Preprints of 4th IFAC Nonlinear Model Predictive Control Conference
International Federation of Automatic Control
Noordwijkerhout, NL. August 23-27, 2012

Simple and Certifiable Quadratic Programming Algorithms for Embedded Linear Model Predictive Control [⋆]

Alberto Bemporad [*] **Panagiotis Patrinos** [*]

[*] *IMT Institute for Advanced Studies Lucca, Piazza San Ponziano 6, 55100 Lucca, Italy.*
{alberto.bemporad,panagiotis.patrinos}@imtlucca.it

Abstract: In this paper we review a dual fast gradient-projection approach to solving quadratic programming (QP) problems recently proposed in [Patrinos and Bemporad, 2012] that is particularly useful for embedded model predictive control (MPC) of linear systems subject to linear constraints on inputs and states. We show that the method has a computational effort aligned with several other existing QP solvers typically used in MPC, and in addition it is extremely easy to code, requires only basic and easily parallelizable arithmetic operations, and a number of iterations to reach a given accuracy in terms of optimality and feasibility of the primal solution that can be estimated quite tightly by solving an off-line mixed-integer linear programming problem.

1. INTRODUCTION

Model Predictive Control (MPC) is a well known methodology for synthesizing feedback control laws that optimize closed-loop performance subject to prespecified operating constraints on inputs, states, and outputs [Mayne and Rawlings, 2009, Bemporad, 2006]. Attracted by such capabilities, in the last few years the automotive and aerospace industries showed increased attention to MPC for a variety of different applications, setting crucial requirements on the optimization algorithms, in particular quadratic programming (QP) solvers, for enabling the use of MPC in real products. For a QP solver to be embedded in the control hardware for implementing a MPC law, it must be *fast* enough to provide a solution within short sampling intervals (such as 1-50 ms), require *simple hardware* (such as a microcontroller or an FPGA), require *little memory* to store the data defining the optimization problem and the code implementing the algorithm itself, a *simple code* that is software-certifiable, especially in safety-critical applications, and good worst-case estimates of the execution time to meet hard *real-time* system requirements.

Such requirements have stimulated extensive research in the MPC community during the last decade, and to date many good algorithms and packages for QP are available to solve linear MPC problems, such as active-set methods [Ricker, 1985, Ferreau et al., 2008, Schmid and Biegler, 1994], interior-point methods [Mattingley and Boyd, 2010, Wang and Boyd, 2010], and the dual piecewise smooth Newton method [Patrinos et al., 2011]. A different approach to meet the above requirements was taken in [Bemporad et al., 2002], where multiparametric quadratic programming is proposed to pre-solve the QP off-line, therefore converting the MPC law into a continuous and piecewise affine function of the state vector. The only drawback of such approach is that it is limited to relatively small problems (typically one or two command inputs, short control and constraint horizons, up to ten states) and to linear time-invariant (LTI) systems.

The authors of [Richter et al., 2009, 2011] were the first to apply the fast gradient method [Nesterov, 1983, 2004] in the context of linear MPC. In [Richter et al., 2009] the authors applied Nesterov's first fast gradient method [Nesterov, 1983] to the primal QP for input-constrained linear MPC problems with simple constraint sets (e.g., a box), giving computable iteration bounds for the level of (primal) suboptimality. In [Richter et al., 2011] Nesterov's first fast gradient method was applied to the dual of state constrained linear MPC problems resulting by relaxing the state equations, giving computable iteration bounds to reach a dual solution whose cost is within certain tolerance from the optimal cost. Successful application of the algorithm proposed in [Richter et al., 2011] relies heavily on being able to easily compute the projection onto the feasible set, limiting its applicability to problems where input and state constraint sets are boxes and the terminal set is a box or an ellipsoid. Results involving fast-gradient methods for MPC were also proposed more recently in [Kögel and Findeisen, 2011a,b]

A new algorithm tailored to embedded linear (possibly time-varying) MPC problems subject to general polyhedral constraints on inputs and states was recently proposed by Patrinos and Bemporad [2012]. The algorithm is based on applying the fast gradient method of [Nesterov, 1983] to the dual problem, that results by relaxing the inequality constraints, and has a convergence rate of $O(1/\nu^2)$, where ν is the iteration counter. Global bounds on the number of iterations were provided to reach a given accuracy not only for dual optimality but also for primal optimality and

[⋆] This work was partially supported by the "HYCON2 – Highly-complex and networked control systems" Network of Excellence, FP7-IST contract no. 257462 and by the European project E-PRICE: Price-based Control of Electrical Power Systems, FP7-IST contract no. 249096.

feasibility, that is the main concern in MPC applications. Based on this bounds, practical termination criteria were also proposed. In this paper we review such a method and explicitly specialize it for QP problems that arise in formulating linear MPC problems in condensed form.

2. NOTATION

Let $\mathbb{R}$, $\mathbb{N}$, $\mathbb{R}^n$, $\mathbb{R}^{m\times n}$, denote the field of real numbers, the set of non-negative integers, the set of column real vectors of length n, the set of m by n real matrices, respectively. Let $\mathbb{S}^n_{++}$ ($\mathbb{S}^n_{+}$) denote the set of symmetric and positive (semi)definite n by n matrices. The transpose of a matrix $A \in \mathbb{R}^{m\times n}$ is denoted by A'. For $z \in \mathbb{R}^n$, $[z]_+$ denotes its Euclidean projection on the nonnegative orthant, i.e., the vector whose i-th coordinate is $\max\{z_i, 0\}$. For a vector $z \in \mathbb{R}^n$, $\|z\|$ denotes the Euclidean norm of z, while if $A \in \mathbb{R}^{m\times n}$, $\|A\|$ denotes the spectral norm of A (unless otherwise stated). For a vector $z \in \mathbb{R}^n$, z_i denotes its ith component, $i \in \mathbb{N}_{[1,n]}$, while for a matrix $A \in \mathbb{R}^{m\times n}$, A_j denotes its jth row, $i \in \mathbb{N}_{[1,m]}$.

