Sparse Kernel Learning-based Nonlinear Predictive Controllers

Yi Liu[1] and Wei Wang[2]

[1]Institute of Process Equipment and Control Engineering, Zhejiang University of Technology, Hangzhou 310032, Zhejiang, China

[2]Institute of Industrial Process Control, Zhejiang University, Hangzhou 310027, Zhejiang, China
Email Address: wangwei2@iipc.zju.edu.cn

Received Oct 5, 2010; Revised Feb 22, 2011

Abstract: Both of deterministic and probabilistic sparse kernel learning (SKL) methods are effective for process modeling. Based on the SKL model with a polynomial kernel, a simple nonlinear control strategy is investigated. First, an SKL identification model is obtained using a polynomial kernel. Then, a predictive control performance index, which is characterized as an even-degree polynomial function of the manipulated input, is formulated. Consequently, the optimal manipulated input can be efficiently obtained by solving a simple root problem of an odd-degree polynomial equation because of its special structure. A comparative study on a benchmark problem shows its superiority to traditional controllers. Also, some attributes of the proposed control strategy can result in a practicable solution for real-time control.

Keywords: Sparse kernel learning, nonlinear predictive control, polynomial kernel.

1 Introduction

The sparse kernel learning (SKL) is a novel nonlinear modeling method originally proposed in the machine learning area [1-2]. Support vector regression (SVR) and relevance vector regression (RVR) represent deterministic and probabilistic modeling methods, respectively. Both of two SKL methods can make predictions based on a linear combination of kernel functions, which are defined on a subset of the training samples called the support vectors or relevance vectors. The sparse kernel representation can avoid the overfitting problem and increase the generalization ability of learning machine [1-2]. Therefore, SKL has attracted more attention with the development of SVR and RVR, and has found increasing applications for process modeling [3-4].

Recently, some SVR model based nonlinear control algorithms have been proposed [5-7]. However, these new nonlinear control algorithms confront with some technical difficulties, e.g., heavy computation load and large memory requirement. Besides, the SVR models adopt the Gaussian kernel function. Thus, the explicit formulation of control

This work was supported by the National Natural Science Foundation of China (No. 61004136) and Zhejiang Provincial Natural Science Foundation of China (No. Y4100457). All correspondence of this paper should be addressed to Dr. Yi Liu.

law cannot be obtained straightforward. Due to this reason, process control engineers may not understand why these new control schemes can work well and prone to reject to use them. On the other hand, for control applications, it is desirable to keep the control scheme as simple as possible for the real-time implementation [8].

On the other hand, compared to the widely used SVR model, to our best knowledge, the RVR-based control methods have rarely been reported. Therefore, this paper aims to discuss and extend a general methodology to design some simple controllers, mainly based on a recent proposed control method [8], for nonlinear processes. Both of deterministic and probabilistic SKL modeling methods are adopted to design simple nonlinear predictive controllers.

2 The SKL-OPC framework and its Control Strategies

Many single-input-single-output nonlinear processes can be described by the following discrete-time model [3]:

$$\begin{aligned} & y(k+1)=f\left[\mathbf{x}(k)\right]=f\left[\mathbf{Y}(k),u(k),\mathbf{U}(k-1)\right] \\ & \text{where } \mathbf{Y}(k)=\left[y(k),\ldots,y(k-n_y+1)\right],\ \mathbf{U}(k-1)=\left[u(k-1),\ldots,u(k-n_u+1)\right] \end{aligned} \quad (2.1)$$

where k is the discrete time index, and $f(\bullet)$ is a general nonlinear function; $y(k)$ and $u(k)$ represent the controlled output and the manipulated input, respectively; and n_y and n_u denote the process orders, respectively.

Applying the SKL identification framework for regression problems to the data set $S_N = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\} \subset \mathbf{X} \times \mathrm{R}$, the prediction $y_m(k+1)$ can be formulated as [1-2]:

$$y_m(k+1)=\mathrm{SKL}\left[\mathbf{x}(k),\ \boldsymbol{\alpha}\right]=\sum_{i=1}^{n}\alpha_i K\left\langle \mathbf{x}(i),\ \mathbf{x}(k)\right\rangle+b \quad (2.2)$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]^{\mathrm{T}}$ are the n-dimension weight vector, and n (always much less than N) is the number of subset of the training samples, called support vectors or relevance vectors. The item <•, •> denotes the dot product and $K\langle\mathbf{x}(i), \mathbf{x}(k)\rangle$ is a kernel function which handles the inner product in the feature space and therefore the explicit form of nonlinear mapping does not need to be known. Note that the bias term b of the regression model is not always necessary [1-2].

