Article:
Bin Abdullah, M. and Rossiter, J. orcid.org/0000-0002-1336-0633 (2019) Using Laguerre functions to improve the tuning and performance of predictive functional control. International Journal of Control, 94 (1). pp. 202-214. ISSN 0020-7179

https://doi.org/10.1080/00207179.2019.1589650

ARTICLE TEMPLATE

Using Laguerre functions to improve the tuning and performance of predictive functional control

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

[a,b] Department of Automatic Control and Systems Engineering, The University of Sheffield, S1 3JD, UK.

ARTICLE HISTORY
Compiled January 31, 2019

ABSTRACT
This paper proposes an alternative parameterisation of the degrees of freedom in a predictive functional control (PFC) law. Using recent insights on the potential of Laguerre functions in traditional MPC (Rossiter et al., 2010; Wang, 2009), it is demonstrated that these functions can also be exploited to give a good effect in PFC. An appropriate design with tuning methodology is developed and this is then demonstrated with a number of numerical examples.

KEYWORDS
MPC; PFC; coincidence horizon; tuning.

1. Introduction

Model predictive control (MPC) has been very popular in the literature (Camacho & Bordons, 1999; Rossiter, 2018) for decades and very widely applied in industry (Richalet , 1993). However, the literature has given much less attention to certain approaches within the MPC portfolio, namely algorithms such as Predictive Functional Control (PFC), (Fallasohi et al., 2010; Fiani & Richalet, 1991; Haber et al., 2011; Richalet et al., 1978, 2009). This may appear somewhat surprising given the evidence that PFC is so widely used in industry, however, the reasoning is simple: the tuning and general properties of PFC are difficult and weak compared to more conventional MPC algorithms (Rossiter & Haber, 2015; Rossiter, 2015; Rossiter et al., 2016) and in consequence, academic authors and reviewers are very wary since most of the academic journals are focusing on the theoretical development of a predictive controller. The same situation can be seen in most of the core textbooks of MPC as PFC concept is barely discussed, for examples Rawlings & Mayne (2009); Wang (2009) to name a few. More specifically, with the exception of a few special cases (Rossiter, 2016), PFC is not conducive to *a priori* stability gaurantees and many reviewers are uncomfortable with this weakness, not withstanding the huge successes in industry especially for many chemical applications (Haber et al., 2011; Richalet et al., 2009).

The prime aim of this paper is to propose a modification to PFC which improves the overall properties and thus gives the user more confidence in the resulting closed-loop behaviour and constraint handling. We do not pretend that a generic *a priori* stability

Corresponding author. Email: MAbdullah2@sheffield.ac.uk

proof is possible and instead emphasise that PFC should not be contrasted with more advanced MPC algorithms such as dual-mode (Rossiter et al., 1998; Rossiter, 2018; Scokaert & Rawlings, 1998) because:

(1) PFC is a competitor with algorithms such as PID which equally have weak *a priori* properties.
(2) PFC is orders of magnitude cheaper than dual-mode MPC and thus a comparison between the two is inappropriate. Besides the application is usually restricted to Single Input and Single Output (SISO) processes; you get what you pay for and if you want a very cheap and simple algorithm do not expect all the analysis and properties of expensive alternatives.

PFC offers properties such as prediction and systematic constraint handling not easily embedded in PID while having coding complexity that is similar to PID and tuning rules that are easy to automate and understand (Richalet et al., 2009, 2011).

In recent years, some authors have been to explore alternative parameterisations of the degrees of freedom with conventional MPC approaches. One could argue this begins with prestabilisation (e.g. Muske & Rawlings (1993)) and indeed the dual-mode approaches as proposed in the 1990s are, in effect, reparameterising the d.o.f. (Rossiter, 2018). Latterly, authors have been looking at functional approaches (Rossiter et al., 2010; Wang, 2009) whereby the future input trajectory is defined as a linear combination of a set of functions (Khan & Rossiter, 2013), where the systematic choice of function is still to a large extent an open question (Muehlebach & D'Andrea, 2017). The advantages of functions such as Laguerre over the more conventional choice of a standard basis set are that they extend the impact of the input changes over a much longer horizon and thus are more likely to be able to capture the shape of the desired closed-loop input trajectory. This simple change can lead to a reduction in the number of d.o.f. needed to manage constraints effectively and thus enables performance to be maintained with a lower computational load.

The obvious question to ask then is: to what extent can a similar parameterisation improve the properties of PFC leading to easier or more consistent tuning? Such an advance would be of significant benefit given the large number of SISO loops which involve some what challenging dynamics and constraints which make PID implementations messy and often poorly tuned whereas, by contrast, PFC may be equally cheap and also able to handle those dynamics and constraints more systematically. Of course, as with PID, the user must accept that any stability analysis is *a posteriori*.

A smaller but important contribution of this paper is also to improve the constraint handling techniques typically adopted in a conventional PFC algorithm as, in order to demonstrate the constraint handling of the proposed Laguerre approaches, it is pertinent to ensure the constraint handling is done as effectively as possible. The existing strategies popular in PFC were developed using engineering intuition and limited computing but as will be seen can be improved using insights available from more modern MPC approaches and while still incurring minimal computational loading and coding, certainly in terms of the functionality available on current cheap processors. Specifically the aim is to avoid the need for an online optimisation as that ensures the code is simple, quick and easy to maintain and validate.

This paper will propose the use of Laguerre functions for PFC and demonstrate how these can be introduced in a systematic manner. Section 2 will give standard background on PFC and briefly highlight the tuning challenges. Section 3 looks at constraint handling and proposes a systematic but simple procedure. Section 4 will introduce Laguerre functions and propose two different mechanisms for introducing

these into PFC. Section 5 will give a number of numerical examples and Section 6 will give a case study on some laboratory hardware. Section 7 contains the conclusions.

2. Background on PFC

This section summarises the key assumptions, notation and principles underlying a conventional PFC algorithm and gives a brief insight into the weaknesses. As much of this is standard in the literature, detailed derivations are omitted.

2.1. Standard PFC algorithm

PFC is premised on the assumption that it is reasonable to desire the closed-loop response to follow (approximately) a first order trajectory from the current position to the desired steady-state target.

Remark 2.1. *For simplicity of exposition, as these issues only introduce more complicated algebra but do not affect the core principles, this paper will not discuss issues linked to non-zero dead-times and time-varying targets. Details are available in the references.*

Therefore, the desired output trajectory is given as:

$$r_{k+n} = R - (R - y_k)\lambda^n, \tag{1}$$

where r_{k+n} denotes the desired n-step ahead value for output y_k at sample k and λ is the desired closed-loop pole (PFC practitioners often use the desired closed-loop time constant in lieu of λ as these are equivalent) and R is the target. The unconstrained PFC law is defined by solving, for a single specified coincidence horizon n:

$$y_{k+n|k} = r_{k+n} \quad \text{with} \quad u_k = u_{k+1|k} = u_{k+2|k} = \cdots, \tag{2}$$

where $y_{k+n|k}, u_{k+n|k}$ are the n-step ahead predicted values for the output and input respectively made at sample k.

