Closed-loop Control with Evolving Gaussian Process Models

Juš Kocijan[1,2] and Dejan Petelin[1]

[1] Jožef Stefan Institute, Ljubljana, Slovenia,
jus.kocijan@ijs.si, dejan.petelin@ijs.si
[2] University of Nova Gorica, Nova Gorica, Slovenia

Abstract. This contribution presents a new development in the design of control system based on evolving Gaussian-process (GP) models. GP models provide a probabilistic, nonparametric modelling approach for black-box identification of nonlinear dynamic systems. They can highlight areas of the input space where prediction quality is poor, due to the lack of data or its complexity, by indicating the higher variance around the predicted mean. GP models contain noticeably less coefficients to be optimised than commonly used parametric models. While GP models are Bayesian models, their output is normal distribution, expressed in terms of mean and variance. Latter can be interpreted as a confidence in prediction and used in many fields, especially in control system. Evolving GP model is the concept approach within which various ways of model adaptations can be used. Successful control system needs as much as possible data about process to be controlled. If the prior knowledge about the system to be controlled is scarce or the system varies with time or operating region, this control problem can be solved with an iterative method which adapts model with information obtained with streaming data and concurrently optimises hyperparameter values. This contribution provides: a survey of adaptive control algorithms for dynamic systems described in publications where GP models have been used for control design, a novel and improved closed-loop controller with evolving GP models and an example for the illustration of proposed control algorithm.

Keywords: Dynamic systems modelling, Gaussian-process regression, Evolving Gaussian-process model, adaptive control.

1 Introduction

Increasingly complex systems are expected to be handled with new technologies among them with control-system technologies. There exist a range of control design methods depending on the sort of system model and amount of information that is available. Various adaptive, self-learning or other learning approaches, are frequent to tackle the problem of low initial prior knowledge about the system to be controlled or in the case of time-variant or nonlinear systems. Often various kinds of computational intelligence methods for model development that result

in so-called black-box models are used for these kinds of control problems. This paper deals with control system design based on Gaussian process (GP) models.

GP models provide a probabilistic, nonparametric modelling approach for black-box identification of nonlinear dynamic systems. They can highlight areas of the input space where model-prediction quality is poor, due to the lack of data or its complexity, by indicating the higher variance around the predicted mean. This property can be incorporated in the closed-loop control design. GP models contain noticeably less coefficients to be optimised than parametric models that are frequently used in control design.

The aim of this chapter is to present an improved closed-loop controller with evolving GP models and place it within contemporary research on adaptive control methods based on GP models and to demonstrate a proposed control algorithm based on GP model.

The chapter is structured as follows. First the modelling with Gaussian processes in general and the modelling of dynamic systems with GP models is explained. Then a short review of adaptive-control methods based on GP models is given. Adaptive control with evolving GP model is introduced next. The control method is demonstrated with an illustrative example. This example demonstrates the performance and the adaptation of closed-loop tracking control in different operating regions with a computer simulation study.

2 Systems Modelling with Gaussian Processes

A GP model is a flexible, probabilistic, nonparametric model that enables the prediction of output-variable distributions. Contrary to parametric modelling methods where a structure is usually presumed and the parameters are optimised with regression, the GP-based modelling is different in the sense that the structure of the mapping function is not presumed, but the data themselves are used to describe the mapping function. The modelled system is, therefore, not approximated by fitting the parameters of the selected basis functions, but rather with the relationship among the measured data. The model of the nonlinear mapping function is called the GP model as the output of the GP model is by prior belief considered to be a GP. GP model's properties and application potentials are reviewed in [28].

A GP is a collection of random variables that have a joint multivariate Gaussian distribution. Assuming a relationship of the form $y = f(\boldsymbol{x})$ between the input $\boldsymbol{x}$ and the output y, we have $y_1, \ldots, y_N \sim \mathcal{N}(0, \boldsymbol{\Sigma})$, where $\Sigma_{pq} = \mathrm{Cov}(y_p, y_q) = C(\boldsymbol{x}_p, \boldsymbol{x}_q)$ gives the covariance between the output points corresponding to the input points described by vectors $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$. Thus, the mean $\mu(\boldsymbol{x})$ and the covariance function $C(\boldsymbol{x}_p, \boldsymbol{x}_q)$ fully specify the Gaussian process.

The value of the covariance function $C(\boldsymbol{x}_p, \boldsymbol{x}_q)$ expresses the correlation between the individual outputs $f(\boldsymbol{x}_p)$ and $f(\boldsymbol{x}_q)$ with respect to the inputs $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$. Note that the covariance function $C(\cdot, \cdot)$ can be any function that generates a positive semi-definite covariance matrix. It is usually composed of two parts,

$$C(\boldsymbol{x}_p, \boldsymbol{x}_q) = C_{\mathrm{f}}(\boldsymbol{x}_p, \boldsymbol{x}_q) + C_{\mathrm{n}}(\boldsymbol{x}_p, \boldsymbol{x}_q), \tag{1}$$

where C_{f} represents the functional part and describes the unknown system we are modelling, and C_{n} represents the noise part and describes the model of the noise.

