A Quadratic Programming Algorithm Based on Nonnegative Least Squares With Applications to Embedded Model Predictive Control

Alberto Bemporad, *Fellow, IEEE*

***Abstract*—This technical note proposes an active set method based on nonnegative least squares (NNLS) to solve strictly convex quadratic programming (QP) problems, such as those that arise in Model Predictive Control (MPC). The main idea is to rephrase the QP problem as a Least Distance Problem (LDP) that is solved via a NNLS reformulation. While the method is rather general for solving strictly convex QP's subject to linear inequality constraints, it is particularly useful for embedded MPC because (i) is very fast, compared to other existing state-of-the-art QP algorithms, (ii) is very simple to code, requiring only basic arithmetic operations for computing LDL^T decompositions recursively to solve linear systems of equations, (iii) contrary to iterative methods, provides the solution or recognizes infeasibility in a finite number of steps.**

***Index Terms*—Active set methods, model predictive control, nonnegative least squares, quadratic programming.**

I. Introduction

Model Predictive Control (MPC) is widely spread in industry to optimize closed-loop response of multivariable systems under constraints on system variables [1]. Except for simple problems in which the MPC control law can be computed in explicit form [2] and even hard-coded on chip [3], embedding an MPC controller in an ECU requires coding and deploying a Quadratic Programming (QP) solver for computing the control signals.

Embedded QP for MPC has stimulated extensive research in the MPC community during the last decade, and to date many good algorithms and packages for QP are available that are able to solve linear MPC problems, such as active-set methods [4, Sec. 24-4], [5] interior-point methods [6], gradient projection methods [1], [7], and the alternating directions method of multipliers (ADMM) [8].

This technical note introduces a new active-set algorithm for strictly convex quadratic programs subject to general linear inequality constraints. The main idea is to recast the QP as a Least Distance Problem (LDP) and to transform this into a Nonnegative Least Squares (NNLS) problem. If the latter has a zero residual, the QP is proved infeasible, otherwise the algorithm provides the primal solution to the original QP problem (and, if needed, its dual solution). To solve the NNLS problem, we extend the well-established algorithm of [9, p. 161], introducing recursive LDL^T decompositions to speed up the solution of the unconstrained least squares problems required at each step of the algorithm. As a result, the proposed QP algorithm is very simple to code, very fast to execute, and provides the solution in a finite number of steps, all very attractive features for embedded MPC applications. The method is compared to several state-of-the-art solvers, including commercial ones, on both random QP tests and on a typical multivariable MPC application example.

This technical note complements the recent paper [2], where an algorithm was proposed to compute *offline* multiparametric QP (mpQP) solutions for getting MPC control laws in explicit form. The paper [2] showed that all polyhedral computations (such as removal of redundant inequalities) can be performed by NNLS; this technical note completes the approach by showing that also the QP's required by the mpQP solver can be solved by NNLS. The QP solution algorithm presented in this technical note has been extended in [10] to handle equality constraints, bilateral inequality constraints, and warm starts for solving mixed-integer quadratic programs.

Notation: Let $\mathbb{R}^n$ denote the set of real vectors of dimension n and $\mathbb{N}$ the set of natural integers, respectively. Let $\mathcal{I} \subset \mathbb{N}$ be a finite set of integers and denote by $\#\mathcal{I}$ its cardinality. For a vector $a \in \mathbb{R}^n$, a_i denotes the i-th entry of a, $a_{\mathcal{I}}$ the subvector obtained by collecting the entries a_i for all $i \in \mathcal{I}$, $\|a\|_2$ the Euclidean norm of a, $\|a\|_1 = \sum_{i=1}^n |a_i|$ the 1-norm of a, the condition $a > 0$ is equivalent to $a_i > 0$, $\forall i = 1, \ldots, n$ (and similarly for $\geq$, $\leq$, $<$), and $\mathrm{diag}(a)$ is the diagonal matrix whose (i,i)-th element is a_i. For a matrix $A \in \mathbb{R}^{n \times m}$, A' denotes its transpose, A_i denotes the i-th row of A, $A_{\mathcal{I}}$ the submatrix of A obtained by collecting the rows A_i for all $i \in \mathcal{I}$, and $A_{\mathcal{I}\mathcal{J}}$ the submatrix of A obtained by collecting the rows and columns of A indexed by $i \in \mathcal{I}$ and $j \in \mathcal{J}$, respectively. For a square matrix $A \in \mathbb{R}^{n \times n}$, A^{-1} denotes the inverse of A (if it exists) and A^{-T} its transpose, $A \succ 0$ ($A \succeq 0$) denotes positive definiteness (semidefiniteness) of A. Matrix I_n denotes the identity matrix of order n, where sometimes the subscript n is dropped if the dimension is clear from the context.

II. Problem Formulation and Main Results

We want to solve strictly convex QPs of the form

$$\min_z \quad V(z) \triangleq \frac{1}{2} z'Qz + c'z \tag{1a}$$

$$\text{s.t.} \quad Gz \leq g \tag{1b}$$

where $Q \succ 0$ is the Hessian matrix, $c \in \mathbb{R}^n$, $G \in \mathbb{R}^{q \times n}$, and $g \in \mathbb{R}^q$. Problem (1) arises when considering linear MPC formulations, see, e.g., [1].

Theorem 1: Consider the QP (1) and let $Q \succ 0$. Let $\mathcal{L}'\mathcal{L}$ be a Cholesky factorization of Q and define

$$M \triangleq G\mathcal{L}^{-1}, \quad d \triangleq g + GQ^{-1}c. \tag{2}$$

Consider the NNLS problem

$$\min_y \quad \frac{1}{2} \left\| \begin{bmatrix} -M' \\ -d' \end{bmatrix} y - \begin{bmatrix} 0 \\ \gamma \end{bmatrix} \right\|_2^2 \tag{3a}$$

$$\text{s.t.} \quad y \geq 0 \tag{3b}$$

Manuscript received December 15, 2014; revised April 15, 2015; accepted July 19, 2015. Date of publication July 21, 2015; date of current version March 25, 2016. Recommended by Associate Editor J. Imura.

