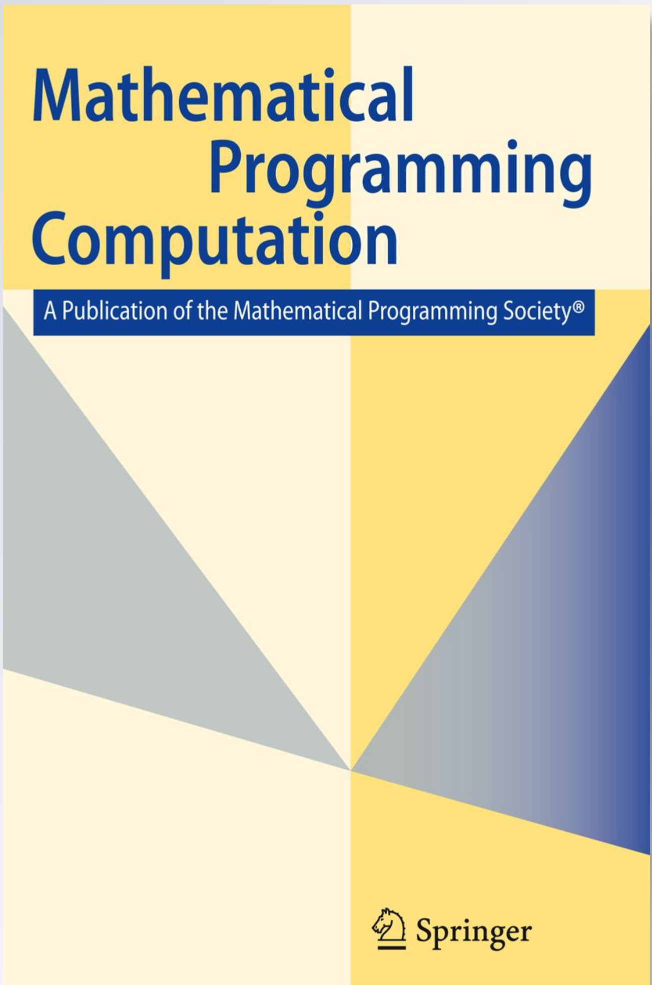
A factorization with update procedures for a KKT matrix arising in direct optimal control

Christian Kirches, Hans Georg Bock, Johannes P. Schlöder & Sebastian Sager

Mathematical Programming Computation
A Publication of the Mathematical Optimization Society

ISSN 1867-2949
Volume 3
Number 4

Math. Prog. Comp. (2011) 3:319-348
DOI 10.1007/s12532-011-0030-z



Springer

Math. Prog. Comp. (2011) 3:319–348
DOI 10.1007/s12532-011-0030-z

FULL LENGTH PAPER

A factorization with update procedures for a KKT matrix arising in direct optimal control

Christian Kirches · Hans Georg Bock · Johannes P. Schlöder · Sebastian Sager

Received: 12 November 2009 / Accepted: 15 July 2011 / Published online: 20 August 2011

Abstract Quadratic programs obtained for optimal control problems of dynamic or discrete-time processes usually involve highly block structured Hessian and constraints matrices, to be exploited by efficient numerical methods. In interior point methods, this is elegantly achieved by the widespread availability of advanced sparse symmetric indefinite factorization codes. For active set methods, however, conventional dense matrix techniques suffer from the need to update base matrices in every active set iteration, thereby loosing the sparsity structure after a few updates. This contribution presents a new factorization of a KKT matrix arising in active set methods for optimal control. It fully respects the block structure without any fill-in. For this factorization, matrix updates are derived for all cases of active set changes. This allows for the design of a highly efficient block structured active set method for optimal control and model predictive control problems with long horizons or many control parameters.

Keywords Matrix factorizations and updates · Block structured active set quadratic programming · Direct methods for optimal control

Mathematics Subject Classification (2010) 90C20 · 90C30 · 93B40 · 15A23 · 65F05

1 Introduction

Quadratic programs obtained for optimal control problems of dynamic or discrete-time processes usually involve highly block structured Hessian and constraints matrices, cf. [7] and [35]. This is true even for unstructured models of the process under

C. Kirches (✉) · H. G. Bock · J. P. Schlöder · S. Sager
Interdisciplinary Center for Scientific Computing (IWR), Heidelberg University,
Im Neuenheimer Feld 368, 69120 Heidelberg, Germany
e-mail: christian.kirches@iwr.uni-heidelberg.de

consideration. For the solution of the arising problems, both active set and interior point methods are popular and both have strong points and weaknesses.

Active set methods are known to allow for very fast solution of a sequence of closely related quadratic problems (QPs) due to their excellent hot starting abilities. This is of significant advantage for the design of fast algorithms, e.g. in sequential quadratic programming (SQP) methods or in model predictive control (MPC). Popular QP techniques often focus on dense problem data without any structure. This contribution is concerned with the task of exploiting the optimal control problem's block structure in active set methods.

For interior point (IP) methods on the other hand, due to the widespread availability of highly efficient symmetric indefinite factorizations such as the codes MA27 and MA57 by Duff [13], the exploitation of both problem and model structure is easily achieved. IP methods however still lack efficient hot starting techniques, which may limit their efficiency especially in MPC applications. For a detailed discussion of the respective merits of active set methods and IP methods we refer to [3]. Popular structure exploiting codes based on IP methods are IPOPT by Wächter and Biegler [42] and the QP codes OOQP by Gertz and Wright [22] and LOQO by Vanderbei [41].

The first key task to be addressed in an active set method for the solution of optimal control QPs is to devise an efficient way of exploiting the structural information available. Several different approaches have been presented in the past.

The classical *condensing* algorithm by Bock and Plitt [7], cf. also Leineweber et al. [35] for a more recent presentation, exploits the optimal control problem's structure in a preprocessing step that yields a considerably smaller but densely populated QP. For a number of optimal control and MPC applications that have been treated recently we refer e.g. to [16,33,43]. The limitations of condensing approaches become noticeable for optimal control problems with long horizons, fine discretizations of the horizon, and for problems with more control parameters than state variables.

Here, an active set method is desirable that solves the structured QP directly. As active set methods tend to perform many iterations, it is crucial that a fast factorization tailored to exploit the particular structure of the QP's KKT system is available. Such a factorization usually comes at the cost of $O(N^3)$ operations, though, where N is the total number of unknowns in the QP. Update procedures that recover the employed factorization in $O(N^2)$ operations after an active set change are employed to make the large number of iterations computationally viable.

For dense QP techniques such as the null space or the range space method, such update procedures are well established. Popular codes include ZQPCVX by Powell [38], QPSOL and its successor QPOPT by Gill et al. [26], QPKWIK by Schmid and Biegler [39], qpOASES for parametric quadratic programming by Ferreau et al. [16]. The code BQPD by Fletcher [18] is a generic active set code that also exploits sparsity to a certain extent. Fletcher [19,20] present efficient factorization and storage schemes for dense factors of general sparse matrices. In [3] a dual active set code based on Schur complements and updates is presented that is designed for systems with optimal control structure. Huynh [32] proposes a sparse active set QP solver based on a block LU factorization of the KKT system that preserves sparsity to a certain degree. Updates based on LUMOD are reported, cf. also Eldersveld and Saunders [15] but the method fills in and requires refactorization after some iterations. Sparse symmetric indefinite factorizations, e.g. Duff and Reid [14] and Duff [13], could also be used to solve the QP's KKT system in sparse active set methods. They do not commonly yield satisfactory performance of the QP code, as fast update procedures are not available in general.

1.1 Relation to own work

Classical condensing techniques for the block structured QP require $O(m^2n^3)$ operations to preprocess the block structured QP into a smaller but dense one. On the dense active set QP solver's part, $O(m^3n^3)$ operations are required for the initial factorization of the QP's KKT matrix, and $O(m^2n^2)$ operations for each active set update of that factorization. In [33] it was observed that, depending on the problem's characteristics, the overhead of the condensing preprocessing step may effectively limit the optimal control problem solver's performance. This motivates our investigation of structured active set techniques. In [34] we examined a block structured factorization due to [40] and its applicability to a KKT matrix arising in an active set QP method for optimal control and model-predictive control. We employed this method to solve a partially convexified and relaxed mixed-integer optimal control problem. The results reported in [34] indicate a significant reduction of the computational effort compared to the classical condensing method. The described linear algebra techniques however lack exploitation of simple bounds and matrix update procedures are not provided.

1.2 New contributions

In this contribution we review from a different point of view the aforementioned block structured factorization. Improving over [34] we show how to efficiently exploit simple bounds on the QP's unknowns in this factorization. We propose for the first time update procedures to this factorization for all kinds of active set changes that may occur when solving a QP with the presented block structure. These updates improve the run time complexity to $O(mn^2)$ as compared to $O(mn^3)$ in [34]. They are based on well established results by Gill and Golub [23] for Cholesky and QR factorizations, and extend the techniques commonly used in dense null-space active set QP solvers, cf. [36]. We extend the run time comparisons of [34] to include a number of generic sparse or structure linear algebra codes, and study the performance gains obtained by the new matrix updates. Computational evidence obtained is promising such that the presented factorization and update procedures are intended to lay the foundation for a fast block structured primal and primal-dual active set QP code to be used in optimal control and model-predictive control. Such a code promises a significant improvement in performance over currently applied techniques.

This contribution is accompanied by MATLAB© source code that contains a proof-of-concept implementation of the presented factorization as well the proposed update procedures for all four investigated cases of active set changes. A data set of sample KKT matrices is provided to enable—up to limited machine precision effects—an exemplary verification of correctness of the computations made by the proposed update procedures.

1.3 Structure of the paper

This contribution is organized as follows. In Sect. 2 we introduce a class of discrete-time optimal control problems, formulate a local quadratic model of the problems, and show connections to SQP methods, to MPC, and to the direct multiple shooting method for continuous time problems. For the solution of the local QP, we employ active set methods and establish the need for an efficient factorization and update of the local QP's KKT system. Section 3 reviews a factorization that fully exploits the block structure of the discrete time optimal control problem class. The main contribution is made in Sect. 4, which comprises the largest part of this paper. Here, we derive for the first time update procedures for the reviewed factorization for all cases of active set changes. In Sect. 5 we consider an exemplary optimal control problem and give per-iteration run times for an active set method using the proposed factorization with matrix updates. The achieved run times are compared to several different approaches such as classical condensing and several more generic sparse or structure-exploiting solvers. Finally, Sect. 6 gives a summary over the achievements presented in this contribution, and concludes with an outlook on further research efforts.

2 A discrete-time linear-quadratic optimal control problem

In this section we introduce a discrete-time linear-quadratic optimal control problem with separable structure. An interpretation as a local quadratic subproblem in the context of SQP methods and of MPC is given. A connection to direct multiple shooting methods for continuous-time problems is mentioned.

2.1 Discrete-time optimal control problem

We start by defining the following class of discrete-time optimal control problems

Definition 21 (*Discrete-time optimal control problem*)

$$
\begin{aligned}
\min_{x,q} \quad & \sum_{i=1}^{m} \Psi_i(x_i, q_i) && \\
\text{s.t.} \quad & x_i^{\text{low}} \le x_i \le x_i^{\text{up}} && \forall i \in \{1, \ldots, m\} \\
& q_i^{\text{low}} \le q_i \le q_i^{\text{up}} && \forall i \in \{1, \ldots, m\} \\
& 0 \le R_i(x_i, q_i) && \forall i \in \{1, \ldots, m\} \\
& 0 = G_i(x_i, q_i) + P_{i+1}(x_{i+1}, q_{i+1}) && \forall i \in \{1, \ldots, m-1\}
\end{aligned}
\tag{1}
$$

in which we minimize a real-valued objective function $\Psi(x, q) = \sum_{i=1}^{m} \Psi_i(x_i, q_i)$ depending on states $x_i \in \mathbb{R}^{n^{\mathrm{x}}}$ of a discrete-time process governed by control parameters $q_i \in \mathbb{R}^{n^{\mathrm{q}}}$. The discrete-time process evolves over m points in time, indexed by $i \in \{1, \ldots, m\}$, and is described by the solution of a state propagation law given in terms of the functions $G_i : \mathbb{R}^{n^{\mathrm{x}}} \times \mathbb{R}^{n^{\mathrm{q}}} \to \mathbb{R}^{n^{\mathrm{x}}}$ and $P_i : \mathbb{R}^{n^{\mathrm{x}}} \times \mathbb{R}^{n^{\mathrm{q}}} \to \mathbb{R}^{n^{\mathrm{x}}}$. Both process states x and control parameters q are subject to simple bounds and to possibly nonlinear constraints $R_i : \mathbb{R}^{n^\mathrm{x}} \times \mathbb{R}^{n^\mathrm{q}} \to \mathbb{R}^{n_i^\mathrm{r}}$, containing for example initial values, boundary conditions, or discretized general path constraints.