3. LINEAR MPC SETUP

Consider the following finite-time optimal control problem formulation for MPC

$$V^*(p) = \min_z \ \ell_N(x_N) + \sum_{k=0}^{N-1} \ell_k(x_k, u_k) \tag{1a}$$

$$\text{s.t. } x_0 = p \tag{1b}$$

$$x_{k+1} = A_k x_k + B_k u_k + f_k \tag{1c}$$

$$F_k x_k + G_k u_k \le c_k \tag{1d}$$

$$k = 0, \dots, N-1 \tag{1e}$$

$$F_N x_N \le c_N \tag{1f}$$

where N is the prediction horizon, $p \in \mathbb{R}^{n_x}$ is the current state vector, $u_k \in \mathbb{R}^{n_u}$ is the vector of manipulated variables at prediction time k , $k = 0, \dots, N-1$, $z \triangleq [\, u_0' \, \cdots \, u_{N-1}' \,]' \in \mathbb{R}^n$, $n \triangleq n_u N$, is the vector of decision variables to be optimized,

$$\ell_k(x, u) = \frac{1}{2} [\, x' \; u' \,] \begin{bmatrix} Q_k & S_k' \\ S_k & R_k \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} + [\, q_k' \; r_k' \,] \begin{bmatrix} x \\ u \end{bmatrix} + s_k, \tag{2a}$$
$$k = 0, \dots, N-1$$

$$\ell_N(x) = \frac{1}{2} x' Q_N x + q_N' x + s_N \tag{2b}$$

are the stage cost and terminal cost, respectively, $Q_N \in \mathbb{S}^{n_x}_{+}$, $\begin{bmatrix} Q_k & S_k' \\ S_k & R_k \end{bmatrix} \in \mathbb{S}^{n_x+n_u}_{+}$, $R_k \in \mathbb{S}^{n_u}_{++}$. By letting $n_{ck} \in \mathbb{N}$ be the number of constraints imposed at prediction time k, matrices $F_k \in \mathbb{R}^{n_{ck}\times n_x}$, $c_k \in \mathbb{R}^{n_{ck}}$, $k = 0, \dots, N$, $G_k \in \mathbb{R}^{n_{ck}\times n_u}$, $k = 0, \dots, N-1$, are the matrices defining the constraints imposed in the MPC problem formulation.

The MPC control law is defined by setting

$$u^*(p) = [I \ 0 \ \dots \ 0] z^*(p) \tag{3}$$

where $z^*(p)$ is the optimizer of problem (1) for the current selection of p.

By eliminating the state sequence from problem (1) using (1c), the optimal control problem (1) can be expressed as the convex quadratic program (QP)

$$V^\star(p) \triangleq \min_z \quad \frac{1}{2} z' M z + (Cp + g)' z + \frac{1}{2} p' Y p \tag{4a}$$

$$\text{s.t.} \quad Gz \le Ep + b \tag{4b}$$

where $M \in \mathbb{S}^n_{++}$ is the Hessian matrix, $C \in \mathbb{R}^{n_x\times n}$ and $g \in \mathbb{R}^m$ define the linear term of the cost function, $Y \in \mathbb{R}^{n_x\times n_x}$ has no influence on the optimizer as it only affects the optimal value of (4a), and by letting $m \triangleq \sum_{k=0}^{n} n_{ck}$, matrices $G \in \mathbb{R}^{m\times n}$, $E \in \mathbb{R}^{m\times n_x}$, $b \in \mathbb{R}^m$ define in a compact form the linear constraints imposed in (1).

Note that in case the matrices defining the dynamics, costs, and constraints in (1) do not vary during the execution of the MPC controller, such as in case the controller is based on a linear time-invariant (LTI) prediction model, the QP matrices in (4) can be precomputed off-line. This is no longer possible in case the matrices in (1) depend on the execution time and/or the current state vector p, such as in the case of MPC of linear time-variant (LTV) systems or nonlinear systems whose dynamics is linearized around the current state p.

The MPC formulation (1) is rather general and covers several instances commonly used in practice, as detailed in Appendix A.

4. ACCELERATED DUAL GRADIENT-PROJECTION

Consider the dual problem of (4)

$$\Phi^\star(p) \triangleq \min_y \quad \frac{1}{2} y' H y + (Dp + d)' y + d_p \tag{5a}$$

$$\text{s.t.} \quad y \ge 0 \tag{5b}$$

where $H = GM^{-1}G'$, $D = GM^{-1}C + E$, $d = GM^{-1}g + b$, $d_p = \frac{1}{2}(Cp+g)'M^{-1}(Cp+g) + p'Yp$. Since (4) is a convex quadratic program, strong duality always holds (without the need of any constraint qualification). Therefore $V^\star(p) = -\Phi^\star(p)$, and in principle one could solve the dual problem (5) to obtain a dual optimal vector $y^\star(p) \in \mathbb{R}^m$ and then calculate the primal optimal vector $z^\star(p) \in \mathbb{R}^n$ by setting

$$z^*(p) = -M^{-1}(G' y^*(p) + Cp + g) \tag{6}$$

Denote by $\Phi(p, y) \triangleq \frac{1}{2} y' H y + (Dp + d)' y + \frac{1}{2} d_p$. Then

$$\nabla_y \Phi(p, y) = Hy + Dp + d$$

is a linear function of both p and y, and in particular it is Lipschitz continuous with respect to y, i.e., $\|\nabla_y \Phi(p, y_1) - \nabla_y \Phi(p, y_2)\| = \|H(y_1 - y_2)\| \le L\|y_1 - y_2\|$, where L can be defined as the maximum eigenvalue λ_H of H, or, in alternative, as an upper bound on λ_H, such as the Frobenius norm $\|H\|_F = \sqrt{\sum_{i,j=1}^m |H_{i,j}|^2}$, or the induced 1-norm $\|H\|_1 = \max_{j\in\mathbb{N}_{[1,m]}} \sum_{i=1}^m |H_{i,j}|$, which by symmetry of H also coincides with the induced ∞-norm $\|H\|_\infty = \max_{i\in\mathbb{N}_{[1,m]}} \sum_{j=1}^m |H_{i,j}|$.

