HYBRID SEQUENTIAL MONTE CARLO / KALMAN METHODS TO TRAIN NEURAL NETWORKS IN NON-STATIONARY ENVIRONMENTS

João F. de Freitas, Mahesan Niranjan and Andrew H. Gee

Cambridge University Engineering Department
Trumpington Street, Cambridge, CB2 1PZ, UK
jfgf@eng.cam.ac.uk

ABSTRACT

In this paper, we propose a novel sequential algorithm for training neural networks in non-stationary environments. The approach is based on a Monte Carlo method known as the sampling-importance resampling simulation algorithm. We derive our algorithm using a Bayesian framework, which allows us to learn the probability density functions of the network weights and outputs. Consequently, it is possible to compute various statistical estimates including centroids, modes, confidence intervals and kurtosis. The algorithm performs a global search for minima in parameter space by monitoring the errors and gradients at several points in the error surface. This global optimisation strategy is shown to perform better than local optimisation paradigms such as the extended Kalman filter.

1. INTRODUCTION

Sequential training methods are important in many applications of neural networks involving real-time signal processing, where data arrival is inherently sequential. In addition, it is often convenient to adopt a sequential training strategy to deal with non-stationarity in signals, so that information from the recent past is given greater weighting than information from the distant past. One way to sequentially estimate neural network models is to use a state space formulation and the extended Kalman filter (EKF) [3, 8]. This involves local linearisation of the output equation, which can be easily performed, since we only need the derivatives of the output with respect to the unknown parameters. This approach has been employed by several authors, including ourselves. Recently, we demonstrated a number of advanced ideas in this context, using a hierarchical Bayesian framework [2]. In particular, we proposed ways of tuning the noise processes to achieve regularisation in sequential learning tasks. However, local linearisation leading to the EKF algorithm is a gross simplification of the probability densities involved.

In recent years, many researchers in the the statistical and signal processing communities have suggested the use of sequential Monte Carlo estimation methods. These methods provide a complete description of the probability distributions involved in the estimation process and tend to improve the accuracy of the analysis. Typical sequential Monte Carlo methods are often based on the sampling-importance resampling (SIR) algorithm [13]. The methods have been applied to a wide range of problems, including target tracking [6, 7], financial analysis [11], diagnostic measures of fit [11], sequential imputation in missing data problems [9], blind deconvolution [10] and medical prognosis [1].

In this paper we propose a hybrid sequential SIR technique, where each sampling trajectory is updated by an EKF, to train neural network models. The method may be viewed as a global optimisation training strategy, whereby we make use of the errors and gradients at several positions in the error surface to update the network weights. It may also be viewed as an integration problem within a Bayesian framework. One of the advantages of the method is that it allows us to compute the distributions of the networks outputs and weights. Consequently, we can generate several types of statistical estimates, including modes, centroids, confidence intervals, error bars, kurtosis, etc.

2. STATE SPACE BAYESIAN INFERENCE

As in our previous work, we start from a state space representation to model the neural network's evolution in time:

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

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

where ($\mathbf{y}_k \in \Re^m$) denotes the output measurements, ($\mathbf{x}_k \in \Re^d$) the input measurements and ($\mathbf{w}_k \in \Re^q$) the neural network weights. The measurements nonlinear mapping $\mathbf{g}(.)$ is approximated by a multi-layer perceptron (MLP). It is widely known that this neural model exhibits the capacity to approximate any continuous function, to an arbitrary precision, as long as it is not restricted in size. Nonetheless, the work may be easily extended to encompass recurrent networks [12], radial basis networks [8] 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 Gaussian noise with covariance R.

We model the evolution of the network weights by assuming that they depend on their previous value $\mathbf{w}_k$ and a stochastic component $\mathbf{d}_k$. The process noise $\mathbf{d}_k$ may represent our uncertainty in how the parameters evolve, modelling errors or unknown inputs such as target manoeuvres. We assume the process noise to be a zero mean, uncorrelated Gaussian process with covariance Q. In Section 3, we propose that this random walk model can be improved by incorporating gradient information.

The noise terms are assumed to be uncorrelated with the network weights and the initial conditions. Further, we assume the evolution of the states (network weights) to correspond to a Markov

We would like to thank Neil Gordon (Defense Research Agency, UK) Michael Isard (Computer Vision Group, Oxford) and Sue Johnson (CUED, Cambridge) for their helpful assistance. 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 and a Trinity College External Studentship (Cambridge).

process with initial probability $\mathrm{p}(\mathbf{w}_0)$ and transition probability $\mathrm{p}(\mathbf{w}_k|\mathbf{w}_{k-1})$. The observations are assumed to be conditionally independent given the states. These are standard assumptions in a large class of tracking and time series problems.

