MEDNARODNA
PODIPLOMSKA ŠOLA
JOŽEFA STEFANA

6. ŠTUDENTSKA KONFERENCA MEDNARODNE PODIPLOMSKE ŠOLE JOŽEFA STEFANA

Zbornik – 1. del

6th JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL STUDENTS' CONFERENCE

Proceedings – Part 1

20. – 22. 05. 2014, LJUBLJANA

Zbornik 6. Študentske konference Mednarodne podiplomske šole Jožefa Stefana
(Proceedings of the 6th Jožef Stefan International Postgraduate School Students Conference)

Uredniki / Editors:
Nejc Trdin
Jernej Pavlič
Božidara Cvetković
Nemanja Aničić
Majda Pavlin
Andraž Rešetič

Fotografije / Photos:
Nikola Simidjievski

Založnik / Publisher:
Mednarodna podiplomska šola Jožefa Stefana, Ljubljana

Dosegljivo na / Attainable at:
http://ipssc.mps.si/2014/proceedingsIPSSC2014Papers.pdf

Ljubljana, maj 2014

Konferenco organizira Študentski Svet Mednarodne podiplomske šole Jožefa Štefana
(The Conference is organized by Jožef Stefan International Postgraduate School - IPS Student Council)

CIP - Kataložni zapis o publikaciji
Narodna in univerzitetna knjižnica, Ljubljana

5/6(082)(0.034.2)
378.046-021.68:001.891(497.4)(082)(0.034.2)

MEDNARODNA podiplomska šola Jožefa Stefana. Študentska konferenca (6 ; 2014 ; Ljubljana)
Zbornik [Elektronski vir] : 1. del = Proceedings : part 1 / 6. študentska konferenca Mednarodne
podiplomske šole Jožefa Stefana = 6th Jožef Stefan International Postgraduate School Students' Conference, 20.-22. 05. 2014, Ljubljana ; [organizira študentski svet Mednarodne podiplomske šole Jožefa Štefana = organized by Jožef Stefan International Postgraduate School - IPS Student Council] ; uredili, edited by Nejc Trdin ... [et al.]. - El. knjiga. - Ljubljana : Mednarodna podiplomska šola Jožefa Stefana, 2014

Način dostopa (URL): http://ipssc.mps.si/2014/proceedingsIPSSC2014Papers.pdf

ISBN 978-961-92871-6-3 (pdf)
1. Trdin, Nejc 2. Mednarodna podiplomska šola Jožefa Stefana (Ljubljana)
273807360

Model predictive control of bioreactor with Evolving Gaussian process model

Martin Stepančič[1,2], **Juš Kocijan**[2,3]

[1] Department of systems and control, Jožef Stefan Institute, Ljubljana, Slovenia

[2] Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

[3] University of Nova Gorica, Nova Gorica, Slovenia

martin.stepancic@ijs.si

Abstract. The paper presents a case study on adaptive nonlinear model predictive control (MPC) based on a Gaussian process (GP) model. MPC requires a model of the controlled system. We identify a NARX GP model using only 15 measurements of inputs and outputs. The model prediction itself is a normally distributed random variable. The information from a normally distributed prediction is used for implementation of probabilistic model predictive control. Our goal is to illustrate the effects on the controlled system performance. By examining the empirical results under the specified requirements, we can infer that the control performance is acceptable.

Keywords: Adaptive model predictive control, Gaussian process model.

1 Introduction

Control systems are most often based on the principle of feedback, whereby the signal to be controlled is compared to a desired reference signal and the discrepancy used to compute corrective control action. The term named *closed-loop control* comes from the information path in the system: process inputs have an effect on the process outputs, which is measured with sensors and processed by the controller to form a control signal. This signal is "fed back" as input to the process, closing the loop. Methods such *model predictive control* (MPC) were developed to make control of nonlinear systems to perform as close as possible to optimality. The idea of MPC is that a control performance test is measured on a model by finding the optimal input signal. The control performance relies on a criterion to be minimized which is called

a *cost function*. When finding an optimal control performance according to the cost function, a part of this input signal is applied to the real process. We will focus specially on the variance obtained from the probabilistic model and we will use this information inside a cost function.