In [Patrinos and Bemporad, 2012], an Accelerated Dual Gradient Projection (GPAD) algorithm was proposed to solve (4), that we briefly recall in this section. The goal is to compute an $(\varepsilon_V, \varepsilon_g)$-optimal solution for (4), defined as follows.

Definition 1. Consider two nonnegative constants ε_V, ε_g. We say that $z \in \mathbb{R}^n$ is an $(\varepsilon_V, \varepsilon_g)$-optimal solution for (4) if

$$V(z,p) - V^\star(p) \le \varepsilon_V \tag{7a}$$
$$\max_{i\in\mathbb{N}_{[1,m]}} G_i z - E_i p - b_i \le \varepsilon_g \tag{7b}$$

where $V(z,p) = \frac{1}{2}z'Mz + (Cp+g)'z + \frac{1}{2}p'Yp$.

Note that although in most dual methods (e.g., [Richter et al., 2011]) the goal is to find an approximate dual solution (i.e., $\Phi(p,y) - \Phi^\star(p) \le \varepsilon_\Phi$), the GPAD algorithm of Patrinos and Bemporad [2012] finds an approximate primal solution as defined in (1), which is very important in MPC applications where the task is to compute the optimal input sequence $z^*(p)$ solving (1) for $p = x(t)$. The GPAD algorithm is the first fast-gradient method of Nesterov [1983] (see also [Nesterov, 2004], [Bertsekas, 2009, Section 6.9], [Tseng, 2008, Alg. 2], [Beck and Teboulle, 2009]) applied to the convex QP problem (5), and is based on the following iterations:

$$w_\nu = y_\nu + \beta_\nu(y_\nu - y_{\nu-1}) \tag{8a}$$
$$\hat z_\nu = -M_G w_\nu - g_P \tag{8b}$$
$$z_\nu = (1-\theta_\nu)z_{\nu-1} + \theta_\nu \hat z_\nu \tag{8c}$$
$$y_{\nu+1} = [w_\nu + G_L\hat z_\nu + p_D]_+ \tag{8d}$$
$$y_0 = y_{-1} = 0,\ z_{-1} = 0$$

where $M_G \triangleq M^{-1}G'$, $g_P \triangleq M^{-1}(Cp+g)$, $G_L \triangleq \frac{1}{L}G$, $p_D \triangleq -\frac{1}{L}(Ep+b)$, $\nu \in \mathbb{N}$ is the iteration counter, and β_ν is a sequence of scalars generated by the following recursions

$$\begin{aligned} \theta_{\nu+1} &= \frac{\sqrt{\theta_\nu^4 + 4\theta_\nu^2} - \theta_\nu^2}{2} \\ \beta_\nu &= \theta_\nu(\theta_{\nu-1}^{-1} - 1) \\ \theta_0 &= \theta_{-1} = 1 \end{aligned} \tag{8e}$$

Note that, after precomputing matrices M_G, G_L (that do not depend on p) and vectors p_D, g_P, the main computational effort of the GPAD iterations is to compute (8b) and (8d), that both require a matrix-vector product whose cost is of order $O(nm)$. For QP problems that derive from MPC formulations (1), this amounts to a complexity of $O(N^2)$, where N is the prediction horizon. A more efficient way of computing (8b) and (8d) was proposed by Patrinos and Bemporad [2012] which requires operations of complexity $O(N)$. Note also that the iterations (8) can be easily executed on m parallel processors. Finally, note that, depending on a CPU/memory tradeoff, the sequence of β_ν, θ_ν, can be either computed on-line via (8e), or pre-computed off line and stored in memory for $\nu = -1, 0, \ldots, N_\nu$, where N_ν is an upper-bound on the number iterations, as detailed in the next section.

Below we characterize the optimal dual solution of the QP problem (4).

Lemma 1. A $y \in \mathbb{R}^m$ is optimal for the dual QP problem (5) if and only if it satisfies the following piecewise affine equation

$$y = [y - \frac{1}{L}(Hy + Dp + d)]_+ \tag{9}$$

Proof. The proof easily follows by the first-order optimality conditions for the convex optimization problem $\min_{y\ge 0}\Phi(p,y)$, expressed in terms of the Euclidean projection operator: $y \in \arg\min_{y\ge0}\Phi(p,y) \iff y = [y - \alpha\nabla_y\Phi(p,y)]_+, \forall\ \alpha > 0$. □

For any $y \in \mathbb{R}^m$ satisfying (9), the primal solution of the QP problem (4) is recovered by (6). Note that by setting $y_0 = y_{-1}$ as in (9) and z_{-1} as in (6), then $w_0 = y_0$, $\hat z_0 = z_{-1}$, $y_1 = y_0$, and hence the iterations (8e) provide $z_\nu = \hat z_\nu = z_{-1}$ and $y_\nu = y_{-1}$, $\forall \nu \in \mathbb{N}$.

