Multiple-step ahead prediction for non linear dynamic systems - A Gaussian Process treatment with propagation of the uncertainty

Article *in* Advances in Neural Information Processing Systems · January 2003

CITATIONS
65

READS
1,809

3 authors, including:

Roderick Murray-Smith
University of Glasgow
303 PUBLICATIONS **10,603** CITATIONS

SEE PROFILE

submitted to *Advances in Neural Information Processing Systems 15*.

Multiple-step ahead prediction for non linear dynamic systems – A Gaussian Process treatment with propagation of the uncertainty

Agathe Girard
Department of Computing Science
University of Glasgow
Glasgow, G12 8QQ
agathe@dcs.gla.ac.uk

Carl Edward Rasmussen
Gatsby Unit, University College London
London, WC1N 3AR
edward@gatsby.ucl.ac.uk

Roderick Murray-Smith
Department of Computing Science
University of Glasgow
Glasgow, G12 8QQ
rod@dcs.gla.ac.uk

Abstract

We consider the problem of multi-step ahead prediction in time series analysis using the non-parametric Gaussian process model. k-step ahead forecasting of a discrete-time non-linear dynamic system can be performed by doing repeated one-step ahead predictions. For a state-space model of the form $y_t = f(y_{t-1}, \ldots, y_{t-L})$, the prediction of y at time $t+k$ is based on the estimates $\hat{y}_{t+k-1}, \ldots, \hat{y}_{t+k-L}$ of the previous outputs. In this paper, we show how, using an analytical Gaussian approximation, we can formally incorporate the uncertainty about future regressor values, thus updating the uncertainty on the current prediction.

1 Introduction

One of the main objectives in time series analysis is forecasting and in many real life problems, one has to predict ahead in time, up to a certain time horizon (sometimes called *lead* time or prediction horizon). Furthermore, knowledge of the uncertainty of the prediction is important. Currently, the multiple-step ahead prediction task is achieved by either explicitly training a *direct* model to predict k steps ahead, or by doing repeated one-step ahead predictions up to the desired horizon, which we call the *iterative method*.

There are a number of reasons why the iterative method might be preferred to the 'direct' one. Firstly, the direct method makes predictions for a fixed horizon only, making it computationally demanding if one is interested in different horizons. Furthermore, the larger k, the more training data we need in order to achieve a good predictive performance, because of the larger number of 'missing' data between t and $t+k$. On the other hand, the iterated method provides any k-step ahead forecast, up to the desired horizon, as well as the

joint probability distribution of the predicted points, and generally does not need as large a training set as the direct approach.

In the Gaussian process modelling approach, one computes predictive distributions whose means serve as output estimates. Gaussian processes (GPs) for regression have historically been first introduced by O'Hagan [1] but started being a popular non-parametric modelling approach after the publication of [5]. In [6], it is shown that GPs can achieve a predictive performance comparable to (if not better than) other modelling approaches like neural networks or local learning methods. We will show that for a k-step ahead prediction which ignores the accumulating prediction variance, the model is not conservative enough, with unrealistically small uncertainty attached to the forecast. An alternative solution is presented for iterative k-step ahead prediction, with propagation of the prediction uncertainty.

2 Predicting with Gaussian processes

We briefly recall some fundamentals of Gaussian processes. For a comprehensive introduction, please refer to [3], [7], or the more recent review [8].

2.1 The GP prior model

A formal definition of a Gaussian process is that of a random function $f(x)$, with mean $m(x)$ and covariance function $C(x^p, x^q)$, if its values $f(x^1), \ldots, f(x^n)$ can be seen as the components of a normally distributed random vector.

Here, we assume that the process is stationary, with constant mean (taken to be zero) and covariance function only depending on the distance between the inputs x. For each n, we have

$$f(x^1), \ldots, f(x^n) \sim \mathcal{N}(0, \Sigma) \tag{1}$$

with $\Sigma_{pq} = \mathrm{Cov}(f(x^p), f(x^q))$ giving the covariance between the points $f(x^p)$ and $f(x^q)$, which is a function of the inputs corresponding to the same cases p and q.

