On a Primal-Dual Newton Proximal Method for Convex Quadratic Programs

Alberto De Marchi[*]

2020-12-12

Abstract

This paper introduces QPDO, a primal-dual method for convex quadratic programs which builds upon and weaves together the proximal point algorithm and a damped semismooth Newton's method. The outer proximal regularization yields a numerically stable method, and we interpret the proximal operator as the unconstrained minimization of the primal-dual proximal augmented Lagrangian function. This allows the inner Newton's scheme to exploit sparse symmetric linear solvers and multi-rank factorization updates. Moreover, the linear systems are always solvable independently from the problem data and exact linesearch can be performed. The proposed method can handle degenerate problems, provides a mechanism for infeasibility detection, and can exploit warm starting, while requiring only convexity. We present details of our open-source C implementation and report on numerical results against state-of-the-art solvers. QPDO proves to be a simple, robust, and efficient numerical method for convex quadratic programming.

Keywords: Semismooth Newton's methods, Proximal point methods, Regularized primal-dual methods, Convex quadratic programming.

[*]Institute for Applied Mathematics and Scientific Computing, Department of Aerospace Engineering, Universität der Bundeswehr München, Germany. E-MAIL: alberto.demarchi@unibw.de, ORCID: 0000-0002-3545-6898.

1 Introduction

Quadratic programs (QPs) are one of the fundamental problems in optimization. In this paper, we consider linearly constrained convex QPs, in the form:

$$\min_{\mathbf{x}} \quad \frac{1}{2}\mathbf{x}^\top \boldsymbol{Q}\mathbf{x} + \mathbf{q}^\top \mathbf{x}, \quad \text{s. t.} \quad \mathbf{l} \leq \boldsymbol{A}\mathbf{x} \leq \mathbf{u}, \tag{1}$$

with $\mathbf{x} \in \mathbb{R}^n$. Matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ and vector $\mathbf{q} \in \mathbb{R}^n$ define the objective function, whereas the constraints are encoded by matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and vectors $\mathbf{l}, \mathbf{u} \in \overline{\mathbb{R}}^m$. We assume (i) the problem data $\boldsymbol{Q}$, $\mathbf{q}$, and $\boldsymbol{A}$ are bounded, (ii) matrix $\boldsymbol{Q}$ is symmetric positive semidefinite, i.e., $\boldsymbol{Q} \succeq 0$, and (iii) vectors $\mathbf{l}$ and $\mathbf{u}$ satisfy $\mathbf{l} \leq \mathbf{u}$, $\mathbf{l} < +\infty$, and $\mathbf{u} > -\infty$ component-wise; cf. [55, 30]. We will refer to the nonempty, closed and convex set

$$\mathcal{C} := \{\mathbf{z} \in \mathbb{R}^m : \mathbf{l} \leq \mathbf{z} \leq \mathbf{u}\}$$

as the constraint set. Note that (1) represents a general convex QP, in that it accomodates also equality constraints and bounds.

1.1 Background

Optimization problems in the form (1) appear in a variety of applications and are of interest in engineering, statistics, finance and many other fields. QPs often arise as sub-problems in methods for general nonlinear programming [9, 29, 40], and greatly vary in terms of problem size and structure.

Convex QPs have been studied since the 1950s [20] and several numerical methods have been developed since then. These differ in how they balance the number of iterations and the cost (e.g., run time) per iteration.

Active-set methods for QPs originated from extending the simplex method for linear programs (LPs) [60]. These methods select a set of binding constraints and iteratively adapt it, seeking the set of active constraints at the solution. Active-set algorithms can be easily warm started and can lead to finite convergence. Moreover, by adding and dropping constraints from the set of binding ones, factorization updates can be adopted for solving successive linear systems. However, these methods may require many iterations to identify the correct set of active constraints. Modern solvers based on active-set methods are qpOASES [18] and NASOQ [12].

First-order methods iteratively compute an optimal solution using only first-order information about the cost function [45, 42]. As these methods consist of computationally cheap and simple steps, they are well suited to applications with limited computing resources [55]. However, first-order algorithms usually require many iterations to achieve accurate solutions and may suffer from ill-conditioning of the problem data. Several acceleration schemes have been proposed to improve their behaviour [1, 58]. The OSQP [55] solver offers an implementation based on ADMM [8].

Interior-point methods consider the problem constraints in the objective function via barrier functions and solve a sequence of parametric sub-problems [9, Chap. 11], [40, §16.6]. Although not easily warm started, the polynomial complexity makes interior-point methods appealing for large scale problems [27]. They usually require few but rather demanding iterations [29, 40]. Recent developments are found in the regularized method IP-PMM [47].

Semismooth Newton's methods apply a nonsmooth version of Newton's method to the KKT conditions of the original problem [49, 48]. In the strictly convex case, i.e., with $\boldsymbol{Q} \succ 0$, this approach performs very well as long as the underlying linear systems are nonsingular. Regularized, or stabilized, semismooth Newton-type methods, such as QPALM [30, 31] and FBstab [34], overcome these drawbacks.

The augmented Lagrangian framework [13, 6, 40], semismooth Newton's methods [49, 23], and proximal techniques [53, 44] are undergoing a revival, as their seamless combination exhibits valuable properties and provides useful features, such as regularization and numerical stability [5, 30, 34]. These ideas form the basis for our approach.

1.2 Approach

In this work we present a numerical method for solving general QPs. The proposed algorithm is based on the proximal point algorithm and a semismooth Newton's method for solving the sub-problems, which are always solvable for any choice of problem data. We therefore impose no restrictions such as strict convexity of the cost function or linear independence of the constraints. As such, our algorithm gathers together the benefits of fully regularized primal-dual methods and semismooth Newton's methods with active-set structure. Our algorithm can exploit warm starting to reduce the number of iterations, as well as factorization caching and multi-rank update techniques for efficiency, and it can obtain accurate solutions.

Our approach, dubbed QPDO from "Quadratic Primal-Dual Optimizer", is inspired by and shares many characteristics with algorithms that have already been proposed, in particular with QPALM [30] and FBstab [34]. On the other hand, they differ on some key aspects. QPALM relates to the proximal method of multipliers [30, Rem. 2], which in turn is associated to the classical (primal) augmented Lagrangian function [52]. Instead, FBstab and QPDO apply the proximal point method, yielding exact primal-dual regularization. However, FBstab reformulates the sub-problem via the (penalized) Fischer-Burmeister NCP function [19, 10], and adopts the squared residual norm as a merit function for the inner iterative loop; this prevents the use of symmetric sparse linear solvers. Instead, QPDO adopts the minimum NCP function, which leads to symmetric linear systems with active-set structure. Then, we show the primal-dual proximal augmented Lagrangian function, introduced in [50, 25] and [15], is a suitable merit function for the proximal sub-problem, which allows us to perform an exact linesearch in a fully primal-dual regularized context. Indeed, we believe, the main contribution of this work consists in recognizing this link, exploiting it to bridge the gap between previously proposed methods, and developing a robust and efficient algorithm that possesses their advantages but does not suffer from their disadvantages.

Notation Before completing this section, let us introduce some notation. $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$, $\mathbb{R}_+$, and $\mathbb{R}_{++}$ denote the sets of natural, integer, real, non-negative real, and positive real numbers, respectively. We denote $\overline{\mathbb{R}} := \mathbb{R} \cup \{-\infty, \infty\}$ the extended real line. The identity matrix and the vector of ones of size n are denoted by $\boldsymbol{I}_n$ and $\mathbf{1}_n$, respectively. We may omit subscripts whenever clear from the context. $[a, b]$, (a, b), $[a, b)$, and $(a, b]$ stand for closed, open, and half-open intervals, respectively, with end points a and b. $[a; b]$, $(a; b)$, $[a; b)$, and $(a; b]$ stand for discrete intervals, e.g., $[a; b] = [a, b] \cap \mathbb{Z}$. Given a vector $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x}^\top$ and $\mathbf{x}^i$ denote its transpose and its i-th component, respectively. We adopt the norms $\|\mathbf{x}\| = \|\mathbf{x}\|_2 := \sqrt{\mathbf{x}^\top \mathbf{x}}$ and $\|\mathbf{x}\|_\infty := \max_{i \in [1;n]} |\mathbf{x}^i|$. Given a set S, $|S|$ denotes its cardinality. In $\mathbb{R}^n$, the relations $<$, $\leq$, $=$, $\geq$, and $>$ are understood component-wise. Given a nonempty closed convex set $\mathcal{C} \subseteq \mathbb{R}^n$, we denote $\chi_\mathcal{C} : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ its characteristic function, namely $\chi_\mathcal{C}(\mathbf{x}) = 0$ if $\mathbf{x} \in \mathcal{C}$ and $\chi_\mathcal{C}(\mathbf{x}) = +\infty$ otherwise, $\mathrm{dist}_\mathcal{C} : \mathbb{R}^n \to \mathbb{R}$ its distance, namely $\mathbf{x} \mapsto \min_{\mathbf{z} \in \mathcal{C}} \|\mathbf{z} - \mathbf{x}\|$, and its projection $\Pi_\mathcal{C} : \mathbb{R}^n \to \mathbb{R}^n$, namely $\mathbf{x} \mapsto \arg\min_{\mathbf{z} \in \mathcal{C}} \|\mathbf{z} - \mathbf{x}\|$.

The algorithm is described with a nested structure: outer and inner iterations are indexed by $k \in \mathbb{N}$, respectively. Given an arbitrary vector $\mathbf{x}$, $\mathbf{x}_k$ denotes that $\mathbf{x}$ depends on k, and analogously for matrices. We denote $\mathbf{y}$ the dual variable associated to the constraints in problem (1). A primal-dual pair $(\mathbf{x}, \mathbf{y})$ will be denoted by $\mathbf{v}$, and we will refer interchangeably to it as a vector or to its components $\mathbf{x}$ and $\mathbf{y}$. An optimal solution to the problem (1) will be denoted as $(\mathbf{x}^\star, \mathbf{y}^\star)$, or $\mathbf{v}^\star$. Optimal solutions of proximal sub-problems will be denoted using an appropriate subscript, according to the iteration. For example, $(\mathbf{x}_k^\star, \mathbf{y}_k^\star)$, and $\mathbf{v}_k^\star$, denote the solution to the proximal sub-problem corresponding to the k-th outer iteration.

Outline The rest of the paper is organized as follows. In Section 2, we outline our algorithmic framework. Sections 3 and 4 develop and present our algorithm in details. In particular, in Section 4.1 we establish our key result, which relates the proximal operator and the primal-dual proximal augmented Lagrangian function. Convergence properties are analyzed in Section 5, while Section 6 juxtaposes QPDO with similar methods. We present details of our implementation in Section 7 and report on numerical experience in Section 8.

2 Algorithm

In this section, we outline our Quadratic Primal-Dual Optimizer (QPDO), which weaves together the proximal point algorithm and a semismooth Newton's method. Effectively, the proximal operator is evaluated by solving a sub-problem via semismooth Newton's method. Thus, the latter constitutes a inner iterative procedure, embedded into the outer proximal point loop. The proposed numerical scheme is sketched in Algorithms 1 and 2, highlighting the nested structure for clarity of presentation. We denote $\mathbf{r}$ and $\mathbf{r}_k$ the outer and inner residuals defined in (4) and (11), respectively, and $\mathbf{v}$ a primal-dual pair $(\mathbf{x}, \mathbf{y})$. We provide more details and investigate its convergence properties in the following sections.