In order to solve (2), the dependence of the output predictions on the assumed values $u_k = u_{k+i|k},\ i \geq 0$ is needed. Prediction algebra is standard in the literature (e.g. Rossiter (2018)) so here we simply assume the solution can be given as:

$$y_{k+n|k} = P_n u_p + Q_n y_p + H_n \underset{\rightarrow k}{u} + d_k; \quad u_p = \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ \vdots \\ u_{k-n_b} \end{bmatrix}; \quad y_p = \begin{bmatrix} y_{k-1} \\ y_{k-2} \\ \vdots \\ y_{k-n_a} \end{bmatrix}; \quad \underset{\rightarrow k}{u} = \begin{bmatrix} u_k \\ u_{k+1|k} \\ u_{k+2|k} \\ \vdots \end{bmatrix}, \tag{3}$$

for suitable P_n, Q_n, H_n, n_b, n_a and d_k is a term to ensure unbiased prediction (typically taken as the difference between the process measurement and an internal model output, although these details are not central here). Note that from (2) we can write $\underset{\rightarrow k}{u} = Lu_k, \quad L = [1, 1, \cdots, 1]^T$. Substituting prediction (3) into (1,2) the PFC control law

can be defined as:

$$\{P_n u_p + Q_n y_p + H_n L u_k + d_k = R - (R - y_k)\lambda^n\} \;\Rightarrow\; u_k = \frac{R - (R - y_k)\lambda^n - P_n u_p - Q_n y_p - d_k}{H_n L}. \tag{4}$$

Remark 2.2. *A main selling point of PFC is the computational simplicity of control law (4). Given H_n, P_n, Q_n are needed for just a single horizon n, the computation of these can be relatively trivial and thus the overall coding requirements are elementary (Richalet et al., 2009).*

2.2. Tuning of PFC

The industrial popularity of PFC is partially down to the intuitive tuning parameters. The designer, at least in principle, chooses the desired closed loop time constant (equivalently λ). The computer can then do a quick search over diffferent choices of coincidence horizon n, displays the associated responses and then the user can determine which value n gives the most desireable closed-loop behaviour. However, herein lies two major weaknesses (Khadir & Ringwood, 2005, 2008; Rossiter & Haber, 2015).

(1) Often the actual closed-loop performance/dynamics are not close (Rossiter et al., 2016; Zabet et al, 2017) to the chosen pole λ which of course draws into question the value of this as a tuning parameter (main exceptions are when $n = 1$ can be chosen which is generally true only for first order plant).
(2) An offline search over different coincidence horizons is somewhat clumsy and difficult to argue as systematic and gives no assurance that a reasonable answer will result.

Remark 2.3. *It is easy to show that PFC suffers from the prediction mismatch (Rossiter, 2018) common in open-loop MPC approaches whereby the optimised predictions may bare little resemblance to the closed-loop behaviour that results. This inconsistency can (not must) lead to poor descision making. The mismatch arises because the prediction assumption on the future input, that this remains constant, is in many cases inconsistent with the actual input trajectory that arises or indeed is required for good behaviour.*

In summary, where a process has close to first order dynamics, PFC works very well. However, as the open-loop dynamics differ more from a first order system, the usefulness of λ as a tuning parameter reduces and the selection of an appropriate coincidence horizon become less obvious. This paper seeks to propose some alternative formulations to reduce these weaknesses and, as will be noted, can be very beneficial when it comes to constraint handling.

3. Systematic constraint handling in PFC with recursive feasibility

Given PFC deploys only very simple coding to enable use on low level processors, the constraint handling is defined to be simple and thus avoids the optimisers common in more mainstream algorithms and instead uses approaches which are simpler even than reference governer strategies (Fiani & Richalet, 1991; Gilbert & Tan, 1991).

Assume constraints, at every sample, on input and states as follows:

$$\underline{\Delta u} \leq \Delta u_k \leq \overline{\Delta u}; \quad \underline{u} \leq u_k \leq \overline{u}; \quad \underline{y} \leq y_k \leq \overline{y}, \tag{5}$$

where $\Delta u_k = u_k - u_{k-1}$ is the input increment or rate.

The simplest PFC approach deals only with input constraints and deploys a saturation approach, that is, if the proposed u_k violates (5), then move to the nearest value which does not. Readers should note that this saturation approach automatically avoids issues with integral windup and the like which can occur when using PID. Also, for systems with stable open-loop dynamics, this approach is usually safe, albeit potentially suboptimal.

The historic PFC literature (Richalet et al., 2011) deploys more involved strategies to cater for state constraints which are akin to reference governer approaches (Gilbert & Tan, 1991), though perhaps a little more cumbersome. The core principle is to deploy nested or parallel PFC loops. The outer loop supplies the target to the inner loop and is used to modify this target when there is an expectation that an unmodified target will lead to a constraint violation. Such an approach requires design and tuning of the outer loop, but also is inherently simplistic and not designed to consider a multitude of different constraints as in (5); consequently there is clear potential in modernising that approach.

First we summarise a core concept adopted as standard in the MPC literature for constraint handling and propose to use this concept in place of the historical PFC choices.

(1) For a suitable horizon m, compute the entire set of future predictions $y_{k+i|k} = P_i u_p + Q_i y_p + H_i \underset{\rightarrow k}{u} + d_k, \quad i = 1, \cdots, m$. Use the compact notation $\underset{\rightarrow k+1}{y} = Pu_p + Qy_p + H\underset{\rightarrow k}{u} + Ld_k$ to capture the output predictions in a single vector where $\underset{\rightarrow k+1}{y} = [y_{k+1|k}, y_{k+2|k}, \cdots]^T$.

(2) Combine the output constraints, output predictions and input constraints into a single set of linear inequalities of the form:

$$Cu_k \leq f_k, \tag{6}$$

$$C = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \\ HL \\ -HL \end{bmatrix}; \quad f = \begin{bmatrix} \overline{u} \\ -\underline{u} \\ \overline{\Delta u} \\ -\underline{\Delta u} \\ L\overline{y} \\ -L\underline{y} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ u_{k-1} \\ -u_{k-1} \\ Pu_p + Qy_p + Ld_k \\ -Pu_p - Qy_p - Ld_k \end{bmatrix},$$

where f_k depends on past data in u_p, y_p and on the limits. The horizon for the output predictions $\underset{\rightarrow k+1}{y}$, and thus the row dimension of H, should be long enough to capture all core dynamics!

(3) The predictions satisfy constraints iff (6) is satisfied and thus a conventional MPC algorithm will ensure this occurs and that is consider inequalities $Cu_k \leq f_k$ explicitly rather than an alternative constraint representation which may be suboptimal or approximate.

The proposed PFC constraint handling algorithm is summarised next. This uses a

single simple loop to select the u_k closest to the unconstrained solution of (4) which satisfies (6).

Algorithm 3.1. *At each sample:*

(1) Define the unconstrained value for u_k from (4).
(2) Define the vector f_k of (6) (it is noted that C does not change).
(3) Use a simple loop covering all the rows of C as follows:
  (a) Check the ith constraint that is the ith row of $Cu_k \leq f_k$ using $a_i = C_i u_k - f_{k,i}$.
  (b) If $a_i > 0$, then set $u_k = (f_{k,i})/C_i$, else leave u_k unchanged.

Theorem 3.1. *In the nominal case (when there is no change in d_k) and for stable open-loop processes, Algorithm 3.1 is guaranteed to be recursively feasible and moreover converge to a feasible value for u_k that is closest to the unconstrained choice.*

Proof. Assume feasibility at initiation and also note that for stable open-loop processes the predicted outputs are convergent for constant future inputs $u_{k+i} = u_k, \forall i > 0$. Consequently, if one has feasibility at sample $k-1$, then the choice $u_k = u_{k-1}$ must be feasible, that is satisfy (6). Hence, as long as u_{k-1} is a possible choice (which it must be as all constraints must satisfy $C_i u_{k-1} \leq f_{k,i}$), recursive feasibility is assured and a feasible solution will lie between u_{k-1} and the unconstrained u_k. Each constraint $C_i u_k \leq f_{k,i}$ will either lower or upper bound u_k; if $u_k < u_{k-1}$ then only the lower bounds can be active and if $u_k > u_{k-1}$ only the upper bounds. Hence, an active constraint $C_i u_k \leq f_{k,i}$ will bring u_k closer to u_{k-1} if violated by the unconstrained u_k but otherwise will have no affect. In consequence, the final u_k will be only as close to u_k as it needs to be to satisfy all the active constraints and thus, is also as close to the original unconstrained u_k as possible. □

