A model-free predictive control method by ?1-minimization

著者	Yamamoto Shigeru
journal or publication title	Proc. of the 10th Asian Control Conference 2015 (ASCC 2015)
page range	1570064495
year	2015-09-08
URL	http://hdl.handle.net/2297/44812

doi: 10.1109/ASCC.2015.7244446

A Model-Free Predictive Control Method by ℓ_1-Minimization

Shigeru Yamamoto
Faculty of Electrical and Computer Engineering
Kanazawa University
Kakuma, Kanazawa, Ishikawa, Japan 920–1192
Email: shigeru@se.kanazawa-u.ac.jp

***Abstract*—We propose a new predictive control method utilizing a sparse solution of a minimization problem defined by both online and stored input/output data of the controlled system. The conventional predictive control methods generally require a mathematical model of the controlled system to predict an optimal future input to control the system. The mathematical model is usually obtained by applying a standard system identification method to the measured input/output data. The proposed method in this paper requires no mathematical model to predict future control input to achieve the desired output. This model-free control method, also called just-in-time predictive control, was originally proposed by Inoue and Yamamoto in 2004 and simplified by Yamamoto in 2014. In this paper, to develop another simplified method, we formulate an ℓ_1-minimization problem.**

I. INTRODUCTION

Predictive control, usually referred to as "model predictive control," is widely used in industrial systems such as chemical processes [1]. To use model predictive control, we need a mathematical model which appropriately represents the system dynamics to predict the future behavior of the system. The mathematical model is derived by many physical modeling techniques and/or useful system identification techniques in the development of online control. Normally, the mathematical model is not updated until a great change occurs in the controlled system.

The so-called just-in-time model predictive control proposed by Stenman in 1999 [2] can be interpreted as a combination of predictive control and adaptive control which constantly updates the mathematical model and/or control parameters based on the online measured input/output data. The method constantly maintains the mathematical model to use measured input/output data and stored past measured input/output data by Just-In-Time modeling [3], [4] (also referred to as model-on-demand [9], lazy learning [10], or instance based learning [11]). The just-in-time method was originally developed for nonlinear system modeling which adaptively identifies a local model (not global) that can represent the current input/output data to use a large amount of stored data obtained from past events.

Instead of the just-in-time "model" predictive control, Inoue and Yamamoto [12], [13] proposed a "model-free" predictive control method in the just-in-time modeling framework. In the method, an optimal control input is directly predicted not using any local models, but by online current measured data and stored past data. 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 weighted average of the neighbors. For this weighted average, several methods are considered in the just-in-time modeling framework. In [14], the author proposed a simple method to calculate the weighted average. In the method, the weights are derived as a solution of a linear equation $b = Aw$ where the vector b represents the current data and the matrix A contains vectors in the neighborhoods of b.

In this paper, another way to obtain a weighted mean of neighborhoods is proposed based on finding a minimum ℓ_1-norm solution to an underdetermined system of the linear equation $b = Aw$ where for a given full rank wide matrix A and a vector b, we need to find a vector w with minimum ℓ_1 norm. This ℓ_1-minimization problem has recently gained much attention in the signal processing and optimization community relating to compressive sensing [15].

II. MODEL-FREE PREDICTIVE CONTROL

We consider a nonlinear auto-regressive model with exogenous (NARX) inputs,

$$\begin{aligned} y(t) &= f(\varphi(t-1)) + \varepsilon(t) && (1)\\ \varphi(t-1) &= [y(t-1), \ldots, y(t-n), \\ &\quad\ u(t-1), \ldots, u(t-m)]^T, && (2)\\ &\quad\ t = 0, 1, 2 \cdots \end{aligned}$$

where $u \in \Re$ is the input; $y \in \Re$ is the output; $\varphi \in \Re^r$ is the regression vector of the size $r = n + m$; $f(\cdot)$ is a nonlinear map; and ε is independent and identically distributed (i.i.d.) noise, respectively. We assume that n and m are unknown together with f, but we can determine them with some uncertainty.

When we are given (1), the goal of control is to make the future output h steps ahead $y(t+1), y(t+2), \ldots, y(t+h)$ follow the desired future (given) reference $r(t+1), r(t+2), \ldots, r(t+h)$. To reach the goal, we need an appropriate future input sequence $u(t), u(t+1), \ldots, u(t+h-1)$. In this paper, we develop a method to predict the future input u and output y to use a large amount of past data $\{u(k),\ y(k)\}$, not to identify f explicitly. As proposed by Inoue and Yamamoto, the just-in-time method is used to find a future sequence of the input u making the future output y track a given reference signal r from the large amount of past data $\{u(k),\ y(k)\}$. The idea is summarized as the next algorithm.

