MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Projected Preconditioning within a Block-Sparse Active-Set Method for MPC

Quirynen, R.; Knyazev, A.; Di Cairano, S.

TR2018-122 August 25, 2018

Abstract

Model predictive control (MPC) often requires solving an optimal control structuredquadratic program (QP), possibly based on an online linearization at each sampling instant. Block-tridiagonal preconditioners have been proposed, combined with the minimal residual (MINRES) method, to result in a simple but efficient implementation of a sparse activeset strategy for fast MPC. This paper presents an improved variant of this PRESAS algorithm, by using a projected preconditioned conjugate gradient (PPCG) method. Based on a standalone C code implementation and using an ARM Cortex-A7 processor, we illustrate the performance of the proposed solver against the current state of the art for embedded predictive control.

IFAC Conference on Nonlinear Model Predictive Control

Projected Preconditioning within a Block-Sparse Active-Set Method for MPC

Rien Quirynen * **Andrew Knyazev** * **Stefano Di Cairano** *

* *Mechatronics, Mitsubishi Electric Research Laboratories, Cambridge, MA, 02139, USA. (e-mail: quirynen@merl.com).*

Abstract: Model predictive control (MPC) often requires solving an optimal control structured quadratic program (QP), possibly based on an online linearization at each sampling instant. Block-tridiagonal preconditioners have been proposed, combined with the minimal residual (MINRES) method, to result in a simple but efficient implementation of a sparse active-set strategy for fast MPC. This paper presents an improved variant of this PRESAS algorithm, by using a projected preconditioned conjugate gradient (PPCG) method. Based on a standalone C code implementation and using an ARM Cortex-A7 processor, we illustrate the performance of the proposed solver against the current state of the art for embedded predictive control.

Keywords: Predictive control, Quadratic programming, Numerical algorithms

1. INTRODUCTION

A block-sparse problem structure arises in a linear or linear time-varying formulation of model predictive control (MPC) (Rawlings et al., 2017). A similarly structured quadratic program (QP) forms the subproblem within a sequential quadratic programming (SQP) method for nonlinear MPC (Ferreau et al., 2017). We are interested in solving the following formulation of a convex constrained linear-quadratic optimal control problem (OCP)

$$\min_{X,U} \quad \sum_{k=0}^{N-1} \frac{1}{2} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^\top \begin{bmatrix} Q_k & S_k^\top \\ S_k & R_k \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} + \begin{bmatrix} q_k \\ r_k \end{bmatrix}^\top \begin{bmatrix} x_k \\ u_k \end{bmatrix} \tag{1a}$$

$$+ \frac{1}{2} x_N^\top Q_N x_N + q_N^\top x_N \tag{1b}$$

$$\text{s.t.} \quad x_0 = \hat{x}_0, \tag{1c}$$

$$x_{k+1} = a_k + A_k x_k + B_k u_k, \ k = 0, \dots, N-1, \tag{1d}$$

$$0 \geq d_k + D_k^{\mathrm{x}} x_k + D_k^{\mathrm{u}} u_k, \ k = 0, \dots, N, \tag{1e}$$

where we define the state vectors as $x_k \in \mathbb{R}^{n_{\mathrm{x}}}$, the control inputs as $u_k \in \mathbb{R}^{n_{\mathrm{u}}}$ and the cost matrices as $Q_k \in \mathbb{R}^{n_{\mathrm{x}} \times n_{\mathrm{x}}}$, $S_k \in \mathbb{R}^{n_{\mathrm{u}} \times n_{\mathrm{x}}}$ and $R_k \in \mathbb{R}^{n_{\mathrm{u}} \times n_{\mathrm{u}}}$. The constraints include the system dynamics with $A_k \in \mathbb{R}^{n_{\mathrm{x}} \times n_{\mathrm{x}}}$, $B_k \in \mathbb{R}^{n_{\mathrm{x}} \times n_{\mathrm{u}}}$, the inequality constraints with $D_k^{\mathrm{x}} \in \mathbb{R}^{n_{\mathrm{c},k} \times n_{\mathrm{x}}}$, $D_k^{\mathrm{u}} \in \mathbb{R}^{n_{\mathrm{c},k} \times n_{\mathrm{u}}}$ and an initial value condition where $\hat{x}_0 \in \mathbb{R}^{n_{\mathrm{x}}}$ denotes the current state estimate.

Most embedded optimization algorithms that have been successfully applied to real-time optimal control rely on direct linear algebra routines. However, it is known that iterative methods can result in a better asymptotic complexity when solving the saddle point linear systems arising in second order optimization methods (Benzi et al., 2005). Iterative solvers, such as the minimal residual (MINRES) or the conjugate gradient (CG) method (Greenbaum, 1997), are suitable for hardware acceleration of the linear system solution, e.g., using an FPGA for fast embedded applications (Boland and Constantinides, 2008), due to their higher ratio of addition and multiplication operations. The use of an iterative method also allows to make a trade-off between computational time and solution accuracy (Knyazev et al., 2015). However, for a general saddle point linear system, Krylov subspace methods tend to converge rather poorly without preconditioning (Benzi et al., 2005).

Block structured preconditioning techniques have recently been proposed within an active-set method for real-time optimal control in (Quirynen et al., 2018). Earlier work in (Shahzad et al., 2010a,b) studied the use of block-diagonal preconditioners, based on techniques that are instead tailored to an inexact interior-point (IP) framework. Unlike IP methods, an active-set solver can considerably benefit from the use of warm or hot-starting techniques to reduce the computational effort when solving a sequence of closely related optimal control problems as discussed in (Bartlett et al., 2000; Ferreau et al., 2017; Wright, 1996). In addition, the cost per iteration is generally of a lower computational complexity by exploiting low-rank updates of the matrix factorizations when changing the current guess for the active set (Ferreau et al., 2014; Kirches et al., 2011). The proposed `PRESAS` solver allows for an initial setup computational complexity of $\mathcal{O}(Nm^3)$ and a per-iteration complexity of $\mathcal{O}(Nm^2)$ for the QP in (1), where m denotes the number of state and control variables in the system (Quirynen et al., 2018).

