ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 111-116

NAR time-series prediction: a Bayesian framework and an experiment

M. Crucianu, Z. Uhry, J.-P. Asselin de Beauville, R. Boné

Laboratoire d'Informatique, Ecole d'Ingénieurs en Informatique pour l'Industrie
Université de Tours, 64 avenue Jean Portalis, 37200 Tours, FRANCE
{crucianu, bone}@univ-tours.fr

Abstract: We extend the Bayesian framework to Multi-Layer Perceptron models of Non-linear Auto-Regressive time-series. The approach is evaluated on an artificial time-series and some common simplifications are discussed.

1. Introduction

Non-linear Auto-Regressive (NAR, [Box et al. 94]) time-series models are a quite common prediction tool. Because of their general approximating capabilities ([Hornik et al. 89]), feed-forward neural networks (NN) of the Multi-Layer Perceptron (MLP) type ([Rumelhart et al. 86]) are often used to develop NAR models. However, we usually build a single NAR model embodied in a single NN. One of the drawbacks of this approach is that it doesn't allow us to compute confidence limits for the predictions. This problem was partly solved in the Bayesian framework, developed for non-sequential problems (see [MacKay 92], [Neal 92]), by taking into account the influence of model variance on the output distribution.

We shortly present here an extention of this Bayesian framework to NAR models. Some results of the full Bayesian treatment of an artificial NAR time-series are discussed, together with the consequences of some common simplifications. We then compare, on the same problem, a classical linear technique and a standard NN approach to time-series modelling.

2. MLP implementation of NAR models: a Bayesian framework

Let us consider the NAR(p) process $\{X_t\}_{t\geq 1}$ given by

$$X_t = g\left(X_{t-1},\ldots,X_{t-p}\right)+\varepsilon_t, \qquad t\geq 1,\ p\geq 1, \tag{1}$$

where ε_t is an i.i.d. variable $N\left(0;\sigma_\varepsilon^2\right)$, independent of $X_{t-1},\ldots,X_1$. We note by D_t a sample $x_1,\ldots,x_t$ of the process. We want to approximate this model by an MLP

ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 111-116

having *k* units in a single hidden layer. The output unit is linear and its activation value is given by

$$(2) \qquad o(x_p,\ldots,x_1) = \sum_{i=1}^{k} v_i \cdot f\left(\sum_{j=1}^{p} w_{ij} \cdot x_j - w_i \right), \qquad f(y) = \left(1+e^{-y}\right)^{-1}.$$

We consider *k* as fixed *a priori* and we do not deal with its choice here (see [MacKay 95]). We now have a set of functions of parameter $\mathbf{w} = \{w_{ij}, w_i, v_i\} \in \mathbf{R}^{p\cdot k} \times \mathbf{R}^k \times \mathbf{R}^k$. In what follows, the output of a single MLP is $g_{\mathbf{w}}(x_p,\ldots,x_1)$.

We are interested in the *posterior* distribution $p(\mathbf{w}|D)$ given by

$$(3) \qquad p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})\cdot p(\mathbf{w})}{p(D)},$$

$p(D|\mathbf{w})$ being the *likelihood* and $p(\mathbf{w})$ the *prior* distribution. We can then compute

$$(4) \qquad p(x_{t+1}|D_t) = \int_{\mathbf{w}} p(x_{t+1}|D_t,\mathbf{w})\cdot p(\mathbf{w}|D_t)\,d\mathbf{w},$$

the distribution for the predicted value. This distribution allows us to evaluate confidence limits for our prediction by taking into account the noise in the data and the variance of the model. To compute the posterior according to (3), we start by introducing two new parameters. First, we consider that the prior over $\mathbf{w}$ is Gaussian, $p(\mathbf{w}|\alpha) = \sqrt{\frac{\alpha}{2\pi}}^{k(p+2)} \cdot e^{-\frac{\alpha}{2}\|\mathbf{w}\|^2}$. Second, we note $\beta = \frac{1}{\sigma_\varepsilon^2}$. We then have

$$(5) \qquad p(\alpha,\beta,\mathbf{w}|D) = \frac{p(D|\beta,\mathbf{w})\cdot p(\mathbf{w}|\alpha)\cdot p(\alpha,\beta)}{p(D)},$$

because $p(D|\alpha,\beta,\mathbf{w}) = p(D|\beta,\mathbf{w})$ and $p(\mathbf{w}|\alpha,\beta) = p(\mathbf{w}|\alpha)$. The likelihood is

$$(6) \qquad p(D_t|\beta,\mathbf{w}) = p(x_t,\ldots,x_{p+1}|\beta,\mathbf{w};x_p,\ldots,x_1)\cdot p(x_p,\ldots,x_1|\beta,\mathbf{w}).$$