Initialization. Determine m, n, the future horizon h_y, and the control horizon h_u. Based on the stored past input and output data (training data), define vectors as follows:

$$\mathbf{a}_i := \begin{bmatrix}\mathbf{y}_\mathrm{p}(t_i)\\ \mathbf{y}_\mathrm{f}(t_i)\\ \mathbf{u}_\mathrm{p}(t_i)\end{bmatrix} \quad (3)$$

$$\mathbf{c}_i := \mathbf{u}_\mathrm{f}(t_i) \quad (4)$$

where $i = 1, 2, \ldots, N$ and

$$\mathbf{y}_\mathrm{p}(t) = [y(t-n+1) \ \cdots \ y(t)]^T \quad (5)$$

$$\mathbf{y}_\mathrm{f}(t) = [y(t+1) \ \cdots \ y(t+h_y)]^T \quad (6)$$

$$\mathbf{u}_\mathrm{p}(t) = [u(t-m) \ \cdots \ u(t-1)]^T \quad (7)$$

$$\mathbf{u}_\mathrm{f}(t) = [u(t) \ \cdots \ u(t+h_u-1)]^T \quad (8)$$

Step 1. Let time $t = 0$.

Step 2. Whenever $t \leq \max(n, m)$, repeat this step. Measure $y(t)$ and apply $u(t)$ with an appropriate value to the system. Increment time as $t \leftarrow t + 1$.

Step 3. Based on the given reference signal to be followed in the future

$$\mathbf{r}(t) = \begin{bmatrix} r(t+1)\\ \vdots \\ r(t+h_y)\end{bmatrix}, \quad (9)$$

define a query vector

$$\mathbf{b} = \begin{bmatrix}\mathbf{y}_\mathrm{p}(t)\\ \mathbf{r}(t)\\ \mathbf{u}_\mathrm{p}(t)\end{bmatrix} \quad (10)$$

where the past output vector $\mathbf{y}_\mathrm{p}(t)$ and the past input vector $\mathbf{u}_\mathrm{p}(t)$ are defined as (5) and (7).

Step 4. By the just-in-time algorithm [5], find the k nearest vectors $\mathbf{a}_i$ to $\mathbf{b}$ together with $\mathbf{c}_i$, which have the same index i with $\mathbf{a}_i$ as follow:

$$\Omega(\mathbf{b}) := \{(\mathbf{a}_{i_j}, \mathbf{c}_{i_j}) \mid j = 1, \ldots, k\} \quad (11)$$

where we assume that all vectors are sorted by the distance to $\mathbf{b}$ as follows:

$$\|\mathbf{a}_{i_1} - \mathbf{b}\| \leq \|\mathbf{a}_{i_2} - \mathbf{b}\| \leq \cdots \leq \|\mathbf{a}_{i_k} - \mathbf{b}\|. \quad (12)$$

Furthermore, from the k nearest vectors $\mathbf{a}_i$ to $\mathbf{b}$, weights w_{i_j} are determined such as

$$w_{i_1} \geq w_{i_2} \geq \cdots \geq w_{i_k} \text{ and } w_{i_1} + w_{i_2} + \cdots + w_{i_k} = 1 \quad (13)$$

Step 5. The expected future input sequence

$$\hat{\mathbf{u}}_\mathrm{f}(t) = [\hat{u}(t|t) \ \ldots \ \hat{u}(t+h_u-1|t)]^T \quad (14)$$

is calculated as follows:

$$\hat{\mathbf{u}}_\mathrm{f}(t) = \sum_{j=1}^{k} w_{i_j} \mathbf{u}_\mathrm{f}(t_{i_j}) = \sum_{j=1}^{k} w_{i_j} \mathbf{c}_{i_j}. \quad (15)$$

Step 6. Apply the first element $\hat{u}(t|t)$ of $\hat{\mathbf{u}}_\mathrm{f}(t)$ to the system as $u(t)$. Increment time as $t \leftarrow t + 1$ and return to Step 3.

Remark 1: Finding k nearest vectors to $\mathbf{b}$ corresponds to finding a pair of past-future input and output sequences similar to the current situation containing the desired trajectory for the future output.

