Article:
Dughman, S.S. and Rossiter, J.A. (2017) Systematic and effective embedding of feedforward of target information into MPC. International Journal of Control, 93 (1). pp. 98-112. ISSN: 0020-7179

https://doi.org/10.1080/00207179.2017.1281439

International Journal of Control
Vol. 00, No. 00, Month 200x, 1–23

Systematic and effective embedding of feedforward of target information into MPC

S.S. Dughman*, J. A. Rossiter*,
(*Received 00 Month 200x; final version received 00 Month 200x*)

* *Department of Automatic Control and Systems Engineering, University of Sheffield, UK, S1 3JD. Email: ssdughman1@sheffield.ac.uk, j.a.rossiter@sheffield.ac.uk*

ISSN: 0020-7179 print/ISSN 1366-5820 online

DOI: 10.1080/0020717YYxxxxxxxx
http://www.informaworld.com

Abstract: Discussions on how to make effective use of advance information on target changes are discussed relatively rarely in the predictive control literature. While earlier work has indicated that the default solutions from conventional predictive control algorithms are often poor, very little work has proposed systematic alternatives. This paper proposes an embedding structure for utilising advance information on target changes within an optimum predictive control law. The proposed embedding is shown to be systematic and beneficial. Moreover, it allows for easy extension to deal with more challenging scenarios such as unreachable set points and guarantees of convergence/stability in the uncertain case.

Keywords: OMPC, guaranteed stability, feedforward, constraint handling

1 Introduction

Model Predictive Control (MPC) has been widely and successfully applied (Qin and Badgwell 2003, Richalet et al. 1978, Fallasohi et al. 2010), primarily because of its ability to handle input and state constraints and multivariable processes in a systematic fashion. Nevertheless, there are some aspects of the algorithm which lack comprehensive systematic approaches and in particular one of these is the effective use of advance information of the target (Goodwin et al. 2011, Dugham and Rossiter 2016). While original work on MPC (Clarke and Mohtadi 1989) argued that advance information was included within the optimisation and therefore assumed this was helpful, it has been noted subsequently (Rossiter and Grinnell 1996, Valencia-Palomo et al. 2014) that in fact this information is often not used systematically and thus can lead to a degradation in performance rather than an improvement.

The reason for this apparent contradiction is relatively simply to understand. In a traditional MPC approach using either open-loop or closed loop predictions (Scokaert and Rawlings 1998, Rossiter et al. 1998), the degrees of freedom (d.o.f.) are focussed on immediate transients. If advance knowledge of the set point changes means that such changes are many samples in the future, these are not contemporaneous with the d.o.f. and thus the d.o.f. are ineffective in using this information (Valencia-Palomo et al. 2014).

The above arguments pertain to the constrained case as well as the unconstrained case. More recent work in the literature has focussed on issues linked to the interaction between set point changes and feasibility (Shead et al. 2010, Ferramosca et al. 2009, Limón et al. 2008). Specifically, feasibility issues tend to apply most to dual-mode MPC approaches as these include a terminal constraint, that is, the predicted state n_c steps into the future must be within a specified set in order to be sure that the predicted trajectories satisfy constraints. A significant change in the target implies a significant change in the terminal constraint and it may not be possible to find a trajectory that satisfies this terminal constraint, as well as transient constraints; this is denoted as infeasibility. Infeasibility due to target changes can take two forms: (i) the target is infeasible only in transients (Rossiter et al. 1996, Rao and Rawlings 1999, Rossiter 2006, Limón et al. 2008) and (ii) the target is permanently unreachable Rawlings et al. (2008), Shead et al. (2010).

A key point to note is that the literature has mainly focussed on what will be denoted here as a regulation problem in that it is assumed the target is fixed. Even when the target changes, this change occurs instantaneously and is assumed fixed in the future. Therefore, there is no advance knowledge of target changes deployed in the proposed algorithms. Where advance information has been deployed in a simplistic manner through the cost function, the default solution is often poor. This paper investigates the potential improvements in performance that are possible if advance information is deployed more effectively and also answers questions over how much advance information is useful as well as effective structures for embedding this information systematically. The paper makes three key contributions, but in related areas. Section 2 will summarise key algorithms, definitions and concepts in the literature associated to dual-mode MPC. Section 3 first investigates the issue of how much advance knowledge is useful in the unconstrained case and then proposes a mechanism for how this information can be embedded effectively thus forming a solid foundation for the constrained case. Hence, this section proposes a new formulation of dual-mode MPC to exploit future target information in a systematic and insightful way. Section 4 extends the new formulation for constraint handling scenarios to those where the desired target is infeasible, either in transients and/or steady-state and again shows the benefits of the proposed embedding structure for advance information. Section 5 focusses on robust design and demonstrates how the proposed embedding structure enables straightforward extensions for parameter uncertainty thus enabling simple MPC algorithms which have robustness and convergence guarantees during constraint handling. Finally, it is emphasised here that the focus here is on simple insights which can be easily understood and implemented. Hence, it is noticeable that the proposed approach requires only a conventional quadratic programming

optimisation, thus one that can be deployed easily within a standard industrial MPC algorithm.

2 Background on MPC tracking algorithms

This section summarises the key modelling assumptions, notation and some typical MPC algorithms in the literature which make use of information about target changes.

2.1 *System model and constraints*

This paper will use a state space model:

$$x_{k+1} = Ax_k + Bu_k, \quad y_k = Cx_k + p_k \tag{1}$$

where x_k, y_k, u_k, p_k are states, process output, process inputs and output disturbance respectively at sample k and A, B, C are matrices defining the model. Assume constraints, at every sample, on input and states as follows:

$$\underline{u} \leq u_k \leq \overline{u}; \quad \underline{x} \leq x_k \leq \overline{x} \tag{2}$$

More complex constraints can also be included without any change to the algorithms and concepts presented here.

2.2 *Performance index*

A typical MPC strategy proposes a sequence of candidate future input moves which are expected to give the best predicted performance, where performance is assessed using a defined performance index. Usually, MPC utilises only the first move of the control candidate sequence and ongoing measurement and optimisation are used to continually improve the planning at each sample. A common performance index (e.g. Rossiter (2003)) penalises the weighted squares of both predicted tracking errors and the control deviations from steady-state, that is:

$$J = \sum_{i=1}^{\infty} (x_{k+i+1} - x_{ss|k+i+1})^T Q (x_{k+i+1} - x_{ss|k+i+1}) + (u_{k+i} - u_{ss|k})^T R (u_{k+i} - u_{ss|k}) \tag{3}$$

where $u_{ss|k}, x_{ss|k}$ are the estimated steady-states of the input and states which enable $y \to r_k$ asymptotically, r_k being the desired target at sample k. Unbiased definitions of $u_{ss|k}, x_{ss|k}$ and their linear dependence on current disturbance estimate p_k and target r_k are well known in the literature (e.g. (Muske and Rawlings 1993)) and can be defined for suitable K_{xr}, K_{ur} as follows:

$$\begin{bmatrix} x_{ss|k} \\ u_{ss|k} \end{bmatrix} = \begin{bmatrix} K_{xr} \\ K_{ur} \end{bmatrix} (r_k - p_k) \tag{4}$$

Remark 1 : This paper focuses on infinite horizon algorithms due to their superior a priori stability properties. To simplify the presentation of the algebra, the disturbance estimate p_k is omitted from the equations hereafter; it is straightforward to include where required and is included in some of the numerical illustrations.

2.3 *Degrees of freedom (d.o.f) and autonomous models for prediction*

It is common to define the degrees of freedom within the predictions as the first n_c predicted control increments (or moves), that is $u_k, \cdots, u_{k+n_c-1}$. For convenience, with infinite horizon

algorithms the d.o.f can be equivalently parametrised (Rossiter et al. 1998) as perturbations c_k about a nominal stabilising control law.

$$\begin{aligned} u_k - u_{ss|k} &= -K(x_k - x_{ss|k}) + c_k; \; i < n_c \\ u_k - u_{ss|k} &= -K(x_k - x_{ss|k}); \qquad i \geq n_c \end{aligned} \tag{5}$$

The predicted state and input evolution is conveniently captured by combining (1,4,5). Hence, with $\Phi = A - BK$, a one-step ahead prediction model is:

$$\begin{aligned} x_{k+1+i} &= \Phi x_{k+i} + [I - \Phi]K_{xr}(r_{k+1+i}) + Bc_k \\ u_{k+i} &= -Kx_{k+i} + [KK_{xr} + K_{ur}](r_{k+1+i}) + c_k \end{aligned} \tag{6}$$

It is convenient to describe the predictions (6)using an autonomous model formulation (Kouvaritakis et al. 2000) whose states also include any information available about the d.o.f. c_k and the future target r_k at sample k. Such a model is given as follows:

$$Z_{k+1} = \Psi Z_k; \quad Z_k = [x_k^T, \underset{\rightarrow k}{c}^T, \underset{\rightarrow k+1}{r}^T]^T; \quad \Psi = \begin{bmatrix} \Phi & [B, 0, .., 0] & [(\Phi - I)K_{xr}, 0, ..., 0] \\ 0 & D_c & 0 \\ 0 & 0 & D_R \end{bmatrix} \tag{7}$$

$$\underset{\rightarrow k}{c} = \begin{bmatrix} c_k \\ c_{k+1} \\ \vdots \\ c_{k+n_c-1} \end{bmatrix}; \quad \underset{\rightarrow k}{r} = \begin{bmatrix} r_{k+1} \\ r_{k+2} \\ \vdots \\ r_{k+n_a} \end{bmatrix}; \quad \underset{\rightarrow k+2}{r} = \underbrace{\begin{bmatrix} 0\,I\,0\cdots\,0 \\ 0\,0\,I\cdots\,: \\ :\,:\,:\;\;:\;\;: \\ 0\,0\,0\cdots\,I \\ 0\,0\,0\cdots\,I \end{bmatrix}}_{D_R} \underset{\rightarrow k+1}{r}; \quad \underset{\rightarrow k+1}{c} = \underbrace{\begin{bmatrix} 0\,I\,0\cdots\,0 \\ 0\,0\,I\cdots\,: \\ :\,:\,:\;\;:\;\;: \\ 0\,0\,0\cdots\,I \\ 0\,0\,0\cdots\,0 \end{bmatrix}}_{D_c} \underset{\rightarrow k}{c} \tag{8}$$

Readers will note that within the definitions of D_R, D_c assumptions have been embedded into the prediction model that $r_{k+n_a+i} = r_{k+n_a}$, that is we know the set point only n_a steps into the future and also $c_{k+n_c+i} = 0, i \geq 0$ (as required by (5)).

