Proceedings Paper:

Rossiter, J. orcid.org/0000-0002-1336-0633 and Abdullah, M. (2019) A new paradigm for Predictive Functional Control to enable more consistent tuning. In: Proceedings of the American Control Conference. 2019 American Control Conference, 10-12 Jul 2019, Philadelphia, USA. IEEE , pp. 366-371. ISBN 9781538679265

A new paradigm for Predictive Functional Control to enable more consistent tuning

John Anthony Rossiter[1] and Muhammad Abdullah[2]

Abstract— **This paper presents two significant contributions to the understanding of Predictive Functional Control (PFC). First, it gives novel insights and explanations into a poorly understood issue, that is the weak link between PFC tuning parameters and the resulting closed-loop behaviour. This new understanding is then exploited to proposed a modification to the existing PFC algorithm which creates a much stronger tuning link while retaining the critical properties of elementary coding and understanding. The efficacy of the proposal is demonstrated on several numerical examples.**

Keywords—Predictive Control, PFC, Tuning, Performance.

I. INTRODUCTION

Predictive functional control (PFC) has been very widely adopted in industry [1], [2], [3] and has been successful because of its relatively simple concept, that helps industrialists and technical staff easily understand its core design principles. The coding requirements, at least for systems with straightforward dynamics, are simple and can be coded in just a few lines on whatever processor is available, including PLCs [4], [5]. Indeed, the downside is that academic rigour and proofs of convergence and stability [6] are generally quite difficult to obtain except for a few special cases [7], [8]; yet, most industrialists would not worry too much about that as long as the control algorithm is effective and cheap.

One purpose of this paper is to unpick one of the theoretical weaknesses in a standard PFC approach and thus explain why the tuning procedure, although simple in practice, in reality gives quite poor links between the expected behaviour compared to what is achieved [9]. Building on this, some insights from more general predictive control are used to suggest how this weakness might be overcome in a manner which is still very simple to implement and code and thus maintaining the cost-effectiveness of PFC. The reader should note however that it is unrealistic to expect generic proofs of stability and/or feasibility with PFC; while these are available with many MPC algorithms they come at substantially greater computational demand and expense.

The main concept in PFC is to treat a 1st order response as an ideal closed-loop behaviour and choose future input values which force the predicted system behaviour to overlap with the target 1st order response at some specific point in the future. For systems with close to first-order dynamics it can be shown that this approach is very effective and indeed, one can even develop strong stability and feasibility results for this case [8], [9], [10]. Conversely, when a system is governed by dynamics which are not close to first-order, it is unsurprising that attempting to force a first-order response is ambitious at best and unwise at worst; indeed for several types of dominant dynamics, it is easy to show that a standard PFC algorithm can be difficult to tune effectively [11], [12].

This paper focusses on a related issue which has strong links to the concepts of recursive feasibility adopted in the mainstream MPC literature [6], [13], [14]; that is can one select at the next time instant a policy which, in essence, replicates the policy selected at the previous sample? Embedding consistency of decision making from one sample to the next enables the user to quickly give proofs for convergence, as the worst case decision making is bounded by that given at the previous sample, and also feasibility (guarantees that predictions satisfy constraints), for the same reasons [6]. It will be shown in section II that for classical PFC, the use of a reformulated first-order dynamic target at each sampling instant implies an inconsistency in decision making which results in the poor tuning properties. However, using the same insights, section III proposes a straightforward modification of PFC which gives a conceptually almost similar algorithm that has much stronger properties and thus more reliable tuning. Section IV provides a conclusion.

II. CLASSICAL PFC CONCEPTS AND RECURSIVE PROPERTIES

This section will introduce a classical PFC and then show how it has good recursive properties with first-order systems but may not do so with higher-order dynamics. Without loss of generality and ease of presentation, this paper utilises a general transfer function model although PFC can take any form of prediction structures [2].

