Math. Prog. Comp. (2015) 7:289–329
DOI 10.1007/s12532-015-0081-7

FULL LENGTH PAPER

A parallel quadratic programming method for dynamic optimization problems

Janick V. Frasch[1] · **Sebastian Sager**[1] · **Moritz Diehl**[2]

Received: 4 December 2013 / Accepted: 15 April 2015 / Published online: 1 May 2015

Abstract Quadratic programming problems (QPs) that arise from dynamic optimization problems typically exhibit a very particular structure. We address the ubiquitous case where these QPs are strictly convex and propose a dual Newton strategy that exploits the block-bandedness similarly to an interior-point method. Still, the proposed method features warmstarting capabilities of active-set methods. We give details for an efficient implementation, including tailored numerical linear algebra, step size computation, parallelization, and infeasibility handling. We prove convergence of the algorithm for the considered problem class. A numerical study based on the open-source implementation qpDUNES shows that the algorithm outperforms both well-established general purpose QP solvers as well as state-of-the-art tailored control QP solvers significantly on the considered benchmark problems.

Keywords Quadratic programming · Dual decomposition · Model predictive control · Structure exploitation · Parallel algorithms

Mathematics Subject Classification 90C20 · 49M15 · 49M29

✉ Janick V. Frasch
frasch@ovgu.de

Sebastian Sager
sager@ovgu.de

Moritz Diehl
moritz.diehl@imtek.uni-freiburg.de

[1] Institute of Mathematical Optimization, Otto von Guericke University Magdeburg, Magdeburg, Germany

[2] Department of Microsystems Engineering, University of Freiburg, Freiburg, Germany

1 Introduction

A large class of practical algorithms for the solution of dynamic optimization problems, as they appear for example in optimal control and dynamic parameter estimation, is based on sequential quadratic programming (SQP) [8,9,12,28]. Particularly for their online variants, model predictive control (MPC) and moving horizon estimation (MHE), fast solution of the arising quadratic programming (QP) subproblems is crucial [12,28,38], exploiting all problem-inherent structure.

In a generic form the QP subproblems arising in this class of methods can be formulated as follows:

$$\min_{x,u} \sum_{k=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^\top \begin{bmatrix} Q_k & S_k \\ S_k^\top & R_k \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} + \begin{bmatrix} q_k \\ r_k \end{bmatrix}^\top \begin{bmatrix} x_k \\ u_k \end{bmatrix} \right) + \frac{1}{2} x_N^\top Q_N x_N + q_N^\top x_N \quad \text{(1a)}$$

$$\text{s.t. } x_{k+1} = A_k x_k + B_k u_k + c_k \quad \forall k \in \mathcal{S}_N \quad \text{(1b)}$$

$$\underline{d}_k \leq D_k \begin{bmatrix} x_k \\ u_k \end{bmatrix} \leq \overline{d}_k \quad \forall k \in \mathcal{S}. \quad \text{(1c)}$$

Discretized state variables are denoted by $x_k \in \mathbb{R}^{n_x}$, while control variables are denoted by $u_k \in \mathbb{R}^{n_u}$ for each discretization stage $k \in \mathcal{S} := \{0, \ldots, N\}$ and $k \in \mathcal{S}_N$, respectively. In general we use subscripts to indicate exclusions from a set, e.g., $\mathcal{S}_{i,j} := \mathcal{S} \backslash \{i, j\}$. For notational simplicity, we hide time-constant parameters in the state vector. The stages are coupled over the discretized time horizon of length N by constraints (1b), while constraints (1c) arise from n_d discretized path constraints. The objective (1a) is often a quadratic approximation to the Lagrangian of the original control or estimation problem. We assume throughout this paper that (1a) is strictly convex, i.e.,

$$\begin{bmatrix} Q_k & S_k \\ S_k^\top & R_k \end{bmatrix} \succ 0 \quad \text{and} \quad Q_N \succ 0.$$

In a typical SQP scheme for dynamic optimization, QP (1) is solved repeatedly, each time with updated (relinearized) data, until a termination criterion is met. While the QP data changes, one often assumes that the active-set, i.e., those equations in (1c) that are satisfied tightly with equality, is similar from one iteration to the next, particularly in later iterations. In the large class of MPC and MHE algorithms that feature linear time-invariant dynamic systems, QP (1) is solved repeatedly for different initial conditions, i.e., only few entries of the vector data of the QP change.

QP (1) is very sparse for long time horizons N. The exploitation of these characteristic, predetermined sparsity patterns in combination with an exploitation of the problem similarity for algorithm warmstarting is one of the central challenges for efficiency of QP solvers in dynamic optimization. A large variety of attempts has been made to address these two requirements and we will briefly highlight several of these classes in the following.

The block-banded structure exhibited by (1) is typically exploited well by tailored interior-point (e.g., [13,32,37]) and first-order methods similar to Nesterov's fast-

gradient method [34] (e.g., [4,39]). However, a well-known drawback of these methods is their limited wamstarting capability to exploit the knowledge of similarity between the solutions of subsequently solved QPs.

Tailored active-set-methods like [14,15], on the other hand, can exploit this similarity for higher efficiency, but typically do not benefit from the problem-inherent sparsity as much as interior-point methods, as they, on the average, require significantly more iterations, each of which is dominated by the solution of a rather large linear system. One popular approach to overcome this bottleneck is to make use of a so-called condensing routine to express overparameterized state variables in terms of the control inputs and thus reduce the vector of optimization variables to only the control inputs [26,30]. In general, this condensing step needs to be performed at every sampling time, with a typically quadratic or cubic runtime complexity in the horizon length (depending on the chosen implementation, cf. [2,3,20]). Furthermore, the initial factorization of the dense QP Hessian is of similar runtime complexity in the horizon length, see [27]. The active-set strategy suggested in [26] can exploit the sparsity structure, but is most useful when only few active-set changes occur between the iterations, since each change requires a new sparse matrix factorization.

In this paper, we follow up on the idea of a different QP algorithm that was presented for linear MPC problems in [16]. This approach aims at combining the benefits in terms of structure exploitation of interior-point methods with the wamstarting capabilities of active-set methods, and comes at only a linear runtime complexity in the horizon length. Based on ideas from [10] and [31], the stage coupling constraints (1b) induced by the MPC structure are dualized and the resulting QP is solved in a two level approach, using a non-smooth/semismooth Newton method in the multipliers of the stage coupling constraints on the higher level, and a primal active-set method in the decoupled parametric QPs of each stage on the lower level. Note that in contrast to classical active-set methods, this approach permits several active-set changes at the cost of one block-banded matrix factorization. We refer to this procedure as a dual Newton strategy. More details on the method are given in Sect. 2.

While in [16] only a limited prototype Matlab implementation was described, this work extends the algorithm to a more general problem class and gives details on an efficient implementation of the method. We further provide a theoretical foundation to the algorithm and prove convergence. We discuss parallelization aspects of the method and present a novel algorithm for the solution of the structured Newton system, that results in an parallel runtime complexity of the dual Newton strategy of $\mathcal{O}(\log N)$ per iteration. Most importantly, we present qpDUNES, an open-source, plain C implementation of the *DU*al *NE*wton *S*tategy. This software comes with interfaces for C/C++ and Matlab. We compare runtimes of qpDUNES to state of the art structure-exploiting QP solvers for dynamic optimization problems based on three challenging benchmark control problems.

This paper bases in parts on the conference proceedings paper [19]. In contrast to [19], however, we provide significant additional details in this paper, such as in-depth algorithmic and theoretical investigations, implementation details, and benchmark results from linear MPC.

2 Method description

For clarity of presentation we group the optimization variables of dynamic optimization QP problems, the system states $x_k \in \mathbb{R}^{n_x}$ and the control inputs $u_k \in \mathbb{R}^{n_u}$, into the stage variables $z_k = \begin{bmatrix} x_k^\top & u_k^\top \end{bmatrix}^\top \in \mathbb{R}^{n_z}$ for each stage $k \in \mathcal{S}_N$, and $z_N = \begin{bmatrix} x_N & 0 \end{bmatrix}^\top \in \mathbb{R}^{n_z}$ for the terminal stage. Here, we define $n_z := n_x + n_u$. Problem (1) then reduces to

$$\min_z \sum_{k=0}^{N} \left(\frac{1}{2} z_k^\top H_k z_k + g_k^\top z_k \right) \tag{P1}$$

$$\text{s.t. } E_{k+1} z_{k+1} = C_k z_k + c_k \quad \forall k \in \mathcal{S}_N \tag{P2}$$

$$\underline{d}_k \leq D_k z_k \leq \overline{d}_k \quad \forall k \in \mathcal{S}, \tag{P3}$$

which is the problem that we consider here. The cost function on each stage consists of a positive definite second-order term $0 \prec H_k \in \mathbb{R}^{n_z \times n_z}$ and a first-order term $g_k \in \mathbb{R}^{n_z}$ for each $k \in \mathcal{S}$. Note that therefore primal solutions (if existent) are unique. Two subsequent stages $k \in \mathcal{S}$ and $k+1 \in \mathcal{S}$ are coupled by first-order terms $C_k, E_{k+1} \in \mathbb{R}^{n_x \times n_z}$ and a constant term c_k. We assume that all C_k have full row rank, i.e., $\text{rank}(C_k) = n_x, \ \forall k \in \mathcal{S}_N$, and that all E_k have the special structure $E_k = \begin{bmatrix} \mathrm{I} & 0 \end{bmatrix}$, where $\mathrm{I} \in \mathbb{R}^{n_x \times n_x}$ is an identity matrix and 0 is a zero matrix of appropriate dimensions. Vectors $\underline{d}_k, \overline{d}_k \in \mathbb{R}^{n_d}$, and a matrix $D_k \in \mathbb{R}^{n_d \times n_z}$ of full row rank denote affine stage constraints.

Unless stated otherwise, we assume in the following that a feasible solution $z^* := \begin{bmatrix} z_0^{*\top}, \ldots, z_N^{*\top} \end{bmatrix}^\top$ of (P) exists that fulfills the linear independence constraint qualification (LICQ). Section 5 discusses the consequences resulting from infeasibility of (P) and how it can be detected.

2.1 Dual decomposition

We decouple the QP stages by dualizing constraints (P2). By introducing the vector of Lagrange multipliers

$$\lambda := \begin{bmatrix} \lambda_1^\top & \lambda_2^\top & \cdots & \lambda_N^\top \end{bmatrix}^\top \in \mathbb{R}^{N n_x}, \tag{3}$$

we can express (P1) and (P2) by the partial Lagrangian function

$$\begin{aligned} \mathcal{L}(z, \lambda) &:= \sum_{k=0}^{N} \left(\frac{1}{2} z_k^\top H_k z_k + g_k^\top z_k \right) + \sum_{k=0}^{N-1} \lambda_{k+1}^\top \left(-E_{k+1} z_{k+1} + C_k z_k + c_k \right) \\ &= \sum_{k=0}^{N} \left(\frac{1}{2} z_k^\top H_k z_k + g_k^\top z_k + \begin{bmatrix} \lambda_k \\ \lambda_{k+1} \end{bmatrix}^\top \begin{bmatrix} -E_k \\ C_k \end{bmatrix} z_k + \lambda_{k+1}^\top c_k \right), \end{aligned}$$

where we define zero matrices $E_0 = C_N = 0 \in \mathbb{R}^{n_x \times n_z}$ and redundant multipliers $\lambda_0 = \lambda_{N+1} := 0 \in \mathbb{R}^{n_x}$ only for notational convenience in this context (note that they are not among the optimization variables of the dual problem defined below).

By Lagrangian duality, the solution of (P) can be computed as

$$\max_{\lambda} \ \min_{z} \ \sum_{k=0}^{N} \left(\frac{1}{2} z_k^\top H_k z_k + g_k^\top z_k + \begin{bmatrix} \lambda_k \\ \lambda_{k+1} \end{bmatrix}^\top \begin{bmatrix} -E_k \\ C_k \end{bmatrix} z_k + \lambda_{k+1}^\top c_k \right)$$
$$\text{s.t. } \underline{d}_k \leq D_k z_k \leq \overline{d}_k \quad \forall k = 0, \ldots, N.$$

Because this Problem is separable in the stage variables z_k, minimization and summation can be interchanged, and a solution to (P) can be obtained by solving

$$\max_{\lambda} f^*(\lambda) := \max_{\lambda} \sum_{k=0}^{N} f_k^*(\lambda), \tag{D}$$

where

$$f_k^*(\lambda) := \min_{z_k} \frac{1}{2} z_k^\top H_k z_k + \left(g_k^\top + \begin{bmatrix} \lambda_k \\ \lambda_{k+1} \end{bmatrix}^\top \begin{bmatrix} -E_k \\ C_k \end{bmatrix} \right) z_k + \lambda_{k+1}^\top c_k$$
$$\text{s.t. } \underline{d}_k \leq D_k z_k \leq \overline{d}_k. \tag{QP$_k$}$$

We refer to (QP$_k$) as the kth stage QP. Note that each (QP$_k$) depends on at most two block components of the vector of dual optimization variables λ defined in (3).

Remark 1 Since λ only enters in the objective of each (QP$_k$), feasibility of (QP$_k$), and thus existence of $f_k^*(\lambda)$ is independent of the choice of $\lambda \in \mathbb{R}^{Nn_x}$. In particular, since the constraints of (QP$_k$) are a subset of (P3), feasibility of (P) implies feasibility of (QP$_k$).

Remark 2 Each $f_k^*(\lambda)$ implicitly defines a vector $z_k^*(\lambda)$, the solution of (QP$_k$).

2.2 Characterization of the dual function

It was shown in [16] (based on results from [5,17,45]) that $f^*(\lambda)$ is concave, piecewise quadratic, and once continuously differentiable. We establish the relevant findings in the following.

Definition 1 For a stage $k \in \mathcal{S}$, the optimal active-set at λ is given by

$$\mathcal{A}_k^*(z_k^*(\lambda)) := \left\{ 1 \leq i \leq n_d \mid D_k^i \cdot z_k^*(\lambda) = \underline{d}_k^i \vee D_k^i \cdot z_k^*(\lambda) = \overline{d}_k^i \right\},$$

i.e., the set of row indices of the constraints of (QP$_k$) that are satisfied as an equality. Here, D_k^i refers to the i^{th} row of D_k, and $\underline{d}_k^i$ and $\overline{d}_k^i$ refer to the i^{th} entry of the vector.

Definition 1 naturally extends to a definition of the active-set in the full space of primal variables by $\mathcal{A}^*(z^*(\lambda)) := \mathcal{A}_0^*(z_0^*(\lambda)) \times \cdots \times \mathcal{A}_N^*(z_N^*(\lambda))$. The finite number of disjoint active-sets further induces a subdivision of the dual λ space.

Definition 2 Each active-set defines a region $A \subseteq \mathbb{R}^{Nn_x}$ in the dual λ space. For a representative $\lambda^{(j)} \in \mathbb{R}^{Nn_x}$ we have

$$A^{(j)} := \left\{ \lambda \in \mathbb{R}^{Nn_x} \mid \mathcal{A}^*(z^*(\lambda)) = \mathcal{A}^*(z^*(\lambda^{(j)})) \right\}.$$

By choosing representatives of pairwise distinct regions we can define an arbitrary, but fixed order that allows us to uniquely identify each region $A^{(j)}$.

The name region is anticipatory, but it will become clear from Corollary 2 that the sets $A^{(j)}$ indeed are connected. The number of regions n_r clearly is finite, as there is only a finite number of distinct active-sets. From Remark 1 we can conclude that each $\lambda \in \mathbb{R}^{Nn_x}$ is contained in a region $A^{(j)}$, and thus

$$\bigcup_{1 \le j \le n_r} A^{(j)} = \mathbb{R}^{Nn_x}.$$

Remark 3 Two distinct regions $A^{(j_1)}$ and $A^{(j_2)}$ do not need to be disjoint. Values of λ that lead to weakly active stage constraints at $z^*(\lambda)$ are contained in two or more regions. These values of λ form the seams of the regions.

Next, we substantiate Remark 2 by characterizing the nature of the dependency of z_k^* on the dual variables λ in the stage problems (QP_k).

Lemma 1 *Let (QP_k) be feasible. Then, the optimal solution of (QP_k), $z_k^*(\lambda)$, is a piecewise affine and continuous function in λ. In particular, the dependency is affine on each region $A^{(j)}$, $1 \le j \le n_r$.*

Proof (*cf.* [16, Lemma 2]; [45]) For stage Lagrange multipliers $\mu_k \in \mathbb{R}^{2n_d}$ the solution of (P) is given by (see, e.g., [18])

$$\begin{bmatrix} H_k & -\underline{D}_k^{*\top} & \overline{D}_k^{*\top} \\ -\underline{D}_k^* & & \\ \overline{D}_k^* & & \end{bmatrix} \begin{bmatrix} z_k^* \\ \mu_k^* \end{bmatrix} = \begin{bmatrix} -\left(g_k + \begin{bmatrix} -E_k \\ C_k \end{bmatrix}^\top \begin{bmatrix} \lambda_k \\ \lambda_{k+1} \end{bmatrix} \right) \\ \underline{d}_k^* \\ -\overline{d}_k^* \end{bmatrix}, \tag{4}$$

where $\underline{D}_k^*$, $\underline{d}_k^*$ and $\overline{D}_k^*$, $\overline{d}_k^*$ consist of the rows of D_k, $\underline{d}_k$, and $\overline{d}_k$ that correspond to the constraints that are active (i.e., fulfilled with equality) at the lower or, respectively, the upper bound in the solution z_k^*, and μ_k^* is the vector of consistent dimension that contains the corresponding multipliers. The remaining stage multiplier entries are 0 in the solution of (P). As λ enters affinely only on the right-hand side, it is clear that for identical active-sets it holds that z_k^* depends affinely on λ. Continuity has been shown in [17]. □

Corollary 2 *Each region $A^{(j)}$, $1 \leq j \leq n_r$, of the dual space is convex and polyhedral.*

Proof Each stage problem (QP_k) is constrained by affine constraints. For a given representative $\lambda^{(j)}$ the set $\mathcal{F}_k := \{z_k \in \mathbb{R}^{n_z} \mid \mathcal{A}_k(z_k) = \mathcal{A}_k^*(z_k^*(\lambda^{(j)}))\}$ is therefore convex and polyhedral. From Lemma 1 we have that z_k^* is affine in λ for a certain (fixed) active-set. A region $A^{(j)}$ is therefore the intersection of $N+1$ (i.e., a finite number) preimages of convex sets, and therefore convex. □

Lemma 1 is the basis for the following exhaustive characterization of the dual function $f^*(\lambda)$, which we take from [16] without proof.

