SIAM J. OPTIM.
Vol. 26, No. 4, pp. 2261–2283

GLOBALLY CONVERGENT PRIMAL-DUAL ACTIVE-SET METHODS WITH INEXACT SUBPROBLEM SOLVES[*]

FRANK E. CURTIS[†] AND ZHENG HAN[†]

Abstract. We propose primal-dual active-set (PDAS) methods for solving large-scale instances of an important class of convex quadratic optimization problems (QPs). The iterates of the algorithms are partitions of the index set of variables, where corresponding to each partition there exist unique primal-dual variables that can be obtained by solving a (reduced) linear system. Algorithms of this type have recently received attention when solving certain QPs and linear complementarity problems since, with rapid changes in the active set estimate, they often converge in few iterations. Indeed, as discussed in this paper, convergence in a finite number of iterations is guaranteed when a basic PDAS method is employed to solve certain QPs for which a reduced Hessian of the objective function is (a perturbation of) an M-matrix. We propose three PDAS algorithms. The novelty of the algorithms is that they allow inexactness in the (reduced) linear system solves at all partitions except optimal ones. Such a feature is particularly important in large-scale settings when one employs iterative Krylov subspace methods to solve these systems. Our first algorithm is convergent when solving problems for which properties of the Hessian can be exploited to derive explicit bounds to be enforced on the (reduced) linear system residuals, whereas our second and third algorithms employ dynamic parameters to obviate the need of such explicit bounds. We prove that when applied to solve an important class of convex QPs, our algorithms converge from any initial partition. We also illustrate their practical behavior by providing the results of numerical experiments on a set of discretized optimal control problems, some of which are explicitly formulated to exhibit degeneracy.

Key words. convex quadratic optimization, large-scale optimization, primal-dual active-set methods, semismooth Newton methods, inexact Newton methods, Krylov subspace methods

AMS subject classifications. 49M05, 49M15, 65K05, 65K10, 65K15

DOI. 10.1137/140993314

1. Introduction.

Convex quadratic optimization problems (QPs) arise in numerous areas of applied mathematics [9, 10, 23, 34, 35, 38, 41, 48, 51]. Consequently, algorithms for solving such problems have been studied for decades. These algorithms generally fall into the categories of active-set [7, 17, 44] and interior-point methods [37, 43, 50, 54]. There are also a variety of methods designed exclusively for bound-constrained QPs (BQPs), which represent an important subclass of the class of QPs considered in this paper. These include active-set [18, 21], interior-point [12, 26], gradient projection [6, 14, 19, 44], or some combination of these methods [8, 25, 42].

In this paper, we propose three primal-dual active-set (PDAS) methods for solving large-scale instances of an important class of QPs. We consider our methods to be enhancements of the method proposed in [28], but note that our methods are also related to the work in [1, 5, 34, 39], the subsequent work in [13, 35, 36], and numerous other articles on the use of PDAS methods [15, 24, 29, 32, 33, 40, 47]. The key feature of the algorithm in [28] (and others just cited) is that it allows any number of changes in the active-set estimate during each iteration. This is in contrast to classical

[*]Received by the editors October 28, 2014; accepted for publication (in revised form) August 10, 2016; published electronically October 26, 2016.
http://www.siam.org/journals/siopt/26-4/99331.html
Funding: This material is based upon work supported by the U.S. Department of Energy, Office of Science, Applied Mathematics, Early Career Research Program under award DE–SC0010615 and by the U.S. National Science Foundation, Division of Mathematical Sciences, Computational Mathematics Program under award DMS–1016291.

[†]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA (frank.e.curtis@lehigh.edu, zh.hansci@gmail.com).

primal or dual active-set methods, where worst-case performance can be plagued by slow adaptation of the active-set estimate [3, 20, 22, 44]. The method in [28] is one in which each iterate corresponds to a partition of an index set of variables into an active set and an inactive set. Such a partition corresponds to a unique primal-dual solution estimate via a reduced linear system whose dimension depends on the size of the inactive set. Solving this reduced linear system represents the main computational expense during each iteration of the algorithm.

Our contributions in this paper are twofold. First, we present a convergence result for a flexible PDAS framework that is applicable to a broad class of QPs. In particular, our convergence result applies for problems with certain sets of equality constraints and upper bounds on a subset of variables and assumes only that a reduced Hessian of the objective has certain structure—which it does in applications of interest, such as certain optimal control problems (see section 5). Second, with the goal of enhancing the algorithm in [28], a novel feature of our algorithms is that the reduced linear systems may be solved *inexactly.* This is of particular interest in large-scale settings when the systems are to be solved iteratively, such as with Krylov subspace methods [11, 45, 46, 49]. We present three algorithms, each involving a set of implementable conditions to control inexactness in such a way that convergence guarantees are ensured. These conditions always allow inexactness in the system solves for any partition that is suboptimal. If a given partition is optimal in that the corresponding primal-dual solution is optimal for the QP, then our algorithms require sufficient accuracy in the subproblem solution so that the algorithm will terminate.

We have two important remarks to make at the outset of this work. First, it will be shown that any QP in our class of interest can be reduced to a BQP by using the affine constraints to eliminate the set of free (i.e., unbounded) variables. Thus, an alternative to the algorithms proposed in this paper is to construct the corresponding BQP and solve it, say, using a PDAS method that follows our proposed strategies for controlling inexactness in the linear system solves. While such an approach might seem simpler than our proposed algorithms, it has a key disadvantage: in certain applications, the problem data in the QP is sparse, but the data in the corresponding BQP (specifically, the reduced Hessian) is dense, which might lead to extra storage requirements and more expensive matrix-vector products. Our methods, on the other hand, involve iterative linear system solves using the original QP problem data, where one need only invoke information related to a reduced residual once a good candidate solution has been obtained. A second important remark is that while some of our techniques for controlling inexactness are not trivial, a simpler heuristic method that merely decreases a dynamic accuracy tolerance as the optimization process proceeds will *not* ensure convergence unless one imposes strong requirements on the employed linear system solver. Indeed, such is the approach employed in our third algorithm, for which our convergence guarantees are significantly weaker than for our first two algorithms. We also provide an example illustrating why the convergence guarantees for such an algorithm *must be* weaker. Overall, our first two algorithms are able to attain stronger convergence guarantees as they involve procedures for computing an upper bound on the norm of the inverse of a particular submatrix during each iteration; our first algorithm computes such an upper bound explicitly, whereas our second algorithm incorporates a dynamic parameter that effectively replaces this upper bound.

This paper is organized as follows. In section 2, we state our problem class of interest, discuss basic concepts and definitions, and outline our PDAS framework. In section 3, we present our algorithms and corresponding subroutines. We also prove that the algorithms attain convergence guarantees for our problem class of interest. We then discuss an implementation of our algorithm in section 4 and provide the results of numerical experiments in section 5. Concluding remarks are provided in section 6.

Notation. We use index sets as subscripts to denote the subvector or submatrix corresponding to the indices in the sets. For example, given an ordered set of indices $\mathcal{S}$, by $x_{\mathcal{S}}$ we denote the subvector of the vector x corresponding to the indices in $\mathcal{S}$, and, with another ordered set of indices $\mathcal{T}$, by $H_{\mathcal{S}\mathcal{T}}$ we denote the submatrix of the matrix H with row indices in $\mathcal{S}$ and column indices in $\mathcal{T}$. Given such an $H_{\mathcal{S}\mathcal{T}}$, its inverse/transpose/inverse-transpose is written as $H_{\mathcal{S}\mathcal{T}}^{T}/H_{\mathcal{S}\mathcal{T}}^{-1}/H_{\mathcal{S}\mathcal{T}}^{-T}$. We also occasionally denote a vector composed of stacked subvectors as an ordered tuple of vectors, i.e., for vectors a and b we occasionally write $(a,b) := [a^T\ b^T]^T$. For a square symmetric matrix S, we write $S \succ 0$ ($S \succeq 0$) to indicate that S is positive definite (semidefinite). The quantities I, e, and e_i denote the identity matrix, column vector of ones, and ith unit column vector (i.e., ith column of I), respectively, where the size of each is determined by the context in which it appears. We denote the p-norm of a vector x by $\|x\|_p$ and denote the p-norm of a matrix H by $\|H\|_p$, whose value is that induced by the vector p-norm, i.e., $\|H\|_p = \sup_{x\neq 0} \|Hx\|_p/\|x\|_p$. In particular, the 1-norm and ∞-norm of a matrix H (i.e., $\|H\|_1$ and $\|H\|_\infty$) are the maximum absolute column sum and maximum absolute row sum of H, respectively. Finally, for a matrix H, we define $[H]_+ := \max\{0, H\}$, where the max should be understood componentwise.

2. Fundamentals.

For a positive integer n and nonnegative integer m, we define an index set of upper-bounded variables $\mathcal{B} := \{1,\dots,n\}$, index set of free variables $\mathcal{F} := \{n+1,\dots,n+m\}$, and index set of equality constraints $\mathcal{E} := \{1,\dots,m\}$. Then, given problem data in terms of $c \in \mathbb{R}^{n+m}$, $H \in \mathbb{R}^{(n+m)\times(n+m)}$, $A \in \mathbb{R}^{m\times(n+m)}$, $b \in \mathbb{R}^m$, and $u \in \mathbb{R}^n$, we consider the quadratic optimization problem

$$\text{(QP)} \qquad \min_{x\in\mathbb{R}^{n+m}}\ c^T \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{F}} \end{bmatrix} + \tfrac{1}{2} \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{F}} \end{bmatrix}^T H \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{F}} \end{bmatrix} \quad \text{s.t.} \quad A \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{F}} \end{bmatrix} = b,\ x_{\mathcal{B}} \le u.$$

We make the following assumption about an instance of (QP). (The convergence guarantees for our proposed algorithms require strengthenings of this assumption; e.g., see Theorem 3 at the end of this section. However, the following weaker assumption serves our immediate purposes.) The assumption refers to the reduced Hessian of the objective function that one obtains by using the affine constraints to eliminate the free variables. This matrix is the Hessian of the objective for the resulting BQP.

Assumption 1. In (QP), the constraint data submatrix $A_{\mathcal{E}\mathcal{F}}$ is invertible and the reduced Hessian in terms of the upper-bounded variables is positive definite, i.e.,

$$(1) \qquad R := \begin{bmatrix} I \\ -A_{\mathcal{E}\mathcal{F}}^{-1} A_{\mathcal{E}\mathcal{B}} \end{bmatrix}^T H \begin{bmatrix} I \\ -A_{\mathcal{E}\mathcal{F}}^{-1} A_{\mathcal{E}\mathcal{B}} \end{bmatrix} \succ 0.$$

Under Assumption 1, it follows that (QP) is feasible and there exists a unique primal point x and unique Lagrange multipliers (y,z) satisfying the Karush–Kuhn–Tucker (KKT) optimality conditions for (QP), which can be written as

$$0 = \text{KKT}(x,y,z) := \left(c + H \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{F}} \end{bmatrix} + A^T y + \begin{bmatrix} z \\ 0 \end{bmatrix},\ A \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{F}} \end{bmatrix} - b,\ \min\{u - x_{\mathcal{B}}, z\} \right).$$

We define a partition $(\mathcal{A}, \mathcal{I})$ of the index set of bounded variables as a pair of mutually exclusive and exhaustive subsets of $\mathcal{B}$, where $\mathcal{A}$ represents an *active* set of variables (i.e., variables equal to their upper bounds) and $\mathcal{I} = \mathcal{B} \setminus \mathcal{A}$ represents the corresponding *inactive* set. Corresponding to a partition $(\mathcal{A}, \mathcal{I})$, we define a *subspace solution*, call it (x, y, z), by the following sequence of operations:

$$\text{Set } \; x_{\mathcal{A}} \leftarrow u_{\mathcal{A}} \; \text{ and } \; z_{\mathcal{I}} \leftarrow 0, \tag{2a}$$

$$\text{then solve } \begin{bmatrix} H_{\mathcal{I}\mathcal{I}} & H_{\mathcal{I}\mathcal{F}} & A_{\mathcal{E}\mathcal{I}}^T \\ H_{\mathcal{F}\mathcal{I}} & H_{\mathcal{F}\mathcal{F}} & A_{\mathcal{E}\mathcal{F}}^T \\ A_{\mathcal{E}\mathcal{I}} & A_{\mathcal{E}\mathcal{F}} & 0 \end{bmatrix} \begin{bmatrix} x_{\mathcal{I}} \\ x_{\mathcal{F}} \\ y \end{bmatrix} = - \begin{bmatrix} c_{\mathcal{I}} \\ c_{\mathcal{F}} \\ -b \end{bmatrix} - \begin{bmatrix} H_{\mathcal{I}\mathcal{A}} \\ H_{\mathcal{F}\mathcal{A}} \\ A_{\mathcal{E}\mathcal{A}} \end{bmatrix} u_{\mathcal{A}} \tag{2b}$$

$$\text{for } \; (x_{\mathcal{I}}, x_{\mathcal{F}}, y),$$

$$\text{then set } \; z_{\mathcal{A}} \leftarrow -H_{\mathcal{A}\mathcal{B}} x_{\mathcal{B}} - H_{\mathcal{A}\mathcal{F}} x_{\mathcal{F}} - A_{\mathcal{E}\mathcal{A}}^T y - c_{\mathcal{A}}. \tag{2c}$$

Fixing the variables indexed by $\mathcal{A}$ at their upper bounds (see (2a)), Assumption 1 ensures that the reduced Hessian in terms of the remaining upper-bounded variables (indexed by $\mathcal{I}$) is positive definite. Hence, under Assumption 1, the matrix on the left-hand side of (2b) is nonsingular, meaning that the subspace solution corresponding to any given partition $(\mathcal{A}, \mathcal{I})$ is unique. We call $(\mathcal{A}, \mathcal{I})$ an *optimal partition* if its subspace solution (x, y, z) satisfies $\mathrm{KKT}(x, y, z) = 0$, i.e., if (x, y, z) is the optimal primal-dual solution for (QP). Otherwise, the partition $(\mathcal{A}, \mathcal{I})$ and its corresponding subspace solution (x, y, z) are *suboptimal.* While the optimal primal-dual solution for an instance of (QP) is unique, there may be more than one optimal partition.