From a Bayesian perspective, the posterior density $\mathrm{p}(W_k|Y_k)$, where $Y_k = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_k\}$ and $W_k = \{\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_k\}$, constitutes the complete solution to the sequential estimation problem. In many applications, it is of interest to estimate one of its marginals, namely the filtering density $\mathrm{p}(\mathbf{w}_k|Y_k)$. If we know this density, we can easily compute various estimates of the network weights recursively, including centroids, modes, medians and confidence intervals.

The filtering density may be estimated recursively in two stages: prediction and update. In the prediction step, the filtering density $\mathrm{p}(\mathbf{w}_{k-1}|Y_{k-1})$ is propagated into the future via the transition density $\mathrm{p}(\mathbf{w}_k|\mathbf{w}_{k-1})$ as follows:

$$\mathrm{p}(\mathbf{w}_k|Y_{k-1}) = \int \mathrm{p}(\mathbf{w}_k|\mathbf{w}_{k-1})\mathrm{p}(\mathbf{w}_{k-1}|Y_{k-1})\mathrm{d}\mathbf{w}_{k-1} \qquad (3)$$

The transition density is defined in terms of the probabilistic model governing the states' evolution (equation (1)) and the process noise statistics. The update stage involves the application of Bayes' rule when new data is observed:

$$\mathrm{p}(\mathbf{w}_k|Y_k) = \frac{\mathrm{p}(\mathbf{y}_k|\mathbf{w}_k)\mathrm{p}(\mathbf{w}_k|Y_{k-1})}{\mathrm{p}(\mathbf{y}_k|Y_{k-1})} \qquad (4)$$

The likelihood density function is defined in terms of the measurements model (equation (2)).

The prediction and update strategy given by equations (3) and (4) provides an optimal solution to the inference problem, but, unfortunately, it entails multi-dimensional integration. To surmount this problem, we use Monte Carlo integration methods. In Monte Carlo simulation, a set of weighted samples, drawn from the posterior density function of the neural network weights, is used to map the integrations, involved in the inference process, to discrete sums. More precisely, we make use of the following Monte Carlo approximation:

$$\hat{\mathrm{p}}(W_k|Y_k) = \frac{1}{N}\sum_{i=1}^{N}\delta(W_k - W_k^{(i)})$$

where $W_k^{(i)}$ represents the samples used to describe the posterior density and, as before, $\delta(.)$ denotes the Dirac delta function. Consequently, any expectations of the form:

$$\mathbf{E}[f_k(W_k)] = \int f_k(W_k)\mathrm{p}(W_k|Y_k)\mathrm{d}W_k$$

may be approximated by the following estimate:

$$\mathbf{E}[f_k(W_k)] \approx \frac{1}{N}\sum_{i=1}^{N} f_k(W_k^{(i)})$$

where the samples $W_k^{(i)}$ are drawn from the posterior density function. Monte Carlo sampling techniques are an improvement over direct numerical approximation in that they automatically select samples in regions of high probability.

We give a give a brief description of sequential Monte Carlo methods here; a complete mathematical derivation is available in [4]. Figure 1 shows the operation of a generic sequential Monte Carlo method. It embraces the standard assumption that we can sample from the prior $p(\mathbf{w}_0)$ and evaluate the likelihood $p(\mathbf{y}_k|\mathbf{w}_k^{(i)})$ of each sample. Only the fittest samples survive in the update stage. These then proceed to be multiplied, according to their likelihood, in the prediction stage. As mentioned earlier, the update and prediction stages are governed by equations (4) and (3) respectively. It is very instructive to approach the problem from an optimisation perspective. Figures 2 and 3 show the windowed global descent in the error function that is typically observed. The diagrams shed light on the roles played by the noise covariances R and Q. R controls the resolution of the update stage, while Q dictates by how much the cloud of samples is expanded in the prediction stage. This expansion allows the algorithm to explore broader regions in parameter space.

