Fast Model Predictive Control Using Efficient Newton Step Computation in Active-Set Method with Tailored Move Blocking

Jiri Burant [1], *Pavel Otta*[1]

[1]Department of Control Engineering, CTU Prague, Czech Republic

{buranji1,ottapav1}@fel.cvut.cz

Abstract. *Model predictive control is an optimization based methodology, which is well known in academia but only slowly adapting in industry. The main reason is an enormous computational burden required compared to other traditional control methods. We present a fast implementation of MPC using active-set method. In this contribution, computational demand is reduced significantly by efficient Newton step and tailored move blocking procedure.*

Keywords

Model Predictive Control, Reference Tracking, Active-Set Method, Newton Step, Move Blocking.

1. Introduction

Model Predictive Control (MPC) is optimization based control method. Main advantages of this method is ability to concern physical limits of the controlled plant and very systematic control design. Therefore, control design for MIMO system is as simple as design for SISO system.

In each sampling period, open-loop control over finite prediction horizon is find by solving an finite optimization problem numerically. The feedback is established by the receding horizon. Receding horizon meas that every sampling period the sequence of optimal control is re-optimized for the receded horizon. Optimization is parametrized by currently measured state. Each step only the first control from the control sequence is applied.

The aim of the MPC is to achieve a control minimizing given cost function, satisfying controlled system dynamics and physical or other additional constraints. However, optimization can be very computationally demanding, since there might be many inputs and states of the system or long prediction horizon is required. The method presented in this paper is focused to minimize the computational and memory requirements for the computation of the Newton step, which usually represents the highest computation load of the MPC algorithm.

In last decade, many outstanding algorithms specialized to MPC have been developed. Richter's FGP [1] – a fast gradient projection algorithm solving condensed problem formulation, *qpOASES* [2] – an higly optimized implementation of multiparametric active-set method focused on condensed formulation as well.

Recently, algorithms for sparse MPC problem formulation and exploiting problem structure have been developed, namely, *qpDUNES* [3] – an implementation of dual Newton step solving the MPC problem via dual decomposition, *fastMPC* [4] – a barrier interior-point algorithm using Newton step solved by tailored Cholesky decomposition, *FORCES* [5] – code generated interior-point method, *ADMMmpc* [6] – an ADMM algorithm using interior-point method in the inner loop. Note that, to the best of our knowledge, efficient active-set method exploiting problem structure have not been reported yet.

In condensed formulation of the problem one has less optimization variables, but problem itself is more complex (ill-conditioned). On the other hand, in sparse formulation one have much more variables, but problem has given sparsity pattern with usually less nonzero elements and less complex (well-conditioned). Note that for condensed problem generic solvers (e.g. quadprog or MOSEK) can be used or a solver can be utilized for MPC problem. For sparse formulation of the MPC problem algorithms *have to* be utilized. That is why the condensed formulation was more popular in the past. More details to the condensed and sparse formulation of the MPC problem is disused later.

Implementation of Newton step plays an important role for the second order method (e.g. interior-point, active-set) based solvers. Our contribution is the implementation of efficient Newton step for active-set method tailored for MPC. Furthermore, we extend basic optimization criterion in more practical manner and refer that output reference tracking satisfying output soft (or hard) constraints cannot be optimized at once in efficient way and that removing weight on input rate from a cost function makes algorithm significantly faster. Move blocking strategy utilized for (all) sparse algorithms is also introduced in this paper.

This paper presents an approach for fast MPC using the active-Set method. The problem is divided into two optimization tasks. In Section 2, the first task of constant reference tracking is introduced. In Section 3, the second task of

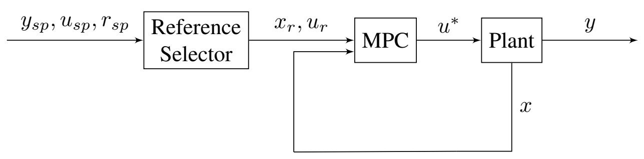


Fig. 1: Proposed controller setup.

the MPC dynamic regulator is described. In the next section, active-set method is sketch to show background for Newton step developed in Section 5. Finally in Section 6, move blocking procedure tailored for sparse structure of the proposed problem is pioneered. The proposed solution is declared to be advantageous for large prediction horizons by numerical experiment on random systems in Section 7.