A. Classical PFC algorithm

The PFC framework is based on the assumption that the system should behave similarly to a desired target trajectory. Although it is possible to use a higher order polynomial as a target trajectory, yet the usual practise is to follow a first-order response due to its simple characterisation of convergence [2]. More precisely, the predicted output follows a first-order response from the current value to the desired steady-state, and thus one could enforce the following equality [9]:

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

where $y_{k+n|k}$ is the n-step ahead system prediction at sample time k, λ is the desired closed-loop pole which controls the convergence rate from output y_k to steady-state target r and

[1,2] J.A. Rossiter and M. Abdullah are with Department Automatic Control and System Engineering, The University of Sheffield, S1 3JD, UK. j.a.rossiter@sheffield.ac.uk

n is a coincidence horizon (a tuning parameter), the point where the system prediction is forced to match the target trajectory [2]. For convenience later, define the implied target sequence based on (1) at sampling k as follows:

$$R_{k+n|k} = \{y_{k+n|k} = (1-\lambda^n)r + \lambda^n y_k, \quad n = 1,2,\cdots\} \qquad (2)$$

The n-step ahead prediction for a transfer function model (e.g. [13], [14], [15]) takes the following form for input u_k:

$$y_{k+n|k} = H\underset{\rightarrow}{u_k} + P\underset{\leftarrow}{u_k} + Q\underset{\leftarrow}{y_k} \qquad (3)$$

where matrix H, P, Q depend on the model parameters and for a transfer function with dimension n_a (denominator) and n_b (numerator):

$$\underset{\rightarrow}{u_k} = \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+n} \end{bmatrix}; \underset{\leftarrow}{u_k} = \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ \vdots \\ u_{k-n_b} \end{bmatrix}; \underset{\leftarrow}{y_k} = \begin{bmatrix} y_k \\ y_{k-1} \\ \vdots \\ y_{k-n_a} \end{bmatrix} \qquad (4)$$

Substituting prediction (3) into equality (1) gives:

$$H\underset{\rightarrow}{u_k} + P\underset{\leftarrow}{u_k} + Q\underset{\leftarrow}{y_k} = (1-\lambda^n)r + \lambda^n y_k \qquad (5)$$

Extracting the n^{th} row from the matrix: H_n, P_n, Q_n along with the constant future input assumption of PFC [2], [3] means $u_{k+i|k} = u_k$ for $i > 0$ and defining $h_n = \sum(H_n)$, the control law is formulated as:

$$\Delta u_k = \frac{1}{h_n}\left[(1-\lambda^n)r + \lambda^n y_k - Q_n\underset{\leftarrow}{y_k} - P_n\underset{\leftarrow}{u_k}\right] \qquad (6)$$

Remark 1: This subsection has ignored details of unbiased prediction [14], the handling of uncertainty and prediction alternatives so as not to distract from the core concepts.

B. Recursive properties with PFC

A core concept within the MPC literature is the so-called *tail*, that is the part of the prediction from the previous sample which has yet to happen. In order to ensure consistent decision making, it is normal to define the degrees of freedom (d.o.f.) in the prediction such that the predictions at subsequent samples can be chosen to match, if desired; that is:

$$y_{k+n|k} = y_{k+n|k+1}, \quad \forall n > 0 \qquad (7)$$

Given PFC deploys the constant future input assumption, such property complies automatically.

A secondary but related concept is that the performance index or control law computation should be such that one could easily default to the *tail* through normal decision making and only move from this where predicted performance improvement is evident. This is where PFC has a weakness:

- In effect that the input trajectory is defined for the entire future as a constant.
- The output trajectory is only computed at a single point from (1), with the rest of the trajectory being ignored.
- The implicit assumption is that the output predictions will follow (1), however with the exception of a single special case, this is not true.

The consequence is that there is a mismatch in the implied assumptions: either the input will deviate away from its tail or the output predictions will do so and thus some important recursive properties are lost. However, the most important possible inconsistency arises from variations in (1) from one sample to the next. To see this, we will illustrate the prediction and coincident points at successive samples.

1) Recursive properties with high-order models: This section will plot the implied targets of (1) calculated at a number of successive sampling instants. For consistent decision making, one would expect the implied target to be the same.

