A novel approach to PFC for nonlinear systems

John Anthony Rossiter [a] Muhammad Saleheen Aftab [a]
George Panoutsos [a] Oscar Gonzalez-Villarreal [b]

[a] *Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, S1 3JD, UK (e-mail: j.a.rossiter@sheffield.ac.uk, msaftab1@sheffield.ac.uk,g.panoutsos@sheffield.ac.uk)*

[b] *School of Aerospace, Transport and Manufacturing Centre for Autonomous and Cyber-Physical Systems, Cranfield University, UK, (email: Oscar.Gonzalez-Villarreal@cranfield.ac.uk)*

Abstract

This paper proposes a computationally efficient predictive control law for non-linear systems, that is one that can easily be coded and implemented on low cost hardware. Moreover, it has a secondary core benefit that the core tuning parameter reduces to a single choice which is: how much faster than open-loop would you like the closed-loop to converge? Conceptually the approach builds on the PFC approach but proposes a very different type of coincidence condition which removes the lag associated to the conventional approach. Simulations demonstrate that for some non-linear systems this is a cheap and simple way of ensuring effective feedback, with constraint handling. [1]

Key words: Predictive functional control, computational efficiency, transparent tuning, nonlinear systems.

1 Introduction

Model predictive control (MPC) [1] is very popular in both the industrial [2,3] and academic communities [4–6]. This is because it makes good intuitive sense combined with delivering reliable results for MIMO (multi-input-multi-output) systems and managed constraint handling. Moreover, although rarely

[1] This is a slightly extended and corrected version of the paper which appeared at the European Control Conference, 2022.

discussed carefully [7], it also has the potential to handle future target information systematically.

Nevertheless, despite the popularity and effectiveness, MPC is still not widely deployed on low level loops where PID continues to dominate. This is as expected. PID tuning [8, 9] is simple enough to be handled without recourse to expensive consultants and moreover, for many practical feedback loops, delivers performance that is adequate. Of course, in addition and critically, PID is much cheaper to purchase, code and implement than MPC in general and thus there needs to be a significant potential benefit before a more expensive alternative would be considered.

This paper focuses on one notable exception to the above observations. There are some SISO (single-input-single-ouput) loops where a simple PID implementation does not deliver adequate performance, perhaps due to challenging dynamics or perhaps due to the need for constraint handling. In such a case, a cheap MPC approach would be competitive in both price and complexity and indeed this is what has been noticed by PFC (predictive functional control) vendors [10, 11] over many years. More specifically:

(1) Being model based, PFC is able, in principle, to exploit model information more systematically than PID and thus improve closed-loop behaviour.
(2) Being prediction based, again in principle, PFC can handle constraints systematically.

Nevertheless, the reader will note the use of words *in principle* to clarify the above statements. A large number of recent works have investigated the tuning [12–15] and constraint handling [6, 17] of PFC and made a number of useful observations and contributions:

- The original PFC algorithm is effective with processes having over-damped behaviour, but tuning is much more difficult with other dynamics.
- Recent work has suggested a number of modified PFC algorithms which are more reliable, consistent and enable better links between the tuning parameters and behaviour.
- The constraint handling in the original PFC algorithm was more akin to approaches used in PID and thus suboptimal at best. Using predictions more systematically is straightforward and enables far better results while still requiring no optimisation.

One of the weaknesses in conventional PFC is the use, in predictions, of a fixed future input whereas it is well known (e.g. [15]) that more nuanced parameterisations of the future input sequence are helpful. A very recent work [18] demonstrated how a very simple PFC algorithm could combine steady-state estimates with a simple exponential parameterisation to give intuitive and effective tuning. This work, for now considered only the linear case whereas,

the original creators of PFC saw one huge advantage of the simplicity of the algorithm being in its potential usage with non-linear systems. The reader may note that nonlinear MPC (NLMPC) in the literature is largely both complex and computationally demanding [19]. Hence, the core contribution of this paper is to demonstrate how this recent new PFC approach [18] can be adapted to the nonlinear case and implemented with almost negligible computing and complexity, certainly when compared to more conventional NLMPC algorithms [19].

The paper is organised as follows. Section 2 gives core background on both PFC and the recent proposed algorithm. Section 3 shows how this algorithm can be modified for the nonlinear case. Section 4 presents a case study on a mixing tank with an endothermic reaction and the paper finishes with some numerical results and conclusions.

2 Background on PFC

2.1 System definition

The non-linear system will be taken to be of the form:

$$\dot{x} = f(x, u) \tag{1}$$

with state x and input u (dimensions n_x, n_u respectively) and $f(.)$ is differentiable. This paper assumes this model can be approximated, at individual sample times, by a discrete linear time varying state-space model:

$$x_{k+1} = A_k x_k + B_k u_k \tag{2}$$

More discussion of the linearisation and the use of deviation variables is in section 3.