Lemma 3 ([16, Lemma 3]) *If all stage QPs (QP_k) are feasible, then the dual function $f^*(\lambda)$ exists and is described by a concave, continuously differentiable, and piecewise quadratic spline in λ space.*

Remark 4 In particular $f^*(\lambda)$ is quadratic on each region $A^{(j)}$.

2.3 Solution by a (non-smooth) Newton method

Because (D) is an unconstrained problem and $f^*(\lambda)$ is a piecewise-quadratic spline we employ a non-smooth Newton method, as originally proposed in [31], and also used in [16]. The dual iterates are updated via the iteration

$$\lambda^{i+1} := \lambda^i + \alpha \Delta\lambda \tag{5}$$

(the Newton iterates λ^i are not to be confused with the region representatives $\lambda^{(j)}$ from Sect. 2.2) for an initial guess λ^0 and a suitably chosen step size α until $f^*(\lambda^i)$ is stationary. The step direction $\Delta\lambda$ is computed from

$$\mathcal{M}(\lambda^i)\, \Delta\lambda = \mathcal{G}(\lambda^i), \tag{6}$$

where $\mathcal{M}(\lambda^i) := -\frac{\partial^2 f^*}{\partial \lambda^2}(\lambda^i)$ and $\mathcal{G}(\lambda^i) := \frac{\partial f^*}{\partial \lambda}(\lambda^i)$. By Remarks 3 and 4, $\mathcal{M}(\lambda)$ is unique everywhere but on the seams of $f^*(\lambda)$ (a null set), so almost everywhere. On the seams we assume that pointwise an arbitrary, but fixed second derivative from the finite number of possible choices is used, thus ensuring general well-definedness of $\mathcal{M}(\lambda)$. As we will see later on, the specific choice of $\mathcal{M}(\lambda)$ on the seams is not crucial for the convergence of the method. In Sect. 3.3 we show that the choice of $\mathcal{M}(\lambda)$ on the seams is done implicitly and automatically by the degeneracy handling mechanism of the stage subproblem solver. We give more details on how the Newton system is set up in Sects. 3.2 and 3.3.

Clearly, stationarity of $f^*(\lambda)$ is equivalent to optimality of (D). Observe that by definition of (D), $z^*(\lambda^i)$ (i.e., the canonical concatenation of all $z_k^*(\lambda^i)$) is always optimal and (P3) are always fulfilled in the spirit of an active-set method. Lemma 4 will show that feasibility of (P2) is identical with stationarity of $f^*(\lambda)$.

The complete QP solution method is given in Algorithm 1, where we denote the Lagrange multipliers of the stage constraints (P3) by $\mu_k \in \mathbb{R}^{2n_d}$ for each stage $k \in \mathcal{S}$.

Algorithm 1: Dual Newton strategy

Input: Initial guess λ^0, termination criteria n_{maxIt}, ϵ_λ
Output: Optimal solution (z^*, λ^*, μ^*)
1 **for** $i = 0 : (n_{\text{maxIt}} - 1)$ **do**
2 Solve all $\text{QP}_k(\lambda^i)$ to obtain $[z_k^*(\lambda^i), \mu_k^*(\lambda^i)]$
3 Set up gradient $\mathcal{G}(\lambda^i)$
4 **if** $\left\|\mathcal{G}(\lambda^i)\right\| \leq \epsilon_\lambda$ **then**
5 **return** $[z_k^*(\lambda^i), \lambda^i, \mu_k^*(\lambda^i)]$
6 Set up Newton matrix $\mathcal{M}(\lambda^i)$
7 Solve Newton system (6)
8 Compute appropriate step size α
9 Update current iterate $\lambda^{i+1} := \lambda^i + \alpha \Delta\lambda$

Note that the parametric solution of the stage problems (QP_k) for the current iterate λ^i in Step 2, and, as we will see in Sect. 3, also the setup of $\mathcal{G}(\lambda^i)$ and $\mathcal{M}(\lambda^i)$ in Steps 3 and 6 permit an independent, concurrent execution on all stages (see also Sect. 6).

We discuss details of the respective steps in Sect. 3 and give a convergence proof for the algorithm in Sect. 4.

2.4 Characterization of the dual Newton iterations

A full QP solution by Algorithm 1 could be visualized as in Fig. 1. Each cell corresponds to a region $A^{(j)}$ in λ-space, for which the primal active-set is constant. Starting from an initial guess λ^0, a Newton direction is computed from Eq. (6), leading to λ^1_{FS}. Using a globalization strategy (see Sect. 3.6), a suitable step rescaling is found, leading to $\lambda^1_{\tilde{\alpha}}$. In contrast to classical active-set methods, multiple active-set changes are possible in one iteration. For future reference we also indicate a minimum guaranteed step, which (in the regular case) leads at least to $\lambda^1_{\alpha_{\min}}$ in the neighboring region. In the second iteration of this illustration, λ^2_{FS} already provides sufficient progress, thus no globalization needs to be applied. In the following iteration λ^* is found. We prove in Lemma 8 that a one-step terminal convergence is guaranteed, once the correct region is identified.

3 Algorithmic details of the dual Newton strategy

The dynamic optimization origin induces a specific structure in Problem (D), that we strive to exploit as explained in the following section.

3.1 Solution of decoupled parametric stage QPs

On each stage $k \in \mathcal{S}$ we have to repeatedly solve a QP of size (n_z, n_d) that only changes in the first-order term (and the negligible constant term) with the current guess of λ. We rephrase problem (QP_k) as

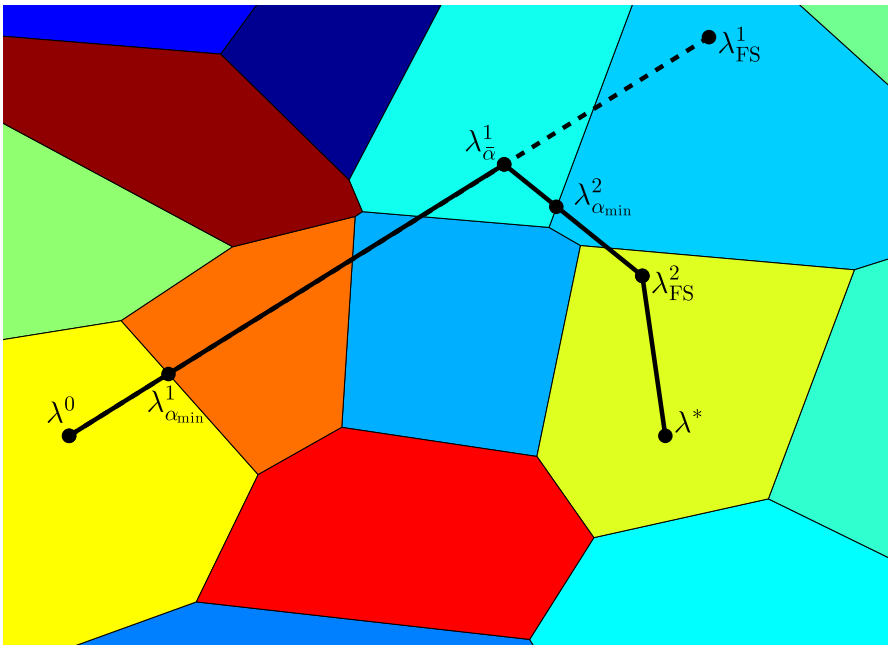


Fig. 1 Steps of the dual Newton strategy in the λ-space

$$\begin{aligned} \min_{z_k} \ & \frac{1}{2} z_k^\top H_k z_k + m_k(\lambda)^\top z_k + p_k(\lambda) \\ \text{s.t.} \ & \underline{d}_k \leq D_k z_k \ \leq \ \overline{d}_k, \end{aligned}$$

with

$$m_k(\lambda) := g_k - E_k^\top \lambda_k + C_k^\top \lambda_{k+1}\,, \tag{7}$$

$$p_k(\lambda) := c_k^\top \lambda_{k+1}\,,\,. \tag{8}$$

In general, H_k and D_k are dense. Such QPs can be solved efficiently (see [7,14]), for example by employing the online active set strategy, which is implemented in the open-source QP solver qpOASES [15].

In the special, yet practically relevant case where H_k is a diagonal matrix and D_k is an identity matrix (i.e., only bounds on states and controls exist) the optimal solution z_k^* can conveniently be computed by component-wise "clipping" of the unconstrained solution as it was presented in [16]:

$$z_k^* = \max(\underline{d}_k, \min(H_k^{-1} m_k, \overline{d}_k)). \tag{9}$$

3.2 Structure of the Newton system

The right-hand-side vector $\mathcal{G} : \mathbb{R}^{Nn_x} \to \mathbb{R}^{Nn_x}$ of the Newton system (6) is easily seen to only depend on two neighboring stages in each block λ_k. It holds that

$$\mathcal{G}(\lambda) \;=\; \left(\frac{\partial f^*}{\partial \lambda}(\lambda)\right)^\top \;=\; \begin{bmatrix} \frac{\partial f_0^*}{\partial \lambda_1}^\top + \frac{\partial f_1^*}{\partial \lambda_1}^\top \\ \frac{\partial f_1^*}{\partial \lambda_2}^\top + \frac{\partial f_2^*}{\partial \lambda_2}^\top \\ \vdots \\ \frac{\partial f_{N-1}^*}{\partial \lambda_N}^\top + \frac{\partial f_N^*}{\partial \lambda_N}^\top \end{bmatrix}(\lambda). \tag{10}$$

The left-hand side Newton matrix $\mathcal{M} : \mathbb{R}^{Nn_x} \to \mathbb{R}^{Nn_x \times Nn_x}$ has a block tri-diagonal structure, because only neighboring multipliers λ_k, λ_{k+1} can have a joint contribution to f^*. At a fixed λ

$$\mathcal{M}(\lambda) = -\frac{\partial^2 f^*}{\partial \lambda^2}(\lambda) = \begin{bmatrix} W_1 & U_1 & & \\ U_1^\top & W_2 & \ddots & \\ & \ddots & \ddots & U_{N-1} \\ & & U_{N-1}^\top & W_N \end{bmatrix}(\lambda), \tag{11}$$

where the diagonal and off-diagonal block components are given by

$$W_k(\lambda) := -\frac{\partial^2 f^*}{\partial \lambda_k^2}(\lambda) \qquad \text{and} \qquad U_k(\lambda) := -\frac{\partial^2 f^*}{\partial \lambda_k \lambda_{k+1}}(\lambda). \tag{12}$$

3.3 Gradient and Hessian computation

Lemma 4 (cf. [6, App.C]) *Let all (QP_k) be feasible. Then the derivative of f_k^* with respect to λ exists and its nonzero entries are given by*

$$\begin{bmatrix} \frac{\partial f_k^*}{\partial \lambda_k} & \frac{\partial f_k^*}{\partial \lambda_{k+1}} \end{bmatrix} = z_k^{*\top} \begin{bmatrix} -E_k \\ C_k \end{bmatrix}^\top + \begin{bmatrix} 0 \\ c_k \end{bmatrix}^\top. \tag{13}$$

Proof The derivative $\frac{\partial f_k^*}{\partial \lambda}$ exists by Lemma 3. From (7) and (8) we observe that only λ_k and λ_{k+1} enter in (QP_k), and therefore the only nonzero blocks of $\frac{\partial f_k^*}{\partial \lambda}$ are given by $\frac{\partial f_k^*}{\partial \breve{\lambda}}$, where we use $\breve{\lambda} := \begin{bmatrix} \lambda_k & \lambda_{k+1} \end{bmatrix}$ to denote the projection of λ onto the respective relevant entries. We derive a closed form for these values by regarding the stage QP Lagrangian

$$\mathcal{L}_k(z_k, \mu_k; \lambda) := \frac{1}{2} z_k^\top H_k z_k + m_k(\lambda)^\top z_k + p_k(\lambda) + \mu_k^\top \begin{bmatrix} D_k z_k - \underline{d}_k \\ \overline{d}_k - D_k z_k \end{bmatrix}.$$

Since $H_k \succ 0$ and (QP_k) is feasible by assumption, it holds that (QP_k) has a (finite) optimal primal and dual solution (z_k^*, μ_k^*), and, by Danskin's Theorem [11], we can interchange optimization and derivation in the sense that

$$\frac{\partial f_k^*}{\partial \check{\lambda}} = \frac{\partial}{\partial \check{\lambda}} \mathcal{L}_k(z_k^*, \mu_k^*; \lambda)$$

holds. We then have

$$\begin{aligned}
\frac{\partial f_k^*}{\partial \check{\lambda}} &= \frac{\partial \mathcal{L}_k(z_k^*, \mu_k^*; \lambda)}{\partial \check{\lambda}} + \frac{\partial \mathcal{L}_k(z_k^*, \mu_k^*; \lambda)}{\partial z_k^*} \times \frac{\partial z_k^*}{\partial \check{\lambda}} + \frac{\partial \mathcal{L}_k(z_k^*, \mu_k^*; \lambda)}{\partial \mu_k^*} \times \frac{\partial \mu_k^*}{\partial \check{\lambda}} \\
&= \frac{\partial \mathcal{L}_k(z_k^*, \mu_k^*; \lambda)}{\partial \check{\lambda}} + \frac{\partial \mathcal{L}_k(z_k^*, \mu_k^*; \lambda)}{\partial z_k^*} \times \frac{\partial z_k^*}{\partial \check{\lambda}} + \begin{bmatrix} D_k z_k - \underline{d}_k \\ \overline{d}_k - D_k z_k \end{bmatrix}^\top \times \frac{\partial \mu_k^*}{\partial \check{\lambda}} \\
&= \left(z_k^{*\top} \begin{bmatrix} -E_k \\ C_k \end{bmatrix}^\top + \begin{bmatrix} 0 \\ c_k \end{bmatrix}^\top \right) + 0 \times \frac{\partial z_k^*}{\partial \check{\lambda}} + 0,
\end{aligned}$$

where the second and the third term vanish because of stationarity and the complementarity requirement of the optimal stage solution (z_k^*, μ_k^*). □

Remark 5 We can see from Lemma 4 that $\| \mathcal{G}(\lambda) \|$ is indeed a measure for both stationarity of $f^*(\lambda)$ and infeasibility of (P2), as claimed in Sect. 2.3.

The second derivative of f^* can be computed as follows:

Lemma 5 *Let $Z_k^* \in \mathbb{R}^{n_z \times (n_z - n_{\text{act}})}$, $k \in \mathcal{S}$ (where n_{act} denotes the number of active constraints, which depends on $z_k^*(\lambda)$) be a basis matrix for the nullspace of the matrix of active constraint rows of (QP_k), D_k^*, which depends by z_k^* on λ. $P_k^* := Z_k^* (Z_k^{*\top} H_k Z_k^*)^{-1} Z_k^{*\top} \in \mathbb{R}^{n_z \times n_z}$ denote the elimination matrix for this nullspace. Then $\mathcal{M}(\lambda)$ is given by*

$$\mathcal{M}(\lambda) = \mathcal{C} \mathcal{P} \mathcal{C}^\top,$$

where $\mathcal{P} := \operatorname{diag}(\begin{bmatrix} P_0^* & P_1^* & \cdots & P_N^* \end{bmatrix})$ and

$$\mathcal{C} := \begin{bmatrix} C_0 & -E_1 & & & \\ & C_1 & -E_2 & & \\ & & \ddots & \ddots & \\ & & & C_{N-1} & -E_N \end{bmatrix} \in \mathbb{R}^{N n_x \times (N+1) n_z}.$$

Proof We compute the Hessian blocks in (11) explicitly. Differentiating (13) once more with respect to λ, we obtain

$$\frac{\partial^2 f^*}{\partial \lambda_k \lambda_{k+1}} = \frac{\partial}{\partial \lambda_k} \left(\frac{\partial f_k^*}{\partial \lambda_{k+1}} + \frac{\partial f_{k+1}^*}{\partial \lambda_{k+1}} \right) = \frac{\partial z_k^*}{\partial \lambda_k} C_k^\top - \underbrace{\frac{\partial z_{k+1}^*}{\partial \lambda_k}}_{=0} E_{k+1}^\top$$

and

$$\frac{\partial^2 f^*}{\partial \lambda_k \lambda_k} = \frac{\partial}{\partial \lambda_k} \left(\frac{\partial f_{k-1}^*}{\partial \lambda_k} + \frac{\partial f_k^*}{\partial \lambda_k} \right) = \frac{\partial z_{k-1}^*}{\partial \lambda_k} C_{k-1}^\top - \frac{\partial z_k^*}{\partial \lambda_k} E_k^\top.$$

Within a fixed active-set the optimal solution of (QP_k) at λ is given by (see, e.g., [35])

$$z_k^*(\lambda) = -P_k^{*-1} m_k(\lambda) = -P_k^{*-1} \left(g_k + E_k^\top \lambda_k + C_k^\top \lambda_{k+1} \right).$$

Accordingly, the Hessian blocks (cf. Eq. (12)) are computed as

$$U_k = -E_k P_k^* C_k^\top \tag{14}$$

and

$$W_k = C_{k-1} P_{k-1}^* C_{k-1}^\top + E_k P_k^* E_k^\top, \tag{15}$$

which concludes the proof. □

Remark 6 From Lemma 5 we can see in detail why $\mathcal{M}(\lambda)$ is naturally well-defined almost everywhere. Only cases where λ leads to weakly active constraints in some (QP_k), we observe ambiguity of Z_k^* and need to resort to a point-wise arbitrary but fixed specification of Z_k^* for well-definedness of $\mathcal{M}(\lambda)$. In practice, this ambiguity is naturally resolved by the degeneracy handling mechanism of the stage QP solver; it will in particular become clear from Theorem 9 that the convergence of the algorithm is not affected by the specific choice of Z_k^*.