3 Outer loop: inexact proximal point method

Our method solves problem (1) using the proximal point algorithm, with inexact evaluation of the proximal operator. In Algorithm 1, this is evaluated by means of a semismooth Newton-type method, which constitutes a inner iterative procedure, further investigated in Section 4. This section focuses on the outer loop corresponding to the proximal point algorithm, which has

Algorithm 1 QPDO: Quadratic Primal-Dual Optimizer

```
input: $\boldsymbol{Q}$, $\mathbf{q}$, $\boldsymbol{A}$, $\mathbf{l}$, $\mathbf{u}$
parameters: $\epsilon > 0$, $\epsilon_0 \geq 0$, $\kappa_\epsilon \in [0,1)$, $0 < \sigma_{\min} \leq \sigma_0$, $0 < \mu_{\min} \leq \mu_0$
guess: $\mathbf{x}_0 \in \mathbb{R}^n$, $\mathbf{y}_0 \in \mathbb{R}^m$
for $k = 0, 1, 2, \dots$ do
    if $\|\mathbf{r}(\mathbf{v}_k)\|_\infty \leq \epsilon$ then
        return $\mathbf{v}_k$
    end if
    find $\mathbf{v}_{k+1}$ such that $\|\mathbf{r}_k(\mathbf{v}_{k+1})\|_\infty \leq \epsilon_k$ by invoking Algorithm 2
    choose $\sigma_{k+1} \in [\sigma_{\min}, \sigma_k]$ and $\mu_{k+1} \in [\mu_{\min}, \mu_k]$
    set $\epsilon_{k+1} \leftarrow \kappa_\epsilon \epsilon_k$
end for
```

Algorithm 2 Inner loop: semismooth Newton's method

```
$\mathbf{v} \leftarrow \mathbf{v}_k$
repeat
    get $\delta\mathbf{v}$ by solving the linear system (17)
    get $\tau$ by solving the piecewise linear equation (18)
    set $\mathbf{v} \leftarrow \mathbf{v} + \tau\,\delta\mathbf{v}$
until $\|\mathbf{r}_k(\mathbf{v})\|_\infty \leq \epsilon_k$
$\mathbf{v}_{k+1} \leftarrow \mathbf{v}$
```

been extensively studied in the literature [53]. We recall some important results and refer to [37, 52, 35, 41] for more details.

3.1 Optimality conditions

Problem (1) can be equivalently expressed as

$$\min_{\mathbf{x}} \quad f(\mathbf{x}) + g(\boldsymbol{A}\mathbf{x}), \tag{2}$$

where

$$f(\mathbf{x}) := \frac{1}{2}\mathbf{x}^\top \boldsymbol{Q}\mathbf{x} + \mathbf{q}^\top \mathbf{x} \qquad \text{and} \qquad g(\mathbf{z}) := \chi_{\mathcal{C}}(\mathbf{z})$$

are the objective function and the characteristic function of the constraint set $\mathcal{C}$, respectively. The necessary and sufficient, first-order optimality conditions of problem (2), and hence problem (1), read

$$\mathbf{0} \in \mathcal{T}(\mathbf{v}) := \begin{pmatrix} \boldsymbol{Q}\mathbf{x} + \mathbf{q} + \boldsymbol{A}^\top \mathbf{y} \\ -\boldsymbol{A}\mathbf{x} + \partial g^*(\mathbf{y}) \end{pmatrix}, \tag{3}$$

where ∂g^* denotes the conjugate subdifferential of g [41]. We will refer to vector $\mathbf{y} \in \mathbb{R}^m$ as the dual variable and to $\mathcal{T} : \mathbb{R}^\ell \rightrightarrows \mathbb{R}^\ell$, $\ell := n + m$, as the KKT operator for problem (1). These optimality conditions, in the form (3), involve the set-valued operator $\mathcal{T}$. However, noticing that, for any $\alpha > 0$, the conditions $\mathbf{v} = \Pi_{\mathcal{C}}(\mathbf{v} + \alpha\mathbf{u})$ and $\mathbf{v} \in \partial g^*(\mathbf{u})$ are equivalent [51, §23], conditions in (3) can be equivalently rewritten. Choosing $\alpha = 1$, we can define the (outer) residual $\mathbf{r} : \mathbb{R}^\ell \to \mathbb{R}^\ell$ and express the KKT conditions for (1) as

$$\mathbf{0} = \mathbf{r}(\mathbf{v}) := \begin{pmatrix} \boldsymbol{Q}\mathbf{x} + \mathbf{q} + \boldsymbol{A}^\top \mathbf{y} \\ \boldsymbol{A}\mathbf{x} - \Pi_{\mathcal{C}}(\boldsymbol{A}\mathbf{x} + \mathbf{y}) \end{pmatrix}. \tag{4}$$

This reformulation can be obtained also by employing the minimum NCP function [56] and rearranging to obtain the projection operator $\Pi_{\mathcal{C}}$. The residual $\mathbf{r}$ is analogous to the natural residual function $\boldsymbol{\pi}$ investigated in [43]. Since it is an error bound for problem (1), i.e., $\operatorname{dist}_{\mathcal{T}^{-1}(\mathbf{0})}(\mathbf{v}) = O(\|\mathbf{r}(\mathbf{v})\|)$ [43, Thm 18], the norm of $\mathbf{r}$ is a sensitive optimality measure and its value can be adopted as a stopping criterion.

3.2 Proximal point algorithm

The proximal point algorithm [53] finds zeros of maximal monotone operators by recursively applying their proximal operator. Since $\mathcal{T}$ is a maximal monotone operator [52, 41], the proximal point algorithm converges to an element $\mathbf{v}^\star$ of the set of primal-dual solutions $\mathcal{T}^{-1}(\mathbf{0})$, if any exists [37, 53]. Starting from an initial guess $\mathbf{v}_0$, it generates a sequence $\{\mathbf{v}_k\}$ of primal-dual pairs by recursively applying the proximal operator $\mathcal{P}_k$:

$$\mathbf{v}_{k+1} = \mathcal{P}_k(\mathbf{v}_k), \qquad \mathcal{P}_k := (\boldsymbol{I} + \boldsymbol{\Sigma}_k^{-1}\mathcal{T})^{-1}. \tag{5}$$

Here, $\{\boldsymbol{\Sigma}_k\}$ is a sequence of non-increasing positive definite matrices, namely, $\boldsymbol{\Sigma}_k \succ 0$ and $\boldsymbol{\Sigma}_k - \boldsymbol{\Sigma}_{k+1} \succeq 0$ for all $k \in \mathbb{N}$. The matrices $\boldsymbol{\Sigma}_k$ control the primal-dual proximal regularization and, similarly to exact penalty methods, these are not required to vanish [52, 53]. Since $\mathcal{T}$ is maximally monotone, the proximal operator $\mathcal{P}_k$ is well defined and single valued for all $\mathbf{v} \in \operatorname{dom} \mathcal{T} = \mathbb{R}^\ell$ [37]. Thus, from (5), evaluating the proximal operator $\mathcal{P}_k$ at $\mathbf{v}_k$ is equivalent to finding the unique $\mathbf{v} \in \mathbb{R}^\ell$ that satisfies

$$\mathbf{0} \in \mathcal{T}_k(\mathbf{v}) := \mathcal{T}(\mathbf{v}) + \boldsymbol{\Sigma}_k(\mathbf{v} - \mathbf{v}_k). \tag{6}$$

This is guaranteed to have a unique solution and to satisfy certain useful regularity properties; see Section 4 below. As a result, we can construct a fast inner solver for these sub-problems based on semismooth Newton's method.

3.3 Early termination

The proximal point algorithm tolerates errors, namely the inexact evaluation of the proximal operator $\mathcal{P}_k$ [53]. Criterion (A_r) in [35] provides conditions for the design of convergent inexact proximal point algorithms [35, Thm 2.1]. Let $\mathbf{v}_k^\star := \mathcal{P}_k(\mathbf{v}_k)$ denote the unique proximal sub-problem solution and $\mathbf{v}_{k+1} \approx \mathbf{v}_k^\star$ the actual recurrence update. Then, the aforementioned criterion requires

$$\|\mathbf{v}_{k+1} - \mathbf{v}_k^\star\| \leq e_k \min\left(1, \|\mathbf{v}_{k+1} - \mathbf{v}_k\|^r\right),$$

where $r \geq 0$ and $\{e_k\}$ is a summable sequence of nonnegative inner tolerances, i.e., $e_k \geq 0$ for all k and $\sum_{k=0}^\infty e_k < +\infty$. However, since $\mathbf{v}_k^\star$ is effectively unknown, this criterion is impractical in its form. Instead, in Algorithm 1 it is required that $\mathbf{v}_{k+1}$ satisfies $\|\mathbf{r}_k(\mathbf{v}_{k+1})\|_\infty \leq \epsilon_k$. Here, $\mathbf{r}_k$ denotes the residual for the k-th sub-problem, and is defined in (11). In Section 5 we will show that this criterion is a simple and viable substitute, which retains the significance of (A_r).

3.4 Warm starting

If a solution $\mathbf{v}^\star$ exists, the (outer) sequence $\{\mathbf{v}_k\}$ generated by (5) converges, by the global convergence of the proximal point algorithm [53]. Then, expectedly, $\mathcal{P}_k(\mathbf{v}_k)$ and $\mathbf{v}_k$ are arbitrarily close for sufficiently large k [34, §4]. This supports the idea of warm starting the inner solver with the current outer estimate $\mathbf{v}_k$, that is, setting $\mathbf{v} \leftarrow \mathbf{v}_k$ in Algorithm 2. For large k, only one or few Newton-type inner iterations are needed to find an approximate sub-problem solution $\mathbf{v}_{k+1}$.

4 Inner loop: semismooth Newton's method

In this section we focus on solving sub-problem (6) via a semismooth Newton's method. For the sake of clarity, and without loss of generality, we consider

$$\boldsymbol{\Sigma}_k := \operatorname{blockdiag}(\sigma_k \boldsymbol{I}_n, \mu_k \boldsymbol{I}_m).$$

for some parameters $\sigma_k, \mu_k \in \mathbb{R}_{++}$.

4.1 Merit function

We now derive the simple yet fundamental result that is the key to develop our method. This provides the NCP reformulation of the proximal sub-problem with a suitable merit function. The former yields symmetric active-set linear systems, while the latter leads to exact linesearch.