4.1 Bounds on the number of iterations

In [Patrinos and Bemporad, 2012, Theorem 2], the authors proved that for any $\nu \in \mathbb{N}_+$ the following inequality holds:

$$Gz_\nu - Ep - b \le \frac{8L}{(\nu+2)^2}\|y^\star(p)\| \begin{bmatrix}1\\ \vdots\\ 1\end{bmatrix} \tag{10}$$

which provides the upper bound $N_g(p,\varepsilon_g)$

$$N_g(p,\varepsilon_g) = \left\lceil \sqrt{\frac{8L\|y^*(p)\|}{\varepsilon_g}} \right\rceil - 2 \tag{11}$$

on the number of iterations that guarantee the averaged primal solution z_ν defined by (8c) to be feasible within a prescribed threshold $\varepsilon_g > 0$.

Regarding suboptimality, that is the distance of $V(z_\nu)$ from $V^\star$, in [Patrinos and Bemporad, 2012, Theorem 3] the authors proved that for any $\nu \in \mathbb{N}_+$

$$-\frac{8L}{(\nu+2)^2}\|y^\star(p)\|^2 \le V(z_\nu,p) - V^\star(p) \le \frac{2L}{(\nu+2)^2}\|y^\star(p)\|^2 \tag{12}$$

which provides the following upper bound

$$N_V(p,\varepsilon_V) = \left\lceil \sqrt{\frac{2L}{\varepsilon_V}}\|y^*(p)\| \right\rceil - 2 \tag{13}$$

on the number of iterations that guarantee the primal solution z_ν to be optimal within a prescribed threshold $\varepsilon_V > 0$ as in (7a). Notice that the lower bound in (12) may be also relevant, as z_ν could be infeasible, therefore one may have $V(z_\nu) \le V^\star$ as well.

Convergence results for the averaged primal solution z_ν have been reported in [Nesterov, 2005, Theorem 3] and [Tseng, 2008, Cor. 2]. A similar approach was followed by Necoara and Suykens [2008]. However these results concern saddle problems with compact dual and/or primal constraint sets, which may not be the case in the present general convex quadratic programming problem in which the dual constraint set is the nonnegative orthant.

The bounds in (11) and (13) depend on (a choice of) the optimal dual solution $y^*(p)$ and therefore on p. A few criteria to determine a *uniform* bound off-line on the number of iterations to reach a certain $(\varepsilon_V, \varepsilon_g)$-optimality, that is a bound for all p in a given polyhedron $\mathcal{P}$ for which the QP problem (4) admits a feasible solution, were reported in [Patrinos and Bemporad, 2012]. In the sequel, we denote by $N_\nu(\varepsilon_g, \varepsilon_V)$ such a uniform bound,

$$N_\nu(\varepsilon_g,\varepsilon_V) \ge \sup_{p\in\mathcal{P}} \max\{N_g(p,\varepsilon_g), N_V(p,\varepsilon_V)\} \tag{14}$$

An alternative criterion based on Lemma 1 and mixed-integer linear programming (MILP) is reported below.

Proposition 1. Let the set $\{x \in \mathbb{R}^{n+nx} : [G\ -E]x \le b\}$ be full dimensional and let $\mathcal{P}$ be a given polyhedron such that $\mathcal{P} \subseteq \{p \in \mathbb{R}^{n_x} : \exists z \in \mathbb{R}^n$ such that $Gz + \epsilon[1\ \ldots\ 1]' \le Ep + b\}$ for some $\epsilon > 0$. Then for all $p \in \mathcal{P}$ and all dual solutions $y^*(p)$ of (5)

$$\|y^*(p)\|_1 \le \Delta_y(\mathcal{P}) \tag{15}$$

where

$$\begin{aligned}\Delta_y(\mathcal{P}) = & \max_{y,p} \sum_{i=1}^{m} y_i \\ & \text{s.t.} \quad p \in \mathcal{P} \\ & \qquad y = [y - \frac{1}{L}(Hy + Dp + d)]_+\end{aligned} \tag{16}$$

Problem (16) computes the maximum 1-norm $\Delta_y(\mathcal{P})$ that a dual vector $y^*(p)$ solving (5) can take. Under the assumptions stated in Proposition 1, Patrinos and Bemporad [2012] proved that such $\Delta_y(\mathcal{P})$ is bounded. Problem (16) can be easily reformulated as an MILP by introducing a binary vector $\delta \in \{0,1\}^m$ such that $[\delta_i = 1] \leftrightarrow [y_i - \frac{1}{L}(H_i y + D_i p + d_i) \geq 0]$ and by imposing

$$\begin{cases} y_i \geq 0 \\ y_i \leq \lambda_i^+ \delta_i \\ \frac{1}{L} H_i y \leq -\frac{1}{L}(D_i p + d_i) + \mu_i^+(1 - \delta_i) \\ \frac{1}{L} H_i y \geq -\frac{1}{L}(D_i p + d_i) + \mu_i^-(1 - \delta_i) \end{cases}$$

where $\mu^-, \mu^+, \lambda^+ \in \mathbb{R}^m$ are suitable vectors of upper and lower bounds.

4.2 QP solution algorithms

A QP solver can be immediately derived based on the GPAD iterations (8) that provides an $(\varepsilon_V, \varepsilon_g)$-optimal solution. We propose here two possible algorithms. The first one, described by Algorithm 1, is based on a stopping criterion that verifies $(\varepsilon_V, \varepsilon_g)$-optimality.