2.4 *Admissible sets*

The predictions from autonomous model (7) are defined as feasible if they satisfy the constraints (2) for all future samples. For convenience, these constraints are represented as a set of matrix inequalities. Standard algorithms are available in the literature for determining these inequalities (e.g. Gilbert and Tan (1991) or recent variants such as Pluymers et al. (2005a)). At this point it is worth introducing an acronym Maximal Controlled Admissible Set (MCAS) which in this paper is taken to be largest volume region in x-space for which one can determine a choice for d.o.f. $\underset{\rightarrow k}{c}$ such that the predictions of (7) satisfy constraints. In mathematical terms, (for suitable M, N, P, d) the set is denoted as S_{MCAS} where:

$$S_{MCAS} = \{x : \exists \underset{\rightarrow k}{c} \;\; s.t \; Mx_k + N\underset{\rightarrow k}{c} + P\underset{\rightarrow k+1}{r} \leq d\} \tag{9}$$

Of particular relevance to this paper is the observation that the MCAS shape and position changes as $\underset{\rightarrow k+1}{r}$ changes (e.g. Rossiter (2006)).

2.5 *An Optimal MPC algorithm*

A typical infinite horizon MPC algorithm (here denoted as OMPC for optimal MPC) minimises the performance index (3) subject to constraints (9) and using an input trajectory/d.o.f. as specified in (5). Of the optimised $\underset{\rightarrow k}{c}$, only the first value c_k is deployed and the optimisation is repeated at every sample. This section gives a brief summary of the algebra required for such an algorithm.

The deviations in states and inputs relative to their estimated steady-state values can be expressed in terms of the augmented state Z_k as:

$$x_k - x_{ss|k} = \underbrace{\big[I, 0, .., 0\big] - \big[0\ 0\ [K_{xr}, 0, 0, ..0]\big]}_{K_{xss}} Z_k \tag{10}$$

$$u_k - u_{ss|k} = \underbrace{-\big[K, 0, .., 0\big] - \big[0\ 0\ [K_{ur}, 0, , ..., 0]\big]}_{K_{uss}} Z_k \tag{11}$$

Substituting (7), (10),(11) into the performance index (3) one can express J in terms of the augmented state as:

$$J = \sum_{i=0}^{\infty} Z_{k+i}^T [\Psi^T K_{xss}^T Q K_{xss} \Psi + K_{zss}^T R K_{zss}] Z_{k+i} \tag{12}$$

It is straightforward to show that this reduces to the following equivalent form:

$$J = Z_k^T S_z Z_k; \quad S_z = \sum_{i=0}^{\infty} (\Psi^i)^T W \Psi^i \tag{13}$$

Critically, a simpler method of computing S_z is via a Lyapunov equation so that $W = \Psi^T K_{xss}^T Q K_{xss} \Psi + K_{zss}^T R K_{zss}$ and $S_z = W + \Psi^T S_z \Psi$.

Algorithm 1: In summary, the OMPC algorithm is given as follows. At every sample, first perform the optimisation:

$$\min_{\underset{\rightarrow k}{c}} J = Z_k^T S_z Z_k \quad \text{s.t.} \quad M x_k + N \underset{\rightarrow k}{c} + P \underset{\rightarrow k+1}{r} \leq d \tag{14}$$

Then, use (5) and the first value c_k of the optimum $\underset{\rightarrow k}{c}$ to determine the current system input u_k.

In many cases the OMPC algorithm is effective, but it is known to contain a number of weaknesses which are tackled in this paper. Three of these are:

(1) Infeasibility, that is occasions where $x_k \notin S_{MCAS}$ can occur due to rapid changes in r_k, p_k and also due to uncertainty. This is more common where the underlying loop control law K is well tuned, which of course is a desirable choice. If $x_k \notin S_{MCAS}$, the algorithm is undefined so an alternative control law is required.
(2) Typical approaches in the literature use $n_a = 1$, that is they assume no advance knowledge of set point changes. A simplistic inclusion of future information ($\underset{\rightarrow k+1}{r}$ with $n_a > 1$) into the algorithm is often detrimental (Valencia-Palomo et al. 2014).
(3) Algorithm modifications (e.g. Kothare et al. (1996)) which cater explicitly for parameter uncertainty tend to require significant online computation and often conservative assumptions on feasible regions and thus are inconsistent with typical industrial code.

3 Effective use of advance information assuming feasibility

3.1 *Ignoring advance knowledge of the target*

Most of the literature using OMPC algorithms ignores advance knowledge of target changes, that is, it tacitly assumes that for the purposes of prediction and optimisation, $n_a = 1$ and $r_{k+i} = r_{k+1}, \forall i > 0$. This also means that, within the predictions, $x_{ss|k}, u_{ss|k}$ are constant. Moreover, it can be shown that (Rossiter 2003) in this case the optimum unconstrained choice of the d.o.f. is $\underset{\rightarrow k}{c} = 0$. The proof is not important here, but rather this gives a useful observation which is a helpful insight for the control operator and as we develop the contributions of this paper. When $n_a = 1$, that is no advance knowledge, optimisation reduces to minimising the weighted norm of the input perturbations $\underset{\rightarrow k}{c}$ so the magnitude of $\underset{\rightarrow}{c}$ is a direct indicator of the impact of constraints on the input choices. If $\underset{\rightarrow k}{c} = 0$, constraints are not affecting the choice of control inputs.

3.2 *Impact of advance knowledge in the unconstrained case*

This section will show that when $n_a \neq 1$, the OMPC algorithm may lose the useful link between the value of $\underset{\rightarrow k}{c}$ and constraints. Moreover, the section derives the explicit impact of the future target values on the control law (that is the control feedforward term). This is straightforward algebra but is useful hereafter and also gives insight.

Theorem 3.1 : *The use of performance index (13) allows the user to formulate the explicit dependence of the control law on future target information.*

Proof: The matrix S_z can be decomposed into its individual block elements which show the links between the states x_k, $\underset{\rightarrow k+1}{r}$, $\underset{\rightarrow k}{c}$ within the cost function:

$$S_z = \begin{bmatrix} S_x & S_{xc} & S_{xr} \\ S_{xc}^T & S_c & S_{cr} \\ S_{xr}^T & S_{cr}^T & S_r \end{bmatrix} \tag{15}$$

Use the decomposition of (15) to expand (13):

$$J = x_k^T S_x x_k + 2x_k^T S_{xc} \underset{\rightarrow k}{c} + \underset{\rightarrow k}{c}^T S_c \underset{\rightarrow k}{c} + \underset{\rightarrow k+1}{r}^T S_r \underset{\rightarrow k+1}{r} + 2x_k^T S_{xr} \underset{\rightarrow k+1}{r} + 2\,\underset{\rightarrow k}{c}^T S_{cr} \underset{\rightarrow k+1}{r} \tag{16}$$

However, it is well known (e.g. Rossiter (2003)) that $S_{xc} = 0$. Moreover, within the optimisation of J one can ignore the terms based on S_x, S_{xr} as these contain no d.o.f. and hence:

$$\arg\min_{\underset{\rightarrow k}{c}} J \equiv \arg\min_{\underset{\rightarrow k}{c}} \{\underset{\rightarrow k}{c}^T S_c \underset{\rightarrow k}{c} + 2\,\underset{\rightarrow k}{c}^T S_{cr} \underset{\rightarrow k+1}{r}\} \tag{17}$$

Next, minimising J w.r.t $\underset{\rightarrow k}{c}$ one finds:

$$\frac{\mathrm{d}J}{\mathrm{d}\,\underset{\rightarrow k}{c}} = 0 \quad \Rightarrow \underset{\rightarrow k}{c} = \underbrace{-S_c^{-1} S_{cr}}_{P_r} \underset{\rightarrow k}{r} = P_r \underset{\rightarrow k+1}{r}. \tag{18}$$

Hence, the feedforward term, in the unconstrained case, is given by P_r and the dependence on $\underset{\rightarrow k+1}{r}$ is explicit. □

A key point to note therefore is that with the use of $n_a > 1$, the optimum values of input perturbations c_k are no longer zero, even in the unconstrained case, and now depend explicitly on the future target values as well as constraints.

Remark 2: The summation of the block elements of the matrix P_r must be equal to zero. This follows immediately from the observation that if the future target is constant then the optimum unconstrained value is $\underset{\rightarrow k}{c} = 0$ and hence $P_r \underset{\rightarrow k+1}{r} = 0$. This is a useful check for coding errors.