Remark 7 Under the assumption of LICQ, a nullspace basis matrix Z_k^* can be constructed as $Z_k^* := \begin{bmatrix} -B^{-1}N \\ I \end{bmatrix}$ by partitioning the matrix of active constraint rows in the solution $z_k^*(\lambda)$, $D_k^* =: \begin{bmatrix} B & N \end{bmatrix} \in \mathbb{R}^{n_{\mathrm{act}} \times n_z}$, into an invertible matrix $B \in \mathbb{R}^{n_{\mathrm{act}} \times n_{\mathrm{act}}}$ (without loss of generality the first n_{act} columns by reordering) and a $n_{\mathrm{act}} \times (n_z - n_{\mathrm{act}})$ matrix N (see [23,35] for details).

Remark 8 It is important to note that P_k^* can be obtained relatively cheaply from a null-space QP solver like qpOASES [15] that directly provides Z_k^* and a Cholesky factor R for $R^\top R = Z_k^{*\top} H_k Z_k^*$; see [14]. For the special case of diagonal Hessian matrices $H_k = \mathrm{diag}(h_k^1, \ldots, h_k^{n_z})$ and simple bounds, the projection P_k^* is simply a diagonal matrix with either $1/h_k^i$ or 0 entries depending on whether the corresponding variable bound is inactive or active. The calculation of the Hessian blocks can then be accelerated even further using diadic products as proposed in [16].

Remark 9 From the construction of $\mathcal{M}$ in Lemma 5 we can see that the dual Newton strategy is, in principle, also capable of dealing with certain indefinite problems of the form (P), as long as the reduced Hessian blocks $Z_k^{*\top} H_k Z_k^*$ remain positive (semi-)definite for all $k \in \mathcal{S}$. This could, for example, be the case when equality stage constraints eliminate the negative definite directions from the stage Hessians. Basic requirement for this work is of course the stage QP solvers' support of such indefinite QPs.

3.4 Solution of the Newton system and regularization

By Lemma 3, $\mathcal{M}(\lambda)$ is positive semidefinite, because $f^*(\lambda)$ is concave. The block-tridiagonal structure of the $\mathcal{M}(\lambda)$, cf. Eq. (11), can be exploited for the efficient solution of the Newton system (6). Observing that a lower triangular factor L of $\mathcal{M}(\lambda) = LL^\top$ possesses the same structural zero-blocks below the diagonal, we suggest to employ a banded Cholesky decomposition. This factorization differs from a regular Cholesky decomposition (see, e.g., [35]) by skipping all redundant blocks left and below the subdiagonal block $U_k^\top$ of each block column k, thus reducing the computational complexity from $O(N^3 n_x^3)$ to $O(N n_x^3)$ floating point operations (FLOPs).

In the case of jointly redundant active constraints in several (QP_k) via the stage coupling constraints (P2), $\mathcal{M}(\lambda)$ may become rank-deficient [31] and the Newton system (6) ill-posed. We propose to overcome this by applying regularization. In our dual Newton software package qpDUNES both a Levenberg-Marquadt-type regularization and an "on-the-fly" regularization are available. On detection of singularity during the initial banded Cholesky factorization, the Levenberg-Marquadt approach replaces $\mathcal{M}(\lambda^i)$ in (6) by

$$\tilde{\mathcal{M}}(\lambda^i) := \mathcal{M}(\lambda^i) + \gamma \cdot I \tag{16}$$

where $\gamma \in \mathbb{R}^+$ is a (small) constant regularization parameter. The "on-the-fly" regularization changes those diagonal elements for which the crucial division step in the Cholesky decomposition cannot be performed due to singularity (similarly to the modified Cholesky factorization described in [35], which is based on [22]). We note that the "on-the-fly" regularization ensures positive definiteness of the resulting $\tilde{\mathcal{M}}(\lambda^i)$ (because it has a unique Cholesky decomposition) and avoids the need of restarting the factorization, but may be numerically less stable.

the latter only regularizes those diagonal elements for which the crucial division step in the Cholesky decomposition cannot be performed due to singularity (similarly to the modified Cholesky factorization described in [35], which is based on [22]). While the former one uses

3.5 A reverse Cholesky factorization for improved stability

In the context of MPC, one expects rather many active constraints in the beginning of the control horizon and few to none towards the end of the horizon in many applications. This may, for example, be a result of the chosen objective being of tracking nature or of the rejection of a perturbation of the controlled process which enters at the first stage. We aim to exploit this knowledge by applying a Cholesky factorization to $\mathcal{M}$ (we omit the λ-dependency in this section for notational convenience) in reverse order, i.e., starting from the last row/column, as detailed in Algorithm 2. Instead of a factorization $\mathcal{M} = LL^\top$, we obtain a Cholesky-like factorization $\mathcal{M} = \mathcal{R}\mathcal{R}^\top$ that is equally well suited for an efficient solution of the Newton system (6). To see this, observe that Algorithm 2 is equivalent to a standard Cholesky factorization applied to

Algorithm 2: Structure-exploiting reverse Cholesky factorization

Input: Newton Hessian matrix $\mathcal{M}$
Output: Cholesky-like factor R for $\mathcal{M} = \mathcal{R}\mathcal{R}^\top$

for $k = N : 1$ **do** /* go by block columns */
  for $j = k \cdot n_x : (k-1) \cdot n_x + 1$ **do** /* go by columns */
    $w = \mathcal{M}_{jj}$
    $\bar{l} = \min(N \cdot n_x, (k+1) \cdot n_x)$ /* end of row fill in */
    for $l = j + 1 : \bar{l}$ **do** /* subtract row tail */
      $w = w - \mathcal{R}_{jl}^2$
    $\mathcal{R}_{jj} = \sqrt{w}$
    $\bar{i} = \max(1, (k-2) \cdot n_x + 1)$ /* end of column fill in */
    for $i = j - 1 : \bar{i}$ **do** /* write rest of column */
      $w = \mathcal{M}_{ij}$
      if $i > (k-1) \cdot n_x$ **then** /* end of row fill in */
        $\bar{l} = \min(N \cdot n_x, (k+1) \cdot n_x)$
      else
        $\bar{l} = \min(N \cdot n_x, k \cdot n_x)$
      for $l = j + 1 : \bar{l}$ **do** /* subtract row tail */
        $w = w - \mathcal{R}_{jl} \cdot \mathcal{R}_{il}$
      $\mathcal{R}_{ij} = w / \mathcal{R}_{jj}$

$\hat{\mathcal{M}} := \mathcal{J}\mathcal{M}\mathcal{J}^\top$ after a full row and column permutation through $\mathcal{J} := \begin{bmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{bmatrix}$. The advantage of applying this reverse Cholesky factorization in the dual Newton strategy is twofold. First, observe that a diagonal block W_k only changes from one Newton iteration to the next if the active-set on stage k or stage $k-1$ changes, and an off-diagonal block U_k only changes if the active-set on stage k changes (in particular note that $\mathcal{M}$ only needs to be recomputed in blocks with active-set changes). As Algorithm 2 only uses data from the last k block rows (and columns) in block iteration k, it is sufficient to restart the factorization from the block row that corresponds to the last active-set change. Furthermore, we can also expect better numerical properties of R, as the principal submatrix corresponding to stages without active state constraints is positive definite (recall that rank-deficiency of $\mathcal{M}$ can only arise from a redundancy in active stage constraints over several stages) and of similar conditioning as the original problem; a significant worsening of the conditioning can only appear in block-rows with active stage constraints, which, in a typical MPC setting, tend to appear rather on the earlier than on the later stages, and thus enter later in Algorithm 2 compared to the standard Cholesky factorization.

To formalize this, we identify the reverse Cholesky factorization with the discrete time Riccati recursion in the following. Let us regard the (possibly regularized) Newton Hessian $\mathcal{M} \succ 0$ in block form as defined in (11). Then, the reverse Cholesky factorization (Algorithm 2) is easily seen to be given by the recursion

$$\begin{aligned} X_{k-1} &= W_{k-1} - U_{k-1} \times X_k^{-1} \times U_{k-1}^\top \\ X_N &= W_N, \end{aligned} \tag{17}$$

where the Cholesky factor $\mathcal{R}$ in block form is given by

$$\mathcal{R} = \begin{bmatrix} \mathcal{R}_{1,1} & \mathcal{R}_{1,2} & & \\ & \mathcal{R}_{2,2} & \ddots & \\ & & \ddots & \mathcal{R}_{N-1,N} \\ & & & \mathcal{R}_{N,N} \end{bmatrix}$$

with upper triangular blocks $\mathcal{R}_{k,k}$ given implicitly (but uniquely) by $X_k =: \mathcal{R}_{k,k}\mathcal{R}_{k,k}^\top \ \forall k \in \mathcal{S}_0$ and dense blocks $\mathcal{R}_{k,k+1} = U_k \times \mathcal{R}_{k+1,k+1} \ \forall k \in \mathcal{S}_{0,N}$. Note that in this context the subscripts $\mathcal{R}_{k,k}$ refer to the block entries of $\mathcal{R}$ rather than to the individual entries (as used in Algorithm 2). We refer to X_k as Cholesky iterates in the following.

For linear time-invariant systems without active constraints it follows from Lemma 5 that

$$\begin{aligned} W_k &= CH^{-1}C^\top + EH^{-1}E^\top \\ U_k &= -EH^{-1}C^\top \end{aligned}$$

for $k \in \mathcal{S}_{0,N}$, where (analogously to QPs (1) and (P)) $C = \begin{bmatrix} A & B \end{bmatrix}$ are the dynamics, $E = \begin{bmatrix} \mathrm{I} & 0 \end{bmatrix}$ is the state selection matrix, and $H = \begin{bmatrix} Q & S^\top \\ S & R \end{bmatrix}$ is the objective Hessian. Due to a possibly different choice of the Hessian on the last interval ($H \equiv P$),

$$W_N = CH^{-1}C^\top + P^{-1}. \tag{18}$$

Theorem 6 *If P is the solution to the discrete time algebraic Riccati equation*

$$P = Q + A^\top PA - (S + A^\top PB)\left(R + B^\top PB\right)^{-1}(S^\top + B^\top PA),$$

then the Cholesky iterates X are constant, i.e., recursion (17) is stationary. In particular, $X_k := P^{-1} + CH^{-1}C^\top = W_N \ \forall k \in \mathcal{S}_0$.

Proof See Appendix.

This proof is easily seen to also extend to the linear time-varying case without active constraints, where we have (cf. Lemma 5):

$$\begin{aligned} W_k &= C_{k-1}H_{k-1}^{-1}C_{k-1}^\top + E_kH_k^{-1}E_k^\top \quad \forall k \in \mathcal{S}_{0,N} \\ U_k &= -E_kH_k^{-1}C_k^\top \quad \forall k \in \mathcal{S}_{0,N} \\ W_N &= C_{N-1}H_{N-1}^{-1}C_N^\top + H_N^{-1}, \end{aligned}$$

with $C_k = \begin{bmatrix} A_k & B_k \end{bmatrix}, \ \forall k \in \mathcal{S}_N$, $E_k = \begin{bmatrix} \mathrm{I} & 0 \end{bmatrix}, \ \forall k \in \mathcal{S}_N$, $H_k = \begin{bmatrix} Q_k & S_k^\top \\ S_k & R_k \end{bmatrix}, \ \forall k \in \mathcal{S}_N$, and $H_N = Q_N$.

Corollary 7 *Let $W_k,\ k \in \mathcal{S}_0$ and $U_k,\ k \in \mathcal{S}_{0,N}$ be computed from an linear time-varying system without (active) state constraints. Then, the Cholesky iterates $X_k,\ k \in \mathcal{S}_0$ from the Cholesky recursion (17) can be identified with the discrete time time-varying algebraic Riccati recursion*

$$P_N = Q_N \tag{19a}$$

$$P_{k-1} = Q_{k-1} + A_{k-1}^\top P_k A_{k-1} - (S_{k-1} + A_{k-1}^\top P_k B_{k-1}) \left(R_{k-1} + B_{k-1}^\top P_k B_{k-1}\right)^{-1} \times (S_{k-1}^\top + B_{k-1}^\top P_k A_{k-1}) \tag{19b}$$

via $X_k = P_k^{-1} + C_{k-1} H_{k-1}^{-1} C_{k-1}^\top \ \forall k \in \mathcal{S}_0$.

Proof See Appendix. □

3.6 Choice of the Newton step size

The piecewise quadratic nature of $f^*(\lambda)$ implies that a globalization strategy is needed for the algorithms to converge reliably. For computational efficiency close to the solution, where we assume the quadratic model of the dual function $f^*(\lambda)$ to be accurate, we propose a line search technique to find an (approximate) solution to

$$\alpha^i := \underset{0 \le \alpha \le 1}{\arg\max}\ f^*(\lambda^i + \alpha \Delta\lambda^i). \tag{20}$$

In contrast to general nonsmooth optimization, an exact line search is possible at reasonable cost in our context. In particular, $f^*(\lambda^i + \alpha \Delta\lambda)$ is a one-dimensional piecewise quadratic function along the search direction $\Delta\lambda$. An exact quadratic model in search direction can be built up by evaluating each $f_k^*(\lambda^i + \alpha_j \Delta\lambda), k \in \mathcal{S}$ at each value of $\alpha_j \in [0, 1]$ that corresponds to an active-set change on this stage. Alongside, slope information in search direction can be obtained by

$$\begin{aligned}\frac{\partial f_k^*(\lambda^i + \alpha_j\, \Delta\lambda)}{\partial \alpha} &= \frac{\partial f_k^*}{\partial \lambda}(\lambda^i + \alpha_j\, \Delta\lambda) \cdot \Delta\lambda \\ &= z_k^*(\lambda^i + \alpha_j \Delta\lambda)^\top \left(C_k{}^\top \Delta\lambda_{k+1} - E_k{}^\top \Delta\lambda_k\right) + c_k{}^\top \Delta\lambda_{k+1},\end{aligned}$$

cf. Eq. (13). As the second derivate is constant within each region, it can be cheaply and accurately computed as the difference quotient of the slope at the left and the right side of the intersection of each region with the search direction $\Delta\lambda$. Note that a parametric active-set strategy like qpOASES traverses the required points $z_k^*(\lambda^i + \alpha_j\, \Delta\lambda)$ for values of α_j corresponding to an active-set change on stage k naturally while computing $z_k^*(\lambda^i + \Delta\lambda)$ for the full step $\Delta\lambda$ (cf. [14]). When employing the clipping operation (9) for the solution of the stage problems (QP_k), the points of active-set changes along the search direction can analogously be determined by a simple ratio test.

With the piecewise quadratic model in search direction built up on each stage $k \in \mathcal{S}$, the dual objective value can be cheaply evaluated at each α_j, where an active-set change occurs on any stage $k \in \mathcal{S}$. Taking the maximum over all these values (e.g., by performing a bisection search) identifies, in conjunction with the slope information, the region containing the maximum dual function value in search direction, and the optimal α^* can be found as the maximum of the one-dimensional quadratic model of this region.

Alternatively, also heuristic backtracking-based search strategies seem appropriate in this context. Particularly the fact that the gradient evaluation $\mathcal{G}(\lambda)$ comes almost at the same cost as a function evaluation $f^*(\lambda)$ can be exploited within the line search.

A search strategy that seemed to perform particularly well in practice was a combination of a fast backtracking line search, allowing to quickly detect very small step sizes, with a bisection interval search for refinement.

While we make use of a backtracking line search to quickly decrease the maximum step size $\alpha_{\max}$, the minimum step size $\alpha_{\min}$ is given by the minimum scaling of the search direction that leads to an active-set change on any stage. While this is intuitively clear, as each region with a constant active-set is quadratic and Newton's method takes a step towards the minimum of a local quadratic function approximation, we give a formal proof for this in the following section, in the context of convergence (Lemma 8). This guaranteed minimum step size is indicated through $\lambda^i_{\alpha_{\min}}$ in Fig. 1.

Remark 10 We obtain all α-values at which active-set changes occur at no extra cost when employing an online active-set strategy to solve each (QP_k). Therefore, taking the minimum over all these α-values over all stages $k \in \mathcal{S}$ provides us with a lower bound for $\alpha_{\min}$. If the solution to (QP_k) is computed by Eq. (9), points of active-set changes can still be obtained cheaply by comparing the unconstrained to the clipped solution in each component.

4 Finite convergence of the algorithm

Convergence of non-smooth Newton methods has been proven before for functions with similar properties [21,31,36]. In the following we show convergence in the specific setting present in this paper; this allows us to present a shorter, more straightforward proof following the generic proof concept for descent methods. We furthermore require these details to establish the theory an infeasibility detection mechanism in Sect. 5, which according to our knowledge is novel to QP solvers based on nonsmooth Newton methods.

A bit of notation is needed throughout this and the following section. By $\mathcal{M}(\lambda)$ we refer to the Hessian matrix of (D), as defined in Eqs. (11–12). The possibly regularized version of $\mathcal{M}(\lambda)$ used in the solution step of the Newton system (6) is denoted by $\tilde{\mathcal{M}}(\lambda)$. We omit the dependency on λ occasionally for notational convenience when it is clear from the context. The active-set of stage constraints at λ is denoted by $\mathcal{A}^*(z^*(\lambda))$.