Algorithm 1 GPAD with termination criterion

```
0. init: $y_0 = y_{-1} = 0$, $z_{-1} = 0$, $\nu = 0$;
1. compute $w_\nu$, $z_\nu$, $y_{\nu+1}$ as in (8);
2. if $G_i z_\nu - E_i p - b_i > \varepsilon_g$ for some $i \in \mathbb{N}_{[1,m]}$ or
   $\Phi(y_{\nu+1}) + V(z_\nu) > \varepsilon_V$ then go to 1,
3. stop. $z_\nu$ = primal solution, $y_{\nu+1}$ = dual solution.
```

Note that Step 2 requires the computation of the constraint violation term $Gz_\nu - Ep - b$ and of the duality gap

$$\begin{aligned}\Delta V_\nu &\triangleq \Phi(y_{\nu+1}) + V(z_\nu) \\ &= \frac{1}{2}(y_{\nu+1}' H y_{\nu+1} + z_\nu' M z_\nu) + (Dp + d)' y_{\nu+1} \\ &\quad + (Cp + g)' z_\nu + d_p\end{aligned} \tag{17}$$

As remarked in [Patrinos and Bemporad, 2012], one could also consider $\hat{z}_\nu$ as a candidate solution, which means to also evaluate (17) at $(\hat{z}_\nu, y_{\nu+1})$ and the violation $G\hat{z}_\nu - Ep - b$, and stop if either $(\hat{z}_\nu, y_{\nu+1})$ or $(z_\nu, y_{\nu+1})$ are $(\varepsilon_V, \varepsilon_g)$-optimal. Moreover, Algorithm 1 could be sensibly sped up by verifying such conditions only every K iterations (for example, $K = 10$), in this case the bounds N_g, N_V, N_ν are increased by K, but computations are sensibly reduced. Finally, in Step 2 of Algorithm 1 an additional termination criterion based on exceeding a given maximum number $N_{\max}$ of iterations can be introduced to prevent an infinite loop in case the given QP problem is infeasible.

A second possible algorithm is described by Algorithm 2 and is based on a fixed number N_ν of iterations, in light of (11), (13), and (14).

Algorithm 2 is particularly adequate for implementation in hard real-time systems, in which one must anyway

Algorithm 2 GPAD with fixed number of iterations

```
0. init: $y_0 = y_{-1} = 0$, $z_{-1} = 0$;
1. for $\nu = 0$ to $N_\nu(\varepsilon_V, \varepsilon_g)$
       compute $w_\nu$, $z_\nu$, $y_{\nu+1}$ as in (8);
2. stop. $z_\nu$ = primal solution, $y_{\nu+1}$ = dual solution.
```

select the computation hardware to afford the worst-case number N_ν of iterations within the sample time of the MPC controller. In this respect, Algorithm 2 is preferable to Algorithm 1 as it avoids computing the duality gap and the amount of constraint violation as in Step 2 of Algorithm 1, which makes each iteration simpler and the computer code more compact.

The above algorithms enjoy the quite interesting property of returning the exact $(0,0)$-optimal solution in case p lies in the critical region $CR_\emptyset$ corresponding to all constraints in (4) being inactive,

$$CR_\emptyset = \{p \in \mathbb{R}^{n_x} : -(GM^{-1}C + E)p \leq b - GM^{-1}g\}$$

In fact, in this case $y^*(p) = 0$ and the exact optimal solution is $z^*(p) = -M^{-1}Cp + g$, and Algorithm 1 returns $w_\nu = 0$, $z_\nu = -M^{-1}Cp + g$, and $y_{\nu+1} = [p_D]_+ = 0$ for all $\nu \in \mathbb{N}$. Therefore, $z^*(p)$ is retrieved in one iteration by Algorithm 1 (or after $N_\nu(\varepsilon_V, \varepsilon_g)$ iterations by Algorithm 2). Retrieving the exact solution in $CR_\emptyset$ is a nice property of Algorithms 1–2, as possible local closed-loop stability properties of MPC are maintained. For enforcing global closed-loop stability properties, one should setup an MPC control law that is robust with respect to ε_V and ε_g perturbations, a topic of current research.

5. NUMERICAL EXAMPLES

5.1 Example 1

We compare the performance of Algorithm 1 against other existing QP solvers: QPOASES [Ferreau et al., 2008], the QP solver of the MPC Toolbox [Bemporad et al., 2004] DANTZGMP, based on the active set method described in [Ricker, 1985], the new QP solver of the MPC Toolbox implementing the QPKWIK algorithm [Schmid and Biegler, 1994], CPLEX 11.2 [ILOG, Inc., 2008], and QUADPROG of the Optimization Toolbox. All test were run on a Macbook Air 2.13 GHz Intel Core Duo running MATLAB R2009b. Algorithm 1, DANTZGMP and QPKWIK were run in compiled Embedded MATLAB code. All QP solvers are cold-started.

Tests were performed on random QP problems with n variables and $2n$ constraints for increasing values of n, with $\varepsilon_V = 10^{-2}$, $\varepsilon_g = 10^{-3}$, $L = \|H\|_F$, and Step 2 executed every $K = 10$ iterations, stopping when the first between $\hat{z}$ and z satisfies $(\varepsilon_g, \varepsilon_V)$-optimality. Figure 1 shows the CPU time (averaged on 20 QP's for each n) spent by each solver (note that for Algorithm 1, DANTZGMP, and QPKWIK, this also includes the time to prepare the matrices for the solver, although in linear MPC problems this is typically an off-line time).

