Banded Null Basis and ADMM for Embedded MPC

Thuy V, Dang * **Keck Voon, Ling** ** **Jan, Maciejowski** ***

* *Interdisciplinary Graduate School (IGS), Nanyang Technological University, Singapore 640798 (e-mail: dang0028@e.ntu.edu.sg).*
** *School of Electrical and Electronic Engineering (EEE), Nanyang Technological University, Singapore 640798 (e-mail: ekvling@ntu.edu.sg).*
*** *University of Cambridge, UK (e-mail: jmm@eng.cam.ac.uk).*

Abstract: In this paper, we propose an improved QP solver for embedded implementations of MPC controllers. We adopt a "reduced Hessian" approach for handling the equality constraints that arise in the well-known "banded" formulation of MPC (in which the predicted states are not eliminated). Our key observation is that a banded basis exists for the null space of the banded equality-constraint matrix, and that this leads to a QP of the same size as the "condensed" formulation of MPC problems, which is considerably smaller than the "banded" formulation. We use the Alternating Direction Method of Multipliers (ADMM) - which is known to be particularly suitable for embedded implementations - to solve this smaller QP problem. Our C implementation results for a particular MPC example (a 9-state, 3-input quadrotor) show that our proposed algorithm is about 4 times faster than an existing well-performing ADMM variant ("indirect indicator" ADMM or "iiADMM") and 3 times faster than the well-known QP solver CVXGEN. The convergence rate and code size of the proposed ADMM variant is also comparable with iiADMM.

Keywords: Banded null basis, ADMM, embedded MPC, structured matrices, sparse QP.

1. INTRODUCTION

Model Predictive Control (MPC) is an optimisation-based control strategy which has been applied widely in industry since its appearance in the 1980s. Recent developments in optimisation method further contribute to the success story of MPC. Available methods, just to mention a few, include multi-parametric programming (see e.g. Bemporad et al. (2002)), Interior Point Methods (see e.g Wright (1997); Wang and Boyd (2010)), Active Set Method (see e.g. Fletcher (1987); Ferreau et al. (2008)), gradient-based methods (see e.g. Nesterov (1983); Richter et al. (2009); Patrinos and Bemporad (2014); Giselsson and Boyd (2015)) or the 'alternating direction method of multipliers' (ADMM) (see e.g. Boyd et al. (2011)).

For a linear time-invariant (LTI) system with quadratic cost function and linear constraints, one can formulate MPC problem as condensed QP formulation which eliminates the states from the decision variables (Maciejowski (2002)). On the other hand, one can also employ a sparse QP formulation which keeps both states and control as decision variables, resulting a bigger size QP problem. The sparse QP has both equality constraints and inequality constraints and all matrices have special structures which can be exploited to reduce the computational cost significantly (see e.g Wright (1997); Wang and Boyd (2010); Domahidi et al. (2012)).

Null space method, sometimes referred to as reduced Hessian approach (see e.g Benzi et al. (2005); Nocedal and Wright (2006)), is not a new technique in optimisation. The beginning idea is to describe the equality constraints by using a null basis and a particular solution. The null basis is not unique and there are several choices such as orthogonal basis or triangular basis (see Benzi et al. (2005) (section 6, p.32)). In spare QP formulation of MPC, the equality constraint matrix has a banded structure. For a banded matrix, there may exist a banded null basis for its null space (see e.g Berry et al. (1985)). To the best of our knowledge, banded null basis has not been exploited in MPC-related optimisation algorithms.

In this paper, we construct a banded null basis based on an algorithm from structural engineering community called Turn-back LU (TBLU) procedure of Berry et al. (1985). The banded null basis is used to obtain a smaller size QP problem than the original sparse QP while maintaining banded structures of matrices. This offers computational advantages. We then employ an ADMM-based algorithm exploiting banded matrices to solve this new QP problem.

We use the MPC example described in Ding et al. (2016) to test our proposed algorithm. This MPC example has polytopic constraints. To deal with this type of constraints in the context of ADMM scheme, there are several existing ADMM variants using slack variables (see e.g Ghadimi et al. (2015); Dang et al. (2015); Manickathu (2016)). Among these, the so-called *indirect indicator ADMM* (iiADMM) is the best in term of convergence rate, see Manickathu (2016). Hence, we will compare our new QP solver with iiADMM. In addition, we also compare with a

Preprint submitted to 20th IFAC World Congress. Received November 10, 2016.

well-known interior-point QP solver package CVXGEN of Mattingley and Boyd (2012). These algorithms are implemented in C and run on ARM-Cortex A9 processor (CPU 667MHz) mounted on Zynq ZC702 board.

Implementation results show that our proposed algorithm is about 4 times faster than iiADMM and 3 times faster than CVXGEN for this particular MPC example. The code size is small and comparable with iiADMM. In term of convergence rate, our proposed ADMM algorithm is also comparable with iiADMM.

The applicability of our proposed algorithm depends on whether or not TBLU can construct a banded null basis. Practical experiments carried out on structural analysis problems in Berry et al. (1985) showed that TBLU always returns a banded null basis. In the context of MPC, we carry out a test over 18 academic and industrial MPC examples which are available in Ferreau and Peyrl (2015), see also Kouzoupis et al. (2015). Test results confirm that a banded null basis is always found by TBLU in all these MPC examples.

2. BACKGROUND

2.1 Spase formulation in MPC

Consider a LTI system represented by the discrete-time state space model

$$x_{k+1} = Ax_k + Bu_k$$

where $x \in \mathbb{R}^{n_x}$ is state vector of dimension n_x, and $u \in \mathbb{R}^{n_u}$ is control vector of dimension n_u.