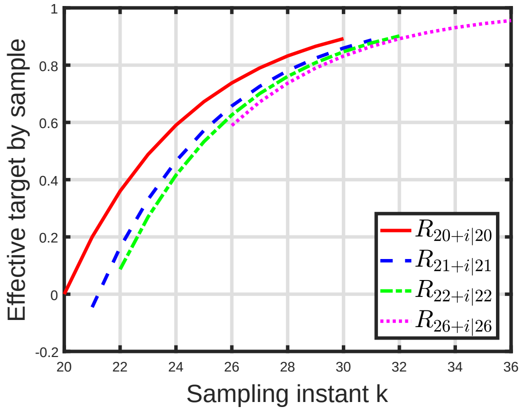

Fig. 1. Implied target sequences $R_{k+i|k}$ for $0 < i < 10$ from (2) for system (8) at successive sampling instants k.

- Take the following second order system (8) with $n = 5$ and $\lambda = 0.8$,

$$G(z) = \frac{0.1z^{-1} - 0.4z^{-2}}{1 - 1.4z^{-1} + 0.45z^{-2}} \qquad (8)$$

Compute the sequences $R_{k+i|k}$ from (2) at different sampling instants k and overlay on Fig. 1. The target sequence at sampling instant $k = 21$ is notably different from that of the previous sample $k = 20$ and hence there is inconsistency in what is being asked of the control law from one sample to the next. This inconsistency continues through future sampling instants, although as the output y_k converges the effective target gets closer and closer to r and so the differences reduce.

- Consider now a third order example (9) with $n = 10$ and $\lambda = 0.92$,

$$G(z) = \frac{3.3z^{-1} + 0.31z^{-2} - 3z^{-3}}{1 - 2.76z^{-1} + 2.54z^{-2} - 0.78z^{-3}} \qquad (9)$$

The compute sequences $R_{k+i|k}$ as in (2) at different sampling instants k in Fig. 2 shows the same pattern as example (8), where the target R_{k+i} keeps changing at different sampling instant k.

2) Recursive properties with first-order models:

$$G(z) = \frac{1.2z^{-1}}{1 - 0.9z^{-1}} \qquad (10)$$

Readers will be interested to know that the observations of Fig. 1 and 2 do not apply to first-order models (or indeed

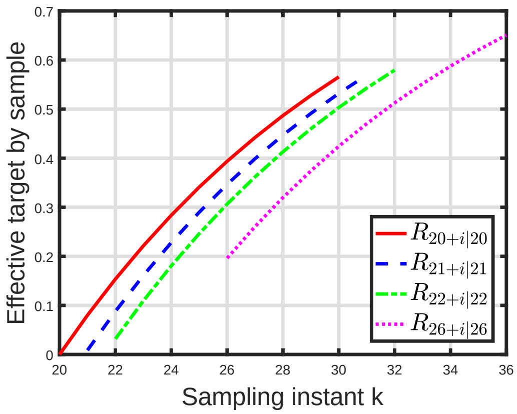

Fig. 2. Implied target sequences $R_{k+i|k}$ for $0 < i < 10$ from (2) for example (9) at successive sampling instants k.

where one can safely use $n = 1$). A corresponding example (10) with $n = 1$ and $\lambda = 0.8$ as in Fig. 3 clearly shows that the target R_{k+i} is now unchanged at different sampling k!

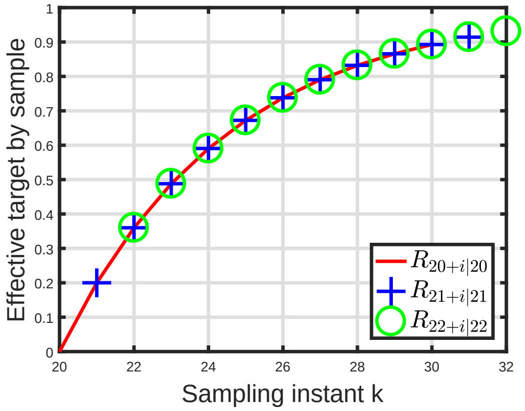

Fig. 3. Implied target sequences $R_{k+i|k}$ for $0 < i < 10$ from (2) for example (10) at successive sampling instants k.

C. *Repercussions of target changes on output behaviour*

A core tenent of PFC is that the user or designer is able to select the desired closed-loop time constant, or equivalently the implied closed-loop λ which appears in (1). The target behaviour is expected to be *embedded* by forcing the predictions to follow that target behaviour. However, herein the reader will notice an immediate inconsistency.

