Comparative Study of Model-Free Predictive Control and Its Database Maintenance for Unstable Systems

Comparative Study of Model-Free Predictive Control and Its Database Maintenance for Unstable Systems

Herlambang SAPUTRA [*] and Shigeru YAMAMOTO [**]

Abstract: This study provides a comparison of three methods, i.e., standard locally weighted averaging (LWA), least-norm solutions, and ℓ_1-minimization, for model-free predictive control based on Just-In-Time modeling and database maintenance for an unstable system. In contrast to conventional model predictive control, the model-free predictive control method does not use any mathematical model; rather, it uses the past input/output data stored in a database. Although conventional stabilizing feedback is used to obtain the input/output data of an unstable system, model-free predictive control is assumed to be used without it. Three methods based on standard LWA, least-norm solutions, and ℓ_1-minimization are statistically compared using a simple model. The results show that the methods of least-norm solutions and ℓ_1-minimization are superior to that of LWA. The method by ℓ_1-minimization yields tracking errors smaller than that by least-norm solutions; however, the method by ℓ_1-minimization requires a long computational time. In addition, the effectiveness of a method of database maintenance is illustrated by numerical simulations.

Key Words: predictive control, data-driven control, Just-In-Time modeling, least-norm solution, ℓ_1-minimization.

1. Introduction

Model predictive control is widely used in industrial systems, including chemical processes [1]. It requires a mathematical model that precisely represents the dynamics of the controlled system to predict future system behavior and determine the control input needed to produce the desired control results. Hence, the key to model predictive control is to obtain a precise model and use many physical modeling techniques and/or useful system identification techniques. Generally, this model is not updated until a significant change occurs within the controlled system.

In contrast with standard model predictive control, model-free predictive control, as proposed by Stenman in 1999 [2], constantly updates the mathematical model based on the so-called Just-In-Time modeling. Just-In-Time modeling, originally proposed in a previous study [3], utilizes both online measured input/output data and data stored in a database to adaptively identify a local, not global, model [4],[5]. This method is also referred to as model on-demand [6],[7], lazy learning [8], or instance based learning [9]. The Just-In-Time technique is applied to prediction of the important parameters in production processes in the steel industry [10]–[13], PID parameter tuning [14],[15], and soft sensors in industrial chemical processes [16].

Inoue and Yamamoto [17] proposed another "model free" predictive control in a Just-In-Time modeling framework. In their method, an optimal control input is directly predicted through online current measured data and stored past data and not by any local models. As in the Just-In-Time modeling method, the neighbors of the current data are searched in the stored data and the predicted control input is derived as a weighted average of the neighbors. Although to select a weight for this weighted averaging, several methods can be used in the Just-In-Time modeling framework, control performance highly depends on the weights. Similar Just-In-Time control methods have been proposed in previous studies [18]–[20]. These studies also use nearest neighbor and locally weighted averaging (LWA) techniques, as in the Just-In-Time modeling method.

Recently, two approaches that substitute the conventional nearest neighbor and LWA techniques have been introduced [21],[22]. In a previous study [21], weights are calculated as a solution for a linear equation. In another previous study [22], weights are computed as a solution of an ℓ_1-minimization problem, which produces a sparse vector with a few nonzero elements. This type of ℓ_1-minimization is currently popular in signal processing community [23], because sparse solutions yield benefit. In control community, sparsity is also utilized in efficient data compression for control signals through rate-limited erasure channels in [24]. In [22], sparsity is utilized to find the nearest neighbor and the weights.

This study aims to compare three methods, i.e., the standard LWA method [17], least-norm solution [21], and ℓ_1-minimization [22], through their application to model-free predictive control of an unstable system. Stabilization by model-free predictive control remains an open problem. Asymptotic stabilization appears to be impossible, except in an ideal case where there is no noise and nonlinearity. The boundedness of all signals in the control system will only be guaranteed in practical applications. In this study, we statistically evaluate the effect of model-free predictive control via many trials. In the case of an unstable system, it is difficult to construct a rich database containing input/output data without feedback control. Hence, to construct a database, we assume that there exists simple feedback control that stabilizes the unstable system.

