Katholieke Universiteit Leuven

Departement Elektrotechniek

ESAT-SISTA/TR 05-163

The potential of interpolation for simplifying predictive control and application to LPV systems [1]

J.A. Rossiter[2] , B. Pluymers[3]

July 2005

Accepted for the Workshop on Assessment and Future Directions of NMPC, Freudenstadt-Lauterbad, Germany, 2005.

[1]This report is available by anonymous ftp from *ftp.esat.kuleuven.ac.be* in the directory *pub/sista/pluymers/reports/NMPC05_paper.pdf*

[2]University of Sheffield, Department of Automatic Control and Systems Engineering, Mappin Street, Sheffield S1 3JD, United Kingdom, E-mail: *j.a.rossiter@sheffield.ac.uk*

[3]K.U.Leuven, Dept. of Electrical Engineering (ESAT), Research group SCD-SISTA, Kasteelpark 10, 3001 Leuven, Belgium, Tel. 32/16/32 10 35, Fax 32/16/32 19 70, WWW: *http://www.esat.kuleuven.ac.be/scd*. E-mail: {*bert.pluymers,johan.suykens,bart.demoor*}*@esat.kuleuven.ac.be*.

Research supported by Research Council KUL: GOA-Mefisto 666, GOA-Ambiorics, several PhD/postdoc & fellow grants; Flemish Government: FWO: PhD/postdoc grants, projects, G.0240.99 (multilinear algebra), G.0407.02 (support vector machines), G.0197.02 (power islands), G.0141.03 (Identification and cryptography), G.0491.03 (control for intensive care glycemia), G.0120.03 (QIT), G.0452.04 (QC), G.0499.04 (robust SVM), research communities (ICCoS, ANMMM, MLDM); AWI: Bil. Int. Collaboration Hungary/ Poland; IWT: PhD Grants, GBOU (McKnow) Belgian Federal Government: Belgian Federal Science Policy Office: IUAP V-22 (Dynamical Systems and Control: Computation, Identification and Modelling, 2002-2006), PODO-II (CP/01/40: TMS and Sustainibility); EU: FP5-Quprodis; ERNSI; Eureka 2063-IMPACT; Eureka 2419-FliTE; Contract Research/agreements: ISMC/IPCOS, Data4s, TML, Elia, LMS, IPCOS, Mastercard; Bert Pluymers is a research assistant with the I.W.T. (Flemish Institute for Scientific and Technological Research in Industry). Dr. Johan Suykens is an associate professor at the Katholieke Universiteit Leuven. Dr. Bart De Moor is a full professor at the Katholieke Universiteit Leuven, Belgium.

Abstract

This paper first introduces several interpolation schemes, which have been derived for the linear time invariant case, but with an underlying objective of trading off performance for online computational simplicity. It is then shown how these can be extended to linear parameter varying systems, with a relatively small increase in the online computational requirements. Some illustrations are followed with a brief discussion on areas of potential development.

The potential of interpolation for simplifying predictive control and application to LPV systems

John Anthony Rossiter[†] and Bert Pluymers[‡]

[†] *Department Automatic Control and Systems Engineering, Mappin Street, University of Sheffield, S1 3JD, UK,*
j.a.rossiter@sheffield.ac.uk
[‡] *Department of Electrical Engineering, ESAT-SCD-SISTA Kasteelpark Arenberg 10, Katholieke Universiteit Leuven, B-3001 Heverlee (Leuven), Belgium,*
bert.pluymers@esat.kuleuven.be

Keywords : Predictive control, LPV systems, interpolation, computational simplicity, feasibility

This paper first introduces several interpolation schemes, which have been derived for the linear time invariant case, but with an underlying objective of trading off performance for online computational simplicity. It is then shown how these can be extended to linear parameter varying systems, with a relatively small increase in the online computational requirements. Some illustrations are followed with a brief discussion on areas of potential development.

1 Introduction

One of the key challenges in predictive control is formulating an optimisation which can be solved fast enough while giving properties such as guaranteed closed-loop stability and recursive feasibility. Furthermore one would really like good expectations on performance. A typical compromise is between algorithm or computational complexity and performance/feasibility. This paper looks at how reparameterising the input sequence using interpolation gives one possible balance, that is, it focuses on maximising feasible regions for a given algorithm/computational complexity without sacrificing asymptotic performance. The paper also considers some of the barriers to progress and hence suggests possible avenues for further research and in particular the potential for application to nonlinear systems. Several types of interpolation will be discussed, including:

1. Simple interpolation between control laws [21, 1]; complexity is linked to the state dimension.
2. Interpolations making use of control strategies which depend explicitly on the current states rather than a few linear feedbacks. One example of this is parametric programming solutions [4].
3. So called Triple mode type strategies [8] use a fixed non-linear strategy as a terminal mode.

Section 2 gives background information. In Section 3, the initial focus is on the conceptual thinking behind interpolation techniques and how they can widen feasibility while restricting complexity; to aid clarity, this is introduced using linear time invariant (LTI) models. Section 4 then focuses on how these concepts can be extended to allow application to nonlinear systems which can be described by an LPV model. Section 5 gives some numerical illustrations and the paper finishes with a discussion.

2 Background

This section introduces notation, the LPV model to be used in the paper, basic concepts of invariance, feasibility and performance, and some prediction equations.

2.1 Model and objective

Define the LPV model and constraints to take the form:

$$x(k+1) = A(k)x(k) + B(k)u(k), \qquad k = 0, \ldots, \infty, \tag{1a}$$

$$[A(k)\ B(k)] \in \Omega \triangleq \mathrm{Co}\{[A_1\ B_1], \ldots, [A_m\ B_m]\}, \tag{1b}$$

$$u(k) \in \mathcal{U} \equiv \{u : A_u u \leq \mathbf{1}\}, \qquad k = 0, \ldots, \infty, \tag{1c}$$

$$x(k) \in \mathcal{X} \equiv \{x : A_x x \leq \mathbf{1}\}, \qquad k = 0, \ldots, \infty, \tag{1d}$$

where **1** is a column vector of appropriate dimensions containing only 1's, which implies that the origin should lie within the imposed constraints. In what follows, when dealing with LTI models ($m = 1$), we will talk about the *nominal case*, while when dealing with LPV models, we will talk about the *uncertain* or *nonlinear case*. In this and future sections, the following feedback law is implicitly assumed :

$$u(k) = -Kx(k); \quad \forall k. \tag{2}$$

We note that the results of this paper have been proven only for feedback gains giving quadratic stabilisability, that is, for feedback K, there must exist a matrix $P = P^{\mathrm{T}} > 0 \in \mathbb{R}^{n_x \times n_x}$

$$\Phi_j^{\mathrm{T}} P \Phi_j \leq P, \quad \forall j, \quad \Phi_j = A_j - B_j K. \tag{3}$$

