Available online at www.sciencedirect.com
SCIENCE DIRECT
automatica
Automatica 41 (2005) 1395–1403
www.elsevier.com/locate/automatica

Brief paper

More efficient predictive control☆

Lars Imsland, Nadav Bar, Bjarne A. Foss*

Department of Engineering Cybernetics, Norwegian University of Science and Technology, N-7491 Trondheim, Norway

Received 11 February 2004; received in revised form 14 September 2004; accepted 8 March 2005
Available online 24 May 2005

Abstract

An approach for constrained predictive control of linear systems (or uncertain systems described by polytopic uncertainty models) is presented. The approach consists of (in general non-convex, but often convex) offline optimization, and very efficient online optimization. Two examples, one being a laboratory experiment, compare the approach to existing approaches, revealing both advantages and disadvantages.

Keywords: Model predictive control; Constrained systems; Invariant sets; Computational efficiency

1. Introduction

Model predictive control (MPC) has gained significant popularity in industry as a tool to optimize system performance while handling constraints explicitly. However, limitations on computational efficiency have restricted the application range. This has lead to a substantial effort to obtain predictive constraint-handling control strategies that have more attractive online computational properties than quadratic programming typically used in traditional linear MPC. In most cases, this is obtained by performing some calculations offline.

Examples of such schemes are explicit MPC (Bemporad, Morari, Dua, & Pistikopoulos, 2002), and efficient robust predictive control (ERPC) (Kouvaritakis, Rossiter, & Schuurmans, 2000). Explicit MPC computes offline via multi-parametric programming an explicit solution to the finite horizon MPC problem. In ERPC, the offline part uses the degrees of freedom on the control horizon to find large invariant ellipsoids for an augmented system, while the online part efficiently minimizes control deviation from unconstrained optimal LQ-control, subject to augmented state membership of the precomputed ellipsoid. ERPC also handles uncertainty; if the ellipsoids are robustly invariant, then the online optimization does not have to consider propagating uncertainty over the horizon, which dramatically reduces computation effort compared to min–max approaches.

Herein, we first present a generalization of the offline problem of ERPC, thus we will denote the new approach generalized ERPC, GERPC. Through this generalization, it is possible to obtain significantly larger invariant ellipsoids. Using the information obtained by solving the offline problem, we specify two online optimization problems, the first being a direct counterpart of the one in Kouvaritakis et al. (2000). The second is a new online problem that reduces sub-optimality at limited additional computational cost. Furthermore, through two examples, one being a laboratory experiment, we compare the merits of GERPC with ERPC and explicit MPC.

2. Model class and control objective

Consider discrete-time linear state-space models subject to input and state constraints

$$x_{k+1} = Ax_k + Bu_k \tag{1a}$$

$$\text{subject to } -\overline{u} < u_k < \overline{u}, \quad x_k \in X, \tag{1b}$$

☆ This paper was not presented in any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Faryar Jabbari under the direction of Editor Roberto Tempo.

* Corresponding author.
E-mail addresses: lars.imsland@itk.ntnu.no (L. Imsland), barc@itk.ntnu.no (N. Bar), bjarne.foss@itk.ntnu.no (B.A. Foss).

0005-1098/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2005.03.010

where the inequalities should be interpreted componentwise. The state and input dimensions are $x \in \mathbb{R}^{n_x}$ and $u \in \mathbb{R}^{n_u}$, and the origin is an equilibrium, $0 < \overline{u}$, and $X \ni 0$ represents state constraints. A strong point of the results herein is that they also hold for polytopic uncertainty models (Boyd, El Ghaoui, Feron, & Balakrishnan, 1994), in the same way as in Kouvaritakis et al. (2000). However, for brevity and simplicity of presentation this is not elaborated on. For the same reason, most of the paper assumes only input constraints.

We will assume that the pair (A, B) is stabilizable. The control objective will be to minimize (while satisfying constraints) the infinite horizon linear quadratic (LQ) cost function,

$$J_{\mathrm{LQ}} = \sum_{i=0}^{\infty} x_{k+i+1}^{\mathrm{T}} Q x_{k+i+1} + u_{k+i}^{\mathrm{T}} R u_{k+i}, \tag{2}$$

where Q and R are positive semi-definite matrices and x_{k+i+1} and u_{k+i} denote predicted values of states and control inputs (subscript k denotes current time).

In ERPC/GERPC, the system is assumed to be prestabilized by a feedback controller K, optimal with respect to (2) in the unconstrained case. The degrees of freedom are expressed as the perturbation, c_k, away from this control. Thus the future (predicted) control input is

$$u_i = \begin{cases} -Kx_i + c_i, & i = k, \ldots, k+n_c-1, \\ -Kx_i, & i \geqslant k+n_c. \end{cases} \tag{3}$$

As a consequence, the online optimization is carried out in terms of the new free variables c_i. Beyond the control horizon n_c, we can set $c_i = 0$ assuming the optimal LQ control is feasible onwards. The system equation for (1) with (3) is

$$x_{k+1} = \Phi x_k + B c_k, \tag{4}$$

where $\Phi = A - BK$.