3.3 *Potentially negative effects of using advance target knowledge with OMPC and systematic choices for n_a: the unconstrained case*

The results of optimising performance suggested that the optimum value of $\underset{\rightarrow k}{c}$ depends upon future target values, through the feedforward P_r, and the obvious inference is that one should get better performance by using this information and moreover using as large a n_a as possible. Surprisingly however, this intuitive expectation is incorrect. In fact, including the feedforward term can cause a deterioration in closed-loop performance as will be shown in this section.

More specifically, this section seeks to give more systematic guidance on how much feed forward information is useful and leads to improved performance and also, what constitutes too much feed forward information which cannot be used effectively and thus can be counter productive. In fact, as the reader will see, the best choice of n_a is linked to both the system dynamics and n_c.

This subsection demonstrates a trial and error method (Algorithm **2** (Dugham and Rossiter 2016)) and a short cut algorithm (algorithm **3**) to select the best amount of the advance knowledge for different systems. While this may seem somewhat simple or lacking rigour, both have the advantage of being easy to code/implement in practice which is a core aim of this paper contribution and moreover gives insights that allow easy extension to the constraint handling case.

Algorithm 2: For values n_a from 1 to n_y, simulate the process (for a specified target) and compute the runtime cost J using performance index (3) by summing terms over the entire runtime (until all terms have converged to zero). Plot the runtime cost vs n_a. Select the smallest n_a giving an acceptable runtime cost on the basis that a smaller n_a may be preferable if the loss in performance is minimal compared to a larger n_a.

Algorithm 3: Determine the closed-loop settling-time n_s using a measure such as settling to within 10% of steady-state. Choose $n_a = n_a^* = n_c + n_s/2$.

The following examples demonstrate how easily these algorithms can be applied and moreover, the fact that the optimum answer is highly dependent on both closed-loop dynamics and n_c. Consider the following systems:

$$A = \begin{bmatrix} 0.8 & 0 \\ 0.2 & 0.2 \end{bmatrix}, B = \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix}; \quad C = \begin{bmatrix} 1.9 & -1 \end{bmatrix} \tag{19}$$

$$A = \begin{bmatrix} 1 & -0.09 \\ 1 & 0 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \quad C = \begin{bmatrix} 1 & -0.1 \end{bmatrix} \tag{20}$$

In both cases, closed-loop simulations are performed with a range of values of n_c, n_a. The responses for model (19) are in figure 1 and for model (20) are in figure 2. Table 1 also summarises the performance using runtime cost (3). In summary:

(1) It is clear that advance information can be used systematically and affects behaviour.
(2) Up to a limit, choosing $n_a > 1$ improves performance compared to $n_a = 1$ but if n_a is chosen too large, such as $n_a \gg n_c$, then closed-loop performance will not improve and indeed can deteriorate (Valencia-Palomo et al. 2014). We should note however, that in the constraint free case, because OMPC is built around an optimal control law, performance will not deteriorate with large n_a, but it would do so when constraints became active.

Table 1. Variation of performance indices for step changes in target over the runtime for a range of n_a and comparison to proposals from Algorithms (**2**,**3**).

System (19) $n_c = 2, R = 0.1, n_s = 18$				Alg. **2**	Alg.**3**
n_a	10	11	12	12	11
J	2.21	2.14	2.10	2.10	2.14
System (19) $n_c = 5, R = 0.1, n_s = 18$				Alg. **2**	Alg.**3**
n_a	13	14	15	15	14
J	2.06	2.04	2.02	2.02 ‘	2.04
System (20) $n_c = 2, R = 10, n_s = 8$				Alg. **2**	Alg.**3**
n_a	5	6	7	7	6
J	1.64	1.60	1.58	1.58	1.60
System (20) $n_c = 5, R = 10, n_s = 6$				Alg. **2**	Alg.**3**
n_a	7	8	9	9	8
J	1.58	1.57	1.57	1.57	1.57

(3) It is possible to use trial and error (Algorithm **2**) to choose an optimum value of n_a for a given set point profile but this would cumbersome to implement in practice whereas the simple guideline of $n_a^* \approx n_c + n_s/2$ suggested by Algorithm **3** is seen to be fairly effective in the unconstrained case and would be easier to deploy in general.

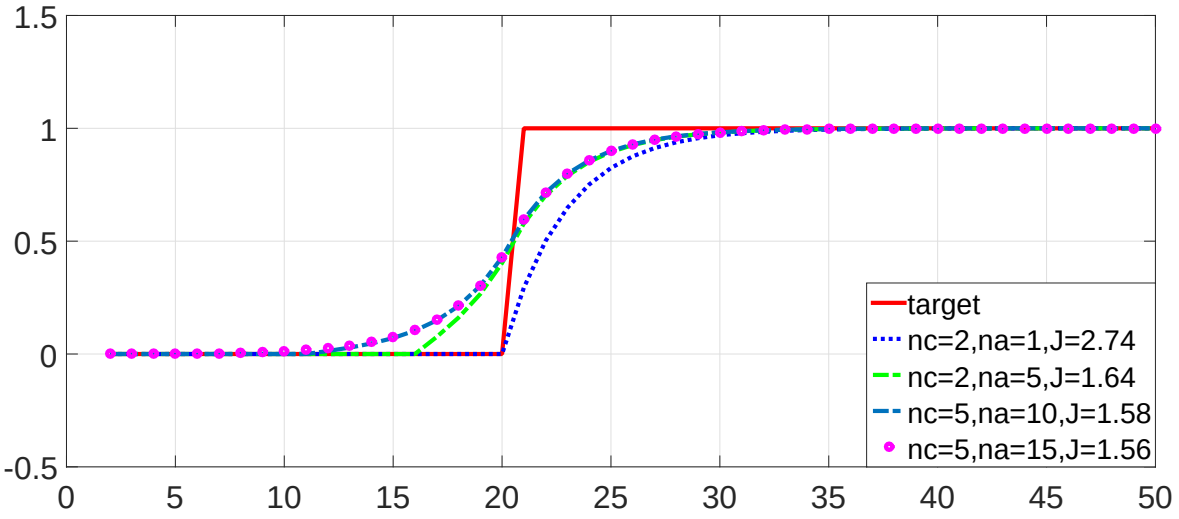

Figure 1. Closed-loop step responses for system (19) with various n_a, n_c.

3.4 *Optimising and embedding the use of feed forward information*

Before consideration is given to the constrained case, it is important to get the unconstrained case right as this will be the foundation for including constraints later. The previous section and some earlier work (Valencia-Palomo et al. 2014) gave an indication of a possible start point which is to determine the feedforward term P_r separately from the online optimisation, that is to determine a choice of P_r which is known to be optimal in the unconstrained case; such a choice would depend on assumptions about the dynamics with the feedback loop and choices for both n_a, n_c.

The proposal hereafter is to embed the optimised feedforward and then add d.o.f. around this for constraint handling, as required. The results are straightforward, but given for completeness as they build a foundation for the next sections.

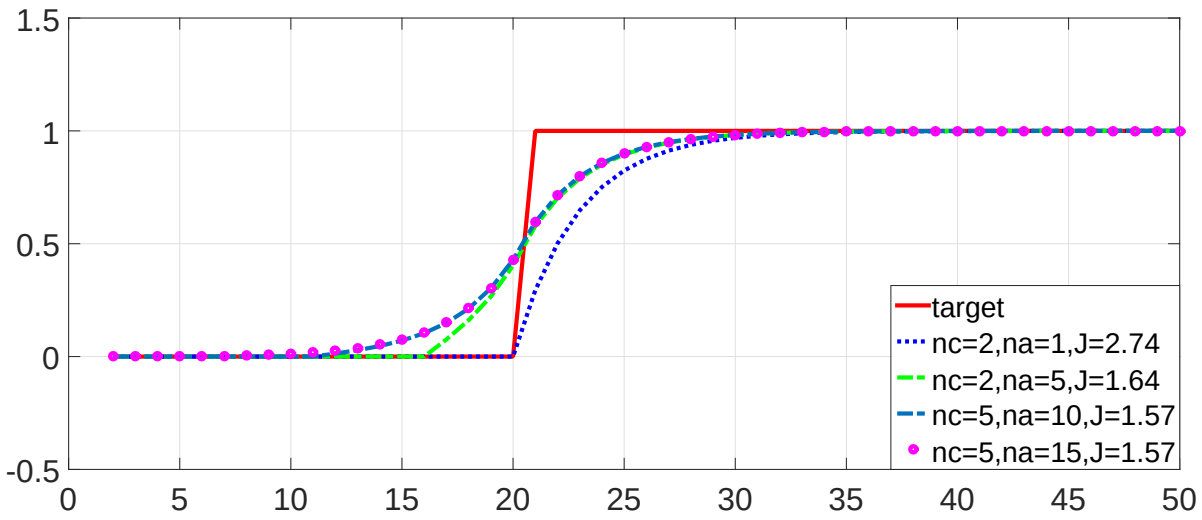

Figure 2. Closed-loop step responses for system (20) with various n_a, n_c.