In practice model (1) is an approximation, so we need to allow for some uncertainty. Here we use standard practice in the literature and define the true process state to be x_p and thus the error term x_e, at each sample, is given as:

$$x_{e,k} = x_{p,k} - x_k \tag{3}$$

As is standard practice in the MPC literature, the error term is used to ensure unbiased prediction and offset free tracking and caters for both parameter uncertainty and disturbances. It assumes that the relevant true state $x_{p,k}$ at the current sample can be measured.

2.2 System prediction

Prediction is well known [6] so details are omitted here. It is sufficient that the reader recognises that with LTV model (2), or indeed similar models, one can easily deduce n-step ahead output predictions as follows, for suitable H, P, L (dimensions implicit from the context).

$$\underset{\rightarrow k+1|k}{x} = H \underset{\rightarrow k}{u} + Px_k + Ld_k \tag{4}$$

where $d_k = x_{e,k}$ and

$$\underset{\rightarrow k}{u} = \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+n-1} \end{bmatrix} \; ; \; \underset{\rightarrow k+1|k}{x} = \begin{bmatrix} x_{k+1|k} \\ x_{k+2|k} \\ \vdots \\ x_{k+n|k} \end{bmatrix}$$

and L is a vector of ones. In the non-linear case, H, P will be time varying and need to be updated every sample, as seen in section 3.

2.3 Conventional PFC control law

This is presented for completeness only and thus very briefly. Conventional PFC is based on the premise of matching the output prediction to a first order response with a given time constant. Hence, assuming the steady-state target is a constant R, define define a target trajectory during transients $r_{k+i|k}$ as:

$$r_{k+i|k} = (1 - \lambda^i)R + \lambda^i x_{p,k}, \quad i = 1, 2, \cdots \tag{5}$$

The PFC control law is determined by ensuring that $x_{k+n|k} = r_{k+n|k}$ and thus, in effect, substitution of (4) into (5) and solving for the degree of freedom which typically is the future value of the input (assumed constant).

However, as mentioned in the introduction [12, 14, 18], this algorithm often fails to give reliable behaviour in that the main tuning parameter λ is often ineffective and indeed, for some open-loop dynamics it is difficult to gain satisfactory beahviour. Thus, some simple alternatives have been proposed and hereafter we introduce one of these.

2.4 Open-loop dynamics PFC (OL)

The simplest predictive algorithm is one which makes no attempt to change the dynamics and focuses solely on ensuring offset free tracking. Such an algorithm is summarised as:

$$u_k = E[u_{ss}] \tag{6}$$

where $E[u_{ss}]$ is the expected steady-state input.

This algorithm gives a useful benchmark for more computationally demanding algorithms, and is especially useful when the open-loop dynamics are benign because it gives a very simple and effective control law. Hence, one would embellish this control law if and only if one wanted faster settling times or a slower change of the input.

Remark 1 *This control law is very simple to code and implement as no detailed prediction is needed, rather just a mechanism to estimate u_{ss}. Thus it provides a route to computationally efficient control of non-linear systems*

2.5 Speeding up OL PFC with exponential input parameterisations

In order to speed up the response it is necessary to over actuate during transients. A simple over-actuation strategy [17, 18] is to parameterise the future inputs as follows:

$$u_k = u_{ss} + \lambda^k \eta; \quad \{0 < \lambda < 1\} \Rightarrow \lim_{k\to\infty} u_k = u_{ss} \tag{7}$$

This has a single degree of freedom (d.o.f.), that is η and thus is amenable to simple optimisation. The parameter λ should be chosen sympathetically with the open-loop dynamics and desired closed-loop dynamics, that is, to converge in a roughly equivalent period; typically chosen the same as in (5).

The selection of η is critical, and hence it was proposed [16] to ensure the associated predictions converge a factor S faster than those associated to the use of (6) alone. The conceptual steps are summarised next.

Algorithm 1 *PFC algorithm to speed up predicted convergence by a factor of S.*

(1) Determine the n-step ahead error E_o between the prediction and target using control law (6).
(2) Determine the n-step ahead error $E_\eta(\eta)$ between the prediction and target using control law (7). *This depends on η.*

(3) Choose η such that $SE_\eta(\eta) = E_o$ where S is a design speed-up factor to be selected.

The algorithm is presented conceptually because, in the non-linear case there will not be fixed algebraic computations for the terms E_o or indeed $E_\eta(\eta)$ and these will need to computed online each sample. However, it is critical to note that the d.o.f. η is a single variable and thus easy to determine efficiently; this will be evident in the numerical examples shown later.