For the noise part it is most common to use the constant covariance function, presuming white noise. The choice of the covariance function for the functional part also depends on the stationarity of the data used for modelling. Assuming stationary data the most commonly used covariance function is the square exponential covariance function. The composite covariance function is therefore

$$C(\boldsymbol{x}_p, \boldsymbol{x}_q) = v_1 \exp\left[-\frac{1}{2}\sum_{d=1}^{D} w_d (x_{dp} - x_{dq})^2\right] + \delta_{pq} v_0, \tag{2}$$

where w_d, v_1 and v_0 are the 'hyperparameters' of the covariance function, D is the input dimension, and $\delta_{pq} = 1$ if $p = q$ and 0 otherwise. In contrast, assuming non-stationary data the polynomial or its special case, the linear covariance function, can be used. Other forms and combinations of covariance functions suitable for various applications can be found in [28]. The hyperparameters can be written as a vector $\boldsymbol{\Theta} = [w_1, \ldots, w_D, v_1, v_0]^T$. The parameters w_d indicate the importance of the individual inputs: if w_d is zero or near zero, it means the inputs in dimension d contain little information and could possibly be neglected.

To accurately reflect the correlations present in the training data, the hyperparameters of the covariance function need to be optimised. Due to the probabilistic nature of the GP models, the common model optimisation approach, where model parameters and possibly also the model structure are optimised through the minimisation of a cost function defined in terms of model error (e.g., mean square error), is not readily applicable. A probabilistic approach to the optimisation of the model is more appropriate. Actually, instead of minimising the model error, the probability of the model is maximised.

GP models can be easily utilised for a regression calculation. Consider a matrix $\boldsymbol{X}$ of N D-dimensional input vectors where $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]^T$ and a vector of the output data $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]$. Based on the data $(\boldsymbol{X}, \boldsymbol{y})$, and given a new input vector $\boldsymbol{x}^*$, we wish to find the predictive distribution of the corresponding output y^*. Based on the training input data $\boldsymbol{X}$, a covariance matrix $\boldsymbol{K}$ of size $N \times N$ is determined. The overall problem of learning unknown parameters from data corresponds to the predictive distribution $p(y^*|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}^*)$ of the new target y^*, given the training data $(\boldsymbol{y}, \boldsymbol{X})$ and a new input $\boldsymbol{x}^*$. In order to calculate this posterior distribution, a prior distribution over the hyperparameters $p(\boldsymbol{\Theta}|\boldsymbol{y}, \boldsymbol{X})$ can first be defined, followed by the integration of the model over the hyperparameters

$$p(y^*|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}^*) = \int p(y^*|\boldsymbol{\Theta}, \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}^*) p(\boldsymbol{\Theta}|\boldsymbol{y}, \boldsymbol{X}) d\boldsymbol{\Theta}. \tag{3}$$

The computation of such integrals can be difficult due to the intractable nature of the nonlinear functions. A solution to the problem of intractable integrals is to adopt numerical integration methods such as the Monte-Carlo approach.

Unfortunately, significant computational efforts may be required to achieve a sufficiently accurate approximation.

In addition to the Monte-Carlo approach, another standard and general practice for estimating hyperparameters is the maximum marginal-likelihood estimation, i.e., to minimise the following negative log-likelihood function [28]:

$$\mathcal{L}(\boldsymbol{\Theta}) = -\frac{1}{2}\log(\mid \boldsymbol{K} \mid) - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}^{-1}\boldsymbol{y} - \frac{N}{2}\log(2\pi). \tag{4}$$

As the likelihood is, in general, nonlinear and multi-modal, efficient optimisation routines usually entail the gradient information. The computation of the derivative of $\mathcal{L}$ with respect to each of the parameters is as follows

$$\frac{\partial\mathcal{L}(\boldsymbol{\Theta})}{\partial\theta_i} = -\frac{1}{2}\text{trace}\left(\boldsymbol{K}^{-1}\frac{\partial\boldsymbol{K}}{\partial\theta_i}\right) + \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}^{-1}\frac{\partial\boldsymbol{K}}{\partial\theta_i}\boldsymbol{K}^{-1}\boldsymbol{y}. \tag{5}$$

For performing a regression, the availability of the training input data described with matrix $\boldsymbol{X}$ and the corresponding output data described with vector $\boldsymbol{y}$ is assumed. As already mentioned, the aim is to find the distribution of the corresponding output y^* for some new input vector $\boldsymbol{x}^* = [x_1(N+1), x_2(N+1), \ldots, x_D(N+1)]^T$.

For the collection of random variables $[y_1, \ldots, y_N, y^*]$ we can write:

$$[\boldsymbol{y}, y^*] \sim \mathcal{N}(0, \boldsymbol{K}^*) \tag{6}$$

with the covariance matrix

$$\boldsymbol{K}^* = \left[\begin{array}{c|c} \boldsymbol{K} & \boldsymbol{k}(\boldsymbol{x}^*) \\ \hline \boldsymbol{k}^T(\boldsymbol{x}^*) & \kappa(\boldsymbol{x}^*) \end{array}\right], \tag{7}$$

where $\boldsymbol{y} = [y_1, \ldots, y_N]$ is a $1 \times N$ vector of training targets. The predictive distribution of the output for a new test input has a normal probability distribution with a mean and variance [28]

$$\mu(y^*) = \boldsymbol{k}(\boldsymbol{x}^*)^T\boldsymbol{K}^{-1}\boldsymbol{y}, \tag{8}$$

$$\sigma^2(y^*) = \kappa(\boldsymbol{x}^*) - \boldsymbol{k}(\boldsymbol{x}^*)^T\boldsymbol{K}^{-1}\boldsymbol{k}(\boldsymbol{x}^*), \tag{9}$$

where $\boldsymbol{k}(\boldsymbol{x}^*) = [C(\boldsymbol{x}_1, \boldsymbol{x}^*), \ldots, C(\boldsymbol{x}_N, \boldsymbol{x}^*)]^T$ is the $N \times 1$ vector of covariances between the test and training cases, and $\kappa(x^*) = C(\boldsymbol{x}^*, \boldsymbol{x}^*)$ is the covariance between the test input itself.

