**Article:**

Rossiter, J.A. orcid.org/0000-0002-1336-0633 and Abdullah, M. (2022) Improving the use of feedforward in predictive functional control to improve the impact of tuning. International Journal of Control, 95 (5). pp. 1206-1217. ISSN 0020-7179

https://doi.org/10.1080/00207179.2020.1843075

# Improving the use of feedforward in Predictive Functional Control to improve the impact of tuning

John Anthony Rossiter[a], Muhammad Abdullah[b]

[a]Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, U.K.; [b]Department of Mechanical Engineering, International Islamic University Malaysia, Jalan Gombak, 53100, Kuala Lumpur, Malaysia.

**ARTICLE HISTORY**
Compiled February 20, 2020

**ABSTRACT**
The classic PFC design is simple and intuitive and yet this paper shows that, counter to expectations, the long standing use of the target information is flawed. Some simple illustrations will demonstrate that what appears sensible can in fact lead to inconsistent decision making with many common process dynamics. Having explored the source of this inconsistency, the paper shows how it can be ameliorated in a systematic fashion and also investigates the impact of the change on loop sensitivity to disturbances. Several numerical examples demonstrate the efficacy of the proposal in the paper.

**KEYWORDS**
predictive functional control; preview control; feed-forward control

## 1. Introduction

Model Predictive Control (MPC) is popular because it deals systematically with both difficult dynamics and also constraint handling [1, 2, 3]. However, classical MPC algorithms tend to be expensive and thus there is a significant market for cheaper alternatives such as Predictive Functional Control (PFC) which, albeit they are not as rigorous, flexible or effective as optimal MPC algorithms, nevertheless can significantly outperform competitors such as PID in many SISO cases [4, 5, 6]. Moreover, PFC is very cheap to code and implement [7, 8] so that use on PLCs or equivalent software/hardware is straightforward.

### *1.1. Concepts of human control and links to PFC*

The main weakness of PFC originates from its main strength, that is the design is intuitive, based on a reflection of how humans control the world around them. Humans tend to actuate in proportion to the error; the larger the error the more aggressively we actuate. The actual amount we actuate is also model based, that is, from experience we have some idea of the actuation move magnitudes required to obtain a desired output response, both in terms of acceleration and steady-state. Hence, in simple terms,

Corresponding author: Email: j.a.rossiter@sheffield.ac.uk

humans anticipate (or predict) the output behaviour that will result from a given actuation change and thus chose the input magnitude to give a prediction with the desired rate of response (or error convergence). One could argue that the human intuition is based on a prediction model somewhat like Figure 1, that is an expectation that with a constant change in the input, the output will move fairly smoothly towards the steady-state. We choose an input value that roughly speaking ensures the prediction matches the ideal behaviour at some point not to far into the future (coincidence point) as this will give the required acceleration; constant updates/feedback will deal with steady-state offset.

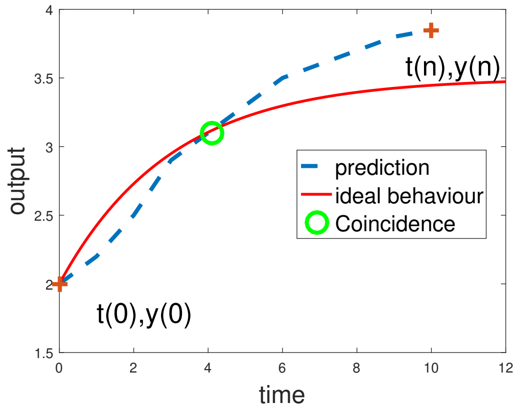



Figure 1.: Ideal output trajectory and typical system behaviour.

PFC practitioners sought to translate the insight of Figure 1 into a mathematical equation that could be implemented in code. Hence, assuming that the ideal target was a first order response , the coincidence of the prediction and target can be ensured with the following equality:

$$y_{k+n|k} = (1 - \lambda^n)r + \lambda^n y_k \tag{1}$$

where $y_{k+n|k}$ is the n-step ahead system prediction at sample time $k$, $\lambda$ is the desired closed-loop pole which controls the convergence rate from output $y_k$ to steady-state target $r$ and $n$ is a coincidence horizon (a tuning parameter), the point where the system prediction is forced to match the target trajectory [5]. To retain human intuition, technicians may be asked for a desired closed-loop time constant/settling time, $T_s$ that has a direct relationship with $\lambda = e^{\frac{-3T}{T_s}}$ (with the sampling period $T$) and can be computed in the code.

**Remark 1.1.** *It is worth emphasising the industrial success of PFC in all likelihood stems from the simplicity of the law (1) and Figure 1 which are very simple to explain and code, and thus of course, straightforward and cheap to implement.*

### *1.2. Weaknesses of PFC and paper contribution*

It can be shown that if the system has well damped open-loop dynamics which are close to 1st-order, then a control law based on solving (1) every sample can be very effective, but critically, where that is not the case then the performance of PFC may be much more erratic and it is more difficult to obtain strong performance assurances [9, 10, 11]. This is unsurprising as the conceptual design is based on a simple assumption of behaviour much like shown in Figure 1; if this assumption is faulty then the corresponding control law to (1) may also be flawed [12, 13].

There are a number of suggestions (e.g. [8, 14]) in the literature for how we can retain the conceptual simplicity while also improving the consistency between the prediction assumptions and reality. To some extent these proposals are quite successful, although the author believes there is a strong reason for exploring methods based in PID prestabilisation [15, 16], more especially as this has strong synergies with methods long since adopted in more classical MPC circles [2, 3]. However, the focus of this paper is not on those issues which link more to the parametrisation of the input trajectory used to solve (1).

A weakness of PFC that has not been explored in the literature is the definition of the coincidence point/target point. A preliminary paper [17] exposed the basic issue but did not study the solution in detail or provide any analysis. This paper will begin in Section 2 by introducing the problems with the default choice of coincidence point. Section 3 will then show how a more sensible choice can be made and define a systematic control structure for implementing this. Numerical illustrations demonstrate the benefits clearly. Section 4 will then explore the repercussions on sensitivity of the proposed adjustment. The paper then finishes with some conclusions.

## 2. Weaknesses of the default concidence point

A core requirement of well posed decision making is consistency from one sample to the next. Chaotic decision making can lead to chaotic behaviours thus it is important that in subsequent samples we support and build on earlier decisions rather than contradicting them. In principle you might expect PFC to lead to consistent decision making, but as this section will show, a lack of appropriate detailed analysis meant the PFC community did not spot where some inconsistencies might exist. This section sets up the mathematical background for PFC carefully before then exposing where the possible flaws lie.

