Les Cahiers du GERAD ISSN: 0711–2440

A symmetric formulation of the linear system arising in interior methods for convex optimization with bounded condition number

A. Ghannad,
D. Orban, M. A. Saunders

G–2020–37

June 2020

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2020
– Bibliothèque et Archives Canada, 2020

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2020
– Library and Archives Canada, 2020

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

A symmetric formulation of the linear system arising in interior methods for convex optimization with bounded condition number

Alexandre Ghannad [a]

Dominique Orban [b]

Michael A. Saunders [c]

[a] *École Centrale de Nantes, 44300 Nantes, France*

[b] *GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal (Québec) Canada, H3C 3A7*

[c] *Systems Optimization Laboratory, Department of Management Science and Engineering, Stanford University, Stanford, CA, 94305–4121, USA*

alexandre.ghannad@gmail.com
dominique.orban@gerad.ca
saunders@stanford.edu

June 2020
Les Cahiers du GERAD
G–2020–37

Abstract: We provide eigenvalues bounds for a new formulation of the step equations in interior methods for convex quadratic optimization. The matrix of our formulation, named $K_{2.5}$, has bounded condition number, converges to a well-defined limit under strict complementarity, and has the same size as the traditional, ill-conditioned, saddle-point formulation. We evaluate the performance in the context of a Matlab object-oriented implementation of PDCO, an interior-point solver for minimizing a smooth convex function subject to linear constraints. The main benefit of our implementation, named PDCOO, is to separate the logic of the interior-point method from the formulation of the system used to compute a step at each iteration and the method used to solve the system. Thus, PDCOO allows easy addition of a new system formulation and/or solution method for experimentation. Our numerical experiments indicate that the $K_{2.5}$ formulation has the same storage requirements as the traditional ill-conditioned saddle-point formulation, and its condition is substantially more favorable than the unsymmetric block 3×3 formulation.

Keywords: Convex optimization, primal-dual interior methods, indefinite linear systems, eigenvalues, condition number, inertia, eigenvalue bounds, regularization

Résumé : Nous développons des bornes sur les valeurs propres d'une nouvelle formulation des équations de Newton dans les méthodes de points intérieurs pour l'optimisation convexe. La matrice de notre formulation, nommée $K_{2.5}$, a un nombre de conditionnement borné, converge vers une limite bien définie sous l'hypothèse de complémentarité stricte, et est de la même taille que la formulation de point de selle mal conditionnée traditionnelle. Nous évaluons sa performance dans le contexte d'une nouvelle implémentation Matlab orientée objet de PDCO, un logiciel de points intérieurs pour la minimisation de fonctions convexes lisses sous contraintes linéaires. L'avantage principal de notre implémentation, nommée PDCOO, est de séparer la logique de la méthode de points intérieurs de la formulation du système utilisé pour calculer un pas à chaque itération et de la méthode utilisée pour résoudre ce système. Ainsi, PDCOO permet d'ajouter facilement une nouvelle formulation et/ou une nouvelle méthode de résolution pour effectuer des essais. Nos résultats numériques indiquent que la formulation $K_{2.5}$ requiert la même quantité de mémoire que la formulation mal conditionnée traditionnelle et que son nombre de conditionnement est significativement meilleur que celui de la formulation non symétrique 3×3.

Acknowledgments: The authors acknowledge Hadrien Godard's contribution to Section 4 during his undergraduate internship at GERAD. Research of the second author was partially supported by an NSERC Discovery Grant.

1 Introduction

We consider the problem

$$\begin{aligned}&\underset{x,\,r}{\text{minimize}}\ \phi(x) + \tfrac{1}{2}\|D_1 x\|^2 + \tfrac{1}{2}\|r\|^2\\ &\text{subject to } Ax + D_2 r = b,\ x \ge 0,\end{aligned} \tag{1}$$

where $\phi : \mathbb{R}^n \to \mathbb{R}$ is $\mathcal{C}^2$ and convex, D_1 and D_2 (if present) are diagonal and positive definite, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and inequalities are understood elementwise. We state (1) with nonnegative x, but all our results can be adapted to general bounds $\ell \le x \le u$.

Nonzero D_1 and D_2 help regularize the problem when ϕ is not strictly convex or the equality constraints are (nearly) dependent. Least-squares problems with bounded variables are accommodated by $D_2 = I$. Otherwise, if D_1 and D_2 are small, (1) may be thought of as a regularized form of

$$\underset{x}{\text{minimize}}\ \ \phi(x) \quad \text{subject to}\ \ Ax = b,\ x \ge 0.$$

A primal-dual interior-point method applied to (1) requires the solution of a large structured linear system at each iteration to compute a step (a search direction for the primal and dual variables). Several formulations of the linear system are used and analyzed in the literature. Recent research includes (Greif, Moulding, and Orban, 2014; Morini, Simoncini, and Tani, 2016) and references therein. A prime computational concern is that the system becomes increasingly ill-conditioned as the iterations proceed. The so-called K_3 formulation yields a large unsymmetric system with 3×3 block structure, and although its condition number typically becomes large in practice, it is provably bounded if strict complementarity is satisfied at the solution. Several attempts have been made in the literature to symmetrize K_3 in order to save computation. We enumerate those attempts in Section 2. One formulation has the same size as K_3 but unbounded condition number (we name it K_{3S}), one has 2×2 block structure and thus saves storage and factorization time but again has unbounded condition number (we name it K_2), and one has the same size as K_3 and provably bounded condition number—we name it $K_{3.5}$.[1]

An efficient sparse symmetric indefinite factorization is required to work with K_{3S}, $K_{3.5}$ or K_2. An efficient sparse LU factorization is required for K_3. Before the advent of such libraries, it was customary to perform further block eliminations and reduce the system to one with matrix $K_1 := ADA^T + D_2^2$, where D is diagonal and positive-definite. The advantage of K_1 is that it is positive definite and therefore possesses a Cholesky factorization. Unfortunately, its condition number is unbounded, and it becomes dense if A contains even one dense column—a common occurrence. Specialized variants of the Cholesky factorization (e.g., Ng and Peyton (1993)) were developed to manage dense columns efficiently. The software PCx of Czyzyk, Mehrotra, Wagner, and Wright (1999) employs K_1 in its interior method for linear optimization, i.e., $\phi(x) = c^T x$, but it does not allow for regularization.

With the advent of efficient sparse symmetric indefinite factorizations such as MA27 (Duff and Reid, 1982) and MA57 (Duff, 2004), the K_2 formulation became a popular alternative to K_1 because its condition, though unbounded, is somewhat more favorable. Matlab's sparse symmetric indefinite factorization `ldl(K)` calls MA57. The software OOQP of Gertz and Wright (2003) implements an approach similar to that of PCx for convex quadratic optimization, i.e., $\phi(x) = c^T x + \frac{1}{2} x^T H x$ with H symmetric positive semi-definite, based on K_2 and MA27 or MA57. In the presence of regularization, K_2 acquires special powers: it becomes *symmetric quasidefinite* (SQD). Vanderbei (1995) establishes that SQD matrices are *strongly factorizable*, i.e., any symmetric permutation possesses an LDL^T factorization with L unit lower triangular and D diagonal but indefinite. Such factorization, sometimes called *signed Cholesky* factorization, is computed by MA27 and MA57 when their pivot tolerance is set to $\mathtt{u} = 0$, and is cheaper than the more general factorization computed with $\mathtt{u} \in (0, 0.5]$. Vanderbei (1999) employs SQD LDL^T factorization in his solver LOQO.

[1] The fraction indicates that square roots of certain diagonal matrices appear in the formulation.

Our contributions here are: (i) to introduce a new formulation named $K_{2.5}$ that has the same memory requirements as K_2, is SQD in the presence of regularization, and has provably bounded condition number under strict complementarity; (ii) to provide bounds on the eigenvalues of $K_{2.5}$ during the interior-point iterations and in the limit; (iii) to illustrate those bounds numerically on several examples and show that they are remarkably tight; (iv) to show by experiment that $K_{2.5}$ performs favorably compared to K_2, K_3 and $K_{3.5}$; and (v) to introduce PDCOO, an object-oriented Matlab implementation of the PDCO solver (PDCO; Saunders, 2019) (designed to solve (1)) that lets users define new formulations of the linear system and new solution methods by way of multiple inheritance for fast experimentation.[2]

The rest of this paper is organized as follows. Section 2 provides background on interior-point methods for convex optimization, the most popular linear system formulations used in practice, and a few definitions. In Section 3, we state basic results on the inertia and eigenvalues of symmetric saddle-point matrices that are used in the derivation of novel results. Section 4 presents the new $K_{2.5}$ formulation of the linear system together with results on its inertia and bounds on its eigenvalues, both during the interior-point iterations and in the limit, in the spirit of Greif et al. (2014). In Section 5, we describe our object-oriented implementation of PDCO. Section 6 reports numerical experiments and contrasts $K_{2.5}$ with the most popular saddle-point formulations. Section 7 summarizes and provides perspectives for future research.

Notation

Lowercase letters x, y denote vectors, and e denotes the vector of ones whose size is given by the context. Uppercase letters A, H denote matrices. Greek letters λ, μ denote scalars. Cursive letters $\mathcal{A}$, $\mathcal{I}$ denote index sets, and $|\mathcal{A}|$ denotes the set cardinality. The identity matrix of size n is denoted by I_n, or just I when there is no ambiguity.

For a vector $x \in \mathbb{R}^n$, $x_{\max}$ and $x_{\min}$ denote the largest and smallest components of x, $X := \operatorname{diag}(x)$, and $\|x\|$ denotes the Euclidean norm. For a matrix A of any shape, $\sigma_{\max}(A)$ and $\sigma_{\min}(A)$ are the largest and smallest singular values of A, while $\lambda_{\max}(H)$ and $\lambda_{\min}(H)$ are the largest and smallest eigenvalues of a symmetric matrix H.

For a symmetric matrix M, the *inertia* of M is defined as the triple of integers $\operatorname{inertia}(M) = (n_+, n_-, n_0)$ representing the number of positive, negative, and zero eigenvalues of M, respectively.

If $\{\alpha_k\}$ and $\{\beta_k\}$ are two positive sequences, we write $\alpha_k = \Theta(\beta_k)$ to indicate that there exist constants $\gamma_1 > \gamma_2 > 0$ such that $\gamma_2\beta_k \le \alpha_k \le \gamma_1\beta_k$ for all sufficiently large k. In particular, $\alpha_k = \Theta(1)$ means that $\{\alpha_k\}$ is bounded and bounded away from zero.

2 Interior methods

To solve (1), an interior method such as PDCO solves approximately a sequence of barrier subproblems of the form

$$\begin{aligned} &\underset{x,\,r}{\text{minimize}}\ \phi(x) + \tfrac{1}{2}\|D_1 x\|^2 + \tfrac{1}{2}\|r\|^2 - \mu \sum_j \log x_j \\ &\text{subject to } Ax + D_2 r = b, \end{aligned} \tag{2}$$

where $\mu > 0$ is a barrier parameter that is initially of order 1 and is reduced steadily toward zero, and x is strictly positive.