Given a partition $(\mathcal{A}, \mathcal{I})$ and its corresponding subspace solution (x, y, z), we define the *violated sets* of indices of variables as given by

$$\mathcal{V}_P := \{i \in \mathcal{I} : x_i > u_i\} \; \text{ and } \; \mathcal{V}_D := \{i \in \mathcal{A} : z_i < 0\}. \tag{3}$$

(Here, the subscripts P and D signify violations of primal and dual bounds, respectively. These violated sets depend on the partition $(\mathcal{A}, \mathcal{I})$; however, for brevity, we do not indicate this dependence in the notation for $\mathcal{V}_P$ and $\mathcal{V}_D$.) The following result has important consequences that we use extensively.

THEOREM 2. *Let (x, y, z) be the subspace solution corresponding to a given partition $(\mathcal{A}, \mathcal{I})$. Then, $(\mathcal{A}, \mathcal{I})$ is optimal for* (QP) *if and only if $\mathcal{V}_P \cup \mathcal{V}_D = \emptyset$.*

Proof. Observing the KKT conditions for (QP), the subspace solution defined by (2) satisfies all KKT conditions, except perhaps a subset of the bounds $x_{\mathcal{I}} \leq u_{\mathcal{I}}$ or $z_{\mathcal{A}} \geq 0$ (which are embedded in the KKT condition $\min\{u - x_{\mathcal{B}}, z\} = 0$). At least one of these bounds is violated if and only if the set $\mathcal{V}_P \cup \mathcal{V}_D$ is nonempty. □

Consider the framework for solving (QP) stated as Algorithm 1 below. Each iteration involves the computation of a subspace solution by (2). If the corresponding primal-dual solution yields a zero (or sufficiently small, corresponding to an arbitrary vector norm $\|\cdot\|$) KKT residual, then the solution is (approximately) optimal and the algorithm terminates. Otherwise, subsets of the corresponding violated sets—the union of which is guaranteed by Theorem 2 to be nonempty—are chosen, the indices of which are switched from active to inactive, or vice versa, to create a new partition. This algorithm represents a generic framework that allows much flexibility, such as in the choices for $\mathcal{C}_P$ and $\mathcal{C}_D$ in step 7. In section 3, we propose three algorithms related to Algorithm 1 that allow inexactness in each subspace solution; along with these algorithms, other details are provided such as how one might choose $\mathcal{C}_P$ and $\mathcal{C}_D$.

The algorithm in [28] can be viewed as a special case of Algorithm 1. In particular, for the case when $m = 0$ (i.e., $\mathcal{F} = \emptyset$ and $\mathcal{E} = \emptyset$), it corresponds to Algorithm 1 with the choice $\mathcal{C}_P \leftarrow \mathcal{V}_P$ and $\mathcal{C}_D \leftarrow \mathcal{V}_D$ in step 7 in each iteration. (Note, however, that

Algorithm 1. PDAS framework.

```
1: Input an initial partition $(\mathcal{A},\mathcal{I})$ and optimality tolerance $\varepsilon_{opt} \geq 0$.
2: loop
3:     Compute the subspace solution $(x,y,z)$ by (2).
4:     if $\|\mathrm{KKT}(x,y,z)\| \leq \varepsilon_{opt}$ then
5:         Terminate and return $(x,y,z)$.
6:     Set $\mathcal{V}_P$ and $\mathcal{V}_D$ by (3).
7:     Choose $\mathcal{C}_P \subseteq \mathcal{V}_P$ and $\mathcal{C}_D \subseteq \mathcal{V}_D$ such that $\mathcal{C}_P \cup \mathcal{C}_D \neq \emptyset$.
8:     Set $\mathcal{A} \leftarrow (\mathcal{A}\backslash\mathcal{C}_D) \cup \mathcal{C}_P$ and $\mathcal{I} \leftarrow (\mathcal{I}\backslash\mathcal{C}_P) \cup \mathcal{C}_D$.
```

the algorithm in [28] is initialized with a primal-dual iterate, not an initial partition, meaning that this correspondence with Algorithm 1 holds only after it has performed one iteration.) The authors of [28] provide convergence results for their algorithm that are similar to those in Theorem 3 below. For our purposes, we state the theorem in a more general setting so that it applies for Algorithm 1 above. A proof is given in Appendix A. For the result, recall that a real symmetric matrix is a P-matrix if all of its principal minors are positive (implying that the matrix is positive definite), and a P-matrix is called an M-matrix if all of its off-diagonal entries are nonpositive.

THEOREM 3. *Suppose that Assumption* 1 *holds and that the matrix R in* (1) *satisfies at least one of the following conditions:*

(a) *R is a P-matrix and, corresponding to any partition $(\mathcal{A},\mathcal{I})$, we have that $\|[R_{\mathcal{I}\mathcal{I}}^{-1}R_{\mathcal{I}\mathcal{A}}]_+\|_1 < 1$ and $e^T R_{\mathcal{I}\mathcal{I}}^{-1} w \geq 0$ for any $w \geq 0$, where the latter inequality holds strictly, i.e., $e^T R_{\mathcal{I}\mathcal{I}}^{-1} w > 0$, whenever $w \neq 0$.*

(b) *$R = M + E$, where M is an M-matrix and $\|E\|_1$ is sufficiently small.*

Then, with $\varepsilon_{opt} \geq 0$ and any initial partition, Algorithm 1 *terminates in a finite number of iterations. In particular, if $\varepsilon_{opt} = 0$, then Algorithm* 1 *terminates in a finite number of iterations with a KKT point for* (QP).

We remark that, under condition (b) in Theorem 3, a notion of how small $\|E\|_1$ must be in the theorem is revealed in the proof in Appendix A.

3. Algorithm descriptions.

In this section, we propose three algorithms for solving (QP). Each algorithm has the same basic structure as Algorithm 1 but allows inexactness in the linear system solves. In the first algorithm that we propose, a tolerance for inexactness is set based on an upper bound on a norm of a particular submatrix. We illustrate that such a bound can be computed efficiently in certain cases of interest. In the second and third algorithms, the inexactness tolerance is set based on a parameter that is updated dynamically within the algorithms. For the first two algorithms, we prove that the guarantees of Theorem 3 are maintained. As for the third algorithm, we argue that while it may have practical advantages, its convergence guarantees are not as strong as for the first two algorithms.

The algorithms in this section employ a relaxation of the operations stated in (2). In particular, corresponding to a partition $(\mathcal{A},\mathcal{I})$, we define an *inexact subspace solution*, call it $(\tilde{x},\tilde{y},\tilde{z})$, by the following operations (where by "$\approx$" in (4b) we are indicating that the "solve" might only be approximate):

$$\text{Set } \tilde{x}_{\mathcal{A}} \leftarrow u_{\mathcal{A}} \text{ and } \tilde{z}_{\mathcal{I}} \leftarrow 0, \tag{4a}$$

$$\text{(4b)}\qquad \text{then solve}\quad \begin{bmatrix} H_{\mathcal{I}\mathcal{I}} & H_{\mathcal{I}\mathcal{F}} & A_{\mathcal{E}\mathcal{I}}^T \\ H_{\mathcal{F}\mathcal{I}} & H_{\mathcal{F}\mathcal{F}} & A_{\mathcal{E}\mathcal{F}}^T \\ A_{\mathcal{E}\mathcal{I}} & A_{\mathcal{E}\mathcal{F}} & 0 \end{bmatrix} \begin{bmatrix} x_{\mathcal{I}} \\ x_{\mathcal{F}} \\ y \end{bmatrix} \approx - \begin{bmatrix} c_{\mathcal{I}} \\ c_{\mathcal{F}} \\ -b \end{bmatrix} - \begin{bmatrix} H_{\mathcal{I}\mathcal{A}} \\ H_{\mathcal{F}\mathcal{A}} \\ A_{\mathcal{E}\mathcal{A}} \end{bmatrix} u_{\mathcal{A}}$$

$$\text{for}\quad (\tilde{x}_{\mathcal{I}}, \tilde{x}_{\mathcal{F}}, \tilde{y}),$$

$$\text{(4c)}\qquad \text{then set}\quad \tilde{z}_{\mathcal{A}} \leftarrow -H_{\mathcal{A}\mathcal{B}}\tilde{x}_{\mathcal{B}} - H_{\mathcal{A}\mathcal{F}}\tilde{x}_{\mathcal{F}} - A_{\mathcal{E}\mathcal{A}}^T\tilde{y} - c_{\mathcal{A}}$$

$$\text{(4d)}\qquad \text{and}\quad \begin{bmatrix} \tilde{r}_{\mathcal{B}} \\ \tilde{r}_{\mathcal{F}} \\ \tilde{t} \end{bmatrix} \leftarrow \begin{bmatrix} c_{\mathcal{B}} \\ c_{\mathcal{F}} \\ -b \end{bmatrix} + \begin{bmatrix} H_{\mathcal{B}\mathcal{B}} & H_{\mathcal{B}\mathcal{F}} & A_{\mathcal{E}\mathcal{B}}^T \\ H_{\mathcal{F}\mathcal{B}} & H_{\mathcal{F}\mathcal{F}} & A_{\mathcal{E}\mathcal{F}}^T \\ A_{\mathcal{E}\mathcal{B}} & A_{\mathcal{E}\mathcal{F}} & 0 \end{bmatrix} \begin{bmatrix} \tilde{x}_{\mathcal{B}} \\ \tilde{x}_{\mathcal{F}} \\ \tilde{y} \end{bmatrix} + \begin{bmatrix} \tilde{z} \\ 0 \\ 0 \end{bmatrix}.$$

The vector $(\tilde{r}, \tilde{t}) = (\tilde{r}_{\mathcal{B}}, \tilde{r}_{\mathcal{F}}, \tilde{t})$ is the residual in the linear system solve that produces $(\tilde{x}, \tilde{y}, \tilde{z})$ via (4b). Under Assumption 1, we find by comparing (2) and (4) that one obtains $(\tilde{x}, \tilde{y}, \tilde{z}) = (x, y, z)$ if and only if $(\tilde{r}, \tilde{t}) = 0$.

In each of the algorithms proposed in this section, we iteratively solve the linear system in (4b) until either it is verified that the partition $(\mathcal{A}, \mathcal{I})$ is optimal (with respect to a tolerance $\varepsilon_{opt} \geq 0$) or the inexact subspace solution is sufficiently accurate in that it leads to a productive update of the partition. In our first algorithm, we provide a strategy in which we identify subsets of the violated sets $\mathcal{V}_P$ and $\mathcal{V}_D$ corresponding to the *exact* subspace solution (x, y, z) *without* having to explicitly compute this exact solution. In this manner, the algorithm fits into the framework of Algorithm 1. In our other algorithms, we do not necessarily identify subsets of these violated sets, though we can still ensure certain convergence guarantees by employing and appropriately updating a dynamic algorithmic parameter.

3.1. An algorithm with a partition-defined subproblem tolerance. Our first algorithm imposes a tolerance on the residual $(\tilde{r}, \tilde{t})$ defined in (4d) that is based on a partition-defined value with which we can guarantee that at any suboptimal partition, a subset of $\mathcal{V}_P \cup \mathcal{V}_D$ corresponding to the *exact* subspace solution (x, y, z) will be identified. In order to motivate the tolerance that we employ, we first explore, for a given partition $(\mathcal{A}, \mathcal{I})$, the relationship between the subspace solution (x, y, z) defined by (2) and an inexact subspace solution $(\tilde{x}, \tilde{y}, \tilde{z})$ defined by (4). Then, once the tolerance is established, we present our first inexact PDAS algorithm, followed by subsections in which we discuss details of its subroutines. It is important to note that we assume that a subroutine is available for computing *exact* solutions of linear systems with $A_{\mathcal{E}\mathcal{F}}$ and its transpose $A_{\mathcal{E}\mathcal{F}}^T$, i.e., we assume products with $A_{\mathcal{E}\mathcal{F}}^{-1}$ and $A_{\mathcal{E}\mathcal{F}}^{-T}$ can be computed. This is a reasonable assumption in certain applications, especially since the matrix $A_{\mathcal{E}\mathcal{F}}$ is fixed, i.e., it does not depend on the partition.