2 Model predictive control

Model predictive control (MPC) is an intuitive and advanced approach for the control of dynamical systems. It requires a model of the controlled process and this model can be as simple as a step response in time-domain or a first-principle one, described with partial differential equations. The model is used by an optimization algorithm which simulates the process output to find a suitable control input which is then partially applied to the process. The devotion to output response optimality is expressed in terms of cost function minimization under some feasibility constraints but it always depends on the model accuracy. A cost function takes three arguments in general: the reference point where the process is wanted to be driven, the simulated output from the model and input to the model.

The usual way of computer-aided control design restricts the process output sampling and input control action to be taken at discrete-time intervals[1]. In a similar way we are dealing with the discrete model. We can present the values of a simulated output signal for a given input as discrete-time values for a finite number of discrete-time steps as shown in Figure 1. H_p is called the *predictive horizon* and is the number of total time steps we take into account for predicting the future signals. MPC control is called also *receding horizon* control because the optimal control input is recalculated by each new discrete-time instant. Another point of MPC is how an input signal is chosen. A possible simple design is to set a parameter for each step till the end of *control horizon* H_u is reached and the sequent input signal is set to a constant value till the end of prediction horizon.

[1] We omit the discretization problem of continuous systems

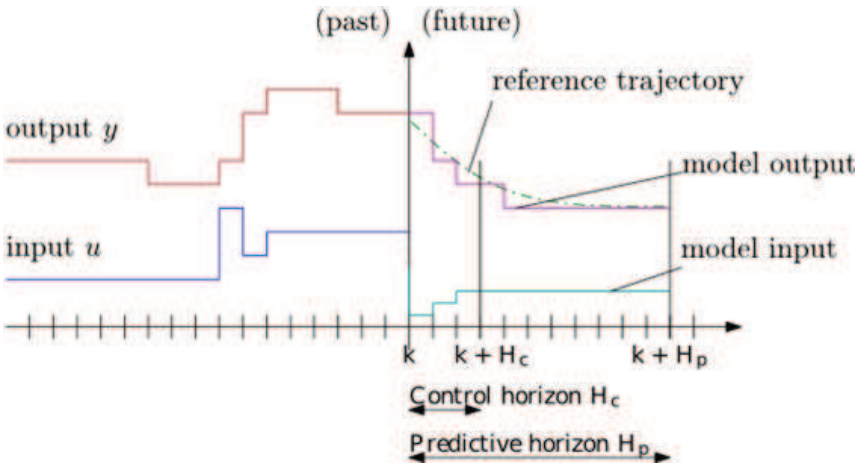

Figure 1 : Illustrative example of input optimization within the receding horizon context

From a practical point of view, the concept of receding horizon uses the predictive horizon as a moving frame inside which a sequence of future H_p discrete input values is chosen to optimize the simulated (model) response with same initial state as the current state of the controlled process. By matching the current state of process with the model, we apply a feedback from the process state to the model and form a *closed-loop* control. The closed-loop concept occurs because using the state of process is a persistent observation of the system output and this information is fed back to the regulator part.

Adaptive controller is the controller that continuously adapts to some changing process. These are meant for the control of time-varying nonlinear systems or for time-invariant nonlinear systems that are modeled as parameter-varying simplified nonlinear models. The designed scheme used for adaptive MPC is shown in Figure 3. The optimizer uses a model to simulate, searching the desired response r by finding a suitable input which will be then partially applied to the plant. Furthermore, the control algorithm is altered to an adaptive one which repeatedly updates the model online. This structure is shown as *model identification* block. The data for identification is made by taking the process input u and output y. A problem occurs when such control system starts without any identification data to build a model. We override this by giving an initial model.