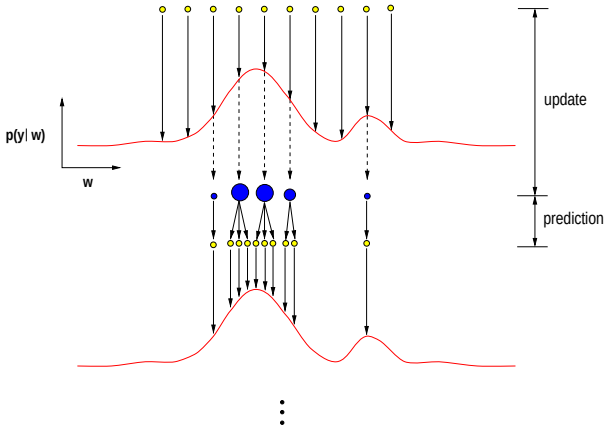

Figure 1: Prediction and update stages in the sequential sampling process. In the update stage, the likelihood of each sample is evaluated. The samples are then propagated according to their likelihood. The size of the black circles indicates the likelihood of a particular sample. In the prediction stage, a process noise term is added to the surviving samples. The samples with higher likelihood are allowed to have more "children". The end result is that the surviving samples provide a better weighted description of the likelihood function.

3. HYBRID SEQUENTIAL SIR

The Monte Carlo conception of optimisation relies solely on probing the error surface at several points as shown in Figures 2 and 3. It fails to embrace all the richness of information implicit in the error surface. For instance, the method could be enhanced by evaluating the gradient and other higher derivatives of the error surface. In order to improve sequential Monte Carlo simulation, we are proposing a new hybrid SIR algorithm. The main feature of the algorithm is that the samples are updated by a gradient descent step in the prediction stage. Within this framework, we can make use of second order statistics by implementing an EKF update:

$$\begin{aligned} \mathbf{w}_{k+1|k} &= \mathbf{w}_k \\ P_{k+1|k} &= P_k^T + Q^* \end{aligned}$$

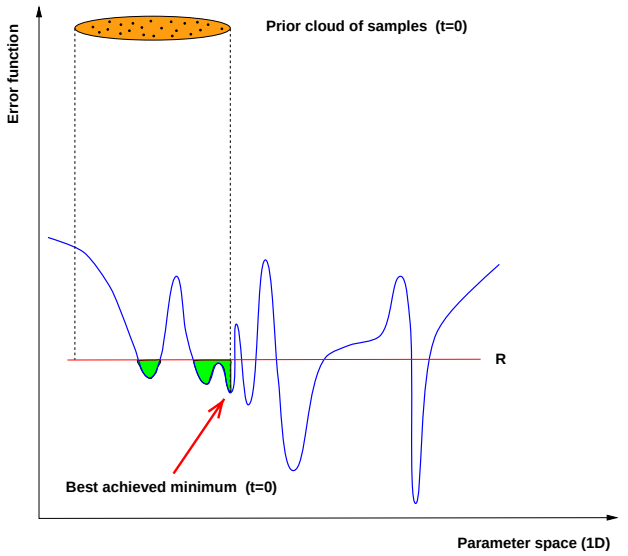

Figure 2: First step of one-dimensional Monte Carlo simulation. A prior cloud (group) of samples is used to explore a region of the error function

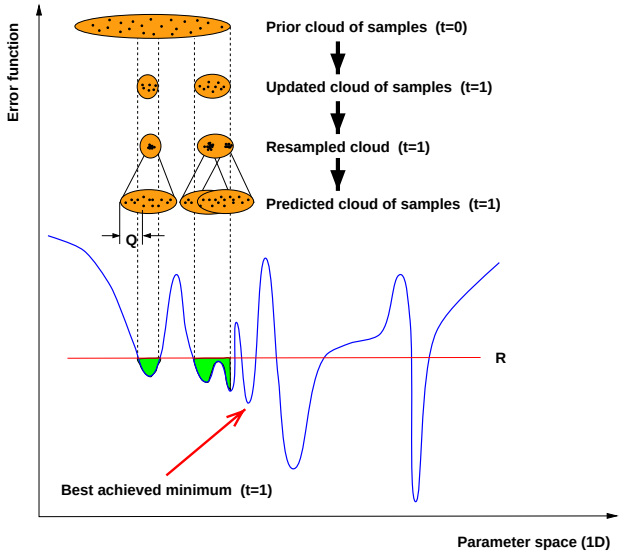

Figure 3: Second step of one-dimensional Monte Carlo simulation. The regions below a threshold (the measurement noise R) of the error function and within the width of the prior cloud of samples determine the size of the updated clouds of samples. The cloud's samples are grouped in the regions of higher likelihood in a resampling stage. Finally, the resampled clouds are expanded by a factor determined by the process noise covariance Q.