We first establish some quantities that will be useful in the statement of our algorithms. Given $(\mathcal{A}, \mathcal{I})$, we define and decompose the KKT system matrix as

$$\text{(5)}\qquad K := \begin{bmatrix} H & A^T \\ A & 0 \end{bmatrix} = \begin{bmatrix} H_{\mathcal{A}\mathcal{A}} & H_{\mathcal{A}\mathcal{I}} & S_{[\mathcal{A}]}^T \\ H_{\mathcal{I}\mathcal{A}} & H_{\mathcal{I}\mathcal{I}} & S_{[\mathcal{I}]}^T \\ S_{[\mathcal{A}]} & S_{[\mathcal{I}]} & Q \end{bmatrix},$$

where (using a subscript $[\cdot]$ to indicate dependence on an index set) we define

$$S_{[\cdot]} := \begin{bmatrix} H_{\mathcal{F}\cdot} \\ A_{\mathcal{E}\cdot} \end{bmatrix} \quad \text{and} \quad Q := \begin{bmatrix} H_{\mathcal{F}\mathcal{F}} & A_{\mathcal{E}\mathcal{F}}^T \\ A_{\mathcal{E}\mathcal{F}} & 0 \end{bmatrix}.$$

With these quantities, one can verify that an alternative representation of the reduced

Hessian R in (1) is that it is the Schur complement of the submatrix Q of K, i.e.,

$$R = H_{\mathcal{B}\mathcal{B}} - S_{[\mathcal{B}]}^T Q^{-1} S_{[\mathcal{B}]}, \quad \text{where} \quad Q^{-1} = \begin{bmatrix} 0 & A_{\mathcal{E}\mathcal{F}}^{-1} \\ A_{\mathcal{E}\mathcal{F}}^{-T} & -A_{\mathcal{E}\mathcal{F}}^{-T} H_{\mathcal{F}\mathcal{F}} A_{\mathcal{E}\mathcal{F}}^{-1} \end{bmatrix}. \tag{6}$$

Similarly, the submatrix of the reduced Hessian corresponding only to the indices in the inactive set $\mathcal{I}$ can be expressed as

$$R_{\mathcal{I}\mathcal{I}} = \begin{bmatrix} I \\ -A_{\mathcal{E}\mathcal{F}}^{-1} A_{\mathcal{E}\mathcal{I}} \end{bmatrix}^T \begin{bmatrix} H_{\mathcal{I}\mathcal{I}} & H_{\mathcal{I}\mathcal{F}} \\ H_{\mathcal{F}\mathcal{I}} & H_{\mathcal{F}\mathcal{F}} \end{bmatrix} \begin{bmatrix} I \\ -A_{\mathcal{E}\mathcal{F}}^{-1} A_{\mathcal{E}\mathcal{I}} \end{bmatrix} = H_{\mathcal{I}\mathcal{I}} - S_{[\mathcal{I}]}^T Q^{-1} S_{[\mathcal{I}]},$$

i.e., it is the Schur complement of Q of the bottom-right 2×2 block submatrix of (5).

We now establish useful relationships between the subspace solution and a given inexact subspace solution. Observing (2) and (4), it follows that $\tilde{x}_{\mathcal{A}} = u_{\mathcal{A}} = x_{\mathcal{A}}$ and $\tilde{z}_{\mathcal{I}} = 0 = z_{\mathcal{I}}$. In addition, defining the residual subvector $\tilde{v} := (\tilde{r}_{\mathcal{F}}, \tilde{t})$, we have

$$x_{\mathcal{I}} = \tilde{x}_{\mathcal{I}} - R_{\mathcal{I}\mathcal{I}}^{-1} \tilde{r}_{\mathcal{I}} + R_{\mathcal{I}\mathcal{I}}^{-1} S_{[\mathcal{I}]}^T Q^{-1} \tilde{v} \tag{7a}$$

$$\begin{aligned} \text{and} \quad z_{\mathcal{A}} = \tilde{z}_{\mathcal{A}} &+ (H_{\mathcal{A}\mathcal{I}} - S_{[\mathcal{A}]}^T Q^{-1} S_{[\mathcal{I}]}) R_{\mathcal{I}\mathcal{I}}^{-1} \tilde{r}_{\mathcal{I}} \\ &- (H_{\mathcal{A}\mathcal{I}} R_{\mathcal{I}\mathcal{I}}^{-1} S_{[\mathcal{I}]}^T - S_{[\mathcal{A}]}^T - S_{[\mathcal{A}]}^T Q^{-1} S_{[\mathcal{I}]} R_{\mathcal{I}\mathcal{I}}^{-1} S_{[\mathcal{I}]}^T) Q^{-1} \tilde{v}. \end{aligned} \tag{7b}$$

Observing (7), it follows that the violated sets $\mathcal{V}_P$ and $\mathcal{V}_D$ corresponding to (x, y, z) can be defined in terms of an inexact subspace solution $(\tilde{x}, \tilde{y}, \tilde{z})$ and its residual $(\tilde{r}, \tilde{t})$. In particular, for an index $i \in \mathcal{I}$, the ith element of x violates its upper bound if the corresponding element on the right-hand side of (7a) is greater than u_i, and, for an index $i \in \mathcal{A}$, the ith element of z violates its lower bound (of zero) if the corresponding element on the right-hand side of (7b) is negative. This revised viewpoint of the elements of $x_{\mathcal{I}}$ and $z_{\mathcal{A}}$ does not immediately yield any benefits since the evaluation of the terms on the right-hand sides of (7) is equivalent to that of solving (4b) exactly. However, it reveals that with an inexact subspace solution $(\tilde{x}, \tilde{y}, \tilde{z})$ and *bounds* on the residual terms in (7), one may identify subsets of $\mathcal{V}_P$ and $\mathcal{V}_D$ without computing (x, y, z) explicitly. The following lemma suggests a strategy for such a procedure.

LEMMA 4. *Given a partition $(\mathcal{A}, \mathcal{I})$, let (x, y, z) be the corresponding subspace solution and let $(\tilde{x}, \tilde{y}, \tilde{z})$ be an inexact subspace solution with residual $(\tilde{r}, \tilde{t})$. Furthermore, suppose that with $\tilde{v} := (\tilde{r}_{\mathcal{F}}, \tilde{t})$ we have $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ satisfying*

$$\alpha_{\mathcal{I}} \geq R_{\mathcal{I}\mathcal{I}}^{-1} \tilde{r}_{\mathcal{I}} - R_{\mathcal{I}\mathcal{I}}^{-1} S_{[\mathcal{I}]}^T Q^{-1} \tilde{v} \tag{8a}$$

$$\begin{aligned} \textit{and} \quad \beta_{\mathcal{A}} \geq\ &(H_{\mathcal{A}\mathcal{I}} - S_{[\mathcal{A}]}^T Q^{-1} S_{[\mathcal{I}]}) R_{\mathcal{I}\mathcal{I}}^{-1} \tilde{r}_{\mathcal{I}} \\ &- (H_{\mathcal{A}\mathcal{I}} R_{\mathcal{I}\mathcal{I}}^{-1} S_{[\mathcal{I}]}^T - S_{[\mathcal{A}]}^T - S_{[\mathcal{A}]}^T Q^{-1} S_{[\mathcal{I}]} R_{\mathcal{I}\mathcal{I}}^{-1} S_{[\mathcal{I}]}^T) Q^{-1} \tilde{v}. \end{aligned} \tag{8b}$$

Then, for the violated sets $\mathcal{V}_P$ and $\mathcal{V}_D$ corresponding to (x, y, z), we have

$$\tilde{\mathcal{V}}_P := \{i \in \mathcal{I} : \tilde{x}_i - \alpha_i > u_i\} \subseteq \mathcal{V}_P \quad \textit{and} \quad \tilde{\mathcal{V}}_D := \{i \in \mathcal{A} : \tilde{z}_i + \beta_i < 0\} \subseteq \mathcal{V}_D. \tag{9}$$

Moreover, if $(\mathcal{A}, \mathcal{I})$ is suboptimal, then there exists $\varepsilon > 0$ (dependent on the partition) such that the relationships (8)–(9) with

$$\|\alpha_{\mathcal{I}}\| = \mathcal{O}(\|(\tilde{r}, \tilde{t})\|) \quad \textit{and} \quad \|\beta_{\mathcal{A}}\| = \mathcal{O}(\|(\tilde{r}, \tilde{t})\|) \tag{10}$$

yield $\tilde{\mathcal{V}}_P = \mathcal{V}_P$ and $\tilde{\mathcal{V}}_D = \mathcal{V}_D$ for any inexact subspace solution with $\|(\tilde{r}, \tilde{t})\| \leq \varepsilon$.

Algorithm 2. PDAS framework with inexact subspace solutions.

```
 1: Input an initial partition $(\mathcal{A},\mathcal{I})$ and optimality tolerance $\varepsilon_{opt} \geq 0$.
 2: **loop**
 3:     Compute an inexact subspace solution $(\tilde{x},\tilde{y},\tilde{z})$ with residual $(\tilde{r},\tilde{t})$ by (4).
 4:     **if** $\|\mathrm{KKT}(\tilde{x},\tilde{y},\tilde{z})\| \leq \varepsilon_{opt}$ **then**
 5:         Terminate and **return** $(\tilde{x},\tilde{y},\tilde{z})$.
 6:     Compute $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ satisfying (8) and (10).
 7:     Set $\tilde{\mathcal{V}}_P$ and $\tilde{\mathcal{V}}_D$ by (9).
 8:     **if** $\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D \neq \emptyset$ **then**
 9:         Choose $\mathcal{C}_P \subseteq \tilde{\mathcal{V}}_P$ and $\mathcal{C}_D \subseteq \tilde{\mathcal{V}}_D$ such that $\mathcal{C}_P \cup \mathcal{C}_D \neq \emptyset$.
10:         Set $\mathcal{A} \leftarrow (\mathcal{A}\backslash\mathcal{C}_D) \cup \mathcal{C}_P$ and $\mathcal{I} \leftarrow (\mathcal{I}\backslash\mathcal{C}_P) \cup \mathcal{C}_D$.
```

Note. If an iteration of the **loop** ends with $\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D = \emptyset$, then it is assumed that the inexact subspace solution subsequently computed in step 3 is improved such that the conditions of Theorem 5 hold.

Proof. By (8) and the relationships in (7), we have that for $i \in \mathcal{I}$ the inequality $\tilde{x}_i - \alpha_i > u_i$ implies $x_i > u_i$, and for $i \in \mathcal{A}$ the inequality $\tilde{z}_i + \beta_i < 0$ implies $z_i < 0$. Hence, by the definitions in (9), it follows that $\tilde{\mathcal{V}}_P \subseteq \mathcal{V}_P$ and $\tilde{\mathcal{V}}_D \subseteq \mathcal{V}_D$.

Now suppose that $(\mathcal{A},\mathcal{I})$ is suboptimal, from which it follows by Theorem 2 that $\mathcal{V}_P \cup \mathcal{V}_D \neq \emptyset$. If $\mathcal{V}_P \neq \emptyset$, then for any $i \in \mathcal{V}_P$ we have $x_i > u_i$. Moreover, by continuity of the linear transformation defined by the inverse of the matrix on the left-hand side of (4b), for this $i \in \mathcal{V}_P$ there exists $\varepsilon_i > 0$ such that for any $(\tilde{r},\tilde{t})$ with $\|(\tilde{r},\tilde{t})\| \leq \varepsilon_i$, the condition (10) implies $\tilde{x}_i - \alpha_i > u_i$. Similar analysis shows that if $\mathcal{V}_D \neq \emptyset$, then for any $i \in \mathcal{V}_D$ there exists $\varepsilon_i > 0$ such that for any $(\tilde{r},\tilde{t})$ with $\|(\tilde{r},\tilde{t})\| \leq \varepsilon_i$, the condition (10) implies $\tilde{z}_i + \beta_i < 0$. Since $\tilde{\mathcal{V}}_P \subseteq \mathcal{V}_P$ and $\tilde{\mathcal{V}}_D \subseteq \mathcal{V}_D$, it follows that with $\varepsilon := \min\{\varepsilon_i : i \in \mathcal{V}_P \cup \mathcal{V}_D\} > 0$, we have $\tilde{\mathcal{V}}_P = \mathcal{V}_P$ and $\tilde{\mathcal{V}}_D = \mathcal{V}_D$. ☐

Lemma 4 proves that at any suboptimal partition $(\mathcal{A},\mathcal{I})$, a subset of the union of violated sets $\mathcal{V}_P \cup \mathcal{V}_D$ can be identified as long as upper bounds $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ are available and are proportional (in terms of a norm $\|\cdot\|$) to the residual vector $(\tilde{r},\tilde{t})$. On the other hand, if a partition $(\mathcal{A},\mathcal{I})$ is optimal, then with a sufficiently small residual we obtain a sufficiently accurate primal-dual solution. Motivated by these observations, we propose the inexact PDAS framework presented as Algorithm 2.