The present paper proposes a new variant of the `PRESAS` solver that further improves the computational performance, i.e., it reduces the overall number of iterations for the residual method. We particularly focus on the numerical properties of the preconditioning when using low-precision arithmetics, which is very common for practical applications of MPC on embedded control hardware. The solver is based on a constraint preconditioner for the projected conjugate gradient (PPCG) method and it corresponds to a direct linear system solution in an ideal setting, i.e., when solving a strictly convex and well-conditioned QP with high-precision arithmetics. In addition, based on

two numerical case studies, we show that a limited number of CG iterations can remain sufficient in practice, e.g., when using single- instead of double-precision arithmetics on an ARM Cortex-A7 processor.

The paper is organized as follows. Section 2 presents the preliminaries on quadratic programming, active-set methods and on preconditioning of iterative solvers. Section 3 describes two block-diagonal preconditioners to be used within PMINRES. Then, Section 4 presents and discusses the optimal control structured constraint preconditioner for the PPCG method. The performance of a resulting `PRESAS-PPCG` solver is illustrated in Section 5, based on case studies of both linear and nonlinear MPC.

2. PRELIMINARIES

We assume that the convex QP in (1) has a unique global solution that is non-degenerate. A solution is degenerate when either the strict complementarity condition or the linear independence constraint qualification (LICQ) does not hold (Nocedal and Wright, 2006). In order to have a unique solution to the QP in (1), the Hessian needs to be positive definite on the null-space of the strictly active constraints at the solution.

2.1 Embedded Optimal Control Algorithms

There is a general trade-off between solvers that make use of second-order information and require only few but computationally expensive iterations, e.g., `qpOASES` (Ferreau et al., 2014), versus first-order methods that are of low complexity but may require many more iterations, such as `PQP` (Di Cairano et al., 2013), `ADMM` (Raghunathan and Di Cairano, 2015) and other gradient or splitting-based methods (Ferreau et al., 2017). In addition, there is an important distinction between optimal control algorithms that target the dense versus the sparse problem formulation. The numerical elimination of the state variables in a condensing routine (Bock and Plitt, 1984) is typically of a computational complexity $\mathcal{O}(N^2m^3)$ but, unlike nonlinear problems, it can be mostly avoided in linear MPC applications. However, even with an offline preparation of the dense QP formulation, solvers applied to this dense QP will have a runtime complexity of $\mathcal{O}(N^2m^2)$ (Kirches et al., 2011). Instead, we focus on directly solving the OCP formulation with the block sparsity structure in (1), similar to the software tools in `FORCES` (Domahidi and Perez, 2014) and `HPMPC` (Frison et al., 2014).

It is important to note that many tailored QP algorithms for real-time optimal control rely on strict convexity of the cost function. This enables the usage of a dual Newton strategy such as in `qpDUNES` (Frasch et al., 2015), sparsity exploiting linear algebra routines such as the block-tridiagonal Cholesky factorization of the Schur complement in (Anderson et al., 1999; Wang and Boyd, 2010) or a particular Riccati recursion for linear-quadratic control problems in (Frison and Jørgensen, 2013; Wright, 1996). In the case of a positive semidefinite cost matrix, regularization needs to be applied, followed by an iterative refinement procedure to obtain a solution to the original problem. This combination of regularization and iterative refinement can be also needed in the presence of ill-conditioned QP matrices. Instead, we here do not assume strict convexity of the cost function and we propose to use an iterative method to solve each linear KKT system.

2.2 Primal Feasible Active-Set Method

The basic idea behind active-set methods is to find an optimal active set by iteratively updating a current guess. When fixing the active constraints at the current solution guess, a corresponding structured equality constrained QP needs to be solved to compute a new search direction

$$\min_{\Delta X,\Delta U} \sum_{k=0}^{N-1} \frac{1}{2}\Delta w_k^\top H_k\, \Delta w_k + \begin{bmatrix}\tilde{q}_k^\top & \tilde{r}_k^\top\end{bmatrix} \Delta w_k \tag{2a}$$

$$+ \frac{1}{2}\Delta x_N^\top Q_N \Delta x_N + \tilde{q}_N^\top \Delta x_N \tag{2b}$$

$$\text{s.t.} \quad \Delta x_0 = 0, \tag{2c}$$

$$\Delta x_{k+1} = A_k \Delta x_k + B_k \Delta u_k,\ k = 0, \dots, N-1, \tag{2d}$$

$$0 = D^{\mathrm{x}}_{k,i}\Delta x_k + D^{\mathrm{u}}_{k,i}\Delta u_k, \quad (k,i) \in \mathcal{W}, \tag{2e}$$

where $\mathcal{W}$ denotes the current guess for the active set, i.e., the *working set*. The variables $\Delta w_k := (\Delta x_k, \Delta u_k) = (x_k - \bar{x}_k, u_k - \bar{u}_k)$ are defined for $k = 0, \dots, N-1$ and $\Delta w_N := \Delta x_N = x_N - \bar{x}_N$, where $\bar{w}_k := (\bar{x}_k, \bar{u}_k)$ denotes the current guess for the optimal solution of the QP in (1). Note that the equality constrained QP in (2) results in the search direction $\bar{w}_k + \alpha \Delta w_k$ for which all constraints in the set $\mathcal{W}$ remain satisfied, regardless of the value for α. A distinction should be made between primal, dual and primal-dual active-set methods (Nocedal and Wright, 2006). In addition, parametric methods have been proposed (Ferreau et al., 2014) in order to exploit the parametric aspect within a homotopy framework. In this work, we consider a primal feasible active-set method.

2.3 Preconditioning of Iterative Solvers

At each iteration of the active-set method, one needs to efficiently solve the saddle point linear system

$$\begin{bmatrix}\mathcal{H} & \mathcal{A}^\top \\ \mathcal{A} & \mathbb{0}\end{bmatrix}\begin{bmatrix}\Delta y \\ \Delta\lambda\end{bmatrix} = -\begin{bmatrix}h \\ a\end{bmatrix} \quad \text{or} \quad \mathcal{K}\, z = b, \tag{3}$$

which corresponds to the first order necessary conditions of optimality for the equality constrained QP (2). In (3), the matrix $\mathcal{A}$ has full rank and $\mathcal{H}$ is symmetric and positive semidefinite. Unlike the prior work on embedded optimization algorithms for optimal control based on direct linear algebra routines in (Frison and Jørgensen, 2013; Kirches et al., 2011; Wang and Boyd, 2010; Wright, 1996), we propose the use of iterative solvers as discussed for general saddle point linear systems in (Benzi et al., 2005; Benzi and Wathen, 2008). Preconditioning is necessary for the good performance of iterative solvers (Knyazev et al., 2015). It results in a modified linear system $\mathcal{T}^{-1}\mathcal{K}\, z = \mathcal{T}^{-1} b$ where $\mathcal{T}$ is the preconditioner, which is such that

(1) computations with the operator $\mathcal{T}^{-1}$ are cheaper than solving the original saddle point linear system in (3),
(2) and the preconditioned matrix $\mathcal{T}^{-1}\mathcal{K}$ approximates the identity or its eigenvalues are tightly clustered (Greenbaum, 1997).

