Article:
Zhang, Z., Xie, L., Lu, S. et al. (2 more authors) (2022) A low-cost pole-placement MPC algorithm for controlling complex dynamic systems. Journal of Process Control, 111. pp. 106-116. ISSN 0959-1524

https://doi.org/10.1016/j.jprocont.2022.02.001

A low-cost pole-placement MPC algorithm for controlling complex dynamic systems

Zhiming Zhang[a], Lei Xie[a,*], Shan Lu[c,*], John Anthony Rossiter[b], Hongye Su[a]

[a]*State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou, China*
[b]*Department of Automatic Control and System Engineering, University of Sheffield, Sheffield, UK*
[c]*Institute of Intelligence Science and Engineering, Shenzhen Polytechnic, Shenzhen, China*

Abstract

Due to the ability to handle constraints systematically and predict system evolution with models, model predictive control (MPC) methods have been widely studied and implemented in many industries. At the same time, low-cost MPC has received widespread attention due to its simple principle and easy implementation. This paper proposes a new low-cost MPC method and uses this method to put forward new insights on the performance improvement of complex dynamic system control. Firstly, using the concept of preprocessing, a novel MPC prediction structure is proposed under the independent model mode. Then through rigorous proofs, the properties of the proposed MPC algorithm are analyzed. Finally, through the study of three industrial cases, the proposal's deployment procedure and efficacy are illustrated in detail. Compared with other low-cost predictive control algorithms, the effectiveness of the method has been presented.

Keywords: Low-cost MPC, pole-placement, complex dynamic systems, prediction structure design

*Corresponding author
Email addresses: leix@iipc.zju.edu.cn (Lei Xie), lushan@szpt.edu.cn (Shan Lu)

1. Introduction

Model predictive control (MPC) [1, 2] has been one of the industry's most successful advanced process control techniques in recent decades. However, the implied high computational loads and optimizations make MPC expensive, and thus its use is primarily evidenced in high throughput processes. In consequence, even though MPC could potentially offer performance benefits more widely, the cheaper and simpler PID still predominates in small-scale and fast-varying industrial processes and feedback loops.

A popular low-cost MPC approach, namely Predictive Functional Control (PFC) [3], is simplified by decreasing the degrees of freedom and avoiding complex optimization problems and thus is a major exception to the above. Specifically, it is competitive in price and complexity with PID, and being model-based, PFC can also deal with constraints systematically. The PFC algorithm can be implemented with very simple code, including handling typical input/output rate and absolute constraints, on devices with limited computing resources such as Programmable Logic Controllers (PLCs). Of course it is unsurprising that PFC, being a simple and very cheap MPC approach, is neither as flexible nor efficient as Dynamic Matrix Control (DMC) [4] and Generalized Predictive Control (GPC) [5]. In addition, the simplifications used to create a low-cost algorithm remove the possibility of a priori generic and rigorous mathematical analysis and proofs of convergence, performance and feasibility, except for a few special cases [6, 7, 8]. However, industrial practitioners appreciate the desired strong link between the selected or desired behavior and what is achieved; unfortunately, this link is often weaker in practice than required to be useful. Consequently, many researchers have proposed several modifications to PFC to improve the link between targeted and achieved behavior for various system types [9, 10].

As a low-cost MPC method, PFC has many attractive properties in specific cases, which can be exploited in the proposed algorithm. The underlying principle of PFC is carefully explored in [6]. It suggests an effective strategy to choose the coincidence point for good closed-loop performance. However, critically it

is noted that the desired tuning can rarely be achieved for high order systems. To improve the links between desired and achieved closed-loop poles with high-order systems, the use of parallel first-order models was used in [11, 12] to facilitate the use of a short coincidence point. However, this approach also requires further design decisions that may not be intuitive, and extensions to handle complex poles are messy.

More recently, two novel and effective modifications have been proposed that enable easier tuning and better closed-loop behavior. The first exploits Laguerre functions, so the control input parametrization is reshaped to improve the consistency between the predictions and the desired closed-loop behavior [13, 14, 15]. However, although a useful concept, the use of Laguerre functions is not suitable for all open-loop dynamics, and an effective generalization is yet to be proposed. The second proposal, namely output feedback PFC, was first proposed in [16, 17] and an offline feedback gain parameter was used to successfully implement PFC in vessel level control with easy tuning. However, while the existing output feedback PFC proposals are suitable for integrator or unstable first-order systems, further developments are necessary for the application to complex dynamic systems, which is the proposal that appears in this paper.

In summary, a unified low-cost MPC framework for higher-order systems does not exist in the literature; using inspiration and insight from PFC, a low- cost pole-placement MPC algorithm is proposed in this paper. The main contributions of this paper are summarized next:

- A novel low-cost MPC algorithm based on transfer function model is proposed, which offers better performance on complex dynamic systems but remains cost-efficient and includes systematic constraints handling.
- Model decomposition is introduced into MPC to achieve the goal of pole-placement. With the pre-conditioning concept, the open-loop poles of an internal closed-loop model are adjusted to the desired ones.
- The rigorous analysis and pseudocode of the proposed algorithm are given, and in the industrial case studies, the approach is seen to outperform PID

and match more expensive MPC algorithms in various single-input-single-output scenarios.

The rest of the paper is organized as follows. Section 2 begins with a brief introduction to the conventional PFC algorithm, which is the foundation of this research; the weak link between desired and achieved behavior is illustrated through a second-order example. Section 3 presents an interesting property of PFC and then uses this to define a compact and efficient MPC algorithm for achieving open-loop pole behavior in a closed-loop fashion. Then, section 4 introduces the pre-conditioning concept and algorithm to place the open-loop poles where desired; some analysis and constraints handling principles are given. Section 5 shows the simulation results for several numerical cases. Following that, conclusions and future work are stated in section 6.

2. Preliminaries

2.1. Conventional predictive functional control

This section will provide a brief introduction to PFC. It is commonly assumed that the predicted future input is constant in the conventional PFC algorithm [3], which is practical and simplifies the formulation as well.

For unification of illustration, at time instant k, the i-step ahead prediction with a constant future input ($u(k+i) = u(k), \forall i \geq 0$) for a transfer function model with delay D takes the following form for input $u(k)$:

$$y_m(k+D+i|k) = H \underrightarrow{u}(k) + P \underleftarrow{u}(k) + Q \underleftarrow{y_m}(k+D), i = 1, 2, ... \qquad (1)$$

where y_m is the model output, H, P and Q depend on the model parameters[1]

[1] Prediction is standard in the literature, but readers could also refer to the Chapter 2 in [2] for further details.

and for a model of order l:

$$\underset{\rightarrow}{u}(k) = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+i-1) \end{bmatrix}; \underset{\leftarrow}{u}(k) = \begin{bmatrix} u(k-1) \\ u(k-2) \\ \vdots \\ u(k-l) \end{bmatrix}; \underset{\leftarrow}{y_m}(k+D) = \begin{bmatrix} y_m(k+D) \\ y_m(k+D-1) \\ \vdots \\ y_m(k+D-l) \end{bmatrix}$$

Remark 1. *Note that notation y_p is used for the actual process output measurement and y_m is the model output, $d(k)$ is an offset correction term; generally one uses $d(k) = y_p(k) - y_m(k)$ to estimate a suitable correction value. The actual process prediction is then given as:*

$$y_p(k+i|k) = y_m(k+i|k) + d(k); \tag{2}$$

Assume the system output is $y_p(k)$ and the setpoint is R, the desired reference trajectory $r(k+D+i|k)$, which is also the desired closed-loop behaviour, is denoted by:

$$r(k+D+i|k) = y_p(k) + (R - y_p(k))(1-\lambda^i), i = 1, 2, \cdots \tag{3}$$

where λ is the desired closed-loop pole (which is equivalent to the desired closed-loop time response (CLTR) parameter) and one notes some lag in response is included in this to cater for the delay D.