Remark 2: There are several methods to select the nearest neighbors and the appropriate weights in Step 4 [7], [8]. They depend on what system generates the data. Without exact information on the controlled system, it is difficult to determine the suitable nearest neighbors and weights.

Remark 3: The steps for the same discrete-time t in model-free control are executed once in every sampling period h. In general, the just-in-time algorithm computation yields a long feedback delay. Hence, model-free control is applicable for systems with slow dynamics.

III. ℓ_1-MINIMIZATION FORMULATION

Step 4 and 5 can be interpreted as finding vectors $\mathbf{a}_{i_j}, \mathbf{c}_{i_j}$ and weights w_{i_j} such that

$$\begin{aligned}&\min_{i_j, w_{i_j}, \hat{\mathbf{u}}_\mathrm{f}} \left\| \sum_{j=1}^{k} w_{i_j} \begin{bmatrix}\mathbf{a}_{i_j}\\ \mathbf{c}_{i_j}\end{bmatrix} - \begin{bmatrix}\mathbf{b}\\ \hat{\mathbf{u}}_\mathrm{f}(t)\end{bmatrix}\right\|_2 \\ &= \min_{t_j, w_{i_j}, \hat{\mathbf{u}}_\mathrm{f}} \left\| \sum_{j=1}^{k} w_{i_j} \begin{bmatrix}\mathbf{y}_\mathrm{p}(t_i)\\ \mathbf{y}_\mathrm{f}(t_i)\\ \mathbf{u}_\mathrm{p}(t_i)\\ \mathbf{u}_\mathrm{f}(t_i)\end{bmatrix} - \begin{bmatrix}\mathbf{y}_\mathrm{p}(t)\\ \mathbf{r}(t)\\ \mathbf{u}_\mathrm{p}(t)\\ \hat{\mathbf{u}}_\mathrm{f}(t)\end{bmatrix}\right\|_2 \end{aligned} \quad (16)$$

where $\|\cdot\|_2$ is the Euclidean norm.

In [14], for given vectors $\mathbf{a}_{i_1}, \ldots \mathbf{a}_{i_k}, \mathbf{c}_{i_1} \ldots \mathbf{c}_{i_k}$ and $\mathbf{b}$, weights $w_{i_1}, \ldots, w_{i_k}$ are found by solving

$$A\mathbf{w} = \mathbf{b} \quad (17)$$

where

$$A = [\mathbf{a}_{i_1} \ \ \mathbf{a}_{i_2} \ \ \ldots \ \ \mathbf{a}_{i_k}] \in \Re^{d \times k}, \quad (18)$$

$$\mathbf{w} = [w_{i_1} \ \ w_{i_2} \ \ \cdots \ \ w_{i_k}]^T \in \Re^{k}, \quad (19)$$

and

$$d = n + h_y + m. \quad (20)$$

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

$$\min_{i_j, x_{i_j}} \left\| \sum_{j=1}^{k} w_{i_j} \mathbf{a}_{i_j} - \mathbf{b} \right\|_2. \quad (21)$$

Also, we can extend the size of A and $\mathbf{b}$ from the neighbor size k to the size of training data N. Then, the new problem is written as follows:

$$\min_{w} \|A\mathbf{w} - \mathbf{b}\| \ \text{ subj. to } \|\mathbf{w}\|_0 = k \quad (22)$$

where

$$A = [\mathbf{a}_1 \ \ \mathbf{a}_2 \ \ \ldots \ \ \mathbf{a}_N] \in \Re^{d \times N}, \quad (23)$$

$$\mathbf{w} = [w_1 \ \ w_2 \ \ \cdots \ \ w_N]^T \in \Re^{N}, \quad (24)$$

and the ℓ_0 norm $\|\mathbf{w}\|_0$ is a total number of non-zero elements in $\mathbf{w}$ which is defined using set cardinality as follows:

$$\|\mathbf{w}\|_0 = \text{card}\{w_i \mid w_i \neq 0\}. \tag{25}$$

Because of the ℓ_0 norm constraint, (22) is the mixed-integer problem which is generally hard to solve. In addition, before solving (22), we need to determine the ℓ_0 norm k, as usual in the just-in-time algorithm, using the Akaike's Final Prediction Error criterion. Instead, in Step 4 of the proposed method, we solve an optimization problem to find $\mathbf{w}$ and k.