[*] Graduate School of Natural Science and Technology, Kanazawa University, Kakuma, Kanazawa, Ishikawa 920-1192, Japan
[**] Faculty of Electrical and Computer Engineering, Kanazawa University, Kakuma, Kanazawa, Ishikawa 920-1192, Japan
E-mail: herlambang_s@moccos.ec.t.kanazawa-u.ac.jp, shigeru@se.kanazawa-u.ac.jp
(Received May 8, 2015)
(Revised July 24, 2015)

In [18] and [25], data-driven control is used to improve stabilizing feedback control. In contrast to these, since our main aim is to show that model-free predictive control has the ability to stabilize unstable systems, when we use model-free predictive control, we do not use the stabilizing controller, unlike in [18] and [25]. In addition to comparing model-free predictive control methods, we investigate the effect of database maintenance. As a method of database maintenance, we propose that the least accessed data in the database should be replaced with the most current data, obtained online to prevent the size of the database from increasing.

2. Model-free Predictive Control

2.1 Basic Idea

We consider the discrete-time system as follows:

$$y(t) = f(\psi(t)) + \varepsilon(t) \tag{1}$$

$$\psi(t) = \left[y(t-1), \cdots, y(t-n), u(t-1), \cdots, u(t-m)\right]^T, \tag{2}$$

where $u \in \Re$ is the control input; $y \in \Re$ is the controlled output; $\psi \in \Re^q$ is the regression vector of the size $q = n + m$; and ε is independent and identically distributed noise. We assume that n and m are unknown together with the nonlinear function f. The control objective is to make the h_y-step output trajectory

$$\mathbf{y}_f(t) = \begin{bmatrix} y(t+1) \\ \vdots \\ y(t+h_y) \end{bmatrix} \tag{3}$$

track the desired reference trajectory

$$\mathbf{r}(t) = \begin{bmatrix} r(t+1) \\ \vdots \\ r(t+h_y) \end{bmatrix}. \tag{4}$$

In general, model predictive control uses a model of the controlled system to find an h_u-step future input sequence

$$\mathbf{u}_f(t) = \begin{bmatrix} u(t) \\ \vdots \\ u(t+h_u-1) \end{bmatrix} \tag{5}$$

to achieve the goal. To use model predictive control, we need to apply a system identification technique to obtain a model. In contrast, model-free predictive control does not require any mathematical models. Model-free predictive control, proposed by a previous study [17], utilizes collected past input/output data of the controlled system as N vectors

$$\mathbf{a}_i := \begin{bmatrix} \mathbf{y}_p(t_i) \\ \mathbf{y}_f(t_i) \\ \mathbf{u}_p(t_i) \end{bmatrix} \in \Re^d, i = 1, 2, \ldots, N, \tag{6}$$

$$\mathbf{c}_i := \mathbf{u}_f(t_i) \in \Re^{h_u}, i = 1, 2, \ldots, N, \tag{7}$$

where

$$d = n + h_y + m, \tag{8}$$

$$\mathbf{y}_p(t) = \begin{bmatrix} y(t-n+1) \\ \vdots \\ y(t) \end{bmatrix}, \tag{9}$$

$$\mathbf{u}_p(t) = \begin{bmatrix} u(t-m) \\ \vdots \\ u(t-1) \end{bmatrix}. \tag{10}$$

The underlying idea of model-free predictive control consists of performing two steps:

(i). selecting k nearest vectors $\mathbf{a}_{i_j}$ to a query vector

$$\mathbf{b} = \begin{bmatrix} \mathbf{y}_p(t) \\ \mathbf{r}(t) \\ \mathbf{u}_p(t) \end{bmatrix} \tag{11}$$

that contains the current situation $\mathbf{u}_p(t)$, $\mathbf{y}_p(t)$, and the desired trajectory for future output $\mathbf{r}(t)$;

(ii). generating expected future input sequence as LWA to use weights x_{i_j} as

$$\hat{\mathbf{u}}_f(t) = \begin{bmatrix} \hat{u}(t|t) \\ \vdots \\ \hat{u}(t+h_u-1|t) \end{bmatrix} \tag{12}$$

$$= \sum_{j=1}^{k} x_{i_j} \mathbf{u}_f(t_{i_j}) = \sum_{j=1}^{k} x_{i_j} \mathbf{c}_{i_j}. \tag{13}$$

In a previous study [17], the so-called Just-In-Time method [6] is utilized for performing the two steps. Basically, all vectors $\mathbf{a}_i$ are sorted according to the distance to $\mathbf{b}$ as

$$d(\mathbf{a}_{i_1}, \mathbf{b}) \leq \cdots \leq d(\mathbf{a}_{i_k}, \mathbf{b}) \leq \cdots \leq d(\mathbf{a}_{i_N}, \mathbf{b}). \tag{14}$$