### *2.1. Classical PFC background*

Using a simple English explanation, the PFC control law is based on the following proposal:

*Find the distance between the target and the current output and assume this distance will follow a first order decay. Thus if the distance at the current sample is $D_k$, then the desired distance n-steps ahead will be $\lambda^n D_k$.*

It is not difficult to see that control law (1) is derived from this, that is:

$$\left.\begin{array}{l} D_k = r_k - y_k \\ D_{k+n} = \lambda^n D_k \\ D_{k+n} = r_k - y_{k+n} \end{array}\right\} \quad \Rightarrow \quad r_k - y_{k+n} = \lambda^n (r_k - y_k) \tag{2}$$

For convenience later, define the implied target sequence $R_{k+n|k}$ based on (2) and a known steady-state target $r_k$ at sampling $k$ as follows:

$$R_{k+n|k} = \{E[y_{k+n|k}] = (1 - \lambda^n) r_k + \lambda^n y_k, \quad n = 1, 2, \cdots\} \tag{3}$$

Prediction is well understood in the literature (e.g. [1, 3, 18]) and thus assuming a difference equation model, n-step ahead output predictions take the form:

$$y_{k+n|k} = H_n \underset{\rightarrow}{u_k} + P_n \underset{\leftarrow}{u_{k-1}} + Q_n \underset{\leftarrow}{y_k} \tag{4}$$

where matrices/vectors $H_n$, $P_n$, $Q_n$ depend on the model parameters and

$$\underset{\rightarrow}{u_k} = \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+n-1} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}}_{L} u_k; \quad \underset{\leftarrow}{u_{k-1}} = \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ \vdots \\ u_{k-n_b} \end{bmatrix}; \quad \underset{\leftarrow}{y_k} = \begin{bmatrix} y_k \\ y_{k-1} \\ \vdots \\ y_{k-n_a} \end{bmatrix} \tag{5}$$

for a denominator/numerator with orders $n_a, n_b$ respectively and assuming future inputs are constant ($u_{k+i|k} = u_k$ for $i > 0$). Substituting prediction (4) into equality (3) gives:

$$H_n L u_k + P_n \underset{\leftarrow}{u_{k-1}} + Q_n \underset{\leftarrow}{y_k} = (1 - \lambda^n) r + \lambda^n y_k \tag{6}$$

Rearranging the nominal PFC control law is given as:

$$u_k = \frac{1}{H_n L}\left[(1 - \lambda^n) r + \lambda^n y_k - Q_n \underset{\leftarrow}{y_k} - P_n \underset{\leftarrow}{u_{k-1}}\right] \tag{7}$$

### *2.2. Catering for uncertainty*

In practice predictions such as (4) would be biased due to parameter uncertainty and disturbances. In order to ensure offset free tracking, these predictions need to be corrected so they are unbiased in the steady-state. A classic method for doing this (but not the only method) is to use an internal model (output $y_m$) and find the difference between the model output and the process output $y_p$ as shown in Figure 2 and from this, define a prediction bias term:

$$d_k = y_{p,k} - y_{m,k} \tag{8}$$

The *process* predictions will be unbiased in the steady-state if defined as:

$$E[y_{p,k+n|k}] = H_n L \underset{\rightarrow}{u_k} + P_n \underset{\leftarrow}{u_{k-1}} + Q_n \underset{\leftarrow}{y_k} + d_k \tag{9}$$

where the past outputs $\underset{\leftarrow}{y_k}$ are taken from the model and $E[.]$ is added here to emphasise that this is an expected value. Consequently, the control law (7) is rearranged as follows.

**Algorithm 2.1.** *The conventional PFC control law, with offset free tracking, is summarised as follows:*

$$u_k = \frac{1}{H_n L}\left[(1 - \lambda^n) r + \lambda^n y_{p,k} - Q_n \underset{\leftarrow}{y_{m,k}} - P_n \underset{\leftarrow}{u_{k-1}} - d_k\right] \tag{10}$$

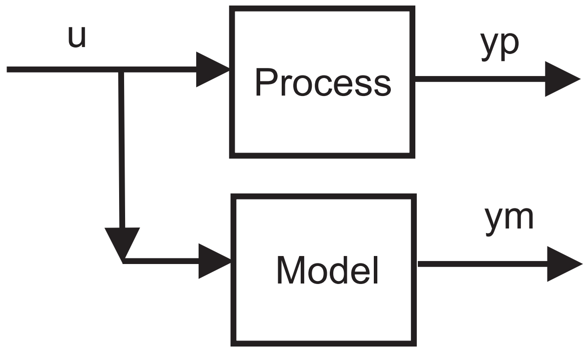



Figure 2.: Internal model structure.

**Remark 2.1.** *For simplicity of presentation this paper assumes no parameter uncertainty so that differences between the process and model are solely due to output disturbances, but control law (10) will ensure offset free tracking for both types of uncertainty, assuming closed-loop stability.*

### *2.3. Concepts of the tail and consistent decision making after target changes*

A well known concept in the conventional MPC literature is the so-called *tail* [3]. So, if a strategy at sample $k$ was to choose a predicted input trajectory as:

$$\underset{\rightarrow k}{u} = [u_k, u_{k+1|k}, u_{k+2|k}, \cdots]^T \tag{11}$$

then we would reasonably expect, assuming no change in target/disturbance/etc., that at the next sample the strategy would be very similar, that is:

$$u_{k+1|k+1} \approx u_{k+1|k}; \quad u_{k+2|k+1} \approx u_{k+2|k}; \quad \cdots \tag{12}$$

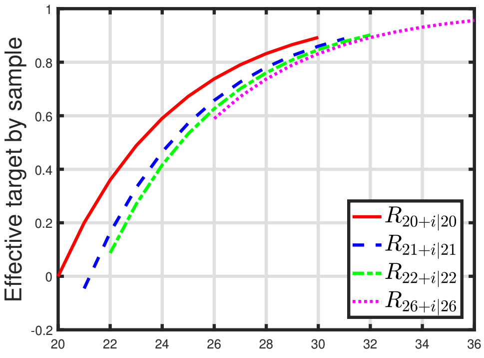
For first order models [9], it is straightforward to show that this consistency is embedded with PFC. However, with higher order models the converse is true, that is, it is possible to show that the proposed trajectories can be quite different. This paper seeks to explain why this occurs and then propose appropriate modifications.

The core issue is not immediately obvious from (1) as this details only $r_k$, but rather is more apparent in the definition of $R_{k+n}$ given in (3). This signal denotes the desired 1st order trajectory that the system is being asked to follow and obviously, for consistency from one sample to the next, one would expect (with no uncertainty) that:

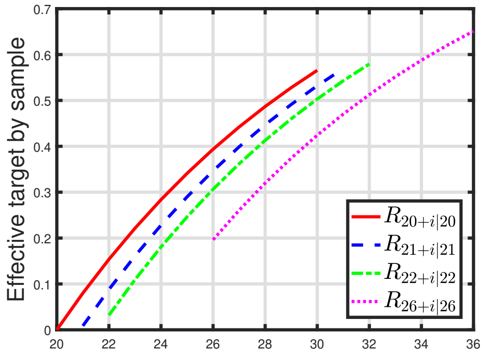
$$R_{k+i|k+1} = R_{k+i|k}, \ \forall i \geq 0 \tag{13}$$

A simple analogy could be driving a car along a road. We do not expect the road we intend to drive along in the near future to keep changing position from one sample to the next; if it did, our steering would become chaotic and we could not plan effectively, much like walking through a bustling crowd.