The problem class under investigation is quite broad, its only restrictive property being *separability* of all involved functions, i.e. objective $\Psi(\cdot)$, constraints $R(\cdot)$, and state propagation law $G(\cdot)$, $P(\cdot)$, with respect to the m points in time. This contribution is concerned with the efficient exploitation of the structures this separability property imposes on the various system matrices. Separable reformulations are available for some types of coupled constraints. One example are coupled boundary conditions such as periodicity constraints for which the residual in the first node $i = 1$ can be introduced as an additional component of the state vector x, which then enters a point constraint on the final node $i = m$.

2.2 Linearly constrained quadratic model

Assuming sufficient differentiability of the problem functions, we may obtain from problem (1) a corresponding linearly constrained convex QP, by using a linear-quadratic model of the Lagrangian of (1) and a linearization of the constraints.

Definition 22 (*Quadratic optimal control problem*)

$$
\begin{aligned}
\min_{w} \quad & \sum_{i=1}^{m} \left(\tfrac{1}{2} w_i^T H_i w_i + g_i^T w_i \right) & & \\
\text{s.t.} \quad & l_i \le w_i \le u_i & & \forall i \in \{1, \dots, m\} \\
& r_i \le R_i w_i & & \forall i \in \{1, \dots, m\} \\
& h_i = G_i w_i + P_{i+1} w_{i+1} & & \forall i \in \{1, \dots, m-1\}
\end{aligned}
\tag{2}
$$

The linear-quadratic model is represented by the Hessians $H_i \in \mathbb{R}^{n \times n}$ and gradients $g_i \in \mathbb{R}^n$. We summarize the states $x \in \mathbb{R}^{m \cdot n^\mathrm{x}}$ and control parameters $q \in \mathbb{R}^{m \cdot n^\mathrm{q}}$ in the vector $w \in \mathbb{R}^{m \cdot n}$, $w(x, q)$ and $w_i = (x_i, q_i)$, and reuse the symbols R, G, and P of the functions in (1) for their linearization matrices $R_i \in \mathbb{R}^{n_i^\mathrm{r} \times n}$, $G_i \in \mathbb{R}^{n^\mathrm{x} \times n}$, $P_i \in \mathbb{R}^{n^\mathrm{x} \times n}$ in (2). We further assume well-posedness of this problem: the Hessians H_i shall be positive semidefinite, and the matrices $P_i \in \mathbb{R}^{n^\mathrm{x} \times n}$ shall be invertible with respect to the state parts x_i of the w_i. This property guarantees that the state propagation law's linearizations create a unique series of process states $(x_2, \dots, x_m)$ given an initial state x_1 and control parameters $(q_1, \dots, q_m)$.

2.2.1 Sequential quadratic programming

Problem (2) may be viewed as subproblem of a SQP method for the solution of a class of nonlinear problems (NLPs) with the introduced structure. This effectively amounts to solving QP (2) for $k = 1, \dots, N$ with a quadratic models H_i and g_i of the Lagrangian as well as with new linearizations R_i, G_i and P_i of the constraints obtained in the point $w^{(k)}$. The optimal solution obtained from solving (2) then is the step $\Delta w^{(k)}$ improving the iterate $w^{(k)}$ towards a KKT point of the NLP,

$$w^{(k+1)} = w^{(k)} + \alpha^{(k)} \Delta w^{(k)},$$

$\alpha^{(k)} \in (0, 1]$ being a suitably selected step length. We refer to the classical papers of [30] and [37] for SQP methods, as well as to [36]. In [35] an SQP method tailored to NLPs from direct multiple shooting is discussed.

2.2.2 Model-predictive control

In a closed-loop MPC setup, QP (2) is solved repeatedly with initial process states x_1^{meas} obtained from measurements of the real-world process, e.g. estimated by Kalman filters or Moving Horizon Estimator (MHE) techniques, cf. [31] and [12]. The measured or estimated process state x_1^{meas} is best incorporated by Initial Value Embedding techniques due to [11], by introducing the additional constraint

$$x_1 - x_1^{\text{meas}} = 0. \tag{3}$$

A discussion of the favorable properties of this linear embedding especially in conjunction with active set methods, where it can provide a tangential predictor of the optimal solution that is valid even across changes of the active set, is given in [11]. In [8] a multi-level scheme for adaptive updating of selected parts of the QP's matrices and vectors is proposed in the nonlinear MPC context. There, efficient factorization and update procedures for the solution of the QP (2) have decisive influence on the achievable feedback delay and feedback rate.

2.2.3 Direct multiple shooting

The process under control may also be a continuous-time dynamic process described by an ODE. Using the direct multiple shooting method by Bock and Plitt [7], this case is easily incorporated. With ODE right hand side $f_i(\cdot)$ on each time interval $[t_i, t_i + 1] \subset \mathbb{R}$ for $i = 1, \ldots, m-1$, we have $m-1$ initial value problems (IVPs)

$$\dot{x}(t) = f_i(t, x(t), q_i), \qquad x(t_i) = x_i \qquad \forall t \in [t_i, t_{i+1}] \subset \mathbb{R}. \tag{4}$$

If we assume continuity of the IVP solutions for the process states $x(t)$ on the whole of $[t_1, t_m] \subset \mathbb{R}$, we have the *matching conditions*

$$x(t_{i+1};\ t_i, x_i, q_i) - x_{i+1} = 0, \qquad i = 1, \ldots, m-1 \tag{5}$$

as a state propagation law for the process, whose evaluation requires solving the ODE to find the value $x(t_{i+1})$ depending on t_i, x_i, and q_i. We denote its linearization with respect to x_i and q_i by $G_i = \begin{bmatrix} G_i^{\text{x}} & G_i^{\text{q}} \end{bmatrix}$ and with respect to x_{i+1} and q_{i+1} it is $P_i = \begin{bmatrix} -I & 0 \end{bmatrix}$. The sensitivities G_i^{x} and G_i^{q} of the IVP solution with respect to x_i and q_i may for example be obtained by solving the variational differential equations

$$\dot{G}_i^{\text{x}}(t) = \frac{\partial f_i}{\partial x_i}(t, x_i(t), q_i) \cdot G_i^{\text{x}}(t) \qquad G_i^{\text{x}}(t_i) = I \quad \forall t \in [t_i, t_{i+1}], \tag{6a}$$

$$\dot{G}_i^{\mathrm{q}}(t) = \frac{\partial f_i}{\partial x_i}(t, x_i(t), q_i) \cdot G_i^{\mathrm{q}}(t) + \frac{\partial f_i}{\partial q_i}(t, x_i(t), q_i) \qquad G_i^{\mathrm{q}}(t_i) = 0 \quad \forall t \in [t_i, t_{i+1}] \tag{6b}$$

along with the IVP (4) according to the principle of internal numerical differentiation (IND), cf. [1] for ODE and index 1 DAE dynamics.

2.3 Primal active set method

We focus here on a primal active set method for the solution of the convex QP (2). The presented techniques can also be used without change in the primal-dual active set method for parametric convex QPs of [6], implemented e.g. in the code qpOASES by Ferreau et al. [16], which is of advantage especially in MPC applications. We only mention the key issues relevant for our contribution here and refer to [4,44] as well as to the textbooks of [17] and [36] for more extensive discussions.

2.3.1 Active set

For a primal feasible point w we define the sets $\mathbb{X}_i$ of simple bounds and $\mathbb{A}_i$ of linear inequality constraints that are *active* in the solution, i.e. satisfied to equality,

$$\mathbb{X}_i := \{j \mid l_{i,j} = w_{i,j} \ \vee \ u_{i,j} = w_{i,j}\}, \quad \mathbb{A}_i := \{j \mid R_{i,j*} w_i = r_{i,j}\} \tag{7}$$

where $R_{i,j*}$ refers to the jth row of the matrix R_i. In addition we define the set of inactive simple bounds $\mathbb{F}_i$

$$\mathbb{F}_i := \{j \mid l_{i,j} < w_{i,j} < u_{i,j}\}. \tag{8}$$

In the following we will use the symbols $\mathbb{A}$, $\mathbb{F}$, and $\mathbb{X}$ also as superscript index sets in order to refer to submatrices. We write e.g. $R^{\mathbb{AF}}$ to refer to the subset of rows of a matrix $R \in \mathbb{R}^{n_{\mathrm{r}} \times n}$ belonging to the active constraints and the subset of columns of R associated with free unknowns.

2.3.2 One iteration of a primal active set method

Primal active set methods start with a primal feasible guess $w^{(0)}$ of the optimal solution and the associated active set, for which the null space Hessian must be positive definite. They generate a series of primal iterates $w^{(k)}$ proceeding towards the optimal solution of the QP. These iterates are obtained from the repeated solution of an *equality constrained quadratic program* (*EQP*) constructed by restricting the QP to be solved to the active set, and by performing active set changes in order to identify the *optimal active set*. The EQP associated with an active set is found from problem (2) by removing all inactive inequalities, and replacing all active inequalities with equalities:

Definition 23 (*Equality constrained quadratic program*)

$$
\begin{aligned}
\min_{w} \quad & \sum_{i=1}^{m} \left(\tfrac{1}{2} w_i^T H_i w_i + g_i^T w \right) & & \\
\text{s.t.} \quad & b_i^{\mathbb{X}} = w_i^{\mathbb{X}} & \forall i \in \{1, \ldots, m\} & \\
& r_i^{\mathbb{A}} = R_i^{\mathbb{A}} w_i & \forall i \in \{1, \ldots, m\} & \\
& h_i = G_i w_i + P_{i+1} w_{i+1} & \forall i \in \{1, \ldots, m-1\} &
\end{aligned} \tag{9}
$$

The vectors $b_i^{\mathbb{X}}$ hold elements from both bounds vectors l_i and u_i as selected by the active set $\mathbb{X}_i$.

In iteration (k), solving the EQP (9) requires solving the associated linear system of KKT conditions for the optimal solution $(w^*, \lambda^*, \mu^*, \nu^*)$, cf. [36]. Here we denote by $\lambda_i \in \mathbb{R}^{n^{\mathrm{x}}}$ the duals of the matching conditions, by $\mu_i \in \mathbb{R}^{n_i^{\mathrm{r}}}$ the point constraints' duals, and by $\nu_i \in \mathbb{R}^{n^{\mathrm{x}}+n^{\mathrm{q}}}$ the duals of the simple bounds. In Sect. 3 we present a factorization that exploits the structure of problem (9).

On the path from the kth iterate $(w^{(k)}, \lambda^{(k)}, \mu^{(k)}, \nu^{(k)})$ to the EQP's optimal solution $(w^*, \lambda^*, \mu^*, \nu^*)$ three cases are now possible:

1. *an active constraint becomes infeasible* ("primally blocking"). A step onto the closest primally blocking is made, and the constraint enters the active set.
2. all active constraints remain feasible but *the multiplier of an inactive constraint becomes non-optimal* ("dually blocking"). A step onto the closest dually blocking constraint is made, and the constraint leaves the active set.
3. all active constraints remain feasible and the multipliers of all inactive constraints remain dually feasible. Then *the optimal active set has been identified* and the point $(w^*, \lambda^*, \mu^*, \nu^*)$ is the optimal solution not only of the EQP but also of the QP itself.

In cases 1 and 2 the algorithm continues in iteration $k + 1$ with the solution of a new EQP for the new active set.

Many details of this coarse description of the active set algorithm remain to be discussed and are still a topic of active research. This includes finding the step direction and length, maintaining linear independence, finding a feasible initial guess, finite termination and prevention of stalling and cycling. Details can be found in e.g. [6,17, 18,25,29,36].

3 A block structured factorization

In this section we address the factorization of the KKT system obtained from EQP (9). We give a new derivation of this factorization that has first been published in [34] and is based on prior work by Steinbach [40] and extend it to efficiently exploit simple bounds on the unknowns. This yields an algorithm for the efficient solution of the EQP (9) that fully respects the block structure of the KKT matrix and completes in $O(mn^3)$ operations.

In [34] a per-node derivation of this factorization can be found along with the associated backsolve procedure. It is based on the ordering

$$(x_1, q_1, \lambda_1, \mu_1, \nu_1,\ x_2, q_2, \lambda_2, \mu_2, \nu_2, \ldots, x_m, q_m, \mu_m, \nu_m)$$

of the primal and dual unknowns which is convenient in a direct multiple shooting context as it joins all unknowns of a node, and gives a banded block KKT matrix of the minimum possible bandwidth $\max_{1\le i\le m}(n + n_i^{\mathbb{X}} + n_i^{\mathrm{r}} + n^{\mathrm{x}})$.