For a given feedback, the constraints (1c,d) at each sample are summarised as:

$$\mathcal{S}_0 = \{x : A_y x \leq \mathbf{1}\}; \quad A_y = \begin{bmatrix} -A_u K \\ A_x \end{bmatrix}. \tag{4}$$

Problem 1 (Cost Objective) *For each of the algorithms discussed, the underlying aims are: to achieve* robust *stability, to optimise performance and to guarantee* robust *satisfaction of constraints. In this paper we will use a single objective throughout, regardless of the algorithm. Hence the algorithm will seek to minimise, subject to robust satisfaction of (4), an upper bound on a performance index of the form:*

$$J = \sum_{k=0}^{\infty} (x(k)^{\mathrm{T}} Q x(k) + u(k)^{\mathrm{T}} R u(k)). \tag{5}$$

2.2 Invariant Sets

Invariant sets [3] are key to the algorithms overviewed in this paper and hence are introduced next.

Definition 1 (Feasibility and robust positive invariance) *Given a system, stabilizing feedback and constraints (1,2,4), a set $\mathcal{S} \subset \mathbb{R}^{n_x}$ is feasible iff $\mathcal{S} \subseteq \mathcal{S}_0$. Moreover, the set is robust positive invariant iff*

$$x \in \mathcal{S} \quad \Rightarrow \quad (A - BK)x \in \mathcal{S}, \quad \forall [A \ B] \in \Omega. \tag{6}$$

Definition 2 (MAS) *The largest feasible invariant set (no other feasible invariant set can contain states outside this set) is uniquely defined and is called the Maximal Admissible Set (MAS, [6]).*

2.2.1 MAS for the nominal case

Define the closed-loop predictions for a given feedback K as:

$$x(k) = \Phi^k x(0); \quad u(k) = -K\Phi^{k-1} x(0); \quad \Phi = A - BK. \tag{7}$$

Then, under mild conditions [6] the MAS for a controlled LTI system is given by

$$\mathcal{S} = \bigcap_{k=0}^{n} \{x : \Phi^k x \in \mathcal{S}_0\} = \{x : Mx \leq \mathbf{1}\}, \tag{8}$$

with n a finite number. In future sections, we will for the sake of brevity use the shorthand notation $\lambda S \equiv \{x : Mx \leq \lambda \mathbf{1}\}$.

2.2.2 MCAS for the nominal case

The MCAS (maximum control admissible set) is defined here for the specific control sequence:

$$\begin{array}{ll} u_i = -Kx_i + c_i, & i = 0, ..., n_c - 1, \\ u_i = -Kx_i, & i \geq n_c, \end{array} \tag{9}$$

which means we define this set as the set of all states for which there exists an input sequence (9) that guarantees robust constraint satisfaction. By computing the predictions given a model/constraints (1,4) and control law (9), it is easy to show that, for suitable M, N, the MCAS is given as ([19, 26]):

$$\mathcal{S}_{\mathrm{MCAS}} = \{x : \exists C \ \text{ s.t. } \ Mx + NC \leq \mathbf{1}\}; \quad C = [c_0^{\mathrm{T}} \ ... \ c_{n_c-1}^{\mathrm{T}}]^{\mathrm{T}}. \tag{10}$$

2.2.3 Ellipsoidal and polyhedral invariant sets for the LPV case

This section establishes that ellipsoidal and polyhedral invariant sets can be computed for the LPV case. We assume the reader is already familiar with the weaknesses of ellipsoidal sets by way of the restricted shape implying restricted use [16], particularly for asymmetric constraints. Nevertheless, given assumption (3), it is obvious that an ellipsoidal set $\mathcal{S}_E = \{x | x^{\mathrm{T}} P x \leq 1\}$ is invariant and scaling of P can ensure that $\mathcal{S}_E \subseteq \mathcal{S}_0$. Larger volume ellipsoids can be calculated using results such as those in [10].

Recent developments [15, 14] have demonstrated the tractability of algorithms to compute MAS (polyhedral invariant sets) for an LPV system. An outline of the conceptual idea is as follows. Search iteratively, beginning with an outer estimate (say $\mathcal{S}_0$), for an $\mathcal{S} = \{x : Mx \leq \mathbf{1}\}$ such that $x \in \mathcal{S} \quad \Rightarrow \Phi_j x \in \mathcal{S}, \ \forall j$. If the current outer estimate for $\mathcal{S}$ does not satisfy this test, then it is augmented with additional rows $M\Phi_j, \ \forall j$ and the test is repeated. Given condition (3), this process converges in finite time. Current work (e.g. [17]) is looking at how to make this algorithm more efficient and also to reduce the complexity of the resulting set $\mathcal{S}$, but that is not a main topic here.

2.3 Background for interpolation

Define a number of alternative stabilizing feedbacks $K_i, i = 1, \ldots, n$ of which K_1 is the preferred choice.

Definition 3 (Invariant sets) *For each K_i, define closed-loop transfer matrices Φ_{ij} and corresponding robust invariant sets $\mathcal{S}_i$ and also define the convex hull $\overline{\mathcal{S}}$:*

$$\Phi_{ij} = A_j - B_j K_i, \quad j = 1, ..., m; \quad \mathcal{S}_i = \{x : x \in \mathcal{S}_i \Rightarrow \Phi_{ij} x \in \mathcal{S}_i, \forall j\}, \tag{11}$$

$$\overline{\mathcal{S}} \triangleq \mathrm{Co}\{\mathcal{S}_1, \ldots, \mathcal{S}_n\}. \tag{12}$$

Definition 4 (Feasibility) *Let $\Phi_i(k) = A(k) - B(k)K_i$, then it can be shown [1] that the following input sequence and the corresponding state predictions are recursively feasible within $\overline{\mathcal{S}}$:*

$$\begin{array}{l} u(k) = -\sum_{i=1}^{n} K_i \prod_{j=0}^{k-1} \Phi_i(k-1-j) x_i, \\ x(k) = \sum_{i=1}^{n} \prod_{j=0}^{k-1} \Phi_i(k-1-j) x_i, \end{array} \tag{13}$$

if one ensures that

$$x(0) = \sum_{i=1}^{n} x_i, \quad \text{with} \quad \left\{ \begin{array}{l} x_i = \lambda_i \hat{x}_i, \\ \sum_{i=1}^{n} \lambda_i = 1, \ \lambda_i \geq 0, \\ \hat{x}_i \in \mathcal{S}_i. \end{array} \right. \tag{14}$$