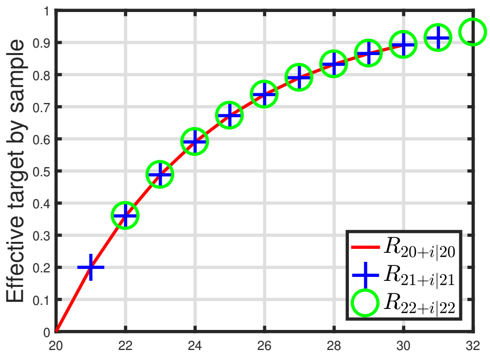
So, it is worth plotting these target signals for some examples and investigating whether they do change or not. This section computes the sequences $R_{k+i|k}$ at succes-




(a) Sequences for example system (14).




(b) Sequences for example system (15).




(c) Sequences for example system (16).

Figure 3.: Implied target sequences $R_{k+i|k}$ for $0 < i < 10$ from (3) at successive sampling instants $k$.

sive sampling instants and overlays on Figs. 3a,3b for the two examples below.

- Define the second order system (14) with $n = 5$ and $\lambda = 0.8$,

$$G(z) = \frac{0.1z^{-1} - 0.4z^{-2}}{1 - 1.4z^{-1} + 0.45z^{-2}} \tag{14}$$

- Define a third order example (15) with $n = 10$ and $\lambda = 0.92$,

$$G(z) = \frac{3.3z^{-1} + 0.31z^{-2} - 3z^{-3}}{1 - 2.76z^{-1} + 2.54z^{-2} - 0.78z^{-3}} \tag{15}$$

In both figures it is clear that the target sequence $R_{k+i|k} \neq R_{k+i|k+1}$; at subsequent samples the coincidence point is therefore notably different which is a bit like the road moving. We are being asked for inconsistent targets from one sample to the next with the differences being especially notable in fast transients and gradually becoming smaller as the closed-loop output approaches steady-state.

**Remark 2.2.** *Readers might note that, in affect, the changes in the target seem to* *be equivalent to a lag. So, the evolution of the target means a gradually slowing down compared to the original target.*

**Remark 2.3.** *As noted above, this behaviour does not occur when applying PFC to first order models, as seen in Figure 3c based on example (16).*

$$G(z) = \frac{1.2z^{-1}}{1 - 0.9z^{-1}} \tag{16}$$

### *2.4. Repercussions of target changes on output behaviour*

The previous section demonstrated clearly how the target trajectory is typically subject to significant lag when PFC is applied to higher order systems. This section elaborates the issue further by considering the control law itself which is based on a coincidence point (1), that is matching the prediction to the target n-steps ahead. What is clear from the following figures is that the coincidence points do not move smoothly as one would like or expect, with the consequence that the resulting closed-loop behaviour could be far from that which was desired.

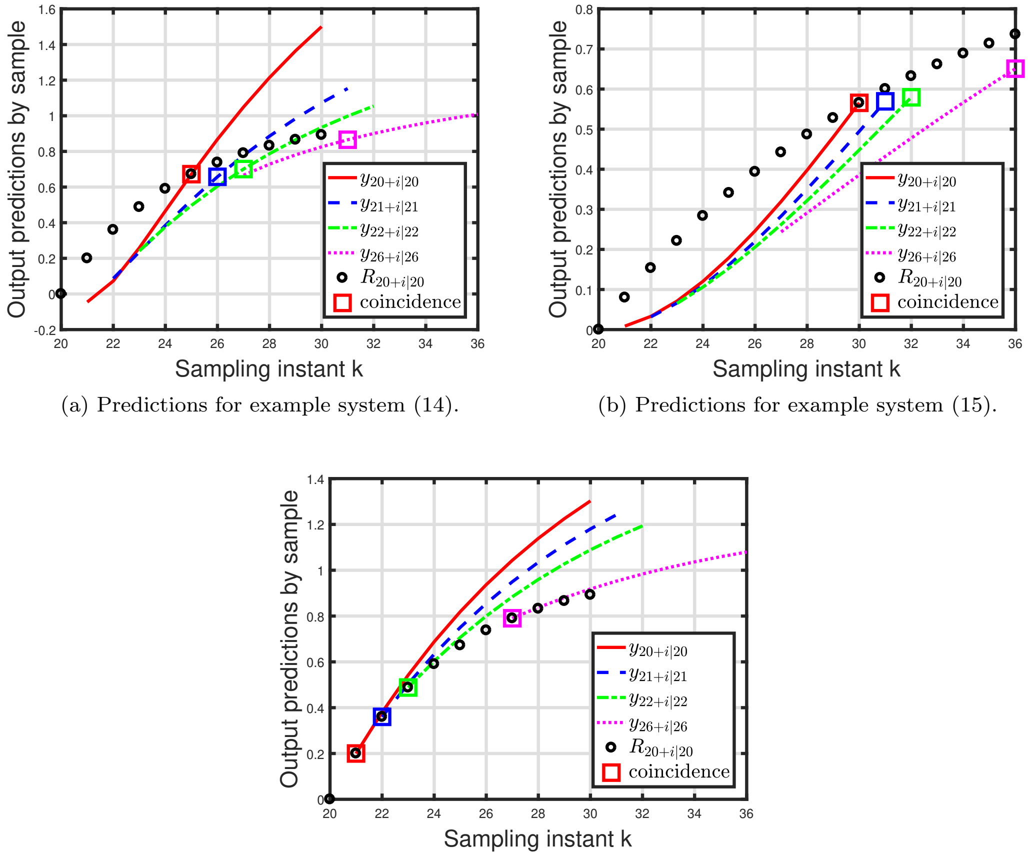
Specifically, Figures 4a,4b show the coincidence points for examples (14,15) along side the corresponding *optimised* predictions at each sample; Figure 4c does the same for the first order example (16).

(1) Again it is clear that one ends up following a lagged version of the orginal target and thus, the closed-loop behaviour will be noticeably slower than the target behaviour. This undermines the role of $\lambda$ as a tuning parameter and a core selling point of PFC which is that, one can *choose* the desired closed-loop time constant.
(2) The coincidence points can be somewhat meandering, going up and down and with inconsistent changes, which is far from what is desired and cannot be a good basis for decision making.
(3) The resulting predictions at subsequent samples change significantly, thus showing inconsistent decision making which is a worrying flaw for a predictive based method, especially when constraint handling and recursive feasibility are to be considered.
(4) The lag does not occur in the coincidence points for the first-order case where the intuitive definition of PFC works best.

In summary, a core tenent of PFC is that the designer is able to select the desired closed-loop time constant and have confidence that the closed-loop behaviour will be close to this as this is the main tuning parameter. This works well with first order systems but clearly, for systems with high order dynamics this is often far from true and thus the core selling point of simple and intuitive tuning, alongside prediction to facilitate constraint handling, is lost.

## 3. The reasons PFC use of feedforward information is flawed and improvements

The previous section has demonstrated clearly the lag that occurs in the target trajectory for PFC with higher order models where inevitably the coincidence horizon must be selected $n \gg 1$. It is perhaps unsurprising then to match this with the observations




(a) Predictions for example system (14).

(b) Predictions for example system (15).

(c) Predictions for example system (16).

Figure 4.: Implied predictions at successive sampling instants $k$ alongside the associated coincidence point $n$ used to determine the PFC control law.

of other papers (e.g. [9, 10]) which have noted that in a conventional PFC agorithm the tuning parameter $\lambda$ has limited impact for many systems. Hence, although the previous section has demonstrated the lagging through the gradual slowing down of the target trajectory, an analysis of the closed-loop shows that this has in fact become embedded into the closed-loop poles rather than the feedforward.