The control of this system is determined by MPC with cost function and constraints as follows

$$\underset{u}{\text{minimize}} \quad \sum_{k=0}^{N-1} \left(x_k^T Q_x x_k + u_k^T R u_k\right) + x_N^T Q_N x_N \quad (1)$$

$$\text{s.t. } G_x x_k \le g_x,\ G_u u_k \le g_u\ (k = 0, 1, ..., N-1), \quad (2)$$

$$G_N x_N \le g_N$$

where x_0 is given; N is the horizon; $Q_x,\ Q_N \in \mathbb{R}^{n_x \times n_x}$ are positive semi-definite matrices; $R \in \mathbb{R}^{n_u \times n_u}$ is positive definite matrix; $G_x \in \mathbb{R}^{n_{ix} \times n_x}$, $G_N \in \mathbb{R}^{n_{it} \times n_x}$, $G_u \in \mathbb{R}^{n_{iu} \times n_u}$, $g_x \in \mathbb{R}^{n_{ix}}$, $g_N \in \mathbb{R}^{n_{it}}$, $g_u \in \mathbb{R}^{n_{iu}}$ are appropriate matrices and vectors describing the constraints of the system (n_{ix}, n_{it}, n_{iu} are number of state constraints, terminal constraints and input constraints).

By using sparse formulation which keeps both the states and control as decision variables

$$z = (u_0; x_1; u_1; x_2; ...; u_{N-1}; x_N)$$

problem (1) can be written as following

$$\underset{z}{\text{minimize}} \quad \frac{1}{2} z^T H z \quad (3)$$

$$\text{s.t. } Fz = f \quad (4)$$

$$Gz \le g \quad (5)$$

where $H \in \mathbb{R}^{n_d \times n_d}$ is positive semi-definite ($n_d = N(n_x + n_u)$ is the number of decision variables). $F \in \mathbb{R}^{n_{eq} \times n_d}$, $G \in \mathbb{R}^{n_{ineq} \times n_d}$, $q \in \mathbb{R}^{n_d}$, $f \in \mathbb{R}^{n_{eq}}$, $g \in \mathbb{R}^{n_{ineq}}$ (where $n_{eq} = Nn_x$, $n_{ineq} = (N-1)(n_{ix}+n_{iu})+n_{it}+n_{iu}$ are total number of equality and inequality constraints respectively); the F, G, f, g, H have the following descriptions

$$F = \begin{bmatrix} -B & I & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & -A & -B & I & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & -A & -B & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \\ 0 & 0 & 0 & \cdots & 0 & -A & -B & I \end{bmatrix}, \quad (6)$$

$$G = \begin{bmatrix} I_{N-1} \otimes \begin{bmatrix} G_u & 0 \\ 0 & G_x \end{bmatrix} & \begin{matrix} 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 \end{matrix} & \begin{matrix} G_u & 0 \\ 0 & G_N \end{matrix} \end{bmatrix}, g = \begin{bmatrix} \mathbf{1}_{N-1} \otimes [g_u; g_x] \\ g_u \\ g_N \end{bmatrix},$$

$$H = \begin{bmatrix} I_{N-1} \otimes \begin{bmatrix} R & 0 \\ 0 & Q_x \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} R & 0 \\ 0 & Q_N \end{bmatrix} \end{bmatrix}, f = \begin{bmatrix} Ax_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

2.2 ADMM algorithm

ADMM has received interest for embedded MPC applications due to its simplicity. A review of ADMM algorithm can be found in Boyd et al. (2011) and references therein. Basically, ADMM algorithm solves the following optimisation problem

$$\underset{u,v}{\text{minimize}} \quad \zeta(u) + \phi(v)$$

$$\text{s.t. } \Psi u + \Upsilon v = c$$

Using augmented Lagrangian defined as

$$L_\rho(u, v, \lambda) = \zeta(u)+\phi(v)+\lambda^T(\Psi u+\Upsilon v-c)+\frac{\rho}{2}\|\Psi u+\Upsilon v-c\|_2^2$$

the ADMM method consists of three main steps:

$$u_{k+1} = \arg \min_u L_\rho(u, v_k, \tau_k)$$

$$v_{k+1} = \arg \min_v L_\rho(u_{k+1}, v, \tau_k)$$

$$\tau_{k+1} = \tau_k + (\Psi u_{k+1} + \Upsilon v_{k+1} - c)$$

where $\tau = \frac{1}{\rho}\lambda$ is a scaled dual variable, and $\rho > 0$ is named as step-size. Stopping criteria are based on primal residual $r_k = ||\Psi u_k + \Upsilon v_k - c||$ and dual residual $d_k = ||\rho \Psi^T \Upsilon (v_{k+1} - v_k)||$, see Boyd et al. (2011).

There are several ADMM algorithms for MPC problem. The differences in these variants are mainly in the way how the MPC problem is formulated as standard ADMM setup. A list of ADMM variants can be found in Manickathu (2016). We compare our proposed algorithm with *indirect indicator* ADMM (iiADMM) as this variant performs best in term of convergence rate among ADMM variants using the slack variable approach, see Manickathu (2016) for the performance comparison of these variants. The iiADMM is summarised in the appendix for easy reference.