We start with a rather obvious result, that nonetheless is crucial for the practical performance of the Dual Newton Strategy.

Lemma 8 (Local one-step convergence) *Let (P) be feasible. Let λ^i be the current dual iterate in Algorithm 1. Let $\mathcal{M}(\lambda^i)$ be positive definite, i.e., no regularization is needed during Step 7 in Algorithm 1, and let $\Delta\lambda$ be the solution of the Newton equation (6). Then, if $\mathcal{A}^*(z^*(\lambda^i)) = \mathcal{A}^*(z^*(\lambda^i + \Delta\lambda))$, it holds that $\lambda^{i+1} := \lambda^i + \Delta\lambda$ solves Problem (D). In particular it holds that*

$$\underset{0\leq\alpha\leq 1}{\arg\max}\ f^*(\lambda^i + \alpha\,\Delta\lambda) = 1. \tag{21}$$

Proof By Lemma 3 f^* is piecewise quadratic in λ. By the construction in Lemma 5 we know that $\mathcal{M}(\lambda^i)$ is constant within each region $A(\lambda^i)$, since the active-set is fixed. By its definition, the Newton step $\Delta\lambda$ points to the maximum of the quadratic function characterizing $A(\lambda^i)$. By concavity of f^* it follows that $\lambda^i + \Delta\lambda$ has to be the maximum of f^* and thus solves (D). The claim (21) follows immediately. □

Lemma 8 is applied twofold in the dual Newton strategy. First, it allows us to make our line search smarter by only considering step sizes that lead to at least one active-set change (or full steps) as mentioned above in Sect. 3.6. Second, it shows that once the correct region of the solution is identified we have a one-step convergence to the exact solution (up to numerical accuracy), cf. Sect. 2.4. Next, we show global convergence.

Theorem 9 (Global convergence) *Let (P) be feasible. Let $\lambda^0 \in \mathbb{R}^{Nn_x}$ and let $\lambda^i \in \mathbb{R}^{Nn_x}$ be defined recursively by $\lambda^{i+1} := \lambda^i + \alpha^i \Delta\lambda^i$, where $\Delta\lambda^i$ is the (possibly regularized) solution to Eq. (6), and α^i is the solution to Eq. (20). Then the sequence $\{\lambda^i\}_{i\in\mathbb{N}_0} \subset \mathbb{R}^{Nn_x}$ converges to the unique maximum $\hat{\lambda}$ with $\mathcal{G}(\hat{\lambda}) = 0$.*

Proof The sequence $\{\lambda^i\}_{i\in\mathbb{N}_0}$ induces a sequence $\{f^i := f^*(\lambda^i)\}_{i\in\mathbb{N}_0} \subseteq \mathbb{R}$. By definition of the exact line search (20) it holds that $f^{i+1} \geq f^i$, i.e., $\{f^i\}_{i\in\mathbb{N}_0}$ is monotonically increasing. Since (P) is feasible, $f^*(\lambda)$ is a bounded, concave function by Lemma 3 and duality theory. By the Bolzano-Weierstrass Theorem $\{f^i\}_{i\in\mathbb{N}_0}$ thus converges to an accumulation point $\hat{f}$.

Due to monotonicity of $\{f^i\}_{i\in\mathbb{N}_0}$ it holds that $\{\lambda^i\}_{i\in\mathbb{N}_0}$ is contained in the superlevel set

$$\mathcal{F} := \left\{\lambda \in \mathbb{R}^{Nn_x} \mid f^*(\lambda) \geq f^*(\lambda^0)\right\},$$

which is compact since $f^*(\lambda)$ is a bounded concave function. A convergent subsequence $\{\lambda^{i^{(1)}}\} \subseteq \{\lambda^i\}_{i\in\mathbb{N}_0}$ therefore has to exist and its limit $\hat{\lambda}$ fulfills $f^*(\hat{\lambda}) = \hat{f}$ because of the induced monotonicity of $f^*(\lambda^{i^{(1)}})$.

What remains to show is that $\hat{\lambda}$ indeed maximizes $f^*(\lambda)$, i.e., $\mathcal{G}(\hat{\lambda}) = 0$. Assume contrarily $\mathcal{G}(\hat{\lambda}) \neq 0$. Because $\tilde{\mathcal{M}}(\lambda^i)$ is strictly positive definite and uniformly bounded away from 0 in norm, it holds that $\hat{\Delta\lambda} = \tilde{\mathcal{M}}(\hat{\lambda})^{-1}\mathcal{G}(\hat{\lambda}) \neq 0$ (cf. Eq. (6)) is an ascent direction. Then $\hat{\alpha} > 0$ holds for the solution of Eq. (20), and by C^1-continuity of $f^*(\lambda)$ we can conclude that there is a $\delta > 0$ with

$$f^*(\hat{\lambda} + \hat{\alpha}\hat{\Delta\lambda}) \geq f^*(\hat{\lambda}) + \delta.$$

Since $\{\lambda^{i^{(1)}}\}$ converges to $\hat{\lambda}$, an index $\bar{i} \in \mathbb{N}$ exists, such that for all $i^{(1)} \geq \bar{i}$ we have $\Delta\lambda^{i^{(1)}}$ sufficiently close to $\hat{\Delta\lambda}$ and $\lambda^{i^{(1)}}$ close enough to $\hat{\lambda}$ such that

$$f^*(\lambda^{i^{(1)}} + \alpha^{i^{(1)}} \Delta\lambda^{i^{(1)}}) \geq f^*(\lambda^{i^{(1)}} + \hat{\alpha} \Delta\lambda^{i^{(1)}}) \geq f^*(\hat{\lambda}) + \delta/2,$$

where the first inequality holds by the maximum property of the line search in each iteration, and the second one holds by continuity of $f^*(\lambda)$. This, however, would be a contradiction to $\hat{\lambda}$ being an accumulation point of a monotonically increasing sequence, so $\mathcal{G}(\hat{\lambda}) = 0$, and our claim holds. □

Lemma 10 *Let $z^*(\lambda^*)$ be a feasible solution for (P), that fulfills the LICQ. Then $\mathcal{M}(\lambda^*)$ is strictly positive definite.*

Proof From Lemma 5 we have that $\mathcal{M}(\lambda^*) = \mathcal{C}\,\mathcal{P}(\lambda^*)\,\mathcal{C}^\top$, where

$$\mathcal{P}(\lambda^*) = \mathcal{Z}^* \, (\mathcal{Z}^{*\top} \mathcal{H}\, \mathcal{Z}^*)^{-1} \, \mathcal{Z}^{*\top}$$

with $\mathcal{Z}^* := \text{block diag}(Z_0^*, Z_1^*, \ldots, Z_N^*)$ and $\mathcal{H} := \text{block diag}(H_0, H_1, \ldots, H_N)$. As in Lemma 5, each Z_k^*, $k \in \mathcal{S}$ denotes a basis matrix for the active constraints in the solution of (QP_k), in this context the solution given the subproblem parameter λ^*. Consider now

$$\lambda^\top \mathcal{M}(\lambda^*)\lambda = \lambda^\top \mathcal{C}\mathcal{Z}^*(\mathcal{Z}^{*\top}\mathcal{H}\mathcal{Z}^*)^{-1}\mathcal{Z}^{*\top}\mathcal{C}^\top\lambda. \tag{22}$$

Since $\mathcal{H}$ is positive definite and $\mathcal{Z}^*$, being a block diagonal composition of basis matrices, has full row rank, we have $\mathcal{Z}^{*\top}\mathcal{H}\mathcal{Z}^* \succ 0$, and thus $(\mathcal{Z}^{*\top}\mathcal{H}\mathcal{Z}^*)^{-1} \succ 0$. Using Eq. (22), this implies $\lambda^\top \mathcal{M}(\lambda^*)\lambda \geq 0$.

Assume $\lambda^\top \mathcal{M}(\lambda^*)\lambda = 0$. Since $(\mathcal{Z}^{*\top}\mathcal{H}\mathcal{Z}^*)^{-1} \succ 0$ it must hold that $\lambda^\top \mathcal{C}\mathcal{Z}^* = 0$.

The columns of $\mathcal{Z}^*$ however are linearly independent and span the nullspace of the active stage constraints, i.e., every vector from the nullspace of $\mathcal{Z}^*$ can be expressed by a linear combination of active stage constraints. If now $\lambda^\top \mathcal{C}$ lies in the nullspace of the active stage constraints this means there is a linear combination of active stage constraints that represents $\lambda^\top \mathcal{C}$, which is a linear combination of the stage coupling equality constraints. Since LICQ holds we can conclude that $\lambda = 0$, and thus $\mathcal{M}(\lambda^*) \succ 0$ holds. □

Corollary 11 (Finite termination of Algorithm 1) *Let $z^*(\lambda^*)$ be a feasible solution for (P) that fulfills the LICQ. Let $\lambda^0, \lambda^1, \ldots$ be computed from Algorithm 1 (in exact arithmetic). Then $\{\lambda^i\}_{i \in \mathbb{N}_0}$ becomes stationary after finitely many iterations, i.e., $\exists \bar{i} \; : \; \lambda^i = \lambda^* \; \forall \, i \geq \bar{i}$.*

Proof From Lemma 1 we know that $z_k^*(\lambda)$ depends continuously on λ. If no stage constraints of (QP_k) are weakly active in λ^*, then λ^* lies in the strict interior of a region A^* (with $\mathcal{M}(\lambda)$ constant on A^*) in the dual λ space. According to Theorem 9 there is a finite iteration index $\bar{i}$ with $\lambda^{\bar{i}} \in A^*$. Due to Lemma 10 we have that $\mathcal{M}(\lambda)$ is non-singular on A^*, and Lemma 8 guarantees convergence in the next iteration.

If there are weakly active constraints in λ^* (i.e., λ^* lies on the boundary between several, but a finite number of, nonempty regions $A^{(j_1)}, \ldots, A^{(j_n)}$ in Fig. 1), then due

to C^1 continuity of f, each quadratic function defining f on $A^{(j_i)}$, $i \in \{1, \ldots, n\}$ needs to have its maximum in λ^*. We can define a ball $B_\epsilon(\lambda^*)$ of fixed radius $\epsilon > 0$ around λ^* with the property that for every $\lambda \in B_\epsilon(\lambda^*)$ the dual function f on a region $A^{(j_i)}$ containing λ is again defined by a quadratic function having its maximum in λ^*. By the identical argument as before we can conclude finite termination of algorithm 1 with Theorem 9, Lemmas 10 and 8. □

5 Infeasibility handling

If the primal problem (P) is infeasible, two possible consequences for the dual problem (D) arise. If a stage k and a selection of stage constraints $\{\underline{d}_k^i \leq D_k^i z_k \leq \overline{d}_k^i\}_{i \in \mathcal{I}_k}$, $\mathcal{I}_k \subseteq \{1, ..., n_d\}$ exists that cannot be satisfied by any $z_k \in \mathbb{R}^{n_z}$, then and only then the dual problem (D) is infeasible as well, as no choice of λ will render (QP_k) feasible (see also Remark 1).

In the following, we describes the behavior of the algorithm when all problems (QP_k) are feasible, and yet (P) is infeasible.

Lemma 12 *Let all (QP_k) be feasible. If (P) is infeasible, the dual function $f^*(\lambda)$ is unbounded and there exists (at least) one region in λ space, $\emptyset \neq A^{\inf} \subseteq \mathbb{R}^{Nn_x}$, with constant $\mathcal{M}_{\inf}$ on $A^{\inf}$ and*

(i) *$\mathcal{M}_{\inf}$ singular, i.e., $\exists\, \lambda \neq 0 \;:\; \mathcal{M}_{\inf}\lambda = 0$,*
(ii) *$\forall\, \bar{f} \in \mathbb{R}\; \exists\, \hat{\lambda} \in A^{\inf} \;:\; f^*(\hat{\lambda}) > \bar{f}$,*
(iii) *for all $A^{(j)}$, defined by a representative $\lambda^{(j)}$, with $\mathcal{M}(\lambda^{(j)}) \succ 0$ it holds*

$$\exists\, \hat{\lambda} \in A^{\inf}\; \forall\, \bar{\lambda} \in A^{(j)} \;:\; f^*(\hat{\lambda}) > f^*(\bar{\lambda}).$$

Proof Since all (QP_k) are feasible, $f^*(\lambda)$ exists and (D) is feasible. Thus, (D) has to be unbounded by duality theory.

Let $\lambda^{(j)} \in A^{(j)}$ for any $A^{(j)}$. The mapping $\lambda \to f^*(\lambda)$ is onto an interval that contains the half-open interval $[f^*(\lambda^{(j)}), +\infty)$ since $f^*(\lambda)$ is continuous and unbounded. Since there is only a finite number of regions by Definition 2, property (ii) holds.

Assume now (i) is violated, i.e., $\exists\, A^{(j)}$ with $\mathcal{M}(\cdot) \succ 0$ on $A^{(j)}$ and $\forall\, \bar{f} \in \mathbb{R}\; \exists\, \lambda \in A^{(j)} : f^*(\lambda) > \bar{f}$. Since $\mathcal{M}(\cdot) \succ 0$ is constant on $A^{(j)}$ we have that $f^*(\lambda)$ is strictly, and even strongly concave on $A^{(j)}$. Therefore $\exists\, \bar{f} < \infty$ with $f^*(\lambda) \leq \bar{f}\; \forall\, \lambda \in \mathcal{A}^{(j)}$, a contradiction. Therefore (i) holds.

In particular $f^*(\lambda)$ is bounded (from above) on each $A^{(j)}$ with $\mathcal{M}(\cdot) \succ 0$. Since there is only a finite number of regions, property (ii) implies (iii). □

Lemma 12 tells us that unboundedness of the dual objective function value can only occur in regions with singular Newton Hessian matrix $\mathcal{M}(\cdot)$. We further characterize these unbounded regions in the following.

Definition 3 (Infinite ray) For $\Delta\lambda \neq 0$ we call a pair $(\bar{\lambda}, \Delta\lambda)$ an infinite ray, if there is a region $A^{\inf} \subseteq \mathbb{R}^{Nn_x}$ represented by $\bar{\lambda} \in A^{\inf}$ such that the ray is

1. contained in the region: $\bar{\lambda} + \delta \cdot \Delta\lambda \in A^{\inf} \ \ \forall \delta > 0$,
2. in the nullspace of the region's Hessian: $\mathcal{M}(\bar{\lambda})\Delta\lambda = 0$,
3. an ascent direction: $\mathcal{G}(\bar{\lambda})^\top \Delta\lambda > 0$.

Clearly every region containing an infinite ray is an unbounded region in the sense of Lemma 12. We additionally have the following characterizations.

Remark 11 Because $\Delta\lambda$ lies in the nullspace of the negated dual Hessian $\mathcal{M}(\cdot)$, the dual gradient $\mathcal{G}(\cdot)$ along a ray $(\bar{\lambda}, \Delta\lambda)$ is constant.

Lemma 13 *Let $(\bar{\lambda}, \Delta\lambda)$ be an infinite ray contained in $A^{\inf}$. For every $\tilde{\lambda} \in A^{\inf}$ it holds $\tilde{\lambda} + \gamma \cdot \Delta\lambda \in A^{\inf} \ \ \forall \gamma > 0$, i.e., $(\tilde{\lambda}, \Delta\lambda)$ is an infinite ray as well.*

Proof Since $A^{\inf}$ is convex by Lemma 2 and both $\tilde{\lambda} \in A^{\inf}$ and $\bar{\lambda} + \gamma \cdot \Delta\lambda \in A^{\inf}$ for each choice of $\gamma > 0$, it holds

$$\beta \cdot \tilde{\lambda} + (1 - \beta) \cdot (\bar{\lambda} + \gamma \cdot \Delta\lambda) \ \in A^{\inf} \ \ \forall \beta \in [0, 1]. \tag{23}$$

From Lemma 2 we also have that $A^{\inf}$ is polyhedral and therefore closed in $\overline{\mathbb{R}}^{Nn_x}$, where $\overline{\mathbb{R}} := \mathbb{R} \cup \{-\infty, +\infty\}$. Thus, the limit of (23) for $\gamma \to \infty$ is contained in $A^{\inf}$, and the claim holds. □

Lemma 13 is interesting because it tells us that $A^{\inf}$ has a cone- or beam-like shape. This will play a role in the following, when we characterize certificates for an unbounded dual problem.

Definition 4 (Ridge) Let $(\lambda^\ddagger, \Delta\lambda^\ddagger)$ be an infinite ray with $\Delta\lambda^\ddagger \neq 0$. We call $(\lambda^\ddagger, \Delta\lambda^\ddagger)$ a ridge if it holds $\Delta\lambda^\ddagger = \gamma \cdot \mathcal{G}(\lambda^\ddagger)$ for a $\gamma > 0$, i.e., if $\Delta\lambda^\ddagger$ is aligned with the gradient of the dual function f^* along the ray it is defining.

Theorem 14 *Let all (QP_k) be feasible. If (P) is infeasible, then a ridge $(\lambda^\ddagger, \Delta\lambda^\ddagger)$ exists.*

Proof From Lemma 12 we know that a non-empty region $A^{\inf}$ exists. We further know that $f^*(\lambda)$ is an unbounded concave quadratic function on $A^{\inf}$. An infinite ray $(\bar{\lambda}, \Delta\lambda)$ in the sense of Definition 3 therefore has to exist in $A^{\inf}$.

From Lemma 13 we have that in this case $(\tilde{\lambda}, \Delta\lambda)$ is an infinite ray as well for every $\tilde{\lambda} \in A^{\inf}$.

