Preprints, 6th IFAC Conference on Nonlinear Model Predictive Control
Madison, WI, USA, August 19-22, 2018

A Numerically Robust Mixed-Integer Quadratic Programming Solver for Embedded Hybrid Model Predictive Control

Alberto Bemporad * **Vihangkumar V. Naik** *

* *IMT School for Advanced Studies Lucca, Italy*
(email: {alberto.bemporad, vihangkumar.naik}@imtlucca.it).

Abstract: The deployment of hybrid model predictive control (MPC) in practical applications requires primarily an efficient and robust on-line Mixed-Integer Quadratic Programming (MIQP) solver that runs in real time. In this paper we propose a new algorithm for solving MIQP problems which is particularly tailored to solve small-scale MIQPs, such as those that arise in embedded hybrid MPC applications. The algorithm couples a branch and bound (B&B) scheme with a recently proposed numerically robust Quadratic Programming (QP) solver based on nonnegative least squares (NNLS) and proximal-point iterations. The resulting MIQP solver supports positive semidefinite Hessian matrices, often appearing in hybrid MPC formulations, and warm starts with respect to both binary and real variables. We show that the speed of execution of our solver is comparable with state-of-the-art commercial solvers, while it is relatively simple to code in an embedded control system.

Keywords: Mixed-integer quadratic programming, quadratic programming, active-set methods, nonnegative least squares, model predictive control, hybrid systems.

1. INTRODUCTION

Since hybrid Model Predictive Control (MPC) was introduced almost two decades ago (Bemporad and Morari, 1999), it has attracted a lot of attention in both academia and industry. The widespread popularity of hybrid systems is mainly due to their ability of modeling a very large spectrum of practical systems where physical processes coexist with switching dynamics, discrete actuators, logic rules, and constraints on system variables. MPC based on hybrid models provides an optimized way of controlling such systems. However, the real-time implementation of hybrid MPC on embedded platforms is still a challenge, as it requires solving a mixed-integer quadratic programming (MIQP) problem at every sample time in a numerically efficient and robust manner.

For this reason, several solution approaches to solve MIQP problems in an embedded control setting have been investigated. In "desktop" applications the numerical package is used to solve (sometimes large-scale) MIQPs with large memory and computing resources available and no stringent limits on execution time. Conversely, in embedded applications, severe restrictions on CPU/memory/time resources pose considerable differences: the number of variables (especially binary variables) should be small, the code must be simple and library-free, the algorithm must be numerically robust also when executed in single precision arithmetic.

As suboptimal solutions are often enough for practical control purposes, several heuristic approaches were introduced to *approximately* solve the mixed-integer problem in a very quick manner. The *feasibility pump* (FP) heuristic was proposed for mixed-integer linear constraints (Fischetti et al., 2005), and for binary and general-integers (Bertacco et al., 2007). Recently, heuristic methods were used with the alternating direction method of multipliers (ADMM) (Takapoui et al., 2016), and accelerated dual gradient projection (GPAD) (Naik and Bemporad, 2017) in order to approximately solve the MIQP problem. However, heuristic approaches are not guaranteed to converge to a feasible solution. Under some assumptions, an approach for finding suboptimal solutions based on operator splitting with convergence guarantees was presented in (Frick et al., 2016).

The Branch and Bound (B&B) approach (Floudas, 1995) to solve MIQP problems to optimality has been used in combination with various algorithms to solve the QP relaxations including dual active-set methods (Axehill and Hansson, 2006), interior-point methods (Frick et al., 2015), an active-set method based on nonnegative least squares (NNLS) (Bemporad, 2015), GPAD (Naik and Bemporad, 2017), and ADMM based on the OSQP solver (Stellato et al., 2018). The algorithms in (Bemporad, 2015), (Naik and Bemporad, 2017), (Stellato et al., 2018) are relatively simple to code, and are demonstrated to be quite effective for small-scale problems. The first two approaches however require a strictly convex objective function, with their numerical performance deteriorating with the condition number of the Hessian matrix. While regularizing the cost function would improve numerical robustness, it would bias the solution away from optimality.

In this paper we propose an MIQP algorithm based on B&B and the numerically robust solver for positive semidefinite QPs recently introduced in Bemporad (2018). The QP solver is based on an active-set method for solving nonnegative least-squares (NNLS) problems, but differently from (Bemporad, 2016) it uses proximal-point iterations that greatly improve robustness without worsening execution time. Compared to the MIQP approach in (Bemporad, 2015), the numerical robustness of the resulting MIQP solver is largely improved and can handle positive semidefinite Hessian matrices. Moreover, it handles equality constraints, supports warm starting of QP relaxations from parent nodes in the search tree, and more general bilateral inequality constraints.

In addition, we include a warm start strategy for binary variables within the B&B setup, which allows prioritizing the exploration of a specific section of the binary tree. This is especially useful in hybrid MPC (Bemporad et al., 1999), moving-horizon estimation (Ferrari-Trecate et al., 2002) and piecewise affine regression (Naik et al., 2017; Mejari et al., 2018), where a good initial guess for the binary variables is available by shifting the optimal solution computed at the previous sample step.

We will show in numerical experiments that the resulted approach gives quite comparable results against well-known commercial solvers when tested on small-scale MIQPs, such as those arising in embedded hybrid MPC applications.

2. PROBLEM FORMULATION

Consider the Mixed-Integer Quadratic Programming (MIQP) problem

$$\begin{aligned} \min_z \quad & V(z) \triangleq \frac{1}{2} z'Qz + c'z && \text{(1a)}\\ \text{s.t.} \quad & \ell \le Az \le u && \text{(1b)}\\ & Gz = g && \text{(1c)}\\ & \bar{A}_i z \in \{\bar{\ell}_i, \bar{u}_i\},\ i = 1, \dots, p && \text{(1d)} \end{aligned}$$

where $z \in \mathbb{R}^n$ is the vector of optimization variables, $Q \in \mathbb{R}^{n \times n}$ is the positive semidefinite Hessian matrix, $Q \succeq 0$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $\ell, u \in \mathbb{R}^m$ describe the linear inequality constraints, $\ell \le u$, and $G \in \mathbb{R}^{q \times n}$, $g \in \mathbb{R}^q$ the linear equality constraints. Binary equality constraints are described by $\bar{A} \in \mathbb{R}^{p \times n}$, $\bar{\ell}, \bar{u} \in \mathbb{R}^p$, $p \le m$. Binary variables $z_i \in \{0, 1\}$ are a special case of (1d), corresponding to setting $\bar{A}_i$ as the i-th row of the identity matrix, $\bar{\ell}_i = 0$, and $\bar{u}_i = 1$. The QP relaxation of problem (1) is obtained by replacing the integrality constraint (1d) with the following linear inequality constraint of the form (1b) as

$$\bar{\ell}_i \le \bar{A}_i z \le \bar{u}_i,\ i = 1, \dots, p. \qquad \text{(1e)}$$