This section will present some analysis which explains more clearly how the lag arises and which then provides insight into how it might be removed. For completeness it is also shown that the same lagging in effect happens with disturbance rejection and thus a conceptually similar solution is also available there. Finally, to enable more systematic sensitivity analysis, in a later section the proposal is reorganised into an analytic form.

### 3.1. Analysis of the causes of the lag

The change in target from one sample to the next is due to an inconsistency between assumptions and indeed, in mainstream MPC concepts of the tail and the importance of consistency from one sample to the next and now long standing [2, 3]. The issue arises because we have two two parallel prediction processes: (i) what we would like the system to do and (ii) what the system actually does. The difficulty is that the first of these is based on coincidence for an n-step ahead prediction while the latter we only take one sample at a time, so one-step ahead. Hence, considering the one-step ahead predictions only:

- The desired trajectory shape one-step ahead is given by:
  $R_{k+1|k} = (1-\lambda)r + \lambda y_k$
- The actual system behaviour (difference equation model) one step ahead is:
  $y_{k+1} = \sum_i b_i u_{k-i+1} - \sum_j a_j y_{k-j+1}$
  whereas $u_k$ is selected to meet (1) which is based on an n-step ahead coincidence.

For consistency we need $R_{k+1|k} = y_{k+1}$ but as will be shown, this often does not happen.

**Theorem 3.1.** *The updated trajectory sequence $R_{k+i|k+1},\ i > 1$ at the next sample is consistent with the sequence $R_{k+i|k}$ from the previous sample if and only if:*

$$y_{k+1} = R_{k+1|k} = (1-\lambda)r + \lambda y_k \tag{17}$$

***Proof.*** The sequences for the trajectories are given as:

$$R_{k+i|k+1} = (1-\lambda^i)r + \lambda^i y_{k+1}; \quad R_{k+i|k} = (1-\lambda^{1+i})r + \lambda^{1+i} y_k \tag{18}$$

It is clear that $R_{k+i|k+1} = R_{k+i|k}$ requires:

$$(1-\lambda^i)r + \lambda^i y_{k+1} = (1-\lambda^{1+i})r + \lambda^{1+i} y_k \tag{19}$$

Substitute in the proposed identity $y_{k+1} = (1-\lambda)r + \lambda y_k$ and we have:

$$(1-\lambda^i)r + \lambda^i[(1-\lambda)r + \lambda y_k] = (1-\lambda^{1+i})r + \lambda^{1+i} y_k \tag{20}$$

which is clearly satisfied. □

**Corollary 3.1.** *For anything other than a first order system, the identity (17) is highly unlikely to be satisfied when* $n \neq 1$ *and thus lagging of the target will occur.*

***Proof.*** Use zero initial conditions and positive $r$ without loss of generality. A generic proof is not possible but we can use knowledge of typical high order systems to make one obvious observation. The one-step ahead prediction implicit in (17) implies a steep initial output gradient, as it matches a first order step response. However, for higher order systems containing 2nd order derivatives and inertia, the initial gradient is typically small and often zero . Consequently, after just one sample, there is often little movement. In consequence:

$$\{\exists u_k = u_{k+i}\ \forall i \ \text{ s.t. } \ y_{k+n|k} = (1-\lambda^n)r\} \quad \Rightarrow \quad \{E[y_{k+1|k}] \ll (1-\lambda)r\} \tag{21}$$

For a non-minimum phase system the result is easier to prove as in that case $y_{k+n|k} = (1-\lambda^n)r \quad \Rightarrow \quad E[y_{k+1|k}] < 0$ for a sensible choice of coincidence horizon and thus $y_{k+1}$ has the wrong sign. □

**Remark 3.1.** *It is useful to illustrate (21) on examples (14,15) respectively.*

*(1) Solving condition (1) with* $\lambda = 0.8, n = 10, r = 1$ *gives* $\{y_{k+10} = -1.478u_k - 0.56u_{k-1} + 1.29y_k - 0.63y_{k-1} = (1-\lambda^{10})r\} \quad \Rightarrow \quad u = -0.455$. *Using this input gives* $y_{k+1} = -0.0455$ *which clearly is far less than the 0.2 required by (17) and moreover is of the wrong sign.*

*(2) Solving (1) with* $\lambda = 0.92, n = 10, r = 1$ *gives* $\{y_{k+10} = 0.22u_k - 0.06u_{k-1} - 0.07u_{k-2} + 28.56y_k - 47.7y_{k-2} + 20.1y_{k-3} = (1-\lambda^{10})r\} \quad \Rightarrow \quad u = 2.579$. *Using this input gives* $y_{k+1} = 0.0086$ *which clearly is only 10% of the 0.08 movement required by (17).*

So in summary, this section has shown that it is fairly typical for the change in the output (one -step ahead) to be far less than the change in the target trajectory. However, in a conventional PFC algorithm, the target trajectory is always reset based on the actual output at each sample and thus, for most systems, the target trajectory is continually changed, or in effect lagged. The amount of lagging is linked to the coincidence horizon, choice of $\lambda$ and system dynamics and thus there does not exist a simple formulae capturing this.

It is also noted that this section has not considered the disturbance signal yet. This is deliberate as it will be more convenient for the reader to introduce that aspect once the proposal for handling target information is properly framed.

## 4. Using systematic feedforward design to compensate for innate PFC lagging

Now that the reason for the inefficacy of the tuning parameter $\lambda$ is better understood, it is possible to consider systematic modifications to ensure that the closed-loop behaviour is closer to what we want. This paper takes a different approach to earlier work [8, 10] which focussed on the loop tuning and instead here considers the role of the feedforward terms. It is well known that feedforward can be useful for improving performance without effecting loop sensitivity and thus there is a good expectation that this approach might be effective.

### *4.1. Modification of PFC control law to remove lag*

The reader will no doubt be thinking that the choice of target is to some extent in the control of the user, so why not manually prevent the drift noted in figures 3a-3b. However, such a change requires a radical overhaul of the original PFC law (1), a rethink in how uncertainty is incorporated and quite possibly will lead to very different loop sensitivity.

The lag arises due to the inconsistency in (21). We can remove this inconsistency by making the target trajectory independent of the current system state/output and depend solely on the target information supplied. It is noted that removing the dependency of the implied target sequence $R_{k+i}$ on the current output measurement $y_{p,k}$ may effect the management of uncertainty, so this issue is dealt with in the following subsection. The proposed algorithm is given next.