In addition, the number k and weights x_{i_j} for $\mathbf{a}_{i_j}$ satisfying

$$x_{i_1} \geq x_{i_2} \geq \cdots \geq x_{i_k} \text{ and } \sum_{j=1}^{k} x_{i_j} = 1 \tag{15}$$

are determined, e.g., using LWA and the Akaike's final prediction error criterion. In a previous study [15], the distance based on the ℓ_1-norm

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{k} |x_i| \tag{16}$$

is defined as

$$d(\mathbf{a}, \mathbf{b}) = \left\| W^{-1}(\mathbf{a} - \mathbf{b}) \right\|_1, \tag{17}$$

$$W = \mathrm{diag}(w_1, \ldots, w_d), \tag{18}$$

where for the ith element of $\mathbf{a}_j$,

$$w_i = \max_{j=1,\ldots N} \mathbf{a}_{ji} - \min_{j=1,\ldots N} \mathbf{a}_{ji}. \tag{19}$$

Moreover, the weight is calculated as

$$\tilde{x}_i = \mathrm{tr}\left(I_d - W^{-1}(\mathbf{a}_i - \mathbf{b})(\mathbf{a}_i - \mathbf{b})^T W^{-1}\right) \tag{20}$$

$$x_i = \tilde{x}_i / \sum_{i}^{k} \tilde{x}_i. \tag{21}$$

Other methods used to select the nearest neighbors have been investigated in [11]. When we have exact information on the system, we can find appropriate neighbours and weights.

In [21], finding the weights x_{i_j} is reformulated as solving the linear equation

$$A\mathbf{x} = \mathbf{b}, \tag{22}$$

where

$$A = \begin{bmatrix} \mathbf{a}_{i_1} & \mathbf{a}_{i_2} & \cdots & \mathbf{a}_{i_k} \end{bmatrix} \in \Re^{d\times k}, \tag{23}$$

$$\mathbf{x} = \begin{bmatrix} x_{i_1} & x_{i_2} & \cdots & x_{i_k} \end{bmatrix}^T \in \Re^{k}. \tag{24}$$

When $d > k$, the solution is given by a least mean square solution as $\mathbf{x} = (A^T A)^{-1} A^T \mathbf{b}$. When $d < k$, the solution is given by the least norm (minimum norm) solution $\mathbf{x} = A^T (AA^T)^{-1}\mathbf{b}$ of

$$\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_2 . \tag{25}$$

The size of the solution $\mathbf{x}$ in (25) (i.e., the neighbor size k) can be extended to the size of database N by introducing

$$A = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \ldots & \mathbf{a}_N \end{bmatrix} \in \Re^{d\times N} \tag{26}$$

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \ldots & x_N \end{bmatrix}^T \in \Re^{N}. \tag{27}$$

as

$$\min_{x} \|A\mathbf{x} - \mathbf{b}\| \text{ subject to } \|\mathbf{x}\|_0 = k, \tag{28}$$

where

$$\|\mathbf{x}\|_0 = \text{card}\,\{x_i \mid x_i \neq 0\} \tag{29}$$

is the l_0 norm defined as the total number of non-zero elements in $\mathbf{x}$. Because of the l_0 norm constraint, (28) is a mixed-integer problem, which is generally difficult to solve in real time.

In [22], (28) is reformulated as an ℓ_1-minimization problem:

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ subject to } A\mathbf{x} - \mathbf{b} = 0. \tag{30}$$

By adopting the ℓ_1-minimization problem, it is not necessary to decide the neighbor size k. To solve the ℓ_1-minimization problem, several methods have been developed. In particular, there are a large number of ℓ_1-minimization algorithms [23] such as gradient projection, homotopy, iterative shrinkage-thresholding, proximal gradient, augmented Lagrange multiplier, and Dual Augmented Lagrange Multiplier (DALM) algorithms[1].

Remark 1 Just-In-Time algorithms generally cause long feedback delays. Hence, model-free predictive control is limited to slow dynamical systems.

2.2 Model-free Predictive Control Algorithm

The fundamental procedure is summarized as follows.

Initialization. Determine n, m, N, h_u, and h_y. Let the discrete-time be $t = 0$.

Step 1. Whenever $t \leq \max(n, m)$, repeat this step. Measure $y(t)$ and apply $u(t)$ with an appropriate value to the controlled system. Increment the discrete-time as $t \leftarrow t + 1$.

Step 2. From the given reference trajectory $\mathbf{r}(t)$, define a query vector (11).

Step 3. Perform one of the three methods given below.

