



Gaussian process for nonstationary time series prediction

Sofiane Brahim-Belhouari*, Amine Bermak

EEE Department, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong

Received 27 June 2003; received in revised form 12 February 2004; accepted 13 February 2004

Abstract

In this paper, the problem of time series prediction is studied. A Bayesian procedure based on Gaussian process models using a nonstationary covariance function is proposed. Experiments proved the approach effectiveness with an excellent prediction and a good tracking. The conceptual simplicity, and good performance of Gaussian process models should make them very attractive for a wide range of problems.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Bayesian learning; Gaussian processes; Prediction theory; Time series

1. Introduction

A new method for regression was inspired by Neal's work on Bayesian learning for neural networks (Neal, 1996). It is an attractive method for modelling noisy data, based on priors over function using Gaussian Processes (GP). It was shown that many Bayesian regression models based on neural networks converge to Gaussian processes in the limit of an infinite network (Neal, 1996). Gaussian process models have been applied to the modelling of noise free (Neal, 1997) and noisy data (Williams and Rasmussen, 1996) as well as to classification problems (MacKay, 1997). In our previous work, Gaussian process for forecasting problem was proposed and compared with radial basis function neural network (Brahim-Belhouari and Vesin, 2001). Bayesian learning has shown an excellent prediction capability for stationary problems. However, most real life time series are often nonstationary and hence improved models

* Corresponding author. Tel.: +852-2358-8844; fax: +852-2358-1485.

E-mail address: eebelhou@ust.hk (S. Brahim-Belhouari).

dealing with nonstationarity are required. In this paper, we present a prediction approach based on Gaussian process, which has been successfully applied to nonstationary time series. In addition, the use of the Gaussian process with different covariance functions for real temporal patterns is investigated. The proposed approach is shown to outperform Radial Basis Function (RBF) neural network. The advantage of the Gaussian process formulation is due to the fact that the combination of the prior and noise models is carried out exactly using matrix operations. We also present a maximum likelihood approach used to estimate the hyperparameters of the covariance function. These hyperparameters allow the control of the form of the Gaussian process.

The paper is organized as follows: Section 2 presents a general formulation of the forecasting problem. Section 3 discusses the application of GP models, with nonstationary covariance function, to prediction problems. Section 4 shows the feasibility of the approach for real nonstationary time series. Section 5 presents some concluding remarks.

2. Prediction problem

Time series are encountered in science as well as in real life. Most common time series are the result of unknown or incompletely understood systems. A time series $x(k)$ is defined as a function x of an independent variable k , generated from an unknown system. Its main characteristic is that its evolution cannot be described exactly. The observation of past values of a phenomenon in order to anticipate its future behavior represents the essence of forecasting. A typical approach is to predict by constructing a prediction model which takes into account previous outcomes of the phenomenon. We can take a set of d such values x_{k-d+1}, \dots, x_k to be the model input and use the next value x_{k+1} as the target.

In a parametric approach to forecasting we express the predictor in terms of a nonlinear function $y(\mathbf{x}, \boldsymbol{\beta})$ parameterized by parameters $\boldsymbol{\beta}$. It implements a nonlinear mapping from input vector $\mathbf{x} = [x_{k-d+1}, \dots, x_k]^T$ to the real value:

$$t^i = y(\mathbf{x}^i, \boldsymbol{\beta}) + \varepsilon^i, \quad i = 1, \dots, n, \quad (1)$$

where ε is a noise corrupting the data points.

Time series processing is an important application area of neural networks. In fact, y can be given by a specified network. The output of the radial basis function (RBF) network is computed as a linear superposition (Bishop, 1995):

$$y(\mathbf{x}, \boldsymbol{\beta}) = \sum_{j=1}^M w_j g_j(\mathbf{x}), \quad (2)$$

where w_j denotes the weights of the output layer. The Gaussian basis functions g_j are defined as

$$g_j(\mathbf{x}) = \exp \left\{ -\frac{\|\mathbf{x} - \boldsymbol{\mu}_j\|^2}{2\sigma_j^2} \right\}, \quad (3)$$

where μ_j and σ_j^2 denote means and variances. Thus we define the parameters as $\beta = [w_j, \mu_j, \sigma_j]^T$ ($j = 1, \dots, M$), which can be estimated using a special purpose training algorithm.

