The Concept for Gaussian Process Model Based System Identification Toolbox

Juš Kocijan, Kristjan Ažman, Alexandra Grancharova

Abstract: *The Gaussian process model is an example of a flexible, probabilistic, nonparametric model with uncertainty predictions. It can be used for the modelling of complex nonlinear systems and recently it has also been used for dynamic systems identification. A need for the supporting software, in particular for dynamic system identification, has been recognised. Consequently, a Matlab toolbox concept for Gaussian Process based System Identification was generated. The use of the supporting software is illustrated with a nonlinear dynamic system identification example.*

Key words: *Matlab toolbox, Dynamic system models, System identification, Gaussian process models.*

INTRODUCTION

While there are numerous methods for the identification of linear dynamic systems from measured data, the nonlinear systems identification requires more sophisticated approaches. The most common choices include artificial neural networks, fuzzy models *etc*. Gaussian process (GP) models present an emerging, complementary method for a nonlinear system identification.

The GP model is a probabilistic, non-parametric black-box model. It differs from most of the other black-box identification approaches as it does not try to approximate the modelled system by fitting the parameters of the selected basis functions but rather searches for the relationship among measured data. GP models are closely related to approaches such as Support Vector Machines and specially Relevance Vector Machines [1].

The output of the GP model is a normal distribution, expressed in terms of mean and variance. The mean value represents the most likely output and the variance can be interpreted as the measure of its confidence. The obtained variance, which depends on the amount and quality of the available identification data, is important information, distinguishing the GP model from other methods. The GP model structure determination is facilitated as only the covariance function and the regressors of the model need to be selected. Another potentially useful attribute of the GP model is the possibility to include various kinds of prior knowledge into the model, see *e.g*. [2] for the incorporation of local models and the static characteristic. Also the number of model parameters, which need to be optimised is smaller than in other black-box identification approaches. The disadvantage of the method is the potential computational burden for optimization, which increases with the amount of data and the number of regressors.

The GP model was first used for solving a regression problem in the late seventies, but it gained popularity within the machine learning community in the late nineties of the twentieth century. Results of a possible implementation of the GP model for the identification of dynamic systems were presented only recently, *e.g*. [3,4]. The investigation of the model with uncertain inputs, which enables the propagation of uncertainty through the model, is given in [5] and illustrated in [6].

If the method is to be interesting for the practical use in the nonlinear black-box identification, a supporting software, available for the users in the community, is a necessity. The purpose of this paper is to provide a concept for a Gaussian process based system identification toolbox for a Matlab programme package. A brief overview of the GP models and their application to dynamic system is given in the next two sections. It is followed by the concept of the toolbox and an illustrative identification example, showing its utility.

GAUSSIAN PROCESS MODEL

A detailed presentation of Gaussian processes can be found *e.g.* in [7]. A Gaussian process is a random process, fully characterized by its mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. For simplicity, a zero-mean process is assumed. Given the inputs $\{\mathbf{x}_1,...,\mathbf{x}_n\}$, the corresponding outputs $f(\mathbf{x}_1),..., f(\mathbf{x}_n)$ can be viewed as a collection of random variables with joint multivariate Gaussian distribution: $f(\mathbf{x}_1),...,f(\mathbf{x}_n) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, where Σ_{pq} gives the covariance between $f(\mathbf{x}_p)$ and $f(\mathbf{x}_q)$ and is a function of the corresponding $\mathbf{x}_p$ and $\mathbf{x}_q$: $\Sigma_{pq}=C(\mathbf{x}_p, \mathbf{x}_q)$. The covariance function $C(.,.)$ can be of any kind, provided that it generates a positive definite covariance matrix $\boldsymbol{\Sigma}$. The Gaussian Process model fits naturally in the Bayesian modelling framework, where it places a prior directly over the space of functions instead of parameterizing $f(\mathbf{x})$. A common choice of covariance function is the squared exponential, *i.e.* Gaussian function:

$$\mathrm{Cov}\left[f(\mathbf{x}_p), f(\mathbf{x}_q)\right] = C\left(\mathbf{x}_p, \mathbf{x}_q\right) = v\ \exp\left[-\frac{1}{2}\sum_{d=1}^{D} w_d \left(\mathbf{x}_p^d - \mathbf{x}_q^d\right)^2\right] + v_0 \delta_{pq} \tag{1}$$

where x_p^d denotes d^{th} component of the D-dimensional input vector $\mathbf{x}_p$, and v, $w_1,...,w_D$ are free parameters, called hyperparameters. The smoothness assumption holds for the covariance function (1), as the points lying closer together in the input space are more correlated than the points lying more far apart. The parameter v controls the vertical scale of the variation and the w_d's are inversely proportional to the horizontal length-scale in dimension d ($\lambda_d=1/\sqrt{w_d}$). Part $v_0\delta_{pq}$ reflects the correlation because of the, presumably white, noise with variance v_0.

Let the input/target relationship be $y=f(\mathbf{x})+\boldsymbol{\varepsilon}$. We assume an additive white noise with variance v_0, $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, v_0)$, and put a GP prior with covariance function (1) and unknown parameters on $f(.)$. Within this probabilistic framework, we can write $y_1,..., y_N, y^* \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_{N+1})$, with $K_{pq}=\boldsymbol{\Sigma}_{pq}+v_0\boldsymbol{\delta}_{pq}$, where $\boldsymbol{\delta}_{pq}=1$ if $p=q$ and 0 otherwise. If we split $y_1,..., y_N, y^*$ into two parts, $\mathbf{y}=[y_1,..., y_N]$ and y^*, we can write

$$\mathbf{y}, y^* \sim \mathcal{N}(0, \mathbf{K}_{N+1}), \tag{2}$$

with

$$\mathbf{K}_{N+1} = \begin{bmatrix} \begin{bmatrix} & & \\ & \mathbf{K} & \\ & & \end{bmatrix} & \begin{bmatrix} \\ \mathbf{k}(\mathbf{x}^*) \\ \\ \end{bmatrix} \\ \begin{bmatrix} & \mathbf{k}(\mathbf{x}^*)^{\mathrm{T}} & \end{bmatrix} & \begin{bmatrix} \kappa(\mathbf{x}^*) \end{bmatrix} \end{bmatrix} \tag{3}$$

where $\mathbf{K}$ is an $N \times N$ matrix giving the covariances between y_p and y_q, for $p,q=1 ... N$, $\mathbf{k}(\mathbf{x}^*)$ is an $N \times 1$ vector giving the covariances between y^* and y_p, $k_p(\mathbf{x}^*)=C(\mathbf{x}^*, \mathbf{x}_p)$, for $p=1 ... N$, and $\boldsymbol{\kappa}(\mathbf{x}^*)=C(\mathbf{x}^*, \mathbf{x}^*)$ is the covariance between the test output and itself.

For modelling purposes, this joint probability can be divided into the marginal and the conditional part. Given a set of N training data pairs, $\{\mathbf{x}_p, y_p\}_{p=1}^{N}$, the marginal term gives the likelihood of the observed data: $\mathbf{y}|\ \mathbf{X} \sim \mathcal{N}(0, \mathbf{K})$, where $\mathbf{y}$ is the $N \times 1$ vector of training targets and $\mathbf{X}$ is the $N \times D$ matrix of corresponding training inputs. The unknown parameters of the covariance function, as well as the noise variance v_0, can be estimated via maximization of the log-likelihood. The conditional part of (2) provides us with the predictive distribution of y^* corresponding to a new given input $\mathbf{x}^*$. We only need to condition the joint distribution on the training data and the new input $\mathbf{x}^*$, $p(y^*|\mathbf{y},\mathbf{X},\mathbf{x}^*) = p(\mathbf{y},y^*)/p(\mathbf{y}|\mathbf{X})$. It can be shown that this distribution is Gaussian with mean and variance

$$\mu(\mathbf{x}^*) = \mathbf{k}(\mathbf{x}^*)^T \mathbf{K}^{-1}\mathbf{t} \tag{4}$$

$$\sigma^2(\mathbf{x}^*) = \kappa(\mathbf{x}^*) - \mathbf{k}(\mathbf{x}^*)^T \mathbf{K}^{-1}\mathbf{k}(\mathbf{x}^*) \tag{5}$$

This way, the predictive mean $\mu(\mathbf{x}^*)$ can be used as an estimate for y^* and the predictive variance $\sigma^2(\mathbf{x}^*)$, or standard deviation $\sigma(\mathbf{x}^*)$, as the uncertainty attached to it.