Definition 5 (Cost) *With $\tilde{x} = [\hat{x}_1^{\mathrm{T}} \ \ldots \ \hat{x}_n^{\mathrm{T}}]^{\mathrm{T}}$, Lyapunov theory can be used to compute an upper bound $\tilde{x}^{\mathrm{T}} P \tilde{x}$ on the infinite-horizon cost J for predictions (13) using:*

$$P \geq \Gamma_u^{\mathrm{T}} R \Gamma_u + \Psi_i^{\mathrm{T}} \Gamma_x^{\mathrm{T}} Q \Gamma_x \Psi_i + \Psi_i^{\mathrm{T}} P \Psi_i, \quad i = 1, \ldots, m, \tag{15}$$

with $\Psi_i = \mathrm{diag}(A_i - B_i K_1, \ldots, A_i - B_i K_n)$, $\Gamma_x = [I, \ldots, I]$, $\Gamma_u = [K_1, \ldots, K_n]$.

3 Interpolation schemes for LTI systems

Interpolation is a different form of methodology to the more usual MPC paradigms. For this technique, one assumes that there exist different static linear state feedback strategies with significantly different properties. For instance one may be tuned for optimal performance and another to maximise feasibility. One then interpolates between the predictions (13) associated with these strategies to get the best performance subject to feasibility. The underlying motivation is that one may achieve large feasible regions with fewer optimisation variables, at some small loss to performance, and hence facilitate application to fast sampling scenarios. This section gives a brief overview of five alternative interpolation schemes that have been applied to the LTI case, with some attention given to their relative merits and de-merits. This will serve as a useful and more palatable context for extensions to the LPV case.

3.1 One degree of freedom interpolations [20, 21]

ONEDOF is a very simple algorithm where the interpolation of (13) are restricted to those for which

$$x = x_1 + x_2; \quad x_1 = (1-\alpha)x; \quad x_2 = \alpha x; \quad 0 \le \alpha \le 1. \tag{16}$$

Such a restriction implies that α is the only d.o.f. and optimisation over one d.o.f. is generally trivial and easy to code. Moreover, if K_1 is the optimal feedback, then one can show that minimising J of (5) over predictions (16,13) is equivalent to minimising $\alpha,\ \alpha \ge 0$, subject of course to feasibility. Two algorithms have been proposed both with an assured feasible region of $\bigcup_i \mathcal{S}_i$.

Algorithm 1 *[ONEDOFa] The first control move is* $u = -[(1-\alpha)K_1 + \alpha K_2]x$ *where:*

$$\alpha = \min_{\alpha} \quad \alpha \quad \text{s.t.} \quad [M_1(1-\alpha) + M_2\alpha]x \le \mathbf{1}; \quad 0 \le \alpha \le 1. \tag{17}$$

Algorithm 2 *[ONEDOFb] The first control move is* $u = -[(1-\alpha)K_1 + \alpha K_2]x$ *where:*

$$\alpha = \min_{\alpha,\beta} \quad \alpha \quad \text{s.t.} \quad \begin{cases} M_1(1-\alpha)x \le (1-\beta)\mathbf{1}, \\ M_2\alpha x \le \beta \mathbf{1}, \\ 0 \le \beta \le 1; \quad 0 \le \alpha \le 1. \end{cases} \tag{18}$$

This is known to be solved by $\alpha = (\mu - 1)/(\mu - \lambda)$ *where* $\mu = \max(M_1 x), \quad \lambda = \max(M_2 x)$.

Summary: It can be shown that ONEDOFa will, in general, outperform ONEDOFb and have a larger feasible region. However, a proof of recursive feasibility has not been found for ONEDOFa whereas it has for ONEDOFb. Convergence proofs only exist for some cases [21], although minor modifications to ensure this are easy to include, e.g. [23]. However, the efficacy of the method relies on the existence of a known controller K_2 with a sufficiently large feasible region!

3.2 GIMPC: MPC using General Interpolation

GIMPC [1] improves on ONEDOF by allowing full flexibility in the decomposition (13) of x, but of course the downside is that the number of optimisation variables increases to $n_x + 1$.

Algorithm 3 (GIMPC) *Take a system* (1)*, constraints* (4)*, cost weighting matrices* Q, R*, controllers* K_i *and invariant sets* $\mathcal{S}_i$ *and compute a suitable* P *from (15). Then, at each time instant, given the current state (for simplicity of notation denoted as* $x(0)$*), solve the following optimization:*

$$\min_{\hat{x}_i, \lambda_i} \ \tilde{x}^{\mathrm{T}} P \tilde{x}, \quad \textit{subject to } (14), \tag{19}$$

and implement the input $u = -\sum_{i=1}^{n} K_i x_i$.

Summary: The increased flexibility in the decomposition of x gives two benefits: (i) a guarantee of both recursive feasibility and convergence is straightforward and (ii) the feasible region is enlarged to $\overline{\mathcal{S}}$. The downside is an increase in the number of optimisation variables.

3.3 GIMPC2 interpolations

GIMPC includes the restriction (14) that $\sum_{i=1}^{n} \lambda_i = 1,\ \lambda_i \ge 0$. However, it has been shown recently [25] that such a restriction is unnecessary when the sets S_i are polyhedral.

Algorithm 4 (GIMPC2) *Using the same notation as algorithm 3, at each time instant, given the current state* x*, solve the following optimization problem on-line*

$$\min_{x_i} \ \tilde{x}^{\mathrm{T}} P \tilde{x}, \quad \textit{subject to} \quad \begin{cases} \sum_{i=1}^{n} M_i x_i \le \mathbf{1}, \\ x = \sum_{i=1}^{n} x_i, \end{cases} \tag{20}$$

and implement the input $u = -\sum_{i=1}^{n} K_i x_i$*, where the* M_i *defines a generalized MAS* $\mathcal{S}'_i$ *with mutually consistent constraints. See Algorithm 7 for details.*

Summary: If the constraints on λ_i implicit in algorithm 3 (or eqn.(14)) are removed one gets two benefits: (i) the feasible region may become substantially larger (illustrated later) than $\overline{\mathcal{S}}$ and moreover (ii) the number of optimisation variables reduces. One still has guarantees of recursive feasibility and convergence. So GIMPC2 outperforms GIMPC on feasibility, performance and computational load. The downside is that the sets $\mathcal{S}'_i$ maybe more complex. This is discussed later, for instance in Algorithm 7.