The user needs to select a coincidence point n_y in a conventional PFC algorithm where the prediction (2) is forced to match the desired trajectory $r(k+D+n_y|k)$ and thus the PFC control law is defined by ensuring:

$$y_p(k+D+n_y|k) = r(k+D+n_y|k) \tag{4}$$

Substituting prediction eqns.(1,2,3) into (4) gives the control law of a conventional PFC algorithm as:

$$u(k) = \frac{1}{H_1}\left[(R - y_p(k))(1-\lambda^{n_y}) + y_p(k) - Q\underset{\leftarrow}{y_m}(k+D) - P\underset{\leftarrow}{u}(k) - d(k)\right] \tag{5}$$

where $H_1 = H \cdot [1, 1, \cdots]^T$ due to the the constant future input assumption. The reader should note that as $y_m(k)$ is based on a model, one can determine the future value of $\underset{\leftarrow}{y_m}(k+D)$ based on solely earlier values of the input at sample k.

2.2. The weak effect of the PFC tuning parameters

The intuitive selling point of conventional PFC is designed to have just one tuning parameter, which is the desired closed-loop pole λ (equivalently CLTR). However, the influence of coincidence point n_y cannot be neglected in the algorithm [6]. The effect of λ and n_y on high-order systems will be discussed in this subsection, presenting some novel analyses of the scenario for a second-order system.

For a second-order stable system such as:

$$(1 - a_1 z^{-1} - a_2 z^{-2}) y(k) = (b_1 z^{-1} + b_2 z^{-2}) u(k), \tag{6}$$

Take the nominal case so that $y_m = y_p$, then the control law according to (5) is given as[2]:

$$u(k) = [(R - y(k))(1 - \lambda^{n_y}) + (1 - Q_1) y(k) - Q_2 y(k-1) - P_1 u(k-1)] \cdot \frac{1}{H_1}$$

where it is easy to show that:

$$Q_1 = \left[\frac{1}{2} + \frac{a_1}{2\sqrt{a_1^2 + 4a_2}}\right] \left(\frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2}\right)^{n_y} + \left[\frac{1}{2} - \frac{a_1}{2\sqrt{a_1^2 + 4a_2}}\right] \left(\frac{a_1 - \sqrt{a_1^2 + 4a_2}}{2}\right)^{n_y}$$

$$Q_2 = \frac{a_2}{\sqrt{a_1^2 + 4a_2}} \left(\frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2}\right)^{n_y} - \frac{a_2}{\sqrt{a_1^2 + 4a_2}} \left(\frac{a_1 - \sqrt{a_1^2 + 4a_2}}{2}\right)^{n_y}$$

$$P_1 = \frac{b_2}{\sqrt{a_1^2 + 4a_2}} \left(\frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2}\right)^{n_y} - \frac{b_2}{\sqrt{a_1^2 + 4a_2}} \left(\frac{a_1 - \sqrt{a_1^2 + 4a_2}}{2}\right)^{n_y}$$

$$\begin{aligned} H_1 = &- \frac{\left(b_1 - \frac{b_1 + b_2}{1 - a_1 - a_2}\right)\left(\frac{a_1 - \sqrt{a_1^2 + 4a_2}}{2} - 1\right) - a_1 b_1 - b_2}{\sqrt{a_1^2 + 4a_2}} \left(\frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2}\right)^{n_y - 1} \\ &+ \frac{\left(b_1 - \frac{b_1 + b_2}{1 - a_1 - a_2}\right)\left(\frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2} - 1\right) - a_1 b_1 - b_2}{\sqrt{a_1^2 + 4a_2}} \left(\frac{a_1 - \sqrt{a_1^2 + 4a_2}}{2}\right)^{n_y - 1} \\ &+ \frac{b_1 + b_2}{1 - a_1 - a_2} \end{aligned}$$

Consequently, the closed-loop pole can be obtained by solving the following

[2]Readers should refer to Appendix 1 for a fuller derivation.

equation:

$$H_1 z^3 + [P_1 - H_1 a_1 + b_1(Q_1 - \lambda^{n_y})]z^2 + [Q_2 b_1 - b_2(\lambda^{n_y} - Q_1) - P_1 a_1 - H_1 a_2]z + (Q_2 b_2 - P_1 a_2) = 0$$

Critically, it is observed that λ has totally lost its effect on closed-loop behavior, except for some specific choices of n_y (specifically $n_y = 1$), which partially explains why tuning for high-order systems can be difficult. Although this section has focussed on a second-order system for ease of illustration, the core point is that the roles of λ and n_y become weak and ambiguous in controlling complex dynamic systems. Consequently, before developing a novel low-cost MPC, it is necessary to improve the tuning transparency and at the same time retain any positive attributes such as cost efficiency and systematic constraints handling.

3. Low-cost MPC for achieving open-loop behavior

In this section, a low-cost MPC algorithm (denoted as open-loop MPC, OLP-MPC) is introduced to realize that the closed-loop behavior of a system is similar to the open-loop behavior of the controlled process as shown in Fig.1.

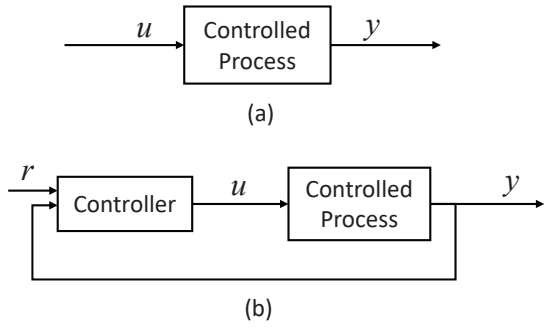

Figure 1: (a) The open-loop behavior of the controlled process; (b) The closed-loop behavior of a system: the controller and the controlled process are connected by feedback control.

The proposed OLP-MPC algorithm uses an internal model control structure as shown in Fig.2, where the controller in Fig.1 is described in detail.

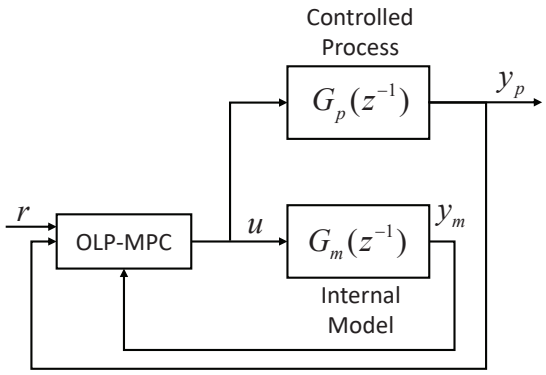

Figure 2: The control structure of OLP-MPC. The controller part in Fig.1(b) is expanded.

For a system as shown in Fig.2, some terminologies are defined as follows.

Definition 1. *Controlled process. An arbitrary stable causal controlled process ($m \leq n$) is defined as:*

$$G_p(z^{-1}) \triangleq \frac{b_1 z^{-1} + b_2 z^{-2} + \cdots + b_m z^{-m}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_n z^{-n}} \triangleq \frac{B(z^{-1})}{A(z^{-1})}, \tag{7}$$

where z^{-1} is the shift operator. Given the input u, the output y_p can be denoted by

$$\begin{aligned} y_p(k) + a_1 y_p(k-1) + a_2 y_p(k-2) + \cdots + a_n y_p(k-n) \\ = b_1 u(k-1) + b_2 u(k-2) + \cdots + b_m u(k-m) \end{aligned}$$

Definition 2. *Internal model. The model used by the model predictive control is often obtained by some identification methods. In the internal model control structure, the model output y_m is parallel to the process output y_p.*

Remark 2. *Model-plant mismatch may exist between the controlled process and the internal model. For the sake of clarity, the mismatch is neglected in the analysis of OLP-MPC, which can be handled by the feedback correction (e.g., Eq.(2)).*

As mentioned before, the OLP-MPC can realize that the closed-loop behavior of a system is similar to the open-loop behavior of the controlled process. The similarity is reflected in that the Bode plots of the two differ only

in amplitude-frequency characteristics, but there is no difference in the phase-frequency characteristics. Hence, given a closed-loop system such as Fig.1, an OLP-MPC controller should be designed as

$$\frac{C(z^{-1})G(z^{-1})}{1+C(z^{-1})G(z^{-1})} = \mathcal{K}G(z^{-1})$$
$$\Longrightarrow C(z^{-1}) = \frac{\mathcal{K}}{1-\mathcal{K}G(z^{-1})}$$

where $\mathcal{K}$ is a real-number gain between the closed-loop system and the controlled process. Since the steady-state gain from the setpoint r to the output y should be 1, $\mathcal{K}$ should satisfies

$$\mathcal{K} = \frac{1}{G(1)}$$

where $G(1)$ is the asymptotic output of $G(z^{-1})$. Then the control law is

$$u(z^{-1}) = \frac{\mathcal{K}}{1-\mathcal{K}G(z^{-1})}\left(r(z^{-1}) - y(z^{-1})\right) \tag{8}$$

Remark 3. *Here the plant-model mismatch between the controlled process and the internal model is not considered in* (8)*, which is important in practice. On the basis of control law* (8)*, the OLP-MPC algorithm is given in Algorithm 1 to handle this issue.*

The first three steps in Algorithm 1 correspond to model prediction, feedback correction and rolling optimization of MPC methodology. When the algorithm in initialized, the input $u(z^{-1})$ can be regarded as zero.