The obtained model, in addition to the mean value, provides information about the confidence in the prediction by the variance of the predictive distribution. Usually, the confidence of the prediction is depicted with a 2σ interval, which is about 95% confidence interval. This confidence region can be seen in

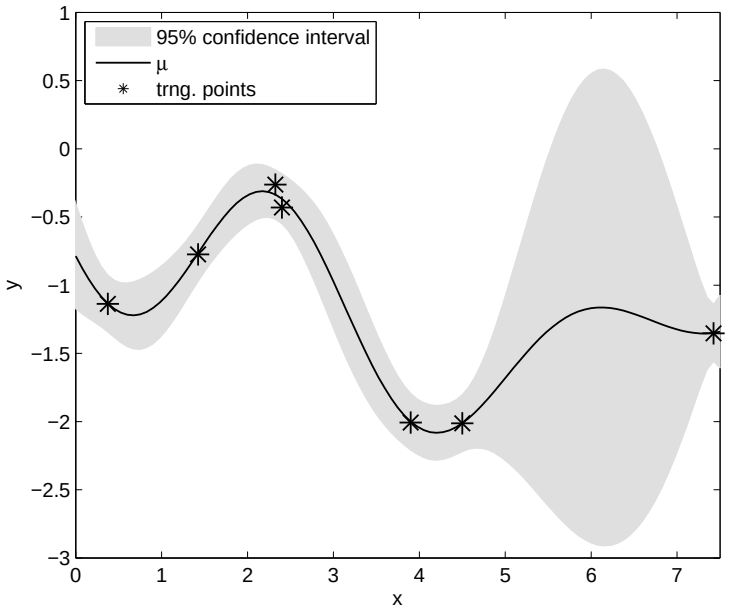

Fig. 1: Using GP models: in addition to the prediction mean value (full line), we obtain a 95% confidence interval (gray band) for the underlying function y.

the example in Fig. 1 as a grey band. It highlights the areas of the input space where the prediction quality is poor, due to the lack of data or noisy data, by indicating a wider confidence band around the predicted mean.

GP models can, like neural networks, be used to model static nonlinearities and can therefore be used for the modelling of dynamic systems [1, 15, 16] as well as time series, if lagged samples of the output signals are fed back and used as regressors. A review of recent developments in the modelling of dynamic systems using GP models and its applications can be found in [14]. It is important to stress that the model prediction in the form of GP is just an approximation when the Gaussian assumption is not fulfilled, which is in line with common engineering practice.

A dynamic GP model is trained as the nonlinear autoregressive model with an exogenous input (NARX) representation, where the output at time instant k depends on the delayed output y and the exogenous control input u:

$$y(k) = f_S(y(k-1), \ldots, y(k-n), u(k-1), \ldots, u(k-n)) + \nu(k), \qquad (10)$$

where f_S denotes a function, $\nu(k)$ is white noise disturbance with normal distribution and the output $y(k)$ depends on the state vector $\boldsymbol{x}(k) = [y(k-1), y(k-2), \ldots, y(k-n), u(k-1), u(k-2), \ldots, u(k-n)]$ at time instant k. This model notation can be generalised to mutivariable cases, i.e., cases with multiple inputs and outputs.

For the validation of obtained dynamic GP model the nonlinear output-error (NOE), also called parallel, model is used. This means that the NARX model is used to predict a further step ahead by replacing the data at instant k with the data at instant $k+1$ and using the prediction $\hat{y}(k)$ from the previous prediction step instead of the measured $y(k)$. This is then repeated indefinitely. The latter possibility is equivalent to simulation. Simulation, therefore, means that only on the basis of previous samples of a process input signal $u(k-i)$ can the model simulate future outputs. Frequently, the mean value of prediction $\hat{y}(k)$ is used to replace $y(k)$, which is called 'naive' simulation. Other possibilities, where the entire distribution is used, are described in, e.g., [16].

3 Adaptive Control Algorithms Based on Gaussian Process Models

Control is the activity that makes a system behave in the desired way. There are many reference books available describing a variety of control methods, their design procedures and their applications. This section provides only a review of some of the adaptive control methods that are based on GP model and were published in literature. Reader is referred to [13] for more comprehensive review of control methods based on GP model.

Adaptive controller is the controller that continuously adapts to some changing process. Adaptive controllers emerged in early sixties of the previous century. At the beginning these controllers were mainly adapting themselves based on linear models with changing parameters. Since then several authors have proposed

the use of nonlinear models as a base to build nonlinear adaptive controllers. These are meant for the control of time-varying nonlinear systems or of time-invariant nonlinear systems that are modelled as parameter-varying simplified nonlinear models.

Various divisions of adaptive control structures are possible. One possible division [12] is into open-loop and closed-loop adaptive systems.

Open-loop adaptive systems are gain-scheduling or parameter-scheduling controllers. Closed-loop adaptive systems can be further divided to dual and non-dual adaptive systems.

Dual adaptive systems [11, 35] are those where the optimisation of the information collection and the control action are pursued at the same time. The control signal should ensure that the system output cautiously tracks the desired set-point value and at the same time excites the plant sufficiently to accelerate the identification process. The solution to the dual control problem is based on dynamic programming and the resulting functional equation is often the Bellman equation. Not a large number of such controllers have been developed.

The difficulties to find the optimal solution for dual adaptive control lead to suboptimal adaptive dual controllers [11, 35] obtained by either various approximations or by reformulating the problem. Such a reformulated adaptive dual control problem is when a special cost function is considered, which consists of two added parts: control losses and an uncertainty measure. This is appealing for application with the GP model that provides measures of uncertainty.