In this paper, we assume discrete linear time-invariant (LTI) system

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t), \quad t > 0. \end{aligned} \tag{1}$$

2. Constant Reference Tracking

The problem presented later in this paper can handle the input and state reference only. However, in real applications, the common requirement is the output reference tracking. This issue is surpassed by modification of the control scheme – adding feedforward tracker of constant reference. The overall control scheme is shown in Fig. 1.

The output tracking optimization problem that intermediate transition from output reference to input and state reference tracking is formed as

$$\min_{x_r,u_r} \quad \frac{1}{2}\|Cx_r - y_{sp}\|^2_{Q_s} + \frac{1}{2}\|u_r - u_{sp}\|^2_{R_s} \tag{2a}$$

$$\text{s.t.} \quad \begin{pmatrix} I-A & -B \\ HC & 0 \end{pmatrix}\begin{pmatrix} x_r \\ u_r \end{pmatrix} = \begin{pmatrix} 0 \\ r_{sp} \end{pmatrix} \tag{2b}$$

$$Eu_r \le e \tag{2c}$$

$$FCx_r \le f, \tag{2d}$$

where y_{sp} is the output reference, weight matrix Q_s is positive semidefinite and R_s is positive definite and $r_{sp} = Hy_{sp}$ is measured to controlled output mapping. Matrices E, F and vectors e, f represent general constraints on inputs and outputs.

By solving this minimization problem beforehand, one can calculate the x_r and u_r, that will be used in the dynamic regulator optimization problem as the reference variables hereafter.

More extensive description of this approach can be found for example in [7].

3. Dynamic Regulator

There are many ways of defining the cost function, depending on the particular demands, in this paper the following formulation is used.

$$\begin{aligned} J(x_k, u_k, \widehat{y}_k) := &\frac{1}{2}\|x_{n_p} - x_r\|^2_{Q_{n_p}} + \frac{1}{2}\|Cx_{n_p} - \widehat{y}_{n_p}\|^2_{P_{n_p}} \\ &+ \frac{1}{2}\sum_{k=1}^{n_p-1} \|x_k - x_r\|^2_Q + \|Cx_k - \widehat{y}_k\|^2_P \\ &+ \frac{1}{2}\sum_{k=0}^{n_p-1} \|\Delta u_k\|^2_S + \|u_k - u_r\|^2_R, \end{aligned} \tag{3}$$

where $x_k \in \mathcal{R}^{n_x}$ is the vector of n_x states. Subscript k denotes time instant of the prediction horizon. u_k is the vector of n_u inputs, Δu_k is a difference between two following control action. n_p denotes the length of prediction horizon. x_r is the state reference, u_r is the input reference. Symbol $\widehat{y}_k$ represents the hard constrained outputs, i.e. output limit value from which further deviation of soft constrained output is penalized. Matrices Q, R, S, P are the weight matrices, they are diagonal and positive definite.

The cost function consists of state and input reference tracking, rate of input change cost and output soft constraints penalty.

$$\min_{x_k,u_k,\widehat{y}_k} \quad J(x_k, u_k, \widehat{y}_k) \tag{4a}$$

$$\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k, \quad x_0 = x(t) \tag{4b}$$

$$\Delta u_k = u_k - u_{k-1}, \quad u_{-1} = u(t-1) \tag{4c}$$

$$\underline{u}_k \le u_k \le \overline{u}_k, \quad k = 0, \dots, n_p - 1 \tag{4d}$$

$$\underline{y}_k \le \widehat{y}_k \le \overline{y}_k, \quad k = 1, \dots, n_p \tag{4e}$$

The symbols $\underline{u}_k, \overline{u}_k$ represent the constraints on the inputs, similarly symbols $\underline{y}_k, \overline{y}_k$ represent the constraints on the variable $\widehat{y}_k$.