The author is with the IMT Institute for Advanced Studies Lucca, 55100 Lucca, Italy (e-mail: alberto.bemporad@imtlucca.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2015.2459211

where γ is any positive scalar, and let

$$r^* \triangleq -\begin{bmatrix} M' \\ d' \end{bmatrix} y^* - \begin{bmatrix} 0 \\ \gamma \end{bmatrix} \tag{4}$$

be the residual obtained at the optimal solution y^* of (3), where $y^* \in \mathbb{R}^q$ and $r^* \in \mathbb{R}^{n+1}$. The following statements hold:

i) If $r^* = 0$ then QP (1) is infeasible;
ii) If $r^* \neq 0$ then

$$z^* \triangleq -Q^{-1}\left(c + \frac{1}{\gamma + d'y^*} G'y^*\right) \tag{5}$$

solves QP (1).

Proof: First, by defining $u \triangleq \mathcal{L}z + \mathcal{L}^{-T}c$, we complete the squares in (1a) by substituting $z = \mathcal{L}^{-1}u - Q^{-1}c$ and recasting (1) into the equivalent constrained LDP

$$\min_u \quad \frac{1}{2}\|u\|_2^2 \tag{6a}$$

$$\text{s.t.} \quad Mu \leq d. \tag{6b}$$

i) If the optimal residual $r^* = 0$ in (4), then y^* satisfies the following conditions:

$$\begin{aligned} M'y^* &= 0 \\ d'y^* &= -\gamma \\ y^* &\geq 0. \end{aligned} \tag{7}$$

By Farkas's Lemma [11, p. 201], (7) is equivalent to infeasibility of (6b), and therefore the LDP problem (6) does not admit a solution, and consequently (1).

ii) To prove the second statement, we follow the reasoning in [9, pp. 165–167]. Consider the KKT conditions for problem (3)

$$-\begin{bmatrix} M & d \end{bmatrix}\begin{bmatrix} -M'y^* \\ -d'y^* - \gamma \end{bmatrix} - w^* = 0 \tag{8a}$$

$$(y^*)'w^* = 0 \tag{8b}$$

$$w^*, y^* \geq 0 \tag{8c}$$

where w^* is the optimal dual variable for problem (3). From (8a) we get

$$-\begin{bmatrix} M & d \end{bmatrix} r^* - w^* = 0 \tag{9}$$

and hence the condition $r^* \neq 0$, together with (8b) and (9), imply that

$$\begin{aligned} 0 < (r^*)'r^* &= (r^*)'\begin{bmatrix} -M' \\ -d' \end{bmatrix} y^* - (r^*)'\begin{bmatrix} 0 \\ \gamma \end{bmatrix} \\ &= (w^*)'y^* - \gamma r^*_{n+1} = -\gamma r^*_{n+1} \end{aligned}$$

i.e., $r^*_{n+1} = -d'y^* - \gamma < 0$. By letting

$$u^* \triangleq -\frac{1}{r^*_{n+1}} r^*_{\{1,\dots,n\}} = -\frac{M'y^*}{\gamma + d'y^*} \tag{10}$$

from (8c) and (9), we obtain

$$0 \leq w^* = -\begin{bmatrix} M & d \end{bmatrix} r^* = -r^*_{n+1}\begin{bmatrix} M & d \end{bmatrix}\begin{bmatrix} \frac{r^*_{\{1,\dots,n\}}}{r^*_{n+1}} \\ 1 \end{bmatrix}$$

and hence $-Mu^* + d \geq 0$, or equivalently u^* is feasible for the LDP problem (6). It remains to prove that u^* is also optimal for (6). To this end, consider the remaining KKT conditions of optimality for problem (6)

$$u^* + M'\lambda^* = 0 \tag{11a}$$

$$(\lambda^*)'(Mu^* - d) = 0 \tag{11b}$$

$$\lambda^* \geq 0. \tag{11c}$$

Let

$$\lambda^* \triangleq -\frac{1}{r^*_{n+1}} y^*. \tag{12}$$

By negativity of r^*_{n+1} and nonnegativity of y^* we get (11c). Moreover, by recalling (10) and (4), we get

$$u^* = \frac{1}{r^*_{n+1}} M'y^* = -M'\lambda^*$$

so that also (11a) is satisfied. To prove (11b), we observe that $(\lambda^*)'(Mu^* - d) = -(1/r^*_{n+1})(\lambda^*)'(Mr^*_{\{1,\dots,n\}} + dr^*_{n+1}) = (1/(r^*_{n+1})^2)(y^*)'[M \quad d]^* = -(1/(r^*_{n+1})^2)(y^*)'w^* = 0$ because of (8b) and (9). In conclusion, u^* is the optimal solution of problem (6), and hence the vector z^* defined in (5) solves (1). ■

The following lemma characterizes the relation between the dual variable w of problem (3) and feasibility/optimality of z in (1b).