As indicated in the note in Algorithm 2, the intention is that each execution of step 3 corresponding to a partition yields a better inexact subspace solution, in the sense that the residual ultimately will be made arbitrarily small in norm. The following result formalizes this assumption in a manner that ensures convergence.

THEOREM 5. *Suppose that the conditions of Theorem* 3 *hold for the partitions generated by Algorithm* 2*. Then, the following hold:*

(i) *If, for any partition, repeated executions of step* 3 *yield* $(\tilde{r},\tilde{t}) \to 0$*, then, with* $\varepsilon_{opt} > 0$*, Algorithm* 2 *terminates after a finite number of partition updates.*

(ii) *If there exists a positive integer J such that, for any partition, at most J executions of step 3 yields* $(\tilde{r},\tilde{t}) = 0$*, then, with* $\varepsilon_{opt} \geq 0$*, Algorithm* 2 *terminates in a finite number of iterations. In particular, if* $\varepsilon_{opt} = 0$*, then Algorithm* 2 *terminates in a finite number of iterations with a KKT point for* (QP)*.*

Proof. If the algorithm encounters an optimal partition, then the conditions in (i) or (ii) ensure that a finite number of executions of step 3 will lead to satisfaction of the termination condition in step 4. On the other hand, given any suboptimal partition, it follows by the conditions in (i) and Lemma 4 that after a finite number of executions of step 3, the sets $\tilde{\mathcal{V}}_P \subseteq \mathcal{V}_P$ and $\tilde{\mathcal{V}}_D \subseteq \mathcal{V}_D$ defined by (9) satisfy $\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D \neq \emptyset$. The result in (i) then follows from Theorem 3 and the fact that after a finite number of partition updates, a partition is identified such that repeated executions of step 3 yield $(\tilde{x}, \tilde{y}, \tilde{z})$ satisfying the termination condition in step 4. The result in (ii) follows in a similar manner due to the additional observation that, given any partition, at most J executions of step 3 occur before the condition in step 4 is satisfied or the partition is modified. $\square$

There remain details that need to be specified for a practical implementation of Algorithm 2. These details are the subjects of the following three subsections. First, since upper bounds $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ in step 6 are easily computed once one obtains an upper bound for the norm of the matrix $R_{\mathcal{I}\mathcal{I}}^{-1}$, we present an algorithm for computing such a bound in certain cases of interest. Second, we present a technique for computing $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ once such a bound is obtained. Third, we outline conditions that one may choose to impose—in addition to $\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D \neq \emptyset$—in the **if** statement in step 8.

3.1.1. Obtaining an upper bound for $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$. In this subsection, given an index set $\mathcal{I}$, we present a technique for computing an upper bound for $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$, which could also provide upper bounds for other norms of $R_{\mathcal{I}\mathcal{I}}^{-1}$. Our technique amounts to solving a linear system of equations by an iterative process. (Under certain conditions, an exact solution of the linear system reveals $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$.) In this manner, it is clear how one may perform Algorithm 2, step 6: for a partition $(\mathcal{A}, \mathcal{I})$, repeated executions of the subroutine in this subsection eventually lead to an upper bound for $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$, which in turn eventually lead to upper bounds $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ via the technique in section 3.1.2.

Given a $p \times p$ symmetric positive definite matrix B, consider the set

$$\mathcal{P}(B) := \{w \in \mathbb{R}^p : w > 0,\ \mathfrak{M}(B)w > 0,\ \|w\|_\infty = 1\},$$

where the symmetric $p \times p$ matrix $\mathfrak{M}(B)$ is defined, for all $\{i, j\} \subseteq \{1, \ldots, p\}$, by

$$[\mathfrak{M}(B)]_{ij} = \begin{cases} |B_{ii}| & \text{if } i = j, \\ -|B_{ij}| & \text{if } i \neq j. \end{cases}$$

The matrix B is called a nonsingular H-matrix if and only if $\mathfrak{M}(B)$ is a nonsingular M-matrix. Moreover, by [53, eq. (5)], the following three statements are equivalent:

1. B is a nonsingular H-matrix.
2. $\mathfrak{M}(B)$ is a nonsingular M-matrix.
3. $\mathcal{P}(B)$ is nonempty.

We also have the following result, which follows from [53, Lemma 1]. (In [53], the author discusses upper bounds for the ∞-norm of a matrix inverse. However, since our matrix is symmetric, we can equivalently refer to its 1-norm.)

LEMMA 6. *If $B \in \mathbb{R}^{p \times p}$ is a symmetric nonsingular H-matrix, then*

$$\|B^{-1}\|_1 \leq (\min\{[Bw]_i : 1 \leq i \leq p\})^{-1} \tag{11}$$

for any $w \in \mathcal{P}(B)$.

By Lemma 6, if $R_{\mathcal{I}\mathcal{I}}$ is an H-matrix, then we can obtain an upper bound for $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$ by iteratively solving the linear system $\mathfrak{M}(R_{\mathcal{I}\mathcal{I}})w = e$ for $w \in \mathbb{R}^{|\mathcal{I}|}$, terminating whenever an element of $\mathcal{P}(R_{\mathcal{I}\mathcal{I}})$ is obtained. We formalize this strategy as the following procedure, for which we have the subsequent result (the last conclusion of which motivates the choice of e as right-hand side vector in (12a)):

$$\text{Iteratively solve } \ \mathfrak{M}(R_{\mathcal{I}\mathcal{I}})w \approx e \ \text{ for } \ w \tag{12a}$$

$$\text{until } \mathfrak{M}(R_{\mathcal{I}\mathcal{I}})w > 0, \tag{12b}$$

$$\text{then return } \|w\|_\infty(\min\{[R_{\mathcal{I}\mathcal{I}}w]_i : 1 \le i \le |\mathcal{I}|\})^{-1}. \tag{12c}$$

Lemma 7. *Suppose Assumption* 1 *holds and R satisfies $R = M + E$, where M is an M-matrix and $\|E\|_1$ is sufficiently small. Then, for any $\mathcal{I} \subseteq \mathcal{B}$, if the iterative solver employed in* (12a) *would otherwise yield $\mathfrak{M}(R_{\mathcal{I}\mathcal{I}})w \to e$, then the procedure* (12) *will terminate finitely and return an upper bound for $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$. Moreover, if $R = M$, then an exact solution in* (12a) *yields $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$ via* (12c).

Proof. Since $R = M + E$, we have $R_{\mathcal{I}\mathcal{I}} = M_{\mathcal{I}\mathcal{I}} + E_{\mathcal{I}\mathcal{I}}$ for some M-matrix $M_{\mathcal{I}\mathcal{I}}$. Since $M_{\mathcal{I}\mathcal{I}}$ is an M-matrix, it is also an H-matrix since $\mathfrak{M}(M_{\mathcal{I}\mathcal{I}}) = M_{\mathcal{I}\mathcal{I}}$. Then, since $M_{\mathcal{I}\mathcal{I}}$ is an H-matrix, it follows that for sufficiently small $\|E\|_1$ we have sufficiently small $\|E_{\mathcal{I}\mathcal{I}}\|_1$ such that $R_{\mathcal{I}\mathcal{I}}$ is also an H-matrix. The result then follows by Lemma 6 since, under the conditions of the lemma, the algorithm eventually computes w satisfying $\mathfrak{M}(R_{\mathcal{I}\mathcal{I}})w > 0$.

Now, if $R_{\mathcal{I}\mathcal{I}}$ is an M-matrix, then $\mathfrak{M}(R_{\mathcal{I}\mathcal{I}}) = R_{\mathcal{I}\mathcal{I}}$ and $\mathfrak{M}(R_{\mathcal{I}\mathcal{I}})^{-1} = R_{\mathcal{I}\mathcal{I}}^{-1} \ge 0$. Thus, for $w = \mathfrak{M}(R_{\mathcal{I}\mathcal{I}})^{-1}e = R_{\mathcal{I}\mathcal{I}}^{-1}e$, one has $\|w\|_\infty = \|R_{\mathcal{I}\mathcal{I}}^{-1}\|_\infty = \|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$, from which it follows that (12c) returns $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$. ∎

We close this subsection by remarking that if $R_{\mathcal{I}\mathcal{I}}$ is strictly diagonally dominant, then $w = e$ yields $\mathfrak{M}(R_{\mathcal{I}\mathcal{I}})w > 0$ and one has

$$\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1 \le (\min\{[R_{\mathcal{I}\mathcal{I}}e]_i : 1 \le i \le |\mathcal{I}|\})^{-1}.$$

This is known as the Ahlberg–Nilson–Varah bound [2, 52].

3.1.2. Obtaining upper bounds $\boldsymbol{\alpha_{\mathcal{I}}}$ and $\boldsymbol{\beta_{\mathcal{A}}}$. Given a partition $(\mathcal{A},\mathcal{I})$, an inexact subspace solution $(\tilde{x}, \tilde{y}, \tilde{z})$ with residual $(\tilde{r}, \tilde{t})$, and a scalar $\gamma \ge 0$ satisfying $\gamma \ge \|R_{\mathcal{I}\mathcal{I}}^{-1}\|_p$ (say, computed via procedure (12)), the following procedure (with $\|\cdot\|_q$ denoting the dual norm of $\|\cdot\|_p$) yields $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ satisfying (8) and (10):

$$\text{Set } \alpha_{\mathcal{I}} \leftarrow (\gamma\|\tilde{r}_{\mathcal{I}} - S_{[\mathcal{I}]}Q^{-1}\tilde{v}\|_q)e \tag{13a}$$

$$\begin{aligned} \text{and } \beta_{\mathcal{A}} \leftarrow {} & [H_{\mathcal{A}\mathcal{I}} - S_{[\mathcal{A}]}^T Q^{-1} S_{[\mathcal{I}]}]_+ \alpha_{\mathcal{I}} \\ & - [H_{\mathcal{A}\mathcal{I}} - S_{[\mathcal{A}]}^T Q^{-1} S_{[\mathcal{I}]}]_- \alpha_{\mathcal{I}} + S_{[\mathcal{A}]} Q^{-1} \tilde{v}. \end{aligned} \tag{13b}$$

The fact that $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ resulting from this procedure satisfy (8) and (10) follows as a straightforward consequence of Hölder's inequality.

As for the cost of procedure (13), recall from (6) that the components of Q^{-1} are independent of the partition. Thus, the computational cost of executing (13) need not be prohibitive, especially if $A_{\mathcal{E}\mathcal{F}}$ is prefactored and/or the matrix $Q^{-1}S_{[\mathcal{B}]}$ is precomputed. We remark that when employing procedure (12) to compute an upper bound for $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$, it is natural to employ procedure (13) with $p = 1$ and $q = \infty$. This is the approach used in our implementation and numerical experiments.

3.1.3. Conditions for executing a partition update. We close our discussion of Algorithm 2 by describing conditions that one may choose to impose—in addition to the requirement that $\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D \neq \emptyset$—in the **if** statement in step 8. As already shown in Theorem 5, the convergence of the algorithm is guaranteed with the condition as stated; i.e., additional conditions are not necessary to ensure convergence. However, in our experience, we have found it worthwhile to impose extra conditions to ensure that any update to the partition that is performed will lead to substantial progress toward a solution, which can be expected when either the inexact subspace solution is sufficiently accurate and/or the modification to the partition will involve a large number of indices switching from active to inactive, or vice versa. We have found the conditions that we state in this section to work well in practice, though one may imagine other possible conditions that could be imposed.

Let $(\tilde{x}', \tilde{y}', \tilde{z}')$ be a given inexact subspace solution with residual $(\tilde{r}', \tilde{t}')$. For example, one may consider $(\tilde{x}', \tilde{y}', \tilde{z}') = (0, 0, 0)$ or the inexact subspace solution corresponding to primal-dual variable values as computed in the previous iteration of Algorithm 2. Given a tolerance $\epsilon_{res} \in (0, 1)$ and a vector norm $\|\cdot\|$, a condition that one may choose to impose is the following, similar to conditions typically found in inexact Newton methods for solving systems of equations [16]:

$$\|(\tilde{r}, \tilde{t})\| \leq \epsilon_{res} \|(\tilde{r}', \tilde{t}')\|. \tag{14}$$

That is, one may choose not to modify the partition until the residual vector $(\tilde{r}, \tilde{t})$ is sufficiently small in norm compared to the reference residual $(\tilde{r}', \tilde{t}')$ corresponding to the reference solution $(\tilde{x}', \tilde{y}', \tilde{z}')$. If the right-hand side of (14) is zero, then $(x, y, z) = (\tilde{x}', \tilde{y}', \tilde{z}')$; otherwise, (14) will eventually be satisfied as long as the employed iterative solver ensures that the residual vanishes, i.e., $(\tilde{r}, \tilde{t}) \to 0$.

