Dynamic Logistic Regression

William D. Penny and Stephen J. Roberts
{w.penny, s.j.roberts}@ic.ac.uk
Department of Electrical and Electronic Engineering, Imperial College, London SW7 2BT, U.K.

Abstract — *We propose an online learning algorithm for training a logistic regression model on nonstationary classification problems. The nonstationarity is captured by modelling the weights in a logistic regression classifier as evolving according to a first order Markov process. The weights are updated using the extended Kalman filter formalism and nonstationarities are tracked by inferring a time-varying state noise variance parameter. We describe an algorithm for doing this based on maximising the evidence of updated predictions. The algorithm is illustrated on a number of synthetic problems.*

I. Introduction

This paper proposes an online learning algorithm for training a logistic regression model on nonstationary classification problems. By nonstationary[1] we mean that the statistics of each class may vary with time or, equivalently from the classification perspective, that the optimal decision boundary changes with time.

The simplest online algorithm for training a logistic regression model is steepest descent. This algorithm has a learning rate parameter which we would wish to be small in regions of stationarity, to allow convergence, and large in regions of nonstationarity, to allow tracking. Our problem therefore, can be seen as equivalent to estimating a time-varying learning rate.

In this paper the weights are modelled as evolving according to a first order Markov process. This state-space approach (where the state is the weight vector) possesses an implicit learning rate which decreases as more data is seen from the same stationary regime and increases when a new regime is encountered. The ability to adapt to a new regime is based on inferring a state-noise variance parameter. We present an algorithm for doing this based on maximising the evidence of model predictions.

This work was supported in part by the UK Engineering and Physical Sciences Research Council (EPSRC), grant number GR/K79062.

[1] We assume the usual technical definition of nonstationarity (see eg. [2]) which for a sequence of input-output pairs, $\{\boldsymbol{x}_t, z_t\}$, is that the joint density $p(\boldsymbol{x}_t, z_t; \boldsymbol{x}_{t+1}, z_{t+1}; ...\boldsymbol{x}_{t+T} z_{t+T})$ may vary with time, t.

In section 2 we briefly review steepest descent and Bayesian algorithms for stationary logistic regression. In section 3 we describe how the Bayesian formalism is extended to the case of nonsatationary logistic regression and in section 4 describe algorithms for inferring the state noise variance. In section 5 we present results from numerical examples. Throughout the paper we consider only the case of binary logistic regression although the methods are readily extended to multiple classes. Further details of this research are available in a technical report [10].

II. Stationary Logistic Regression

In a logistic regression model the predicted class label, $\hat{Z}_t$, is generated according to

$$y_t = P(\hat{Z}_t = 1 \mid \boldsymbol{w}) = g(\boldsymbol{w}^T \boldsymbol{x}_t) \tag{1}$$

where $g(a_t)$ is the logistic function

$$g(a_t) = \frac{\exp(a_t)}{1 + \exp(a_t)} \tag{2}$$

and a_t is the 'activation' at time t, where $\boldsymbol{w}$ is a column vector of weights and $\boldsymbol{x}_t$ is a column vector of inputs. Note that this is a stationary model because the weight vector $\boldsymbol{w}$ is not time varying.

A. Steepest Descent Learning

The weights in a logistic regression model can be adapted using a steepest descent algorithm which maximises the likelihood, $L(z_t)$, of the class labels where

$$L(z_t) = P(\hat{Z}_t = z_t \mid \boldsymbol{w}) = y_t^{z_t}(1 - y_t)^{1 - z_t} \tag{3}$$

Maximising the likelihood can be achieved by minimising the negative log likelihood or *cross entropy*

$$G(z_t, y_t) = -\ln L(z_t) = -[z_t \ln y_t + (1 - z_t)\ln(1 - y_t)] \tag{4}$$

The corresponding steepest descent learning rule is [1]

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \alpha \boldsymbol{x}_t (z_t - y_t) \tag{5}$$

where α is the learning rate and z_t is the true class label. A problem with steepest descent learning for stationary logistic regression is that there are two conflicting requirements on the learning rate parameter; at the start of learning we wish α to be large so that learning is fast but at the end of learning α must be small to ensure that the learning process converges. Whilst it is true that α can be set to be proportional to $1/t$, methods based on this approach are entirely *ad hoc* and have practical difficulties e.g. how is the constant of proportionality chosen ?