**Algorithm 4.1.**

- **Target initialisation:** *Without loss of generality and using superposition, take the case with zero initial conditions and a change in the target at $k = 0$: $r_k = 0,\ k \leq 0$ and $r_k = r,\ k > 0$. The implied target sequence $R_{k+i|k}$ is given as:*

$$\begin{aligned} R_{k+i|k} &= \underbrace{[(1-\lambda), (1-\lambda^2),\ (1-\lambda^3), \cdots]}_{R_0} r \\ &= [R_{k+1|k}, R_{k+2|k}, R_{k+3|k}, \cdots] \end{aligned} \tag{22}$$

- **Simple target update:** *Assuming no change in steady-state target $r$, at the next sample $k+1$, update this sequence by removing the first term and hence:*

$$\begin{aligned} R_{k+i|k+1} &= [(1-\lambda^2), (1-\lambda^3),\ (1-\lambda^4), \cdots] r \\ &= [R_{k+2|k}, R_{k+3|k}, R_{k+4|k}, \cdots] \end{aligned} \tag{23}$$

- **Updates when target changes:** *Next consider a scenario where the target changes, so for example $r_k - r_{k-1} \neq 0, k = h$. The target trajectory needs to retain the history information about earlier targets and be modified to take account of this change, hence:*

$$\begin{aligned} R_{h+i|h} = &[(1-\lambda), (1-\lambda^2),\ (1-\lambda^3), \cdots](r_h - r_{h-1}) \\ &+ [R_{h+2|h-1}, R_{h+3|h-1}, R_{h+4|h-1}, \cdots] \end{aligned} \tag{24}$$

**Remark 4.1.** *The critical observation about Algorithm 4.1 is that the target trajectory is solely dependent on target information and no longer has any dependence on the current system output, thus is quite different to (1).*

The modifed PFC algorithm is now defined.

**Algorithm 4.2.** *The nominal PFC control law is given from:*

$$E[y_{p,k+n|k}] = R_{k+n|k} \tag{25}$$

*Where the target trajectory $R_{k+i|k}$ is updated as in Algorithm 4.1.*

It is clear that as the target no longer has an explicit dependence on the ouput measurement condition and instead solely uses history information from the target, therefore the the target sequence is consistently defined from one sample to another and no lag is induced.

### *4.2. Modified PFC control law with handling of uncertainty*

In order to cater for uncertainty such as disturbances and parameter uncertainty, it is necessary to use unbiased predictions as outlined in section 2.2. Hence, control law (25) could be expanded to read:

$$E[y_{p,k+n|k}] = R_{k+n|k} = y_{m,k+n|k} + d_k \tag{26}$$

where the subscript $p$ is used to denote actual system output value and subscript $m$ for the model output (Figure 1). In practice, the user estimates the values of $y_{p,k+n|k}$ using the following:

$$d_k = y_{p,k} - y_{m,k}; \quad E[y_{p,k+n|k}] = y_{m,k+n|k} + d_k \tag{27}$$

Thus, rearranging (26) gives a possible modified law based on the following:

$$y_{m,k+n|k} = R_{k+n|k} - d_k \tag{28}$$

However, returning to the orginal PFC control law of (1) and adding in the disturbance/bias correction term, gives:

$$y_{m,k+n|k} + d_k = (1 - \lambda^n)r + \lambda^n[y_{m,k} + d_k] \tag{29}$$

or alternatively

$$y_{m,k+n|k} = (1 - \lambda^n)[r - d_k] + \lambda^n y_{m,k} \tag{30}$$

Equation (30) raises an interesting question because the bias correction term appears in the control law in an analogous way to the target information, and hence the modification of the PFC control law proposed in (25) could be equally applied to this term. Hence, defining an initial 'bias correction' future term as follows:

$$B_{k+i|k} = \underbrace{[1 - \lambda, 1 - \lambda^2, 1 - \lambda^3, \cdots]}_{R_0} d_k \tag{31}$$

one can very quickly see an analogous update expression to (24), that is:

$$B_{h+i|h} = R_0[(d_h - d_{h-1})] + [B_{h+2|h-1}, B_{h+3|h-1}, B_{h+4|h-1}, \cdots] \tag{32}$$

The proposed modified PFC control law is then given in the following algorithm.

**Algorithm 4.3.**

*(1) Update the target trajectory and bias correction terms using (24,32).*
*(2) Define the control law by solving for input $u_k$:*

$$y_{m,k+n|k} = R_{k+n|k} - B_{k+n|k} = H_n \underrightarrow{u_k} + P_n \underleftarrow{u_k} + Q_n \underleftarrow{y_k} \tag{33}$$

For completeness, it is important to establish the proposed control law (33) achieves offset free tracking in the presence of uncertainty.

**Theorem 4.1.** *Assuming closed-loop stability and in the unconstrained case, PFC Algorithm 4.3 will ensure the system outputs converge to the steady-state target.*

***Proof.*** In the unconstrained case, control law (33) is fixed and thus normal linear analysis can be deployed. A simple proof considers the steady-state this loop reaches and confirms whether or not that is consistent with zero offset. Steady-state assumes that past and future inputs are constant and no recent changes in the target and bias correction terms. Thus, at steady-state (subscript $ss$), the following identities must hold:

$$y_{m,k} = y_{m,k+n|k} = G_{ss}u_k = R_{k+n|k} - B_{k+n|k}; \tag{34}$$

Recursive use of update equation (24,32) with $r_k, d_k$ constant will lead to $R_{k+i|k} = r, B_{k+i|k} = d, \ \forall i$, and hence (34) can be rewritten as:

$$y_{m,k} = G_{ss}u_k = r - d \tag{35}$$

that is, no steady-state offset. □

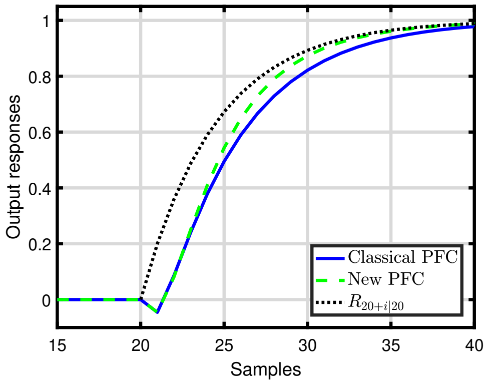
### *4.3. Numerical illustrations for proposed algorithm*

This section will demonstrate the efficacy of the proposed algorithm using examples (14), (15). Specifically it will illustrate clearly that by removing the lag in the target trajectories, the closed-loop behaviour is now closer to the orginal desired dynamic in terms of its convergence; clearly fast transients for high order systems will rarely be able to follow a first order dynamic closely.

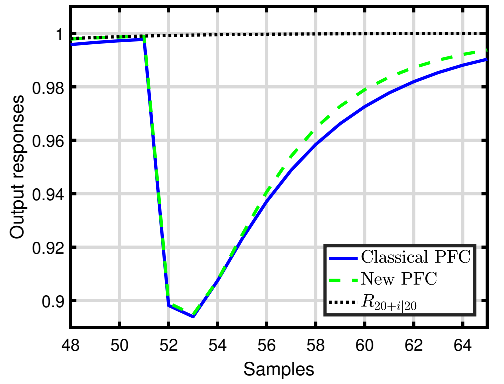
The closed responses for a change in the target and a change in the disturbance for examples (14,15) are in in Fig. 5a, 5b and Fig 6a, 6b respectively. Both examples have the expected slow transient response, especially given one is non-minimum phase. However, thereafter the trajectories clearly approach the desired trajectory much better than a classical PFC law and the speed-up is also retained for disturbance rejection.