In nonparametric methods, predictions are obtained without representing the unknown system as an explicit parameterized function. A new method for regression was inspired by Neal’s work (1996) on Bayesian learning for neural networks. It is an attractive method for modelling noisy data, based on priors over function using Gaussian Processes.

3. Gaussian process models

The Bayesian analysis of forecasting models is difficult because a simple prior over parameters implies a complex prior distribution over functions. Rather than expressing our prior knowledge in terms of a prior for the parameters, we can instead integrate over the parameters to obtain a prior distribution for the model outputs in any set of cases. The prediction operation is most easily carried out if all the distributions are Gaussian. Fortunately, Gaussian process are flexible enough to represent a wide variety of interesting model structure, many of which would have a large number of parameters if formulated in more classical fashion. A Gaussian process is a collection of random variables, any finite set of which have a joint Gaussian distribution (MacKay, 1997). For a finite collection of inputs, $\mathbf{x} = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}]^T$, we consider a set of random variables $\mathbf{y} = [y^{(1)}, \dots, y^{(n)}]^T$, to represent the corresponding function values. A Gaussian process is used to define the joint distribution between the y ’s:

$$\wp(\mathbf{y}|\mathbf{x}) \sim \exp(-\frac{1}{2} \mathbf{y}^T \Sigma^{-1} \mathbf{y}), \tag{4}$$

where the matrix Σ is given by the covariance function C :

$$\Sigma_{pq} = \text{cov}(y^{(p)}, y^{(q)}) = C(\mathbf{x}^{(p)}, \mathbf{x}^{(q)}), \tag{5}$$

cov stands for covariance operator.

3.1. Predicting with Gaussian process

The goal of Bayesian forecasting is to compute the distribution $\wp(y^{(n+1)}|D, \mathbf{x}^{(n+1)})$ of output $y^{(n+1)}$ given a test input $\mathbf{x}^{(n+1)}$ and a set of n training points $D = \{\mathbf{x}^{(i)}, t^{(i)} | i = 1, \dots, n\}$. Using Baye’s rule, we obtain the posterior distribution for the $(n+1)$ Gaussian process outputs. By conditioning on the observed targets in the training set, the predictive distribution is Gaussian (Williams and Rasmussen, 1996):

$$\wp(y^{(n+1)}|D, \mathbf{x}^{(n+1)}) \sim N(\mu_{y^{(n+1)}}, \sigma_{y^{(n+1)}}^2), \tag{6}$$

where the mean and variance are given by

$$\begin{aligned} \mu_{y^{(n+1)}} &= \mathbf{a}^T \mathbf{Q}^{-1} \mathbf{t}, \\ \sigma_{y^{(n+1)}}^2 &= C(\mathbf{x}^{(n+1)}, \mathbf{x}^{(n+1)}) - \mathbf{a}^T \mathbf{Q}^{-1} \mathbf{a}, \end{aligned}$$

$$Q_{pq} = C(\mathbf{x}^{(p)}, \mathbf{x}^{(q)}) + r^2 \delta_{pq},$$

$$a_p = C(\mathbf{x}^{(n+1)}, \mathbf{x}^{(p)}), \quad p = 1, \dots, n,$$

r^2 is the unknown variance of the Gaussian noise. In contrast to classical methods, we obtain not only a point prediction but a predictive distribution. This advantage can be used to obtain the prediction intervals that describe a degree of belief of the predictions.