Theorem 3.2: *Minimisation of performance index (17) gives the same optimum* $\underset{\rightarrow k}{c}$ *as the following optimisation.*

$$\underset{\rightarrow}{\tilde{c}} = \arg\min_{\underset{\rightarrow}{\tilde{c}}} \quad \tilde{J} = \underset{\rightarrow k}{\tilde{c}} S_c \underset{\rightarrow k}{\tilde{c}}; \quad \underset{\rightarrow k}{c} = \underset{\rightarrow k}{\tilde{c}} + P_r \underset{\rightarrow k+1}{r} \tag{21}$$

Proof: A parameterisation of the input perturbations c_k which includes the optimum feedforward (18)) and further d.o.f. for constraint handling can be defined as:

$$\underset{\rightarrow k}{c} = \underset{\rightarrow k}{\tilde{c}} + P_r \underset{\rightarrow k+1}{r} \tag{22}$$

Hence the term $\underset{\rightarrow k}{\tilde{c}}$ is the deviation from the unconstrained optimum. The cost function is given by substituting (22) into (17). Hence

$$J \equiv [\underset{\rightarrow k}{\tilde{c}} + P_r \underset{\rightarrow k+1}{r}]^T S_c [\underset{\rightarrow k}{\tilde{c}} + P_r \underset{\rightarrow k+1}{r}] + 2[\underset{\rightarrow k}{\tilde{c}} + P_r \underset{\rightarrow k+1}{r}]^T S_{cr} \underset{\rightarrow k+1}{r} \tag{23}$$

From (18) it is known that the unconstrained optimum choice is $\underset{\rightarrow k}{\tilde{c}} = 0$ and therefore the performance index must be a quadratic with no-affine term. Therefore, for some constant C,

$$J = [\underset{\rightarrow k}{\tilde{c}}]^T S_c [\underset{\rightarrow k}{\tilde{c}}] + C \tag{24}$$

which implies minimising J and minimising $\tilde{J}$ must give the same $\underset{\rightarrow k}{\tilde{c}}$. □

Corollary 3.3: *An equivalent MCAS for control perturbations (22) is straightforward to construct. This follows directly from substitution of (22) into (9).*

$$Mx_k + N \underset{\rightarrow k}{c} + P \underset{\rightarrow k+1}{r} \leq d \quad \Rightarrow \quad Mx_k + N[\underset{\rightarrow k}{\tilde{c}} + P_r \underset{\rightarrow k+1}{r}] + P \underset{\rightarrow k+1}{r} \leq d \tag{25}$$

Now the constrained optimisation can focus on the computation of just $\underset{\rightarrow k}{\tilde{c}}$ as the bias term to deal with advance information systematically is fully embedded! Specifically, $\tilde{c}_k \neq 0$ will be required iff constraints are active and the magnitude of $\tilde{c}_k$ is an indicator of how far one is from the unconstrained optimal associated to the advance knowledge scenario!

Algorithm 4: The constrained OMPC algorithm with systematic incorporation of advance knowledge can now be summarised as:

$$\min_{\underset{\rightarrow}{\tilde{c}}} \quad \tilde{J} = \underset{\rightarrow k}{\tilde{c}}^T S_c \underset{\rightarrow k}{\tilde{c}} \quad s.t. \quad Mx_k + N\underset{\rightarrow k}{\tilde{c}} + [NP_r + Q]\underset{\rightarrow k+1}{r} \leq d \tag{26}$$

The optimised $\tilde{c}_k$ is used in conjunction with (22,5) to determine u_k.

3.5 *Summary*

The key point here is that, the default OMPC algorithm has the nice property that a choice of $\underset{\rightarrow}{c} = 0$ implies that the unconstrained optimal is feasible and thus one has a clear view on the impact of constraints as there is a direct link to the magnitude of $\underset{\rightarrow}{c}$. However, including advance knowledge destroyed this link (see eqn.(22)). By re-parameterising the degrees of freedom in terms of $\underset{\rightarrow}{\tilde{c}}$ this nice property is recovered and moreover, the nominal optimal solution, incorporating advance knowledge, is embedded within the predictions. Hence the required online optimisation, a standard quadratic program (QP), is solely dealing with constraint handling and not trying to achieve mixed objectives of performance optimisation and handling advance information alongside constraint handling.

4 Unreachable targets and advance knowledge

The previous section focussed on effective use of advance knowledge when target changes are feasible so that optimisation (26) always has a solution. This section extends the discussion to scenarios where infeasibility occurs, that is, where the change in the steady-state $x_{ss|k}, u_{ss|k}$ is too rapid, so the prediction class (5) is not sufficiently large to meet constraints. Infeasibility can take two common forms:

(1) Transient infeasibility. That is, the target is reachable asymptotically but a much larger n_c is required (Rao and Rawlings 1999) to find a feasible solution. Assuming n_c cannot be increased, an alternative algorithm is needed to maintain feasibility and convergence. A common proposal (Rossiter et al. 1996, Rossiter 2006, Limón et al. 2008, Shead et al. 2010)) is to include extra d.o.f that capture changes in the steady state. A contractive constraint may be deployed to ensure convergence.

(2) Persistent infeasibility or so called unreachable targets (Shead et al. 2010, Rawlings et al. 2008). In this case, the target cannot be reached, even asymptotically, without violating some constraints and thus an alternative parameterisation allowing changes to the target steady-state is needed, again along with a modified objective.

A key point to note here is that the majority of the work in the literature tackling these two issues assume that $n_a = 1$; in this paper proposals are given which embed advance knowledge (i.e. $n_a > 1$) while also taking account of transient or permanent infeasibility and moreover, while retaining a simple QP based optimisation.

Remark 3: Reference governor approaches (Gilbert et al. 1994, Aghaei et al. 2013) have analogies to both the above in that a governor deploys a transient change in the target to maintain feasibility. However these will be discussed no further as their focus is on simplicity so they often lead to a loss in performance.

4.1 *Input parameterisation for unreachable targets*

In the case where the asymptotic target is unreachable, then the input parameterisation of (5) is invalid, that is, infeasible. The proposal here is to replace this parameterisation with:

$$\begin{array}{ll} u_{k+i} - u_{ss|k+i} = -K(x_{k+i} - x_{ss|k+i}) + c_{k+i}, & i \leq n_c \\ u_{k+i} - u_{ss|k+i} = -K(x_{k+i} - x_{ss|k+i}) + c_\infty & i > n_c \end{array} \tag{27}$$

Lemma 4.1: *The inclusion of the term c_∞ within the input parameterisation of (27) leads to a constant offset between the predicted steady-state output and the desired target.*

Proof: Substitute the asymptotic input parameterisation of (27) into the model dynamics (1,4). It is clear that if $c_\infty = 0$ there is no offset and hence

$$\lim_{k\to\infty} x_k = x_{ss} = K_{xr} r_{k+n_a} \quad \Rightarrow \quad \lim_{k\to\infty} y_k = r_{k+n_a} \tag{28}$$

Using superposition one can then determine that with (27)

$$\lim_{k\to\infty} y_k = r_{k+n_a} + \delta y_\infty; \quad \delta y_\infty = \underbrace{[C(I-\Phi)^{-1}B]^{-1}}_{G_\infty} c_\infty \qquad \square \tag{29}$$

Corollary 4.2: *The inclusion of c_∞ is equivalent to deploying an artificial target $\hat{r}$ which is deviated from the true target by δy_∞. Hence, one can also find an equivalent c_∞ for a specified artificial target $\hat{r}$ as follows:*

$$c_\infty = G_\infty^{-1}(\hat{r}_k - r_{k+n_a})) \tag{30}$$

Lemma 4.3: *Minimising performance index (3) with with input parameterisation (27) and $n_a = 1$ gives the same optimising values for c_k as minimising the following cost:*

$$J_c = \underset{\rightarrow k}{c}^T S \underset{\rightarrow k}{c} + \sum_{i=1}^{\infty} c_\infty^T S c_\infty \tag{31}$$

Proof: Minimising the true performance index J has been shown earlier (Theorem 3.1) to be equivalent to minimising:

$$J_c = \sum_{i=0}^{\infty} c_{k+i}^T S c_{k+i} \tag{32}$$

Then, noting that in effect parameterisation (27) implies $c_{k+n_c+i} = c_\infty, i > 0$ the result drops out. The $\underset{\rightarrow k+1}{r}$ has been excluded from J_c because here $n_a = 1$. $\square$

Corollary 4.4: *Combining input parameterisation of (27) with the observation of Theorem 3.2 and Lemma 4.3 one can form an equivalent cost function for $n_a > 1$ of the form:*

$$\tilde{J}_c \equiv \underset{\rightarrow k}{\tilde{c}}^T S \underset{\rightarrow k}{\tilde{c}} + \sum_{i=1}^{\infty} c_\infty^T S c_\infty \tag{33}$$

Equivalent means the optimum control law from minimising $\tilde{J}_c$ is the same as the optimal control law from minimising J_c.

In summary, one key point of this section is to show how an additional d.o.f. can be added which allows the MPC algorithm to cater for unreachable points. Moreover, this d.o.f. has been

added in such a way that gives clarity to the impact of constraints in that, the optimised values of $\tilde{c}_k$ are zwero, if and only iff the unconstrained optimal is feasible. Nevertheless, a more significant contribution is to show how optimal trajectories, which include advance knowledge of targets, can also be embedded effectively. The $\underset{\rightarrow}{\tilde{c}}$ terms indicate the deviation from the unconstrained optimal, with advance knowledge, during transients and the c_∞ term gives the deviation from the unreachable target.

4.2 *Objective function with steady-state offset*

It is well recognised (Rossiter et al. 1996, Rawlings et al. 2008) that the performance index of (33) is not useful in itself because whenever $c_\infty \neq 0$ this J_c is unbounded and hence minimising J_c is equivalent to minimising the offset component of $c_\infty^T S c_\infty$. Indeed one could choose simply to do that, but such an objective would effectively ignore the impact of transient behaviour on overall performance and thus may lead to relatively poor decisions. Consequently, there is a need for a performance index which captures the following requirements:

(1) Has an objective measure of transient performance.
(2) Is always feasible and thus includes the d.o.f. c_∞ to allow deviation from unreachable asymptotic targets.
(3) Includes advance information about target changes.