Remark 4. *It should be pointed out that with Algorithm 1 there is no performance improvement of the closed-loop system since the phase-frequency characteristics has not been changed. However, it may be useful if the open-loop behaviour are satisfactory, assumed stable, and the algorithm will give offset-free control with minimal coding.*

4. Pole-placement MPC

Through the proposed algorithm in the last section, the behavior of the closed-loop system can be similar to that of the controlled process or the internal model, i.e., the phase-frequency characteristics are consistent. Based on

Algorithm 1 Open-loop poles MPC algorithm (OLP-MPC)

Input: The identified model $G_m(z^{-1}) = \frac{B(z^{-1})}{A(z^{-1})}$, setpoint $r(z^{-1})$.

1: Given the input $u(z^{-1})$, predict the internal model output.

$$y_m(z^{-1}) = G_m(z^{-1})u(z^{-1})$$

2: Correct the controlled process prediction according to the mismatch correction term $d(z^{-1})$:

$$y_p(z^{-1}) = y_m(z^{-1}) + d(z^{-1})$$

3: Calculate the control law.

$$\begin{aligned} u(z^{-1}) &= \frac{\mathcal{K}}{1 - \mathcal{K}G(z^{-1})} \left(r(z^{-1}) - y_p(z^{-1})\right) \\ &= \frac{r(z^{-1}) - d(z^{-1})}{G_m(1)} \end{aligned}$$

4: Go back to step 2.

the OLP-MPC algorithm, this section will discuss how to realize the pole placement of the closed-loop system to achieve faster response. The innovative core idea is to make the behavior of the closed-loop system consistent with that of the internal model. At the same time, the forward channel part of the internal model is the same as the controlled process. First the decomposition of the internal model is introduced. Then the pole-placement-based OLP-MPC (denoted as Pole-Placement MPC, PP-MPC) is elaborated. Some details such as poles design and constraints handling are also discussed.

4.1. Decomposition of the internal model

First, the internal model in Fig.2 is decomposed into two part, as shown in Fig.3.

The main difference between Fig.2 and Fig.3 is the expansion of the internal model. The internal model is decomposed into two series components. For the internal model $G_m(z^{-1})$, $G_{m1}(z^{-1})$ is minimum phase part and with no delay,

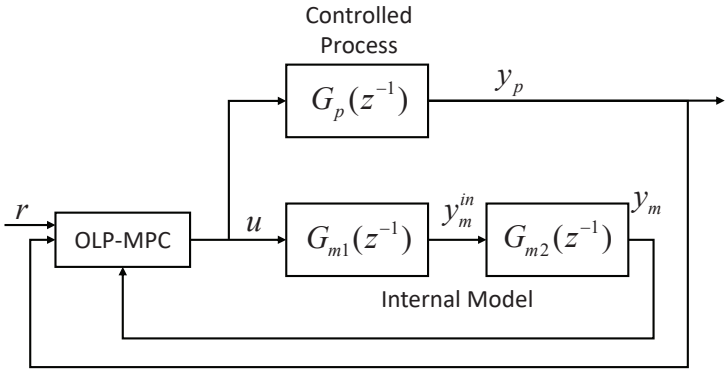

Figure 3: Expand the internal model of the OLP-MPC. $G_m(z^{-1}) = G_{m1}(z^{-1})G_{m2}(z^{-1})$.

and the non-minimum phase zeros part and delay are extracted into $G_{m2}(z^{-1})$.

$$\begin{aligned} G_m(z^{-1}) = \frac{B(z^{-1})}{A(z^{-1})} &= \frac{B_-(z^{-1})B_+(z^{-1})z^{-D}}{A(z)} = G_{m1}(z^{-1})G_{m2}(z^{-1}) \\ G_{m1}(z^{-1}) &\triangleq \frac{B_-(z^{-1})}{A(z)} \\ G_{m2}(z^{-1}) &\triangleq B_+(z^{-1})z^{-D} \end{aligned} \tag{9}$$

The importance of this decomposition is that pole-placement design for $G_{m1}(z^{-1})$ can be done relatively simply, as the difficult dynamics are in $G_{m2}(z^{-1})$. Next subsection will show how to develop a PP-MPC algorithm.

4.2. Pole-placement control structure

This subsection gives the proposed PP-MPC control structure and illustrates how to place the given desirable poles. The core point is that an inner feedback loop will only be placed around $G_{m1}(z^{-1})$, which makes a classical design relatively straightforward. As $G_{m2}(z^{-1})$ is FIR (Finite Impulse Response), it is stable for any convergent input and thus an input giving good asymptotic stability for $G_{m1}(z^{-1})$ will necessarily give equivalent asymptotic stability for $G_m(z)$.

The proposed PP-MPC control structure is now shown in Fig.4, where $K(z^{-1})$ is a feedback function which closes the loop around $G_{m1}(z^{-1})$ to form

the inner loop G_m^{in}. Define a new control law input $v(z^{-1})$ as the target to this inner loop so that:

$$\begin{aligned} u(z^{-1}) &= v(z^{-1}) - K(z^{-1}) \cdot y_m^{in}(z^{-1}) \\ &= v(z^{-1}) \left(1 - K(z^{-1}) G_m^{in}(z^{-1})\right) \end{aligned} \tag{10}$$

Remark 5. *In Algorithm 1, the control law of u is pretty simple, which is mainly associated with the setpoint. Inspired by this, in Fig.4, the control law of v should be simple as well. On the other hand, input v cannot be delivered to $G_p(z^{-1})$ directly, but with the relationship* (10)*, the behavior from system $G_p(z)$ is consistent with that of the internal model since the transfer functions from u to y_m and u to y_p are the same.*

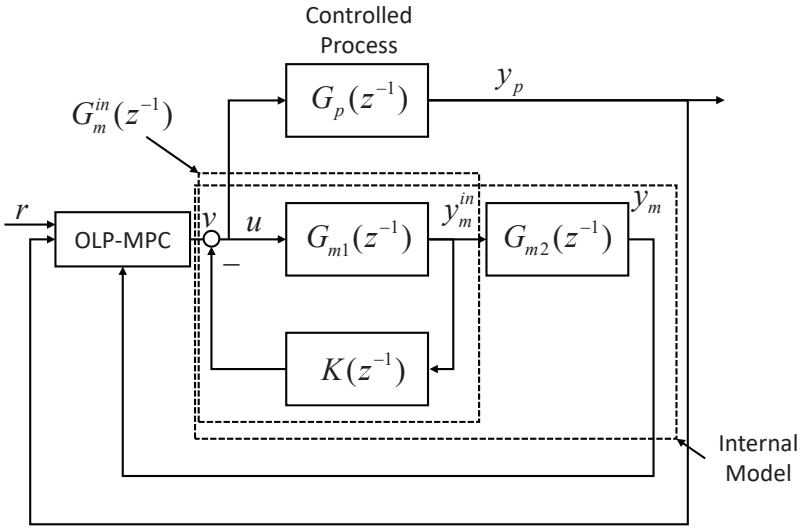

Figure 4: The control structure of PP-MPC. After decomposing the model $G_m(z^{-1})$, a feedback function $K(z^{-1})$ is added around $G_{m1}(z^{-1})$.