3.2. Training a Gaussian process

There are many possible choices of prior covariance functions. From a modelling point of view, the objective is to specify prior covariances which contain our prior beliefs about the structure of the function we are modelling. Formally, we are required to specify a function which will generate a nonnegative definite covariance matrix for any set of input points. Gaussian process procedure can handle interesting models by simply using a covariance function with an exponential term (Williams and Rasmussen, 1996):

$$C_{\text{exp}}(\mathbf{x}^{(p)}, \mathbf{x}^{(q)}) = w_0 \exp \left\{ -\frac{1}{2} \sum_{l=1}^d w_l (x_l^{(p)} - x_l^{(q)})^2 \right\}. \quad (7)$$

This function expresses the idea that cases with nearby inputs will have highly correlated outputs. The simplest nonstationary covariance function so that $C(\mathbf{x}^{(p)}, \mathbf{x}^{(q)}) \neq C(|\mathbf{x}^{(p)} - \mathbf{x}^{(q)}|)$, is the one corresponding to a linear trend:

$$C_{\text{ns}}(\mathbf{x}^{(p)}, \mathbf{x}^{(q)}) = v_0 + v_1 \sum_{l=1}^d x_l^{(p)} x_l^{(q)}. \quad (8)$$

Addition and multiplication of simple covariance functions are useful in constructing a variety of covariance functions. It was found that the sum of both covariance function (Eqs. (7) (8)) works well. Let us assume that a form of covariance function has been chosen, but that it depends on undetermined hyperparameters $\boldsymbol{\theta} = (v_0, v_1, w_0, w_1, \dots, w_d, r^2)$. We would like to learn these hyperparameters from the training data. In a maximum likelihood framework, we adjust the hyperparameters so as to maximize the log likelihood of the hyperparameters:

$$\begin{aligned} \log \wp(D|\boldsymbol{\theta}) &= \log \wp(t^{(1)}, \dots, t^{(n)} | \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}, \boldsymbol{\theta}) \\ &= -\frac{1}{2} \log \det Q - \frac{1}{2} \mathbf{t}^\top Q^{-1} \mathbf{t} - \frac{n}{2} \log 2\pi. \end{aligned} \quad (9)$$

It is possible to express analytically the partial derivatives of the log likelihood, which can form the basis of an efficient learning scheme. These derivatives are (MacKay, 1997)

$$\frac{\partial}{\partial \theta_i} \log \wp(D|\boldsymbol{\theta}) = -\frac{1}{2} \text{tr} \left(Q^{-1} \frac{\partial Q}{\partial \theta_i} \right) + \frac{1}{2} \mathbf{t}^\top Q^{-1} \frac{\partial Q}{\partial \theta_i} Q^{-1} \mathbf{t}. \quad (10)$$

We initialize the hyperparameters to random values (in a reasonable range) and then use an iterative method, for example conjugate gradient, to search for optimal values of the hyperparameters. We found that this approach is sometimes susceptible to

local minima. To overcome this drawback, we randomly selected a number of starting positions within the hyperparameters space.

4. Experimental results

In order to test the performance of the GP models using different covariance functions and compare it with RBF network, we used a respiration signal as real nonstationary time series. The data set contains 725 samples representing a short recording period of the respiratory rhythm via a thoracic belt. We obtained 400 patterns for training and 325 for testing candidate models. Predictive patterns were generated by windowing 6 inputs and one output. We conducted experiments for GP models using exponential covariance function (Eq. (7)) and a nonstationary covariance function given by the sum of (Eqs. (7) and (8)). The RBF network uses 30 centers chosen according to the validation set. The hyperparameters were adapted to the training data using conjugate gradient search algorithm. Results of prediction for both GP structures are given in Figs. 1 and 2. Time window is divided into a training part (time < 400) and test part (time > 400).