An overview on algebraic and application-specific preconditioners can be found in (Benzi et al., 2005). Here, we focus on two block-diagonal preconditioning techniques.

2.4 PRESAS for Embedded Predictive Control

A primal active-set method requires the availability of an initial point, which is already primal feasible. In the general case of a constrained quadratic program, this is a nontrivial task that corresponds to a *Phase* 1 procedure as described, for example, in (Fletcher, 1987; Nocedal and Wright, 2006). However, when solving the parametric OCP in (1) within receding horizon based control or estimation, it becomes relatively easy to satisfy the initial value condition and the continuity constraints based on a forward simulation using a shifted version of the previous control trajectory. In addition, one may introduce slack variables in order to always be able to satisfy all state-dependent inequality constraints in (1e).

The resulting `PRESAS` solver is relatively easy to implement, because there is no explicit Phase 1 procedure needed for initialization and a primal active-set method automatically maintains the linear independence of the constraint gradients in its working set (Nocedal and Wright, 2006). The computational performance of this solver has recently been illustrated for multiple numerical case studies of linear MPC on an ARM Cortex-A53 processor, and was shown to be competitive with other state of the art solvers in (Quirynen et al., 2018).

3. OPTIMAL CONTROL STRUCTURED BLOCK-DIAGONAL PRECONDITIONING

The saddle point linear system in (3) describes the first order necessary conditions of optimality for the equality constrained QP in (2). It has a particular sparsity structure because $\mathcal{H}$ corresponds to the block-diagonal Hessian matrix and the constraint matrix $\mathcal{A}$ reads as

$$\mathcal{A} = \begin{bmatrix} -\mathbb{1} & \mathbb{0} & & \\ E_0^{\mathrm{x}} & E_0^{\mathrm{u}} & & \\ A_0 & B_0 & -\mathbb{1} & \mathbb{0} \\ & & & \ddots \end{bmatrix} = \begin{bmatrix} [-\mathbb{1} \ \mathbb{0}] & & \\ E_0 & & \\ C_0 & [-\mathbb{1} \ \mathbb{0}] & \\ & & \ddots \end{bmatrix}, \tag{4}$$

where E_k^{x} and E_k^{u} denote the active inequality constraints for each interval k, corresponding to the working set in $\mathcal{W}$. For notational convenience, we define the block matrices $C_k := [A_k \ B_k]$ and $E_k := [E_k^{\mathrm{x}} \ E_k^{\mathrm{u}}]$.

3.1 Preconditioner 1: Augmented Lagrangian (AL)

To obtain a good performance for the iterative solver, we can use the standard block-diagonal preconditioner

$$\mathcal{T}_{\mathrm{a}} = \begin{bmatrix} \mathcal{H} + \mathcal{A}^\top \Gamma \mathcal{A} & \mathbb{0} \\ \mathbb{0} & \Gamma^{-1} \end{bmatrix} = \begin{bmatrix} \mathcal{H} + \gamma \mathcal{A}^\top \mathcal{A} & \mathbb{0} \\ \mathbb{0} & \gamma^{-1} \mathbb{1} \end{bmatrix}, \tag{5}$$

where Γ is a symmetric positive definite weighting matrix. A popular choice for the weighting matrix, which follows an augmented Lagrangian type technique (Benzi and Wathen, 2008), is $\Gamma = \gamma \mathbb{1}$ where $\gamma > 0$ is a scalar and $\mathbb{1}$ denotes the identity matrix. The application of this augmented Lagrangian type preconditioner $\mathcal{T}_{\mathrm{a}}$ in (5) requires the factorization of the block-tridiagonal matrix $\mathcal{H} + \gamma \mathcal{A}^\top \mathcal{A}$ which reads as

$$\mathcal{H} + \gamma \mathcal{A}^\top \mathcal{A} = \begin{bmatrix} \hat{H}_0 + \gamma \, G_0^\top G_0 & -\gamma \, C_0^\top & \\ -\gamma \, C_0 & \hat{H}_1 + \gamma \, G_1^\top G_1 & -\gamma \, C_1^\top \\ & -\gamma \, C_1 & \ddots \end{bmatrix}, \tag{6}$$

where $G_k = \begin{bmatrix} E_k \\ C_k \end{bmatrix}$, $H_k = \begin{bmatrix} Q_k & S_k^\top \\ S_k & R_k \end{bmatrix}$, $\hat{H}_k = \begin{bmatrix} Q_k + \gamma \mathbb{1} & S_k^\top \\ S_k & R_k \end{bmatrix}$.

The structured matrix in Eq. (6) is positive definite for any value $\gamma > 0$ such that a block-tridiagonal Cholesky decomposition (Anderson et al., 1999) can be applied. The asymptotic computational complexity for this matrix factorization is $\mathcal{O}(Nm^3)$, where $m = n_{\mathrm{x}} + n_{\mathrm{u}}$ is defined. However, such computational cost is incurred only once per QP solution for the initial guess of active constraints. At each iteration of the active-set method, in which one constraint is added or removed from the working set, a corresponding row is added or removed from the Jacobian matrix $\mathcal{A}$ in (4). Given this modification of the matrix $\mathcal{H} + \gamma \mathcal{A}^\top \mathcal{A}$, a rank-one update to its block-tridiagonal Cholesky factorization can be computed at a computational complexity of $\mathcal{O}(Nm^2)$ in each iteration of the `PRESAS` solver (Quirynen et al., 2018).