- Fig. 1 and 2 show that the implied target changes every sample so that, in effect one is no longer following the expected target stated at the outset, but some alternative lagged version. The lag is critical as this means that in effect, the control law computations are following a slower target than expected and desired.
- Fig. 3 shows that this lag does not occur in the first-order case (or where $n = 1$) and in this case, the implied target is the same from one sample to the next.

The effect of this implied lagging can be demonstrating by looking at the system predictions associated to control law (6) at successive samples alongside the implied coincidence of (1), n steps into the future. Fig. 4-6 show the corresponding output predictions for Fig. 1-3, respectively and the chosen coincidence points, over-lapped with the sequence R_{k+i} from the initial sample k.

1) For example (8) (see Fig. 4) the coincidence points deviate away from $R_{20+i|20}$ and also show a somewhat meandering path which calls into question the efficacy of λ as a tuning parameter given the implied coincidence point is somewhat inconsistent from one sample to the next and thus does not overlap well with the original desired dynamic of $R_{20+i|20}$.
2) For example (9), Fig. 5 shows even greater deviations between the coincidence points and the original target and thus it is wholly unsurprising that the eventual closed-loop dynamic achieved is not close to the original target.
3) Example (10) (see Fig. 6) is the exception. Although there are some changes in the optimised predictions, it is noted that all the coincidence points lie upon the original target $R_{20+i|20}$ and therefore, in this case, the desired dynamic is achieved in the closed-loop.

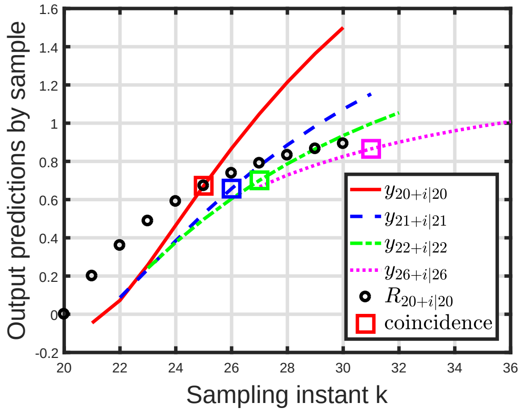

Fig. 4. Implied predictions for example (8) at successive sampling instants k alongside the associated coincidence point n used to determine the PFC control law.

D. *The reasons why the PFC tuning parameter of desired time constant is flawed with non-first-order models*

It is self evident from Fig. 1,2,4,5 that for many cases, the definition of the PFC control law through coincidence points in (1) alongside an initially slow responding underlying system dynamic and $n \gg 1$, leads to the implied target trajectory gradually drifting away from the original target; in effect the target behavior is much slower and thus unsurprisingly the resulting closed-loop behaviour is also much slower. While other works [9], [10], [14] have noticed the inconsistency between the target λ and the achieved closed-loop pole, this is the first work to our knowledge which fully exposes why this inconsistency is happening.

The reader should note that the change from one sample to the next is a consequence of two parallel prediction processes which are inconsistent for most systems.

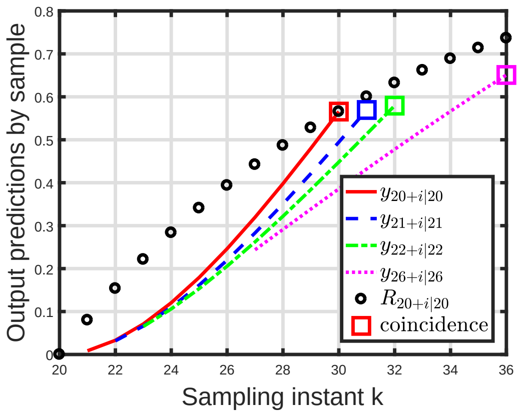

Fig. 5. Implied predictions for example (9) at successive sampling instants k alongside the associated coincidence point n used to determine the PFC control law.