3.1 Saddle-point problem form

It is well known that solving the KKT system of EQP (9) can also be regarded as a saddle-point problem with mn nonnegative eigenvalues associated with the Hessian blocks H_i (positive only if the H_i are positive definite), and with $\sum_{i=1}^{m}(n_i^{\mathbb{X}} + n_i^{\mathrm{r}}) + (m-1)n^{\mathrm{x}}$ negative eigenvalues associated with the active simple bounds, active point constraints, and matching conditions, see e.g. [36] for the dense case. For a survey on saddle-point problems and numerical methods for their solution we refer to [5]. In the following, we derive the block structured factorization of [34] based on this notion using the ordering

$$(x_1, q_1, \ldots, x_m, q_m,\ \lambda_1, \ldots, \lambda_{m-1},\ \mu_1, \nu_1, \ldots, \mu_m, \nu_m)$$

of the unknowns. We consider the following general form of a saddle-point problem,

$$\begin{bmatrix} H & M^T & C^T \\ M & 0 & 0 \\ C & 0 & 0 \end{bmatrix} \begin{bmatrix} w \\ \lambda \\ \zeta \end{bmatrix} = \begin{bmatrix} g \\ h \\ c \end{bmatrix}, \tag{10}$$

with $\zeta := (\mu_1, \nu_1, \ldots, \mu_m, \nu_m)$ and $c := (b_1^{\mathbb{X}}, r_1^{\mathbb{A}}, \ldots, b_m^{\mathbb{X}}, r_m^{\mathbb{A}})$ and the following block structured submatrices H, M, and C of the KKT system of EQP (9),

$$H_{i,i} := \begin{bmatrix} H_i^{\mathbb{FF}} & H_i^{\mathbb{FX}} \\ H_i^{\mathbb{XF}} & H_i^{\mathbb{XX}} \end{bmatrix}, \qquad 1 \le i \le m, \tag{11a}$$

$$M_{i,i} := \begin{bmatrix} G_i^{\mathbb{F}} & G_i^{\mathbb{X}} \end{bmatrix}, \quad M_{i,i+1} := \begin{bmatrix} P_{i+1}^{\mathbb{F}} & P_{i+1}^{\mathbb{X}} \end{bmatrix}, \quad 1 \le i \le m-1, \tag{11b}$$

$$C_{i,i} := \begin{bmatrix} 0 & I_i^{\mathbb{X}} \\ R_i^{\mathbb{AF}} & R_i^{\mathbb{AX}} \end{bmatrix}, \qquad 1 \le i \le m. \tag{11c}$$

Subscript indices denote the populated block row and column. All remaining block entries of H, M, and C are understood to be zero. The blocks H and C of system (10) hence take block diagonal shape and M takes block bidiagonal shape, its first lower subdiagonal being occupied by the blocks $P_i^{\mathbb{F}}$, $P_i^{\mathbb{X}}$. The block H is positive semidefinite by definition of problem (2), and M, C have full row rank by choice of a linear independent active set.

3.2 TQ factorization step

A first and well-known step to factorizing system (10) is to apply a TQ factorization to C, see e.g. [5] or [36],

$$C\left[\tilde{Z}\ \tilde{Y}\right]=\left[0\ \tilde{T}\right], \qquad \tilde{Q}:=\left[\tilde{Z}\ \tilde{Y}\right] \tag{12}$$

wherein $\tilde{Z}$, $\tilde{Y}$ are column orthogonal bases of the null-space and the range-space of C respectively, and $\tilde{T}$ is a factor of C satisfying $C\tilde{Y}=\tilde{T}$. This factorization can e.g. be obtained from a QR factorization by reversal of the column order. As C is block diagonal, so are the factors $\tilde{Z}$ and $\tilde{Y}$,

$$\begin{bmatrix} 0 & I_i^{\mathbb{X}} \\ R_i^{\mathbb{AF}} & R_i^{\mathbb{AX}} \end{bmatrix}\begin{bmatrix} Z_i & Y_i & 0 \\ 0 & 0 & I \end{bmatrix}=\begin{bmatrix} 0 & 0 & I \\ 0 & T_i & R_i^{\mathbb{AX}} \end{bmatrix}, \quad \tilde{Z}_i:=\begin{bmatrix} Z_i \\ 0 \end{bmatrix}, \quad \tilde{Y}_i:=\begin{bmatrix} Y_i & 0 \\ 0 & I \end{bmatrix},$$
$$1\le i\le m, \tag{13}$$

The factor T_i is southeast triangular; its shape turns out to be of advantage in Sect. 4. Premultiplying (10) by $\operatorname{diag}(Q, I, I)^T$ and writing $Zw^Z+Yw^Y=w$ we obtain

$$\begin{bmatrix} Y^THY & Y^THZ & Y^TM^T & T^T \\ Z^THY & Z^THZ & Z^TM^T & 0 \\ MY & MZ & 0 & 0 \\ T & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} w^Y \\ w^Z \\ \lambda \\ \zeta \end{bmatrix}=\begin{bmatrix} Y^Tg \\ Z^Tg \\ h \\ c \end{bmatrix}, \tag{14}$$

3.3 Schur complement step

The difference of our approach compared to the classical dense null-space method now is the subsystem

$$\begin{bmatrix} Z^THZ & Z^TM^T \\ MZ & 0 \end{bmatrix}\begin{bmatrix} w^Z \\ \lambda \end{bmatrix}=\begin{bmatrix} Z^T(g-HYw^Y) \\ h-MYw^Y \end{bmatrix}=:\begin{bmatrix} \tilde{g} \\ \tilde{h} \end{bmatrix} \tag{15}$$

which needs to be solved as a part of (14). We required positive definiteness of the reduced Hessian Z^THZ in (2), such that (15) is a saddle point problem by itself, with positive eigenvalues according to the column dimension of Z and with $(m-1)n^{\mathrm{x}}$ negative eigenvalues due to the matching conditions M. We propose to use a Schur complement step to resolve this system, premultiplying the first row of (15) by $MZ(Z^THZ)^{-1}$ and subtracting it from the second to find

$$\begin{bmatrix} Z^THZ & Z^TM^T \\ 0 & -MZ(Z^THZ)^{-1}Z^TM^T \end{bmatrix}\begin{bmatrix} w^Z \\ \lambda \end{bmatrix}=\begin{bmatrix} \tilde{g} \\ \tilde{h}-MZ(Z^THZ)^{-1}\tilde{g} \end{bmatrix}. \tag{16}$$

By computing a Cholesky factorization $Z^T H Z = U^T U$ and defining $\hat{M} := MZU^{-1}$ we may write (16) after the Schur complement step as

$$\begin{bmatrix} U^T U & Z^T M^T \\ 0 & -\hat{M}\hat{M}^T \end{bmatrix} \begin{bmatrix} w^Z \\ \lambda \end{bmatrix} = \begin{bmatrix} \tilde{g} \\ \tilde{h} - \hat{M} U^{-T} \tilde{g} \end{bmatrix}. \tag{17}$$

As $Z^T H Z$ is block diagonal, so is the Cholesky factor U. As both Z and with U also U^{-1} are block diagonal, $\hat{M}$ inherits the bidiagonal block structure of M, and the symmetric positive definite system $\hat{M}\hat{M}^T$ is block tridiagonal,

$$(\hat{M}\hat{M}^T)_{i,i} = \hat{G}_i \hat{G}_i^T + \hat{P}_{i+1} \hat{P}_{i+1}^T =: A_i, \quad 1 \le i \le m-1, \tag{18a}$$

$$(\hat{M}\hat{M}^T)_{i+1,i} = \hat{G}_{i+1} \hat{P}_{i+1}^T =: B_{i+1}, \quad 1 \le i \le m-2, \tag{18b}$$

$$(\hat{M}\hat{M}^T)_{i,i+1} = (\hat{G}_{i+1} \hat{P}_{i+1}^T)^T = B_{i+1}^T, \quad 1 \le i \le m-2, \tag{18c}$$

with all remaining blocks of $(\hat{M}\hat{M}^T)$ being zero and with the definitions $\hat{G}_i := G_i^{\mathbb{F}} Z_i U_i^{-1}$ and $\hat{P}_{i+1} := P_{i+1}^{\mathbb{F}} Z_{i+1} U_{i+1}^{-1}$ for $1 \le i \le m-1$.

3.4 Block tridiagonal factorization step

For the symmetric and positive definite, block tridiagonal system $(\hat{M}\hat{M}^T)$ a tailored block Cholesky factorization $\tilde{V}^T \tilde{V} = \hat{M}\hat{M}^T$ may be used as described e.g in [2]. The complete block Cholesky factor $\tilde{V}$, with blocks

$$\tilde{V}_{i,i} := V_i, \quad 1 \le i \le m-1, \tag{19a}$$

$$\tilde{V}_{i,i+1} := D_{i+1}, \quad 1 \le i \le m-2, \tag{19b}$$

and again all other blocks being zero, can be computed iteratively in a loop over $i = 1, \ldots, m-1$ as follows:

$$A_i = V_i^T V_i, \qquad D_{i+1} := V_i^{-T} B_i^T, \qquad A_{i+1} := A_{i+1} - D_{i+1} D_{i+1}^T, \tag{20}$$

denoting the diagonal blocks of $\hat{M}\hat{M}^T$ by A_i and the side diagonal ones by B_i as defined in (18).

3.5 Computational effort and applicability

The computational effort involved in the described factorization of the saddle point problem (10) can be classified as follows. We note first that it is indeed sufficient to compute the following smaller TQ decompositions $R_i^{\mathbb{AF}} \begin{bmatrix} Z_i & Y_i \end{bmatrix} = \begin{bmatrix} 0 & T_i \end{bmatrix}$ instead of (12). Many active simple bounds on the unknowns hence significantly reduce the computational effort of this first step as the column dimension of $R_i^{\mathbb{AF}}$ shrinks. The same is true for both the row and column dimension of the ensuing Cholesky factorizations $Z_i^T H_i Z_i = U_i^T U_i$. The described factorization hence is appealing for use in an active set method. In particular for problems with many block local parameters, resulting e.g. from (partial) convexifications of mixed-integer problems as in [33,34], this advantage becomes noticeable.

The described factorization allows for the independent and even parallelized computation of the TQ factor's blocks Q_i, T_i and the Cholesky factor's blocks U_i. All factorizations can be computed in-place if the KKT matrix blocks themselves are no longer required. The matrices to be factored are of block local size only, i.e. their dimensions are independent of the number m of blocks. This leads to $O(m)$ runtime of this factorization, advantageous for the efficient treatment of long horizons or fine discretizations.

Note finally that requiring Z^THZ to be positive definite in order to employ a Schur complement step is not a significant restriction in applicability of our method. This requirement is in fact a subset of a well-known sufficient optimality condition for nonlinear programming, and a common assertion in most QP codes.

4 Matrix update procedures for active set exchanges

In this section we derive for the first time updates for the presented factorization for all cases of active set changes, namely for adding or removing a simple bound on an unknown of a node, and for adding or removing an inequality point constraint on a node. All updates complete in $O(mn^2)$ time as opposed to a refactorization in $O(mn^3)$ time, and respect the block structure of the KKT matrix. The first part of each of the four update procedures is identical to that for a dense null space active set method, as the factorization of Sect. 3 starts with a null space projection step. These updates need to be performed for a single node of the KKT system only, though, which already gives a significant advantage over purely dense active set techniques. The extension of these update procedures to the Schur complement step, to the tridiagonal system blocks, and to the factorization of that system is a new contribution.

4.1 Existing update techniques

It is common knowledge that Cholesky factorizations of dense matrices can be recovered in $O(n^2)$ time after certain operations such as a rank one modification $\tilde{A} = A + \alpha yy^T$, which is referred to as an *update* if $\alpha > 0$ and as a *downdate* if $\alpha < 0$. Techniques for maintaining a Cholesky factorization after appending a row and column or after removing the last row and column are well known. The same is true for QR factorizations of dense matrices when adding or removing an arbitrary row or column. A concise summary and comparison of relevant algorithms can already be found in [23], cf. also the references therein. More recent work also focuses on multiple-rank updates and on algorithms for sparse matrices, cf. [10].

For primal active-set codes for dense QPs which employ the null-space method for solving the EQP's KKT system, such as [16,18,27], it effective update procedures for the involved matrix factorizations exist and details can for example be found in [36]. Schur complement updates for a dual active-set code can be found in [3]. Overviews over algorithmic applications for existing sparse matrix and update techniques are given in [24] and [28].

4.2 Projections, Givens rotations, and permutations

We first need to introduce some notation and basic facts about a certain projection matrix, about Givens rotations, and about permutations of the components x_{ij} of the unknown x_i on a node i of the problem at hand.

4.2.1 Projections

Projections $A\mathscr{P}$ or $\mathscr{P}^T A$ are used to denote that a matrix A loses a column or a row. Some identities for these projections are given that ease the derivation of the update procedures.

Definition 41 (*Projection matrix* $\mathscr{P}$) For a matrix $A \in \mathbb{R}^{m\times n}$, the *column cutting* projection $\mathscr{P}$ and the *row cutting* projection $\mathscr{P}^T$ are defined as

$$\mathscr{P} := \begin{bmatrix} I^{n-1} \\ 0^T \end{bmatrix} \in \mathbb{R}^{n\times n-1}, \qquad \mathscr{P}^T := \begin{bmatrix} I^{m-1} & 0 \end{bmatrix} \in \mathbb{R}^{m-1\times m}. \tag{21}$$

The projection $\mathscr{P}$ is multiplied onto A from the right to cut off the last column, and it is multiplied onto A from the left to cut off the last row. Note that the row and column dimensions of $\mathscr{P}$ and $\mathscr{P}^T$ depend on those of the matrix A and are defined only in the context of usage.

The projection matrices $\mathscr{P}$ and $\mathscr{P}^T$ satisfy the identities of Lemma 42, of which (3–5) will be of particular interest in the upcoming proofs.