3.2 Preconditioner 2: Schur Complement (SC)

Alternatively, a Schur complement type preconditioner can be used that reads as follows

$$\mathcal{T}_{\mathrm{s}} = \begin{bmatrix} \tilde{\mathcal{H}} & \mathbb{0} \\ \mathbb{0} & \mathcal{A} \tilde{\mathcal{H}}^{-1} \mathcal{A}^\top \end{bmatrix}, \tag{7}$$

where $\tilde{\mathcal{H}} \approx \mathcal{H}$ such that $\tilde{\mathcal{H}} \succ 0$. This preconditioner $\mathcal{T}_{\mathrm{s}}$ in (7) also results in a block-tridiagonal structure

$$\mathcal{A} \tilde{\mathcal{H}}^{-1} \mathcal{A}^\top = \begin{bmatrix} \tilde{Q}_0^{-1} & -\tilde{Q}_0^{-1} E_0^{\mathrm{x}\top} & -\tilde{Q}_0^{-1} A_0^\top & \\ -E_0^{\mathrm{x}} \tilde{Q}_0^{-1} & E_0 \tilde{H}_0^{-1} E_0^\top & E_0 \tilde{H}_0^{-1} C_0^\top & \\ -A_0 \tilde{Q}_0^{-1} & C_0 \tilde{H}_0^{-1} E_0^\top & C_0 \tilde{H}_0^{-1} C_0^\top + \tilde{Q}_1^{-1} & -\tilde{Q}_1^{-1} E_1^{\mathrm{x}\top} \\ & & -E_1^{\mathrm{x}} \tilde{Q}_1^{-1} & \ddots \end{bmatrix}, \tag{8}$$

where, for simplicity, we assumed that each of the Hessian matrices is block diagonal, i.e., $S_k = \mathbb{0}$ for $k = 0, \ldots, N-1$. Note that each of the block matrices is of different dimensions in (8), corresponding to the number of active constraints in each block. Similar to before and as discussed in (Quirynen et al., 2018), a rank-one update can be computed for the block-tridiagonal Cholesky factorization of the matrix $\mathcal{A} \tilde{\mathcal{H}}^{-1} \mathcal{A}^\top$ as part of the preconditioner in (7), resulting in a computational complexity of $\mathcal{O}(Nm^2)$ at each iteration of the active-set method.

In the case that the Hessian block matrix H_k is positive semidefinite, then a positive definite approximation $\tilde{H}_k = \begin{bmatrix} \tilde{Q}_k & \mathbb{0} \\ \mathbb{0} & \tilde{R}_k \end{bmatrix} \succ 0$ needs to be computed. One can apply an *on-the-fly* regularization in the form of a modified Cholesky factorization, e.g., as in (Frasch et al., 2015; Nocedal and Wright, 2006). Alternatively, we can apply the structure-exploiting regularization technique tailored to optimal control from (Verschueren et al., 2017). We further restrict to a simple but standard regularization procedure of the form $\tilde{H}_k = H_k + \epsilon \mathbb{1}$. The value for $\epsilon > 0$ needs to be chosen sufficiently small such that $\tilde{H}_k \approx H_k$ but it also needs to be large enough such that $\tilde{H}_k \succ 0$. Note that the Hessian matrices do not need to be diagonal and one can even include off-diagonal matrices S_k, without changing the block-tridiagonal sparsity structure in (8).

3.3 Preconditioned MINRES Algorithm

Both the Schur-complement based $\mathcal{T}_\text{s}$ and the augmented Lagrangian preconditioner $\mathcal{T}_\text{a}$ are symmetric positive definite, given a sufficiently large choice for $\Gamma = \gamma\,\mathbb{1}$. Therefore, these preconditioning techniques can be used within the preconditioned minimal residual (PMINRES) algorithm as described in (Greenbaum, 1997). PMINRES is based on the three-term recurrence in the Lanczos iteration for symmetric matrices and specifically requires a positive definite preconditioner (Benzi et al., 2005). On the other hand, the generalized minimal residual (GMRES) algorithm is based on the Arnoldi iteration, for which the computational cost generally grows with each iteration, but it does not need a positive definite preconditioner. For simplicity, we further focus on using PMINRES in combination with either of the block-diagonal preconditioners in Eq. (5) or (7).

For the augmented Lagrangian preconditioner $\mathcal{T}_\text{a}$, as discussed in detail by (Greif and Schoetzau, 2006), the eigenvalues of the preconditioned matrix become more tightly clustered around ± 1 as the value of $\gamma > 0$ increases. For a sufficiently large value of γ, MINRES can therefore converge within two iterations in the ideal setting, but choosing γ too large may result in ill-conditioning of the matrix $\mathcal{T}_\text{a}$. For the Schur-complement based preconditioner $\mathcal{T}_\text{s}$, when $\tilde{\mathcal{H}} = \mathcal{H}$ is invertible, the preconditioned matrix has three distinct eigenvalues 1, $\frac{1}{2}(1+\sqrt{5})$ and $\frac{1}{2}(1-\sqrt{5})$, and therefore MINRES converges within three iterations (Murphy et al., 2000). The eigenvalues become clustered when, e.g., regularization is needed to make $\tilde{\mathcal{H}} \approx \mathcal{H}$ positive definite (Benzi et al., 2005; Murphy et al., 2000). Similar to before, the eigenvalues become more tightly clustered for a smaller value of $\epsilon > 0$ in $\tilde{\mathcal{H}} = \mathcal{H} + \epsilon\,\mathbb{1}$, while trying to avoid numerical issues that can be caused by ill-conditioning of the matrix $\mathcal{T}_\text{s}$, when choosing the value of $\epsilon > 0$ too small. The effect of these choices on the overall performance of the algorithm will be illustrated in the case studies of Section 5.

4. PROJECTED PRECONDITIONED CONJUGATE GRADIENT METHOD WITHIN PRESAS

The parameters, γ or ϵ, in the above mentioned preconditioners can be difficult to choose, especially in case of using lower precision arithmetics or in the presence of ill-conditioned QP matrices. In what follows, we therefore present a preconditioned iterative solver for which the performance is less dependent on a careful choice for the regularization parameter. We propose to use conjugate gradient (CG) iterations in the null space of the active constraints, based on a particular projection operator that preserves the block-structured sparsity in optimal control. For this purpose, let us briefly review the issues that arise when applying the CG method to a standard reduced formulation of the linear KKT system.