Remark 3.1. *Because this approach (Algorithm 3.1) deploys a very simple* for-loop, *coding is simple and very fast and certainly far more simple than traditional MPC approaches which often use a quadratic program but equally, more systematic and probably quicker than the ad-hoc approaches common with PID. Nevertheless, the usage is only limited for single input single output (SISO) process, as PFC is rarely used to control a multi input multi output (MIMO) system due to its limited capability (Richalet et al., 2009). Nevertheless, this offset only occurs in the implied prediction, where the closed-loop response will only used the first sample of the input. Since the manipulated input u_k value is updated at each sample time, the final output still converges to the steady state target R.*

4. PFC using Laguerre functions

The main weakness of conventional PFC was the assumption with the predictions that the future input is constant. This same weakness is present in conventional MPC algorithms such as GPC and in fact equivalent restrictions also exist in the d.o.f. within dual-mode strategies. In an effort to ameloirate these and instead propose future input trajectories which were likely to be closer to those required in closed-loop, a few authors considered Laguerre function parameterisations (Khan & Rossiter, 2013; Rossiter et al., 2010; Wang, 2009). It was shown that despite being a relatively simple change in formulation, this helped significantly with trade-offs between the number of d.o.f. in the prediction class and the feasibility (ability to deal with constraints).

The purpose of this paper is to propose and demonstrate the potential benefits of a similar concept when applied to PFC. However we should note some fundamental differences:

- In PFC we deploy just one d.o.f. and thus can use just a single Laguerre function.
- A different trade off will be investigated, that is between tuning parameter λ and closed-loop performance achieved during both unconstrained scenarios and constraint handling scenarios

4.1. Definition of input trajectories using Laguerre functions

As we are not using the whole set of Laguerre functions and just a single one, it is easier just to state that single function. For a given pole a, the first Laguerre function is given as:

$$L_a(z) = L_a(1)[1 + az^{-1} + a^2z^{-2} + a^3z^{-3} + \cdots]. \tag{7}$$

(Typically $L(1) \neq 1$ when defining multiple Laguerre functions (Rossiter et al., 2010) although for this paper this detail is optional.) An underlying assumption within this paper is that the closed-loop input will converge to the steady-state with close to first order dynamics and thus with dynamics that can be represented by $L_a(z)$ plus some constant w. That is, ideally the future input predictions are defined as:

$$u_k = w_k + \eta_k; \quad u_{k+1|k} = w_k + a\eta_k; \quad u_{k+2|k} = w_k + a^2\eta_k; \quad \cdots, \tag{8}$$

where η_k is a scaling factor to be selected on-line, and w_k is a value to be defined rather than a d.o.f.. The reader will note that in effect w_k is the implied steady-state/asymptotic value for u_{k+i} within the predictions.

An almost equivalent definition could use the input increments and hence:

$$\Delta u_k = \nu_k; \quad \Delta u_{k+1|k} = a\nu_k; \quad \Delta u_{k+2|k} = a^2\nu_k; \quad \cdots, \tag{9}$$

although in this case the implied input trajectory would be:

$$u_k = u_{k-1} + \nu_k; \quad u_{k+1|k} = u_{k-1} + (1+a)\nu_k; \quad u_{k+2|k} = u_{k-1} + (1+a+a^2)\nu_k; \quad \cdots. \tag{10}$$

Next note the properties of the geometric sequence $1, 1+a, 1+a+a^2, \cdots$. It is known that

$$S_a = \sum_{i=0}^{\infty} a^i = \frac{1}{1-a} \quad \text{and} \quad \sum_{i=0}^{n} a^i = \frac{1-a^{n+1}}{1-a} = (1-a^{n+1})S_a. \tag{11}$$

Lemma 4.1. *The choices of (8,9) are not exactly equivalent and thus would lead to different results in general.*

Proof. Consider the implied control increments with the choices of (8,9). First the

choice (8) gives:

$$\begin{bmatrix} u_k \\ u_{k+1|k} \\ u_{k+2|k} \\ \vdots \end{bmatrix} = \begin{bmatrix} w_k + \eta_k \\ w_k + a\eta_k \\ w_k + a^2\eta_k \\ \vdots \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} \Delta u_k \\ \Delta u_{k+1} \\ \Delta u_{k+2} \\ \vdots \end{bmatrix} = \begin{bmatrix} w_k - u_{k-1} + \eta_k \\ (a-1)\eta_k \\ (a^2-a)\eta_k \\ \vdots \end{bmatrix}. \tag{12}$$

Using an equivalent notation, we can rewrite sequence (10) as follows:

$$\begin{bmatrix} \Delta u_k \\ \Delta u_{k+1|k} \\ \Delta u_{k+2|k} \\ \vdots \end{bmatrix} = \begin{bmatrix} \nu_k \\ a\nu_k \\ a^2\nu_k \\ \vdots \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} u_k \\ u_{k+1|k} \\ u_{k+2|k} \\ \vdots \end{bmatrix} = \begin{bmatrix} u_{k-1} + \nu_k \\ u_{k-1} + (1-a^2)S_a\nu_k \\ u_{k-1} + (1-a^3)S_a\nu_k \\ \vdots \end{bmatrix}. \tag{13}$$

Hence it is clear that there is a significant difference in these choices of parameterisation:

(1) Choice (8) allows the first increment Δu_k to be out of proportion to the remaining increments whereas for choice (9) this ratio is fixed. Choice (8) may be better where a disproportionately large (or small) first increment is needed.
(2) Irrespective of the choice of w, the relative sizes of respective elements are the same for all except the first increment, so for example, taking (12,13) in turn:

$$\{\frac{\Delta u_{k+i|k}}{\Delta u_{k+i+1|k}} = \frac{1}{a}\} \quad \text{or} \quad \{\frac{\Delta u_{k+i|k}}{\Delta u_{k+i+1|k}} = \frac{a^i - a^{i-1}}{a^{i+1} - a^i} = \frac{1}{a}\}. \tag{14}$$

Hence the two choices would be equivalent if and only if both $w_k - u_{k-1} + \eta_k = \nu_k$ and $(a-1)\eta_k = a\nu_k$ which would require a specific choice of w_k. □

Theorem 4.2. *The choice of parameterisation (8) has more flexibility than the choice (9) and thus, in general, is to be prefered.*

Proof. This follows immediately from the previous Lemma. Appropriate choices of η_k, ν_k make the increments identical for all bar the first, that is Δu_k. In this instance, parameterisation (8) has an additional d.o.f. w_k which can be exploited if desired and if not assigned an equivalent value to that implied in (9). □

Proposal: As was noted above, w_k is the asymptotic value of the input within the predictions and hence an obvious choice of w_k which eliminates the need for more design decisions is to set $w_k = E[u_{ss}]$, that is the expected steady-state input value required to remove offset. This would ensure the output predictions have zero offset asymptotically.

Corollary 4.3. *The prediction classes for PFC given in (2) and with Laguerre based on input increments (9) suffer from a critical weakness. In both cases the asymptotic value of the predicted output (not to be mixed up with the closed-loop output which has no asymptotic offset assuming stability) is highly unlikely to be close to the desired target of R. This is because, the value of u_k satisfying the PFC law definition $r_{k+n} = y_{k+n}$ in general will be inconsistent with $u_k = E[u_{ss}]$.*

4.2. Unbiased prediction and steady-state estimation

The output predictions with a single control increment were given in (3). With Laguerre based predictions, that is where $\underset{\rightarrow k}{u}$ is taken from, for example (12), this prediction can be reformulated as:

$$y_{k+n|k} = P_n u_p + Q_n y_p + \underbrace{H_n \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \end{bmatrix}}_{H_{nw}} w_k + \underbrace{H_n \begin{bmatrix} 1 \\ a \\ a^2 \\ \vdots \end{bmatrix}}_{H_{na}} \eta + d_k. \tag{15}$$

The reader will note from Corollary 4.3 that w_k is defined using the expected steady-state. In simple terms, a typical PFC computation of this value is given from:

$$y_{ss} = G_{ss} u_{ss} + d_k \quad \rightarrow \quad u_{ss} = \frac{y_{ss} - d_k}{G_{ss}} = \frac{R - d_k}{G_{ss}}, \tag{16}$$

where G_{ss} is the model steady-state gain, y_{ss} is the desired output steady-state (typically R) and d_k the offset between the model output and process measurement.