Lemma 42 (Identities for the projection matrices) *The projection matrices* $\mathscr{P}$ *and* $\mathscr{P}^T$ *of Definition 41 satisfy the following identities.*

1. *For* $A \in \mathbb{R}^{m\times n}$ *with* $A = \begin{bmatrix} \tilde{A} & a \end{bmatrix}$, *it holds* $A\mathscr{P} = \tilde{A} \in \mathbb{R}^{m\times n-1}$, *i.e. the matrix A loses the last column.*
2. *For* $A \in \mathbb{R}^{m\times n}$ *with* $A = \begin{bmatrix} \tilde{A} \\ a^T \end{bmatrix}$, *it holds* $\mathscr{P}^T A = \tilde{A} \in \mathbb{R}^{m-1\times n}$, *i.e. the matrix A loses the last row.*
3. *For triangular* $A \in \mathbb{R}^{n\times n}$ *for which both A and* $\mathscr{P}^T A\mathscr{P}$ *are regular, it holds that* $\left(\mathscr{P}^T A\mathscr{P}\right)^{-1} = \mathscr{P}^T A^{-1}\mathscr{P}$.
4. *For orthogonal matrices* $O \in \mathbb{R}^{n\times n}$ *it holds* $O\mathscr{P}\mathscr{P}^T O^T = I^n - oo^T$ *where* $o \in \mathbb{R}^n$ *is the last column vector of the matrix O.*
5. *For a square matrix* $A \in \mathbb{R}^{n\times n}$ *with zero bottom row,* $A = \begin{bmatrix} \tilde{A} \\ 0^T \end{bmatrix}$, *it holds that*
$$\mathscr{P}\mathscr{P}^T = \begin{bmatrix} I^{n-1} & 0 \\ 0^T & 0 \end{bmatrix} \in \mathbb{R}^{n\times n} \text{ and hence } \mathscr{P}\mathscr{P}^T A = A \text{ and } A^T\mathscr{P}\mathscr{P}^T = A^T.$$

All identities are easily verified by direct computation.

4.2.2 Orthogonal eliminations

Orthogonal eliminations, also referred to as Givens rotations or plane rotations, will be employed to restore the triangular structure of certain factors of various matrices.

Definition 43 (*Orthogonal eliminations* $\mathcal{O}_i^j$) The orthogonal elimination matrix $\mathcal{O}_i^j(v) \in \mathbb{R}^{n\times n}$, $i, j \in \{1, \dots, n\}$, $i \neq j$ for a vector $v \in \mathbb{R}^n$ is defined element-wise as

$$\left(\mathcal{O}_i^j(v)\right)_{kl} := \begin{cases} v_i/\rho & \text{if } (k,l) = (i,i) \vee (k,l) = (j,j), \\ v_j/\rho & \text{if } (k,l) = (i,j), \\ -v_j/\rho & \text{if } (k,l) = (j,i), \\ 1 & \text{if } k = l \wedge k \neq i \wedge l \neq j, \\ 0 & \text{otherwise}, \end{cases} \tag{22}$$

where $\rho := \sqrt{v_i^2 + v_j^2}$.

The purpose of an orthogonal elimination matrix $\mathcal{O}_i^j(v)$ is to eliminate element j of the vector v by modifying element i only, and Gill and Golub [23] provide more details on the properties of these matrices. The form (22) is not used in an actual numerical code for issues of efficiency and numerical stability. We refer the reader to [28] for instructions on how to compute Givens matrices accurately and for fast multiplication algorithms.

Lemma 44 (Identities for the orthogonal eliminations) *The orthogonal elimination matrices* $\mathcal{O}_i^j(v)$ *of Definition 43 satisfy the following identities:*

1. *Orthogonality:* $\mathcal{O}_i^j(v)\mathcal{O}_i^j(v)^T = I$.
2. *Modification of element* i*:* $\left(\mathcal{O}_i^j(v)v\right)_i = \rho$.
3. *Elimination of element* j*:* $\left(\mathcal{O}_i^j(v)v\right)_j = 0$.
4. *All other elements remain:* $\left(\mathcal{O}_i^j(v)v\right)_k = v_k$ *for* $k \neq i,\ k \neq j$.

Again, all identities may easily be verified by direct computation.

4.2.3 Permutations of the free unknowns

For the matrix updates adding or removing a simple bound we will make assumptions on the index $j \in \{1, \dots, n\}$ of the unknown component x_{ij} to be fixed to or freed from its lower or upper simple bound. In order to satisfy these assumptions in a numerical code, a suitable permutation must be applied to a vector x_i of node unknowns and to the existing KKT factorization as well. Lemma 45 shows that the effect of these permutations is limited to the TQ factorization's column base matrix Q_i. In particular, the southeast triangular matrix T_i is not affected.

Lemma 45 (Factorization after permutation of the free unknowns) *Let* $\Pi_i \in \mathbb{R}^{n_i^{\mathbb{F}} \times n_i^{\mathbb{F}}}$ *be permutation matrices such that* $\hat{x}_i^{\mathbb{F}} = \Pi_i x_i^{\mathbb{F}}$ *are the permuted vectors of free* *unknowns. Then the block matrices $\hat{Q}_i$, $\hat{\hat{G}}_i$, $\hat{\hat{P}}_i$, $\hat{B}_i$ and the block factors $\hat{T}_i$, $\hat{R}_i$, $\hat{A}_i$ of the permuted KKT system's factorization satisfy*

$$\hat{Q}_i = \Pi_i Q_i \tag{23}$$

and

$$\hat{T}_i = T_i,\ \hat{R}_i = R_i,\ \hat{\hat{G}}_i = \hat{G}_i,\ \hat{\hat{P}}_i = \hat{P}_i,\ \hat{A}_i = A_i,\ \hat{B}_i = B_i. \tag{24}$$

Proof We first consider the block matrix entries of the permuted KKT system itself. For invariance under the permutations Π_i of the free unknowns $x_i^{\mathbb{F}}$, it must hold that

$$\hat{R}_i^{\mathbb{AF}} := R_i^{\mathbb{AF}} \Pi_i^T, \quad \hat{G}_i^{\mathbb{F}} := G_i^{\mathbb{F}} \Pi_i^T, \quad \hat{P}_i^{\mathbb{F}} := P_i^{\mathbb{F}} \Pi_i^T, \quad \text{and} \quad \hat{H}_i^{\mathbb{FF}} := \Pi_i H_i^{\mathbb{FF}} \Pi_i^T. \tag{25}$$

This is easily seen by multiplication with $\hat{x}_i$ and elimination of $\Pi_i^T \Pi_i = I$. For the block local TQ factorizations of the free unknowns part of the active point constraints' Jacobians $R_i^{\mathbb{AF}}$, observe

$$\begin{aligned}\left[0\ T_i\right] = R_i^{\mathbb{AF}} \left[Z_i\ Y_i\right] = R_i^{\mathbb{AF}} \underbrace{\Pi_i^T \Pi_i}_{=I} \left[Z_i\ Y_i\right] &= \hat{R}_i^{\mathbb{AF}} \left[\Pi_i Z_i\ \Pi_i Y_i\right] \\ &=: \hat{R}_i^{\mathbb{AF}} \left[\hat{Z}_i\ \hat{Y}_i\right] = \left[0\ \hat{T}_i\right],\end{aligned} \tag{26}$$

which proves the claimed relations for $\hat{Y}_i$ and $\hat{Z}_i$, hence for $\hat{Q}_i$ (Eq. 23), and in particular shows invariance of the southeast triangular factor $\hat{T}_i = T_i$ (Eq. 24). For the Cholesky factors U_i of the projected Hessians $\tilde{H}_i$, we find

$$\hat{U}_i^T \hat{U}_i = \hat{Z}_i^T \hat{H}_i^{\mathbb{FF}} \hat{Z}_i = Z_i^T \underbrace{\Pi_i^T (\Pi_i}_{=I} H_i^{\mathbb{FF}} \underbrace{\Pi_i^T) \Pi_i}_{=I} Z_i = Z_i^T H_i^{\mathbb{FF}} Z_i = U_i^T U_i, \tag{27}$$

hence the Cholesky factors $\hat{U}_i$ of the permuted KKT system's factorization are identical to the factors U_i of the original one. The Schur complements $\hat{\hat{G}}_i$ and $\hat{\hat{P}}_i$ after permutation are unaffected as well,

$$\hat{\hat{G}}_i = \hat{G}_i^{\mathbb{F}} \hat{Z}_i \hat{U}_i^{-1} = (G_i^{\mathbb{F}} \underbrace{\Pi_i^T) \Pi_i}_{=I} Z_i U_i^{-1} = G_i^{\mathbb{F}} Z_i U_i^{-1} = \hat{G}_i, \tag{28}$$

and analogously for $\hat{\hat{P}}_i$. Invariance carries over to the blocks $\hat{A}_i$ and $\hat{B}_i$ of the positive definite block tridiagonal system by (Eq. 18), and hence also to the Cholesky factor blocks $\hat{V}_i$, $\hat{D}_i$ by (Eq. 20).

4.3 Adding a simple bound

In the following we distinguish values *after* the active set update by a bar (as in $\overline{T}$) from their counterpart values *before* the update. The dimensions $n^{\mathbb{F}}$ and $n^{\mathbb{X}}$ denote the number of free and fixed unknowns, while $n^{\mathbb{Y}}$ and $n^{\mathbb{Z}}$ denote the range-space and the null-space dimensions of the TQ factorization of the point constraint matrix $R_i^{\mathbb{AF}}$. If a primally blocking simple bound is encountered as a result of the step length determination, we may assume it w.l.o.g. to fix the last component $j = n^{\mathbb{F}}$ of the vector $w_i^{\mathbb{F}}$ of unknowns by applying a suitable permutation to $w_i^{\mathbb{F}}$, to the columns of the matrix Q_i prior to the update, and to the KKT matrix, cf. Lemma 45.

4.3.1 TQ factorization update

The new bounds's unit row vector $e_{n^{\mathbb{F}}}$ is added to the TQ factorization of the constraints matrix (13) comprising the simple bounds and the decoupled point constraints

$$\begin{bmatrix} e_{n^{\mathbb{F}}}^T & 0^T \\ 0 & I_i^{\mathbb{X}} \\ R_i^{\mathbb{AF}} & R_i^{\mathbb{X}} \end{bmatrix} \begin{bmatrix} Z_i & Y_i & \\ & & I_i^{\mathbb{X}} \end{bmatrix} = \begin{bmatrix} t^{\mathbb{Z}^T} & t^{\mathbb{Y}^T} & \\ & & I_i^{\mathbb{X}} \\ & T_i & R_i^{\mathbb{AX}} \end{bmatrix}. \tag{29}$$

We eliminate the elements of t using a sequence $\mathscr{O}_{\mathrm{ZT}}$ of $n^{\mathbb{F}} - 1$ Givens rotations

$$\mathscr{O}_{\mathrm{Z}} := \mathscr{O}_2^{1\,T} \cdot \ldots \cdot \mathscr{O}_{n^{\mathbb{Z}}}^{n^{\mathbb{Z}}-1\,T},\ \mathscr{O}_{\mathrm{T}} := \mathscr{O}_{n^{\mathbb{Z}}+1}^{n^{\mathbb{Z}}\;\;T} \cdot \ldots \cdot \mathscr{O}_{n^{\mathbb{F}}}^{n^{\mathbb{F}}-1\,T},\ \mathscr{O}_{\mathrm{ZT}} := \begin{bmatrix} \mathscr{O}_{\mathrm{Z}} & 0 \\ 0 & I \end{bmatrix} \mathscr{O}_{\mathrm{T}}, \tag{30}$$

to transform it to the unit vector $e_{n^{\mathbb{F}}}$ as follows,

$$\begin{bmatrix} e_{n^{\mathbb{F}}}^T & 0^T \\ 0 & I_i^{\mathbb{X}} \\ R_i^{\mathbb{AF}} & R_i^{\mathbb{AX}} \end{bmatrix} \begin{bmatrix} Z_i & Y_i & \\ & & I_i^{\mathbb{X}} \end{bmatrix} \begin{bmatrix} \mathscr{O}_{\mathrm{ZT}} & \\ & I \end{bmatrix} = \begin{bmatrix} 0^T & 0^T & 1 & \\ & & & I_i^{\mathbb{X}} \\ & \overline{T}_i & r & R_i^{\mathbb{AX}} \end{bmatrix} = \begin{bmatrix} 0^T & 0^T & \overline{I}_i^{\mathbb{X}} \\ & \overline{T}_i & \overline{R}_i^{\mathbb{AX}} \end{bmatrix}, \tag{31}$$

having thus restored the required shape of the right hand side matrix as follows: The last $n^{\mathbb{F}} - n^{\mathbb{Z}}$ rotations of the sequence $\mathscr{O}_{\mathrm{T}}$ introduce an additional reverse subdiagonal into T_i. Shifting it to the left we obtain the new TQ southeast triangular factor $\overline{T}_i$ and the null space dimension shrinks by one. The remaining column r enters $\overline{R}_i^{\mathbb{AX}}$, now belonging to first fixed component $w_{i,1}^{\mathbb{X}}$ of the unknown.

