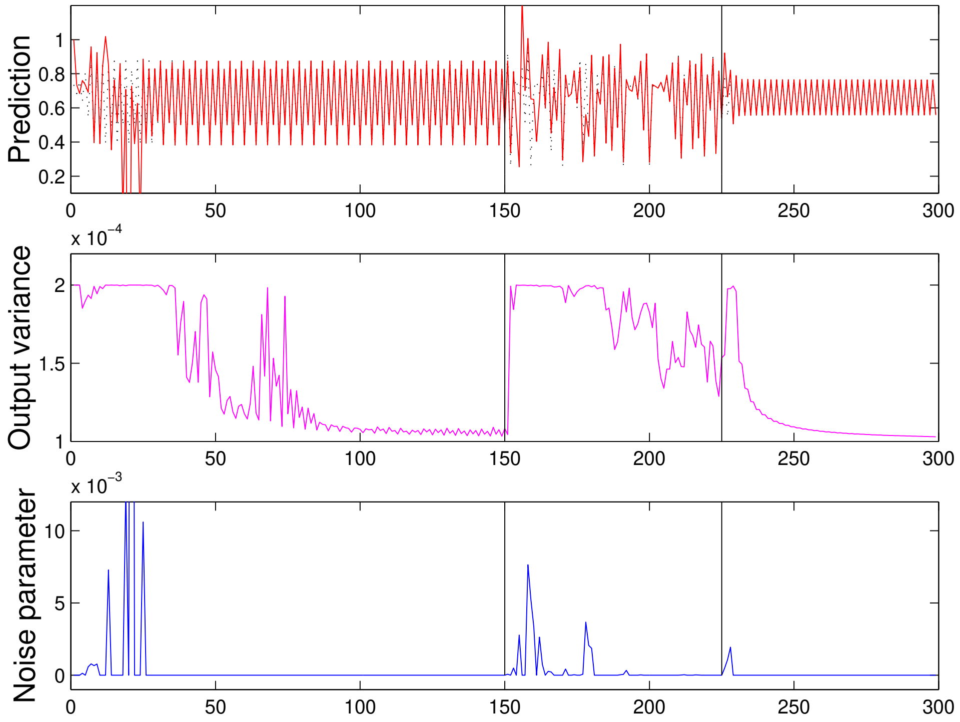

Figure 5: Performance of the evidence maximisation for a non-stationary chaotic quadratic map. The top plot shows the true data [$\cdots$] and the prediction [—], the middle plot shows the output confidence intervals while the bottom plot shows the value of the adaptive noise parameter.

	NSE	**Floating point operations**
EKF	31.76	21,814,011
EKFQ	1.37	22,968,178

Table 2: Simulation results for 100 runs of the quadratic chaotic map.

7.3 Experiment 3: Automatic Relevance Determination

To test the ARDI framework using multiple process noise hyper-parameters, we generated a data sequence with the modified quadratic map. We then trained an MLP with 20 neurons in the hidden layer with two inputs. One of the inputs was the correct y_k signal. The other input corresponded to a zero mean random signal with variance equal to 2. As depicted in Figure 9, the variation of the noise parameters linked to each input allows us to detect the spurious input. This algorithm should be used with caution. If the output prediction is far from reasonable, then we cannot expect the information given by the noise parameters to be meaningful. During our simulations, we also found out that the noise parameters depend on the magnitude of the inputs. Consequently, one should also monitor the values of the input weights before a decision to neglect a particular input is adopted.

http://svr-www.eng.cam.ac.uk/~jfgf/software.html .