Lemma 1: Let $y \in \mathbb{R}^q$ such that $d'y \neq -\gamma$, and define $z \in \mathbb{R}^n$ and $w \in \mathbb{R}^q$ as

$$z = -\frac{1}{\gamma + d'y}\mathcal{L}^{-1}M'y - Q^{-1}c \tag{13a}$$

$$w = MM'y + (\gamma + d'y)d \tag{13b}$$

similarly to (5) and (8a), respectively. Then, for all $i \in \{1, \dots, q\}$ and $\epsilon \geq 0$, the condition

$$w_i \geq -(\gamma + d'y)\epsilon \tag{14a}$$

implies that

$$G_i z - g_i \leq \epsilon. \tag{14b}$$

Moreover, given the dual problem

$$\max_{\lambda \geq 0} \Psi(\lambda), \ \Psi(\lambda) \triangleq -\frac{1}{2}\lambda'GQ^{-1}G'\lambda - d'\lambda - \frac{1}{2}c'Q^{-1}c \tag{15}$$

of QP (1) it holds that

$$V(z) = \Psi(\lambda) + \frac{1}{(\gamma + d'y)^2} w'y \tag{16}$$

and hence, if

$$w'y = 0 \tag{17}$$

$$\lambda = \frac{1}{\gamma + d'y} y \tag{18}$$

is such that $\lambda \geq 0$, then z is (super-)optimal, in that

$$V(z) \leq V(z^*). \tag{19}$$

Proof: The fact that condition (14a) implies (14b) simply follows by left-multiplying z as defined in (13a) by G, subtracting g, and recalling (2). To prove (19), consider that by (13a) the primal cost $V(z) = (1/2)\lambda'MM'\lambda - (1/2)c'Q^{-1}c = \Psi(\lambda) + \lambda'MM'\lambda + d'\lambda = \Psi(\lambda) + (1/(\gamma + d'y)^2)w'y$, which proves (16). For all dual feasible $\lambda \geq 0$, $\Psi(\lambda)$ is a lower bound on the optimal primal cost $V(z^*)$. Together with condition (17), this implies (19). ■

Remark 1: In case $r^*_{n+1} = -(\gamma + d'y^*) \neq 0$, the solution λ^* of the dual QP problem (15) is given by (12). In case $\gamma = 1$, Problem (3) can be rewritten as the following QP

$$\min_{y} \quad \frac{1}{2} y' \left(GQ^{-1}G' + dd'\right) y + d'y \tag{20a}$$

$$\text{s.t.} \quad y \geq 0. \tag{20b}$$

Note that, although closely related, Problems (15) and (20) are different because of the extra term $(1/2)y'dd'y$.

The alternative QP formulation (20) suggests the following corollary of Theorem 1.

Corollary 1: Given the QP (1) with $Q = Q' \succ 0$, the convex QP problem (20) always admits an optimal solution y^*. If the optimal cost of (20) is $-(1/2)$ then (1) is infeasible, otherwise z^* defined in (5) is the optimal solution of (1).

Proof: The cost function (20a) is equal to $(1/2)y'(G'Q^{-1}G + dd')y + d'y + (1/2) - (1/2) = (1/2)\|\left[\begin{smallmatrix} -M' \\ -d' \end{smallmatrix}\right]y - \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]\|_2^2 - (1/2) \geq -(1/2)$, so (20) always admits a solution y^*. If the optimal cost is $-(1/2)$ then the residual defined in (4) is zero and by Theorem 1 the QP problem (1) is infeasible. Otherwise, y^* solves (3) and therefore, by Theorem 1, z^* in (5) solves (1). ■

The following Corollary 2 of Theorem 1 provides a further criterion to detect infeasibility of the QP problem (1).

Corollary 2: For any subset of constraint indices $\mathcal{P} \subseteq \{1, \ldots, q\}$ and $\gamma > 0$, if the solution $s_{\mathcal{P}}$ of the least-squares problem

$$s_{\mathcal{P}} = \arg\min_{z_{\mathcal{P}}} \left\| \begin{bmatrix} M'_{\mathcal{P}} \\ d'_{\mathcal{P}} \end{bmatrix} z_{\mathcal{P}} + \begin{bmatrix} 0 \\ \gamma \end{bmatrix} \right\|_2^2 \tag{21}$$

is such that the squared residual $\|M'_{\mathcal{P}} s_{\mathcal{P}}\|^2 + (\gamma + d'_{\mathcal{P}} s_{\mathcal{P}})^2 = 0$, then the QP problem (1) is infeasible.

Proof: As proved in part i) of Theorem 1, if such a residual is zero then the polyhedron $\mathcal{C} \stackrel{\Delta}{=} \{u \in \mathbb{R}^n : \ M_{\mathcal{P}} u \leq d_{\mathcal{P}}\}$ is empty. Hence, also the polyhedron $\{u \in \mathbb{R}^n : \ Mu \leq d\} = \mathcal{C} \cap \{u \in \mathbb{R}^n : M_{\{1,\ldots,q\}\setminus\mathcal{P}} u \leq d_{\{1,\ldots,q\}\setminus\mathcal{P}}\}$ is empty, and therefore problem (1) is infeasible. ■

We now exploit Theorem 1, Corollary 2, and Lemma 1 to derive an active-set method to solve the QP problem (1) and its dual (15) within a feasibility tolerance ϵ. The method is described in Algorithm 1 and extends the well-known and simple, yet very effective, NNLS solution algorithm described in [9, p. 161].

Regarding the choice of the parameter $\gamma > 0$, one can use $\gamma = 1$ as in [9]. Although the particular choice of γ is not critical, we observed better numerical conditioning by adapting γ during the iterations to the value

$$\gamma = 1 + \|d_{\mathcal{P}}\|_1. \tag{22}$$

Algorithm 1 QP solver based on NNLS

Input: Inverse Cholesky factor $\mathcal{L}^{-1}$ of Q, $M = G\mathcal{L}^{-1}$, vectors c and g, feasibility tolerance $\epsilon \geq 0$.