4.1 Conjugate Gradient for Reduced Linear System

The equality constrained QP in (2) could be solved by eliminating the constraints and solving the resulting reduced problem formulation. Let us define the matrix $\mathcal{Z}$ as a basis for the null space of the constraint matrix $\mathcal{A}$ such that $\mathcal{A}\mathcal{Z} = 0$. In addition, note that $a = 0$ in the right-hand side of Eq. (3) for a primal feasible active-set method such that the reduced linear system reads as

$$\mathcal{Z}^\top \mathcal{H}\mathcal{Z}\,\Delta y_z = -\mathcal{Z}^\top h, \tag{9}$$

where $\Delta y = \mathcal{Z}\,\Delta y_z$, i.e., the solution vector Δy is in the null space of the active constraint matrix $\mathcal{A}$.

Note that one could directly apply the conjugate gradient (CG) method with standard preconditioning techniques to this reduced linear system in (9) as described in (Coleman and Verma, 2001; Gould et al., 2001). This idea is generally not advised for optimal control problems because the block-structured sparsity is destroyed, similar to the use of a condensing routine (Bock and Plitt, 1984), resulting in expensive CG iterations of complexity $\mathcal{O}(N^2m^2)$ instead of the desired runtime complexity of $\mathcal{O}(Nm^2)$. In addition, forming a null space basis matrix $\mathcal{Z}$, the reduced Hessian $\mathcal{Z}^\top\mathcal{H}\mathcal{Z}$ and constructing and applying a preconditioner for this reduced Hessian would generally be expensive (Gould et al., 2001). Sparsity exploiting update procedures for the null space basis matrix and for the factorization in a Schur complement step of an active-set strategy have been proposed in (Kirches et al., 2011), but they are relatively complex to implement.

4.2 Projected Preconditioned Conjugate Gradient (PPCG)

Instead of the reduced linear system from Eq. (9), we propose to apply a projection operator $\mathcal{P}$, onto the null space of A, to the linear system in Eq. (3):

$$\mathcal{P}\,\mathcal{H}\,\Delta y = -\mathcal{P}\,h, \tag{10}$$

where $\mathcal{P}\mathcal{A}^\top = 0$ has been used (Bakhvalov and Knyazev, 1994). One possible definition of the projection operator reads as $\mathcal{P} := \mathcal{Z}\left(\mathcal{Z}^\top\mathcal{Z}\right)^{-1}\mathcal{Z}^\top$, but it requires again the computation of a basis matrix $\mathcal{Z}$ for the null space of A. The idea of the projected preconditioned conjugate gradient (PPCG) method from (Coleman and Verma, 2001; Dollar, 2005; Gould et al., 2001) is to use the alternative projection operator

$$\mathcal{P}_{\mathcal{A}:\tilde{\mathcal{H}}} := \mathbb{1} - \mathcal{A}^\top\left(\mathcal{A}\tilde{\mathcal{H}}^{-1}\mathcal{A}^\top\right)^{-1}\mathcal{A}\tilde{\mathcal{H}}^{-1}, \tag{11}$$

where $\mathbb{1}$ denotes the identity matrix. The projection in (11) does not rely on any basis matrix $\mathcal{Z}$ and it preserves a block-structured problem sparsity, as discussed in the next subsection. The projection operator in (11) is defined for a particular preconditioner $\tilde{\mathcal{H}} \succ 0$ and $\tilde{\mathcal{H}} \approx \mathcal{H}$ that can be applied directly to Eq. (10), resulting in

$$\tilde{\mathcal{H}}^{-1}\,\mathcal{P}_{\mathcal{A}:\tilde{\mathcal{H}}}\,\mathcal{H}\,\Delta y = -\tilde{\mathcal{H}}^{-1}\,\mathcal{P}_{\mathcal{A}:\tilde{\mathcal{H}}}\,h. \tag{12}$$

One can then apply the conjugate gradient method to the preconditioned projected linear system in (12).

The operation $\left(\tilde{\mathcal{H}}^{-1}\,\mathcal{P}_{\mathcal{A}:\tilde{\mathcal{H}}}\right)b_1$ in (12) can alternatively be computed by solving the linear system

$$\begin{bmatrix}\tilde{\mathcal{H}} & \mathcal{A}^\top \\ \mathcal{A} & \mathbb{0}\end{bmatrix}\begin{bmatrix}z_1 \\ z_2\end{bmatrix} = \begin{bmatrix}b_1 \\ 0\end{bmatrix}, \quad \text{or} \quad \mathcal{T}_\text{c}\,z = b, \tag{13}$$

for a certain vector b_1. The matrix $\mathcal{T}_\text{c}$ is also referred to as a *constraint preconditioner* for the KKT system in (3). The solution of the linear system (13) corresponds to

$$z_1 := \tilde{\mathcal{H}}^{-1}\left(b_1 - \mathcal{A}^\top z_2\right),\; z_2 := \left(\mathcal{A}\tilde{\mathcal{H}}^{-1}\mathcal{A}^\top\right)^{-1}\mathcal{A}\tilde{\mathcal{H}}^{-1}b_1, \tag{14}$$

such that $z_1 = \left(\tilde{\mathcal{H}}^{-1}\,\mathcal{P}_{\mathcal{A}:\tilde{\mathcal{H}}}\right) b_1$ holds. We further use the improved variant of the PPCG method based on the residual update strategy in (Dollar, 2005; Gould et al., 2001), in order to avoid significant round-off errors in the range of $\mathcal{A}^\top$. One could additionally use an iterative refinement step to improve the numerical stability.

4.3 Block-structured Sparse Constraint Preconditioner

The standard block-tridiagonal sparsity structure of the positive definite matrix $\mathcal{A}\tilde{\mathcal{H}}^{-1}\mathcal{A}^\top$ as well as the block-diagonal structure of the positive definite matrix $\tilde{\mathcal{H}}$ can be exploited in Eq. (14) for the implementation of a novel variant of `PRESAS` in combination with the PPCG method for fast MPC applications. In addition, note that the approximate Hessian matrix $\tilde{\mathcal{H}}$ in the preconditioner is often chosen to be diagonal, in practice. As described earlier in Section 3, the block-tridiagonal Cholesky factorization for the matrix $\mathcal{A}\tilde{\mathcal{H}}^{-1}\mathcal{A}^\top$ can be also efficiently maintained based on a rank-one factorization update for each active-set change. Similar to the techniques in (Quirynen et al., 2018), this alternative variant of the `PRESAS` solver requires an initial setup computational complexity of $\mathcal{O}(Nm^3)$ and a per-iteration complexity of $\mathcal{O}(Nm^2)$ for solving the optimal control structured QP in (1).