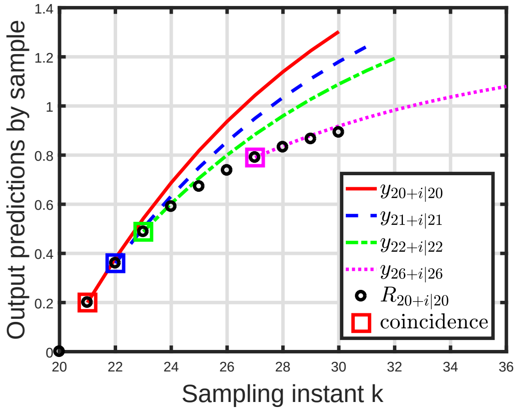

Fig. 6. Implied predictions for example (10) at successive sampling instants k alongside the associated coincidence point n used to determine the PFC control law.

- Trajectory shape one-step ahead given by: $y_{k+1|k} = (1-\lambda)r + \lambda y_k$
- Actual system behaviour: $y_{k+1} = \sum_i b_i u_{k-i+1} - \sum_j a_j y_{k-j+1}$ where u_k is selected to meet (1).

In general $\sum_i b_i u_{k-i+1} - \sum_j a_j y_{k-j+1} \neq (1-\lambda)r + \lambda y_k$.

Theorem 1: The desired trajectory sequence $R_{k+1|k+1}$ at the next sample is consistent with the second prediction from the previous sample $R_{k+2|k}$, that is:

$$R_{k+1|k+1} = (1-\lambda)r + \lambda y_{k+1} = (1-\lambda^2)r + \lambda^2 y_k \qquad (11)$$

if and only if $y_{k+1} = (1-\lambda)r + \lambda y_k$.

Proof: Although a generic proof is not possible, it is fairly obvious that for typical high order dynamics the 1-step ahead response to a change in input is quite small, so meeting a coincidence point computation (1) n steps into the future will likely mean the implied $y_{k+1|k}$ is much smaller than target, that is (without loss of generality the reader can assume zero initial conditions and positive system gain) for illustration:

$$\left\{ \begin{array}{c} u_k \quad \Rightarrow \quad y_{k+n|k} = (1-\lambda^n)r + \lambda^n y_k \\ y_{k+1} = \sum_i b_i u_{k-i+1} - \sum_j a_j y_{k-j+1} \end{array} \right\} \qquad (12)$$
$$\Rightarrow \quad y_{k+1} \ll (1-\lambda)r + \lambda y_k$$

As a consequence, y_{k+1} has not changed from y_k as much as required except for first-order processes:

$$(1-\lambda)r + \lambda y_{k+1} \neq (1-\lambda^2)r + \lambda^2 y_k \qquad (13)$$

and thus some lag in the set point trajectories is introduced, as seen in Fig. 1 and 2. □

However, of course the actual y_{k+1} is typically different, where u_k is selected to satisfy (1) and $n \neq 1$ (as $n > 1$ is typically essential especially for non-minimum phase system and higher order model [9]).

III. IMPROVING THE TUNING EFFICACY OF PFC

Some recent works [5], [10] used parallel prediction via partial fraction expansions and exploited the PFC properties for first-order systems as a means of improving the tuning process. However, that method still required an arbitrary selection of some parameters/poles which could impact significantly on the overall behaviour. Ideally, PFC should be defined to have a few design variables as possible to simplify the process while ensuring it as intuitive as possible for users. Other recent work [16] is considering the use of alternative parameterisations for the degrees of freedom in the prediction, again as a means of embedding the desired tuning more logically. However, that approach does not yet explicitly deal with the trajectory lag issues discussed here.

Hence, this paper will exploit the new insights given by the previous section into why classical PFC often does not deliver the targeted poles and will, as a preliminary work, focus solely on a classical PFC formulation such as defined in (6). It has been shown that the classical formulation can lead to a drift in the implied target, primarily due to the implied mismatch illustrated in (12) when a process has a slow initial response (slow that is compared to 1st order dynamics). The most obvious proposal therefore is to consider mechanisms which avoid the drift in the implied target trajectory used in the control law computations, so that the sequence R_k remains the same, irrespective of the actual system behaviour. This is not as immediately trivial as the reader might think due to the requirement for ensuring effective handling of uncertainty within the control law formulation.

