A FAST SUBOPTIMAL MULTI PARAMETRIC QUADRATIC PROGRAMMING ALGORITHM FOR PREDICTIVE CONTROL

J.A. Rossiter *

* *University of Sheffield, Dept. Automatic Control and Systems Engineering, S1 3JD, UK*

Abstract: Multi parameteric quadratic programming gives a full offline solution to a time varying quadratic programming (QP) problem arising during constrained predictive control. However, coding and implementation of this solution may be more burdensome than simply solving the original QP. This paper proposes an algorithm, which by accepting a small amount of suboptimality in the predicted control trajectories, achieves a large decrease in both the online computation and data storage requirements.

Keywords: Efficient predictive control, constraints, quadratic programming

1. INTRODUCTION

Obstacles to more widespread use of predictive control (Mayne *et al.*, 2000) are the computational demand and the lack of transparency due to the use of an online optimisation. Typical MPC algorithms are based on the minimisation of a quadratic performance index subject to linear constraints, that is a quadratic programming (QP) problem. In many cases (Rossiter *et al.*, 2002), the control law must have a simpler implementation: (i) coding limitations may make a QP optimiser inpractical and (ii) computational time restricts implementation on fast systems.

A recent contribution (Bemporad *et al.*, 2002), denoted multi parametric quadratic programming (MPQP), solves offline, all possible QP problems that can arise on line and shows that the solution is piecewise affine, that is, within computable regions of the state space, the the input trajectory has a known affine dependence on the current state. Hence solving QP is equivalent to set membership tests; if the state is inside region 'k', then use the associated law. A major strength of MPQP is visibility; implementation is akin to gain scheduling in that the control law is modified according to the current state. Unfortunately, MPQP may not be efficient due to the potentially large number of regions/laws to be stored and hence it could be harder to implement than the original QP optimiser.

This paper proposes one means of reducing the data storage requirements and implementation time of MPQP by allowing a small degree of suboptimality. Other authors are also considering this problem (e.g.(Bemporad *et al.*, 2001; Grieder *et al.*, 2003; Tondel *et al.*, 2003)) but from a different perspective. The paper gives a brief overview of MPC and MPQP, shows how a suboptimal algo-

rithm can reduce complexity with little detriment to performance.

2. BACKGROUND

2.1 Modelling and MPC

This paper makes use of state-space models,

$$\mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k; \quad \mathbf{y}_k = C\mathbf{x}_k \tag{1}$$

where $\mathbf{x}, \mathbf{u}, \mathbf{y}$ are the state, input and output respectively. Associated to the model are constraints:

$$\underline{\mathbf{u}} \leq \mathbf{u}_k \leq \overline{\mathbf{u}}; \quad \underline{\mathbf{x}} \leq \mathbf{x}_k \leq \overline{\mathbf{x}} \tag{2}$$

Define the d.o.f. via a dummy variable $\mathbf{c}$:

$$\begin{array}{ll} \mathbf{u}_k = -K\mathbf{x}_k + \mathbf{c}_k; & k = 0, ..., n_c - 1 \\ \mathbf{u}_k = -K\mathbf{x}_k; & k \geq n_c \end{array} \tag{3}$$

where K is the optimal unconstrained feedback. Let $\mathbf{C}^T = [\mathbf{c}_0^T, \mathbf{c}_1^T, ..., \mathbf{c}_{n_c-1}^T]$. Then an MPC optimisation (Scokaert *et al.*, 1998; Rossiter *et al.*, 1998) of performance subject to constraint satisfaction can be written as:

$$\min_{\mathbf{C}} \quad J = \mathbf{C}^T S \mathbf{C} \quad \text{s.t.} \quad M_c\mathbf{C} + M_x\mathbf{x}_0 - \mathbf{v} \leq 0 \tag{4}$$

Details of matrices S, M_x, M_c and $\mathbf{v}$ are omitted for brevity. The first component of $\mathbf{C}$, that is $\mathbf{c}_0$ is used to formulate the control law of (3).