1. $v \leftarrow \mathcal{L}^{-T} c$;
2. $d \leftarrow g + Mv$;
3. $\mathcal{P} \leftarrow \emptyset$; $y \leftarrow 0$; $\gamma \leftarrow 1$;
4. $w \leftarrow M(M'_{\mathcal{P}} y_{\mathcal{P}}) + (\gamma + d'_{\mathcal{P}} y_{\mathcal{P}})d$;
5. **if** $w_i \geq -(\gamma + d'_{\mathcal{P}} y_{\mathcal{P}})\epsilon, \forall i \in \{1, \ldots, q\}$, **or** $\mathcal{P} = \{1, \ldots, q\}$ or $\|M'_{\mathcal{P}} y_{\mathcal{P}}\|_2^2 + (\gamma + d'_{\mathcal{P}} y_{\mathcal{P}})^2 = 0$ **then go to** Step 13;
6. $i \leftarrow \arg\min_{i \in \{1,\ldots,m\}\setminus\mathcal{P}} w_i$, $\mathcal{P} \leftarrow \mathcal{P} \cup \{i\}$; $\gamma \leftarrow \gamma + |d_i|$;
7. $s_{\mathcal{P}} \leftarrow$ solution of LS problem (21), $s_{\{1,\ldots,q\}\setminus\mathcal{P}} \leftarrow 0$;
8. **if** $s_{\mathcal{P}} \geq 0$ **then** $y \leftarrow s$ and **go to** Step 4;
9. $j \leftarrow \arg\min_{h \in \mathcal{P}:\ s_h \leq 0} \{y_h/(y_h - s_h)\}$;
10. $y \leftarrow y + (y_j/(y_j - s_j))(s - y)$;
11. $\mathcal{I} \leftarrow \{h \in \mathcal{P} : y_h = 0\}$, $\mathcal{P} \leftarrow \mathcal{P} \setminus \mathcal{I}$; $\gamma \leftarrow \gamma - \|d_{\mathcal{I}}\|_1$;
12. **go to** Step 7;
13. **if** the residual in Step 7 is nonzero **then** $\lambda^* \leftarrow -1/(\gamma + d'_{\mathcal{P}} y_{\mathcal{P}})y$; $u^* \leftarrow M'_{\mathcal{P}} \lambda^*_{\mathcal{P}}$; $z^* \leftarrow \mathcal{L}^{-1}(u^* - v)$; **otherwise** QP problem (1) is infeasible;
14. **end**.

Output: Primal solution z^* solving (1) and dual solution λ^* solving (15), or infeasibility status.

As proved in [9], for $\epsilon = 0$ Algorithm 1 convergences after a finite number of steps, providing the optimal solution vector z^*. The same holds for $\epsilon > 0$ sufficiently small. In general, for $\epsilon > 0$, Algorithm 1 may stop even earlier, providing a vector z^* that, by virtue of Lemma 1, satisfies $G_i z^* - g_i \leq \epsilon, \forall i \in \{1, \ldots, q\}$. Note that in case Step 13 is executed with $\mathcal{P} = \emptyset$ then $\lambda^* = 0$, $u^* = 0$, and $z^* = -Q^{-1}c$.

The LS problem (21) solved at Step 7 corresponds to the unconstrained quadratic optimization problem

$$s_{\mathcal{P}} = \arg\min \frac{1}{2} z'_{\mathcal{P}} \begin{bmatrix} M_{\mathcal{P}} & d_{\mathcal{P}} \end{bmatrix} \begin{bmatrix} M'_{\mathcal{P}} \\ d'_{\mathcal{P}} \end{bmatrix} z_{\mathcal{P}} + \begin{bmatrix} M_{\mathcal{P}} & d_{\mathcal{P}} \end{bmatrix} \begin{bmatrix} 0 \\ \gamma \end{bmatrix} \tag{23}$$

or, equivalently, to solving the symmetric linear system

$$(M_{\mathcal{P}} M'_{\mathcal{P}} + d_{\mathcal{P}} d'_{\mathcal{P}}) s_{\mathcal{P}} = -\gamma d_{\mathcal{P}}. \tag{24}$$

Hence, each time Step 4 is executed, we have that $y_{\mathcal{P}} = s_{\mathcal{P}} \geq 0$, $y_i = 0, \forall i \in \{1, \ldots, q\} \setminus \mathcal{P}$, and hence

$$w'y = y'_{\mathcal{P}} w_{\mathcal{P}} = s'_{\mathcal{P}} \left[(M_{\mathcal{P}} M'_{\mathcal{P}} + d_{\mathcal{P}} d'_{\mathcal{P}}) s_{\mathcal{P}} + \gamma d_{\mathcal{P}}\right] = 0.$$

If in addition $d'_{\mathcal{P}} y_{\mathcal{P}} > -\gamma$, then λ defined as in (18) is nonnegative, so that by Lemma 1 (super-)optimality of z as in (13a) follows after each execution of Step 4, and in particular of z^* when Algorithm 1 terminates with a nonzero residual.

Remark 2: Regarding computation and memory requirements of Algorithm 1, the largest number of operations is spent at Steps 4 and 7, where the latter involves solving an unconstrained least-squares problem with $\#\mathcal{P}$ variables and $n + 1$ inequalities. When used in the context of MPC of linear time-invariant systems, only c, g in (1) may change on-line, while matrices $\mathcal{L}^{-1}$ and $M = G\mathcal{L}^{-1}$ can be precomputed offline.

Remark 3: Algorithm 1 is cold-started at Step 3 from the null combination $\mathcal{P}$ of active constraints. Conditions for warm-starting the algorithm with $\mathcal{P} \neq \emptyset$ and a corresponding suitable value of y have been proposed in [10]. Alternatively, as suggested in [12], one can very effectively warm start a NNLS algorithm like Algorithm 1 by running first a finite number of iterations of an accelerated gradient-projection algorithm, like the one proposed in [7].