First of all, the problem is rewritten into the vector form.

$$\begin{aligned} J(x, u, \widehat{y}) = &\frac{1}{2}\|x - x_R\|^2_{\mathbb{Q}} + \frac{1}{2}\|u - u_R\|^2_{\mathbb{R}} \\ &+ \frac{1}{2}\|\Delta u\|^2_{\mathbb{S}} + \frac{1}{2}\|Cx - \widehat{y}\|^2_{\mathbb{P}} \end{aligned} \tag{5}$$

The vectors and matrices are formed by stacking the values for every k, forming larger vectors(matrices). $x = [x_1^T, x_2^T, \dots, x_{n_p}^T]^T$, $u = [u_0^T, u_1^T, \dots, u_{n_p-1}^T]^T$, $x_R = [x_r^T, \dots, x_r^T]^T$, $u_R = [u_r^T, \dots, u_r^T]^T$, $\Delta u = [\Delta u_0^T, \Delta u_1^T, \dots, \Delta u_{n_p-1}^T]^T$ and $\widehat{y} = [\widehat{y}_1^T, \widehat{y}_2^T, \dots, \widehat{y}_{n_p}^T]^T$, $\mathbb{Q} = \text{blkdiag}(I_{n_p} \otimes Q, Q_{n_p})$, $\mathbb{R} = I_{n_p} \otimes R$, $\mathbb{S} = I_{n_p} \otimes S$, $\mathbb{P} = \text{blkdiag}(I_{n_p} \otimes P, P_{n_p})$

Condensed Formulation

One can formulate condensed optimization problem from (10). The problem is then solved only in control variables and also those, which have to be bounded, namely u and $\widehat{y}$.

Next step is to transcript the relation between u and x, defined by the system dynamics into compact matrix form.

$$x = \widehat{P}x_0 + \widehat{V}u \tag{6}$$

$$\widehat{P} = \begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^{n_p} \end{bmatrix}, \widehat{V} = \begin{bmatrix} B & 0 & 0 & \dots & 0 \\ AB & B & 0 & \dots & 0 \\ A^2B & AB & B & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{n_p-1}B & A^{n_p-2}B & A^{n_p-3}B & \dots & B \end{bmatrix}$$

By substituting (6) into criteria (5), the final equation in matrix form is obtained.

$$\min_{u,\widehat{y}} \quad \frac{1}{2}\begin{bmatrix} u \\ \widehat{y} \end{bmatrix}^T H \begin{bmatrix} u \\ \widehat{y} \end{bmatrix} + \begin{bmatrix} u \\ \widehat{y} \end{bmatrix}^T f \tag{7}$$

$$\text{s.t.} \quad \underline{u} \le u \le \overline{u} \tag{8}$$

$$\underline{y} \le \widehat{y} \le \overline{y} \tag{9}$$

Where $H = \begin{bmatrix} \widehat{V}^T\mathbb{Q}\widehat{V} + \mathbb{R} + \mathbb{K}^T\mathbb{S}\mathbb{K} & -\widehat{V}^T\mathbb{C}^T\mathbb{P} \\ -\mathbb{P}\mathbb{C}\widehat{V} & \mathbb{P} \end{bmatrix}$

and $f = \begin{bmatrix} \widehat{V}^T\mathbb{Q}\widehat{P}x_0 - \widehat{V}^T\mathbb{Q}x_R - \mathbb{R}u_R - \mathbb{K}^T\mathbb{S}e \\ -\mathbb{P}\mathbb{C}\widehat{P}x_0 \end{bmatrix}$.

Note that the new problem formulation is minimized just over the variable u and $\widehat{y}$, the size of the Hessian H is therefore $n_p \times (n_u + n_y)$. Also note that output constraints remain simple – this is of our particular interest due to specific algorithm in which Newton step computation will be used. Alternatively, when common input blocking is considered, the problem size is $n_c \times (n_u + n_y)$, where n_c is control horizon.