3. Offline: augmenting the state space for enlarging the region of attraction

The unconstrained LQ controller K will typically have a rather small region where it does not hit the constraints (and hence stability is guaranteed). The objective of this section is to enlarge this region of attraction by augmenting the state space, generalizing a similar approach in Kouvaritakis et al. (2000) (see below). Denoting this augmented variable $z = (x, f)$, the augmenting variable $f \in \mathbb{R}^p$ is imposed with the following dynamics:

$$f_{k+1} = F x_k + G f_k. \tag{5}$$

Let c_k be computed by $c_k = D f_k$. The overall dynamics is then described by

$$z_{k+1} = \Psi z_k, \quad \Psi = \begin{pmatrix} \Phi & BD \\ F & G \end{pmatrix}. \tag{6}$$

Note that in the special case of $D = I$, $F = 0$ and $G = 0$ we recover the original LQ controller. For $F = 0$, $G = M$ where M has the "time recession" structure

$$M = \begin{pmatrix} 0 & I & 0 & \cdots & 0 \\ 0 & 0 & I & \cdots & 0 \\ & & \ddots & & \\ 0 & \cdots & 0 & 0 & I \\ 0 & \cdots & 0 & 0 & 0 \end{pmatrix}, \tag{7}$$

and $D = [I, 0, \ldots, 0]$ we recover the offline problem of ERPC (Kouvaritakis et al., 2000). In this case, the variable f is interpreted as the future c_i's, i.e. $f_k^{\mathrm{T}} = (c_k^{\mathrm{T}}, c_{k+1}^{\mathrm{T}}, \ldots, c_{k+n_c-1}^{\mathrm{T}})$ thus the dimension is $p = n_u n_c$. This predictive control interpretation is not so straightforward in the general case of (5), but a connection can still be made. We obtain (in general) an infinite control horizon, since for a given x_k and f_k, the future c_{k+i}'s are given as $c_{k+i} = D f_{k+i}$, $i = 0, 1, \ldots$ where $f_{k+i+1} = F x_{k+i} + G f_{k+i}$. This is as opposed to the finite horizon given by the time recession matrix (7). However, the GERPC "control degrees of freedom" (f_k) are still finite.

We want to find D, F and G that gives the largest possible region of attraction for the original system. We do this by looking for positive-definite matrices Q_z defining invariant ellipsoids $\mathscr{E}_z := \{z \mid z^{\mathrm{T}} Q_z^{-1} z \leqslant 1\}$, where the projection of the ellipsoid $\mathscr{E}_z$ onto the state space (see Kouvaritakis et al. (2000)), $\mathscr{E}_{xx} := \{x \mid x^{\mathrm{T}} (T Q_z T^{\mathrm{T}})^{-1} x \leqslant 1\}$, T defined by $x = Tz$, should be as large as possible. This is achieved with the following optimization problem:

$$\min_{Q_z, D, F, G} \ \ln \det (T Q_z T^{\mathrm{T}})^{-1} \tag{8a}$$

$$\text{subject to} \ \begin{pmatrix} Q_z & Q_z \Psi^{\mathrm{T}} \\ \Psi Q_z & Q_z \end{pmatrix} \geqslant 0, \tag{8b}$$

$$\bar{u}_j^2 - [-K_j^{\mathrm{T}} (De_j)^{\mathrm{T}}] Q_z [-K_j^{\mathrm{T}} (De_j)^{\mathrm{T}}]^{\mathrm{T}} \geqslant 0, \tag{8c}$$

where (8b) is a Lyapunov inequality guaranteeing invariance, and (8c) makes sure the input constraints are feasible inside the ellipsoid. With constant D, F and G (as in ERPC), this is a convex problem, for which efficient algorithms exist (Boyd et al., 1994). However, treating D, F and G as variables makes the problem non-convex (in general), since they are multiplied with Q_z. This increase in complexity can be rewarded by significantly larger ellipsoids. The non-convex optimization problem is similar to "static output feedback"-type problems, and BMI solvers for such problems can be used, as briefly outlined in Drageset, Imsland, and Foss (2003).

Recently (after this work was submitted), it has been shown (Cannon & Kouvaritakis, 2005) that in the case with $F = 0$ and $p \geqslant n_x$, the problem admits a convex LMI formulation. This is a huge advantage compared to solving a non-convex optimization problem.

State constraints can be added to the offline problem as long as they can be expressed as affine functions of Q_z (and possibly P_z, depending on the solver used). For brevity this is not included here, but the first example includes a state constraint.

4. Online problem: minimizing cost

Although the augmented part of the autonomous system specifies a valid (dynamic) controller, it is not optimizing. In this section, we look at how the online cost can be minimized (although sub-optimally) while retaining stability. We first review the application of the method suggested in Kouvaritakis et al. (2000), where the minimizing *f* is found from the feasible ones (inside the ellipsoid $\mathscr{E}_z$). The second is a new method and that searches for *f*'s that ensure feasibility at the next sample. In this section, we will assume that the structural constraint $F=0$ is imposed in (8). This means that $f_{k+1}=Gf_k$, and thus not dependent on the evolution of x_k. This assumption is mainly done for simplicity of presentation—similar (but slightly weaker) results hold in the case with non-zero F.

