Proceedings of the 36th
Conference on Decision & Control
San Diego, California USA • December 1997

Predictive LQ control with constraints

B. Kouvaritakis M. Cannon
Department of Engineering Sciences
Parks Road
Oxford, OX1 3PJ, UK
email: Basil.Kouvaritakis @ eng.ox.ac.uk

J.A. Rossiter M.J. Rice
Department of Mathematical Sciences
Loughborough University
Leicestershire LE11 3TU, UK
e-mail: J.A.Rossiter@lboro.ac.uk,

Abstract

Recent work [13] has developed necessary and sufficient conditions for stable input and output predictions. Other work [14] has shown how to force a predictive controller onto the LQ optimal control law, in the constraint free case whereas a more recent algorithm [5], achieved this via a different means. Here the latter algorithm [5] is presented in a new way to illustrate the connections between it and the earlier work of [13], [14]. Furthermore examples are given to contrast the performance of the three algorithms.

1 Introduction

Predictive controllers have often been criticised for being suboptimal and lacking a guarantee of stability, both of which attributes an LQ controller possesses. Their popularity therefore seems largely due to the simplicity of the idea and the ability to incorporate constraints into the control law both explicitly and 'optimally'.

Since the main ideas on predictive control were clarified in the mid-eighties, researchers have sought to overcome these two main weaknesses, in the first case the lack of a stability guarantee and more recently the lack of 'LQ optimality'. It was soon understood that a guarantee of stability (for the nominal case) could be given if some form of terminal constraint was appended into the performance index; essentially this made the performance index to be finite when computed over an infinite horizon - this implies also that the optimal predictions were both stable and convergent to the correct set point.

In the earliest works (e.g. [1],[2],[3],[4],[6], [10], [7]) the terminal constraint was sufficient for stable predictions, but not necessary and moreover it is easy to show that even when constraints are not active, these approaches cannot reproduce the LQ optimal control law except asymptotically as the number of degrees of freedom tends to infinity. Hence, in this paper we focus on two of the most recent developments of [13] and [14], [5]. The first of these derives necessary and sufficient conditions for stable input/ouput prediction pairs; the predictions transpire to be simple functions of the initial condition the system unstable poles and zeros. As these predictions are necessary and sufficient for stability, they form a natural base for handling constraints. The disadvantage is that the LQ optimal solution can only be recovered if a large number of degrees of freedom are deployed. The second work [14] approaches the problem from the opposite direction, that is to say it begins with the optimal LQ solution and perturbs away from this solution to handle constraints. Its advantage is clearly that it reduces to the optimal LQ solution when constraints are inactive, its disadvantage is that the perturbations about the optimal for handling constraints are restricted to the have the optimal closed-loop poles which is only a sufficient condition for stability, hence in the face of tight constraints infeasibility/instability is more likely.

In this paper we develop further the work in [5] which also uses perturbations on an L_2 optimal solution. These perturbations however are chosen from the class of stable predictions and thus define a wider class than that used in [14]. The algorithm presented here combines the following desirable attributes:

- Reduces to the LQ optimal solution if feasible.
- Deploys necessary/sufficient conditions on perturbations about the optimal to handle constraints

Furthermore here we undertake a comparison between the earlier work of [13], [14] and the proposed algorithm and illustrate its superiority for a number of numerical examples.

The paper will be arranged as follows: Section 2 comprises notation, definitions and a summary of [13] and an input/output formulation of [14]. Section 3 introduces the proposed algorithm and Section 4 gives numerical examples which together with appropriate comparisons leads to a conclusions section.