Sparse Formulation

The problem formulation might be rewritten into vector form, which is more suitable for the matrix computations.

$$\min_{x,u,\widehat{y}} \quad J(x,u,\widehat{y}) \tag{10a}$$

$$\text{s.t.} \quad 0 = \mathbb{A}x + \mathbb{B}u + d \tag{10b}$$

$$\Delta u = \mathbb{K}u - e \tag{10c}$$

$$\underline{u} \le u \le \overline{u} \tag{10d}$$

$$\underline{y} \le \widehat{y} \le \overline{y} \tag{10e}$$

Where $\mathbb{B} = I_{n_p} \otimes B, \mathbb{C} = I_{n_p} \otimes C$,

The structure of the other matrices and vectors is following.

$$\mathbb{A} = \begin{bmatrix} -I & & & & \\ A & -I & & & \\ & A & -I & & \\ & & \ddots & \ddots & \\ & & & A & -I \end{bmatrix}, d = \begin{bmatrix} Ax_0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\mathbb{K} = \begin{bmatrix} I & & & & \\ -I & I & & & \\ & -I & I & & \\ & & \ddots & \ddots & \\ & & & -I & I \end{bmatrix}, e = \begin{bmatrix} Iu_{-1} \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

4. Active-Set Method

To solve the MPC problem with constraints, the active-set method has been chosen. It is an iterative optimization method, which within each iteration changes a working set – set of constraints that are active at the current iteration. The goal is to find the active set – set that contains exactly those constraints that are active in the solution. In every step, a constraint is either added or removed from this set. The extensive description of this method can be found in the literature (e.g. [8]).

To compute the Newton step within the active-set method from the formulation presented in (10), the step in the variables u and $\widehat{y}$ is introduced.

$$u = u^- + p_u \tag{11a}$$

$$\widehat{y} = \widehat{y}^- + p_{\widehat{y}}. \tag{11b}$$

This step is carried out in each iteration, the p_u and $p_{\widehat{y}}$ are computed, subsequently the variables u and $\widehat{y}$ are updated. Variables u^- and $\widehat{y}^-$ are the values of u and y from the previous iteration. This change inequality constraints in (10) into equality constraints

$$\mathbb{F}_i p_u = 0 \tag{12a}$$

$$\mathbb{G}_i p_{\widehat{y}} = 0. \tag{12b}$$

Matrices $\mathbb{F}_i, \mathbb{G}_i$ depends on set of active constraints at i-th iteration of active-set algorithm. They have as many rows as number of active constraints with one at each of them, i.e. the matrices have full row rank.

5. Newton Step Computation

KKT System

To apply the constraints, the Lagrange multipliers are introduced.

$$
\begin{aligned}
L :=& \frac{1}{2}(\|x - x_R\|^2_{\mathbb{Q}} + \|u^- + p_u - u_R\|^2_{\mathbb{R}} + \|\Delta u\|^2_{\mathbb{S}} + \\
& \|\mathbb{C}x - \widehat{y}^- - p_{\widehat{y}}\|^2_{\mathbb{P}}) + \lambda^T(\mathbb{A}x + \mathbb{B}u^- + \mathbb{B}p_u + d) + \\
& \mu^T(\mathbb{K}u^- + \mathbb{K}p_u - e - \Delta u) + \gamma^T \mathbb{F}p_u + \xi^T \mathbb{G}p_{\widehat{y}}. \quad (13)
\end{aligned}
$$

By the partial derivations of the previous equation (13), the KKT matrix is obtained as follows

$$
\begin{bmatrix}
\mathbb{R} & & & & \mathbb{B}^T & \mathbb{K}^T & \mathbb{F}^T & \\
 & \mathbb{P} & & -\mathbb{P}\mathbb{C} & & & & \mathbb{G}^T \\
 & & \mathbb{S} & & & -\mathbb{I} & & \\
 & -\mathbb{C}^T\mathbb{P} & & \mathbb{Q} + \mathbb{C}^T\mathbb{P}\mathbb{C} & \mathbb{A}^T & & & \\
\mathbb{B} & & & \mathbb{A} & & & & \\
\mathbb{K} & & -\mathbb{I} & & & & & \\
\mathbb{F} & & & & & & & \\
 & \mathbb{G} & & & & & &
\end{bmatrix}
\begin{bmatrix} p_u \\ p_{\widehat{y}} \\ \Delta u \\ x \\ \lambda \\ \mu \\ \gamma \\ \xi \end{bmatrix}
=
\begin{bmatrix} -\mathbb{R}(u^- - u_R) \\ -\mathbb{P}\widehat{y}^- \\ 0 \\ \mathbb{Q}x_R + \mathbb{C}^T\mathbb{P}\widehat{y}^- \\ -\mathbb{B}u^- - d \\ -\mathbb{K}u^- + e \\ 0 \\ 0 \end{bmatrix}
$$