Let y, z be Lagrange multipliers associated with the equality constraints and bounds in (1), and let $X := \operatorname{diag}(x)$, $Z := \operatorname{diag}(z)$, with z strictly positive. For the current value of $\mu > 0$, interior methods

[2] github.com/optimizers/PDCOO

compute an approximate solution to the optimality conditions for (2), which are perturbed optimality conditions for (1):

$$\nabla\phi(x) + D_1^2 x - A^T y - z = 0 \tag{3a}$$
$$Ax + D_2^2 y = b \tag{3b}$$
$$Xz = \mu e \tag{3c}$$
$$(x, z) > 0, \tag{3d}$$

from which we eliminated $r = D_2 y$. Linesearch-based interior methods for (1) apply Newton's method for nonlinear equations to (3). At an approximate solution (x, y, z) with $(x, z) > 0$, they compute search directions $\Delta x, \Delta y, \Delta z$ from systems of the form

$$\begin{bmatrix} -(H + D_1^2) & A^T & I \\ A & D_2^2 & \\ Z & & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} r_d \\ r_p \\ r_c \end{bmatrix}, \tag{K3}$$

with $H = \nabla^2\phi(x)$, $r_p := b - Ax - D_2^2 y$, $r_d := \nabla\phi(x) + D_1^2 x - A^T y - z$, and $r_c := \mu e - Xz$. The matrix K_3 in (K3) is unsymmetric but structurally symmetric. Its eigenvalues are real because it is similar to

$$K_{3.5} := DK_3D^{-1} = \begin{bmatrix} -(H + D_1^2) & A^T & Z^{\frac{1}{2}} \\ A & D_2^2 & \\ Z^{\frac{1}{2}} & & X \end{bmatrix}, \qquad D = \begin{bmatrix} I & & \\ & I & \\ & & Z^{-\frac{1}{2}} \end{bmatrix}, \tag{K3.5}$$

which is symmetric. Both K_3 and $K_{3.5}$ have size $(2n+m) \times (2n+m)$. Forsgren (2002) credits a private communication with Michael Saunders for the formulation (K3.5).

It is customary in the literature to symmetrize (K3) as

$$\begin{bmatrix} -(H + D_1^2) & A^T & I \\ A & D_2^2 & \\ I & & Z^{-1}X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} r_d \\ r_p \\ Z^{-1} r_c \end{bmatrix}, \tag{K3S}$$

or to perform one step of block elimination and obtain

$$\begin{bmatrix} -(H + D_1^2 - X^{-1}Z) & A^T \\ A & D_2^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} r_d - X^{-1} r_c \\ r_p \end{bmatrix}, \quad \Delta z = X^{-1}(r_c - Z\Delta x). \tag{K2}$$

A computational advantage of (K3.5), (K3S) and (K2) is that when D_2^2 and $H + D_1^2$ are positive definite, the block matrix is symmetric quasi-definite (SQD) (Vanderbei, 1995), permitting efficient solution by sparse signed-Cholesky factorization $PKP^T = LDL^T$ with D diagonal indefinite, and permutation P chosen to promote sparsity in L.

If x is feasible for (1), we denote the sets of active and inactive bounds by

$$\mathcal{A}(x) := \{j = 1, \dots, n \mid x_j = 0\} \quad \text{and} \quad \mathcal{I}(x) := \{j = 1, \dots, n \mid x_j > 0\}. \tag{4}$$

Definition 1 (LICQ) *If x is feasible for (1), we say that the linear independence constraint qualification is satisfied at x if $\begin{bmatrix} A^T & E \end{bmatrix}$ has full column rank, where E contains the columns of I_n corresponding to indices in $\mathcal{A}(x)$.*

Definition 2 (Strict complementarity) *If x is feasible for (1) and $(x, z) \geq 0$ satisfies $x_j z_j = 0$ for all $j = 1, \dots, n$, we say that x and z are strictly complementary if $x_j + z_j > 0$ for each j, i.e., if x_j and z_j do not vanish simultaneously.*

In a typical interior-point method, the iterates roughly follow a *central path*—a smooth parametrized curve $\mathcal{C} = \{(x(\mu), y(\mu), z(\mu)) \mid \mu \geq 0\}$ such that

1. at $\mu = 0$, $(x(0), y(0), z(0))$ is a solution of (1) under standard assumptions;
2. the following estimates hold as $\mu \searrow 0$:

$$x_i = \Theta(\mu) \quad (i \in \mathcal{A}), \qquad x_i = \Theta(1) \quad (i \in \mathcal{I}), \tag{5a}$$
$$z_i = \Theta(1) \quad (i \in \mathcal{A}), \qquad z_i = \Theta(\mu) \quad (i \in \mathcal{I}). \tag{5b}$$

If we assume without loss of generality that the variables are ordered as $x = (x_\mathcal{A}, x_\mathcal{I})$ and that $z = (z_\mathcal{A}, z_\mathcal{I})$ is ordered similarly, we have in the limit $x = (0, x_\mathcal{I})$, $z = (z_\mathcal{A}, 0)$. Accordingly, we decompose

$$X = \begin{bmatrix} X_\mathcal{A} & \\ & X_\mathcal{I} \end{bmatrix}, \quad Z = \begin{bmatrix} Z_\mathcal{A} & \\ & Z_\mathcal{I} \end{bmatrix}, \quad A = \begin{bmatrix} A_\mathcal{A} & A_\mathcal{I} \end{bmatrix}, \quad H = \begin{bmatrix} H_{\mathcal{A}\mathcal{A}} & H_{\mathcal{A}\mathcal{I}} \\ H_{\mathcal{A}\mathcal{I}}^T & H_{\mathcal{I}\mathcal{I}} \end{bmatrix},$$

and by complementarity, (K3) approaches the well-defined limit

$$\begin{bmatrix} -(H + D_1^2)_{\mathcal{A}\mathcal{A}} & -(H + D_1^2)_{\mathcal{A}\mathcal{I}} & A_\mathcal{A}^T & I & \\ -(H + D_1^2)_{\mathcal{I}\mathcal{A}} & -(H + D_1^2)_{\mathcal{I}\mathcal{I}} & A_\mathcal{I}^T & & I \\ A_\mathcal{A} & A_\mathcal{I} & D_2^2 & & \\ Z_\mathcal{A} & & & 0 & \\ & 0 & & & X_\mathcal{I} \end{bmatrix}.$$

If strict complementarity is satisfied in the limit, the above matrix is nonsingular and the condition number of K_3 remains uniformly bounded.

By the same logic, we conclude that (K3S) and (K2) have unbounded condition number, whereas (K3.5) also approaches a well-defined limit. Thus, employing (K3S) does not appear to have any advantage and we no longer consider it. Although its condition is unbounded, (K2) has the advantage of being $(n + m) \times (n + m)$, and it has been used extensively in the literature—see, e.g., (Friedlander and Orban, 2012; Fourer and Mehrotra, 1993; Gertz and Wright, 2003).

We refer to (Greif et al., 2014) for a complete description of (K3.5) and a comparison with (K3) and (K2).

The question arises whether there is a formulation of size $(n + m) \times (n + m)$ that remains well conditioned. The derivation and eigenvalue analysis of such a formulation are our main contributions.

3 Preliminary results

We first state a general result on the inertia of a saddle-point matrix.

Lemma 1 (Forsgren, 2002, Proposition 2**)** *Let* $A = A^T \in \mathbb{R}^{q \times q}$, $B \in \mathbb{R}^{t \times q}$, $C = C^T \in \mathbb{R}^{t \times t}$ *positive semidefinite,*

$$K := \begin{bmatrix} -A & B^T \\ B & C \end{bmatrix},$$

and $r := \mathrm{rank}(\begin{bmatrix} B & C \end{bmatrix})$. *Let the columns of* U *form a basis for* $\mathrm{Null}(C)$, *the columns of* N *form a basis for* $\mathrm{Null}(U^T B)$, *and* p *be the dimension of* $\mathrm{Null}(C)$. *Finally, let* $C^\dagger$ *denote the pseudo-inverse of* C. *Then*

$$\mathit{inertia}(K) = \mathit{inertia}\Big(-N^T (A + B^T C^\dagger B) N\Big) + (r,\, p - t + r,\, t - r).$$

In addition, $\mathrm{rank}(U^T B) = p - t + r$.

When $C = 0$, Lemma 1 reduces to Lemma 2, which we cite for completeness.

Lemma 2 (Gould, 1985, Lemma 3.4) *Let $A = A^T \in \mathbb{R}^{q\times q}$, $B \in \mathbb{R}^{t\times q}$ and*

$$K := \begin{bmatrix} -A & B^T \\ B & O \end{bmatrix}.$$

Let $r := \mathit{rank}(B)$ and the columns of N form a basis for $\mathrm{Null}(B)$. *Then*

$$\mathit{inertia}(K) = \mathit{inertia}(-N^T AN) + (r,\, r,\, t-r).$$

The following result can be used to derive eigenvalue bounds of regularized saddle-point matrices. It is inspired from earlier results by Rusten and Winther (1992) and Silvester and Wathen (1994).

Proposition 1 (Friedlander and Orban, 2012, Theorem 5.1) *Let $H = H^T \in \mathbb{R}^{n\times n}$ positive definite, $A \in \mathbb{R}^{m\times n}$, $I \in \mathbb{R}^{m\times m}$ the identity matrix, $\delta > 0$, $\lambda_{\max}$ and $\lambda_{\min}$ respectively the largest and smallest eigenvalues of H, $\sigma_{\max}$ and $\sigma_{\min}$ respectively the largest and smallest singular values of A. Let*

$$K := \begin{bmatrix} -H & A^T \\ A & \delta I \end{bmatrix}. \tag{6}$$

The eigenvalues of K are contained in the intervals $[\gamma^-_{\min}, \gamma^-_{\max}]$ and $[\gamma^+_{\min}, \gamma^+_{\max}]$, where $\gamma^-_{\min} \le \gamma^-_{\max} < 0 < \gamma^+_{\min} \le \gamma^+_{\max}$ and

$$\begin{aligned}
\gamma^-_{\min} &= \tfrac{1}{2}\left[\delta - \lambda_{\max} - \sqrt{[\lambda_{\max}+\delta]^2 + 4\sigma^2_{\max}}\right] \\
\gamma^-_{\max} &= -\lambda_{\min} \\
\gamma^+_{\min} &= \tfrac{1}{2}\left[\delta - \lambda_{\max} + \sqrt{[\lambda_{\max}+\delta]^2 + 4\sigma^2_{\min}}\right] \\
\gamma^+_{\max} &= \tfrac{1}{2}\left[\delta - \lambda_{\min} + \sqrt{[\lambda_{\min}+\delta]^2 + 4\sigma^2_{\max}}\right].
\end{aligned}$$

In addition, δ is the smallest positive eigenvalue of K if and only if A does not have full row rank. Its associated eigenspace is $\{0\} \times \mathrm{Null}(A^T)$ and its geometric multiplicity is $m - \mathit{rank}(A)$.