3. BANDED NULL BASIS AND ADMM FOR MPC

3.1 Null space and the reduced Hessian QP

Null space method in optimisation is usually referred to as reduced Hessian approach, see e.g. Benzi et al. (2005) and Nocedal and Wright (2006). Basically, the solution

of equality constraints of QP problem (3), $Fz = f$, can be parametrised by $z = z_0 + Yy$ where z_0 is a particular solution, i.e., $Fz_0 = f$, and Y is a null basis of F, i.e., $FY = 0$. Then the QP problem (3) can be solved via solving the following equivalent problem

$$\underset{y}{\text{minimize}} \quad \frac{1}{2}y^T\tilde{H}y + (z_0^T H_0)y \tag{7}$$

$$\text{s.t.} \quad \tilde{G}y \leq g - Gz_0 \triangleq \tilde{g} \tag{8}$$

where $\tilde{H} = Y^T HY,\ \tilde{G} = GY,\ H_0 = HY$.

The number of decision variables of QP problem (7) is Nn_u, i.e., $y \in \mathbb{R}^{Nn_u}$ (see Lemma 1 below) whereas it is $N(n_x + n_u)$ in the original QP (3). The number of decision variables of this new QP problem is the same as the condensed QP. If $n_u << n_x$, the reduction in number of decision variables can be quite substantial.

Lemma 1: F is full row rank, i.e $rank(F) = Nn_x$, where F is the equality constraint matrix (see (6)) with dimension $(Nn_x) \times (N(n_x + n_u))$. It follows that the dimension of its null space is $N(n_x + n_u) - Nn_x = Nn_u$. (Proof is shown in the appendix).

We adopt an usually used assumption in the reduced Hessian approach:

Assumption 1: The reduced Hessian $\tilde{H} = Y^T HY$ is positive definite.

3.2 Construct a banded null basis

The structure of matrices H, G in original QP problem are block diagonal, and the equality constraint matrix F is banded. Since some matrices in the QP problems ((3) and (7)) and the null basis are not square matrices, we does not mean banded with respect to the main diagonal but with respect to a line from upper left corner to the lower right corner of the matrix. The structure of matrices $\tilde{H}$, $\tilde{G}$ and H_0 in the new QP problem (7) depends on the structure of Y. If Y is a banded matrix, the matrices $\tilde{H}$, $\tilde{G}$ and H_0 will have a banded pattern. In order to construct a banded null basis Y for the null space of F, we use Turn-back LU (TBLU) (see Algorithm 1) of Berry et al. (1985).

Algorithm 1: Turn-back LU (Berry et al. (1985)).
Given $F \in \mathbb{R}^{m\times n}$ $(m < n)$, we want to construct a banded null basis $Y \in \mathbb{R}^{n\times r}$ $(r = n - m)$, i.e $FY = 0_{m\times r}$. Denote F_i, $i = 1, 2, ..., n$ is the $i-th$ column. Given S is a subset of $\{1, 2, ..., n\}$, F_S is a set of columns of F whose indexes are in S, $Y_i(S)$ are elements in $i-th$ column of Y whose indexes are in S. The main steps in TBLU are as follows

1. Find column indexes $c_1, c_2, ..., c_r$ so that the set of columns $\{F_1, F_2, ..., F_{c_k}\} \setminus \{F_{c_1}, .., F_{c_{k-1}}\}$ are linearly dependent ($k = 1, 2, ..., r$ is iteratively increased to find a corresponding c_k, $\{F_{c_1}, .., F_{c_{k-1}}\} = \emptyset$ if $k = 1$)
2. Turn-back step:
2.1. $k = 1,\ \mathcal{T} = \emptyset$
2.2. Set column F_{c_k} as active column
2.3. Find $t_k = \max\{j | j < c_k \& F_{S_k} \triangleq \{F_{c_k},\ F_{c_k-1}, ..., F_j\} \setminus \{F_{\mathcal{T}}\}$ are linearly dependent$\}$ ($S_k = \{c_k, c_k - 1, ..., j\}$ is a set of indexes). Find linear dependent coefficients vector b_k for F_{S_k}. Assign $Y_k(S_k) = b_k$
2.4. $k = k + 1$, $\mathcal{T} = \mathcal{T} \cup \{t_k\}$. If $k \leq r$, return to 2.2, else quit and return Y.

In MPC with LTI systems, F is a fixed matrix. Hence, Y can be computed offline. To the best of our knowledge, there is no theoretical result to confirm whether or not TBLU returns a banded null basis for a general banded matrix, but it has been reported that TBLU is the most effective algorithm for computing a banded null basis (see Berry et al. (1985) and the references therein). As proved in Berry et al. (1985), Y is full column rank.

In our particular MPC example (quadrotor camera field-of-view problem in Ding et al. (2016)), using TBLU we successfully obtain a banded null basis and its pattern is show in Fig. 1. The pattern of other matrices in QP (7) are shown in Fig. 2.

3.3 Compute a particular solution z_0 of $Fz = f$

Recall that $z = (u_0; x_1;\ u_1\ x_2\ ...\ u_{N-1}; x_N)$, and the constraints $Fz = f,\ f = [Ax_0; 0; ...; 0]$ (x_0 is given), is the compact form of following constraints

$$\begin{aligned} x_1 &= Ax_0 + Bu_0 \\ x_2 &= Ax_1 + Bu_1 \\ &\vdots \\ x_N &= Ax_{N-1} + Bu_{N-1} \end{aligned}$$

Then a particular solution of $Fz = f$ is any $z = (u_0; x_1;\ u_1\ x_2\ ...\ u_{N-1}; x_N)$ whose elements satisfy the above constraints. One such particular set is: $u_0 = u_1 = ... = u_{N-1} = 0$ and $x_1 = Ax_0,\ x_2 = Ax_1, ..., x_N = Ax_{N-1}$.