III. Recursive LDL^T Decomposition

Solving the LS problem (21) at Step 7 is the most time-consuming operation of Algorithm 1. The symmetric linear system (24) can be solved by a variety of techniques, such as QR factorization, Conjugate Gradient (CG) or Cholesky factorization. Here, to solve (21), we adopt instead the LDL^T factorization of the left-hand side matrix of (24)

$$LDL' = \begin{bmatrix} M_{\mathcal{P}} & d_{\mathcal{P}} \end{bmatrix} \begin{bmatrix} M'_{\mathcal{P}} \\ d'_{\mathcal{P}} \end{bmatrix} \tag{25}$$

where L is a lower-triangular matrix of dimension $\#\mathcal{P}$ with all ones on its diagonal and D is a diagonal matrix. The factorization (25) can

be computed for example using [13, Algorithm 4.1.1]. The drawback in using such a classical LDL^T decomposition procedure is that it assumes that all diagonal elements in D are nonzero, i.e., that the matrix to be factorized in (25) is full rank. Moreover, since the set $\mathcal{P}$ changes incrementally by adding one component (Step 6) or removing one or more components (Step 11), one should take advantage of the way $\mathcal{P}$ is modified incrementally to compute the LDL^T factorization in (25) more efficiently in a recursive way.

Iterative QR factorization methods to solve (21) have been proposed in [9, Ch. 24]. The following lemma provides a way of computing instead the LDL^T transformation recursively of a generic matrix AA', $A \in \mathbb{R}^{p\times m}$, with L lower triangular with unit diagonal and D diagonal, with $D_{ii} \geq 0$, which also covers the case in which AA' is rank-deficient.

Theorem 2: Let $A \in \mathbb{R}^{p\times n}$. Then there exists a lower triangular matrix $L^p \in \mathbb{R}^{p\times p}$ with unit diagonal and a diagonal matrix $D^p \in \mathbb{R}^{p\times p}$ with $D_{ii} \geq 0$, satisfying the following recursions:

$$L^1 = 1, \;\; D^1 = \|A_1\|_2^2 \tag{26a}$$

$$L^{k+1} = \begin{bmatrix} L^k & 0 \\ (\ell^{k+1})' & 1 \end{bmatrix} \tag{26b}$$

$$D^{k+1} = \begin{bmatrix} D^k & 0 \\ 0 & d^{k+1} \end{bmatrix} \tag{26c}$$

$$(L^k D^k)\ell^{k+1} = A_{1:k}A'_{k+1} \tag{26d}$$

$$d_{k+1} = \|A_{k+1}\|_2^2 - (\ell^{k+1})' D^k \ell^{k+1} \tag{26e}$$

for $k = 1, \ldots, p-1$, and such that

$$A_{1:k}A'_{1:k} = L^k D^k (L^k)', \quad \forall k = 1, \ldots, p. \tag{27}$$

Proof: We prove the theorem by induction on k. For $k = 1$, (27) trivially follows from (26a). Assume that (26)–(27) are satisfied for a given k and consider matrix $A_{1:k+1}$. Consider the linear system

$$A_{1:k}A'_{1:k}x^{k+1} = A_{1:k}A'_{k+1} \tag{28}$$

which is the condition for vector $x^{k+1} \in \mathbb{R}^k$ to be an optimal solution of the LS problem

$$\min_{x\in\mathbb{R}^k} \left\| A'_{1:k}x - A'_{k+1} \right\|_2^2. \tag{29}$$

Since (29) is always solvable, a vector x^{k+1} satisfying (28) exists, and hence a vector $\ell^{k+1} = (L^k)' x^{k+1}$ exists that solves (26d). For all components $D_{ii} = 0$ of D^k, the product $D_{ii}\ell_i^{k+1} = 0$ for any value of ℓ_i^{k+1}, in particular we can set $\ell_i^{k+1} = 0$, $i = 1, \ldots, k$. System (26d), by left-multiplying by $(L^k)^{-1}$, can be rewritten also as

$$D^k \ell^{k+1} = c^{k+1} \tag{30}$$

where $c^{k+1} \in \mathbb{R}^k$ is the unique solution of $L^k c^{k+1} = A_{1:k}A'_{k+1}$. Since (26d) is solvable, (30) is also solvable, and hence necessarily $c_i^{k+1} = 0$ for all i such that $D_{ii} = 0$. In conclusion, the following vector ℓ^{k+1} defined as

$$\ell_i^{k+1} = \begin{cases} \frac{c_i^{k+1}}{D_{ii}}, & \text{if } D_{ii} \neq 0 \\ 0, & \text{if } D_{ii} = 0 \end{cases} \tag{31}$$

is a solution of (26d). By defining d^{k+1} as in (26e), we get

$$\begin{aligned} &L^{k+1}D^{k+1}(L^{k+1})' \\ &\quad = \begin{bmatrix} L^k D^k (L^k)' & L^k D^k \ell^{k+1} \\ (\ell^{k+1})' D^k (L^k)' & \ell^{k+1})' D^k (L^k)\ell^{k+1} + d^{k+1} \end{bmatrix} \\ &\quad = \begin{bmatrix} A_{1:k}A'_{1:k} & A_{1:k}A'_{k+1} \\ A_{k+1}A'_{1:k} & A_{k+1}A'_{k+1} \end{bmatrix} = A_{1:k+1}A'_{1:k+1} \end{aligned}$$

which proves that (27) is satisfied also for $k+1$. ■

Corollary 3: Let $A \in \mathbb{R}^{p\times n}$ and let $LDL' = AA'$ be a decomposition of AA', with $L \in \mathbb{R}^{p\times p}$ lower triangular with unit diagonal and $D \in \mathbb{R}^{p\times p}$ diagonal, $D_{ii} \geq 0, \forall i = 1, \ldots, p$. Then, for any $b \in \mathbb{R}^n$, a solution $x \in \mathbb{R}^p$ of the least-squares problem

$$\min_x \|A'x - b\|_2^2 \tag{32}$$

is given by solving

$$Lc = Ab \tag{33a}$$

$$\ell_i = \begin{cases} \frac{c_i}{D_{ii}} & \text{if } D_{ii} \neq 0 \\ 0 & \text{if } D_{ii} = 0 \end{cases}, \; i = 1, \ldots, p \tag{33b}$$

$$L'x = \ell. \tag{33c}$$

Proof: The proof immediately follows by substituting $A_{1:k}$ with A, A'_{k+1} with b, c^{k+1} with c, and ℓ^{k+1} with ℓ in the proof of Theorem 2. ■

Theorem 2, through (26b)–(26e), provides a way of updating the LDL^T decomposition when a row A_{k+1} is added. The following Lemma 2 tackles the case in which a row is deleted from A.