Let $A^{\inf}_\cup := A^{\inf,1} \cup A^{\inf,2} \cup \ldots \cup A^{\inf,n}$ be the union of all unbounded regions in the sense of Lemma 12. Since $f^*(\lambda)$ is concave, $A^{\inf}_\cup$ is connected. For the same reason it also holds that the superlevel set

$$\bar{A}^{\inf}_\cup := \{\lambda \in A^{\inf}_\cup \mid f^*(\lambda) \geq \bar{f}\} \tag{24}$$

is convex if we choose $\bar{f} \in \mathbb{R}$ sufficiently large. Since $f^*(\lambda)$ is continuous and unbounded on $\bar{A}^{\inf}_\cup$, it holds that $\bar{A}^{\inf}_\cup$ is (Nn_x)-dimensional, i.e., full-dimensional; otherwise a directional vector $e \in \mathbb{R}^{Nn_x}$ would exist with $\tilde{\lambda} \in \text{int}(\bar{A}^{\inf}_\cup)$, i.e., $f^*(\tilde{\lambda}) > \bar{f}$ and $f^*(\tilde{\lambda} + \epsilon \cdot e) < \bar{f} \ \ \forall \epsilon > 0$, a violation of continuity.

Assume for the moment that every region $A^{\text{inf},j}$ only contains one infinite ray (up to translation and scaling). Consider the nonempty intersection of $\bar{A}^{\text{inf}}_{\cup}$ with a $(Nn_x - 1)$-dimensional Hyperplane $F \subset \mathbb{R}^{Nn_x}$. If $F \cap \bar{A}^{\text{inf}}_{\cup}$ does not contain a singular ray, $f^*(\lambda)$ has to be bounded on $F \cap \bar{A}^{\text{inf}}_{\cup}$, as $f^*(\lambda)$ is composed from only a finite number of concave quadratic functions. In particular $f^*(\lambda)$ attains a maximum $\hat{\lambda}$ somewhere in the intersection.

This maximum $\hat{\lambda}$ is characterized by the fact that the dual gradient $\mathcal{G}(\hat{\lambda})$ is orthogonal to F if $\hat{\lambda} \in \text{int}(F \cap \bar{A}^{\text{inf}}_{\cup})$. Since the dual λ space is (Nn_x)-dimensional $\mathcal{G}(\hat{\lambda})/\|\mathcal{G}(\hat{\lambda})\|$ is uniquely defined by F and vice versa. Recall that $(\hat{\lambda}, \Delta\lambda)$ is an infinite ray in every $\hat{\lambda} \in \bar{A}^{\text{inf}}_{\cup}$ for one fixed $\Delta\lambda$ as shown above. Since $\mathcal{G}(\lambda)$ is continuous, varying (i.e., "rotating") F eventually has to yield a $\hat{\lambda}$ with $\mathcal{G}(\hat{\lambda}) = \gamma \Delta\lambda$ for some $\gamma > 0$.

If now there is a region A^{inf}_j with more than one infinite ray (i.e., the directional vectors $\Delta\lambda$ of the infinite rays span a space of dimensionality $k > 1$) the same argument can be applied, but with a $(Nn_x - k)$-dimensional Hyperplane $F \subset \mathbb{R}^{Nn_x}$. In this case $\mathcal{G}(\hat{\lambda})$ is not uniquely defined anymore, but lies in the normal space of F. By the same argument as above there has to be an F whose normal space coincides with the space spanned by the directional vectors $\Delta\lambda$ of the infinite rays of A^{inf}_j, and thus again there is an infinite ray $(\hat{\lambda}, \Delta\lambda)$ coinciding with the gradient $\mathcal{G}(\hat{\lambda})$ at its base point $\hat{\lambda}$. □

Remark 12 Let $(\lambda^\ddagger, \mathcal{G}(\lambda^\ddagger)) \subseteq A^\ddagger \subseteq \bar{A}^{\text{inf}}_{\cup}$ be a ridge. Clearly, $\mathcal{M}(\lambda^\ddagger)$ is rank deficient (and constant on $A^\ddagger$) and thus regularized with $\delta I \succ 0$ by Algorithm 1. By Theorem 14 and Definitions 3 and 4 it holds $\mathcal{M}(\lambda^\ddagger)\mathcal{G}(\lambda^\ddagger) = 0$ and we have

$$\begin{aligned}
\mathcal{G}(\lambda^\ddagger) &= \frac{1}{\delta}\,\delta\,\mathcal{G}(\lambda^\ddagger) + \frac{1}{\delta}\underbrace{\mathcal{M}(\lambda^\ddagger)\,\mathcal{G}(\lambda^\ddagger)}_{=0} = \frac{1}{\delta}\,(\underbrace{\mathcal{M}(\lambda^\ddagger) + \delta\, I}_{\succ 0})\,\mathcal{G}(\lambda^\ddagger) \\
&\Leftrightarrow\ \tilde{\mathcal{M}}(\lambda^\ddagger)^{-1}\mathcal{G}(\lambda^\ddagger) = \frac{1}{\delta}\,\mathcal{G}(\lambda^\ddagger) \\
&\Leftrightarrow\ \Delta\lambda^\ddagger = \frac{1}{\delta}\,\mathcal{G}(\lambda^\ddagger),
\end{aligned}$$

i.e., Algorithm 1 only performs gradient steps and thus remains on the ridge. Therefore we can see a ridge as the analogon to a fixed point in the case where the dual function $f^*(\lambda)$ is unbounded.

Lemma 15 (Minimality of the ridge gradient) *Let* $(\lambda^\ddagger, \Delta\lambda^\ddagger)$ *be a ridge. Then*

$$\|\mathcal{G}(\lambda^\ddagger)\|_2 \leq \|\mathcal{G}(\lambda)\|_2 \tag{25}$$

for all $\lambda \in \mathbb{R}^{Nn_x}$. *Furthermore, inequality* (25) *is strict except for those* λ *that lie on a ridge, i.e., for which there is a ridge* $(\bar{\lambda}, \overline{\Delta\lambda})$ *such that* $\lambda = \bar{\lambda} + \gamma\,\overline{\Delta\lambda}$ *for a* $\gamma \geq 0$. *In particular the ridge* $(\lambda^\ddagger, \Delta\lambda^\ddagger) \subseteq A^\ddagger$ *is unique up to scaling of* $\Delta\lambda^\ddagger$ *and translations from the nullspace of* $\mathcal{M}(\lambda^\ddagger)$.

Proof Let $\lambda \in \mathbb{R}^{Nn_x}$ and let $(\lambda^\ddagger, \Delta\lambda^\ddagger)$ be a ridge. Due to concavity of $f^*(\cdot)$ we have $(\mathcal{G}(\lambda) - \mathcal{G}(\lambda^\ddagger))^\top(\lambda^\ddagger - \lambda) \geq 0$. Since the $\mathcal{G}(\cdot)$ is constant along the ridge, $(\lambda^\ddagger, \Delta\lambda^\ddagger)$

and $\Delta\lambda^\ddagger$ and $\mathcal{G}(\lambda^\ddagger)$ only differ by a positve scalar factor, and also

$$(\mathcal{G}(\lambda) - \mathcal{G}(\lambda^\ddagger))^\top \frac{1}{\gamma}(\lambda^\ddagger + \gamma\, \mathcal{G}(\lambda^\ddagger) - \lambda) \geq 0$$

for all $\gamma > 0$. Since concavity of $f^*(\cdot)$ clearly also holds on the extended domain $\overline{\mathbb{R}}^{Nn_x}$, where $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$, we have $(\mathcal{G}(\lambda) - \mathcal{G}(\lambda^\ddagger))^\top \mathcal{G}(\lambda^\ddagger) \geq 0$ in the limit for $\gamma \to \infty$. This is equivalent to

$$\|\mathcal{G}(\lambda^\ddagger)\|_2^2 = \mathcal{G}(\lambda^\ddagger)^\top \mathcal{G}(\lambda^\ddagger) \leq \mathcal{G}(\lambda)^\top \mathcal{G}(\lambda^\ddagger) \leq \|G(\lambda)\|_2 \cdot \|\mathcal{G}(\lambda^\ddagger)\|_2,$$

where the last inequality is the Cauchy-Schwarz inequality. Thus (25) holds.

For the proof of the second part of the Lemma, we note that the Cauchy-Schwarz inequality is strict unless $G(\lambda)$ is a positive multiple of $\mathcal{G}(\lambda^\ddagger)$, so $\|\mathcal{G}(\lambda)\|_2 = \|\mathcal{G}(\lambda^\ddagger)\|_2$ implies $\mathcal{G}(\lambda) = \mathcal{G}(\lambda^\ddagger)$. Let us therefore consider an arbitrary $\lambda \in \mathbb{R}^{Nn_x}$ with $\mathcal{G}(\lambda) = \mathcal{G}(\lambda^\ddagger)$. Due to concavity we have

$$\begin{aligned} f^*(\beta\,\lambda + (1-\beta)\,\lambda^\ddagger) &\leq f^*(\lambda) + \mathcal{G}(\lambda)^\top(\beta\,\lambda + (1-\beta)\,\lambda^\ddagger - \lambda) \\ &= f^*(\lambda) + (1-\beta)\cdot \mathcal{G}(\lambda)^\top(\lambda^\ddagger - \lambda) \end{aligned} \tag{26}$$

for $\beta \in [\,0\,,\,1\,]$, and analogously

$$f^*(\beta\,\lambda + (1-\beta)\,\lambda^\ddagger) \leq f^*(\lambda^\ddagger) - \beta \cdot \mathcal{G}(\lambda^\ddagger)^\top(\lambda^\ddagger - \lambda). \tag{27}$$

We can multiply (26) by β and (27) by $(1-\beta)$, add both inequalities up and, using $\mathcal{G}(\lambda) = \mathcal{G}(\lambda^\ddagger)$, obtain

$$f^*(\beta\,\lambda + (1-\beta)\,\lambda^\ddagger) \leq \beta\, f^*(\lambda) + (1-\beta)\, f^*(\lambda^\ddagger).$$

By concavity of $f^*(\cdot)$ also the converse inequality holds and we have

$$f^*(\beta\,\lambda + (1-\beta)\,\lambda^\ddagger) = \beta\, f^*(\lambda) + (1-\beta)\, f^*(\lambda^\ddagger), \tag{28}$$

i.e., linearity of $f^*(\cdot)$ on the interval between λ and $\lambda^\ddagger$. Once more since $\mathcal{G}(\cdot)$ is constant along the ridge $(\lambda^\ddagger, \Delta\lambda^\ddagger)$, (28) holds analogously for all $\tilde{\lambda}^\ddagger := \lambda^\ddagger + \gamma\,\Delta\lambda^\ddagger$ in place of $\lambda^\ddagger$, i.e., f^* is linear on the interval $[\,\lambda\,,\,\lambda^\ddagger + \gamma\,\Delta\lambda^\ddagger\,]$ for all choices of $\gamma > 0$. Since the space of linear functions from $\mathbb{R}^{Nn_x}$ to $\mathbb{R}$ is closed, linearity also holds in the limit for $\gamma \to \infty$, which is the half-open interval $\{\lambda + \gamma \cdot \Delta\lambda^\ddagger \mid \gamma \in [\,0\,,\,\infty\,)\}$. Since $\mathcal{G}(\lambda) = \mathcal{G}(\lambda^\ddagger)$ (which itself is a positive multiple of $\Delta\lambda^\ddagger$) we therefore have that $(\lambda, \mathcal{G}(\lambda))$ is itself a ridge, which is moreover parallel to $(\lambda^\ddagger, \Delta\lambda^\ddagger)$.

Since $f^*(\cdot)$ is specifically linear on $[\,\lambda\,,\,\lambda^\ddagger\,]$, $(\lambda, \mathcal{G}(\lambda))$ differs from $(\lambda^\ddagger, \Delta\lambda^\ddagger)$ only by a shift that lies in the nullspace of $\mathcal{M}(\lambda^\ddagger)$ as we claimed above. □

Theorem 16 (Convergence to an infinite ray) *If (P) is infeasible, then exactly one of the two following statements is true:*

1. *(QP_k) is infeasible for at least one $k \in \mathcal{S}$;*
2. *Algorithm 1 converges to an infinite ray (i.e., the distance of the iterates to the half-open set characterized by the ray vanishes) if the regularization parameter δ is chosen sufficiently large.*

Proof Let (P) be infeasible. Because of the special time coupling structure of (P) either there exists a minimal infeasible set (a selection of constraints of minimal size that cannot be fulfilled at the same time) that is contained in the set of local stage constraints of one (QP_k), or all minimal infeasible sets consist of local stage constraints (P3) of several stages $k_1 < k_2 < \cdots < k_n$ and the corresponding time coupling constraints between k_1 and k_n (a subset of constraints (P2)). In the former case Statement 1 holds, and infeasibility is detected by the stage QP solver on first execution. In the latter case we have that the partial dual function $f^*(\lambda)$ exists and can be evaluated. In particular $f^*(\lambda)$ is unbounded by Lemma 12. In the remainder of the proof we show that in this case Algorithm 1 indeed converges to an infinite ray.

As we have seen in the proof of Theorem 9 the iterates λ^i of Algorithm 1 defined by Eqs. (5), (6), and (20) induce a monotonically increasing sequence $\{f^*(\lambda^i)\}_{i\in\mathbb{N}_0}$. We claim that $\{f^*(\lambda^i)\}_{i\in\mathbb{N}_0}$ does not converge if $f^*(\lambda)$ is unbounded.

Assume to the contrary that $\{f^*(\lambda^i)\}_{i\in\mathbb{N}_0}$ converges. Clearly $\mathcal{G}(\lambda)$ does not vanish since $\mathcal{G}(\lambda) = 0$ would imply the existence of a feasible solution of (P) by Remark 5. By Lemma 15 we further know the existence of a

$$0 < G_{\min} := \min_{\lambda\in\mathbb{R}^{Nn_x}} \|\mathcal{G}(\lambda)\|. \tag{29}$$

If now $\{f^*(\lambda^i)\}_{i\in\mathbb{N}_0}$ converges, clearly the updates $\alpha^i \cdot \tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i)$, have to vanish. Since there are only finitely many different $\tilde{\mathcal{M}}(\cdot) \succ 0$ and $\mathcal{G}(\cdot)$ is bounded away from 0 by Eq. (29) this implies that

$$\alpha^i = \arg\max_{0\le\alpha\le 1} f^*(\lambda^i + \alpha\Delta\lambda)\,,$$

with $\Delta\lambda = \tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i)$, cf. Eq. (20), has to drop to 0.

At each iteration i three cases for α^i could appear:

(i) $\alpha^i = 1$;

(ii) $\alpha^i = 0$. We have $\mathcal{G}(\lambda^i) \neq 0$ and $\tilde{\mathcal{M}}(\lambda^i) \succ 0$, so $\tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i)$ is an ascent direction. By C^1-continuity of $f^*(\cdot)$ an ascent is possible and thus $\alpha^i \neq 0\ \forall i \in \mathbb{N}_0$;

(iii) $0 < \alpha^i < 1$. Then by maximality α^i fulfills $\mathcal{G}(\lambda^i + \alpha^i\Delta\lambda)^\top\Delta\lambda = \mathcal{G}(\lambda^i + \alpha^i\Delta\lambda)^\top\tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i) = 0$. Regard $\phi(\alpha) := \mathcal{G}(\lambda^i + \alpha\Delta\lambda)^\top\tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i)$ as a function in α. Clearly $\phi(0) = \mathcal{G}(\lambda^i)^\top\tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i)$ is bounded away from 0, since there are only finitely many different $\tilde{\mathcal{M}}(\cdot) \succ 0$ and $\mathcal{G}(\cdot)$ is bounded away from 0 by Eq. (29). Further $\phi(\alpha)$ is continuous and its derivative $\phi'(\alpha) = -\Delta\lambda^\top\mathcal{M}(\lambda^i+\alpha\Delta\lambda)\Delta\lambda$ is bounded from below (for a fixed $\Delta\lambda$), again since there are only finitely many different $\tilde{\mathcal{M}}(\cdot)$; note that the ascent direction $\Delta\lambda$ cannot

grow to infinity for $i \to \infty$ since $f^*(\lambda)$ is concave and $f^*(\lambda^i)$ is increasing. Therefore α^i has to be bounded away from 0 by a problem-data specific constant that is independent from the iteration index i.

Since also α^i does not converge to 0 we have a contradiction, and our claim that $\{f^*(\lambda^i)\}_{i\in\mathbb{N}_0}$ diverges (even monotonically) holds true. In particular every fixed function value $\bar{f}$ will be exceeded and for every suitably large $\bar{f}$ the iterates of Algorithm 1 will remain in $\bar{A}_{\cup}^{\inf}$ (as defined in Eq. (24)) after a finite number of iterations.

In the remainder of the proof we first show local and subsequently global attractiveness of a ridge in the sense of Theorem 14.

Let $A^{\ddagger}$ denote the region that contains a ridge $(\lambda^{\ddagger}, \mathcal{G}(\lambda^{\ddagger}))$. For two subsequent iterates both contained in $A^{\ddagger}$ (characterized by a constant Newton matrix $\mathcal{M}$) we have the exact linear model

$$\begin{aligned}\mathcal{G}(\lambda^{i+1}) &= \mathcal{G}(\lambda^i + \alpha^i \tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i)) \\ &= \mathcal{G}(\lambda^i) - \alpha^i \mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i).\end{aligned} \tag{30}$$

The cone-like quadratic shape of $A^{\ddagger}$ (cf. Lemma 13) implies that each subsequent iterate λ^{i+1} is contained in $A^{\ddagger}$ if $\lambda^i \in A^{\ddagger}$. The linear expansion of the gradient (30), becomes $\mathcal{G}(\lambda^{i+1}) = \left(\mathcal{I} - \mathcal{M}\,\tilde{\mathcal{M}}^{-1}\right)\mathcal{G}(\lambda^i)$. Clearly $\mathcal{I} - \mathcal{M}\,\tilde{\mathcal{M}}^{-1} = \mathcal{I} - \left(\tilde{\mathcal{M}} - \delta\,\mathcal{I}\right)\tilde{\mathcal{M}}^{-1} = \delta\,\tilde{\mathcal{M}}^{-1}$ is positive definite. On the other hand $\mathcal{M}\,\tilde{\mathcal{M}}^{-1} = \mathcal{I} - \delta\,\tilde{\mathcal{M}}^{-1}$ is positive semidefinite, so $\|\mathcal{G}(\lambda^{i+1})\| \le \|\mathcal{G}(\lambda^i)\|$.