Adaptions can be made if the set-point is not reachable, e.g. see [12]

0-7803-3970-8/97 $10.00 © 1997 IEEE

2 Background

In this paper we shall adopt the notation of bold for vectors and not bold for z-transforms. e.g.

$$\mathbf{m} = \begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ m_l \end{bmatrix}; \quad m = [1\ z^{-1}\ \dots z^{-l}]\mathbf{m} \qquad (1)$$

For simplicity take the nth order model to be the standard z-transform model (or difference equation model)

$$\begin{gathered} a(z)y_t = b(z)u_{t-1} \\ y_o + a_1 y_{-1} + \dots + a_n y_{-n} = b_1 u_{-1} + \dots + b_n u_{-n} \end{gathered} \qquad (2)$$

where z^{-1} is the delay operator. As is usual in predictive control augment the model with an integrator $1/\Delta$, $\Delta = 1 - z^{-1}$ and let $A = a\Delta$, $\Delta u_t = u_t - u_{t-1}$, hence

$$\begin{gathered} A(z)y_t = b(z)\Delta u_{t-1} \\ y_0 + A_1 y_{-1} + \dots + A_{n+1} y_{-n-1} = b_1 \Delta u_{-1} + \dots + b_n \Delta u_{-n} \end{gathered} \qquad (3)$$

Define the Hankel and Toeplitz matrices H_d, Γ_d as

$$[H_d \mid \Gamma_d] = \left[\begin{array}{ccc|cccc} d_r & d_{r-1} & \dots & d_o & 0 & \dots & 0 \\ 0 & d_r & \dots & d_1 & d_0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \vdots & d_r & d_{r-1} & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{array}\right] \qquad (4)$$

Define vectors of input/output predictions as

$$\mathbf{y}_N = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}; \quad \mathbf{\Delta u}_N = \begin{bmatrix} \Delta u_0 \\ \vdots \\ \Delta u_N \end{bmatrix} \qquad (5)$$

and

$$\lim_{N\to\infty} \mathbf{y}_N = \mathbf{y}; \quad \lim_{N\to\infty} \mathbf{\Delta u}_N = \mathbf{\Delta u} \qquad (6)$$

Define vectors of past/known data to be

$$\mathbf{Y}_p = \begin{bmatrix} y_{-n} \\ \vdots \\ y_0 \end{bmatrix}; \quad \mathbf{\Delta U}_p = \begin{bmatrix} \Delta u_{-n} \\ \vdots \\ \Delta u_{-1} \end{bmatrix} \qquad (7)$$

Define the infinite horizon performance index to be

$$J = \sum_{i=1}^{\infty} y_i^2 + \lambda \Delta u_{i-1}^2 = \mathbf{y}^T\mathbf{y} + \lambda \mathbf{\Delta u}\mathbf{\Delta u} \qquad (8)$$

Finally simulating model (3) forward in time its is easy to derive the following

$$A(z)y(z) = b(z)\Delta u(z) + p(z) \qquad (9)$$

where $y(z)$, $\Delta u(z)$ contain predicted future values only, i.e.

$$y(z) = \sum_{i=1}^{\infty} y_i z^{-i}; \quad \Delta u(z) = \sum_{i=0}^{\infty} \Delta u_i z^{-i} \qquad (10)$$

and $p(z) = [1, z^{-1}, \dots, z^{-n}][H_A \mathbf{Y}_p - H_b \mathbf{\Delta U}_p]$. It now follows that the z-transform of the error, $e(z) = r(z) - y(z)$, where $r(z)$ is the z-transform of the setpoint trajectory, is given by

$$e(z) = \frac{r}{1 - z^{-1}} - y(z) = \frac{q(z) - b(z)\Delta u(z)}{A(z)} \qquad (11)$$

$$q(z) = a(z)r - p(z)$$

Note for clarity of presentation it has been assumed in eqn.(11) that all future values of the set-point trajectory are the same and equal to r.

Hereafter for ease of notation we will drop the argument of $.(z)$.