Lemma 2: Given the LDL^T decomposition

$$\begin{bmatrix} A_1 \\ a' \\ A_2 \end{bmatrix} \begin{bmatrix} A'_1 & a & A'_2 \end{bmatrix} = LDL' \tag{34}$$

where

$$L = \begin{bmatrix} L_1 & 0 & 0 \\ \ell & 1 & 0 \\ L_3 & L_4 & L_2 \end{bmatrix}, \; D = \begin{bmatrix} D_1 & 0 & 0 \\ 0 & \delta & 0 \\ 0 & 0 & D_2 \end{bmatrix} \tag{35}$$

$L \in \mathbb{R}^{p\times p}$ lower triangular with unit diagonal, $D \in \mathbb{R}^{p\times p}$ diagonal, $D_{ii} \geq 0, \forall i = 1, \ldots, p$, an LDL^T decomposition

$$L_- D_- L'_- = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \begin{bmatrix} A'_1 & A'_2 \end{bmatrix} \tag{36}$$

with $L_- \in \mathbb{R}^{(p-1)\times(p-1)}$ lower triangular with unit diagonal and $D_- \in \mathbb{R}^{(p-1)\times(p-1)}$ diagonal, $D_{-ii} \geq 0, \forall i = 1, \ldots, p-1$, is given by

$$L_- = \begin{bmatrix} L_1 & 0 \\ L_3 & \bar{L}_2 \end{bmatrix}, \; D_- = \begin{bmatrix} D_1 & 0 \\ 0 & \bar{D}_2 \end{bmatrix} \tag{37}$$

where $\bar{L}_2$, $\bar{D}_2$ provide the LDL^T decomposition $\bar{L}_2\bar{D}_2\bar{L}'_2 = \delta L_4 L'_4 + L_2 D_2 L'_2$, with $\bar{L}_2$ lower triangular with unit diagonal and $\bar{D}_2$ diagonal.

Proof: Since $\delta L_4 L'_4 + L_2 D_2 L'_2 = \begin{bmatrix} L_2 D_2^{1/2} & \delta^{1/2} L_4 \end{bmatrix} \cdot \begin{bmatrix} D_2^{1/2} L'_2 \\ \delta^{1/2} L'_4 \end{bmatrix}$, by Theorem 2 an LDL^T decomposition $\bar{L}_2\bar{D}_2\bar{L}'_2$ of $\delta L_4 L'_4 + L_2 D_2 L'_2$ exists such that $\bar{L}_2$ is lower triangular with unit diagonal and $\bar{D}_2$ is diagonal. Since $LDL' = \begin{bmatrix} L_1 D_1 & 0 & 0 \\ \star & \star & \star \\ L_3 D_1 & \delta L_4 & L_2 D_2 \end{bmatrix} \cdot \begin{bmatrix} L'_1 & \star & L'_3 \\ 0 & \star & L'_4 \\ 0 & \star & L'_2 \end{bmatrix} = \begin{bmatrix} L_1 D_1 L'_1 & \star & L_1 D_1 L'_3 \\ \star & \star & \star \\ L_3 D_1 L'_1 & \star & L_3 D_1 L'_3 + \delta L_4 L'_4 + L_2 D_2 L'_2 \end{bmatrix} = \begin{bmatrix} A_1 A'_1 & \star & A_1 A'_2 \\ \star & \star & \star \\ A_2 A'_1 & \star & A_2 A'_2 \end{bmatrix}$ and $L_- D_- L'_- = \begin{bmatrix} L_1 D_1 & 0 \\ L_3 D_1 & \bar{L}_2\bar{D}_2 \end{bmatrix} \cdot \begin{bmatrix} L'_1 & L'_3 \\ 0 & \bar{L}'_2 \end{bmatrix} = \begin{bmatrix} L_1 D_1 L'_1 & L_1 D_1 L'_3 \\ L_3 D_1 L'_1 & L_3 D_1 L'_3 + \bar{L}_2\bar{D}_2\bar{L}'_2 \end{bmatrix}$, Eq. (36) is satisfied by inspection. ■

Note that one does not need to take square roots of D_2 and δ as $\bar{L}_2$, $\bar{D}_2$ can be determined as in Theorem 2 by replacing $A_{1:k}A'_{k+1}$ in (26d) with $(\delta(L_4)_{1:k}(L'_4)_{k+1} + (L_2)_{1:k}D_2(L'_2)_{k+1})$ and $\|A_{k+1}\|_2^2$ in (26e) with $(\delta(L_4)_{k+1}(L'_4)_{k+1} + (L_2)_{k+1}D_2(L'_2)_{k+1})$.