3.4 Algorithm 2: B-ADMM

We then employ ADMM-based QP solver using slack variable technique Ghadimi et al. (2015) to solve problem (7). Slack variables $\nu \geq 0$ are introduced in (8) giving $\tilde{G}y + \nu = \tilde{g}$. We then have the proposed algorithm, named as B-ADMM (Banded Null Basis ADMM), as following

---

Algorithm 2: B-ADMM
I. Offline tasks such as computing $Y, \tilde{H}, \tilde{G}, H_0$
II. Online tasks
1.1. Compute a particular solution z_0 (*see* 3.3)
1.2. $q^T = z_0^T H_0,\ \tilde{g} = g - Gz_0$
Do
2. $y_{k+1} = -(\tilde{H} + \rho\tilde{G}^T\tilde{G})^{-1}[q + \rho\tilde{G}^T(\nu_k + \theta_k - \tilde{g})]$ (9)
3. $\nu_{k+1} = \max\{0, -\tilde{G}y_{k+1} + \tilde{g} - \theta_k\}$ (10)
4. $\theta_{k+1} = \theta_k + \tilde{G}y_{k+1} + \nu_{k+1} - \tilde{g}$ (11)
While: Convergence criteria are not met.
5. **Obtain:** y_{stop}. **Return solution** $z = z_0 + Yy_{stop}$

The step-size ρ is chosen heuristically [1] based on Ghadimi et al. (2015):

[1] Optimal step-size selection method in Ghadimi et al. (2015) is not applicable for our particular MPC example since $\tilde{G}$ in (8) does not satisfy the full row rank condition which is one of conditions for (12) to be an optimal step-size.

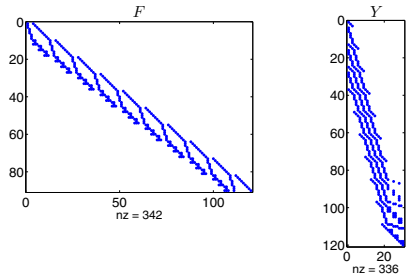

Fig. 1. Sparsity pattern of F and its null space basis Y in our example in Section 4.

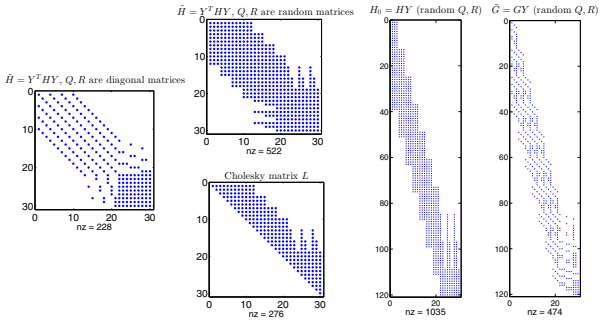

Fig. 2. Banded pattern of matrices in QP (7).

$$\rho^* = \left(\sqrt{\lambda_{\min,>0}(\tilde{G}\tilde{H}^{-1}\tilde{G}^T)\lambda_{\max,>0}(\tilde{G}\tilde{H}^{-1}\tilde{G}^T)}\right)^{-1} \quad (12)$$

3.5 Computational efficiency for B-ADMM

Once we have obtained the banded null basis and matrices in (7), we can exploit the structure of matrices to carry out (9)-(11) efficiently. Exploitation of (10)-(11) is quite straight forward since the computations consist mainly of matrix-vector multiplications.

To carry out (9), one way is to compute $(\tilde{H} + \rho\tilde{G}^T\tilde{G})^{-1}$ offline, then (9) consists of mainly matrix-vector multiplication operations. However, matrix inversion operation destroys banded pattern resulting in a computational complexity of $O(N^2 n_u^2)$ for (9). But this approach is highly parallelable.

An alternative is to exploit the bandedness of $\tilde{H}_1 = \tilde{H} + \rho\tilde{G}^T\tilde{G}$ and solve the following system of linear equation

$$\tilde{H}_1 y_{k+1} = b_k \quad (13)$$

where $b_k = q + \rho\tilde{G}^T(\nu_k + \theta_k - \tilde{g})$. An efficient method for solving system of linear equations having a banded matrix is Cholesky factorisation. Let L be the Cholesky factorisation of $\tilde{H}_1$, i.e., $\tilde{H}_1 = L^T L$. We have L is an upper banded matrix (see Fig. 2) with bandwidth b_L (depends on null basis Y). For LTI system, $\tilde{H}_1$ is constant and L can be computed offline. Then, solving (13) is done via solving following

$$L^T y_{temp} = b_k, \; L y_{k+1} = y_{temp}$$

which are forward and backward substitutions. The computational complexity of this approach is $O(N b_L^2)$. Our experience suggests that solving (9) by offline computing the Cholesky factorization is more effective than offline computing the inverse of matrix $\tilde{H}_1$, especially when N is large.

In this paper, we only report the timing of B-ADMM that uses Cholesky factorisation in (9).