Let $y_r(k)$ be the desired output at time k. Then, the control law can be obtained by minimizing the one-step-ahead weighted predictive control performance index (CPI) [8]

$$\begin{aligned} & \min\ J\left[u(k)\right]=\left[E(k+1)\right]^2+\lambda\left[u(k)-u(k-1)\right]^2 \\ & \text{where } E(k+1)=y_r(k+1)-y_m(k+1)-e(k) \end{aligned} \quad (2.3)$$

where $E(k+1)$ is the total error of SKL predictive model at time k, and $\lambda > 0$ is the control effort weighting factor, representing a penalty factor associated with the manipulated variable increment. The intention is to obtain the SKL identification model and then to achieve the manipulated input $u(k)$ by minimizing the CPI. This is the basic idea and general framework of SKL-OPC [8].

Two traditional methods to design the simple controller have been proposed in the literature [8]. One adopted an approximate gradient descent (GD) method to iteratively optimize the CPI and then get the solution. The other utilized the Taylor linearization (TL) method with respect to *u*(*k*) at the point *u*(*k*-1) to obtain an approximate control law. They are noted GD and TL methods, respectively. These two approaches can be adopted under the SKL-OPC framework to develop their corresponding kernel control strategies [8]. However, both the two control strategies can be regarded as indirect methods (i.e., using some kind of approximation to avoid numerical difficulty and obtain the control strategy). To overcome this shortcoming, a direct method to design the simple control strategy also under the SKL-OPC framework is formulated below.

Applying the polynomial kernel function, the prediction can be formulated below [8]

$$\begin{aligned} y_m(k+1) &= \sum_{i=1}^{n} \alpha_i \left(\left\langle \mathbf{x}(i), \mathbf{x}(k) \right\rangle + 1 \right)^p + b \\ &= \sum_{j=0}^{p} C_p^j \sum_{i=1}^{n} \alpha_i \left[x_{n_y+1}(i) \right]^{p-j} \left[\left\langle \bar{\mathbf{x}}(i), \bar{\mathbf{x}}(k) \right\rangle + 1 \right]^j \left[u(k) \right]^{p-j} + b \end{aligned} \tag{2.4}$$

$$\text{where } \bar{\mathbf{x}}(k) = \left[\mathbf{Y}(k), \mathbf{U}(k-1) \right],\; C_p^j = \frac{p!}{j!(p-j)!}$$

where the integer *p* denotes the polynomial degree. For simplicity, some terms in equation (2.4) are denoted as [8]

$$\begin{cases} \beta_j = C_p^j \sum_{i=1}^{N_{\mathrm{SV}}} \alpha_i \left[x_{n_y+1}(i) \right]^{p-j} \left[\left\langle \bar{\mathbf{x}}(i), \bar{\mathbf{x}}(k) \right\rangle + 1 \right]^j,\; 0 \le j \le p-1 \\ \beta_p = \sum_{i=1}^{N_{\mathrm{SV}}} \alpha_i \left[\left\langle \bar{\mathbf{x}}(i), \bar{\mathbf{x}}(k) \right\rangle + 1 \right]^p + b + he(k) - y_r(k+1) \end{cases} \tag{2.5}$$

Consequently, the CPI at time *k* is formulated as follows [8]

$$J\left[u(k)\right] = \left\{ \sum_{j=0}^{p} \beta_j \left[u(k) \right]^{p-j} \right\}^2 + \lambda \left[u(k) - u(k-1) \right]^2 \tag{2.6}$$

This CPI is an even degree polynomial function of the input, thus the optimal solution is derived [8].

$$\partial J\left[u(k)\right] / \partial u(k) = \sum_{l=0}^{2p-1} A_l \left[u(k) \right]^{2p-1-l} = 0$$

$$\text{where } \begin{cases} A_l = \sum_{j+m=l} \beta_j \beta_m (p-m),\; l = 0, \cdots, 2p-3 \\ A_{2p-2} = \sum_{j+m=2p-2} \beta_j \beta_m (p-m) + \lambda \\ A_{2p-1} = \sum_{j+m=2p-1} \beta_j \beta_m (p-m) - \lambda u(k-1) \end{cases} \tag{2.7}$$

The control strategy adopts the polynomial kernel function and utilizes the rooting method to obtain the optimal process input $u(k)$, thus the controller here is referred to as SKL-PKR for abbreviation. When an SVR model and a RVR model are adopted, the related controllers are named SVR-PKR and RVR-PKR, respectively.