4 A new system formulation

We multiply the first block equation of (K3) by X and subtract the third block equation to obtain

$$\begin{bmatrix} -(X(H+D_1^2)+Z) & XA^T \\ A & D_2^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} Xr_d - r_c \\ r_p \end{bmatrix}. \tag{7}$$

This is a stable transformation if the original problem is sensibly scaled to make $\|x\| = O(1)$, $\|y\| = O(1)$, $\|z\| = O(1)$. Now consider the similarity transform

$$\begin{bmatrix} X^{-\frac{1}{2}} & \\ & I \end{bmatrix} \begin{bmatrix} -(X(H+D_1^2)+Z) & XA^T \\ A & D_2^2 \end{bmatrix} \begin{bmatrix} X^{\frac{1}{2}} & \\ & I \end{bmatrix} \begin{bmatrix} \Delta \bar{x} \\ \Delta y \end{bmatrix} = \begin{bmatrix} X^{-\frac{1}{2}}(Xr_d - r_c) \\ r_p \end{bmatrix},$$

which becomes

$$\begin{bmatrix} -(X^{\frac{1}{2}}(H+D_1^2)X^{\frac{1}{2}}+Z) & X^{\frac{1}{2}}A^T \\ AX^{\frac{1}{2}} & D_2^2 \end{bmatrix} \begin{bmatrix} \Delta \bar{x} \\ \Delta y \end{bmatrix} = \begin{bmatrix} X^{\frac{1}{2}}r_d - X^{-\frac{1}{2}}r_c \\ r_p \end{bmatrix}, \tag{K2.5}$$

where $\Delta x = X^{\frac{1}{2}}\Delta \bar{x}$. As components of x approach zero, the similarity transform zeros out columns of A, but if the LICQ is satisfied as a solution is approached, $AX^{\frac{1}{2}}$ remains of full row rank. As a bonus, we learn that the matrix of (7), though unsymmetric, has real eigenvalues.

Saunders (2019) suggests (K2.5) as an alternative to (K3) that is symmetric, smaller and reasonably well-conditioned. A further benefit is that the matrix $K_{2.5}$ in (K2.5) is SQD.

Korzak (1999) states a related matrix for the special case of linear optimization.

4.1 Eigenvalues of $\mathrm{K}_{2.5}$

We study the inertia of $K_{2.5}$ and bounds on its eigenvalues during the interior-point iterations and in the limit.

Proposition 2 (Inertia during the iterations) *Assume that $(x, z) > 0$ and that $\begin{bmatrix} AX^{\frac{1}{2}} & D_2^2 \end{bmatrix}$ has full row rank. The inertia of $K_{2.5}$ in (K2.5) is $(m, n, 0)$.*

Proof. Set $q = m$, $t = n$, and $r = t$ in Lemma 1. □

The rank assumption of Proposition 2 is satisfied if A has full row rank or D_2 is nonsingular. In the following, we assume $D_1 = \delta_1 I_n$ and $D_2 = \delta_2 I_m$, where $\delta_1 \geq 0$ and $\delta_2 > 0$. The results below are easily generalized to arbitrary positive definite diagonal matrices D_1 and D_2.

Let $\lambda_{\max} \geq \lambda_{\min} \geq 0$ be the extreme eigenvalues of H, and $\sigma_{\max} \geq \sigma_{\min} > 0$ be the extreme singular values of A.

Theorem 1 (Eigenvalues during the iterations) *The eigenvalues of $K_{2.5}$ in (K2.5) are in the intervals $[\rho^-_{\min}, \rho^-_{\max}]$ and $[\rho^+_{\min}, \rho^+_{\max}]$, where $\rho^-_{\min} \leq \rho^-_{\max} < 0 < \rho^+_{\min} \leq \rho^+_{\max}$, and*

$$\begin{aligned}
\rho^-_{\min} &= \tfrac{1}{2}\left[\delta_2^2 - \eta_{\max} - \sqrt{[\eta_{\max} + \delta_2^2]^2 + 4\sigma^2_{\max} x_{\max}}\right] \\
\rho^-_{\max} &= -\max((\lambda_{\min}(H) + \delta_1^2) x_{\min},\ \min_j(\delta_1^2 x_j + z_j)) \\
\rho^+_{\min} &= \tfrac{1}{2}\left[\delta_2^2 - \eta_{\max} + \sqrt{[\eta_{\max} + \delta_2^2]^2 + 4\sigma^2_{\min} x_{\min}}\right] \\
\rho^+_{\max} &= \tfrac{1}{2}\left[\delta_2^2 - \eta_{\min} + \sqrt{[\eta_{\min} + \delta_2^2]^2 + 4\sigma^2_{\max} x_{\max}}\right],
\end{aligned}$$

with $\eta_{\min} := (\lambda_{\min}(H) + \delta_1^2)\, x_{\min} + z_{\min}$ and $\eta_{\max} := (\lambda_{\max}(H) + \delta_1^2)\, x_{\max} + z_{\max}$.

If $\sigma_{\min} = 0$, then $\rho^+_{\min} = \delta_2^2$. Also, $AX^{\frac{1}{2}}$ is row-rank-deficient if and only if δ_2^2 is the smallest positive eigenvalue of (K2.5) with geometric multiplicity $m - \mathrm{rank}(AX^{\frac{1}{2}})$.

Proof. Let $W := X^{\frac{1}{2}}(H + D_1^2)X^{\frac{1}{2}} + Z$. Then,

$$\begin{aligned}
\lambda_{\max}(W) &= \max_{\|u\|=1} u^T X^{\frac{1}{2}}(H + D_1^2)X^{\frac{1}{2}} u + u^T Z u \\
&\leq \lambda_{\max}(H + D_1^2)\|X^{\frac{1}{2}}\|^2 + z_{\max} \\
&\leq \lambda_{\max}(H + D_1^2)\, x_{\max} + z_{\max} \\
&= \eta_{\max}.
\end{aligned}$$

By definition,

$$\sigma_{\max}(AX^{\frac{1}{2}}) = \max_{\|u\|=1} \|AX^{\frac{1}{2}} u\| \leq \sigma_{\max}(A) \max_{\|u\|=1} \|X^{\frac{1}{2}} u\| = \sigma_{\max}(A)\sqrt{x_{\max}}.$$

We now apply Proposition 1 to (K2.5) and obtain

$$\begin{aligned}
\gamma^-_{\min} &= \tfrac{1}{2}\left[\delta_2^2 - \lambda_{\max}(W) - \sqrt{(\delta_2^2 + \lambda_{\max}(W))^2 + 4\sigma_{\max}(AX^{\frac{1}{2}})^2}\right] \\
&\geq \tfrac{1}{2}\left[\delta_2^2 - \eta_{\max} - \sqrt{(\delta_2^2 + \eta_{\max})^2 + 4\sigma_{\max}(A)^2\, x_{\max}}\right] \\
&= \rho^-_{\min}.
\end{aligned}$$

Regarding the upper bound on the negative eigenvalues, observe that because H is positive semidefinite and $x > 0$, we have both

$$\lambda_{\min}(W) \geq \lambda_{\min}(X^{\frac{1}{2}} D_1^2 X^{\frac{1}{2}} + Z) = \min_j \delta_1^2 x_j + z_j,$$
$$\lambda_{\min}(W) \geq \lambda_{\min}(X^{\frac{1}{2}} (H + D_1^2) X^{\frac{1}{2}}) \geq (\lambda_{\min}(H) + \delta_1^2)\, x_{\min},$$

so that

$$\gamma_{\max}^- = -\lambda_{\min}(W) \leq -\max\bigl(\min_j((\lambda_{\min}(H) + \delta_1^2)x_j),\, \min_j(\delta_1^2 x_j + z_j)\bigr) = \rho_{\max}^-.$$

Similarly,

$$\sigma_{\min}(AX^{\frac{1}{2}}) = \min_{\|u\|=1} \|AX^{\frac{1}{2}} u\| \geq \sigma_{\min}(A) \min_{\|u\|=1} \|X^{\frac{1}{2}} u\| = \sigma_{\min}(A) \sqrt{x_{\min}}.$$

The remaining bounds are obtained as in the proof of (Friedlander and Orban, 2012, Theorem 5.1), where each occurrence of $\lambda_{\max}(W)$ and $\lambda_{\min}(W)$ is replaced by $\eta_{\max}$ and $\eta_{\min}$ respectively. □

Note that in Theorem 1, we do not use $-\eta_{\min}$ as the upper bound on the negative eigenvalues. Indeed, if there exist i and j such that $x_i = 0$ and $x_j > 0$ in the limit, as is typical, complementarity ensures that $z_j = 0$. Thus in the limit, $x_{\min} = z_{\min} = \eta_{\min} = 0$. As the next results show, $K_{2.5}$ approaches a nonsingular matrix in the limit, so that the negative eigenvalues are bounded away from zero and $-\eta_{\min}$ is not a useful bound. If strict complementarity is satisfied in the limit, then $x_j + z_j > 0$ for all components j and, consequently, $\rho_{\max}^- < 0$. The bound $(\lambda_{\min}(H) + \delta_1^2)x_{\min}$, though it appears to vanish in the limit, becomes useful when we bound the negative eigenvalues of the limit of $K_{2.5}$ below.

By complementarity, the limiting value of $K_{2.5}$ is

$$\begin{bmatrix} -Z_{\mathcal{A}} & & \\ & -G_{\mathcal{I}\mathcal{I}} & X_{\mathcal{I}}^{\frac{1}{2}} A_{\mathcal{I}}^T \\ & A_{\mathcal{I}} X_{\mathcal{I}}^{\frac{1}{2}} & D_2^2 \end{bmatrix}, \qquad G_{\mathcal{I}\mathcal{I}} := X_{\mathcal{I}}^{\frac{1}{2}} (H_{\mathcal{I}\mathcal{I}} + D_{1,\mathcal{I}}^2) X_{\mathcal{I}}^{\frac{1}{2}}. \tag{8}$$

Proposition 3 (Limiting inertia) *Assume* $\begin{bmatrix} A_{\mathcal{I}} X_{\mathcal{I}}^{\frac{1}{2}} & D_2^2 \end{bmatrix}$ *has full row rank, strict complementarity is satisfied, and* $G_{\mathcal{I}\mathcal{I}}$ *is positive definite. Then the inertia of* (8) *is* $(m, n, 0)$ *and the inertia of the bottom block* 2×2 *submatrix of* (8) *is* $(m, |\mathcal{I}|, 0)$.

Proof. The result follows from Proposition 2, the fact that $\begin{bmatrix} 0 & A_{\mathcal{I}} X_{\mathcal{I}}^{\frac{1}{2}} & D_2^2 \end{bmatrix}$ has full row rank, and the facts that $z_{\mathcal{A}} > 0$ and the leading block 2×2 submatrix of (8) is nonsingular. □

We now turn our attention to eigenvalues in the limit.