In our experience, we have also found it beneficial to avoid modifying the partition until there is consistency between the sets $\tilde{\mathcal{V}}_P$ and $\tilde{\mathcal{V}}_D$ in (9) and

$$\tilde{\mathcal{V}}'_P := \{i \in \mathcal{I} : \tilde{x}_i > u_i\} \text{ and } \tilde{\mathcal{V}}'_D := \{i \in \mathcal{A} : \tilde{z}_i < 0\}. \tag{15}$$

Specifically, we have found it beneficial to avoid modifying the partition until the number of elements of the violated sets that have been identified, namely, $|\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D|$, is proportional to the number of elements of the primal-dual components that violate their bounds, namely, $|\tilde{\mathcal{V}}'_P \cup \tilde{\mathcal{V}}'_D|$. Given a parameter $\theta \in (0, 1]$, we employ

$$|\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D| \geq \theta |\tilde{\mathcal{V}}'_P \cup \tilde{\mathcal{V}}'_D|. \tag{16}$$

Observe that procedure (13) yields $\alpha_{\mathcal{I}} \geq 0$ and $\beta_{\mathcal{A}} \geq 0$, from which it follows that $|\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D| \leq |\tilde{\mathcal{V}}'_P \cup \tilde{\mathcal{V}}'_D|$. This justifies the restriction that $\theta \in (0, 1]$.

3.2. Algorithms with dynamic subproblem tolerances. We are now prepared to present our second and third inexact PDAS algorithms. For a given partition $(\mathcal{A}, \mathcal{I})$, the main idea underlying our first method—i.e., Algorithm 2 presented in section 3.1—was to use properties of inexact subspace solutions and their corresponding residuals in order to construct explicit subsets of the violated sets $\mathcal{V}_P$ and $\mathcal{V}_D$ corresponding to the exact subspace solution, all without having to explicitly compute this exact solution. Unfortunately, however, the procedure that we proposed required an explicit upper bound for a norm of $R_{\mathcal{I}\mathcal{I}}^{-1}$, which may be expensive to compute in certain situations, especially when a tight bound is needed to identify elements of the violated sets. By contrast, the algorithms that we propose in this section do not require explicit upper bounds of this type; instead, they involve dynamic parameters either to estimate such an upper bound or to control inexactness directly.

Our second algorithm, stated as Algorithm 3 below, employs a dynamic parameter that plays a role similar to the upper bound for a norm of $R_{\mathcal{I}\mathcal{I}}^{-1}$ as employed in Algorithm 2 (via procedure (13)). In the worst case, this dynamic parameter will increase large enough such that it is, in fact, an upper bound for a norm of $R_{\mathcal{I}\mathcal{I}}^{-1}$ for any $\mathcal{I}$; indeed, the convergence guarantees that we present are based on this feature. However, we have designed the update for this dynamic parameter such that we rarely see such behavior in practice. Indeed, in practice, we often observe that the algorithm

Algorithm 3. PDAS framework with inexact subspace solutions (dynamic).

```
1: Input an initial partition $(\mathcal{A},\mathcal{I})$, optimality tolerance $\varepsilon_{opt} > 0$, dynamic parameter
   $\gamma > 0$, update factor $\delta_\gamma > 1$, optimality tolerance history length $\bar{j} \in \mathbb{N}$, and
   sufficient reduction factor $\kappa \in (0,1)$.
2: Initialize a partition update counter $j \leftarrow 0$ and $\mathrm{KKT}_j \leftarrow \infty$ for $j \in \{-1,\dots,-\bar{j}\}$.
3: loop
4:     Compute an inexact subspace solution $(\tilde{x},\tilde{y},\tilde{z})$ with residual $(\tilde{r},\tilde{t})$ by (4).
5:     if $\|\mathrm{KKT}(\tilde{x},\tilde{y},\tilde{z})\| \le \varepsilon_{opt}$ then
6:         Terminate and return $(\tilde{x},\tilde{y},\tilde{z})$.
7:     Compute $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$ by procedure (13) (even if $\gamma < \|R_{\mathcal{I}\mathcal{I}}^{-1}\|_p$).
8:     Set $\tilde{\mathcal{V}}_P$ and $\tilde{\mathcal{V}}_D$ by (9) (even if $\tilde{\mathcal{V}}_P \not\subseteq \mathcal{V}_P$ or $\tilde{\mathcal{V}}_D \not\subseteq \mathcal{V}_D$).
9:     if $\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D \neq \emptyset$ then
10:        Choose $\mathcal{C}_P \subseteq \tilde{\mathcal{V}}_P$ and $\mathcal{C}_D \subseteq \tilde{\mathcal{V}}_D$ such that $\mathcal{C}_P \cup \mathcal{C}_D \neq \emptyset$.
11:        Set $\mathcal{A} \leftarrow (\mathcal{A}\backslash\mathcal{C}_D) \cup \mathcal{C}_P$ and $\mathcal{I} \leftarrow (\mathcal{I}\backslash\mathcal{C}_P) \cup \mathcal{C}_D$.
12:        Set $\mathrm{KKT}_j \leftarrow \|\mathrm{KKT}(\tilde{x},\tilde{y},\tilde{z})\|$.
13:        if $\mathrm{KKT}_j \ge \kappa \max\{\mathrm{KKT}_{j-1},\dots,\mathrm{KKT}_{j-\bar{j}}\}$ then
14:            Set $\gamma \leftarrow \delta_\gamma \gamma$.
15:        Set $j \leftarrow j+1$.
```

Note. If an iteration of the **loop** ends with $\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D = \emptyset$, then it is assumed that the inexact subspace solution subsequently computed in step 4 is improved such that the conditions of Theorem 8 hold.

terminates for relatively small values of this dynamic parameter. As in Algorithm 2, the norm used in the optimality test in Algorithm 3 (step 5) can be any vector norm; we also add that, in this context, it is natural to use the same norm in step 12. On the other hand, the norm to which we refer in step 7 should be the norm $\|\cdot\|_p$ with $p \ge 1$ employed in procedure (13) for computing the values $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$.

The purpose of the sequence $\{\mathrm{KKT}_j\}$ computed in Algorithm 3 is to monitor progress in reducing the KKT error over the sequence of iterations in which the partition is modified. Specifically, if a KKT error computed in step 12 is not less than the most recent $\bar{j}$ such computed KKT errors, then the dynamic parameter γ is increased. As can be seen in procedure (13), this has the effect of yielding larger values for $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{A}}$, which in turn has the effect of producing more conservative estimates (i.e., $\tilde{\mathcal{V}}_P$ and $\tilde{\mathcal{V}}_D$) of the violated sets (i.e., $\mathcal{V}_P$ and $\mathcal{V}_D$).

We have the following theorem related to convergence properties of Algorithm 3.

THEOREM 8. *Suppose that the conditions of Theorem* 3 *hold for the partitions generated by Algorithm* 3. *Then, the following hold:*

(i) *If, for any partition, repeated executions of step* 4 *yield* $(\tilde{r},\tilde{t}) \to 0$, *then, with* $\varepsilon_{opt} > 0$, *Algorithm* 3 *terminates after a finite number of partition updates.*

(ii) *If there exists a positive integer J such that, for any partition, at most J executions of step* 4 *yields* $(\tilde{r},\tilde{t}) = 0$, *then, with* $\varepsilon_{opt} \ge 0$, *Algorithm* 3 *terminates in a finite number of iterations. In particular, if* $\varepsilon_{opt} = 0$, *then Algorithm* 3 *terminates in a finite number of iterations with a KKT point for* (QP).

Proof. If the algorithm encounters an optimal partition, then the conditions in (i) or (ii) ensure that a finite number of executions of step 4 will lead to satisfaction of the termination condition in step 5. On the other hand, given any suboptimal partition, it follows by the conditions in either (i) or (ii) that after a finite number of executions of step 4, the sets $\tilde{\mathcal{V}}_P$ and $\tilde{\mathcal{V}}_D$ defined by (9) satisfy $\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D \neq \emptyset$ (even if $\tilde{\mathcal{V}}_P \not\subseteq \mathcal{V}_P$ or $\tilde{\mathcal{V}}_D \not\subseteq \mathcal{V}_D$). Consequently, given any suboptimal partition, either Algorithm 3 will eventually terminate or a partition update will be performed. If the algorithm terminates finitely, then there is nothing left to prove. Hence, in order to derive a contradiction, suppose that an infinite number of partition updates are performed. If $\{\mathrm{KKT}_j\} \to 0$, then, under the conditions in either (i) or (ii), the condition in step 5 eventually will be satisfied; this would cause the algorithm to terminate finitely, a contradiction to our supposition that an infinite number of partition updates are performed. Thus, we may assume that $\{\mathrm{KKT}_j\}$ is bounded below by a positive constant, which, by the condition in step 13, implies that $\gamma \to \infty$. However, once

$$\gamma \geq \bar{\gamma} := \max_{\mathcal{I} \subseteq \mathcal{B}} \|R_{\mathcal{I}\mathcal{I}}^{-1}\|_p, \tag{17}$$

it follows that we will always have $\tilde{\mathcal{V}}_P \subseteq \mathcal{V}_P$ and $\tilde{\mathcal{V}}_D \subseteq \mathcal{V}_D$, implying that convergence can be guaranteed in the same manner as in the proof of Theorem 5. Since this contradicts our supposition that an infinite number of partition updates are performed, we have that the algorithm will terminate finitely, as desired. ∎

One important observation about Algorithm 3 is that it is nontrivial to choose an initial value for the dynamic parameter γ such that all iterations performed in the algorithm may be identical to those that would be performed by Algorithm 2. For example, assuming that it may be computed efficiently, one may consider an initial value of $\gamma \leftarrow \|R^{-1}\|_1$ (independent of any partition), but this value does not necessarily satisfy (17). To see this, consider the following example.

Example 9. Let

$$R = \begin{bmatrix} \frac{7}{6} & \frac{1}{6} & \frac{29}{12} \\ \frac{1}{6} & \frac{1}{6} & \frac{5}{12} \\ \frac{29}{12} & \frac{5}{12} & \frac{143}{24} \end{bmatrix} \quad \text{and} \quad \mathcal{I} = \{1, 2\}.$$

One can verify that $\|R^{-1}\|_1 = \frac{31}{4} < 8 = \|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$.

Our third inexact PDAS algorithm, presented as Algorithm 4 below, employs the straightforward heuristic of defining a dynamic tolerance for the residuals in the (reduced) linear system solves that decreases as the optimization process proceeds. In this algorithm, it is reasonable to choose the norm in step 7 as the same norm used in the KKT errors in steps 5 and 11. (We also note that a relative residual test—rather than an absolute residual test—could be employed instead, as we do in our implementation and numerical experiments.)

We have the following convergence result for Algorithm 4.

Theorem 10. *Suppose that the conditions of Theorem* 3 *hold for the partitions generated by Algorithm* 4 *and that* $\zeta = 0$. *If, for any partition, a finite number of executions of step* 4 *lead to the condition in step* 7 *being satisfied, then, with* $\varepsilon_{opt} \geq 0$, *Algorithm* 4 *terminates in a finite number of iterations. If* $\varepsilon_{opt} = 0$, *then Algorithm* 4 *terminates in a finite number of iterations with a KKT point for* (QP).

