DYNAMICAL LEARNING WITH THE EM ALGORITHM FOR NEURAL NETWORKS

JFG de Freitas, M Niranjan and AH Gee

Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
England
E-mail: jfgf@eng.cam.ac.uk

Abstract

In this paper, we derive an EM algorithm for nonlinear state space models. We use it to estimate jointly the neural network weights, the model uncertainty and the noise in the data. In the E-step we apply a forward-backward Rauch-Tung-Striebel smoother to compute the network weights. For the M-step, we derive expressions to compute the model uncertainty and the measurement noise. We find that the method is intrinsically very powerful, simple and stable.

1 Introduction

In environments where data is available in batches, it is possible to address the general problem of Bayesian learning with Gaussian approximations, in a principled way, using the expectation maximisation (EM) algorithm (Dempster, Laird and Rubin 1977) and dynamical models. Moreover, such an approach allows us to treat non-stationary data sets. This paper will focus on this learning strategy. In particular, it aims to extend the current work on EM learning for dynamical linear systems to the problem of computing the weights of a multi-layer perceptron (MLP), the initial conditions and the noise variances jointly.

The application of the EM algorithm to learning and inference in linear dynamical systems has occupied the attention of several researchers in the past. Chen (Chen 1981) was one of the pioneers in this field. In particular, he applied the EM algorithm to linear state space models known in the statistics literature as MIMIC models. In these models one observes multiple indicators and multiple causes of a single latent variable. Chen's MIMIC model was implemented in a simulation study relating social status and participation.

Watson and Engle (Watson and Engle 1983) have suggested using the EM algorithm, in conjunction with the method of scoring, for the estimation of linear dynamic factor, MIMIC and varying coefficient regression models. They evaluated their paradigm experimentally by estimating common factors in wage range data from several industries in Los Angeles, USA.

In 1982, Shumway and Stoffer (Shumway and Stoffer 1982) proposed the use of the EM algorithm and linear state space models for time series smoothing and forecasting with missing observations. To demonstrate their method, they considered a health series representing total expenditures for physician services as measured by two different sources. The time series produced by each source have similar values but exhibit missing observations at different periods. In Shumway and Stoffer's approach, the two series are automatically merged into an overall expenditure series, which is then used for forecasting. Nine years later, Shumway and Stoffer (Shumway and Stoffer 1991) extended their work to switching linear dynamic models. In essence, they derived a state space representation with measurement matrices that switch according to a time varying independent random process. They illustrate their method on an application involving the tracking of multiple targets.

The method of learning and inference in linear state space models via the EM algorithm has also played a role in the fields of speech analysis and computer vision. Digalakis, Rohlicek and Ostendorf (Digalakis, Rohlicek and Ostendorf 1993) applied it to the speech recognition problem. They made a connection between this method and the Baum-Welch estimation algorithm for hidden Markov models (HMMs). North and Blake (North and Blake 1998) have implemented the method to learn linear dynamic state space models used for tracking contours in images. Rao and Ballard (Rao and Ballard 1997) have also explored the relevance of the EM algorithm together with state space estimation in the field of vision. They have developed an hierarchical network model of visual recognition that encapsulates these concepts.

Ghahramani (Ghahramani 1997) has embedded the EM method for learning dynamic linear systems in a graphical models framework. He treats computationally intractable models, such as factorial HMMs and switching state space models, by resorting to Gibbs sampling and variational approximations. In another paper, Roweis and Ghahramani (Roweis and Ghahramani 1999) make use of the EM algorithm and linear state space representations to present a unified view of linear Gaussian models including factor analysis, mixtures of Gaussians, standard and probabilistic versions of principal component analysis, vector quantisation, Kalman smoothing and linear hidden Markov models.

This paper is organised as follows. Section 2 introduces the nonlinear state space modelling scheme adopted in the paper. The application of extended Kalman smoothing to estimate the weights of an MLP is discussed in Section 3. Section 4 presents a brief derivation of the EM algorithm, which is used as a step towards the derivation of the EM algorithm for nonlinear state space models in Section 5. Section 6 examines some of the results obtained with experiments on synthetic and real data.

2 Nonlinear State Space Model

To investigate the application of the EM algorithm to state space learning, we shall focus on the following nonlinear state space representation:

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= A\boldsymbol{\theta}_t + \mathbf{u}_t \\ \mathbf{y}_t &= \widehat{\mathbf{f}}(\mathbf{x}_t, \boldsymbol{\theta}_t) + \mathbf{v}_t \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^d$ denotes the input data, $\mathbf{y} \in \mathbb{R}^c$ denotes the output data and $\boldsymbol{\theta} \in \mathbb{R}^m$ denotes the model states. The measurements nonlinear mapping $\widehat{\mathbf{f}}(\mathbf{x}_t, \boldsymbol{\theta}_t)$ corresponds to a multi-layer perceptron (MLP) whose weights are the model states $\boldsymbol{\theta}$. The framework may be easily extended to encompass recurrent networks, radial basis networks and many other approximation techniques. We assume the measurement $(\mathbf{v}_t)$ and process $(\mathbf{u}_t)$ noise terms to be zero mean Gaussian with covariances R and Q respectively. The matrix A contains information about how the states evolve. It is particularly useful in tracking applications. However, when the above model is employed merely for parameter estimation in neural network models with stationary data, there is no need for the matrix A.

Despite the fact that the data is processed in batches, the model of equation (1) allows the weights to be time varying. It is, therefore, possible to deal with non-stationary data sets. In the event of the data being stationary, we should expect the process noise term to vanish. Consequently, if we know that the data is stationary, the estimate of the process noise can be used to determine how well the model explains the data. In Section 6, we demonstrate this method on a few stationary data sets.

Our objective is to estimate the model states (MLP weights) $\widehat{\boldsymbol{\theta}}_t$ and the set of parameters $\boldsymbol{\varphi} \triangleq \{R, Q, A, \boldsymbol{\mu}, \Pi\}$ given the measurements $\{\mathbf{x}_{1:N}, \mathbf{y}_{1:N}\}$[1], where $\boldsymbol{\mu}$ and Π denote the mean and covariance of the Gaussian prior $p(\boldsymbol{\theta}_0|\boldsymbol{\varphi})$.

3 The extended Kalman smoother

One of the earliest implementations of the extended Kalman filter (EKF) to train MLPs is due to Singhal and Wu (Singhal and Wu 1988). The algorithm's computational complexity is of the order $\mathcal{O}(cm^2)$ multiplications per time step. Shah, Palmieri and Datum (Shah, Palmieri and Datum 1992) and Puskorius and Feldkamp (Puskorius and Feldkamp 1991) have proposed various approximations to the weights covariance so as to simplify this problem. Here, we extend the work in this area by proposing an algorithm to estimate the noise covariances R and Q and the initial conditions $\boldsymbol{\mu}$ and Π. The method is also more accurate as it involves smoothing instead of plain filtering.