3.4 Interpolations to simplify parametric programming (IMPQP)

One of the active areas of research within parametric programming [4] solutions to MPC is the issue of how to speed up the search for the active region [7], or to reduce the number of regions in the first place. Here we show how interpolation can tackle the latter of these issues, although it does assume one can find the maximal controlled admissible set (MCAS) defined in eqn.(10) and from this, the parametric regions contributing to the outer boundary. Hence it is still restricted to those problems where such computations are feasible.

The idea of interpolation MPQP (IMPQP) [23] is fairly simple. On the outer boundary of the MCAS, in some given region, the optimal predicted control law can be summarised as:

$$x \in \mathcal{R}_i \quad \Rightarrow \quad u = -Kx + c_i; \quad c_i = -K_i x + p_i. \tag{21}$$

For an x not on the boundary, but for which a scaled version (by $1/\rho$) would lie in $\mathcal{R}_i$ on the boundary, then the following control law can be shown, by linearity, to give recursive feasibility and convergence:

$$\frac{x}{\rho} \in \mathcal{R}_i \quad \Rightarrow \quad u = -Kx + \rho c_i; \quad c_i = -K_i x + p_i. \tag{22}$$

Algorithm 5 (IMPQP - suboptimal) Offline: *First compute the MPQP solution and find the regions contributing to the boundary. Summarise the boundary of the MCAS in the form $M_b x \leq \mathbf{1}$ and form a lookup table of all regions contributing to each boundary facet.*
Online: *Identify the active facet from $\rho = \max_j M_b(j,:)x$. With this ρ, find a feasible and convergent control law from (22). (In fact one can use algorithm ONEDOFb to improve the choice of c further still.)*

Algorithm 6 (IMPQP - optimal) Offline: *As above but in addition, define the MCAS as in* (10).
Online: *Identify ρ as above and define the suboptimal C from (22). Perform the interpolation*

$$\min_{\alpha} \alpha \quad \text{s.t.} \quad Mx + N\alpha C \leq \mathbf{1}, \tag{23}$$

and implement $u = -Kx + \alpha e_1^T C$. (There are some minor subtleties to guarantee convergence [23].)

Summary: For many MPQP solutions, optimal or otherwise, the IMPQP algorithm can be used to reduce complexity by requiring storage only of boundary regions and their associated control laws. Montecarlo studies demonstrated that, despite a huge reduction in set storage requirements, the closed-loop behaviour was nevertheless often close to optimal.

3.5 Other algorithms

3.5.1 Triple mode algorithms

This class of algorithm [8, 20] is less obviously an interpolation and is included here solely because it tackles the same underlying issue: how can I improve feasible regions without a large increase in the required numbers of d.o.f.? Triple mode strategies look at ways of defining a single, perhaps LTV, control trajectory with as large as possible a feasible region. This is then used like a terminal mode. The power of the approach is that by allowing an LTV law in the terminal mode, one can gradually tune up the feedback, in a prescribed way, as the state moves nearer to the origin. There are of course strong parallels with algorithms such as [27, 28]. In summary the 'predicted' control law takes the form

$$\begin{aligned} u_i &= -Kx + c_i \quad i = 0, ..., n_c - 1, \\ u_i &= -K_i x \quad i = n_c, ..., m_c + n_c - 1, \\ u_i &= -Kx \quad i \geq n_c + m_c. \end{aligned} \tag{24}$$

The objective is to design [24, 5] the intermediate controllers K_i to get some compromise between the volume of the feasible region and the associated cost J. The d.o.f. in the optimisation are the variables c_i which are far fewer in number, for a given feasible region, than without the intermediate mode.

3.5.2 Using vertices

One other well known use of interpolation, for which it is easy to demonstrate convergence and feasibility, is based on vertices of some invariant set. Find a feasible trajectory for a number of vertices v_i, then inside the convex hull of these vertices one can find a feasible control law as a convex combination of those trajectories. This technique is not covered here but clearly, as with parametric methods, may suffer from issues of complexity, especially where n_x is large.

4 Extensions to the LPV case

The previous section dealt solely with the nominal case. However, the focus of this workshop is the nonlinear case. Hence this section shows how the interpolation methods presented can be extended (e.g. [16]) for application to nonlinear systems which can be represented by an LPV model. Application for more general nonlinearities forms future work.

4.1 Invariance

A major tool used to establish recursive feasibility is the existence of a known feasible invariant set, say $\mathcal{S}_i$. All the interpolation algorithms made use of such sets, including IMPQP where such a set was posed as an MAS or MCAS. The conjecture put forward next is that all of the interpolation algorithms can be carried across to the LPV case with only small changes, as long as one can compute the corresponding invariant sets. Hence, this section first covers, briefly, how robust polyhedral invariant sets might be computed to meet the needs of the various algorithms.

4.2 Invariant sets and interpolation for GIMPC and ONEDOFb

The GIMPC and ONEDOFb algorithms work on terms of the form $\max_j M(j,:)x_i$. As long as the invariant set is valid, this value is unique and, as a consequence, one can determine the sets $\mathcal{S}_i$ associated to each K_i independently. This is useful because one can then find the set of minimal complexity and with an efficient algorithm.

Summary: Extension of GIMPC and ONEDOFb to the LPV case is straightforward, as long as polyhedral sets $\mathcal{S}_i$ exist and one replaces J with a suitable upper bound [1]. The implied online computational load increases marginally because the sets $\mathcal{S}_i$ for the LPV case are likely to be more complex.

4.3 Invariant sets and interpolation for GIMPC2 and ONEDOFa

The algorithm of [15] was defined to find the MAS of an LPV system for a single control law. Due to issues of possible combinatorial explosion in the number of inequalities, it was necessary to include within the iteration frequent checks for and removal of redundant constraints. However, for the GIMPC2 algorithm, constraints may need to be retained [25] even where they are redundant in the individual MAS, because the implied constraints in the rows of matrices M_i must match in the combined forms of (17,20). For the LPV case, due to combinatorial explosions, one can not simply define the MAS as in (8) and then remove redundant constraints afterwards. Hence an efficient algorithm to deal with this complexity is required. Here one possibility, forming an augmented system, is discussed.