Let us express the proximal sub-problem (6) in the form

$$\mathbf{0} \in \begin{pmatrix} \boldsymbol{Q}\mathbf{x} + \mathbf{q} + \boldsymbol{A}^\top \mathbf{y} + \sigma_k(\mathbf{x} - \mathbf{x}_k) \\ -\boldsymbol{A}\mathbf{x} + \mu_k(\mathbf{y} - \mathbf{y}_k) + \partial g^*(\mathbf{y}) \end{pmatrix}. \tag{7}$$

Similarly to (4), for any given $\alpha > 0$, this can be rewritten as

$$\mathbf{0} = \begin{pmatrix} \boldsymbol{Q}\mathbf{x} + \mathbf{q} + \boldsymbol{A}^\top \mathbf{y} + \sigma_k(\mathbf{x} - \mathbf{x}_k) \\ \boldsymbol{A}\mathbf{x} + \mu_k(\mathbf{y}_k - \mathbf{y}) - \Pi_{\mathcal{C}}(\mathbf{w}_k) \end{pmatrix}, \tag{8}$$

where we denote

$$\mathbf{w}_k := \boldsymbol{A}\mathbf{x} + \mu_k(\mathbf{y}_k - \mathbf{y}) + \alpha \mathbf{y}. \tag{9}$$

Then, for any positive $\alpha \neq \mu_k$, the conditions in (8) are equivalent to

$$\mathbf{0} = \begin{pmatrix} \boldsymbol{Q}\mathbf{x} + \mathbf{q} + \frac{1}{\alpha}\boldsymbol{A}^\top[\mathbf{w}_k - \Pi_{\mathcal{C}}(\mathbf{w}_k)] + \sigma_k(\mathbf{x} - \mathbf{x}_k) \\ \frac{\alpha - \mu_k}{\alpha}[\mathbf{w}_k - \Pi_{\mathcal{C}}(\mathbf{w}_k)] + (\mu_k - \alpha)\mathbf{y} \end{pmatrix}, \tag{10}$$

namely their unique solutions coincide. Now, we observe that the right-hand side of (10) is the gradient of the function

$$f(\mathbf{x}) + \frac{1}{2\alpha} \operatorname{dist}^2_{\mathcal{C}}(\mathbf{w}_k) + \frac{\sigma_k}{2}\|\mathbf{x} - \mathbf{x}_k\|^2 + \frac{\mu_k - \alpha}{2}\|\mathbf{y}\|^2.$$

By construction, this is a continuously differentiable function whose gradient vanishes at the unique solution of the proximal sub-problem. Furthermore, for any $\alpha \in (0, \mu_k)$, it is strictly convex and hence admits a unique minimizer. This must coincide with the unique proximal point. Therefore, this function is a suitable merit function for the sub-problem. The particular choice $\alpha := \mu_k/2$ inherits all these properties and leads to the inner optimality conditions

$$\mathbf{0} = \mathbf{r}_k(\mathbf{v}) := \begin{pmatrix} \boldsymbol{Q}\mathbf{x} + \mathbf{q} + \boldsymbol{A}^\top \mathbf{y} + \sigma_k(\mathbf{x} - \mathbf{x}_k) \\ \boldsymbol{A}\mathbf{x} + \mu_k(\mathbf{y}_k - \mathbf{y}) - \Pi_{\mathcal{C}}(\boldsymbol{A}\mathbf{x} + \mu_k(\mathbf{y}_k - \mathbf{y}/2)) \end{pmatrix}, \tag{11}$$

with $\mathbf{r}_k : \mathbb{R}^\ell \to \mathbb{R}^\ell$ the inner residual, and the associated merit function

$$\mathcal{M}_k(\mathbf{v}) := f(\mathbf{x}) + \frac{1}{\mu_k}\operatorname{dist}^2_{\mathcal{C}}(\boldsymbol{A}\mathbf{x} + \mu_k(\mathbf{y}_k - \mathbf{y}/2)) + \frac{\sigma_k}{2}\|\mathbf{x} - \mathbf{x}_k\|^2 + \frac{\mu_k}{4}\|\mathbf{y}\|^2. \tag{12}$$

In fact, $\mathcal{M}_k : \mathbb{R}^\ell \to \mathbb{R}$ is the primal-dual proximal augmented Lagrangian function [50, 25, 15]; see Appendix A for a detailed derivation. This underlines once again the strong relationship between the proximal point algorithm and the augmented Lagrangian framework, pioneered in [52]. On the one hand, by (12), the dual regularization parameter μ_k controls the constraint penalization [21, §3.2]. On the other hand, this provides an "interpretation of the augmented Lagrangian method as an adaptive constraint regularization process" [3, §2].

The inner residual $\mathbf{r}_k$ in (11) is piecewise affine, hence strongly semismooth on $\mathbb{R}^\ell$ [48, 33]. Effectively, it can be employed as stopping criterion in place of $\|\nabla \mathcal{M}_k(\cdot)\|$. In fact, given the unique, bounded, and nonsingular matrix $\boldsymbol{T}_k$ defined by

$$\boldsymbol{T}_k := \left[\begin{array}{c|c} \boldsymbol{I} & \frac{2}{\mu_k}\boldsymbol{A}^\top \\ \hline \mathbf{0} & -\boldsymbol{I} \end{array}\right], \tag{13}$$

we have the identity $\nabla \mathcal{M}_k(\cdot) = \boldsymbol{T}_k \mathbf{r}_k(\cdot)$.

The availability of a suitable merit function allows us to adopt a damped Newton-type method and design a linesearch-based globalization strategy, in contrast with [46, 23, 34]. Since $\mathcal{M}_k$ is continuously differentiable and piecewise quadratic, an exact linesearch procedure can be carried out, which yields finite convergence [57].

Finally, we highlight that the method asymptotically reduces to a sequence of regularized semismooth Newton's steps applied to the original, unperturbed optimality system, on the vein of [2]. This closely relates to the concept of exact regularization [22]. In fact, the proximal primal-dual regularization is exact; see Proposition 1 and compare [3, Thm 1].

Proposition 1. *Let $k \in \mathbb{N}$ be arbitrary.*

(i) Suppose $\mathbf{v}_k^\star$ solves sub-problem (11) for $\mathbf{v}_k := \mathbf{v}_k^\star$ and for some $\sigma_k \geq 0$ and $\mu_k > 0$. Then, $\mathbf{v}_k^\star$ solves the original problem (4).

(ii) Alternatively, suppose $\mathbf{v}_k^\star$ solves sub-problem (11) for $\mathbf{y}_k := \mathbf{y}_k^\star$, $\sigma_k := 0$, and for some $\mu_k > 0$. Then, $\mathbf{v}_k^\star$ solves the original problem (4).

(iii) Conversely, suppose $\mathbf{v}^\star$ solves the original problem (4). Then, $\mathbf{v}^\star$ solves the sub-problem (11) for $\mathbf{v}_k := \mathbf{v}^\star$ and for any $\sigma_k \geq 0$ and $\mu_k > 0$.

Proof. The proof is immediate by direct comparison of (4) and (11). □

4.2 Search direction

A semismooth Newton's direction $\delta\mathbf{v} = (\delta\mathbf{x}, \delta\mathbf{y})$ at $\mathbf{v} = (\mathbf{x}, \mathbf{y})$ solves

$$\boldsymbol{V}_k(\mathbf{v})\delta\mathbf{v} = -\mathbf{r}_k(\mathbf{v}). \tag{14}$$

Here, the matrix $\boldsymbol{V}_k(\mathbf{v})$ is an element of the generalized Jacobian [51, §23] of $\mathbf{r}_k$ at $\mathbf{v}$, which has the form

$$\boldsymbol{V}_k(\mathbf{v}) = \left[\begin{array}{c|c} \boldsymbol{Q} + \sigma_k \boldsymbol{I} & \boldsymbol{A}^\top \\ \hline (\boldsymbol{I} - \boldsymbol{P}_k(\mathbf{v}))\boldsymbol{A} & -\mu_k(\boldsymbol{I} - \boldsymbol{P}_k(\mathbf{v})/2) \end{array}\right]. \tag{15}$$

In turn, the diagonal matrix $\boldsymbol{P}_k(\mathbf{v})$ with entries

$$\boldsymbol{P}_k^{ii}(\mathbf{v}) := \begin{cases} 1 & \text{if } \mathbf{l}^i < \mathbf{w}_k^i < \mathbf{u}^i \\ 0 & \text{otherwise} \end{cases}, \quad i = 1, \dots, m, \tag{16}$$

is an element of the generalized Jacobian of $\Pi_{\mathcal{C}}$ at $\mathbf{w}_k$. Owing to the selection of $\boldsymbol{P}_k(\mathbf{v})$ with binary entries, the linear system (14) can be rewritten in symmetric form, similar to those arising in active-set methods [32]. To this end, we notice that, if $\boldsymbol{P}_k^{ii}(\mathbf{v}) = 1$, the corresponding inner residual in (11) simplifies into $\mathbf{r}_k^{n+i}(\mathbf{v}) = -\mu_k \mathbf{y}^i/2$, and the linear equation in (14) gives $\delta\mathbf{y}^i = -\mathbf{y}^i$. This yields the crucial observation that, by (16), it holds $\boldsymbol{P}_k(\mathbf{v})\delta\mathbf{y} = -\boldsymbol{P}_k(\mathbf{v})\mathbf{y}$ for all $\mathbf{v} \in \mathbb{R}^\ell$. Then, an equivalent yet symmetric linear system is obtained, whose solution is the search direction $\delta\mathbf{v}$ at $\mathbf{v}$:

$$\left[\begin{array}{c|c} \boldsymbol{Q} + \sigma_k \boldsymbol{I} & \boldsymbol{A}^\top(\boldsymbol{I} - \boldsymbol{P}_k(\mathbf{v})) \\ \hline (\boldsymbol{I} - \boldsymbol{P}_k(\mathbf{v}))\boldsymbol{A} & -\mu_k(\boldsymbol{I} - \boldsymbol{P}_k(\mathbf{v})/2) \end{array}\right] \begin{pmatrix} \delta\mathbf{x} \\ \delta\mathbf{y} \end{pmatrix} = \begin{pmatrix} \boldsymbol{A}^\top \boldsymbol{P}_k(\mathbf{v})\mathbf{y} \\ \mathbf{0} \end{pmatrix} - \mathbf{r}_k(\mathbf{v}). \tag{17}$$

The active-set structure introduced by $\boldsymbol{P}_k$ allows us to obtain a symmetric linear system and adopt multi-rank factorization updates [24, 14] while maintaining structure and sparsity of the coefficient matrix [55, 12]. The linear system in (17) always admits a unique solution, since the coefficient matrix is symmetric quasi-definite [59], independent of the problem data.