The quality of the constraint preconditioner is defined by the accuracy of the approximation $\tilde{\mathcal{H}} \approx \mathcal{H}$ such that $\tilde{\mathcal{H}} \succ 0$. However, unlike the case for the Schur complement type block-diagonal preconditioner, the eigenvalues of the preconditioned matrix $\mathcal{T}_{\mathrm{c}}^{-1}\mathcal{K}$ can be shown to be

$$\{1\} \,\cup\, \sigma\left(\left(\mathcal{Z}^\top\tilde{\mathcal{H}}\mathcal{Z}\right)^{-1}\mathcal{Z}^\top\mathcal{H}\mathcal{Z}\right), \tag{15}$$

where the notation $\sigma(\cdot)$ denotes the spectrum of a matrix. As described in (Gould et al., 2001), typical choices are $\tilde{\mathcal{H}} = \mathbb{1}$ or $\tilde{\mathcal{H}} = \mathtt{diag}(\mathcal{H})$. Here, we use again a regularized Hessian approximation of the form $\tilde{\mathcal{H}} = \mathcal{H} + \epsilon\,\mathbb{1}$, where $\epsilon > 0$ is chosen sufficiently small without causing ill-conditioning of the constraint preconditioner $\mathcal{T}_{\mathrm{c}}$.

Two conclusions can be made straightforwardly from the expression for the eigenvalues of the preconditioned matrix in Eq. (15). First, all eigenvalues are equal to 1 in case that $\mathcal{H} = \tilde{\mathcal{H}} \succ 0$, i.e., only one CG iteration is needed. Secondly, all eigenvalues remain equal to 1 also when augmented Lagrangian type regularization is applied to the Hessian, i.e., $\tilde{\mathcal{H}} = \mathcal{H} + \mathcal{A}^\top\Gamma\mathcal{A} \succ 0$ such that the reduced Hessian reads $\mathcal{Z}^\top\tilde{\mathcal{H}}\mathcal{Z} = \mathcal{Z}^\top\mathcal{H}\mathcal{Z}$ as in (Dollar, 2005). The latter is very common for MPC applications, e.g., when performing a slack reformulation of inequality constraints based on an exact L1 penalty. The full Hessian matrix will be positive semidefinite as a result of such slack reformulation, but a regularization of the slack variables in the constraint preconditioner does not change the reduced Hessian and therefore it does not affect the convergence of the PPCG method. Note that the number of iterations can be larger than one in practice for both of these cases because of numerical round-off errors that are caused by ill-conditioning and/or the use of low-precision arithmetics.

5. NUMERICAL CASE STUDIES

We consider two different numerical case studies. The first case study concerns a linear MPC problem formulation for a chain of spring-connected masses as in (Wang and Boyd, 2010). The second case study consists of a non-linear MPC problem formulation for the swing-up of an inverted pendulum, based on the real-time iteration (RTI) scheme in the `ACADO` code generation tool (Quirynen et al., 2014). We illustrate the computational performance of the `PRESAS-PPCG` solver based on a standalone C code implementation that can be easily embedded. For this purpose, we present average computation times of closed-loop MPC simulations for both case studies in order to compare the newly proposed solver against the `PRESAS-AL` and `PRESAS-SC` methods from (Quirynen et al., 2018) and other state of the art optimal control algorithms. Motivated by real-world control applications, the computation times in this section have been obtained using an ARM Cortex-A7 processor in the Raspberry Pi 2. [1]

5.1 Case Study 1: MPC on Chain of Oscillating Masses

This first test problem consists of the chain of oscillating masses, which is often used as a benchmark example for fast MPC solvers (Quirynen et al., 2018; Wang and Boyd, 2010). The linear time-invariant system dynamics and corresponding OCP formulation, of the form in (1), are described in more detail in (Wang and Boyd, 2010). The full state of the system consists of the displacement and velocity of the n_{m} masses, i.e., $x(t) \in \mathbb{R}^{2n_{\mathrm{m}}}$ such that the state dimension can be varied by changing the amount of masses. A number of actuators $n_{\mathrm{u}} < n_{\mathrm{m}}$ apply tensions between certain masses while respecting the actuator limitations as well as constraints on the position and velocity of each of the masses. In order to guarantee the QP to remain feasible at each sampling instant, a slack variable is introduced for the state constraints on each control interval. Based on a sufficiently large penalization of this additional variable in the objective, a feasible solution can be found whenever possible. During the closed-loop MPC simulations, a uniformly distributed but reproducible disturbance acts as a random force on each of the masses as in (Wang and Boyd, 2010).

Figure 1 shows the average computation times per QP solution during the closed-loop simulations of linear MPC for different numbers of masses n_{m} and for a varying control horizon length N. It includes the computation times for `ADMM`, `qpOASES`, `HPMPC` and the 3 variants of `PRESAS`. The considered `ADMM` algorithm (Raghunathan and Di Cairano, 2015) and `qpOASES` (Ferreau et al., 2014) both solve the small but dense QP after numerically eliminating the state variables. The timing results for `ADMM` and `qpOASES` include the computation time for this condensing routine. Note that the black dashed lines in Figure 1 illustrate, respectively, a computational cost that increases with a 1^{st}, 2^{nd} or 3^{rd} order of complexity. The typical runtime complexity of $\mathcal{O}(N^2m^2)$ for the two dense solvers, `ADMM` and `qpOASES`, can be observed. On the other hand, the computational complexity of $\mathcal{O}(Nm^3)$ for the interior-point method in

[1] The Raspberry Pi 2 uses a BCM2836 SoC with a 900 MHz 32-bit quad-core ARM Cortex-A7 processor, with 256 KB shared L2 cache.