Remark 4.1. *Since the value of d_k is updated at each sample, the Independent Model (IM) structure is capable to handle some level of disturbance and parameter uncertainty and give zero asymptotic offset. Although the loop and its signals are impacted by measurement noise, as measurement noise is typically assumed to be a zero mean random variable, the impact of this on offset is typically ignored in the MPC literature. Of course there is an impact on loop sensitivity which is not a topic of this paper but an interested reader may refer to the work of Abdullah & Rossiter (2018b).*

4.3. Two PFC algorithms using Laguerre function predictions

Having defined both the input and output predictions, the PFC algorithm follows the same lines as in section 2, that is, choose the d.o.f. η such that (1,2) are satisfied.

Algorithm 4.1. LPFC: *The PFC control law using a Laguerre parameterisation of the input trajectory is defined as follows:*

(1) Define the input trajectory from (12) with $w_k = u_{ss,k}$ as defined in (16).
(2) Define the output prediction n-steps ahead using (15).
(3) Substituting prediction (15) into (1,2) the PFC control law can be defined as:

$$P_n u_p + Q_n y_p + H_{nw} w_k + H_{na} \eta + d_k = R - (R - y_k)\lambda^n, \tag{17}$$

$$\Rightarrow \quad \eta_k = \frac{R - (R - y_k)\lambda^n - P_n u_p - Q_n y_p - H_{nw} w_k - d_k}{H_{na}}; \quad u_k = w_k + \eta_k.$$

For completeness, as this has some relevance with constaint handling, a second algorithm is also defined.

Algorithm 4.2. LPFC2: *An alternative PFC control law using a Laguerre parameterisation of the input increment trajectory is defined as follows:*

(1) Define the input trajectory from (13).
(2) Define the output prediction n-steps ahead using (15) with (13) and hence define

$$y_{k+n|k} = P_n u_p + Q_n y_p + H_{nw} u_{k-1} + \underbrace{H_n \begin{bmatrix} 1 \\ (1-a^2)S_a \\ (1-a^3)S_a \\ \vdots \end{bmatrix}}_{H_{nn}} \nu_k + d_k. \tag{18}$$

(3) Substituting output predictions (18) into (1,2) the PFC control law can be defined as:

$$P_n u_p + Q_n y_p + H_{nw} u_{k-1} + H_{nn}\nu_k + d_k = R - (R - y_k)\lambda^n, \tag{19}$$

$$\Rightarrow \quad \nu_k = \frac{R - (R - y_k)\lambda^n - P_n u_p - Q_n y_p - H_{nw} u_{k-1} - d_k}{H_{nn}}; \quad u_k = u_{k-1} + \nu_k.$$

4.4. Constraint handling

The procedure for constraint handling is analogous to that discussed in section 3 and thus is not presented in detail. The core conceptual difference is that the d.o.f is now one, as either the parameter η_k or ν_k can be selected depending on the choice of input parameterisation and this means that, in principle, the input constraints need to be checked along the entire prediction horizon. However, given the maximum magnitude increments occur at the first sample, only Δu_k one needs to be checked and similarly, the maximum/minimum of u_{k+i} has a simple dependence on η_k, ν_k, w_k so again only one value needs to be checked.

As in Algorithm 3.1, the aim is to choose the d.o.f. as close as possible to their unconstrained values, and subject to (5). Nevertheless, there are some subtleties which are worth highlighting and link to feasibility.

Lemma 4.4. *For the nominal case (where the is no large change in d_k), recursive feasibility is guaranteed with input prediction class (13), irrespective of the choice of target R.*

Proof. Assuming feasibility at sample $k-1$, then the choice $\Delta u_{k+i|k} = \Delta u_{k+i|k-1}$, $\forall i \geq 0$ will give rise to predictions which satisfy constraints (5). The choice $\nu_k = a\nu_{k-1}$ will enable this choice of future inputs and thus a feasible solution exists at the current sample and, clearly, this statement can be made recursively. □

Lemma 4.5. *For the nominal case, recursive feasibility is not guaranteed with input prediction class (12).*

Proof. The potential weakness with input prediction class (12) is emphasised in the first term $u_k = u_{ss,k} + \eta_k$ as this contains a value, specifically $u_{ss,k}$ which may or may not be feasible. Moreover, consideration of the implied increments shows that $\Delta u_k = u_{ss,k} - u_{k-1} + \eta_k$ could be very large if there is a significant change in $u_{ss,k}$

(that is, $u_{ss,k} \neq u_{ss,k-1}$). To be more precise, the sequence of proposed inputs from the previous sample can be laid alongside the proposed sequence at the current sample:

$$\underset{\rightarrow k|k-1}{u} = \begin{bmatrix} u_{ss,k-1} + a\eta_{k-1} \\ u_{ss,k-1} + a^2\eta_{k-1} \\ u_{ss,k-1} + a^3\eta_{k-1} \\ \vdots \end{bmatrix} \quad : \quad \underset{\rightarrow k|k}{u} = \begin{bmatrix} u_{ss,k} + \eta_k \\ u_{ss,k} + a\eta_{k-1} \\ u_{ss,k} + a^2\eta_k \\ \vdots \end{bmatrix}. \tag{20}$$

From these it is clear that one can only ensure $\underset{\rightarrow k|k-1}{u} = \underset{\rightarrow k|k}{u}$ if $u_{ss,k} = u_{ss,k-1}$ (equivalently w_k) is unchanged. Without the ability to remain on the same prediction class, recursive feasibility cannot be assured. □

Theorem 4.6. *In order to ensure recursive feasibility while using prediction class (12), the user must retain the option to modify $u_{ss,k}$ as required.*

Proof. It is a consequence of Lemma 4.4 whereby the option to choose $u_{ss,k} = u_{ss,k-1}$ enables the selection of $\underset{\rightarrow k|k-1}{u} = \underset{\rightarrow k|k}{u}$, hence garanteeing feasibility. □

Readers may note that the requirement in Theorem 4.6 is analogous to reference governer strategies (Gilbert & Tan, 1991) and is unsurprising and indeed also a well known issue within mainstream MPC. That is, large changes in the target can give rise to transient infeasibility where there is a terminal constraint as implicit with input trajectory (12) and this is easiest dealt with by slowing the change in target (equivalently modifying the implied terminal constraint). This paper will use examples to compare such a strategy with the use of (13) which is more analogous to GPC (Clarke & Mohtadi, 1989) in not having an implied terminal constraint and thus does not require this additional check.

5. Simulation studies

The simulation studies will be sectioned into three main themes and comparisons will be made between PFC, LPFC and LPFC2.

(1) Compare the predictions arising from the different algorithms and the extent to which these give good expectations of good performance and consistent decision making.
(2) Consideration of tuning efficacy and performance.
(3) Consideration of constraint handling efficacy.