DYNAMIC SYSTEMS MODELLING

The presented GP model was originally used for the modelling of static nonlinearities, but it can be extended to model the dynamic systems as well [3,5]. Our task is to model the dynamic system (2) with output y, where input

$$\mathrm{x}(k)=[y(k-1),y(k-2),\ldots,y(k-L),u(k-1),u(k-2),\ldots,u(k-L)]^T \tag{8}$$

is the vector of regressors that determines nonlinear ARX model structure, and be able to make multi-step ahead model prediction.

One way to do multi-step ahead prediction is to make iterative one-step ahead predictions up to desired step whilst feeding back the predicted output. Two general approaches to iterated one-step ahead prediction are possible using the GP model. In the first only the mean values of the predicted output are fed back to the input. In this, so called "naive" approach, the input vector $\mathbf{x}$ into the GP model at time step k is:

$$\mathrm{x}(k)=[\hat{y}(k-1),\hat{y}(k-2),\ldots,\hat{y}(k-L),u(k-1),u(k-2),\ldots,u(k-L)]^T \tag{9}$$

In the second, so called "exact", approach the complete output distributions are fed back. More on the GP model simulation and differences of approaches can be found *e.g.* in [5,3].

There are two additional forms of the GP model which need to be mentioned. The first is a GP model with incorporated local linear dynamic models named as the Local Model Gaussian Process model (LMGP model). The LMGP model [8,2] is a hybrid between the GP model and the Local Model Network, achieved with specially adopted Gaussian covariance function. The second is a parametric, linear parameter varying model, named Fixed Structure Gaussian Process model (FSGP model) [9], with its varying parameters represented with the GP models.Both models are included in the concept of the toolbox.

THE CONCEPT FOR THE TOOLBOX

The toolbox is going to be developed as a Matlab toolbox package, as Matlab has become one of the most used software tools in the field of dynamic systems identification and because of the available mathematical and software tools available within.

The toolbox should consist of several function groups, assisting the designer in all steps of the identification procedure: GP model structure selection, model training, model simulation and validation of the obtained model. Also the groups of functions, supporting special forms of the GP model, *e.g*. LMGP and FSGP model, would be added.

More detailed concept can be given with the list of tasks of the contained functions:

1. Functions for training and general use of the GP model:
   - Training of the GP model
   - Minimisation routine for a continuous differentiable multivariate function based on conjugate gradient method with line searches
   - GP model one-step-ahead prediction calculation
2. Covariance functions:

- Gaussian covariance function
- Linear covariance function
- Constant covariance function
- Periodical covariance function
- Covariance function for the incorporation of the white noise
- Covariance function for the incorporation of the coloured noise
- Summation of the covariance functions
- Multiplication of the covariance functions

3. Functions for the training of the LMGP model:
   - Training of the LMGP model
   - LMGP model one-step-ahead prediction calculation
4. Functions for the simulation of the trained GP model:
   - Simulation of a GP model without distribution propagation
   - Simulation of a GP model with analytical approximation of distribution propagation (for Gaussian covariance function)
   - Simulation of a GP model with numerical approximation of distribution propagation
5. Functions for the simulation of the LMGP model:
   - Simulation of a LMGP model without distribution propagation
   - Simulation of a LMGP model with analytical approximation of distribution propagation
   - Simulation of a LMGP model with numerical approximation of distribution propagation
6. Functions for training and application of FSGP model:
   - Training of the FSGP model
   - FSGP model one-step-ahead prediction calculation, function for the use with Matlab/Simulink
   - Matlab/Simulink FSGP model template for dynamic system simulation
7. Miscellaneous support functions:
   - Data normalisation
   - Data denormalisation
   - Input matrix composition
   - Plot of simulation responses with confidence band
   - Plot of identification residuals with confidence band
   - Performance measures calculation
   - Statistical validation measures

Let us stress that this is not a complete list of functions to be included in the toolbox. The toolbox development is an iterative procedure and more functions are expected to be added by the time the toolbox becomes functional. To illustrate the utility of the listed functions, an example with a typical identification procedure is given in the next section.