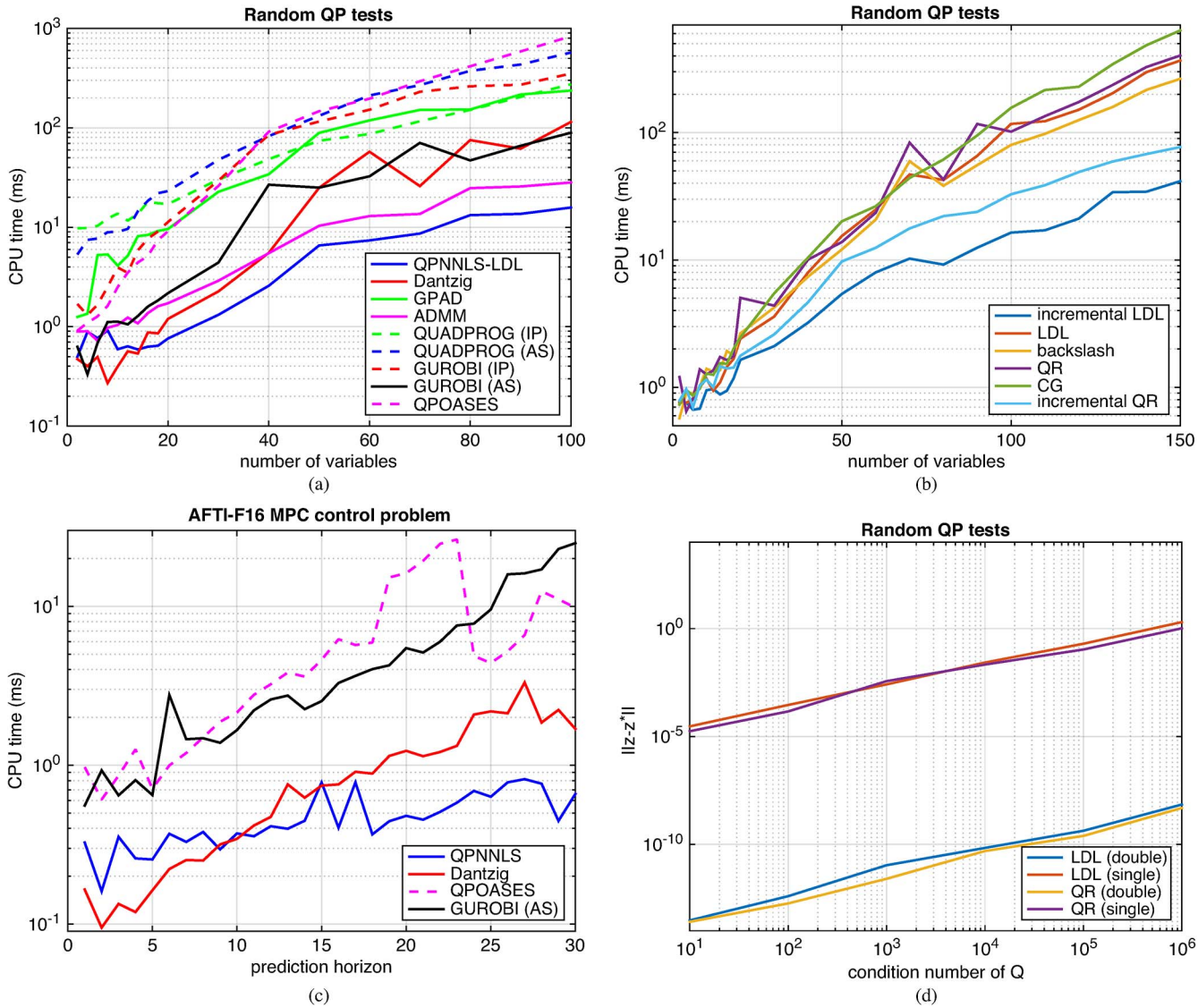

Fig. 1. Results of numerical tests. (a) Comparison of different QP solvers on random QP problems with n variables, $q = 5n$ constraints, condition number $\kappa = 10^4$ of Q: worst-case measured CPU time on 100 instances for each n. (b) Comparison of different methods to solve the LS problem (21) at Step 7 of Algorithm 1 on random QP problems with n variables, $q = 5n$ constraints, condition number $\kappa = 10^4$ of Q. For each n the time reported is the worst-case of 100 instances. (c) Worst-case CPU time (ms) during the simulation for the AFTI-F16 aircraft problem as a function of the prediction horizon. (d) Worst-case difference $\|z - z^*\|_2$ over 100 random QP problems ($n = 10$, $q = 50$) as a function of κ.

IV. Numerical Experiments

A. Quadratic Programs

We compare the performance of the QP solver developed in the previous section (labeled as QPNNLS) against different state-of-the-art solvers.[1]

[1]We compared against: 1) Dantzig's active set algorithm [4, Sect. 24–4] applied to solve the dual problem (15) (Dantzig), 2) the accelerated gradient projection method of [7] (GPAD) applied to the dual QP (15) with diagonal preconditioning [14, Section 2.3.1], 3) the alternative direction method of multipliers [8] (ADMM), run with $\rho = 0.2$ for a fixed number $K = 300$ iterations in all instances (with checking of feasibility and optimality criteria disabled for speedup), 4) QUADPROG's interior point method, 5) QUADPROG's active set method of the Optimization Toolbox for MATLAB V7.1, 6) GUROBI's v6.0 [15] interior-point method, 7) GUROBI's dual simplex method, and 8) QPOASES [5]. Algorithm 1 and methods 1, 2, 3 have been implemented in Embedded MATLAB code and compiled. The numerical experiments were obtained on a Macbook Pro 3 GHz Intel Core i7 with 16GB RAM running MATLAB R2014b.

Fig. 1(a) shows the worst-case CPU time obtained on random feasible QP problems with n variables $q = 5n$ constraints, and condition number $\kappa = 10^4$ of the primal Hessian Q. For each n, the reported CPU time is the worst-case over 100 QP instances. The QPNNLS Algorithm 1 is executed by updating the LDL^T decomposition recursively in accordance with Theorem 2 and Lemma 2. In all instances, Algorithm 1 and Dantzig's dual QP method execute the same number of iterations.