Proof. Under the conditions of the theorem, it follows that after a finite number of partition updates the algorithm behaves as if (exact) subspace solutions were computed via (2). Hence, the result follows as that for Theorem 3. ∎

Despite the fact that Algorithm 4 employs a straightforward heuristic for controlling inexactness and has the advantage that one need never execute procedure (13), it

Algorithm 4. PDAS framework with inexact subspace solutions (dynamic).

```
1: Input an initial partition $(\mathcal{A},\mathcal{I})$, optimality tolerance $\varepsilon_{opt} > 0$, dynamic parameter
   $\zeta \geq 0$, update factor $\delta_\zeta > 1$, optimality tolerance history length $\bar{j} \in \mathbb{N}$, and
   sufficient reduction factor $\kappa \in (0,1)$.
2: Initialize a partition update counter $j \leftarrow 0$ and $\mathrm{KKT}_j \leftarrow \infty$ for $j \in \{-1,\ldots,-\bar{j}\}$.
3: loop
4:    Compute an inexact subspace solution $(\tilde{x},\tilde{y},\tilde{z})$ with residual $(\tilde{r},\tilde{t})$ by (4).
5:    if $\|\mathrm{KKT}(\tilde{x},\tilde{y},\tilde{z})\| \leq \varepsilon_{opt}$ then
6:       Terminate and return $(\tilde{x},\tilde{y},\tilde{z})$.
7:    if $\|(\tilde{r},\tilde{t})\| \leq \zeta$ then
8:       Set $\tilde{\mathcal{V}}'_P$ and $\tilde{\mathcal{V}}'_D$ by (15) (even if $\tilde{\mathcal{V}}'_P \not\subseteq \mathcal{V}_P$ or $\tilde{\mathcal{V}}'_D \not\subseteq \mathcal{V}_D$).
9:       Choose $\mathcal{C}_P \subseteq \tilde{\mathcal{V}}'_P$ and $\mathcal{C}_D \subseteq \tilde{\mathcal{V}}'_D$ such that $\mathcal{C}_P \cup \mathcal{C}_D \neq \emptyset$.
10:      Set $\mathcal{A} \leftarrow (\mathcal{A}\backslash\mathcal{C}_D) \cup \mathcal{C}_P$ and $\mathcal{I} \leftarrow (\mathcal{I}\backslash\mathcal{C}_P) \cup \mathcal{C}_D$.
11:      Set $\mathrm{KKT}_j \leftarrow \|\mathrm{KKT}(\tilde{x},\tilde{y},\tilde{z})\|$.
12:      if $\mathrm{KKT}_j \geq \kappa \max\{\mathrm{KKT}_{j-1},\ldots,\mathrm{KKT}_{j-\bar{j}}\}$ then
13:         Set $\zeta \leftarrow \zeta/\delta_\zeta$.
14:      Set $j \leftarrow j+1$.
```

Note. If an iteration of the **loop** ends with $\|(\tilde{r},\tilde{t})\| > \zeta$, then it is assumed that the inexact subspace solution subsequently computed in step 4 is improved such that the norm of the corresponding residual vector is sufficiently less than that corresponding to the previous inexact subspace solution. Further, to guarantee convergence, the stronger assumptions in Theorem 10 (that require $\zeta = 0$) must hold.

has two key disadvantages vis-à-vis Algorithms 2 and 3. First, as a practical matter, our experience suggests that it is much more difficult to choose values for ζ and δ_ζ that lead to good performance on a wide range of problems. Second, our convergence guarantee for Algorithm 4 is significantly weaker than those for Algorithms 2 and 3. The following example illustrates why it is not possible to obtain the same convergence guarantees in Theorem 10 as we have stated in Theorems 5 and 8. In particular, the example shows that in order to ensure that $\tilde{\mathcal{V}}'_P \subseteq \mathcal{V}_P$, $\tilde{\mathcal{V}}'_D \subseteq \mathcal{V}_D$, and $\tilde{\mathcal{V}}'_P \cup \tilde{\mathcal{V}}'_D \neq \emptyset$, one may need a linear system residual that is exactly zero.

Example 11. Let

$$m = 0, \quad H = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}, \quad c = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}, \quad \text{and} \quad u = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

With $(\mathcal{A},\mathcal{I}) = (\{1,2\},\{3\})$, it follows from (2) that the subspace solution is

$$x = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{and} \quad z = \begin{bmatrix} -3 \\ 0 \\ 0 \end{bmatrix},$$

from which it follows that $\mathcal{V}_P = \emptyset$ and $\mathcal{V}_D = \{1\}$. In particular, $(\mathcal{A},\mathcal{I})$ is suboptimal. Moreover, from (4), any inexact subspace solution $(\tilde{x},\tilde{z})$ satisfies

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \tilde{x}_3 - x_3 \end{bmatrix} + \begin{bmatrix} \tilde{z}_1 - z_1 \\ \tilde{z}_2 - z_2 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \tilde{r}_3 \end{bmatrix},$$

which implies that $(\tilde{z}_1 - z_1) = 0$ and $\tilde{r}_3 = 2(\tilde{x}_3 - x_3) = 2(\tilde{z}_2 - z_2)$. Hence, in order to have $\tilde{\mathcal{V}}'_P \subseteq \mathcal{V}_P$, $\tilde{\mathcal{V}}'_D \subseteq \mathcal{V}_D$, and $\tilde{\mathcal{V}}'_P \cup \tilde{\mathcal{V}}'_D \neq \emptyset$, one must have

$$\begin{aligned} \tilde{x}_3 &= x_3 + \tfrac{1}{2}\tilde{r}_3 = 1 + \tfrac{1}{2}\tilde{r}_3 \leq 1 \\ \text{and} \quad \tilde{z}_2 &= z_2 + \tfrac{1}{2}\tilde{r}_2 = 0 + \tfrac{1}{2}\tilde{r}_3 \geq 0, \end{aligned}$$

i.e., one must have $\tilde{r}_3 = 0$. (On the other hand, one can verify that with $\gamma = 1 \geq \|H_{33}^{-1}\|_1$, Algorithms 2 and 3 obtain $\tilde{\mathcal{V}}_P = \emptyset = \mathcal{V}_P$ and $\tilde{\mathcal{V}}_D = \{1\} = \mathcal{V}_D$ for $|\tilde{r}_3| < 3$.)

4. An implementation.

We have written implementations of Algorithms 1, 2, 3, and 4 along with the subroutines described as procedures (12) and (13). Also, for comparison purposes, we have written an implementation of the semismooth Newton method with inexact subproblem solves (call it Algorithm `SSN`) from [30]. We discuss common and distinguishing details of the implementations in this section.

All algorithms are implemented in a single piece of software in Python 2.7.3. The software uses the infrastructure in `cvxopt` for matrix storage and manipulation, as well as the implementation of MINRES [45] provided in `Scipy` for (approximately) solving the linear systems (2b) and (4b). For Algorithms 2 and 3, the matrix $A_{\mathcal{E}\mathcal{F}}$ is factored at the start of each run using `cvxopt`'s interface to `LAPACK`, the factors of which are employed to compute products with $A_{\mathcal{E}\mathcal{F}}^{-1}$ and $A_{\mathcal{E}\mathcal{F}}^{-T}$ as they are needed.

In all algorithms, the initial point provided to MINRES in the first PDAS iteration is a randomly generated vector, whereas in subsequent PDAS iterations, the initial point is set by extracting the appropriate elements of the primal-dual solution corresponding to the previous PDAS iterate. For Algorithm 1 in which "exact" solutions are required, each run of MINRES terminates once the 2-norm of the residual for the linear system (2b) is reduced below a prescribed tolerance $\bar{\epsilon}_{num} > 0$. Similarly, for Algorithm 4, MINRES terminates once either (14) holds or the 2-norm of the residual for the linear system (4b) is reduced below $\epsilon_{num} > \bar{\epsilon}_{num}$. Finally, for Algorithms 2 and 3, MINRES terminates either once (14) and (16) both hold or the 2-norm of the residual for the linear system (4b) is reduced below ϵ_{num}. (The only exceptions occur when $\|(\tilde{r}, \tilde{t})\|_2 \leq \epsilon_{num}$, but $|\tilde{\mathcal{V}}_P \cup \tilde{\mathcal{V}}_D| = |\tilde{\mathcal{V}}'_P \cup \tilde{\mathcal{V}}'_D| = 0$ and the algorithm's termination criterion is not satisfied, in which case MINRES is forced to continue.) The termination criterion in our tests were problem-specific; see section 5.

For Algorithms 2 and 3, the evaluation of vectors in (13) requires products with $A_{\mathcal{E}\mathcal{F}}^{-1}$ and $A_{\mathcal{E}\mathcal{F}}^{-T}$ (i.e., solves with the factors of $A_{\mathcal{E}\mathcal{F}}$), meaning that it is not economical to compute these quantities after every MINRES iteration. Hence, our software performs 10 MINRES iterations in step 3 (resp., 4) of Algorithm 2 (resp., 3).

For Algorithm 3, we implemented an additional strategy for updating γ that utilizes intermediate vectors computed by MINRES. The purpose of this strategy is to use problem information to quickly adjust γ if the initial value is set too low for a given run of the algorithm. In particular, with an intermediate solution $\tilde{x}_{\mathcal{I}}$ and product $R_{\mathcal{I}\mathcal{I}}\tilde{x}_{\mathcal{I}}$, both provided at no extra cost by MINRES, we set

$$\gamma \leftarrow \max\left\{\gamma, \frac{\|\tilde{x}_{\mathcal{I}}\|_1}{\|R_{\mathcal{I}\mathcal{I}}\tilde{x}_{\mathcal{I}}\|_1}\right\}.$$

This update is motivated by the fact that

$$\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1 = \max_{\|x\|_1 = 1} \|R_{\mathcal{I}\mathcal{I}}x\|_1^{-1}.$$

We remark that our use of MINRES as the iterative solver for (2b) and (4b) is not required; any iterative method for solving symmetric indefinite systems could be employed. We also note that while preconditioning would be a critical aspect of any efficient implementation of any of our algorithms, we did not implement a preconditioner in our software. This is reasonable as the purpose of our numerical experiments is merely to illustrate the convergence behavior of our algorithms despite inexactness in the subspace solutions; thus, the absolute numbers of MINRES iterations required to obtain the (inexact) subspace solutions in our software is not a focus of our experiments. More important are the relative numbers of iterations required by our methods in comparison to those required by Algorithms 1 and `SSN`.

5. Numerical results.

In this section, we report on the performance of our implementations of Algorithms 1, 2, 3, 4, and `SSN` when they were employed to solve instances of two optimal control problems. (Hereafter, Algorithm 1 refers to the variant of that algorithm in which the choice $\mathcal{C}_P \leftarrow \mathcal{V}_P$ and $\mathcal{C}_D \leftarrow \mathcal{V}_D$ is made during every iteration.) The problems are the same as those in [28]. We first report on the performance of the algorithms for straightforward instances of the problems, then report on their performance on instances that were intentionally degenerate.

For consistency with notation common in optimal control, let the state variable be y and the control variable be u, and let $x = (x_1, x_2)$ denote the coordinate axes in $\mathbb{R}^2$. (The reader should be aware that the pair (x, y) here should not be confused with the primal-dual variable pair (x, y) in previous sections.) Then, given a domain $\Omega \in \mathbb{R}^2$, reference functions $\overline{y} \in L^2(\Omega)$ and $\overline{u} \in \{L^2(\Omega), L^2(\partial\Omega)\}$, upper bound function $\psi \in L^2(\Omega)$, and regularization parameter $\beta > 0$, we consider the test problems

$$\begin{aligned} &\min_{y,u} \ \tfrac{1}{2}\|y - \overline{y}\|^2_{L^2(\Omega)} + \tfrac{\beta}{2}\|u - \overline{u}\|^2_{L^2(\Omega)} \\ &\text{s.t.} \begin{cases} -\Delta y = u & \text{in } \ \Omega, \\ y = 0 & \text{on } \ \partial\Omega, \\ u \le \psi & \text{in } \ \Omega, \end{cases} \end{aligned} \tag{18}$$

and, with n denoting the unit outer normal to Ω along $\partial\Omega$,

$$\begin{aligned} &\min_{y,u} \ \tfrac{1}{2}\|y - \overline{y}\|^2_{L^2(\Omega)} + \tfrac{\beta}{2}\|u - \overline{u}\|^2_{L^2(\partial\Omega)} \\ &\text{s.t.} \begin{cases} -\Delta y + y = 0 & \text{in } \ \Omega, \\ \frac{\partial y}{\partial n} = u & \text{on } \ \partial\Omega, \\ u \le \psi & \text{on } \ \partial\Omega. \end{cases} \end{aligned} \tag{19}$$

In particular, we consider instances with $\Omega = (0, 1)^2$, $\beta = 10^{-5}$, and $\psi = 0$. As for the reference functions $\overline{y}$ and $\overline{u}$, we consider values as stated in the subsections below.

In order to illustrate the performance of the implementations on instances of various sizes, we generated discretized versions of problems (18) and (19) at various levels of discretization. In particular, we generated instances of both problems with the numbers of grid points along each dimension in the set $\{4, 8, 16, 32, 64\}$. A five-point-star discretization of Δ was used and the functions y, u, $\overline{y}$, $\overline{u}$, and ψ were discretized by means of grid functions at the nodal points. It is easily verified that all of the resulting problem instances have the form (QP) satisfying Assumption 1. Table 1 contains the sizes of each problem instance in terms of the numbers of grid points per dimension (`g`), variables (`n`), and equality constraints (`m`).

To avoid mesh-dependent performance, the termination criterion for each problem was chosen based on condensed first-order optimality conditions [27, 31]. Letting p represent adjoint variables and letting $\{p_h, u_h, \psi_h\}$ represent the adjoint, control, and

Table 1
Problem sizes.

	Problem (18)		Problem (19)	
g	n	m	n	m
4	32	16	48	32
8	128	64	128	96
16	512	256	384	320
32	2048	1024	1280	1152
64	8192	4096	4608	4352

Table 2
Input parameters for our implementations of Algorithms 1, 2, 3, 4, *and* SSN.