Several non-first order dynamics examples will be used to emphasise a variety of challenging characteristics (for example non-minimum phase and unstable dynamics), as it is known that conventional PFC is often inadequate for such systems. These examples are:

(1) A second order non-minimum phase system with zero at 0.4:

$$G_1 = \frac{-0.04z^{-1} + 0.1z^{-2}}{1 - 1.4z^{-1} + 0.45z^{-2}}; \quad n = 5; \quad \lambda = 0.7. \tag{21}$$

(2) A third order over-damped system with poles at 0.9241, 0.9869 ± 0.03339:

$$G_2 = 10^{-3}\frac{-0.1665z^{-1} + 0.0157z^{-2} - 0.1505}{1 - 2.898z^{-1} + 2.7993z^{-2} - 0.9012z^{-3}}; \quad n = 25; \quad \lambda = 0.96. \quad (22)$$

(3) A second order unstable system with poles at 0.63 and 1.27.

$$G_3 = \frac{0.1z^{-1} - 0.2z^{-2}}{1 - 1.9z^{-1} + 0.8z^{-2}}; \quad n = 5; \quad \lambda = 0.7. \quad (23)$$

Remark 5.1. *The selection of λ depends on a user preference where the range should be in between $0 < \lambda < 1$. This parameter is directly related to the desired closed-loop time response (CLTR), where $\lambda = e^{-3T_s/CLTR}$ and T_s is the sampling time (Richalet et al., 2009). Obviously, a smaller value of λ leads to a faster convergence and vice versa. For the coincidence horizon n, this work follows the conjecture given in (Rossiter & Haber, 2015) where the point is selected in between 40% to 80% rise of the step input response to the steady-state value.*

5.1. Prediction consistency and recursive decision making

A core underlying assumption in well posed MPC algorithms is that the *optimised* prediction at the current sample is reasonable and hence, could be re-used at the next sample. However, as these examples will show that is often not the case with a conventional PFC law due to the prediction assumption that the future input is constant. Where anything other than an open-loop dynamic is required, a constant input cannot deliver the desired dynamic and thus to embed this into the predictions automatically creates a mismatch in expectations.

Figure 1 shows the optimised predictions for example G_1 at the first sample from which it is clear that although all the predictions satisfy $r_{k+n} = y_{k+n}$ $(n = 5)$ as defined by (2) but elsewhere they differ significantly from the target trajectory r_k within the exception of LPFC which remains close (apart from in early transients for which, due to the non-minimum phase characteristic, tracking r_k is impossible) and also has the correct asymptotic value. The input trajectories show that the flexibility in LPFC allows a large initial input to get a fast transient and then a gradual decay to the desired steady-state. By contrast, PFC tries to manage everything with a constant input and thus fails. LPFC2 has a different weakness: as the increments Δu_k all have the same sign, the required early increments to satisfy (2) inevitably lead to an asymptotic input trajectory which grows too large and ironically, also imply a less aggressive initial input move which could imply relatively slow transients compared to PFC and LPFC.

Figure 2 shows similar behaviour for example G_2, although in this case the disparaity between the target r_k is amplified much further due to the relatively slow underlying dominant dynamics of the open-loop poles (real part 0.98), and thus open-loop predictions, compared to the desired pole ($\lambda = 0.96$).

The figure for G_3 is omitted as, being open-loop unstable, the predictions are divergent. It so happens that, in the constraint free case, effective decision making may still result from using these predictions over a short output horizon as evidenced by the numerous examples in the literature using both GPC and PFC on open-loop unstable processes (e.g. Rossiter et al. (1998)). Moreover,

Remark 5.2. *It is worth noting that, an IM structure is not suitable for open-loop*

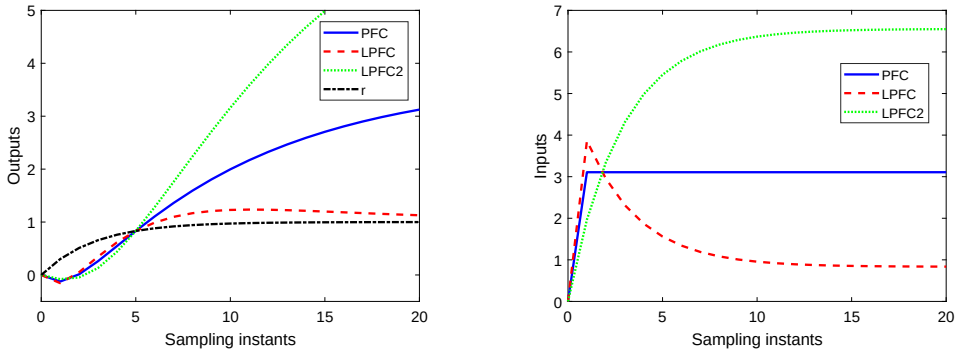

Figure 1. Output and input predictions for PFC, LPFC, LPFC2 for example G_1.

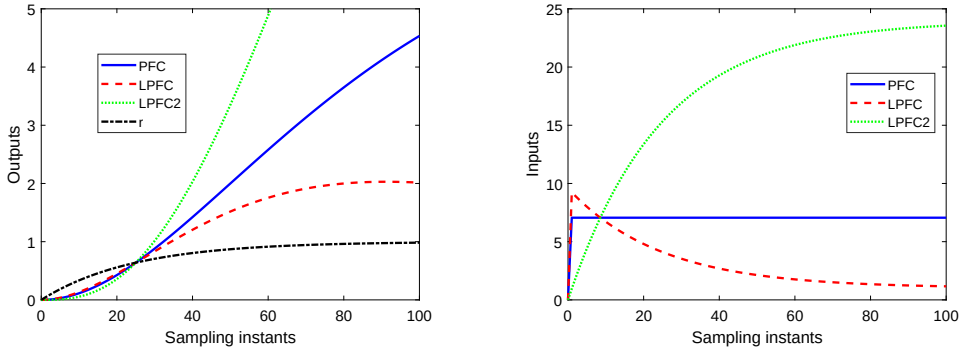

Figure 2. Output and input predictions for PFC, LPFC, LPFC2 for example G_2.

unstable processes as would be evident should any uncertainty or noise be introduced. In such cases a classical PFC approach often deploys a cascade structure (Richalet et al., 2009), but for convenience, the results of this paper carry across directly as long as a suitable alternative prediction model is used (Clarke & Mohtadi, 1989; Rossiter, 2018). Such details are not core to this paper and thus omitted.

5.2. Tuning efficacy of PFC, LPFC and LPFC

The tuning efficacy of PFC has been discussed elsewhere (Rossiter & Haber, 2015; Rossiter et al., 2016) and so this section illustrates whether the proposed adaptions of LPFC, LPFC2 change any of those insights or not. The underlying issue is that, with the exception of cases where one can choose $n = 1$, the tuning parameter λ may have a weak correlation with the closed-loop pole that results.

Figures 3, 4, 5, 6 overlap the closed-loop behaviour with the original target for examples G_1, G_2, G_3. From these figures three obvious conclusions are clear.

(1) None of the algorithms is able to get close to the desired dynamic/target trajctory when $n \gg 1$.
(2) LPFC is marginally faster during the intermediate transients than PFC whereas LPFC2 has slow initial transients but ultimately converges to the steady-state slightly more quickly.
(3) Nevertheless, the closed-loop responses do speed up with the choice of smaller λ

(see figure 6).

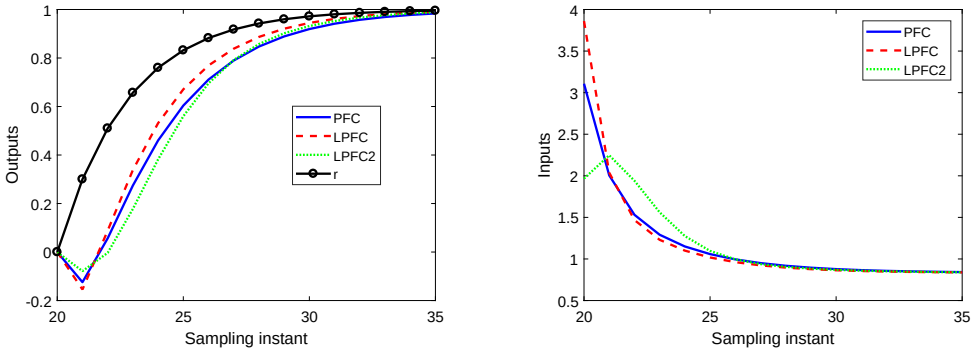

Figure 3. Closed-loop output and input behaviour for PFC, LPFC, LPFC2 for example G_1.