2.1 Infinite horizon predictive control (IHPC), [13]

This algorithm finds necessary and sufficient conditions for the stability of $\mathbf{y}$, $\mathbf{\Delta u}$ that is the stability of y, Δu of eqn.(9). Separating A, b into parts comprising unstable roots A^+, b^+ and stable roots a^-, b^- eqn.(11) becomes

$$A^+ a^- e = q - b^+ b^- \Delta u \qquad (12)$$

The class of stable solutions e, Δu to (12) is given by

$$\Delta u = \frac{A^+ c + \psi_p}{b^-}; \quad e = \frac{-b^+ c + \phi_p}{a^-} \qquad (13)$$

where

$$q - A^+\phi = b^+\psi; \quad \phi \in \Pi; \quad \psi \in \Pi \qquad (14)$$

and Π is the set of polynomials of either finite order or with coefficients which converge to zero. It is easy to show

$$\phi = \phi_p - b^+ c \quad \text{and} \quad \psi = \psi_p + A^+ c; \qquad c \in \Pi \quad (15)$$

where the particular solutions ϕ_p, ψ_p can be taken to be

$$\begin{bmatrix} \boldsymbol{\phi}_p \\ \boldsymbol{\psi}_p \end{bmatrix} = [\Gamma'_{A^+}, \Gamma'_{b^+}]^{-1} \begin{bmatrix} \mathbf{q} \\ O \end{bmatrix} = \left[\begin{array}{c|c} P_1 & P_3 \\ \hline P_2 & P_4 \end{array}\right] \left[\begin{array}{c} \mathbf{q} \\ \hline O \end{array}\right] \qquad (16)$$

where $\phi_p = [1, z^{-1}, \dots]\boldsymbol{\phi}_p$, $\psi_p = [1, z^{-1}, \dots]\boldsymbol{\psi}_p$ and $P_i, i = 1, 2, 3, 4$ are matrices conformal to the dimensions of $\boldsymbol{\phi}_p$, $\boldsymbol{\psi}_p$, $\mathbf{q}$. In IHPC $c(z)$ is taken to have n_c degrees of freedom, i.e.

$$c(z) = c_0 + c_1 z^{-1} + \dots c_{n_c - 1} z^{-n_c + 1} \qquad (17)$$

Algorithm 2.1 *The IHPC algorithm is defined by minimising the performance index (8) w.r.t. to c where y, Δu are given in (13). Of the optimising c the first element is used to compute Δu_0 and the computation is repeated at the next sampling instant.*

Remark 2.1 *The predictions deployed in IHPC are necessary and sufficient for stability and therefore release as much of the degrees of freedom as possible for meeting constraints and minimising the cost.*

2.2 Constrained LQR (CLQR) [14]

This algorithm was originally presented only in the state-space framwork. Here, for ease of comparison/discussion we present the algorithm in its polynomial form. This algorithm assumes knowledge of the LQ optimal as a start point; define the LQ optimal predictions minimising (8) to be

$$\Delta u = \frac{m}{s}; \quad e = \frac{k}{s} \tag{18}$$

where s is the optimal closed-loop pole polynomial and, for simplicity, m, k can be found as linear functions of q (eqn.11), that is nth order solutions of

$$Ak + bm = sq; \tag{19}$$

It is easy to solve for P_5, P_6 such that

$$\mathbf{k} = P_5\mathbf{q}, \quad \mathbf{m} = P_6\mathbf{q} \tag{20}$$

The predictions deployed for minimising J subject to constraints are then given as

$$\Delta u = \frac{m}{s} + \frac{Ac}{s}; \quad e = \frac{k}{s} - \frac{bc}{s}. \tag{21}$$

where the degrees of freedom are in c which can be chosen as in eqn.(17).