Hybrid systems can efficiently be modeled using the Mixed Logical Dynamical (MLD) framework (Bemporad and Morari, 1999). The tool HYSDEL (Torrisi and Bemporad, 2004) allows one to describe a hybrid dynamical model and get the equivalent MLD transformation. The hybrid MPC problem based on MLD models can be recast as a MIQP problem of the form (1a)-(1d). Very often the associated Hessian matrix Q is only positive semidefinite. The MIQP solution approach described in this paper does not require regularizing Q to make the problem strictly convex.

3. SOLUTION OF QP RELAXATIONS

Solving problem (1a)-(1d) using B&B requires solving QP relaxations, in which constraints (1d) are either relaxed to inequality constraints $\bar{\ell}_i \le \bar{A}_i z \le \bar{u}_i$ or to equality constraints $A_i z = \bar{\ell}_i$ or $A_i z = \bar{u}_i$. To solve QP relaxations of the form (1a)-(1c), we slightly extend the solver in (Bemporad, 2018) to handle bilateral constraints of the form (1b), as summarized in Algorithm 1. It is based on the idea of solving proximal-point iterations, where each iteration consists of solving a regularized QP. The latter that is recast as a least distance problem (LDP) and solved using a nonnegative least squares (NNLS) algorithm.

3.1 Outer proximal-point iterations

Let $Q = Q' \succeq 0$ in (1). For any $\epsilon > 0$, the sequence $\{z_k\}$ of solutions to the following strictly convex quadratic programs

$$\begin{aligned} z_{k+1} = \arg\min_z \ & \frac{1}{2} z'Qz + c'z + \frac{\epsilon}{2}\|z - z_k\|_2^2 \\ \text{s.t.} \ & \ell \le Az \le u \\ & Gz = g \end{aligned} \qquad (2)$$

converges to a solution z^* of the QP relaxation of the form (1a)-(1c) as k tends to infinity (Bemporad, 2018, Corollary 1).

3.2 Inner active-set solver

The dual of the QP problem (2) is the following convex QP

$$\min_{\lambda_\ell, \lambda_u, \mu} \frac{1}{2} \begin{bmatrix} \lambda_\ell \\ \lambda_u \\ \mu \end{bmatrix}' \begin{bmatrix} -A \\ A \\ G \end{bmatrix} (Q + \epsilon I)^{-1} \begin{bmatrix} -A \\ A \\ G \end{bmatrix}' \begin{bmatrix} \lambda_\ell \\ \lambda_u \\ \mu \end{bmatrix} + \begin{bmatrix} d_\ell^k \\ d_u^k \\ f^k \end{bmatrix}' \begin{bmatrix} \lambda_\ell \\ \lambda_u \\ \mu \end{bmatrix} \qquad \text{(3a)}$$

$$\text{s.t.} \quad \lambda_\ell, \lambda_u \ge 0,\ \mu \text{ free}, \qquad \text{(3b)}$$

where $\lambda_\ell, \lambda_u \in \mathbb{R}^m$, $\mu \in \mathbb{R}^q$, and $d_\ell^k, d_u^k \in \mathbb{R}^m$, $f^k \in \mathbb{R}^q$ are defined as follows:

$$\begin{bmatrix} d_\ell^k \\ d_u^k \\ f^k \end{bmatrix} \triangleq \begin{bmatrix} -\ell \\ u \\ g \end{bmatrix} + \begin{bmatrix} -A \\ A \\ G \end{bmatrix} (Q + \epsilon I)^{-1} (c - \epsilon z_k). \qquad \text{(3c)}$$

The following theorem shows how the QP problem (2) is equivalent to a least-squares problem in which some of the variables are constrained to be nonnegative, extending (Bemporad, 2016, Th. 1) to the case of equality constraints, bilateral inequalities as in (Bemporad, 2015) and subsequently to scaling of d_ℓ^k, d_u^k, f^k for improved numerical robustness as in (Bemporad, 2018).

Theorem 1. Consider the QP (1a)–(1c) and let $Q \succeq 0$, $\epsilon > 0$. Let $L'L$ be a Cholesky factorization of $Q + \epsilon I$ and define $M \triangleq AL^{-1}$, $N \triangleq GL^{-1}$. Let γ, β be any positive scalars. Consider the Partially Nonnegative Least Squares (PNNLS) problem

$$\min_{y_\ell, y_u, \nu} \frac{1}{2} \left\| \begin{bmatrix} -M' \\ \beta(d_\ell^k)' \end{bmatrix} y_\ell + \begin{bmatrix} M' \\ \beta(d_u^k)' \end{bmatrix} y_u + \begin{bmatrix} N' \\ \beta(f^k)' \end{bmatrix} \nu + \begin{bmatrix} 0 \\ \beta\gamma \end{bmatrix} \right\|_2^2 \qquad \text{(4a)}$$

$$\text{s.t.} \quad y_\ell, y_u \ge 0,\ \nu \text{ free} \qquad \text{(4b)}$$

with $y_\ell, y_u \in \mathbb{R}^m$, $\nu \in \mathbb{R}^q$, and let

$$\delta^* \triangleq \beta(\gamma + (d_\ell^k)' y_\ell^* + (d_u^k)' y_u^* + (f^k)' \nu^*) \qquad \text{(5a)}$$

$$r^* \triangleq \begin{bmatrix} M'(y_\ell^* - y_u^*) - N' \nu^* \\ -\delta^* \end{bmatrix} \qquad \text{(5b)}$$

where $r^* \in \mathbb{R}^{n+1}$ is the residual obtained at the optimal solution (y_ℓ^*, y_u^*, ν^*) of (4), where $y_\ell^*, y_u^* \in \mathbb{R}^m$, $\nu^* \in \mathbb{R}^q$. The following statements hold:

i) If $r^* = 0$ then QP (1a)–(1c),(1e) is infeasible;

ii) If $r^* \neq 0$ then

$$z^* \triangleq -(Q+\epsilon I)^{-1}\left(c - \epsilon z_k + \frac{A'(y_u^* - y_\ell^*) + G'\nu^*}{\beta\delta^*}\right) \tag{6}$$

and

$$\lambda_\ell^* \triangleq \frac{1}{\beta\delta^*}y_\ell^*,\ \lambda_u^* \triangleq \frac{1}{\beta\delta^*}y_u^*,\ \mu^* \triangleq \frac{1}{\beta\delta^*}\nu^*$$

solve QP (2) and its dual (3), respectively.