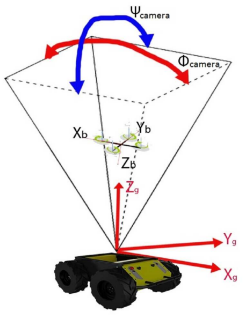

Fig. 3. Camera field-of-view quadrotor problem Ding et al. (2016)

Table 1. *Aver. runtime (ms)/ aver. iteration* for camera-field-of-view problem. D-ADMM: using a dense null basis, B-ADMM: using a banded null basis. ##: CVXGEN is unable to generate C-code for big problem.

	iiADMM	D-ADMM	B-ADMM	CVXGEN
N=5	7.4 /25	5.7 /30	2.5 /30	7.95 /5
N=10	25.4 /41	26.9 /35	7.0 /35	22.9 /5
N=15	48.5 /51	64.8 /37	11.1 /37	35.6 /5
N=20	69.4 /53	112.9 /38	17.9 /38	##
N=25	89.1 /54	192.4 /40	22.4 /40	##

4. IMPLEMENTATION RESULTS AND DISCUSSION

4.1 MPC problem: camera-field-of-view quadrotor

The example is from Ding et al. (2016). In this example, a quadrotor flies within a space named camera field-of-view. This results in polytopic constraints in MPC problem. The system models are occurrences

$$A = \text{diag}\{A_e, A_e, A_e\}, \; B = \text{diag}\{B_e, B_e, B_e\} \quad (14)$$

where

$$A_e = \begin{bmatrix} 1 & 0 & 0 \\ 0 & dT & 1 \\ 0 & 0 & (dT)^2/2 \end{bmatrix}, \; B_e = \begin{bmatrix} 1 \\ 0 \\ (dT)^2/2 \end{bmatrix}, \; dT = 0.01$$

We have $n_x = 9$, $n_u = 3$, and the polytopic constraints are $G_x x \leq f_x$ and $G_u u \leq g_u$ where $G_x \in \mathbb{R}^{6\times 9}, G_u \in \mathbb{R}^{6\times 3}$, i.e $n_{ix} = 6$, $n_{iu} = 6$, is obtained from the cited paper. The MPC parameters are $Q_x = I_{n_x}$, $R = 0.5 I_{n_u}, N = 5, 10, 15, 20, 25$.

We use MATLAB Coder R2015b to generate C-code from m-file versions of iiADMM and B-ADMM. C-code of CVXGEN is generated and downloaded from its website. Then Xilinx Software Development Kit (SDK) 14.6 is used for implementation on the ARM-Cortex A9 processor.

4.2 Results

A banded null basis Y has been successfully constructed for this MPC problem. Its pattern is shown in Fig. 1.

Table 2. Code size (kB)

	iiADMM	D-ADMM	B-ADMM	CVXGEN
N=5	91	105	82	1552
N=10	100	216	102	3488
N=15	108	400	122	5314
N=20	117	659	143	##
N=25	126	991	163	##

In Table 1 and Table 2 we show the performance in term of online computational time, number of iterations and code size, running on the ARM-Cortex A9 processor (667 MHz, single core, bare metal, double precision). The runtime is averaged over 100 random initial states.

From Table 1, it can be seen that our proposed algorithm B-ADMM is about 4 times faster than iiADMM and about 3 times faster than CVXGEN in this particular MPC problem. In terms of number of iterations, our algorithms is comparable with iiADMM [2].

Null basis is not unique. Suppose we could not obtain a banded null basis, we would have used a non-banded (dense) null basis (MATLAB command $Y = null(F)$) which results in dense matrices in the reduced size QP problem (7). In this case. we have a dense null basis ADMM algorithm (D-ADMM). We see that runtime of D-ADMM are quite big. Its runtime is up to 9 times more than B-ADMM and up to 2 times more than iiADMM. It means that the reduced Hessian approach, although it reduces number of decision variables, is not quite effective if we do not have a banded null basis. In addition, we observe that the two null-space-based ADMM algorithms D-ADMM and B-ADMM have the same number of iterations [3].

Fig. 4 shows the timing per ADMM iteration. It shows that the time per iteration (TPI) of B-ADMM (green diamond-line) is the least, and grows linearly with horizon N. If a dense null basis is used, the growth rate is N^2 (red square dot line). For iiADMM (blue circle-dash line), we implement it in the most efficient way for a fair comparison (see Appendix A for its implementation details), and its TPI is more than B-ADMM's TPI and also grows linearly w.r.t horizon.

Comparing code size shown in Table 2, iiADMM is the best. Code size of B-ADMM is slightly bigger than iiADMM. Code size of these two algorithms grow slowly w.r.t horizon, whereas code size of D-ADMM grows quite fast, mainly because dense matrices in D-ADMM require more memory than banded matrices in B-ADMM. CVXGEN uses second order QP solver and usually generates a big code size comparing with first order methods such as ADMM.

4.3 Random MPC problem test

We generate 500 random MPC problems with polytopic constraints by generating 500 random sets of matrices $A, B, Q, R, G_x, G_u, G_N = G_x$ with the same dimensions as the camera field-of-view quadrotor problem. To ensure that the generated QPs are feasible, we first generate a random initial state and a particular solution z_0 for (4), then substitute z_0 to the inequality (5) and choose g_x, g_N, g_u such that this inequality is feasible.

[2] Stopping criteria for ADMM: max{primal residual, dual residual $\leq 10^{-4}$}. Time for optimality check (about 15% and 25% of total time in iiADMM and B-ADMM) is not counted . Number of iterations of CVXGEN is fixed at 5.

[3] The heuristic step-size is also the same.