Remark 1 *From the KKT matrix (namely $\mathbb{Q}+\mathbb{C}^T\mathbb{P}\mathbb{C}$) it can be seen that output tracking and (soft) constrained output at the same time might cause singularity of that matrix. It would prevent us from eliminating states later.*

Reduction by Null-Space Method

Using the null-space method, the variables γ and ξ are eliminated. The matrices Z and Y are introduced, thanks to them the problem will be altered according to the active constraints. Then the first equation is multiplied by $\mathbb{Z}^T$, and the second equation is multiplied by $\mathbb{Y}^T$.

$$p_u = \mathbb{Z}\tilde{p}_u \quad (14a)$$

$$p_{\widehat{y}} = \mathbb{Y}\tilde{p}_{\widehat{y}} \quad (14b)$$

The resulting system of equations is following.

$$
\begin{bmatrix}
\Psi & & & & \Gamma^T & \Pi^T \\
 & \Theta & & \Upsilon^T & & \\
 & & \mathbb{S} & & & -\mathbb{I} \\
 & \Upsilon & & \mathbb{M} & \mathbb{A}^T & \\
\Gamma & & & \mathbb{A} & & \\
\Pi & & -\mathbb{I} & & &
\end{bmatrix}
\begin{bmatrix} \tilde{p}_u \\ p_{yz} \\ \Delta u \\ x \\ \lambda \\ \mu \end{bmatrix}
=
\begin{bmatrix} k \\ l \\ 0 \\ m \\ n \\ q \end{bmatrix}
\quad (15)
$$

Where the matrices have following meaning.

$$
\begin{aligned}
\Psi &= \mathbb{Z}^T\mathbb{R}\mathbb{Z}, & \Gamma &= \mathbb{B}\mathbb{Z}, & \Pi &= \mathbb{K}\mathbb{Z}, \\
\Theta &= \mathbb{Y}^T\mathbb{P}\mathbb{Y}, & \Upsilon &= -\mathbb{C}^T\mathbb{P}\mathbb{Y}, & \mathbb{M} &= \mathbb{Q} + \mathbb{C}^T\mathbb{P}\mathbb{C}
\end{aligned}
\quad (16)
$$

Matrix Ψ is diagonal, matrix Γ is block diagonal, rectangular, matrix Π is block di-diagonal (it has blocks on the main diagonal and under it). Matrix M is block diagonal and square, matrix Θ is square, diagonal, matrix V is block diagonal, rectangular.

$$
\begin{aligned}
k &= -\mathbb{Z}^T\mathbb{R}(u^- - u_R), \quad l = -\mathbb{Y}^T\mathbb{P}\widehat{y}^- \\
m &= \mathbb{Q}x_R + \mathbb{C}^T\mathbb{P}\widehat{y}^-, \quad n = -\mathbb{B}u^- - d, \quad q = -\mathbb{K}u^- + e
\end{aligned}
\quad (17)
$$

Matrix equation (15) is then reduced to smaller problem, depending only on the dual variables λ a μ. In the final algorithm, this smaller problem is solved first. The result are the values of λ and μ, which are subsequently used to compute the rest of the variables.

Reduction by Range-Space Method

First of all, it is needed to express the variables $\tilde{p}_u, \tilde{p}_{\widehat{y}}, \Delta u, x$ from the equation (15).

$$
\begin{aligned}
\tilde{p}_u &= \Psi^{-1}(k - \Gamma^T\lambda - \Pi^T\mu) && (18a) \\
\tilde{p}_{\widehat{y}} &= \Theta^{-1}(l - \Upsilon^T x) && (18b) \\
\Delta u &= \mathbb{S}^{-1}\mu && (18c) \\
x &= -\mathbb{M}^{-1}(\mathbb{A}^T\lambda + \Upsilon\tilde{p}_{\widehat{y}} - m) && (18d)
\end{aligned}
$$

The next step is to express the variable x using just variables λ and μ, because the equation for x will be substituted into the equation for λ. Therefore, the equation (18b) will be substituted in the equation for (18d).

$$x = -\mathbb{M}^{-1}(\mathbb{A}^T\lambda + \Upsilon\Theta^{-1}(l - \Upsilon^T x) - m) \quad (19)$$