Smoothing often entails forward and backward filtering over a segment of data so as to obtain improved averaged estimates. Various techniques have been proposed to accomplish this goal (Gelb 1974, Jazwinski 1970). In our work, we make use of the well known Rauch-Tung-Striebel smoother (Rauch, Tung and Striebel 1965). The forward filtering stage involves computing the estimates $\widehat{\boldsymbol{\theta}}_t$ and P_t, over a segment of N samples, with the following EKF

[1] We adopt the notation $\mathbf{z}_{1:N} \triangleq \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N\}$. In addition, we suppress the input variables $\mathbf{x}$ in the arguments of the probability distributions.

recursions:

$$
\begin{aligned}
\widehat{\boldsymbol{\theta}}_{t+1|t} &= A\widehat{\boldsymbol{\theta}}_t \\
P_{t+1|t} &= AP_tA' + Q \\
K_{t+1} &= P_{t+1|t}G'_{t+1}\big(R + G_{t+1}P_{t+1|t}G'_{t+1}\big)^{-1} \\
\widehat{\boldsymbol{\theta}}_{t+1} &= \widehat{\boldsymbol{\theta}}_{t+1|t} + K_{t+1}\big(\mathbf{y}_{t+1} - \widehat{\mathbf{f}}(\mathbf{x}_{t+1}, \widehat{\boldsymbol{\theta}}_{t+1|t})\big) \\
P_{t+1} &= P_{t+1|t} - K_{t+1}G_{t+1}P_{t+1|t}
\end{aligned}
$$

where K denotes the Kalman gain, A' the transpose of A and G the Jacobian matrix:

$$
G = \frac{\partial \widehat{\mathbf{f}}(\boldsymbol{\theta}, \mathbf{x})}{\partial \boldsymbol{\theta}}\bigg|_{(\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}})} = \begin{bmatrix} \frac{\partial \widehat{\mathbf{f}}_1(\boldsymbol{\theta},\mathbf{x})}{\partial \boldsymbol{\theta}_1} & \frac{\partial \widehat{\mathbf{f}}_2(\boldsymbol{\theta},\mathbf{x})}{\partial \boldsymbol{\theta}_1} & \cdots & \frac{\partial \widehat{\mathbf{f}}_c(\boldsymbol{\theta},\mathbf{x})}{\partial \boldsymbol{\theta}_1} \\ \frac{\partial \widehat{\mathbf{f}}_1(\boldsymbol{\theta},\mathbf{x})}{\partial \boldsymbol{\theta}_2} & & & \\ \vdots & & & \vdots \\ \frac{\partial \widehat{\mathbf{f}}_1(\boldsymbol{\theta},\mathbf{x})}{\partial \boldsymbol{\theta}_m} & & \cdots & \frac{\partial \widehat{\mathbf{f}}_c(\boldsymbol{\theta},\mathbf{x})}{\partial \boldsymbol{\theta}_m} \end{bmatrix}'
$$

Subsequently, the Rauch-Tung-Striebel smoother makes use of the following backward recursions:

$$
\begin{aligned}
J_{t-1} &= P_{t-1}A'P^{-1}_{t|t-1} \\
\widehat{\boldsymbol{\theta}}_{t-1|N} &= \widehat{\boldsymbol{\theta}}_{t-1}J_{t-1}\big(\widehat{\boldsymbol{\theta}}_{t|N} - A\widehat{\boldsymbol{\theta}}_{t-1}\big) \\
P_{t-1|N} &= P_{t-1} + J_{t-1}\big(P_{t|N} - P_{t|t-1}\big)J'_{t-1} \\
P_{t,t-1|N} &= P_tJ'_{t-1} + J_t\big(P_{t+1,t|N} - AP_t\big)J'_{t-1}
\end{aligned}
$$

where the parameters, covariance and cross-covariance are defined as follows:

$$
\begin{aligned}
\widehat{\boldsymbol{\theta}}_{t|N} &= \mathbb{E}\big(\boldsymbol{\theta}_t|\mathbf{y}_{1:N}\big) \\
P_{t|N} &= \mathbb{E}\big((\boldsymbol{\theta}_t - \widehat{\boldsymbol{\theta}}_t)(\boldsymbol{\theta}_t - \widehat{\boldsymbol{\theta}}_t)'|\mathbf{y}_{1:N}\big) \\
P_{t,t-1|N} &= \mathbb{E}\big((\boldsymbol{\theta}_t - \widehat{\boldsymbol{\theta}}_t)(\boldsymbol{\theta}_{t-1} - \widehat{\boldsymbol{\theta}}_{t-1})'|\mathbf{y}_{1:N}\big)
\end{aligned}
$$

They may be initialised with the following values:

$$
\begin{aligned}
\widehat{\boldsymbol{\theta}}_{N|N} &= \widehat{\boldsymbol{\theta}}_N \\
P_{N|N} &= P_N \\
P_{N,N-1|N} &= \big(I - K_NG'_N\big)AP_{N-1}
\end{aligned}
$$

The extended Kalman smoother provides a minimum variance Gaussian approximation to the posterior probability density function $p(\boldsymbol{\theta}_{0:t}|\mathbf{y}_{1:t})$ (de Freitas, Niranjan and Gee 1997). In many cases, however, the posterior is multi-modal. This problem can be circumvented by implementing a mixture of Kalman smoothers, where each individual smoother approximates a particular mode.

4 The EM Algorithm

So far, we have shown that given a set of parameters $\boldsymbol{\varphi} = \{R, Q, A, \boldsymbol{\mu}, \Pi\}$ and a matrix of N measurements $\mathbf{y}_{1:N}$, it is possible to compute the expected values of the states with an extended Kalman smoother. In this section, we present a treatment of the EM algorithm that will allow us to learn the parameters $\boldsymbol{\varphi}$ of nonlinear state space models.

The EM algorithm is an iterative method for finding a mode of the likelihood function $p(\mathbf{y}_{1:N}|\boldsymbol{\varphi})$. Roughly speaking, it proceeds as follows: (E-step 1) estimate the states $\boldsymbol{\theta}_{0:N}$ given a set of parameters $\boldsymbol{\varphi}$, (M-step 1) estimate the parameters given the new states, (E-step 2) re-estimate the states with the new parameters, and so forth. The most remarkable attribute of the EM algorithm is that it ensures an increase in the likelihood function at each iteration. However, as the EKF can only provide an approximation to the true states $\boldsymbol{\theta}_{0:N}$ in the E step, the EM algorithm to train MLPs is not necessarily guaranteed to converge.