4.1. Feasibility now

As future control flexibility (the *f*) is part of the current augmented state, the ellipsoidal stability constraint can be applied at current time rather than at the end of the control horizon, as is common in other (e.g. QP-based) MPC approaches. This reduces online optimization to minimizing a performance index based on the future degrees of freedom in the input, J_f, subject to membership of the precomputed ellipsoid

$$\min_f J_f \quad \text{subject to} \quad z^{\mathrm{T}} Q_z^{-1} z \leqslant 1, \tag{9}$$

with $z=(x_k, f)$. Here, J_f penalizes the future control perturbations,

$$J_f = \sum_{i=0}^{\infty} c_{k+i}^{\mathrm{T}} W c_{k+i}, \tag{10}$$

where $W>0$ is given by $W = B^{\mathrm{T}} P B + R, \quad P = Q + K^{\mathrm{T}} R K + \Phi^{\mathrm{T}} P \Phi$. It can be shown (Kouvaritakis, Cannon, & Rossiter, 2002) that J_f and the LQ cost (2) differ by a bias term, thus minimizing the two indices is equivalent. The infinite sum (10) has a limit that is readily computed, $\sum_{i=0}^{\infty} c_{k+i}^{\mathrm{T}} W c_{k+i} = f^{\mathrm{T}} \Gamma f$ where Γ is the positive-definite solution of the discrete Lyapunov equation $G^{\mathrm{T}} \Gamma G - \Gamma = -D^{\mathrm{T}} W D$. This turns the online problem into minimizing a quadratic function subject to one ellipsoidal constraint, which can be solved extremely efficiently using a Newton–Raphson method to determine a single Lagrange multiplier (Kouvaritakis et al., 2002).

4.2. Feasibility at next sample

The ellipsoidal constraint in (9) leads to sub-optimality. According to Kouvaritakis et al. (2002), this sub-optimality can be reduced by allowing a line search outside the ellipsoid subject to feasibility at the next time instant (i.e., by "scaling" *f*). As this "scaling" can be performed explicitly, it only adds marginally to computational complexity. In view of the improved performance due to the "scaling" of *f*, it is tempting to look for other algorithms that search for *f* outside $\mathscr{E}_z$ in more general ways, subject to feasibility at the next time instant. The straightforward convex optimization problem that solves this, is

$$\min_f J_f \quad \text{subject to} \quad \begin{cases} z^{\mathrm{T}} \Psi^{\mathrm{T}} Q_z^{-1} \Psi z & \leqslant 1, \\ Df & \leqslant \bar{u} + K x_k, \\ -Df & \leqslant \bar{u} - K x_k, \end{cases} \tag{11}$$

where two constraints are added to ensure that the computed control satisfies input constraints. Denoting the optimal solution of (9) by $f^{\star}_{(9)}$ and the optimal solution of (11) by $f^{\star}_{(11)}$, it is clear that $J_f(f^{\star}_{(11)}) \leqslant J_f(f^{\star}_{(9)})$. In practice, the eigenvalues of Ψ will be strictly <1, in which case the inequality is strict (when not zero). For the rest of this section, this is assumed.

Apart from the two linear constraints, this optimization problem has the same structure as (9). It can be solved by a general code for nonlinear optimization, or a taylored QP-algorithm allowing ellipsoidal constraints (Cannon, Kouvaritakis, & Rossiter, 2001). If the ellipsoidal constraint can be replaced with a polytopic one, the optimization problem becomes a standard QP. Notably, this also allows for more than one step outside $\mathscr{E}_z$, generalizing to "Triple mode MPC" (Rossiter, Kouvaritakis, & Cannon, 2001). However, using a general optimization routine does not enjoy the same efficiency as using a Newton–Raphson method to solve (9). We therefore propose a more efficient method of solving (11), but where the solution might not always be the optimal. However, when initialized properly, it is always better than the solution of (9), and when possible suboptimality (compared to the optimal solution of (11)) occurs, at least one of the inputs is at its constraint.

Let f_{feas} be a feasible (not necessarily optimal) solution of (11). This can be found by solving (9), or from the previous timestep via (5). Such a f_{feas} exists at the first iteration, if we start inside $\mathscr{E}_{xx}$.

Algorithm 1.

(i) *Solve the optimization problem obtained by removing the linear constraints from* (11) *using e.g. a Newton–Raphson method. Call the solution* $f^{\star}$. *Obtain* f_{feas} (*e.g. by solving* (9)).

(ii) *Check if* $f^{\star}$ *satisfies the linear constraints of* (11). *If they do, then* $f^{\star}$ *is the optimal solution to* (11), *and we are finished. If they do not, then go to* (iii).

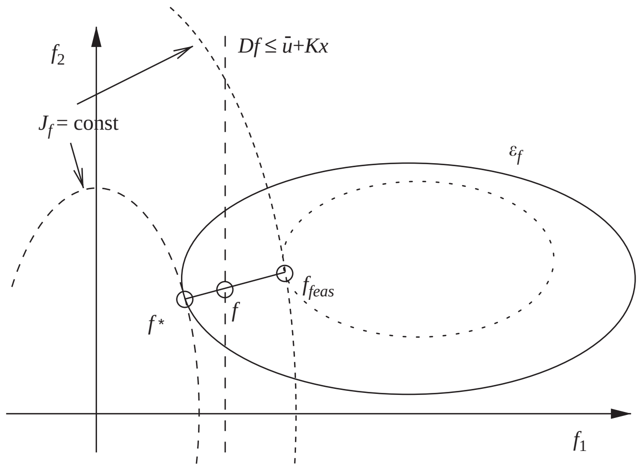

Fig. 1. Sketch of situation in proof of Theorem 1, for $n_u = 1$, $p = 2$, $D = [1, 0]$.