Many adaptive controllers in general are based on the separation principle [35] that implies separate estimation of system model, i.e., system parameters, and the application of this model for control design. When the identified model used for control design and adaptation is presumed to be the same as the true system then the adaptive controller of this kind is said to be based on certainty equivalence principle and such adaptive control is named non-dual adaptive control. The control actions in non-dual adaptive control do not take any active actions that will influence the uncertainty.

When using the GP model for the adaptive control, different from gain-scheduling control, the GP model is identified on-line and this model is used in the control algorithm. The block scheme showing the general principle of adaptive control with the GP model identification is given in Fig. 2. It is sensible that advantages of GP models are considered in the control design, which relates the GP model-based adaptive control at least to suboptimal dual adaptive control principles. The uncertainty of model predictions obtained with the GP model prediction are dependent, among others, on local learning-data density, and the model complexity is automatically related to the amount and the distribution of the available data – more complex models need more evidence to make them likely. Both aspects are very useful in sparsely-populated transient regimes. Moreover, since weaker prior assumptions are typically applied in a nonparametric model, the bias is typically lower than in parametric models.

The above ideas are indeed related to the work done on the dual adaptive control, where the main effort has been concentrated on the analysis and de-

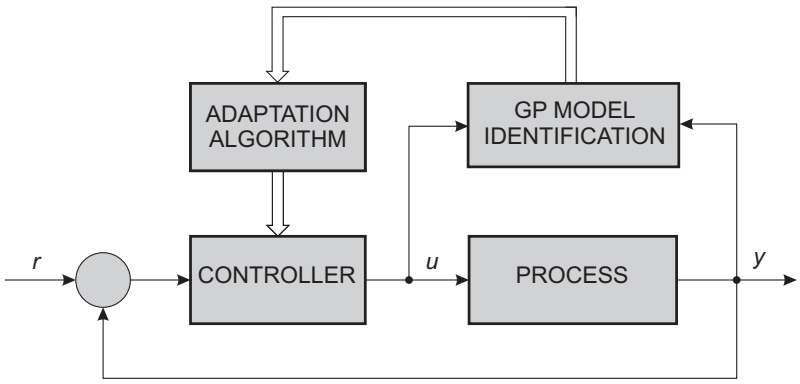

Fig. 2: General block scheme of the closed-loop system with adaptive controller

sign of adaptive controllers based on the use of the uncertainty associated with parameters of models with a fixed structure [11, 31].

The major differences in up-to-now published adaptive systems based on GP models are in the way how the on-line model identification is pursued.

Increasing the size of the covariance matrix, i.e., 'blow-up model', with the in-streaming data and repeating model optimisation is used in papers [19], [18], [30], [29] and [31], where more attention is devoted to control algorithms and their benefits based on information gained from the GP model and not on the model identification itself.

Another adaptive control algorithm implementation is control with feedback for cancelling nonlinearities with the on-line learning of the inverse model. This sort of adaptive control with the increasing covariance matrix with the in-streaming data is described in [21]. Two sorts of on-line learning for the mentioned feedforward contained control is described in [22]. The first sort is with moving window strategy, where the old data are dropped from the on-line learned model, while the new data is accommodated, the second one accommodates only new data with sufficient information gain. These applications of referenced inverse GP models do not use entire information from the prediction distribution, but they focus on the mean value of prediction.

A lot of GP model-based adaptive-control algorithms from the referenced publications are based on the Minimum Variance controller. One of the reasons is that the Minimum Variance controller explores the variance that is readily available with the GP model prediction.

The Minimum Variance controller in general [12] looks for a control signal $u(k)$ in time instant k, that will minimise the following cost function:

$$J_{\mathrm{MV}} = E(\|\boldsymbol{r}(k) - \boldsymbol{y}(k+p)\|^2). \tag{11}$$

In this case, J_{MV} refers to the covariance of the error between the vector of set-points $\boldsymbol{r}(k)$ and the controlled outputs p-time steps in the future, $\boldsymbol{y}(k+p)$. The desired controller is thus the one that minimises these variances, hence the name

Minimum Variance control. The optimal control signal $\boldsymbol{u}_{\text{opt}}$ can be obtained by minimising selected cost function. The minimisation can be done analytically, but also numerically, using any appropriate optimisation method.

The cost function (11) can be expanded with a penalty terms and generalised to multiple-input multiple-output case leading to Generalised Minimum Variance control [30].

$$J_{\text{GMV}} = E(||\boldsymbol{r}(k+1) - \boldsymbol{y}(k+1)||^2_{\boldsymbol{Q}}) + ||\boldsymbol{u}_k||^2_{\boldsymbol{R}}, \tag{12}$$

where matrix $\boldsymbol{Q}$ is positive definite matrix and $\boldsymbol{R}$ is polynomial matrix with the backward shift operator q^{-1}. The matrix $\boldsymbol{Q}$ elements and matrix $\boldsymbol{R}$ polynomial coefficients can be used as tuning parameters.

The method named Gaussian Process Dynamic Programming (GPDP) is a Gaussian-process-model-based adaptive control algorithm with a proximity to dual adaptive control. The details of the method are described in [8]. The following description is summarised from [8] and [10]. The evolution of the method can be followed through time with publications [27], [9], [26],[10] and [8].

GPDP is an approximate dynamic programming method, where cost functions, so-called value functions in the dynamic programming recursion are modelled by GPs.

The reader is referred to [8] for details and a demonstration of the method. Unfortunately, according to the method's authors [8], ALGPDP cannot be directly applied to a dynamic system because it is often not possible to experience arbitrary state transitions. Moreover GPDP method does not scale that well to high dimensions.

More promising for engineering control applications is Probabilistic Inference and Learning for COntrol (PILCO) method, described in [5], [6] and [7].