Similar to the control law in Algorithm 1, a simple control law form of v can be given by

$$v(z^{-1}) = \frac{r(z^{-1}) - d(z^{-1})}{G_m^{in}(1) G_{m2}(1)} \tag{11}$$

where v is mainly dependent on the setpoint. Given a desired closed-loop be-

havior $G_d(z^{-1})$. The controller should be designed as

$$\begin{aligned}\frac{C(z^{-1})G(z^{-1})}{1+C(z^{-1})G(z^{-1})} &= G_d(z^{-1}) \\ \Longrightarrow C(z^{-1}) &= \frac{G_d(z^{-1})}{G_p(z^{-1})(1-G_d(z^{-1})}\end{aligned}$$

Hence the control law of u is

$$u(z^{-1}) = \frac{G_d(z^{-1})}{G_p(z^{-1})} r(z^{-1}) \tag{12}$$

Combine equations (10) (11) and (12), the control law of v is given by

$$\begin{aligned}v(z^{-1}) &= \frac{r(z^{-1})}{G_m^{in}(1)G_{m2}(1)} \\ &= \frac{1}{1-K(z^{-1})G_m^{in}(z^{-1})} \frac{G_d(z^{-1})}{G_p(z^{-1})} r(z^{-1})\end{aligned} \tag{13}$$

Then the feedback function can be represented by

$$\begin{aligned}K(z^{-1}) &= \frac{G_{m1}(z^{-1})G_{m2}(z^{-1}) - G_d(z^{-1})G_m^{in}(1)G_{m2}(1)}{G_m^{in}(z^{-1})G_{m1}(z^{-1})G_{m2}(z^{-1})} \\ &= \frac{1}{G_m^{in}(z^{-1})} - \frac{G_d(z^{-1})G_m^{in}(1)G_{m2}(1)}{G_m^{in}(z^{-1})G_{m1}(z^{-1})G_{m2}(z^{-1})}\end{aligned} \tag{14}$$

In the $G_m^{in}(z^{-1})$ loop, the transfer function can be denoted by

$$\begin{aligned}G_m^{in}(z^{-1}) &= \frac{G_{m1}(z^{-1})}{1+K(z^{-1})G_{m1}(z^{-1})} \\ \Longrightarrow K(z^{-1}) &= \frac{1}{G_m^{in}(z^{-1})} - \frac{1}{G_{m1}(z^{-1})}\end{aligned}$$

Here it is assumed that the $G_m^{in}(z^{-1})$ can be represented by

$$G_m^{in}(z^{-1}) \triangleq \frac{\sum_{i=1}^{\tilde{m}} \tilde{b}_i z^{-i}}{1+\sum_{i=1}^{\tilde{n}} \tilde{a}_i z^{-i}} \tag{15}$$

And the desired closed-loop behavior $G_d(z^{-1})$ is

$$G_d(z^{-1}) = \frac{G_m^{in}(z^{-1})G_{m2}(z^{-1})}{G_m^{in}(1)G_{m2}(1)} \tag{16}$$

160 By comparing (15) and (16), it can be found that the behavior of $G_d(z^{-1})$ is similar to that of $G_m^{in}(z^{-1})$, hence, the core part of the PP-MPC is the design of $G_m^{in}(z^{-1})$, which will be introduced in the next subsection.

Algorithm 2 Pole-placement MPC algorithm

Input: Take the identified model $G_m(z^{-1}) = \frac{B(z^{-1})}{A(z^{-1})}$, select the setpoint $r(z^{-1})$ and choose desired model $G_m^{in}(z^{-1})$.

1: Decompose the identified model.

$$A, B_-, B_+, D \leftarrow \textsc{Decompose}(A, B)$$

2: Solve for $K(z^{-1})$ as in (14).

3: Given the input $v(z^{-1})$, predict the internal model output:

$$y_m(z^{-1}) = G_m^{in}(z^{-1})G_{m2}(z^{-1})u(z^{-1})$$

4: Correct the controlled process prediction according to the mismatch correction term $d(z^{-1})$:

$$y_p(z^{-1}) = y_m(z^{-1}) + d(z^{-1})$$

5: Calculate the control law

$$v(z^{-1}) = \frac{r(z^{-1}) - d(z^{-1})}{G_m^{in}(1)G_{m2}(1)} \tag{17}$$

6: Determine the control input value of $u(z^{-1})$ from eqn.(10).

7: Go back to step 3.

Function DECOMPOSE(A, B)

1: $D \leftarrow 0$

2: **while** $B(1) == 0$ **do**

3: $\quad D \leftarrow D + 1$

4: $\quad B \leftarrow B[2 : end]$

5: **end while**

6: $roots\_plus \leftarrow$ the roots of $B(z)$ which is greater than one.

7: $roots\_minus \leftarrow$ the roots of $B(z)$ which is less than one.

8: $B_- \leftarrow$ polynomial expansion of $roots\_minus$.

9: $B_+ \leftarrow$ polynomial expansion of $roots\_plus$.

10: **return** A, B_-, B_+, D

end function

According to the assumption that the internal model is stable, it is obvious that it is stable from $u(z^{-1})$ to $y_m(z^{-1})$, and the key point is to explore the stability from $v(z^{-1})$ to $u(z^{-1})$.

Theorem 1. *There exists a feasible $K(z^{-1})$ such that $G_m^{in}(z^{-1})$ can be designed to a stable model:*

Proof. In Fig.4, $G_m^{in}(z^{-1})$ can be presented as (15) Then we have:

$$\begin{aligned} K(z^{-1}) &= \frac{1}{G_m^{in}(z^{-1})} - \frac{1}{G_{m1}(z^{-1})} \\ &= \frac{1+\sum_{i=1}^{\tilde{n}} \tilde{a}_i z^{-i}}{\sum_{i=1}^{\tilde{m}} \tilde{b}_i z^{-i}} - \frac{A(z^{-1})}{B_-(z^{-1})} \end{aligned} \tag{18}$$

By definition $G_m^{in}(z^{-1})$ is stable, so we need only ensure that $K(z^{-1})$ does not incorporate unstable modes into the loop. Computation of $u(z^{-1})$ gives:

$$\begin{aligned} u(z^{-1}) &= \frac{1}{1+K(z^{-1})\frac{B_-(z^{-1})}{A(z^{-1})}} v(z^{-1}) \\ &= \frac{\sum_{i=1}^{\tilde{m}} \tilde{b}_i z^{-i}}{1+\sum_{i=1}^{\tilde{n}} \tilde{a}_i z^{-i}} \frac{A(z^{-1})}{B_-(z^{-1})} v(z^{-1}) \end{aligned} \tag{19}$$

which is obviously stable as by definition $B_-(z^{-1})$ has stable roots and the roots of $1+\sum_{i=1}^{\tilde{n}} \tilde{a}_i z^{-i} = 0$ are also stable by definition. □

The proposed PP-MPC algorithm is the combination of both Algorithm 1 and Theorem 1 together, which is given in Algorithm 2. Theorem 1 allows us to form an inner loop, or internal model $G_m^{in}(z^{-1})$ with desirable poles. If the OLP-MPC algorithm is applied to $G_m^{in}(z^{-1})$, then clearly both y_m and y_p will behave according to those poles.

Remark 6. *Compared to the PFC algorithm, there are no parameters λ and n_y anymore, instead of which the user selects the desired model $G_m^{in}(z^{-1})$. It should be noted that the transient closed-loop behaviour is also influenced by $G_{m2}(z^{-1})$, and this could be taken into account of in the offline stage while assessing different choices for $G_m^{in}(z^{-1})$.*

4.3. Poles design

This subsection introduces the design of the model $G_m^{in}(z^{-1})$. Since the closed-loop poles in $G_d(z^{-1})$ is consistent with the poles of $G_m^{in}(z^{-1})$, the core part is to determine the form of $G_m^{in}(z^{-1})$.