Whenever $\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) \neq 0$ we have $\delta\,\mathcal{G}(\lambda^i)^\top\tilde{\mathcal{M}}^{-1}\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) > 0$ for any $\delta > 0$. This can be easily verified, e.g., by using the fact that $0 \preceq \mathcal{M} =: B\,B^\top$ has a (not necessarily full rank) symmetric factorization. Assuming $\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) \neq 0$ it then holds[1]

$$\begin{aligned}&\delta\,\mathcal{G}(\lambda^i)^\top\tilde{\mathcal{M}}^{-1}\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) > 0 \\ \Leftrightarrow\ &\mathcal{G}(\lambda^i)^\top\left(\mathcal{I} - \mathcal{M}\,\tilde{\mathcal{M}}^{-1}\right)\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) > 0 \\ \Leftrightarrow\ &\mathcal{G}(\lambda^i)^\top\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) - \mathcal{G}(\lambda^i)^\top\tilde{\mathcal{M}}^{-1}\mathcal{M}\,\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) > 0 \\ \Rightarrow\ &2\cdot\mathcal{G}(\lambda^i)^\top\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) - \mathcal{G}(\lambda^i)^\top\tilde{\mathcal{M}}^{-1}\mathcal{M}\,\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) > 0.\end{aligned}$$

This implies that

$$\begin{aligned}\|\mathcal{G}(\lambda^{i+1})\|^2 &= \mathcal{G}(\lambda^i)^\top\left(\mathcal{I} - \mathcal{M}\,\tilde{\mathcal{M}}^{-1}\right)^\top\left(\mathcal{I} - \mathcal{M}\,\tilde{\mathcal{M}}^{-1}\right)\mathcal{G}(\lambda^i) \\ &= \|\mathcal{G}(\lambda^i)\|^2 - 2\cdot\mathcal{G}(\lambda^i)^\top\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) \\ &\quad + \mathcal{G}(\lambda^i)^\top\tilde{\mathcal{M}}^{-1}\mathcal{M}\,\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) \\ &< \|\mathcal{G}(\lambda^i)\|^2\end{aligned}$$

[1] We make use of the symmetry and positive semidefiniteness of $\mathcal{M}\,\tilde{\mathcal{M}}^{-1} = \mathcal{I} - \delta\,\tilde{\mathcal{M}}^{-1}$.

for all iterates with $\mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}(\lambda^i) \neq 0$, and together with (30) we can conclude that the sequence of gradients converges to a limit $\mathcal{G}^*$ that fulfills

$$\begin{aligned} & \mathcal{M}\,\tilde{\mathcal{M}}^{-1}\mathcal{G}^* = 0 \\ \Leftrightarrow \quad & \left(\tilde{\mathcal{M}} - \delta\,\mathcal{I}\right)\tilde{\mathcal{M}}^{-1}\mathcal{G}^* = 0 \\ \Leftrightarrow \quad & \delta\,\tilde{\mathcal{M}}^{-1}\mathcal{G}^* = \mathcal{G}^*, \end{aligned}$$

the ridge property. Recall that by Lemma 15 the ridge is unique.

It remains to show that the iterates λ^i of Algorithm (1) also converge globally to $A^\ddagger$. To see this, we consider the auxiliary function

$$\tilde{f}(\lambda) := f^*(\lambda) - \mathcal{G}(\lambda^\ddagger)^\top\lambda.$$

This function is clearly piecewise quadratic and convex, just like f^*, and its second derivative equals the second derivative of f^*, while the first derivative of $\tilde{f}$ is given by $\frac{\partial\tilde{f}}{\partial\lambda} = \mathcal{G}(\lambda)^\top - \mathcal{G}(\lambda^\ddagger)^\top$. It is furthermore bounded, since due to concavity we have for every $\lambda \in \mathbb{R}^{Nn_x}$

$$\tilde{f}(\lambda) \leq \tilde{f}(\lambda^\ddagger) + (\mathcal{G}(\lambda^\ddagger) - \mathcal{G}(\lambda^\ddagger))^\top(\lambda - \lambda^\ddagger) = \tilde{f}(\lambda^\ddagger).$$

The iterates λ^i of Algorithm (1) are increasing in $\tilde{f}$ if and only if the step directions $\tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i)$ are always ascent directions, i.e., if

$$\alpha^i(\mathcal{G}(\lambda^i) - \mathcal{G}(\lambda^\ddagger))^\top\tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i) \;>\; 0$$

holds. We have shown above that α^i is bounded away from 0 by an iteration-independent constant and therefore we equivalently have the condition

$$\delta\,\mathcal{G}(\lambda^i)^\top\tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i) \;>\; \delta\,\mathcal{G}(\lambda^\ddagger)^\top\tilde{\mathcal{M}}(\lambda^i)^{-1}\mathcal{G}(\lambda^i), \tag{31}$$

where δ is the regularization parameter in Algorithm 1. Since $\delta\,\tilde{\mathcal{M}}(\lambda^i)^{-1}$ is positive definite, each of its finitely many distinct values $\delta\,\tilde{\mathcal{M}}_{(j)}^{-1}$ defines a scalar product that induces a norm on $\mathbb{R}^{Nn_x}$. Applying the Cauchy-Schwarz inequality to (31) we have that the iterates λ^i are increasing iff

$$\begin{aligned} & \|G(\lambda^i)\|^2_{\delta\tilde{\mathcal{M}}_{(j)}^{-1}} \;>\; \|G(\lambda^\ddagger)\|_{\delta\tilde{\mathcal{M}}_{(j)}^{-1}} \cdot \|G(\lambda^i)\|_{\delta\tilde{\mathcal{M}}_{(j)}^{-1}} \\ \Leftrightarrow \quad & \|G(\lambda^i)\|_{\delta\tilde{\mathcal{M}}_{(j)}^{-1}} \;>\; \|G(\lambda^\ddagger)\|_{\delta\tilde{\mathcal{M}}_{(j)}^{-1}}. \end{aligned} \tag{32}$$

While λ^i is not contained in a ridge, we have $\|G(\lambda^i)\|_2 > \|G(\lambda^\ddagger)\|_2$ from Lemma 15; furthermore $I - \delta\,\tilde{\mathcal{M}}_{(j)}^{-1} = I - \delta\,(\mathcal{M}_{(j)} + \delta I)^{-1}$ vanishes for sufficiently large choices of δ, fulfilling (32) and thus showing that the iterates λ^i are also ascending in the auxiliary function $\tilde{f}$. Since $\tilde{f}$ is bounded, we can conclude that the region

$A^{\ddagger}$, containing a ridge, is eventually reached, thus concluding the proof of global convergence. □

Theorem 16 can be used algorithmically to detect infeasibility of (P). If any (QP_k) is infeasible, it will be detected by the stage QP solver (either qpOASES, or during the clipping operation) on the first execution and (P) is then immediately known to be infeasible. Otherwise, in the case of infeasibility through the coupling constraints, we know by Theorem 16 that the iterates of Algorithm 1 will eventually converge to a ridge. Here, we could gradually increase the regularization parameter δ, if the iterates remain in regions with singular Hessians. A check whether $\mathcal{G}(\lambda^i)$ is in the nullspace of $\mathcal{M}(\lambda^i)$ (i.e., $\mathcal{M}(\lambda^i)\mathcal{G}(\lambda^i) \approx 0$) every few (regularized) iterations without an active-set change, combined with a check whether any active-set change occurs at all in the Newton direction will eventually conclude infeasibility as stated by Theorem 16. Note that the check for active-set changes in the Newton direction can cheaply be performed both in qpOASES and the clipping QP solver by simply considering the signs in the ratio test of the active and inactive constraints, cf. Sect. 3.6. Additionally, practical infeasibility can be concluded, when the objective function value exceeds a certain large threshold (caused by an explosion of the norm of the iterates λ^i).

Remark 13 In practice, the dual iterates λ^i in Algorithm 1 were indeed always observed to grow very fast and reach a ridge quickly in infeasible problems because of the initially small regularization.

We note that the theoretical result of Theorem 16 is somewhat unsatisfactory, since it requires a modification of the regularization parameter δ (or even to only do gradient steps in regions with a singular Hessian). We initially aimed at proving Theorem 16 for generic regularized Newton steps, i.e., independent of δ. Despite some considerable effort, we could not derive a formal argument that links condition (32) with Lemma 15, since the ordering relations in the Euclidian norm on the one hand and the norm induced by $\delta\,\tilde{\mathcal{M}}_{(j)}^{-1}$ on the other thand might be different. Still, we are confident that also regularized Newton updates in general, independent of the choice of δ, are increasing in $\tilde{f}$ on a global scale (though not necessarily monotonically), and thus formulate the following conjecture.

Conjecture 17 *Theorem 16 holds independently for all choices of the regularization parameter* $\delta > 0$.

6 Concurrency in the dual Newton strategy

One important advantage of the Dual Newton Strategy is that, as opposed to conventional active-set or interior-point methods, it is an easily parallelizable algorithm. Analyzing Algorithm 1 in this respect, we observe that all stage QPs can be solved concurrently in $N + 1$ threads in Step 2. Next, each block of the dual gradient $\mathcal{G}(\lambda)$ only depends on the solution of two neighboring stage QPs (cf. Eq. 10), and therefore the setup can be done concurrently in N threads (Step 3). Also in the setup of the symmetric Newton matrix M, Step 6, each diagonal block only depends on the solution of two adjacent stage QP solutions, while each off-diagonal block only depends on the

solution of one stage QP; therefore the workload of the setup of M can be distributed on N threads almost equally. During the line search procedure in Step 8 of Algorithm 1, the expensive steps in each iteration consist of solving all stage QPs for the new step size guess and computing the corresponding gradient, both of which can be spread over $N + 1$ threads with almost equal workload.

Therefore, so far all steps of significant workload can be run fully in parallel in each major iteration of Algorithm 1, except for the solution of the structured linear system, Step 7. In a sequential implementation, solving the system based on a structure-exploiting reverse Cholesky factorization as proposed in Sect. 3.5 seems most efficient. This algorithm, however, is completely serial on the higher level and therefore cannot be expected to scale well with the number of available computing threads on modern and forthcoming multi-core CPU architectures.

We therefore propose a parallelizable solution algorithm for the structured Newton system in Sect. 6.1, as an alternative to Algorithm 2. The overall cost of this algorithm is roughly twice as high in number of FLOPs as Algorithm 2, but it comes at a parallel complexity of only $\mathcal{O}(\log N)$ on N threads; therefore, the overall time complexity of one iteration of the dual Newton strategy comes down to only $\mathcal{O}(\log N)$ FLOPs, compared to $\mathcal{O}(N^3)$ in active-set methods, and $\mathcal{O}(N)$ in common tailored interior-point method.

6.1 A parallel solution algorithm for the Newton system

The core idea behind the algorithm proposed in this section is to exploit the block-tridiagonal structure of the Newton matrix $\mathcal{M}$ in Eq. (6) in a cyclic reduction strategy. We note that a similar and to some extent more general algorithm has been proposed in [44] for the parallel solution of a linear system of similar structure, that originated form a related but slightly different problem. The work [44] contains a deeper analysis of the applicability and extensions of the cyclic reduction strategy, as well as extensive numerical comparisons. In the following we only present the central algorithmic idea in our context and refer to [44] for complementary reading. Note that in contrast to the setting of [44], we have general applicability, since $\mathcal{M}$ is positive definite and therefore the diagonal minors are invertible.

We consider the Newton system given by

$$\begin{bmatrix} D_1 & U_1 & & & & \\ U_1^\top & D_2 & U_2 & & & \\ & U_2^\top & D_3 & U_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & U_{N-2}^\top & D_{N-1} & U_{N-1} \\ & & & & U_{N-1}^\top & D_N \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \vdots \\ \lambda_{N-1} \\ \lambda_N \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ \vdots \\ g_{N-1} \\ g_N \end{bmatrix}, \tag{33}$$

where $\lambda_i \in \mathbb{R}^{n_x}$, $i \in \mathcal{S}_0$ denote the block components of λ defined in Eq. (3). We assume the Newton matrix $\mathcal{M}$ strictly positive definite (otherwise we regularize). We can use the 2nd, 4th,... equation of (33) to eliminate $\lambda_2, \lambda_4, \ldots$ concurrently from (33). This yields a reduced-size system of equations, which is again block-tridiagonal,

$$\begin{bmatrix} \bar{D}_1 & \bar{U}_1 & & & \\ \bar{U}_1^\top & \bar{D}_3 & \bar{U}_3 & & \\ & \ddots & \ddots & \ddots & \\ & & & & \bar{U}_{N-1} \\ & & & \bar{U}_{N-1}^\top & \bar{D}_N \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_3 \\ \lambda_5 \\ \vdots \\ \lambda_{N-1} \\ \lambda_N \end{bmatrix} = \begin{bmatrix} \bar{g}_1 \\ \bar{g}_3 \\ \bar{g}_5 \\ \vdots \\ \bar{g}_{N-1} \\ \bar{g}_N \end{bmatrix},$$

with block components given by

$$\bar{D}_i := D_i - U_{i-}^\top D_{i-}^{-1} U_{i-} - U_i D_{i+}^{-1} U_i^\top \tag{34a}$$

$$\bar{U}_i := -U_i^\top D_{i+}^{-1} U_{i+} \tag{34b}$$

$$\bar{g}_i := g_i - U_{i-}^\top D_{i-}^{-1} g_{i-} - U_i^\top D_{i+}^{-1} g_{i+}, \tag{34c}$$

where $i- = i-1$, and $i+ = i+1$. Applying this reduction step recursively, we obtain a system

$$\begin{bmatrix} \bar{\bar{D}}_1 & \bar{\bar{U}}_1 \\ \bar{\bar{U}}_1^\top & \bar{\bar{D}}_N \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_N \end{bmatrix} = \begin{bmatrix} \bar{\bar{g}}_1 \\ \bar{\bar{g}}_N \end{bmatrix}$$

after $\Theta(\log N)$ iterations, from which we can eliminate λ_N, yielding

$$\left(\bar{\bar{D}}_1 - \bar{\bar{U}}_1 \bar{\bar{D}}_N^{-1} \bar{\bar{U}}_1^\top\right) \lambda_1 = \bar{\bar{g}}_1 - \bar{\bar{U}}_1 \bar{\bar{D}}_N^{-1} \bar{\bar{g}}_N, \tag{35}$$

a dense system of size $n_x \times n_x$. This system can efficiently be solved by a direct Cholesky decomposition, followed by two backsolve steps.

With λ_1 we can recover λ_N from

$$\lambda_N = \bar{\bar{D}}_N^{-1} \left(\bar{\bar{g}}_N - \bar{\bar{U}}_1^\top \lambda_1\right), \tag{36}$$

and in general, we can recover λ_i from λ_{i-} and λ_{i+} by

$$\lambda_i = D_i^{-1} \left(g_i - U_{i-}^\top \lambda_{i-} - U_i \lambda_{i+}\right), \tag{37}$$

concurrently in reverse level order of the previous elimination procedure. Here λ_{i-} and λ_{i+} denote the λ-blocks preceding and succeeding λ_i in the system of equations remaining in the reduction step that eliminated λ_i.

The complete solution algorithm can be summarized as follows:

Algorithm 3: A parallel solution algorithm for Eq. (6)

Input: Newton system given by $g^{(0)} = \left[g_1^\top, \ldots, g_N^\top\right]^\top$,
$D^{(0)} = \text{blockdiag}(D_1, \ldots, D_N)$, $U^{(0)} = \text{blockdiag}(U_1, \ldots, U_{N-1})$

Output: Solution $\lambda = \left[\lambda_1^\top, \ldots, \lambda_N^\top\right]^\top$ to Equation (6)

$k_{\max} = \lceil \log_2(N-1) \rceil$
for $k = 1 : k_{\max}$ **do** `/* factor step */`
  for $i = 2^{k-1} : 2^k : N-1$ **do in parallel**
    $i- = i - 2^{k-1}$
    $i+ = \min(i + 2^k, N)$
    compute $D_i^{(k)}, U_i^{(k)}, g_i^{(k)}$ from Eq. (34) with
      $D_\bullet = D_\bullet^{(k-1)},\ U_\bullet = U_\bullet^{(k-1)},\ g_\bullet = g_\bullet^{(k-1)} \quad \forall \bullet \in \{i, i-, i+\}$
Compute λ_1 from Eq. (35) with $\bar{\bar{U}}_1 = U_1^{(k_{\max})},\ \bar{\bar{g}}_1 = g_1^{(k_{\max})}$,
  $\bar{\bar{D}}_1 = D_1^{(k_{\max})},\ \bar{\bar{D}}_N = D_N^{(k_{\max})}$ using a Cholesky decomposition
Compute λ_N from Eq. (36) with
  $\bar{\bar{D}}_N = D_N^{(k_{\max})},\ \bar{\bar{g}}_N = g_N^{(k_{\max})},\ \bar{\bar{U}}_1 = U_1^{(k_{\max})}$
for $k = k_{\max} : -1 : 1$ **do** `/* solve step */`
  for $i = 2^{k-1} : 2^k : N-1$ **do in parallel**
    recover λ_i using Eq. (37) with
      $D_i = D_i^{(k-1)},\ g_i = g_i^{(k-1)},\ U_{i-} = U_{i-}^{(k-1)},\ U_i = U_i^{(k-1)}$

Remark 14 Obviously the products of the block matrices $U_\bullet$ and $D_\bullet^{-1}$ in Equations (34) and (35) are most efficiently computed by a backsolve with a Cholesky factor of the diagonal blocks of the reduced size system, $D_\bullet$. If the system of Eq. (33) is linearly dependent, i.e., if $\mathcal{M}$ is rank deficient, the Cholesky factorization of one of these blocks will fail (otherwise, i.e., if all $D_\bullet$ have full rank (35–37) would constitute an linear injective mapping $\lambda \to \mathcal{G}$), and we can restart the factorization with a regularized $\mathcal{M}$, analogously to Sect. 3.4.