A further important observation is that the tuning is now based on a simple intuitive statement: how much faster than open-loop dynamics would you like to be?

Remark 2 *In the linear case, because explicit and fixed algebraic relationships are possible, it is possible to make the tuning even more precise as discussed in [16]. Here we are extending and applying the concept to the non-linear case where relationships are time varying, and thus those additional steps are not considered for now. Of specific interest one should note that the actual closed-loop speed-up achieved will be different to the ratio of the prediction errors $SE_\eta(\eta) = E_o$ so some offline analysis will be needed.*

2.6 Constraint handling

One can incorporate constraint handling into Algorithm 1 in a systematic and computationally simple way by comparing system predictions against constraints for a sufficiently large horizon; this is standard in the literature [6, 18, 20].

$$\begin{aligned} \underline{u} \ &\leq \mathbf{u}_k \leq \ \overline{u} \\ \underline{\Delta u} &\leq \Delta \mathbf{u}_k \leq \Delta \overline{u} \\ \underline{x} \ &\leq \mathbf{y}_k \leq \ \overline{x} \end{aligned} \tag{8}$$

Critically it is noted that as the predictions have a single d.o.f. η, the selection of η can be determined using a simple *for loop* and thus done very efficiently. One might also note that with (7) the maximum input and input rate will occur at the first or second sample, and thus the number of inequalities to be checked for the input constraints is very small.

Remark 3 *Constraints limit the input amplitudes available and thus will also impact on the speed-up achievable in some scenarios, especially with large changes in target.*

2.7 Summary of proposed algorithm

This section has summarised the core conceptual steps and algebra needed to implement the proposed algorithm 1. The user needs to define the following design parameters.

(1) What is the prediction horizon n? Good practice [12] suggests something like 2 time constants.
(2) What is the speed up factor S? Clearly this depends entirely on what the user wants but we would not expect much bigger than 2-3 or significant over actuation is inevitable and this is rarely implementable in practice.
(3) The parameter λ used in (7) is needed. Typically this should be close to the target closed-loop pole and partially overlaps with the choice of n.

Having defined the core parameters, the remaining steps are linked to computation of the expected errors which is discussed in the following sections.

The reader should be reminded however that a core requirement for the efficacy of the proposed approach is that the open-loop behaviour is broadly acceptable (that is almost meets the performance requirements) so can be used as a valid benchmark.

3 Background on linearisation and prediction with non-linear models

In NLMPC it is necessary to form predictions for a non-linear model. As PFC, by design, is intended to be simple, here we take a very simple approach to this process, accepting that more accurate but also more demanding numerical integration approaches are possible.

3.1 Linearisation about a trajectory

Hence, we use superposition to find predictions by separating the nominal trajectory (x_k, u_k) from the deviations part. It is implicit hereafter that the nominal or baseline trajectory is that associated to input prediction (6).

(1) Simple difference equations are used to simulate the non-linear model and thus to form a baseline prediction based on some assumed future input. Let these values be: x_k, u_k for states x_k and inputs u_k and k the sample number.

(2) The model is linearised about all points x_k, u_k on the baseline prediction to form state-space models of the form:

$$\delta \dot{x}_k = A_k \delta x_k + B_k \delta u_k \tag{9}$$

where $\delta x_k = \hat{x}_k - x_k, \delta u_k = \hat{u}_k - u_k$ are deviations relative to the baseline prediction and A_k, B_k are the linearised model parameters at the kth sample of the baseline prediction. The full predicted state and input values are $\hat{x}_k = x_k + \delta x_k, \hat{u}_k = u_k + \delta u_k$.

Remark 4 *In order to derive the matrices A_k, B_k, we need to undertake partial differentiation of a model which is based on the first derivative, for example, assume that:*

$$\dot{\hat{x}} = f(\hat{x}, \hat{u}) \approx f(x, u) + \frac{\partial f}{\partial x}\delta x + \frac{\partial f}{\partial u}\delta u \tag{10}$$

$$\dot{\hat{x}} = f(\hat{x}, \hat{u}); \quad \dot{\hat{x}} = \dot{x} + \dot{\delta x} = f(x, u) + \dot{\delta x} \tag{11}$$

$$\delta \dot{x} = \frac{\partial f}{\partial x}\delta x_k + \frac{\partial f}{\partial u}\delta u_k = A_k \delta x_k + B_k \delta u_k \tag{12}$$

3.2 Prediction using deviation variables

Once one has determined the models (9) for a notional trajectory, one can easily determine the impact of small deviations in the input, that is $\delta u_k \neq 0$. Predictions can be found by recursive use of (9) as follows:

$$\delta \dot{x}_k \approx \frac{\delta x_{k+1}}{\delta t} = A_k \delta x_k + B_k \delta u_k \tag{13}$$