To gain more insight into the EM method, let us express the likelihood function as follows:

$$p(\mathbf{y}_{1:N}|\boldsymbol{\varphi}) = p(\mathbf{y}_{1:N}|\boldsymbol{\varphi})\frac{p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi})}{p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi})} = \frac{p(\boldsymbol{\theta}_{0:N},\mathbf{y}_{1:N}|\boldsymbol{\varphi})}{p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi})}$$

Taking the logarithms of both sides yields the following identity:

$$\ln p(\mathbf{y}_{1:N}|\boldsymbol{\varphi}) = \ln p(\boldsymbol{\theta}_{0:N},\mathbf{y}_{1:N}|\boldsymbol{\varphi}) - \ln p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi})$$

Let us treat $\boldsymbol{\theta}_{0:N}$ as a random variable with distribution $p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi}^{\text{old}})$, where $\boldsymbol{\varphi}^{\text{old}}$ is the current guess. If we then take expectations on both sides of the previous identity, while remembering that the left hand side does not depend on $\boldsymbol{\theta}_{0:N}$, we get:

$$\ln p(\mathbf{y}_{1:N}|\boldsymbol{\varphi}) = \mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N},\mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) - \mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi})\big) \tag{2}$$

where the expectations involve averaging over the matrix $\boldsymbol{\theta}_{0:N}$ under $p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi}^{\text{old}})$. For example:

$$\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N},\mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) = \int \big(\ln p(\boldsymbol{\theta}_{0:N},\mathbf{y}_{1:N}|\boldsymbol{\varphi})\big)p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi}^{\text{old}})\mathrm{d}\boldsymbol{\theta}_{0:N}$$

It is well known that the second term on the right side of equation (2) is maximised for $\boldsymbol{\varphi}^{\text{old}}$. That is:

$$\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi}^{\text{old}})\big) \geq \mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N},\boldsymbol{\varphi})\big)$$

for any $\boldsymbol{\varphi}$. To apply the EM algorithm, we need to compute the first term on the right hand side of equation (2) repeatedly. The aim is to maximise this term at each iteration. One method of maximising it is discussed in detail in the next section. For the time being, let us assume that we can maximise it, that is:

$$\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N},\mathbf{y}_{1:N}|\boldsymbol{\varphi}^{\text{new}})\big) \geq \mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N},\mathbf{y}_{1:N}|\boldsymbol{\varphi}^{\text{old}})\big)$$

Then, it follows that the likelihood function also increases at every iteration. To demonstrate this important result, consider the change in likelihood for a single iteration:

$$\ln p(\mathbf{y}_{1:N}|\boldsymbol{\varphi}^{\text{new}}) - \ln p(\mathbf{y}_{1:N}|\boldsymbol{\varphi}^{\text{old}}) = \Big(\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi}^{\text{new}})\big) - \mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi}^{\text{old}})\big)\Big)$$
$$- \Big(\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N}, \boldsymbol{\varphi}^{\text{new}})\big) - \mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N}, \boldsymbol{\varphi}^{\text{old}})\big)\Big)$$

The right hand side of the above equation is positive because we are averaging under the distribution $p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N}, \boldsymbol{\varphi}^{\text{old}})$. Consequently, the likelihood function is guaranteed to increase at each iteration. The EM algorithm's name originates from the steps that are required to increase $\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big)$, namely compute the Expectation and then Maximise it. The EM algorithm thus involves the following steps:

Initialisation : Start with a guess for $\boldsymbol{\varphi}^0$.

E-step : Determine the expected log-likelihood density function of the complete data given the current estimate $\boldsymbol{\varphi}^{\text{old}}$:

$$\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) = \int \big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N}, \boldsymbol{\varphi}^{\text{old}}) d\boldsymbol{\theta}_{0:N}$$

M-step : Compute a new value of $\boldsymbol{\varphi}$ that maximises the expected log-likelihood of the complete data. The maximum can be found by simple differentiation of the expected log-likelihood with respect to $\boldsymbol{\varphi}$.

5 The EM algorithm for nonlinear state space models

To derive the EM algorithm for nonlinear state space models, we need to develop an expression for the likelihood of the completed data. We assume that the likelihood of the data given the states, the initial conditions and the evolution of the states can be represented by Gaussian distributions. In particular, if the initial mean and covariance of the states is given by $\boldsymbol{\mu}$ and Π, then:

$$p(\boldsymbol{\theta}_0|\boldsymbol{\varphi}) = \frac{1}{(2\pi)^{m/2}|\Pi|^{1/2}} \exp\Big[-\frac{1}{2}(\boldsymbol{\theta}_0 - \boldsymbol{\mu})'\Pi^{-1}(\boldsymbol{\theta}_0 - \boldsymbol{\mu})\Big]$$
$$p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1}, \boldsymbol{\varphi}) = \frac{1}{(2\pi)^{m/2}|Q|^{1/2}} \exp\Big[-\frac{1}{2}(\boldsymbol{\theta}_t - A\boldsymbol{\theta}_{t-1})'Q^{-1}(\boldsymbol{\theta}_t - A\boldsymbol{\theta}_{t-1})\Big]$$
$$p(\mathbf{y}_t|\boldsymbol{\theta}_t, \boldsymbol{\varphi}) = \frac{1}{(2\pi)^{c/2}|R|^{1/2}} \exp\Big[-\frac{1}{2}(\mathbf{y}_t - \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t))'R^{-1}(\mathbf{y}_t - \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t))\Big]$$

Under the model assumptions of uncorrelated noise sources and Markov state evolution, the likelihood of the complete data is given by:

$$p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi}) = p(\boldsymbol{\theta}_0|\boldsymbol{\varphi}) \prod_{t=1}^{N} p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1}, \boldsymbol{\varphi}) \prod_{t=1}^{N} p(\mathbf{y}_t|\boldsymbol{\theta}_t, \boldsymbol{\varphi})$$

Hence, the log-likelihood of the complete data is given by the following expression:

$$
\begin{aligned}
\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi}) \quad = \quad & -\sum_{t=1}^{N} \Big[\frac{1}{2}(\mathbf{y}_t - \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t))' R^{-1} (\mathbf{y}_t - \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)) \Big] - \frac{N}{2} \ln |R| \\
& -\sum_{t=1}^{N} \Big[\frac{1}{2}(\boldsymbol{\theta}_t - A\boldsymbol{\theta}_{t-1})' Q^{-1} (\boldsymbol{\theta}_t - A\boldsymbol{\theta}_{t-1}) \Big] - \frac{N}{2} \ln |Q| \\
& -\frac{1}{2}(\boldsymbol{\theta}_0 - \boldsymbol{\mu})' \Pi^{-1} (\boldsymbol{\theta}_0 - \boldsymbol{\mu}) - \frac{1}{2} \ln |\Pi| - \frac{Nc + (N+1)m}{2} \ln(2\pi)
\end{aligned}
\tag{3}
$$

As discussed in the previous section, all we need to do now is to compute the expectation of $\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})$ and then differentiate the result with respect to the parameters $\boldsymbol{\varphi}$ so as to maximise it. The EM algorithm for nonlinear state space models will thus involve computing the expected values of the states and covariances with the extended Kalman smoother and then maximising the parameters $\boldsymbol{\varphi}$ with the formulae obtained by differentiating the expected log-likelihood.