Step 4' (the proposed method)

Solve the optimization problem:

$$\min_{\mathbf{w}} \|\mathbf{w}\|_1 \text{ subj. to } A\mathbf{w} - \mathbf{b} = 0, \tag{26}$$

where $\|\mathbf{w}\|_1 = \sum_{i=1} |w_i|$.

When $A \in \Re^{d\times N}$ is of full row rank and $d < N$, infinite many solutions $\mathbf{w}$ for $A\mathbf{w} = \mathbf{b}$ exist. Among them, by penalizing $\|\mathbf{w}\|_1$ we can find a sparse solution with many zero elements in the vector. This optimization problem is referred to as ℓ_1-minimization.

To solve ℓ_1-minimization, several methods have been developed. In particular, a number of accelerated algorithms have been proposed. In [15], we have a comprehensive review of representative approaches, namely, Gradient Projection, Homotopy, Iterative Shrinkage-Thresholding, Proximal Gradient, and Augmented Lagrange Multiplier. In this paper, we use the DALM (Dual Augmented Lagrange Multiplier) method described in next section (Solvers in MATLAB are available at `http://www.eecs.berkeley.edu/~yang/software/l1benchmark/l1benchmark.zip`).

IV. Simulations

In this section, we show several simulation results to illustrate the performance of the model-free predictive control by solving ℓ_1-minimization. We use the DALM algorithm to solve ℓ_1-minimization.

A. Linear System

We consider a linear system

$$y(t) = y(t-1) - 0.16y(t-2) - 1.5u(t-1) + \varepsilon(t) \tag{27}$$

where ε is a noise term whose entries are i.i.d. according to Gaussian distribution with zero mean and variance of 0.05. The transfer function from u to y has stable poles at 0.8 and 0.2.

The training data was created to use $u(k)$ with a 300 i.i.d. random sequence generated from a uniform distribution $[-5, 5]$. We stored 300 pairs of $u(t)$ and $y(t)$ as shown in Fig. 1.

We first set the order of the system and horizons as $n = 2$, $m = 1$, $h_y = 3$, and $h_u = 3$. The order n and m are the same with the linear system (27). The reference input is generated by

$$r(t) = 2\sin\frac{2\pi}{40}t \tag{28}$$

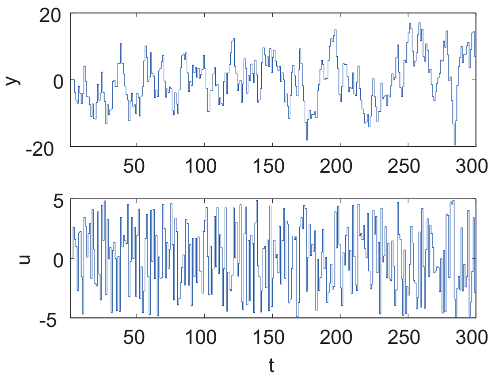
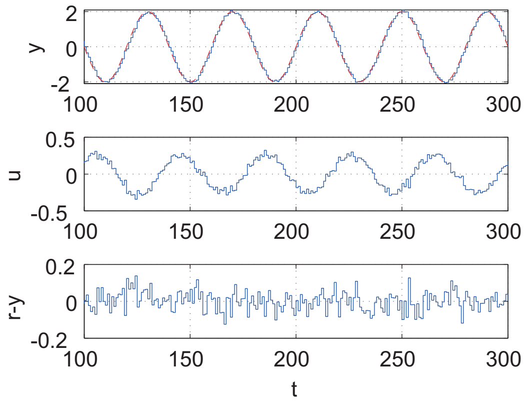
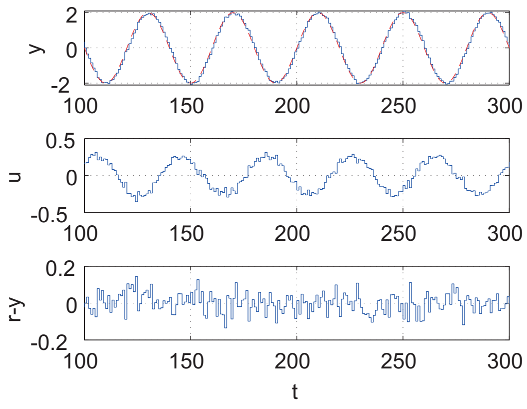


Fig. 1: Stored measurement data of the linear system (27) for model-free predictive control.



(a) When the same order $n = 2$ and $m = 1$ is used.



(b) When an over-estimated order $n = 3$ and $m = 2$ is used.