Summarising one deduces (for suitable period δt):

$$\begin{aligned}
\delta x_{k+1} &\approx [A_k \delta x_k + B_k \delta u_k]\delta t \\
\delta x_{k+2} &\approx [A_{k+1} \delta x_{k+1} + B_{k+1} \delta u_{k+1}]\delta t \\
\delta x_{k+3} &\approx [A_{k+2} \delta x_{k+2} + B_{k+2} \delta u_{k+2}]\delta t \\
&\vdots
\end{aligned} \tag{14}$$

Next, making substitutions and assuming that $\delta x_k = 0$:

$$
\begin{aligned}
\delta x_{k+1} &\approx [B_k \delta u_k]\delta t \\
\delta x_{k+2} &\approx [A_{k+1} B_k \delta u_k \delta t + B_{k+1}\delta u_{k+1}]\delta t \\
\delta x_{k+3} &\approx [A_{k+2}[A_{k+1} B_k \delta u_k \delta t + B_{k+1}\delta u_{k+1}]\delta t \\
&\qquad + B_{k+2}\delta u_{k+2}]\delta t \\
&\vdots
\end{aligned}
\tag{15}
$$

3.3 Prediction with input parameterised using an exponential

The predictions of (15) are somewhat clumsy to use, but in the context of the PFC algorithm to be used here, the predicted input (7) has been parameterised as follows:

$$
\delta u_{k+i} = \lambda^i \eta \tag{16}
$$

for a given λ and η the d.o.f. to be selected online. Substitute (16) into (15) then:

$$
\begin{aligned}
\delta x_{k+1} &= \underbrace{B_k \delta t}_{\alpha_k}\ \eta \\
\delta x_{k+2} &= \underbrace{[A_{k+1}\alpha_k + B_{k+1}\lambda]\delta t}_{\alpha_{k+1}}\ \eta \\
\delta x_{k+3} &= \underbrace{[A_{k+2}\alpha_{k+1} + B_{k+2}\lambda^2]\delta t}_{\alpha_{k+2}}\ \eta \\
&\vdots \\
\delta x_{k+n+1} &= \underbrace{[A_{k+n}\alpha_{k+n-1} + B_{k+n}\lambda^n]\delta t}_{\alpha_{k+n}}\ \eta
\end{aligned}
\tag{17}
$$

It is noted that the main computation here is the simple recursion of:

$$
\alpha_{k+n} = [A_{k+n}\alpha_{k+n-1} + B_{k+n}\lambda^n]\delta t \tag{18}
$$

Algorithm 2 *Predictions for the non-linear model $\dot{x} = f(x,u)$ with input parameterisation* (16) *are computed as follows:*

(1) Estimate the required steady-state input u_{ss} and simulate the model $\dot{x} = f(x,u)$ forward (using numerical integration) over the required horizon using $u_k = u_{ss}, \forall k > 0$.

(2) For the nominal trajectory (x_k, u_k) determined in step 1, form the matrices A_k, B_k at every sample using:

$$
A_k = \frac{\partial f}{\partial x}; \quad B_k = \frac{\partial f}{\partial u}
$$

(3) Use recursion (17) *to determine* α_{k+n} *for the required horizon* n*. The* n*-step ahead predictions for the state are now given as:*

$$\hat{x}_{k+n} = x_{k+n} + \alpha_{k+n}\eta \tag{19}$$

Remark 5 *It is implicit from the use of first order Taylor series and simple difference equations for the numerical integration that the trajectories do not deviate a long way from the baseline. If they do, then the approximation errors would grow and could impact on behaviour. This means the sample period* δt *should be small enough.*

We can define the final algorithm more precisely.

Algorithm 3 *The PFC algorithm 1 can be combined with Algorithm 2 as follows.*

(1) Find the baseline trajectory (x_k, u_k) *using* (6) *and also the associated state space matrices* (12) *and prediction* (19)*.*
(2) Determine the n-step ahead error $E_o = R - x_{k+n}$*.*
(3) Determine the n-step ahead error $E_\eta(\eta) = R - \hat{x}_{k+n}$*.*
(4) Choose η *such that* $SE\eta(\eta) = E_o$*:*

$$S[R - x_{k+n} - \alpha_{k+n}\eta] = R - x_{k+n} \tag{20}$$

(5) The control value to be implemented at the current sample is: $u_k = u_{ss} + \lambda^0\eta$*.*

4 Description of case study

This section describes a simple mixing tank with an endothermic reaction. Chemical A is produced by a reaction in the tank, but this reaction is endothermic thus cooling down the tank contents. The rate of reaction is also temperature dependent (the main non-linear characteristic), so to maximise the reaction rate, the temperature needs to be maintained and thus heat must be supplied. Consequently, the tank can be described by two equations, one for the concentration and a second for the temperature. The objective is to control the concentration (output C_A) by manipulation of supplied heating (input W).