Given a stable S-domain transfer function $G_m(s)$, it can be discretized to a Z-domain transfer function $G_m(z^{-1})$ at a given sampling time. Then $G_m(z^{-1})$ is decomposed into two part as (9). The part $G_{m1}(z^{-1})$ can be designed into $G_m^{in}(z^{-1})$ with proper $K(z^{-1})$. In summary, the main step of placing the desired poles are

- Determine the form of the process model and the desired internal model. If the two models are with the form in S-domain, discretize them. The poles of the desired internal model can be chosen according to Bode plot analyze or other methods. In addition, the desired poles are usually not expected slower than open-loop poles [18]. In our practice, the closed-loop poles that are 3 to 5 times faster than the open-loop poles is a good choice. That is, the real part of the slowest pole should be further away from the imaginary axis, and the imaginary part of the pole should not be too far from the real axis to cause oscillations.

- Calculate the feedback function according to the process model and the desired internal model.

4.4. Constraints handling

Predictive control algorithms are widely applied in industry for their ability to handle constraints. For PP-MPC, a simple method to deal with rate or absolute constraint is adopted instead of solving a linear or quadratic programming problem.

Theorem 2. *All the input constraints on $u(k)$ and the output constraints on $y(k)$ in the future can be converted to equivalent constraints on $v(k)$ such as:*

$$v_{min}(k) \leq v(k) \leq v_{max}(k)$$

Proof. For suitable parameters n_i, d_i, eqn. (19) can be reformulated as:

$$u(z^{-1}) = \frac{n_0 + n_1 z^{-1} + n_2 z^{-2} + \cdots}{1 + d_1 z^{-1} + d_2 z^{-2} + \cdots} v(z^{-1})$$

Assuming $u_{min} \leq u \leq u_{max}$, $\Delta u_{min} \leq \Delta u \leq \Delta u_{max}$ and $y_{min} \leq y \leq y_{max}$, $\Delta y_{min} \leq \Delta y \leq \Delta y_{max}$, one can map these to constraints on $v(k)$ as follows:

$$u_{min} + \Sigma_1 \leq n_0 v(k) \leq u_{max} + \Sigma_1$$

$$\Delta u_{min} + u(k-1) + \Sigma_1 \leq n_0 v(k) \leq \Delta u_{max} + u(k-1) + \Sigma_1$$

$$y_{min} + \Sigma_2 \leq h_i v(k) \leq y_{max} + \Sigma_2$$

$$\Delta y_{min} + y_m(k-1) + \Sigma_2 \leq h_i v(k) \leq \Delta y_{max} + y_m(k-1) + \Sigma_2$$

where $\Sigma_1 = \sum_{i=1} d_i u(k-i) - \sum_{i=1} n_i v(k-i)$, $\Sigma_2 = p_i \underleftarrow{v}(k) + q_i \underleftarrow{y_m}(k+D)$ and h_i, q_i, p_i ($i = 1, 2, ...$ and should be large enough) are prediction parameters relative to the internal model. h_i is the sum of the ith row of H. q_i and p_i are the ith row of Q and P respectively defined in Eq(1). It is implicit that predictions over a suitable constraint horizon are needed for both $y(k)$ and $u(k)$, but this detail is now standard in the MPC literature. □

Remark 7. *It is very important to analyse the stability of the PP-MPC algorithm especially when constraints are considered. Readers can refer to Appendix 2 for the detail. Nevertheless the following remarks are in order:*

- *For the nominal case, infeasibility of the inequalities is caused by changes in the target and thus simple reference governing, that is slowing down target changes, is sufficient to maintain feasibility and thus implicitly also stability.*

- *Guarantees of recursive feasibility and thus stability in the presence of uncertainty and constraints are non-trivial in general and do not exist even for common but far more expensive MPC implementations in practice. However, again simple reference governing is known to be very effective.*

4.5. Computational cost

The low cost of the proposed method is reflected in two ways. Firstly, the proposed method needs not to solve the optimization problem. In the

constrained MPC algorithms, the optimization result is obtained by solving a quadratic programming (QP) problem or a nonlinear programming (NLP) problem. Even if it is an unconstrained optimization problem, it is still necessary to obtain the result through matrix calculation. Secondly, compared with the conventional MPC algorithms, the proposed method is independent of prediction horizon or control horizon in terms of time complexity. While the complexity of conventional MPC is often "cubic in the horizon"[19].

5. Case studies: Applications of PP-MPC on high-order systems

In this section, three cases are studied to illustrate the effectiveness of the PP-MPC algorithm. In the first case, PP-MPC is adopted to control the superheated steam temperature. Steps for constraints handling and design of desired model are described in detail. The second case is the control of an isothermal continuous stirred tank reactor. Compared with other methods, it shows the capability of the proposed method in dealing with nonlinear process and interference. The third case is the tray temperature control case of distillation column, which reflects the ability of the method in dealing with multiple input multiple output system.

5.1. Case I: Superheated steam temperature

This subsection shows how to implement Algorithm 2 taking superheated steam temperature control as an example. A superheated steam temperature model can be described by [20]:

$$G(s) = \frac{1.4}{(40s+1)^5}. \tag{20}$$

whose units are seconds. The dynamic of the superheated steam temperature is slow for its high order and large time constant. The desired model should be faster even with some overshoots, which is given by

$$G_m^{in}(s) = \frac{s+0.5}{s^2+s+2} \tag{21}$$

To get a relative fast closed-loop response, the desired closed-loop poles are selected aa $s_{1,2} = -\frac{1}{2} \pm \frac{\sqrt{7}}{2}i$ in (21), which is with an overshoot and a large attenuation rate. Meanwhile, the order of the desired model is lower compared with (20). With sampling period 0.1 sec, the parameters in Algorithm 2 are given as

$$\begin{cases} A = 1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3} + a_4 z^{-4} + a_5 z^{-5} \\ B_- = b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + b_4 z^{-4} \\ B_+ = 1 + 1.2376 z^{-1} \\ D = 0 \end{cases} \tag{22}$$

where

$a_1 = -0.0916, a_2 = 0.0034, a_3 = -6.1442 \times 10^{-5}, a_4 = 5.6268 \times 10^{-7},$
$a_5 = -2.0612 \times 10^{-9}, b_1 = 5.7259 \times 10^{-9}, b_2 = 5.5344 \times 10^{-10}, b_3 = 6.5312 \times 10^{-12},$
$b_4 = 6.9895 \times 10^{-15}$

and on basis of eqn.(18), the feedback function is:

$$K(z^{-1}) = \frac{N_0 + N_1 z^{-1} + N_2 z^{-2} + N_3 z^{-3} + N_4 z^{-4} + N_5 z^{-5}}{D_0 + D_1 z^{-1} + D_2 z^{-2} + D_3 z^{-3} + D_4 z^{-4} + D_5 z^{-5}} \tag{23}$$

where

$N_0 = 1.7461 \times 10^8, N_1 = -1.5991 \times 10^7, N_2 = 5.8576 \times 10^5, N_3 = -1.0729 \times 10^4,$
$N_4 = 98.2509, N_5 = -0.3599, D_0 = 1, D_1 = -1.7892, D_2 = 0.7237,$
$D_3 = 0.0853, D_4 = 0.0010, D_5 = 1.1043 \times 10^{-6}$

Taking the setpoint change as from 0 to 1, Fig.5 shows the result of this numerical case. At the time 2 sec, the setpoint changes from 0 to 1. The blue line shows the unconstrained closed-loop response which achieves the desired closed-loop poles. Fig.5 also shows the ability of PP-MPC to handle constraints systematically; in this case output constraints ($y \leq 1.1$) are emphasised to make it clear that this is more challenging than simple input saturation; the PP-MPC algorithm incorporates these effectively through Theorem 2.