The key proposal here is to build on the performance index of (33) which already includes transient performance and implicitly has included information about advance knowledge through the deployment of $\tilde{c}_k$. However, we desire a reduced emphasis on the asymptotic predicted error so that this does not swamp the transient terms.

A performance index J_p which gives a balance between transient behaviour and expected asymptotic offset is:

$$J_p = W_1(c_\infty^T S c_\infty) + \underset{\rightarrow k}{\tilde{c}}^T S \underset{\rightarrow k}{\tilde{c}} \tag{34}$$

where W_1 is a weighting matrix to be selected. Here, the term $(c_\infty^T S c_\infty$ penalises asymptotic offset and the term $\underset{\rightarrow k}{\tilde{c}}^T S \underset{\rightarrow k}{\tilde{c}}$ penalises transient performance, including information on $\underset{\rightarrow k+1}{r}$. The weighting W_1 allows the user to determine the emphasis they wish to place on each term.

4.3 *Constraint handling*

The dynamics now include an additional term as compared to (7), that is the term $c_{k+n_c+i} = c_\infty, i \geq 0$ and hence the autonomous model and inequalities capturing constraint information need minor modifications to include this. Define the autonomous prediction model as follows:

$$Z_{k+1} = \Psi Z_k; \quad \Psi = \begin{bmatrix} \Phi & [B,0,..,0] & 0 & [(\Phi - I)K_{xr}, 0, 0..., 0] \\ 0 & D_c & E_c & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & D_R \end{bmatrix}; \quad Z_k = [x_k^T, \underset{\rightarrow k}{\tilde{c}}^T, c_\infty, \underset{\rightarrow k}{r}^T]^T \tag{35}$$

D_c, E_C are shift matrices for $\tilde{c_k}, c_\infty$ analogous to that in (8).

Using model (35) it is straightforward to apply the admissible set algorithm (Pluymers et al. 2005a) to find a the new MCAS (denoted S_{MCASU} for MCAS Unreachable) of the following format:

$$S_{MCASU} = \{x : \exists(\underset{\rightarrow k}{\tilde{c}}, c_\infty) \;\; s.t \; Mx_k + N\underset{\rightarrow k}{\tilde{c}} + Vc_\infty + P\underset{\rightarrow k+1}{r} \leq d\} \tag{36}$$

However, readers should note that a standard admissible set algorithm may not terminate in finite or reasonable time due to the implied steady-state being on a constraint boundary by virtue of $c_\infty \neq 0$ and thus some termination condition needs to be added.

The proposed OMPC algorithm can now be summarised.

Algorithm 5: An OMPC algorithm with both advance knowledge handling and the potential to manage unreachable targets is summarised in the following optimisation.

$$\min_{\underset{\rightarrow}{\tilde{c}},c_\infty} \quad W_1(c_\infty^T S c_\infty) + \underset{\rightarrow k}{\tilde{c}}^T S \underset{\rightarrow k}{\tilde{c}} \quad s.t. \quad (\underset{\rightarrow k}{\tilde{c}}, c_\infty) \in S_{MCASU} \tag{37}$$

Use the optimised $\underset{\rightarrow}{\tilde{c}}$ in conjunction with (22) to determine c_k and implement the first move u_k of the control law as defined in (27).

4.4 *Guarantees of feasibility and performance*

This section establishes that algorithm **5** has guarantees of recursive feasibility and asymptotic convergence to a point which minimises the weighted offset.

Lemma 4.5: *The proposed OMPC algorithm* **5** *with advance knowledge handling maintains feasibility irrespective of changes in the target.*

Proof: The proof follows in a straightforward fashion from the assumption of feasibility at start up and the inclusion of c_∞. If there were no change in target, that is $r_{k+n_a+1} = r_{k+n_a}$, then one can use standard MPC arguments to show that the optimum (assumed feasible) solution from sample k can be carried forward to sample $k+1$ and thus feasibility is retained. In the case where $r_{k+n_a+1} \neq r_{k+n_a}$, one can always introduce a non-zero value of c_∞ such that the implied artificial target $\hat{r}_{k+n_a+1} = r_{k+n_a}$, thus again retaining feasibility. □

Theorem 4.6: *The proposed algorithm* **5** *is convergent to the point which minimises the weighted offset.*

Proof: At steady-state the optimised values for c_k are all identical and therefore the optimisation is capped by:

$$\min_{\underset{\rightarrow k}{c},c_\infty} J_p \quad \leq c_\infty^T [(W_1 + nI)S] c_\infty \tag{38}$$

Any optimised value of c_k such that $c_k^T S c_k < c_\infty^T S c_\infty$ would be a contradiction of the system being in steady-state and thus, noting the relationship of (30), the optimisation has minimised a weighted norm of the offset. □

It is not the purpose of this paper to consider guarantees in the presence of disturbances as that case is altogether much more demanding. The focus here is on a simple approach to deal with the basic requirements. However, it is worth noting that the flexibility afforded in c_∞ is often sufficient to deal with any transient infeasibility caused by changes in disturbances.

4.5 *Numerical example with an unreachable target*

This section gives numerical examples which demonstrate the efficacy of algorithm **5** for handling both advance knowledge and unreachable targets in a single simple optimisation.

A common scenario is the infeasibility of terminal constraints due to a too fast or too large change of asymptotic target. These examples demonstrate how the proposed algorithm smoothly introduces an artificial target during transients but moves to the correct steady-state asymptotically. Moreover, it is clear that the algorithm continues to embed information about target moves

Table 2. Performance indices for step changes in target for example system (39) using Algorithm **5**.

	J with $n_a = 1$	J with $n_a = 3$	J with $n_a = 5$
Example (39)	42.05	38.62	31.92

in a systematic fashion, thus improving performance compared to more conventional approach with $n_a = 1$.

Consider the system and constraints

$$A = \begin{bmatrix} 0.8 & 0.1 \\ -0.2 & 0.9 \end{bmatrix}, B = \begin{bmatrix} 0.1 \\ 0.8 \end{bmatrix}; \quad C = \begin{bmatrix} 1.9 & -1 \end{bmatrix}; \quad -1 \leq u \leq 1.35; \quad \begin{bmatrix} -0.8 \\ -2.5 \end{bmatrix} \leq x \leq \begin{bmatrix} 4 \\ 4 \end{bmatrix} \tag{39}$$

with $r = 1, n_c = 2, n_a = 1, R = 0.1I, Q = C^T C$.

- Figure 3 shows responses when the target is reachable in steady-state, but not in transients, with the use of $n_a = 1$.
- Figure 4 shows responses when the target is reachable in steady-state, but not in transients, with the use of $n_a = 5$.
- Figure 5 shows responses when the target is unreachable (now $r = 1.1$) in steady-state with the use of $n_a = 5$.

It is clearly shown in figures 3, 4 and table 2 that algorithm **5** provides effective control for a constrained system both with no advance knowledge ($n_a = 1$) and with advance knowledge. Readers will note that for figures 3,4 the term c_∞ (lower figure) is non-zero during transients only, as expected whereas it remains non-zero permanently in figure 5. The term $\tilde{c}_k$ is also non-zero demonstrating the benefits of incorporating advance information systematically, even within these challenging scenarios.

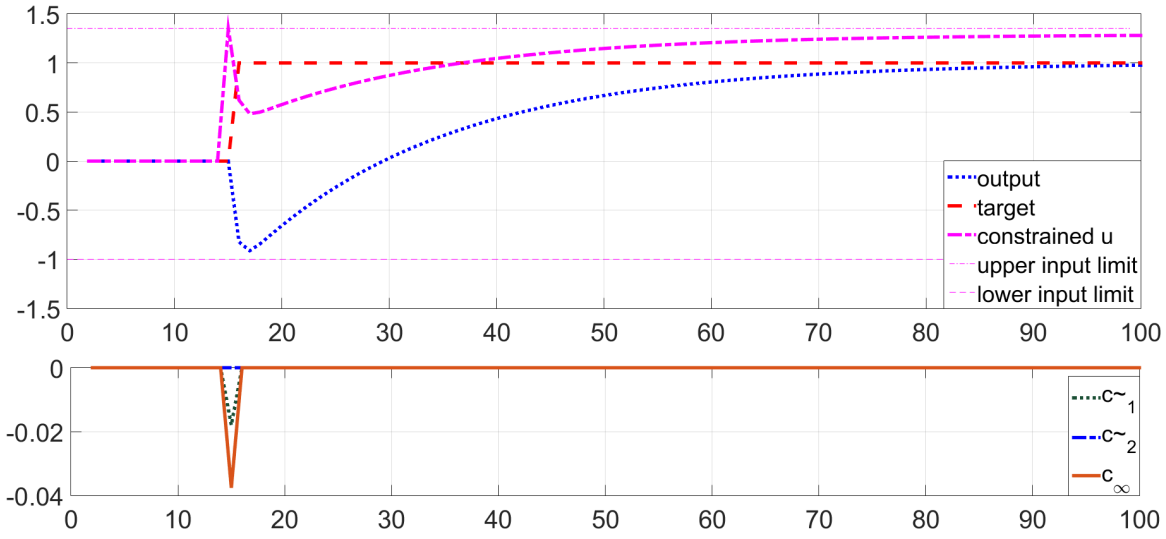

Figure 3. Closed-loop step responses for system (39) using Algorithm **5** for unreachable target in transient with $n_a = 1$.