2.2 Multi parametric quadratic programming

Consider the minimisation (4). The optimising $\mathbf{C}$ depends upon $\mathbf{x}$. Define regions $\mathcal{S}_i, \quad i = 0, 1, ...$:

$$\mathcal{S}_i = \{\mathbf{x} : M_i\mathbf{x} - \mathbf{d}_i \leq 0\} \tag{5}$$

such that within each region the active set is the same and hence the $\mathbf{C}$ optimising (4) has a known affine dependence on $\mathbf{x}$. The following algorithm gives the optimal control:

Algorithm 2.1. Find i s.t. $\mathbf{x} \in \mathcal{S}_i$, then

$$\mathbf{x} \in \mathcal{S}_i \;\Rightarrow\; \mathbf{c} = \mathbf{e}_1^T(-\hat{K}_i\mathbf{x} + \mathbf{t}_i) \;\Rightarrow\; \mathbf{u} = -K_i\mathbf{x} + \mathbf{p}_i \tag{6}$$

where $\mathbf{e}_1^T = [I, 0, 0, ...]$, and $K_i,\ \mathbf{t}_i,\ \mathbf{p}_i$ have obvious definitions (Bemporad *et al.*, 2002).

If the number of sets $\mathcal{S}_i$ is small, then MPQP is very efficient. However, if the number of sets is large then the implied search and data storage requirements can become unmanageable.

2.3 ONEDOF algorithms

ONEDOF algorithms (Rossiter *et al.*, 2001) reduce the computational burden of constraint handling by: (i) allowing some suboptimality and (ii) making good use of offline information. In ONEDOF rather than minimising w.r.t. the control values $\mathbf{u}_k$, one uses a mixture of input trajectories associated to different control laws. For instance, with just two laws, the optimisation reduces to a single set membership test. The main weakness is possible restrictions to the feasible region as sometimes nonlinear predictions are needed in transients to enlarge feasible regions.

The simplest algorithm makes use of a law $\mathbf{u} = -K_f\mathbf{x}$ with a large MAS and also the optimal control law $\mathbf{u} = -K\mathbf{x}$. One then takes a linear mix of the predicted control trajectories arising from each law and minimises the cost J over a scalar mixing variable, subject to constraint satisfaction. In summary the procedure reduces to:

Algorithm 2.2. (**ONEDOF)**

(1) Define the predicted future state and input trajectories (subscripts o, f denote optimal and feasible predictions) and α is the d.o.f. .

$$\mathbf{X} = (1-\alpha)\mathbf{X}_o + \alpha\mathbf{X}_f; \quad \mathbf{U} = (1-\alpha)\mathbf{U}_o + \alpha\mathbf{U}_f \tag{7}$$

(2) Substituting predictions (7) into the constraints of (2) gives the inequalities $\mathbf{M}\alpha - \mathbf{d} \leq 0$.

(3) Minimising J w.r.t. α is equivalent to:

$$\min_{\alpha} \quad \alpha \quad \text{s.t.} \quad \begin{cases} \mathbf{M}\alpha - \mathbf{d} \leq 0 \\ 0 \leq \alpha \leq 1 \end{cases} \tag{8}$$

(4) Compute the current control action from

$$\mathbf{u}_k = \mathbf{e}_1^T[(1-\alpha)\mathbf{U}_o + \alpha\mathbf{U}_f]. \tag{9}$$

The minimisation (8) is trivial and hence efficient. Moreover, examples (Rossiter *et al.*, 2001) show that, where feasible, suboptimality is often negligible.

2.4 Summary

(1) MPQP suffers no loss in the size of the feasible region or optimality but may be inefficient.

(2) ONEDOF is very efficient, but may have a smaller feasible region and some suboptimality.

This paper combines the large feasible region of MPQP with the efficiency of ONEDOF.

3. IMPROVING THE EFFICIENCY OF MPQP

We can use inferences from control strategies associated to the MPQP feasibility boundary to gain a significant reduction in complexity and surprisingly little deterioration in performance.

3.1 MPQP and its region of attraction

The following lemmata are obvious and, due to space restrictions, stated without proof.

Lemma 3.1. The MPQP control law is defined and stabilising only in the region $\mathcal{S}_{max}$ where

$$\mathcal{S}_{max} = \bigcup_i \mathcal{S}_i = \{\mathbf{x} : M_{max}\mathbf{x} - \mathbf{d}_{max} \leq 0\} \quad (10)$$

Assume hereafter that $\mathcal{S}_{max}$ is in minimal form so that redundant constraints are removed. For a given pair K, n_c, $\mathcal{S}_{max}$ is the maximal volume feasible region.

Lemma 3.2. Each row of the linear inequalities describing $\mathcal{S}_{max}$ can be determined by forcing equality of a subset of the inequalities in $M_c\mathbf{C}+M_x\mathbf{x}-\mathbf{v} \leq 0$.