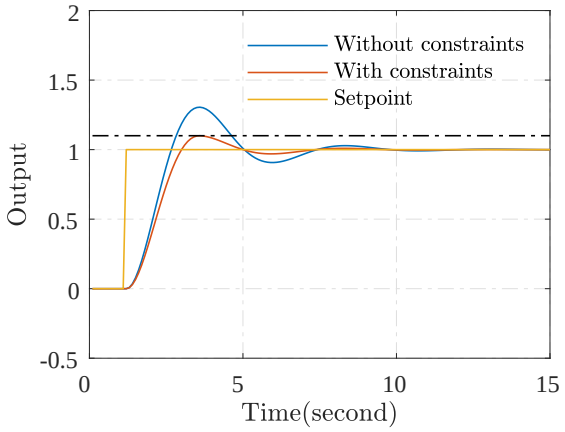

Figure 5: Comparison of constrained and unconstrained output in the superheated steam temperature control case. The output (red line) is limited to 1.1 with a constraint.

5.2. Case II: Continuous stirred tank reactor

In this case, an isothermal continuous stirred tank reactor (CSTR) is studied. As shown in Fig. 6, the isothermal series/parallel Van de Vusse reaction [21, 22] is taking place. Two reactions occur in the reaction scheme:

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C$$
$$A + A \xrightarrow{k_3} D$$

where A is the reactant. The desired product B can be degraded to C, which is a by-product and needs to be suppressed. In the second reaction, D is another by-product. k_i $(i = 1, 2, 3)$ are the reaction rate constants for the two reactions. The dynamic of the system can be described as:

$$\begin{aligned} \frac{dC_A(t)}{dt} &= \frac{F_r(t)}{V}[C_{Ai} - C_A(t)] - k_1 C_A(t) - k_3 C_A^2(t) \\ \frac{dC_B(t)}{dt} &= -\frac{F_r(t)}{V} C_B(t) + k_1 C_A(t) - k_2 C_B(t) \end{aligned} \tag{24}$$

where F_r is the feed flow rate of reactant A, V is the reactor volume which can be regarded as a constant during the reaction. $C_\bullet$ is the concentration of the reactant or product. C_{Ai} is concentration of A in the feed flow. Consequently,

to get desired C_B (the controlled variable), one needs to adjust the feed flow rate F_r to eliminate the disturbance C_{Ai}.

Choose $k_1 = 5/6\ min^{-1}$, $k_2 = 5/3\ min^{-1}$, $k_3 = 1/6\ L \cdot mol^{-1} \cdot min^{-1}$ and $C_{Ai} = 10\ mol \cdot L^{-1}$, $V = 700\ L$. The working point is $C_A = 2.917\ mol \cdot L^{-1}$, $C_B = 1.1\ mol \cdot L^{-1}$, $F_r = 380\ L \cdot min^{-1}$. Then when the input increases 10% on the basis of the input working point, the model identified by [22] is

$$G_m(s) = \frac{0.3199(-0.3520s + 1)}{(0.5619s + 1)(0.3086s + 1)} \tag{25}$$

Noted that the output of the model is the percentage deviation from the output working point.

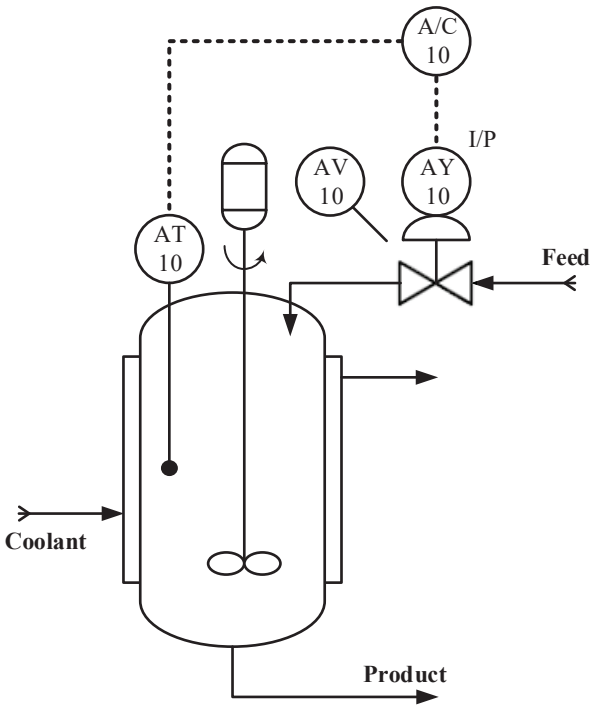

Figure 6: An isothermal continuous stirred tank reactor system.

In general terms, PFC always aims for a first-order reference trajectory which is difficult to achieve when controlling a higher-order system, whereas it will be shown that PP-MPC outperforms PFC in such scenario. In this case, a PID

controller is also used for comparison which is with the following form:

$$G_C(s) = K_c \frac{\tau_I s + 1}{\tau_I} \tag{26}$$

Fig. 7 shows the closed-loop step responses for a change in target when using PID, PFC, PP-MPC and OLP-MPC controllers. The parameters used in the first three controllers are listed in Table.1. To make comparative results, the PID parameters are well tuned on the basis of [23]. λ and n_y are also chosen carefully in PFC. Two poles are used for PP-MPC, namely $s_{1,2} = -1.4920 \pm 1.3542i$.

PID	$K_c = 2.2087,\ \tau_I = 1.3382$
PP-MPC	$\tilde{a}_1 = -1.9702,\ \tilde{a}_2 = 0.9706$
PFC	$\lambda = 0.9851,\ n_y = 100$

Table 1: Tuning parameters for the CSTR case study

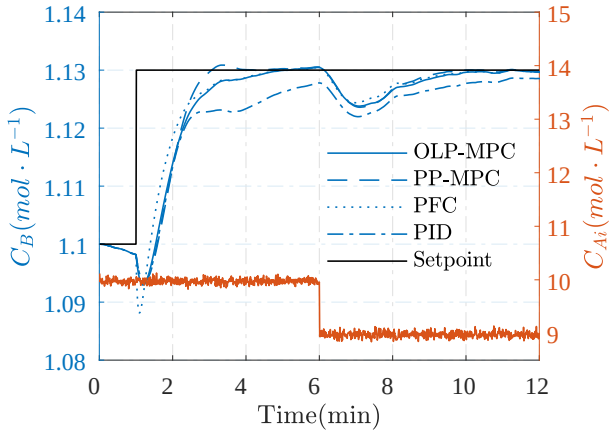

Figure 7: Results with different controllers in the continuous stirred tank reactor control case.

The setpoint changes from 1.1 $mol \cdot L^{-1}$ to 1.13 $mol \cdot L^{-1}$ at the first minute. The controlled output gets close to the setpoint increasingly without apparent overshoot. Due to the prediction property, the convergence time of PFC and PP-MPC is faster than PID (PP-MPC is faster). It should be noted that the coincidence point of PFC should be selected carefully as too short a horizon

can lead to even worse behaviour due to the non-minimum phase characteristic. At the sixth minute, the concentration of A in the feed flow changes from $C_{Ai} = 10\ mol \cdot L^{-1}$ to $C_{Ai} = 9\ mol \cdot L^{-1}$ and this acts like a disturbance; all four controllers can deal with the disturbance. Note that the process is simulated in accordance with process (24) hence model parameters uncertainty and nonlinearity may exist which also shows the robustness of the proposed method.

The inverse part of the responses is also worth discussing. PFC has the deepest inverse response, which is almost inevitable if one wants to get a fast closed-loop response corresponding to a first-order reference. A large coincidence horizon may improve this problem but tuning would default back to the open-loop dynamic of the original system (essentially it becomes the OLP-MPC algorithm). PP-MPC and PID have similar inverse responses, although thereafter PP-MPC shows the best performance and convergence. This is readily comprehensible since PP-MPC uses a pair of mild under-damped complex poles.