Algorithm 2.2 *The CLQR algorithm is defined by minimising the performance index (8) w.r.t. to c where y, Δu are given in (21). Of the optimising c the first element is used to compute Δu_0 and the computation is repeated at the next sampling instant.*

Remark 2.2 *The predictions deployed in CLQR contain the LQ optimal hence, in the case that constraints are inactive, the optimum c is zero.*

3 Optimal Constrained predictive control (OCPC)

The weakness of IHPC is that it does not allow the recovery of the LQ optimal unless a large number of degrees of freedom are deployed (though in practice it comes very close with only a few d.o.f.). On the other hand the weakness of CLQR is that the predictions deployed for handling constraints (eqn. 21) are sufficient for stability but not necessary and hence may run into problems when constraints are very tight. Here we propose to combine the best features of the two approaches, that is to develop an algorithm which can recover the LQ optimal but also uses necessary and sufficient conditions for stable predictions.

To help clarify the original two algorithms, consider the two components in the prediction equations. In each case the predictions are given as

$$\Delta u = \Delta u^{(1)} + \Delta u^{(2)}; \quad y = y^{(1)} + y^{(2)} \tag{22}$$

where $\Delta u^{(1)}$, $y^{(1)}$ are particular solutions and $\Delta u^{(2)}$, $y^{(2)}$ contain the degrees of freedom.

IHPC		CLQR	
$\Delta u^{(1)}$	$y^{(1)}$	$\Delta u^{(1)}$	$y^{(1)}$
$\frac{\psi_p}{b^-}$	$\frac{\phi_p}{A^-}$	$\frac{m}{s}$	$\frac{k}{s}$

Table 1. Particular solutions for IHPC and CLQR predictions

IHPC		CLQR	
$\Delta u^{(2)}$	$y^{(2)}$	$\Delta u^{(2)}$	$y^{(2)}$
$\frac{A^+c}{b^-}$	$\frac{-b^+c}{A^-}$	$\frac{Ac}{s}$	$\frac{-bc}{s}$

Table 2. Degrees of freedom in IHPC and CLQR predictions

It seems clear that the advantage of CLQR is in its choice of particualr solution $\Delta u^{(1)}$, $y^{(1)}$ whereas the advantage of IHPC is in its choice of the perturbation term $\Delta u^{(2)}$, $y^{(2)}$. Hence a logical approach would be to combine these two !

OCPC			
$\Delta u^{(1)}$	$y^{(1)}$	$\Delta u^{(2)}$	$y^{(2)}$
$\frac{m}{s}$	$\frac{k}{s}$	$\frac{A^+c}{b^-}$	$\frac{-b^+c}{A^-}$

Table 3. Predictions in OCPC

Remark 3.1 *Let the predictions be given as*

$$\Delta u = \frac{m}{s} + \frac{A^+c}{b^-}; \quad e = \frac{k}{s} + \frac{-b^+c}{A^-} \tag{23}$$

where m, k, s are as defined in (18). Then the perturbation term is based on necessary and sufficient conditions and the base term is the LQ optimal. Hence if *these predictions are used for minimising J (8) then in the case that constraints are inactive, the optimal c is zero, but in the case that constraints are active the perturbations are based on necessary and sufficient conditions for stable input/output pairs and maximise the control space for avoiding constraint violations. $c(z)$ can be chosen as in eqn.(17).*

Algorithm 3.1 *The OCPC algorithm is defined by minimising the performance index (8) w.r.t. to c where y, Δu are given by eqn.(22) with the $\Delta u^{(1)}$, $y^{(1)}$, $\Delta u^{(2)}$, $y^{(2)}$ of table 3. Of the optimising c the first element is used to compute Δu_0 and the computation is repeated at the next sampling instant.*

Remark 3.2 *The above algorithm still retains the LQ optimal as its particular solution. If constraints are very difficult to meet, even this assumption should be avoided and one should revert to IHPC. However, if constraints can be satisified easily, then OCPC provides the best mechanism for optimising performance as, in some cases, IHPC cannot find a solution close to the LQ optimal.*

4 Examples

In this section the results of the paper are illustrated by way of a number of numerical examples.

Three types of comparison will be performed.

- Case 1: simulations with constraints inactive.
- Case 2: simulations with constraints just active.
- Case 3: simulations with constraints difficult to satisfy.