Note that sparseness is generally regarded as a good feature for the learning machine. Using a sparse model, the predictions for new samples depend only on the kernel function evaluated with a subset of the training samples. This means that the memory requirement is small and, correspondingly, that the computation time on the test samples is short [2]. As a result, a controller based on the SKL model will be easier to implement.

Generally, SVR and sparse least-squares SVR (SLSSVR) can both be described as being in equation (2.2). The RVR method, as an alternative SKL algorithm with Bayesian technique, shares the similar function form as SVR and thus can also be formulated in this form. Moreover, RVR is implemented through a probabilistic manner and has no limitation of the kernel function form [2]. Recently, it has been reported that the sparseness of RVR is much better than that of SVR and SLSSVR, while the prediction performance of these learning models are comparative [2]. Thus, these SKL algorithms are utilized to identify the process model in this paper.

Both of SVR and SLSSVR are deterministic SKL modeling methods. In other words, they are not probabilistic. Based on the statistical viewpoint, all statistical inferences and decisions should be made through the probabilistic manner. For this reason, it is necessary to express the uncertain characteristic of the process data in both modeling and prediction steps [2]. It should be better to adopt the RVR method to identify a process when its measurement noise is significant. Especially, the RVR model can be developed offline and only to be used for online prediction because it has involved a high computational complexity [2,4]. Therefore, this paper adopts the RVR model to extend the application of SKL-PKR controllers.

3 Comparison Study of a Benchmark Problem

In this section, a benchmark example is illustrated to demonstrate the performance of the proposed SKL-PKR controllers, using both of deterministic and probabilistic SKL modeling methods, and meanwhile to verify the simplicity and advantage. The simulation environment is Matlab V7.1 with CPU main frequency 1.7 GHz and 512 M memory. The benchmark example is a spring-mass-damper system with a hardening spring [9].

$$\frac{dy}{dt}+\frac{d^2y}{dt^2}+y+y^3=u \tag{3.1}$$

First, the deterministic environment is investigated. A sequence of 100 samples is used to train an SVR model and an RVR model, respectively. The output and input vectors are chosen as $y(k+1)$ and $\mathbf{x}(k) = [y(k), y(k\text{-}1), u(k), u(k\text{-}1)]$, respectively [9]. An offline SVR identification model is obtained using the cross-validation approach [1]. And an RVR model can be also obtained using some optimal methods [2,4]. It takes less than 30 minutes to obtain the SVR and RVR models, respectively. Compared to the other data-driven modeling methods, the identification procedure is easier to implement.

To provide a suitable comparison with traditional techniques, a well-tuned proportional-integral-derivative (PID) controller and a neural network-generalized predictive control (NN-GPC) are also adopted here. The design parameters of nonlinear GPC strategy selected is specified in the following index [9].

$$J\left[u(k)\right]=\sum_{i=N_1}^{N_2}\left[y_r(k+i)-y(k+i)\right]^2+\lambda\sum_{i=1}^{N_u}\left[u(k+i)-u(k+i-1)\right]^2 \tag{3.2}$$

The performance comparison of all controllers, including SVR-PKR, PID and NN-GPC, is shown in figure 3.1 (The response of SVR-PKR and RVR-PKR are similar and thus only the former is plotted.). The integral absolute error (IAE) of setpoint tracking, listed in table 3.1, is used to quantify the performance of controllers. Both of the SVR-PKR and RVR-PKR controllers have good tracking performance and little overshoot. The PID controller is slower for tracking; on the other hand, the NN-GPC strategy has larger overshoot. NN-GPC has better instantaneous response than SVR-PKR because of its multi-step-ahead prediction and optimization properties. Therefore, it is meaningful to develop efficient algorithms for extending SVR-PKR and RVR-PKR to multi-step-ahead prediction to achieve better performance.