A. Classical PFC control law with handling of uncertainty

In order to cater for uncertainty such as disturbances and parameter uncertainty, it is common to rewrite (1) in an equivalent form as follows (readers should note that alternatives do exist and we chose the formulation that is most convenient for purpose):

$$y_{p,k+n|k} = (1-\lambda^n)r + \lambda^n y_{p,k} \qquad (14)$$

where the subscript p is used to denote actual system output value. The model (denoted by subscript m) and process are simulated in parallel as indicated in Fig. 7.

In practice, the user estimates the values of $y_{p,k+n|k}$ using the following:

$$d_k = y_{p,k} - y_{m,k}; \quad E[y_{p,k+n|k}] = y_{m,k+n|k} + d_k \qquad (15)$$

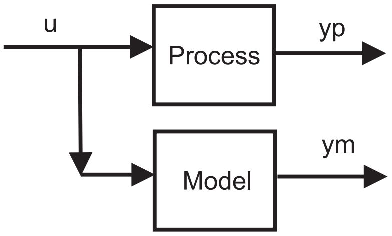

Fig. 7. Independent model prediction structure.

Thus the control law of (14) can be rewritten as:

$$\begin{aligned} & y_{m,k+n|k} + d_k = (1-\lambda^n)r + \lambda^n[y_{m,k}+d_k] && (16) \\ \text{or} \quad & y_{m,k+n|k} = (1-\lambda^n)[r-d_k] + \lambda^n y_{m,k} && (17) \end{aligned}$$

B. Modification of PFC control law to remove lag

The lag noted in section II-B arises due to the inconsistency in (12). We can remove this inconsistency by making the target fixed in time, that is, removing the dependency of the implied target sequence R_{k+i} on the current output measurement $y_{p,k}$, but obviously while still defining the control law so that it caters for uncertainty.

Algorithm 1: Without loss of generality and ultimately using superposition, consider the case where the system has zero initial conditions and there is a change in the target, that is $r_k = 0,\ k \leq 0$ and $r_k = r,\ k > 0$. Hence, at sample time k the implied target sequence $R_{k+i|k}$ can be formulated as:

$$\begin{aligned} R_{k+i|k} &= \underbrace{[(1-\lambda),(1-\lambda^2),\ (1-\lambda^3),\cdots]}_{R_0} r \\ &= [R_{k+1|k}, R_{k+2|k}, R_{k+3|k}, \cdots] \end{aligned} \qquad (18)$$

At the next sample $k+1$, we would simply update this sequence in logical manner by removing the first term which is now in the past, so that

$$\begin{aligned} R_{k+i|k+1} &= [(1-\lambda^2),(1-\lambda^3),\ (1-\lambda^4),\cdots]r \\ &= [R_{k+2|k}, R_{k+3|k}, R_{k+4|k}, \cdots] \end{aligned} \qquad (19)$$

Next consider a scenario where the target changes, so for example $r_k - r_{k-1} \neq 0, k = h$. Such a change implies an associated target change so, using superposition, then:

$$\begin{aligned} R_{h+i|h} = [(1-\lambda),(1-\lambda^2),\ (1-\lambda^3),\cdots](r_h - r_{h-1}) \\ + [R_{h+2|h-1}, R_{h+3|h-1}, R_{h+4|h-1}, \cdots] \end{aligned} \qquad (20)$$

Changes in disturbance estimate d_k impact the target sequence in an analogous fashion so that the overall sequence catering for uncertainty is updated each sample as follows:

$$\begin{aligned} R_{h+i|h} = R_0[(r_h - r_{h-1}) - (d_h - d_{h-1})] \\ + [R_{h+2|h-1}, R_{h+3|h-1}, R_{h+4|h-1}, \cdots] \end{aligned} \qquad (21)$$

Algorithm 2: The PFC control law is given from:

$$y_{m,k+n|k} = R_{k+n|k} \qquad (22)$$

Where the reader notes the removal of the explicit dependence on the initial condition and instead the use of history information from the target to ensure the target sequence is consistently defined, and thus removing any lag in the implied target.

C. Analysis of properties of proposed PFC law

The most important property to establish is offset free tracking, or equivalently, that the control law will successfully reject both parameter uncertainty and disturbances.