Theorem 2 (Limiting eigenvalues) *Assume strict complementarity is satisfied and* $G_{\mathcal{I}\mathcal{I}}$ *is positive definite. If* $\begin{bmatrix} A_{\mathcal{I}} X_{\mathcal{I}}^{\frac{1}{2}} & D_2^2 \end{bmatrix}$ *has full row rank,* (8) *has* $|\mathcal{A}|$ *negative eigenvalues equal to the components of* $-z_{\mathcal{A}}$. *The remaining eigenvalues are in the intervals* $[\nu_{\min}^-, \nu_{\max}^-]$ *and* $[\nu_{\min}^+, \nu_{\max}^+]$, *where* $\nu_{\min}^- \leq \nu_{\max}^- < 0 < \nu_{\min}^+ \leq \nu_{\max}^+$ *and*

$$\begin{aligned}
\nu_{\min}^- &= \tfrac{1}{2}\left[\delta_2^2 - \eta_{\max,\mathcal{I}} - \sqrt{[\eta_{\max,\mathcal{I}} + \delta_2^2]^2 + 4\sigma_{\max}^2 x_{\max,\mathcal{I}}}\right] \\
\nu_{\max}^- &= -(\lambda_{\min}(H) + \delta_1^2) x_{\min,\mathcal{I}} \\
\nu_{\min}^+ &= \tfrac{1}{2}\left[\delta_2^2 - \eta_{\max,\mathcal{I}} + \sqrt{[\eta_{\max,\mathcal{I}} + \delta_2^2]^2 + 4\sigma_{\min}^2 x_{\min,\mathcal{I}}}\right] \\
\nu_{\max}^+ &= \tfrac{1}{2}\left[\delta_2^2 - \eta_{\min,\mathcal{I}} + \sqrt{[\eta_{\min,\mathcal{I}} + \delta_2^2]^2 + 4\sigma_{\max}^2 x_{\max,\mathcal{I}}}\right],
\end{aligned}$$

where $\sigma_{\min} = 0 \Rightarrow \nu^{+}_{\min} = \delta_2^2$*;*

$$x_{\min,\mathcal{I}} := \min_{j\in\mathcal{I}} x_j, \qquad \eta_{\min,\mathcal{I}} := (\lambda_{\min}(H) + \delta_1^2)\, x_{\min,\mathcal{I}},$$
$$x_{\max,\mathcal{I}} := \max_{j\in\mathcal{I}} x_j, \qquad \eta_{\max,\mathcal{I}} := (\lambda_{\max}(H) + \delta_1^2)\, x_{\max,\mathcal{I}}.$$

Further, $A_{\mathcal{I}} X_{\mathcal{I}}^{\frac{1}{2}}$ *is row-rank deficient iff* δ_2^2 *is the smallest positive eigenvalue of* (8) *with geometric multiplicity* $|\mathcal{I}| - \mathrm{rank}(A_{\mathcal{I}} X_{\mathcal{I}}^{\frac{1}{2}})$*.*

Proof. The result follows from Theorem 1, the block-diagonal structure of (8), and complementarity. □

We close this section by examining special cases. The proofs are straightforward and follow from the continuity of eigenvalues by setting $\delta_1 = \delta_2 = 0$.

Corollary 1 (Eigenvalues during the iterations without regularization) *When* $\delta_1 = \delta_2 = 0$*, the eigenvalues of* $K_{2.5}$ *in* (K2.5) *are in the intervals* $[\rho^{-}_{\min,0}, \rho^{-}_{\max,0}]$ *and* $[\rho^{+}_{\min,0}, \rho^{+}_{\max,0}]$*, where* $\rho^{-}_{\min,0} \le \rho^{-}_{\max,0} < 0 \le \rho^{+}_{\min,0} \le \rho^{+}_{\max,0}$*, and*

$$\begin{aligned}
\rho^{-}_{\min,0} &= -\tfrac{1}{2}\left[\eta_{\max,0} + \sqrt{\eta^2_{\max,0} + 4\sigma^2_{\max} x_{\max}}\right] \\
\rho^{-}_{\max,0} &= -\max(\lambda_{\min}(H) x_{\min},\, z_{\min}) \\
\rho^{+}_{\min,0} &= \tfrac{1}{2}\left[-\eta_{\max,0} + \sqrt{\eta^2_{\max,0} + 4\sigma^2_{\min} x_{\min}}\right] \\
\rho^{+}_{\max,0} &= \tfrac{1}{2}\left[-\eta_{\min,0} + \sqrt{\eta^2_{\min,0} + 4\sigma^2_{\max} x_{\max}}\right],
\end{aligned}$$

with $\eta_{\min,0} := \lambda_{\min}(H)\, x_{\min} + z_{\min}$*,* $\eta_{\max,0} := \lambda_{\max}(H)\, x_{\max} + z_{\max}$*, and* $\sigma_{\min} = 0 \Rightarrow \rho^{+}_{\min,0} = 0$*. In the case of linear optimization,* $\lambda_{\min}(H) = \lambda_{\max}(H) = 0$*, so that* $\eta_{\min,0} = z_{\min}$*,* $\eta_{\max,0} = z_{\max}$*, and* $\rho^{-}_{\max,0} = -z_{\min}$*.*

Corollary 2 (Limiting eigenvalues without regularization) *Under the assumptions of Theorem 2, when* $\delta_1 = \delta_2 = 0$*,* (8) *has* $|\mathcal{A}|$ *negative eigenvalues equal to the components of* $-z_{\mathcal{A}}$*. The remaining eigenvalues are in the intervals* $[\nu^{-}_{\min,0}, \nu^{-}_{\max,0}]$ *and* $[\nu^{+}_{\min,0}, \nu^{+}_{\max,0}]$*, where* $\nu^{-}_{\min,0} \le \nu^{-}_{\max,0} \le 0 \le \nu^{+}_{\min,0} \le \nu^{+}_{\max,0}$ *and*

$$\begin{aligned}
\nu^{-}_{\min,0} &= -\tfrac{1}{2}\left[\eta_{\max,\mathcal{I},0} + \sqrt{\eta^2_{\max,\mathcal{I},0} + 4\sigma^2_{\max} x_{\max,\mathcal{I}}}\right] \\
\nu^{-}_{\max,0} &= -\lambda_{\min}(H) x_{\min,\mathcal{I}} \\
\nu^{+}_{\min,0} &= \tfrac{1}{2}\left[-\eta_{\max,\mathcal{I},0} + \sqrt{\eta^2_{\max,\mathcal{I},0} + 4\sigma^2_{\min} x_{\min,\mathcal{I}}}\right] \\
\nu^{+}_{\max,0} &= \tfrac{1}{2}\left[-\eta_{\min,\mathcal{I},0} + \sqrt{\eta^2_{\min,\mathcal{I},0} + 4\sigma^2_{\max} x_{\max,\mathcal{I}}}\right],
\end{aligned}$$

with $x_{\min,\mathcal{I}} := \min_{j\in\mathcal{I}} x_j$*,* $x_{\max,\mathcal{I}} := \max_{j\in\mathcal{I}} x_j$*,* $\eta_{\min,\mathcal{I},0} := \lambda_{\min}(H)\, x_{\min,\mathcal{I}}$ *and* $\eta_{\max,\mathcal{I},0} := \lambda_{\max}(H)\, x_{\max,\mathcal{I}}$*. In addition,* $A_{\mathcal{I}} X_{\mathcal{I}}^{\frac{1}{2}}$ *is row-rank deficient iff zero is an eigenvalue of* (8) *with geometric multiplicity* $|\mathcal{I}| - \mathrm{rank}(A_{\mathcal{I}} X_{\mathcal{I}}^{\frac{1}{2}})$*.*

In the case of linear optimization, $\lambda_{\min}(H) = \lambda_{\max}(H) = 0$*, so that* $\eta_{\min,\mathcal{I},0} = \eta_{\max,\mathcal{I},0} = 0$*, and*

$$\nu^{-}_{\min,0} = -\sigma_{\max}\sqrt{x_{\max,\mathcal{I}}}, \qquad \nu^{-}_{\max,0} = 0,$$
$$\nu^{+}_{\min,0} = \sigma_{\min}\sqrt{x_{\min,\mathcal{I}}}, \qquad \nu^{+}_{\max,0} = \sigma_{\max}\sqrt{x_{\max,\mathcal{I}}},$$

where $\sigma_{\min} = 0 \Rightarrow \nu^{+}_{\min,0} = 0$*.*

5 An interior solver for convex optimization

In PDCO, μ is regarded as an extra variable and updated according to $\mu \leftarrow (1-\alpha)\mu$, where α is the steplength for the current iteration ($0 < \alpha \le 1$). PDCO and PDCOO work with the problem

$$\begin{aligned} &\min_{x,\,r} \ \phi(x) + \tfrac{1}{2}\|D_1 x\|^2 + \tfrac{1}{2}\|r\|^2 \\ &\text{s.t. } Ax + D_2 r = b, \quad \ell \le x \le u \end{aligned} \tag{9}$$

with general bounds $\ell,\, u \in \mathbb{R}^n$. The bounds are equivalent to constraints $x - x_1 = \ell$, $x + x_2 = u$, $x_1 \ge 0$, $x_2 \ge 0$. The K_3 system now reads

$$\begin{bmatrix} -(H + D_1^2) & A^T & I & -I \\ A & D_2^2 & & \\ Z_1 & & X_1 & \\ -Z_2 & & & X_2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z_1 \\ \Delta z_2 \end{bmatrix} = \begin{bmatrix} r_d \\ r_p \\ r_{c,1} \\ r_{c,2} \end{bmatrix},$$

where rows and columns are zero in the diagonal matrices X_1, Z_1 or X_2, Z_2 if the corresponding elements of ℓ or u are infinite. The resulting $K_{2.5}$ system is

$$\begin{bmatrix} -(X_1^{\frac{1}{2}} X_2^{\frac{1}{2}}(H + D_1^2) X_1^{\frac{1}{2}} X_2^{\frac{1}{2}} + X_2 Z_1 + X_1 Z_2) & X_1^{\frac{1}{2}} X_2^{\frac{1}{2}} A^T \\ A X_1^{\frac{1}{2}} X_2^{\frac{1}{2}} & D_2^2 \end{bmatrix} \begin{bmatrix} \Delta \bar{x} \\ \Delta y \end{bmatrix} = \begin{bmatrix} X_1^{\frac{1}{2}} X_2^{\frac{1}{2}} r_d - X_1^{-\frac{1}{2}} X_2^{\frac{1}{2}} r_{c,1} + X_1^{\frac{1}{2}} X_2^{-\frac{1}{2}} r_{c,2} \\ r_p \end{bmatrix}, \tag{10}$$

where $\Delta x = X_1^{\frac{1}{2}} X_2^{\frac{1}{2}} \Delta \bar{x}$.

If a problem has linear inequality constraints $b_\ell \le Ax \le b_u$, we add slack variables to reformulate those constraints as $Ax - s = 0$, and $b_\ell \le s \le b_u$.

We now describe our object-oriented implementation of PDCO, named PDCOO and available from `github.com/optimizers/PDCOO`. Below, we refer to the combination of a transformation of (K3) and of a method to solve those linear equations as a *variant*. In the original PDCO, the user indicates by an integer which variant should be used during the iterations. While functional, this approach makes the process of adding a new variant fragile as multiple parts of the code must be changed. PDCOO restructures the entire method by separating variants from the main interior-point loop, and by separating each variant into a transformation of (K3) and a method to solve the linear system. A variant of the user's choosing and the main loop are subsequently assembled by way of multiple inheritance. The top-level source repository contains the main interior-point loop in `pdcoO.m` and the folders `Formulations`, `Solvers`, and `Variants`. Initially, `Variants` is empty. `Formulations` contains several files, each of which contains a small amount of code to implement a transformation of (K3). Among them, we find `K2.m`, `K25.m`, `K3.m`, and `K35.m`. The implementation of a formulation consists of the definition of a Matlab class that implements the `Solve_Newton()` method, and, in essence, looks as in Listing 1.

The `K25` class features an abstract method named `Solver()`. An abstract method is a method that is declared as belonging to the `K25` class but that is expected to be implemented by a subclass or by another class that will be combined with `K25` by way of multiple inheritance. The main interior-point loop itself is implemented inside the `pdcoO` class, and the latter defines the abstract method `Solve_Newton()`.

The constructor of `K25` simply initializes a boolean variable to indicate that, unlike other variants, it is not specifically designed for cases where H is diagonal.

The important method is `Solve_Newton()`. Its role is to assemble the matrix of the formulation of (K3) of interest, store it in the attribute `M`, assemble the corresponding right-hand side, store it in

```
classdef K25 < handle
properties
M     % a matrix that represents K2.5
rhs   % the right-hand side corresponding to K2.5
sol   % a vector to contain the solution of the system
end

methods (Abstract)
Solver(o)  % an abstract method to be implemented by the solver
end

methods
function o = K25(options)  % constructor
o.diagHess = false;      % K2.5 is not specifically for diagonal H
end

function Solve_Newton(o)
% ... construct K2.5 and store it in o.M
% ... construct the right-hand side of (K25) and store it in o.rhs
Solver(o);  % call solver, which stores the solution in o.sol
% ... recover solution of (K3) from o.sol
end
end
end
```

Listing 1: The K25 formulation.

the attribute `rhs`, call the linear system solver, which has not yet been defined at this point, but which will store the solution in the attribute `sol`, and finally to extract the solution of (K3) from `sol`.