A common choice of covariance function is[1]

$$\Sigma_{pq} \;=\; C(x^p, x^q) \;=\; v_1 \exp\left[-\frac{1}{2}\sum_{d=1}^{D} w_d (x_d^p - x_d^q)^2\right] \;. \tag{2}$$

The role of the covariance function in the GP framework is similar to that of the kernels used in the Support Vector Machines community. This choice corresponds to a prior assumption that the underlying function f is *smooth* and *continuous*. It accounts for a high correlation between the outputs of cases with nearby inputs. The parameter v_1 gives the overall scale of correlations and the w parameters allow a different distance measure for each input dimension d. For a given problem, these parameters will be adjusted to the data at hand and, for irrelevant inputs, the corresponding w_d will tend to zero[2].

2.2 Gaussian Processes for regression

Given this prior and a set of data $\mathcal{D} = \{t^i, x^i\}_{i=1}^N$, we now wish to predict the function value $y^* = f(x^*)$ corresponding to the input x^*.

Assuming an additive uncorrelated Gaussian white noise with variance v_0 relates the targets to the function outputs, the distribution over the targets, $p(\mathbf{t}|\Theta)$, is $\mathcal{N}(0, K)$, where K is the $N \times N$ covariance matrix for the training data such that $K_{pq} = \Sigma_{pq} + v_0 \delta_{pq}$.

[1] More discussion about possible covariance functions can be found in [3].

[2] Automatic Relevance Determination idea developed by MacKay and Neal [5]

In a Maximum Likelihood framework, we adjust the vector of hyperparameters $\Theta = [w_1 \dots w_D \ v_1 \ v_0]^T$ so as to maximise the log-likelihood $\mathcal{L}(\Theta) = \log p(\mathbf{t}|\Theta)$, where $\mathbf{t}$ is the vector of targets.

Therefore, for a new x^*, the predictive distribution of the corresponding output is simply obtained by conditioning on the training data to obtain $p(y^*|x^*, \mathcal{D})$. The joint distribution of the variables being Gaussian, this conditional distribution is also Gaussian with mean and variance

$$\mu_y(x^*) = \mathbf{k}(x^*)^T \ K^{-1} \ \mathbf{t} \quad (3)$$

$$\sigma_y^2(x^*) = k(x^*) - \mathbf{k}(x^*)^T \ K^{-1} \ \mathbf{k}(x^*) \quad (4)$$

where $\mathbf{k}(x^*) = [C(x^*, x^1), \dots, C(x^*, x^N)]^T$ is the $N \times 1$ vector of covariances between the new point and the training targets and $k(x^*) = C(x^*, x^*)$.

The predicted mean $\mu_y(x^*)$ is in general used as an estimate of the output, $\hat{y}^*$, with uncertainty $\sigma_y^2(x^*)$.

3 Iterative k-step ahead prediction of time series

The iterative prediction method consists in using the previous and current data to predict k steps ahead: it predicts only one time step ahead, using the output of the current prediction as the input to the prediction of the next time step, until the prediction k steps ahead is made.

For a state-space model of the form $y_t = f(y_{t-1}, \dots, y_{t-L})$, where L is the (fixed) lag, the prediction of y at $t+k$ is based on $\hat{y}_{t+k-1}, \dots, \hat{y}_{t+k-L}$ where $\hat{y}$ is the estimate of y (given by equation (3) in the GP framework). In this manner, only the output estimates are used and the uncertainty induced by each successive prediction (given by equation (4) for each y) is not accounted for.

To formally incorporate the uncertainty information about the future regressor, we now need to consider $y_{t+k} = f(y_{t+k-1}, \dots, y_{t+k-L})$ where each y is normally distributed with mean (3) and variance (4).

3.1 Prediction at a random input: Gaussian approximation

Let x^* be the state vector composed of the delayed outputs, such that $x^* = [y_{t-1}, \dots, y_{t-L}]^T \sim \mathcal{N}(\mu_{x^*}, \Sigma_{x^*})$[3].