The proof is a slight modification in (Bemporad, 2018, Appendix A), specifically M is replaced by $\left[\begin{smallmatrix}-M & M\end{smallmatrix}\right]'$, d^k by $\left[\begin{smallmatrix}d_\ell^k & d_u^k\end{smallmatrix}\right]'$, y by $\left[\begin{smallmatrix}y_\ell & y_u\end{smallmatrix}\right]'$, w by $\left[\begin{smallmatrix}w_\ell & w_u\end{smallmatrix}\right]'$, and λ by $\left[\begin{smallmatrix}\lambda_\ell & \lambda_u\end{smallmatrix}\right]'$.

3.3 Warm starting

At each proximal-point iteration, as well as when starting to solve a new QP relaxation after a solution is available from the parent node in the B&B search tree, the QP problem (2) can be warm-started from an initial guess z_k and the corresponding problem (4) from sets of active constraints $\mathcal{P}_u, \mathcal{P}_\ell \subseteq \{1,\dots,m\}$, $\mathcal{P}_u \cap \mathcal{P}_\ell = \emptyset$, along with the initial guess $y_\ell \geq 0, y_u \geq 0, w_\ell, w_u \in \mathbb{R}^m$ and $\nu \in \mathbb{R}^q$ which satisfies the following conditions:

$$w_\ell = -M(M'(y_u - y_\ell) + N'\nu) + \beta\delta d_\ell^k \tag{7a}$$
$$w_u = M(M'(y_u - y_\ell) + N'\nu) + \beta\delta d_u^k \tag{7b}$$
$$y_\ell' w_\ell = y_u' w_u = 0 \tag{7c}$$
$$y_{\ell i} \geq 0,\ w_{\ell i} = 0,\ \forall i \in \mathcal{P}_\ell \tag{7d}$$
$$y_{ui} \geq 0,\ w_{ui} = 0,\ \forall i \in \mathcal{P}_u \tag{7e}$$
$$y_{\ell i} = 0,\ \forall i \in \{1,\dots,m\} \setminus \mathcal{P}_\ell \tag{7f}$$
$$y_{ui} = 0,\ \forall i \in \{1,\dots,m\} \setminus \mathcal{P}_u \tag{7g}$$
$$\nu = -\begin{bmatrix} N' \\ \beta(f^k)' \end{bmatrix}^{\#} \begin{bmatrix} M'(y_u - y_\ell) \\ \delta \end{bmatrix} \tag{7h}$$

where $\#$ denotes pseudoinversion, and

$$\delta = \beta(\gamma + (d_\ell^k)'y_\ell + (d_u^k)'y_u + (f^k)'\nu). \tag{7i}$$

Terms (7a)-(7g), (7i) are derived from the Karush-Kuhn-Tucker (KKT) conditions of problem (4). Condition (7h) is derived from the proof of (Bemporad, 2018, Appendix A) with the modifications mentioned in Section 3.2.

3.4 Parameter selection and scaling

The parameters $\beta,\gamma > 0$ are selected as in (Bemporad, 2018)

$$\gamma = 1 + \|f^k\|_1 + \|d_{\ell\mathcal{P}_\ell}^k\|_1 + \|d_{u\mathcal{P}_u}^k\|_1,\ \beta = \frac{1}{\gamma},$$

which are known to provide good numerical conditioning. Unlike first-order methods which are well-known to be very sensitive to scaling, our numerical experiments suggest that this algorithm is less sensitive to the scaling. However, in order to ensure better numerical robustness of the overall MIQP solver, we use the scaling applied to the constraints of type (1b), (1c) as

$$\hat{M}_i \triangleq \frac{1}{\|M_i\|_2},\ \hat{N}_i \triangleq \frac{1}{\|N_i\|_2}$$
$$\hat{M}_i\ell_i \leq \hat{M}_i A_i z \leq \hat{M}_i u_i$$
$$\hat{N}_i G_i z = \hat{N}_i g_i.$$