Fig. 2: Simulation results of model-free predictive control using ℓ_1 minimization for the linear system (27) with $n = 2$ and $m = 1$.

Figure 2a shows the output y, the control input u, and the tracking error $e = r - y$ in a simulation result by ℓ_1 minimization when $n = 2$, $m = 1$, $h_y = 3$, and $h_u = 3$. In this case, $A \in \Re^{5\times 296}$.

When we used an over-estimated order $n = 3$, $m = 2$ (in this case, $A \in \Re^{8\times 296}$), we obtained a similar result with that in Fig. 2a as shown in Fig. 2b. It is difficult to distinguish the difference.

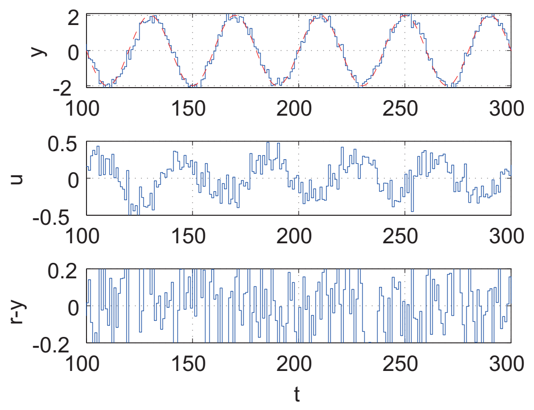


Fig. 3: A noisy case result of model-free predictive control using ℓ_1 minimization for the linear system (27). The variance of noise is four times larger than that in Fig. 2.

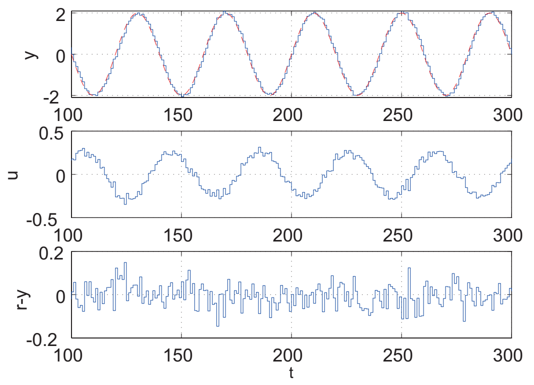


Fig. 4: A simulation result of model-free predictive control by solving the linear equation (17) in [14] for the linear system (27).

We were also interested in the noisy case where the measurement contains much noise. Here, we apply the i.i.d. noise with Gaussian distribution with zero mean and variance of 0.2 (four times larger than that in Fig. 2) when creating the training data and control. Figure 3 shows a simulation result of model-free predictive control using ℓ_1 minimization for the noisy case.

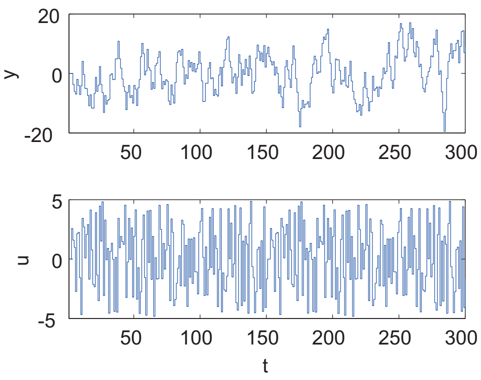
To compare the proposed method with that in [14], we solved (17) in Step 4 for (27). Figure 4 shows a simulation result of model-free predictive control when we used the same parameters with that in Fig. 2a and $k = 10$. To find k nearest vectors $\mathbf{a}_i \in \Re^d$, we used the infinity distance $\|\mathbf{a}_i - \mathbf{b}\|_\infty = \max_{j=1,\dots,d} |a_{ij} - b_j|$.