4.1 Core model equations

The concentration model depends on the flow rates into and out of the tank (assumed equal) and the reaction rate:

$$V\frac{dC_A}{dt} = \gamma C_A V e^{0.05(T-T_i)} + F(C_{A0} - C_A) \tag{21}$$

where V is tank volume, F is the flow rate, T is the temperature in the tank (assume well mixed), T_i is the temperature of the in flow, C_{A0} is the concentration of the inflow and C_A is the concentration in the tank. The variable γ is linked to the reaction rate.

The basic heat equation is:

$$V\rho C_p\frac{dT}{dt} = F\rho C_p(T_i - T) - \beta[\gamma C_A V e^{0.05(T-T_i)}] + W \tag{22}$$

where ρ is fluid density, C_p is fluid heat capacity, W the heat supply and β a variable linked to the rate of reaction and thus how much heat is absorbed by the reaction.

For convenience, the model equations (21),(22) can be re-arranged as follows.

$$\frac{dC_A}{dt} = \gamma C_A e^{0.05(T-T_i)} + \frac{F}{V}(C_{A0} - C_A) \tag{23}$$

$$\frac{dT}{dt} = \frac{F}{V}(T_i - T) - \beta[\frac{\gamma}{\rho C_p}C_A e^{0.05(T-T_i)}] + \frac{W}{V\rho C_p} \tag{24}$$

For the purposes of this paper the following values were used: $\rho = 10^3 kgm^{-3}$, $C_p = 4000 J/kgdeg$, $V = 5m^3$, $F = 0.01m^3s^{-1}$, $\beta = 10^7 deg$, $\gamma = 0.005s^{-1}, T_i = 20deg$. It is also noted that the inlet concentration and inlet temperature are not considered to be degrees of freedom in this paper.

4.2 Steady-state estimates

For the proposed algorithm, we need to determine an estimate of the steady-state, assuming that the provided heating is constant, that is $W = W_{ss}$. Moreover, assume that the required steady-state concentration is known as $C_{A,ss}$. There will be an implied steady-state temperature T_{ss}.

A steady-state exists if the derivatives in (23),(24) are zero:

$$0 = \gamma C_{A,ss} e^{0.05(T_{ss}-T_i)} + \frac{F}{V}(C_{A0} - C_{A,ss}) \tag{25}$$

$$0 = \frac{F}{V}(T_{ss} - T_i) - \beta[\frac{\gamma}{\rho C_p} C_{A,ss} e^{0.05(T_{ss}-T_i)}] + \frac{W_{ss}}{V\rho C_p} \tag{26}$$

Using (25) to solve for the steady-state temperature gives:

$$e^{0.05(T_{ss}-T_i)} = -\frac{F}{V}\frac{(C_{A0} - C_{A,ss})}{\gamma C_{A,ss}} \tag{27}$$

$$(T_{ss} - T_i) = 20 \log\left(\frac{F}{V}\frac{(C_{A,ss} - C_{A0})}{\gamma C_{A,ss}}\right) \tag{28}$$

Now we can use (26) and (28) to find the required power input to maintain this temperature.

$$\beta[\gamma V C_{A,ss} e^{0.05(T_{ss}-T_i)}] - F\rho C_p(T_{ss} - T_i) = W_{ss} \tag{29}$$

4.3 Linearisation of case study

It is clear that both model equations (23), (24) take the form:

$$\dot{x} = f(x, u); \quad x = \begin{bmatrix} C_A \\ T \end{bmatrix}; \quad u = W \tag{30}$$

Hence we can linearise as in section 3. The partial derivatives can be computed as follows:

$$\frac{\partial}{\partial C_A}(\dot{C}_A) = \gamma e^{0.05(T-T_i)} - \frac{F}{V} \tag{31}$$

$$\frac{\partial}{\partial T}(\dot{C}_A) = 0.05\gamma C_A e^{0.05(T-T_i)} \tag{32}$$

$$\frac{\partial}{\partial C_A}(\dot{T}) = -\beta[\frac{\gamma}{\rho C_p} e^{0.05(T-T_i)}] \tag{33}$$

$$\frac{\partial}{\partial T}(\dot{T}) = -\frac{F}{V} - 0.05\beta[\frac{\gamma}{\rho C_p} C_A e^{0.05(T-T_i)}] \tag{34}$$