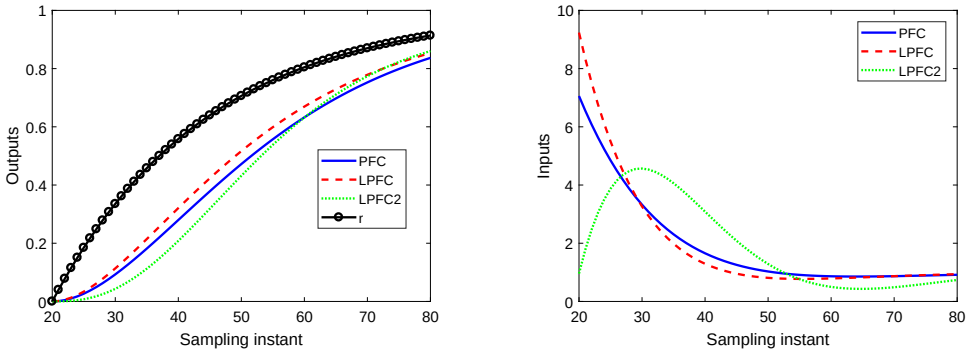

Figure 4. Closed-loop output and input behaviour for PFC, LPFC, LPFC2 for example G_2.

5.3. Constraint handling

The reader may be puzzled as to why the improved prediction consistency illustrated in section 5.1 seems to have minimal impact on the efficacy of tuning and hence question whether the move to LPFC has any benefits? The answer to this becomes clear when one considers the constraint handling scenarios. When doing constraint handling and in particular wanting assurances of recusive feasibility (see section 4.4), consistency between predicitions and the actual closed-loop behaviour is essential. The decision on how to limit u_k, η, ν_k to ensure predictions meet constraints will be ill-posed if those predictions are not representative and, of particular concern, these decisions could be unnecessarily conservative thus leadng to far slower performance than necessary.

- This section will demonstrate how the inconsistency in PFC predictions can lead to extreme conservativism, whereas this is less likely to occur with LPFC.
- The section also shows a scenario where LPFC2 might be preferred to LPFC as it gives a much more straightforward mechanism for coping with large target changes.

Add constraints $\overline{u} = 1.2, \overline{\Delta u} = 0.5, \overline{y} = 1.1$ to example G_1 and limits $\overline{u} = 2, \overline{\Delta u} =$

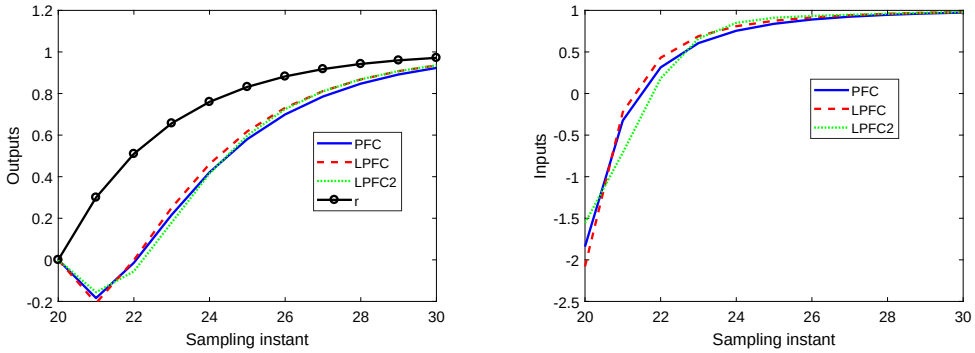

Figure 5. Closed-loop output and input behaviour for PFC, LPFC, LPFC2 for example G_3.

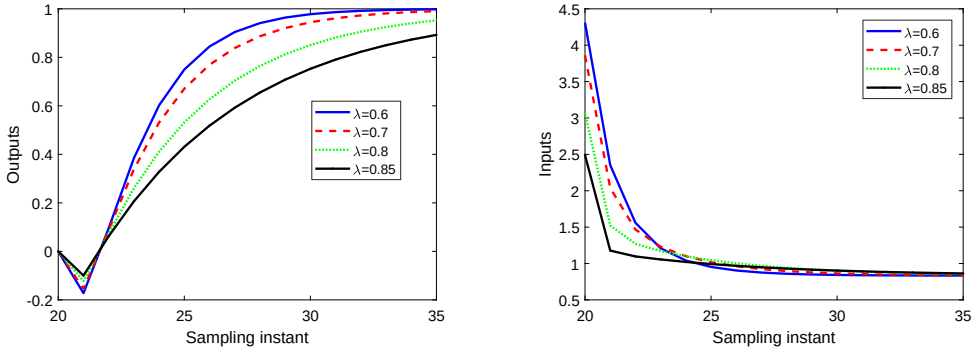

Figure 6. Closed-loop output and input behaviour for LPFC for example G_1 with various λ.

$0.4, \overline{y} = 1.1$ to example G_2. Then the corresponding closed-loop simulations for a target of $R = 1$ are shown in Figures 7, 8. It is clear that LPFC has by far the best performance because it exploits the input most effectively. By contrast, because PFC assumes a constant future input, the input values available become highly restricted to be close to the steady-state because otherwise the long-range output predictions would exceed the upper output limit; this is obvious in the input figures where for PFC the input goes barely above its steady-state value. LPFC2 has a slow initial transient, again because the shape of input trajectory (9) will only meet the upper constraints in the long term if, for example, the first term $\Delta u_k = a\nu_k \ll \overline{u}$. However, in the medium term LPFC2 can exploit input values beyond u_{ss} and thus eventually converges in a timescale not dissimilar from LPFC.

Now consider a case where a simplistic implementation of LPFC fails, whereas PFC and LPFC2 do not. Add rate constraints to the input predictions for LPFC as given in (12).

$$\begin{bmatrix} |\Delta u_k| \\ |\Delta u_{k+1}| \\ \vdots \end{bmatrix} = \begin{bmatrix} |w_k - u_{k-1} + \eta_k| \\ |(a-1)\eta_k| \\ \vdots \end{bmatrix} \leq \begin{bmatrix} \overline{\Delta u} \\ \overline{\Delta u} \\ \vdots \end{bmatrix}. \tag{24}$$

Now, consider the case where $\overline{\Delta u} = 0.1, w_k = 1, u_{k-1} = 0, a = 0.8$. The first two

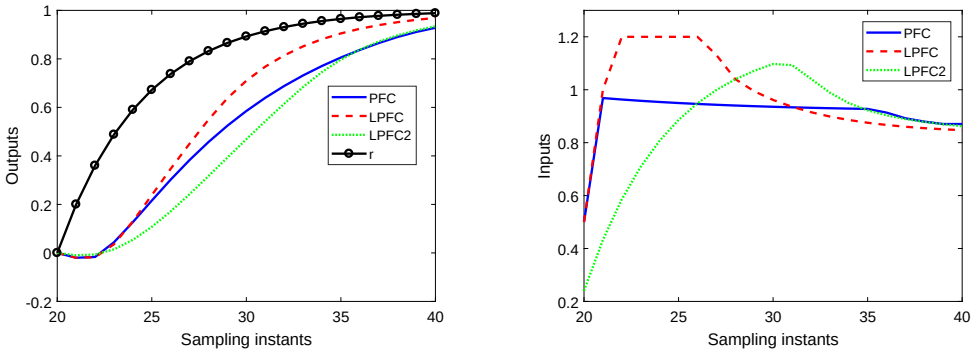

Figure 7. Closed-loop output and input behaviour for constrained LPFC for example G_1.