Theorem 3.1. If $\mathbf{x}$ lies on the facet of $\mathcal{S}_{max}$, then one can identify the associated region $\mathcal{S}_i$ and hence the underlying optimal control law far more efficiently than via a search over all the MPQP regions.

Proof: If $\mathbf{x}$ lies on a facet, then for some integer j:

$$\mathbf{e}_j^T[M_{max}\mathbf{x} - \mathbf{d}_{max}] = 0 \quad (11)$$

Finding j requires an equivalent computation of just one set membership test. One can now identify the region $\mathcal{S}_i$ in which $\mathbf{x}$ lies by doing a search over only those regions contributing to the ith facet. In general, the number of regions on a facet would be far fewer than the total number of regions. □

Hence an efficient procedure to find the optimal control law (for $\mathbf{x}$ on a facet) is:

Algorithm 3.1. **MPQP for boundary states**

(1) Identify the active facet from (11).

(2) From those regions which contribute to the jth facet, identify in which one $\mathbf{x}$ lies and implement the appropriate control law from (6).

Algorithm 3.1 is far more efficient that MPQP. First the number of facets will in general be less that the number of regions and moreover identifying the active facet requires only the computation of one inequality for each facet. Second, in general there will be only a few active sets on each facet, so the region search will be, relative to MPQP, simple.

3.2 Control law for states not on a facet

Here we extend algorithm 3.1 for interior points. Assume that $0 \in \mathcal{S}_{max}$. The following lemma can be used to establish the 'nearest' facet to a non-boundary state.

Lemma 3.3. Assume that $\mathbf{d}_{max}^T = [1, 1, ...]$ and the rows of M_{max} are m_j^T. Define P_j as the minimal volume polytopes containing the jth facet of $\mathcal{S}_{max}$ and the origin. Compute the values $\gamma_j = \mathbf{m}_j^T\mathbf{x}$ and compute j for which γ_j is a maximum. Then $\mathbf{x}$ lies in polytope P_j.

Corollary 3.1. Given $\mathbf{x} \in P_j$ and $\gamma_j = \mathbf{m}_j^T\mathbf{x}$ find the region $\mathcal{S}_i$ such that $\mathbf{x}/\gamma_j \in \mathcal{S}_i$. Then the control move

$$\mathbf{u} = -K_i\mathbf{x} + \gamma_j\mathbf{p}_i; \quad (12)$$

is the first move of a predicted control law with feasible and convergent predictions.

The control law implied by Corollary 3.1 requires only information about the boundary and therefore is far more efficient than MPQP: (i) only boundary regions and associated laws need be stored; (ii) set membership searchs are simplified to checking only those on a known facet. However, the cost of this simplification is suboptimality.

4. REDUCING SUBOPTIMALITY WITH ONEDOF

This section proposes a novel implementation of the ONEDOF algorithm to gain significant improvements in performance on algorithm 3.1 and control (12).

Algorithm 4.1. **ONEDOF:**

(1) Find the feasible $\mathbf{C}$ implied by Corollary (3.1). (γ_j times the $\mathbf{C}$ for $\mathbf{x}/\gamma_j$.)
(2) Solve the minimisation

$$\min_{\alpha} \quad \alpha \quad \text{s.t.} \quad \alpha M_c \mathbf{C} + M_x \mathbf{x} - \mathbf{d} \leq 0 \qquad (13)$$

(3) Implement the control law: $\mathbf{u} = -K\mathbf{x} + \mathbf{e}_1^T \alpha \mathbf{C}$.

This algorithm requires only a a trivial linear program, that is, in just one variable and therefore equivalent in complexity to a single set membership test. Despite this simplicity, one can often gain significantly by of way optimality compared to the control law of (12).

5. EXAMPLES

This section will illustrate two things. First the efficacy of control law (12) (denote as SCALING) and algorithm ONEDOF in giving near optimal performance and vast improvements in online efficiency.

5.1 Example 1

Take the discrete model and input and state limits

$$\mathbf{x}_{k+1} = \begin{bmatrix} 1 & 0.05 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0.0025 \\ 0.05 \end{bmatrix}; \quad y_k = [1 \;\; 0]\mathbf{x}_k$$

$$-1 \leq u_k \leq 1; \quad -0.5 \leq [0 \;\; 1]\mathbf{x}_k \leq 0.5 \qquad (14)$$

Compute the optimal control law with $Q = C^T C$, $R = 1$ as $K = [0.9653 \;\; 1.3655]$.