Algorithm 1 Robust QP solver based on NNLS and proximal-point iterations

Input: QP matrices Q, c, A, G, vectors u, ℓ, g, regularization term $\epsilon > 0$, feasibility tolerance $\sigma > 0$, stop tolerance $\eta > 0$, initial guess z_0.

```
 1. Compute inverse Cholesky factor $\mathcal{L}^{-1}$, $\mathcal{L}'\mathcal{L} = (Q+\epsilon I)$;
 2. $M \leftarrow A\mathcal{L}^{-1}$, $N \leftarrow G\mathcal{L}^{-1}$; $\mathcal{P}_\ell, \mathcal{P}_u \leftarrow \emptyset$;
 3. Factorize $NN'$ as $LDL' = NN'$;
 4. $k, h \leftarrow 0$;
 5. $v_k \leftarrow \mathcal{L}^{-T}(c - \epsilon z_k)$; $W_\mathcal{P} \leftarrow \begin{bmatrix} -M_{\mathcal{P}_\ell} \\ M_{\mathcal{P}_u} \\ N \end{bmatrix}$; $d_\ell^k \leftarrow -(\ell + Mv_k)$;
    $d_u^k \leftarrow u + Mv_k$; $\theta_\mathcal{P}^k \leftarrow \begin{bmatrix} d_{\ell\mathcal{P}_\ell}^k \\ d_{u\mathcal{P}_u}^k \\ g + Nv_k \end{bmatrix}$; $\gamma \leftarrow \|\theta_\mathcal{P}^k\|_1$; $\beta \leftarrow \frac{1}{\gamma}$;
 6. $[L, D] \leftarrow \mathsf{rankone}(L, D, 1, \beta\theta_\mathcal{P}^k)$;
 7. $(y_\ell, y_u, \nu, \mathcal{P}_\ell, \mathcal{P}_u, \gamma, \beta) \leftarrow \mathsf{get\_feasible}(\mathcal{P}_\ell, \mathcal{P}_u)$; $h \leftarrow h+1$;
 8. $\delta \leftarrow \beta(\gamma + (\theta_\mathcal{P}^k)' \begin{bmatrix} y_{\ell\mathcal{P}_\ell} \\ y_{u\mathcal{P}_u} \\ \nu \end{bmatrix})$, $a \leftarrow W_\mathcal{P}' \begin{bmatrix} y_{\ell\mathcal{P}_\ell} \\ y_{u\mathcal{P}_u} \\ \nu \end{bmatrix}$,
    $\begin{bmatrix} w_\ell \\ w_u \end{bmatrix} \leftarrow Ma \begin{bmatrix} -I \\ I \end{bmatrix} + \beta\delta \begin{bmatrix} d_\ell^k \\ d_u^k \end{bmatrix}$;
 9. $\rho \leftarrow a'a + \delta^2$;
10. **if** $\rho = 0$ **then** QP problem (1) is infeasible; **go to** Step 18;
11. **if** $\min\{w_{\ell_i}, w_{u_i}\} \geq -\beta\delta\sigma$, $\forall i \in \{1,\dots,m\}$, **or** $\mathcal{P}_\ell \cup \mathcal{P}_u = \{1,\dots,m\}$ **then go to** Step 14;
12. $i_\ell \leftarrow \arg\min_{i \in \{1,\dots,m\} \setminus \mathcal{P}_\ell} w_{\ell i}$,
    $i_u \leftarrow \arg\min_{i \in \{1,\dots,m\} \setminus \mathcal{P}_u} w_{ui}$,
    **if** $w_{\ell i_\ell} \leq w_{u i_u}$ **then** $\mathcal{P}_\ell \leftarrow \mathcal{P}_\ell \cup \{i_\ell\}$; $\gamma \leftarrow \gamma + |d_{\ell i_\ell}|$;
    **otherwise** $\mathcal{P}_u \leftarrow \mathcal{P}_u \cup \{i_u\}$; $\gamma \leftarrow \gamma + |d_{u i_u}|$;
    $\bar{\beta} = \beta$, $\beta = \frac{1}{\gamma}$;
13. update $LDL^T$ factorization **go to** Step 7;
14. $k \leftarrow k+1$;
15. $\begin{bmatrix} \lambda_{\ell k} \\ \lambda_{uk} \\ \mu_k \end{bmatrix} \leftarrow \frac{1}{\delta} \begin{bmatrix} y_\ell \\ y_u \\ \nu \end{bmatrix}$; $u_k \leftarrow \frac{1}{\delta}a$; $z_k \leftarrow \mathcal{L}^{-1}(u_k - v_k)$;
16. **if** $\|z_k - z_{k-1}\| > \eta$ **then**
    $[L, D] \leftarrow \mathsf{rankone}(L, D, -1, \beta d_\mathcal{P}^k)$; **go to** Step 5;
17. $z^* \leftarrow z_k$, $\lambda_\ell^* \leftarrow \lambda_{\ell_k}$, $\lambda_u^* \leftarrow \lambda_{u_k}$, $\mu^* \leftarrow \mu_k$, $\mathcal{P}_\ell^* \leftarrow \mathcal{P}_\ell$, $\mathcal{P}_u^* \leftarrow \mathcal{P}_u$;
18. **end**.

19. **procedure**$(y_\ell, y_u, \nu, \mathcal{P}_\ell, \mathcal{P}_u, \gamma, \beta) \leftarrow \mathsf{get\_feasible}(\mathcal{P}_\ell, \mathcal{P}_u)$
  19.1. solve the LS problem
        $\begin{bmatrix} s_{\ell\mathcal{P}_\ell} \\ s_{u\mathcal{P}_u} \\ s_\nu \end{bmatrix} \leftarrow \arg\min_z \left\| \begin{bmatrix} W_\mathcal{P}' \\ \beta(\theta_\mathcal{P}^k)' \end{bmatrix} z + \begin{bmatrix} 0_n \\ \beta\gamma \end{bmatrix} \right\|_2^2$;
        $s_{\ell\{1,\dots,m\} \setminus \mathcal{P}_\ell} \leftarrow 0$, $s_{u\{1,\dots,m\} \setminus \mathcal{P}_u} \leftarrow 0$;
  19.2. **if** $s_{\ell\mathcal{P}_\ell}, s_{u\mathcal{P}_u} \geq 0$ **then** $\begin{bmatrix} y_\ell \\ y_u \\ \nu \end{bmatrix} \leftarrow \begin{bmatrix} s_\ell \\ s_u \\ s_\nu \end{bmatrix}$ and **go to** Step 19.8;
  19.3. $j_\ell \leftarrow \arg\min_{h \in \mathcal{P}_\ell:\ s_{\ell h} \leq 0} \left\{ \frac{y_{\ell h}}{y_{\ell h} - s_{\ell h}} \right\}$,
        $j_u \leftarrow \arg\min_{h \in \mathcal{P}_u:\ s_{uh} \leq 0} \left\{ \frac{y_{uh}}{y_{uh} - s_{uh}} \right\}$;
  19.4. $\begin{bmatrix} y_\ell \\ y_u \\ \nu \end{bmatrix} \leftarrow \begin{bmatrix} y_\ell \\ y_u \\ \nu \end{bmatrix} + \min\{j_\ell, j_u\} \cdot \left( \begin{bmatrix} s_\ell \\ s_u \\ s_\nu \end{bmatrix} - \begin{bmatrix} y_\ell \\ y_u \\ \nu \end{bmatrix} \right)$;
  19.5. $\mathcal{I}_\ell \leftarrow \{h \in \mathcal{P}_\ell : y_{\ell h} = 0\}$, $\mathcal{P}_\ell \leftarrow \mathcal{P}_\ell \setminus \mathcal{I}_\ell$;
        $\gamma \leftarrow \gamma - \|d_{\ell\mathcal{I}_\ell}\|_1$;
        $\mathcal{I}_u \leftarrow \{h \in \mathcal{P}_u : y_{uh} = 0\}$; $\mathcal{P}_u \leftarrow \mathcal{P}_u \setminus \mathcal{I}_u$;
        $\gamma \leftarrow \gamma - \|d_{u\mathcal{I}_u}\|_1$; $\bar{\beta} = \beta$, $\beta = \frac{1}{\gamma}$;
  19.6. update $LDL^T$ factorization
  19.7. **go to** Step 19.1;
  19.8. **end procedure**.
```