PILCO is a policy search method and an explicit value function model is not required as in GPDP method. The general idea of the method is to learn the system model with Reinforcement learning and control the closed-loop system, taking into account the probabilistic model of the process. The algorithm can be divided into three layers: a top level for the controller adaptation, an intermediate layer for the approximate inference for long-term predictions and a bottom layer for learning the model dynamics. A start state $\boldsymbol{x}_0$ is required by the algorithm in the beginning.

The PILCO method was applied to real systems, e.g. robotic systems [7].

4 Evolving GP-model-based control

An adaptive Minimum Variance controller based on evolving Gaussian process model [23] is presented in this section. The basic idea of control based on evolving system model is that system GP model evolves with in-streaming data and the information about system from the model is used for its control. One option is that the information can be in the form of GP model prediction for one or several steps ahead which is then used to calculate optimal control input in the controlled system. The other option would be, for example, prediction of some

particular part of the model, e.g., parameters, like in [2], and on-line calculation of controller.

The proposed control is actually a variation of the control proposed in [24]. The main difference is an on-line learning method used for adapting GP model. It should be noted that an efficient adaptation of the GP model is crucial, as the calculation of the optimal control input is based on the GP model's prediction. However, a noticeable drawback of GP model identification is the computation load that increases with the third power of the amount of input data due to the calculation of the inverse of the covariance matrix. This computational complexity restricts the amount of training data to, at most, a few thousand cases. To overcome the computational-limitation issues, only a subset of the most informative data is to be used. In the literature, such a subset is called the active set or the basis vectors set [4] and its elements are called basis vectors [4], inducing variables [25] or basis functions [17]. The basic idea is to retain the bulk of the information contained in the full training dataset, but reduce the size of the covariance matrix so as to facilitate a less computationally demanding implementation of the GP model. As the data is in-streaming the GP model should be adapted continuously. In other words, the on-line learning method should processes every new piece of streaming data sequentially.

In the previous version of the controller [24] we used Csato's method Sparse online Gaussian processes [4]. It's main disadvantage is twofold. The first one is a lack of ability to update hyperparameter values in an online fashion. In other words, for adequately learning optimised hyperparameters are needed in advance, so their values should be optimised before we apply controller to the system. Usually, there is some available data that can be used for optimisation of hyperparameter values, but obtained values are optimal only for the current data presenting the system's dynamics. But, if the data presents only one region of the system's dynamics or if the system is time variant, obtained hyperparameter values most likely will not be optimal enough in other regions or time spans. The second disadvantage is potential computational instabilities [33] or unguaranteed convergence of the algorithm for nonlinear systems [3]. Therefore, we propose evolving GP models [23] to be used for adapting controller's GP model.

The basic idea of evolving GP models is that all influential parts of the GP model should be adapted on-line. The GP model is fully determined by the training data and the covariance function. In the case of sparse approximation which we use, the training data is actually represented by the active set. Usually, the training data, especially in case of dynamic systems, has more than one input dimensions, so-called regressors, which have various influence on output data. Moreover, some regressors may be fallacious and can present additional noise to model. Thus, selection of regressors is important part of modelling as well. As described in Section 2, with selection of appropriate type or a combination of various types of covariance function a prior knowledge of the system is included in the model. Nevertheless, with optimization of hyperparameter values the model is even more adjusted to real system. So, there are four parts that can evolve:

– regressors,

– active set,
– type of covariance function and
– hyperparameter values.

Although the proposed concept considers all four parts that can evolve, our implementation is somewhat more facile. In dynamic nonlinear systems, where the nonlinear mapping between input and output can not be easily formulated, frequently the squared exponential covariance function is used presuming smoothness and stationarity of the system. That means the covariance function is fixed and does not need to evolve. Furthermore, the squared exponential covariance function can be used with automatic relevance determination (ARD) which is able to find influential regressors [28]. With the optimization of the hyperparameter values, uninfluential regressors have smaller values and as a consequence have smaller influence to the result. Therefore, all available regressors can be used. Consequently, only the active set and hyperparameter values have left to be adapted sequentially.

With every incoming data, first its information gain regarding the current GP model is estimated. This is done in two phases. In the first phase a prediction of the current input data is calculated based on the current GP model. If the difference between mean value of prediction and current data output is small enough, it means that the current GP model can accurately predict the output of the incoming data. Furthermore, if also the variance of the prediction is small, the current data most probably does not contain any new information regarding the current GP model, thus there is no need to include current data into the GP model. Otherwise, we include current data into the GP model. If this inclusion causes the excess of the pre-set maximal active set size, the less informative data in the active set should be removed. The less informative data is found by calculating the negative log-likelihood from Eq. (4) for all subsets of the active set of length $m = n-1$, where m is the pre-set maximal size of the active set and n is the size of the exceeded active set[3]. The subset with the lowest negative log-likelihood is preserved and the remaining data is removed. After every update of the active set, the hyperparameter values are optimised by minimising negative log-likelihood. This can be done with any suitable optimisation method. In our case the conjugate gradients optimisation method is used.

The described on-line learning method is, besides the selected control algorithm, the main difference between the control we propose and PILCO method [5]. Due to the nature of the policy search methods, PILCO is implemented as batch learning method. That means the model is updated at the end of every cycle, during which new measurements are collected. As the number of collected measurements is usually high enough that a sequential learning is not convenient,

[3] It should be noted that for calculation of the negative log-likelihood the inverse of the covariance matrix is needed. In our case the inverse of the covariance matrix should be calculated for every subset. But, this computational demanding task can be speed-up by calculating the Cholesky decomposition of the exceeded active set of length n only once and then downdates it for every data in it by using low rank updates [32].

but rather off-line learning is used, since it can consider all recently collected data. By default, a full GP model is used, but in case the number of data points exceeds a particular threshold a Sparse Pseudo-input Gaussian process [34] can be used.