For $n_c = 2$ with state constraints and for $n_c = 6$ without state constraints, table 1 (and figures 1-3) gives the number of MPQP regions and facets. Comparing worst case scenarios, algorithm 4.1 gives a reduction in complexity over MPQP of a factor of at least 5. Set membership search complexity is at most 18 as opposed to 85 regions for $n_c = 2$ and at most 2 compared to 385 for $n_c = 6$! In both cases there are far fewer facets than regions; moreover locating the active facet from Lemma (3.3) is trivial.

Closed-loop simulations were performed for initial states near the feasibility boundary. The runtime cost was computed, based on J, for each algorithm and is summarised in table 2. Clearly ONEDOF has negligible suboptimality. Closed-loop simulations are given in figure 4 ((a,b) for one initial point and (c,d) for the other; the solid line for MPQP, dotted line for algorithm 4.1 and dashed line for the algorithm of eqn.(12)) and these show only small differences in behaviour.

MPQP	ONEDOF/SCALING	
Regions	Facets	Maximum number regions per facet
$n_c = 2$ with state constraints		
85	30	18
$n_c = 6$ without state constraints		
385	62	2

Table 1. Maximum search complexity

Initial state	MPQP	ONEDOF	SCALING
$n_c = 2$ with state constraints			
$\begin{bmatrix} -1.02 \\ 0.465 \end{bmatrix}$	17.91	18.08	19.87
$\begin{bmatrix} 1.1 \\ -0.13 \end{bmatrix}$	28.57	28.53	30.16
$n_c = 6$ without state constraints			
$\begin{bmatrix} -3.8 \\ 2.47 \end{bmatrix}$	200.98	202.29	206.30
$\begin{bmatrix} 3.2 \\ -1 \end{bmatrix}$	184.49	184.49	186.15

Table 2. Run time costs for example 1

5.2 Example 2

This model represents a simplified boiler for electricity generation:

$$A = \begin{bmatrix} 0.98 & 0 & 0.019 \\ 0.075 & 0.607 & 0.001 \\ 0 & 0 & 0.607 \end{bmatrix}; \quad B = \begin{bmatrix} 0.005 & -0.021 \\ 0.39 & 00.05 \end{bmatrix}$$

$$C = \begin{bmatrix} [1.69 & 13.22 & 0 \\ 0.84 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 \\ -1 \end{bmatrix} \leq \mathbf{u} \leq \begin{bmatrix} 1 \\ 1 \end{bmatrix}; \quad Q = C^T C, \quad R = I \qquad (15)$$

A summary of the number of regions, facets and maximum regions per facet is given in table 3 for $n_c = 3$. Simulations are performed for initial points close to the boundary of $\mathcal{S}_{max}$ and the closed-loop run time costs are given in table 4. Once again it is evident that the ONEDOF algorithm has given only a small deterioration in performance relative to the huge increase in efficiency - search complexity down from 677 regions to 5!

MPQP	ONEDOF/SCALING	
Regions	Facets	Maximum number regions per facet
677	126	5

Table 3. Search complexity.

Initial state	MPQP	ONEDOF	SCALING
$\begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}$	56.51	68.61	149.6
$\begin{bmatrix} 0.6 \\ -0.6 \\ 0.6 \end{bmatrix}$	32.22	32.66	37.07
$\begin{bmatrix} 0.65 \\ -0.65 \\ -0.65 \end{bmatrix}$	32.50	32.61	36.68
$\begin{bmatrix} 0.55 \\ 0.55 \\ -0.55 \end{bmatrix}$	63.51	73.05	151.30

Table 4. Run time costs for example 2.

6. CONCLUSION

This paper has shown how the MPQP algorithm can be combined with the ONEDOF algorithm to obtain an algorithm with the same size of feasibility region, near optimal performance and yet a comparatively small computational load and far smaller data storage requirement. This algorithm has now been tested (and shown to be effective) on a large number of systems and the results are reported in (Rossiter *et al.*, 2004).