Output: Primal solution z^* solving (1a)-(1d) dual solution $(\lambda_\ell^*, \lambda_u^*, \mu^*)$, optimal active sets $\mathcal{P}_\ell^*$, $\mathcal{P}_u^*$ or infeasibility status.

3.5 Stopping criteria and optimality

At Step 16 the iterates of Algorithm 1 stop when

$$\|z_k - z_{k-1}\| \leq \eta.$$

This condition corresponds to satisfying the optimality condition

$$\|Qz_k + c + A'(\lambda_{u_k} - \lambda_{\ell_k}) + G'\mu_k\| \leq \epsilon\eta \tag{9}$$

of the QP problem of the form (1a)–(1c). In fact, from (6), we have

$$(Q + \epsilon I)z_k = -c + \epsilon z_{k-1} - A'(\lambda_{u_k} - \lambda_{\ell_k}) - G'\mu_k$$
$$\epsilon(z_k - z_{k-1}) = -Qz_k - c - A'(\lambda_{u_k} - \lambda_{\ell_k}) - G'\mu_k$$

and hence $\|z_k - z_{k-1}\| \leq \eta$ implies (9).

In Algorithm 1, Steps 1-4 performs initialization, followed by active-set iterations starting at Step 5. The active sets $\mathcal{P}_u, \mathcal{P}_\ell$ change for every iteration while solving problem (4) at a given proximal-point iteration k; this is done via rank-one updates of LDL^T factorization (Bemporad, 2018, Section III. C). Feasible values as in (7) are computed at Steps 7, 8. Steps 9, 10 check for infeasibility. The inner active set iterations continue from Step 5 until w is feasible (within tolerance) at Step 11 or if all inequality constraints have entered the active set, followed by computation of solution z_k at Steps 14, 15. Step 16 continues executing proximal-point iterations until the criteria (9) is satisfied followed by computing the optimal solution at Step 17.

4. MIQP SOLVER

Algorithm 2 describes a standard B&B method to solve the MIQP problem (1a)-(1d). It uses Algorithm 1 for solving the QP relaxations (1a)-(1c),(1e). It consists of three basic sets J, $I_{\bar{\ell}}$, $I_{\bar{u}}$, contains the indices of binary constraints of type $\bar{\ell}_J \leq \bar{A}_J z \leq \bar{u}_J$, $\bar{A}_{I_{\bar{\ell}}} z = \bar{\ell}_{I_{\bar{\ell}}}$ and $\bar{A}_{I_{\bar{u}}} z = \bar{u}_{I_{\bar{u}}}$ respectively where $J = \{1, \ldots, p\} \setminus (I_{\bar{\ell}} \cup I_{\bar{u}})$, $I_{\bar{\ell}} \cap I_{\bar{u}} = \emptyset$. The tuple $\mathcal{T}$ holds the sets $I_{\bar{\ell}}$, $I_{\bar{u}}$ for each relaxed QP subproblem. The stack $\mathcal{S}$ holds these tuples with the order of remaining QP subproblems to be solved. The best cost associated with an integer feasible solution is denoted by V_0, z^* denotes the optimal value of the MIQP problem. Vector z_k is the initial value of the optimization vector, which is initialized with an arbitrary initial guess z_0.

At Step 1, the set J is initialized with all p binary constraints, which is equivalent to (1e) and represents the root-node of the B&B tree. At Step 2.1, the last tuple $\mathcal{T}$ is popped from the stack $\mathcal{S}$. Step 2.2 solves this QP relaxation using Algorithm 1.

Step 2.3 checks if the solution of the QP problem is feasible and the solution $V^* \leq V_0$. Step 2.3.1 checks if all the binary constraints are satisfied and updates the optimal values V^* and ξ^*. The binary constraint to be branched upon j is picked from the set J having the largest fractional part at Step 2.2.3.1. At Step 2.2.3.2, the index j is removed from the set J, and two tuples $\mathcal{T}_0$, $\mathcal{T}_1$ are created corresponding to the two subproblems with j-th binary constraint equal to $\bar{\ell}_j$ and $\bar{u}_j$ respectively. In case of binary constraints that derive from imposing binary variables $z_j \in \{0,1\}$, z_j^* is replaced by either 0 or 1 in Step 2.2.3.2 for warm start. Step 2.2.3.3 decides the priority among the two subproblems $\mathcal{T}_0$, $\mathcal{T}_1$ based on the value of $\bar{A}_j z^*$ of the relaxed solution. Once all QP problems are solved, the value of V_0 is checked at Step 3. If V_0 is still having $+\infty$ then (1) is infeasible, in the other case the optimal cost $\mathcal{V}^*$ and the optimal solution ξ^* is returned.