An illustrative example that shows operation of minimum variance control based on evolving GP model is given in the following section.

5 Illustrative example

To assess the proposed controller we used the nonlinear dynamic system [20] described by

$$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k) + \nu, \tag{13}$$

where u is system's input signal, y is system's output signal, ν is white noise of normal distribution with standard deviation 0.005 that contaminates the system response and the sampling time is one second. The nonlinearity in the region of interest for the benchmark system is depicted by a gray mesh in Fig. 3.

The requirement of the closed-loop control is to follow the set-point, depicted in Fig. 4, as close as possible. We start off with an empty GP model's active set and with some default hyperparameter values $\log\theta = [0; 0; 1; -1]$, which are quite different comparing to the optimal ones. The set-point signal is a combination of periodic pulses in three various regions. The first region is between 0.5 and 1.5, the second one between 3 and 4, and the last one between -0.5 and -1.5. The priority of such a signal is to show that the proposed approach for control system based on evolving GP models is able to learn from scratch, without any prior model, and to update regarding the dynamics changes. The data stream contains only 388 data points (shown in Fig. 4), which serves the demonstration requirements. We pre-set the maximal active size to 50 data points. The used control cost function is variation of minimum variance cost function from Eq. (11)

$$J(k) = [r(k) - \{y(k-1) - E(\hat{y}(k-1))\} - E(\hat{y}(k))]^2. \tag{14}$$

The term $\{y(k-1) - E(\hat{y}(k-1))\}$ is to make the control algorithm insensitive to errors in the steady-state gain with subtracting the discrepancy between the latest plant output and the latest model most likely output.

As the controller has no prior knowledge about the system, the system's output oscillates at the beginning, Fig. 4. Nevertheless, the controller observes enough data to successfully, but with some overshoot, follow the first step. Afterwards, the controller easily follows the set-point signal, even in the second and the third region, where the nonlinearity is locally different. The complete nonlinearity, including all three regions, can be seen in Fig. 3, where orange pluses, green crosses and blue stars denote first, second and third region respectively. However, at the beginning of these regions also some overshoots appear, but the output quickly gets settled.

The final active set is depicted in Fig. 3. The most informative data points, selected from in-streaming data with the proposed evolving method, are denoted

with red circles. It is clear that the selected data points are evenly distributed through all nonlinear space, which indicates that the proposed method successfully adapts the GP model according to operating regions. The times when the GP model is updated are depicted in Fig. 5 as dashed light blue lines. It can be seen that most updates occur, as expected, at changes of the set-point signal, especially at changes in dynamics. Similarly holds for hyperparameter values, whose traces through the process are also shown in Fig. 5, denoted as coloured solid lines. It can be seen that hyperparameter values are mostly changing in a region of the first three steps, when most new information about the system is obtained. Once near-optimal values are reached, hyperparameter values are changing in a much smaller scale.

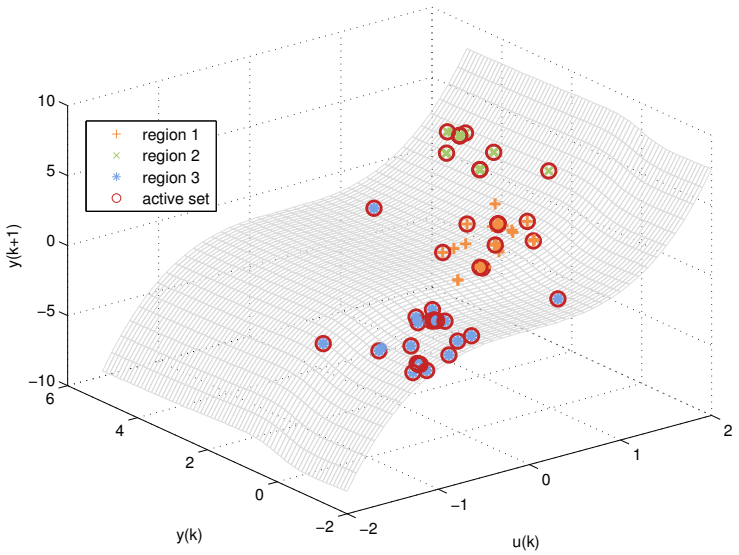

Fig. 3: Observed data and the most informative data - active set showed on surface of selected nonlinear system. The gray mesh denotes the nonlinear mapping of the system to be controlled, orange pluses, green crosses and blue stars denote first, second and third region respectively, while red circles denote the active set.

The main purpose of this implementation of the closed-loop control is the model adaptation according to system's dynamics. Therefore, once enough data about the system is obtained, the controller easily follows the set-point signal and adapts the GP model. But the controller can be further improved to somehow explore unknown space, especially at the beginning of the process or in any other cases when it is not possible to follow the set-point signal due to the lack

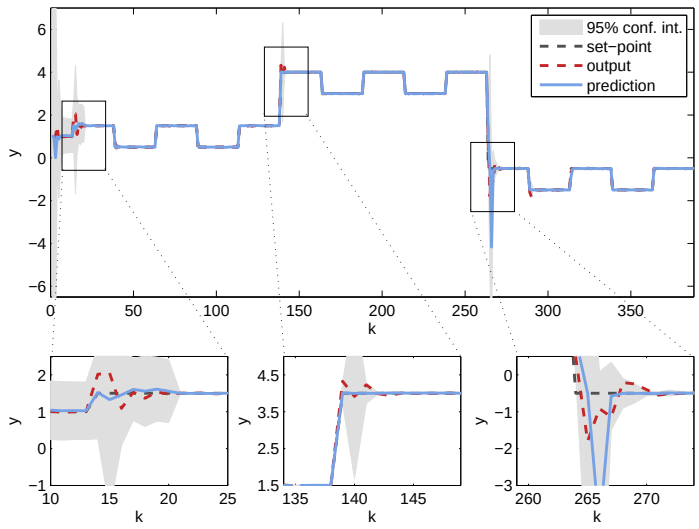

Fig. 4: Simulation of controller based on evolving GP model. The black dashed line denotes the set-point signal and the red dashed line denotes the output of the system, while the blue solid line denotes a mean value of the prediction and a gray band denotes the 95% prediction confidence interval based on the current GP model.