After the adjustments, the equation for x is dependent only on λ.

$$x = -[\mathbb{M} - \Upsilon\Theta^{-1}\Upsilon^T]^{-1}(\mathbb{A}^T\lambda + \Upsilon\Theta^{-1}l - m) \quad (20)$$

To solve this equation, it is necessary that the member $[\mathbb{M} - \Upsilon\Theta^{-1}\Upsilon^T]$ is positively definite. This is accomplished, which can be seen after examining these matrices, $\mathbb{Q}+\mathbb{C}^T\mathbb{P}\mathbb{C} - \mathbb{C}^T\mathbb{P}\mathbb{Y}(\mathbb{Y}\mathbb{P}\mathbb{Y}^T)^{-1}\mathbb{Y}^T\mathbb{P}\mathbb{C}$. It is plain to see that the worst case is when no inputs are blocked. In this case the expression $\mathbb{Q}$ is the only remaining member and matrix $\mathbb{Q}$ is positively definite.

The system of equations for λ and μ has the following form.

$$
\begin{bmatrix} \Phi & \Sigma \\ \Sigma^T & \Omega \end{bmatrix}
\begin{bmatrix} \mu \\ \lambda \end{bmatrix}
=
\begin{bmatrix} \psi \\ \sigma \end{bmatrix}
\quad (21)
$$

Remark 2 *Complexity of the problem drops significantly when weight on rate of input change is removed from the cost function.*

The matrices of those equations have following meaning.

$$\Omega = \Gamma\Psi^{-1}\Gamma^T + \mathbb{A}[\mathbb{M} - \Upsilon\Theta^{-1}\Upsilon^T]^{-1}\mathbb{A}^T \tag{22a}$$
$$\Sigma = \Pi\Psi^{-1}\Gamma^T \tag{22b}$$
$$\Phi = \Pi\Psi^{-1}\Pi^T + \mathbb{S}^{-1} \tag{22c}$$

$$\sigma = -\mathbb{A}[\mathbb{M} - \Upsilon\Theta^{-1}V^T]^{-1}(\Upsilon\Theta^{-1}l - m) - \Gamma\Psi^{-1}k - n \tag{22d}$$
$$\psi = \Pi\Psi^{-1}k - q \tag{22e}$$

Matrix Ω is block tridiagonal, matrix Σ is block di-diagonal, it has blocks on the main diagonal and blocks under the main diagonal. Matrix Φ is block tridiagonal.

Remark 3 *The structure of the problem is extraordinary – it can be rearranged into block-pentagonal matrix instead. As a consequence, a decomposition exists such that the structure remains.*

There are efficient algorithms for solving block-pentagonal either iterative or direct – for example Cholesky decomposition followed by substitution and back-substitution can be used.

6. Tailored Move Blocking

In case of the move blocking, it is possible to exclude from optimization those inputs, which are the same as the inputs in the previous steps. Thanks to that, the dimension of the problem is reduced. Along with the inputs, some states are also excluded from the optimization. It is possible to use the blocking for several sequences of different length.

The blocking procedure is then done due to the following transform

$$x = Mx_B + NGu_B + d \tag{23}$$
$$u = Gu_B, \tag{24}$$

where G is common blocking matrix (e.g. [9]).

By applying this to (10) the following reduced optimization is obtained.

$$\begin{aligned}
\min_{x, u_B, \widehat{y}_B} \quad & J(x_B, u_B, \widehat{y}_B) \\
\text{s.t.} \quad & 0 = \mathbb{A}_B x_B + \mathbb{B}_B u_B + d_B \\
& \Delta u_B = \mathbb{K} G u_B - e \\
& \underline{u}_B \le u_B \le \overline{u}_B \\
& \underline{y}_B \le \widehat{y}_B \le \overline{y}_B,
\end{aligned} \tag{25}$$

where $\mathbb{A}_B = \mathbb{A}M$, $\mathbb{B}_B = (\mathbb{A}N + \mathbb{B})G$, $d_B = (\mathbb{A} + I)d$ and with cost function defined by

$$\begin{aligned}
J(x_B, u_B, \widehat{y}_B) = & \frac{1}{2}\|Mx_B + NGu_B + d - x_{R_B}\|_{\mathbb{Q}}^2 \\
& + \frac{1}{2}\|Gu_B - u_{R_B}\|_{\mathbb{R}}^2 + \frac{1}{2}\|\Delta u_B\|_{\mathbb{S}}^2 \\
& + \frac{1}{2}\|Cx_B - \widehat{y}_B\|_{\mathbb{P}}^2
\end{aligned} \tag{26}$$

Since all matrices in (25) will reduce their sizes, the computational demand for Newton step is decreased as well.