This problem can be overcome by taking a Bayesian approach instead of a maximum likelihood approach. The Bayesian algorithm, as we shall see, posseses an implicit learning rate parameter which naturally decreases as learning progresses.

B. Bayesian Learning

In Bayesian learning we consider the evolution of the distribution of weight values, $p(\hat{\boldsymbol{w}})$. This distribution is often approximated as a Gaussian. Before data point $(\boldsymbol{x}_t, z_t)$ arrives the weights are distributed according to the prior distribution $N(\hat{\boldsymbol{w}}_{t-1}, \boldsymbol{\Sigma}_{t-1})$ where $N(\boldsymbol{m}, \boldsymbol{C})$ is a normal distribution with mean $\boldsymbol{m}$ and covariance $\boldsymbol{C}$. After arrival of the new data point this prior is updated to a posterior, $N(\hat{\boldsymbol{w}}_t, \boldsymbol{\Sigma}_t)$, according to Bayes rule.

The mean and covariance of this posterior can be evaluated using a variational approximation due to Jaakkola and Jordan [4], by Newton's method as described by Spiegelhalter and Lauritzen [13] or by the Extended Kalman Filter as described by Niranjan [9]. In this paper, we consider the latter two methods.

In Bayesian learning the prior weight distribution (ie. before the class label is observed and the weights are updated) implies a prior distribution on the activation $P(a_t) = N(\bar{a}_t, s_t^2)$ where $\bar{a}_t = \hat{\boldsymbol{w}}_{t-1}^T \boldsymbol{x}_t$ is the mean activation and s_t^2 is the variance of the activation given by

$$s_t^2 = \boldsymbol{x}_t^T \boldsymbol{\Sigma}_{t-1} \boldsymbol{x}_t \tag{6}$$

This gives rise to the concept of a 'moderated' output, $\tilde{y}_t$, defined as

$$\tilde{y}_t = P(\hat{Z}_t = 1) = \int P(\hat{Z}_t = 1 \mid a_t) p(a_t) da_t \tag{7}$$

This integral can be accurately approximated by [8]

$$\tilde{y}_t = g(K(s_t)\bar{a}_t) \tag{8}$$

where

$$K(s_t) = \left(1 + \frac{\pi s_t^2}{8}\right)^{-1/2} \tag{9}$$

The unmoderated output, y_t, is given by $y_t = g(\bar{a}_t)$. The 'moderation' changes the actual output, y_t, to a moderated output, $\tilde{y}_t$, which is nearer to 0.5 by an amount which is dependent on the prior uncertainty on the model parameters. Moderated outputs are typically better than unmoderated outputs in terms of the likelihood of predictions [8].

If the posterior distribution is approximated by a Gaussian, its mean and covariance can be found via Newton's method as formulated by Spiegelhalter and Lauritzen [13]. The update equations are

$$\boldsymbol{\Sigma_t} = \boldsymbol{\Sigma}_{t-1} - \frac{y_t(1-y_t)}{1+y_t(1-y_t)s_t^2} (\boldsymbol{\Sigma}_{t-1}\boldsymbol{x}_t)(\boldsymbol{\Sigma}_{t-1}\boldsymbol{x}_t)^T \tag{10}$$

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \boldsymbol{\Sigma}_t \boldsymbol{x}_t (z_t - y_t) \tag{11}$$

A comparison of equation 11 with equation 5 shows that the *effective* learning rate is equal to the covariance matrix $\boldsymbol{\Sigma}_t$. We can therefore evaluate an average effective learning rate as $<\alpha>= \frac{1}{p} Tr(\boldsymbol{\Sigma}_t)$ where p is the number of weights. The learning rate is therefore adaptive; as the classifier sees more data from the same stationary regime the covariance reduces which, in turn, reduces the learning rate.