The corresponding linearised state-space model is given as:

$$A_k = \begin{bmatrix} \frac{\partial}{\partial C_A}(\dot{C}_A) & \frac{\partial}{\partial T}(\dot{C}_A) \\ \frac{\partial}{\partial C_A}(\dot{T}) & \frac{\partial}{\partial T}(\dot{T}) \end{bmatrix}; \quad B_k = \begin{bmatrix} 0 \\ \frac{1}{V\rho C_p} \end{bmatrix} \tag{35}$$

In summary, the parameters needed for (17) depend upon the current values of C_A, T as evident from (31)-(34) and thus can easily and quickly be updated

with the values in (x_k, u_k) as required. Consequently the prediction equations outlined in Algorithm 2 and used in (20) of Algorithm 3 can be determined.

5 Simulation results

This section demonstrates the efficacy of the proposed Algorithm 3 on the case study given in section 4. A core selling point is the intuitive nature of the tuning whereby one can request performance as a relative measure compared against open-loop behaviour, the so called speed up factor. A second selling point is the computational simplicity; the main computing requirement is the recursion in (17) which, in terms of modern computing, is not significant.

This section will present results for the nominal case and also with significant parameter uncertainty to demonstrate that, as expected with most MPC approaches, the algorithm is robust to some uncertainty. We also include some comparisons with a conventional NMPC approach to highlight some of the differences.

5.1 Open-loop behaviour

In order to form a benchmark, this section begins by demonstrating the performance achieveable with control law (6) which ensures offset free tracking with open-loop dynamics. The corresponding behaviour is shown in Figure 1 where it is clear that:

- The settling time is around 2000 sec.
- There is no offset in C_A.
- The closed-loop input signal W is constant.

To demonstrate the impact of uncertainty, the true plant parameters are changed slightly from those in subsection 4.1 to: $V = 4.9m^3,\ F = 0.012m^3s^{-1},\ \beta = 1.1 \times 10^7 deg,\ , T_i = 21deg.$ The corresponding simulation is given in Figure 2. Unsurprisingly the behaviour is slightly different, but again there is no offset in the steady-state but now the feedback takes a while to determine the correct steady-state value for W which slows down the overall settling time.

5.2 Speeding up behaviour

Next we deploy Algorithm 3 and use a target speed up of a factor of $\beta = 2$ with $n = 15, \lambda = 0.9$ and sampling period of $5s$. The corresponding behaviour

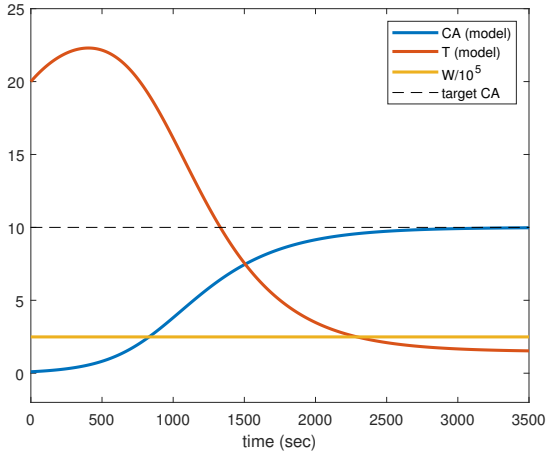

Fig. 1. Closed-loop responses for the nominal case using control law (6).

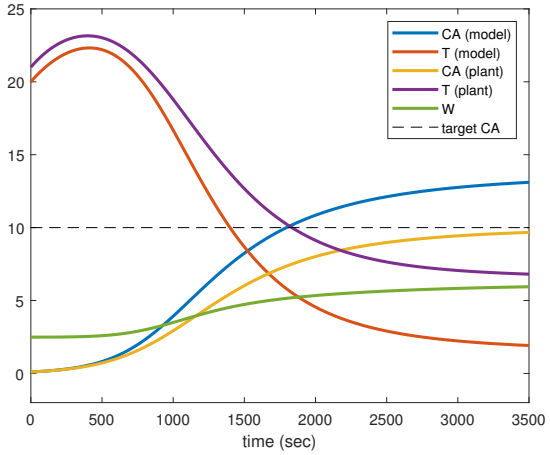

Fig. 2. Closed-loop responses for the robust case using control law (6).

for the nominal case is shown in Figure 3 alongside the results for $\beta = 1$ where it is clear that:

- The Settling time is closer to 1000 sec and thus nearly twice as fast as in Figure 1.
- There is no offset.
- The closed-loop input signal is very aggressive (off the scale) and indeed the transient temperature has risen by nearly 20 degrees to facilitate the faster rise in the reaction rate.