[1] MATLAB solvers are available at http://www.eecs.berkeley.edu /˜yang/software/l1benchmark/l1benchmark.zip

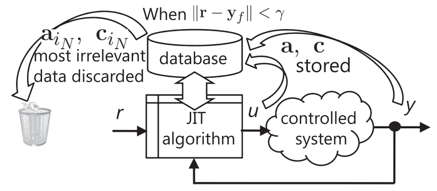
When $\|\mathbf{r} - \mathbf{y}_f\| < \gamma$
$\mathbf{a}_{i_N}$, $\mathbf{c}_{i_N}$
most irrelevant data discarded
database
$\mathbf{a}$, $\mathbf{c}$ stored
r
JIT algorithm
u
controlled system
y

Fig. 1 Database maintenance in model-free predictive control.

Step 3a (by LWA). Using the sorted vectors $\mathbf{a}_{i_1} \ldots, \mathbf{a}_{i_k}$ satisfying (14) and LWA in the Just-In-Time algorithm [6], determine the number k and weights $x_{i_1}, \ldots, x_{i_k}$ as (15).

Step 3b (by least-norm solution). For the sorted vectors $\mathbf{a}_{i_1} \ldots, \mathbf{a}_{i_k}$ satisfying (14), determine weights $x_{i_1}, \ldots, x_{i_k}$ by solving (22).

Step 3c (by ℓ_1-minimization). Using all vectors $\mathbf{a}_1 \ldots, \mathbf{a}_N$, solve the ℓ_1-minimization problem (30), and determine k and index $i_1, \ldots, i_k$ according to

$$|x_{i_1}| \geq \cdots \geq |x_{i_k}| \geq \cdots \geq |x_{i_N}|. \tag{31}$$

Step 4. The expected future input sequence is calculated by (12).

Step 5. Apply the first element $\hat{u}(t|t)$ of $\hat{\mathbf{u}}_f(t)$ to the system as $u(t)$. Increment the discrete-time as $t \leftarrow t+1$, and return to Step 2.

Remark 2 In this paper, to compare the ability of model-free predictive control to stabilize unstable systems, only when we first construct a database that stores the input/output (training) data of the given unstable system, we use a standard feedback control, rather than model-free predictive control. Hence, in Section 3, to obtain the the input/output (training) data, we used

$$u(t) = K(z)(r(t) - y(t)) + v(t) \tag{32}$$

with a controller $K(z)$ and a persisting exciting signal v to the system. Here, $K(z)$ is the discrete transfer function and "z" means the forward shift operator as $y(t) = zy(t-1)$ in the time domain.

2.3 Database Maintenance

We often see the poor control results by model-free predictive control when we control an unstable system because the database does not include enough rich data. To resolve this problem, we store the latest data obtained from real-time online control data in the database as shown in Fig. 1. To prevent the size of the database from increasing, we discard the least relevant data from the database. For example, in Step 5, at time t the most irrelevant data $\mathbf{a}_{i_N}$ and $\mathbf{c}_{i_N}$ in the database are replaced with

$$\begin{bmatrix} \mathbf{y}_p(t-h) \\ \mathbf{y}_f(t-h) \\ \mathbf{u}_p(t-h) \end{bmatrix} \text{ and } \mathbf{u}_f(t-h) \tag{33}$$

where $h = \max(h_y, h_u)$. However, because this method records u which produced unsatisfactory control results (i.e., large difference $r - y$) in the database, it often generates a poor control performance. Hence, we update the database only when (33) yields small tracking errors that are less than a prescribed level, i.e.,

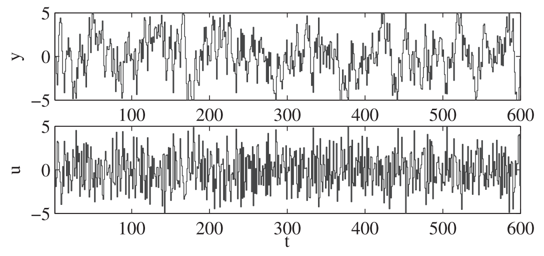

Fig. 2 Stored measurement data of the stabilized system for model-free predictive control. Top plot: y. Bottom plot: u.