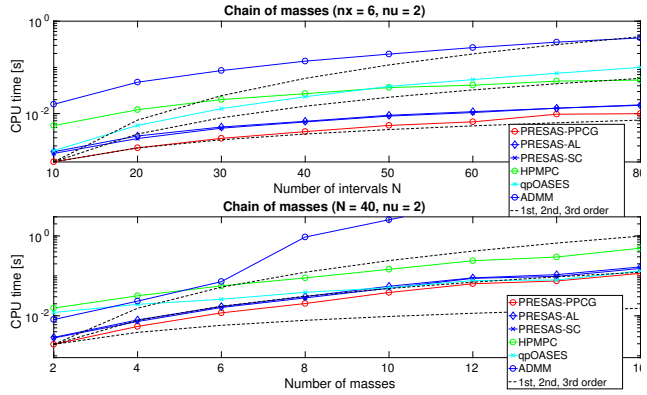

Fig. 1. Average computation times for MPC on the chain of masses with L2 slack reformulation: varying number of intervals N and number of masses n_{m}.

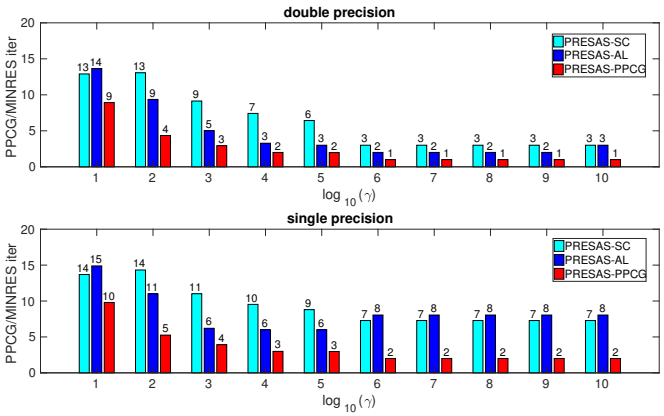

Fig. 2. Average number of iterations for PPCG or MINRES per linear system solution within the closed-loop MPC simulations for a chain of $n_{\mathrm{m}} = 3$ masses.

HPMPC (Frison et al., 2014) and the per iteration complexity of $\mathcal{O}(Nm^2)$ for the proposed variants of the PRESAS solver can be observed in Figure 1. Even though the PRESAS-AL and PRESAS-SC solvers are very competitive with state of the art optimal control algorithms, it can be observed that the proposed PRESAS-PPCG method further outperforms these algorithms. This performance scales well with the number of control intervals and the number of state variables in this particular case study.

Note that PRESAS-AL requires a suitable choice for its parameter γ, while PRESAS-SC and the proposed PRESAS-PPCG solver typically need diagonal regularization using a parameter $\epsilon = \frac{1}{\gamma}$. Figure 2 illustrates the numerical dependency of the number of PMINRES or PPCG iterations on this parameter value, either using single- or double-precision arithmetics. As expected, the value for γ should be sufficiently large but not too large in order to maintain a good conditioning of the preconditioner. In case of double precision arithmetics, the parameter design appears to be much less sensitive. For a sufficiently large value of γ, the SC, AL and PPCG variant of PRESAS, respectively, require 3, 2 and 1 iteration of the residual method on average for each solution of a saddle point linear system. When using single-precision arithmetics, the numerical performance is reduced for all three preconditioners. However, the proposed PRESAS-PPCG method shows a consistently lower number of iterations for the range of values for the parameter γ and allows an average of only two PPCG iterations for a sufficiently small regularization parameter $\epsilon = \frac{1}{\gamma}$ in this case study.

5.2 Case Study 2: Nonlinear MPC of Inverted Pendulum

This second numerical case study involves a swing-up of a pendulum to its upward unstable position, using the nonlinear system dynamics and the optimal control problem formulation from (Quirynen et al., 2014). Both the actuated force and the cart position are constrained to remain within their respective bound values. A slack reformulation of the position constraints is used to guarantee each QP subproblem to remain feasible. The swing-up maneuver results in a relatively high number of online active-set changes with respect to the amount of state $n_{\mathrm{x}} = 4$ and control variables $n_{\mathrm{u}} = 2$. The nonlinear MPC (NMPC) algorithm is implemented using the Gauss-Newton based real-time iteration scheme in the ACADO code generation tool as discussed in (Quirynen et al., 2014).

Table 1. Timing results (ms) for inverted pendulum swing-up ($T_{\mathrm{s}} = 50$ ms and $N = 40$): double/single precision on ARM Cortex-A7.

Double precision	Time [ms] (mean/max)	# sol iter (mean/max)	# res iter (mean/max)
PRESAS-PPCG	1.82/6.90	2.5/12	1.0/1
PRESAS-AL	3.28/14.59	2.5/12	2.0/3
PRESAS-SC	3.67/15.81	2.5/12	3.0/3
qpOASES	16.96/34.86	4.2/14	-
ADMM	15.87/97.57	54.2/446	-
Single precision	Time [ms]	# sol iter	# res iter
PRESAS-PPCG	1.81/7.37	2.5/12	1.9/2
PRESAS-AL	4.75/22.19	2.5/12	6.4/8
PRESAS-SC	6.85/24.67	5.4/21	5.0/6
qpOASES	16.37/38.86	5.2/16	-
ADMM	11.68/69.61	54.5/446	-

Table 1 shows the average and worst-case computation times for each QP solution within the RTI algorithm for nonlinear MPC of the inverted pendulum, including the solvers ADMM, qpOASES and all three variants of PRESAS based on single- versus double-precision arithmetics on an ARM Cortex-A7 processor. Note that ADMM provides a rather low solution accuracy compared to the accuracy of the solutions that are provided by PRESAS and qpOASES, which are similar to each other. When switching from double- to single-precision arithmetics, there can be an increased efficiency of the floating-point operations because, e.g., the compiler can more effectively use SIMD instructions. For example, this is observed for the computation time of the ADMM solver. However, the use of single-precision arithmetics may additionally lead to numerical issues that are caused by round-off errors, resulting in an overall increased number of iterations. One can observe from Table 1 that these competing effects result in a comparable computational performance of the proposed PRESAS-PPCG solver for both single- and double-precision arithmetics, which outperforms the alternative optimal control algorithms on this particular case study.