Theorem 2: Assuming closed-loop stability, the use of Algorithm 2 in conjunction with update (21) will ensure the system outputs converge to a reachable steady-state target.

Proof: The control law leads to fixed term control law which thus reaches a fixed steady-state. A simple proof can be based therefore on assessing the steady-state and checking whether that is inconsistent or not with zero offset. Steady-state assumes that past and future inputs (including those arising from (22) now) must be the same. First, assuming no changes in the target and set point, then $R_{k+n|k} = R_{k+n+1|k+1} = R_{k+n+2|k+2}, \cdots = r - d_k$ where d_k=$y_{p,k} - y_{m,k}$. At steady-state (subscript ss), from (22), the following identities must hold:

$$y_{m,k} = y_{m,k+n|k} = G_{ss}u_k = R_{k+n|k}; \quad y_p = y_m + d_k \qquad (23)$$

which alongside the definition of $R_{k+n|k}$ implies that $y_p = r$ as required! □

D. Numerical illustrations

This section will utilise the same examples (8), (9) and demonstrate that the replacement of control law (1) with Algorithm 2 for removing lag from the overall process and thus gives a better consistency with the desired closed-loop dynamic λ. The illustrations will also include an external and time invariant output disturbance to demonstrate that Algorithm 2 does indeed deliver offset free tracking in the uncertain case.

The responses for example (8) are shown in Fig. 8 and 9. It is clear that the proposed algorithm is much more faithful to the original target trajectory than the classical algorithm as, despite the initial slow response due to the non-minimum phase characteristic, the output response then approaches the original target within settling time whereas the classical approach does not. The desired speed up is also obtained during the output disturbance rejection. A similar response is shown in Fig 10 and 11 for example (9).

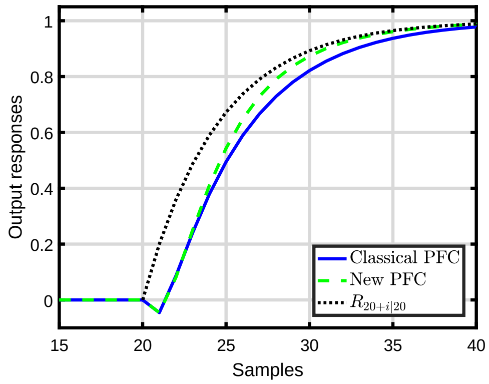

Fig. 8. Closed-loop output tracking responses for classical PFC and Algorithm 2 PFC on system (8).

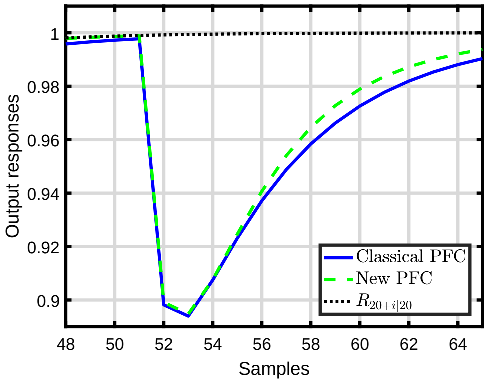

Fig. 9. Closed-loop disturbance (with amplitude of -0.1) rejection responses for classical PFC and Algorithm 2 PFC on system (8).

IV. CONCLUSIONS

This paper has made two notable and novel contributions in the field of predictive functional control.

1) Firstly, it has given explanations and illustrations for why the main tuning parameter that is the desired closed-loop time constant/settling time is often ineffective when the coincidence horizon exceeds one. In particular, it has shown how inconsistency between the very rapid initial response of a first order system (and the ideal target) as compared to the more typical slow initial response for high order systems, means that the effective target deployed by PFC is continually lagged more and more in each sample. This repetitive lagging/delay in the target leads to the closed-loop response lagging behind the originally desired target and thus having slower dynamics than intended.
2) Secondly, the paper proposes a straightforward modification to PFC to overcome this repetitive lagging. The main idea is to frame the PFC objective slightly different so it is not based on an instantaneous measure of the distance from the target, but rather a measure with some memory of the targets used at previous samples. The computation in this step is trivial and thus the resulting algorithm is no more complex than the classical PFC to code as seen in (21) and (22).