Clearly for case 1 we would expect OCPC and CLQR to outperfrom IHPC. If constraints are active it is not clear which algorithm will do best and if constraints are almost impossible to meet, we would expect IHPC to retain feasibility longer than both OCPC and CLQR as both its particular solution and degrees of freedom use necessary and sufficient conditions. OCPC on the other hand uses the LQ optimal for its particular solution.

Let the constraints be given as

$$\begin{array}{c} \underline{\Delta u} \le \Delta u_i \le \overline{\Delta u} \\ \underline{u} \le u_i \le \overline{u} \end{array} \qquad (24)$$

For each example a simulation for a unit set-point change will be tested with the limits first set very large, then they will be reduced so that they become active. Finally as they are reduced further the algorithms will begin to go infeasible (at which point the stability guarantee is lost). The runtime costs

$$J_{run} = \sum_{i=0}^{runtime} y_i^2 + \lambda \Delta u_{i-1}^2 \qquad (25)$$

will be tabulated for all the above cases.

4.1 Example 1

The model is given as

$$\begin{aligned} a(z) &= 1 - 1.2z^{-1} + 0.9z^{-2} + 0.162z^{-3} \\ b(z) &= 1 - 1z^{-1} - 0.56z^{-2} \end{aligned}$$

The limits by simulation run are given in table 4 and the runtime costs by simulation run are given in table 5 for $n_c = 2, \lambda = 1$.

	Input Constraints			
	$\underline{\Delta u}$	$\overline{\Delta u}$	$\underline{u}$	$\overline{u}$
Run 1	-1000	1000	-1000	1000
Run 2	-0.19	0.19	-0.5	0.5
Run 3	-0.15	0.15	-0.5	0.5
Run 4	-0.05	0.05	-0.2	0.2

Table 4. Values of limits by run for example 1

	Algorithm		
	IHPC	CLQR	OCPC
Run 1	0.6160	0.6114	0.6114
Run 2	0.9769	0.8454	0.8454
Run 3	1.5009	infeas	0.9937
Run 4	infeas	infeas	2.8441

Table 5. Values of J_{run} for example 1

It can be seen that the OCPC and CLQR are equivalent and both superior to IHPC for the case when the constraints are inactive. As the constraints become more stringent, the CLQR loses feasibility first and IHPC second. Here, for tight constraints, OCPC is the best of the three, remaining feasible for the tightest constraints. For completeness the simulation plots are given in Figure 1. for Run 1 (the LQ optimal) and in Figure 2 for Run 4 (IHPC only); the outputs, inputs, input increments and setpoint are all plotted on the same graph.

There are algorithms e.g. ([12], [11], [14]) for adjusting the objective function and/or the set point (and hence q) in the case of feasibility problems but obviously such algorithms are automatically suboptimal

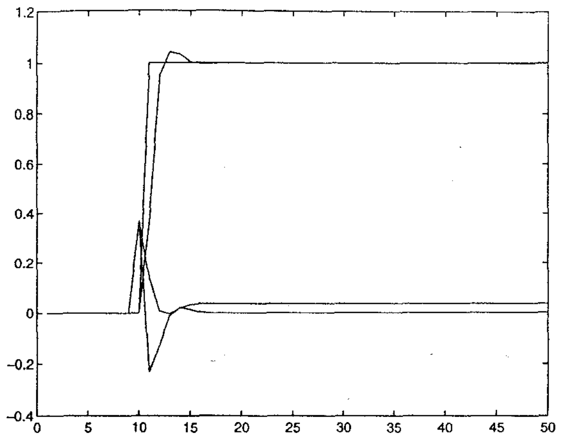

Figure 1. OCPC unconstrained optimal responses for example 1, Run 1.