4.3 Exact linesearch

Given a primal-dual pair $\mathbf{v}$ and a search direction $\delta\mathbf{v}$, we seek a stepsize $\tau > 0$ to effectively update $\mathbf{v}$ to $\mathbf{v} + \tau\,\delta\mathbf{v}$ in Algorithm 2. Similarly to $\mathcal{M}_k$, the function $\psi_k : \tau \mapsto \mathcal{M}_k(\mathbf{v} + \tau\delta\mathbf{v})$ is continuously differentiable, piecewise quadratic, and strictly convex. Thus, the optimal stepsize $\tau := \arg\min_{t\in\mathbb{R}} \psi_k(t)$ is found as the unique zero of ψ_k', i.e., $\psi_k'(\tau) = 0$. Since ψ_k' is a piecewise linear, strictly monotone increasing function, the exact linesearch procedure amounts to solving a piecewise linear equation of the form

$$0 = \alpha_k \tau + \beta_k + \frac{2}{\mu_k}\delta\mathbf{w}^\top \left[\mathbf{w}_k + \tau\delta\mathbf{w}_k - \Pi_{\mathcal{C}}\left(\mathbf{w}_k + \tau\delta\mathbf{w}_k\right)\right] \tag{18}$$

with respect to $\tau \in \mathbb{R}$. Here, the coefficients are given by

$$\alpha_k := \delta\mathbf{x}^\top(\boldsymbol{Q} + \sigma_k \boldsymbol{I})\delta\mathbf{x} + \mu_k \delta\mathbf{y}^\top \delta\mathbf{y}/2, \tag{19a}$$

$$\beta_k := \delta\mathbf{x}^\top[\boldsymbol{Q}\mathbf{x} + \mathbf{q} + \sigma_k(\mathbf{x} - \mathbf{x}_k)] + \mu_k \delta\mathbf{y}^\top \mathbf{y}/2, \tag{19b}$$

$$\mathbf{w}_k := \boldsymbol{A}\mathbf{x} + \mu_k\left(\mathbf{y}_k - \mathbf{y}/2\right), \tag{19c}$$

$$\delta\mathbf{w}_k := \boldsymbol{A}\delta\mathbf{x} - \mu_k \delta\mathbf{y}/2, \tag{19d}$$

whose derivation is reported in Appendix B. Thanks to its peculiar structure, (18) can be solved efficiently and exactly (up to numerical precision), e.g., by sorting and linear interpolation, cf. [30, Alg. 2].

We underline that the stepsize τ is unique and strictly positive, since $\mathcal{M}_k$ is strictly convex and $\delta\mathbf{v}$ is a descent direction for $\mathcal{M}_k$ at $\mathbf{v}$. This follows from the observation that

$$\psi_k'(0) = \delta\mathbf{v}^\top \nabla\mathcal{M}_k(\mathbf{v}) = \delta\mathbf{v}^\top \boldsymbol{T}_k \mathbf{r}_k(\mathbf{v}) = -\delta\mathbf{v}^\top \boldsymbol{T}_k \boldsymbol{V}_k(\mathbf{v})\delta\mathbf{v} < 0,$$

since $\partial^2\mathcal{M}_k(\mathbf{v}) \ni \boldsymbol{T}_k\boldsymbol{V}_k(\mathbf{v}) \succ 0$.

5 Convergence analysis

This section discusses the convergence of QPDO as outlined in Algorithms 1 and 2. Our analysis relies on well-established results for Newton's and proximal point methods; in particular, we refer to [53, 35, 57].

First, we focus on the inner loop, described in Algorithm 2 and detailed in Section 4. Since linear system (17) is always solvable, the search direction $\delta\mathbf{v}$ exists and is unique. Similarly, there exists a unique, positive optimal stepsize τ which solves (18). Thus, all steps are well-defined. It remains to show that the condition $\|\mathbf{r}_k(\mathbf{v})\|_\infty \leq \epsilon_k$ is eventually satisfied. Since $\mathcal{M}_k$ is continuously differentiable, strictly convex, and piecewise quadratic, the semismooth Newton's method with exact linesearch exhibits finite convergence [57, Thm 3]. Thus, $\nabla\mathcal{M}_k(\mathbf{v}) = \mathbf{0}$ after finitely many iterations. Then, by $\nabla\mathcal{M}_k(\cdot) = \boldsymbol{T}_k\mathbf{r}_k(\cdot)$ with $\boldsymbol{T}_k$ nonsingular, it reaches $\mathbf{r}_k(\mathbf{v}) = \mathbf{0}$. Hence, for any $\epsilon_k > 0$, the inner stopping criterion is eventually satisfied, and the inner loop terminates.

Let us consider now the outer loop, sketched in Algorithm 1. This consists of inexact proximal point iterations [53], hence global and local convergence properties of the outer loop can be derived based on [35, Thm 2.1]. Recall that, by construction, the regularization parameters are positive and non-increasing. Also, by $\epsilon_0 \in \mathbb{R}_+$ and $\kappa_\epsilon \in [0, 1)$, the sequence $\{\epsilon_k\} \subseteq \mathbb{R}_+$ is summable, since $\sum_{k\in\mathbb{N}} \epsilon_k = \sum_{k\in\mathbb{N}} \kappa_\epsilon^k \epsilon_0 = \epsilon_0/(1-\kappa_\epsilon) < +\infty$. The following result shows that criterion (A_r) [35] holds.

Lemma 1. *Let $\mathcal{T}^{-1}(\mathbf{0})$ be nonempty, any $\mathbf{v}_0 \in \mathbb{R}^\ell$ be given, and the sequence $\{\mathbf{v}_k\}$ be generated by Algorithm 1. Then, there exists a summable sequence $\{e_k\} \subseteq \mathbb{R}_+$ such that*

$$\|\mathbf{v}_{k+1} - \mathbf{v}_k^\star\| \leq e_k \quad \forall k.$$

Proof. By the inner stopping condition, for all $k \in \mathbb{N}$ it holds $\|\mathbf{r}_k(\mathbf{v}_{k+1})\| \leq \epsilon_k$, with summable $\{\epsilon_k\} \subseteq \mathbb{R}_+$. Morever, since $\mathcal{M}_k$ is Σ_k-strongly convex, we have that, for some $\tilde{\eta}_k > 0$, it is

$$\tilde{\eta}_k \|\mathbf{v} - \mathbf{v}_k^\star\| \leq \|\nabla\mathcal{M}(\mathbf{v}) - \nabla\mathcal{M}(\mathbf{v}_k^\star)\| = \|\nabla\mathcal{M}(\mathbf{v})\| = \|\boldsymbol{T}_k\,\mathbf{r}_k(\mathbf{v})\|$$

for all $\mathbf{v} \in \mathbb{R}^\ell$. By the boundedness of $\boldsymbol{\Sigma}_k$ away from zero, matrix $\boldsymbol{T}_k$ is bounded and there exists a constant $\eta > 0$ such that the bound $\|\mathbf{v} - \mathbf{v}_k^\star\| \leq \eta\|\mathbf{r}_k(\mathbf{v})\|$ holds for all $k \in \mathbb{N}$ and $\mathbf{v} \in \mathbb{R}^\ell$. Thus, in particular, for all $k \in \mathbb{N}$ it is

$$\|\mathbf{v}_{k+1} - \mathbf{v}_k^\star\| \leq \eta\|\mathbf{r}_k(\mathbf{v}_{k+1})\| \leq \eta\epsilon_k.$$

Let $e_k := \eta\epsilon_k$, and the proof is complete. □

Notice that we choose $r = 0$ in (A_r) for the sake of simplicity, although this may prevent faster convergence [35]. Then, since problem (6) is a polyhedral variational inequality [17, §3D], we can invoke [35, Thm 2.1].

Theorem 1. *Let $\mathcal{T}^{-1}(\mathbf{0})$ be nonempty, any $\mathbf{v}_0 \in \mathbb{R}^\ell$ be given, and the sequence $\{\mathbf{v}_k\}$ be generated by Algorithm 1. Then, the sequence $\{\mathbf{v}_k\}$ is well-defined and converges linearly to a solution $\mathbf{v}^\star \in \mathcal{T}^{-1}(\mathbf{0})$.*

6 Relationship with similar methods

Our approach is inspired by and shares many features with other recently developed methods. This section elaborates upon their relationship with QPDO.

FBstab [34] "synergistically combines the proximal point algorithm with a primal-dual semismooth Newton-type method" to solve convex QPs. By adopting the Fischer-Burmeister [19, 10] NCP function, FBstab does not depend on an estimate of the active set, which may result in a more regular behavior than QPDO. In contrast, adopting the minimum NCP function, QPDO can exploit factorization updates, perform exact line search by solving a piecewise linear equation, and handle simultaneously bilateral constraints.

QPALM is a "proximal augmented Lagrangian based solver for convex QPs" [30]; recent advancements allow to handle nonconvex QPs as well [31]. Given a primal-dual estimate $\overline{\mathbf{v}}$, the exact, unique resolvent update $\mathbf{v}^\triangle$ of QPALM [30, Eq. 6], with $\boldsymbol{\Sigma} = \text{blockdiag}(\sigma^{-1}\boldsymbol{I}, \mu^{-1}\boldsymbol{I})$, is given by

$$\mathbf{x}^\triangle = \arg\min\nolimits_{\mathbf{x}\in\mathbb{R}^n}\ \varphi(\mathbf{x}), \tag{20a}$$

$$\mathbf{y}^\triangle = \overline{\mathbf{y}} + \mu^{-1}\left[\boldsymbol{A}\mathbf{x}^\triangle - \Pi_{\mathcal{C}}\left(\boldsymbol{A}\mathbf{x}^\triangle + \mu\overline{\mathbf{y}}\right)\right]. \tag{20b}$$

Herein function φ is given by [30, Eq. 8]

$$\varphi(\mathbf{x}) := f(\mathbf{x}) + \frac{1}{2\mu}\,\text{dist}_{\mathcal{C}}^2\left(\boldsymbol{A}\mathbf{x} + \mu\overline{\mathbf{y}}\right) + \frac{\sigma}{2}\|\mathbf{x} - \overline{\mathbf{x}}\|^2$$

and closely resembles $\mathcal{M}_k$ in (12). Since (20a) yields $\nabla\varphi(\mathbf{x}^\triangle) = \mathbf{0}$, combining with (20b) and rearranging give

$$\mathbf{0} = \boldsymbol{Q}\mathbf{x}^\triangle + \mathbf{q} + \boldsymbol{A}^\top\mathbf{y}^\triangle + \sigma\left(\mathbf{x}^\triangle - \overline{\mathbf{x}}\right), \tag{21a}$$

$$\mathbf{0} = \boldsymbol{A}\mathbf{x}^\triangle + \mu\left(\overline{\mathbf{y}} - \mathbf{y}^\triangle\right) - \Pi_{\mathcal{C}}\left(\boldsymbol{A}\mathbf{x}^\triangle + \mu\overline{\mathbf{y}}\right). \tag{21b}$$

Conditions (21) and (11) differ only in the argument of $\Pi_{\mathcal{C}}$, where the term $-\mu\mathbf{y}/2$ is missing in (21b). This underlines the primal-dual nature of QPDO, that may better cope with changes in the active set [32] and control the quality of both primal and dual variables during iterations [26, 2], without any additional computational effort.

OSQP is a "solver for convex quadratic programs based on the alternating direction method of multipliers" [55]. Rearranging from [55, Alg. 1], with parameters $\alpha = 1$, $\rho = \mu^{-1}$, and given primal-dual estimate $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$ and constraint estimate $\overline{\mathbf{z}}$, the (unique) primal-auxiliary update $(\mathbf{x}^\diamondsuit, \mathbf{s}^\diamondsuit)$ satisfies

$$\mathbf{0} = \boldsymbol{Q}\mathbf{x}^\diamondsuit + \mathbf{q} + \boldsymbol{A}^\top\mathbf{s}^\diamondsuit + \sigma(\mathbf{x}^\diamondsuit - \overline{\mathbf{x}}), \tag{22a}$$

$$\mathbf{0} = \boldsymbol{A}\mathbf{x}^\diamondsuit + \mu(\overline{\mathbf{y}} - \mathbf{s}^\diamondsuit) - \overline{\mathbf{z}}. \tag{22b}$$

Then, the constraint and dual updates are given by $\mathbf{z}^\diamondsuit = \Pi_{\mathcal{C}}\left(\overline{\mathbf{z}} + \mu\mathbf{s}^\diamondsuit\right)$ and $\mathbf{y}^\diamondsuit = \mathbf{s}^\diamondsuit + \mu^{-1}\left(\overline{\mathbf{z}} - \mathbf{z}^\diamondsuit\right)$, respectively. Although conditions (22) resemble (11), an auxiliary variable $\mathbf{s}$ substitutes the dual variable $\mathbf{y}$ and the projection in (11) is replaced by the constraint estimate $\overline{\mathbf{z}}$. This makes sub-problem (22) a linear system and results in a first-order method.