Parameter	Value	Algorithm(s)
ϵ_{opt}	$5 \times 10^{-5}/h$ for (18) $5 \times 10^{-5}/\sqrt{h}$ for (19)	1, 2, 3, 4, SSN
$\bar{\epsilon}_{num}$	$1 \times 10^{-5}/(2h)$ for (18) $1 \times 10^{-5}/(2\sqrt{h})$ for (19)	1
ϵ_{num}	$5 \times 10^{-5}/(2h)$ for (18) $5 \times 10^{-5}/(2\sqrt{h})$ for (19)	2, 3, 4
ϵ_{res}	10^{-2} 10^{-1}	2, 3 4, SSN
θ	0.8	2, 3
γ, δ_γ	10^2, 1.2	3
ζ, δ_ζ	ϵ_{res}, 1.2	4
$\bar{j}$	5	3, 4
κ	0.9	3, 4

bound vectors corresponding to the mesh size parameters $h \in \{2^{-4}, \ldots, 2^{-64}\}$, for problem (18) these conditions become

$$h\|p_h - \beta u_h - \max\{p_h - \beta\psi_h, 0\}\|_2 \le \epsilon_{opt},$$

while for problem (19) they become

$$\sqrt{h}\|(p_h|_\Gamma) - (\beta u_h|_\Gamma) - \max\{(p_h|_\Gamma) - \beta\psi_h, 0\}\|_2 \le \epsilon_{opt}.$$

For the input parameters, we used the values in Table 2. We experimented with various values for these parameters and the ones employed here are merely ones that worked well in our experiments. Of course, to obtain the best performance, it would be ideal to tune these parameters for individual applications.

Finally, it is important to note that for each instance of each problem, all algorithms were initialized with the same initial partition.

5.1. Results for basic formulations. We first present results for instances where, as in [28], the reference functions were set as

$$\bar{y}(x_1, x_2) = \sin(5x_1) + \cos(4x_2) \quad \text{and} \quad \bar{u} = 0. \tag{20}$$

All algorithms were initialized with all control variables being active at their bounds. The performance of the algorithms in solving problem (18) is reported in Table 3 and their performance in solving problem (19) is reported in Table 4. In each table,

Table 3
Results when solving problem (18) *with* $(\overline{y}, \overline{u})$ *from* (20).

	Algorithm 1		Algorithm 2		Algorithm 3		Algorithm 4		Algorithm SSN	
g	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.
4	2	87	2	68	2	68	2	61	2	68
8	3	454	3	367	3	367	4	449	3	367
16	3	1686	3	1370	3	1370	4	1577	3	1370
32	3	6181	3	4694	3	4702	4	5963	3	4832
64	4	29911	3	16269	3	16261	4	20227	3	16308

Table 4
Results when solving problem (19) *with* $(\overline{y}, \overline{u})$ *from* (20).

	Algorithm 1		Algorithm 2		Algorithm 3		Algorithm 4		Algorithm SSN	
g	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.
4	5	525	5	481	5	481	5	423	5	476
8	7	2729	7	2493	7	2493	7	2303	7	2523
16	8	10226	8	9564	8	9564	8	9032	8	9774
32	9	36659	8	30867	8	31098	8	30371	8	31746
64	9	121853	8	103507	8	104660	8	103604	8	106147

we report the numbers of grid points per dimension, PDAS iterations required before termination (`Iter.`), and total Krylov (i.e., MINRES) iterations required before termination (`Kry.`). (It should be noted that, due to diagonal dominance of the matrices involved, all runs of Algorithm 2 required only one Krylov iteration per PDAS iteration to compute the upper bound on $\|R_{\mathcal{I}\mathcal{I}}^{-1}\|_1$ via procedure (12). Hence, the computational expense of this subroutine for all instances was negligible.) While we do not explicitly report the CPU time required by each solver to terminate when solving each problem instance, we remark that, as should be expected in general, the relative CPU times required by all solvers was proportional to the relative Krylov iterations required. Hence, in these results, one should compare solvers primarily based on the numbers of Krylov iterations required.

The results in Tables 3 and 4 show that all algorithms were competitive in terms of iterations required in this experiment. This is a credit to PDAS and semismooth Newton frameworks, which, as exhibited in this experiment, can converge in very few iterations regardless of problem size. That being said, the results do provide evidence for the benefits of allowing inexactness in the subspace solutions. In particular, when applied to solve these test instances, Algorithms 2, 3, and 4 consistently require many fewer Krylov iterations. Algorithm `SSN` is also competitive.

5.2. Results for degenerate formulations. We now present results for degenerate instances of (18) and (19). In particular, following a similar strategy as in [4], we construct degenerate instances by ensuring $(u_*)_h = \psi_h$ with half of the optimal dual variables associated with these bounds being zero and the other half being randomly generated positive numbers. The state variable $(y_*)_h$ is then set so as to satisfy the affine equality constraints and the reference function values are chosen so that the optimality conditions are satisfied at this primal-dual optimal solution. For both problems, all algorithms were initialized to have half of the bounds active.

Using the same headings as the tables in the previous subsection, the performance of the algorithms in solving problem (18) is reported in Table 5 and their performance in solving problem (19) is reported in Table 6.

As in the previous section, the results in Tables 5 and 6 show that while all algorithms are competitive in terms of numbers of iterations, the algorithms that allow

TABLE 5
Results when solving degenerate instances of problem (18).

	Algorithm 1		Algorithm 2		Algorithm 3		Algorithm 4		Algorithm SSN	
g	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.
4	3	137	3	77	3	77	2	26	2	40
8	4	641	3	240	3	240	2	177	3	289
16	4	2154	3	802	3	802	3	649	3	951
32	6	9586	3	3273	3	3174	2	2130	3	4152
64	5	16797	3	10101	3	8414	3	7512	3	11958

TABLE 6
Results when solving degenerate instances of problem (19).

	Algorithm 1		Algorithm 2		Algorithm 3		Algorithm 4		Algorithm SSN	
g	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.	Iter.	Kry.
4	2	334	2	309	3	420	4	447	3	422
8	3	1593	3	1588	2	1214	4	1392	4	1905
16	5	7503	3	4884	3	5184	3	3524	3	5258
32	5	21062	3	12823	3	14328	5	16046	3	14969
64	5	62470	3	35152	3	39934	4	37778	3	43288

inexact subspace solutions yield benefits in terms of reduced overall Krylov iterations required, a good proxy for CPU time in general. All other algorithms required many fewer Krylov iterations than Algorithm 1, with slightly improved performance seen for Algorithms 2, 3, and 4 when compared to Algorithm SSN.

6. Conclusion.

In this paper, we have proposed a set of primal-dual active-set algorithms for solving certain structured quadratic optimization problems. The distinguishing feature of the algorithms is that they attain global convergence guarantees while allowing inexactness in the (reduced) linear system solves in each iteration. In each iteration, the first algorithm sets a requirement for the accuracy in the linear system solve through the use of subroutines for computing an upper bound for the inverse of a particular submatrix, whereas the second and third make use of dynamic algorithmic parameters for controlling the level of inexactness in each iteration. We have implemented our algorithms and have provided the results of numerical experiments on a pair of optimal control test problems, illustrating that our algorithms can converge in a similar number of PDAS iterations as an algorithm that employs "exact" linear system solves, but with lower per-iteration computational costs. We also compared our implementations against that of a semismooth Newton method that allows inexact linear system solves [30]. This approach was also competitive in our experiments, though we believe that the manner in which our methods impose inexactness criteria that guarantee productivity in each active-set update can be advantageous.

Appendix A. Convergence of Algorithm 1.

In this appendix, we prove Theorem 3 by proving two theorems, the first corresponding to condition (a) of the theorem and the second corresponding to condition (b). These theorems can be seen as generalizations of [28, Theorem 3.3] and [28, Theorem 3.4], respectively, so that the conclusions can be shown to apply when solving problem (QP) with Algorithm 1. (By contrast, the algorithm and results in [28] were stated in terms of solving BQPs only, and the algorithm in [28] is stated such that all elements of the violated sets change index set membership in each iteration, which is not required in Algorithm 1.)

THEOREM 12. *Suppose Assumption* 1 *holds, R is a P-matrix, and, corresponding to any partition $(\mathcal{A}, \mathcal{I})$, we have that $\|[R_{\mathcal{I}\mathcal{I}}^{-1} R_{\mathcal{I}\mathcal{A}}]_+\|_1 < 1$ and $e^T R_{\mathcal{I}\mathcal{I}}^{-1} w \geq 0$ for any*

$w \geq 0$, where the latter inequality holds strictly, i.e., $e^T R_{\mathcal{II}}^{-1} w > 0$, whenever $w \neq 0$. Then, with $\varepsilon_{opt} \geq 0$ and any initial partition, Algorithm 1 *terminates in a finite number of iterations. In particular, if $\varepsilon_{opt} = 0$, then Algorithm* 1 *terminates in a finite number of iterations with a KKT point for* (QP).

Proof. It suffices to prove the result for $\varepsilon_{opt} = 0$. Thus, in the proof, we show that Algorithm 1 yields a KKT point in a finite number of iterations.

In order to derive a contradiction, suppose that Algorithm 1 generates an infinite number of partitions. For a given partition $(\mathcal{A}, \mathcal{I})$ considered in the algorithm, let $(\mathcal{A}^+, \mathcal{I}^+)$ be the subsequent partition in the algorithm. Furthermore, let (x, y, z) and (x^+, y^+, z^+), respectively, be the subspace solutions corresponding to these partitions. For any index $i \in \mathcal{A}^+$, we have by step 8 of Algorithm 1 that either

$$\begin{aligned} i \in \mathcal{A} &\implies x_i = u_i = x_i^+ \\ \text{or } \; i \in \mathcal{I} &\implies x_i > u_i = x_i^+. \end{aligned}$$

Hence, it follows that $[x^+ - x]_{\mathcal{A}^+} \leq 0$. Similarly, for any index $i \in \mathcal{I}^+$, we have by step 8 of Algorithm 1 that either

$$\begin{aligned} i \in \mathcal{I} &\implies z_i = 0 = z_i^+ \\ \text{or } \; i \in \mathcal{A} &\implies z_i < 0 = z_i^+. \end{aligned}$$

Hence, it follows that $[z^+ - z]_{\mathcal{I}^+} \geq 0$. Now, since (x, y, z) and (x^+, y^+, z^+) are subspace solutions, it follows from (2) that

$$R[x^+ - x]_{\mathcal{B}} + [z^+ - z] = 0, \tag{21}$$

which implies that

$$[x^+ - x]_{\mathcal{I}^+} = -R_{\mathcal{I}^+\mathcal{I}^+}^{-1}(R_{\mathcal{I}^+\mathcal{A}^+}[x^+ - x]_{\mathcal{A}^+} + [z^+ - z]_{\mathcal{I}^+}).$$

Moreover, from nonsingularity of the matrix R, it follows that if $[x^+ - x]_{\mathcal{A}^+}$ and $[z^+ - z]_{\mathcal{I}^+}$ are both zero, then $[x^+ - x]_{\mathcal{I}^+}$ and $[z^+ - z]_{\mathcal{A}^+}$ are both zero. By the partition update rule in step 8 of Algorithm 1, this occurs if and only if the violated sets $\mathcal{V}_P$ and $\mathcal{V}_D$ corresponding to (x, y, z) satisfy $\mathcal{V}_P \cup \mathcal{V}_D$, which, by Theorem 2, occurs if and only if (x, y, z) is a KKT point for (QP), i.e., $\text{KKT}(x, y, z) = 0$. However, in such a case, the algorithm would have terminated with (x, y, z), contradicting the supposition that an infinite number of partitions are generated. Hence, it follows that $[x^+ - x]_{\mathcal{A}^+}$ and $[z^+ - z]_{\mathcal{I}^+}$ cannot both be zero.