The first $n^{\mathbb{Z}} - 1$ rotations of the sequence $\mathscr{O}_{\mathrm{Z}}$ affect the basis matrix Z_i, whose last column is eliminated to zero and cut off as the null space dimension shrinks,

$$\begin{bmatrix} Z_i & Y_i \end{bmatrix} \mathscr{O}_{\mathrm{Z}} = \begin{bmatrix} \overline{Z}_i & \overline{Y}_i \\ & 1 \end{bmatrix}, \quad \overline{Z}_i = \mathscr{P}^T Z_i \mathscr{O}_{\mathrm{Z}} \mathscr{P}. \tag{32}$$

The projected Hessian's factor after applying the rotations $\mathscr{O}_{\mathrm{Z}}$ would be $U_i\mathscr{O}_{\mathrm{Z}}\mathscr{P}$ which can be seen from

$$\begin{aligned}\overline{U}_i^T\overline{U}_i = \overline{Z}_i^T\overline{H}_i^{\mathbb{F}\mathbb{F}}\overline{Z}_i &= \mathscr{P}^T\mathscr{O}_{\mathrm{Z}}^T Z_i^T\mathscr{P}(\mathscr{P}^T H_i^{\mathbb{F}\mathbb{F}}\mathscr{P})\mathscr{P}^T Z_i\mathscr{O}_{\mathrm{Z}}\mathscr{P} && \text{(33a)}\\ &= \mathscr{P}^T\mathscr{O}_{\mathrm{Z}}^T Z_i^T H_i^{\mathbb{F}\mathbb{F}} Z_i\mathscr{O}_{\mathrm{Z}}\mathscr{P} \quad \text{(by Lemma 42, 5.)} && \text{(33b)}\\ &= (U_i\mathscr{O}_{\mathrm{Z}}\mathscr{P})^T(U_i\mathscr{O}_{\mathrm{Z}}\mathscr{P}), && \text{(33c)}\end{aligned}$$

which no longer is a Cholesky type factor. The sequence $\mathscr{O}_{\mathrm{Z}}$ introduces a subdiagonal of nonzero elements into $U_i\mathscr{O}_{\mathrm{Z}}\mathscr{P}$, destroying its shape. A second sequence $\mathscr{O}_{\mathrm{U}}$ of $n^{\mathbb{Z}}-1$ Givens rotations

$$\mathscr{O}_{\mathrm{U}} := \mathscr{O}_{n^{\mathbb{Z}}}^{n^{\mathbb{Z}}-1}\cdot\ldots\cdot\mathscr{O}_2^1 \tag{34}$$

is used to restore the triangular form of the new projected Hessian factor $\overline{U}_i$,

$$\overline{U}_i = \mathscr{P}^T\mathscr{O}_{\mathrm{U}}U_i\mathscr{O}_{\mathrm{Z}}\mathscr{P}. \tag{35}$$

whose last row is eliminated to zero as the row dimension shrinks by one, reflecting the shrinked null space dimension.

4.3.2 Schur complement update

The principal procedure so far can also be found in e.g. in [36] for the dense null-space active set method. We continue our presentation by contributing an extension of the matrix update procedures to the further steps of Sect. 3. We denote by g the last column of $G_i^{\mathbb{F}}$ belonging to the component $w_{ij}^{\mathbb{F}}$ that gets fixed to its bound, by z^T the last row of the null space basis matrix Z_i, and by o the the last column of the Givens sequence $\mathscr{O}_{\mathrm{Z}}$. For the update to the Schur complement parts $\hat{G}_i$ and $\hat{P}_i$ we find constructively

$$\begin{aligned}\overline{\hat{G}}_i = \overline{G}_i^{\mathbb{F}}\overline{Z}_i\overline{U}_i^{-1} &= G_i^{\mathbb{F}}\mathscr{P}\left(\mathscr{P}^T Z_i\mathscr{O}_{\mathrm{Z}}\mathscr{P}\right)\left(\mathscr{P}^T\mathscr{O}_{\mathrm{U}}U_i\mathscr{O}_{\mathrm{Z}}\mathscr{P}\right)^{-1} && \text{(36a)}\\ &= G_i^{\mathbb{F}}\mathscr{P}\left(\mathscr{P}^T Z_i\mathscr{O}_{\mathrm{Z}}\mathscr{P}\right)\mathscr{P}^T(\mathscr{O}_{\mathrm{U}}U_i\mathscr{O}_{\mathrm{Z}})^{-1}\mathscr{P} \quad \text{(Lemma 42, 3.)} && \text{(36b)}\\ &= G_i^{\mathbb{F}}(\mathscr{P}\mathscr{P}^T Z_i)\left(\mathscr{O}_{\mathrm{Z}}\mathscr{P}\mathscr{P}^T\mathscr{O}_{\mathrm{Z}}^T\right)U_i^{-1}\mathscr{O}_{\mathrm{U}}^T\mathscr{P} && \text{(36c)}\\ &= G_i^{\mathbb{F}}Z_i\left(\mathscr{O}_{\mathrm{Z}}\mathscr{P}\mathscr{P}^T\mathscr{O}_{\mathrm{Z}}^T\right)U_i^{-1}\mathscr{O}_{\mathrm{U}}^T\mathscr{P} \quad \text{(Lemma 42, 5.)} && \text{(36d)}\\ &= \left(G_i^{\mathbb{F}}Z_i - gz^T\right)\left(I - oo^T\right)U_i^{-1}\mathscr{O}_{\mathrm{U}}^T\mathscr{P} \quad \text{(Lemma 42, 4.)} && \text{(36e)}\\ &= \left(G_i^{\mathbb{F}}Z_i - gz^T\right)U_i^{-1}\mathscr{O}_{\mathrm{U}}^T\mathscr{P} - \left(G_i^{\mathbb{F}}Z_i - gz\right)o\underbrace{\left(o^T U_i^{-1}\mathscr{O}_{\mathrm{U}}^T\mathscr{P}\right)}_{=0} && \text{(36f)}\\ &= (G_i^{\mathbb{F}}Z_iU_i^{-1})\mathscr{O}_{\mathrm{U}}^T\mathscr{P} - g\underbrace{(z^T U_i^{-1}\mathscr{O}_{\mathrm{U}}^T\mathscr{P})}_{=0} = \hat{G}_i\mathscr{O}_{\mathrm{U}}^T\mathscr{P}. && \text{(36g)}\end{aligned}$$

In step (36f) consider that $o^T U_i^{-1} \mathscr{O}_{\mathrm{U}}^T$ is the last row of the inverse of the updated Cholesky factor $\mathscr{O}_{\mathrm{U}} U_i \mathscr{O}_{\mathrm{Z}}$ in (Eq. 35). Since U_i is upper triangular, so is its inverse, and of its last row only the last element is nonzero. This element is cut off such that we have $o^T U_i^{-1} \mathscr{O}_{\mathrm{U}}^T \mathscr{P} = 0$. For step (36g) consider that

$$\begin{bmatrix} q_1^T & q_2 \end{bmatrix} = z^T U_i^{-1} \mathscr{O}_{\mathrm{U}}^T \;\Leftrightarrow\; \begin{bmatrix} q_1^T & q_2 \end{bmatrix} \mathscr{O}_{\mathrm{U}} U_i \mathscr{O}_{\mathrm{Z}} = z^T \mathscr{O}_{\mathrm{Z}} = \begin{bmatrix} 0^T & \zeta \end{bmatrix} \tag{37}$$

holds by construction of the rotation sequence $\mathscr{O}_{\mathrm{Z}}$ in (32). Since $\mathscr{O}_{\mathrm{U}} U_i \mathscr{O}_{\mathrm{Z}}$ is upper triangular, it also holds that $0 = q_1^T = z^T U_i^{-1} \mathscr{O}_{\mathrm{U}}^T \mathscr{P}$.

Summarizing, we have shown that the updated matrices $\overline{\hat{G}}_i$ and $\overline{\hat{P}}_i$ are obtained from the old ones $\hat{G}_i$ and $\hat{P}_i$ by applying the sequence $\mathscr{O}_{\mathrm{U}}$ of Givens rotations used to restore the projected Hessian factor's triangular shape. Note that $\overline{\hat{G}}_i$ and $\overline{\hat{P}}_i$ lose a column compared to $\hat{G}_i$ and $\hat{P}_i$, reflecting that the number of free variables $n^{\mathbb{F}}$ decreased.

4.3.3 Block tridiagonal system update

For the tridiagonal system blocks A_{i-1}, A_i (exemplarily), and B_i affected by the updates to $\hat{G}_i$ and $\hat{P}_i$ we denote by o^T the last row of the sequence $\mathscr{O}_{\mathrm{U}}$ and find

$$\begin{aligned}
\overline{A}_i = \overline{\hat{G}}_i \overline{\hat{G}}_i^T + \hat{P}_{i+1} \hat{P}_{i+1}^T &= \hat{G}_i \mathscr{O}_{\mathrm{U}}^T \mathscr{P} \left(\hat{G}_i \mathscr{O}_{\mathrm{U}}^T \mathscr{P} \right)^T + \hat{P}_{i+1} \hat{P}_{i+1}^T \\
&= \hat{G}_i \left(\mathscr{O}_{\mathrm{U}}^T \mathscr{P} \mathscr{P}^T \mathscr{O}_{\mathrm{U}} \right) \hat{G}_i^T + \hat{P}_{i+1} \hat{P}_{i+1}^T \\
&= \hat{G}_i \left(I - o o^T \right) \hat{G}_i^T + \hat{P}_{i+1} \hat{P}_{i+1}^T \quad \text{(Lemma 42, 4.)} \\
&= \hat{G}_i \hat{G}_i^T - \left(\hat{G}_i o \right) \left(\hat{G}_i o \right)^T + \hat{P}_{i+1} \hat{P}_{i+1}^T = A_i - g g^T
\end{aligned}$$

where $\hat{g} := \hat{G}_i o$. By the same mechanism we find with $\hat{p} := \hat{P}_i o$ that $\overline{A}_{i-1} = A_{i-1} - \hat{p} \hat{p}^T$ and $\overline{B}_i = B_i - \hat{g} \hat{p}^T$. Writing this as

$$\begin{bmatrix} \ddots & & & \\ & \overline{A}_{i-1} & \overline{B}_i^T & \\ & \overline{B}_i & \overline{A}_i & \\ & & & \ddots \end{bmatrix} = \begin{bmatrix} \ddots & & & \\ & A_{i-1} & B_i^T & \\ & B_i & A_i & \\ & & & \ddots \end{bmatrix} - \begin{bmatrix} \vdots \\ 0 \\ \hat{p} \\ \hat{g} \\ 0 \\ \vdots \end{bmatrix} \begin{bmatrix} \vdots \\ 0 \\ \hat{p} \\ \hat{g} \\ 0 \\ \vdots \end{bmatrix}^T \tag{38}$$

we see that these modifications take the form of a subtraction of a dyadic product from a 2×2 subblock of the block tridiagonal system (18). We're left with devising a downdate procedure for the block Cholesky factorization of this system. As this situation will arise for all four update procedures proposed, this task shall be postponed until they have all been presented.

4.4 Adding a point constraint

If any of the inactive decoupled point constraints on a node i becomes active, the row $r^T := R_j^{\mathbb{F}}$ and its TQ factorization, $j \in \{1, \dots, n^{\mathrm{r}}\}$ is appended to the end of $R_i^{\mathbb{AF}}$,

$$\begin{bmatrix} R_i^{\mathbb{AF}} \\ r^T \end{bmatrix} \begin{bmatrix} Z_i & Y_i \end{bmatrix} = \begin{bmatrix} 0 & T_i \\ v^T & u^T \end{bmatrix}. \tag{39}$$

To restore the triangular structure of the right hand side of (39), a series of $n^{\mathbb{Z}} - 1$ Givens rotations

$$\mathscr{O}_{\mathrm{T}} := \mathscr{O}_2^{1\,T} \cdot \dots \cdot \mathscr{O}_{n^{\mathbb{Z}}}^{n^{\mathbb{Z}}-1\,T} \tag{40}$$

is applied. Elements of v^T are eliminated step by step by modifying the element to the right, until the transformed last element γ remains and the southeast triangular structure has been restored,

$$\overline{R}_i^{\mathbb{AF}} \begin{bmatrix} \overline{Z}_i & \overline{Y}_i \end{bmatrix} := \begin{bmatrix} R_i^{\mathbb{AF}} \\ r^T \end{bmatrix} \begin{bmatrix} Z_i & Y_i \end{bmatrix} \begin{bmatrix} \mathscr{O}_{\mathrm{T}} & 0 \\ 0 & I^{n^{\mathbb{Y}}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & T_i \\ 0^T & \gamma & u^T \end{bmatrix} =: \begin{bmatrix} 0 & \overline{T}_i \end{bmatrix}. \tag{41}$$