The formulation's abstract method `Solver()` is implemented by one of the classes stored under the `Solvers` folder, which provide a solution method for the linear system. Listing 2 shows the essential parts of one such class designed for SQD systems such as (K2.5) in which the system is solved by way of a signed Cholesky factorization. The latter is computed via Matlab's `ldl()` function with the pivot threshold parameter set to zero to prevent the factorization from pivoting for stability.

```
classdef LDL < handle
properties
L
end

methods
function o = LDL(options)
o.need_precon = false;  % this method requires no preconditioner
end

function Solver(o)
thresh = 0;     % tells MA57 to keep its sparsity-preserving order
[o.L, D, P, S] = ldl(o.M, thresh);  % o.M was set in the K25 class
o.sol = S * (P * (o.L' \ (D \ (o.L \ (P' * (S * o.rhs))))));
end
end
end
```

Listing 2: The LDL solver.

The above definitions allow the user to assemble a complete solver by putting together the `pdcoO`, `K25` and `LDL` classes using multiple inheritance. That is achieved either by writing a class by hand that inherits from the previous three, or by calling the `build_variant()` function. The latter takes three arguments: the location of the top PDCOO folder, a string to specify the formulation class, and a string to specify the solver class. If we assume that `/home/user/pdcoo` is the path to the top PDCOO folder, the call `build_variant('/home/user/pdcoo', 'K25', 'LDL')` assembles a complete solver by creating a new file under the `Variants` folder named `pdco_K25_LDL.m` that contains the declaration of the `pdco_K25_LDL` class shown in Listing 3.

```
classdef pdco_K25_LDL < pdcoO & K25 & LDL  % inherit from three classes
properties
end

methods
function o = pdco_K25_LDL(slack, opts_pdco, opts_form, opts_solv)
o = o@pdcoO(slack, opts_pdco);
o = o@K25(opts_form);
o = o@LDL(opts_solv);
end
end
end
```

Listing 3: The assembled solver.

The constructor of the assembled solver takes as arguments a model in which linear inequalities were converted to equalities and bounds by way of slack variables, as explained above, along with options to pass to the interior-point solver class, the formulation class and the linear system solver class.

PDCOO expects optimization problems to adhere to a specific format defined in the `model` package, available from `github.com/optimizers/model`. The `model` package defines a base class `nlpmodel` and a number of subclasses specialized either by problem type or provenance. Most relevant to the present paper, the `lpmodel` and `qpmodel` classes are used to represent linear and quadratic optimization models, respectively. The latter can be read from files in MPS or QPS format by a reader included with PDCOO. The additional class `slackmodel` takes as input a model and adds slack variables as specified above. The user should pass an instance of `slackmodel` to PDCOO so the problem has the form (9).

Finally, a complete session might look as in Listing 4. The numerical experiments of Section 6 are carried out using commands similar to those of Listing 4. The commands first import the relevant classes from the `model` package, build the variant of interest, read a problem from an MPS file and add slack variables, set a number of PDCO options, instantiate the variant, and finally solve the problem. The PDCO options are the same as those described by Orban (2015) and set regularization parameters, an initial guess, scaling factors `xsize` and `zsize`, and an initial barrier parameter.

6 Numerical experiments

We illustrate the eigenvalue bounds of the previous sections on a selection of problems from the TOMLAB collection.[3] Our test set consists of the linear optimization collection[4] and the quadratic optimization collection[5] (90 and 130 problems respectively). We exclude `cre-c` and `qforplan`, which take substantially more time to solve than the rest, for a total of 218 problems.

In a first set of experiments, we compute all eigenvalues of $K_{2.5}$ at each iteration of PDCO on the linear problems `small009` and `nsic2`, which both satisfy the LICQ and strict complementarity at the solution. The results on those two problems are representative of what we have observed on problems satisfying the LICQ and strict complementarity. The original formulation of `small009` has $1,135$ variables and 710 constraints, and we add 298 slack variables. Problem `nsic2` has 463 variables and 465 constraints, and we add 434 slack variables.

[3] `tomopt.com/tomlab`
[4] `tomopt.com/docs/models/tomlab_models034.php`
[5] `tomopt.com/docs/models/tomlab_models036.php`

```
import model.lpmodel;
import model.slackmodel;

classname = build_variant(pdcoo_home, 'K25', 'LDL');

% read .mps file and add slack variables
mps_data = readmps('afiro.mps');
lp = mpstolp(mps_data);
slack = slackmodel(lp);

% define PDCOO options
Anorm = normest(slack.gcon(slack.x0), 1.0e-3);
options_pdco.d1 = 1.0e-2;
options_pdco.d2 = 1.0e-2;
options_pdco.x0 = slack.x0;
options_pdco.x0(slack.jLow) = slack.bL(slack.jLow) + 1;
options_pdco.x0(slack.jUpp) = slack.bU(slack.jUpp) - 1;
options_pdco.x0(slack.jTwo) = (slack.bL(slack.jTwo) + ...
slack.bU(slack.jTwo)) / 2;
options_pdco.xsize = max(norm(options_pdco.x0, inf), 1);
options_pdco.zsize = max(norm(slack.gobj(slack.x0), inf) + ...
sqrt(slack.n) * Anorm, 1);
options_pdco.z0 = options_pdco.zsize * ones(slack.n, 1);
options_pdco.y0 = zeros(slack.m, 1);
options_pdco.mu0 = options_pdco.zsize;
options_pdco.Maxiter = min(max(30, slack.n), 100);
options_form = struct();  % no particular options for the formulation
options_solv = struct();  % no particular options for the solver

Problem = eval([classname,
'(slack, options_pdco, options_form, options_solv)']);
Problem.solve;
```

Listing 4: An example session with PDCOO.

Eigenvalues are computed using Matlab's `eigs()` function, from which we request the entire spectrum. We plot eigenvalues on a symmetric logarithmic scale using dots, and superpose curves representing the inner and outer bounds of Theorem 1. Evaluation of the bounds requires computing $\sigma_{\min}(A)$ and $\sigma_{\max}(A)$ using Matlab's `svds()` function, from which we request the extreme singular values only.

Figures 1 and 2 illustrate the results for problem `small009`, where $K_{2.5}$ has size $2,143$. We note that the inner bounds are especially tight throughout the iterations, while the outer bounds are tight in the early stages and looser later. The positive eigenvalues are safely bounded away from zero, essentially by δ_2^2. The negative eigenvalues, though small in magnitude, are also bounded away from zero. The upper bound on negative eigenvalues only depends on δ_1. In the case of linear optimization, it becomes

$$\rho^-_{\max} = -\max(\delta_1^2 x_{\min}, \min_j(\delta_1^2 x_j + z_j))$$

during the iterations and

$$\nu^-_{\max} = -\delta_1^2 x_{\min,\mathcal{I}}$$

in the limit. Thus, some negative eigenvalues can be perilously close to zero unless the problem is scaled so that $x_{\min,\mathcal{I}}$ is not too small at the solution. Fortunately, that behavior occurs only in the last few iterations.

Figures 3 and 4 illustrate the eigenvalues and bounds for problem `nsic2`, where $K_{2.5}$ has size $1,362$.

If A does not have full row rank, $\sigma_{\min}(A) = 0$ and the lower bound on positive eigenvalues of Theorem 1 and Theorem 2 becomes $\rho^+_{min} = \nu^+_{\min} = \delta_2^2$. Figures 5 and 6 illustrate that bound for several values of δ_2 on quadratic problem `qbrandy`, where the number of variables is 249, the number of original

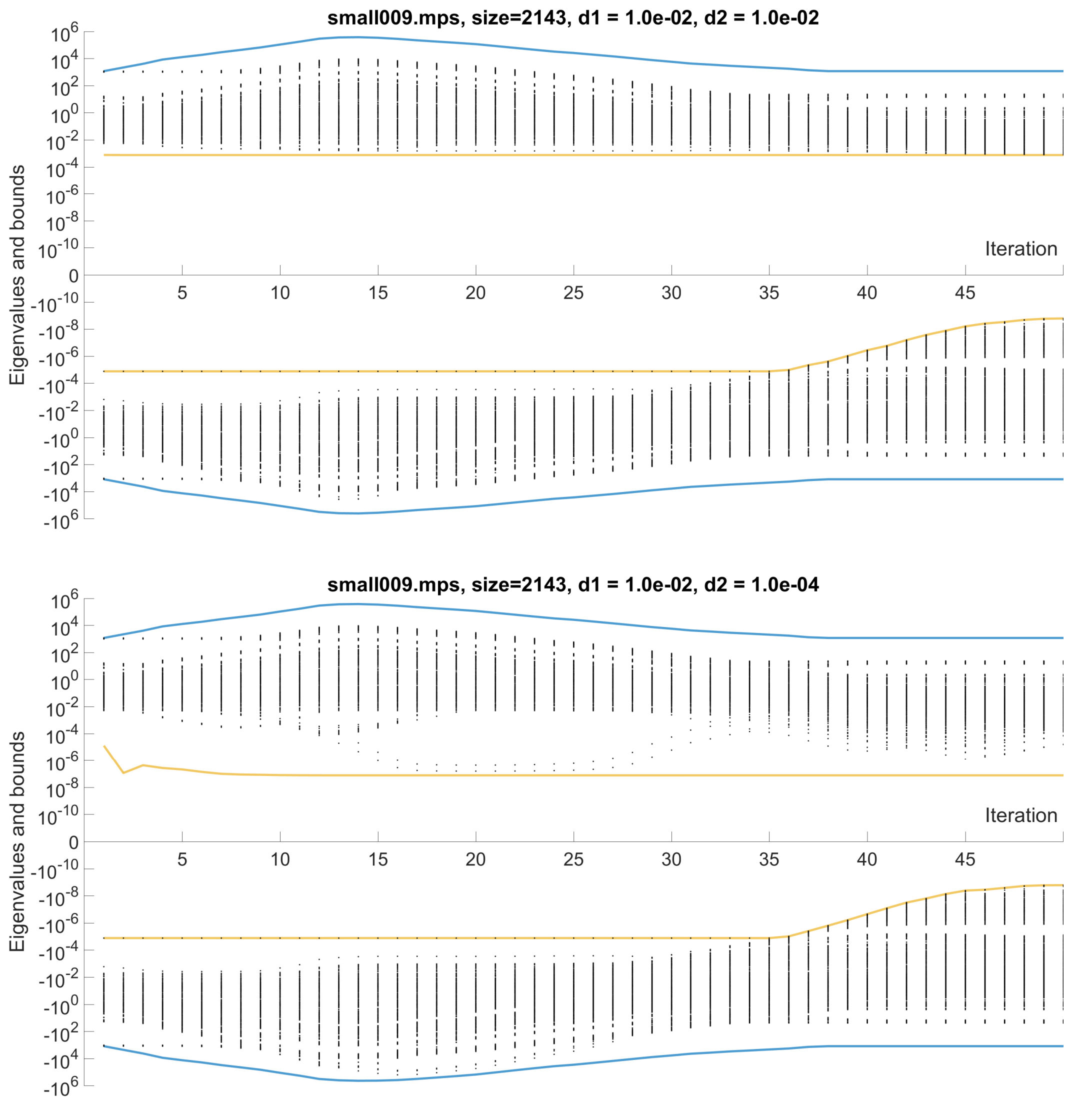

Figure 1: Eigenvalues and bounds of Theorem 1 for linear problem `small009`.