For brevity, we now define $\Delta x := x^+ - x$ and $\Delta z := z^+ - z$. From the discussion above and letting $e \in \mathbb{R}^n$ denote a vector of ones, we have

$$\begin{aligned} & e^T \Delta x_{\mathcal{B}} \\ &= e_{\mathcal{A}^+}^T [\Delta x]_{\mathcal{A}^+} + e_{\mathcal{I}^+}^T [\Delta x]_{\mathcal{I}^+} \\ &= e_{\mathcal{A}^+}^T [\Delta x]_{\mathcal{A}^+} - e_{\mathcal{I}^+}^T R_{\mathcal{I}^+\mathcal{I}^+}^{-1} R_{\mathcal{I}^+\mathcal{A}^+} [\Delta x]_{\mathcal{A}^+} - e_{\mathcal{I}^+}^T R_{\mathcal{I}^+\mathcal{I}^+}^{-1} [\Delta z]_{\mathcal{I}^+} \\ &\leq \; -(1 - \|[R_{\mathcal{I}^+\mathcal{I}^+}^{-1} R_{\mathcal{I}^+\mathcal{A}^+}]_+\|_1)\|[\Delta x]_{\mathcal{A}^+}\|_1 - e_{\mathcal{I}^+}^T R_{\mathcal{I}^+\mathcal{I}^+}^{-1} [\Delta z]_{\mathcal{I}^+} < 0, \end{aligned}$$

where, by the conditions of the theorem, the last inequality is strict since $[\Delta x]_{\mathcal{A}^+}$ and $[\Delta z]_{\mathcal{I}^+}$ cannot both be zero. Thus, the quantity $e^T x_{\mathcal{B}}$ strictly decreases in each iteration of Algorithm 1. However, this is a contradiction to the supposition that an infinite number of iterates are generated since x is uniquely determined by the partition and there are only a finite number of partitions of $\mathcal{B}$. Consequently, we have proved that Algorithm 1 must terminate finitely with a KKT point. ∎

THEOREM 13. *Suppose Assumption* 1 *holds and* R *satisfies* $R = M + E$*, where* M *is an* M*-matrix and* $\|E\|_1$ *is sufficiently small. Then, with* $\varepsilon_{opt} \geq 0$ *and any initial partition, Algorithm* 1 *terminates in a finite number of iterations. In particular, if* $\varepsilon_{opt} = 0$*, then Algorithm* 1 *terminates in a finite number of iterations with a KKT point for* (QP)*.*

Proof. As in the proof of Theorem 12, it suffices to prove the result for $\varepsilon_{opt} = 0$. Furthermore, again as in the proof of Theorem 12, we suppose—in order to derive a contradiction—that Algorithm 1 generates an infinite number of partitions. Borrowing notation and conclusions from the proof of Theorem 12, we have $[\Delta x]_{\mathcal{A}^+} \leq 0$ and $[\Delta z]_{\mathcal{I}^+} \geq 0$. Moreover, for sufficiently small $\|E\|_1$, the matrix R is nonsingular,

$$R_{\mathcal{I}^+\mathcal{I}^+}^{-1} R_{\mathcal{I}^+\mathcal{A}^+} = M_{\mathcal{I}^+\mathcal{I}^+}^{-1} M_{\mathcal{I}^+\mathcal{A}^+} + \mathcal{O}(\|E\|_1) \quad \text{and} \quad R_{\mathcal{I}^+\mathcal{I}^+}^{-1} = M_{\mathcal{I}^+\mathcal{I}^+}^{-1} + \mathcal{O}(\|E\|_1).$$

Since M is an M-matrix, we have $M_{\mathcal{I}^+\mathcal{I}^+}^{-1} M_{\mathcal{I}^+\mathcal{A}^+} \leq 0$ and $M_{\mathcal{I}^+\mathcal{I}^+}^{-1} \geq 0$. Hence, since $[\Delta x]_{\mathcal{A}^+}$ and $[\Delta z]_{\mathcal{I}^+}$ cannot both be zero, we have for sufficiently small $\|E\|_1$ that

$$\begin{aligned}
&e^T \Delta x_{\mathcal{B}} \\
&= e_{\mathcal{A}^+}^T [\Delta x]_{\mathcal{A}^+} + e_{\mathcal{I}^+}^T [\Delta x]_{\mathcal{I}^+} \\
&= e_{\mathcal{A}^+}^T [\Delta x]_{\mathcal{A}^+} - e_{\mathcal{I}^+}^T R_{\mathcal{I}^+\mathcal{I}^+}^{-1} R_{\mathcal{I}^+\mathcal{A}^+} [\Delta x]_{\mathcal{A}^+} - e_{\mathcal{I}^+}^T R_{\mathcal{I}^+\mathcal{I}^+}^{-1} [\Delta z]_{\mathcal{I}^+} \\
&= e_{\mathcal{A}^+}^T [\Delta x]_{\mathcal{A}^+} - e_{\mathcal{I}^+}^T M_{\mathcal{I}^+\mathcal{I}^+}^{-1} M_{\mathcal{I}^+\mathcal{A}^+} [\Delta x]_{\mathcal{A}^+} - e_{\mathcal{I}^+}^T M_{\mathcal{I}^+\mathcal{I}^+}^{-1} [\Delta z]_{\mathcal{I}^+} + \mathcal{O}(\|E\|_1) < 0.
\end{aligned}$$

The remainder of the proof follows in the same manner as that of Theorem 12. □

Acknowledgments. The authors would like to thank the two anonymous referees, whose comments and suggestions helped to make significant improvements. They would also like to thank the associate editor sincerely for their efforts.

REFERENCES

[1] M. AGANAGIĆ, *Newton's method for linear complementarity problems*, Math. Program., 28 (1984), pp. 349–362.

[2] J. H. AHLBERG AND E. N. NILSON, *Convergence properties of the spline fit*, J. SIAM, 11 (1963), pp. 95–104.

[3] R. A. BARTLETT AND L. T. BIEGLER, *QPSchur: A dual, active-set, Schur-complement method for large-scale and structured convex quadratic programming*, Optim. Eng., 7 (2006), pp. 5–32.

[4] M. BERGOUNIOUX, M. HADDOU, M. HINTERMÜLLER, AND K. KUNISCH, *A comparison of methods and a Moreau-Yosida based active set strategy for constrained optimal control problems*, SIAM J. Optim., 11 (2000), pp. 495–521.

[5] M. BERGOUNIOUX, K. ITO, AND K. KUNISCH, *Primal-dual strategy for constrained optimal control problems*, SIAM J. Control Optim., 37 (1999), pp. 1176–1194.

[6] D. P. BERTSEKAS, *Projected Newton methods for optimization problems with simple constraints*, SIAM J. Control Optim., 20 (1982), pp. 221–246.

[7] M. J. BEST, *An algorithm for the solution of the parametric quadratic programming problem*, in Applied Mathematics and Parallel Computing, Physica-Verlag HD, Heidelberg, Germany, 1996, pp. 57–76.

[8] E. G. BIRGIN AND J. M. MARTÍNEZ, *Large-scale active-set box-constrained optimization method with spectral projected gradients*, Comput. Optim. Appl., 23 (2002), pp. 101–125.

[9] I. M. BOMZE, *On standard quadratic optimization problems*, J. Global Optim., 13 (1998), pp. 369–387.

[10] B. E. BOSER, I. M. GUYON, AND V. N. VAPNIK, *A training algorithm for optimal margin classifiers*, in Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92, New York, 1992, ACM Press, pp. 144–152.

[11] P. N. BROWN AND Y. SAAD, *Hybrid Krylov methods for nonlinear systems of equations*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 450–481.

[12] S. Cafieri, M. D'Apuzzo, M. Marino, A. Mucherino, and G. Toraldo, *Interior-point solver for large-scale quadratic programming problems with bound constraints*, J. Optim. Theory Appl., 129 (2006), pp. 55–75.

[13] F. E. Curtis, Z. Han, and D. P. Robinson, *A Globally Convergent Primal-Dual Active-Set Framework for Large-Scale Convex Quadratic Optimization*, Tech. report 12T-013, COR@L Laboratory, Department of ISE, Lehigh University, 2012.

[14] Y.-H. Dai and R. Fletcher, *Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming*, Numer. Math., 100 (2005), pp. 21–47.

[15] J. C. De los Reyes, *A primal-dual active set method for bilaterally control constrained optimal control of the Navier-Stokes equations*, Numer. Funct. Anal. Optim., 25 (2005), pp. 657–683.

[16] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[17] H. J. Ferreau, H. G. Bock, and M. Diehl, *An online active set strategy to overcome the limitations of explicit MPC*, Internat. J. Robust Nonlinear Control, 18 (2008), pp. 816–830.

[18] R. Fletcher, *Practical Methods of Optimization*, 2nd ed., Wiley-Interscience, New York, 2000.

[19] A. Friedlander and J. M. Martínez, *On the maximization of a concave quadratic function with box constraints*, SIAM J. Optim., 4 (1994), pp. 177–192.

[20] M. Fukushima and P. Tseng, *An implementable active-set algorithm for computing a B-stationary point of a mathematical program with linear complementarity constraints*, SIAM J. Optim., 12 (2002), pp. 724–739.

[21] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, Emerald Group, Bingley, UK, 1982.

[22] D. Goldfarb and A. Idnani, *A numerically stable dual method for solving strictly convex quadratic programs*, Math. Program., 27 (1983), pp. 1–33.

[23] N. I. M. Gould and P. L. Toint, *A Quadratic Programming Bibliography*, RAL Numerical Analysis Group Internal Report 2000-1, Rutherford Appleton Laboratory, Chilton, UK, 2000.

[24] R. Griesse and S. Volkwein, *A primal-dual active set strategy for optimal boundary control of a nonlinear reaction-diffusion system*, SIAM J. Control Optim., 44 (2005), pp. 467–494.

[25] W. W. Hager and H. Zhang, *A new active set algorithm for box constrained optimization*, SIAM J. Optim., 17 (2006), pp. 526–557.

[26] M. Heinkenschloss, M. Ulbrich, and S. Ulbrich, *Superlinear and quadratic convergence of affine-scaling interior-point Newton methods for problems with simple bounds without strict complementarity assumption*, Math. Program., 86 (1999), pp. 615–635.

[27] M. Hintermüller, *Mesh-independence and fast local convergence of a primal-dual active-set method for mixed control-state constrained elliptic control problems*, ANZIAM J., 49 (2007), pp. 1–38.

[28] M. Hintermüller, K. Ito, and K. Kunisch, *The primal-dual active set strategy as a semismooth Newton method*, SIAM J. Optim., 13 (2003), pp. 865–888.

[29] M. Hintermüller, V. A. Kovtunenko, and K. Kunisch, *Obstacle problems with cohesion: A hemivariational inequality approach and its efficient numerical solution*, SIAM J. Optim., 21 (2011), pp. 491–516.

[30] M. Hintermüller and G. Stadler, *An infeasible primal-dual algorithm for total bounded inf-convolution-type image restoration*, SIAM J. Sci. Comput., 28 (2006), pp. 1–23.

[31] M. Hintermüller and M. Ulbrich, *A mesh independence result for semismooth Newton methods*, Math. Program., 101 (2004), pp. 151–184.

[32] S. Hüeber, A. Matei, and B. I. Wohlmuth, *Efficient algorithms for problems with friction*, SIAM J. Sci. Comput., 29 (2007), pp. 70–92.

[33] S. Hüeber, G. Stadler, and B. I. Wohlmuth, *A primal-dual active set algorithm for three-dimensional contact problems with Coulomb friction*, SIAM J. Sci. Comput., 30 (2008), pp. 572–596.

[34] K. Ito and K. Kunisch, *Optimal control of elliptic variational inequalities*, Appl. Math. Optim., 41 (2000), pp. 343–364.

[35] K. Ito and K. Kunisch, *The primal-dual active set method for nonlinear optimal control problems with bilateral constraints*, SIAM J. Control Optim., 43 (2004), pp. 357–376.

[36] K. Ito and K. Kunisch, *Convergence of the primal-dual active set strategy for diagonally dominant systems*, SIAM J. Control Optim., 46 (2007), pp. 14–34.

[37] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, *An interior-point method for large-scale ℓ_1-regularized least squares*, IEEE J. Selected Topics Signal Process., 1 (2007), pp. 606–617.

[38] D. Krishnan, P. Lin, and A. M. Yip, *A primal-dual active-set method for non-negativity constrained total variation deblurring problems*, IEEE Trans. Image Process., 16 (2007), pp. 2766–2777.

[39] K. Kunisch and F. Rendl, *An infeasible active set method for quadratic problems with simple bounds*, SIAM J. Optim., 14 (2003), pp. 35–52.

[40] K. Kunisch and A. Rösch, *Primal-dual active set strategy for a general class of constrained optimal control problems*, SIAM J. Optim., 13 (2002), pp. 321–334.

[41] P. Lötstedt, *Solving the minimal least squares problem subject to bounds on the variables*, BIT, 24 (1984), pp. 205–224.

[42] J. J. Moré and G. Toraldo, *Algorithms for bound constrained quadratic programming problems*, Numer. Math., 55 (1989), pp. 377–400.

[43] Y. Nesterov and A. Nemirovskii, *Interior Point Polynomial Algorithms in Convex Programming*, Stud. Appl. Numer. Math., SIAM, Philadelphia, 1994.

[44] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer Ser. Oper. Res. Financ. Eng., Springer, New York, 2006.

[45] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629.

[46] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.

[47] G. Stadler, *Semismooth Newton and augmented Lagrangian methods for a simplified friction problem*, SIAM J. Optim., 15 (2004), pp. 39–62.

[48] J. A. K. Suykens and J. Vandewalle, *Least squares support vector machine classifiers*, Neural Process. Lett., 9 (1999), pp. 293–300.

[49] H. A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Cambridge Monogr. Appl. Comput. Math., Cambridge University Press, New York, 2003.

[50] R. J. Vanderbei, *LOQO: An interior point code for quadratic programming*, Optim. Methods Softw., 11 (1999), pp. 451–484.

[51] V. Vapnik and C. Cortes, *Support vector networks*, Machine Learning, 20 (1995), pp. 273–297.

[52] J. Varah, *A lower bound for the smallest singular value of a matrix*, Linear Algebra Appl., 11 (1975), pp. 3–5.

[53] R. S. Varga, *On diagonal dominance arguments for bounding $\|A^{-1}\|_\infty$*, Linear Algebra Appl., 14 (1976), pp. 211–217.

[54] S. J. Wright, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, 1997.