For non-sequential problems, one can easily express the likelihood as a product ([MacKay 92]). Fortunately, this can also be performed for NAR models

$$(7) \qquad p(x_t,\ldots,x_{p+1}|\beta,\mathbf{w};x_p,\ldots,x_1) = \prod_{i=p+1}^{t} p(x_i|\beta,\mathbf{w};x_{i-1},\ldots,x_{i-p}).$$

If we note $E_{D_t}(\mathbf{w}) = \frac{1}{2}\cdot \sum_{i=p+1}^{t} \left[x_i - g_{\mathbf{w}}(x_{i-1},\ldots,x_{i-p})\right]^2$ and remember (1) we obtain

$$(8) \qquad p(D_t|\beta,\mathbf{w}) = \sqrt{\frac{\beta}{2\pi}}^{t-p} \cdot e^{-\beta\cdot E_{D_t}(\mathbf{w})} \cdot p(x_p,\ldots,x_1|\beta,\mathbf{w}).$$

The conditional distribution for the first *p* values is difficult to obtain. However, since *p* is fixed and $p << t$, one may consider $p(x_p,\ldots,x_1|\beta,\mathbf{w}) = p(x_p,\ldots,x_1)$ or simply neglect it. We can eventually compute the distribution for the predicted value,

$$(9) \qquad p(x_{t+1}|D_t) = \int_{\alpha,\beta,\mathbf{w}} p(x_{t+1}|D_t,\alpha,\beta,\mathbf{w})\cdot p(\alpha,\beta,\mathbf{w}|D_t)\,d\alpha\,d\beta\,d\mathbf{w}$$

ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 111-116

for every *t*. If for all $t > t_0$ the changes to $p(\alpha,\beta,\mathbf{w}|D_t)$ can be ignored, we may stop re-estimating it and thus significantly reduce the complexity of the computation.

In contrast, in the standard NN approach one simply looks for the parameter $(\alpha,\beta,\mathbf{w})_{MP}$ that maximizes the likelihood (the prior $p(\mathbf{w})$ is unknown) or the posterior distribution ($p(\mathbf{w})$ is available), and then uses it to evaluate $\hat{x}_{t+1}$. Indeed, if $p(\alpha,\beta,\mathbf{w}|D_t)$ is concentrated around $(\alpha,\beta,\mathbf{w})_{MP}$, we can approximate the integral in (9) by $p(x_{t+1}|D_t,\alpha_{MP},\beta_{MP},\mathbf{w}_{MP})$. But we can no longer obtain confidence limits.

Several approximations were proposed for non-sequential problems ([MacKay 92]) in order to make the Bayesian approach more tractable. Similar approximations are very helpful for NAR time-series:

1° If $p(\alpha,\beta|D)$ is concentrated around (α_{MP},β_{MP}), then α,β and $\mathbf{w}$ can be processed separately and the posterior becomes $p(\mathbf{w}|D) \cong p(\mathbf{w}|\alpha_{MP},\beta_{MP},D)$.

2° We can perform a Gaussian approximation for $p(\mathbf{w}|\alpha_{MP},\beta_{MP},D)$. For an MLP having *k* units in a single hidden layer, the parameter space $\mathbf{W}$ is composed of several equivalent subspaces ($k!\cdot 2^k$ or just $k!$, depending on whether the activation function of the hidden units is symmetric or not). The Gaussian approximation will only hold on these subspaces.

3° Eventually, a Gaussian approximation can be used for $p(x_{t+1}|D)$. The most probable output value and its associated confidence limits can be obtained.

3. Experimenting with an NAR time-series

We performed an experiment on a synthetic NAR time-series. Our purpose was to evaluate the Bayesian approach and to test the approximations just mentioned. We also wanted to compare a linear prediction technique and a standard NN method.

The NAR time-series should have a significantly non-linear but stable behavior. After an important amount of tests, we selected

$$X_t = 0.9\cdot X_{t-1} - (X_{t-1} - X_{t-2})\cdot e^{-20\cdot X_{t-1}^2} + \varepsilon_t \qquad t > 2, \tag{10}$$

with $\sigma_\varepsilon = 0.1$, $X_1 = \varepsilon_1$ and $X_2 = 0.9\cdot X_1 + \varepsilon_2$ (see figure 1 for a sample). Note that we used the same model for the initial values in the Bayesian computation.