The factor $\overline{T}_i$ gains a row and column as the range space dimension increases. The sequence $\mathscr{O}_{\mathrm{T}}$ affects the old null space basis Z_i only and we find

$$\overline{Z}_i := Z_i \mathscr{O}_{\mathrm{T}} \mathscr{P}, \qquad \overline{Y}_i := \begin{bmatrix} z & Y_i \end{bmatrix} \tag{42}$$

where z is the last column of $Z_i \mathscr{O}_{\mathrm{T}}$. As in Sect. 4.3.1 the projected Hessian factor's triangular shape needs to be recovered by the sequence

$$\mathscr{O}_{\mathrm{U}} := \mathscr{O}_{n^{\mathbb{Z}}-1}^{n^{\mathbb{Z}}} \cdot \dots \cdot \mathscr{O}_1^2, \tag{43}$$

resulting in the following update procedure to find $\overline{U}_i$,

$$\overline{U}_i := \mathscr{P}^T \mathscr{O}_{\mathrm{U}} U_i \mathscr{O}_{\mathrm{T}} \mathscr{P}. \tag{44}$$

For the matrices $\hat{G}_i$ (exemplarily) and $\hat{P}_i$ the relation $\hat{G}_i U_i = G_i^{\mathbb{F}} Z_i$ holds, so $\overline{\hat{G}}_i$ loses a column like $\overline{Z}_i$ did. We find with t denoting the last row of $\mathscr{O}_{\mathrm{T}}$ that

$$\begin{aligned}
\overline{\hat{G}}_i &= G_i^{\mathbb{F}} \overline{Z}_i \overline{U}_i^{-1} && \text{(45a)} \\
&= G_i^{\mathbb{F}} (Z_i \mathscr{O}_{\mathrm{T}} \mathscr{P})(\mathscr{P}^T \mathscr{O}_{\mathrm{U}} U_i \mathscr{O}_{\mathrm{T}} \mathscr{P})^{-1} && \text{(45b)} \\
&= G_i^{\mathbb{F}} Z_i (\mathscr{O}_{\mathrm{T}} \mathscr{P} \mathscr{P}^T \mathscr{O}_{\mathrm{T}}^T) U_i^{-1} \mathscr{O}_{\mathrm{U}}^T \mathscr{P} \qquad \text{(Lemma 42, 3.)} && \text{(45c)} \\
&= (G_i^{\mathbb{F}} Z_i U_i^{-1} - G_i^{\mathbb{F}} Z_i t t^T U_i^{-1}) \mathscr{O}_{\mathrm{U}}^T \mathscr{P} \qquad \text{(Lemma 42, 4.)} && \text{(45d)} \\
&= \hat{G}_i \mathscr{O}_{\mathrm{U}}^T \mathscr{P} - G_i^{\mathbb{F}} Z_i t \underbrace{(t^T U_i^{-1} \mathscr{O}_{\mathrm{U}}^T \mathscr{P})}_{=0}. && \text{(45e)}
\end{aligned}$$

With the reasoning of Sect. 4.3.2, we find that in the last step (45e) that the second term vanishes. With this result we proceed as in Sect. 4.3.3.

4.5 Deleting a simple bound

If an active simple bound on w_{ij} becomes inactive on node i, we may assume by Lemma 45 that it is the bound $j = n^{\mathbb{F}} + 1$ on the first of the fixed components of w_i by applying a suitable permutation to $w_i^{\mathbb{X}}$, to v_i, and to the KKT matrix blocks. In the extended constraints matrix (13), the first column $r^{\mathbb{AX}}$ of $R_i^{\mathbb{AX}}$ becomes the last one of T_i,

$$\begin{bmatrix} & & & I_i^{\mathbb{X}} \\ R_i^{\mathbb{AF}} & r^{\mathbb{AX}} & \overline{R}^{\mathbb{AX}} \end{bmatrix} \begin{bmatrix} Z_i & Y_i & & \\ & & 1 & \\ & & & I_i^{\mathbb{X}} \end{bmatrix} = \begin{bmatrix} & & & I_i^{\mathbb{X}} \\ 0 & T_i & r^{\mathbb{AX}} & \overline{R}_i^{\mathbb{AX}} \end{bmatrix}. \tag{46}$$

The sequence $\mathscr{O}_{\mathrm{T}}$ of $n^{\mathbb{Y}}$ Givens rotations

$$\mathscr{O}_{\mathrm{T}} := \mathscr{O}_{n^{\mathbb{F}}}^{n^{\mathbb{F}}+1\,T} \cdot \ldots \cdot \mathscr{O}_{n^{\mathbb{Z}}+1}^{n^{\mathbb{Z}}+2\,T} \tag{47}$$

restores the triangular shape of the right hand side of (46) by eliminating each element on the reverse subdiagonal using the element to the right,

$$\begin{bmatrix} 0 & \overline{T}_i \end{bmatrix} := \begin{bmatrix} T_i & r^{\mathbb{AX}} \end{bmatrix} \mathscr{O}_{\mathrm{T}}. \tag{48}$$

This sequence leaves Z_i unaffected, but affects the first column of Y_i which becomes the new last one of $\overline{Z}_i$ as the null space dimension grows by one,

$$\overline{Z}_i := \begin{bmatrix} Z_i & z \\ 0^T & \zeta \end{bmatrix} \quad \text{where} \quad \begin{bmatrix} z & \overline{Y}_i \\ \zeta & y^T \end{bmatrix} := \begin{bmatrix} Y_i & 0 \\ 0^T & 1 \end{bmatrix} \mathscr{O}_{\mathrm{T}}. \tag{49}$$

To find the new projected Hessian factor $\overline{U}_i$ we denote its new column's by $u \in \mathbb{R}^{n^{\mathbb{Z}}}$ and $\rho \in \mathbb{R}$, and denote the elements of the new Hessian $\overline{H}_i^{\mathbb{FF}}$ by $h \in \mathbb{R}^{n^{\mathbb{F}}}$ and $\eta \in \mathbb{R}$,

$$\overline{U}_i =: \begin{bmatrix} U_i & u \\ 0^T & \rho \end{bmatrix}, \qquad \overline{H}_i^{\mathbb{FF}} =: \begin{bmatrix} H_i^{\mathbb{FF}} & h \\ h^T & \eta \end{bmatrix}. \tag{50}$$

From expanding $\overline{Z}_i^T \overline{H}_i^{\mathbb{FF}} \overline{Z}_i = \overline{U}_i^T \overline{U}_i$ we find that

$$\begin{bmatrix} Z_i^T & 0 \\ z^T & \zeta \end{bmatrix} \begin{bmatrix} H_i^{\mathbb{FF}} & h \\ h^T & \eta \end{bmatrix} \begin{bmatrix} Z_i & z \\ 0^T & \zeta \end{bmatrix} = \begin{bmatrix} U_i^T U_i & U_i^T u \\ u^T U_i & u^T u + \rho^2 \end{bmatrix}. \tag{51}$$

Hence we can compute the factor's new entries u and ρ from the now unfixed entries h and η of the Hessian $\overline{H}_i^{\mathbb{FF}}$ of the free variables,

$$U_i^T u = Z_i^T (H_i^{\mathbb{FF}} z + h\zeta), \tag{52a}$$

$$\rho = \sqrt{z^T (H_i^{\mathbb{FF}} z + h\zeta) + \zeta(h^T z + \eta\zeta) - u^T u}. \tag{52b}$$

Since the initial $n^{\mathbb{Z}}$ components of $\overline{Z}_i$ and $\overline{U}_i$ remained unchanged from Z_i and U_i, we find for $\hat{G}_i$ (exemplarily) and $\hat{P}_i$ that $\overline{\hat{G}}_i = \left[\, \hat{G}_i \;\, \hat{g} \,\right]$ where $\hat{g}$ is a new additional column that can be found as follows:

$$\left[\, \hat{G}_i \;\, \hat{g} \,\right] \overline{U}_i = \left[\, G_i^{\mathbb{F}} \;\, g_1^{\mathbb{X}} \,\right] \overline{Z}_i \tag{53a}$$

$$\iff \left[\, \hat{G}_i U_i \;\; \hat{G}_i u + \hat{g}\rho \,\right] = \left[\, G_i^{\mathbb{F}} \;\, g_1^{\mathbb{X}} \,\right] \begin{bmatrix} Z_i & z \\ 0^T & \zeta \end{bmatrix} = \left[\, G_i^{\mathbb{F}} Z_i \;\; G_i^{\mathbb{F}} z + g_1^{\mathbb{X}} \zeta \,\right] \tag{53b}$$

thus $\hat{g} = (G_i^{\mathbb{F}} z + g_1^{\mathbb{X}} \zeta - \hat{G} u)/\rho$. Finally, for the tridiagonal system blocks A_{i-1} (exemplarily), B_i, and A_i we find

$$\overline{A}_i = \overline{\hat{G}}_i \overline{\hat{G}}_i^T + \hat{P}_{i+1} \hat{P}_{i+1}^T = \hat{G}_i \hat{G}_i^T + \hat{g}\hat{g}^T + \hat{P}_{i+1} \hat{P}_{i+1}^T \tag{54}$$

thus they are affected by a rank one update this time. With this result we proceed as in Sect. 4.7.

4.6 Deleting a point constraint

If a decoupled point constraint on node i becomes inactive, a row $j \in \{1, \ldots, n_i^{\mathrm{r}}\}$ is removed from $R_i^{\mathbb{AF}}$ and the triangular factor T_i. This yields

$$\overline{R}_i^{\mathbb{AF}} \left[\, Z_i \;\, Y_i \,\right] = \left[\, 0 \;\, \tilde{T}_i \,\right] \tag{55}$$

after which we restore the triangular structure of $\tilde{T}_i$, that has been destroyed by the removal of row j, using the series of $n^{\mathbb{Y}} - j$ Givens rotations

$$\mathscr{O}_{\mathrm{T}} := \mathscr{O}_{n^{\mathbb{Z}}+n^{\mathbb{Y}}-j+1}^{n^{\mathbb{Z}}+n^{\mathbb{Y}}-j+2^T} \cdot \ldots \cdot \mathscr{O}_{n^{\mathbb{Z}}+2}^{n^{\mathbb{Z}}+1}. \tag{56}$$

This results in

$$\left[\, 0 \;\, \overline{T}_i \,\right] := \left[\, \tilde{T}_i \,\right] \mathscr{O}_{\mathrm{T}}. \tag{57}$$

We find that T_i shrinks by one row and column reflecting the increased dimension of the null space. The null space basis Z_i remains unaffected by $\mathscr{O}_{\mathrm{T}}$ and we have

$$\overline{Z}_i := \left[\, Z_i \;\, y \,\right], \qquad \left[\, y \;\, \overline{Y}_i \,\right] := Y_i \mathscr{O}_{\mathrm{T}} \tag{58}$$

where y denotes the first column of Y_i. The factorization $\overline{U}_i^T \overline{U}_i$ of the projected Hessian $\overline{Z}_i^T H_i^{\mathbb{FF}} \overline{Z}_i$ is easily restored: Denoting the new elements of the Hessian factor with a vector u and a scalar ρ as in (52) we find

$$\overline{Z}_i^T H_i^{\mathbb{FF}} \overline{Z}_i = \overline{U}_i^T \overline{U}_i \quad \Longleftrightarrow \quad \begin{bmatrix} Z_i^T H_i^{\mathbb{FF}} Z_i & Z_i^T H_i^{\mathbb{FF}} y \\ y^T H_i^{\mathbb{FF}} Z_i & y^T H_i^{\mathbb{FF}} y \end{bmatrix} = \begin{bmatrix} U_i^T U_i & U_i^T u \\ u^T U_i & u^T u + \rho^2 \end{bmatrix} \tag{59}$$

and from this we determine the new column $\begin{bmatrix} u^T & \rho \end{bmatrix}^T$ of the Hessian factor $\overline{U}_i$ to be

$$u := U_i^{-T} Z_i^T H_i^{\mathbb{FF}} y, \qquad \rho := \sqrt{y^T H_i^{\mathbb{FF}} y - u^T u}. \tag{60}$$

Since the initial $n^{\mathbb{Z}}$ components of $\overline{Z}_i$ and $\overline{U}_i$ remained unchanged, we find $\overline{\hat{G}}_i = \begin{bmatrix} \hat{G}_i & \hat{g} \end{bmatrix}$ where $\hat{g}$ is a new additional column that is determined as follows. From

$$\begin{bmatrix} \hat{G}_i & \hat{g} \end{bmatrix} \overline{U}_i = G_i^{\mathbb{F}} \overline{Z}_i \quad \Longleftrightarrow \quad \begin{bmatrix} \hat{G}_i U_i & \hat{G}_i u + \hat{g} \rho \end{bmatrix} = \begin{bmatrix} G_i^{\mathbb{F}} Z_i & G_i^{\mathbb{F}} y \end{bmatrix} \tag{61}$$

we determine that new new column $\hat{g}$ of $\hat{G}_i$ is

$$\overline{\hat{g}} = (G_i^{\mathbb{F}} y - \hat{G}_i u)/\rho. \tag{62}$$

Finally, for the tridiagonal system blocks A_{i-1}, B_i, and A_i (exemplarily) we find

$$\overline{A}_i = \overline{\hat{G}}_i \overline{\hat{G}}_i^T + \hat{P}_{i+1} \hat{P}_{i+1}^T = \hat{G}_i \hat{G}_i^T + \hat{g} \hat{g}^T + \hat{P}_{i+1} \hat{P}_{i+1}^T, \tag{63}$$

thus A_i is affected by a rank one update again. With this result we proceed as in Sect. 4.7.