In summary, this paper identifies a known weakness which is the poor link between the PFC tuning parameters and the resulting closed-loop behaviour. By exposing the causes, this paper has proposed a solution which has been shown to be effective on non-simple case studies.

REFERENCES

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

[2] J. Richalet, and D. O'Donovan, *Predictive Functional Control: principles and industrial applications*. Springer-Verlag, 2009.

[3] J. Richalet, and D. O'Donovan, "Elementary predictive functional control: A tutorial", Int. Symposium on Advanced Control of Industrial Processes, pp. 306–313, 2011.

[4] 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), pp. 229–239, 2008.

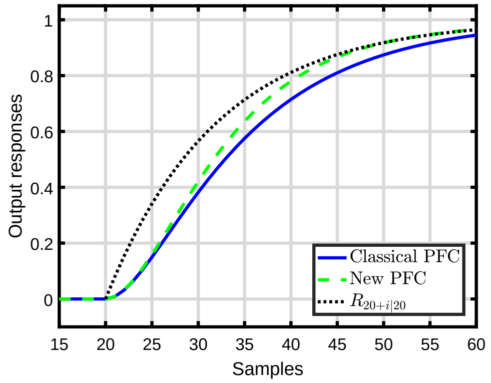

Fig. 10. Closed-loop output tracking responses for classical PFC and Algorithm 2 PFC on system (9).

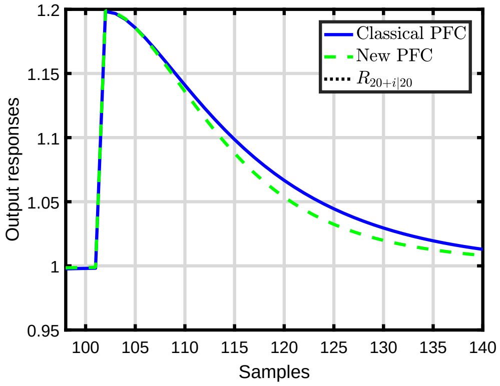

Fig. 11. Closed-loop disturbance (with amplitude of 0.2) rejection responses for classical PFC and Algorithm 2 PFC on system (9).

[5] K. Zabet, J. A. Rossiter, R. Haber, and M. Abdullah, Pole-placement Predictive Functional Control for under-damped systems with real numbers algebra, *ISA Transactions*, 71(2), pp. 403–414, 2017.

[6] D. Q. Mayne, J.B. Rawlings, C. V. Rao and P. O. M. Scokaert, Constrained model predictive control: Stability and optimality, *Automatica*, 36, pp. 789–814, 2000.

[7] M. Khadir, J. Ringwood, "Stability issues for first order predictive functional controllers: extension to handle higher order internal models", International Conference on Computer Systems and Information Technology pp. 174-179, 2005.

[8] J. A. Rossiter, A priori stability results for PFC, *International Journal of Control*, 90(2), pp. 305–313, 2016.

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

[10] J. A. Rossiter, R. Haber, and K. Zabet, Pole-placement predictive functional control for over-damped systems with real poles, *ISA Transactions*, 61, pp. 229–239, 2016.

[11] J. A. Rossiter, Input shaping for PFC: how and why?, *Journal of Control and Decision*, pp. 1–14, Sep. 2015.

[12] M. Abdullah and J. A. Rossiter, Input shaping predictive functional control for different types of challenging dynamics processes, *Processes*, 6(8), 118, 2018.

[13] D. W. Clarke, C. Mohtadi, P. S. Tuffs, Generalized predictive control. Part I. The basic algorithm, *Automatica* 23(2), pp. 137–148, 1987.

[14] J. A. Rossiter, *A First Course in Predictive Control*, Second Edition, Taylor and Francis, 2018.

[15] J. A. Rossiter, Notes on multi-step ahead prediction based on the principle of concatenation, *Proc. of the IMechE, Part I: Journal of Systems and Control Engineering* 207(4), pp. 261–263, 1993.

[16] M. Abdullah, J. A. Rossiter and R. Haber, "Development of constrained predictive functional control using Laguerre function based prediction", IFAC World Congress, pp. 10705-10710, 2017.