Algorithm 7 (Method to find mutually consistent MAS for the LPV case)

1. *Define an augmented system*

$$X(k+1) = \Psi(k)X(k); \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \tag{25}$$

$$\Psi(k) = \begin{bmatrix} A(k)-B(k)K_1 & 0 & \dots & 0 \\ 0 & A(k)-B(k)K_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A(k)-B(k)K_n \end{bmatrix}.$$

Define a set $\hat{\Omega}$ with $\Psi \in \hat{\Omega}$, describing the allowable variation in Ψ due to the variations implied by $[A(k)\ B(k)] \in \Omega$.

2. *Constraints (4) need to be written in terms of augmented state X as follows:*

$$A_u \underbrace{[-K_1, -K_2, \cdots]}_{\hat{K}} X(k) \le \mathbf{1}, \qquad k = 0, \dots, \infty, \tag{26a}$$

$$A_x [I, I, \cdots] X(k) \le \mathbf{1}, \qquad k = 0, \dots, \infty. \tag{26b}$$

3. *Assume that an outer approximation to the MAS is given by (26). Then letting* $u = -\hat{K}X$, *this reduces to* $\mathcal{S}_o = \{X : M_o X \leq \mathbf{1}\}$ *where the definition of* M_o *is obvious.*

4. *Follow steps 2-5 of Algorithm in [15] to find the robust MAS as* $\mathcal{S}_a = \{X : M_a X \leq \mathbf{1}\}$.

Remark 1 (Feasible region for robust GIMPC2) *Given the constraint* $x = \sum_{i=1}^{n} x_i$, *then one can find a projection of* $\mathcal{S}_a$ *to* x*-space from* X*-space as follows:*

$$\mathcal{S}_{G2} = \{x : \ \exists X \text{ s.t. } M_a X \leq \mathbf{1}, \ x = [I, I, \ldots, I]X\}. \tag{27}$$

Remark 2 *There are alternative ways of forming an augmented system/states using the observation that* $x = \sum x_i \quad \Rightarrow \quad x_r = x - x_1 - x_2 - ... - x_{r-1}$. *Investigations into preferred choices are ongoing.*

The basic suggestion hereafter is unsurprisingly quite simple.

Algorithm 8 (GIMPC2 for the LPV case) *Given a system* (1), *constraints* (4), *cost weighting matrices* $Q = Q^{\mathrm{T}} > 0, R = R^{\mathrm{T}} > 0$, *asymptotically stabilizing controllers* K_i, *corresponding polyhedral robust invariant sets* $\mathcal{S}_a = \{X : M_a X \leq \mathbf{1}\}$ *and* P *satisfying* (15), *solve on-line at each time instant, the following problem:*

$$\min_{\hat{x}_i} \tilde{x}^{\mathrm{T}} P \tilde{x}, \textit{subject to} \quad \begin{cases} x = [I, I, \ldots, I]X, \\ M_a X \leq \mathbf{1}, \end{cases} \tag{28}$$

and implement input $u = -[K_1, \ K_2, \ \ldots \ , \ K_n]X$.

Theorem 1 *Algorithm 8 guarantees robust satisfaction of* (4) *and is recursively feasible and asymptotically stable for all initial states* $x(0) \in \mathcal{S}_{G2}$.

Proof: First show that, irrespective of the values $A(k), B(k)$ (or $\Psi(k)$), that nevertheless

$$x(k) \in \mathcal{S}_{G2} \quad \Rightarrow \quad x(k+1) \in \mathcal{S}_{G2}. \tag{29}$$

This follows from the invariance and feasibility of $\mathcal{S}_a$. Asymptotic stability follows similar lines: it is known that one can always choose new state components at the next time step as those as suggested in (14). However, repeated choice of the same decomposition must give convergence due to the quadratic stability (3) associated to each individual control law, and hence also to augmented system Ψ. Deviation away from this scheme will only occur where the cost $J = \tilde{x}^{\mathrm{T}} P \tilde{x}$ can be made smaller still. Hence, the optimal value of the cost function (28) acts as a Lyapunov function of the closed-loop system. □

Remark 3 *One can identify the underlying MAS,* $\mathcal{S}_i$ *from* $\mathcal{S}_a$ *by setting* $x_j = 0, \ j \neq i$ *in turn, that is* $M_1 = M_a[I, 0, \ldots, 0]^{\mathrm{T}}, \ \ldots, \ M_n = M_a[0, I, 0, \ldots]^{\mathrm{T}}$. *These* M_i *may contain redundant rows.*

Summary: Extension to the LPV case is not straightforward for GIMPC2 and ONEDOFa because the form of constraint inequalities implicit in the algorithms is $M_1 x_1 + M_2 x_2 + ... \leq \mathbf{1}$ and this implies a fixed and mutual consistent structure in M_i; they can no longer be computed independently! This requirement can make the matrices M_i far larger than would be required by say GIMPC. Once consistent sets $\mathcal{S}_i$ have been defined, the interpolation algorithms GIMPC2 and ONEDOFa are identical to the LTI case, so long as the cost J is replaced by a suitable upper bound.

4.4 Extension of IMPQP to the LPV case

Extension of IMPQP to the LPV case is straightforward if one can compute the MCAS for the robust case, but a few technical details, analogous to the previous section, are required before this can be done. Although alternative procedures are possible, the neatest algorithm makes use of an autonomous model [11] (that is model (1) in combination with control law (9)) to represent d.o.f. during transients. Hence, define a model for the 'predicted' closed-loop dynamics as:

$$z_{k+1} = \Psi z_k; \quad z = \begin{bmatrix} x \\ C \end{bmatrix}; \quad \Psi = \left[\begin{array}{c|c} \Phi & B \quad 0 \\ \hline 0 & U \end{array}\right]; \ U = \begin{bmatrix} 0 & I_{(n_c-1)n_u \times (n_c-1)n_u} \\ 0 & 0 \end{bmatrix}. \tag{30}$$

Naturally, given (1), Ψ has an LPV representation. Define the equivalent constraint set $\mathcal{S}_0$:

$$\mathcal{S}_0 = \{x : \tilde{A}_y z \leq \mathbf{1}\}; \quad \tilde{A}_y = \begin{bmatrix} A_u K & B & 0 \\ A_x & 0 & 0 \end{bmatrix}. \tag{31}$$

One can now form the MAS for system (30) with constraints (31) using the conventional algorithm. This set, being linear in both x and C, will clearly take the form of (10) and therefore can be deployed in an MPQP algorithm. One also needs to reformulate the predicted cost for the autonomous model. One can either form a tight upper bound as in [1] or take a more simple minded, but suboptimal approach, using something like $J = C^T C$. Guaranteed convergence is easy to establish for both these options, assuming recursive feasibility which is automatic with the correct definition of the robust MCAS.