Remark 15 We note that Algorithm 3 can also be employed in the factorization step of tailored interior-point methods, thus obtaining also $\mathcal{O}(\log N)$ parallel time complexity versions of this class of methods. This has been established in [29] for a structure-exploiting variant of Mehrotra's predictor-corrector scheme [33].

7 Open-source software implementation

The dual Newton strategy has been implemented in the open-source software package qpDUNES, which is available for download at [1]. It is a plain, self-contained C code written according to the C90 standard to enlarge compatibility with embedded hardware platforms. It comes with its own linear algebra module and efficient data storage formats to better exploit the problem intrinsic structures. Memory allocation

is performed on a global scale to enable reusability of memory blocks and to enable switching between dynamic memory allocation for maximum flexibility and static memory allocation for increased performance and deployment on embedded hardware. A code generation routine for the linear algebra modules tailored to the structure and dimensions of a specific problem instance for even higher efficiency is currently under development. Application of such code generation techniques has lead to significant performance increases in related areas like interior-point solvers [13,32] and NMPC-Controllers [24].

Problems can be set up and solved from a C/C++ environment as well as conveniently from MATLAB. qpDUNES provides set-up and solve routines (both cold- and wamstarted) for multi-stage QPs, as well as for linear time-invariant and for linear time-varying MPC problems in both environments.

At the time of writing qpDUNES only supports box-constrained QPs with diagonal quadratic terms in the objective function. A version that employs a modified version of the open-source parametric active-set strategy qpOASES [15] and thus can deal with general convex multi-stage QPs is only in an experimental state, as for high efficiency, a deep integration on memory level is crucial.

8 Numerical performance

8.1 Double integrator

The first benchmark example is motivated by the energy optimal control of a cart on a rail in the context of a Badminton robot [41]. The dynamics of the system boil down to a simple double integrator with two states, position and velocity, and acceleration as control input. The optimization problem obtained after discretization is a convex QP in the form of (P) with

$$Q_k = \sigma \times \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}, \quad A_k = \begin{bmatrix} 1 & 0.01 \\ & 1 \end{bmatrix}, \quad B_k = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix}, \quad D_k = \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}$$

and

$$R_k = \begin{bmatrix} 1 \end{bmatrix}, \quad S_k = 0, \quad q_k = r_k = c_k = 0, \quad \underline{d}_k = \begin{bmatrix} -1.9 & -3 & -30 \end{bmatrix}^\top = -\overline{d}_k$$

for all stages (either $k \in \mathcal{S}$ or $k \in \mathcal{S}_0$). Additionally, we have the initial value constraint fixing $x_0 = \begin{bmatrix} -1 & 0 \end{bmatrix}^\top$ and two arrival constraints demanding the cart to arrive at position 0 at a certain index $\bar{k}$ and staying there for at least 10 ms (one time discretization step), giving the robot arm time to hit the shuttlecock. A small regularization term $\sigma > 0$ in this formulation ensures positive definiteness of the matrices Q_k.

This benchmark problem is particularly interesting, because it directly shows the limitations of the dual Newton strategy. Purely energy-minimal operation of the badminton robot would correspond to $\sigma = 0$, which results in a non-strictly convex QPs. To be able to treat this problem with the dual Newton strategy, regularization is

always required. Despite this drawback, we still believe that the dual Newton strategy is well suited to this problem, since in almost all practical applications regularization is beneficial and often even leads to more desirable properties of the obtained solutions.

Additionally to the small regularization parameter (leading to a rather badly conditioned dual function), we choose arrival times close to infeasibility, ensuring that many state constraints become active in the solution. As discussed in Sect. 3.4, scenarios with many active state constraints tentatively are particularly challenging for the dual Newton strategy. Also note in this context that the unconstrained optimum (favoring no action) lies far outside the feasible region.

We compare the average computation time over the first 20 MPC iterations again for horizon lengths of $N = 50, 100, 150, 200$, where the cart is forced to arrive at the desired position at $\bar{k} = 50$. We add random noise to the simulated system dynamics, and choose $\sigma = 10^{-4}$. If the MPC problem was rendered infeasible by the (precomputed) noise vector, we discard this instance and generate a new noise vector.

We report computation times on a standard desktop PC featuring a 3.4 GHz Intel i7 CPU under a Ubuntu 13.04 Linux for our method qpDUNES in comparison against FORCES [13], a very recent and highly efficient structure exploiting interior-point method that uses automatic code generation to create a custom solver tailored for the dynamics of a MPC problem. We run FORCES with default settings, as we did not observe any significant performance improvement in other configurations. It was indicated in [40] that FORCES will outperform even very efficient active-set methods on prediction horizons of the considered length. It was indicated in [13] that FORCES also outperforms other tailored interior-point methods rigorously.

For completeness we include a comparison against the popular solvers CPLEX 12.5.1 [25], and Matlab's quadprog. This is obviously an unfair comparison, since these two solvers are general-purpose and do not know about the sparsity patterns in advance. Still we include the results to stress the impact that structure-exploiting algorithms may have. In CPLEX both active-set methods and interior-point methods are available. We chose to use CPLEX in automatic mode (with parallelization switched off), as fixing CPLEX to either method rather lead to a performance decrease than increase. Matlab's quadprog was also run in default configuration, using its interior-point solver. We note that the active-set method was observed to perform orders of magnitude worse than the interior-point method and is therefore not included in the comparison.

All benchmarks were run from Matlab R2013a, as FORCES is currently only provided via a mex-interface. CPLEX and quadprog were called as precompiled libraries for Linux through their default Matlab interface. FORCES was downloaded through its Matlab interface as a custom solver tailored to the model dynamics. Both FORCES and qpDUNES were compiled to a `.mex` file using gcc 4.7.3 with standard code optimization options.

We perform the MPC simulation 1000 times, with different random noise vectors, and report averaged computation times in milliseconds in Fig. 2a. We observe that both customized solvers, qpDUNES and FORCES, perform observably well. Still, qpDUNES performs yet a factor of 4.9–10.6 better on this benchmark problem than the runner-up FORCES.

To get a glimpse on worst-case computation times (comparisons that tentatively favor interior-point methods), we additionally report solution times of a single QP

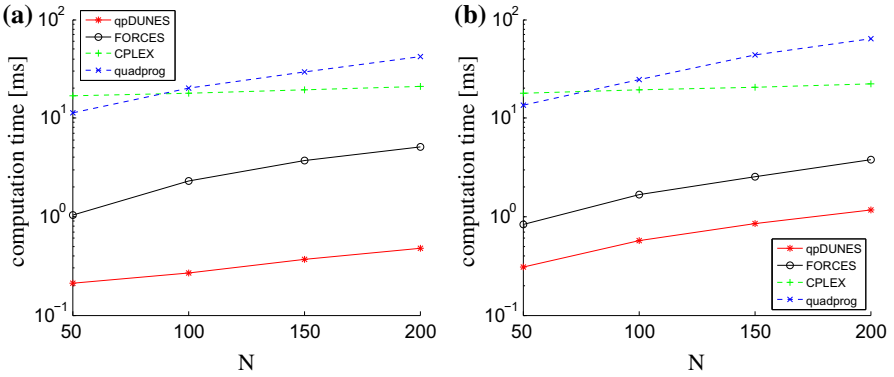


Fig. 2 Computation times of the double integrator benchmark. **a** Cold started benchmark. **b** Cold started benchmark

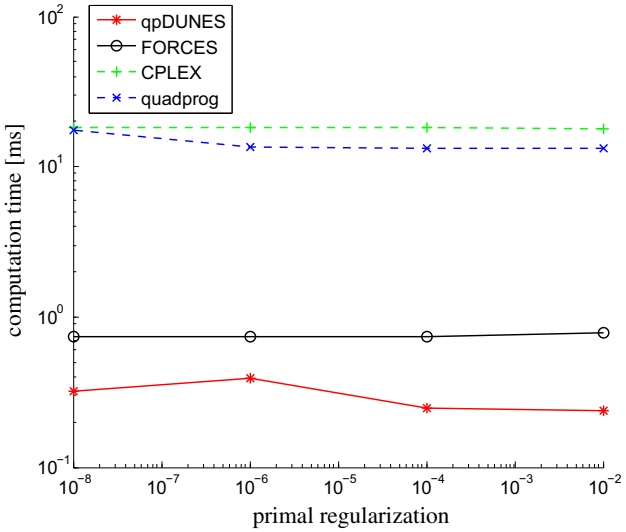
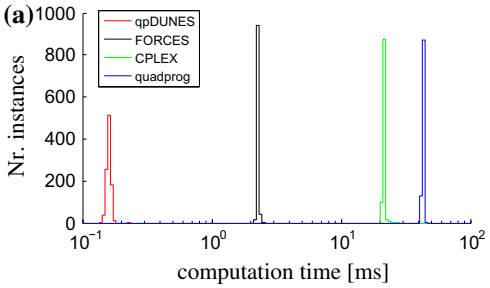


Fig. 3 Computation times of one cold started QP solution for different primal regularization parameters

without any prior knowledge about the solution (i.e., cold started). We chose horizon lengths of $N = 50, 100, 150, 200$ and forced the cart to arrive at the desired position at $\bar{k} = 45$ (almost the minimum time possible).

Even though qpDUNES is not tailored for a single QP solution, it was observed that qpDUNES outperforms the other considered solvers by a factor of 2.7–3.3 even on this benchmark scenario, cf. Fig. 2b. The numbers reported are in milliseconds, and averaged over 1000 identical (cold started) runs.

In a third comparison, we analyze the effect of different primal regularization parameters in the dual Newton strategy. For a horizon length of $N = 50$, and an arrival index of $\bar{k} = 45$, we compare values for σ between 10^{-2} and 10^{-8} (possibly causing very ill-conditioned dual functions) in Fig. 3. Again, computation times are reported



(b)

Solver	Computation time
qpDUNES	0.16 ms
FORCES	2.27 ms
CPLEX	21.41 ms
quadprog	42.83 ms

Fig. 4 Chain of masses benchmark problem. **a** Computation time histogram. Peaks from *left to right* qpDUNES, FORCES, CPLEX, quadprog. **b** Average computation times of one QP solution

in milliseconds, and averaged over 1000 identical (cold started) runs. As expected, we observe that the choice of σ barely has any influence on the performance of the interior-point methods. The dual Newton strategy in contrast is sensitive to the choice of σ, and there is a tendency to higher computational demand for smaller primal regularization parameters (i.e., a more ill-conditioned dual problem), as anticipated. Still, for all considered regularization parameters, qpDUNES outperformed its competitors.

8.2 Chain of masses

The second benchmark example is taken from [42]. MPC is used to control a chain of six oscillating masses that are connected by springs to each other and to a wall on each side. We use the same parameters as stated in [42], and thus end up with an MPC problem of 12 states, 3 control inputs and a prediction horizon of 30 intervals. We simulate the MPC problem on 100 time steps.

We computed 1000 random noise vectors that perturb the positions of the masses. We again compare the computation times obtained from qpDUNES with those obtained from the solvers FORCES, CPLEX, and quadprog.

Figure 4b shows average computation times for one QP solution, and Fig. 4a shows a histogram of the average iteration times over the different instances. It can be seen that by merit of the efficient warmstarting, qpDUNES is at least one order of magnitude faster than the other solvers considered, even in presence of perturbations. We note that qpDUNES was rarely observed to exceed 4 iterations, while FORCES needed 6–9 iterations in most cases. Since the factorization in qpDUNES can be warmstarted, and the structure of the band-matrix is similar to the one occurring in interior-point methods, each iteration in the dual Newton strategy is roughly at most as expensive as an iteration of an interior-point method. It should be noted that in this benchmark problem significantly less constraints become active, a fact from which qpDUNES benefits overproportionally.

8.3 Hanging chain

To comment on the weaknesses of the dual Newton strategy, we consider a third problem, which has been used for benchmarking in several papers before, see [14,40,

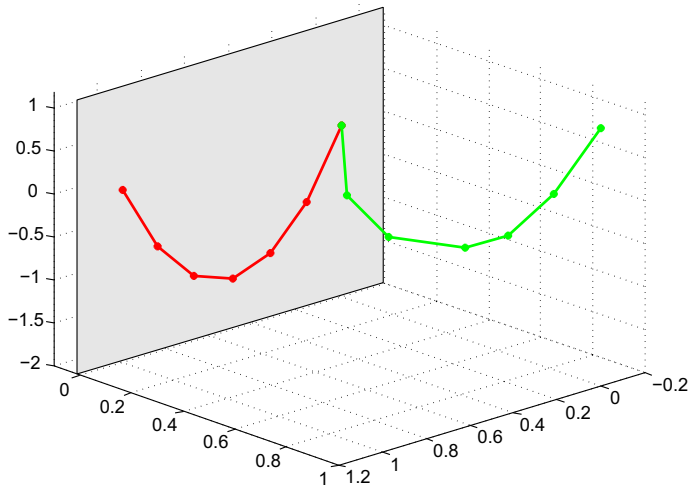


Fig. 5 Hanging chain scenario. The chain in steady state is depicted in *red* on the *left*, while the initial state is drawn in *green* on the *right* (color figure online)

43]. The problem features again a chain of masses, yet in three-dimensional space, connected by springs, that assumes its steady state very close to a wall, cf. Fig. 5. One end of the chain is fixed at the origin, while the other one is free and can be controlled by its velocities in x, y, and z direction. Note that, as in [40], we placed the wall closer to the equilibrium position than it has been considered in the original setting from [43]. This means that potentially a large amount of state constraints becomes active in the solution, and thus we yield a more challenging problem, particularly for the dual Newton strategy. As in [14] we perform linear MPC based on a linearization in the steady state, trying to stabilize the problem quickly at its equilibrium. For the detailed model equations we refer to [43].

We consider a chain of 5 Masses, which results in a system of 33 states (the free masses' positions and velocities) and 3 controls (the last masses' velocities), on a varying horizon length between 30 and 60 intervals. Noise is added on the velocities of each mass in each simulation step (adding noise directly on the positions might result in an infeasible problem very easily because of the closeness between the equilibrium and the constraining wall). We explicitly note that systems of such a ratio between state dimensionality and horizon length are not the targeted application domain for the dual Newton strategy. Still, if we consider average computation times over 50 MPC iterations, we observe that the dual Newton strategy performs reasonably well, cf. Fig. 6a. Our solver qpDUNES performs a factor of 3.4–5.6 faster in this comparison than FORCES, the tightest competitor. It is interesting to observe that the computation time is more or less constant over all considered horizon lengths. This can be explained by the efficient warmstarting of the factorization in the MPC context, cf. Sect. 3.5. Moreover, a longer horizon will cause that the equlibrium is reached (first at the end of the prediction horizon) at an earlier point in the simulation, thus making the multiplier shift more effective from this point on.

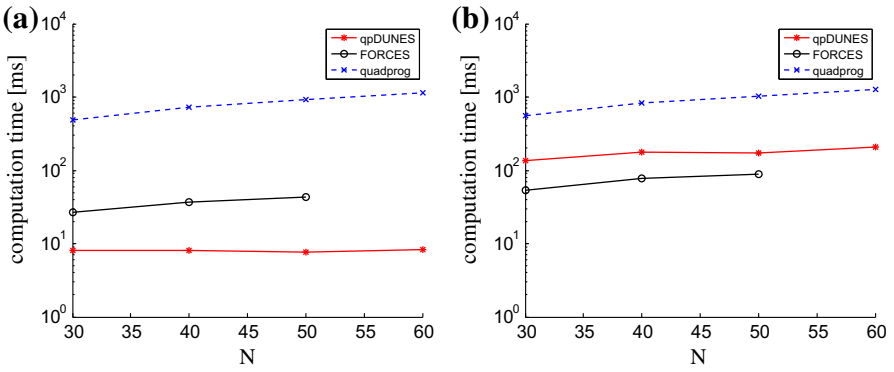


Fig. 6 The hanging chain benchmark problem. **a** Average computation times. **b** Maximum computation times

In this comparison we did not include CPLEX, as its Barrier method experienced numerical problems in almost all considered noise scenarios and therefore terminated early and suboptimally. We were also not able to obtain FORCES code for a problem of 60 or more intervals length with this number of states; note that this is not a numerical limitation of the solver, but rather a technical limitation of FORCES' download server, which generates the code specifically for each problem instance.

Despite all these shortcomings of the comparison, we believe that this very challenging problem yields some important insights when we consider maximum computation times in Fig. 6b. Here we observe that FORCES outperforms qpDUNES by a factor of 2.5–2 on the considered horizon lengths. Because the objective is driving the optimization variables to points with many state constraints active (about 70 active constraints in the solution for the $N = 50$ setting), qpDUNES takes many iterations that require regularization of the dual Hessian matrix. Overall, qpDUNES needed about twice as many iterations in the maximum than FORCES; additionally, many line search iterations were required when the dual Hessian needed to be regularized.

Nonetheless, this comparison also yields some positive news for the dual Newton strategy. We observed that the high computation times of qpDUNES were exclusively observed in the first MPC iteration, when qpDUNES was cold-started, and the initial condition is far away from the equilibrium. When excluding this first iteration from the comparison, qpDUNES outperformed FORCES again by a factor of 3–6.1 also in maximum computation times for one MPC iteration. Therefore we are still convinced that the dual Newton strategy is of practical relevance.

9 Conclusions and outlook

We presented a novel algorithm, the dual Newton strategy, for the solution of convex quadratic programming problems that arise in optimal control and estimation contexts. Besides theoretic contributions, we introduced qpDUNES, an open-source implementation of the dual Newton strategy, that yielded very promising results in comparison with state of the art solvers on a number of benchmark problems.