5.3. Case III: A distillation column

The distillation is a basic unit operation in the chemical/petrochemical industry[24]. In this subsection, a MIMO process in distillation column control is studied to further evaluate the proposed method. As shown in Fig.8, there are two inputs (reflux flow rate L and vapor boil up flow rate V) and two outputs (temperatures of the tray 21 T_{21} and the temperatures of the tray 7 T_7) in the process[25]. As the vapor boil up flow rate increases, the tray temperature rises, while an increase in the reflux flow rate causes the tray temperature to drop. The system is identified as

$$\begin{bmatrix} T_{21} \\ T_7 \end{bmatrix} = G(s) \begin{bmatrix} V \\ L \end{bmatrix} \tag{27}$$

where

$$G(s) = \begin{bmatrix} G_{11}(s) & G_{12}(s) \\ G_{21}(s) & G_{22}(s) \end{bmatrix} = \begin{bmatrix} \frac{32.63}{(99.6s+1)(0.35s+1)} & \frac{-33.89}{(98.02s+1)(0.42s+1)} \\ \frac{34.84}{(110.5s+1)(0.33s+1)} & \frac{-18.85}{(75.43s+1)(0.3s+1)} \end{bmatrix}$$

The desired poles for the two outputs are placed as

$$G_V(s) = \frac{1}{1000s^2 + 20s + 1}$$
$$G_L(s) = \frac{-1}{1000s^2 + 20s + 1} \tag{28}$$

where the poles are at $s = -0.01 \pm 0.03i$. In order to increase the output at the top of the tower, the temperature of tray 21 can be increased and the temperature of tray 7 can be lowered. The target values of the system output T_{21} and T_7 are set as $T_{21} = 1$ and $T_7 = -1$ with the desired dynamic (28). Fig.9 compares the step response of the open-loop model (27) and desired model (28). The response time of the desired model is faster with a slight overshoot.

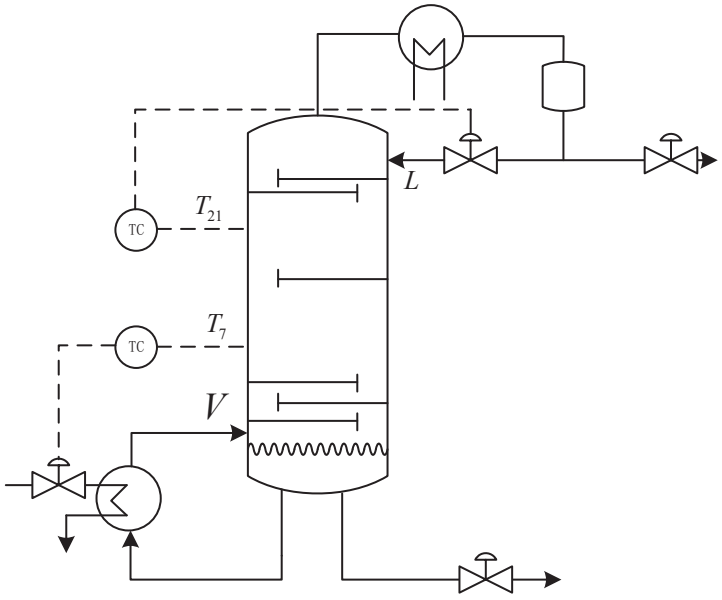

Figure 8: Distillation column control structure. Two inputs and two outputs control relationship is established.

Fig.10 shows the input and output of the system. It is seen that with PP-MPC algorithm, the output dynamic is improved effectively. The temperature of the two trays converge to the target rapidly, and reflux flow rate decreases to increase the output quantity of the distillation column. In Fig.10, two types

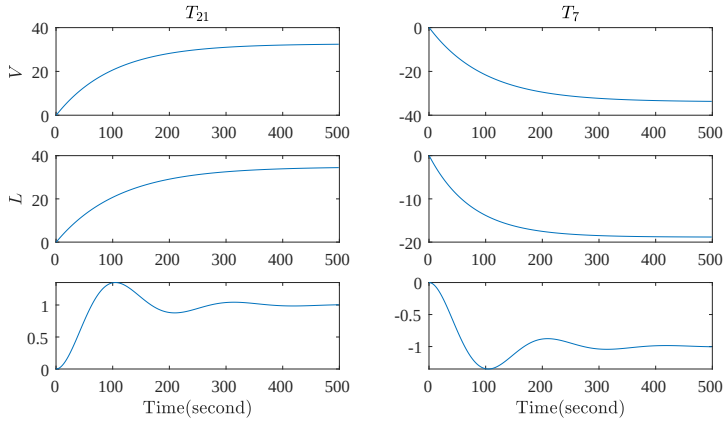

Figure 9: Step response of the distillation column system. The open-loop model in the first two rows, and the desired model in the third row.

control results of the PP-MPC algorithm are displayed: with no model mismatch and with model mismatch. It can be seen that under the no model mismatch condition, the system output perfectly tracks the desired response. Then the measurement of the system output is multiplied by 0.9 to simulate the model mismatch and test the ability of the proposed method on tracking offset-free. In summary, it is illustrated from this case that the proposed method has the potential to implement in MIMO systems.

6. Conclusion

A novel low cost pole-placement MPC Algorithm is proposed and its efficacy is illustrated through some cases. The specific novelty lies in the combination of open-loop predictive control and desired model to obtain two core attributes: (i) the closed-loop poles of the system can be analyzed and (ii) maintain the ability to handle constraints. A further novelty is the structure used in desired model, where a particular decomposition is used to ensure the conditioning and reliability of the pole placement part. The case studies demonstrate that good closed-loop performance is achieved and moreover, the proposed approach retains the simple coding and low implementation cost. It is also demonstrated

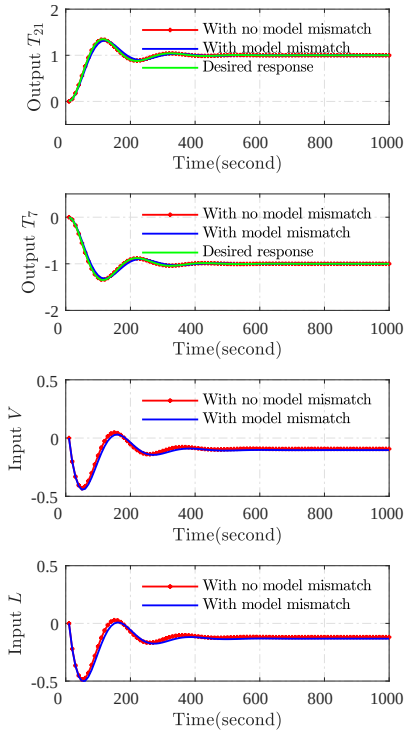

Figure 10: The output and input results of setpoint tracking. Model mismatch is added to show the offset-free property of PP-MPC.

that, being model based, the approach can outperform the obvious competitor of PID.

Future work will be focused on the robust analysis of the proposed PP-MPC algorithm and the extension to both open-loop unstable and MIMO systems as well as exploring systematic approaches for the pole placement part of the design. It would also be interesting to explore subtle changes in the structure/decomposition scheme and the potential for adding simple feedforward to improve fast transient performance.

Acknowledgement

This work is supported by National Key Research and Development Program of China (NO. 2018YFB1702802).

References

[1] F. Borrelli, A. Bemporad, M. Morari, Predictive control for linear and hybrid systems, Cambridge University Press, 2017.

[2] J. A. Rossiter, Model-based predictive control: a practical approach, CRC press, 2017.

[3] J. Richalet, D. O'Donovan, Predictive functional control: principles and industrial applications, Springer Science & Business Media, 2009.

[4] C. R. Cutler, B. L. Ramaker, Dynamic matrix control?? a computer control algorithm, in: joint automatic control conference, no. 17, 1980, p. 72.

[5] D. W. Clarke, C. Mohtadi, P. Tuffs, Generalized predictive controlpart i. the basic algorithm, Automatica 23 (2) (1987) 137–148.

[6] J. A. Rossiter, R. Haber, The effect of coincidence horizon on predictive functional control, Processes 3 (1) (2015) 25–45.

[7] J. A. Rossiter, A priori stability results for pfc, International journal of control 90 (2) (2017) 289–297.

[8] J. A. Rossiter, M. Abdullah, A new paradigm for predictive functional control to enable more consistent tuning, in: 2019 American Control Conference (ACC), IEEE, 2019, pp. 366–371.

[9] X. Hu, H. Zou, J. Tao, F. Gao, Multimodel fractional predictive functional control design with application on an industrial heating furnace, Industrial & Engineering Chemistry Research 57 (42) (2018) 14182–14190.