Alternatively, we could use the Extended Kalman Filter (EKF) paradigm to approximate the posterior distribution. This produces an update equation for the covariance which is identical to that obtained from Newton's method. But the update for the mean is different and is given by

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \frac{\boldsymbol{\Sigma}_{t-1}}{1+y_t(1-y_t)s_t^2} \boldsymbol{x}_t (z_t - y_t) \tag{12}$$

This is different to equation 11 in that learning will be slower for larger activation uncertainty, s_t^2, or output uncertainty, $y_t(1-y_t)$.

III. Nonstationary Logistic Regression

We now consider a nonstationary logistic regression model where the weights evolve dynamically and the classifications are made according to the usual logistic function

$$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \boldsymbol{n}_t \tag{13}$$

$$y_t = g(\boldsymbol{w}_t^T \boldsymbol{x}_t) \tag{14}$$

where $\boldsymbol{n}_t$ is 'state' noise generated from a zero mean normal distribution with isotropic covariance matrix $q_t \boldsymbol{I}$. The important difference between this nonstationary model and the stationary model of equation 1 is that the weights $\boldsymbol{w}_t$ are now time-varying.

We now take a Bayesian-learning approach to the problem of estimating the weights online. The situation is exactly the same as for the stationary case except that the prior distribution is now $N(\hat{\boldsymbol{w}}_{t-1}, \boldsymbol{\Sigma}_{t-1} + q_t \boldsymbol{I})$. This is because we are adding state noise before the new observation is made (see equation 13). If we use the Newton updates to estimate the posterior distribution then the mean and covariance are given by equations 10, 11 and 6 except that $\boldsymbol{\Sigma}_{t-1}$ is replaced by

$\mathbf{\Sigma}_{t-1} + q_t \boldsymbol{I}$. Our dynamic nonstationary learning rule is hence:

$$\begin{aligned}\mathbf{\Sigma}_{\boldsymbol{t}} &= (\mathbf{\Sigma}_{t-1} + q_t \boldsymbol{I}) - \frac{y_t(1-y_t)}{1+y_t(1-y_t)s_t^2} \\ &\quad [(\mathbf{\Sigma}_{t-1} + q_t \boldsymbol{I})\boldsymbol{x}_t][(\mathbf{\Sigma}_{t-1} + q_t \boldsymbol{I})\boldsymbol{x}_t]^T \qquad (15)\end{aligned}$$

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \mathbf{\Sigma}_t \boldsymbol{x}_t (z_t - y_t) \qquad (16)$$

where s_t^2 is the variance of the activation and is given by

$$s_t^2 = \boldsymbol{x}_t^T (\mathbf{\Sigma}_{t-1} + q_t \boldsymbol{I}) \boldsymbol{x}_t \qquad (17)$$

The EKF learning rule is identical except that equation 16 is replaced by equation 12. The use of an isotropic covariance matrix for the state noise, $q_t \boldsymbol{I}$, assumes that the weight changes are of similar magnitude. Whilst this may be a reasonable assumption if we consider just the input weights, it may not be so reasonable if we consider the input weights *and* the bias weight. However, by scaling the input data to have zero mean the bias weights will fluctuate around zero. Alternatively, one could use a separate noise variance parameter for the bias weight or, indeed, have separate noise variance parameters for every weight as discussed in the context of prediction problems by DeFreitas and Niranjan [3].

IV. Estimating state noise

We now describe an online method for updating the state noise parameter, q_t, based on maximising the 'evidence' of the observations. This approach has been used in the context of Kalman Filters and Extended Kalman Filters and was developed originally by Jazwinski [6] and has since been re-discovered by DeFreitas and Niranjan and placed in the context of regularization in neural networks [3].