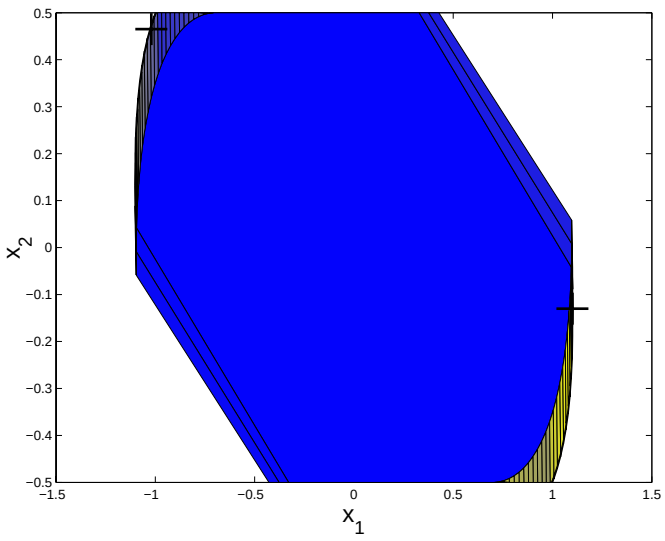

Fig. 1. Regions for example 1 with $n_c = 2$

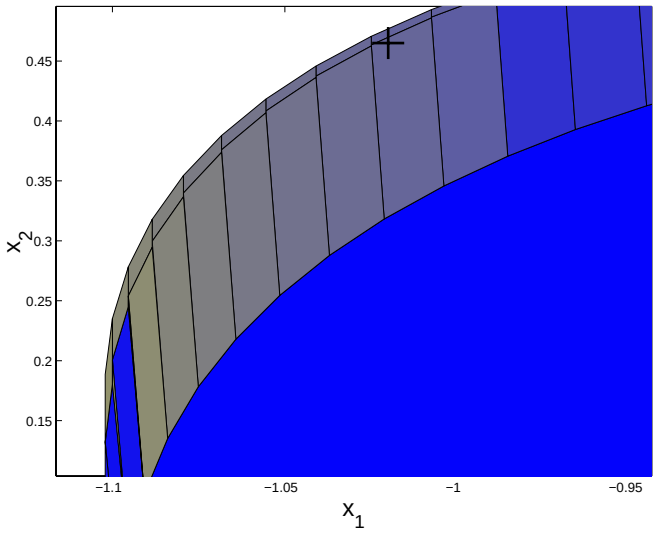

Fig. 2. Blown up view of regions for example 1 with $n_c = 2$

REFERENCES

Bemporad, A., M. Morari, V. Dua, E.N. Pistokopoulos, The explicit linear quadractic regulator for constrained systems, *Automatica*, 2002, 38, 1, pp 3-20

Bemporad, A. F. Borrelli and M. Morari, The explicit solution of constrained LP-based receding horizon control, Proc. European Control Conference, 2001.

Grieder, P., F. Borrelli, F. Torrisi and M. Morari, Computation of the constrained infinite time linear quadratic regulator, ACC, 2003

Mayne, D.Q., J.B. Rawlings, C.V. Rao and P.O.M. Scokaert, Constrained model predictive control: stability and optimality, *Automatica*, 2000, 36, pp789-814

Rossiter, J.A., M.J. Rice and B.Kouvaritakis, A numerically robust state-space approach to stable predictive control strategies, *Automatica*, 1998, 38, 1, 65-73

Rossiter, J.A., P.W. Neal and L. Yao, Applying predictive control to fossil fired power station, Proceedings Inst. MC, 2002, 24(3), 177-194.

Rossiter, J.A., B. Kouvaritakis and M. Cannon, Computationally efficient algorithms for constraint handling with guaranteed stability and near optimality, IJC, 2001, 74, 17, 1678-1689

Rossiter, J.A. and P. Grieder, Using Interpolation to Simplify Explicit Model Predictive Control, to appear ACC04

Scokaert, P.O.M. and J. B. Rawlings, Constrained linear quadratic regulation, *IEEE Trans AC*, 1998, 43, 8, pp1163-1168

Tøndel, P., T.A. Johansen, and A. Bemporad. Evaluation of piecewise affine control via binary search tree. *Automatica*, 39(5):945–950, 2003.