[10] Z. Zhang, L. Xie, H. Su, Rail pressure controller design of gdi basing on predictive functional control, Control Theory and Technology 17 (2) (2019) 176–182.

[11] J. Rossiter, R. Haber, K. Zabet, Pole-placement predictive functional control for over-damped systems with real poles, ISA transactions 61 (2016) 229–239.

[12] K. Zabet, J. Rossiter, R. Haber, M. Abdullah, Pole-placement predictive functional control for under-damped systems with real numbers algebra, ISA transactions 71 (2017) 403–414.

[13] M. Abdullah, J. A. Rossiter, Utilising laguerre function in predictive functional control to ensure prediction consistency, in: 2016 UKACC 11th International Conference on Control (CONTROL), IEEE, 2016, pp. 1–6.

[14] M. Abdullah, J. Rossiter, R. Haber, Development of constrained predictive functional control using laguerre function based prediction, IFAC-PapersOnLine 50 (1) (2017) 10705–10710.

[15] M. Abdullah, J. A. Rossiter, Using laguerre functions to improve the tuning and performance of predictive functional control, International Journal of Control (2019) 1–13.

[16] Z. Zhang, J. A. Rossiter, L. Xie, H. Su, Predictive functional control for integrator systems, Journal of the Franklin Institute 357 (7) (2020) 4171 – 4186.

[17] M. Aftab, J. Rossiter, Z. Zhang, Predictive functional control for unstable first-order dynamic systems, in: Proceedings of the 14th International Conference on Automatic Control and Soft Computing, Springer, 2020.

[18] P. J. Gawthrop, Linear predictive pole-placement control: practical issues, in: IEEE CONFERENCE ON DECISION AND CONTROL, Vol. 1, Citeseer, 2000, pp. 160–165.

[19] S. Richter, Computational complexity certification of gradient methods for real-time model predictive control, ETH Zurich, 2012.

[20] R. Hỳl, R. Wagnerová, Design and implementation of cascade control structure for superheated steam temperature control, in: 2016 17th International Carpathian Control Conference (ICCC), IEEE, 2016, pp. 253–258.

[21] J. Van de Vusse, Plug-flow type reactor versus tank reactor, Chemical Engineering Science 19 (12) (1964) 994–996.

[22] P. Balaguer, V. Alfaro, O. Arrieta, Second order inverse response process identification from transient step response, ISA transactions 50 (2) (2011) 231–238.

[23] W. L. Luyben, Tuning proportional- integral controllers for processes with both inverse response and deadtime, Industrial & engineering chemistry research 39 (4) (2000) 973–976.

[24] I.-L. CHIEN, B. A. OGUNNAIKE, Modeling and control of a temperature-based high-purity distillation column, Chemical Engineering Communications 158 (1) (1997) 71–105.

[25] A. N. Venkat, Distributed model predictive control: theory and applications, Ph.D. thesis, Citeseer (2006).

[26] J. B. Rawlings, K. R. Muske, The stability of constrained receding horizon control, IEEE transactions on automatic control 38 (10) (1993) 1512–1516.

[27] W.-H. Chen, P. J. Gawthrop, Constrained predictive pole-placement control with linear models, Automatica 42 (4) (2006) 613–618.

Appendix 1

The one step ahead prediction of the system 6 is:

$$y(k+1) = a_1 y(k) + a_2 y(k-1) + b_1 u(k) + b_2 u(k-1). \tag{29}$$

And the two step ahead prediction is:

$$\begin{aligned} y(k+2) &= a_1 y(k+1) + a_2 y(k) + b_1 u(k+1) + b_2 u(k) \\ &= (a_1^2 + a_2) y(k) + a_1 a_2 y(k-1) + (a_1 b_1 + b_1 + b_2) u(k) + a_2 b_2 u(k-1). \end{aligned} \tag{30}$$

If regarding eqn.(29) as a parameter equation:

$$y(k+1) = \alpha_1 y(k) + \beta_1 y(k-1) + \gamma_1 u(k) + \lambda_1 u(k-1) \tag{31}$$

and eqn.(30) as another parameter equation:

$$y(k+2) = \alpha_2 y(k) + \beta_2 y(k-1) + \gamma_2 u(k) + \lambda_2 u(k-1), \tag{32}$$

More generally there exists prediction equations of the form:

$$y(k+n) = \alpha_n y(k) + \beta_n y(k-1) + \gamma_n u(k) + \lambda_n u(k-1), \tag{33}$$

The recurrence relations used to determine the parameters are:

$$\begin{cases} \alpha_{n+2} = a_1 \alpha_{n+1} + a_2 \alpha_n, \alpha_1 = a_1, \alpha_2 = a_1^2 + a_2 \\ \beta_{n+2} = a_1 \beta_{n+1} + a_2 \beta_n, \beta_1 = a_2, \beta_2 = a_1 a_2 \\ \gamma_{n+2} = a_1 \gamma_{n+1} + a_2 \gamma_n + b_1 + b_2, \gamma_1 = b_1, \gamma_2 = a_1 b_1 + b_1 + b_2 \\ \lambda_{n+2} = a_1 \lambda_{n+1} + a_2 \lambda_n, \lambda_1 = b_2, \lambda_2 = a_1 b_2 \end{cases} \tag{34}$$

since, for example:

$$\begin{aligned} y(k+3) =& a_1 y(k+2) + a_2 y(k+1) + b_1 u(k+2) + b_2 u(k+1) \\ =& a_1 \alpha_2 y(k) + a_1 \beta_2 y(k-1) + a_1 \gamma_2 u(k) + a_1 \lambda_2 u(k-1) \\ &+ a_2 \alpha_1 y(k) + a_2 \beta_1 y(k-1) + a_2 \gamma_1 u(k) + a_2 \lambda_1 u(k-1) \\ &+ b_1 u(k+2) + b_2 u(k+1) \end{aligned} \tag{35}$$

In the second order case, the recurrence relation can be solved to get the general term formula of Fibonacci-like sequence.

Appendix 2

Similar to most of the existing MPC algorithms [26, 27], the regulation problem is discussed in this section. A stable model (15) can be converted to a state space model as follows:

$$\begin{cases} x(k+1) = Ax(k) + Bv(k) \\ y(k) = Cx(k) \end{cases} \tag{36}$$

According to eqn.(17), the objective function of PP-MPC can be described as:

$$J = \min_{v(k)} \|Cx(\infty)\|^2 \tag{37}$$

Considering input and output(or states) constraints:

$$\begin{aligned} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} v(k) \leq \begin{bmatrix} v_{max} \\ -v_{min} \end{bmatrix}, k = 0 \\ \begin{bmatrix} C & 0 \\ 0 & -C \end{bmatrix} x(k) \leq \begin{bmatrix} y_{max} \\ -y_{min} \end{bmatrix}, k = 1, 2, ..., \end{aligned} \tag{38}$$

for a stable model, the input constraints are always feasible as long as $v(k)$ is in the finite set, while the state constraints may be infeasible, but they can be converted into a feasible set by adjusting the horizon according to [26]. Then the following result is stated.

Theorem 3. *In the nominal case, $x(k) = 0$ is an stable solution of the PP-MPC controller with objective function* (37) *and feasible constraints* (38) *for every x_0.*

Proof. Since the constraints are always feasible, once all constraints are inactive, the value of the objective function at k is

$$J_k = x^T(\infty) C^T C x(\infty) \tag{39}$$

where $x(\infty) = A^\infty x_0 + \frac{I - A^\infty}{I - A} B v(k) = \frac{B}{I - A} v(k)$.

At $k + 1$, the value of the objective function is

$$J_{k+1} = x^T(\infty + 1) C^T C x(\infty + 1) \tag{40}$$

where $x(\infty+1) = Ax(\infty) + Bv(k+1) = \frac{B}{I-A}v(k+1)$. In the nominal case, $v(k)$ equals to $v(k+1)$ in PP-MPC hence J_{k+1} is no worse than J_k. At the same time, to minimize the objective function, $v(k)$ should be zero, which means x_k will converge to zero according to the stable assumption and give the asymptotic stability. □