Summary: Application of IMPQP to the LPV case can be done through the use of an autonomous model to determine the MCAS. Apart from the increase in offline complexity and obvious changes to the shape of the parametric solution, there is little conceptual difference between the LTI and LPV solutions.

4.5 Polyhedral invariant sets for triple mode MPC

The results discussed in previous sections strongly indicate that the combination of the results of [5, 8] with the invariant set work of [15] is possible. We aim to report on this soon.

4.6 Summary

We summarize the changes required to extend nominal interpolation algorithms to the LPV case.

1. The simplest ONEDOF interpolations can make use of a robust MAS, in minimal form, and apart from this no changes from the nominal algorithm are needed. The simplest GIMPC algorithm is similar except that the cost needs to be represented as a minimum upper bound.

2. More involved ONEDOF interpolations require non-minimal representations of the robust MAS to ensure consistency between respective $\mathcal{S}_i$, and hence require many more inequalities. The need to compute these simultaneously also adds significantly to the offline computational load.

3. The GIMPC2 algorithm requires both mutual consistency of the MAS and the cost to be replaced by a minimum upper bound.

4. Interpolation MPQP requires the robust MCAS which can be determined using an autonomous model representation, although this gives a large increase in the dimension of the invariant set algorithm. It also needs an upper bound on the predicted cost.

It should be noted that recent results [17] indicate that in the LPV case the number of additional constraints can often be reduced significantly with a modest decrease in feasibility.

5 Numerical Example

In this section we use a double integrator example with non-linear dynamics, to demonstrate the various interpolation algorithms. As the developments for the nominal case are subsumed by those for the LPV case, to save space, only the LPV case will be illustrated in detail. For comparison purposes we will adopt as a benchmark the algorithm of [26] (denoted OMPC) but modified to make use of robust MCAS [18].

5.1 Model and constraints

We consider the nonlinear model:

$$\begin{aligned} x_{1,k+1} &= x_{1,k} + 0.1(1 + (0.1x_{2,k})^2)x_{2,k}, \\ x_{2,k+1} &= x_{2,k} + (1 + 0.005x_{2,k}^2)u_k, \end{aligned} \tag{32a}$$

subject to constraints

$$-0.5 \leq u_k \leq 1, \qquad [-10 \ -10]^{\mathrm{T}} \leq x_k \leq [8 \ 8]^{\mathrm{T}}, \qquad \forall k. \tag{32b}$$

Using the imposed state constraints we obtain an LPV system bounding the non-linear behaviour :

$$A_1 = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}, B_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad A_2 = \begin{bmatrix} 1 & 0.2 \\ 0 & 1 \end{bmatrix}, B_2 = \begin{bmatrix} 0 \\ 1.5 \end{bmatrix}. \tag{33}$$

The nominal model is chosen as $[A_1\ B_1]$, which is the model that is valid around the origin. This model is chosen as the design basis for the two feedback controllers, the first of which is chosen as the LQR-optimal controller $K_1 = [0.4858\ 0.3407]^{\mathrm{T}}$ for weighting matrices $Q = \mathrm{diag}(1, 0.01), R = 3$ and the second of which is chosen to be the sub-optimal $K_2 = [0.3\ 0.4]^{\mathrm{T}}$ that is chosen for its large feasible region. Both controllers are robustly asymptotically stabilizing for system (33) and are hence also stabilizing for system (32).

5.2 Feasible Regions

First we demonstrate the efficacy of the interpolation algorithms in extending the feasible region beyond $\mathcal{S}_1$. First of all, for completeness, figure 1(a) shows the increase in feasibility achieved by using polyhedral sets as opposed to the more conventional ellipsoidal invariant sets that many authors use for the robust case. The further increase in feasible region when switching from GIMPC to GIMPC2 can also be observed. Figure 1(b) shows the controllable feasible regions for GIMPC2 and IMPQP for $n_c = 0, \ldots, 6$.

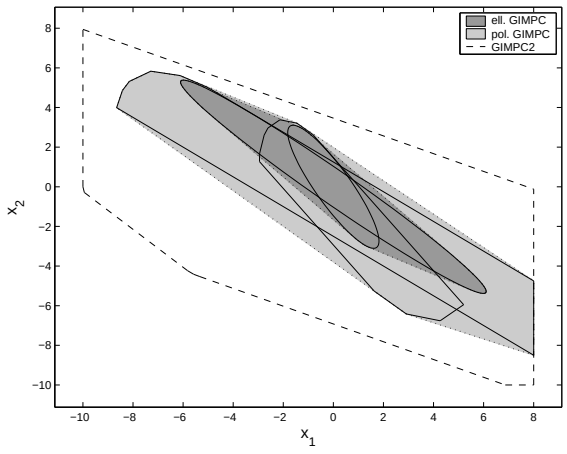
It is perhaps unsurprising that GIMPC2 has a larger region than GIMPC which in turn is larger than that for ONEDOFb. It is however surprising to see that, for this example, GIMPC2 gives far better feasibility than IMPQP with $n_c \leq 6$.

5.3 Computational Load

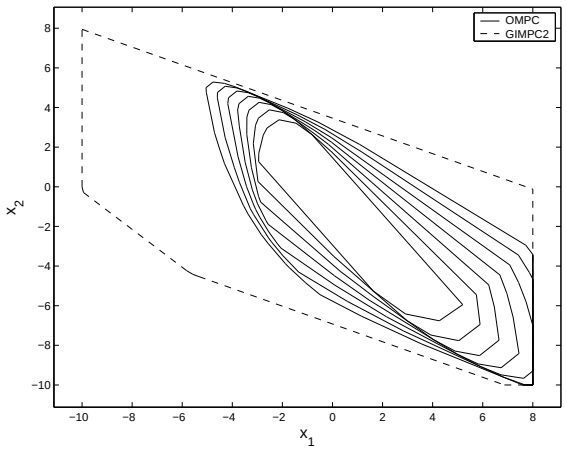
One might expect the computational load to increase significantly with the move from the nominal to the uncertain case. In fact, as indicated earlier, the optimisations take the same form and the only increase is due to the complexity of the invariant sets, which admittedly can be significant. However, some initial results have been obtained recently [17] that show how this number can be reduced significantly, largely eliminating the gap between the nominal and the robust case. For completeness table 1 shows the numbers of d.o.f. and the numbers of inequalities for each algorithm when applied to the LPV case. IMPQP is excluded from this table as the online computation is linked to the number of regions and hence is fundamentally different.