## 5. Closed-loop interpretations and sensitivity analysis

It is clear from the previous figures that the closed-loop responses have changed, and consequently so has sensitivity and the closed-loop poles. Hence, it is important to look at both the poles and sensitivity more closely to gain insights into the impact of changes in the PFC algorithm. Moreover, the reader may have noticed that the update expression, for example (24), implicitly requires a very long vector with history information which is clumsy, and potentially requiring exccessive memory. This section first demonstrates how the proposed use of feedforward information can be presented in a more systematic transfer function relationship that lends itself to loop analysis and much simpler implementation. From this the loop sensitivity analysis is developed followed by numerical illustrations and conjectures.

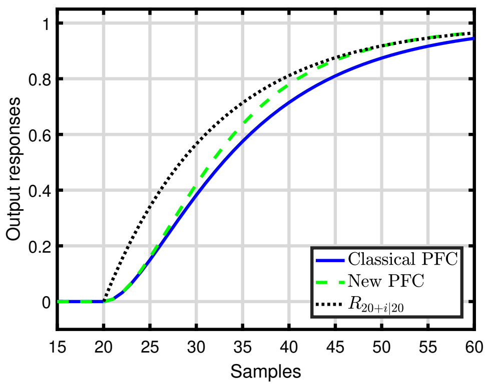



(a) Responses for a change in the target.

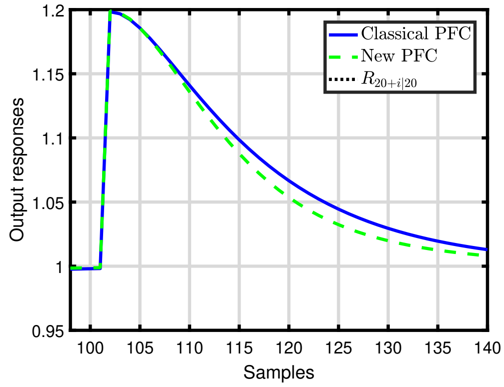



(b) Responses for a change in the disturbance.

Figure 5.: Closed-loop responses for classical PFC and Algorithm 4.3 on system (14).




(a) Responses for a change in the target.




(b) Responses for a change in the disturbance.

Figure 6.: Closed-loop responses for classical PFC and Algorithm 4.3 on system (15).

### *5.1. Analytic expressions for feedforward action*

The proposal of this paper is that feedforward information utilises the expressions in (24,32) within the control law of (33). Here it is shown that expressions (24,32) have a more compact realisation.

**Lemma 5.1.** *The expression (24) capturing the inclusion of target information into the control law can be represented with a difference equation model of the form:*

$$R_{k+n|k} = r_k - \lambda^n w_k \tag{36}$$

*where*

$$w_k = \lambda w_{k-1} + \delta r_k; \quad \delta r_k = r_k - r_{k-1} \tag{37}$$

***Proof.*** First expand out (24) in terms of all the impacts of past information, and hence the entire future trajectory can be captured with the infinite sequence:

$$\begin{array}{rcl} R_{k+i|k}, \{i = 1, 2, \cdots\} & = & [(1-\lambda), (1-\lambda^2),\ (1-\lambda^3), \cdots](r_k - r_{k-1}) \\ & & \quad +[(1-\lambda^2), (1-\lambda^3),\ (1-\lambda^4), \cdots](r_{k-1} - r_{k-2}) \\ & & \quad +[(1-\lambda^3), (1-\lambda^4),\ (1-\lambda^5), \cdots](r_{k-2} - r_{k-3}) \\ & & \quad + \cdots \end{array} \tag{38}$$

It can be reduced to a simpler form by considering just a single coincidence horizon or point in the future:

$$R_{k+n|k} = (1-\lambda^n)\delta r_k + (1-\lambda^{n+1})\delta r_{k-1} + (1-\lambda^{n+2})\delta r_{k-2} + \cdots \tag{39}$$

which, noting that $r_k = \sum_i \delta r_{k-i}$, can be rearranged to be:

$$R_{k+n|k} = r_h - \lambda^n \delta r_k - \lambda^{n+1}\delta r_{k-1} - \lambda^{n+2}\delta r_{k-2} - \cdots \tag{40}$$

Finally, consider the output of the difference equation (37):

$$w_k = \lambda w_{k-1} + \sum_{i=0}^{\infty} \lambda^i \delta r_{k-i} \tag{41}$$

Hence, substituting (41) into (40) gives:

$$R_{k+n|k} = r_k - \lambda^n w_k \tag{42}$$

□

**Corollary 5.1.** *Disturbance information can be included into the control law (33) using a model of the form:*

$$R_{k+n|k} - B_{k+n|k} = r_k - d_k - \lambda^n w_k; \quad w_k = \lambda w_{k-1} + \delta r_k - \delta d_k; \tag{43}$$

*This is given without proof as analogous to the previous lemma.*

**Theorem 5.1.** *A transfer function representation of (43) is given from* $T_k = R_{h+n|h} - B_{k+n|k}$ *with:*

$$T(z) = \left[1 - \frac{(1-z^{-1})\lambda^n}{(1-\lambda z^{-1})}\right](r(z) - d(z)) \tag{44}$$

***Proof.*** We have that $R_{k+n|k} - B_{k+n|k} = r_k - d_k - \lambda^n w_k$ and also that $w_k = \lambda w_{k-1} + \delta r_k - \delta d_k$. The latter of these can be rearranged using z-transforms to take the form:

$$w(z) = \frac{(1-z^{-1})}{(1-\lambda z^{-1})}(r(z) - d(z)) \tag{45}$$

as $\delta r(z) = (1-z^{-1})r(z), \delta d(z) = (1-z^{-1})d(z)$. It then follows immediately, by taking z-transforms of all terms, that:

$$T(z) = \underbrace{\left[1 - \frac{(1-z^{1})\lambda^n}{(1-\lambda z^{-1})}\right]}_{F_t(z)}(r(z) - d(z)) \tag{46}$$

□

**Theorem 5.2.** *The PFC control law (33) can be expressed in transfer function form.*

***Proof.*** The control law is given as:

$$y_{m,k+n|k} = R_{k+n|k} - B_{k+n|k} = T_k = H_n L u_k + P_n \underset{\leftarrow}{u_{k-1}} + Q_n \underset{\leftarrow}{y_{m,k}} \tag{47}$$

which can be rearranged as:

$$\underbrace{[H_n L, P_n]}_{D_t} \underset{\leftarrow}{u_k} = -Q_n \underset{\leftarrow}{y_{m,k}} + T_k \tag{48}$$

Rewriting in terms of z-transforms (e.g. [3]) gives:

$$D_t(z)u(z) = -Q_n(z)y_m(z) + T(z) = -Q_n(z)y_m(z) + F_t(z)[r(z) - d(z)] \tag{49}$$

□

### *5.2. Sensitivity functions for the proposed PFC and conventional PFC*

The previous section has described the transfer function expression for the proposed PFC control law in (49). The conventional PFC algorithm of (10) has a similar form of expression which here will be denoted as:

$$D_c(z)u(z) = -\underbrace{[Q_n(z) - \lambda^n]}_{Q_c(z)} y_m(z) + F_c(z)[r(z) - d(z)]; \quad F_c = 1 - \lambda^n \tag{50}$$

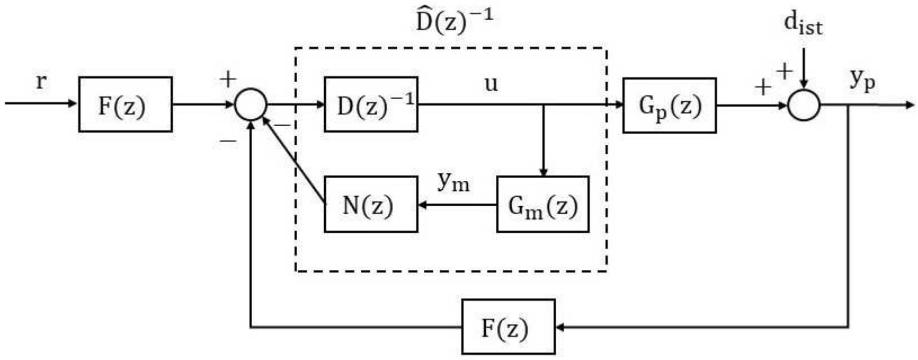



Figure 7.: Block diagram representation of PFC laws (49,50).