Algorithm 2 MIQP solver

Input: MIQP problem matrices $Q = Q' \succeq 0$, c, A, $\bar{A}$, G and vectors ℓ, u, $\bar{\ell}$, $\bar{u}$, g, initial value z_0; feasibility tolerance $\epsilon \geq 0$.

```
1. set $V_0 \leftarrow +\infty$; $z^* \leftarrow \emptyset$; $J \leftarrow \{1,\ldots,p\}$; $I_{\bar{\ell}} \leftarrow \emptyset$; $I_{\bar{u}} \leftarrow \emptyset$; $z_k \leftarrow z_0$; $\mathcal{T} \leftarrow \{I_{\bar{\ell}}, I_{\bar{u}}, z_k\}$; $\mathcal{S} \leftarrow \{\mathcal{T}\}$;
2. while $\mathcal{S} \neq \emptyset$ do:
   2.1. $\{I_{\bar{\ell}}, I_{\bar{u}}, z_k\} \leftarrow$ last element $\mathcal{T}$ of $\mathcal{S}$; $\mathcal{S} \leftarrow \mathcal{S} \setminus \{\mathcal{T}\}$;
   2.2. execute Algorithm 1 to solve QP problem for $z^*$,$V^*$;
   2.3. if the QP problem is feasible and $V^* \leq V_0$ then
      2.3.1. if $\bar{A}_i z^* \in \{\bar{\ell}_i, \bar{u}_i\}$, $\forall i \in J$ or $I_{\bar{\ell}} \cup I_{\bar{u}} = \{1,\ldots,p\}$ then $V_0 \leftarrow V^*$, $\xi^* \leftarrow z^*$; otherwise
         2.2.3.1. $j \leftarrow \arg\min_{i \in J} \left|\bar{A}_i z^* - \frac{\bar{\ell}_i + \bar{u}_i}{2}\right|$;
         2.2.3.2. $J \leftarrow J \setminus j$, $\mathcal{T}_0 \leftarrow \{I_{\bar{\ell}} \cup \{j\}, I_{\bar{u}}, z^*\}$; $\mathcal{T}_1 \leftarrow \{I_{\bar{\ell}}, I_{\bar{u}} \cup \{j\}, z^*\}$;
         2.2.3.3. if $\bar{A}_j z^* \leq \frac{\bar{\ell}_j + \bar{u}_j}{2}$ then append $\{\mathcal{T}_1, \mathcal{T}_0\}$ to $\mathcal{S}$ otherwise append $\{\mathcal{T}_0, \mathcal{T}_1\}$ to $\mathcal{S}$;
3. if $V_0 = +\infty$ then (1) infeasible otherwise $\mathcal{V}^* \leftarrow V_0$;
4. end.
```

Output: Solution ξ^* of the MIQP problem (1), optimal cost $\mathcal{V}^*$, or infeasibility status.

5. WARM STARTING BINARY VARIABLES

For simplicity we focus on the case of constraints (1d) having the form $z_i \in \{0,1\}$ for some indices $i \in \mathcal{B}$, $\mathcal{B} \subseteq \{1,\ldots,n\}$, with $card\,\mathcal{B} = p$. We want to introduce a strategy in the MIQP solver that exploits warm starts on (all or some of) the binary variables z_i, which can be immediately extended to more general constraints of the form (1d).

Let $\tilde{I}_{\bar{\ell}}, \tilde{I}_{\bar{u}} \subseteq \mathcal{B}$ be the sets containing the indices of warm started binary variables set equal to 0 or 1 respectively, $\tilde{I}_{\bar{\ell}} \cap \tilde{I}_{\bar{u}} = \emptyset$, $\tilde{I}_{\bar{\ell}} \cup \tilde{I}_{\bar{u}} \neq \emptyset$. After solving the relaxed problem (1a)-(1c), (1e) at the root node to compute the optimal solution z^* (steps up to Step 2.2 of Algorithm 2), and assuming that $z_i^* \notin \{0,1\}$ for some $i \in \mathcal{B}$, Steps 2.2.3.1-2.2.3.3 of Algorithm 2 are replaced by Algorithm 3, which is described below.

Steps 1.1-1.3 are executed if the warm started binary variables have fractional values. In this case, the priority for branching is first given to these warm started variables at Step 1.1. At Step 1.2, the selected j is removed from the set J, and two new subproblems are created. These subproblems are pushed on $\mathcal{S}$ with the order defined in Step 1.3. Once all the warm started binaries are branched upon, Steps 1.4-1.6 are executed for further branching on the binary variables which were not warm started.

Step 2 is executed right after the relaxed problem at the root node is solved and two subproblems are pushed on the stack, and executed only once as ensured by condition in 2. At Step 2.1, all the warm started values are removed from set J. The binary constraints j having the largest fractional part at is picked (from the indices of non warm

```
Algorithm 3 Warm start for binary variables
Input: Sets $\tilde{I}_{\bar{\ell}}$, $\tilde{I}_{\bar{u}}$ containing the values corresponding to the warm start, $J$, $I_{\bar{\ell}}$, $I_{\bar{u}}$; Optimal solution $z^*$ of the QP problem;
1. if $J \cap (\tilde{I}_{\bar{\ell}} \cup \tilde{I}_{\bar{u}}) \neq \emptyset$ then:   (first branch on the warm started binary variables)
    1.1. $j \leftarrow \inf(J \cap (\tilde{I}_{\bar{\ell}} \cup \tilde{I}_{\bar{u}}))$;
    1.2. $J \leftarrow J \setminus j$, $\mathcal{T}_0 \leftarrow \{I_{\bar{\ell}} \cup \{j\}, I_{\bar{u}}, z^*\}$; $\mathcal{T}_1 \leftarrow \{I_{\bar{\ell}}, I_{\bar{u}} \cup \{j\}, z^*\}$;
    1.3. if $j \in \tilde{I}_{\bar{\ell}}$ then append $\{\mathcal{T}_1, \mathcal{T}_0\}$ to $\mathcal{S}$ otherwise append $\{\mathcal{T}_0, \mathcal{T}_1\}$ to $\mathcal{S}$;
    otherwise
    1.4. $j \leftarrow \arg\min_{i \in J} \left|z_i^* - \frac{1}{2}\right|$;
    1.5. $J \leftarrow J \setminus j$, $\mathcal{T}_0 \leftarrow \{I_{\bar{\ell}} \cup \{j\}, I_{\bar{u}}, z^*\}$; $\mathcal{T}_1 \leftarrow \{I_{\bar{\ell}}, I_{\bar{u}} \cup \{j\}, z^*\}$;
    1.6. if $z_j^* \leq \frac{1}{2}$ then append $\{\mathcal{T}_1, \mathcal{T}_0\}$ to $\mathcal{S}$ otherwise append $\{\mathcal{T}_0, \mathcal{T}_1\}$ to $\mathcal{S}$;
2. if $\mathbf{card}(J) = p - 1$ then   (just once below the root node)
    2.1. set $J \leftarrow \{1, \ldots, p\} \setminus (\tilde{I}_{\bar{\ell}} \cup \tilde{I}_{\bar{u}})$;
    2.2. $j \leftarrow \arg\min_{i \in J} \left|z_i^* - \frac{1}{2}\right|$;
    2.3. $J \leftarrow J \setminus j$, $\mathcal{T}_0 \leftarrow \{\tilde{I}_{\bar{\ell}} \cup \{j\}, \tilde{I}_{\bar{u}}, z^*\}$; $\mathcal{T}_1 \leftarrow \{\tilde{I}_{\bar{\ell}}, \tilde{I}_{\bar{u}} \cup \{j\}, z^*\}$;
    2.4. if $z_j^* \leq \frac{1}{2}$ then append $\{\mathcal{T}_1, \mathcal{T}_0\}$ to $\mathcal{S}$ otherwise append $\{\mathcal{T}_0, \mathcal{T}_1\}$ to $\mathcal{S}$;   (pushed at the top of $\mathcal{S}$ with the highest priority)
Output: Two subproblems $\mathcal{T}_1, \mathcal{T}_0$ (in appropriate order) are pushed at the top of the stack $\mathcal{S}$.
```