Remark 4 *Since in the first term of* (26) *new cross term occurs, the problem becomes slightly more complex.*

Example Let us show one case for which input blocks are determined by $n_B = [1, 2, 3]$, obviously $n_p = 6$ and $n_c = 3$ then. For this case the vectors and matrices in (23) will look like $x_B = [x_{1_B}^T, x_{3_B}^T, x_{6_B}^T]^T$, $u_B = [u_{0_B}^T, u_{1_B}^T, u_{2_B}^T]^T$, $\widehat{y}_B = [\widehat{y}_{1_B}^T, \widehat{y}_{3_B}^T, \widehat{y}_{6_B}^T]^T$

$$M = \begin{bmatrix} I & & \\ A & & \\ & I & \\ & A & \\ & A^2 & \\ & & I \end{bmatrix}, \; N = \begin{bmatrix} 0 & & & & & \\ & B & & & & \\ & & 0 & & & \\ & & & B & & \\ & & & AB & B & \\ & & & & & 0 \end{bmatrix}, \; G = \begin{bmatrix} I & & \\ & I & \\ & I & \\ & & I \\ & & I \\ & & I \end{bmatrix}$$

This example is illustrated in Fig. 2.

Remark 5 *Size of the first block n_{B_1} has to be one. Otherwise, M contains zeros in the first row, which cause singularity of $\mathbb{A}_B$.*

Remark 6 *As shown in the example, number of state samples has to be at least equal to the number of input samples. Position of this minimum samples is given by size of input blocks. Other state samples can be added without loss of sparsity pattern.*

7. Numerical Experiment

In order to test the effectivity of the approach, presented in this paper, it has been compared with equivalent Newton step computation in condensed formulation [10]. The comparison has been provided in MATLAB environment. Simulations were run on quad-core processor i5-4460 CPU @ 3.20GHz. A series of tests has been ran on ten randomly generated systems with the following properties, $n_x = 10$, $n_u = 10$, $n_y = 10$. The systems were generated by MATLAB function *drss()*. Prediction horizon n_p was in range $[2, 80]$. The number of active constraints were tested in full range.

The images in Fig. 3 show that the complexity of the problem for condensed formulation is strongly dependent on the number of active constraints compared to the proposed sparse formulation. Therefore, for no active constraint (the

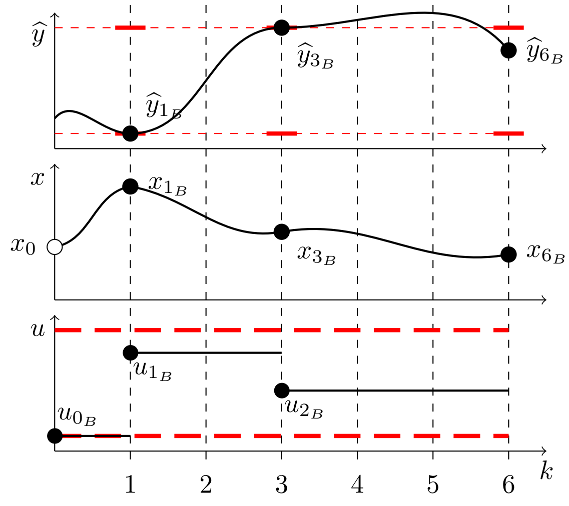
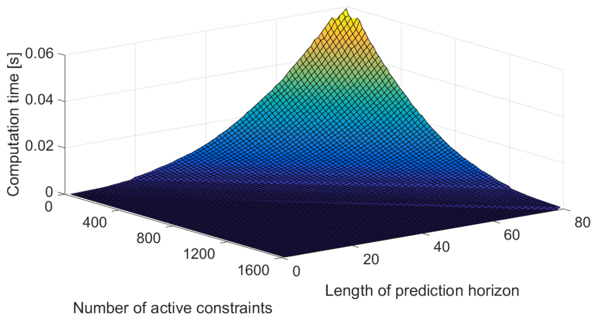
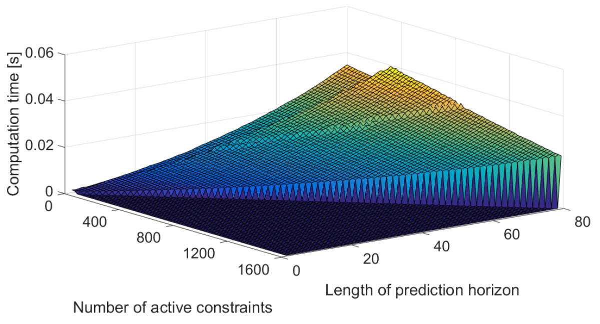