5 Robust MPC with tracking

This section demonstrates how the results of the earlier sections can be extended to deal with parameter uncertainty, while retaining guarantees of recursive feasibility and convergence. This is significant as the algorithm proposed here utilises an online optimisation which is a straightforward QP with a potentially small number of decision variables. Typical approaches to robust

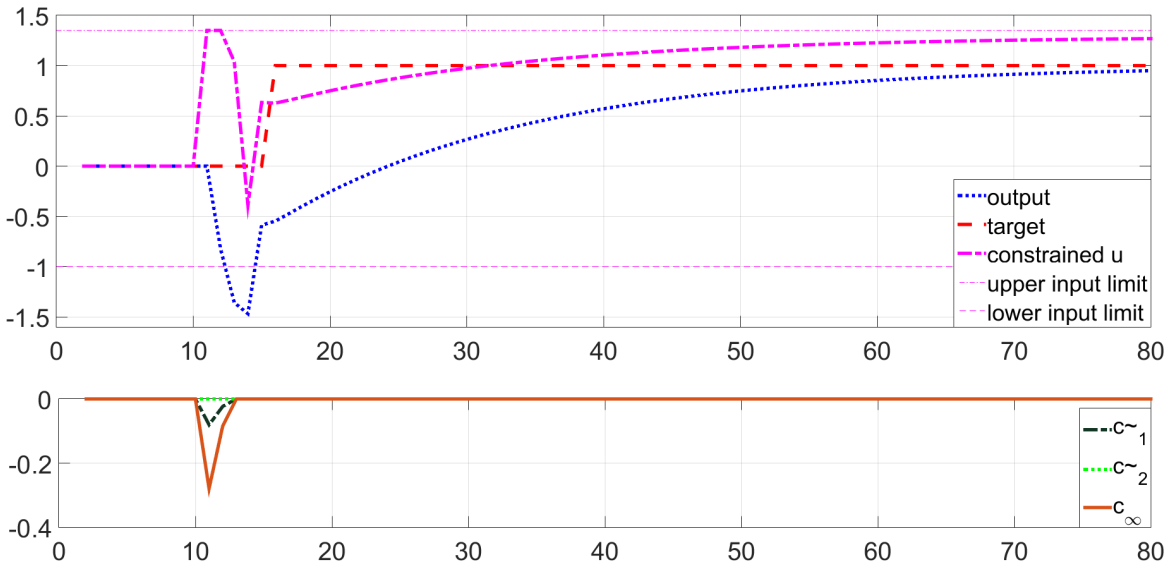

Figure 4. Closed-loop step responses for system (39) using Algorithm **5** for unreachable target in transient with $n_a = 5$.

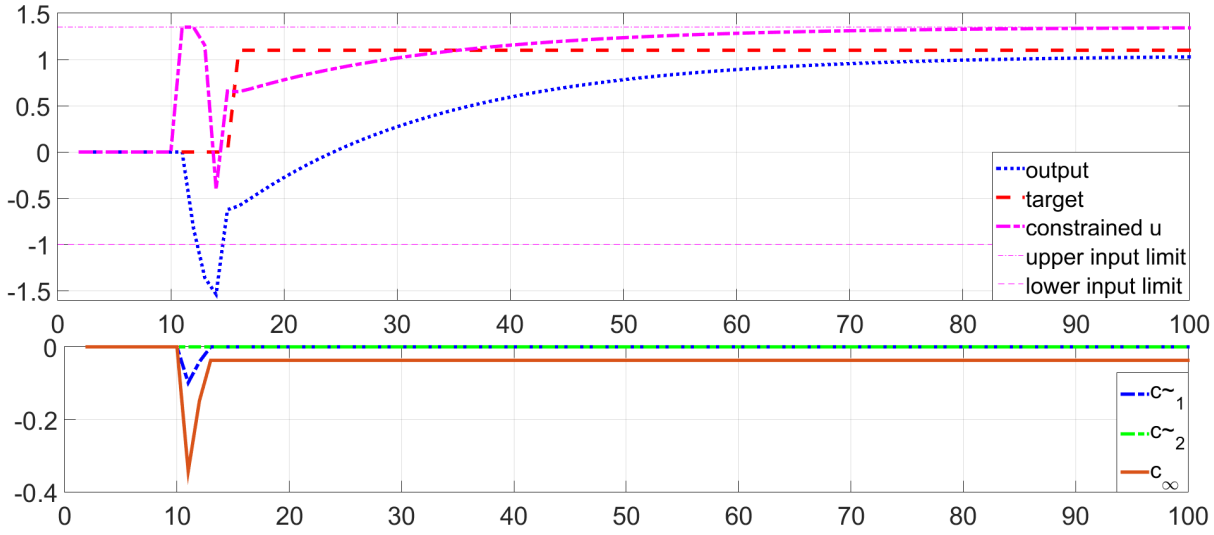

Figure 5. Closed-loop step responses for system (39)using Algorithm **5** for unreachable target in steady state for $n_a = 5$.

MPC often require much more demanding algebra and optimisations (Cheng and Jia 2004, Rakovic and Mayne 2005, Chisci et al. 2001).

The proposal here is to build on the robust invariant set approach developed in Pluymers et al. (2005a) and later demonstrated to be effective with a simple dual mode MPC algorithm for regulation Pluymers et al. (2005b). This section will demonstrate how the basic approach can be extended to cater for both tracking scenarios and unreachable set points.

5.1 *LPV system model and input predictions*

Consider the discrete time LPV system

$$x_{k+1} = A(k)x_k + B(k)u_k. \tag{40}$$

$A(k), B(k)$ are matrices defining the model. Parameter uncertainty is quantified with $[A(k)\ B(k)] \in \Omega = Co\{[A_1\ B_1], ..., [A_m\ B_m]\}$, where Co refers to convex hull of the extreme models, in which $[A\ B] \in \Omega$, hence with $0 \leq \lambda_i \leq 1,\ \sum \lambda_i = 1$ and $[A\ B] = \sum_{i=1}^{L} \lambda_i [A_i\ B_i]$.

The dual-mode predictions for system (40) with control law (27) can be described as:

$$\begin{array}{c} x_{k+1} = \Phi(k)x_k + [I - \Phi(k)]K_{xr}(r_{k+1}) + B(k)c_k \\ u_k = -Kx_k + [KK_{xr} + K_{ur}](r_{k+1}) + c_k \end{array} \tag{41}$$

where $\Phi(k) = A(k) - B(k)K$. It is noted that $[A(k)\ B(k)] \in \Omega \quad \Rightarrow \quad \Phi(k) \in Co\{\Phi_1, ..., \Phi_m\}$.

5.2 *The state feedback controller K*

It is common in the robust literature (Kothare et al. 1996) for the state feedback controller K to be determined on line so that K is varying every sample. The aim here is to combine this with the ideas summarised in the previous section and maintain algorithm simplicity and thus here assume that K is fixed (for example as in Kouvaritakis et al. (2000)). Nevertheless (Pluymers et al. 2005a) there must exist an invariant set for the uncertain unconstrained closed-loop dynamic $x_{k+1} = \Phi(k)x_k$. This can be checked using the following condition.

$$\exists P = P^T > 0 \quad s.t. \ \ \Phi_i^T P \Phi_i \leq P, \qquad i = 1, ..., m. \tag{42}$$

Algorithms for identifying a K to satisfy (42) and simultaneously optimise a nominal cost function are readily available but one could equally argue that the corresponding K for the nominal case may also be preferred, if it satisfies (42).

5.3 *The closed loop dynamics*

It is important to capture the variability in the closed-loop trajectories due to the uncertainty in the model parameters. One can capture this uncertainty efficiently with a set of linear inequalities (Pluymers et al. 2005a), as long as the dynamics and constraints can be captured in appropriate form. The basic algorithm requires a one step ahead state evolution equation (analogous to (35)) and a statement of constraint dependence on the state at each sampling instant.

Lemma 5.1: *The uncertain system predictions of (41) can be captured in a single mode autonomous model of the following form.*

$$Z_{k+1} = \Psi(k)Z_k; \ \ Z_k = [x_k^T, \underset{\rightarrow k}{\tilde{c}}^T, c_\infty, \underset{\rightarrow k}{r}^T]^T; \quad \Psi_i = \begin{bmatrix} \Phi_i & [B_i, 0..., 0] & \Gamma_i.G_\infty & \Gamma_i \\ 0 & D_C & E_C & 0 \\ 0 & 0 & D_R & 0 \\ 0 & 0 & 0 & D_R \end{bmatrix} \tag{43}$$

where $\Gamma_i = [B_i, 0, ..., 0]Pr + [(\Phi_i - I)K_{xri}, 0, .., 0]$ *and* $\Psi(k) \in Co\{\Psi_1, ...\Psi_m\}$. *These predictions are stable and must converge to the specified artificial target of* $\hat{r}$.

Proof: The definition of Ψ_i is analogous to (35). Quadratic invariance of the closed-loop dynamic $\Phi(k)$ is sufficient to ensure the quadratic invariance of the augmented dynamic $\Psi(k)$ as the additional dynamics in $\Psi(k)$ as compared to $\Phi(k)$ relate to the variables $\underset{\rightarrow k}{\tilde{c}}, c_\infty, \underset{\rightarrow k}{r}$. These dynamics are governed solely by shift matrices and thus must converge to fixed, possibly non-zero, values. The asymptotic control law is defined as $u_k - u_{ss} = -K(x_k - x_{ss}) + c_\infty$ and by definition c_∞ is the value that ensures the associated steady-state output is $\hat{r}$ (if $c_\infty = 0$ then the system converges to r_{k+n_a}). □

5.4 *The derivation of a robust MCAS*

It is shown in Pluymers et al. (2005a) that one can define a robustly invariant set to guarantee robust stability for the regulation case. This paper extends this set for tracking scenario by deploying a similar concept but with the autonomous model of (43) which includes as states both the degrees of freedom $\underset{\rightarrow k}{\tilde{c}}$ and also the future target values $\underset{\rightarrow k}{r}, c_\infty$. A major difference is convergence to a non-zero steady-state.