As it can be seen from Fig. 1, predictive patterns using a GP model with an exponential covariance function match perfectly with the real signal during the training

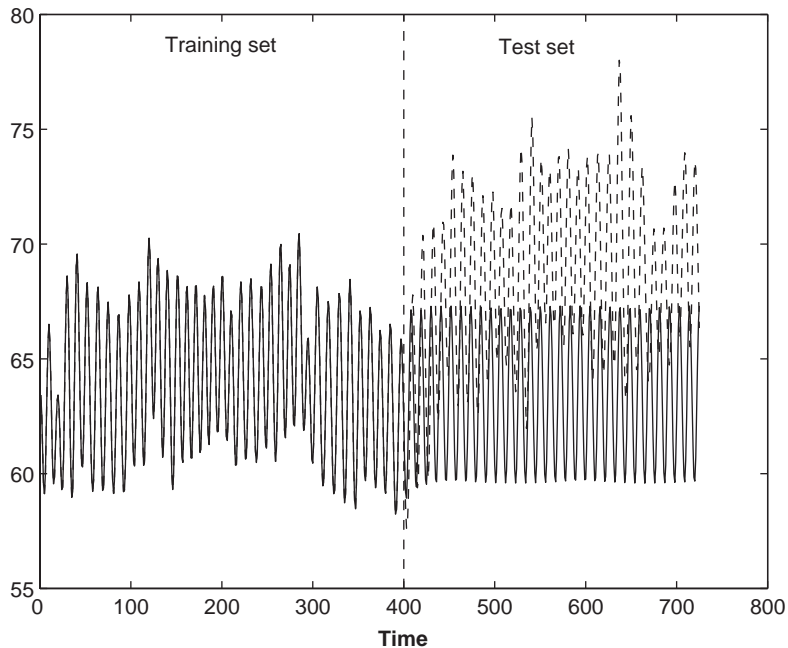


Fig. 1. Respiration signal (dashed line) and the mean of the predictive distribution using a GP model with an exponential covariance function (solid line).

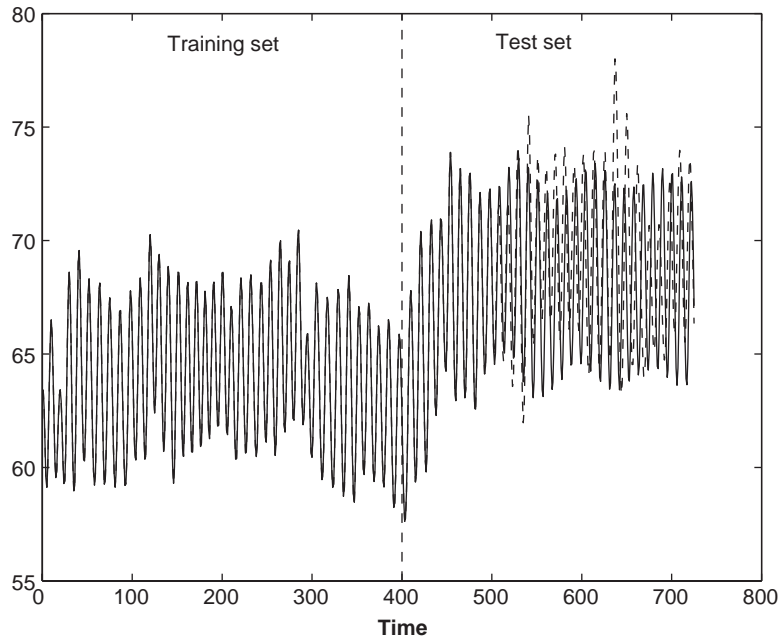


Fig. 2. Respiration signal (dashed line) and the mean of the predictive distribution using a GP model with a nonstationary covariance function (solid line).

Table 1
Prediction Performance expressed in terms of mean square test error

Methods	Mean square test error
GP_{ns}	3.83
GP_{exp}	6.37
RBF	7.18

period, but deviate significantly during the test period. In Fig. 2, the GP model using a nonstationary covariance function shows a good tracking performance.

The prediction performance is quantified by the mean square test error and is shown in Table 1. We can see from this table that the GP model using a nonstationary covariance function performs better than the one using exponential covariance function and RBF network.