5.1 Computing the expectation of the log-likelihood

If we take the expectation of the log-likelihood for the complete data, by averaging over $\boldsymbol{\theta}_{0:N}$ under the distribution $p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N}, \boldsymbol{\varphi}^{\text{old}})$, we get the following expression:

$$
\begin{aligned}
\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) \quad = \quad & -\frac{N}{2} \ln |R| - \frac{N}{2} \ln |Q| - \frac{1}{2} \ln |\Pi| - \frac{Nc + (N+1)m}{2} \ln(2\pi) \\
& -\sum_{t=1}^{N} \frac{1}{2} \mathbb{E}\Big[\mathbf{y}_t' R^{-1} \mathbf{y}_t - \mathbf{y}_t' R^{-1} \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t) \\
& \qquad - \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)' R^{-1} \mathbf{y}_t + \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)' R^{-1} \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t) \Big] \\
& -\sum_{t=1}^{N} \frac{1}{2} \mathbb{E}\Big[\boldsymbol{\theta}_t' Q^{-1} \boldsymbol{\theta}_t - \boldsymbol{\theta}_t' Q^{-1} A \boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}_{t-1}' A' Q^{-1} \boldsymbol{\theta}_t \\
& \qquad + \boldsymbol{\theta}_{t-1}' A' Q^{-1} A \boldsymbol{\theta}_{t-1} \Big] \\
& -\frac{1}{2} \mathbb{E}\Big[\boldsymbol{\theta}_0' \Pi^{-1} \boldsymbol{\theta}_0 - \boldsymbol{\theta}_0' \Pi^{-1} \boldsymbol{\mu} - \boldsymbol{\mu}' \Pi^{-1} \boldsymbol{\theta}_0 + \boldsymbol{\mu}' \Pi^{-1} \boldsymbol{\mu} \Big]
\end{aligned}
$$

We need to digress briefly to compute the expectation of the measurements mapping $\widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)$. We should recall that the EKF approximation to this mapping is given by:

$$
\widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t) = \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t) + \frac{\partial \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)}{\partial \boldsymbol{\theta}_t} \Big|_{(\boldsymbol{\theta}_t = \widehat{\boldsymbol{\theta}}_{t|N})} (\boldsymbol{\theta}_t - \widehat{\boldsymbol{\theta}}_{t|N}) + \cdots
$$

Consequently, if we take expectations on both sides of the equation, we get:

$$
\mathbb{E}\big(\widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)\big) \approx \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)
$$

and

$$
\begin{aligned}
&\mathbb{E}\big((\widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t) - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t))(\widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t) - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t))' \big) \\
&\qquad \approx \mathbb{E}\Big[\Big(\widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t) + \frac{\partial \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)}{\partial \boldsymbol{\theta}_t} \Big|_{(\boldsymbol{\theta}_t = \widehat{\boldsymbol{\theta}}_{t|N})} (\boldsymbol{\theta}_t - \widehat{\boldsymbol{\theta}}_{t|N}) - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t) \Big) \\
&\qquad\quad \Big(\widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t) + \frac{\partial \widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)}{\partial \boldsymbol{\theta}_t} \Big|_{(\boldsymbol{\theta}_t = \widehat{\boldsymbol{\theta}}_{t|N})} (\boldsymbol{\theta}_t - \widehat{\boldsymbol{\theta}}_{t|N}) - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t) \Big)' \Big] \\
&\qquad = G_t P_{t|N} G_t'
\end{aligned}
$$

Hence, under the distribution $p(\boldsymbol{\theta}_{0:N}|\mathbf{y}_{1:N}, \boldsymbol{\varphi}^{\text{old}})$, it follows that:

$$
\mathbb{E}\big(\widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)\widehat{\mathbf{f}}(\boldsymbol{\theta}_t, \mathbf{x}_t)'\big) \approx G_t P_{t|N} G_t' + \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)\widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)'
$$

Using this approximation and the fact that the trace and expectation operators are linear, the expectation of the log-likelihood becomes:

$$
\begin{aligned}
\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) \approx & -\frac{N}{2}\ln|R| - \frac{N}{2}\ln|Q| - \frac{1}{2}\ln|\Pi| - \frac{Nc + (N+1)m)}{2}\ln(2\pi) \\
& - \sum_{t=1}^{N} \frac{1}{2}\text{tr}\Big(R^{-1}\Big[\mathbf{y}_t\mathbf{y}_t' - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)\mathbf{y}_t' - \mathbf{y}_t\widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)' \\
& \qquad\qquad + \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)\widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)' + G_t P_{t|N} G_t' \Big] \Big) \\
& - \sum_{t=1}^{N} \frac{1}{2}\text{tr}\Big(Q^{-1}\Big[\widehat{\boldsymbol{\theta}}_{t|N}\widehat{\boldsymbol{\theta}}_{t|N}' + P_{t|N} - 2A(\widehat{\boldsymbol{\theta}}_{t|N}\widehat{\boldsymbol{\theta}}_{t-1|N}' + P_{t,t-1|N})' \\
& \qquad\qquad + A(\widehat{\boldsymbol{\theta}}_{t-1}\widehat{\boldsymbol{\theta}}_{t-1|N}' + P_{t-1|N})A' \Big] \Big) \\
& - \frac{1}{2}\text{tr}\Big(\Pi^{-1}\Big[\widehat{\boldsymbol{\theta}}_{0|N}\widehat{\boldsymbol{\theta}}_{0|N}' + P_{0|N} - 2\widehat{\boldsymbol{\theta}}_{0|N}\boldsymbol{\mu}' + \boldsymbol{\mu}\boldsymbol{\mu}' \Big] \Big)
\end{aligned}
$$

Completing squares and using the following abbreviations:

$$
\begin{aligned}
\Gamma &= \sum_{t=1}^{N} \widehat{\boldsymbol{\theta}}_{t|N}\widehat{\boldsymbol{\theta}}_{t|N}' + P_{t|N} \\
\Delta &= \sum_{t=1}^{N} \widehat{\boldsymbol{\theta}}_{t-1|N}\widehat{\boldsymbol{\theta}}_{t-1|N}' + P_{t-1|N} \\
\Upsilon &= \sum_{t=1}^{N} \widehat{\boldsymbol{\theta}}_{t|N}\widehat{\boldsymbol{\theta}}_{t-1|N}' + P_{t,t-1|N}
\end{aligned}
$$

we get our final expression for the approximate expectation of the log-likelihood:

$$
\begin{aligned}
\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) \quad \approx \quad & -\frac{N}{2}\ln|R| - \frac{N}{2}\ln|Q| - \frac{1}{2}\ln|\Pi| - \frac{Nc + (N+1)m}{2}\ln(2\pi) \\
& - \sum_{t=1}^{N} \frac{1}{2}\text{tr}\Big(R^{-1}\Big[\big(\mathbf{y}_t - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)\big)\big(\mathbf{y}_t - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t)\big)' \\
& \qquad\qquad + G_t P_{t|N} G_t' \Big] \Big) \\
& - \frac{1}{2}\text{tr}\Big(Q^{-1}\Big[\Gamma - 2A\Upsilon' + A\Delta A' \Big] \Big) \\
& - \frac{1}{2}\text{tr}\Big(\Pi^{-1}\Big[(\widehat{\boldsymbol{\theta}}_{0|N} - \boldsymbol{\mu})(\widehat{\boldsymbol{\theta}}_{0|N} - \boldsymbol{\mu})' + P_{0|N} \Big] \Big)
\end{aligned} \tag{4}
$$

5.2 Differentiating the expected log-likelihood

To maximise the expected value of the log-likelihood with respect to the parameters $\boldsymbol{\varphi}$, we need to compute the derivatives with respect to each parameter individually. This is done in the subsequent sections, where we make use of some results of matrix differentiation (Graham 1981).