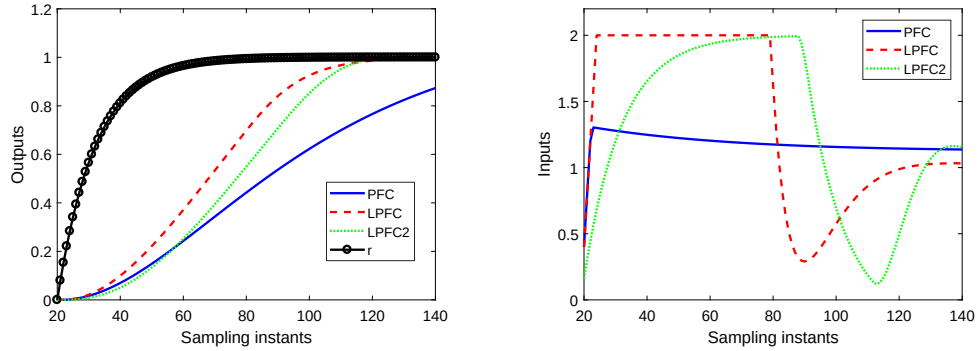

Figure 8. Closed-loop output and input behaviour for constrained LPFC for example G_2.

inequalities reduce to:

$$\begin{bmatrix} |1+\eta_k| \\ |-0.2\eta_k| \\ \vdots \end{bmatrix} \leq \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} \quad \Rightarrow \quad \begin{matrix} -1.1 \leq \eta_k \leq -0.9 \\ |\eta| \leq 0.5 \end{matrix} . \tag{25}$$

Clearly these two conditions are inconsistent and thus LPFC is infeasible and the main factor is the input rate constraint. A simple scenario which would give rise to this inconsistency is when w_k changes significantly from one sample to the next (say due to a large change in set-point R). The requirement for the input sequence to both meet the new steady-state (while following the given decay rate) and beginning from the current u_{k-1} can easily be in conflict with rate constraints $\overline{\Delta u}$. This issue is well understood in the main MPC literature and a reference governing or softening approach will be needed to avoid infeasibility.

Remark 5.3. *The reader may also be able to come up with scenarios where a large change in w_k gives rise to inconsistency between output constraints and other constraints. In general the easiest solution, which is standard in the mainstream MPC literature, is to relax the implied terminal constraint, that is slow the rate of change of w_k as much as required.*

6. Demonstrations on laboratory equipment

This section demonstrates and compares the performance of PFC and LPFC in controlling a real laboratory process that is a Quanser SRV02 servo based unit (see Figure 9). This system is powered by a Quanser VoltPAQX1 amplifier that comes with National Instrument ELVIS II+ multifunctional data acquisition device. The control algorithms are employed using National Instrument LabVIEW software in personal computer which is connected to the plant via USB wire.

Figure 9. Quanser SRV02 servo based unit.

The control objective for this case is to track the desired angular position of the servo, $\theta(t)$ by regulating the supplied voltage, $V(t)$. The mathematical model is derived based on differential equation and given as (for more detail explanation and derivation refer to Quanser (2012)):

$$J_{eq}\frac{d^2}{dt^2}\theta(t) + B_{eq}\frac{d}{dt}\theta(t) = A_m V(t), \tag{26}$$

where $J_{eq} = 0.00213$ kgm^2 is the equivalent moment of inertia, $B_{eq} = 0.0844$ Nms/rad is the equivalent viscous damping parameter and $A_m = 0.129$ Nm/V is the actuator gain. By rearrange the model (26) and discretise it with sampling time $0.02s$, the transfer function of angular position to voltage input can be formed as:

$$G_4 = \frac{0.0095z^{-1} + 0.0073z^{-2}}{1 - 1.45z^{-1} + 0.45z^{-2}}. \tag{27}$$

Both of the controllers are tuned with $\lambda = 0.7$ and $n = 8$. Similar conclusions as in previous section can be seen from Figure 10, where LPFC is more effective in tracking the set point and converging faster (closer to the desired λ) than PFC.

Next, the following constraints are added to the process: -6 V $< V <$ 6 V, -3 V $< \Delta V <$ 3 V, -0.8 rad $< \theta <$ 0.8 rad. Due to the advantage of employing better prediction consistency and well-posed decision making, LPFC manages to utilise the extra d.o.f in its future input trajectory to satisfy all the constraints better than PFC (refer Figure 11). It is notable that the constraint handling weaknesses implicit in PFC (linked to the assumed steady-state input in the predictions implying output constraint violations), means that PFC is very slow to converge to the output limit of 0.8 whereas LPFC is not affected by this weakness!

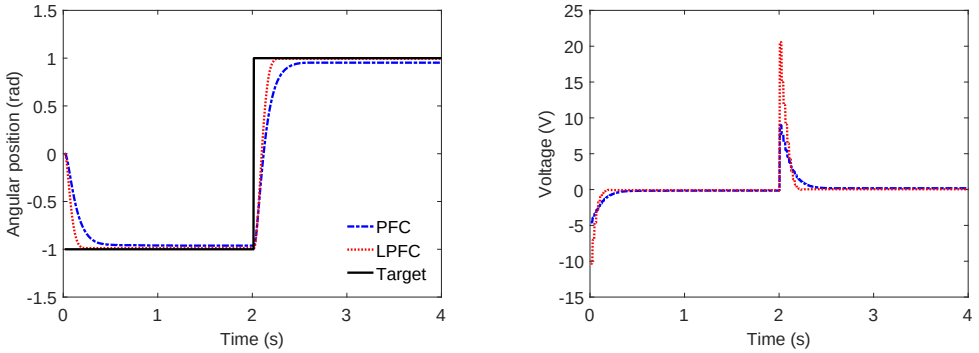

Figure 10. Performance of PFC and LPFC in tracking the Quanser SVR02 servo position.

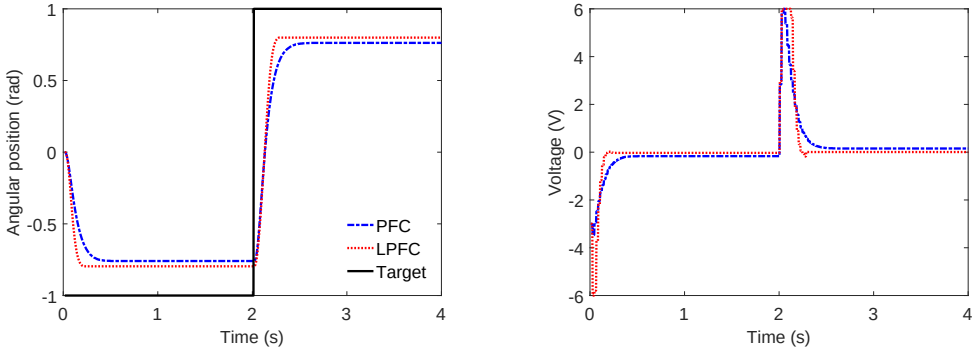

Figure 11. Constrained performance of PFC and LPFC in tracking the Quanser SVR02 servo position.