7 Implementation details

QPDO has been implemented in C code[1] and provides an interface to MATLAB. This section discusses some relevant aspects of the program, such as the linear solver, parameters update rules, infeasibility detection, and problem scaling.

7.1 Linear solver

The linear system (17) is solved with CHOLMOD [11]. This linear solver is analogous to the one adopted in QPALM [30], for the sake of comparison. Let $(\mathbf{r}_k^{\text{dual}}, \mathbf{r}_k^{\text{prim}})$ partition the inner residual $\mathbf{r}_k$ in (11). Then, formally solving for $\delta\mathbf{y}$ in (17), we obtain the expression (omitting subscripts and arguments)

$$\begin{aligned}
\delta\mathbf{y} &= \mu^{-1}(\boldsymbol{I} - \boldsymbol{P}/2)^{-1}\left[(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{A}\delta\mathbf{x} + \mathbf{r}^{\text{prim}}\right] \\
&= \mu^{-1}(\boldsymbol{I} + \boldsymbol{P})\left[(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{A}\delta\mathbf{x} + \mathbf{r}^{\text{prim}}\right] \\
&= \mu^{-1}(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{A}\delta\mathbf{x} + \mu^{-1}(\boldsymbol{I} + \boldsymbol{P})\mathbf{r}^{\text{prim}},
\end{aligned}$$

[1]Available online at https://github.com/aldma/qpdo.

where the second and third lines are due to the binary structure of $\boldsymbol{P}$. Substituting $\delta\mathbf{y}$ and rearranging, we obtain a linear system for $\delta\mathbf{x}$:

$$\left[\boldsymbol{Q} + \sigma\boldsymbol{I} + \mu^{-1}\boldsymbol{A}^\top(\boldsymbol{I}-\boldsymbol{P})\boldsymbol{A}\right]\delta\mathbf{x} = \boldsymbol{A}^\top\boldsymbol{P}\mathbf{y} - \mu^{-1}\boldsymbol{A}^\top(\boldsymbol{I}-\boldsymbol{P})\mathbf{r}^{\text{prim}} - \mathbf{r}^{\text{dual}}.$$

This has a symmetric, positive definite coefficient matrix and can be solved by CHOLMOD [11]. On the one hand, this approach allows multi-rank factorization updates [14], thus avoiding the need for a full re-factorization at every inner iteration. On the other hand, sparsity pattern may be lost and significant fill-in may arise due to the matrix-matrix product $\boldsymbol{A}^\top\boldsymbol{A}$. For this reason, the current implementation may benefit from solving (17) via sparse symmetric linear solvers with multi-rank factorization updates, which is a topic for future research.

7.2 Parameters selection

Solving convex QPs via the proximal point algorithm imposes mild restrictions on the sequence of primal-dual regularization parameters $\{\boldsymbol{\Sigma}_k\}$. As mentioned in Section 3.2, there are no additional requirements other than being non-increasing and positive definite. However, similarly to forcing sequences in augmented Lagrangian methods [13], the sequence of regularization parameters greatly affects the behaviour of QPDO, and a careful tuning can positively impact the performance. For instance, although faster convergence rates can be expected if $\boldsymbol{\Sigma}_k \to \mathbf{0}$ [35], numerical stability and machine precision should be taken into account. Following [31, §5.3] and [55, §5.2], our implementation considers only diagonal matrices of the form $\boldsymbol{\Sigma}_k = \text{blockdiag}(\sigma_k\boldsymbol{I}, \text{diag}(\boldsymbol{\mu}_k))$, and refer to the effect of σ_k and $\boldsymbol{\mu}_k$ as primal and dual regularization, respectively.

Dual regularization This term proves critical for the practical performance of the method. We argue, these parameters have such impact since they strike the balance between the number of inner and outer iterations, seeking easy-to-solve sub-problems, effective warm starting, or rapid constraints satisfaction. Therefore, we carefully initialize and update these parameters, guided by the interpretation as a constraint penalization offered by the augmented Lagrangian framework; cf. Section 4.1. In our implementation, we consider a vector $\boldsymbol{\mu}_k$ to gain a finer control over the constraint penalization [13]. Given a (primal) initial guess $\mathbf{x}_0 \in \mathbb{R}^n$, we initialize as in [7, §12.4]:

$$\begin{aligned}
\mathbf{d}_0 &:= \boldsymbol{A}\mathbf{x}_0 - \Pi_{\mathcal{C}}(\boldsymbol{A}\mathbf{x}_0),\\
\boldsymbol{\mu}_0^i &:= \Pi_{[\mu_0^{\min},\mu_0^{\max}]}\left(\kappa_\mu \frac{\max(1,(\mathbf{d}_0^i)^2/2)}{\max(1,|f(\mathbf{x}_0)|)}\right), \; i \in [1;m],
\end{aligned}$$

where $\mu_0^{\max} \geq \mu_0^{\min} > 0$ and $\kappa_\mu \geq 0$. Then, following [31, §5.3], we monitor the primal residual $\mathbf{r}_{\text{prim}}(\mathbf{v}) := \boldsymbol{A}\mathbf{x} - \Pi_{\mathcal{C}}(\boldsymbol{A}\mathbf{x}+\mathbf{y})$ from (4) and update the dual regularization parameter $\boldsymbol{\mu}_k$ accordingly. If $|\mathbf{r}^i_{\text{prim}}(\mathbf{v}_{k+1})| > \max\left(\theta_\mu|\mathbf{r}^i_{\text{prim}}(\mathbf{v}_k)|, \epsilon_{\text{opt}}\right)$, we set

$$\boldsymbol{\mu}^i_{k+1} = \Pi_{[\mu_{\min},\boldsymbol{\mu}^i_k]}\left(\delta_\mu \frac{\|\mathbf{r}_{\text{prim}}(\mathbf{v}_{k+1})\|_\infty}{|\mathbf{r}^i_{\text{prim}}(\mathbf{v}_{k+1})|}\boldsymbol{\mu}^i_k\right),$$

where $\theta_\mu \in (0,1)$, $\mu_{\min} > 0$, and $\delta_\mu \geq 0$. Otherwise, we set $\boldsymbol{\mu}_{k+1}^i = \boldsymbol{\mu}_k^i$. These rules adapt the constraint penalization on the current residual, seeking a uniform, steady progression towards feasibility, while making sure the sequences $\{\boldsymbol{\mu}_k^i\}$, $i \in [1;m]$, are non-increasing and bounded away from zero. In our implementation, the default values are $\mu_0^{\min} = 10^{-4}$, $\mu_0^{\max} = 10^4$, $\kappa_\mu = 0.1$, $\mu_{\min} = 10^{-8}$, $\delta_\mu = 10^{-2}$ and $\theta_\mu = 0.1$.

Primal regularization This term turns out to be less crucial with respect to the dual counterpart. For this reason, it is associated to a scalar value and tuned independently from the residual. Starting from $\sigma_0 > 0$, we apply

$$\sigma_{k+1} = \max(\sigma_{\min}, \kappa_\sigma \sigma_k),$$

where $\sigma_{\min} > 0$ and $\kappa_\sigma \in [0,1]$. In our implementation the default values are $\sigma_0 = 0.1$, $\sigma_{\min} = 10^{-7}$, and $\kappa_\sigma = 0.1$.

Early termination The inner tolerance ϵ_k also affects the performance of QPDO, since it balances sub-problem accuracy and early termination. In Algorithm 1, these aspects relate to the parameters ϵ_0 and κ_ϵ, which drive $\{\epsilon_k\}$ to zero. However, finite precision should also be taken into account. In fact, although the semismooth Newton's method converges in finitely many iterations, the solution provided is exact up to round-off errors and numerical precision. Therefore, we deviate from Algorithm 1 in this respect and employ the update rule

$$\epsilon_{k+1} = \max(\epsilon_{\min}, \kappa_\epsilon \epsilon_k),$$

where $0 \leq \epsilon_{\min} \leq \epsilon_{\text{opt}}$. In our implementation, the default values are $\epsilon_0 = 1$, $\kappa_\epsilon = 0.1$, $\epsilon_{\min} = 10^{-14}$, and $\epsilon_{\text{opt}} = 10^{-6}$.

7.3 Infeasibility detection

A routine for detecting primal and dual infeasibility of problem (1) is included in Algorithm 1. This allows the algorithm to terminate with either a primal-dual solution or a certificate of primal or dual infeasibility, for some given tolerances. We adopt the mechanism developed in [4, §5.2], which holds whenever the proximal point algorithm is employed to solve the KKT conditions (3). Problem (1) is declared primal or dual infeasible based on some conditions on $\Delta\mathbf{x}_k := \mathbf{x}_{k+1} - \mathbf{x}_k$ and $\Delta\mathbf{y}_k := \mathbf{y}_{k+1} - \mathbf{y}_k$, $k \geq 0$. The reader may refer to [55, §3.4], [30, §V.C], and [34, §4.1], and [47, §4] for analogous applications.

7.4 Preconditioning

Preconditioning, or scaling, the problem may alleviate ill-conditioning and mitigate numerical issues, especially when the problem data span across many orders of magnitude. In our implementation, we closely follow [31, §5.2] and scale the problem data by performing the Ruiz's equilibration procedure [54] on the constraint matrix $\boldsymbol{A}$. This procedure iteratively scales the rows and columns of a matrix in order to make their infinite norms approach one. By default, QPDO performs 10 scaling iterations. Slightly different routines are adopted, e.g., in [55,

§5.1] and [47, §5.1.2]. Note that, by default, if the problem is initially scaled, the termination conditions for both, optimality and infeasibility, refer to the original, unscaled problem.

8 Numerical results

We discuss details of our open-source implementation of QPDO and present computational results on random problems and the Maros–Mészáros set [36]. We test and compare QPDO against the open-source, full-fledged solvers OSQP [55] and QPALM [30, 31]. Indeed, "the construction of appropriate software is by no means trivial and we wish to make a thorough job of it" [13]; we plan to improve our current implementation and to report comprehensive numerical results in due time.

8.1 Setup

We consider the tolerance $\epsilon_{\text{opt}} = 10^{-5}$, and set the tolerances in OSQP and QPALM to $\epsilon_{\text{abs}} = \epsilon_{\text{opt}}$ and $\epsilon_{\text{rel}} = 0$. In addition, we set the maximum number of iterations and the time limit to 10^{12} and 100 s, respectively, for every solver, and we leave all the other settings to the internal defaults. It is worth mentioning that, since no initial guess is provided, all the solvers start with $\mathbf{v}_0 = \mathbf{0}$. We deem optimal a primal-dual pair $\mathbf{v}^\star$ returned by a solver if it satisfies the condition $\|\mathbf{r}(\mathbf{v}^\star)\|_\infty \leq \epsilon_{\text{opt}}$, otherwise we consider it a failure.