5.2.1 Maximum with respect to A

Differentiating the expected log-likelihood with respect to A yields:

$$\begin{aligned}\frac{\partial}{\partial A}\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) &\approx -\frac{1}{2}\frac{\partial}{\partial A}\mathrm{tr}\left(Q^{-1}\left[\,\Gamma - 2A\Upsilon' + A\Delta A'\,\right]\right) \\ &= -\frac{1}{2}\left(-2Q^{-1}\Upsilon + 2Q^{-1}A\Delta\right)\end{aligned}$$

Equating this result to zero yields the value of A that maximises the approximate log-likelihood:

$$A = \Upsilon\Delta^{-1} \tag{5}$$

5.2.2 Maximum with respect to R

Differentiating the expected log-likelihood with respect to R^{-1} gives:

$$\begin{aligned}\frac{\partial}{\partial R^{-1}}\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) &\approx \frac{\partial}{\partial R^{-1}}\left(\frac{N}{2}\ln|R^{-1}| - \sum_{t=1}^{N}\frac{1}{2}\mathrm{tr}\left(R^{-1}\left[\,G_t P_{t|N} G_t'\right.\right.\right. \\ &\qquad \left.\left.\left. + (\mathbf{y}_t - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t))(\mathbf{y}_t - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t))'\,\right]\right)\right) \\ &= \frac{N}{2}R - \sum_{t=1}^{N}\frac{1}{2}\left(G_t P_{t|N} G_t'\right. \\ &\qquad \left. + (\mathbf{y}_t - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t))(\mathbf{y}_t - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t))'\right)\end{aligned}$$

Hence, by equating the above result to zero, the approximate maximum of the log-likelihood with respect to R is given by:

$$R = \frac{1}{N}\sum_{t=1}^{N}\left(G_t P_{t|N} G_t' + (\mathbf{y}_t - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t))(\mathbf{y}_t - \widehat{\mathbf{f}}(\widehat{\boldsymbol{\theta}}_{t|N}, \mathbf{x}_t))'\right) \tag{6}$$

5.2.3 Maximum with respect to Q

Following the same steps, the derivative of the expected log-likelihood with respect to Q^{-1} is given by:

$$\frac{\partial}{\partial Q^{-1}}\mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) \approx \frac{N}{2}Q - \frac{1}{2}\left(\Gamma - 2A\Upsilon' + A\Delta A'\right)$$

Hence, equating to zero and using the result that $A = \Upsilon\Delta^{-1}$, the approximate maximum of the log-likelihood with respect to Q is given by:

$$Q = \frac{1}{N}\left(\Gamma - \Upsilon\Delta^{-1}\Upsilon'\right) \tag{7}$$

5.2.4 Maximum with respect to $\boldsymbol{\mu}$

It is also possible to treat the initial conditions as parameters and improve their estimates in the M-step of the EM algorithm. Finding the derivative of the expected log-likelihood with respect to the initial states gives:

$$\frac{\partial}{\partial \boldsymbol{\mu}} \mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) \approx \frac{1}{2}\Pi^{-1}\Big(-2\widehat{\boldsymbol{\theta}}_{0|N} + 2\boldsymbol{\mu}\Big)$$

Hence, the initial value for the states should be:

$$\boldsymbol{\mu} = \widehat{\boldsymbol{\theta}}_{0|N} \tag{8}$$

5.2.5 Maximum with respect to Π

The derivative of the expected log-likelihood with respect to the inverse of the initial covariance gives:

$$\frac{\partial}{\partial \Pi^{-1}} \mathbb{E}\big(\ln p(\boldsymbol{\theta}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\varphi})\big) \approx \frac{\Pi}{2} - \frac{1}{2}\Big((\widehat{\boldsymbol{\theta}}_{0|N} - \boldsymbol{\mu})(\widehat{\boldsymbol{\theta}}_{0|N} - \boldsymbol{\mu})' + P_{0|N}\Big)$$

Therefore, the initial covariance should be updated as follows:

$$\Pi = P_{0|N} \tag{9}$$

5.3 The E and M steps for nonlinear state space models

We can now prescribe the EM algorithm for nonlinear state space models as follows:

Initialisation : Start with a guess for $\boldsymbol{\varphi} = \{R, Q, A, \boldsymbol{\mu}, \Pi\}$.

E-step : Determine the expected values $\widehat{\boldsymbol{\theta}}_{t|N}$, $P_{t|N}$ and $P_{t,t-1|N}$, given the current parameter estimate $\boldsymbol{\varphi}^{\text{old}}$, using the extended Kalman smoothing equations described in Section 4.2.

M-step : Compute new values of the parameters $\boldsymbol{\varphi} = \{R, Q, A, \boldsymbol{\mu}, \Pi\}$ using equations (5) to (9).

The complexity of this algorithm is $\mathcal{O}(m^3 N)$ operations per iteration.

6 Experiments

6.1 Simple regression example

For the purposes of demonstrating the method, we address the problem of learning the following nonlinear mapping from (x_1, x_2) to y:

$$y = 4\sin(x_1 - 2) + 2x_2 + 5 + \eta$$

where x_1 and x_2 were chosen to be two normal random sequences of 700 samples each. The noise process η was sampled from a zero mean Gaussian distribution with variance $R = 0.5$.

An MLP with four sigmoidal neurons in the hidden layer and a linear neuron in the output layer was used to approximate the measurements mapping. After 50 iterations, as shown in Figure 1, the estimate of observation variance R converges to the true value. In addition, the trace of the process noise covariance Q goes to zero. Note that since the data is stationary, the trace of Q should tend to zero. That is, the trace of Q can be used to provide an estimate of how well the model fits the data. The innovations covariance (variance of the evidence function $p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \widehat{\boldsymbol{\theta}}_{t|t-1}, Q_{t-1}, R_{t-1})$) tends to R over the entire data set, as shown in Figure 2. The top plot of this figure shows that the MLP approximates the true function without fitting the noise. That is, it generalises well. Figure 1 also shows how the log-likelihood increases at each step, thereby demonstrating that the algorithm converges well.