(iii) *Pick the f on the line between $f^\star$ and f_{feas} that is closest to $f^\star$ and satisfies the constraints.*

Theorem 1. *The f produced by Algorithm* 1 *guarantees feasibility at next time step, satisfies the control constraints, and when it is not equal to $f^\star$, at least one input is at a constraint. Furthermore $f^{\mathrm{T}}\Gamma f < f_{\text{feas}}^{\mathrm{T}}\Gamma f_{\text{feas}}$, thus f is always a better solution than f_{feas}.*

Proof. If $f^\star$ satisfies input constraints, then the first part of the theorem is obvious. The last part, $J_f(f^\star) < J_f(f_{\text{feas}})$ follows since f_{feas} is strictly contained in $\mathscr{E}_f$ (see below). Thus we focus on the f chosen in part (iii). The quadratic constraint in (11) defines an ellipsoid $\mathscr{E}_f$ in f-space. If $\mathscr{E}_f$ contains the origin, then $f^\star = 0$. If $\mathscr{E}_f$ does not contain the origin, $f^\star$ is the point on the border of $\mathscr{E}_f$ that is closest (in Γ-norm sense) to the origin. The point f_{feas} is strictly inside $\mathscr{E}_f$, since $\Psi^{\mathrm{T}} Q_z^{-1}\Psi < Q_z^{-1}$. Furthermore, since $\mathscr{E}_f$ is convex, the line between $f^\star$ and f_{feas} is inside $\mathscr{E}_f$, thus also the f chosen in (iii) is inside $\mathscr{E}_f$ and ensures feasibility at the next time step.

Since the input constraints are linear, and the line between $f^\star$ and f_{feas} crosses (since $f^\star$ is infeasible with respect to input constraints) a constraint at least once, the f chosen in (iii) will always be at a constraint.

We can express f as the convex combination $f = \alpha f_{\text{feas}} + (1-\alpha) f^\star$, for some $\alpha \in [0, 1)$. For $\alpha \in (0, 1)$, $J_f(f) < \max\{J_f(f_{\text{feas}}), J_f(f^\star)\} = J_f(f_{\text{feas}})$ since J_f is strictly convex, which establishes $J_f(f) < J_f(f_{\text{feas}})$ for $\alpha \in [0, 1)$. □

The situation in the proof is illustrated in Fig. 1. The "linesearch" for f in Algorithm 1 can be implemented explicitly, and hence very efficiently. If $n_u = 1$ and $f^\star$ exists but violates the input constraints, then it merely amounts to choosing the right input constraint, and f_{feas} is not needed at all.

If the $f^\star$ is feasible, the solution of Algorithm 1 will always be better (less sub-optimal) than the one found from "scaling". If the solution is at a constraint ($f^\star$ is unfeasible), then, depending on the geometry of the problem, both scaling and Algorithm 1 might be the best. Our conjecture (supported by the example in the next section) is that most often Algorithm 1 is best, since the linesearch is in a direction close to perpendicular[1] to the level sets of J_f.

4.3. Algorithm and stability

The overall approach can be summarized as follows:

Algorithm 2. *Offline*: *Solve the optimization problem* (8) *to find n_c, D, G and Q_z that give a suitable invariant set*.

Online: *Perform the minimization* (9) *or* (11) *to find f, implement $u_k = -Kx_k + c_k$, where $c_k = Df$, and move on to the next time step.*

Theorem 2 (*Closed-loop stability*). *If for system* (1) *there exist K, Q_z, p, D and G such that $x_k \in \mathscr{E}_{xx}$, the closed-loop application of Algorithm* 2 *is feasible and asymptotically stabilizing*.

The proof is similar to the proof of Theorem 4.1 in Kouvaritakis et al. (2000), and omitted for brevity.

5. Examples

In this section we will present two examples, where the performance of the suggested control algorithm will be compared to two control schemes that also are tailored towards efficient predictive control under constraints. We will first briefly review these.

Efficient robust predictive control. As mentioned above, the offline problem of ERPC as proposed in Kouvaritakis et al. (2000) is a special (convex) case of the GERPC offline problem obtained by choosing $F = 0$, $D = [I, 0, \ldots, 0]$ and $G = M$ where M is defined in (7). The online problem has the same structure as (9).

Explicit MPC. Explicit MPC refers to the explicit solution of the finite horizon constrained LQR problem as piecewise affine functions (controllers) defined on polytopic partitions of the state space (Bemporad et al., 2002). The offline problem of finding these controllers and the corresponding polytopes can be solved as a multi-parametric QP problem, see Tøndel, Johansen, and Bemporad (2003a) for a recent algorithm. The online problem consists of finding which polytope the present state is within. If done with a "brute force" approach, this might involve a significant number of arithmetic operations when the number of polytopes is large. However, by using smart data structures to store the

[1] A third option could be to perform the linesearch in the "steepest descent"-direction, but this is not elaborated on here.