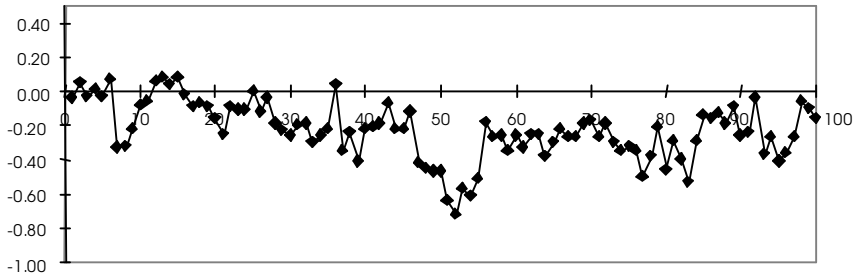

Figure 1: A 100-point sample of the NAR time-series.

ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 111-116

The number of hidden units (k) in the MLPs was fixed to 5 for all the experiments. We considered α and β as independent and set $p(1/\alpha) =$ uniform on $[0;5]$ (quite arbitrarily), $p(1/\beta) =$ uniform on [0;Var (time-series)] (we use prior information).

The posterior distribution $p(\alpha,\beta|D)$ obtained for t = 50 is presented in figure 2. A Gaussian approximation is obviously inappropriate for $p(\alpha|D)$; α_{MP} doesn't give much information about the entire distribution. For t = 200 we still have two different peaks. For t = 400, however, the posterior is nearly Gaussian.

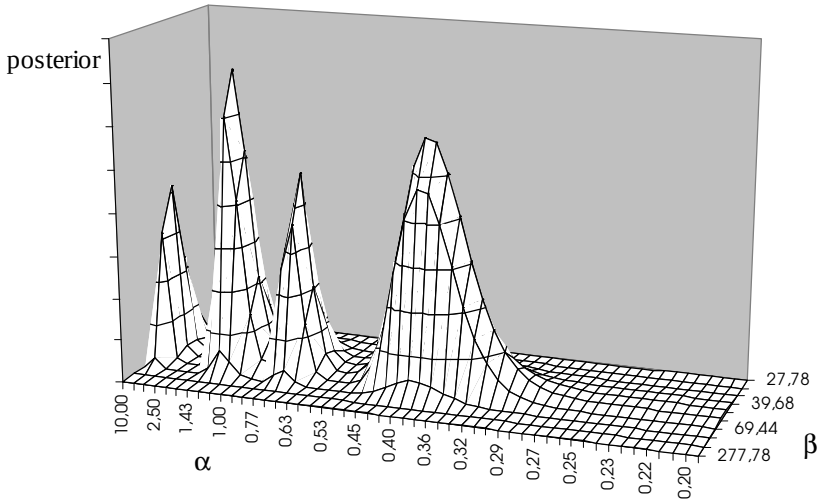

Figure 2: The posterior distribution $p(\alpha,\beta|D)$ for t = 50.

We noticed that the full Bayesian computation of the output distribution (9) was extremely time-consuming. We decided to perform this computation only for t = 50, 200 and 400. Figure 3a presents the distributions for the predicted values x_{t+1}.

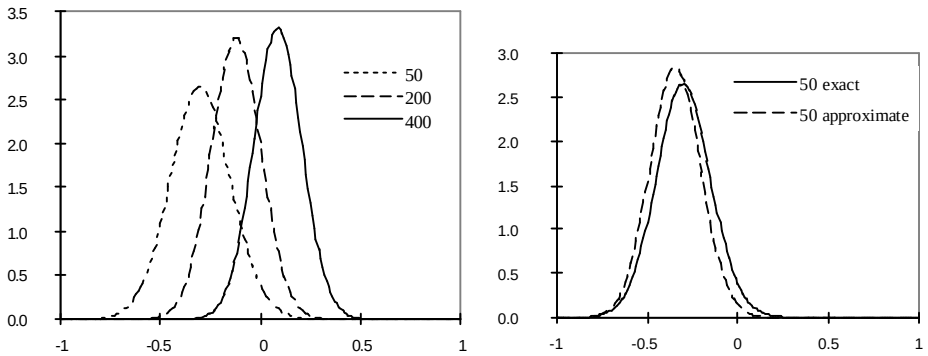

Figure 3: (a) exact and (b) comparison of exact and approximate output distributions.

The three distributions are nearly Gaussian. Table 1 shows the predicted values, with their confidence limits, and compares them to the true values. The confidence interval

ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 111-116

shrinks when t increases: model variance diminishes as more data becomes available, so that the output variance is more and more explained by noise alone.

	$t = 50$	$t = 200$	$t = 400$
predicted x_{t+1}^{MP}	−0.3	−0.12	0.09
95% confidence limits	[**−0.6**; 0.0]	[−0.37; 0.13]	[−0.15; 0.33]
actual x_{t+1}	**−0.636**	−0.169	0.234

Table 1: Most probable predicted values, associated confidence limits and the true values.