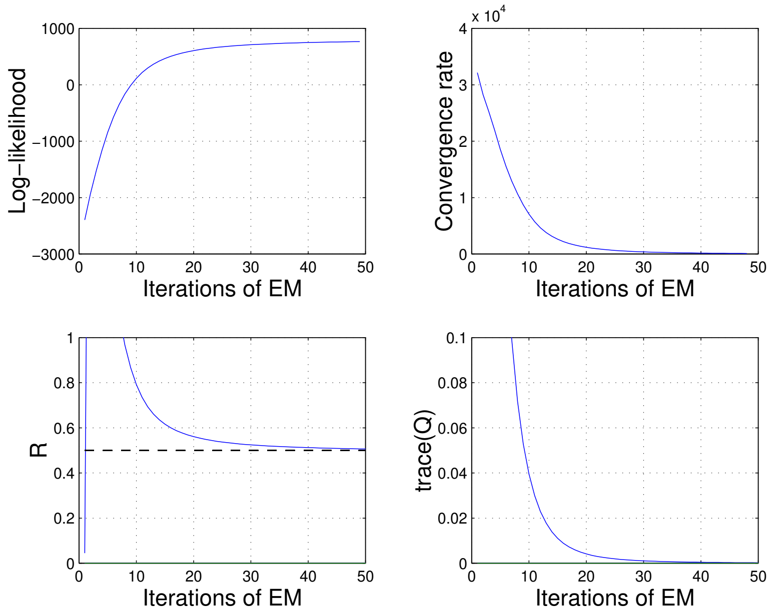

Figure 1: The top plots show the log-likelihood function and the convergence rate (log-likelihood slope) for the simple regression problem. The bottom plots show the convergence of the measurements noise covariance R and the trace of the process noise covariance Q.

6.2 Robot arm mapping

This data set is often used as a benchmark to compare neural network algorithms[2]. It involves implementing a model to map the joint angle of a robot arm (x_1, x_2) to the position of the end of the arm (y_1, y_2). The data were generated from the following model:

$$\begin{aligned} y_1 &= 2.0\cos(x_1) + 1.3\cos(x_1 + x_2) + \epsilon_1 \\ y_2 &= 2.0\sin(x_1) + 1.3\sin(x_1 + x_2) + \epsilon_2 \end{aligned}$$

[2]The data set can be found at `http://wol.ra.phy.cam.ac.uk/mackay/`

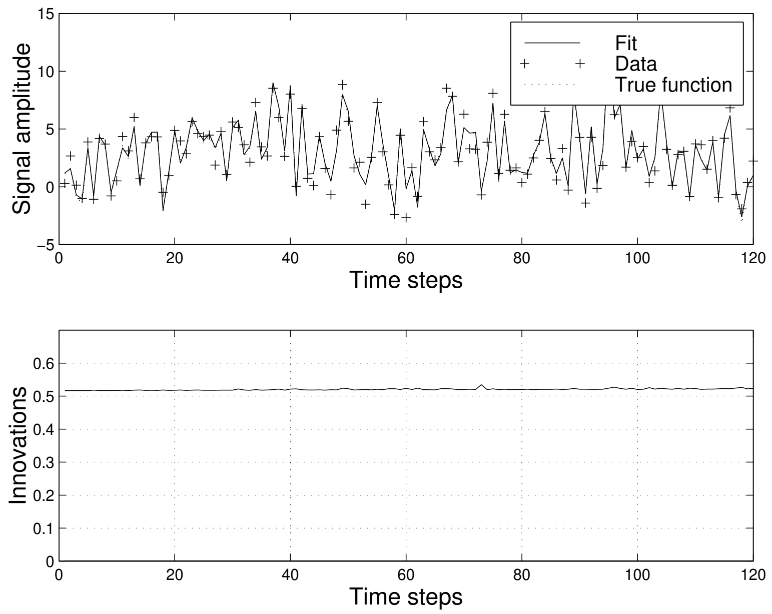

Figure 2: The top plot shows that the MLP fit, for the regression example, approximates the true function (the former is almost exactly on top of the latter); it does not fit the noise. The bottom plot shows that the uncertainty in the predictions (innovations) converges to the uncertainty engendered by the measurement noise.

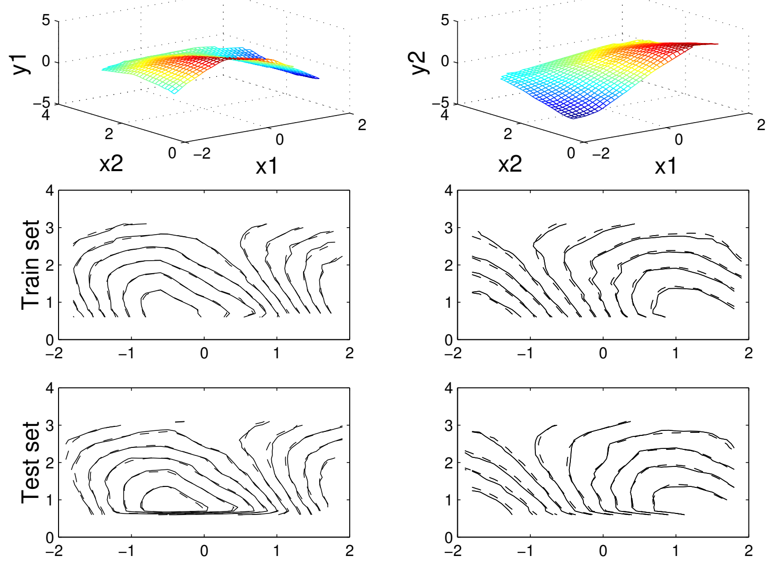

Figure 3: The top plots show the training data surfaces corresponding to each coordinate of the robot arm's position. The Middle and bottom plots show the training and validation data [- -] and the respective MLP mappings [—].

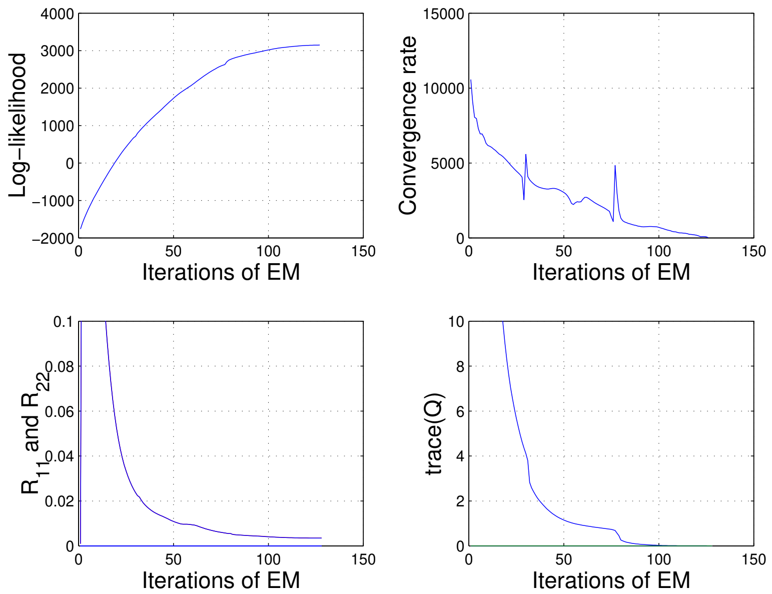

Figure 4: The top plots show the log-likelihood function and the convergence rate (log-likelihood slope) for the robot arm problem. The bottom plots show the convergence of the diagonal entries of the measurements noise covariance R (almost identical) and the trace of the process noise covariance Q.

where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, $\sigma = 0.05$. We use the first 200 observations of the data set to train our models and the last 200 observations to test them.