$$K_{k+1} = P_{k+1|k}G_{k+1}[R^* + G_{k+1}^T P_{k+1|k} G_{k+1}]^{-1}$$
$$\mathbf{w}_{k+1} = \mathbf{w}_{k+1|K} + K_{k+1}(\mathbf{y}_{k+1} - \mathbf{g}(\mathbf{x}_k, \mathbf{w}_{k+1|k}))$$
$$P_{k+1} = P_{k+1|k} - K_{k+1}G_{k+1}^T P_{k+1|k}$$

where K_{k+1} is known as the Kalman gain matrix and R^* and Q^* are two tuning parameters, whose roles are explained in great detail in [2]. Here, it suffices to say that they control the rate of convergence of the EKF algorithm and the generalisation performance of the neural network. In the general multiple input, multiple output (MIMO) case, $\mathbf{g} \in \Re^m$ is a vector function and G represents the Jacobian matrix:

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

Since the EKF is a suboptimal estimator based on linearisation of a nonlinear mapping, strictly speaking, P_k is an approximation to the covariance matrix. In mathematical terms:

$$P_k \approx \mathbf{E}[(\mathbf{w}_k - \hat{\mathbf{w}}_k)^T(\mathbf{w}_k - \hat{\mathbf{w}}_k)|Y_k]$$

The EKF step, before the resampling stage, allows us to incorporate the latest measurements into the prior. That is, it is equivalent to using the prior $p(\mathbf{w}_k|\mathbf{w}_{k-1}, \mathbf{y}_k)$ before the resampling stage. Another advantage of this method is that the covariance of the weights changes over time. Because the extended Kalman filter is a minimum variance estimator, the covariance of the weights decreases with time. Consequently, as the tracking improves, the variation of the network weights is reduced. This annealing process improves the efficiency of the search for global minima in parameter space and reduces the variance of the estimates. It should be pointed out that the weights need to be resampled with their respective covariances.

The EKF updates may, unfortunately, introduce biases. That is, the estimates will depend on a few modes of the posterior density function. However, these updates lead to a substantial reduction on computational time. One way of avoiding the bias problem could be to define the error surface in terms of a Hamiltonian that accounts for the approximation errors and the momentum of each trajectory. This is the basis of the Hybrid Monte Carlo algorithm [5]. In this algorithm, each trajectory is updated by approximating the Hamiltonian differential equations by a leapfrog discretisation scheme. The discretisation may, however, introduce biases.

4. EXPERIMENTS

We generated input-output data vectors from the following nonlinear, non-stationary state space model:

$$\mathbf{w}_{k+1} = 0.5\mathbf{x}_k + 25\frac{\mathbf{x}_k}{1+\mathbf{x}_k^2} + 8\cos(1.2k) + \mathbf{d}_k$$
$$\mathbf{y}_k = \frac{\mathbf{x}_k^2}{20} + 6\mathcal{S}(0.05k) + 3 + \mathbf{v}_k$$

where $\mathcal{S}(.)$ represents a square wave function. We chose variances equal to $2\sin(0.1k)$ and 0.01 for the measurement and process noises respectively. Subsequently, we proceeded to train an MLP with the hybrid SIR algorithm so as to approximate the measurements equation. That is, we used $\mathbf{x}_k$ as the network input and $\mathbf{y}_k$ as the network output. We chose a two-layer architecture with 10 sigmoidal hidden nodes and a linear output node. To compare the performance of the algorithm to a standard gradient descent method, we also trained the MLP with an EKF[1]. Figure 4 shows

[1] Further details and software are available at the following address: `http://svr-www.eng.cam.ac.uk/~jfgf/software.html`.

the Monte Carlo posterior mean and EKF estimates. The one-step-ahead squared errors corresponding to these estimates were 53 and 71 respectively. Figure 5 shows the estimate of the output proba-