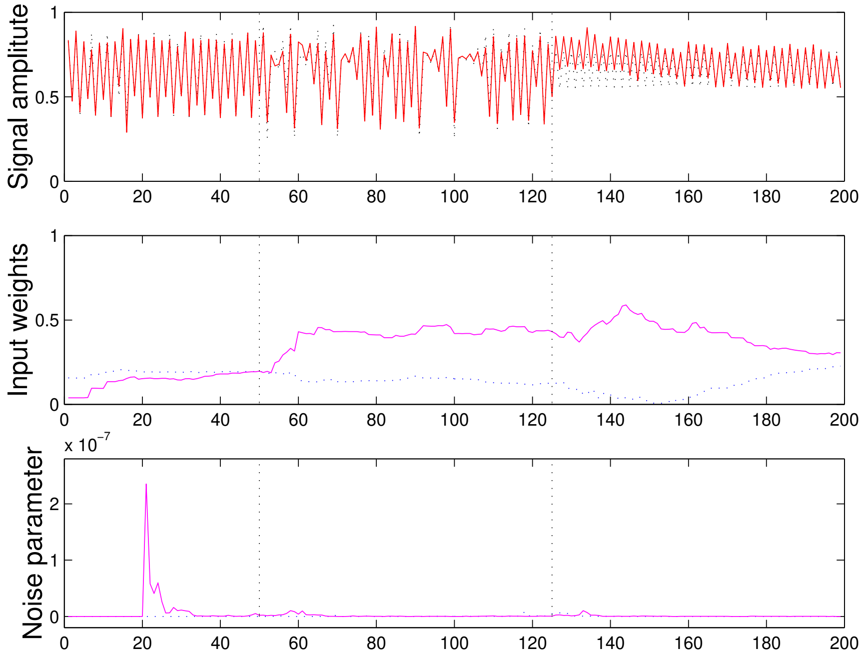

Figure 6: Automatic relevance determination by monitoring the input weights adaptive noise covariance. The top plot shows the actual data [$\cdots$] and the prediction [—]. The middle plot shows the average of the weights for the relevant [$\cdots$] and irrelevant [—] inputs. The bottom plot shows the average of the adaptive noise parameters for the relevant [$\cdots$] and irrelevant [—] inputs.

7.4 Application: Pricing Financial Options

Derivatives are financial instruments whose value depends on some basic underlying cash product, such as interest rates, equity indices, commodities, foreign exchange, bonds, etc. An option is a particular type of derivative that gives the holder the right, but not the obligation, to do something. For example, a call option allows the holder to buy a cash product, at a specified date in the future, for a price determined in advance. The price at which the option is exercised is known as the strike price, while the date at which the option lapses is often referred to as the maturity time. Put options, on the other hand, allow the holder to sell the underlying cash product.

In recent years, the mathematical modelling of financial derivatives has become increasingly important for two reasons. Firstly, financial institutions have much interest in developing more sophisticated pricing models for options contracts (Hull 1997). Secondly, options data offers an excellent source of difficult and challenging problems to the statistical and neural computing communities (Hutchinson, Lo and Poggio 1994, Ingber and Wilson 1998, Niranjan 1996). So far, the research results seem to provide clear evidence that there is a nonlinear and non-stationary 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 model (Black and Scholes 1973).

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 non-stationary aspects of the problem. Niranjan (1996) uses a Kalman filtering framework to sequentially propagate the estimated parameters and their corresponding uncertainties.

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. The one-step-ahead 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 train the MLPs with the conventional EKF and EKFQ algorithms. We treat the cash product's value normalised by the strike price and time to maturity as its inputs. The network's output consists of the call and put option prices normalised by the strike price[3]. We used five pairs of call and put option contracts on the FTSE100 index (from February 1994 to December 1994) to evaluate our pricing algorithms. The parameters where estimated by the following methods:

Trivial : This method simply involves using the current value of the option as the next prediction.

RBF-EKF : Represents a regularised radial basis function network with 4 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 (normalised 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 6 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 6 sigmoidal hidden units and a linear output neuron, trained with the EKF with evidence maximisation and sequentially updated priors. The states covariance P was given by a diagonal matrix with entries equal to 10. The weights prior corresponded to a zero mean Gaussian density with covariance equal to 1.

Figure 7 shows the one-step-ahead predictions obtained with the EKFQ algorithm. In Table 3, we compare the one-step-ahead normalised square errors obtained with each method. It should be mentioned that the square errors were only measured over 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.

[3] A matlab demo, corresponding to this example, is available at the following web site: `http://svr-www.eng.cam.ac.uk/~jfgf/software.html`.