The code to generate the numerical results is currently available upon reasonable request to the author. All the experiments were carried out on a desktop running Ubuntu 16.04 with Intel Core i7-8700 and 16 GB RAM.

Metrics Let S, P, and $t_{s,p}$ denote the set of solvers, the set of problems, and the time required for solver $s \in S$ to return a solution for problem $p \in P$. The shifted geometric mean (sgm) $\widehat{t}_s$ of the run times for solver $s \in S$ on P is defined by

$$\widehat{t}_s := \exp\left(\frac{1}{|P|}\sum_{p\in P} \ln\left(t_{s,p} + t_{\text{shift}}\right)\right) - t_{\text{shift}}$$

with the shift $t_{\text{shift}} = 1$ s [38]. Here, when solver s fails to solve problem p, the term $t_{s,p}$ is set to the time limit. We also adopt the performance profiles [16] to compare the solver timings. These plot the function $f_s^{\text{r}} : \mathbb{R} \to [0,1]$, $s \in S$, defined by

$$f_s^{\text{r}}(\tau) := \frac{|\{p \in P : t_{s,p} \leq \tau\, t_p^{\min}\}|}{|P|}, \qquad t_p^{\min} := \min_{s\in S} t_{s,p}.$$

Considering $t_{s,p} = +\infty$ when solver s fails on problem p, $f_s^{\text{r}}(\tau)$ is the fraction of problems solved by solver s within τ times the best timing. Since performance profiles may be misleading when more than two solvers are compared [28], we will compare QPDO to QPALM and OSQP separately.

Table 1: Comparison on different problem classes with timings, as shifted geometric mean, and failure rates.

			QPDO	QPALM	OSQP
Random QPs	Timing (sgm)	[s]	0.090	0.079	0.112
	Failure rate	[%]	0.00	0.00	0.00
Random Eq. QPs	Timing (sgm)	[s]	0.988	1.045	1.648
	Failure rate	[%]	0.00	0.00	4.00
Maros–Mészáros	Timing (sgm)	[s]	0.061	0.435	3.141
	Failure rate	[%]	0.00	6.85	24.66

Furthermore, performance profiles do not provide the percentage of problems that can be solved (for some given tolerance ϵ_{opt}) within a given time t. Thus, on the vein of data profiles [39, §2.2], we plot the function $f_s^{\text{a}} : \mathbb{R} \to [0,1]$, $s \in S$, defined by

$$f_s^{\text{a}}(t) := \frac{|\{p \in P : t_{s,p} \leq t\}|}{|P|}.$$

Considering $t_{s,p} = +\infty$ when solver s fails on problem p, $f_s^{\text{a}}(t)$ is the fraction of problems solved by solver s within the time t. Note that, in contrast to f_s^{r}, the time profile $t \mapsto f_s^{\text{a}}(t)$ is independent from other solvers and displayed with the actual timings of s.

8.2 Random problems

We considered QPs in the form of (1) with randomly generated problem data. In each problem instance, the number of variables is $n = \lceil 10^a \rceil$, with a uniformly distributed, and ranges between 10^1 and 10^3, i.e., $a \sim \mathcal{U}(1,3)$. The number of constraints is $m = 10\,n$. The linear cost is normally distributed, i.e., $\mathbf{q}_i \sim \mathcal{N}(0,1)$. The cost matrix is $\boldsymbol{Q} = \boldsymbol{P}\boldsymbol{P}^\top$, where $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ has 10% nonzero entries $\boldsymbol{P}_{ij} \sim \mathcal{N}(0,1)$. The constraint matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ contains 10% nonzero entries $\boldsymbol{A}_{ij} \sim \mathcal{N}(0,1)$. The bounds are uniformly distributed, i.e., $\mathbf{l}_i \sim \mathcal{U}(-1,0)$ and $\mathbf{u}_i \sim \mathcal{U}(0,1)$. We also investigated equality-constrained QPs. For these problems, n ranges between 10^2 and 10^4, $m = \lceil n/10 \rceil$, and $\mathbf{l}_i = \mathbf{u}_i \sim \mathcal{N}(0,1)$. We generated 250 instances of each problem class.

Results Computational results are summarized in Table 1 and shown in Figs. 1 and 2. Performance profiles suggest that, for both problem classes, QPALM exhibits the best performance, with QPDO slightly behind and OSQP third. However, the time profiles in Fig. 2 show that, on equality-constrained QPs, QPDO scales better than the other solvers. Indeed, QPDO is the first to complete the test set of random problems. OSQP reaches the time limit on few problems, due to the relative high accuracy requirement. Overall, all solvers prove competitive.

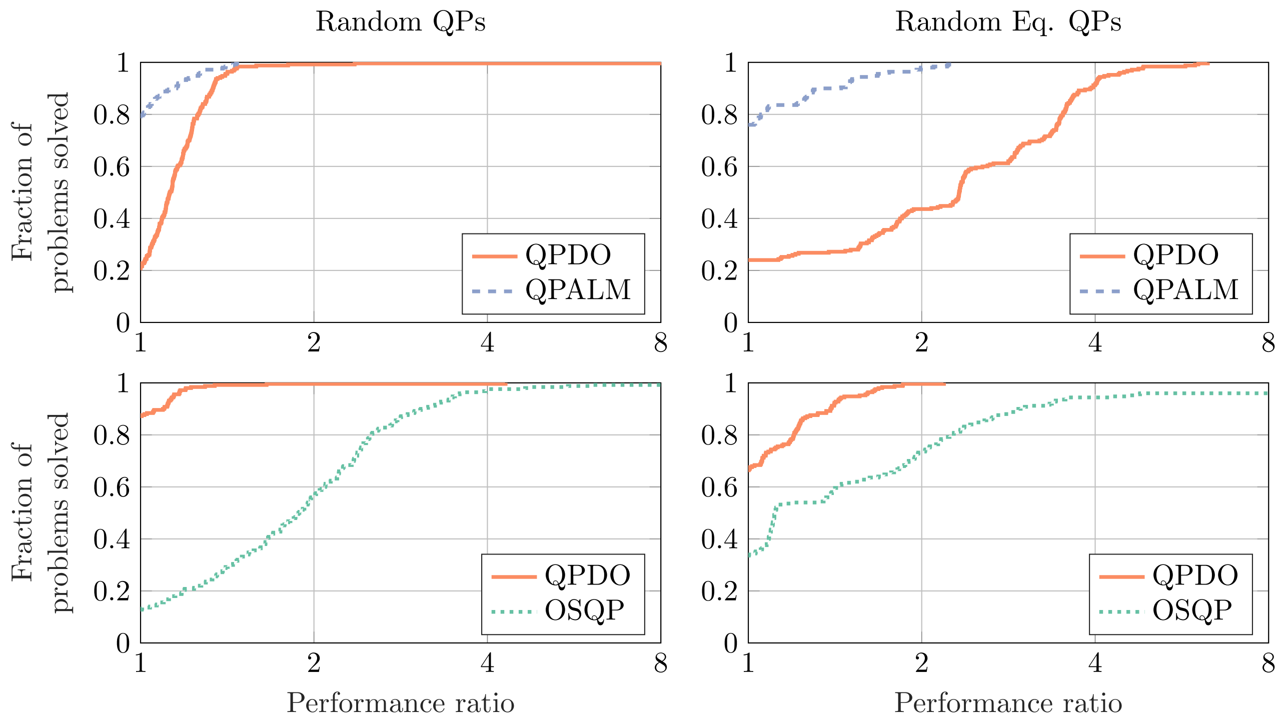

Figure 1: Comparison on random problems with performance profiles: fraction of problems solved by each solver as a function of performance ratio.

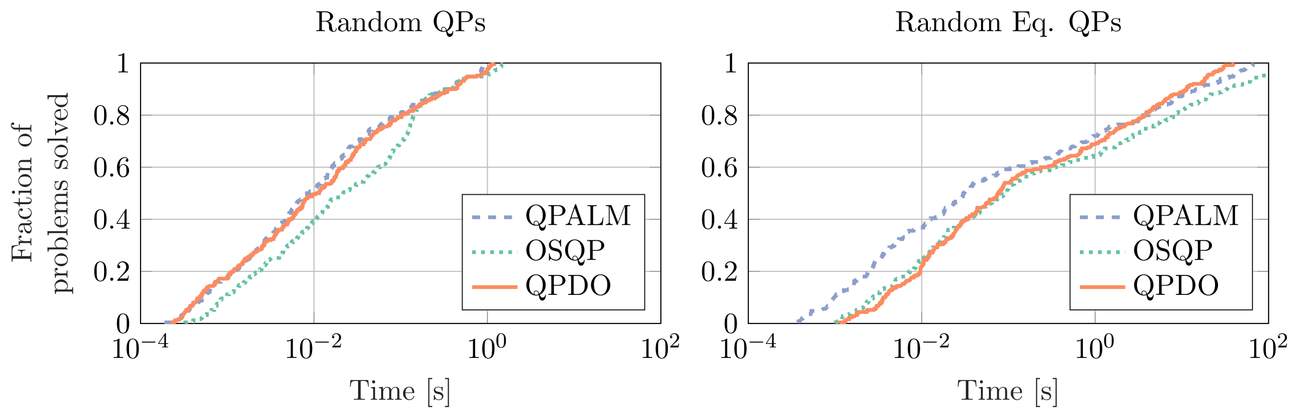

Figure 2: Comparison on random problems with data profiles: fraction of problems solved by each solver as a function of run time.

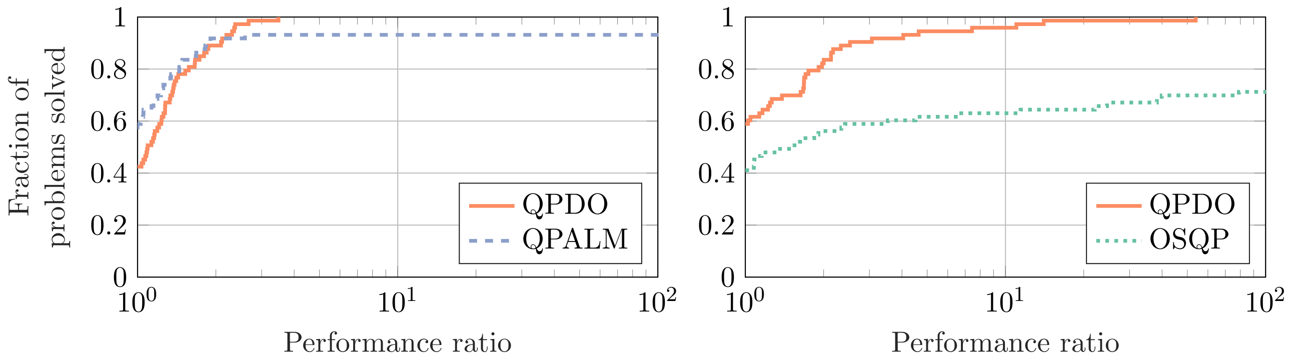

Figure 3: Comparison on Maros–Mészáros problems with performance profiles: fraction of problems solved by each solver as a function of performance ratio.