The prediction performance would be greatly enhanced if the maximum of signal local trends are within the training window. Consequently, a large training window is required to estimate reliable statistics about the underlying process. However, requiring a vast number of data patterns n is time consuming as the inversion of an $n \times n$ matrix (Q) is needed. In fact, finding the predictive mean (Eq. (6)) and evaluating

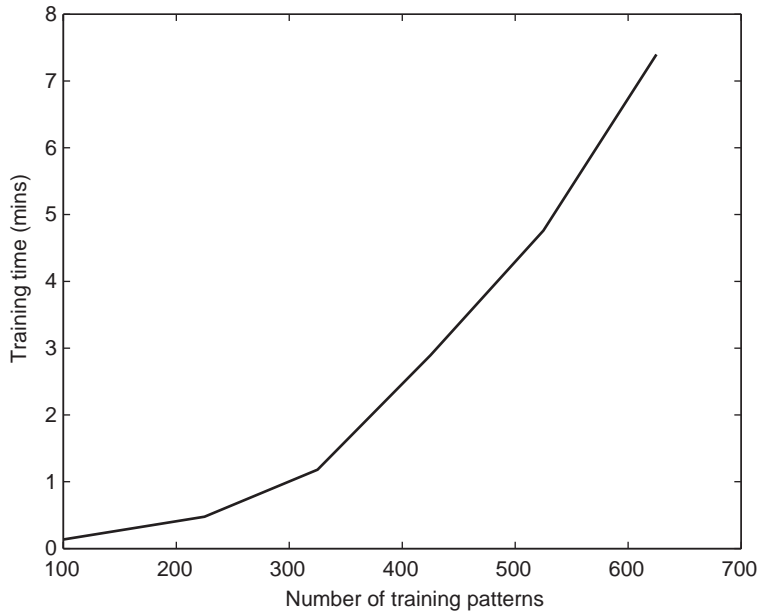


Fig. 3. Training time of the GP model as function of the number of training patterns.

the gradient of the log likelihood (Eq. (10)) require the evaluation of Q^{-1} . Fig. 3 shows the training times (on a SUN ULTRA SPARC 10) for different widths of the training window. The training time increases dramatically with the number of training patterns (for $n=400$, the time required to perform the prediction was about 156 s). As a consequence, new approximation methods are needed to deal with this computational cost when the number of data patterns becomes large.

5. Concluding remarks

In this paper we proposed a forecasting method based on Gaussian process models. We have shown that reasonable prediction and tracking performance can be achieved in the case of nonstationary time series. In addition, Gaussian process models are simple, practical and powerful Bayesian tools for data analysis.

One drawback with our prediction method based on Gaussian processes is the computational cost associated with the inversion of an $n \times n$ matrix. The cost of direct methods of inversion may become prohibitive when the number of the data points n is greater than 1000 (MacKay, 1997). Further research is needed to deal with this issue.

It is of interest to investigate the possibility of improving further the performance by using a more complicated parameterized covariance function. A multi-model forecasting approach with GP predictors can be proposed as an on-line learning technique.

Acknowledgements

The work described in this paper was supported in part by a Direct Allocation Grant (project No. DAG02/03.EG05) and a PDF Grant from HKUST.

References

- Bishop, C.M., 1995. *Neural Networks for Pattern Recognition*. Clarendon press, Oxford.
- Brahim-Belhouari, S., Vesin, J-M., 2001. Bayesian learning using Gaussian process for time series prediction. 11th IEEE Statistical Signal Processing Workshop, pp. 433–436.
- MacKay, D.J.C., 1997. Gaussian Processes—A Replacement for Supervised Neural Networks. Lecture Notes for a Tutorial at NIPS'97 UK, <http://www.inference.phy.cam.ac.uk/mackay/BayesGP.html>.
- Neal, R.M., 1996. *Bayesian Learning for Neural Networks*. Springer, New York.
- Neal, R.M., 1997. Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. Technical Report, No. 9702, University of Toronto.
- Williams, C.K.I., Rasmussen, C.E., 1996. Gaussian process for regression. Touretzky, D.S., Mozer, M.C., Hasselmo, M.E., (Eds.) In: *Advances in Information Processing Systems*, MIT Press, Cambridge, MA, 8th Edition, pp. 111–116. <http://www.cs.toronto.edu/~care/gp.html>.