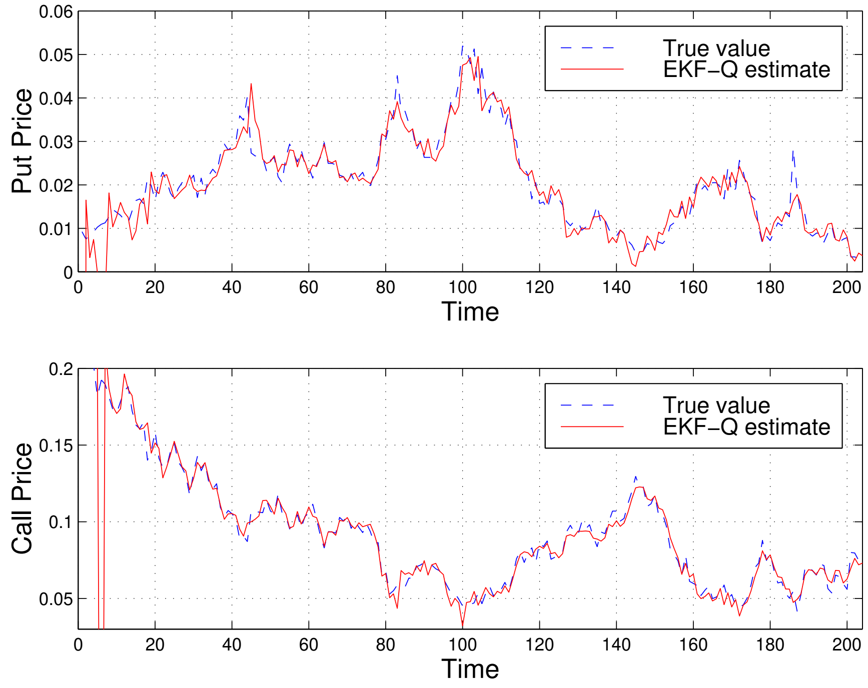

Figure 7: Tracking put and call option prices with an MLP trained with the EKFQ algorithm.

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

Table 3: One-step-ahead prediction errors on call options.

8 Conclusions

We have shown that the Bayesian view of Kalman filtering has a rich set of tools to offer. Tuning the noise parameters in a systematic way leads to many interesting attributes such as regularisation and automatic relevance determination. In addition, the Bayesian inference framework provides an elegant, unifying theory to the problem of sequential learning.

Although we did not estimate the drift function in this paper, we have proved elsewhere that linear drift functions may be estimated via extended Kalman smoothing and the EM algorithm (de Freitas et al. 1998). The method for estimating the linear drift functions with the EM algorithm is simple, stable and elegant but exhibits very slow convergence and is only applicable to stationary environments.

We showed that distributed learning rates, adaptive noise parameters and adaptive smoothing regularisers are mathematically equivalent. This result sheds light on many areas of the machine learning field. It places many diverse approaches to estimation and regularisation within a unified framework.

There are many avenues for further research. These include stating our theoretical results in a more rigorous mathematical formulation, implementing static and dynamic mixtures of models, implementing other model structures such as recurrent networks and, finally, testing the algorithms on additional financial and engineering problems.

9 Acknowledgements

We would like to thank Analytical Mechanics Associates (Virginia, USA) for their helpful assistance in providing some of the references. João FG de Freitas is financially supported by two University of the Witwatersrand Merit Scholarships, a Foundation for Research Development Scholarship (South Africa) and a Trinity College External Research Studentship (Cambridge).

A Bayesian Derivation of the Kalman Filter

Consider the following linear Gauss-Markov process

$$\begin{aligned} \mathbf{w}_{k+1} &= \mathbf{w}_k + \mathbf{d}_k &(43)\\ \mathbf{y}_k &= H_k \mathbf{w}_k + \mathbf{v}_k &(44)\end{aligned}$$

where the covariance of the process noise $\mathbf{d}_k$ is given by Q_k, the covariance of the measurement noise $\mathbf{v}_k$ is given by R_k and the covariance of the states $\mathbf{w}_k$ is given by P_k. The remaining symbols have been defined in Section 2. It is possible to apply Bayes rule to estimate the posterior density function for the model parameters after the new data arrives. In doing so, the posterior is given by:

$$p(\mathbf{w}_{k+1}|Y_{k+1}) = \frac{p(\mathbf{y}_{k+1}|\mathbf{w}_{k+1})}{p(\mathbf{y}_{k+1}|Y_k)} p(\mathbf{w}_{k+1}|Y_k) \qquad (45)$$

That is,

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

Assuming Gaussian approximations to the probability density functions and uncorrelated zero mean noise processes, representations for the likelihood, evidence and prior can be derived in terms of the first and second order statistics.