B. Switched Linear System

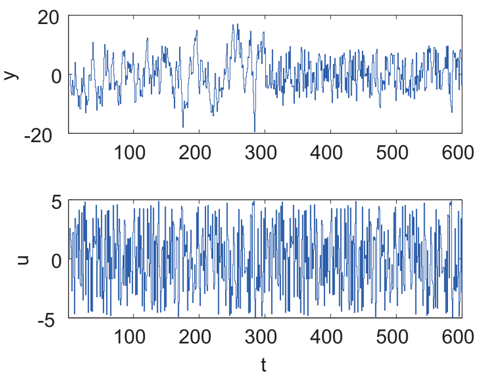
We consider a switched linear system

$$y(t) = \begin{cases} y(t-1) - 0.16y(t-2) \\ \quad - 1.5u(t-1) + \varepsilon(t), & \text{for } 0 \le t \le t_s \\ 0.3y(t-1) + 0.1y(t-2) \\ \quad - 1.8u(t-1) + \varepsilon(t), & \text{for } t_s + 1 \le t \end{cases} \tag{29}$$

where ε is a noise term whose entries are i.i.d. according to Gaussian distribution with zero mean and variance 0.05. This system is exactly the linear system (27) before t_s and it changes the dynamics with stable poles at 0.5 and -0.2 after t_s.



(a) 300 samples used in Fig. 6a.



(b) 600 samples used in Fig. 6b.

Fig. 5: Stored measurement data of the switched system (29).

To obtain the training data, we set $t_s = 150$ and we stored 300 pairs of $u(t)$ and $y(t)$ as shown in Fig. 5a for $u(k)$ with a 300 i.i.d. random sequence generated from a uniform distribution $[-5, 5]$.

By setting $t_s = 200$, we performed a simulation. Figure 6a shows the output y, the control input u, and the tracking error $e = r - y$ in a simulation result by ℓ_1 minimization when $n = 2$, $m = 1$, $h_y = 3$, and $h_u = 3$. In this case, $A \in \Re^{5\times 296}$. The tracking errors are worse than the linear system in Fig. 2a. In particular, after $t_s = 150$ it shows large tracking errors.

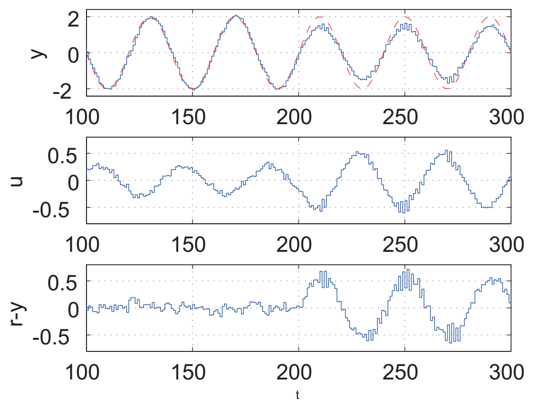
Next, by setting $t_s = 300$, we increased training data (the number of samples) to 600 pairs of $u(t)$ and $y(t)$, as shown in Fig. 6b.

When we used the training data shown in Fig. 6b, we obtained the simulation result in Fig. 6b where $t_s = 200$, $n = 2$, $m = 1$, $h_y = 3$ and $h_u = 3$, and $A \in \Re^{5\times 296}$. Although the tracking errors are worse than the linear system in Fig. 2a, it shows slightly better tracking errors than Fig. 6a.

C. Nonlinear System

We consider a nonlinear system

$$y(t) = \frac{y(t-1)}{1 + y(t-1)^2} + u(t-1)^3 + \varepsilon(t) \tag{30}$$


(a) Stored measurement data (300 samples) in Fig. 5a are used.

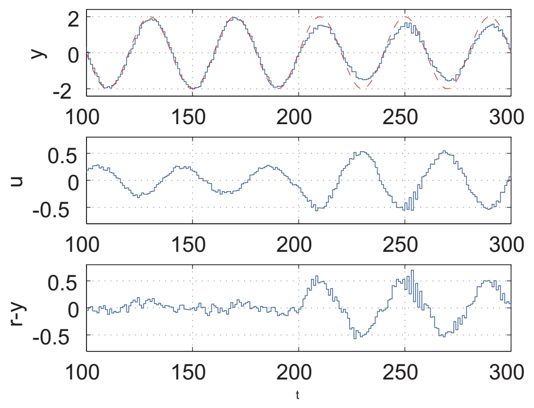

(b) Stored measurement data (600 samples) in Fig. 5b are used.

Fig. 6: Simulation results of model-free predictive control using ℓ_1 minimization for the switched system (29).