We note, however, that maximising the evidence of the observations in a classification context, as opposed to a prediction context, may not be the optimal strategy. This is because, in the logistic regression model, the observations are class labels which are *binary random variables* (drawn from a binomial distribution with a 'hit' probability given by the output of the logistic unit). Each class label therefore contains much less information (at most 1 bit) than the equivalent observation in the prediction context (a Gaussian random variable). The resulting estimates of state noise variance are commensurately noisier.

We therefore investigate an alternative strategy: maximising the evidence of *updated predictions* which, we believe, is a more appropriate criterion for classification problems.

The state noise variance parameter, q_t, which maximises this criterion can be found by a line search procedure which is also described in this section.

A. Maximising Evidence of Updated Predictions

For reasons previously discussed we choose to maximise the evidence of the model trained on the new class label, rather than maximising the evidence of the new class label itself. This can be achieved by minimising

$$G_N(\hat{y}, \tilde{y}) = -\sum_{t=1}^{N} [\hat{y}_t \ln \tilde{y}_t + (1-\hat{y}_t)\ln(1-\tilde{y}_t)] \qquad (18)$$

where $\hat{y}_t = g(\boldsymbol{w}_t^T \boldsymbol{x}_t)$. Note that this 'updated prediction' is based on the updated weights whereas the moderated prediction is based on the weights from the previous time step (if this were not the case, the optimal value of q would be $q = 0$).

This criterion may also be understood from a 'variance-matching' perspective. The maximum evidence criterion corresponds to setting the state noise so that the observed prediction variance is matched by the estimated prediction variance. In the prediction context (see, for example, section 5.3.1 in [3]) these variances parameterise a Gaussian distribution whereas in the classification context the variances parameterise a binomial distribution.

The evidence of *updated predictions* is maximised when the following variance matching criterion is minimised

$$V(\hat{y}, \tilde{y}) = \sum_{t=1}^{N} \hat{y}_t(1-\hat{y}_t) - \tilde{y}_t(1-\tilde{y}_t) \qquad (19)$$

Note that the corresponding criterion for maximising the evidence of *class labels* contains the term $z_t(1 - z_t)$ which is zero. This provides another explanation of why we should maximise the evidence of updated predictions rather than of class labels.

B. Line Search Method

The value of q which minimises $G_N(\hat{y}, \tilde{y})$ can be found with a line search algorithm. Line searches are iterative schemes which, for example, fit a quadratic polynomial to a cost function evaluated at three distinct points and the new parameter estimate is given as the minimum of that quadratic function. The next iteration uses this new point and two of the previous points. In this paper we use Brent's line search algorithm ([11], page 402).

The state noise can be estimated by moving a window along one sample at a time and performing a new line search. This is, however, rather computationally intensive and may be unnecessary as q is likely to evolve on a slower time-scale than the weights. We therefore choose to move the window along N samples at a time and then to re-estimate q ie. $q_t = q_{t-1}$ unless t is exactly divisible by N in which case q_t is re-estimated using line search.

We note that if the window size, N, is set too small then it is possible for the state noise estimates to become infinite. This therefore suggests both that reasonably large values of N should be used and that there may be a need to impose a maximum state noise estimate q_{max}.

In this paper we use the following heuristic for setting N and q_{max}. Firstly, different segments of the time series are used to train different logistic regression models using a batch method (eg. Bayesian logistic regression using the evidence formalism [8]). This allows weight vectors and covariance matrices to be estimated for each segment. We then calculate the difference in weight vector between consecutive segments and form an outer product. This acts as an estimate of the empirical covariance. We then subtract the estimated covariance from the empirical covariance and infer that whatever covariance remains is due to state noise (and not just parameter uncertainty from having finite data sets). The average diagonal entries correspond to estimated q values and we set q_{max} to the median value (a conservative but robust estimate). The window length N is then initially set to a small value, say, 10 and is increased if the line search estimate of q is greater than q_{max}.