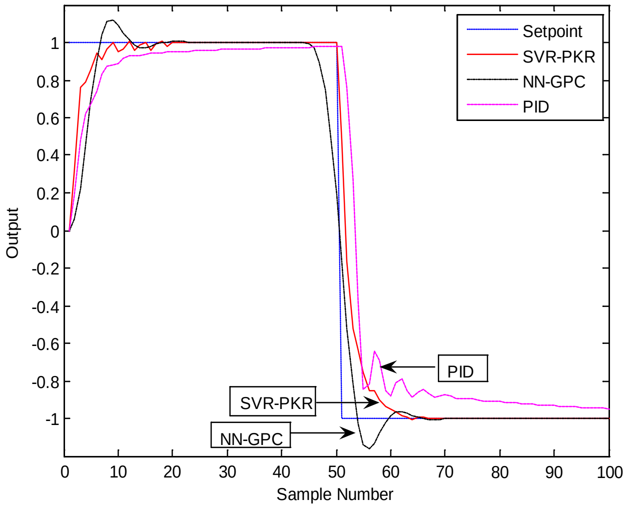

Figure 3.1: Set-points tracking comparison with traditional controllers

Among controllers, NN-GPC is the most time-consuming, and the computation load of PID is the smallest. The whole procedure of 100 steps by SVR-PKR just takes 3.8 s (i.e., about 0.04 s for one step), which is much less than the sampling period of this system (i.e., 0.2 s). The computation time of RVR-PKR is less than SVR-PKR since its model is more sparse.

Actually, almost all process measurements are essentially random variables because they are always contaminated by random white noises. Therefore, the following part aims to compare the performance of RVR-PKR and SVR-PKR controllers in the noise environment. To simulate the scenario, the measured process input and output of the same sequence are supposed to be corrupted by Gaussian noise. The average simulation results show that the RVR method can obtain a more sparse model with better identification performance than SVR in the noise environment.

Table 3.1. Performance comparison of RVR-PKR and SVR-PKR controllers in the deterministic and noise scenarios

Scenarios	Control strategies	IAE	Time, s
No noise	RVR-PKR	6.6	3.1
	SVR-PKR	6.7	3.8
	PID	16.1	1.5
	NN-GPC	8.2	8.2
Noise	RVR-PKR	9.8	3.2
	SVR-PKR	10.2	4.7

Details about the IAE comparison of both controllers are tabulated as table 3.1. All the simulation results are the average of 20 running times. Due to the probabilistic scenario, the related parameters of both controllers are changed each simulation and not listed here. The results show that the RVR-PKR controller can achieve a litter better tracking performance than SVR-PKR. Moreover, the computation time required by RVR-PKR for 100 control steps is much less than SVR-PKR. This is very important for real-time control, especially for those processes required fast control response.

4 Conclusions

A simple nonlinear control framework named as SKL-OPC is proposed and different strategies are developed. The SVR-PKR and RVR-PKR control strategies are two natural extensions of the SKL-OPC framework, under deterministic and probabilistic scenarios. Based on the comparison results of a benchmark problem, the tracking performance and the aspect on practical implement are evaluated. Therefore, these two simple SKL-PKR controllers are alternative for nonlinear processes.

References

[1] J. S. Taylor and N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, Cambridge, UK, 2004.

[2] C. M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, New York, 2006.

[3] Y. Liu, N.P. Hu, H.Q. Wang, and P. Li, Soft chemical analyzer development using adaptive least-squares support vector regression with selective pruning and variable moving window

size, *Industrial & Engineering Chemistry Research* **48** (2009), 5731–5741.

[4] Z.Q. Ge and Z.H. Song, Nonlinear soft sensor development based on relevance vector machine, *Industrial & Engineering Chemistry Research* **49** (2010), 8685–8693.

[5] W.M. Zhong, G.L. He, D.Y. Pi and Y.X. Sun, SVM with polynomial kernel function based nonlinear model one-step-ahead predictive control, *Chinese Journal of Chemical Engineering* **13** (2005), 373–379.

[6] S. Iplikci, Support vector machines-based generalized predictive control, *International Journal of Robust Nonlinear Control* **16** (2006), 843–862.

[7] R.D. Zhang and S.Q. Wang, Support vector machine based predictive functional control design for output temperature of coking furnace, *Journal of Process Control* **18** (2008), 439–448.

[8] Y. Liu, Y.C. Gao, Z.L. Gao, H.Q. Wang, and P. Li, Simple nonlinear predictive control strategy for chemical processes using sparse kernel learning with polynomial form, *Industrial & Engineering Chemistry Research* **49** (2010), 8209–8218.

[9] M. Nørgaard, O. Ravn, N. K. Poulsen and L. K. Hansen, Neural Networks for Modelling and Control of Dynamic Systems, Springer-Verlag, London, 2000.

Yi Liu obtained his Ph.D. in control science and engineering at Zhejiang University in June 2009. He is now an assistant professor at Zhejiang University of Technology. His research interests are in the fields of kernel learning adaptive modeling & control algorithms with applications to chemical processes.

Wei Wang is now a Ph.D. candidate in control science and engineering at Zhejiang University. His research interests are in the fields of data-based process control algorithms with applications to chemical processes.