started binary variables) at Step 2.2. At Step 2.3, two new subproblems are created with j-th variable branched upon along with the binary variables belonging to sets $\tilde{I}_{\bar{\ell}}$ and $\tilde{I}_{\bar{u}}$ fixed equal to $\bar{\ell}_i$ and $\bar{u}_i$ respectively. These two subproblems are pushed on the top of $\mathcal{S}$ as in Step 2.4. The binary variables $z_j \in \{0, 1\}$, z_j^* is replaced by either 0 or 1 in Steps 1.2, 1.5 and 2.3 for warm start.

According to a last-in-first-out strategy, these two problems are solved before exploring nodes below the root node which were formerly pushed on $\mathcal{S}$. This step will force the tree exploration from these two new subproblems, and further branching will be done only if necessary, until leaf nodes are reached. If the resulting leaves are feasible then the value of V_0 (the best known integer-feasible solution) is updated appropriately with V^*. In this case, finite value of V_0 upfront can reduce the number of QP subproblems solved in the course of finding the global solution of MIQP problem using the proposed B&B algorithm.

This step is followed by convention of solving the problems from the stack $\mathcal{S}$ below the root node using Step 1 of Algorithm 3. We ensure not to solve the nodes leading to the leaves already solved while exploring the tree now from beneath the root node.

We illustrate the flow of Algorithm 2+3 with an example with 3 binary variables, warm started using the sequence $(1, 0, \star)$, that is the warm start $z_1 = 1$, $z_2 = 0$. The flow of B&B is shown in Figure 1, where the number inside the node denotes the order in which the QP relaxation is executed. First the root node is solved and then two problems are pushed on the stack with high priority to the $(1, \star, \star)$ node.

Then, the leaf problem $(1, 0, 0)$ is solved first (QP #2), and $(1, 0, 1)$ immediately after (QP #3). Once these two leaves have been solved, problem $(1, \star, \star)$ is popped from the stack (due to the last-in-first-out method), but not solved (depicted as a dashed circle in Fig. 1), because we have already explored the children nodes of $(1, 0, \star)$. Indeed, problems $(1, 0, 0)$ and $(1, 0, 1)$ are ignored when popped again from stack herein. Therefore, problem $(1, 1, \star)$ is popped from the stack and solved (QP #4).

We remark that we have saved solving 2 QP relaxations without compromising the optimality of the MIQP solution. In general, using this approach at least $p - 1$ QP relaxations can be saved.

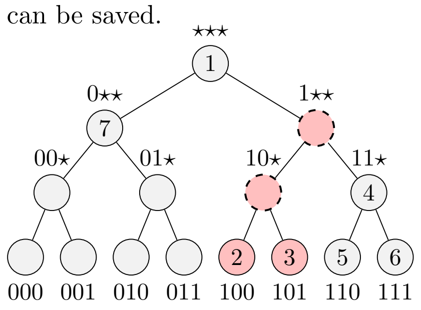

Fig. 1. Illustration example with 3 binary variables and $(1, 0, \star)$ as a binary warm start. The numbers indicate the order in which the QP relaxations are solved, dashed nodes correspond to QP subproblems that are ignored

When solving hybrid MPC in receding horizon fashion, at every sampling instance a sequence z is optimized over the whole horizon of length N, the first computed value of the input is applied to the system, then the horizon window is shifted by one sample time and the procedure repeated. Let $z^* = (v_{0|t}^*, \delta_{0|t}^*, \zeta_{0|t}^*, \ldots, v_{N-1|t}^*, \delta_{N-1|t}^*, \zeta_{N-1|t}^*)$ be the optimal solution computed at time t, where v_k, ζ_k, δ_k are the vectors of inputs, real-valued and binary auxiliary variables respectively, corresponding to the optimal trajectory of the MLD model used by hybrid MPC (Bemporad and Morari, 1999). We can exploit the shifted binary values optimized at the previous step as the binary warm start, in particular $\delta_{0|t+1} = \delta_{1|t}^*$, $\ldots$, $\delta_{N-2|t+1} = \delta_{N-1|t}^*$ (and similarly for binary inputs, if any), and use Algorithm 2 with Steps 2.2.3.1–2.2.3.3 replaced by Algorithm 3.

6. NUMERICAL RESULTS

We report numerical experiments carried out on a desktop computer with Intel Core i7-4700MQ CPU 2.40 GHz and 8 GB RAM, using MATLAB R2015a. Algorithms 2, 3 are implemented in interpreted MATLAB code and Algorithm 1 in compiled Embedded MATLAB code.

We consider the hybrid MPC problem from (Bemporad and Morari, 1999, Example 5.1), with the default settings as in `bm99sim.m` which is a part of the the Hybrid Toolbox for MATLAB (Bemporad, 2003). This demo considers unit weight on the output of the system and all the other weights are zero. This hybrid MPC problem has been tested for the prediction horizon N from 2 to 15.