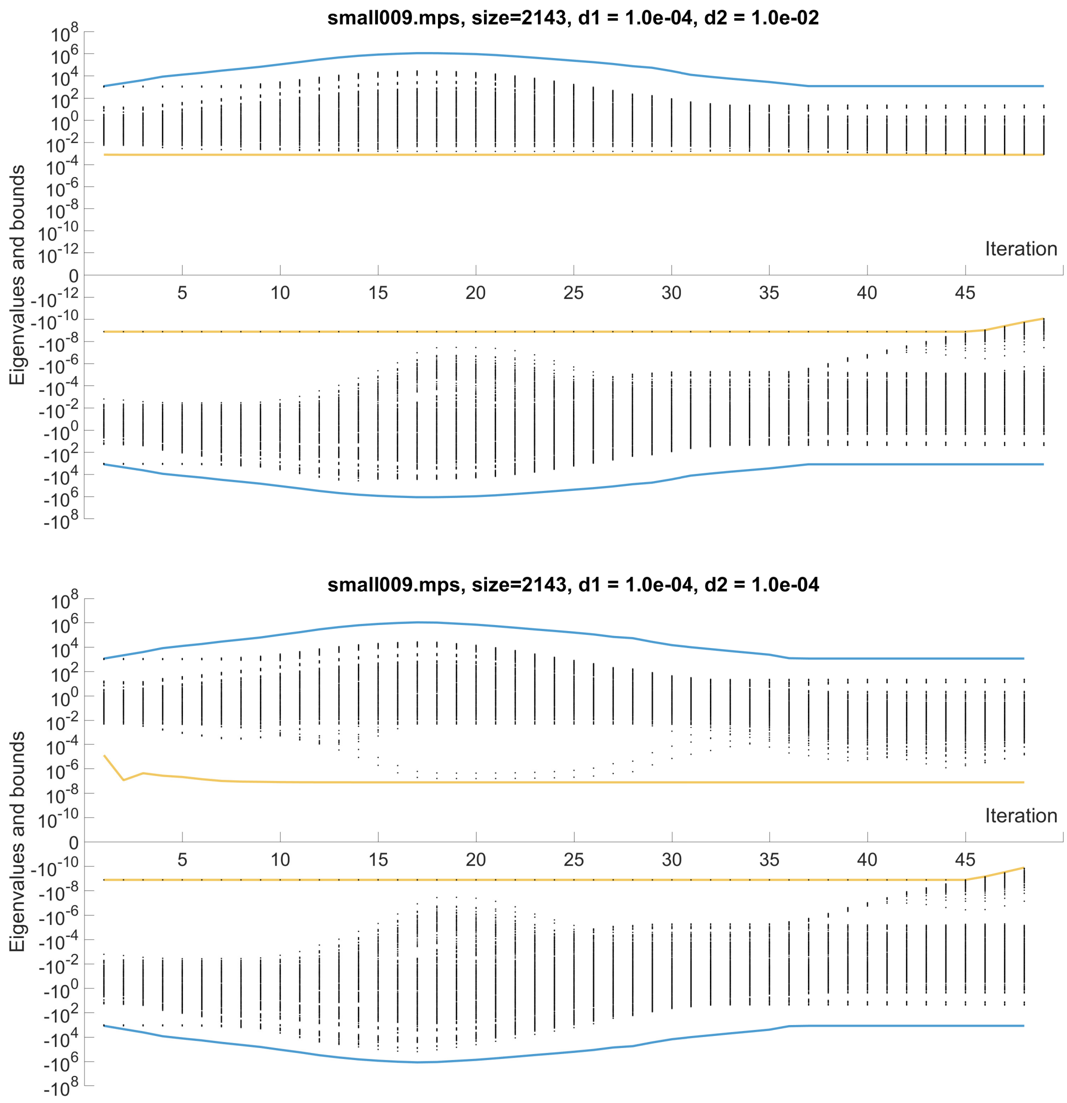

Figure 2: Eigenvalues and bounds of Theorem 1 for linear problem `small009` (continued).

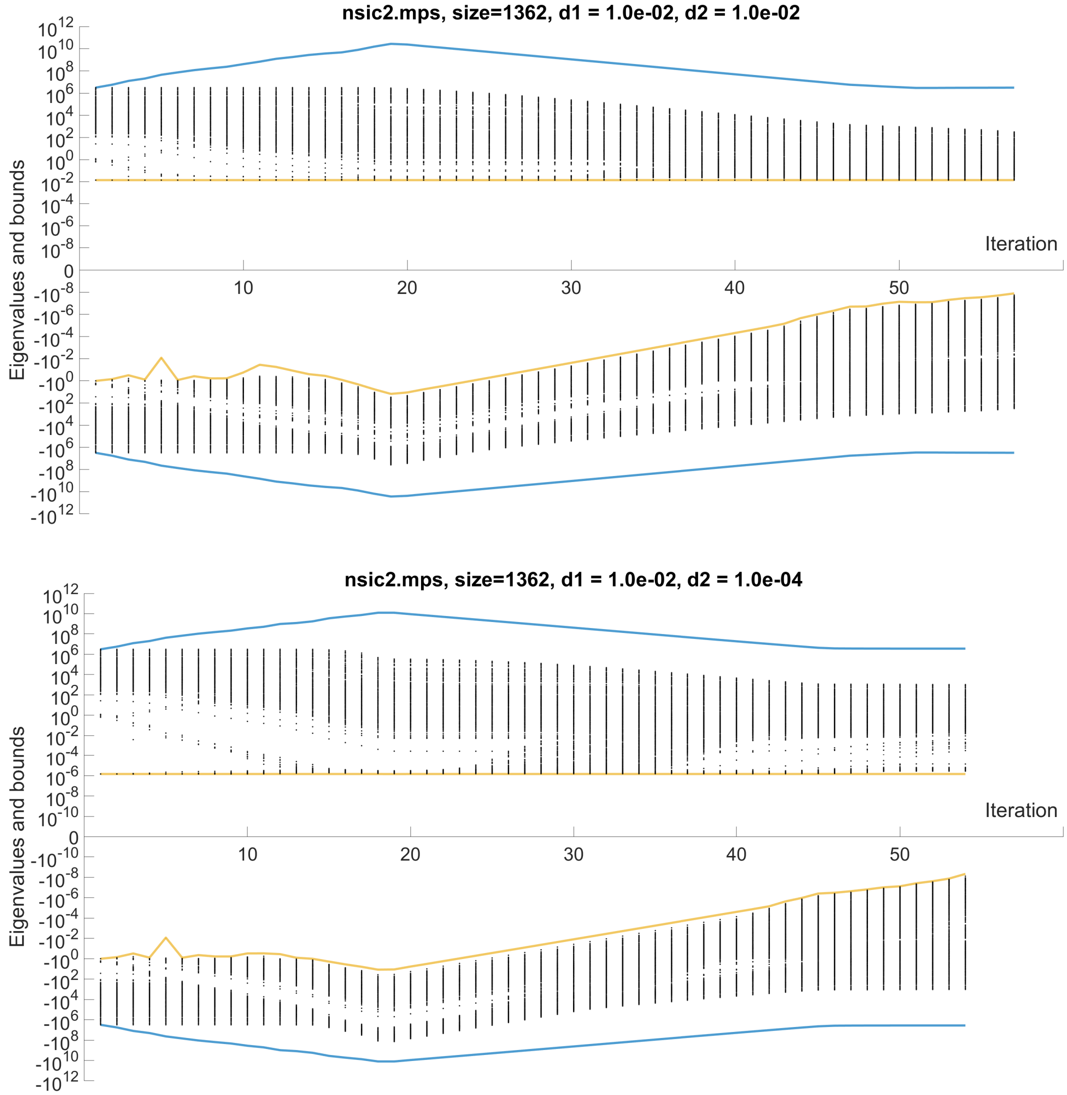

Figure 3: Eigenvalues and bounds of Theorem 1 for linear problem `nsic2`.

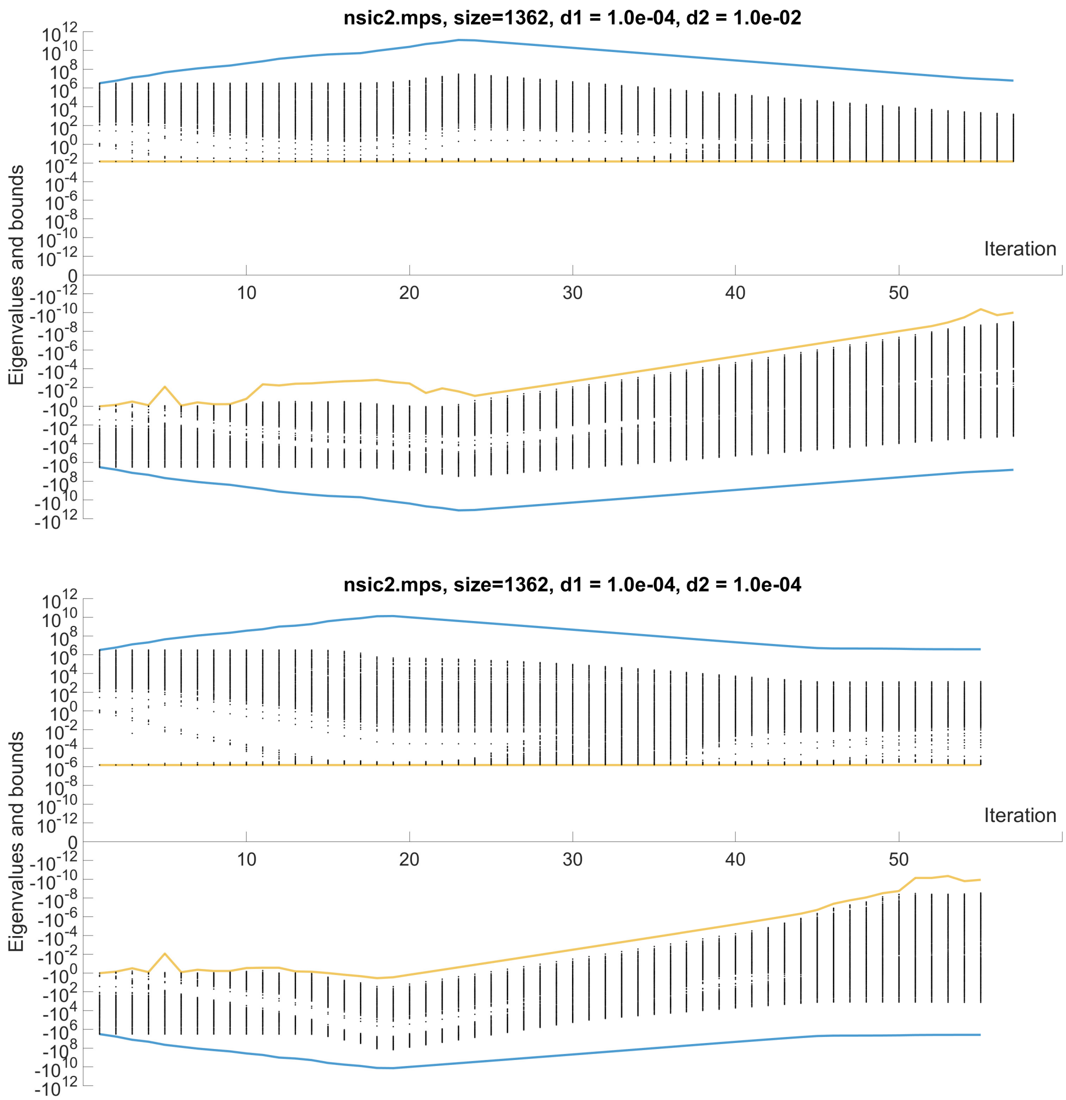

Figure 4: Eigenvalues and bounds of Theorem 1 for linear problem `nsic2` (continued).

constraints is 220, and we add 83 slack variables. The resulting Jacobian A, including the contributions of the slack variables, has size 220×303 and rank 193. In the figure, note that the lower bound on the positive eigenvalues is constant, tight, and decreases with δ_2. The reason why the bound visibly differs from δ_2^2 is that each problem is scaled prior to solution, and the scaling affects the value of δ_2 effectively used during the iterations.