4.7 Modifying the tridiagonal block Cholesky factorization

We conclude the presentation of the proposed update procedures by deriving modifications for the tridiagonal block Cholesky factorization of system (18). Our modification procedures will treat a rank one update or downdate of two neighboring blocks A_{i-1} and A_i together with their subdiagonal block B_i. As we have seen, this situation arises as common final part of all four cases of active set changes. In order to derive a *downdate* procedure for this factorization, we carry out a single step of the block tridiagonal Cholesky factorization of Sect. 3.4, incorporating the subtraction of the dyadic product.

4.7.1 Downdating a diagonal block

For downdating a single diagonal block A_i

$$\overline{V}_{i-1}^T \overline{V}_{i-1} = \overline{A}_{i-1} = A_{i-1} - pp^T, \tag{64}$$

we employ method C3 of [23], which works by forming

$$\begin{bmatrix} v & V_{i-1} \\ \delta_\mathrm{n} & 0^T \end{bmatrix} \tag{65}$$

with $v = V_{i-1}^{-T} p$, $\delta_\mathrm{n}^2 := 1 - v^T v$ and transforming $\begin{bmatrix} v^T & \delta_\mathrm{n} \end{bmatrix}$ into the scaled unit vector $\delta_0 e_\mathrm{n}^T$ by applying a sequence of Givens plane rotations

$$\mathscr{O}_\mathrm{V} = \mathscr{O}^1_{n^\mathbb{Z}+1} \cdot \cdots \cdot \mathscr{O}^{n^\mathbb{Z}}_{n^\mathbb{Z}+1} \tag{66}$$

to (65), yielding

$$\mathscr{O}_\mathrm{V} \begin{bmatrix} v & V_{i-1} \\ \delta_\mathrm{n} & 0^T \end{bmatrix} =: \begin{bmatrix} G_{11} & g_{12} \\ g_{21}^T & \gamma_{22} \end{bmatrix} \begin{bmatrix} v & V_{i-1} \\ \delta_\mathrm{n} & 0^T \end{bmatrix} = \begin{bmatrix} 0 & \overline{V}_{i-1} \\ \delta_0 & r^T \end{bmatrix}. \tag{67}$$

Correctness can be verified by multiplying each side of (Eq. 67) by its transpose and comparing entries, which yields $\overline{V}_{i-1}^T \overline{V}_{i-1} + pp^T$. By comparison of elements in (Eq. 67) we find with $\overline{V}_{i-1} = G_{11} V_{i-1}$ an explicit expression for $\overline{V}_{i-1}$ and also find the identity $g_{21} = v$ to be used in the next section.

4.7.2 Downdating a side diagonal block

From the block tridiagonal Cholesky algorithm we have the identity $\overline{D}_i = \overline{V}_{i-1}^{-T} \overline{B}_i^T$ which for the subdiagonal block's update leads to $\overline{D}_i = G_{11}^{-T} V_{i-1}^{-T} \overline{B}_i^T$. The difficulty here lies with finding G_{11}^{-T}, as G_{11} is not orthogonal. To this end, we expand to

$$\begin{bmatrix} \overline{D}_i \\ d \end{bmatrix} = \mathscr{O}_\mathrm{V} \begin{bmatrix} v & V_{i-1} \\ \delta_\mathrm{n} & 0^T \end{bmatrix}^{-T} \begin{bmatrix} 0 \\ B_i^T - pg^T \end{bmatrix} \tag{68}$$

Forming the inverse and carrying out the right matrix–vector product yields

$$\begin{bmatrix} \overline{D}_i \\ d \end{bmatrix} = \mathscr{O}_\mathrm{V} \begin{bmatrix} V_{i-1}^{-T}(B_i^T - pg^T) \\ -\frac{1}{\delta_\mathrm{n}} v^T V_{i-1}^{-T}(B_i^T - pg^T) \end{bmatrix} = \mathscr{O}_\mathrm{V} \begin{bmatrix} D_i - vg^T \\ -\frac{1}{\delta_\mathrm{n}}(D_i - vg^T) \end{bmatrix} \tag{69}$$

which can easily be computed by alongside the updated factor $\overline{V}_{i-1}$ by applying the sequence of Givens plane rotations.

4.7.3 The dyadic downdate to the following diagonal block

We now compute $\overline{D}_i^T \overline{D}_i$ for the third step. We define $\beta := 1/(1 - v^T v) = 1/\delta_n^2$ for brevity.

$$\overline{D}_i^T \overline{D}_i = \overline{B}_i V_{i-1}^{-1} G_{11}^{-1} G_{11}^{-T} V_{i-1}^{-T} \overline{B}_i^T \tag{70a}$$
$$= \overline{B}_i V_{i-1}^{-1} (I - g_{21} g_{21}^T)^{-1} V_{i-1}^{-T} \overline{B}_i^T \tag{70b}$$
$$= \overline{B}_i V_{i-1}^{-1} \left(I + \beta v v^T \right) V_{i-1}^{-T} \overline{B}_i^T \tag{70c}$$
$$= (D_i^T - g v^T)(D_i - v g^T) + \beta (D_i^T v - g v^T v)(v^T D_i - v^T v g^T) \tag{70d}$$

Here we have exploited orthogonality of G_{V} and applied the Sherman–Morrison formula in step (70c) to find $(G_{11}^T G_{11})^{-1}$. Expanding, collecting identical terms and using the identity $1 + \beta v^T v = \beta$ we find

$$\overline{D}_i^T \overline{D}_i = D_i^T D_i - \beta D_i^T v g^T - \beta g v^T D_i + \beta v^T v g g^T + \beta D_i^T v v^T D_i \tag{71a}$$
$$= D_i^T D_i - \beta D_i^T v g^T - \beta g v^T D_i + \beta g g^T + \beta D_i^T v v^T D_i - g g^T \tag{71b}$$
$$= D_i^T D_i + \beta (D_i^T v - g)(D_i^T v - g)^T - g g^T \tag{71c}$$

Using this relation, the third step of the block tridiagonal Cholesky factorization reads:

$$\overline{V}_i^T \overline{V}_i = \overline{A}_i - \overline{D}_i^T \overline{D}_i - g g^T = A_i - D_i^T D_i - d d^T = V_i^T V_i - d d^T \tag{72}$$

with a vector $d = \frac{1}{\delta_{\mathrm{n}}}(D_i^T v - g)$. This vector may also be written as $d = \frac{1}{\delta_{\mathrm{n}}} \tilde{D}_i^T v - \delta_{\mathrm{n}} g$ in terms of the matrix $\tilde{D}_i := D - v g^T$, which may be computationally more convenient.

We have thus derived a rank one *downdate* for the blocks A_{i-1} and B_i under the tridiagonal block Cholesky factorization, depending on the vectors g and p defining the original dyadic downdate. With this, we can start over in Sect. 4.7.1 for A_i and B_{i+1}. The derived procedure eventually has to be carried out for all blocks $i-1, i, \ldots, m-1$ of (Eq. 18). As the initial vectors p and g forming the dyadic products to be subtracted affect the nodes $i-1$ and i only, the downdate for all nodes except node $i-1$ is carried out with $g = 0$, allowing for simplified computations.

We finally note that the same derivation can be carried out with minor modifications for the *update* $\overline{A}_{i-1} = A_{i-1} + p p^T$, $\overline{B}_i = B_i + g p^T$, $\overline{A}_i = A_i + g g^T$, which is required for the third and fourth kind of active set change. We omit the derivation for brevity, as it does not provide any additional insight.

5 Numerical results

In this section we consider an exemplary optimal control problem and solve it with a direct multiple shooting approach as briefly mentioned in Sect. 2.2.3. The nonlinear programs resulting from the direct multiple shooting discretization and parameterization are solved by a standard line search SQP method using an L-BFGS approximation of the Hessian of the Lagrangian. The QP subproblems arising in this SQP method are solved using a primal-dual parametric active set method, cf. [6] and [16] for more details. The KKT system arising in each iteration of this active set method is solved using the factorization of Sect. 3 and using the updates of Sect. 4. We report average per-iteration run times for several instances of this problem with differing size and compare to several different other algorithms, including the classical condensing preprocessing in conjunction with a dense null-space primal active set solver.

5.1 An exemplary problem

In order to exemplarily investigate the relative performance of the proposed factorization and update procedures, we consider a time-optimal control problem from automobile test driving due to [21]. The problem models a single-track vehicle evading an obstacle on a predefined track, aiming to complete this track in minimal time. For more details on this problem we refer to [21,33]. The problem's dimensions are $n^{\mathrm{x}} = 10$ states and $n^{\mathrm{q}} = 3 + n^{\mu}$ controls. The size of this problem after a direct multiple shooting discretization is influenced by the choice of the number $n^{\mu} \geq 1$ of available gears and the choice of the number m of direct multiple shooting nodes, resulting in a structured QP with $m(n^{\mathrm{x}} + n^{\mathrm{q}})$ unknowns, $(m-1)n^{\mathrm{x}}$ matching conditions, $n_i^{\mathrm{r}} = 3$ point constraints on the shooting nodes $1 \leq i \leq m-1$, and $n_m^{\mathrm{r}} = 4$ terminal point constraints.

5.2 Evaluated approaches at solving the KKT system

We examine 16 instances of this problem combined from the choices $n^{\mu} \in \{4, 8, 12, 16\}$ and $m \in \{21, 41, 81, 161\}$. Dimensions and sparsity of the Hessians H and the constraint Jacobians M, C of the resulting QPs can be found in Table 1. Examined approaches at the solution of the QP subproblems comprise

- Preprocessing the structured QP using the classical condensing algorithm, cf. [7,35], and solving the obtained smaller but densely populated QP using the dense null-space active set QP solver QPOPT [26], in version 1.0–10.
- Solving the KKT system of the structured QP using the multifrontal sparse symmetric indefinite code MA57, see [13], in the version available from the Harwell Subroutine Library.
- Solving the KKT system of the structured QP using the code UMFPACK, cf. [9], in version 5.1. Note that [32] presents KKT repair techniques for a sparse LU factorization computed by UMFPACK, which give significant speed-ups but obviously fill in after several iterations.
- Solving the KKT system of the structured QP using the banded LU factorization code DGBTRF available in LAPACK, see [2].
- Solving the KKT system of the structured QP using the factorization proposed in Sect. 3 of this paper.
- Solving the KKT system of the structured QP using the factorization proposed in Sect. 3 of this paper for the first active set QP iteration, and updating this

Table 1 Dimensions and sparsity of the Hessians H and the constraints M, C for the exemplary problem

m	n^{μ}	Hessian of the Lagrangian				Jacobian of the constraints			
		Size	Elements	Nonzeros		Rows	Elements	Nonzeros	
21	4	336	112,896	5,262	4.7%	264	88,704	1,906	2.1%
	8	420	176,400	7,878	4.5%	264	110,880	2,465	2.2%
	12	504	254,016	11,918	4.7%	264	133,056	3,024	2.3%
	16	588	345,744	16,254	4.7%	264	155,232	3,584	2.3%
41	4	656	430,336	10,382	2.4%	524	343,744	3,806	1.1%
	8	820	672,400	15,814	2.4%	524	429,680	4,924	1.1%
	12	984	968,256	22,718	2.3%	524	515,616	6,044	1.2%
	16	1,148	1,317,904	31,934	2.4%	524	601,552	7,166	1.2%
81	4	1,296	1,679,616	20,622	1.2%	1,044	1,353,024	7,607	0.6%
	8	1,620	2,624,400	30,950	1.2%	1,044	1,691,280	9,845	0.6%
	12	1,944	3,779,136	46,478	1.2%	1,044	2,029,536	12,087	0.6%
	16	2,268	5,143,824	60,478	1.2%	1,044	2,367,792	14,325	0.6%
161	4	2,576	6,635,776	41,048	0.6%	2,084	5,368,384	15,208	0.3%
	8	3,220	10,368,400	62,316	0.6%	2,084	6,710,480	19,688	0.3%
	12	3,864	14,930,496	92,208	0.6%	2,084	8,052,576	24,169	0.3%
	16	4,508	20,322,064	121,924	0.6%	2,084	9,394,672	28,648	0.3%

factorization using the techniques proposed in Sect. 4 of this paper in all subsequent active set iterations of the same SQP step.

All algorithms were run with their default settings. We did not make use of an accelerated BLAS library. The computing environment is a single core of an Intel Core i7 CPU at 2.67 GHz running a 64-bit Ubuntu Linux 9.10. Condensing and the proposed factorization and updates are implemented in ANSI C99, and compiled with applicable machine-specific optimization flags enabled.