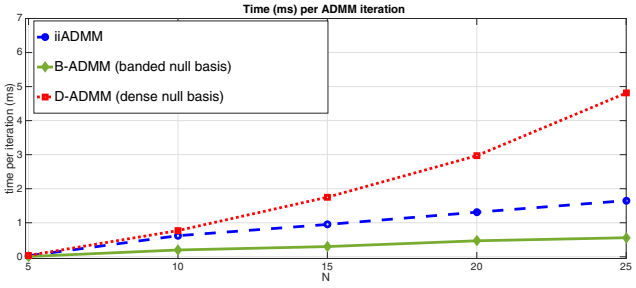

Fig. 4. Time per ADMM iteration versus prediction horizon

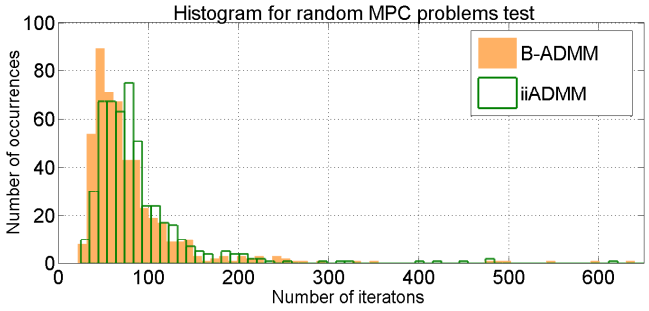

Fig. 5. Histogram of number of iterations for 500 random MPC problems benchmarking ($n_x = 9,\ n_u = 3,\ n_{ix} = 6,\ n_{iu} = 6,\ N = 10$).

Tests are carried out on a PC to obtain the convergence property (in terms of the number of iterations) of each algorithm. Fig. 5 shows the histogram of number of iterations. It suggests that B-ADMM is slightly better than iiADMM, in term of number of iterations. We believe that the reduction in the number of decision variables may have a positive impact on reducing the number of iterations needed of our proposed ADMM algorithm (null space method reduces the number of decision variables, and using the banded null basis reduces the runtime per iteration for the null space method).

4.4 Numerical instability of B-ADMM

Theoretically, we have $\tilde{H} = Y^T H Y \succ 0$ (Assumption 1) and this implies $\tilde{H}_1 \succ 0$ ($\tilde{H}_1 = \tilde{H} + \rho \tilde{G}^T \tilde{G} = Y^T (H + \rho G^T G) Y$). However, in our random test experiment, we encountered one case with $\tilde{H}_1$ becomes only (numerically) positive semi-definite matrix which makes B-ADMM fail at the first step (9) (either matrix inversion or Choleskly factorisation operations fail). This is due to numerical errors that arise in the constructing the banded null basis Y. For example, in the quadrotor example, theoretically we should obtain $FY = 0$ but in fact we obtained $||FY||_F = 1.3.10^{-16}$ ($||.||_F$: Frobenius matrix norm). If Y is not ideal, then $\tilde{H}_1$ possibly becomes (numerically) positive semi-definite matrix. For iiADMM, all random QPs generated are solved.

4.5 Banded null basis in MPC problems

The applicability of our proposed algorithm depends on the existence of a banded null basis. As aforementioned, there has been no theoretical result to confirm whether TBLU will always return a banded null basis for a banded matrix. In the 500 random MPC problems, TBLU always return a banded null basis. However, real world MPC problems usually have dynamics that are not completely random. We then carry out a test over 18 academic and industrial examples which are available in Ferreau and Peyrl (2015) to see if TBLU can give a banded null basis. Test results show that TBLU can always construct a banded null basis in these MPC problems (see Fig. 9).

5. CONCLUSION

We exploit banded structure of equality constraint matrix in the sparse QP formulation of MPC by its banded null basis. We use this banded null basis to reduce the size of sparse QP while maintaining structured matrices, and use ADMM to solve the new QP. The proposed ADMM algorithm, exploiting structured matrices, is about 4 times faster than the current best ADMM variant and 3 times faster than QP solver CVXGEN in a particular MPC problem. Implementation results also show that the proposed ADMM-based algorithm can obtain a convergence rate (number of iterations) comparable with iiADMM. Runtime per iteration of B-ADMM grows linearly w.r.t to horizon. Banded null basis test carried on the 18 MPC problems in literature shows that TBLU always returns a banded null basis in these MPC problems.

REFERENCES

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

Benzi, M., Golub, G.H., and Liesen, J. (2005). Numerical solution of saddle point problems. *Acta Numerica*, 14(1), 1–137.

Berry, M., Heath, M., Kaneko, I., Lawo, M., Plemmons, R., and Ward, R. (1985). An algorithm to compute a sparse basis of the null space. *Numerische Mathematik*, 47(4), 483–504.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1), 1–122.

Dang, T.V., Ling, K.V., and Maciejowski, J.M. (2015). Embedded ADMM-based QP solver for MPC with polytopic constraints. In *Proc. European Control Conference (ECC)*, 3446–3451.

Ding, W., Ganesh, M.R., Severinghaus, R.N., Corso, J.J., and Panagou, D. (2016). Realtime model predictive control for keeping a quadrotor visible on the camera field-of-view of a ground robot. In *Proc. American Control Conference (ACC)*, 2259–2264.

Domahidi, A., Zgraggen, A.U., Zeilinger, M.N., Morari, M., and Jones, C.N. (2012). Efficient interior point methods for multistage problems arising in receding horizon control. In *Proc. 51st IEEE Conference on Decision and Control (CDC)*, 668–674.

Ferreau, H.J. and Peyrl, H. (2015). mpcbenchmarking webpage. In *https://github.com/ferreau/mpcBenchmarking*.

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

Fletcher, R. (1987). *Practical Methods of Optimization*. John Wiley & Sons, New York.