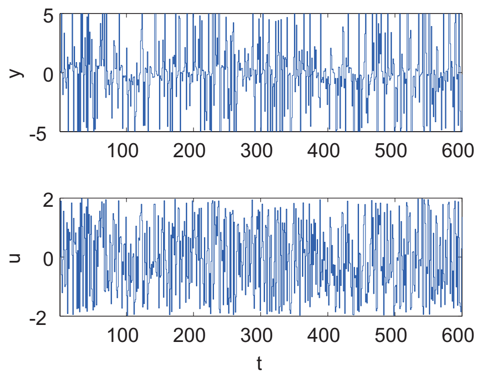
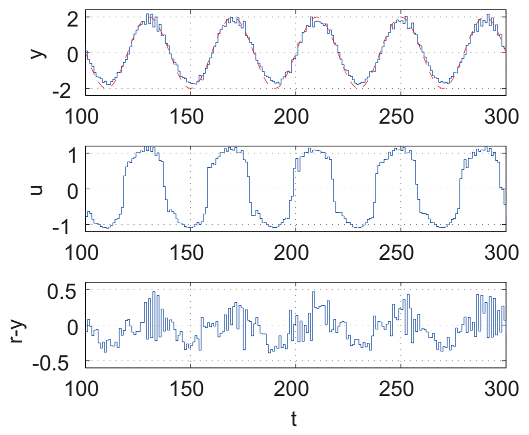


Fig. 7: Stored measurement data of the nonlinear system (30).

where ε is a noise term whose entries are i.i.d. according to Gaussian distribution with zero mean and variance of 0.05. The linearized system around origin is not stabilized.

The training data was created to use $u(k)$ with a 600 i.i.d. random sequence generated from a uniform distribution $[-2, 2]$. We stored 600 pairs of $u(t)$ and $y(t)$ as shown in Fig. 7.


(a) model-free predictive control using ℓ_1 minimization

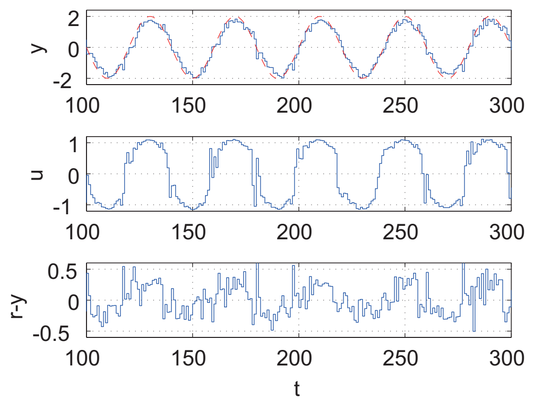

(b) model-free predictive control by solving (17) in [14]

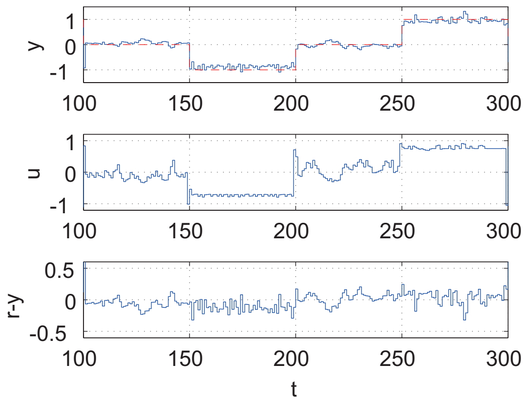
Fig. 8: Simulation results for the nonlinear system (30).

When we directly use Step 4' for all stored data in Fig. 7, very poor results are obtained. Hence, before solving the ℓ_1 minimization problem in Step 4', we modified the algorithm to select nearest data according to the method in [14] by the infinity distance $\|\mathbf{a}_i - \mathbf{b}\|_\infty$. Figure 8a shows a simulation result where $n = 1$, $m = 1$, $h_y = 1$, $h_u = 1$, and $A \in \Re^{5\times 10}$. To compare the proposed method with that in [14], we also show a result in Fig. 8b where $n = 1$, $m = 1$, $h_y = 1$, and $h_u = 1$, and $k = 10$. As we see in the two figures, the tracking errors are larger than those in linear systems and there is no significant difference between the two methods.