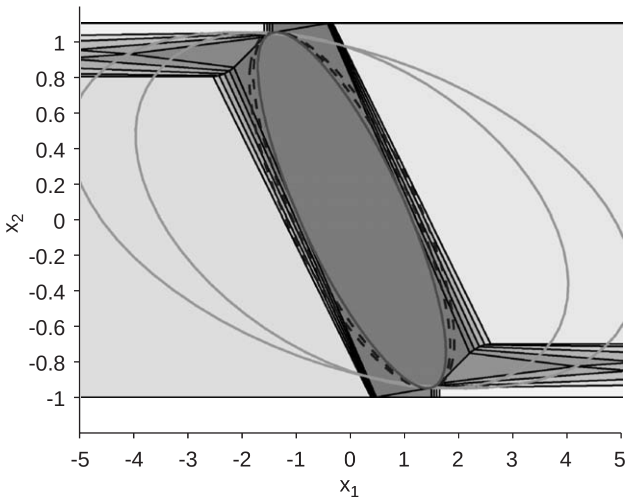

Fig. 2. The shaded areas (the polytopes) are the region where the explicit MPC controller with horizon 5 is defined. The dotted ellipsoids are the region of attraction for ERPC with horizon 2 and 5 and the outer ellipsoids the region of attraction for GERPC with horizon 2 and 5. The innermost ellipsoid is the largest invariant ellipsoid where the LQ controller is unconstrained.

problem data and exploiting the structure, it is possible to significantly reduce the online computational effort (Tøndel, Johansen, & Bemporad, 2003b). Nevertheless, storing all the polytopes and corresponding controllers might require a significant amount of memory.

5.1. Double integrator example

Consider the discretized double integrator model

$$A=\begin{pmatrix}1 & T_s\\ 0 & 1\end{pmatrix},\quad B=\begin{pmatrix}T_s^2\\ T_s\end{pmatrix},$$

where $T_s = 0.05$ s is the sampling time. There are constraints on input, $|u| \leqslant 1$, and on velocity, $|x_2| \leqslant 1$. The weighting matrices are chosen as $Q = \mathrm{diag}\{1, 0\}$ and $R = 1$. We will in the following compare the region of attraction, online (and offline) computational demand, online memory use and sub-optimality of GERPC, ERPC and explicit MPC.

5.1.1. Offline

Region of attraction. The obtained "regions of attractions" for the three approaches are shown in Fig. 2. The region for explicit MPC (stretching indefinitely in the x_1-direction) is significantly larger than for GERPC, which again is larger than for ERPC. For ERPC, increasing the horizon only marginally increased the region of attraction beyond the region of attraction for the LQ-controller. The fixed structure of G in the case of ERPC seemingly makes enlargement most pronounced in one particular direction, and in this case this comes in conflict with the state constraint. This example clearly shows the importance of flexibility in G since the GERPC ellipsoids are both rotated and "fattened" as compared to ERPC.

Offline computational complexity. The difference in offline computing time between a convex solver for ERPC and a non-convex solver for GERPC is as expected very large. As mentioned above, the explicit MPC solution is found using multi-parametric quadratic programming (mpQP), for which the computational complexity grows exponentially with problem size. In conclusion, it is probably safe to say that the ERPC offline problem has least complexity for a given system and horizon, while mpQP and a BMI solver for GERPC is harder to compare. In our experience the mpQP problem is about an order of magnitude faster than our BMI algorithm, for the same control horizon.[2]

5.1.2. Online

Memory usage. The demand on online memory is about the same for ERPC and GERPC (there is a slight difference due to the size of the Γ matrix). Therefore, we compare GERPC with explicit MPC. The memory requirements for horizons $n_c = 5$ can be summarized as follows:[3]

GERPC (reals)	"crude" explicit MPC (reals, controllers plus polytopes)	"smart" explicit MPC (reals plus integers)
56	$182 + 756$	$240 + 255$

As we can see, the difference is considerable in favor of GERPC, and will be even larger for longer horizons, as the size of the matrices grows quadratically while the number of polytopes grows exponentially. In the right-most column, we see that by using smart data structures and exploiting problem structure (Tøndel et al., 2003b), "smart" explicit MPC considerably reduces memory demand as compared to the straightforward implementation.

Online computational complexity. A count of "worst case" floating point operations is shown below for the given example (initial condition at $(-4, 0)$):

GERPC	"crude" explicit MPC	"smart" explicit MPC
12896	1008	44

For the Newton–Raphson method used by GERPC (for solving the optimization problem (9)), the worst case number of iterations were 13 in our implementation, with 992 floating point operations (counted with the `flops`-command of

[2] However, using a convex formulation Cannon and Kouvaritakis, 2005 for GERPC makes offline computation similar to ERPC, with about twice as many optimization variables.

[3] For GERPC, storing the results of some matrix calculations done offline will increase efficiency. This is not taken into account here.

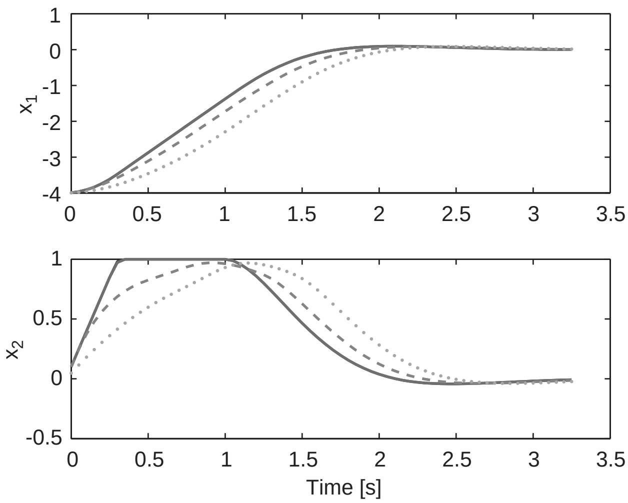

Fig. 3. The states: solid line is infinite horizon MPC and explicit MPC (they are barely distinguishable), dotted line is GERPC (with scaling), and dashed line is GERPC with Algorithm 1. Initial condition $(-4, 0)$.