Ghadimi, E., Teixeira, A., Shames, I., and Johansson, M. (2015). Optimal parameter selection for the alternating direction method of multipliers (ADMM): Quadratic problems. *IEEE Transactions on Automatic Control*, 60(3), 644–658.

Giselsson, P. and Boyd, S. (2015). Metric selection in fast dual forward–backward splitting. *Automatica*, 62, 1–10.

Kouzoupis, D., Zanelli, A., Peyrl, H., and Ferreau, H. (2015). Towards proper assessment of QP algorithms for embedded model predictive control. In *Proc. 2015 European Control Conference (ECC)*,, 2609–2616.

Maciejowski, J.M. (2002). *Predictive control: with constraints*. Prentice Hall.

Manickathu, D. (2016). Comparison of ADMM formulations for MPC problems. *ETH Master Thesis*.

Mattingley, J. and Boyd, S. (2012). CVXGEN: A code generator for embedded convex optimization. *Optimization and Engineering*, 13(1), 1–27.

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

Nocedal, J. and Wright, S.J. (2006). *Numerical Optimization*. Springer.

Patrinos, P. and Bemporad, A. (2014). An accelerated dual gradient-projection algorithm for embedded linear model predictive control. *IEEE Transactions on Automatic Control*, 59(1), 18–33.

Richter, S., Jones, C.N., and Morari, M. (2009). Real-time input-constrained MPC using fast gradient methods. In *Proc. of the 48th IEEE Conference on Decision and Control held jointly with the 28th Chinese Control Conference. CDC/CCC 2009.*, 7387–7393.

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

Wright, S.J. (1997). *Primal-dual interior-point methods*. SIAM.

APENDIX A: INDIRECT INDICATOR ADMM FOR MPC

The name *indirect indicator* is given in Manickathu (2016). The setup is as following

$$\underset{z,s}{\text{minimize}} \quad \frac{1}{2}z^T H z + h^T z + \mathcal{I}_{Fz=f} + \mathcal{I}_{s\geq 0} \tag{15}$$

$$\text{s.t.} \quad Gz + s = g \tag{16}$$

For this setup, the first step of ADMM requires solving following KKT equation

$$\begin{bmatrix} H + \rho G^T G & F^T \\ F & 0 \end{bmatrix} \begin{bmatrix} z_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} -(h + \rho G^T(\tau_k + s_k - g)) \\ f \end{bmatrix} \tag{17}$$

where θ_{k+1} is Lagrangian multiplier associated with an equality-constrained minimisation in the first step of

ADMM. The ADMM iterates is then as following

Algorithm 3: iiADMM

$$\mathbf{z_{k+1}} = solve\ (17) \tag{18}$$
$$s_{k+1} = \max\{\mathbf{0}, -Gz_{k+1} - \tau_k + g\} \tag{19}$$
$$\tau_{k+1} = \tau_k + Gz_{k+1} + s_{k+1} - g \tag{20}$$

The step-size is heuristic selected, based on Ghadimi et al. (2015), as

$$\rho^*_{A2} = \left(\sqrt{\lambda_{\min,>0}(GH^{-1}G^T)\lambda_{\max,>0}(GH^{-1}G^T)}\right)^{-1}$$

There are several choices to solve (17). The runtime of iiADMM depends on how we solve this equation. For a fair comparison with our new algorithm, we present here a method for solving (17) that gives iiADMM the least runtime in our experiment. We use a similar method as in Wang and Boyd (2010) to solve (17) by carrying out following steps

$$v_{temp} = h + \rho G^T(s_k + \tau_k - g) \tag{21}$$
$$F\bar{H}_2^{-1}F^T\theta_{k+1} = -F\tilde{H}^{-1}v_{temp} - f \tag{22}$$
$$z_{k+1} = -\bar{H}_2^{-1}(F^T\theta_{k+1} + v_{temp}) \tag{23}$$

where $\bar{H}_2 = H + \rho G^T G$ is a block-diagonal matrix, and $\bar{H}_2^{-1}$ has following form

$$\bar{H}_2^{-1} = \begin{bmatrix} I_N \otimes \begin{bmatrix} \tilde{R} & 0 \\ 0 & \tilde{Q} \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} \tilde{R} & 0 \\ 0 & \tilde{Q}_N \end{bmatrix} \end{bmatrix}$$

where $\tilde{R} = (R + \rho G_u^T G_u)^{-1}, \tilde{Q} = (Q + \rho G_x^T G_x)^{-1}$, $\tilde{Q}_N = (Q_N + \rho G_N^T G_N)^{-1}$. And we also have $\tilde{F} = F\bar{H}_2^{-1}F^T$ in (22) has the banded structure. Then, we solve (22) by a similar way to solving (9) using Cholesky factorization. For (21) and (23), all of involved matrices have special structures and then the computational complexity is $O(N(n_x+n_u)^2)$. Overall, the complexity for solving (17) is $O(N(n_x + n_u)^2)$ if structured matrices are exploited. For the second and third step in iiADMM, the matrices have structured pattern and the exploitation is straight forward. The computation cost of these steps also are $O(N(n_x + n_u)^2)$.

APPENDIX B: PROOF OF LEMMA 1

Recall that $F = \begin{bmatrix} -B & I & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & -A & -B & I & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & -A & -B & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \\ 0 & 0 & 0 & \cdots & 0 & -A & -B & I \end{bmatrix}$ of dimension $(Nn_x) \times (N(n_x + n_u))$.