Now, the predictive distribution at x^* is given by

$$p(y^*|\mu_{x^*}, \Sigma_{x^*}) = \int p(y^*|x^*)p(x^*)dx^* \ , \quad (5)$$

omitting the conditioning on the training data, and where $p(y^*|x^*)$ is as specified by (3) and (4).

Given that this integral is analytically intractable ($p(y^*|x^*)$ is a complicated function of x^*), we opt for an analytical Gaussian approximation of $p(y^*|\mu_{x^*}, \Sigma_{x^*})$. In our illustrative example, we will perform a numerical approximation of the integral by a simple Monte-Carlo approach.

[3] At time t, μ_{x^*} is the $1 \times L$ vector of predicted means $[\hat{y}_{t-1}, \dots, \hat{y}_{t-L}]^T$ and Σ_{x^*} the $L \times L$ input covariance matrix with the predicted variances on its diagonal and the cross-covariance terms, given by $\mathrm{cov}(x^*, y) = \left|\frac{\partial \mu_y(x^*)}{\partial x^*}\right|_{\mu_{x^*}}^T \Sigma_{x^*}$.

Within the Gaussian approximation, we only need to compute the mean and variance of $y^*|\mu_{x^*}, \Sigma_{x^*}$ (that we will now denote by y for brevity), which are given by the law of iterated expectations and that of the conditional variance respectively:

$$E[y] = E_{x^*}[E_y[y|x^*]] = E_{x^*}[\mu_y(x^*)] \quad (6)$$

$$\mathrm{var}(y) = E_{x^*}[\mathrm{var}_y(y|x^*)] + \mathrm{var}_{x^*}(E_y[y|x^*]) = E_{x^*}[\sigma_y^2(x^*)] + \mathrm{var}_{x^*}(\mu_y(x^*)) \quad (7)$$

where E_{x^*} indicates the expectation under x^*. Now, $\mu_y(x^*)$ and $\sigma_y^2(x^*)$ are functions of the random argument x^* and we need further approximation to be able to compute these moments.

3.2 Predictive mean $E[y]$ and variance $\mathrm{var}(y)$

We approximate $\mu_y(x^*)$ by its first order Taylor expansion around μ_{x^*}:

$$\mu_y(x^*) \simeq \mu_y(\mu_{x^*}) + \left.\frac{\partial \mu_y(x^*)}{\partial x^*}\right|_{\mu_{x^*}}^T (x^* - \mu_{x^*}) + o(x^{*2}) \quad (8)$$

where $\mu_y(\mu_{x^*}) = \mathbf{k}(\mu_{x^*})^T K^{-1}\mathbf{t}$. Inserting $\mu_y(x^*)$ in equation (6), we get

$$E[y] \approx E_{x^*}\left[\mu_y(\mu_{x^*}) + \left.\frac{\partial \mu_y(x^*)}{\partial x^*}\right|_{\mu_{x^*}}^T (x^* - \mu_{x^*})\right] = \mu_y(\mu_{x^*}) \,. \quad (9)$$

Therefore, we see that within a first order Taylor expansion, predicting at random x^* does not provide any correction over the zeroth order. Of course, this approximation depends on the curvature of $\mu_y(x^*)$ and on the variance of x^*: in one dimension, it can easily be seen geometrically that for $\mu_y(x^*)$ with high curvature, if σ_{x^*} is 'small', then the first order Taylor approximation will be good, otherwise a second order approximation will be better suited.

To compute $\mathrm{var}(y)$, we first need to calculate $E_{x^*}[\sigma_y^2(x^*)]$. For this, we need to approximate $\sigma_y^2(x^*)$ and a natural choice is its second order Taylor expansion around μ_{x^*}:

$$\sigma_y^2(x^*) \simeq \sigma_y^2(\mu_{x^*}) + \left.\frac{\partial \sigma_y^2(x^*)}{\partial x^*}\right|_{\mu_{x^*}}^T (x^* - \mu_{x^*}) + \frac{1}{2}(x^* - \mu_{x^*})^T \left.\frac{\partial^2 \sigma_y^2(x^*)}{\partial x_d^* \partial x_{d'}^*}\right|_{\mu_{x^*}} (x^* - \mu_{x^*}) + o(x^{*3}) \quad (10)$$