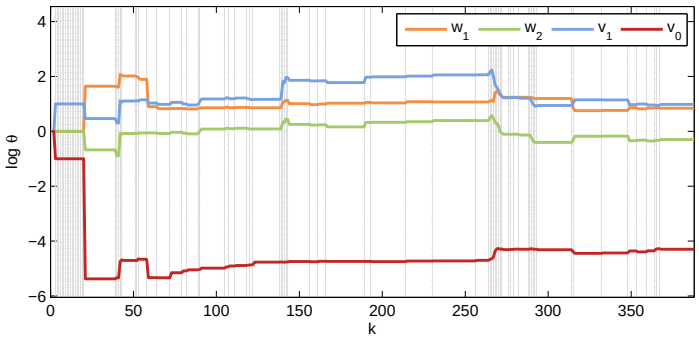

Fig. 5: Traces of the hyperparameter values changing through time and the active set updates. Coloured solid lines denote hyperparameter values and dashed gray lines denote time instants when the GP model was updated with new data.

of information about the dynamics. With such an improvement the controller is able to follow almost arbitrary changes in process dynamics.

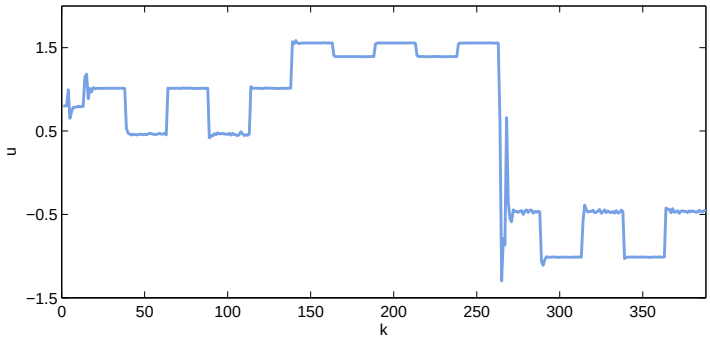

Fig. 6: Optimal control signal $u(k)$ based on the minimum variance controller using GP model's predictions.

6 Conclusions

In this chapter an adaptive closed-loop control based on evolving GP models is described. Evolving GP model is the model where not only hyperparameters, but also the content of covariance matrix is changing with new in-streaming data that have enough information content in comparison with the previously contained data. This way the model keeps-up with the dynamics which changes with the change of operating region.

The evolving GP model is upgraded with minimum variance controller that completes the proposed adaptive closed-loop system.

The illustrative example shows the satisfactory performance of the adaptive control with no initial knowledge and with a rapid change of operating region during its operation.

The proposed control algorithm which, we believe, is an improvement over similar and previously proposed algorithms is meant as a step forward in control of nonlinear and time-variable dynamic systems with possible presence of uncertainties and disturbances. Potential applications of the proposed control are foreseen in fields of rehabilitation engineering and biotechnology, robotics, process and power engineering and elsewhere where nonlinear and time-variable dynamic systems can be found.

Acknowledgement

This work has been supported by the Slovenian Research Agency, grant No. P2-0001

References

1. Ažman, K., Kocijan, J.: Application of Gaussian processes for black-box modelling of biosystems. ISA Transactions 46, 443–457 (2007)