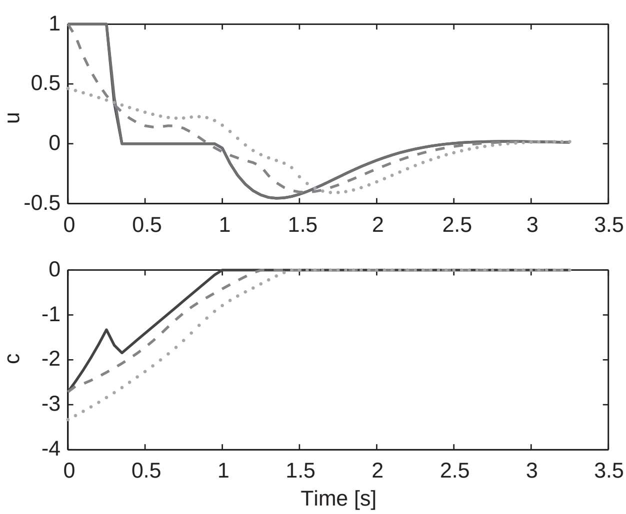

Fig. 4. Control input u and perturbation for unconstrained optimal control c: solid line is infinite horizon MPC and explicit MPC (they are barely distinguishable, c not shown for explicit MPC), dotted line is GERPC (with scaling), and dashed line is GERPC with Algorithm 1. Initial condition $(-4, 0)$.

Matlab 5) per iteration. Enhancing performance using either scaling (Kouvaritakis et al., 2002) or Algorithm 1, only adds minimally to this number—“scaling” added 129 floating point operations, and using Algorithm 1 adds even less. The “smart” explicit MPC controller is extremely efficient in this case. However, all these numbers are small, compared to a QP MPC controller, which (implemented using `quadprog` in Matlab) with horizon length 5 uses about 106 000 floating point operations.

Sub-optimality. Simulations of the system from initial condition $(-4, 0)$ with infinite horizon MPC, explicit MPC, GERPC and GERPC with scaling are shown in Figs. 3 and 4.

We would expect that both explicit MPC and GERPC for initial conditions far from the origin will show sub-optimality. However, for this example it is hard to find initial conditions where explicit MPC shows significant sub-optimality. GERPC on the other hand, is not optimal, probably since it stays away from the constraints. As we see from the table below, the scaling technique of Kouvaritakis et al. (2002) reduces sub-optimality slightly, while Algorithm 1 reduces it significantly.

Initial condition	Cost				
	GERPC (9)	GERPC w/scaling	GERPC Algorithm 1	explicit MPC	infinite horizon MPC
$(-4, 0)$	832.72	831.72	677.89	609.44	609.39
$(-2, .6)$	378.24	374.57	234.63	204.46	204.46

5.2. Lab helicopter example

This example compares GERPC and ERPC applied to control a laboratory helicopter (Quanser 3-DOF Helicopter). This process has significant non-linearities, but we use a linear model for simplicity. Using two “local” PD controllers to decouple pitch and elevation (reducing non-linearity), the following six states (λ is “travel”, p is pitch angle of helicopter and e is elevation angle), 2 input (setpoints to pitch and elevation controllers) model is used:

$$\begin{bmatrix} \dot{\lambda} \\ \ddot{\lambda} \\ \dot{p} \\ \ddot{p} \\ \dot{e} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -.451 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -19.8 & -7.28 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -3.09 & -4.98 \end{bmatrix} \begin{bmatrix} \lambda \\ \dot{\lambda} \\ p \\ \dot{p} \\ e \\ \dot{e} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 19.8 & 0 \\ 0 & 0 \\ 0 & 4.98 \end{bmatrix} \begin{bmatrix} p_c \\ e_c \end{bmatrix}.$$

The weights are chosen as $Q = \mathrm{diag}\{1, 1, 1, 10, 1, 10\}$ and $R = \mathrm{diag}\{95, 95\}$.

A comparison of the volumes obtained by GERPC and ERPC is shown in the table below. The volume factor $V_f = \sqrt{\det(T Q_z T^{\mathrm{T}})^{-1}}$ is a measure of the size of the ellipsoids. A three-dimensional projection of the ellipsoids is shown in Fig. 5 (most of the other projections showed similar proportions). We clearly see that even for an extremely short control horizon, GERPC is able to obtain significantly larger regions of attractions than ERPC.

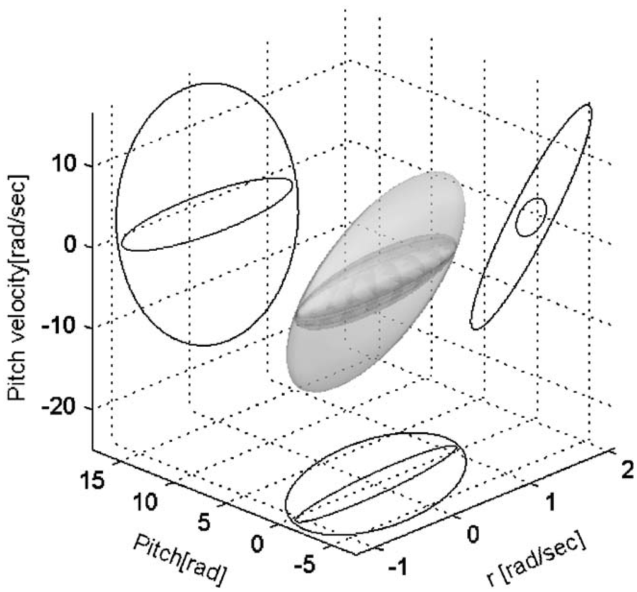

Fig. 5. Projections of six-dimensional ellipsoids. GERPC horizon $n_c = 2$ (outer ellipsoid) and ERPC for $n_c = 13$.