When δ_2 reaches 10^{-6}, several negative eigenvalues appear to lie outside their bounds for iterations 19 to 24. When $\delta_2 = 10^{-8}$, the effect is more dramatic and numerous positive and negative eigenvalues appear to lie outside their bounds. However, this misleading effect is due to accumulated rounding errors in the `eigs()` function. In order to verify our claim, we ran the eigenvalue computation in extended precision using the Matlab Symbolic Math toolbox with the *variable precision arithmetic* (vpa) set to 64 digits.[6] Because `eigs()` does not accept vpa input, we computed eigenvalues using `eig()`. The resulting eigenvalues and bounds are shown in the bottom plot of Figure 6.

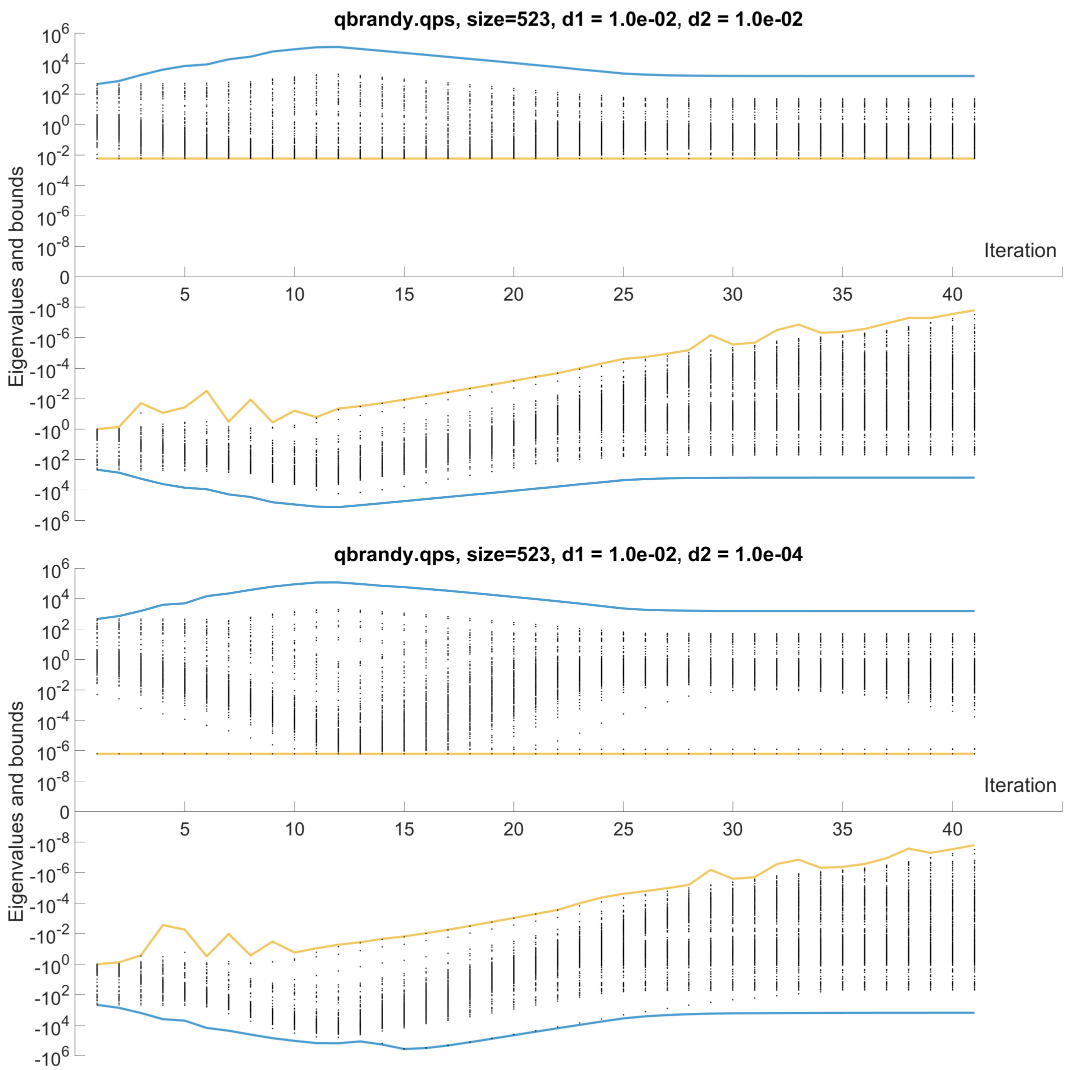

Figure 5: Eigenvalues and bounds of Theorem 1 for quadratic problem `qbrandy`, where $\sigma_{\min}(A) = 0$. The value of δ_1 is fixed to 10^{-2} and a range of values for δ_2 is selected.

[6] By default, Matlab uses 16 digits in double precision; 64 digits corresponds to octuple precision.

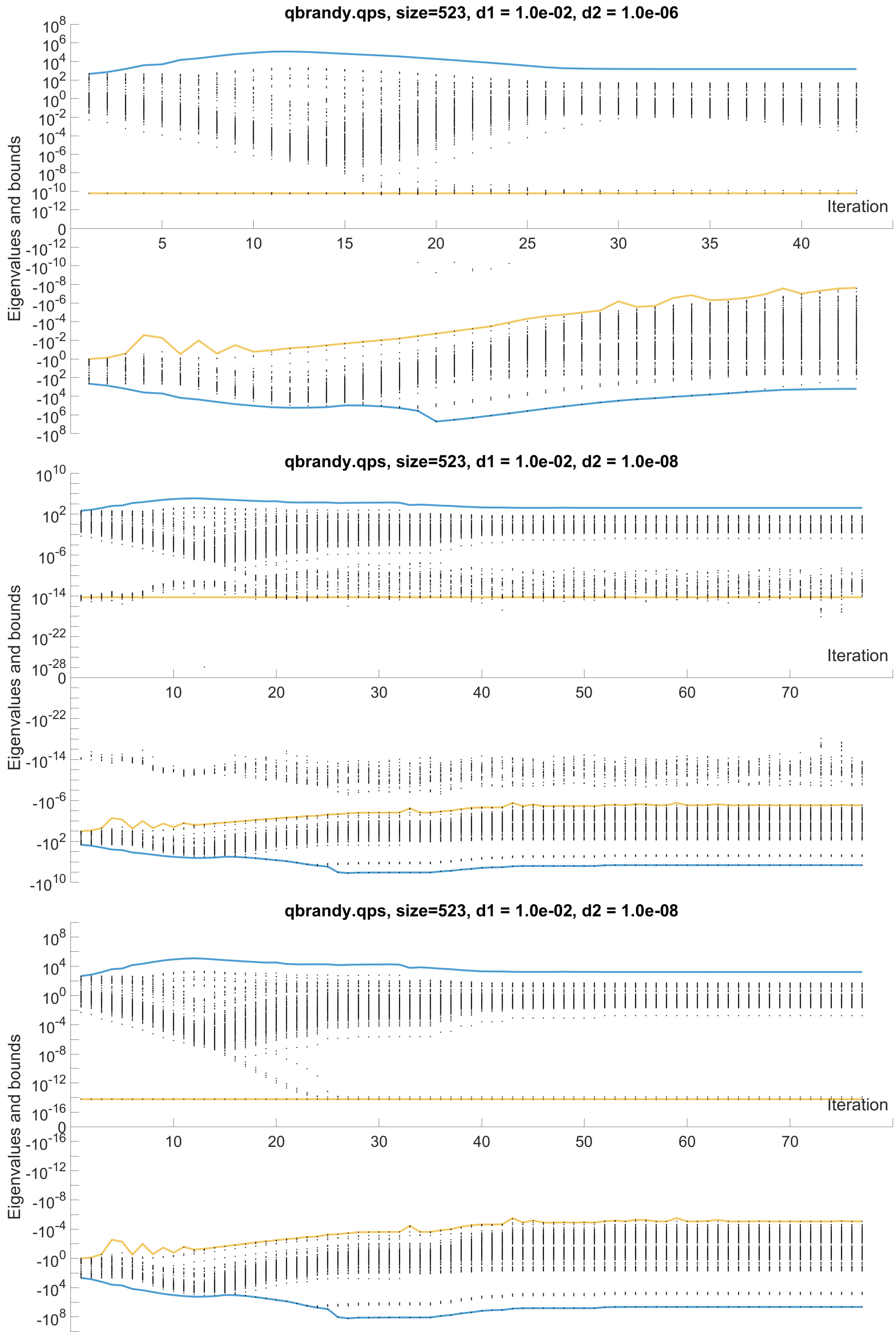

Figure 6: Eigenvalues and bounds of Theorem 1 for quadratic problem `qbrandy`, where $\sigma_{\min}(A) = 0$ (continued). We use $\delta_1 = 10^{-2}$ and a range of values for δ_2. In the bottom plot, the eigenvalues are computed using extended precision.

6.1 Comparison with K_3, $K_{3.5}$ and K_2

Greif et al. (2014) provide eigenvalue bounds and condition number estimates for (K3.5). If strict complementarity is satisfied in the limit, $K_{3.5}$ remains well-conditioned like $K_{2.5}$, but it is substantially larger without being usefully more sparse. In this section, we compare the eigenvalue distribution and condition number of the formulations (K3), (K3.5), (K2) and (K2.5).

Figure 7 compares the evolution of the condition number of each formulation on problems `small009`, `nsic2` and `qbrandy`. The top three plots show the condition number at each iteration. The solid curve is an *ad hoc* upper bound on $\mathrm{cond}(K_{2.5})$ obtained from the bounds of Theorem 1 by computing the ratio of the largest to the smallest bound in absolute value, i.e., $\max(\rho^+_{\max}, |\rho^-_{\min}|)/\min(\rho^+_{\min}, |\rho^-_{\max}|)$.