2. Ažman, K., Kocijan, J.: Fixed-structure Gaussian process model. International Journal of Systems Science 40(12), 1253–1262 (2009)
3. Cornford, D., Csato, L., Opper, M.: Sequential, Sparse Learning in Gaussian Processes. In: Proceedings of the 7th International Conference on GeoComputation. vol. 44. Southampton, UK (2003)
4. Csató, L., Opper, M.: Sparse online Gaussian processes. Neural computation 14(3), 641–668 (2002)
5. Deisenroth, M.P.: Efficient Reinforcement Learning using Gaussian Processes. Ph.D. thesis, Karlsruhe Institute of Technology, Karlsruhe (2010)
6. Deisenroth, M.P., Rasmussen, C.E.: PILCO: A model-based and data-efficient approach to policy search. In: Proceedings of the 28th International Conference on Machine Learning (ICML 2011). Bellevue, WA (2011)
7. Deisenroth, M.P., Rasmussen, C.E., Fox, D.: Learning to control a low-cost manipulator using data-efficient reinforcement learning. In: Proceedings of the International Conference on Robotics: Science & Systems (R:SS 2011). Los Angeles, CA (2011)
8. Deisenroth, M.P., Rasmussen, C.E., Peters, J.: Gaussian process dynamic programming. Neurocomputing 72(7–9), 1508–1524 (2009)
9. Deisenroth, M., Peters, J., Rasmussen, C.: Approximate dynamic programming with Gaussian processes. In: Proceedings of American Control Conference (ACC). pp. 4480–4485. Seattle, WA (2008)
10. Deisenroth, M., Rasmussen, C.: Bayesian Inference for Efficient Learning in Control. In: Proceedings of Multidisciplinary Symposium on Reinforcement Learning (MSRL). Montreal, Canada (2009)
11. Filatov, N., Unbehauen, H.: Survey of adaptive dual control methods. IEE Proceedings - Control theory and applications 147(1), 119–128 (2000)
12. Isermann, R., Lachman, K.H., Matko, D.: Adaptive Control Systems. Systems and Control Engineering, Prentice Hall International (1992)
13. Kocijan, J.: Control algorithms based on Gaussian process models: A state-of-the-art survey. In: Kolemisevska-Gugulovska, T.D., Stankovski, M.J. (eds.) Special International Conference on Complex systems: Synergy of Control, Communications and Computing - Proceedings of COSY 2011 Papers, September 16-20, 2011, Ohrid, Macedonia. pp. 69–80. The Society for Electronics, Telecommunications, Automation, and Informatics of Macedonia, Skopje, Macedonia (September 2011)
14. Kocijan, J.: Dynamic GP models: an overview and recent developments. In: Recent Researches in Applied Mathematics and Economics: proceedings of the 6th International Conference on Applied Mathematics, Simulation, Modelling, (ASM'12). pp. 38–43. Vougliameni, Greece (2012)
15. Kocijan, J., Girard, A., Banko, B., Murray-Smith, R.: Dynamic systems identification with Gaussian processes. Mathematical and Computer Modelling of Dynamic Systems 11(4), 411–424 (2005)
16. Kocijan, J., Likar, B.: Gas-liquid separator modelling and simulation with Gaussian-process models. Simulation Modelling Practice and Theory 16(8), 910–922 (2008)
17. Lázaro-Gredilla, M., Quiñonero Candela, J., Rasmussen, C.E., Figueiras-Vidal, A.R.: Sparse spectrum Gaussian process regression. The Journal of Machine Learning Research 11, 1865–1881 (2010)
18. Murray-Smith, R., Sbarbaro, D., Rasmussen, C., Girard, A.: Adaptive, cautious, predictive control with Gaussian process priors. In: Proceedings of 13th IFAC Symposium on System Identification. Rotterdam, Netherlands (2003)

19. Murray-Smith, R., Sbarbaro, D.: Nonlinear adaptive control using nonparametric Gaussian process prior models. In: Proceedings of IFAC 15th World Congress. Barcelona (2002)
20. Narendra, K., Parthasarathy, K.: Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Networks 1(1), 4–27 (1990)
21. Nguyen-Tuong, D., Peters, J.: Learning robot dynamics for computed torque control using local Gaussian processes regression. In: Symposium on Learning and Adaptive Behaviors for Robotic Systems. pp. 59–64 (2008)
22. Nguyen-Tuong, D., Seeger, M., Peters, J.: Real-time local GP model learning, vol. 264, chap. From Motor Learning to Interaction Learning in Robots, pp. 193–207. Springer (2010)
23. Petelin, D., Grancharova, A., Kocijan, J.: Evolving Gaussian process models for prediction of ozone concentration in the air. Simulation Modelling Practice and Theory 33, 68–80 (2013)
24. Petelin, D., Kocijan, J.: Control system with evolving Gaussian process model. In: Proceedings of IEEE Symposium Series on Computational Intelligence, SSCI 2011. IEEE, Paris (2011)
25. Quinonero-Candela, J., Rasmussen, C.E.: A Unifying View of Sparse Approximate Gaussian Process Regression. Journal of Machine Learning Research 6, 1939–1959 (2005)
26. Rasmussen, C.E., Deisenroth, M.P.: Probabilistic inference for fast learning in control. In: Recent Advances in Reinforcement Learning, Lecture Notes on Computer Science, vol. 5323, pp. 229–242. Springer (2008)
27. Rasmussen, C.E., Kuss, M.: Gaussian processes in reinforcement learning. In: Thurn, S., Saul, L., Schoelkopf, B. (eds.) Advances in Neural Information Processing Systems conference. vol. 16, pp. 751–759. MIT Press (2004)
28. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge, MA (2006)
29. Sbarbaro, D., Murray-smith, R.: An adaptive nonparametric controller for a class of nonminimum phase non-linear system. In: Proceedings of IFAC 16th World Congress. Prague, Czech Republic (2005)
30. Sbarbaro, D., Murray-Smith, R., Valdes, A.: Multivariable generalized minimum variance control based on artificial neural networks and Gaussian process models. In: International Symposium on Neural Networks. Springer (2004)
31. Sbarbaro, D., Murray-Smith, R.: Self-tuning control of nonlinear systems using Gaussian process prior models. In: Murray-Smith, R., Shorten, R. (eds.) Switching and Learning in Feedback Systems, Lecture Notes in Computer Science, vol. 3355, pp. 140–157. Springer, Heidelberg (2005)
32. Seeger, M.: Low Rank Updates for the Cholesky Decomposition. Tech. rep., University of California at Berkeley (2008)
33. Seeger, M., Williams, C.K.I., Lawrence, N.D.: Fast Forward Selection to Speed Up Sparse Gaussian Process Regression. In: Ninth International Workshop on Artificial Intelligence and Statistics. Society for Artificial Intelligence and Statistics (2003)
34. Snelson, E., Ghahramani, Z.: Sparse Gaussian Processes using Pseudo-inputs. In: Neural Information Processing Systems (2005)
35. Wittenmark, B.: Adaptive dual control. In: Control Systems, Robotics and Automation, Encyclopedia of Life Support Systems (EOLSS), Developed under the auspices of the UNESCO. Eolss Publishers, Oxford, UK (Jan 2002)