The purpose of Figure 1 is to show that despite the fact that Algorithm 1 is mainly conceived for implementation simplicity and for predictability of the number of iterations, it has a computation complexity that is comparable

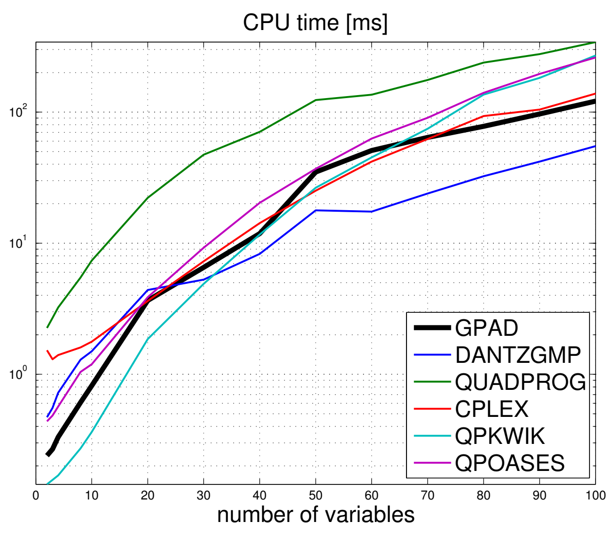

Fig. 1. Comparison of the GPAD algorithm against other QP solvers

to other solvers typically used in MPC. We do not aim at comparing the exact computation complexity, which would require defining equivalent conditions of $(\varepsilon_g, \varepsilon_V)$-optimality for the other solvers, a task that is hardly doable as most of such solvers use active set methods.

5.2 Example 2

To test the tightness of the theoretical bounds on the number of iterations, we consider the double integrator problem suggested in [Bemporad et al., 2002] with horizon $N = 2$ and set $\varepsilon_V = 10^{-2}$, $\varepsilon_g = 10^{-3}$, L to the maximum eigenvalue of matrix H, $K = 1$, and the stopping criterion verified on z only. Over the box $\mathcal{P} = \{p \in \mathbb{R}^2 : \|p\|_\infty \leq 15\}$, Problem (16) provides $\Delta_y(\mathcal{P}) = 34.90$, corresponding to the theoretical upper-bound $N_\nu = 10718$ on the number of iterations. Extensive simulations with random vectors $p \in \mathcal{P}$ provide a worst-case $\|y^*(p)\|_1 = 29.53$ and worst-case number of iterations $\nu = 1991$.

6. CONCLUSION

In this paper we have reviewed the GPAD algorithm developed by Patrinos and Bemporad [2012] and particularized to the case of the condensed QP problems that arise from linear MPC formulations. Although the new solver has a computation performance comparable to many other existing QP solvers, it is extremely simple to code and only involves (easily parallelizable) products, sums, and comparisons, and rather tight upper-bounds can be computed off-line on the maximum number of iterations by solving a mixed-integer programming problem. These features make the approach quite suitable for embedded MPC applications.

REFERENCES

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

A. Bemporad. Model-based predictive control design: New trends and tools. In *Proc. 45th IEEE Conf. on Decision and Control*, pages 6678–6683, San Diego, CA, 2006.

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

A. Bemporad, M. Morari, and N.L. Ricker. *Model Predictive Control Toolbox for Matlab – User's Guide*. The Mathworks, Inc., 2004. `http://www.mathworks.com/access/helpdesk/help/toolbox/mpc/`.

D.P. Bertsekas. *Convex optimization theory*. Athena Scientific, 2009.

H.J. Ferreau, H.G. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008.

ILOG, Inc. *CPLEX 11.2 User Manual*. Gentilly Cedex, France, 2008.

M. Kögel and R. Findeisen. A fast gradient method for embedded linear predictive control. In *Proc. 18th IFAC World Congress*, pages 1362–1367, 2011a.

M. Kögel and R. Findeisen. Fast predictive control of linear systems combining Nesterov's gradient method and the method of multipliers. In *Proc. 50th IEEE Conf. on Decision and Control and European Control Conf.*, pages 501–506, Orlando, USA, 2011b.

J. Mattingley and S. Boyd. CVXGEN: A code generator for embedded convex optimization. *Optimization and Engineering*, pages 1–27, 2010.

D.Q. Mayne and J.B. Rawlings. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, LCC, Madison,WI, 2009.

I. Necoara and J. Suykens. Application of a smoothing technique to decomposition in convex optimization. *IEEE Transactions on Automatic Control*, 53(11):2674–2679, 2008.

Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

Y. Nesterov. *Introductory lectures on convex optimization: A basic course*. Kluwer Academic Publishers, 2004.

Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

P. Patrinos and A. Bemporad. An accelerated dual gradient-projection algorithm for linear model predictive control. 2012. Submitted for publication.

P. Patrinos, P. Sopasakis, and H. Sarimveis. A global piecewise smooth Newton method for fast large-scale model predictive control. *Automatica*, 47(9):2016–2022, 2011.

S. Richter, C.N. Jones, and M. Morari. Real-time input-constrained MPC using fast gradient methods. In *Proc. 48th IEEE Conf. on Decision and Control*, pages 7387–7393, 2009.

S. Richter, M. Morari, and C.N. Jones. Towards computational complexity certification for constrained MPC based on Lagrange relaxation and the fast gradient method. In *Proc. 50th IEEE Conf. on Decision and Control and European Control Conf.*, pages 5223–5229, Orlando, USA, 2011.

N.L. Ricker. Use of quadratic programming for constrained internal model control. *Ind. Eng. Chem. Process Des.*

Dev., 24(4):925–936, 1985.

C. Schmid and L.T. Biegler. Quadratic programming methods for reduced Hessian SQP. *Computers & Chemical Engineering*, 18(9):817–832, 1994.

P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. Technical report, Department of Mathematics, University of Washington, 2008.

Y. Wang and S. Boyd. Fast model predictive control using online optimization. *IEEE Trans. Contr. Systems Technology*, 18(2):267–278, 2010.