	GIMPC	GIMPC2	OMPC
No. inequalities	22	63	506
No. d.o.f.	$n_x + 1 = 3$	$n_x = 2$	$n_c = 6$

Table 1: Numbers of inequalities and d.o.f. required by GIMPC, GIMPC2 and OMPC for model (33).

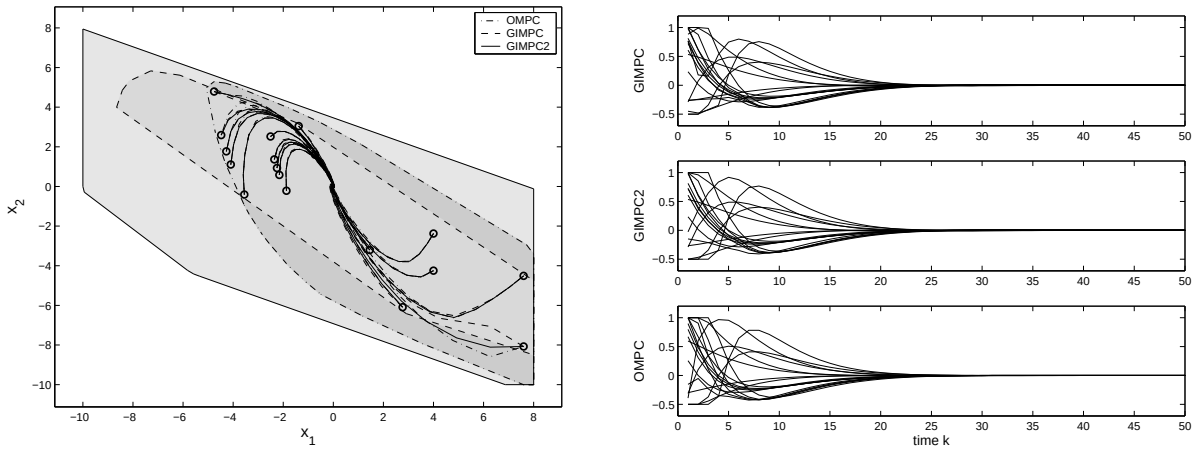


(a) Feasible regions of GIMPC using ellipsoidal and polyhedral invariant sets and GIMPC2.



(b) Feasible regions of IMPQP for $n_c = 0, \ldots, 6$ and GIMPC2.

Figure 1: Feasible regions for different algorithms for model (33) using feedback laws K_1 and K_2.



(a) State trajectories for the 3 different algorithms.

(b) Input sequences for the 3 different algorithms.

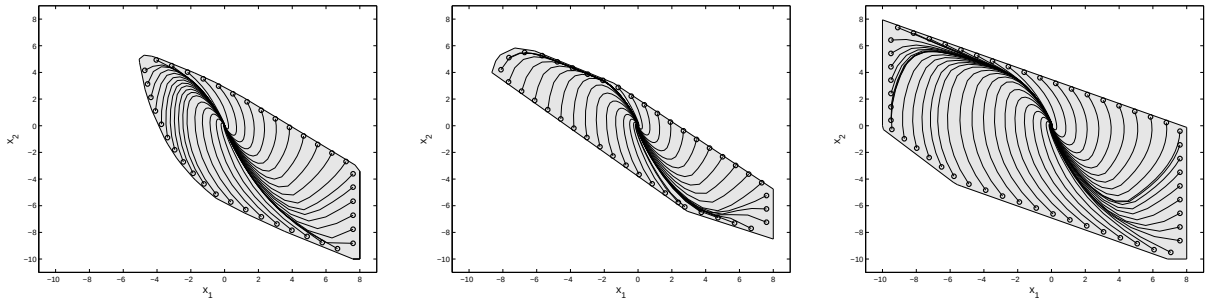
Figure 2: Trajectories for GIMPC, GIMPC2 and OMPC for plant model (32) using feedback laws K_1 and K_2 and design model (33), starting from initial states at the boundary and the inside of the intersection of the feasible regions.

5.4 Control Performance

Illustrations are only valid within the feasible region of the associated algorithm, which are clearly different for the three algorithms given here. However, it is useful to consider how the closed-loop performance compares to 'optimal', which here could be defined as the performance of OMPC. Hence, Figure 2 depicts simulation results for GIMPC, GIMPC2 and OMPC, starting from initial states on the boundary of the intersection of the different feasible regions, showing that all three algorithms are stabilizing and result in nearly identical trajectories. The average control cost (according to (5)) of algorithms GIMPC and GIMPC2 is respectively 1.7% and 0.3% higher than OMPC with $n_c = 6$.

5.5 Robust closed-loop behaviour

Some evidence is also provided that each of these algorithms is indeed robustly feasible and convergent for the entire feasible region. Hence, closed-loop trajectories, both of state and inputs, are computed for several points on the feasibility boundary and these are displayed in Figure 3. Clearly all state trajectories remain feasible for every algorithm.



(a) State trajectories for OMPC.

(b) State trajectories for GIMPC.

(c) State trajectories for GIMPC2.

Figure 3: Trajectories for OMPC, GIMPC and GIMPC2 for plant model (32) using feedback laws K_1 and K_2 and design model (33), starting from initial states at the boundaries of the respective feasible regions.

6 Conclusions and future directions

This paper has demonstrated that it is relatively simple to apply interpolation techniques to nonlinear systems which can be represented, locally, by an LPV model. Most significantly, it is clear that the interpolation algorithms allow a degree of performance optimisation allied to guarantees of recursive feasibility and convergence, while at the same time requiring relatively trivial online computation. The main requirement of the proposed algorithms is the ability to compute either the MAS or MCAS, but with certain structural restrictions to ensure consistency across several sets. It has also been noticed that some algorithms such as GIMPC2 give far larger feasible regions than might be intuitively expected.

There are several interesting future research directions. Clearly LPV models only include a small class of non linear processes and hence there is interest in whether interpolation concepts can be used effectively for more complicated non-linearities. Also, many papers consider only parameter uncertainty, or disturbance rejection whereas in a practical environment, both of these must be tackled. Work in progress is extending the algorithms in this paper to that case. On a similar vein, it is still unclear what may be a good mechanism for identifying the underlying feedbacks K_i or strategies which give large feasible regions. Triple mode ideas may be an effective way forward, but in general few other proposals exist in the literature. Finally of course, interpolation has been studied fairly extensively on low order systems and hence there is a need to consider their efficacy on high order processes.