Lemma 5.2: *: Constraints at each sample instant can be summarised with the following inequalities:*

$$GZ_k \leq f \tag{44}$$

for appropriate G, f.

Proof: This follows from a straightforward substitution from all constraint equations such as (2) and expression in terms of state variable Z_k. The definitions of G, f, one row for each constraint, follow automatically.

$$G = \begin{bmatrix} -K & [I,0..,0] & \alpha_i.G_\infty & \alpha_i \\ K & -[I,0..,0] & -\alpha_i.G_\infty & -\alpha_i \\ 0 & 0 & [0,0..,0,K_{uri}].G_\infty & [0,..0,K_{uri}] \\ 0 & 0 & [0,..,0,-K_{uri}].G_\infty & [0,..,0,-K_{uri}] \\ C & 0 & 0 & 0 \\ -C & 0 & 0 & 0 \\ 0 & 0 & [0,..,0,CK_{xri}].G_\infty & [0,..,0,CK_{xri}] \\ 0 & 0 & -[0,..,0,CK_{xri}].G_\infty & -[0,..0,CK_{xri}] \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & -I \end{bmatrix}; \quad i = 1,2,\cdots \tag{45}$$

where $f = \left[\bar{u}^T, \underline{u}^T, \bar{u}^T, \underline{u}^T, \bar{x}^T, \underline{x}^T, \bar{x}^T, \underline{x}^T, \bar{r}^T, \underline{r}^T\right]^T$ and $\alpha_i = [1,0,...,0]Pr + [KK_{xri} + K_{uri}, 0, .., 0]$ □.

Theorem 5.3: *One can deploy the algorithm of Pluymers et al. (2005a) with sample constraints (44) and autonomous model (43) and the algorithm will converge, as long as condition (42) is satisfied.*

Proof: It is known from condition (42) combined with the convergence to fixed values within n_c, n_a steps of states c_k, r_k, that the predictions of (43) must converge to a fixed steady-state. The algorithm of Pluymers et al. (2005a) shows therefore that asymptotically, adding predictions for higher horizons results in redundant constraints beyond a certain horizon and therefore the algorithm will terminate. □

Let the robust MCAS (denoted S_{RMCAS}) be given as:

$$S_{RMCAS} = \left\{ x : \exists \underset{\rightarrow k}{\tilde{c}}, c_\infty \;\; s.t \;\; Mx + N\underset{\rightarrow k}{\tilde{c}} + Vc_\infty + P\underset{\rightarrow k}{r} \leq d \right\} \tag{46}$$

Readers should note that where the steady-state is on a boundary strict convergence may require a tolerance to be deployed. This will occur with unreachable set points which, by definition, imply the asymptotic steady-state is on a boundary.

5.5 *Robust tracking MPC algorithm*

This section summarises the proposed robust tracking MPC algorithm with advance knowledge.

Algorithm 6: Define the performance index as in (34). Define the robust MCAS as in (46). Perform the quadratic programming optimisation:

$$\min_{c_\infty, \underset{\rightarrow k}{\tilde{c}}} \quad J \quad s.t. \;\; Mx + N\underset{\rightarrow k}{\tilde{c}} + Vc_\infty + P\underset{\rightarrow k}{r} \leq d \tag{47}$$

Implement the first block element of $\underset{\rightarrow k}{\tilde{c}}$ in (27) to compute the control law.

Algorithm **6** gives guaranteed convergence and recursive feasibility, including cases of unreachable set points because, by definition, the satisfaction of RMCAS of (46) ensures recursive feasibility. In consequence, one can use conventional approaches (Rawlings et al. 2008) to show that c_k converges to a weighted minimum. Convergence of $\underset{\rightarrow k}{\tilde{c}}$ implies convergence of the state x_k due to condition (42) and dynamics (43).

5.6 *Numerical examples*

This section demonstrates that the proposed algorithm **6** is both robust to parameter uncertainty and handles advance information about the target effectively. Conversely, an algorithm which does not embed the parameter uncertainty gives less effective performance and indeed could lose feasibility. For ease of comparison, the paper uses the uncertain system model that was presented in Lim et al. (2014).

$$A = Co\left\{\begin{bmatrix}0.8 & 0.0\\ 21.8 & 1.4\end{bmatrix}, \begin{bmatrix}0.8 & 0.0\\ 16.9 & 1.3\end{bmatrix}\right\}; B = Co\left\{\begin{bmatrix}0.0\\ 0.2\end{bmatrix}, \begin{bmatrix}0.0\\ 0.2\end{bmatrix}\right\} \tag{48}$$

The system input/state constraints are:

$$\begin{array}{c} -2.5 \leq u \leq 3.5; \\ -0.5 \leq u_{ss} \leq 0.5; \end{array} \quad \begin{bmatrix}-0.5\\ -5\end{bmatrix} \leq x \leq \begin{bmatrix}0.5\\ 5\end{bmatrix} \tag{49}$$

A nominal model $A = 0.5A_1 + 0.5A_2$ and $B = 0.5B_1 + 0.5B_2$ is used to define the feedback controller $K = \begin{bmatrix}93.58 & 5.76\end{bmatrix}$ as the LQ-Optimal for $Q = diag(1, 0.01),\ R = 0.01$.

- Figure 6 demonstrates algorithm **6** for system (48) with no advance knowledge ($n_a = 1$). It is seen that although the target ($r = 1$) is unreachable during transients ($c_\infty \neq 0$), nevertheless the algorithm performs well and converges to the correct steady-state without any constraint violations.
- Figure 7 presents algorithm **6** with advance knowledge ($n_a = 3$). The target is unreachable during transients but again the algorithm performs well handling both the advance target information effectively and avoiding constraint violations. The response is faster compared to the response figure 6 which did not use advance knowledge. Moreover, the response requires less control effort. The performance benefits, of using advance knowledge with the proposed robust MPC algorithm are further evidenced in table 3, which compares the runtime costs.
- Figure 8 uses advance knowledge ($n_a = 3$) and a target $r = 1.3$ which is unreachable at steady state. Algorithm **6** performs well handling both the advance target information effectively and avoiding constraint violations while the output reaches the nearest value to the desired target which is the minimum value ($G_\infty c_\infty$).

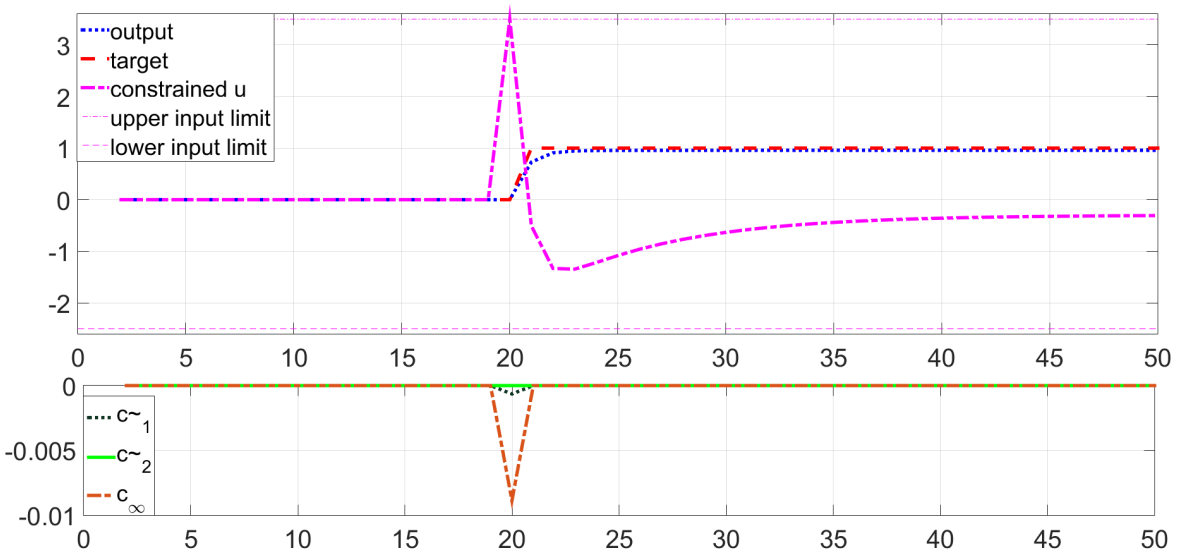

Figure 6. Closed-loop for step response of system (48) with algorithm **6** and with no advance knowledge.

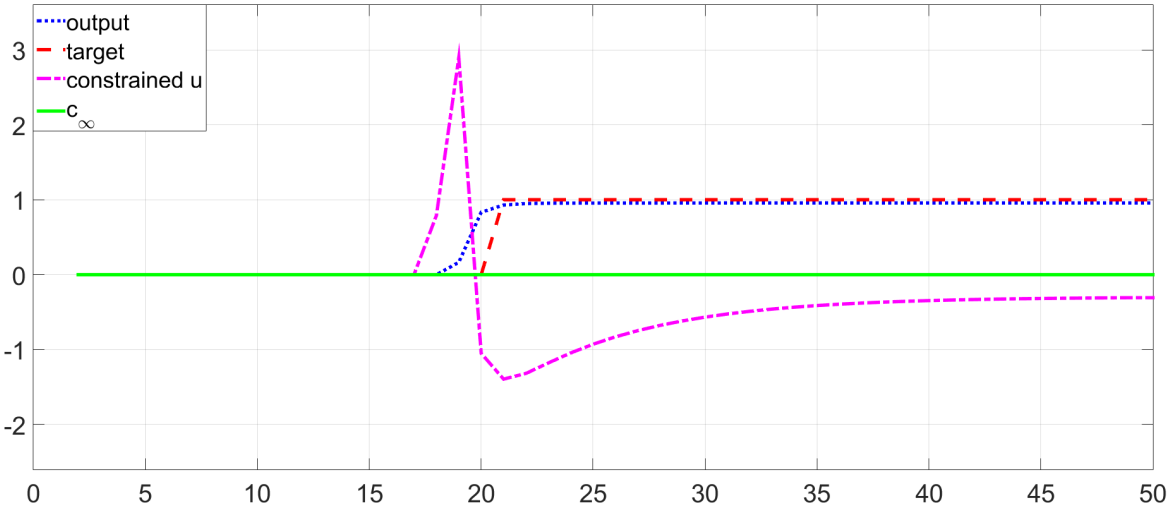

Figure 7. Closed-loop for step response of system (48) with algorithm **6** and with advance knowledge.