Appendix A. LINEAR MPC FORMULATIONS

This appendix shows how to recast several instances of linear MPC commonly used in practice, such as for disturbance rejection and reference tracking, into the linear MPC formulation (1).

A.1 Measured disturbances

Measured disturbances $d_k \in \mathbb{R}^{n_d}$ entering the dynamics in the following form

$$x_{k+1} = A_k x_k + B_k u_k + E_k d_k + f_k$$

are easily modeled by defining a new affine term $\bar{f}_k = E_k d_k + f_k$, $k = 0, \dots, N-1$ in (1c). Assuming that the MPC controller is applied at time t, $p = x(t)$, an anticipative action (preview) on the measured disturbance $d(t)$, $d(t+1)$, $d(t+N-1)$ can be imposed by setting $d_k = d(t+k), \forall\, k = 0, \dots, N-1$, in alternative to $d_k \equiv d(t)$ (causal action, no preview). In the latter case (causal action) and when dynamics (A, B, f) are LTI, matrices M, C, G, B in (4) do not depend on the current time t and can be precomputed off line.

A.2 Incremental formulation

In practical implementations of MPC an incremental formulation is often used to solve tracking problems and/or to enforce constraints on input increments $\Delta u_k = u_k - u_{k-1}$, $k = 0, \dots, N-1$, where u_{-1} is the optimal input computed as in (3) and applied at the previous time step $t-1$. The MPC problem can be recast as in (1) by defining $\bar{u} = \Delta u$ as the new vector of free control moves, $\bar{x} = \left[\begin{smallmatrix} x' & v' \end{smallmatrix}\right]'$ as the new state vector, with $v_0 = u_{-1}$, along with the extended prediction model

$$\bar{A}_k = \left[\begin{smallmatrix} A_k & B_k \\ 0 & I \end{smallmatrix}\right], \ \bar{B}_k = \left[\begin{smallmatrix} B_k \\ I \end{smallmatrix}\right], \ \bar{f}_k = \left[\begin{smallmatrix} f_k \\ 0 \end{smallmatrix}\right] \qquad \text{(A.1)}$$

Weights and constraints involving input increments are included in the MPC formulation by suitably redefining the stage costs $\ell_k(\bar{x}_k, \bar{u})$ and the constraint matrices F_k, c_k, $k = 0, \dots, N$, and G_k, $k = 0, \dots, N-1$.

A.3 Reference tracking

Very often MPC is used for making an output vector $y_k = C_k x_k + D_k u_k$ track a given reference trajectory r_k^y, with $y_k, r_k^y \in \mathbb{R}^{n_y}$, by penalizing the stage cost

$$\ell_k = \frac{1}{2}(y_k - r_k^y)' Q_k^y (y_k - r_k^y) + \frac{1}{2}\Delta u_k' R_k^{\Delta_u} \Delta u_k \qquad \text{(A.2)}$$

with $Q_k^y \in \mathbb{S}_+^{n_y}$, and with no penalty on the terminal state x_N as in (2b) to avoid a bias from the origin. This is achieved by adopting the extended model (A.1) and by replacing the weights in (2a) with the following set of new weights

$$\begin{aligned} \bar{Q}_k &= \left[\begin{smallmatrix} C_k' Q_k^y C_k & C_k' Q_k^y D_k \\ D_k' Q_k^y C_k & D_k' Q_k^y D_k \end{smallmatrix}\right], & \bar{R}_k &= R_k^{\Delta_u} + D_k' Q_k^y D_k \\ \bar{S}_k &= \left[\begin{smallmatrix} D_k' Q_k^y C_k & D_k' Q_k^y D_k \end{smallmatrix}\right] & \bar{q}_k &= -\left[\begin{smallmatrix} C_k & D_k \end{smallmatrix}\right]' Q_y r_k^y \\ \bar{r}_k &= -D_k' Q_y r_k^y, & s_k &= \frac{1}{2}(r_k^y)' Q_y r_k^y \end{aligned} \qquad \text{(A.3)}$$

The same reasoning used for measured disturbances d_k applies here for enabling/disabling preview on the reference signal r^y.

A.4 Integral action

In order to get zero offsets in steady-state while tracking a constant reference signal r_k and/or rejecting a constant measured disturbance signal d_k, the MPC controller can be equipped with integral action. This amounts to augmenting the prediction model (1c) with the dynamics of the integral of the tracking error

$$I_{k+1} = I_k + (y_k - r_k^y)$$

and to adding the penalty $\frac{1}{2} I_k' Q_k^I I_k$ in the stage cost ℓ_k defined in (A.2), $Q_k^I \in \mathbb{S}_+^{n_y}$, $k = 0, \dots, N-1$. Accordingly, in (A.3) matrix $\bar{Q}_k$ is augmented as $\left[\begin{smallmatrix} \bar{Q}_k & 0 \\ 0 & Q_I \end{smallmatrix}\right]$, $\bar{S}_k$ as $\left[\begin{smallmatrix} \bar{S}_k & 0_{n_u \times n_y} \end{smallmatrix}\right]$, and $\bar{q}_k$ as $\left[\begin{smallmatrix} \bar{q}_k' & 0_{1 \times n_y} \end{smallmatrix}\right]'$.