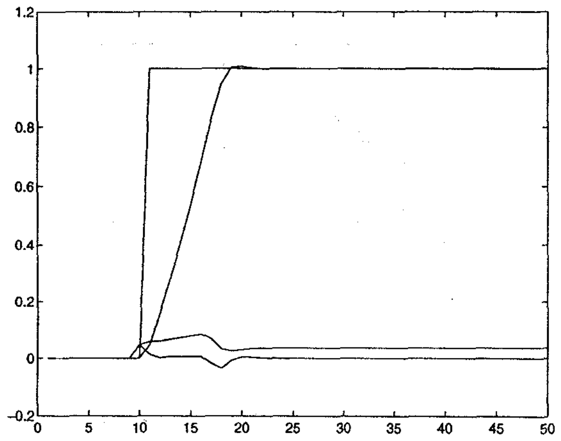

Figure 2. OCPC constrained optimal responses for example 1, Run 4

4.2 Example 2

The model is given as

$$
\begin{aligned}
a(z) &= 1 + 0.23z^{-1} - 0.95z^{-2} + 0.16z^{-3} \\
b(z) &= 1 - 0.08z^{-1} - 0.001z^{-2}
\end{aligned}
$$

with $\lambda = 1$; $n_c = 2$. The limits by simulation run are given in table 6 and the runtime costs by simulation run are given in table 7.

	Input Constraints			
	$\underline{\Delta u}$	$\overline{\Delta u}$	$\underline{u}$	$\overline{u}$
Run 1	-1000	1000	-1000	1000
Run 2	-0.25	0.25	-1	1
Run 3	-0.2	0.2	-1	1
Run 4	-0.15	0.15	-1	1

Table 6. Values of limits by run for example 2

	Algorithm		
	IHPC	CLQR	OCPC
Run 1	0.7693	0.7655	0.7655
Run 2	1.1545	1.1535	1.1535
Run 3	1.3746	infeas	1.3741
Run 4	1.7557	infeas	infeas

Table 7. Values of J_{run} for example 2

It can be seen that the OCPC and CLQR are equivalent and both superior to IHPC for the case when the constraints are inactive. As the constraints become more stringent, the CLQR loses feasibility, then OCPC next while IHPC stays feasible for the tightest constraints. The plots of the LQ optimal (figure 3) and for Run 4 IHPC (figure 4) are given below.

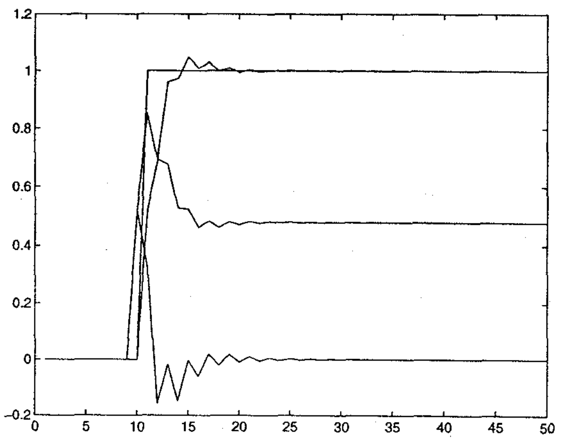

Figure 3. OCPC Unconstrained optimal responses for example 2, Run 1

Remark 4.1 *Preliminary studies indicate that, counter to intuition which suggests that IHPC will remain feasible for the tightest constraints and that OCPC should outperform CLQR, it is not obvious which algorithm remains feasible for the tightest constraints nor which gives the best performance in general. Early numerical test results show that it is highly example dependant, sometimes IHPC is best, sometimes OCPC and sometimes CLQR. The variable n_c (the number of d.o.f) also seems to play a significant role. A more formal analysis and comparison of the three methods forms the topic of further research.*