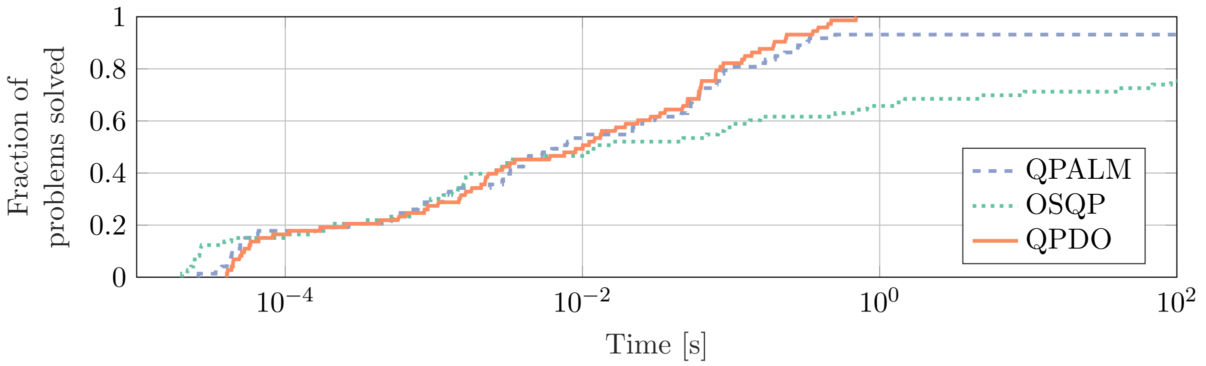

Figure 4: Comparison on Maros–Mészáros problems with data profiles: fraction of problems solved by each solver as a function of run time.

8.3 Maros–Mészáros problems

We considered the Maros–Mészáros test set [36] of hard QPs. Selecting those with $n \leq 10^3$ yields 73 problems, with $2 \leq n \leq 1000$, $3 \leq m \leq 1750$, and the number of nonzeros $6 \leq \mathrm{nnz}(\boldsymbol{Q}) + \mathrm{nnz}(\boldsymbol{A}) \leq 22292$.

Results Computational results are summarized in Table 1 and shown in Figs. 3 and 4. On this test set, QPDO demonstrates its robustness. OSQP is very fast for some problems but has a high failure rate. As a first-order method, OSQP builds upon computationally cheap iterations, but it may take many to cope with ill-conditioning and the relatively high accuracy requirements. QPALM fails on some problems, presumably due to linear algebra issues; its relatively high timing in Table 1 is associated to the failure rate. Considering only the problems it solved, QPALM's timing (sgm) is 0.050 s, whereas QPDO takes 0.054 s. Indeed, this proves QPDO is both reliable and effective.

9 Conclusions

This paper presented a primal-dual Newton-type proximal method for convex quadratic programs. We build upon a simple yet crucial result: a suitable merit function for the proximal sub-problem is found in the proximal primal-dual augmented Lagrangian function. This allows us to effectively weave the proximal point method together with semismooth Newton's, yielding structured symmetric linear systems, exact linesearch, and the possibility to apply sparse multi-rank factorization updates. Requiring only convexity, the method is simple and easily warm started, can exploit sparsity pattern, is robust to early termination, and can detect infeasibility. We have implemented our method QPDO in a general-purpose solver, written in open-source C code. We benchmarked it against state-of-the-art QP solvers, comparing run times and failure rates. QPDO proved reliable, effective, and competitive.

A Primal-dual proximal augmented Lagrangian function

We show that the merit function $\mathcal{M}$ in (12) for the sub-problem (11) is indeed the primal-dual proximal augmented Lagrangian function, proposed and investigated in [50, 25, 15]. Let us reformulate (1) as the equivalent problem

$$\min_{\mathbf{x},\mathbf{z}} \quad f(\mathbf{x}) + g(\mathbf{z}) \qquad \text{s.t.} \quad \boldsymbol{A}\mathbf{x} = \mathbf{z}, \tag{23}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{z} \in \mathbb{R}^m$ are decision variables. The Lagrangian $\mathcal{L}^z$, the augmented Lagrangian $\mathcal{L}^z_\mu$, and the primal-dual augmented Lagrangian $\mathcal{M}^z_\mu$ functions for (23) are given by

$$\begin{aligned}
\mathcal{L}^z(\mathbf{x},\mathbf{z},\mathbf{y}) &:= f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^\top(\boldsymbol{A}\mathbf{x} - \mathbf{z}),\\
\mathcal{L}^z_\mu(\mathbf{x},\mathbf{z},\mathbf{y}) &:= \mathcal{L}^z(\mathbf{x},\mathbf{z},\mathbf{y}) + \frac{1}{2\mu}\|\boldsymbol{A}\mathbf{x} - \mathbf{z}\|^2,\\
\mathcal{M}^z_\mu(\mathbf{x},\mathbf{z},\mathbf{y},\overline{\mathbf{y}}) &:= \mathcal{L}^z_\mu(\mathbf{x},\mathbf{z},\overline{\mathbf{y}}) + \frac{1}{2\mu}\|\mathbf{z} - \boldsymbol{A}\mathbf{x} + \mu(\mathbf{y} - \overline{\mathbf{y}})\|^2,
\end{aligned}$$

for some given parameter $\mu > 0$ and dual estimate $\overline{\mathbf{y}} \in \mathbb{R}^m$; cf. [13, 6] and [50, 25]. Introducing a primal proximal regularization, we define

$$\mathcal{M}^z_{\mu,\sigma}(\mathbf{x},\mathbf{z},\mathbf{y},\overline{\mathbf{x}},\overline{\mathbf{y}}) := \mathcal{M}^z_\mu(\mathbf{x},\mathbf{z},\mathbf{y},\overline{\mathbf{y}}) + \frac{\sigma}{2}\|\mathbf{x} - \overline{\mathbf{x}}\|^2 \tag{24}$$

for some given parameter $\sigma > 0$ and primal estimate $\overline{\mathbf{x}} \in \mathbb{R}^n$. In the context of primal-dual augmented Lagrangian methods, the function $\mathcal{M}^z_{\mu,\sigma}$ is to be jointly minimized with respect to $\mathbf{x}$, $\mathbf{z}$, and $\mathbf{y}$ [50, 25]. Following [15], we consider the explicit minimization over the auxiliary variable $\mathbf{z}$. The minimizer $\mathbf{z}_\mu$ of $\mathcal{M}^z_{\mu,\sigma}$ in (24) is readily obtained as

$$\begin{aligned}
\mathbf{z}_\mu(\mathbf{x},\mathbf{y},\overline{\mathbf{y}}) &:= \arg\min_{\mathbf{z}} \mathcal{M}^z_{\mu,\sigma}(\mathbf{x},\mathbf{y},\mathbf{z},\overline{\mathbf{x}},\overline{\mathbf{y}}) \\
&= \arg\min_{\mathbf{z}} \left\{ g(\mathbf{z}) + \mathbf{y}^\top(\boldsymbol{A}\mathbf{x} - \mathbf{z}) + \frac{1}{2\mu}\|\boldsymbol{A}\mathbf{x} - \mathbf{z}\|^2 + \frac{1}{2\mu}\|\mathbf{z} - \boldsymbol{A}\mathbf{x} + \mu(\mathbf{y} - \overline{\mathbf{y}})\|^2 \right\} \\
&= \arg\min_{\mathbf{z}} \left\{ g(\mathbf{z}) + \frac{1}{2\mu}\|\boldsymbol{A}\mathbf{x} - \mathbf{z} + \mu\overline{\mathbf{y}}\|^2 + \frac{1}{2\mu}\|\mathbf{z} - \boldsymbol{A}\mathbf{x} + \mu(\mathbf{y} - \overline{\mathbf{y}})\|^2 \right\} \\
&= \arg\min_{\mathbf{z}} \left\{ g(\mathbf{z}) + \frac{1}{\mu}\|\mathbf{z} - \boldsymbol{A}\mathbf{x} - \mu(\overline{\mathbf{y}} - \mathbf{y}/2)\|^2 + \frac{\mu}{4}\|\mathbf{y}\|^2 - \frac{\mu}{2}\|\overline{\mathbf{y}}\|^2 \right\} \\
&= \Pi_{\mathcal{C}}\left(\boldsymbol{A}\mathbf{x} + \mu(\overline{\mathbf{y}} - \mathbf{y}/2)\right). \qquad (25)
\end{aligned}$$

Considering $\mathcal{M}^z_{\mu,\sigma}$ on the manifold defined by $\mathbf{z}_\mu$ in (25), we get the primal-dual proximal augmented Lagrangian function $\mathcal{M}_{\mu,\sigma}$. This yields

$$\begin{aligned}
\mathcal{M}_{\mu,\sigma}(\mathbf{v},\overline{\mathbf{v}}) &:= \mathcal{M}^z_{\mu,\sigma}(\mathbf{x},\mathbf{z}_\mu(\mathbf{x},\mathbf{y},\overline{\mathbf{y}}),\mathbf{y},\overline{\mathbf{x}},\overline{\mathbf{y}}) \\
&= f(\mathbf{x}) + \frac{1}{\mu}\|\mathbf{z}_\mu(\mathbf{x},\mathbf{y},\overline{\mathbf{y}}) - \boldsymbol{A}\mathbf{x} - \mu(\overline{\mathbf{y}} - \mathbf{y}/2)\|^2 + \frac{\mu}{4}\|\mathbf{y}\|^2 - \frac{\mu}{2}\|\overline{\mathbf{y}}\|^2 + \frac{\sigma}{2}\|\mathbf{x} - \overline{\mathbf{x}}\|^2 \\
&= f(\mathbf{x}) + \frac{1}{\mu}\operatorname{dist}^2_{\mathcal{C}}\left(\boldsymbol{A}\mathbf{x} + \mu(\overline{\mathbf{y}} - \mathbf{y}/2)\right) + \frac{\sigma}{2}\|\mathbf{x} - \overline{\mathbf{x}}\|^2 + \frac{\mu}{4}\|\mathbf{y}\|^2 - \frac{\mu}{2}\|\overline{\mathbf{y}}\|^2,
\end{aligned}$$

which matches $\mathcal{M}_k$ in (12), up to the constant term $-\mu\|\overline{\mathbf{y}}\|^2/2$.