A.1 Prior Gaussian Density Function

The mean is given by:

$$\begin{aligned} \mathbf{E}(\mathbf{w}_{k+1}|Y_k) &= \mathbf{E}[\mathbf{w}_k + \mathbf{d}_k|Y_k] \\ &= \mathbf{E}[\mathbf{w}_k|Y_k] \\ &= \hat{\mathbf{w}}_k \end{aligned}$$

and the covariance is given by:

$$\begin{aligned} \mathrm{Cov}(\mathbf{w}_{k+1}|Y_k) &= \mathbf{E}[(\mathbf{w}_k + \mathbf{d}_k - \hat{\mathbf{w}}_k)(\mathbf{w}_k + \mathbf{d}_k - \hat{\mathbf{w}}_k)^T|Y_k] \\ &= P_k + Q_k + 2\mathbf{E}[(\mathbf{w}_k - \hat{\mathbf{w}}_k)\mathbf{d}_k|Y_k] \\ &= P_k + Q_k \end{aligned}$$

Hence, the prior is given by:

$$\text{Prior} = p(\mathbf{w}_{k+1}|Y_k) \sim \mathcal{N}(\hat{\mathbf{w}}_k, P_k + Q_k) \qquad (46)$$

A.2 Evidence Gaussian Density Function

The mean is given by:

$$\begin{aligned}
\mathbf{E}(\mathbf{y}_{\mathrm{k+1}}|\mathrm{Y_k}) &= \mathbf{E}[H_{k+1}\mathbf{w}_{\mathrm{k+1}} + \mathbf{v}_{\mathrm{k+1}}|\mathrm{Y_k}] \\
&= H_{k+1}\mathbf{E}[\mathbf{w}_{\mathrm{k+1}}|\mathrm{Y_k}] \\
&= H_{k+1}\hat{\mathbf{w}}_{k+1|k} \\
&= H_{k+1}\hat{\mathbf{w}}_{k}
\end{aligned}$$

and the covariance is given by:

$$\begin{aligned}
\mathrm{Cov}(\mathbf{y}_{\mathrm{k+1}}|\mathrm{Y_k}) &= \mathbf{E}[(H_{k+1}\mathbf{w}_{\mathrm{k+1}} + \mathbf{v}_{\mathrm{k+1}} - \mathrm{H_{k+1}}\hat{\mathbf{w}}_{\mathrm{k}}) \\
&\quad (H_{k+1}\mathbf{w}_{\mathrm{k+1}} + \mathbf{v}_{\mathrm{k+1}} - \mathrm{H_{k+1}}\hat{\mathbf{w}}_{\mathrm{k}})^{\mathrm{T}}|\mathrm{Y_k}] \\
&= \mathbf{E}[(H_{k+1}(\mathbf{w}_{\mathrm{k+1}} - \hat{\mathbf{w}}_{\mathrm{k}}) + \mathbf{v}_{\mathrm{k+1}})(\mathrm{H_{k+1}}(\mathbf{w}_{\mathrm{k+1}} - \hat{\mathbf{w}}_{\mathrm{k}}) + \mathbf{v}_{\mathrm{k+1}})^{\mathrm{T}}|\mathrm{Y_k}] \\
&= H_{k+1}(P_k + Q_k)H_{k+1}^T + R_{k+1}
\end{aligned}$$

Hence, the evidence is given by:

$$\text{Evidence: } \mathrm{p}(\mathbf{y}_{\mathrm{k+1}}|\mathrm{Y_k}) \sim \mathcal{N}(\mathrm{H_{k+1}}\hat{\mathbf{w}}_{\mathrm{k+1}}, \mathrm{H_{k+1}}(\mathrm{P_k} + \mathrm{Q_k})\mathrm{H}_{\mathrm{k+1}}^{\mathrm{T}} + \mathrm{R_{k+1}}) \tag{47}$$