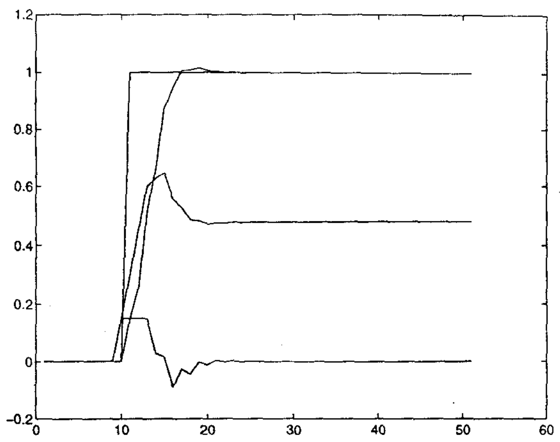

Figure 4. IHPC Constrained optimal responses for example 2, Run 4

4.3 Conclusion

The algorithms of [13], [14] approach the predictive control problem from different viewpoints where either LQ optimality or control freedom for meeting constraints are the most important consideration. Further development of a combination of the two philosophies [5] has been shown to yield an algorithm which can achieve LQ optimality when it is feasible but then maximises the remaining control freedom for constraint handling. This new algorithm is expected to outperform both earlier algorithms except in the case where constraints are very hard to meet in which case sometimes [13] should still offer the best approach. Early numerical studies suggest that all three algorithms have advantages and which to use is very example dependant.

Acknowledgements: The authors wish to thank EPSRC for financial support.

References

[1] D.W. Clarke and R. Scattolini, *Constrained receding horizon predictive control* Proc. IEE, Pt. D, Vol.138, No.4, pp347-354, 1991

[2] J. R. Gossner, B. Kouvaritakis and J.A. Rossiter 1995, *Cautious stable predictive control: a guaranteed stable predictive control algorithm with low input activity and good robustness*, To appear IJC

[3] S.S. Keerthi and E.G. Gilbert, 1988, *Optimal Infinite horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving horizon approximations*, Journal Optimisation Theory and Applications, 57, 2, pp265-293

[4] B. Kouvaritakis, J.A. Rossiter and A.O.T. Chang, 1992, *Stable Generalized predictive control: an algorithm with guarenteed stability* Proc. IEE, Vol.139, No.4, pp349-262

[5] B. Kouvaritakis, J.A. Rossiter and M. Cannon, 1996, , Internal Report No. , Dept. Engineering Science, Oxford University.

[6] E. Mosca, and J. Zhang, 1992, *Stable redesign of predictive control* Automatica, Vol.28, No. 6, pp1229-1233

[7] J.B. Rawlings and K.R. Muske, 1993, *The stability of constrained receding horizon control* IEEE Trans. AC, Vol.38, No.10, pp1512-1516

[8] J.A.Rossiter and B.Kouvaritakis, 1993, *Constrained generalized predictive control*, Proc IEE Pt.D, Vol.140, No.4, pp243-254

[9] J.A. Rossiter and B. Kouvaritakis, 1994, *Numerical robustness and efficiency of generalised predictive control algorithms with guaranted stability*, Proc. IEE Pt. D., Vol. 141, No. 3, pp154-162

[10] J.A. Rossiter, 1994, *GPC controllers with guaranteed stability and mean-level control of unstable plant*, 33rd IEEE Conference on Decision and Control, Orlando, USA, pp3579-3580

[11] J.A.Rossiter, B.Kouvaritakis and J.R. Gossner, 1995, *Feasibility and stability results for constrained stable generalised predictive control* Automatica, 31,6, pp863-877

[12] J.A.Rossiter, J.R. Gossner and B.Kouvaritakis, 1996, *Guaranteeing feasibility in constrained stable generalised predictive control*, Proc. IEE Pt.D, 143, 5, pp463-469

[13] J.A.Rossiter, J.R. Gossner and B.Kouvaritakis, 1996, *Infinite horizon stable predictive control*, IEEE Trans. AC, Vol. 41, No.10, pp1522-1527

[14] P.O.M.Scokaert and J. B. Rawlings, *Infinite horizon linear quadratic control with constraints*, Proceedings IFAC96, San Fransisco,. 1996, Vol. M, pp109-114