$$\|\mathbf{r}(t-h) - \mathbf{y}_f(t-h)\| < \gamma. \tag{34}$$

3. Simulations and Discussions

In this section, we present several simulation results to evaluate the effect by database updates on model-free predictive control for unstable systems and to compare the three methods in Step 3. We first used the system

$$y(t) = 1.2y(t-1) + u(t-1) + \varepsilon(t) \tag{35}$$

with the unstable pole 1.2. The training data was created to use stabilizing feedback (32) with a static gain $K = 0.5$ and $r(t) = 0$. The resulting stabilized system is

$$y(t) = 0.7y(t-1) + v(t-1) + \varepsilon(t). \tag{36}$$

To apply 100 sets of random sequences $\varepsilon(t)$ according to Gaussian distribution with zero mean, variance $\sigma^2 = 0.05^2$, and random sequence $v(t)$ generated from a uniform distribution $[-3, 3]$ to the stabilized system, we generated 100 databases containing samples ($N = 600$) of the control input $u(t)$ and output $y(t)$. An example of the input/output data is shown in Fig. 2. Throughout the simulations, we set the order of the system and horizons as $n = 1$, $m = 1$, $h_y = 1$, and $h_u = 1$, and used two types of the references r: the sinusoidal signal

$$r(t) = 2\sin\frac{2\pi}{40}t \tag{37}$$

and the square signal

$$r(t) = \begin{cases} 0 & 0 \le t < 50 \\ 1 & 50 \le t < 100 \\ 0 & 100 \le t < 150 \\ -1 & 150 \le t < 200 \\ \vdots & \vdots \end{cases} \tag{38}$$

We used (14) and (20) as LWA for Step 3a and fixed the neighbor size $k = 4$. We adopted the distance defined by (17) for all methods to sort vectors. In Step 3b, we fixed $k = 10$. Since $d = n + h_y + m = 3 < k$, Step 3b provides the least-norm solution. In Step 3c, we used the DALM method [23] to solve (30).

To use the generated 100 databases and another 100 random sequences $\varepsilon(t)$, we simulated the three methods for model-free predictive control. To compare these methods, we calculated the sum of the squares of the tracking error $e(t) = r(t) - y(t)$ for the interval $t \in [a, b]$ as

$$\sum_{t=a}^{b} e(t)^2. \tag{39}$$

To denote the sequence of signal $e(a), \dots, e(b)$, we adopt the "colon" notation in Matlab as $e(a{:}b)$. In Fig. 3, we show the boxplots[2] of 100 samples of the sum of the squares of $e(101{:}500)$.

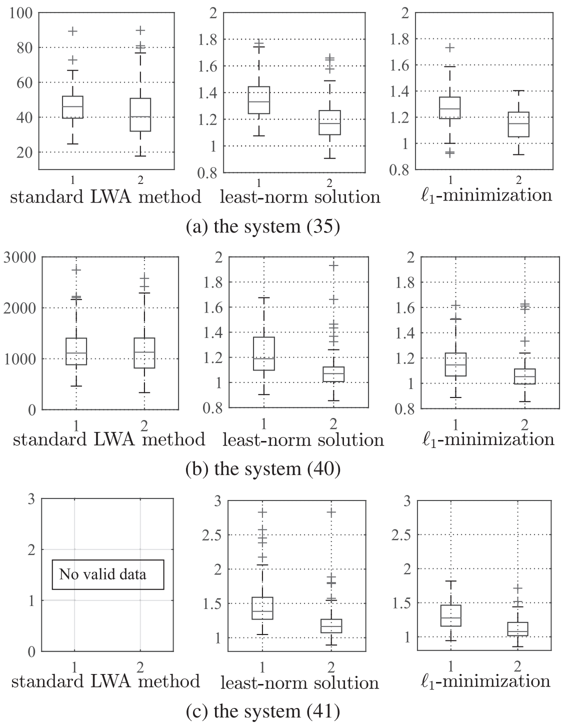

Fig. 3 Boxplot of the sum of squares of the tracking error $e(t) = r(t) - y(t)$ for the sinusoidal (label 1) and square references (label 2).

To investigate more, we also used

$$y(t) = y(t-1) + y(t-2) + 4u(t-1) + \varepsilon(t), \tag{40}$$
$$y(t) = 2y(t-1) - 3y(t-2) + 2u(t-1) + \varepsilon(t). \tag{41}$$

The training data was also created to use stabilizing feedback

$$u(t) = -0.35y(t-1) + v(t), \tag{42}$$
$$u(t) = -0.5y(t) + 1.3y(t-1) + v(t), \tag{43}$$

respectively, so that the resulting stabilized systems are

$$y(t) = y(t-1) - 0.4y(t-2) + 4v(t-1) + \varepsilon(t), \tag{44}$$
$$y(t) = y(t-1) - 0.4y(t-2) + 2v(t-1) + \varepsilon(t), \tag{45}$$

respectively. Obtained boxplots are shown in Fig. 3. In particular, the standard LWA method (Step 3a) shows worse tracking errors and cannot stabilize (41).

From Fig. 3, we conclude as follows.