A.3 Likelihood Gaussian Density Function

The mean is given by:

$$\begin{aligned}
\mathbf{E}(\mathbf{y}_{\mathrm{k+1}}|\mathbf{w}_{\mathrm{k+1}}) &= \mathbf{E}[H_{k+1}\mathbf{w}_{\mathrm{k+1}} + \mathbf{v}_{\mathrm{k+1}}|\mathbf{w}_{\mathrm{k+1}}] \\
&= H_{k+1}\mathbf{E}[\mathbf{w}_{\mathrm{k+1}}|\mathbf{w}_{\mathrm{k+1}}] \\
&= H_{k+1}\mathbf{w}_{\mathrm{k+1}}
\end{aligned}$$

and the covariance is given by:

$$\begin{aligned}
\mathrm{Cov}(\mathbf{y}_{\mathrm{k+1}}|\mathbf{w}_{\mathrm{k+1}}) &= \mathbf{E}[(H_{k+1}\mathbf{w}_{\mathrm{k+1}} + \mathbf{v}_{\mathrm{k+1}} - \mathrm{H_{k+1}}\mathbf{w}_{\mathrm{k+1}}) \\
&\quad (H_{k+1}\mathbf{w}_{\mathrm{k+1}} + \mathbf{v}_{\mathrm{k+1}} - \mathrm{H_{k+1}}\mathbf{w}_{\mathrm{k+1}})^{\mathrm{T}}|\mathbf{w}_{\mathrm{k+1}}] \\
&= R_{k+1}
\end{aligned}$$

Hence, the likelihood is given by:

$$\text{Likelihood: } \mathrm{p}(\mathbf{y}_{\mathrm{k+1}}|\mathbf{w}_{\mathrm{k+1}}, \mathrm{M_j}, \mathrm{R_k}, \mathrm{Q_k}) \sim \mathcal{N}(\mathrm{H_{k+1}}\mathbf{w}_{\mathrm{k+1}}, \mathrm{R_{k+1}}) \tag{48}$$

A.4 Posterior Gaussian Density Function

Substituting the equations for the means and covariances of the evidence, prior and likelihood into equation (46) and completing squares in the exponent of the Gaussian exponential, yields the posterior density function:

$$\mathrm{p}(\mathbf{w}_{\mathrm{k+1}}|\mathrm{Y_{k+1}}, \mathrm{M_j}, \mathrm{R_k}, \mathrm{Q_k}) = \mathrm{A_{k+1}} \exp\Big(-\frac{1}{2}(\mathbf{w}_{\mathrm{k+1}} - \hat{\mathbf{w}}_{\mathrm{k+1}})\mathrm{P}_{\mathrm{k+1}}^{-1}(\mathbf{w}_{\mathrm{k+1}} - \hat{\mathbf{w}}_{\mathrm{k+1}})\Big) \tag{49}$$

where the coefficients A_{k+1} are represented by the following expression:

$$A_{k+1} = \frac{|H_{k+1}(P_k + Q_k)H_{k+1}^T + R_{k+1}|^{1/2}}{(2\pi)^{q/2}|R_{k+1}|^{1/2}|P_k + Q_k|^{1/2}}$$

and $\hat{\mathbf{w}}_{k+1}$ and P_{k+1} are given by:

$$\hat{\mathbf{w}}_{k+1} = \hat{\mathbf{w}}_k + K_{k+1}(\mathbf{y}_{\mathrm{k+1}} - \mathrm{H_{k+1}}\hat{\mathbf{w}}_{\mathrm{k}}) \tag{50}$$

$$P_{k+1} = P_k + Q_k - K_{k+1}H_{k+1}(P_k + Q_k) \tag{51}$$

where K_k is known as the Kalman gain:

$$K_{k+1} = \frac{(P_k + Q_k)H_{k+1}^T}{R_{k+1} + H_{k+1}(P_k + Q_k)H_{k+1}^T} \tag{52}$$

B Computing Derivatives for the Jacobian Matrix

For the network depicted in Figure 10, the output layer mapping is given by:

$$y = w_1 + w_2 o_{11} + w_3 o_{12}$$

and consequently, the derivatives with respect to the weights are given by:

$$\begin{aligned}
\frac{\partial y}{\partial w_1} &= 1 \\
\frac{\partial y}{\partial w_2} &= o_{11} \\
\frac{\partial y}{\partial w_3} &= o_{12}
\end{aligned}$$

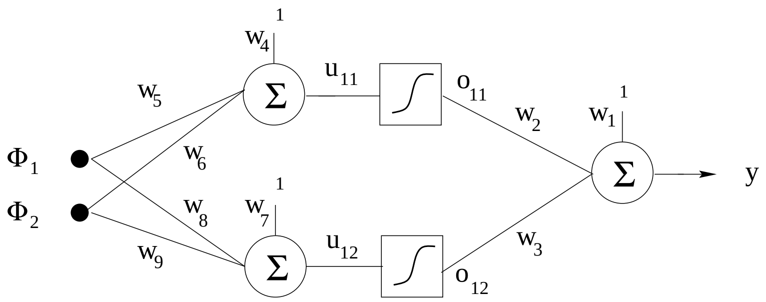

Figure 8: Simple MLP structure for regression problems.