Table 3. The Run Time costs for System $n_a = 1$, $n_a = 3$.

System (48) $n_c = 2, R = 0.01$		
n_a	1	3
J	1.262	0.256

6 Conclusions

This paper has made three main contributions. Following a brief review of the literature on approaches to tracking within MPC, it is clear that very few papers have utilised advance information on target changes and the common assumption is that no advance information is available.

First it is shown, perhaps surprisingly, that the default inclusion of advance knowledge is not often helpful and thus this paper proposes a pragmatic algorithm for determining how much advance knowledge is likely to be useful and moreover, a corresponding 'fixed' feed forward

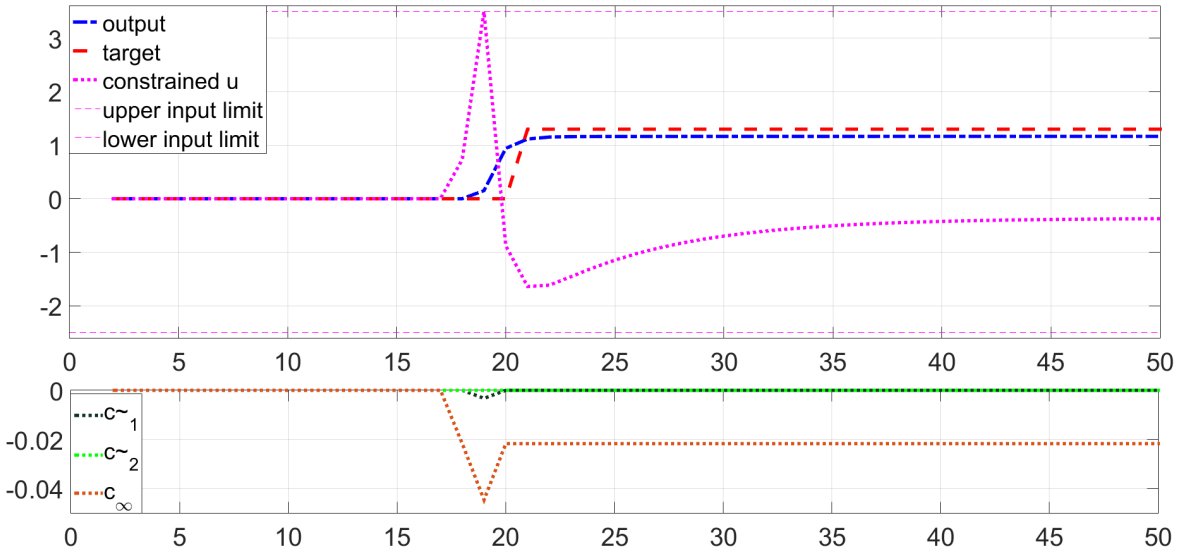

Figure 8. Closed-loop for step response of system (48) with algorithm **6** and with advance knowledge for an unreachable target at steady state.

design.

An argument is made that during constraint handling, it is better to construct predictions which embed the default *optimal unconstrained feedforward* rather than entering the future target values directly. This ensures the optimal behaviour is embedded and gives transparency to the role of the degrees of freedom. The efficacy and simplicity of this approach is demonstrated.

At times, the desired target will be unreachable and in such scenarios a default MPC algorithm becomes ill-defined. This paper proposes a simple alternative which caters for both transient and permanent infeasibility in the target without the need to change the algorithm online. Moreover, it has shown how, even in this case, the systematic embedding of advance information is straightforward and beneficial.

Finally, the paper has shown how all the previous contributions can be extended in a straightforward manner to cater for parameter uncertainty and thus give robust guarantees of feasibility and convergence while utilising a simple QP optimisation online.

References

Aghaei, S., Sheikholeslam, F., Farina, M., and Scattolini, R. (2013). An mpc-based reference governor approach for offset-free control of constrained linear systems. *International Journal of Control*, 86, 1534–1539.

Cheng, X. and Jia, D. (2004). Robust stability constrained model predictive control. In *ACC*, volume 2, 1580–1585.

Chisci, L., Rossiter, J., and Zappa, G. (2001). Systems with persistent disturbances: Predictive control with restricted. *Automatica*, 37(7), 1019–1028.

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

Dugham, S. and Rossiter, J. (2016). Systematic and simple guidance for feed forward design in model predictive control. *International Conference on Control Science and Systems Engineering*.

Fallasohi, H., Ligeret, C., and 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.

Ferramosca, A., Limón, D., Alvarado, I., Alamo, T., and Camacho, E.F. (2009). Mpc for tracking

with optimal closed-loop performance. *Automatica*, 45(8), 1975–1978.

Ferramosca, A., Limón, D., Alvarado, I., Alamo, T., Castaño, F., and Camacho, E.F. (2011). Optimal mpc for tracking of constrained linear systems. *International Journal of Systems Science*, 42(8), 1265–1276.

Gilbert, E., Kolmanovsky, I., and Tan, K.T. (1994). Nonlinear control of discrete-time linear systems with state and control constraints: A reference governor with global convergence properties. *Conference on Decision and Control*, 144–149.

Gilbert, E. and 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.

Goodwin, G., Carrasco, D., Mayne, D., Salgado, M., and Seron, M. (2011). Preview and feedforward in model predictive control: Conceptual and design issues. In *18th IFAC World Congress*, 5555–5560.

Kothare, M.V., Balakrishnan, V., and Morari, M. (1996). Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32(10), 1361–1379.

Kouvaritakis, B., Rossiter, J.A. and Schuurmans, J. (2000), Efficient robust predictive control *T*rans. IEEE AC, 45(8), pp1545–1549

Limón, D., Alvarado, I., Alamo, T., and Camacho, E.F. (2008). Mpc for tracking piecewise constant references for constrained linear systems. *Automatica*, 44(9), 2382–2387.

Lim, J. S., Kim, J-S and Lee, Y.I. (2014), Robust tracking model predictive control for input-constrained uncertain linear time invariant systems, *International Journal of Control*, 87(), 120–130.

Muske, K.R. and Rawlings, J.B. (1993), Model predictive control with linear models, *AIChE Journal*, 39(2), 262–287.

Pluymers, B., Rossiter, J., Suykens, J., and De Moor, B. (2005a). The efficient computation of polyhedral invariant sets for linear systems with polytopic uncertainty. In *ACC*, 804–809. IEEE.

Pluymers, B., Rossiter, J., Suykens, J., and De Moor, B. (2005b). A simple algorithm for robust mpc. In *IFAC World Congress*.

Qin, S. and Badgwell, T. (2003). A survey of industrial model predictive control technology. *Control Engineering Practice*, 11, 733–764.

Rakovic, S. and Mayne, D. (2005). A simple tube controller for efficient robust model predictive control of constrained linear discrete time systems subject to bounded disturbances. In *IFAC World Congress*.

Rao, C.V. and Rawlings, J.B. (1999). Steady states and constraints in model predictive control. *AIChE Journal*, 45(6), 1266–1278.

Rawlings, J.B., Bonn, D., Jrgensen, J., Venkat, A., and Jrgensen, S. (2008). Unreachable set-points in model predictive control. *IEEE Transactions on Automatic Control*, 53(9), 2209–2215. .

Richalet, J., Rault, A., Testud, J., and Papon, J. (1978), "Model predictive heuristic control: applications to industrial processes," *Automatica*, 14, 413–428.

Rossiter, J., Kouvaritakis, B., and Gossner, J.R. (1996). Guaranteeing feasibility in constrained stable generalised predictive control. *IEE Proceedings-Control Theory and Applications*, 143(5), 463–469.

Rossiter, J. and Grinnell, B. (1996). Improving the tracking of generalized predictive control controllers. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 210(3), 169–182.

Rossiter, J., Kouvaritakis, B., and Rice, M. (1998). A numerically robust state-space approach to stable predictive control stratgies. *Automatica*, 34(1), 65–73.

Rossiter, J.A. (2003). *Model-based predictive control: a practical approach*. CRC press.

Rossiter, J. (2006). A global approach to feasibility in linear mpc,. *Proc. UKACC ICC*.

Scokaert, P.O. and Rawlings, J.B. (1998). Constrained linear quadratic regulation. *Automatic*

Control, IEEE Transactions on, 43(8), 1163–1169.

Shead, L., Muske, K., and Rossiter, J. (2010). Conditions for which mpc fails to converge to the correct target. In *Journal of Process Control*, volume 20, 1207–1219.

Valencia-Palomo, G., Rossiter, J., and López-Estrada, F. (2014). Improving the feed-forward compensator in predictive control for setpoint tracking. *ISA transactions*, 53(3), 755–766.