EXAMPLE

The following example illustrates the use of described toolbox functions. Consider the system described by the following nonlinear state space model:

$$\dot{y} = -\tanh(y + u^3), \tag{10}$$

The output of the model is state y, disturbed with Gaussian white noise with variance 0.0025. The sampling time, determined according to the system dynamics, was selected as $T_s = 0.5$. The Euler approximation of the system (10) is:

$$y(k+1) = y(k) - T_s \tanh(y(k) + u^3(k)), \tag{11}$$

The control signal u was generated by a random number generator with normal distribution in the magnitude range $u \in [-1; 1]$. Its rate of change was $T_u = 6T_s$, *i.e.* the signal was kept constant for six time instants. The number N of the input signal samples, used for training, which determines the dimension of the covariance matrix, is $N = 200$. We would like to obtain a Gaussian process model for the discrete-time system (11).

Based on the generated data set, the discrete-time system (11) is approximated with Gaussian process with zero mean and the covariance function of the form (1). The maximum likelihood framework was used to determine the hyperparameters. The optimization method applied for identification of the Gaussian process model was the conjugate gradient method with line searches [7]. Models of various orders were fitted, but the optimization found the first order model as the most appropriate. The optimisation gave the following set of hyperparameters at found maximum likelihood:

$$\Theta = [\, w_1, w_2, v, v_0] = [0.3952, 0.9754, 1.0333, 0.0354], \tag{12}$$

A validation control input signal, different from the one used for the identification, was generated by random number generator with normal distribution. For the validation of the GP model the simulation and not one-step-ahead prediction was used. The response of the Gaussian process model to the validation signal is shown in Fig. 1 in the form of prediction means together with the 95% confidence band, corresponding to the interval $\pm 2\sigma$.

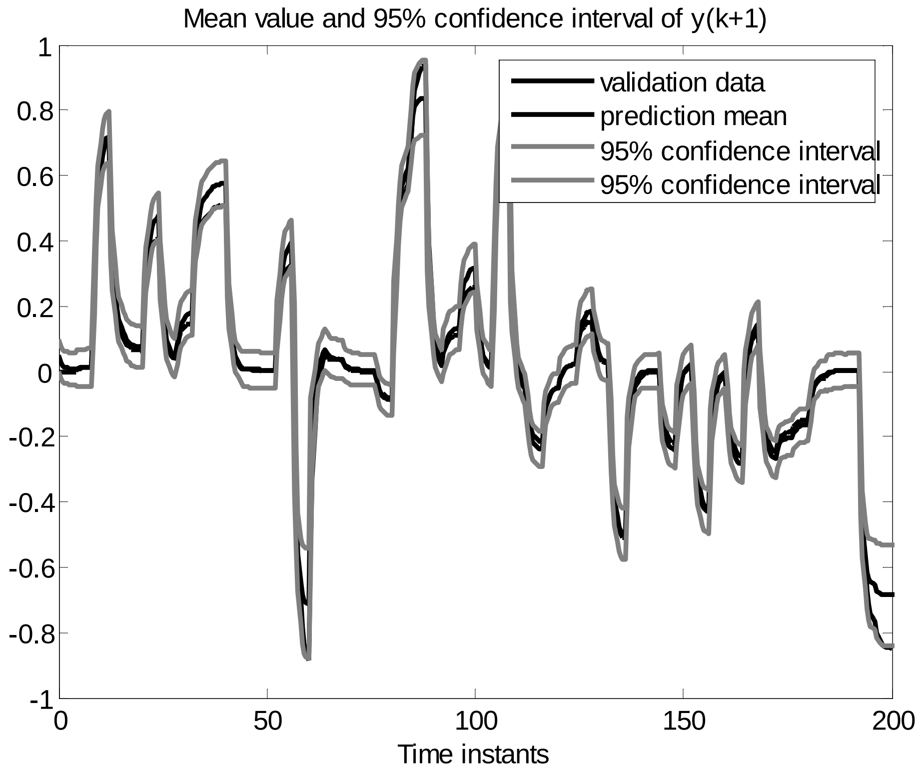

Fig. 1. Response of the Gaussian process model to the excitation signal used for validation.