In practice, we find that, as expected, maximising the evidence of *updated predictions* is much more robust to choices of N and q_{max} than is maximising the evidence of *class labels* and find that the above heuristic scheme is often not necessary. All of the numerical examples in this paper estimate the state noise variance by maximising the evidence of updated predictions.

V. Results

A. Continuous drift

We generated a 'continuous drift' data set as follows. Two Gaussians were first defined as having means $\boldsymbol{m}_1 = [1,1]^T$ and $\boldsymbol{m}_2 = [-1,-1]^T$ and covariances $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma} = diag(v,v)$ and $N_D = 300$ data points were drawn from each Gaussian. The data were then randomly ordered to create a set of 600 input vectors, $\{\boldsymbol{x}_t\}$. The weights in a logistic regression model were then initialised by assuming class 1 data points came from Gaussian 1 and class 2 from Gaussian 2 and using the relations in [1] (section 3.1.3). The variance v was chosen so that the 70% of the data could be correctly classified (the appropriate value of v was found via Monte Carlo simulation). The weights were then allowed to evolve according to the diffusion in equation 13 where the state noise at time t has isotropic variance, $q(t)$. Various profiles for $q(t)$ were tried, the one we report here being zero except between $t = 200$ and $t = 220$ where $q(t) = 0.1$. The true posterior probability at time t was then evaluated by passing the input at time t through a logistic node, $y_t = g(\boldsymbol{w}_t^T \boldsymbol{x}_t)$. Class labels, z_t, were then generated from this posterior probability.

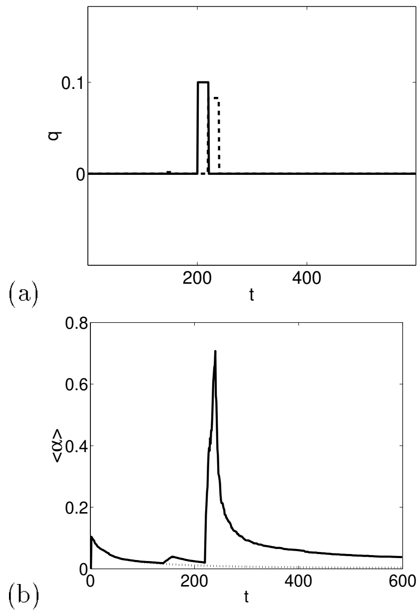

Fig. 1. *(a) Estimate of state noise, q, and (b) effective learning rate of nonstationary rule (solid line) and stationary rule (dotted line). The nonstationary rule increases the effective learning rate in response to the change in classification regime at t =* 200*, whereas the effective learning rate for the stationary rule decreases monotonically as it believes that all data points belong to the same classification regime.*

We then trained logistic regression models using stationary, nonstationary and steepest descent learning algorithms. In the nonstationary model the state noise was estimated using the line search method to maximise the evidence of updated predictions.

Figure 1(a) shows the true state noise and the estimated state noise trajectories and Figure 1(b) shows the corresponding learning rates for the stationary and nonstationary algorithms. Figure 2 shows the true weights and the weights as estimated by the nonstationary rule and Figure 3 compares the cumulative prediction error (as measured by the cross entropy between moderated predictions and true class labels) with that obtained by the stationary and steepest descent algorithms.

The reason why nonstationary learning performs so much better is that the learning rate is increased when the new classification regime is encountered (see Figure 1(b)). The EKF approximation was preferred to Newton's method as it was less sensitive to overestimates of state noise (due to the denominator terms in equation 12).