Figure 3 shows the 3D plots of the training data and the contours of the training and test data. The contour plots also include the typical approximations that were obtained using our algorithm and an MLP with 2 linear output neurons and 20 sigmoidal hidden neurons. Figure 4 illustrates the convergence of the algorithm. In this particular run the training and test mean square errors were 0.0057 and 0.0081 (the minimum bound being $2\sigma^2 = 0.005$). Our mean square errors are of the same magnitude as the ones reported by other researchers (Andrieu, de Freitas and Doucet 1999, Holmes and Mallick 1998, Mackay 1992, Neal 1996, Rios Insua and Müller 1998). Figure 4 also shows the two diagonal entries of the measurements noise covariance and the trace of the process noise covariance. They behave as expected.

6.3 Classification with medical data

Here, we consider an interesting nonlinear classification data set[3] collected as part of a study to identify patients with muscle tremor (Roberts, Penny and Pillot 1996, Spyers-Ashby, Bain and Roberts 1998). The data was gathered from a group of patients (9 with, primarily, Parkinson's disease or multiple sclerosis) and from a control group (not exhibiting the disease). Arm muscle tremor was measured with a 3-D mouse and a movement tracker in three linear

[3]The data is available at `http://www.ee.ic.ac.uk/hp/staff/sroberts.html`

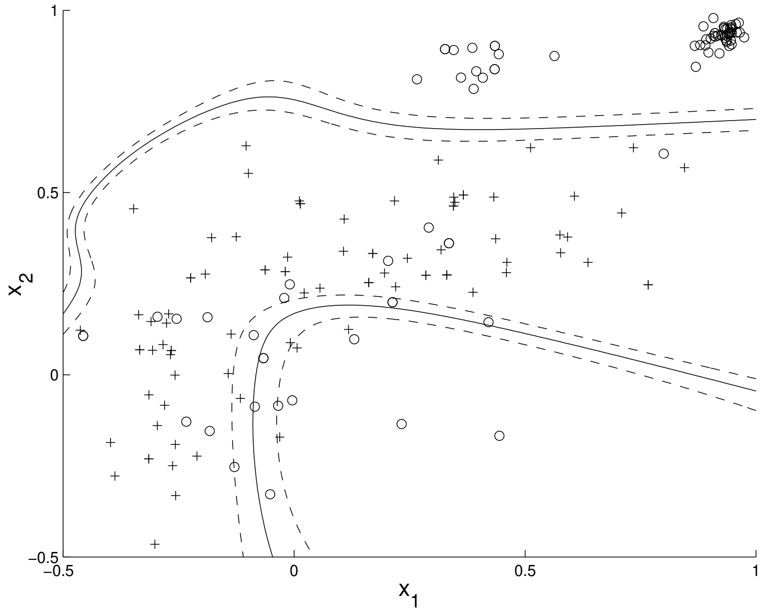

Figure 5: Classification boundaries (–) and confidence intervals (- -) for the MLP classifier. The circles indicate patients, while the crosses represent the control group.

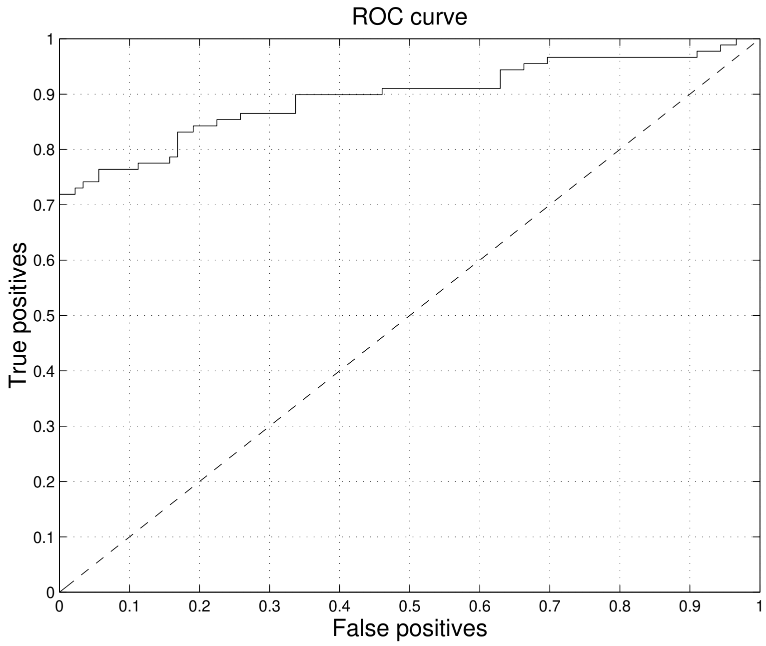

Figure 6: Receiver operating characteristic (ROC) of the classifier for the tremor test data.

and three angular directions. The time series of the measurements were parameterised using a set of autoregressive models. The number of features was then reduced to two (Roberts et

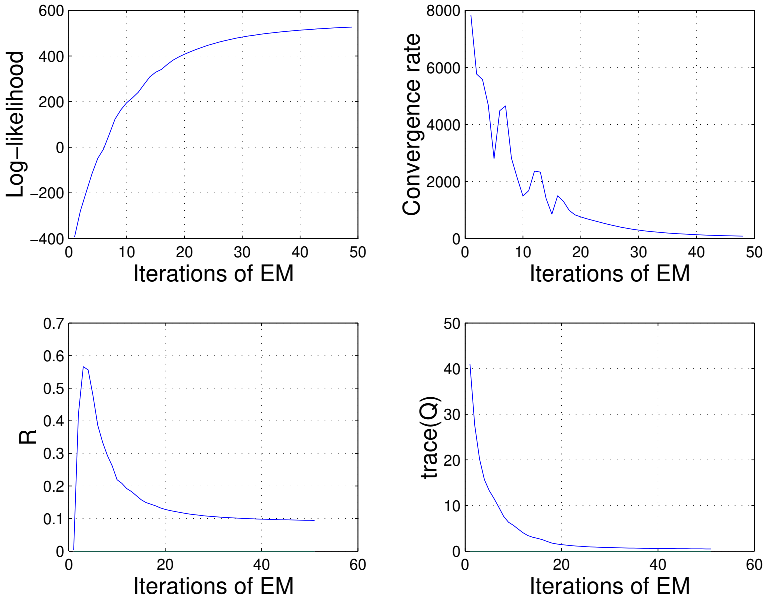

Figure 7: The top plots show the log-likelihood function and the convergence rate (log-likelihood slope) for the tremor data classification problem. The bottom plots show the convergence of the measurements noise covariance R and the trace of the process noise covariance Q.

al. 1996). Figure 5 shows a plot of these features for patient ($\circ$) and control groups ($+$). The figure also shows the decision boundaries (solid lines) and confidence intervals (dashed lines) obtained with an MLP, consisting of 10 sigmoidal hidden neurons and an output linear neuron. We should point out, however, that having an output linear neuron leads to a classification framework based on discriminants. An alternative and more principled approach, which we do not pursue here, is to use a logistic output neuron so that the classification scheme is based on probabilities of class membership.