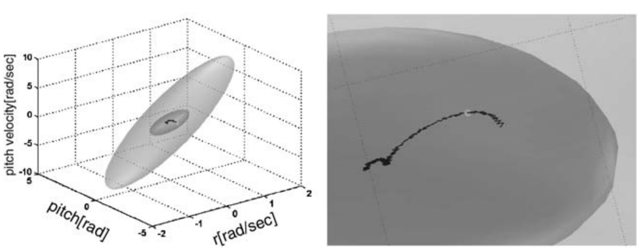

Fig. 6. A projection of two ellipsoids (the GERPC region of attraction and the largest invariant ellipsoid for unconstrained LQ) and one state trajectory, and (right) a (rotated) zoom. The point where the control becomes unconstrained is marked with a dot.

Algorithm	n_c	Offline calc. time (s)	dim Q_z	$V_f\,[\times 10^3]$
GERPC	2	600	10×10	55.3
ERPC	2	0.9	10×10	0.6
ERPC	8	48.7	22×22	1.4
ERPC	13	560	32×32	2.7

In order to get a reasonable region of attraction, we had to increase n_c for ERPC. For the given real-time system environment, however, the online optimization problem of ERPC with $n_c \geqslant 10$ was too computationally demanding, and lead to computer exceptions and hence controller failure.

Fig. 6 shows three of the states from a laboratory trial starting outside the invariant ellipsoid given by the LQ-controller, and entering it after about 0.4 s. This is confirmed by

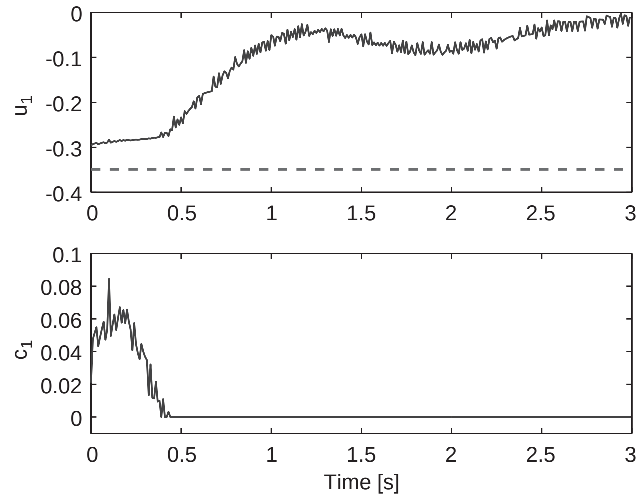

Fig. 7. At top, the first input (the pitch reference, u_1) with the constraint ($-20°$), with the corresponding c_1 (cf. (3)).

Fig. 7, where we see that the "perturbation" to the first control also is zero after 0.45 s.

6. Discussion

Offline problem: The examples clearly indicates that the generalized offline problem achieved considerably larger regions of attractions than ERPC. The generalization of the offline problem gives more freedom in "shaping" the ellipsoid, which can be especially helpful in the presence of state constraints as in the first example. Furthermore, we believe that the generalized offline problem in the multiple input case (as in the second example) has more freedom for exploiting possible couplings between the inputs than the ERPC offline problem, since G is a general matrix as opposed to the "diagonal structure" of the M matrix of ERPC. However, the regions of attraction obtained by GERPC and ERPC will always be a subset of the set where the explicit MPC controller is defined, but this set is not necessarily a region of attraction, as discussed below.

In the results herein, we used BMI solvers for solving the offline optimization problem. As mentioned earlier, in many cases the offline problem admits a convex parametrization (Cannon & Kouvaritakis, 2005). This is the case for the first example, but not the second (with the chosen control horizon). Using the convex LMI formulation for the first example, the GERPC ellipsoids can approximate the explicit MPC region arbitrarily close, at the expense of online cost. However, the LMI formulation (Cannon & Kouvaritakis, 2005) also allows for a trade-off between size of ellipsoids and online sub-optimality. This feature is exploited in Imsland and Rossiter (2005) to combine GERPC with "Triple-mode MPC" (Rossiter et al., 2001).

Online problem. The first example reveals that while GERPC/ERPC requires limited memory online, explicit

MPC performs better as far as computational complexity and sub-optimality is concerned. However, it is not hard to imagine examples where the memory requirements of explicit MPC is prohibitive.

In the first example, we were able to reduce sub-optimality in GERPC considerably by the new online algorithm suggested in Algorithm 1, at negligible additional computational cost. The fact that scaling is not as effective as in Kouvaritakis et al. (2002) in this case, is probably due to a combination of the following: The state constraint (state constraints were not considered in the examples in Kouvaritakis et al. (2002)), that due to the larger ellipsoidal regions of attraction we start further away from the origin, and that the ellipsoid (Lyapunov matrix) calculated by the BMI solver is badly conditioned (in (x, f)-space). Applying the LMI formulation (Cannon & Kouvaritakis, 2005) to the offline problem of the first example such that the ellipsoid is about the same size, gives better online performance of GERPC, and less difference between "scaling" and Algorithm 1. This is probably related to the conditioning problem mentioned above.