5.3 Achieved run times

For each of the 16 problem instances and each of the 5 examined algorithmic approaches, the optimal control problem of Sect. 5.1 was solved to an acceptable KKT tolerance of 10^{-8} and the average run time per QP iteration was computed afterwards and is shown in Table 2. Only the QP code's share of the overall solution time was taken into account, i.e. run time spent for ODE system solution, sensitivity generation, or computation of the Hessian approximation was omitted. As can be seen clearly, the factorization proposed in Sect. 3 is the fastest of the examined algorithmic approaches by a wide margin for the largest examined problem instances. The dense active set solver QPOPT is faster for the smallest instances, although the condensing preprocessing has to be taken into account. Due to its $O(m^3)$ runtime complexity, condensing quickly falls behind for longer horizons. Both generic sparse solvers fall behind in performance as the size of the problem grows. This may be attributed to a

Table 2 Average run times in milliseconds per active set iteration (columns 4–9)

m	n^{μ}	Condensing (once)	QPOPT (dense)	MA57	UMFPACK	LAPACK DGBTRF	Sect. 3	With updates
21	4	4.49	0.247	1.366	1.396	1.734	0.737	0.457
	8	7.14	0.359	1.992	6.119	2.664	0.632	0.439
	12	10.3	0.365	2.550	6.129	4.116	0.765	0.426
	16	14.0	0.656	3.316	6.487	7.488	0.681	0.447
41	4	24.9	1.010	2.589	9.588	3.426	1.035	0.717
	8	42.6	1.392	3.429	11.58	5.304	1.130	0.750
	12	64.0	1.916	4.832	11.88	14.12	1.154	0.751
	16	90.6	2.763	6.299	13.09	14.15	1.234	0.797
81	4	158	4.337	4.995	19.21	6.204	1.922	1.261
	8	289	5.478	6.637	23.11	9.360	2.161	1.384
	12	451	9.101	9.278	23.97	14.31	2.703	1.402
	16	662	15.60	12.17	26.39	27.32	2.327	1.502
161	4	1,128	17.10	10.18	38.90	13.55	3.755	2.410
	8	2,224	26.92	13.88	46.35	22.67	4.115	2.601
	12	3,577	38.04	19.21	47.25	30.78	4.311	2.706
	16	5,322	55.30	25.39	53.79	59.06	4.586	2.907

Run time in milliseconds per SQP iteration for the condensing preprocessing (column 3), done for QPOPT only

lack of sparsity in the sensitivity matrices G computed by the direct multiple shooting approach. Collocation schemes would create larger, but sparser systems here. These better suit the generic sparse methods, while the proposed factorization could not easily exploit the additional in-block sparsity created by collocation. UMFPACK and DGBTRF in addition do not exploit the symmetry of the KKT system.

6 Summary and outlook

In this contribution, we have addressed the issue of solving a certain class of block structured quadratic programs using a a primal or primal-dual active set method. The examined class of QPs is widely encountered in optimal control and model predictive control, and its connections to SQP methods and the direct multiple shooting method have briefly been introduced.

A new derivation of a factorization for the block structured KKT system of the examined class of QPs, first published in [34], has been given taking the point of view of a nested saddle-point problem, and has been extended to exploit simple bounds on the unknowns. The factorization fully respects the block structure and completes in $O(mn^3)$ operations, where m is the number of nodes and n is the number of process states and control parameters per node. The factorization is as widely applicable as the popular dense null space method, given a QP with the required block structure.

We have seen that in active set methods, update procedures that recover the KKT system's factorization after a change of the active set are of vital importance. The main part of the contribution is concerned with the detailed first-time derivation of update procedures for the block structured factorization. We have shown for the first time how to recover this factorization after any of the four possible changes to the active set, namely adding or deleting a simple bound, and adding or deleting a point constraint. The presented update procedures are built on the well established updates for the dense null space method, but extend to two further interconnected factorization steps, namely a Schur complement and a block tridiagonal Cholesky factorization.

Using the proposed updates, an active set loop of a block structured primal active set method has to compute this factorization only once for the initially guessed active set, and may then compute cheap updates of this factorization after every iteration of the active set loop. The proposed algorithms hence allow for a block structured active set method with a setup run time complexity of $O(mn^3)$ and a per iteration run time complexity of $O(mn^2)$. This is in sharp contrast to classical condensing approaches, which require up to $O(m^3n^3)$ operations for the setup and $O(m^2n^2)$ operations per iteration.

We compared the computational demand of the described algorithms to several alternative approaches. These included classical condensing plus a dense active set method, as well as a number of more generic sparse and structured linear algebra codes for the solution of the structured KKT system. Computational evidence obtained for an exemplary optimal control problem suggests that the proposed factorization with update procedures is competitive in terms of per-iteration run times. The presented matrix updates give an additional speed-up that has becomes increasingly important as the control dimension n^{q} grows.

Our contribution is accompanied by a proof-of-concept MATLAB© implementation of all the proposed update procedures. It is intended to provide an additional means of exemplary verification of the derived updates. A realization in ANSI C as part of a block structured primal-dual active set QP code is envisioned and will be the topic of a forthcoming publication.

Acknowledgments The authors would like to thank the two anonymous referees whose extensive comments have helped to significantly improve this paper. In particular, the alternative presentation of the structured factorization in Sect. 3 was suggested by one of the referees. The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement no. FP7-ICT-2009-4 248940. This work has been supported by the Heidelberg Graduate School of Mathematical and Computational Methods for the Sciences (HGS MathComp) funded by Deutsche Forschungsgemeinschaft (DFG).

References

1. Albersmeyer, J., Bock, H.: Efficient sensitivity generation for large scale dynamic systems. Technical report, SPP 1253 Preprints, University of Erlangen (2009)
2. Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide, 3rd edn. Society for Industrial and Applied Mathematics, Philadelphia (1999)
3. Bartlett, R., Biegler, L.: QPSchur: a dual, active set, Schur complement method for large-scale and structured convex quadratic programming algorithm. Optim. Eng. **7**, 5–32 (2006)

4. Bartlett, R., Wächter, A., Biegler, L.: Active set vs. interior point strategies for model predictive control. In: Proceedings of the American Control Conference, Chicago, IL, pp. 4229–4233 (2000)
5. Benzi, M., Golub, G., Liesen, J.: Numerical solution of saddle-point problems. Acta Numerica **14**, 1–137 (2005)
6. Best, M.: An algorithm for the solution of the parametric quadratic programming problem. In: Fischer, H., Riedmüller, B., Schäffler, S. (eds.) Applied Mathematics and Parallel Computing—Festschrift for Klaus Ritter, Chap. 3, pp. 57–76. Physica-Verlag, Heidelberg (1996)
7. Bock, H., Plitt, K.: A multiple shooting algorithm for direct solution of optimal control problems. In: Proceedings of the 9th IFAC World Congress, pp. 242–247. Pergamon Press, Budapest (1984)
8. Bock, H., Diehl, M., Kostina, E., Schlöder, J. : Constrained optimal feedback control for DAE. In: Biegler, L., Ghattas, O., Heinkenschloss, M., Keyes, D., van Bloemen Waanders, B. (eds.) Real-Time PDE-Constrained Optimization, Chap. 1, pp. 3–24. SIAM, Philadelphia (2007)
9. Davis, T.: Algorithm 832: UMFPACK—an unsymmetric-pattern multifrontal method with a column pre-ordering strategy. ACM Trans. Math. Softw. **30**, 196–199 (2004)
10. Davis, T., Hager, W.: Multiple-rank modifications of a sparse Cholesky factorization. SIAM J. Matrix Anal. Appl. **22**(4), 997–1013 (2000)
11. Diehl, M., Bock, H., Schlöder, J., Findeisen, R., Nagy, Z., Allgöwer, F.: Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. J. Proc. Contr. **12**(4), 577–585 (2002)
12. Diehl, M., Kuehl, P., Bock, H., Schlöder, J.: Schnelle Algorithmen für die Zustands- und Parameterschätzung auf bewegten Horizonten. Automatisierungstechnik **54**(12), 602–613 (2006)
13. Duff, I.: MA57—a code for the solution of sparse symmetric definite and indefinite systems. ACM Trans. Math. Softw. **30**(2), 118–144 (2004)
14. Duff, I., Reid, J.: The multifrontal solution of indefinite sparse symmetric linear equations. ACM Trans. Math. Softw. **9**(3), 302–325 (1983)
15. Eldersveld, S., Saunders, M.: A block-LU update for large scale linear programming. SIAM J. Matrix Anal. Appl. **13**, 191–201 (1992)
16. Ferreau, H., Bock, H., Diehl, M.: An online active set strategy to overcome the limitations of explicit MPC. Int. J. Robust Nonlinear Control **18**(8), 816–830 (2008)
17. Fletcher, R.: Practical Methods of Optimization, 2nd edn. Wiley, Chichester (1987)
18. Fletcher, R.: Resolving degeneracy in quadratic programming. Numerical Analysis Report NA/135, University of Dundee, Dundee, Scotland (1991)
19. Fletcher, R.: Approximation Theory and Optimization. Dense Factors of Sparse Matrices, pp. 145–166. Tributes to M.J.D. Powell. Cambridge University Press (1997)
20. Fletcher, R.: Numerical Analysis 1997. Block Triangular Orderings and Factors for Sparse Matrices in LP, pp. 91–110. Pitman Research Notes in Mathematics, vol. 380. Longman, Harlow (1998)
21. Gerdts, M.: Solving mixed-integer optimal control problems by Branch&Bound: a case study from automobile test-driving with gear shift. Optimal Control Appl. Methods **26**, 1–18 (2005)
22. Gertz, E., Wright, S.: Object-oriented software for quadratic programming. ACM Trans. Math. Softw. **29**, 58–81 (2003)
23. Gill, P., Golub, G., Murray, W., Saunders, M.A.: Methods for modifying matrix factorizations. Math. Comput. **28**(126), 505–535 (1974)
24. Gill, P., Murray, W., Saunders, M., Wright, M.: Sparse matrix methods in optimization. SIAM J. Sci. Stat. Comput. **5**(3), 562–589 (1984)
25. Gill, P., Murray, W., Saunders, M., Wright, M.: A practical anti-cycling procedure for linearly constrained optimization. Math. Program. **45**(1–3), 437–474 (1989)
26. Gill, P., Murray, W., Saunders, M., Wright, M.: Inertia-controlling methods for general quadratic programming. SIAM Rev. **33**(1), 1–36 (1991)
27. Gill, P., Murray, W., Saunders, M.: User's Guide For QPOPT 1.0: A Fortran Package for Quadratic Programming (1995)
28. Golub, G., van Loan, C.: Matrix Computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)
29. Hall, J., McKinnon, K.: The simplest examples where the simplex method cycles and conditions where the EXPAND method fails to prevent cycling. Math. Program. Ser. A & B **100**(1), 133–150 (2004)
30. Han, S.: Superlinearly convergent variable-metric algorithms for general nonlinear programming problems. Math. Program. **11**, 263–282 (1976)

31. Haseltine, E., Rawlings, J.: Critical evaluation of extended Kalman filtering and moving-horizon estimation. Ind. Eng. Chem. Res. **44**, 2451–2460 (2005)
32. Huynh, H.: A large-scale quadratic programming solver based on block-LU updates of the KKT system. PhD thesis, Stanford University (2008)
33. Kirches, C., Sager, S., Bock, H., Schlöder, J.: Time-optimal control of automobile test drives with gear shifts. Optimal Control Appl. Methods **31**(2), 137–153 (2010)
34. Kirches, C., Bock, H., Schlöder, J., Sager, S.: Block structured quadratic programming for the direct multiple shooting method for optimal control. Optim. Methods Softw. **26**(2), 239–257 (2011)
35. Leineweber, D., Bauer, I., Schäfer, A., Bock, H., Schlöder, J.: An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization (Parts I and II). Comput. Chem. Eng. **27**, 157–174 (2003)
36. Nocedal, J., Wright, S.: Numerical Optimization, 2nd edn. Springer, Berlin, Heidelberg, New York (2006)
37. Powell, M.: Algorithms for nonlinear constraints that use Lagrangian functions. Math. Program. **14**(3), 224–248 (1978)
38. Powell, M.: ZQPCVX: a Fortran subroutine for convex quadratic programming. Technical report, Department of Applied Mathematics and Theoretical Physics, Cambridge University (1983)
39. Schmid, C., Biegler, L.: Quadratic programming methods for tailored reduced Hessian SQP. Comput. Chem. Eng. **18**(9), 817–832 (1994)
40. Steinbach, M.: Fast recursive SQP methods for large-scale optimal control problems. PhD thesis, Ruprecht-Karls-Universität Heidelberg (1995)
41. Vanderbei, R.: LOQO: an interior point code for quadratic programming. Optim. Methods Softw. **11**(1–4), 451–484 (1999)
42. Wächter, A., Biegler, L.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. **106**(1), 25–57 (2006)
43. Wirsching, L., Albersmeyer, J., Kühl, P., Diehl, M., Bock, H.: An adjoint-based numerical method for fast nonlinear model predictive control. In: Chung, M., Misra, P. (eds.) Proceedings of the 17th IFAC World Congress, Seoul, Korea, July 6–11, 2008. IFAC-PapersOnLine, vol. 17, pp. 1934–1939 (2008)
44. Wright, S.: Applying new optimization algorithms to model predictive control. In: Fifth International Conference on Chemical Process Control—CPC V, pp. 147–155. CACHE Publications (1997)