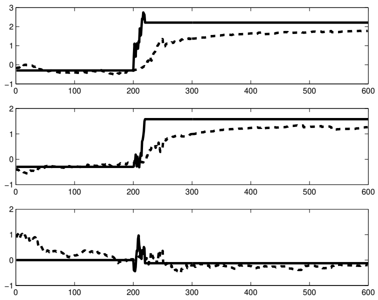

Fig. 2. *Estimation of weights using nonstationary learning rule with the state noise estimated by the line search method; true weights (solid line), estimated weights (dashed line).*

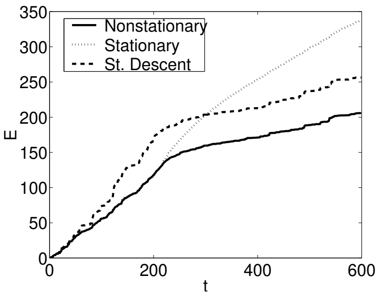

Fig. 3. *Cumulative prediction error, E, of nonstationary, stationary and steepest descent learning rules on a data set which has a slow transition between classification regimes starting at $t = 200$.*

B. Discrete change

We generated data sets A and B, shown in Figure 4. Each data set consists of examples drawn from one of two isotropic Gaussians corresponding to one of two class labels. Each data set is optimally separated by a linear decision boundary shown in the Figure. A data set was then formed by drawing 150 points from data set A followed by 150 points from data set B. Thus at $t = 150$ the optimal decision boundary switches from that shown in Figure 4(a) to that in Figure 4(b). Note that if the points in the overall data set were randomly ordered then it could only be classified with an accuracy of 50% ie. the chance rate. If, however, the temporal ordering is preserved near perfect classification is possible.

Figure 5(a) shows the estimated state noise trajectory and Figure 5(b) shows the corresponding learning rates for the stationary and nonstationary algorithms. Figure 6 compares the cumulative prediction error with that obtained by the stationary and steepest descent algorithms. Again, the nonstationary algorithm is seen to be the superior method as it is able to increase the implicit learning rate when a new classification regime is encountered.

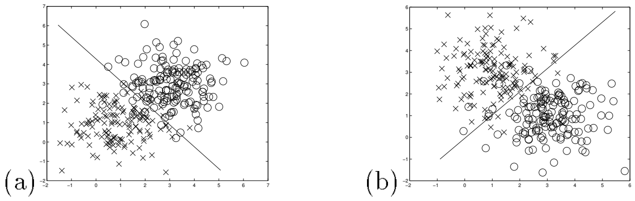

Fig. 4. *Two data sets (a) A and (b) B used to form the discrete change data set.*

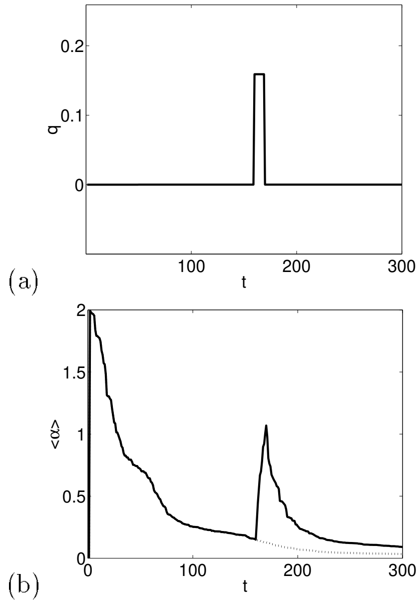

Fig. 5. *(a) Estimate of state noise, q, and (b) effective learning rate of the nonstationary rule (solid line) and the stationary rule (dotted line). The nonstationary rule increases the effective learning rate in response to the change in classification regime at $t = 150$, whereas the effective learning rate for the stationary rule decreases monotonically as it believes that all data points belong to the same classification regime.*

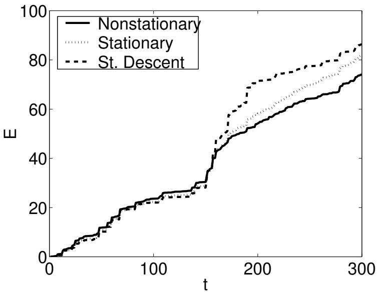

Fig. 6. *Cumulative prediction error, E, of nonstationary, stationary and steepest descent learning rules on a data set which has a discrete transition between classification regimes at $t = 150$.*

VI. Discussion

We have investigated an online learning algorithm for training a logistic regression model on nonstationary classification problems. The nonstationarity is captured by modelling the weights in a logistic regression classifier as evolving according to a first order Markov process. The weights were updated using the extended Kalman filter formalism. Similar approaches exist in the statistics literature and are known as Dynamic Generalised Linear Models (DGLIMs) [15],[14].