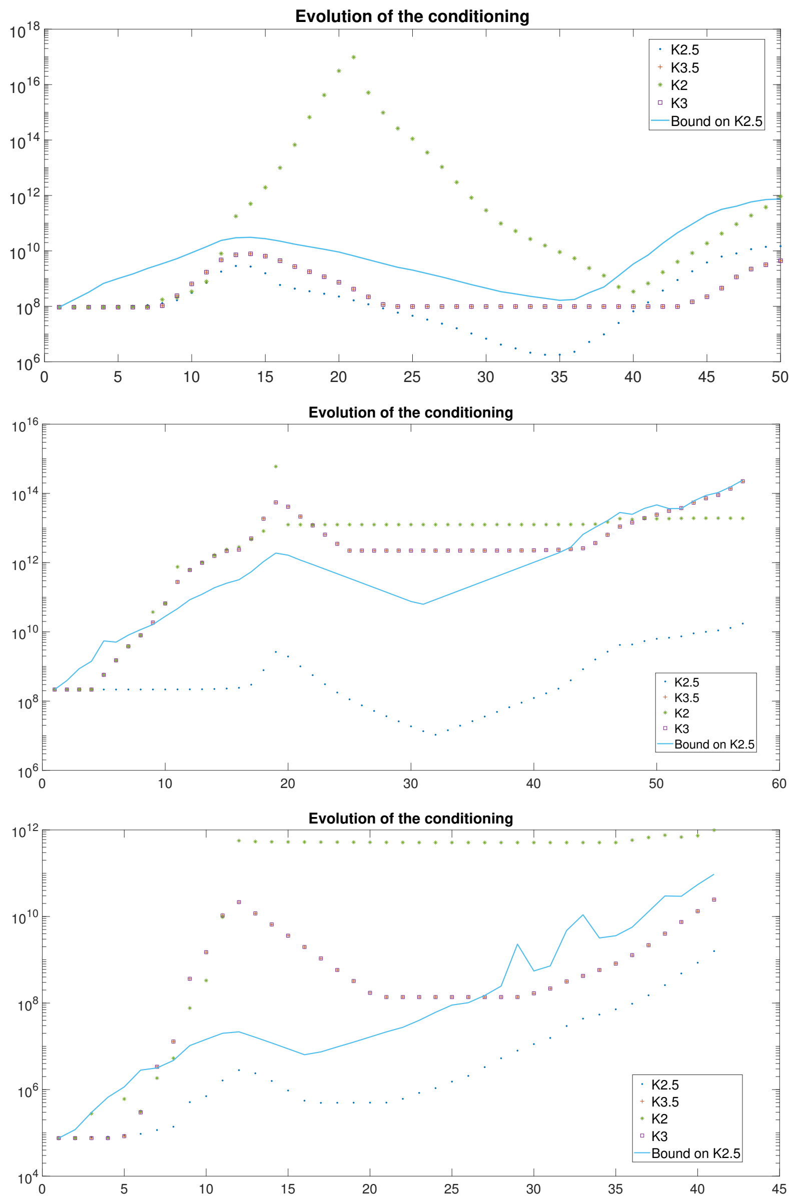

Figure 7: Top three plots: evolution of the condition number of (K3), (K3.5), (K2) and (K2.5) on problems `small009`, `nsic2` and `qbrandy`. The solid blue curve is an upper bound on $\mathrm{cond}(K_{2.5})$ obtained from the bounds of Theorem 1. All plots use $\delta_1 = \delta_2 = 10^{-2}$.

In all three cases, we see that the bound is quite loose and the condition number of $K_{2.5}$, though it may be large, is more favorable than that of the other formulations.

Figure 8 compares PDCOO performance on the entire test set with the formulations (K3), (K3.5), (K2) and (K2.5). The top plot shows a Dolan and Moré (2002) performance profile where the metric is the condition number computed at the final iteration on the entire test set. Although all solvers solved the entire test set successfully, the profiles do not attain 100% because `eigs()` failed to converge and return the extreme eigenvalues on a small proportion of problems. Nevertheless, the profile indicates that $\mathrm{cond}(K_{2.5})$ is substantially more favorable than for the other formulations, including K_3, on our test set. The bottom plot shows a time performance profile. The plot suggests that the run time of all formulations is roughly comparable, with a slight advantage in favor of K_2 and K_3. The horizontal $\log_2$ scale indicates that the run times do not differ by more than a factor of about two. Assembling $K_{2.5}$ at each iteration contributes to the cost, as it requires scaling the columns of A by $X^{1/2}$. A factorization-free implementation combined with an iterative method to compute an inexact step might overcome this expense.

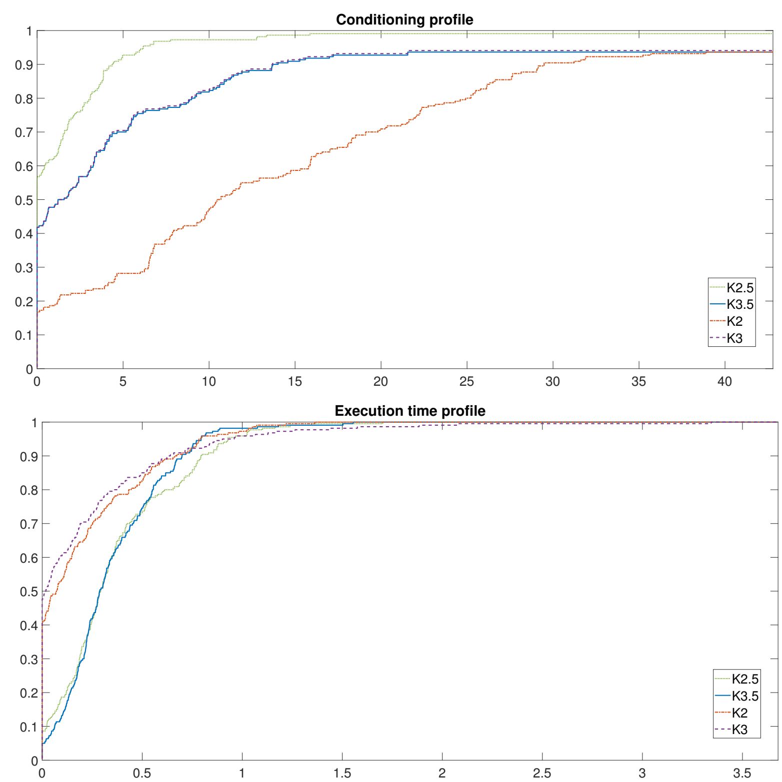

Figure 8: Top: performance profile where the metric is the condition number of the system at the final iteration. Bottom: time performance profile. Both plots use $\delta_1 = \delta_2 = 10^{-2}$.

7 Discussion

An advantage of the formulation (K2.5) is that it has the same size and storage requirements as (K2). According to the bounds of Theorems 1 and 2 and the numerical experiments of Section 6, its other advantage is that its condition number is similar to, and often substantially more favorable than, that of (K3). Moreover, the expense of forming (K2.5), and in particular of scaling A, does not have a significant effect on the solution time. We experimented here with a factorization-based implementation of PDCOO. However, it would be instructive to study (K2.5) in the context of inexact steps computed by an iterative solver such as MINRES (Paige and Saunders, 1975; Regev and Saunders, 2020).

Preconditioners will be crucial. For example, Orban (2015) and Greif, He, and Liu (2017) include an incomplete $\mathrm{LDL}^{\mathrm{T}}$ preconditioner, while di Serafino and Orban (2019) employ constraint preconditioners. We expect that Theorem 2 will provide guidelines to design further preconditioners based on estimates of the active set.

References

J. Czyzyk, S. Mehrotra, M. Wagner, and S. J. Wright. PCx: an interior-point code for linear programming. Optim. Method Softw., 11/12(1–4):397–430, 1999. DOI: 10.1080/10556789908805757.

D. di Serafino and D. Orban. Constraint-preconditioned Krylov solvers for regularized saddle-point systems. Cahier du GERAD G–2019–72, GERAD, Montréal, QC, Canada, 2019.

E. Dolan and J. Moré. Benchmarking optimization software with performance profiles. Math. Program., Series B, 91:201–213, 2002. DOI: 10.1007/s101070100263.

I. S. Duff. MA57: a Fortran code for the solution of sparse symmetric definite and indefinite systems. ACM Trans. Math. Software, 30(2):118–144, 2004. See `ldl` in MATLAB.

I. S. Duff and J. K. Reid. MA27–a set of Fortran subroutines for solving sparse symmetric sets of linear equations. Technical Report AERE R10533, AERE Harwell Laboratory, Harwell, Oxfordshire, England, 1982.

A. Forsgren. Inertia-controlling factorizations for optimization algorithms. Appl. Numer. Math., 43(1):91–107, 2002. DOI: 10.1016/S0168-9274(02)00119-8.

R. Fourer and S. Mehrotra. Solving symmetric indefinite systems in an interior-point method for linear programming. Math. Program., 62(1):15–39, 1993. DOI: 10.1007/BF01585158.

M. P. Friedlander and D. Orban. A primal-dual regularized interior-point method for convex quadratic programs. Math. Program. Comp., 4(1):71–107, 2012. DOI: 10.1007/s12532-012-0035-2.

E. M. Gertz and S. J. Wright. Object-oriented software for quadratic programming. ACM Transactions on Mathematical Software, 29(1):58–81, 2003. DOI: 10.1145/641876.641880.

N. I. M. Gould. On practical conditions for the existence and uniqueness of solutions to the general equality quadratic-programming problem. Math. Program., 32(1):90–99, 1985. DOI: 10.1007/BF01585660.

C. Greif, E. Moulding, and D. Orban. Bounds on the eigenvalues of matrices arising from interior-point methods. SIAM J. Optim., 24(1):49–83, 2014. DOI: 10.1137/120890600.

C. Greif, S. He, and P. Liu. SYM-ILDL: Incomplete LDLT factorization of symmetric indefinite and skew-symmetric matrices. ACM Trans. Math. Softw., 44(1):Article 1, 2017.

J. Korzak. Eigenvalue relations and conditions of matrices arising in linear programming. Computing, 62(1): 45–54, 1999. DOI: 10.1007/s006070050012.

B. Morini, V. Simoncini, and M. Tani. Spectral estimates for unreduced symmetric KKT systems arising from interior point methods. Numer. Linear Algebra Appl., 23(5):776–800, 2016. DOI: 10.1002/nla.2054.

E. G. Ng and B. W. Peyton. Block sparse Cholesky algorithms on advanced uniprocessor computers. SIAM J. Sci. Comput., 14(5):1034–1056, 1993. DOI: 10.1137/0914063.

D. Orban. Limited-memory LDL^T factorization of symmetric quasi-definite matrices with application to constrained optimization. Numer. Algor., 70(1):9–41, 2015. DOI: 10.1007/s11075-014-9933-x.

C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. SIAM J. Numer. Anal., 12(4):617–629, 1975. DOI: 10.1137/0712047.

PDCO. PDCO: MATLAB convex optimization software. `http://stanford.edu/group/SOL/software/pdco`, 2019.

S. Regev and M. A. Saunders. minres20: MATLAB software for MINRES and several preconditioners, 2020. URL `http://stanford.edu/group/SOL/software/minres`.

T. Rusten and R. Winther. A preconditioned iterative method for saddlepoint problems. SIAM J. Matrix Anal. Appl., 13(3):887–904, 1992. DOI: 10.1137/0613054.

M. A. Saunders. CME 338 class notes 9: PDCO: Primal-dual interior methods, 2019. URL http://stanford.edu/class/cme338/notes.html.

D. Silvester and A. Wathen. Fast iterative solution of stabilised Stokes systems. II. Using general block preconditioners. SIAM J. Numer. Anal., 31(5):1352–1367, 1994. DOI: 10.1137/0731070.

R. J. Vanderbei. Symmetric quasi-definite matrices. SIAM J. Optim., 5:100–113, 1995. DOI: https://doi.org/10.1137/0805005.

R. J. Vanderbei. LOQO: an interior point code for quadratic programming. Optim. Method Softw., 11(1–4): 451–484, 1999. DOI: 10.1080/10556789908805759.