with $\sigma_y^2(\mu_{x^*}) = k(\mu_{x^*}) - \mathbf{k}(\mu_{x^*})^T\, K^{-1}\, \mathbf{k}(\mu_{x^*})$. Using the formula giving the expectation of a quadratic form under a Gaussian, we arrive at

$$E_{x^*}[\sigma_y^2(x^*)] \approx \sigma_y^2(\mu_{x^*}) + \frac{1}{2}\mathrm{Tr}\left\{\left.\frac{\partial^2 \sigma_y^2(x^*)}{\partial x_d^* \partial x_{d'}^*}\right|_{\mu_{x^*}} \Sigma_{x^*}\right\} \,. \quad (11)$$

Then, with the term $\mathrm{var}_{x^*}(\mu_y(x^*))$ obtained by using (8), we have

$$\mathrm{var}(y) = \sigma_y^2(\mu_{x^*}) + \mathrm{Tr}\left\{\Sigma_{x^*}\left(\frac{1}{2}\left.\frac{\partial^2 \sigma_y^2(x^*)}{\partial x_d^* \partial x_{d'}^*}\right|_{\mu_{x^*}} + \left.\frac{\partial \mu_y(x^*)}{\partial x^*}\right|_{\mu_{x^*}} \left.\frac{\partial \mu_y(x^*)}{\partial x^*}\right|_{\mu_{x^*}}^T\right)\right\} \quad (12)$$

with

$$\left.\frac{\partial \mu_y(x^*)}{\partial x_d^*}\right|_{\mu_{x^*}} = \left.\frac{\partial \mathbf{k}(x^*)}{\partial x_d^*}\right|_{\mu_{x^*}}^T K^{-1}\,\mathbf{t} \quad (13)$$

$$\left.\frac{\partial^2 \sigma_y^2(x^*)}{\partial x_d^* \partial x_{d'}^*}\right|_{\mu_{x^*}} = \left.\frac{\partial^2 K(x^{*p}, x^{*q})}{\partial x_d^{*p} \partial x_{d'}^{*q}}\right|_{\mu_{x^*}} - \left.\frac{\partial K(x^*, \mathbf{x})}{\partial x_d^*}\right|_{\mu_{x^*}}^T K(\mathbf{x}, \mathbf{x})^{-1} \left.\frac{\partial K(\mathbf{x}, x^*)}{\partial x_{d'}^*}\right|_{\mu_{x^*}} \quad (14)$$

Compared to the predicted variance obtained in the 'non-random' input case, we now have a correction term of the zeroth order which involves the computation of the first and second derivatives of the covariance function (2). Close to the training points, the term $\partial \mathbf{k}(x^*)/\partial x_d^*$ will be close to zero so the contribution from $\partial \mu_y(x^*)/\partial x_d^*$ to the variance will be very small.

4 Illustrative examples

The first example is intended to provide a basis for comparing our Gaussian approximation to the Monte-Carlo sampling from the true distribution (numerical approximation of the integral (5)) when the uncertainty is propagated as we predict ahead in time. The second one, inspired from real-life problems, enables us to analyse the difference between the iterative k-step ahead prediction when propagating the uncertainty and when using only the output estimates.

The predictive performance of the different methods is assessed computing the average absolute error (AE), the average squared error (SE) and average minus log predictive density[4] ($mLPD$).

4.1 A $2D$ non-linear dynamic system

We present an example of a non-linear, second order, dynamic system Ω, composed of a blend of two affine models, $\Omega : \dot{\mathbf{x}} = f_1(\mathbf{x}) + f_2(\mathbf{x})$, where $f_i(\mathbf{x}) = \phi(\mathbf{x})(A_i\mathbf{x} + d_i)$, $\phi(\mathbf{x}) = \exp(- \parallel \mathbf{x} - \mathbf{c} \parallel /\sigma_i^2)$ and $\mathbf{c}_i = A_i^{-1}d_i$, with $A_1 = A_2 = \begin{bmatrix} 2 & 5 \\ -10 & -3 \end{bmatrix}$, $d_1 = \begin{bmatrix} 10 \\ -10 \end{bmatrix}$, $d_2 = -d_1$, $\sigma_1 = 1$, $\sigma_2 = 2$. This system has two stable equilibrium points and a stable limit cycle. Because the data was acquired by starting the simulation at a random initial position, and simulating for a fixed period, we find a number of interesting features which are typical of many real world examples when modelling from observed data.