6. CONCLUSIONS

This paper proposed a new algorithm for solving optimal control structured quadratic programming problems as they typically arise in both linear and nonlinear real-time MPC applications. The solver is based on a block-sparse constraint preconditioner for the projected preconditioned conjugate gradient (PPCG) method within the primal active-set strategy (PRESAS). We discussed how this new solver exhibits favorable numerical properties, especially when using low-precision arithmetics, compared to prior work on the `PRESAS` algorithm based on block-diagonal preconditioning. Its competitiveness with state of the art optimal control algorithms has been illustrated for two case studies of MPC on an ARM Cortex-A7 processor.

REFERENCES

Anderson, E., C., Z.B., Bischof, Blackford, S., Demmel, J., Dongarra, J., Croz, J.D., Greenbaum, A., Hammarling, S., McKenney, A., and Sorensen, D. (1999). *LAPACK Users' Guide*. SIAM, Philadelphia, PA, third edition.

Bakhvalov, N.S. and Knyazev, A.V. (1994). Preconditioned iterative methods in a subspace for linear algebraic equations with large jumps in the coefficients. *Contemporary Mathematics*, 157–162.

Bartlett, R., Wächter, A., and Biegler, L. (2000). Active Set vs. Interior Point Strategies for Model Predictive Control. In *Proc. ACC*, 4229–4233. Chicago, Il.

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

Benzi, M. and Wathen, A.J. (2008). *Some Preconditioning Techniques for Saddle Point Problems*, 195–211. Springer Berlin Heidelberg, Berlin, Heidelberg.

Bock, H.G. and Plitt, K.J. (1984). A multiple shooting algorithm for direct solution of optimal control problems. In *Proc. IFAC World Congr.*, 242–247. Pergamon Press.

Boland, D. and Constantinides, G.A. (2008). An FPGA-based implementation of the MINRES algorithm. In *Inter. Conf. Field Prog. Logic and Appl.*, 379–384.

Coleman, T.F. and Verma, A. (2001). A preconditioned conjugate gradient approach to linear equality constrained minimization. *Comp. Opt. Appl.*, 20(1), 61–72.

Di Cairano, S., Brand, M., and Bortoff, S.A. (2013). Projection-free parallel quadratic programming for linear model predictive control. *Inter. J. of Control*, 86(8), 1367–1385.

Dollar, H. (2005). *Iterative Linear Algebra for Constrained Optimization*. Ph.D. thesis, University of Oxford.

Domahidi, A. and Perez, J. (2014). FORCES professional. embotech GmbH (`http://embotech.com`).

Ferreau, H.J., Almer, S., Verschueren, R., Diehl, M., Frick, D., Domahidi, A., Jerez, J.L., Stathopoulos, G., and Jones, C. (2017). Embedded optimization methods for industrial automatic control. In *IFAC World Congr.*

Ferreau, H.J., Kirches, C., Potschka, A., Bock, H.G., and Diehl, M. (2014). qpOASES: a parametric active-set algorithm for quadratic programming. *Math. Progr. Comp.*, 6(4), 327–363.

Fletcher, R. (1987). *Practical Methods of Optimization*. Wiley, Chichester, 2nd edition.

Frasch, J.V., Sager, S., and Diehl, M. (2015). A parallel quadratic programming method for dynamic optimization problems. *Math. Progr. Comp.*, 7(3), 289–329.

Frison, G. and Jørgensen, J.B. (2013). Efficient implementation of the Riccati recursion for solving linear-quadratic control problems. In *2013 IEEE Inter. Conf. Control Appl. (CCA)*, 1117–1122.

Frison, G., Sorensen, H.B., Dammann, B., and Jørgensen, J.B. (2014). High-performance small-scale solvers for linear model predictive control. In *Proc. European Control Conf. (ECC)*, 128–133.

Gould, N.I.M., Hribar, M.E., and Nocedal, J. (2001). On the solution of equality constrained quadratic programming problems arising in optimization. *SIAM J. Sci. Comput.*, 23(4), 1376–1395.

Greenbaum, A. (1997). *Iterative Methods for Solving Linear Systems*. SIAM.

Greif, C. and Schoetzau, D. (2006). Preconditioners for saddle point linear systems with highly singular (1,1) blocks. *Electr. Trans. on Num. Anal.*, 22, 114–121.

Kirches, C., Bock, H.G., Schlöder, J.P., and Sager, S. (2011). A factorization with update procedures for a KKT matrix arising in direct optimal control. *Math. Progr. Comp.*, 3(4), 319–348.

Knyazev, A., Fujii, Y., and Malyshev, A. (2015). Preconditioned continuation model predictive control. In *Proc. IEEE Conf. Control Appl. (CCA)*.

Murphy, M.F., Golub, G.H., and Wathen, A.J. (2000). A note on preconditioning for indefinite linear systems. *SIAM J. Scientific Computing*, 21(6), 1969–1972.

Nocedal, J. and Wright, S.J. (2006). *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, 2 edition.

Quirynen, R., Knyazev, A., and Di Cairano, S. (2018). Block structured preconditioning within an active-set method for real-time optimal control. In *Proc. European Control Conf. (ECC)*.

Quirynen, R., Vukov, M., Zanon, M., and Diehl, M. (2014). Autogenerating microsecond solvers for nonlinear MPC: a tutorial using ACADO integrators. *Optimal Control Applications and Methods*, 36, 685–704.

Raghunathan, A.U. and Di Cairano, S. (2015). ADMM for convex quadratic programs: Q-linear convergence and infeasibility detection. *arXiv:1411.7288*.

Rawlings, J.B., Mayne, D.Q., and Diehl, M.M. (2017). *Model Predictive Control: Theory, Computation, and Design*. Nob Hill, 2nd edition edition.

Shahzad, A., Kerrigan, E.C., and Constantinides, G.A. (2010a). A fast well-conditioned interior point method for predictive control. In *49th IEEE Conference on Decision and Control (CDC)*, 508–513.

Shahzad, A., Kerrigan, E., and Constantinides, G. (2010b). Preconditioners for inexact interior point methods for predictive control. In *American Control Conference (ACC), 2010*, 5714–5719.

Verschueren, R., Zanon, M., Quirynen, R., and Diehl, M. (2017). A sparsity preserving convexification procedure for indefinite quadratic programs arising in direct optimal control. *SIAM J. of Optim.*, 27(3), 2085–2109.

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. (1996). Applying new optimization algorithms to model predictive control. In J. Kantor, C. Garcia, and B. Carnahan (eds.), *Fifth Intern. Conf. on Chemical Process Control – CPC V*, 147–155.