A.5 Soft constraints

In order to prevent that the QP (4) becomes infeasible for particular choices of $p = x(t)$, it is common practice to "soften" the constraints in (1) by introducing slack variables σ_k, $\sigma_k \in \mathbb{R}$, $\sigma_k \geq 0$, $k = 0, \dots, N$:

$$\begin{aligned} F_k x_k + G_k u_k &\leq c_k + V_k \sigma_k, \ k = 0, \dots, N-1 \\ F_N x_N &\leq c_N + V_N \sigma_N \end{aligned} \qquad \text{(A.4)}$$

where $V_k \in n_{ck}$ are vectors with nonnegative entries (the largest the ith entry of V_k, the relatively softer the ith constraint is at prediction time k), $k = 0, \dots, N$, and by weighting $\rho_k \sigma_k^2$ in the stage costs, where $\rho_k \in \mathbb{R}$, $\rho_k > 0$ is a (large) weighting factor, $k = 0, \dots, N$. A problem with soft constraints as in (A.4) can be recast in the general form (1) by letting the prediction horizon be $\bar{N} = N+1$, $\bar{u} = \left[\begin{smallmatrix} u_k \\ \sigma_k \end{smallmatrix}\right]$ the new input vector, and

$$\begin{aligned} \bar{\ell}_k(x, \bar{u}) &= \frac{1}{2} \left[\begin{smallmatrix} x' & \bar{u}' \end{smallmatrix}\right] \left[\begin{smallmatrix} Q_k & S_k' & 0 \\ S_k & R_k & 0 \\ 0 & 0 & \rho_k \end{smallmatrix}\right] \left[\begin{smallmatrix} x \\ \bar{u} \end{smallmatrix}\right] + \left[\begin{smallmatrix} q_k' & r_k' & 0 \end{smallmatrix}\right] \left[\begin{smallmatrix} x \\ \bar{u} \end{smallmatrix}\right] + s_k \\ &\quad k = 0, \dots, \bar{N}-2 \\ \bar{\ell}_{\bar{N}-1}(x, \bar{u}) &= \frac{1}{2} x' Q_N x + q_N' x + s_N + \rho_N \sigma_N^2 + u_N' u_N \\ \bar{\ell}_{\bar{N}}(x) &= 0 \text{ (no terminal penalty)} \end{aligned}$$

and constraints

$$\begin{aligned} F_k x_k + [G_k \ \ -V_k] \bar{u} &\leq c_k, \ k = 0, \dots, \bar{N}-2 \\ F_N x_k + [0 \ \ -V_N] \bar{u} &\leq c_N \\ F_{\bar{N}} = 0, \ c_{\bar{N}} &= 0 \qquad \text{(no terminal constraint)} \end{aligned}$$

Clearly, the optimal value for the (unconstrained) additional variable u_N is always zero.

A.6 Blocking moves

In MPC practice, to limit the computation burden it is often useful to limit the number of degrees of freedom by "blocking" control moves after a certain input horizon N_u, $1 \leq N_u < N$:

$$u_k = K_k x_k + H_k u_{N_u-1},\ k = N_u, \ldots, N-1 \qquad \text{(A.5)}$$

For example the input signal is frozen and held constant after prediction time $N_u - 1$ by setting $K_k = 0$, $H_k = I$, while in the LTI case infinite horizon LQR cost is obtained by setting K_k equal to the LQR gain and $H_k = 0$. The same formulation as in (1) is obtained by setting $\bar{N} = N_u$ as the prediction horizon, and by expressing x_{N_u+h} as a function of x_{N_u-1}, u_{N_u-1}, $\forall h = 0, \ldots, N - N_u$. Considering for simplicity that dynamics (1c) and the blocking rule (A.5) are both LTI, $x_{N_u+h} = \mathcal{A}_h x_{N_u-1} + \mathcal{B}_h u_{N_u-1}$, $\mathcal{A}_h \triangleq (A+BK)^{h+1}$, $\mathcal{B}_h \triangleq \sum_{i=0}^{h}(A+BK)^i H)$, $h = 0, \ldots, N - N_u$. Then, define the new stage costs $\bar{\ell}_k = \ell_k$, $\forall k = 0, \ldots, \bar{N}-2$, $\bar{\ell}_{\bar{N}-1} = \ell_{N_u-1}(x_{N_u-1}, u_{N_u-1}) + \ell_N(\mathcal{A}_{N-N_u} x_{N_u-1} + \mathcal{B}_{N-N_u} u_{N_u-1}, K\mathcal{A}_{N-N_u} x_{N_u-1} + K\mathcal{B}_{N-N_u} u_{N_u-1} + H u_{N_u-1}) + \sum_{h=0}^{N-N_u} \ell_{\bar{N}+h}(\mathcal{A}_h x_{N_u-1} + \mathcal{B}_h u_{N_u-1}, K\mathcal{A}_h x_{N_u-1} + K\mathcal{B}_h u_{N_u-1} + H u_{N_u-1})$, $\bar{\ell}_{\bar{N}}(x_{\bar{N}}) = 0$. After incorporating the constraints imposed at prediction time $\bar{N}_u, \ldots, N$ in a single constraint for step $\bar{N}-1$, by suitably defining augmented constraint matrices $\bar{F}_{\bar{N}-1}$, $\bar{G}_{\bar{N}-1}$, $\bar{c}_{\bar{N}-1}$, and by removing the terminal constraint ($G_N = 0$, $c_N = 0$), we recast the MPC problem with blocked moves as in (1). Note that while the number of optimization variables is decreased, the number of constraints remains unchanged (the number of constraints involving only inputs can be decreased in case constant blocking $K_k = 0$, $H_k = I$ is used in (A.5)).

A.7 Stability constraints

For LTI formulations, various techniques have been proposed in the literature to enforce the asymptotic stability of the closed-loop MPC scheme a priori, that is by design. For example a polyhedral constraint on the terminal state x_N can be immediately modeled within the formulation (1). Note that, although in (1a) we do not assume that stage costs are nonnegative, for convergence analysis this might be required, which imposes a certain structure on the selection of Q_k, S_k, R_k, q_k, r_k, s_k.