When using the approximations mentioned before, we obtain a slightly different output distribution for $t = 50$ (figure 3b), but very similar ones for $t = 200$ or $t = 400$.

We then trained several MLPs by standard backpropagation on the first 200 values of the time-series. The weights were randomly initialized in $[-1;1]$. Training was stopped as soon as the residuals passed randomness tests (runs above and below the median, runs up and down, Box-Pierce). Note that the rather strong amount of noise in the training data makes overfitting unlikely. The residuals also passed a test for Gaussian distribution; estimated standard deviation was 0.107 for the residuals — as compared to 0.1 for the noise injected. The minimal RMS error obtained by an MLP on the training set was 0.0116. However, the difference between the true NAR function and its approximation by the best-fitted MLP is significant (figure 4a). In fact, the transfer function for the MLP is nearly linear on the domain of interest. Indeed, the training set (figure 4b) covers a quasi-linear region of the target NAR function (the non-linear character is not manifest). Preliminary tests have confirmed that such MLPs could approximate well enough the true NAR function on $[-1;1]^2$.

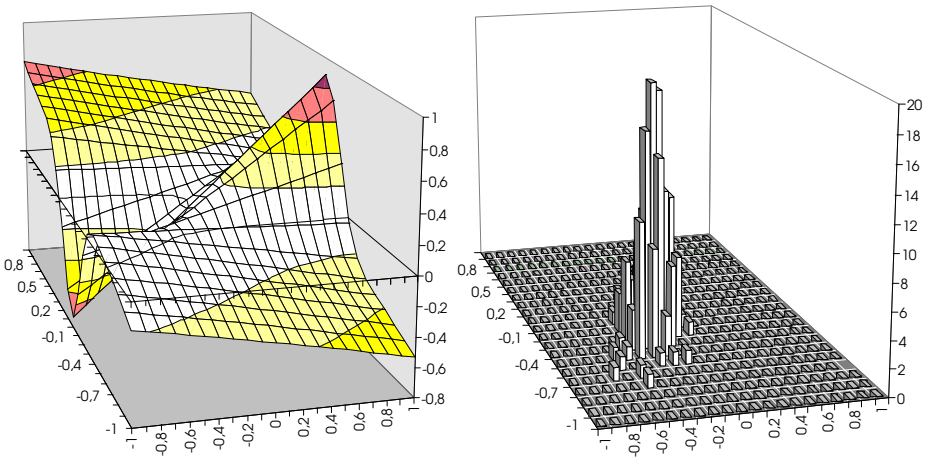

Figure 4: (a) true NAR function minus MLP approximation, (b) histogram of the training set.

We can then expect to obtain good linear models for this data set. Several ARMA models were fitted to the data. The residuals passed the randomness tests only for the AR(3) and ARMA(1,1) models. The estimated standard deviation of the residuals was

ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 111-116

0.109 and the RMS error on the training set 0.012. We see that the difference between the MLPs and the best linear models is not significant for this time-series.

4. Conclusion

The Bayesian framework developed for non-sequential problems was extended to MLP models of NAR time-series. This approach allows us to propose confidence limits for the predictions, by taking into account input noise and model variance. Note that the prior distribution is very important: usually, when the prior is stronger, the confidence limits are closer, but unfortunately the bias is higher.

We noticed that a full Bayesian treatment presents a very high computational cost. However, common simplifications can be successfully applied if enough data is available. Other approximation techniques are being evaluated, such as the use of a set of NN to obtain the output distribution. This seems to be necessary if we want to make the Bayesian approach tractable for more complex time-series.

Acknowledgements

The authors are grateful to Serge Déjerine for the interesting discussions and his permanent support.

References

Box, G. E. P., Jenkins, G. M., Reinsel, G. C. (1994) *Time Series Analysis: Forecasting and Control*, 3rd edition, Prentice-Hall Inc., Englewood Cliffs, NJ 07632, USA.

Hornik, M., Stinchcombe, M., White, H. (1989) Multilayer feedforward networks are universal approximators, *Neural Networks* 2: 359-366.

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

MacKay, D. J. C. (1995) Probable networks and plausible predictions — a review of practical Bayesian methods for supervised neural networks, *Technical Report*, Cavendish Laboratory, Cambridge, UK.

Neal, R. M. (1992) Bayesian training of backpropagation networks by the hybrid Monte-Carlo method, *Technical Report CRG-TR-92-1*, Department of Computer Science, University of Toronto, Canada.

Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986) Learning internal representations by error propagation, in Rumelhart, D. E., McClelland, J. L.(eds.), *Parallel distributed processing: explorations in the microstructure of cognition*, Vol.2: Psychological and biological models, Cambridge, MA: MIT Press, pp. 7-57.