The size of the confidence intervals for the decision boundary is given by the noise variance (σ^2). These intervals are a measure of uncertainty on the threshold that we apply to the linear output neuron. Our confidence of correctly classifying a sample occurring within these intervals should be very low. The receiver operating characteristic (ROC) curve, shown in Figure 6, indicates that we can expect to detect patients with a 70% confidence without making any mistakes. The percentage of classification errors in the test set was found to be 15.17. This error is of the same magnitude as previous results (Roberts and Penny 1998). Finally, the convergence properties of the EM algorithm for this application are illustrated in Figure 7.

7 Conclusions

In this paper, we derived an EM algorithm to estimate the neural network weights, measurement noise and model uncertainty (in the case of stationary data) jointly. We found that it performs well in terms of model accuracy and generalisation ability. Further research avenues include extending the method to other types of noise processes, establishing theoretical convergence bounds and investigating ways of efficiently initialising the algorithm so as to avoid local minima. The latter problem can be circumvented, to a certain extent, by stochastic global optimisation algorithms. Finally, we have made the software available at `http://www-svr.eng.cam.ac.uk/~jfgf`.

Acknowledgements

We would like to thank Mari Ostendorf (Boston University), Vassilis Digalakis (Technical University of Crete), David Melvin (Cambridge Clinical School), Ben North (Oxford University) and the anonymous reviewers for their valuable comments. We are very grateful to Zoubin Ghahramani (University of Toronto) for his help. We would also like to thank David Stoffer (University of Pittsburgh) for making available some of his technical reports and Stephen Roberts and Will Penny (Imperial College of London) for the tremor data.

João FG de Freitas is financially supported by two University of the Witwatersrand Merit Scholarships, a Foundation for Research Development Scholarship (South Africa), an ORS Award (UK) and a Trinity College External Research Studentship (Cambridge).

References

Andrieu, C., de Freitas, J. F. G. and Doucet, A. (1999). Robust full Bayesian learning for neural networks, *Technical Report CUED/F-INFENG/TR 343*, Cambridge University Engineering Department.

Chen, C. F. (1981). The EM algorithm to the multiple indicators and multiple causes model via the estimation of the latent variable, *Journal of the American Statistical Association* **76**(375): 704–708.

de Freitas, J. F. G., Niranjan, M. and Gee, A. H. (1997). Hierarchical Bayesian-Kalman models for regularisation and ARD in sequential learning, *Technical Report CUED/F-INFENG/TR 307*, Cambridge University Engineering Department.

Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society Series B* **39**: 1–38.

Digalakis, V., Rohlicek, J. R. and Ostendorf, M. (1993). ML estimation of a stochastic linear system with the EM algorithm and its application to speech recognition, *IEEE Transactions on Speech and Audio Processing* **1**(4): 431–442.

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

Gelman, A., Carlin, J. B., Stern, H. S. and Rubin, D. B. (1995). *Bayesian Data Analysis*, Chapman and Hall.

Ghahramani, Z. (1997). Learning dynamic Bayesian networks, *in* C. L. Giles and M. Gori (eds), *Adaptive Processing of Temporal Information*, Vol. 1387 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag.

Graham, A. (1981). *Kronecker Products and Matrix Calculus with Applications*, Ellis Horwood Limited.

Holmes, C. C. and Mallick, B. K. (1998). Bayesian radial basis functions of variable dimension, *Neural Computation* **10**(5): 1217–1233.

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

Mackay, D. J. C. (1992). A practical Bayesian framework for backpropagation networks, *Neural Computation* **4**(3): 448–472.

Meng, X. L. and Rubin, D. B. (1992). Recent extensions to the EM algorithm, *in* J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F. Smith (eds), *Bayesian Statistics*, Vol. 4, Oxford University Press, pp. 307–320.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*, Lecture Notes in Statistics No. 118, Springer-Verlag, New York.

North, B. and Blake, A. (1998). Learning dynamical models using expectation-maximisation, *Proceedings of the 6th International Conference on Computer Vision*, Mumbai, India, pp. 129–134.

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.

Rao, R. P. N. and Ballard, D. H. (1997). Dynamic model of visual recognition predicts neural response properties in the visual cortex, *Neural Computation* **9**(4): 721–763.

Rauch, H. E., Tung, F. and Striebel, C. T. (1965). Maximum likelihood estimates of linear dynamic systems, *AIAA Journal* **3**(8): 1445–1450.

Rios Insua, D. and Müller, P. (1998). Feedforward neural networks for nonparametric regression, *Technical Report 98–02*, Institute of Statistics and Decision Sciences, Duke University.

Roberts, S. J. and Penny, W. D. (1998). Bayesian neural networks for classification: How useful is the evidence framework?, To appear in *Neural Networks*.

Roberts, S. J., Penny, W. D. and Pillot, D. (1996). Novelty, confidence and errors in connectionist systems, *IEE Colloquium on Intelligent Sensors and Fault Detection*, number 261, pp. 10/1–10/6.

Roweis, S. and Ghahramani, Z. (1999). A unifying review of linear Gaussian models, *Neural Computation* **11**(2): 305–345.

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

Shumway, R. H. and Stoffer, D. S. (1982). An approach to time series smoothing and forecasting using the EM algorithm, *Journal of Time Series Analysis* **3**(4): 253–264.

Shumway, R. H. and Stoffer, D. S. (1991). Dynamic linear models with switching, *Journal of the American Statistical Association* **86**(415): 763–769.

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.

Spyers-Ashby, J. M., Bain, P. and Roberts, S. J. (1998). A comparison of fast Fourier transform (FFT) and autoregressive (AR) spectral estimation techniques for the analysis of tremor data, *Journal of Neuroscience Methods* **83**(1): 35–43.

Watson, M. W. and Engle, R. F. (1983). Alternative algorithms for the estimation of dynamic factor, MIMIC and varying coefficient regression models, *Journal of Econometrics* **23**(3): 385–400.