The nonstationarities were tracked by inferring a time-varying state noise variance parameter using a maximum evidence algorithm widely used for nonstationary prediction problems [5],[3] but adapted to the classification context by choosing to maximise the evidence of updated predictions rather than of observations.

An alternative approach to estimating state-noise parameters is outlined in Li and Bar-Shalom [7] where a mixture of models, each having a fixed state-noise value is employed. In its simplest incarnation two models would be used; one with zero state noise and one with a 'maximum' state noise. Estimates of state noise are then obtained by estimating mixing coefficients.

Our observation that state-noise estimation is easier if the targets are 'updated predictions', rather than binary class labels, suggests the following heuristic estimation scheme; if the model is tracking the data at all, then the estimated weights at time t should be closer to the true weights at time $t-N$ than are the estimated weights at time $t-N$. Therefore, for the purposes of estimating state noise, the model at time t could be used to generate pseudo-targets for previous time steps ie. $z_{t-N} \approx y(\boldsymbol{x}_{t-N}, \boldsymbol{w}_t)$. Experimentation with such an approach shows that good results can be obtained as long as some initial tracking ability is present in the model. This therefore suggests a multiple-pass algorithm where, firstly, a weight trajectory is estimated and then, secondly, a state noise trajectory is estimated and these two steps are iterated until a consistent solution is reached. Such a scheme is reminiscent of the forward and backward recursions involved in the EM (Expectation-Maximisation) estimation [12] of model parameters and seems a promising avenue for further research. However such 'smoothing' approaches (as opposed to filtering approaches) mean that the algorithms are no longer 'online'.

References

[1] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, 1995.

[2] C. Chatfield. *The Analysis of Time Series: An Introduction.* Chapman and Hall, 1996.

[3] J.F.G. DeFreitas, M. Niranjan, and A.H. Gee. Hierarchical Bayesian-Kalman Models for Regularisation and ARD in Sequential Learning. Technical Report CUED/F-INFENG/TR 307, Department of Engineering, Cambridge University, 1998. Also avaiable from http://svr-www.eng.cam.ac.uk/ jfgf/publications.html.

[4] T. S. Jaakkola and M. I. Jordan. A variational approach to Bayesian logistic regression models and their extensions. Technical Report 9702, MIT Computational Cognitive Science, January 1997.

[5] A. Jazwinski. *Stochastic Processes and Filtering Theory.* Academic Press, 1970.

[6] A.H. Jazwinski. Adaptive Filtering. *Automatica*, 5:475–485, 1969.

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

[8] D.J.C. Mackay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992.

[9] M. Niranjan. Risk prediction in pregnancy. Talk at the Isaac Newton Institute, November 1997.

[10] W.D. Penny and S.J. Roberts. Nonstationary Logistic Regression. Technical report, Department of Electrical Engineering, Imperial College, 1999. Available from http://www.ee.ic.ac.uk/research/neural/wpenny.html.

[11] W. H. Press, S.A. Teukolsky, W.T. Vetterling, and B.V.P. Flannery. *Numerical Recipes in C, second edition.* Cambridge, 1992.

[12] S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11(2):305–346, 1999.

[13] D. J. Spiegelhalter and S. L. Lauritzen. Sequential updating of conditional probabilities on directed graphical structures. *Networks*, 20:579–605, 1990.

[14] M. West and J. Harrison. *Bayesian Forecasting and Dynamic Models.* Springer, 2nd edition, 1997.

[15] M. West, J. Harrison, and H.S. Migon. Dynamic Generalized Linear Models and Bayesian Forecasting. *Journal of the American Statistical Association*, 80(389):73–83, 1985.