We have each "row" (a block of rows) of F contains identity matrix. Using elementary column operations, we can transform F to the row echelon form. Then, the full row rank of F is directly confirmed. We then have $rank(F) = Nn_x$. By rank theorem, we have the dimension of its null space is $N(n_x + n_u) - Nn_x = Nn_u$.

APPENDIX C: ADDITIONAL RANDOM TESTS

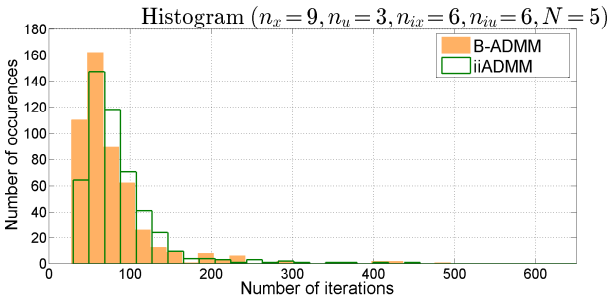

Fig. 6. Test #2: Same dimension as quadrotor, $N = 5$

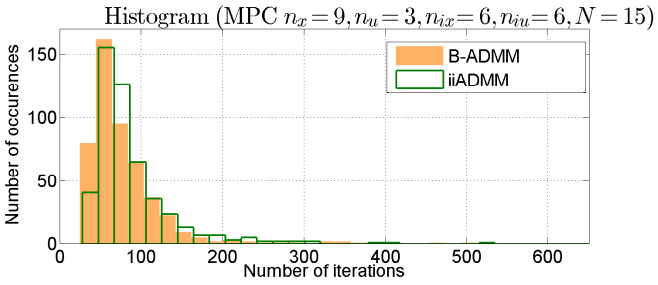

Fig. 7. Test #3: same dimension as quadrotor, $N = 15$

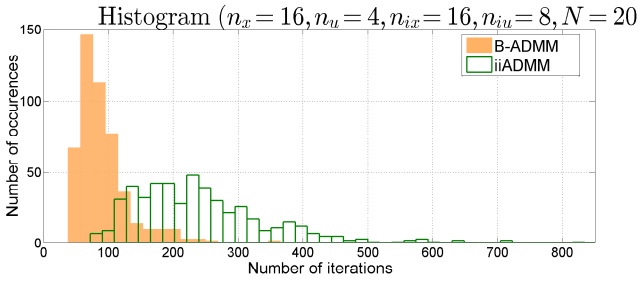

Fig. 8. Test #4: systems with $n_u << n_x$: $n_x = 16, n_u = 4, n_{ix} = 16, n_{iu} = 8,\ N = 20$

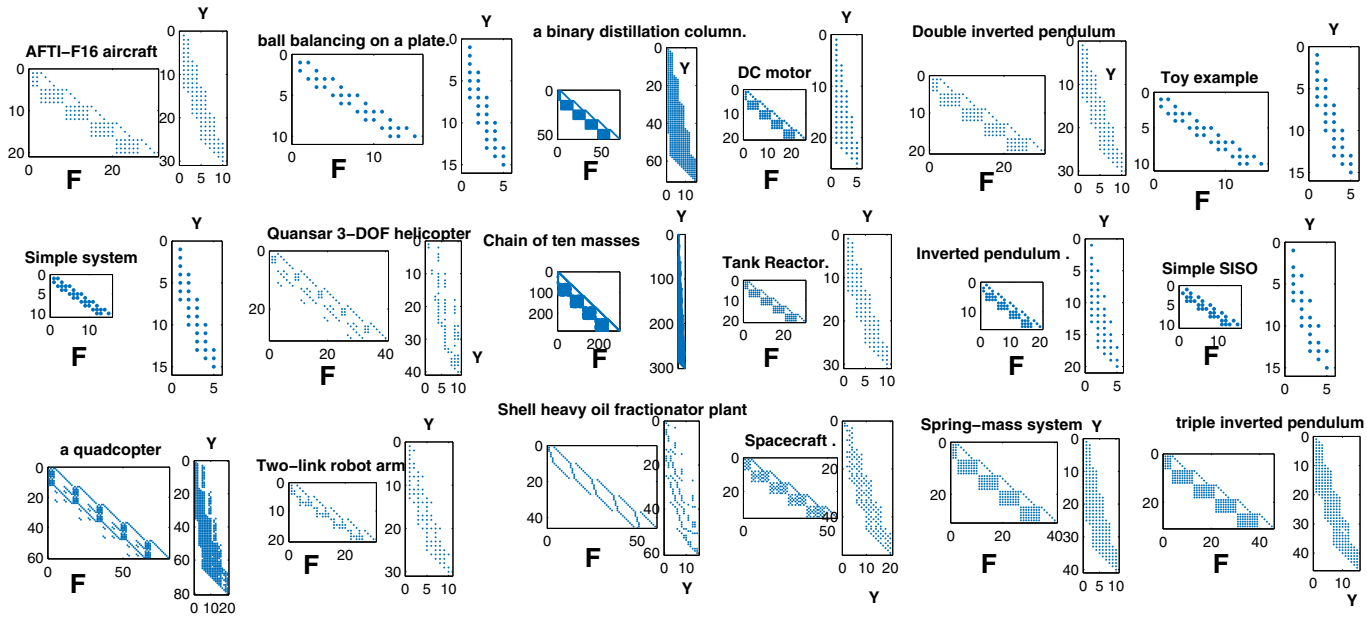

Fig. 9. Banded null basis in 18 MPC problems ($N = 5$)