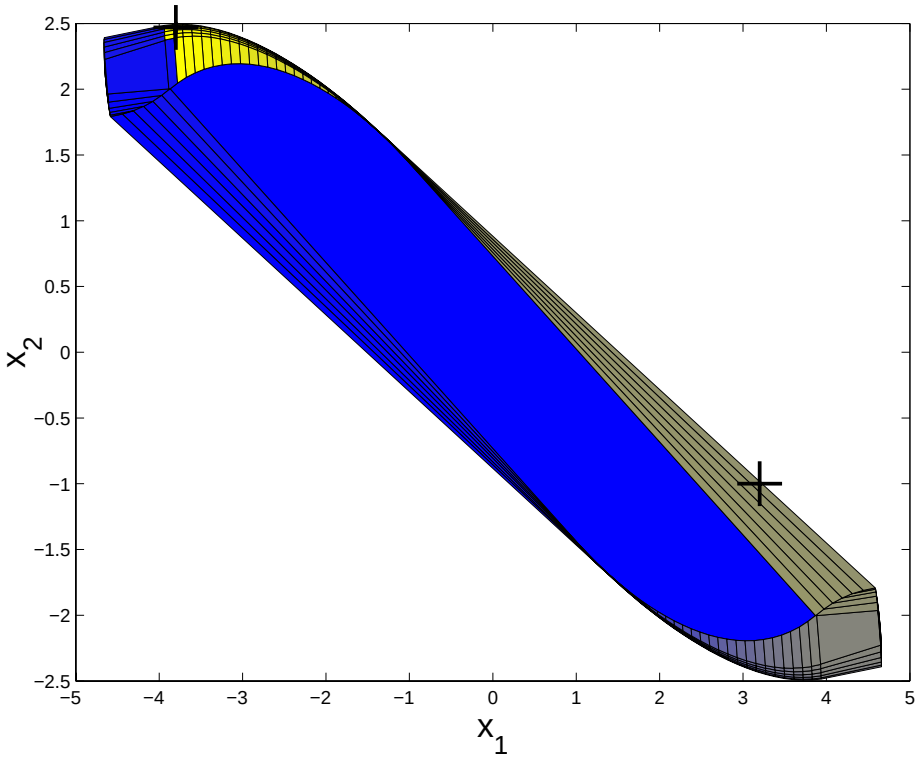

Fig. 3. Regions for example 1 with $n_c = 2$ and no state constraints

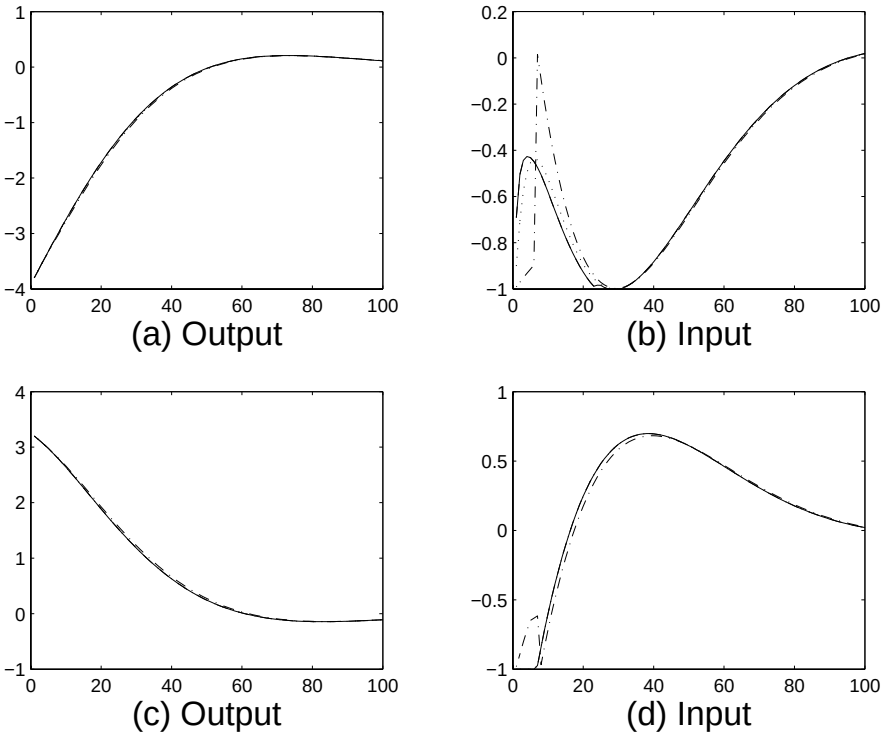

Fig. 4. Simulations for example 1 with $n_c = 6$ and no state constraints