- Model-free predictive control by the least-norm solution (Step 3b) and ℓ_1-minimization (Step 3c) yields less tracking errors than the standard LWA method (Step 3a). In the standard LWA method, we have several tunable parameters (the neighbor size k, weight for the distance, etc.). Hence, there is a possibility to obtain better results using more appropriate parameter values.
- Although ℓ_1-minimization (Step 3c) is the best in view of the tracking error, the computational time by ℓ_1-

[2] In all boxplots, the bottom of the box represents the first quartile, and the top of the box the third quartile. A horizontal line near the middle of the box indicates the median. A vertical line extends to the maximum value and another vertical line extends to the minimum value. Potential outliers are indicated by "+".

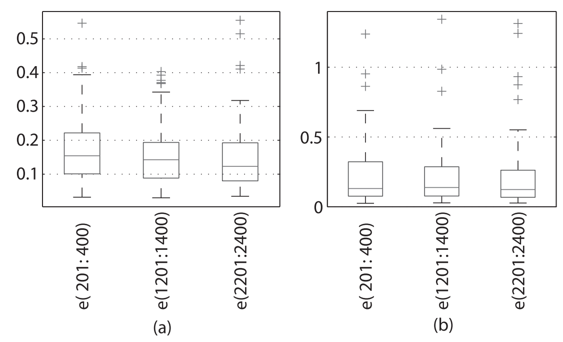

Fig. 4 Boxplot of the sum of squares of the tracking error $e(t) = r(t) - y(t)$ when we used the method by the least-norm solution to evaluate the effect of database maintenance: (a) the sinusoidal reference and (b) the square reference.

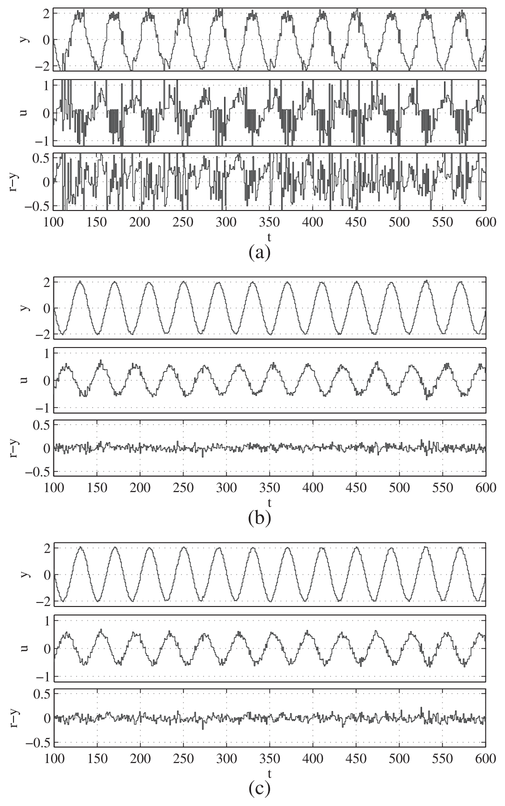

Fig. 5 Simulation results of model-free predictive control for the sinusoidal reference signal using a fixed database and the (a) standard LWA method, (b) least-norm solution, and (c) ℓ_1-minimization.

minimization is much longer than that by the other methods. For (35), the average computational ratios of Step 3b to Step 3a and Step 3c to Step 3a were approximately 0.999 and 14.21, respectively.

- In all methods, the tracking error for the square reference signal is smaller than that for the sinusoidal one because the former is a piecewise constant.

Furthermore, to evaluate the effectiveness of database maintenance for model-free predictive control based on the least-norm solution, for (35) we used 100 small-sized databases ($N = 200$) and compared the sum of squares of the tracking error over three intervals: $e(201{:}400)$, $e(1201{:}1400)$, and $e(2201{:}2400)$. We show a typical result in Fig. 4, which we obtained when we used 100 sets of random sequences $\varepsilon(t)$ according to Gaussian distribution with zero mean and variance $\sigma^2 = 0.01^2$. The variance was smaller than that ($\sigma^2 = 0.05^2$) in the first simulation results. To obtain the results, we used the level of database maintenance $\gamma = 6 \times 10^{-4}$ for the sinusoidal reference and $\gamma = 5 \times 10^{-4}$ for the square reference. The results were sensitive to γ. From Fig. 4, we conclude as follows.