B Exact linesearch coefficients

We prove that the right-hand side of (18) coincides with $\psi_k'(\tau)$ for all $\tau \in \mathbb{R}$, with the coefficients given in (19). Let $\mathbf{w}_k := \boldsymbol{A}\mathbf{x} + \mu_k(\mathbf{y}_k - \mathbf{y}/2)$ and $\delta\mathbf{w}_k := \boldsymbol{A}\delta\mathbf{x} - \mu_k\mathbf{y}/2$; cf. (19). Then, from (12), we have

$$\begin{aligned}
\psi'(\tau) &= \delta\mathbf{v}^\top \nabla\mathcal{M}(\mathbf{v} + \tau\delta\mathbf{v}) \\
&= \begin{pmatrix} \delta\mathbf{x} \\ \delta\mathbf{y} \end{pmatrix}^\top \begin{pmatrix} \boldsymbol{Q}(\mathbf{x} + \tau\delta\mathbf{x}) + \mathbf{q} + \frac{2}{\mu_k}\boldsymbol{A}^\top\left[\mathbf{w}_k + \tau\delta\mathbf{w}_k - \Pi_{\mathcal{C}}\left(\mathbf{w}_k + \tau\delta\mathbf{w}_k\right)\right] + \sigma_k(\mathbf{x} + \tau\delta\mathbf{x} - \mathbf{x}_k) \\ -\left[\boldsymbol{A}(\mathbf{x} + \tau\delta\mathbf{x}) + \mu_k(\mathbf{y}_k - \mathbf{y} - \tau\delta\mathbf{y}) - \Pi_{\mathcal{C}}\left(\mathbf{w}_k + \tau\delta\mathbf{w}_k\right)\right] \end{pmatrix} \\
&= \delta\mathbf{x}^\top\left[\boldsymbol{Q}\mathbf{x} + \mathbf{q} + \sigma_k(\mathbf{x} - \mathbf{x}_k)\right] + \frac{\mu_k}{2}\delta\mathbf{y}^\top\mathbf{y} + \tau\delta\mathbf{x}^\top\left(\boldsymbol{Q} + \sigma_k\boldsymbol{I}\right)\delta\mathbf{x} + \tau\frac{\mu_k}{2}\delta\mathbf{y}^\top\delta\mathbf{y} \\
&\quad + \left[\frac{2}{\mu_k}\boldsymbol{A}\delta\mathbf{x} - \delta\mathbf{y}\right]^\top \left[\mathbf{w}_k + \tau\delta\mathbf{w}_k - \Pi_{\mathcal{C}}\left(\mathbf{w}_k + \tau\delta\mathbf{w}_k\right)\right] \\
&= \alpha_k\tau + \beta_k + \frac{2}{\mu_k}\delta\mathbf{w}_k^\top\left[\mathbf{w}_k + \tau\delta\mathbf{w}_k - \Pi_{\mathcal{C}}\left(\mathbf{w}_k + \tau\delta\mathbf{w}_k\right)\right].
\end{aligned}$$

Acknowledgment

This work would not have been possible without the support and enthusiasm of my supervisor, Matthias Gerdts. The author would like to thank Axel Dreves, whose comments on an early draft greatly improved the presentation.

References

[1] Alnur Ali, Eric Wong, and J. Zico Kolter. A semismooth Newton method for fast, generic convex programming. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 70–79, Sydney, 8 2017.

[2] Paul Armand and Riadh Omheni. A globally and quadratically convergent primal–dual augmented Lagrangian algorithm for equality constrained optimization. *Optimization Methods and Software*, 32(1):1–21, 2017.

[3] Sylvain Arreckx and Dominique Orban. A regularized factorization-free method for equality-constrained optimization. *SIAM Journal on Optimization*, 28(2):1613–1639, 2018.

[4] Goran Banjac, Paul Goulart, Bartolomeo Stellato, and Stephen Boyd. Infeasibility detection in the alternating direction method of multipliers for convex optimization. *Journal of Optimization Theory and Applications*, 183(2):490–519, 2019.

[5] Alberto Bemporad. A numerically stable solver for positive semidefinite quadratic programs based on nonnegative least squares. *IEEE Transactions on Automatic Control*, 63(2):525–531, 2 2018.

[6] Dimitri P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.

[7] Ernesto G. Birgin and José Mario Martínez. *Practical Augmented Lagrangian Methods for Constrained Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 4 2014.

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. now, 2011.

[9] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 3 2004.

[10] Bintong Chen, Xiaojun Chen, and Christian Kanzow. A penalized Fischer-Burmeister NCP-function. *Mathematical Programming*, 88(1):211–216, 6 2000.

[11] Yanqing Chen, Timothy A. Davis, William W. Hager, and Sivasankaran Rajamanickam. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Trans. Math. Softw.*, 35(3):1–14, 10 2008.

[12] Kazem Cheshmi, Danny M. Kaufman, Shoaib Kamil, and Maryam Mehri Dehnavi. NASOQ: Numerically accurate sparsity-oriented QP solver. *ACM Transactions on Graphics*, 39(4), 7 2020.

[13] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM Journal on Numerical Analysis*, 28(2):545–572, 4 1991.

[14] Timothy A. Davis and William W. Hager. Multiple-rank modifications of a sparse cholesky factorization. *SIAM Journal on Matrix Analysis and Applications*, 22(4):997–1013, 2001.

[15] N. K. Dhingra, S. Z. Khong, and M. R. Jovanović. The proximal augmented Lagrangian method for nonsmooth composite optimization. *IEEE Transactions on Automatic Control*, 64(7):2861–2868, 7 2019.

[16] Elizabeth D. Dolan and Jorge J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[17] Asen L. Dontchev and R. Tyrrell Rockafellar. *Implicit functions and solution mappings*. Monogr. Math. Springer, 2009.

[18] Hans Joachim Ferreau, Christian Kirches, Andreas Potschka, Hans Georg Bock, and Moritz Diehl. qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, 12 2014.

[19] Andreas Fischer. A special Newton-type optimization method. *Optimization*, 24:269–284, 1992.

[20] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1–2):95–110, 1956.

[21] M. P. Friedlander and D. Orban. A primal-dual regularized interior-point method for convex quadratic programs. *Mathematical Programming Computation*, 4(1):71–107, 3 2012.

[22] Michael P. Friedlander and Paul Tseng. Exact regularization of convex programs. *SIAM Journal on Optimization*, 18(4):1326–1350, 2008.

[23] Matthias Gerdts and Martin Kunkel. A nonsmooth Newton's method for discretized optimal control problems with state and control constraints. *Journal of Industrial & Management Optimization*, 4(2):247–270, 2008.

[24] Philip E. Gill, Gene H. Golub, Walter Murray, and Michael A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, 4 1974.

[25] Philip E. Gill and Daniel P. Robinson. A primal-dual augmented Lagrangian. *Computational Optimization and Applications*, 51(1):1–25, 1 2012.

[26] Philip E. Gill and Daniel P. Robinson. A globally convergent stabilized SQP method. *SIAM Journal on Optimization*, 23(4):1983–2010, 2013.

[27] Jacek Gondzio. Interior point methods 25 years later. *European Journal of Operational Research*, 218(3):587–601, 2012.

[28] Nicholas Gould and Jennifer Scott. A note on performance profiles for benchmarking software. *ACM Trans. Math. Softw.*, 43(2), 9 2016.

[29] Nick I. M. Gould, Dominique Orban, and Philippe L. Toint. Numerical methods for large-scale nonlinear optimization. *Acta Numerica*, 14:299–361, 5 2005.

[30] Ben Hermans, Andreas Themelis, and Panagiotis Patrinos. QPALM: A Newton-type proximal augmented Lagrangian method for quadratic programs. In *2019 IEEE 58th Conference on Decision and Control (CDC))*, pages 4325–4330, 11 2019.

[31] Ben Hermans, Andreas Themelis, and Panagiotis Patrinos. QPALM: A proximal augmented Lagrangian method for nonconvex quadratic programs, 10 2020.

[32] Michael Hintermüller, K. Ito, and Karl Kunisch. The primal-dual active set strategy as a semismooth Newton method. *SIAM Journal on Optimization*, 13(3):865–888, 2002.

[33] Alexey F. Izmailov and Mikhail V. Solodov. *Newton-Type Methods for Optimization and Variational Problems*. Springer, 2014.

[34] Dominic Liao-McPherson and Ilya Kolmanovsky. FBstab: A proximally stabilized semismooth algorithm for convex quadratic programming. *Automatica*, 113:108801, 2020.

[35] Fernando Javier Luque. Asymptotic convergence analysis of the proximal point algorithm. *SIAM Journal on Control and Optimization*, 22(2):277–293, 1984.

[36] István Maros and Csaba Mészáros. A repository of convex quadratic programming problems. *Optimization Methods and Software*, 11(1–4):671–681, 1999.

[37] George J. Minty. Monotone (nonlinear) operators in hilbert space. *Duke Mathematical Journal*, 29(3):341–346, 9 1962.

[38] Hans D. Mittelmann. Benchmarks for optimization software. Accessed 19 Nov 2020.

[39] Jorge J. Moré and Stefan M. Wild. Benchmarking derivative-free optimization algorithms. *SIAM Journal on Optimization*, 20(1):172–191, 2009.

[40] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, 2nd edition, 2006.

[41] Daniel O'Connor and Lieven Vandenberghe. Primal-dual decomposition by operator splitting and applications to image deblurring. *SIAM Journal on Imaging Sciences*, 7(3):1724–1754, 2014.

[42] Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, 6 2016.

[43] Jong-Shi Pang. Error bounds in mathematical programming. *Mathematical Programming*, 79(1):299–332, 10 1997.

[44] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, 2014.

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

[46] S. Pieraccini, M. G. Gasparo, and A. Pasquali. Global Newton-type methods and semismooth reformulations for NCP. *Applied Numerical Mathematics*, 44(3):367 – 384, 2003.

[47] Spyridon Pougkakiotis and Jacek Gondzio. An interior point-proximal method of multipliers for convex quadratic programming. *Computational Optimization and Applications*, 11 2020.

[48] Liqun Qi and Houyuan Jiang. Semismooth Karush-Kuhn-Tucker equations and convergence analysis of Newton and quasi-Newton methods for solving these equations. *Mathematics of Operations Research*, 22(2):301–325, 1997.

[49] Liqun Qi and Jie Sun. A nonsmooth version of Newton's method. *Mathematical Programming*, 58(1):353–367, 1 1993.

[50] Daniel P. Robinson. *Primal-Dual Methods for Nonlinear Optimization.* PhD thesis, University of California, San Diego, 9 2007.

[51] R. Tyrrell Rockafellar. *Convex Analysis.* Princeton University Press, Princeton, NJ, 1997.

[52] Ralph Tyrrell Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of operations research*, 1(2):97–116, 5 1976.

[53] Ralph Tyrrell Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5):877–898, 1976.

[54] Daniel Ruiz. A scaling algorithm to equilibrate both rows and columns norms in matrices. Technical Report RAL-TR-2001-034, Rutherford Appleton Laboratory, Oxon, UK, 9 2001.

[55] Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. OSQP: An operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 2 2020.

[56] Defeng Sun and Liqun Qi. On NCP-functions. *Computational Optimization and Applications*, 13(1):201–220, 4 1999.

[57] J. Sun. On piecewise quadratic Newton and trust region problems. *Mathematical Programming*, 76(3):451–467, 3 1997.

[58] Andreas Themelis and Panagiotis Patrinos. SuperMann: A superlinearly convergent algorithm for finding fixed points of nonexpansive operators. *IEEE Transactions on Automatic Control*, 64(12):4875–4890, 12 2019.

[59] Robert J. Vanderbei. Symmetric quasidefinite matrices. *SIAM Journal on Optimization*, 5(1):100–113, 1995.

[60] Philip Wolfe. The simplex method for quadratic programming. *Econometrica*, 27(3):382–398, 1959.