The identification data consist of 6 simulations (200 points per simulation) with different starting points and the two test trajectories result from two different simulations.

Assuming a model of the type $x_{t+1} = f(x_t)$, where x is two-dimensional, we create the input and target matrices corresponding to each trajectory. We then model each dimension with a separate Gaussian process: $x_{t+1}^i = f^i(x_t)$, for $i = 1, 2$. Figure 1 shows the predictions when predicting from $k = 1$ to $k = 10$ steps ahead, when starting the simulation at six different points. Figure 2 shows the true trajectory along with the mean predictions with their 2σ uncertainties (crosses) when propagating the uncertainty with the Gaussian approximation and 30 samples from the true predictive distribution.

These figures clearly show that the Gaussian approximation is valid, with error bars encompassing the samples from the true distribution. This is confirmed quantitatively by the losses (see table 1).

4.2 Prediction of a pH process simulation

We apply the method on a pH neutralisation process benchmark ([2]). The training and test data consists of pH values (outputs y of the process) and a control input signal (u).

[4]To evaluate these losses in the case of Monte-Carlo sampling, we use the sample mean and sample variance.

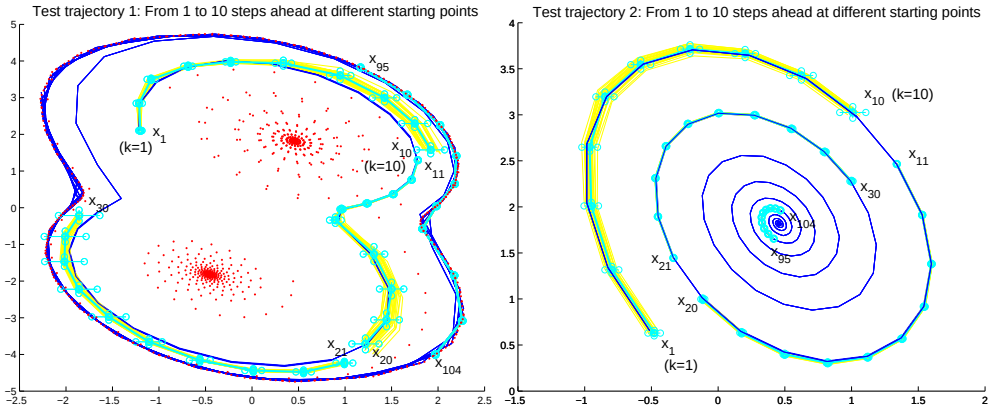

Figure 1: Iterative method in action: simulation from 1 to 10 steps ahead for different starting points of the test trajectories. Mean predictions with 2σ error bars (circles), along with 30 samples from the true distribution, when propagating the uncertainty, for the two test trajectories. Also shows (left) the training data (dots).

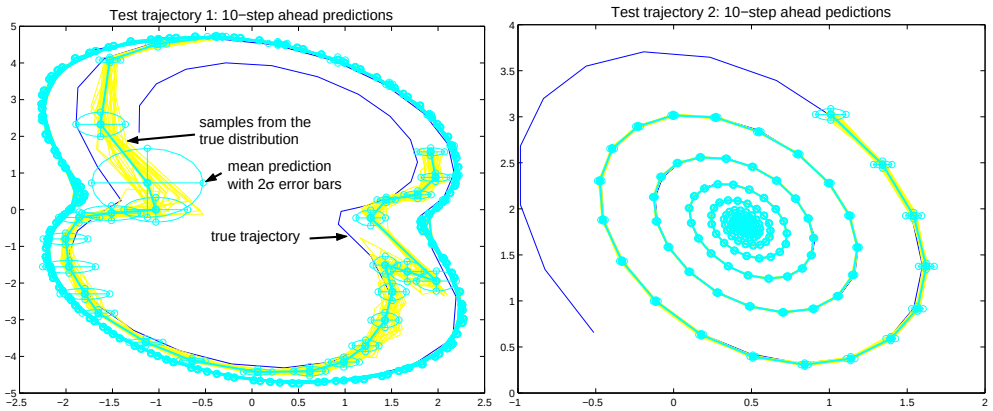

Figure 2: 10-step ahead prediction for the two test trajectories (the first point here corresponds to x_{10} in figure 1, etc). Same legend as figure 1.