The hidden layer mapping for the top neuron is:

$$o_{11} = \frac{1}{1 + \exp(-u_{11})}, \qquad \text{where } u_{11} = w_4 + w_5\Phi_1 + w_6\Phi_2$$

The corresponding derivatives with respect to the weights are:

$$\begin{aligned}
\frac{\partial y}{\partial w_4} &= \frac{\partial y}{\partial o_{11}} \frac{\partial o_{11}}{\partial u_{11}} \frac{\partial u_{11}}{\partial w_4} \\
&= w_2 o_{11}(1 - o_{11}) \\
\frac{\partial y}{\partial w_5} &= w_2 o_{11}(1 - o_{11})\Phi_1 \\
\frac{\partial y}{\partial w_6} &= w_2 o_{11}(1 - o_{11})\Phi_2
\end{aligned} \tag{53}$$

The derivatives with respect to the weights of the other hidden layer neuron can be trivially calculated following the same procedure.

References

Anderson, B. D. and Moore, J. B. (1979). *Optimal Filtering*, Prentice-Hall, New Jersey.

Bar-Shalom, Y. and Li, X. R. (1993). *Estimation and Tracking: Principles, Techniques and Software*, Artech House, Boston.

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

Black, J. V. and Reed, C. M. (1996). A hybrid parametric, non-parametric approach to Bayesian target tracking, *Technical report*, DRA, Malvern, UK.

Blom, H. A. P. and 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. and Weigend, A. S. (1991). Bayesian back-propagation, *Complex Systems* **5**: 603–643.

Candy, J. V. (1986). *Signal Processing: The Model Based Approach*, McGraw-Hill, New York.

Carter, C. K. and Kohn, R. (1994). On Gibbs sampling for state space models, *Biometrika* **81**(3): 541–553.

de Freitas, J. F. G., Gaylard, A., Stevens, A. L., Ridley, J. N. and Landy, C. F. (1996). Identification of vibrating structures and fault detection using neural networks, *International Conference on Neural Networks*, Washington.

de Freitas, J. F. G., Macleod, I. M. and Maltz, J. S. (1996). Neural networks for pneumatic actuator fault detection, Accepted for Publication by the Transactions of the South African IEE.

de Freitas, J. F. G., Niranjan, M. and Gee, A. H. (1998). The EM algorithm and neural networks for nonlinear state space estimation, *Technical Report CUED/F-INFENG/TR 313*, Cambridge University, http://svr-www.eng.cam.ac.uk/~jfgf.

Gelb, A. (ed.) (1974). *Applied Optimal Estimation*, MIT Press.

Ghahramani, Z. and Jordan, M. (1995). Factorial Hidden Markov Models, *Technical Report 9502*, MIT Artificial Intelligence Lab, MA.

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

Ho, Y. C. and Lee, R. C. K. (1964). A Bayesian approach to problems in stochastic estimation and control, *IEEE Transactions on Automatic Control* **AC-9**: 333–339.

Hull, J. C. (1997). *Options, Futures, and Other Derivative*, third edn, Prentice Hall.

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

Ingber, L. and Wilson, J. K. (1998). Volatility of volatility of financial markets, To be published by the Journal of Mathematical and Computer Modelling.

Isard, M. and Blake, A. (1996). Contour tracking by stochastic propagation of conditional density, *European Conference on Computer Vision*, Cambridge, UK, pp. 343–356.

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

Jaynes, E. T. (1986). Bayesian methods: General background, *in* J. H. Justice (ed.), *Maximum Entropy and Bayesian Methods in Applied Statistics*, Cambridge University Press, pp. 1–25.

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

Jazwinski, A. H. (1970). *Stochastic Processes and Filtering Theory*, Academic Press.

Jazwinski, A. H. and Bailie, A. E. (1967). Adaptive filtering, *Technical Report 67-6*, Analytical Mechanics Associates, Maryland.