Similar observations arise in the uncertain case as seen in Figure 4. The speed up is achieved as requested, alongside offset free tracking, but at the expense of aggressive heating during transients.

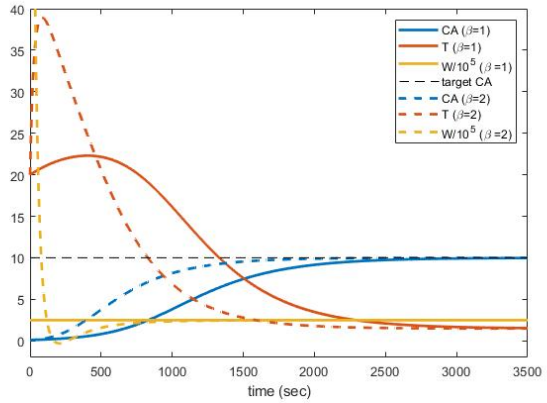

Fig. 3. Closed-loop responses for the nominal case using control law (20) alongside the open-loop behaviour.

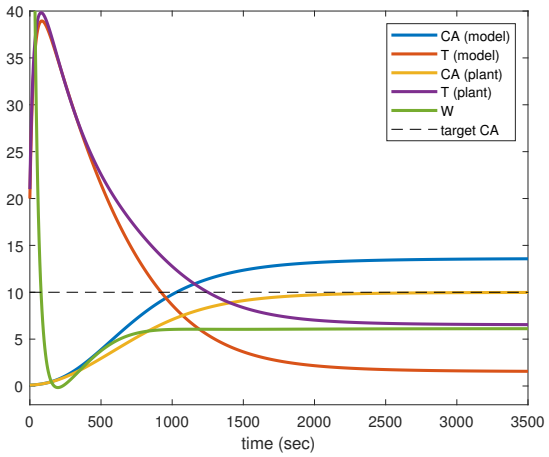

Fig. 4. Closed-loop responses for the uncertain case using control law (6).

5.3 Constraint handling

For completeness, Figure 5 gives a simulation for the uncertain case and with some input constraint handling to demonstrate that this is straightforward to implement. Here we implement a reasonable upper limit on the heating available to be not significantly bigger than the steady-state requirements. Unsurprisingly this results in a slight slow down in performance, that is, the target speed up of a factor of 2 is not achievable in this case.

5.4 Off-the shelf NMPC

For completeness we illustrate the differences with a much more expensive and complicated off-the-shelf NMPC approach in Figure 6. In this case the

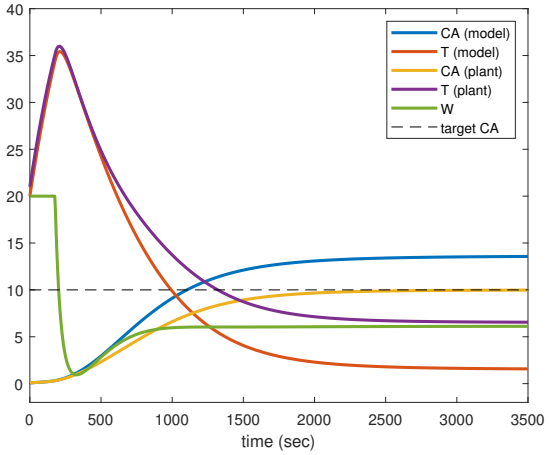

Fig. 5. Closed-loop responses for the uncertain case using control law (6) and with input saturation.

user is able to change the performance index weights to achieve different performances, so we show how simple changes to the input weighting lead to different closed-loop behaviours. Nevertheless, what is most interesting here is that: i) the overall speed of response and smoothness of the output behaviour is similar to PFC and ii) NMPC is much less aggressive in its use of the heating and indeed more flexible in general to tune and trade-off the input and output behaviour.

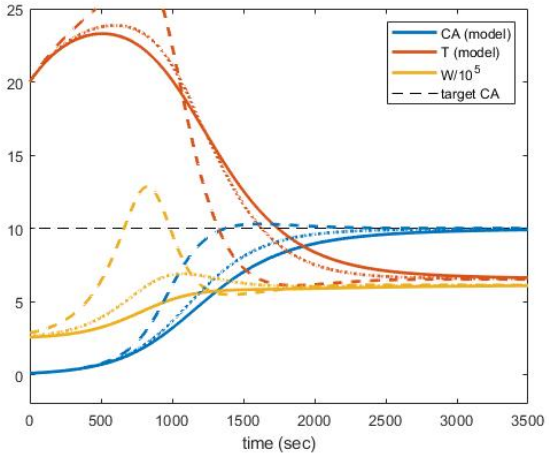

Fig. 6. Closed-loop responses for the uncertain case using NMPC and three alternative choices of weights (solid, dashed and dotted refer to different weighting choices).