Stability and robustness. GERPC/ERPC gives (robust) stability guarantees, while, on the other hand, it is well known that finite horizon MPC (as implemented by explicit MPC) might enter "blind alleys" if the horizon is not long enough. Thus, the fact that the MPC controller is defined for an initial condition does not imply that this initial condition is within the region of attraction. However, the stability of the piecewise affine controller can be checked (conservatively) using, e.g. piecewise quadratic Lyapunov functions and LMIs, or one can enforce stability by design, by either finding the horizon length that guarantees stability, or adding stability constraints (at the cost of a more complex controller).

While GERPC/ERPC straightforwardly can handle uncertainty by the use of polytopic models, this is not so straightforward for explicit MPC. Some results in that direction have appeared recently (Bemporad, Borrelli, & Morari, 2003), however restricted to 1- and ∞-norm type objective functions.

7. Conclusion

By considering feasibility through a non-convex optimization problem offline, we are able to pose efficient online optimization algorithms for constrained optimization of linear systems and systems described by polytopic uncertainty models. Compared to ERPC larger regions of attractions were achieved. Furthermore, a less sub-optimal online problem was proposed. The new approach offers few advantages to explicit MPC in terms of online computational efficiency or sub-optimality, but it uses less online memory. In addition, the new approach gives (in the same way as ERPC) stability guarantees, and model uncertainty can be handled.

Acknowledgements

The authors gratefully acknowledge Dr. Petter Tøndel for helpful discussions and providing the results for explicit MPC in the double integrator example, and Dr. Mark Cannon and Professor Basil Kouvaritakis for valuable discussions on ERPC. The Gas Technology Center NTNU-SINTEF is acknowledged for financial support.

References

Bemporad, A., Borrelli, F., & Morari, M. (2003). Min–max control of constrained uncertain discrete-time linear systems. *IEEE Transactions on Automatic Control*, *48*(9), 1600–1606.

Bemporad, A., Morari, M., Dua, V., & Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, *38*(1), 3–20.

Boyd, S., El Ghaoui, L., Feron, E., & Balakrishnan, V. (1994). Linear matrix inequalities in system and control theory. In: *SIAM studies in applied mathematics* (vol. 15). Philadelphia; PA, SIAM.

Cannon, M., & Kouvaritakis, B. (2005). Optimising prediction dynamics for robust MPC. In: *Proceedings of the 16th IFAC world congress*, Prague, Czech Republic.

Cannon, M., Kouvaritakis, B., & Rossiter, J. A. (2001). Efficient active set optimization in triple mode MPC. *IEEE Transactions on Automatic Control*, *46*(8), 1307–1312.

Drageset, S., Imsland, L., & Foss, B. A. (2003). Efficient model predictive control with prediction dynamics. In: *Proceedings of the seventh European control conference ECC*, 2003. Cambridge, England.

Imsland, L., & Rossiter, J. A. (2005). Time varying terminal control. In: *Proceedings of the 16th IFAC world congress*, Prague, Czech Republic.

Kouvaritakis, B., Cannon, M., & Rossiter, J. A. (2002). Who needs QP for MPC anyway?. *Automatica*, *38*(5), 879–884.

Kouvaritakis, B., Rossiter, J. A., & Schuurmans, J. (2000). Efficient robust predictive control. *IEEE Transactions on Automatic Control*, *45*(8), 1545–1549.

Rossiter, J. A., Kouvaritakis, B., & Cannon, M. (2001). Computationally efficient algorithms for constraint handling with guaranteed stability and near optimality. *International Journal Control*, *74*(17), 1678–1689.

Tøndel, P., Johansen, T. A., & Bemporad, A. (2003a). An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica*, *39*, 489–497.

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

Lars Imsland received his Ph.D. degree in Electrical Engineering at the Department of Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU) in 2002. For parts of his Ph.D. studies, he was a visiting research scholar at Institute for Systems Theory in Engineering at University of Stuttgart, Germany. After his Ph.D. studies, he worked two years as a post-doctoral researcher at the Gas Technology Center NTNU-SINTEF, before joining SINTEF ICT, Applied Cybernetics in Trondheim, Norway as a Research Scientist. His main research interests are in the area of theory and application of non-linear and optimizing control and estimation.

Nadav S. Bar received his M.Sc. degree from Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), in 2003. His Ph.D. research at the same institution involves dynamic modeling of the metabolism cycle in vertebrates. He is currently a visiting scholar in the Medical Faculty at the University of Washington, Seattle, working on the protein synthesis elongation process during the translation of the mRNA. The Ph.D. is a collaboration between the Department of Engineering Cybernetics, NTNU and BioMar, Prediktor, and SINTEF.

Bjarne A. Foss is a Professor at the Department of Engineering Cybernetics at the Norwegian University of Science and Technology since 1989. He received his P.hD. from the same institution in 1987. He has been a visiting professor at the University of Texas at Austin. His main interests are in modeling and optimization-based control in particular related to process control applications and commercialization of such technology. He is a co-founder of two companies: Cybernetica AS and Cyberlab.org AS. Currently he leads the Gas Technology Center NTNU-SINTEF. He is a senior member of IEEE, Associate Editor of Journal of Process Control and member of the The Norwegian Academy of Technical Sciences.