The goodnes of the fit of the validation signal was assessed by average squared error as the most frequently used performance measure for comparison with other black-box methods and log-density error as the performance measure usually used for GP models. Their values are *ASE* = 0.0017 and *LD* = −2.1476, respectively.

CONCLUSIONS AND FUTURE WORK

The Gaussian process model is an example of a flexible, probabilistic, nonparametric model with inherent uncertainty prediction. It has been used for the modelling of complex nonlinear systems and recently also for the dynamic systems identification. This paper presents the concept for the development of a Gaussian Process Based System Identification Toolbox for the use with Matlab.

The toolbox concept consists of the sets of functions, corresponding to the specific identification tasks, and the list of needed functions to be included into the toolbox to fulfil the needs met at the dynamic system identification: model training, simulation, validation and application.

The next step, according to a linear software developing life cycle, is the preparation of the software specifications, followed by the software coding in Matlab syntax. The existing software code is likely to be redesigned and/or incorporated into the new software.

REFERENCES

[1] Quinonero-Candela, J. , C.E. Rasmussen. Analysis of Some Methods for Reduced Rank Gaussian Process Regression. In: Murray-Smith, R. and R. Shorten (Eds.), Switching and Learning in Feedback Systems, Lecture Notes in Computer Science, Vol. 3355, 2005.

[2] Solak, E., R. Murray-Smith, W.E. Leithead, D.J. Leith, and C.E. Rassmusen, Derivative observations in Gaussian Process Models of Dynamic Systems. In: Becker, S., S. Thrun, K. Obermayer (Eds.), Advances in Neural Information Processing Systems 15, MIT Press, 2003, 529–536.

[3] Kocijan, J., A. Girard, B. Banko, R. Murray-Smith. Dynamic Systems Identification with Gaussian Processes. Mathematical and Computer Modelling of Dynamical Systems, Vol. 11, 2005, 411–424.

[4] Gregor**čič**, G., G. Lightbody. Gaussian Processes for Modelling of Dynamic Non-linear Systems. In: Proc. Irish Signals and Systems Conference, Cork, Ireland, 2002.

[5] Girrard, A., Approximate Methods for Propagation of Uncertainty with Gaussian Process Models. PhD thesis, University of Glasgow, Glasgow, UK, 2004.

[6] Ažman, K., J. Kocijan. An example of Gaussian process model identification. In: Budin, L., S. Ribari**ć** (Eds.), Proc. 28th International conference MIPRO – CIS , Opatija, Croatia, 2005, 79–84.

[7] Rasmussen, C.E. and Williams, C.K.I., Gaussian Processes for machine learning. The MIT Press, Cambridge, MA, 2006.

[8] Ažman, K., J. Kocijan, Comprising Prior Knowledge in Dynamic Gaussian Process Models, In: B. Rachev, A. Smrikarov (Eds.), Proceedings of the International Conference on Computer Systems and Technologies - CompSysTech 2005, 16.-17.6., Varna, Bulgaria, 2005, sec IIIB.2.

[9] Ažman, K., J. Kocijan, An Application of Gaussian Process Models for Control Design, In: *Proc. International Control Conference ICC 2006*, 30.8.-1.9., Glasgow, Scotland, UK, 2006.

ABOUT THE AUTHORS

Prof. Juš Kocijan, PhD, Jožef Stefan Institute, Ljubljana, and University of Nova Gorica, Nova Gorica, Slovenia, Phone: +386 5 3315 285, E-mail: jus.kocijan@ijs.si.
Kristjan Ažman, M.Sc., Jožef Stefan Institute, Ljubljana, Slovenia.
Assoc. Prof. Alexandra Grancharova, PhD, Institute of Control and System Research, Bulgarian Academy of Sciences, Sofia, Bulgaria.