We envisage to utilize qpDUNES as a QP subproblem solver in an SQP procedure for nonlinear model predictive control in the near future. We further want to extend the software implementation by several features that were presented in this paper, yet are still to be implemented in an reliable fashion, like the infeasibility detection (so far only prototypic), an efficient support for more general QPs and an efficient parallel implementation.

Besides these short term goals, algorithmic extensions are conceivable. Since the dual Newton strategy can efficiently be warmstarted, one way to avoid singular Newton matrices at all would be by relaxing all stage constraints and penalizing their violation by an adaptive weighted term in the objective. Even further, the dual Newton strategy could be modified to treat general convex programming problems, using a convex subproblem solver.

Acknowledgments This research was supported by Research Council KUL: PFV/10/002 Optimization in Engineering Center OPTEC, GOA/10/09 MaNet and GOA/10/11 Global real-time optimal control of autonomous robots and mechatronic systems. Flemish Government: IOF/KP/SCORES4CHEM, FWO: PhD/postdoc grants and projects: G.0320.08 (convex MPC), G.0377.09 (Mechatronics MPC); IWT: PhD Grants, projects: SBO LeCoPro; Belgian Federal Science Policy Office: IUAP P7 (DYSCO, Dynamical systems, control and optimization, 2012–2017); EU: FP7- EMBOCON (ICT-248940), FP7-SADCO (MC ITN-264735), ERC ST HIGHWIND (259 166), Eurostars SMART, ACCM.

Appendix

Proof (Theorem 6) The proof is done by calculation. We start from the the Cholesky recursion property, apply the assumed relation between Cholesky and Riccati iterates, and transform the expression into the form of the Riccati recursion. We have

$$\begin{aligned} & X_{k-1} = W_{k-1} - U_{k-1} X_k^{-1} U_{k-1}^\top \\ \Leftrightarrow \quad & P^{-1} + CH^{-1}C^\top = CH^{-1}C^\top + EH^{-1}E^\top \\ & \qquad - EH^{-1}C^\top \left(P^{-1} + CH^{-1}C^\top\right)^{-1} CH^{-1}E^\top, \end{aligned}$$

and therefore

$$P^{-1} = EH^{-1}E^\top - EH^{-1}C^\top \left(P^{-1} + CH^{-1}C^\top\right)^{-1} CH^{-1}E^\top. \tag{38}$$

Using the Schur complement $\bar{Q} = Q - S^\top R^{-1} S$ it is well known from elementary linear algebra that the inverse H^{-1} can be expressed by

$$H^{-1} = \begin{bmatrix} Q & S^\top \\ S & R \end{bmatrix}^{-1} = \begin{bmatrix} \bar{Q}^{-1} & -\bar{Q}^{-1} S^\top R^{-1} \\ -R^{-1} S \bar{Q}^{-1} & R^{-1} + R^{-1} S \bar{Q}^{-1} S^\top R^{-1} \end{bmatrix}.$$

Using this, $C = \begin{bmatrix} A & B \end{bmatrix}$, and the special structure of $E = \begin{bmatrix} I & 0 \end{bmatrix}$, we first see that the identities

$$\begin{aligned} EH^{-1}E^\top &= \bar{Q}^{-1}, \\ CH^{-1}E^\top &= \left(A - BR^{-1}S\right)\bar{Q}^{-1} =: \bar{C}\bar{Q}^{-1} \end{aligned}$$

and

$$\begin{aligned} CH^{-1}C^\top &= \left(A - BR^{-1}S\right)\bar{Q}^{-1}\left(A^\top - S^\top R^{-1}B^\top\right) + BR^{-1}B^\top \\ &= \bar{C}\bar{Q}^{-1}\bar{C}^\top + BR^{-1}B^\top \end{aligned}$$

hold. Thus, (38) can be written as

$$P^{-1} = \bar{Q}^{-1} - \bar{Q}^{-1}\bar{C}^\top\left(P^{-1} + BR^{-1}B^\top + \bar{C}\bar{Q}^{-1}\bar{C}^\top\right)^{-1}\bar{C}\bar{Q}^{-1}.$$

Applying the Woodbury matrix identity with $Y := P^{-1} + BR^{-1}B^\top$ we can express the right-hand-side term by

$$P^{-1} = \left(\bar{Q} + \bar{C}^\top Y^{-1}\bar{C}\right)^{-1}.$$

Thus

$$\begin{aligned} P &= \bar{Q} + \bar{C}^\top Y^{-1}\bar{C} \\ &= Q - S^\top R^{-1}S + \left(A^\top - S^\top R^{-1}B^\top\right)\left(P^{-1} + BR^{-1}B^\top\right)^{-1}\left(A - BR^{-1}S\right). \end{aligned}$$

Note that the inverse matrices of Q, R, P and Y exist (and are real-valued), since we assume that QP (1) is convex. Applying the Woodbury identity once again (however, in opposite direction) on $\left(P^{-1} + BR^{-1}B^\top\right)^{-1}$, and introducing $\bar{R} := \left(R + B^\top PB\right)$, we get

$$\begin{aligned} P &= Q - S^\top R^{-1}S \\ &\quad + \left(A^\top - S^\top R^{-1}B^\top\right)\left(P - PB\left(R + B^\top PB\right)^{-1}B^\top P\right)\left(A - BR^{-1}S\right) \\ &= Q - S^\top R^{-1}S \\ &\quad + A^\top\left(P - PB\bar{R}^{-1}B^\top P\right)A \\ &\quad - S^\top R^{-1}B^\top\left(P - PB\bar{R}^{-1}B^\top P\right)A \\ &\quad - A^\top\left(P - PB\bar{R}^{-1}B^\top P\right)BR^{-1}S \\ &\quad + S^\top R^{-1}B^\top\left(P - PB\bar{R}^{-1}B^\top P\right)BR^{-1}S. \end{aligned} \tag{39}$$

Using the identity $I = \bar{R}\bar{R}^{-1} = \bar{R}^{-1}\bar{R}$, we further have

$$\begin{aligned} &S^\top R^{-1}B^\top\left(P - PB\bar{R}^{-1}B^\top P\right)BR^{-1}S - S^\top R^{-1}S \\ &\quad = S^\top R^{-1}B^\top PBR^{-1}S - S^\top R^{-1}B^\top PB\bar{R}^{-1}B^\top PBR^{-1}S - S^\top R^{-1}S \end{aligned}$$

$$
\begin{aligned}
&= S^\top R^{-1}\bar{R}\bar{R}^{-1}B^\top PBR^{-1}S \\
&\quad - S^\top R^{-1}B^\top PB\bar{R}^{-1}B^\top PBR^{-1}S - S^\top \bar{R}^{-1}\bar{R}R^{-1}S \\
&= S^\top R^{-1}\left(R + B^\top PB\right)\bar{R}^{-1}B^\top PBR^{-1}S \\
&\quad - S^\top R^{-1}B^\top PB\bar{R}^{-1}B^\top PBR^{-1}S - S^\top \bar{R}^{-1}\left(R + B^\top PB\right)R^{-1}S \\
&= S^\top R^{-1}R\bar{R}^{-1}B^\top PBR^{-1}S + S^\top R^{-1}B^\top PB\bar{R}^{-1}B^\top PBR^{-1}S \\
&\quad - S^\top R^{-1}B^\top PB\bar{R}^{-1}B^\top PBR^{-1}S \\
&\quad - S^\top \bar{R}^{-1}RR^{-1}S - S^\top \bar{R}^{-1}B^\top PBR^{-1}S \\
&= -S^\top \bar{R}^{-1}S
\end{aligned}
$$

and

$$
\begin{aligned}
&- A^\top\left(P - PB\bar{R}^{-1}B^\top P\right)BR^{-1}S \\
&\quad = -A^\top PBR^{-1}S + A^\top PB\bar{R}^{-1}B^\top PBR^{-1}S \\
&\quad = -A^\top PB\bar{R}^{-1}\bar{R}R^{-1}S + A^\top PB\bar{R}^{-1}B^\top PBR^{-1}S \\
&\quad = -A^\top PB\bar{R}^{-1}\left(R + B^\top PB\right)R^{-1}S + A^\top PB\bar{R}^{-1}B^\top PBR^{-1}S \\
&\quad = -A^\top PB\bar{R}^{-1}RR^{-1}S - A^\top PB\bar{R}^{-1}B^\top PBR^{-1}S + A^\top PB\bar{R}^{-1}B^\top PBR^{-1}S \\
&\quad = -A^\top PB\bar{R}^{-1}S.
\end{aligned}
$$

Analogously,

$$
-S^\top R^{-1}B^\top\left(P - PB\bar{R}^{-1}B^\top P\right)A = -S^\top \bar{R}^{-1}B^\top PA.
$$

Therefore (39) is equivalent to

$$
\begin{aligned}
P &= Q - S^\top \bar{R}^{-1}S \\
&\quad - S^\top \bar{R}^{-1}B^\top PA - A^\top PB\bar{R}^{-1}S + A^\top\left(P - PB\bar{R}^{-1}B^\top P\right)A \\
&= Q + A^\top PA - \left(S^\top + A^\top PB\right)\bar{R}^{-1}\left(S + B^\top PA\right),
\end{aligned}
$$

which concludes the proof. □

Proof (Corollary 7) The proof follows exactly the lines of the proof to Theorem 6, but keeps the matrix block indices. In particular, one has

$$
\begin{aligned}
P_{k-1}^{-1} &= E_{k-1}H_{k-1}^{-1}E_{k-1}^\top - E_{k-1}H_{k-1}^{-1}C_{k-1}^\top\left(P_k^{-1} + C_{k-1}H_{k-1}^{-1}C_{k-1}^\top\right)^{-1} \\
&\quad \times C_{k-1}H_{k-1}^{-1}E_{k-1}^\top
\end{aligned}
$$

in place of (38) and transforms it into (19b) using the same matrix identities. □

References

1. qpDUNES Website. http://mathopt.de/qpdunes 2012–2014
2. Andersson, J.: A general-purpose software framework for dynamic optimization. PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center, Kasteelpark Arenberg 10, 3001-Heverlee, Belgium, Oct (2013)
3. Andersson, J.A.E., Frasch, J.V., Vukov, M., Diehl, M.: A condensing algorithm for nonlinear MPC with a quadratic runtime in horizon length. Automatica (2013, under review)
4. Bemporad, A., Patrinos, P.: Simple and certifiable quadratic programming algorithms for embedded linear model predictive control. In: Proceedings of the 4th IFAC Nonlinear Model Predictive Control Conference, pp 14–20, (2012)
5. Berkelaar, A.B., Roos, K., Terlaky, T.: Recent advances in sensitivity analysis and parametric programming. In: The Optimal Set and Optimal Partition Approach to Linear and Quadratic Programming. Kluwer Publishers, Dordrecht (1997)
6. Bertsekas, D.P., Tsitsiklis, J.N.: Parallel and Distributed Computation: Numerical Methods. Prentice Hall, New Jersey (1989)
7. Best M.J.: Applied mathematics and parallel computing. In: An Algorithm for the Solution of the Parametric Quadratic Programming Problem, pp. 57–76. Physica-Verlag, Heidelberg (1996)
8. Biegler, L.T.: Advances in nonlinear programming concepts for process control. J. Process Control **8**(5), 301–311 (1998)
9. Bock, H.G.: Recent advances in parameter identification techniques for ODE. In: Deuflhard, P., Hairer, E. (eds.) Numerical Treatment of Inverse Problems in Differential and Integral Equations. Birkhäuser, Boston (1983)
10. Dai, Y.-H., Fletcher, R.: New algorithms for singly linearly constrained quadratic programs subject to low and upper bounds. Math. Program. **106**(3), 403–421 (2006)
11. Danskin, J.M.: The Theory of Max-Min and Its Application to Weapons Allocation Problems. Springer-Verlag, Berlin, New York (1967)
12. Diehl, M., Bock, H.G., Schlöder, J.P., Findeisen, R., Nagy, Z., Allgöwer, F.: Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. J. Process Control **12**(4), 577–585 (2002)
13. Domahidi, A., Zgraggen, A., Zeilinger, M.N., Morari, M., Jones, C.N.: Efficient interior point methods for multistage problems arising in receding horizon control. In: IEEE Conference on Decision and Control (CDC), pp. 668–674. Maui (2012)
14. Ferreau, H.J., Bock, H.G., Diehl, M.: An online active set strategy to overcome the limitations of explicit MPC. Int. J. Robust Nonlinear Control **18**(8), 816–830 (2008)
15. Ferreau, H.J., Kirches, C., Potschka, A., Bock, H.G., Diehl, M.: qpOASES: a parametric active-set algorithm for quadratic programming. Math. Program. Comput. (2013)
16. Ferreau, H.J., Kozma, A., Diehl, M.: A parallel active-set strategy to solve sparse parametric quadratic programs arising in MPC. In: Proceedings of the 4th IFAC Nonlinear Model Predictive Control Conference, pp. 74–79 (2012)
17. Fiacco, A.V.: Introduction to Sensitivity and Stability Analysis in Nonlinear Programming. Academic Press, New York (1983)
18. Fletcher, R.: Practical Methods of Optimization, 2nd edn. Wiley, Chichester (1987)
19. Frasch, J.V., Vukov, M., Ferreau, H.J., Diehl, M.: A new quadratic programming strategy for efficient sparsity exploitation in SQP-based nonlinear MPC and MHE. In: Proceedings of the 19th IFAC World Congress (2014)
20. Frison G., Jorgensen J.: A fast condensing method for solution of linear-quadratic control problems. In: Proceedings of the 52nd IEEE Conference on Decision and Control (2013)
21. Gerdts, M., Kunkel, M.: A nonsmooth Newton's method for discretized optimal control problems with state and control constraints. J. Indus. Manage. Optim. **4**, 247–270 (2008)
22. Gill, P.E., Murray, W., Wright, M.H.: Practical Optimization. Academic Press, London (1981)
23. Gill, P.E., Murray, W.: Numerically stable methods for quadratic programming. Math. Program. **14**, 349–372 (1978)
24. Houska, B., Ferreau, H.J., Diehl, M.: An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range. Automatica **47**(10), 2279–2285 (2011)
25. IBM Corp. IBM ILOG CPLEX V12.1, User's Manual for CPLEX (2009)

26. Kirches, C., Bock, H.G., Schlöder, J.P., Sager, S.: Block structured quadratic programming for the direct multiple shooting method for optimal control. Optim. Methods Softw. **26**, 239–257 (2010)
27. Kirches, C., Wirsching, L., Bock, H.G., Schlöder, J.P.: Efficient direct multiple shooting for nonlinear model predictive control on long horizons. J. Process Control **22**(3), 540–550 (2012)
28. Kühl, P., Diehl, M., Kraus, T., Schlöder, J.P., Bock, H.G.: A real-time algorithm for moving horizon state and parameter estimation. Comput. Chem. Eng. **35**(1), 71–83 (2011)
29. Do Duc, L.: Parallelisierung von Innere-Punkte-Verfahren mittels Cyclic Reduction. Bachelor's thesis, OVGU Magdeburg (2014)
30. Leineweber, D.B.: Efficient Reduced SQP Methods for the Optimization of Chemical Processes Described by Large Sparse DAE Models. Fortschritt-Berichte VDI Reihe 3, Verfahrenstechnik, vol. 613. VDI Verlag, Düsseldorf (1999)
31. Li, W., Swetits, J.: A new algorithm for solving strictly convex quadratic programs. SIAM J. Optim. **7**(3), 595–619 (1997)
32. Mattingley, J., Boyd, S.: Convex Optimization in Signal Processing and Communications, chapter Automatic Code Generation for Real-Time Convex Optimization. Cambridge University Press, Cambridge (2009)
33. Mehrotra, S.: On the implementation of a primal-dual interior point method. SIAM J. Optim. **2**(4), 575–601 (1992)
34. Nesterov, Y.: Introductory Lectures on Convex Optimization: A Basic Course, vol. 87 of Applied Optimization. Kluwer Academic Publishers, Amsterdam (2004)
35. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Series in Operations Research and Financial Engineering, 2nd edn. Springer, New York (2006)
36. Qi, L., Sun, J.: A nonsmooth version of Newton's method. Math. Program. **58**, 353–367 (1993)
37. Rao, C.V., Wright, S.J., Rawlings, J.B.: Application of interior-point methods to model predictive control. J. Optim. Theory Appl. **99**, 723–757 (1998)
38. Rawlings, J.B., Mayne, D.Q.: Model Predictive Control: Theory and Design. Nob Hill, San Francisco (2009)
39. Richter, S., Jones, C.N., Morari, M.: Real-time input-constrained MPC using fast gradient methods. In: Proceedings of the IEEE Conference on Decision and Control, Shanghai, China (2009)
40. Vukov, M., Domahidi, A., Ferreau, H.J., Morari, M., Diehl, M.: Auto-generated algorithms for nonlinear model predicitive control on long and on short horizons. In: Proceedings of the 52nd Conference on Decision and Control (CDC) (2013)
41. Wang, X., Stoev, J., Pinte, G., Swevers, J.: Energy optimal point-to-point motion using model predictive control. In: Proceedings ASME 2012 5th Annual Dynamic Systems and Control Conference (2012)
42. Wang, Y., Boyd, S.: Fast model predictive control using online optimization. IEEE Trans. Control Syst. Technol. **18**(2), 267–278 (2010)
43. Wirsching, L., Bock, H.G., Diehl, M.: Fast NMPC of a chain of masses connected by springs. In: Proceedings of the IEEE International Conference on Control Applications, Munich, pp. 591–596 (2006)
44. Wright, S.: Partitioned dynamic programming for optimal control. SIAM J. Optim. **1**(4), 620–642 (1991)
45. Zafiriou, E.: Robust model predictive control of processes with hard constraints. Comput. Chem. Eng. **14**(4–5), 359–371 (1990)