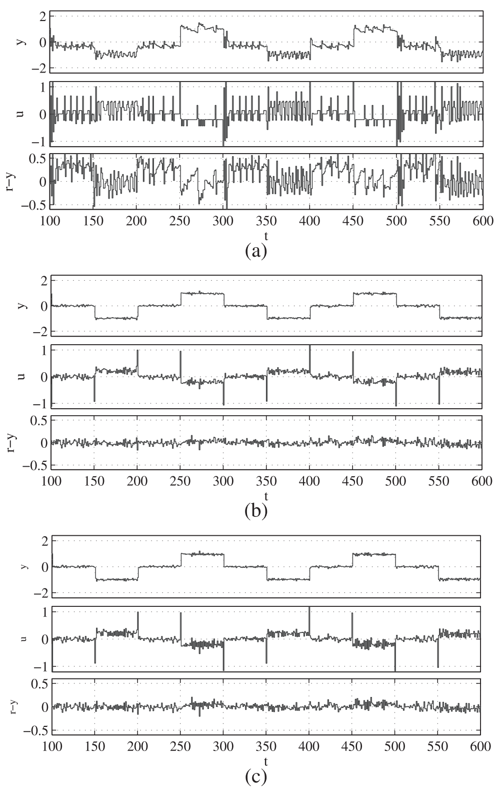

Fig. 6 Simulation results of model-free predictive control for the square reference signal using a fixed database and the (a) standard LWA method, (b) least-norm solution, and (c) ℓ_1-minimization.

- The interquartile range indicated by the boxes became smaller through database maintenance.
- The maximum of data points indicated by the end of the upper whiskers also became smaller through database maintenance.
- There are outliers indicated by "+". In particular, there exist large valued outliers in the results for the square reference.
- The distribution of the tracking errors for the square reference is poorer than that for the sinusoidal reference, unlike the distribution shown in Fig. 3; this is because of the piecewise constant reference. The procedure of database maintenance sweeps away important data for another setpoint r. The growth of the ratio of the current setpoint data causes the degradation of the control results when the setpoint r changes.

Finally, we show examples of simulation results of (35) in Figs. 5 and 6. In both figures, the dashed line indicates the reference signal r; the solid line is the output y; and the top, middle, and bottom are output y, input u, and error e, respectively.

4. Conclusion

In this study, we compared the three methods based on

LWA, least-norm solutions, and ℓ_1-minimization in model-free predictive control using Just-In-Time modeling for an unstable system. The least-norm solutions and ℓ_1-norm solutions gave much smaller tracking errors than the LWA. Since ℓ_1-minimization requires much longer computational time, we concluded that the method using least-norm solutions is the best for practical usage. Furthermore, we determined that database maintenance yields better results when working with a small-sized database.

Acknowledgments

This work was partially supported by JSPS KAKENHI Grant Number 23360183. The first author would like to thank to DIKTI Scholarship Indonesia and State Polytechnic of Sriwijaya Indonesia. The authors would like to thank the Associate Editor and the anonymous reviewers for valuable comments on the paper.

References

[1] C.E. Garcia, D.M. Prett, and M. Morari, "Model predictive control: Theory and practice — a survey," *Automatica*, Vol. 25, No. 3, pp. 335–348, 1989.

[2] A. Stenman, "Model-free predictive control," *Proc. 38th IEEE Conference on Decision and Control*, pp. 3712–3717, 1999.

[3] G. Cybenko, "Just-in-time learning and estimation," *NATO ASI Seris of Computer and Systems Sciences*, Vol. 153, pp. 423–434, 1996.

[4] A. Stenman, F. Gustafsson, and L. Ljung, "Just in time models for dynamical systems," *Proc. 35th IEEE Conference on Decision and Control*, Vol. 1, pp. 1115–1120, 1996.

[5] A. Stenman, A.V. Nazin, and F. Gustafsson, "Asypmtotic properties of just-in-time models," *Proc. 11th IFAC Symposium on System Identification*, pp. 1249–1254, 1997.

[6] A. Stenman, *Model on Demand: Algorithms, Analysis and Applications*, PhD thesis, Department of Electrical Engineering Linkoping University, 1999.

[7] M.W. Braun, D.E. Rivera, and A. Stenman, "A model-on-demand identification methodology for nonlinear process systems," *International Journal of Control*, Vol. 74, No. 18, pp. 1708–1717, 2001.

[8] G. Bontempi, M. Birattari, and H. Bersini, "Lazy learning for local modelling and control design," *International Journal of Control*, Vol. 72, No. 7-8, pp. 643–658, 1999.

[9] D.W. Aha, D. Kibler, and M.K. Albert, "Instance-based learning algorithms," *Machine learning*, Vol. 6, No. 1, pp. 37–66, 1991.