Remark 6.1. *The process used for this example has an integrator where one of its poles reside on the unit circle. Using the conventional PFC approach is not recommended since then the open loop prediction is divergent which inevitably will imply output constraint violations. The normal practice is to employ a cascade structure where the prediction is stabilised first before implementing the control law (Richalet et al., 2009). However, this cascade from of PFC is more complex to tune and has less clear cut properties so the discussion is beyond the scope of this paper. Conversely, LPFC can still be used since its future input dynamics will converge to zero assuming one uses the appropriate choice of* $u_{ss} = 0$ *(Abdullah & Rossiter , 2018a).*

Remark 6.2. *It is noted that there is a small offset error in Figure 10, which is far more obvious when using the conventional PFC. The main reason behind this is because the Independent Model (IM) structure used in the control law is not very effective in handling parameter uncertainty of the open-loop divergent process. This sensitivity can be improved by implementing other alternative prediction structures such as CARIMA or T-filter (Rossiter, 2018), but again the discussion of this topic is beyond the remit of this paper.*

7. Conclusions

This paper has taken the start point that Laguerre based input parameterisations have been effective for mainstream MPC algorithms and thus it is worth investigating their efficacy with PFC. We have proposed the use of Laguerre based predictions for PFC and evaluated the efficacy of this, compared to a more conventional PFC algorithm, using two different parameterisations: (i) one based on the inputs and (ii) another based on the input increments.

The most obvious conclusion is that using Laguerre functions can offer some clear benefits and more specifically, because it enables better consistency between the predictions at each sample and the resulting closed-loop behaviour, it enables more accurate constraint handling because satisfaction of constraints by the predictions is now a better representation of the actual future. By contrast, especially with regard to output constraints, a traditional PFC algorithm may have to adopt some quite conservative assumptions in order to ensure recursive feasibility and thus sacrifices performance.

The paper gives a proposal for a more formal, but computationally simple, constraint handling policy which has more rigour than traditional PFC constraint handling approaches. Then using this, a comparison of the two alternative parameterisations indicates that in most cases, mapping Laguerre directly onto the inputs is preferable to mapping onto the input increments as this enables faster transients and a better usage of the full input range. However, the one downside is the need to utilise an implied terminal constraint and, as is well known in the literature, terminal constraints can cause conflicts with other constraints and thus, at times, need to be managed carefully. Further investigations into computationally simple and efficient ways of doing this for PFC form an immediate future work.

Some other aspects which would be interesting to address next include: (i) to what extent does using a Laguerre parameterisation change, for better or worse, the underlying loop sensitivity compared to conventional PFC and (ii) is there potential to exploit other input parameterisations and if so, how can one choose these systematically and in a computationally simple manner.

References

Abdullah, M., Rossiter, J. A. & Haber, R. (2017). Development of constrained predictive functional control using Laguerre function based prediction. In *2017 20th IFAC World Congress*, (p. 10705–10710).

Abdullah, M. & Rossiter, J. A. (2018). Alternative method for predictive functional control to handle an integrating process. In *2018 UKACC Control Conference*.

Abdullah, M. & Rossiter, J. A. (2018). A formal sensitivity analysis for Laguerre based predictive functional control. In *2018 UKACC Control Conference*.

Camacho, E. F. & Bordons, C. (1999). *Model Predictive Control*. Berlin: Springer.

Changenet, C., Charver, J. N., Gehin, D., Sicard, F. & Charmel, B. (2008). Predictive functional control of an expansion valve for controlling the evaporator superheat, *IMechE Journal of Systems and Control Engineering*, 222(6), 571–582.

Clarke, D. W. & Mohtadi, C. (1989). Properties of generalized predictive control. *Automatica*, 25(6), 859–875.

Fallasohi, H., Ligeret, C., & Lin-shi, X. (2010). Predictive functional control of an expansion valve for minimizing the superheat of an evaporator, *International Journal of Refrigeration*, 33, 409–418.

Fiani, P. & Richalet, J. (1991). Handling input and state constraints in predictive functional

control, in *Proceedings of the 30th IEEE Conference on Decision and Control*, (p. 985–990).
Gilbert, E. & Tan, K. (1991). Linear systems with state and control constraints: The theory and application of maximal admissible sets. *IEEE Transactions Automatic Control*, 36, 1008–1020.
Haber, R., Bars, R. & Schmitz, U. (2011). *Predictive control in process engineering: From the basics to the applications, Chapter 11: Predictive functional control*, Weinheim: Wiley-VCH.
Haber, R., Schmitz, U. & Zabet, K. (2014). Implementation of PFC (predictive functional control) in petrochemical plant. In *2014 IFAC World Congress*, (p. 5333–5338).
Haber, R., Rossiter, J. A. & Zabet, K. (2016). An alternative for PID control: Predictive functional control - A tutorial. In *2016 American Control Conference*, (p. 6935–6940).
Khadir, M., & Ringwood, J. (2005). Stability issues for first order predictive functional controllers: Extension to handle higher order internal models. In *International Conference on Computer systems and Information Technology*, (p. 174-179).
Khadir, M. & Ringwood, J. (2008). Extension of first order predictive functional controllers to handle higher order internal models. *Int. Journal of Applied Mathematics and Comp. Science*, 18(2), 229–239.
Khan, B., & Rossiter, J.A. (2013). Alternative parameterisation within predictive control: A systematic selection. *International Journal of Control*, 86(8), 1397–1409.
Rawlings J.B. & Mayne, D.Q. (2009). *Model predictive control: Theory and design*. Nob Hill Publication.
Muehlebach, M. & DAndrea, R. (2017). Basis functions design for the approximation of constrained linear quadratic regulator problems encountered in model predictive control, in *2017 IEEE 56th Annual Conference on Decision and Control*, (p. 6189–6196) .
Muske, K.R. & Rawlings, J.B. (1993), Model predictive control with linear models, *AIChE Journal*, 39(2), 262–287.
Quanser user manual SRV02 rotary servo based unit set up and configuration. Quanser Inc, 2012.
Richalet, J., Rault, A., Testud, J., & Papon, J. (1978). Model predictive heuristic control: Applications to industrial processes, *Automatica*, 14, 413–428.
Richalet, J. (1993). Industrial applications of model based predictive control, *Automatica*, 29(5), 1251–1274.
Richalet, J. & O' Donovan, D. (2009). *Predictive functional control: Principles and industrial applications*. Springer-Verlag.
Richalet, J. & O' Donovan, D. (2011). Elementary predictive functional control: A tutorial. In *Int. Symposium on Advanced Control of Industrial Processes*, (p. 306–313).
Rossiter, J.A., Wang, L., & Valencia-Palomo, G. (2010). Efficient algorithms for trading off feasibility and performance in predictive control. *International Journal of Control*, 83(4), 789–797.
Rossiter, J.A., & Haber, R. (2015). The effect of coincidence horizon on predictive functional control, *Processes*, 3(1), 25–45.
Rossiter, J.A. (2015). Input shaping for PFC: how and why?, *Journal of Control and Decision*, 1–14.
Rossiter, J.A. (2016). A priori stability results for PFC. *International Journal of Control*, 90(2), 305–313.
Rossiter, J.A., Haber, R., & Zabet, K. (2016). Pole-placement predictive functional control for over-damped systems with real poles, *ISA Transactions*, 61, 229–239.
Rossiter, J.A., Kouvaritakis, B., & Rice, M. (1998). A numerically robust state-space approach to stable predictive control strategies. *Automatica*, 34(1), 65–73.
Rossiter, J.A. (2018). *A first course in predictive control: 2nd edition*. CRC press.
Scokaert, P.O. & Rawlings, J.B. (1998). Constrained linear quadratic regulation. *IEEE Transactions on Automatic Control*, 43(8), 1163–1169.
Wang, L. (2009). *Model predictive control system design and implementation using MATLAB*. Springer Science & Business Media.
Zabet, K., Rossiter,J.A., Haber, R., & Abdullah, M. (2017). Pole-placement predictive func-

tional control for under-damped systems with real numbers algebra, *ISA Transactions*, 71(2), 403–414.