The interesting point to note here is that, by inspection of both (10,48), one can see that:

$$D_t(z) = D_c(z); \quad Q_n(z) \neq Q_c(z); \quad F_t(z) \neq F_c(z) \tag{51}$$

Consequently, the two laws differ solely in how they use the feedforward information on the target and the bias information.

**Remark 5.1.** *It should be noted that subscript t is use to denote the proposed algorithm and subscript c is for the traditional PFC.*

**Lemma 5.2.** *Algorithms 2.1, 4.3, summarised in control laws (49,50) have different sensitivity and different closed-loop pole polynomials.*

***Proof.*** This follows by noting that the $F_c(z)$ terms multiplies onto bias information, that is $d(z) = y_p(z) - y_m(z)$ and thus in effect introduces a further feedback loop. Hence, the control law (50) can be expressed as:

$$D_c(z)u(z) = F_c(z)r(z) - F_c(z)y_p(z) - \underbrace{[Q_c(z) - F_c(z)]}_{N_c(z)} y_m(z) \tag{52}$$

As $F_t(z) \neq F_c(z)$ and $Q_n(z) \neq Q_c(z)$, different parameters in the feedback loop must lead to different sensitivities for control law (49). The appropriate block diagram for both algorithm is shown in Figure 7 (note that no subscripts are assigned to keep it general). □

**Theorem 5.3.** *Assuming no parameter uncertainty, the closed-loop poles for (49) can be determined from:*

$$P_t(z) = \hat{D}_t(z)a(z) + F_t(z)b(z) \tag{53}$$

***Proof.*** From Figure 7, we can represent $\hat{D}_t(z)^{-1} = [D_t(z) + N_t(z)y_m(z)]^{-1}$ where

$y_m(z) = \frac{b(z)}{a(z)} u(z)$. Thus, the closed-loop pole can be computed as:

$$P_t(z) = 1 + F_t(z)\hat{D}_t(z)^{-1}G_p(z) \tag{54}$$

Then, combine (54) with system/process model $G_p(z) = \frac{b(z)}{a(z)}$, ones can get (53). □

**Corollary 5.2.** *The closed-loop poles for control law (50) have the analogous structure, but with $F_c(z)$ rather than $F_t(z)$ and $\hat{D}_c(z)$ rather than $\hat{D}_t(z)$.*

$$P_c(z) = \hat{D}_c(z)a(z) + F_c(z)b(z) \tag{55}$$

**Theorem 5.4.** *The output sensitivity to disturbance transfer function for Algorithm (49) can be form as:*

$$S_{yd} = a(z)P_t(z)^{-1}\hat{D}_t(z) \tag{56}$$

***Proof.*** Based on Figure 7, the output sensitivity to disturbance can be formed by finding a transference from $y_p$ to $D_{ist}$ as:

$$S_{yd} = [1 + P_t(z)\hat{D}_t(z)^{-1}G_p(z)]^{-1} \tag{57}$$

Substituting the pole (53) and system/process model $G_p(z) = \frac{b(z)}{a(z)}$ into (57), ones can form the transfer function as (56). □

**Corollary 5.3.** *Similarly, the output sensitivity to disturbance for control law (50) can be formed with $P_c(z)$ and $\hat{D}_c(z)$.*

$$S_{yd} = a(z)P_c(z)^{-1}\hat{D}_c(z) \tag{58}$$

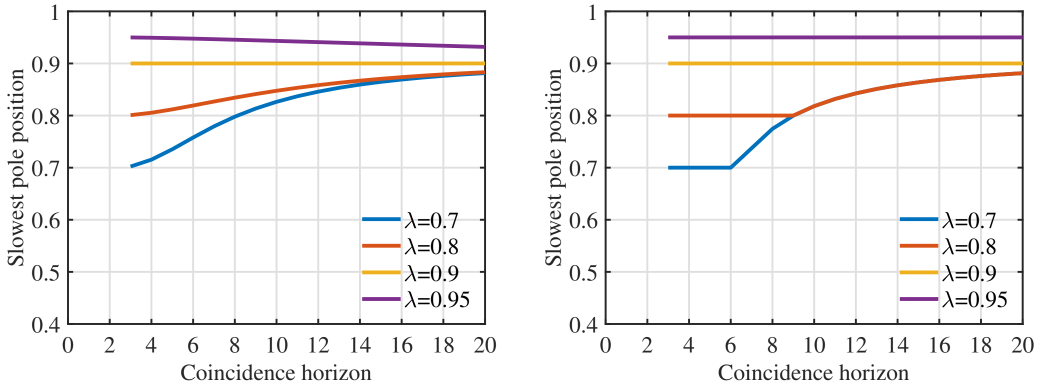
### *5.3. Numerical Illustrations*

This section uses examples (14,15) and compares:

(1) The dominant closed-loop pole achieved for various pairs of $n, \lambda$ with Algorithms 4.3,2.1 (see Figures 8 and 9).
(2) The sensitivity plot to output disturbances (see Figure 10).

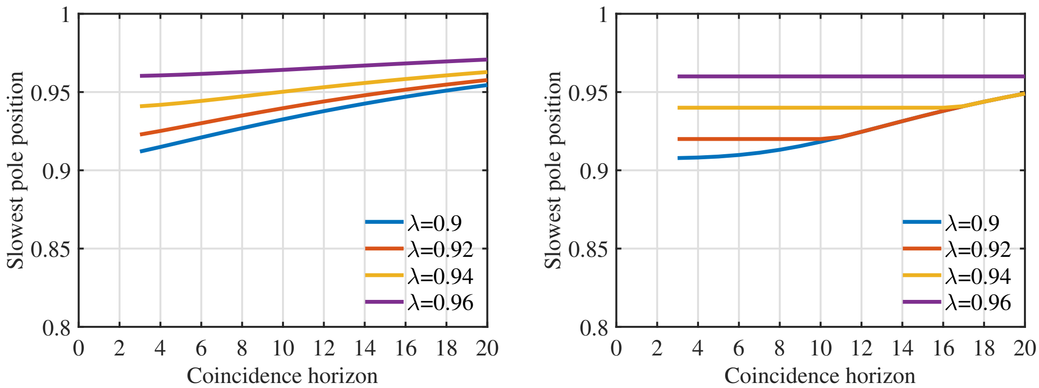
Considering Figures 8 and 9, as expected, for a smaller coincidence horizon $n$, the dominant closed-loop pole is closer to the desired one $\lambda$. Conversely, with larger $n$, the system will eventually converge to its open loop behaviour. However, one can clearly see that Algorithm (4.3) managed to reduce the lag in tracking the desired $\lambda$, especially within a smaller $n$, for both cases where the dominant closed-loop pole is constant within this range. Hence, it may provide faster system convergence compared to the traditional PFC as discussed in Section 4.3. Besides, this information also is very useful to setup a guideline in selecting a suitable tuning parameters $n, \lambda$ for a system.