Kadirkamanathan, V. and Kadirkamanathan, M. (1995). Recursive estimation of dynamic modular RBF networks, *in* D. S. Touretzky, M. C. Mozer and M. E. Hasselmo (eds), *Advances in Neural Information Processing Systems 8*, pp. 239–245.

Kadirkamanathan, V. and Kadirkamanathan, M. (1996). Kalman filter based estimation of dynamic modular networks, *in* S. Usui, Y. Tohkura, S. Katagiri and E. Wilson (eds), *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing VI*, pp. 180–189.

Kadirkamanathan, V. and Niranjan, M. (1992). Application of an architecturally dynamic network for speech pattern classification, *Proceedings of the Institute of Acoustics*, Vol. 14, pp. 343–350.

Kadirkamanathan, V. and Niranjan, M. (1993). A function estimation approach to sequential learning with neural networks, *Neural Computation* **5**: 954–975.

Kalman, R. E. and Bucy, R. S. (1961). New results in linear filtering and prediction theory, *Transactions of the ASME (Jounal of Basic Engineering)* **83D**: 95–108.

Li, X. R. and 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. (1992). Bayesian interpolation, *Neural Computation* **4**(3): 415–447.

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

Mackay, D. J. C. (1994b). Hyperparameters: Optimise or integrate out?, *in* G. Heidbreder (ed.), *Maximum Entropy and Bayesian Methods*.

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

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* pp. 693–698.

Moody, J. (1997). Neural networks for financial predictions, Cambridge University Neural Networks Summer School.

More, J. J. (1977). The Levenberg-Marquardt algorithm: Implementation and theory, *in* A. Watson (ed.), *Numerical Analysis*, Lecture Notes in Mathematics 630, Springer Verlag, pp. 105–116.

Myers, K. A. and Tapley, B. D. (1976). Adaptive sequential estimation of unknown noise statistics, *IEEE Transactions on Automatic Control* pp. 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 and T. Petsche (eds), *Advances in Neural Information Processing Systems*, pp. 960–966.

Niranjan, M., Cox, I. J. and Hingorani, S. (1994). Recursive estimation of formants in speech, *Proceedings of the International Conference on Acoustics, Speech and Signal Processing.*

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

Puskorius, G. V. and Feldkamp, L. A. (1994). Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks, *IEEE Transactions on Neural Networks* **5**(2): 279–297.

Puskorius, G. V., Feldkamp, L. A. and Davis, L. I. (1996). Dynamic neural network methods applied to on-vehicle idle speed control, *Proceedings of the IEEE* **84**(10): 1407–1420.

Reynard, D., Wildenberg, A., Blake, A. and Marchant, J. (1996). Learning dynamics of complex motions from image sequences, *European Conference on Computer Vision*, Cambridge, UK, pp. 357–368.

Ruck, D. W., Rogers, S. K., Kabrisky, M., Maybeck, P. S. and 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.

Shah, S., Palmieri, F. and 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 and D. V. Hinkley (eds), *Time Series Models in Econometrics, Finance and Other Fields*, Chapman and Hall, pp. 1–67.

Sibisi, S. (1989). Regularization and inverse problems, *Maximum Entropy and Bayesian Methods*, Kluwer Academic Publishers, pp. 389–396.

Singhal, S. and Wu, L. (1988). Training multilayer perceptrons with the extended Kalman algorithm, *in* D. S. Touretzky (ed.), *Advances in Neural Information Processing Systems*, Vol. 1, San Mateo, CA, pp. 133–140.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions, *Journal of the Royal Statistical Society* **B** **36**(1): 111–147.

Stone, M. (1978). Cross-validation: A review, *Math. Operationsforsch. Statist. Ser. Statistics* **9**(1): 127–139.

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

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

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

Wahba, G. and Wold, S. (1969). A completely automatic French curve: Fitting spline functions by cross-validation, *Communications on Statistics, Series A* **4**(1): 1–17.

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

Williams, R. J. (1992). Some observations on the use of the extended Kalman filter as a recurrent network learning algorithm, *Technical Report NU-CCS-92-1*, College of Computer Science, Northeastern University, Boston.

Wolpert, D. H. (1993). On the use of evidence in neural networks, *in* S. J. Hanson, J. D. Cowan and C. L. Giles (eds), *Advances in Neural Information Processing Systems*, Vol. 5, San Mateo, CA, pp. 539–546.