6 Conclusions and future work

This paper has modified a recently proposed PFC Algorithm for the non-linear case and demonstrated that it can be applied affectively. The algorithm de-

ploys an intuitive tuning factor, denoted speed-up, which is easy for workers to relate to and thus negates the need for experts to manage the implementation. The user can easily explore the impact of different speed-up choices on other aspects of behaviour such as input activity and decide upon the desired trade-off.

Other core benefits are that, despite being a full non-linear control law, the required on-line computations are relatively minor and thus can easily be coded in low cost processors. Also, in line with other predictive control laws, the incorporation of systematic constraint handling is straightforward and can be managed with a simple for loop. Moreover, the algorithm demonstrates the expected robustness to some parameter uncertainty similarly to other MPC approaches.

Future work will look at whether this concept can be usefully applied to scenarios where the model and measurement information is less precise, or fuzzy, such as where feedback is based on images rather than specific numerical values. It is also important to present a more complete and balanced comparison with tuning using alternative and conventional PFC approaches and this constitutes work in progress.

Funding: EPSRC Future Manufacturing Hub - Manufacture using Advanced Powder Processes (MAPP) through grant EP/P006566/1.

References

[1] D. Q. Mayne, Model predictive control: Recent developments and future promise," Automatica, vol. 50, no. 12, pp. 2967-2986, Dec 2014.

[2] E. Fernandez-Camacho and C. Bordons-Alba, Model Predictive Control in the Process Industry. Springer London, 1995.

[3] S. Qin and T. A. Badgwell, A survey of industrial model predictive control technology," Control Engineering Practice, vol. 11, no. 7, pp. 733-764, Jul 2003.

[4] E. F. Camacho and C. Bordons, Model Predictive Control. Springer, 1999.

[5] J. M. Maciejowski, Predictive Control with Constraints. Pearson Education, 2002.

[6] J. Rossiter, A First Course in Predictive Control. CRC Press, 2018.

[7] S. S. Dughman and J. A. Rossiter (2020) Systematic and effective embedding of feedforward of target information into MPC, International Journal of Control, 93:1, 98-112.

[8] K. J. Astrom and T. Hägglund, PID controllers: theory, design, and tuning, Instrument society of America Research Triangle Park, NC, 1995.

[9] A. Visioli, Practical PID Control. Springer London, 2006.

[10] J. Richalet, A. Rault, J. Testud, and J. Papon, Model predictive heuristic control," Automatica (Journal of IFAC), vol. 14, no. 5, pp. 413-428, 1978.

[11] J. Richalet and D. O'Donovan, Predictive Functional Control: Principles and Indus- trial Applications. Springer Science and Business Media, 2009.

[12] J. Rossiter and R. Haber, The effect of coincidence horizon on predictive functional control," Processes, vol. 3, no. 1, pp. 25-45, 2015.

[13] J. Rossiter, Input shaping for pfc: how and why?" Journal of control and decision, vol. 3, no. 2, pp. 105-118, 2016.

[14] J. A. Rossiter and M. Abdullah, A new paradigm for predictive functional control to enable more consistent tuning," in 2019 American Control Conference (ACC). IEEE, Jul 2019.

[15] J.A. Rossiter and M.S. Aftab, A Comparison of Tuning Methods for Predictive Functional Control. Processes 2021, 9, 1140. https://doi.org/10.3390/pr9071140.

[16] J.A. Rossiter and M.S. Aftab, Exploiting Laguerre polynomials and steady-state estimates to facilitate tuning of PFC. ECC 2022.

[17] M. Abdullah, J. Rossiter, and R. Haber, Development of constrained predictive functional control using Laguerre function based prediction, IFAC-PapersOnLine, vol. 50, no. 1, pp. 10 705-10 710, 2017.

[18] M. Abdullah and J. A. Rossiter, Using Laguerre functions to improve the tuning and performance of predictive functional control," International Journal of Control, pp. 1-13, 2019.

[19] J.B. Rawlings, E.S. Meadows, K.R. Muske, Nonlinear Model Predictive Control: A Tutorial and Survey, IFAC Proceedings Volumes, Volume 27, Issue 2, 1994, Pages 185-197, ISSN 1474-6670, https://doi.org/10.1016/S1474-6670(17)48151-1.

[20] M.S. Aftab and J.A. Rossiter, Pre-stabilised predictive functional control for open-loop unstable dynamic systems, In Proceedings of 7th IFAC Conference on Nonlinear Model Predictive Control 2021, 11-14 Jul 2021.