For the output sensitivity to disturbance as shown in Figure 10, the new Algorithm (4.3) improves the system sensitivity in the low and mid frequency range and produces almost the same sensitivity in the higher frequency range for both examples. The main reason behind this is because its target trajectory now becomes independent on the




(a) Algoritm (2.1). (b) Algoritm (4.3).

Figure 8.: Dominant closed-loop pole with different pairs of $n, \lambda$ on system (14). Note: $n < 3$ leads to unstable pole




(a) Algoritm (2.1). (b) Algoritm (4.3).

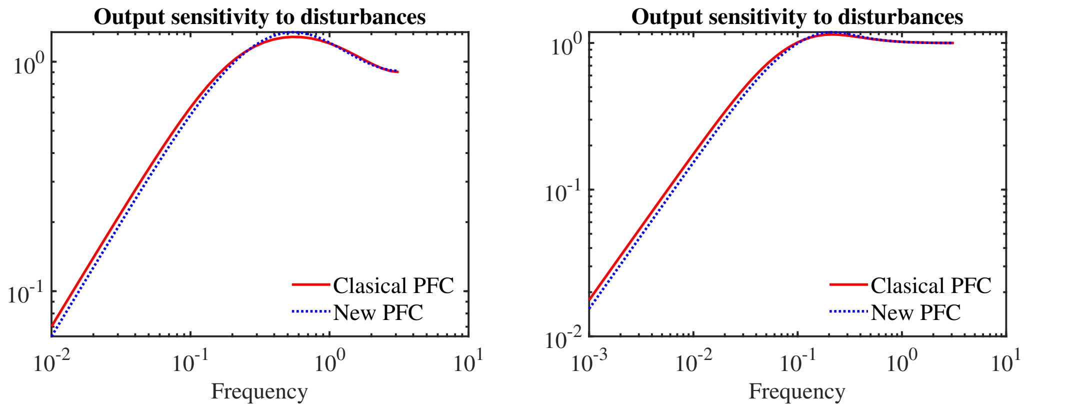
Figure 9.: Dominant closed-loop pole with different pairs of $n, \lambda$ on system (15).

measured output $y_m$. Hence, Algorithm (4.3) may provide faster response in handling disturbances as compared to Algorithm 2.1 as discussed in the previous section (and Figures 5 and 6).

## 6. Conclusions

This paper makes two important contributions to algorithms for Predictive Functional Control.

(1) An important insight is that the main tuning parameter which conventionally is set as the desired closed-loop time settling time, or equivalently desired closed-loop pole, usually cannot be delivered. While this fact had been observed in the past it had not been explained. This paper gives clear and novel insight into how lag enters a conventional PFC law, and thus causes the closed-loop response to be slower than expected/desired.




(a) Example (14).

(b) Example (15).

Figure 10.: Comparison of output sensitivity to disturbance between algorithm (2.1,4.3).

(2) Once the cause of the lagged responses has been understood, the paper is able to propose systematic modifications to avoid this. This requires an imaginative change from the existing control law definition, but in fact the change retains the same fundamental concepts and intuition. It introduces the need for a history variable, but it is shown that this can be handled with a first order model which thus is simple to code and implement, thus the resulting algorithm is no more complex than the classical PFC to code.

The responses for the proposed PFC algorithm are encouraging and demonstrate improved tuning properties. It is not surprising that loop sensitivity is modified, as the behaviour is now different, so for completeness the paper also illustrates the sensitivity or the proposed modification.

In terms of future work, this paper has not explicitly considered constraint handling although the basic algorithm would not require anything novel compared to what exists in the literature and thus is not worth including here. However, there have been other modifications to PFC in the literature in recent years which used a different approach to tackle tuning efficacy, and thus it would be interesting to incoporate the proposal here into some of those and discern whether even better tuning efficacy can result.

## References

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

[2] D. Mayne, J. Rawlings, C. Rao, P. Scokaert, Constrained model predictive control: Stability and optimality, Automatica 36 (2000) 789–814.

[3] J. A. Rossiter, A First Course in Predictive Control: second edition, Taylor and Francis, 2018.

[4] J. Richalet, A. Rault, J. Testud, J. Papon, Model predictive heuristic control:applications to industrial processes, Automatica 5 (14) (1978) 413–428.

[5] J. Richalet, D. Donovan, Predictive Functional Control: principles and industrial applications, Springer-verlag, 2009.

[6] M. Abdullah, J. A. Rossiter, Input shaping predictive functional control for different types of challenging dynamics processes, Processes 6 (8) (2018) 118.
[7] M. Khadir, J. Ringwood, Extension of first order predictive functional controllers to handle higher order internal models, Int. Journal of Applied Mathematics and Comp. Science 18 (2) (2008) 229–339.
[8] K. Zabet, J. Rossiter, R. Haber, M. Abdullah, Pole-placement Predictive Functional Control for under-damped systems with real numbers algebra, ISA Transations 71 (2) (2017) 403–414.
[9] J. A. Rossiter, R. Haber, The effect of coincidence horizon on predictive functional control, Processes 31 (1) (2015) 25–45.
[10] J. A. Rossiter, R. Haber, K. Zabet, Pole-placement predictive functional control for over-damped systems with real poles, ISA Transactions 61 (2016) 229–239.
[11] J. A. Rossiter, A priori stability results for PFC, International Journal of Control 90 (2) (2016) 305–313.
[12] J. A. Rossiter, Input shaping for PFC: how and why?, J. Control and Decision (2015) 1–14.
[13] M. Abdullah, J. Rossiter, Input shaping predictive functional control for different types of challenging dynamics processes 6 (8) (2018) 118.
[14] M. Abdullah, J. Rossiter, R. Haber, Development of constrained predictive functional control using Laguerre function based prediction, in: IFAC world congress, 2017, pp. 10705–10710.
[15] Z. Zhang, J. Rossiter, L. Xie, H. Su, Predictive functional control for integral systems, in: International Symposium on Process System Engineering, 2018, p. 6.
[16] M. Saleheen Aftab, J. A. Rossiter, Predictive Functional Control with Explicit Pre-conditioning for Oscillatory Dynamic Systems, in: IFAC world congress - submitted, 2020, p. 6.
[17] J. A. Rossiter, M. Abdullah, A new paradigm for Predictive Functional Control to enable more consistent tuning, in: American Control Conference, 2019.
[18] J. A. Rossiter, Notes on multi-step ahead prediction based on the principle of concatenation, Proc. of the IMechE Part I 207 (4) (1993) 361–263.