Table 1: Losses for the Gaussian approximation (GP) and the sampling from the true distribution (MC) on the two test trajectories. The losses are given for each of the two dimensions.

		AE dim. 1	SE dim. 2	SE dim. 1	SE dim. 2	mLPD dim. 1	mLPD dim. 2
Traj. 1	GP	0.037	0.071	0.016	0.061	-0.040	6.471
(Fig. 2, left)	MC	0.037	0.072	0.016	0.061	3.059	3.474
Traj. 2	GP	0.003	0.003	$0.282.10^{-4}$	$0.337.10^{-4}$	-4.559	-3.656
(Fig. 2, right)	MC	0.003	0.003	$0.272.10^{-4}$	$0.358.10^{-4}$	-3.745	-3.763

With a model of the form $y_t = f(y_{t-8}, \ldots, y_{t-1}, u_{t-8}, \ldots, u_{t-1})$, we have a training input matrix of size 1228×16, with the corresponding vector of targets and a different set of 15952 test data (all data had their mean removed). After maximization of the likelihood, the ARD tool indicates that the lagged outputs contributing the most are y_{t-1},y_{t-2} and y_{t-3} whose hyperparameters are one order of magnitude larger than the those corresponding to other outputs. In the same manner, more *weight* is given to u_{t-5}.

Figure 3 shows the simulation from 1 to 10 steps ahead starting at two different points and figure 4 the plots the 10-step ahead predicted points, at the beginning and at the end of the validation set. On both figures, we plot the true data and the mean predictions with their 2σ uncertainties obtained when propagating the uncertainty (circles) or not (crosses). For the 10-step ahead predictions, we have the following losses. The average absolute error and squared error are 0.3255 and 0.4090 respectively, whether one propagates the uncertainty or not. On the other hand, the average minus log predictive density is much better when propagating the uncertainty (0.6527 against 2763.1!).

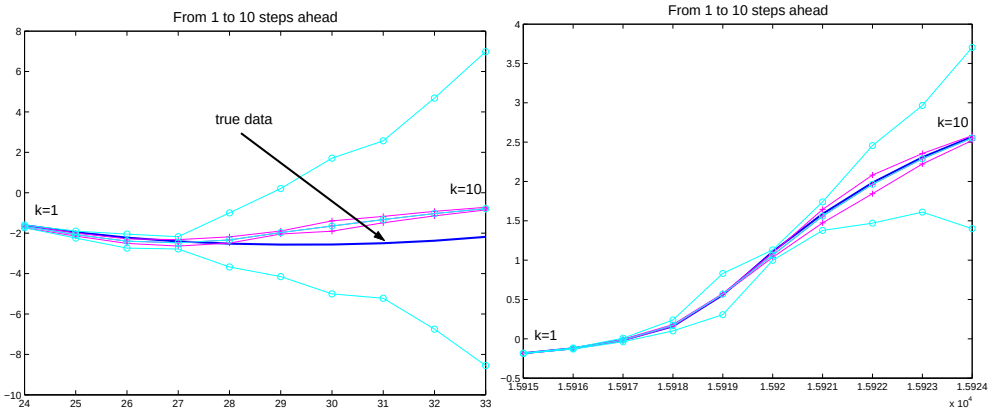

Figure 3: From 1 to 10 steps ahead: two windows of predicted means with their 2σ uncertainties with (circles) and without (crosses) propagation of the uncertainty at different times within the validation set.