Fig. 1(b) compares different methods to solve the LS problem (21) at Step 7 of Algorithm 1, namely: 1) incremental LDL^T updates based on Theorem 2 and Lemma 2; 2) computation of the LDL^T decomposition from the original data M, d from scratch at every iteration; 3) MATLAB's backslash built-in function to solve linear systems in a least-square sense; 4) MATLAB's "economy-size" QR decomposition of matrix $\begin{bmatrix} M'_{\mathcal{P}} \\ d'_{\mathcal{P}} \end{bmatrix}$, (5) the basic CG method without preconditioning of [16, Algorithm 5.2], with stopping tolerance equal to 10^{-8}, (6) the incremental QR factorization Method 1 described in [9, Ch. 24].

B. MPC Problems

In order to test the QP solver of Algorithm 1 in an MPC problem, we consider the AFTI-F16 aircraft control example described in [17], under the settings of the demo `aft16.m` in the Hybrid Toolbox for MATLAB [18]. We consider a prediction and control horizon of N steps, with hard input and soft constraints enforced over the prediction horizon, leading to a QP problem (1) with $n = 2N + 1$ optimization variables (the extra variable is needed to soften output constraints, that is weighted with a penalty of 10^4) and $q = 4N + 2(N - 1)$ constraints. The reference trajectory is 0 for the constrained output, and switches between ± 10 deg for the second output, over a 12 s simulation interval.

We compare Algorithm 1 with incremental LDL^T updates against Dantzig's active set algorithm [4, Sect. 24-4] applied to solve the dual problem (15), GUROBI's dual simplex method, and QPOASES [5] with warm starting from the active set of the previous optimal solution.[2] The worst-case CPU time encountered during the simulation for different prediction horizons N is plotted in Fig. 1(c).[3]

C. Numerical Robustness

We test the robustness of Algorithm 1 by comparing results obtained in single and double precision as a function of the condition number κ of matrix Q. Fig. 1(d) shows the worst-case norm of the difference obtained over 100 random tests of the obtained solution with respect to QUADPROG's double-precision solution.

Numerical experiments (not reported here) have shown that robustness can be improved by changing the way the solution is reconstructed in Step 13 of Algorithm 1. For example, one can calculate the solution $\lambda^*_{\mathcal{P}}$ of (15) by solving the symmetric linear system $M_{\mathcal{P}} M'_{\mathcal{P}} \lambda^*_I = -d_{\mathcal{P}}$ via QR factorization of $M'_{\mathcal{P}}$.

V. Conclusion

This technical note has introduced a new active set method for solving QP problems. The method is attractive in several application domains where fast and simple to code QP solvers are required, such as in embedded MPC applications.

[2]For QPOASES we do not consider the CPU time spent at time $t = 0$, which is the largest one of the entire simulation, in order not to penalize the benefits of warm starting provided by the method.

[3]The QP solvers of QUADPROG (both the interior-point and the dual simplex methods) failed in solving many of the QP problems generated during the simulations, and ADMM could not provide feasible solutions z with $\max(Gz - g) \leq 10^{-2}$ even with $K = 3000$ iterations, so they are not considered in the comparison.

References

[1] A. Bemporad and P. Patrinos, "Simple and certifiable quadratic programming algorithms for embedded linear model predictive control," in *Proc. 4th IFAC Nonlinear Model Predictive Control Conf.*, F. A. M. Lazar, Ed., 2012, pp. 14–20.

[2] A. Bemporad, "A multiparametric quadratic programming algorithm with polyhedral computations based on nonnegative least squares," *IEEE Trans. Autom. Control*, vol. 60, no. 11, pp. 2892–2903, Nov. 2015.

[3] A. Bemporad, A. Oliveri, T. Poggi, and M. Storace, "Ultra-fast stabilizing model predictive control via canonical piecewise affine approximations," *IEEE Trans. Autom. Control*, vol. 56, no. 12, pp. 2883–2897, Dec. 2011.

[4] G. Dantzig, *Linear Programming and Extensions*. Princeton, NJ, USA: Princeton Univ. Press, 1998.

[5] H. Ferreau, H. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit MPC," *Int. J. Robust Nonlin. Control*, vol. 18, no. 8, pp. 816–830, 2008.

[6] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 2, pp. 267–278, 2010.

[7] P. Patrinos and A. Bemporad, "An accelerated dual gradient-projection algorithm for embedded linear model predictive control," *IEEE Trans. Autom. Control*, vol. 59, no. 1, pp. 18–33, Jan. 2014.

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[9] C. Lawson and R. Hanson, *Solving Least Squares Problems*, vol. 161. Philadelphia, PA, USA: SIAM, 1974.

[10] A. Bemporad, "Solving mixed-integer quadratic programs via nonnegative least squares," in *Proc. 5th IFAC Conf. NMPC*, Sevilla, Spain, 2015, pp. 73–79.

[11] R. Rockafellar, *Convex Analysis*. Princeton, NJ, USA: Princeton Univ. Press, 1970.

[12] A. Bemporad and M. Paggi, "Optimization algorithms for the solution of the frictionless normal contact between rough surfaces," *Int. J. Solids Struct.*, vol. 69–70, pp. 94–105, Sep. 2015.

[13] G. Golub and C. V. Loan, *Matrix Computations*, 4th ed. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 2013.

[14] D. Bertsekas, *Nonlinear Programming*, 2nd ed. New York, NY, USA: Athena Scientific, 1999.

[15] Gurobi Optimization, Inc., Gurobi Optimizer Reference Manual, 2014. [Online]. Available: http://www.gurobi.com

[16] J. Nocedal and S. Wright, *Numerical Optimization*. New York, NY, USA: Springer-Verlag, 1999.

[17] A. Bemporad, A. Casavola, and E. Mosca, "Nonlinear control of constrained linear systems via predictive reference management," *IEEE Trans. Autom. Control*, vol. AC-42, no. 3, pp. 340–349, 1997.

[18] A. Bemporad, Hybrid Toolbox—User's Guide, Dec. 2003. [Online]. Available: http://cse.lab.imtlucca.it/bemporad/hybrid/toolbox