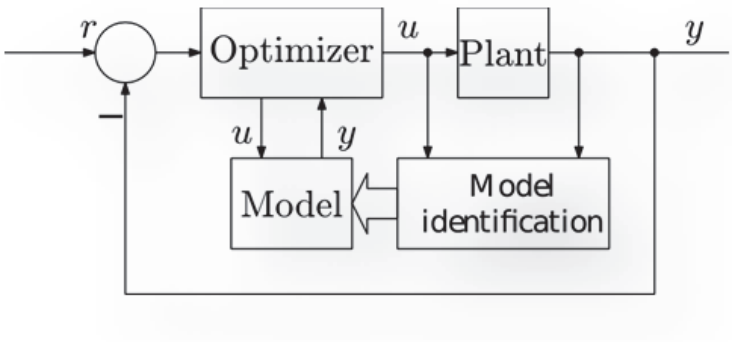

Figure 2 : Scheme of adaptive MPC algorithm

3 Model identification

The MPC control algorithm requires a model of the controlled system. We consider a black-box dynamic model in the NARX representation [1-2], where the output at time step k depends on the delayed outputs y and the exogenous control inputs u:

$$y(k) = f\big(y(k-1), \dots, y(k-L), u(k-1), \dots, u(k-L)\big) + \epsilon(k), \qquad (1)$$

where f denotes a function, ϵ is white noise and the output $y(k)$ depends on the state vector $\boldsymbol{x}(k) = [y(k-1), \dots, y(k-L), u(k-1), \dots, u(k-L)]^T$ [1]. Assuming the signal is known up to k, we wish to predict the output of the system l steps ahead, i.e., we need to find the predictive distribution of $y(k+1)$ corresponding to $x(k+1)$, if a probabilistic model is taken into account. Multi-step-ahead predictions of a system modelled by (1) can be achieved by iteratively making repeated one-step-ahead predictions, up to the desired horizon [1]. One of possible implementations of a NARX model is the Gaussian process model which will be presented in subsection 3.1.

3.1 GP model

GP model is a probabilistic, non-parametric model based on the principles of Bayesian probability [3]. It is probabilistic because its prediction is normally distributed and it is non-parametric because it has no structural evidence of a modeled system. This kind of modeling is classified as supervised learning and during the building phase it depends on a learning set. In our case, the learning set can be percieved as the model itself. The learning set $\mathcal{U}$ of our model is composed from delayed input and output signal measurements of the process. This kind of

data is followed from the NARX model form. Each element $\{\boldsymbol{x}_i, y_i\} \in \mathcal{U}$ is split into an input vector $\boldsymbol{x}_i$ and its predictive target y_i for $i = 1, \dots, N$ where N is the size of learning set $\mathcal{U}$. The output values y_i are assumed to be noisy measurements of an underlying function $f(\boldsymbol{x}_i)$ with conditional probability distribution $p(y_i|f_i) = \mathcal{N}(f_i, \sigma^2)$. Let $\boldsymbol{f} = [f(\boldsymbol{x}_1), \dots, f(\boldsymbol{x}_N)]^T$ and $\boldsymbol{f} = [y_1, \dots, y_N]^T$, then the learning set $\mathcal{U}$ is used to form a joined Gaussian distribution of function values $\boldsymbol{f}$ [4]. This is a Gaussian process and it is defined as a collection of random variables with joined Gaussian distribution $p(y_i|\mathcal{U}) = \mathcal{N}(\mathbf{0}, \mathbf{K})$ where K is a (semi-positive definite) *covariance matrix* which inherits the input part of the learning set $\mathcal{U}$ by mapping its paired inputs $\boldsymbol{x}_i, \boldsymbol{x}_j$ with a *covariance function* $\boldsymbol{k}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Intuitively, the covariance function $\boldsymbol{k}$ returns a scalar value, representing how two inputs from $\mathcal{U}$ are related to each other. For now, we keep in mind just what covariance function does, but not how it is made. A common aim in regression is to predict the output y^* from a new input $\boldsymbol{x}^*$ given the learning set $\mathcal{U}$ and a known covariance function $\boldsymbol{k}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. It can be shown that the single posterior distribution $p(y^*|\mathcal{U}, \boldsymbol{x}^*)$ can be analytically solved [4], hence we get the form of GP model prediction:

$$p(y^*|\boldsymbol{x}^*, \mathcal{U}) = \mathcal{N}(y^*| \ \mathbf{k}^{*T}\mathbf{K}^{-1}\boldsymbol{y} \ \ , k(\boldsymbol{x}^*, \boldsymbol{x}^*) - \mathbf{k}^{*T}\mathbf{K}^{-1}\mathbf{k}^*) \qquad (2)$$

where $\mathbf{k}^* = [k(\boldsymbol{x}_1, \boldsymbol{x}^*), \dots, k(\boldsymbol{x}_N, \boldsymbol{x}^*)]^T$ is the vector of covariance function values between the inputs $\boldsymbol{x}_i \in \mathcal{U}, i = 1, \dots, N$ and the prediction input $\boldsymbol{x}^*$.

The *covariance function* design was omitted but it is essentially the main part of GP model structure along the learning set $\mathcal{U}$. Inference in GP firstly involves finding the form of covariance function $\boldsymbol{k}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ to provide a Bayesian interpretation of *kernel methods* [2][3]. Its value expresses the correlation between the individual outputs y_i and y_j with respect to inputs $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$[3]. Usually, the covariance function is used along with some parameters, i.e. *hyperparameters*. The use hyperparameters can highlight or neglect the regressors from an input vector $\boldsymbol{x}_i$. Assuming stationary data is contaminated with white noise, most commonly used covariance function is a composition of the square exponential (SE) covariance function with "automatic

[2] The theory of kernel methods will not be discussed here. For more information, some surveys into kernel methods are provided [4-6].

relevance determination" [7] (ARD) hyperparameters and an additional term δ_{ij} for the white noise assumption [3]:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = v_0 \exp\left(-\frac{1}{2}\sum_{d=1}^{D} \vartheta_d (x_{id} - x_{jd})^2\right) + v_1 \delta_{ij}, \quad (3)$$

where ϑ_d are the automatic relevance determination hyperparameters, v_1 and v_0 are hyperparameters of the covariance function, D is the input dimension, and $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise. The method of setting hyperparameters $\boldsymbol{\theta} = [v_0, v_1, \vartheta_1, \dots, \vartheta_d]$ will not be discussed here, but can be further provided in [3-4][7].

3.2 Evolving GP model

This subsection is summarized from [3]. The Evolving GP model (EGP) is inspired by Evolving systems [8], which are self-developing systems, adapting on-line both, structure and parameter values of the model from incoming data [8]. We use the term Evolving GP models in sense of sequential adapting of both, the "structure" of GP model and hyperparameter values. This enables fast and efficient GP model adaptation to the time-varying system. In comparison with the learning set $\mathcal{U}$ of a GP model, the learning set of an EGP model $\mathcal{U}_A$ is said to be an *active set* with the property that only a subset $\mathcal{U}_A \subset \mathcal{U}$ of entire learning dataset $\mathcal{U}$ is used for modeling with EGP.

Similarly as in [9] we decided to use fixed squared exponential (SE) covariance function with ARD (3) because its functionality is able to find influential regressors. With the optimization of the hyperparameter values, uninfluential regressors have consequently smaller influence to the result. Therefore, all available regressors can be used and consequently, only the active set $\mathcal{U}_A$ and hyperparameter values $\boldsymbol{\theta}$ are to be adapted sequentially. In general the proposed method consists of three main steps to adapt the GP model sequentially: Update of active learning set, hyperparameter optimization, covariance matrix inverse calculation.