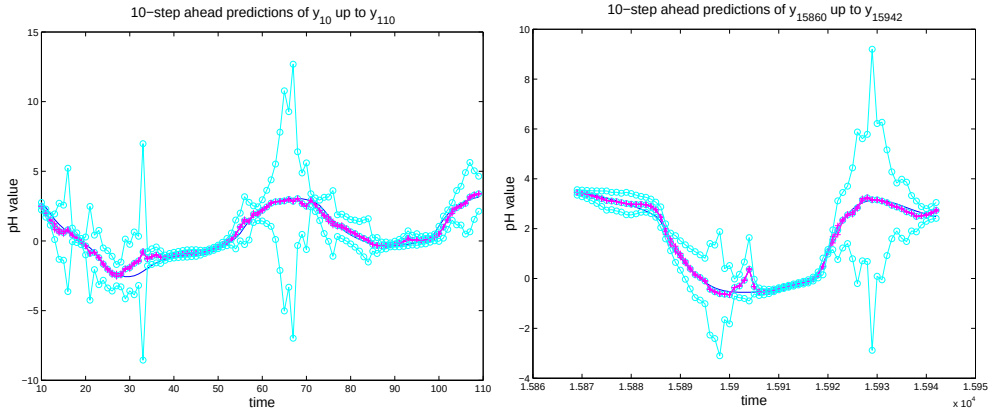

Figure 4: 10-step ahead mean predictions with their 2σ error bars, when propagating the uncertainty (circles) and when using the previous estimates only (crosses). These predictions are taken at the beginning and at the end of the test set.

As we predict ahead and propagate the uncertainty, the uncertainty does not necessarily increase monotically with k, but is also affected by changes in the control signal, see figure 3. On the other hand, we see that when the uncertainty is not propagated, the model is too confident with very small error bars. In this example, the important role of the control signal has to be noted, partly explaining the big changes in uncertainties observed on the 10-step ahead predictions plot (figure 4).

5 Conclusions

We have presented an approximation which allows us to use knowledge of the variance on inputs to Gaussian process models to achieve more realistic prediction variance. This is useful in its own right in the case of noisy model inputs.

Iterating this approach allows us to use it as a method for efficient propagation of uncertainty in multi-step ahead time-series predictions. In experiments on simulated dynamic systems, comparing our Gaussian approximation to Monte Carlo simulations, we found that the propagation method is comparable to Monte Carlo simulations, and that both approaches achieved more realistic error bars than a naive approach which ignores the uncertainty on current state.

This method can help understanding the underlying dynamics of a system, as well as being useful, for instance, in a model predictive control framework where knowledge of the accuracy of the model predictions over the whole prediction horizon is required (see [4] for a model predictive control law based on Gaussian processes taking account of the prediction uncertainty).

Acknowledgements

The authors gratefully acknowledge the support of the *Multi-Agent Control* Research Training Network - EC TMR grant HPRN-CT-1999-00107.

References

[1] O'Hagan, A. (1978) Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society B* **40**:1-42.

[2] Henson, M. A. & Seborg, D. E. (1994) Adaptive nonlinear control of a pH neutralisation process. *IEEE Trans Control System Technology* **2**:169-183.

[3] MacKay, D. J. C. (1997) Gaussian Processes - A Replacement for Supervised Neural Networks?. Lecture notes for a tutorial at NIPS 1997.

[4] Murray-Smith, R. & Sbarbaro-Hofer, D. (2002) Nonlinear adaptive control using non-parametric Gaussian process prior models. *15th IFAC World Congress on Automatic Control, Barcelona*

[5] Neal, R. M. (1995) *Bayesian Learning for Neural Networks* PhD thesis, Dept. of Computer Science, University of Toronto.

[6] Rasmussen, C. E. (1996) *Evaluation of Gaussian Processes and other Methods for Non-Linear Regression* PhD thesis, Dept. of Computer Science, University of Toronto.

[7] Williams, C. K. I. & Rasmussen, C. E. (1996) Gaussian Processes for Regression *Advances in Neural Information Processing Systems 8* MIT Press.

[8] Williams, C. K. I. (2002) Gaussian Processes *To appear in The handbook of Brain Theory and Neural Networks, Second edition* MIT Press.