On a slightly different tack, although relatively simple interpolations can be applied to LPV systems and give substantial feasible regions, these rely on invariant sets or parametric solutions which can require prohibitive offline computation and moreover may be so complicated as to eradicate the expected benefit. Hence important future developments also need to include efficient algorithms for computing low complexity, but large, invariant sets or solutions, as well as alternative ways of dealing with high order systems. Initial results in this direction have been obtained [17], but still need further development.

Acknowledgments: To the Royal Society and the Royal Academy of Engineering of the United Kingdom. Research Council KULeuven: GOA-Mefisto 666; Flemish Government: FWO: G.0240.99, G.0407.02, G.0197.02, G.0141.03, G.0491.03, G.0120.03 (QIT), research communities (ICCoS, ANMMM); AWI: Bil. Int. Collaboration Hungary/ Poland; IWT: PhD Grants, Soft4s, Belgian Federal Government: DWTC (IUAP IV-02 (1996-2001) and IUAP V-22 (2002-2006)), PODO-II (CP/40: TMS and Sustainibility); EU: CAGE; ERNSI; Eureka 2063-IMPACT; Eureka 2419-FliTE; Contract Research/agreements: Data4s, Electrabel, Elia, LMS, IPCOS, VIB. Bert Pluymers is a research assistant with the IWT at the KULeuven. The authors would like to thank Prof. Bart De Moor (KULeuven) and Prof. Johan Suykens (KULeuven) for their support and guidance.

References

[1] Bacic, M. and Cannon, M. and Lee, Y. I. and Kouvaritakis, B., General Interpolation in MPC and Its Advantages, IEEE Transactions on Automatic Control, 48, 6, 2003, pp1092–1096.

[2] Blanchini, F., Ultimate Boundedness Control for Uncertain Discrete-Time Systems via Set-Induced Lyapunov Functions, IEEE Transactions on Automatic Control, 1994, 39, pp428–433.

[3] Blanchini, F., Set Invariance in Control, Automatica, 1999, 35, pp1747–1767.

[4] Borelli, F., Constrained Optimal Control for Linear and Hybrid Systems, Springer-Verlag, Berlin, 2003.

[5] Cannon, M. and B. Kouvaritakis, Optimizing prediction dynamics for robust MPC, IFAC world congress, Prague, 2005.

[6] Gilbert, E. G. and Tan. K. T., Linear Systems with State and Control Constraints : The Theory and Application of Maximal Output Admissable Sets, IEEE Transactions on Automatic Control, 1991, 36, 9, pp1008–1020.

[7] Jones, C., P. Grieder and S. Rakovic, A Logarithmic-Time Solution to the Point Location Problem for Closed-Form Linear MPC, IFAC World Congress, Prague, 2005.

[8] Imsland, L. and J.A. Rossiter, Time varying terminal control, IFAC World Congress, Prague, 2005.

[9] Kerrigan, E., Robust Constraint Satisfaction: Invariant Sets and Predictive Control, PhD thesis, Cambridge, 2000.

[10] Kothare, M. V. and Balakrishnan, V. and Morari, M., Robust Constrained Model Predictive Control using Linear Matrix Inequalities, Automatica, 32, pp1361–1379, 1996.

[11] Kouvaritakis, B., J.A. Rossiter and J. Schuurmans, Efficient robust predictive control, 2000, IEEE Transactions on Automatic Control, 45, 8, pp1545–1549.

[12] Lee, Y. I. and Kouvaritakis, B., Robust Receding Horizon Predictive Control For Systems With Uncertain Dynamics and Input Saturation, Automatica, 2000, 36, pp1497–1504.

[13] Michalska, H. and D. Mayne, Robust receding horizon control of constrained nonlinear systems, IEEE Transactions on Automatic Control, 38, pp1623–1633, 1993.

[14] F. Blanchini, S. Miani, and C. Savorgnan. Polyhedral lyapunov functions computation for robust and gain scheduled design. *Proceedings of the Symposium on nonlinear Control Systems (NOLCOS) 2004, Stuttgart, Germany.*

[15] Pluymers, B. and Rossiter, J.A. and Suykens, J.A.K. and De Moor, B., The Efficient Computation of Polyhedral Invariant Sets for Linear Systems with Polytopic Uncertainty, American Control Conference, Portland, 2005.

[16] Pluymers, B. and Rossiter, J.A. and Suykens, J.A.K. and De Moor, B. Interpolation Based MPC for LPV Systems using Polyhedral Invariant Sets, American Control Conference, Portland, 2005.

[17] Pluymers, B., J.A. Rossiter, J. and Suykens, J. A. K. and De Moor, B., Reduced Complexity Polyhedral Invariant Sets for LPV Systems, Internal Report 04-230, ESAT-SISTA, K. U. Leuven (Leuven, Belgium).

[18] Pluymers, B., J.A. Rossiter, J.A.K. Suykens, B. De Moor, A simple algorithm for robust MPC, IFAC World Congress, Prague, 2005.

[19] Rossiter, J.A., M.J.Rice and B. Kouvaritakis, A numerically robust state-space approach to stable predictive control strategies, Automatica, 1998, 34, 1, pp65–73.

[20] Rossiter, J.A., B. Kouvaritakis and M. Cannon, Computationally efficient algorithms for constraint handling with guaranteed stability and near optimality, 2001, International Journal of Control, 74, 17, pp1678–1689.

[21] Rossiter, J.A. and Kouvaritakis, B. and Bacic, M., Interpolation based computationally efficient predictive control, International Journal of Control, 77, 3, pp290–301, 2004.

[22] Rossiter, J.A., Model based predictive control: a practical approach, CRC Press, 2003.

[23] Rossiter, J.A. and p. Grieder, Using interpolation to improve efficiency of multiparametric predictive control, Automatica, 41,4, 2005.

[24] Rossiter, J.A., B. Kouvaritakis and M. Cannon, Linear time varying terminal laws in MPQP, UKACC, 2004.

[25] Rossiter, J.A., Y. Ding, B. Pluymers, J.A.K. Suykens, B. De Moor, Interpolation based MPC with exact constraint handling: the uncertain case, to appear ECC/CDC, Sevilla, Spain, 2005.

[26] Scokaert, P.O.M. and Rawlings, J.B., Constrained linear quadratic regulation, IEEE Transactions on Automatic Control, 43, 8, pp1163–1168,1998.

[27] Tan, K.T. and E.G. Gilbert, Multimode controllers for linear discrete time systems with general state and control constraints, in Optimisation Techniques and Applications (World Scientific, Singapore), pp433–442, 1992.

[28] Wan, Z. and Kothare, M. V., Efficient robust constrained model predictive control with a time varying terminal constraint set, Systems and Control Letters, 48, pp375–383, 2003.