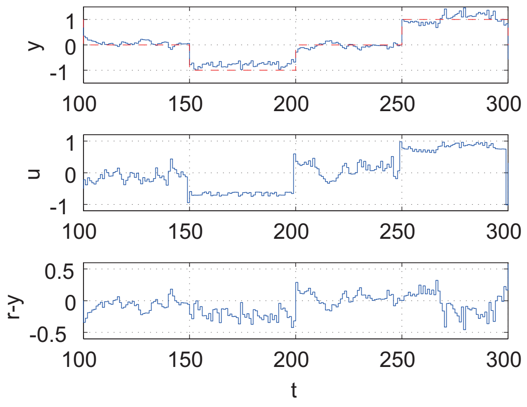
Next, we changed the reference signal as

$$r(t) = \begin{cases} 0, & 0 \le t < 50 \\ 1, & 50 \le t < 100 \\ 0, & 100 \le t < 150 \\ -1, & 150 \le t < 200 \\ \vdots & \vdots \end{cases} \tag{31}$$

For the step reference signal r, we applied two methods. Figure 9a shows a simulation result where $n = 1$, $m = 1$, $h_y = 1$, $h_u = 1$, and $A \in \Re^{5\times 10}$. Figure 9b shows a simulation result by [14] with the same parameters, respectively. As we see in these figures, the tracking errors by the proposed method in Fig. 9a are smaller than those in Fig. 9b by [14].



(a) model-free predictive control using ℓ_1 minimization



(b) model-free predictive control by solving (17) in [14]

Fig. 9: Simulation results for the nonlinear system (30) when the reference is (31).

V. Conclusion

In this paper, a new method is proposed for model-free predictive control in the just-in-time modeling framework in which online data and stored data of the input/output of the controlled system are utilized. In the original model-free predictive control proposed by Inoue and Yamamoto [12], [13], as in just-in-time modeling, we need to determine several tuning parameters for obtaining the k nearest neighbors. Our proposed method can avoid such oppressiveness for the users similarly as in [14]. In contrast to l_2-norm minimization in [14], we introduce an l_1-norm minimization problem to obtain the weighting average in model-free predictive control.

Acknowledgment

This work was partially supported by JSPS KAKENHI Grant Number 23360183.

References

[1] Carlos E. García, David M. Prett, Manfred Morari: Model predictive control: Theory and practice?A survey, *Automatica*, Vol. 25, Iss. 3, pp. 335–348, 1989

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

[3] A. Stenman, F. Gustafsson and H. Ljung: Just in time models for dynamical systems, *35th IEEE Conference on Decision and Control*, pp. 1115–1120, 1996

[4] A. Stenman, A. V. Nazin, and F. Gustafsson: Asymptotic properties of Just-In-Time models, *Preprints of the 11th IFAC Symposium on System Identification*, pp. 1249–1254, 1997

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

[6] G. Cybenko: Just-in-time learning and estimation. In S. Bittani and G. Picci, editors, Identification, Adaption, Learning, NATO ASI series, pp. 423–434, Springer, 1996

[7] Q. Zheng and H. Kimura: Locally Weighted Regression Based on k Biparticle Neighbors, *42nd Japan Joint Automatic Control Conference*, pp. 143–144, 1999

[8] Q. Zheng and H. Kimura: Just-In-Time Modeling for Function Prediction and its Applications, *Asian. J. Control*, Vol. 3, No. 1, pp. 35–44, 2001

[9] M. W. Braun, D. E. Rivera, A. Stenman: A 'Model-on-Demand' identification methodology for non-linear process systems, *International Journal of Control*, Vol. 74, Iss. 18, pp. 1708–1717, 2001

[10] G. Bontempi, M. Birattari and H. Bersini: Lazy Learning for local modeling and design; *International Journal of Control*, Vol. 72, No. 7/8, pp. 634–658, 1999

[11] David W. Aha, Dennis Kibler, and Marc K. Albert, Instance-based learning algorithms, *Machine Learning*, Vol. 6, Iss. 1, pp. 37–66, January 1991.

[12] D. Inoue and S. Yamamoto: An Operation Support System based on Database-Driven On-Demand Predictive Control; *Proceedings of 2004 SICE Annual Conference*, pp. 2024–2027, 2004

[13] D. Inoue and S. Yamamoto: A Memory-Based Predictive Control Approach to a Braking Assist Problem, *Proceedings of SICE Annual Conference 2005*, pp. 3186–3189, 2005

[14] S. Yamamoto: A New Model-Free Predictive Control Method Using Input and Output Data, The 3rd International Conference on Key Engineering Materials and Computer Science (KEMCS 2014), Singapore, August 5, 2014. In Advanced Materials Research, Vol. 1042, pp. 182–187, 2014

[15] A. Y. Yang, A. Ganesh, Z. Zhou, S. Sastry and Y. Ma: A Review of Fast ℓ_1-Minimization Algorithms for Robust Face Recognition, CoRR, abs/1007.3753, http://arxiv.org/abs/1007.3753, 2010