In our specific case we have an EGP of NARX form whose incoming data consists from an input vector $\boldsymbol{x}_i$ of delayed inputs and outputs and its target value y_i of the current output. For every new incoming data, the novelty of the data according to the current GP model is verified. This is simply done by predicting the output mean value $E[y_i^*]$ of the incoming input vector $\boldsymbol{x}_i$ and comparing to the measured value

y_i. If the condition $|E[y_i^*] - y_i| > \zeta_{EGP}$ is true for a pre-set threshold ζ_{EGP}, the element $\{\boldsymbol{x}_i, y_i\} \in \mathcal{U}$ is added to the active set $\mathcal{U}_A$. A method for excluding elements must be used if the active learning set has to be limited to a maximum size. This methodology will not be discussed here but more information about excluding elements from an active set is available from [3,9-10].

4 Case study

4.1 Bioreactor

The adaptive MPC-GP method will be examined with a simplified model of bioreactor [12]. It is an open-loop stable, nonlinear and second order system, desribed with difference equations:

$$x_1(k+1) = 0.5\,\frac{x_1(k)x_2(k)}{x_1(k)+x_2(k)} - 0.5\,u(k)\,x_1(k), \tag{4}$$

$$x_2(k+1) = 0.5\,\frac{x_1(k)x_2(k)}{x_1(k)+x_2(k)} - 0.5\,u(k)\,x_2(k) + 0.05\,u(k), \tag{5}$$

$$y(k) = x_1(k) + \epsilon(k), \tag{6}$$

where u is system input, limited to $[0,0.7]$, x_1 and x_2 are system states, and the output y is contaminated with a normally distributed noise ϵ with $p(\epsilon) = \mathcal{N}(0,0.001)$.

4.2 Control design

The cost function:

$$J = \sum_{i=1}^{H_p} (r(k+i) - E[y^*(k+i)])^2 + \gamma_v \,\mathrm{var}[y^*(k+i)] \tag{7}$$

is used to find the optimal control input u.

Because we need an initial GP model to perform effectively a simple proportional (P) regulator was used to train a GP model in closed-loop in the first $0 \leq k \leq 30$ time steps. At $k > 30$ the adaptive MPC-EGP regulator was activated and replaced the proportional one. The error threshold for EGP model update is set to $\zeta_{EGP} = 0.021$ and we restricted the EGP active learning set $\mathcal{U}_A$ to a maximum of 15 learning points.

Just a representative segment of the closed-loop performance is shown in Figure 3 for prediction horizon $H_p = 8$, control horizon $H_u = 1$ and cost function parameter $\gamma_v = 0.14$.