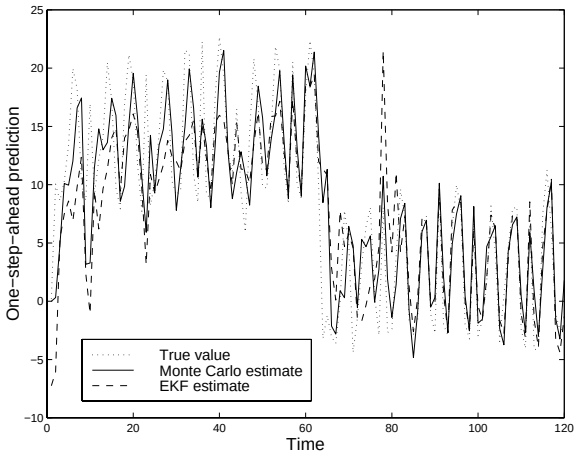

Figure 4: Actual output data [· · ·], Monte Carlo posterior mean estimate using 50 samples [—] and EKF estimate [- -].

bility density function.

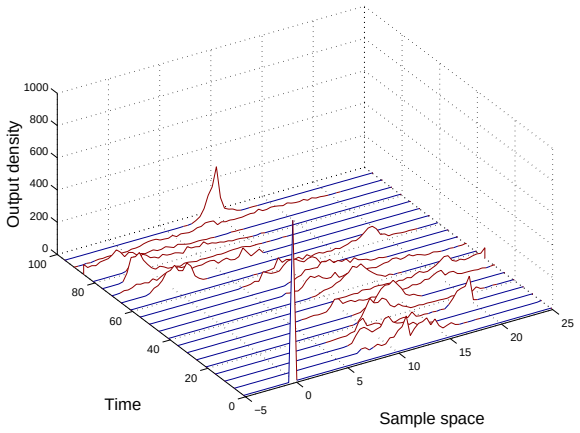

Figure 5: Output probability density function.

The results clearly indicate that the hybrid SIR algorithm results in lower prediction errors than gradient descent based methods. In terms of computational speed, it took approximately 1.5 seconds to process the data with the EKF, while the Monte Carlo simulation, with 50 samples, lasted approximately 200 seconds using a 200MHz Pentium processor. In a recent application to financial options pricing, we also observed that the hybrid SIR method provides better estimation results than the extended Kalman filter [4].

5. SUMMARY

We have described a new sequential algorithm for training neural networks that leads to improved training results. The algorithm is, particularly, suitable to applications involving non-stationarity in the data. Directions for further research include designing algorithms to estimate the noise hyper-parameters and improving the computational efficiency of the algorithm.

6. REFERENCES

[1] C Berzuini, N G Best, W R Gilks, and C Larizza. Dynamic conditional independence models and Markov Chain Monte Carlo methods. *Journal of the American Statistical Association*, 92(440):1403–1412, December 1997.

[2] J F G de Freitas, M Niranjan, and A H Gee. Hierarchichal Bayesian-Kalman models for regularisation and ARD in sequential learning. Technical Report CUED/F-INFENG/TR 307, Cambridge University, http://svr-www.eng.cam.ac.uk/˜jfgf, December 1997.

[3] J F G de Freitas, M Niranjan, and A H Gee. Regularisation in sequential learning algorithms. In M I Jordan, M J Kearns, and S A Solla, editors, *Advances in Neural Information Processing Systems*, volume 10, MIT Press, 1998.

[4] J F G de Freitas, M Niranjan, A H Gee, and A Doucet. Sequential Monte Carlo methods for optimisation of neural network models. Technical Report CUED/F-INFENG/TR 328, Cambridge University, http://svr-www.eng.cam.ac.uk/˜jfgf, July 1998.

[5] S Duane, A D Kennedy, B J Pendleton, and D Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987.

[6] N J Gordon, D J Salmond, and A F M Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings-F*, 140(2):107–113, April 1993.

[7] M Isard and A Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision*, pages 343–356, Cambridge, UK, 1996.

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

[9] A Kong, J S Liu, and W H Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

[10] J S Liu and R Chen. Blind deconvolution via sequential imputations. *Journal of the American Statistical Association*, 90(430):567–576, June 1995.

[11] M K Pitt and N Shephard. Filtering via simulation: Auxiliary particle filters. Technical report, Department of Statistics, Imperial College of London, England, October 1997. Available at http://www.nuff.ox.ac.uk/economics/papers.

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

[13] D B Rubin. Using the SIR algorithm to simulate posterior distributions. In J M Bernardo, M H DeGroot, D V Lindley, and A F M Smith, editors, *Bayesian Statistics 3*, pages 395–402, 1988.