[10] Q. Zheng and H. Kimura, "A new just-in-time modeling method and its application to rolling set-up modeling," *Trans. Society of Instrument and Control Engineers*, Vol. 37, No. 2, pp. 640–646, 2001 (in Japanese).

[11] Q. Zheng and H. Kimura, "Just-in-time modeling for function prediction and its aplication," *Asian Journal of Control*, Vol. 3, No. 1, pp. 35–44, 2001.

[12] M. Kishi, K. Kimura, J. Ohta, and S. Yamamoto, "Shrinkage prediction of a steel production via model-on-demand," *Proc. 11th IFAC Symposium on Automation in Mining, Mineral and Metal Processing*, pp. 447–450, 2004.

[13] H. Shigemori, M. Kano, and S. Hasebe, "Optimum quality design system for steel products through locally weighted regression model," *Journal of Process Control*, Vol. 21, No. 2, pp. 293–301, 2011.

[14] J. Ohta and S. Yamamoto, "Database-driven tuning of PID controllers," *Trans. Society of Instrument and Control Engineers*, Vol. 40, pp. 664–669, 2004 (in Japanese).

[15] T. Yamamoto, K. Takao, and T. Yamada, "Design of a data-driven PID controller," *IEEE Transactions on Control Systems Technology*, Vol. 17, No. 1, pp. 29–39, 2009.

[16] K. Fujiwara, M. Kano, S. Hasebe, and A. Takinami, "Soft-sensor development using correlation-based just-in-time modeling," *AIChE Journal*, Vol. 55, No. 7, pp. 1754–1765, 2009.

[17] D. Inoue and S. Yamamoto, "Support for drivers via just-in-time predictive control and fault detection based on a nearest neighbor method during braking to stop trains," *Transactions of the Japan Society of Mechanical Engineers. C*, Vol. 72, pp. 2756–2761, 2006.

[18] K. Fukuda, S. Ushida, and K. Deguchi, "Just-in-time control of image-based inverted pendulum systems with a time-delay," *Proc. SICE-ICASE 2006*, pp. 4016–4021, 2006.

[19] E. Konaka, "Design of discrete predictive controller using approximate nearest neighbor method," *Proc. 18th IFAC World Congress*, pp. 10213–10218, 2011.

[20] T. Kosaki and M. Sano, "Networked just-in-time control of a parallel mechanism with pneumatic linear drives," *Communication in Control Science and Engineering (CCSE)*, Vol. 2, pp. 19–25, 2014.

[21] S. Yamamoto, "A new model-free predictive control method using input and output data," *3rd International Conference on Key Engineering Materials and Computer Science (KEMCS 2014)* (Advanced Materials Research Vol. 1042, ed.), pp. 182–187, Trans Tech Publication, 2014.

[22] S. Yamamoto, "A model-free predictive control method by l_1-minimization," *Proc. 10th Asian Control Conference 2015*, 1570064495, 2015.

[23] A.Y. Yang, Z. Zhou, A.G. Balasubramanian, S.S. Sastry, and Y. Ma, "Fast l_1-minimization algorithms for robust face recognition," *IEEE Transactions on Image Processing*, Vol. 22, No. 8, pp. 3234– 3246, 2013.

[24] M. Nagahara, D.E. Quevedo, and J. Ostergaard, "Sparse packetized predictive control for networked control over erasure channels," *IEEE Trans. Automatic Control*, Vol. 59, No. 7, pp. 1899–1905, 2014.

[25] N. Nakpong and S. Yamamoto, "Just-in-time predictive control for a two-wheeled robot," *Proc. 2012 Tenth Int. Conf. ICT and Knowledge Engineering*, No. 3, pp. 95–98, 2012.

Herlambang SAPUTRA

He received the B.Ed. degree from University of Sriwijaya in physics education, Palembang, Indonesia, in 2003, and M.Cs degrees from Gadjah Mada University in Computer Science, Yogyakarta, Indonesia, in 2005. He is currently a student and pursuing the Ph.D. degree in electrical engineering and computer science at Kanazawa University, Ishikawa, Japan.

Shigeru YAMAMOTO (Member)

He received the B.S., M.S., and Ph.D. degrees from Osaka University, Japan, in 1987, 1989, and 1996, respectively. In 1989, he joined Osaka University, as a Research Associate and became an Assistant Professor in 1998 and an Associate Professor in 2000. Since 2007, he has been a Professor at Kanazawa University, Japan. His research interests include control systems theories and applications. He is a member of JSME, ISCIE, ASME, IEEJ and IEEE.