Fig. 2: Example on move blocking.

worst case scenario) the proposed algorithm performed better for $n_p > 40$ – note that this value is strongly dependent on the particular implementation – however, there is a value of n_p from which the proposed computation procedure is faster. On the other hand, the condensed Newton step remain faster when many constraints are permanently active.



(a) Condensed setup



(b) Proposed setup

Fig. 3: Trends in computation times depending on prediction horizon length and number of active constraints (implemented in MATLAB).

8. Conclusion

At the beginning, this paper gently follows [4] where a similar idea of efficient Newton step is presented for interior-point method. We adopt the idea for active-set method and extends it for more practical problems yet computationally still reasonable. Furthermore, we show that this procedure cannot be used for output tracking satisfying output constraints at the same time. Therefore, we encourage solve the problems separately, namely constant reference tracking (feedforward part) on one hand and dynamic regulation (feedback part) on the other hand. Last but not least, extension of move blocking for the proposed method is discussed. Move blocking is a well known strategy commonly used for problems in condensed form to decrease problem size even more. To the best of our knowledge, the strategy have not been adopted for problems in the sparse form yet. Thus as the proposed problem may be still computationally challenging, we introduce move blocking technique tailored for sparse formulation of the problem in such a way that the problem sparsity structure remain untouched.

The results presented in this paper are still preliminary, more details on implementation of overall active-set algorithm and both theoretical and practical proves of efficiency will be addressed in forthcoming journal publication.

Acknowledgements

This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS16/232/OHK3/3T/13.

References

[1] S. Richter, S. Mariethoz, and M. Morari, “High-speed online MPC based on a fast gradient method applied to power converter control,” in *American Control Conference (ACC)*, 2010, pp. 4737–4743.

[2] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, “qpOASES: A parametric active-set algorithm for quadratic programming,” *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[3] J. V. Frasch, S. Sager, and M. Diehl, “A parallel quadratic programming method for dynamic optimization problems,” *Mathematical Programming Computation*, vol. 7, no. 3, pp. 289–329, 2015.

[4] Y. Wang and S. Boyd, “Fast model predictive control using online optimization,” *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267–278, 2010.

[5] A. Domahidi, “FORCES: Fast optimization for real-time control on embedded systems,” http://forces.ethz.ch, Oct. 2012.

[6] L. E. Sokoler, G. Frison, M. S. Andersen, and J. B. Jrgensen, “Input-constrained model predictive control via the alternating direction method of multipliers,” in *Control Conference (ECC), 2014 European*, June 2014, pp. 115–120.

[7] L. Shead, K. Muske, and J. Rossiter, “Conditions for which mpc fails to converge to the correct target,” in *Proc IFAC World Congress*, 2008, pp. 6968–6973.

[8] J. Nocedal and S. J. Wright, *Numerical Optimization, Second Edition*. Springer New York, 2006.

[9] R. Cagienard, P. Grieder, E. Kerrigan, and M. Morari, “Move blocking strategies in receding horizon control,” *Journal of Process Control*, vol. 17, no. 6, pp. 563–570, 2007.

[10] O. Šantin and V. Havlena, “Combined partial conjugate gradient and gradient projection solver for MPC,” *IEEE International Conference on Control Applications (CCA)*, pp. 1270–1275, 2011.