N	NNLS		NNLS*		GUROBI		CPLEX	
	avg	max	avg	max	avg	max	avg	max
2	2.0	2.6	2.0	2.6	1.6	2.0	3.1	6.0
3	3.1	5.1	2.5	4.8	2.7	3.0	6.3	11.5
4	5.3	8.8	3.1	6.9	3.1	3.9	8.9	15.7
5	8.1	16.4	3.9	13.0	3.9	4.8	10.8	15.7
6	13.6	25.5	5.1	18.3	4.7	7.3	11.1	17.1
7	21.7	46.2	6.4	30.2	5.6	9.5	17.5	80.4
8	29.7	71.0	8.1	43.4	7.2	13.2	15.5	80.2
9	49.5	115.9	11.1	69.8	8.8	15.3	22.8	110.6
10	76.2	146.1	14.4	103.2	11.1	17.6	35.1	95.3
11	121.3	254.6	20.6	179.1	13.0	23.9	37.3	102.5
12	155.8	410.8	26.9	263.4	14.9	31.2	61.7	103.7
13	247.6	607.9	35.5	384.9	17.5	36.6	47.3	119.5
14	304.9	893.7	46.3	562.4	21.4	67.4	81.6	150.6
15	484.2	1242.3	61.7	766.9	25.9	109.8	89.9	181.1

Table 1. Hybrid MPC problem: CPU time (ms) per sampling step for different prediction horizons N

The performance of Algorithm 1 is compared against commercial solvers GUROBI (Gurobi Optimization, Inc., 2014) and CPLEX (IBM, Inc., 2014) with presolvers enabled. The maximum norm of the error in the input trajectories is less than 10^{-4}.

Table 1 report the CPU time taken by the different solvers, where the columns NNLS and NNLS* denote Algorithm 1+2 without warm starting binary variables and Algorithm 1+2+3 with warm start of binary variables, respectively. For $N = 10$, the MIQP problem has $n = 40$, $p = 10$ (i.e., 30 continuous variables and 10 binary variables), and $m = 160$ linear inequalities.

7. CONCLUSIONS

Motivated by embedded hybrid MPC applications, this paper has proposed a new MIQP solver based on branch and bound that is able to exploit warm start on binary variables. A numerically efficient and robust QP solver based on nonnegative least squares and proximal point iterations is used for solving the QP relaxations, which does not require the Hessian matrix to be positive definite; this is particularly useful in hybrid MPC problems based on quadratic costs where the quadratic cost is only positive semidefinite due to some variables having zero weight in the MPC cost function. The reported numerical results demonstrate that presented framework for warm starting binary variables for hybrid MPC problems using the same QP solution algorithm provides better performance in terms of both average and maximum computation time.

REFERENCES

Axehill, D. and Hansson, A. (2006). A mixed integer dual quadratic programming algorithm tailored for MPC. In *Proc. 45th IEEE Conference on Decision and Control*, 5693–5698. San Diego, CA, USA.

Bemporad, A. (2003). *Hybrid Toolbox – User's Guide.* `http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox`.

Bemporad, A. (2015). Solving mixed-integer quadratic programs via nonnegative least squares. In *5th IFAC Conf. on Nonlinear Model Predictive Control*, 73–79. Sevilla, Spain.

Bemporad, A. (2016). A quadratic programming algorithm based on nonnegative least squares with applications to embedded model predictive control. *IEEE Trans. Automatic Control*, 61(4), 1111–1116.

Bemporad, A. (2018). A numerically stable solver for positive semidefinite quadratic programs based on nonnegative least squares. *IEEE Trans. Automatic Control*, 63(2), 525–531.

Bemporad, A., Mignone, D., and Morari, M. (1999). Moving horizon estimation for hybrid systems and fault detection. In *Proc. American Contr. Conf.*, 2471–2475. Chicago, IL.

Bemporad, A. and Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3), 407–427.

Bertacco, L., Fischetti, M., and Lodi, A. (2007). A feasibility pump heuristic for general mixed-integer problems. *Discrete Optimization*, 4(1), 63–76.

Ferrari-Trecate, G., Mignone, D., and Morari, M. (2002). Moving horizon estimation for hybrid systems. *IEEE Trans. Automatic Control*, 47(10), 1663–1676.

Fischetti, M., Glover, F., and Lodi, A. (2005). The feasibility pump. *Mathematical Programming*, 104(1), 91–104.

Floudas, C.A. (1995). *Nonlinear and Mixed-Integer Optimization.* Oxford University Press.

Frick, D., Domahidi, A., and Morari, M. (2015). Embedded optimization for mixed logical dynamical systems. *Computers & Chemical Engineering*, 72, 21–33.

Frick, D., Jerez, J.L., Domahidi, A., Georghiou, A., and Morari, M. (2016). Low-complexity iterative method for hybrid MPC. *ArXiv e-prints.*

Gurobi Optimization, Inc. (2014). *Gurobi Optimizer Reference Manual.* URL `http://www.gurobi.com`.

IBM, Inc. (2014). *IBM ILOG CPLEX Optimization Studio 12.6 – User Manual.*

Mejari, M., Naik, V.V., Piga, D., and Bemporad, A. (2018). Regularized moving-horizon PWA regression for LPV system identification. In *Proc. 18th IFAC Symposium on System Identification.* Stockholm, Sweden.

Naik, V.V. and Bemporad, A. (2017). Embedded mixed-integer quadratic optimization using accelerated dual gradient projection. In *Proc. 20th IFAC World Congress*, 10723–10728. Toulouse, France.

Naik, V.V., Mejari, M., Piga, D., and Bemporad, A. (2017). Regularized moving-horizon piecewise affine regression using mixed-integer quadratic programming. In *Proc. 25th Mediterranean Conf. on Control and Automation*, 1349–1354. Valletta, Malta.

Stellato, B., Naik, V.V., Bemporad, A., Goulart, P., and Boyd, S. (2018). Embedded mixed-integer quadratic optimization using the OSQP solver. In *Proc. European Control Conference (ECC)*, 1536–1541. Limassol, Cyprus.

Takapoui, R., Moehle, N., Boyd, S., and Bemporad, A. (2016). A simple effective heuristic for embedded mixed-integer quadratic programming. In *Proc. American Contr. Conf.*, 5619–5625. Boston, MA, USA.

Torrisi, F. and Bemporad, A. (2004). HYSDEL — A tool for generating computational hybrid models. *IEEE Trans. Contr. Systems Technology*, 12(2), 235–249.