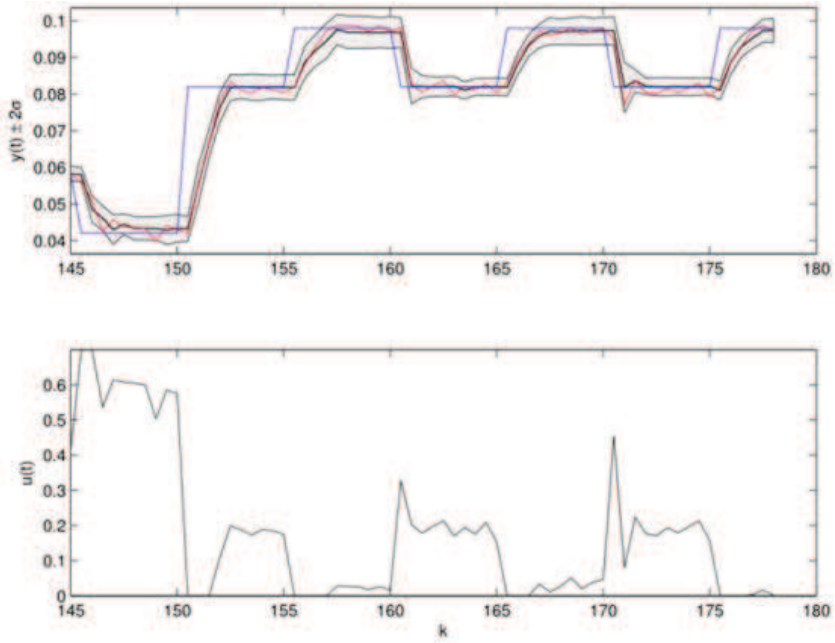

Figure 3: Closed-loop control of bioreactor. The upper window contains a reference signal (blue), process output (red) and one-step prediction mean with double std. deviation (black with gray gap). The lower window is control input.

5 Conclusion

Our goal was to illustrate the controlled system performance using an EGP model with a limited learning set to 15 data inputs. The results from Figure 3: Closed-loop control of bioreactor. The upper window contains a reference signal (blue), process output (red) and one-step prediction mean with double std. deviation (black with gray gap). The lower window is control input. show that the performance is acceptable. Using a larger prediction horizon is unnecessary for this specific case.
One should note that we implemented an adaptive control algorithm which adapts the GP model on-line and its prediction could predict a much smaller uncertainty compared to an offline GP model.

References:

[1] J. Kocijan. Control Algorithms Based on Gaussian Process Models: A State-of-the-Art Survey. In *Special International Conference on Complex systems: synergy of control com-munications and computing*, September 16-20, 2011, Ohrid, Republic of Macedonia.

[2] R. Sałat, M. Awtoniuk, and K. Korpysz, "Black-Box system identification by means of Support Vector Regression and Imperialist Competitive Algorithm," in Przeglad Elektrotechniczny, 2013.

[3] D. Petelin and J. Kocijan: Evolving Gaussian process models for predicting chaotic timeseries. In *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, 2014.

[4] C. Bishop, Pattern Recognition and Machine Learning. Springer, 2006.

[5] G. Pillonetto, F. Dinuzzo, T. Chen, G. D. Nicolao, and L. Ljung, Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica*, vol. 50, no. 3, pp. 657–682, 2014.

[6] C. Campbell, Kernel methods: a survey of current techniques. *Neurocomputing*, vol. 48, pp. 63–84, 2002.

[7] C. E. Rasmussen and C. K. I. Williams, Gaussian Processes for Machine Learning. MIT Press, 2006.

[8] P. Angelov, D. P. Filev, and N. Kasabov: Evolving intelligent systems: Methodology and applications. In *IEEE Press Series on Computational Intelligence*, Wiley IEEE Press, April 2010.

[9] D. Petelin, A. Grancharova, and J. Kocijan: Evolving Gaussian process models for prediction of ozone concentration in the air. *Simulation Modelling Practice and Theory*, vol. 33, pp. 68 – 80, 2013.

[10] D. Petelin and J. Kocijan: Control system with evolving Gaussian process models. In *Evolving and Adaptive Intelligent Systems (EAIS), 2011 IEEE Workshop on*, pp. 178–184, 2011

[11] K. Ažman and J. Kocijan: Application of gaussian processes for black-box modelling of Biosystems, *ISA Transactions*, vol. 64, pp. 443–457, 2007

For wider interest

Bioreactor processes are the core manufacturing process in the biotech industry. Delays and process upsets can result in the loss of money in revenue through lost product and downtime. Because the bioreactor is such a critical component, keeping it running is essential to the profitability of a biotech operation. For the efficient operation high-quality control is necessary. Processes demonstrating highly nonlinear behaviour such bioreactors can be operated in regimes closer to the process optimum, where simple controllers may fail. Unfortunately, the precise and appropriate model for MPC requires significant time and effort to construct and the proposed adaptive MPC using a probabilistic black-box model might be an efficient solution.