Computational Optimization and Applications
https://doi.org/10.1007/s10589-024-00614-3

Proximal-stabilized semidefinite programming

Stefano Cipolla[1] · **Jacek Gondzio**[2]

Received: 29 February 2024 / Accepted: 3 October 2024

Abstract
A regularized version of the primal-dual Interior Point Method (IPM) for the solution of Semidefinite Programming Problems (SDPs) is presented in this paper. Leveraging on the proximal point method, a novel Proximal Stabilized Interior Point Method for SDP (PS-SDP-IPM) is introduced. The method is strongly supported by theoretical results concerning its convergence: the worst-case complexity result is established for the inner regularized infeasible inexact IPM solver. The new method demonstrates an increased robustness when dealing with problems characterized by ill-conditioning or linear dependence of the constraints without requiring any kind of pre-processing. Extensive numerical experience is reported to illustrate advantages of the proposed method when compared to the state-of-the-art solver.

Keywords Semidefinite programming · Interior point method · Proximal point methods · Primal–dual regularization

Mathematics Subject Classification 90C22 · 90C51 · 90C46 · 90C25 · 49N15

1 Introduction

Semidefinite Programming (SDP), see [40] for a detailed introduction to the area, is a powerful mathematical framework that extends linear programming to optimization problems involving positive semidefinite matrices. It finds applications in various fields such as control theory, machine learning, combinatorial optimization [2, 39], and quantum information theory [15, 25], to mention just a few. SDP provides a versatile approach for solving optimization problems where the objective function

✉ Stefano Cipolla
s.cipolla@soton.ac.uk

✉ Jacek Gondzio
j.gondzio@ed.ac.uk

[1] University of Southampton School of Mathematical Sciences, Southampton, UK

[2] The University of Edinburgh School of Mathematics and Maxwell Institute for Mathematical Sciences , Edinburgh, UK

Published online: 14 October 2024

and constraints can be expressed in terms of symmetric matrices, and the feasibility conditions involve positive semidefiniteness.

While SDP has proven to be a valuable tool in approximating and modelling challenging problems, it is also important to be aware of numerical instabilities that can arise during the solution of SDP problems. Indeed, understanding and addressing numerical challenges in SDP solvers is crucial for obtaining robust algorithms, especially when dealing with Interior Point Methods (IPM) [26] which are characterized by a severe inherent ill-conditionings of the related linear algebra problems [11, 18].

This work contributes to the understanding of the tight interplay between numerical linear algebra techniques and optimization algorithms when developing robust IPM-type SDP solvers able to deal with instances which would otherwise challenge the standard IPM-based solvers.

1.1 Motivations and problem statement

We consider a solution of a primal–dual pair of semi-definite programming problems in the form:

$$\begin{aligned} \min_{X\in\mathscr{S}^n} \ & \frac{h}{2}X\bullet X + C\bullet X \\ \text{s.t. } & \mathscr{A}X = \mathbf{b} \\ & X \succeq 0, \end{aligned} \qquad \begin{aligned} \max_{X,\ S\in\mathscr{S}^n,\ \mathbf{y}\in\mathbb{R}^m} \ & \mathbf{y}^T\mathbf{b} - \frac{h}{2}X\bullet X \\ \text{s.t. } & hX + C - \mathscr{A}^T\mathbf{y} - S = 0 \\ & S \succeq 0, \end{aligned} \tag{1}$$

where $h \in \mathbb{R}_+$, given $\{A_i\}_{i=1}^m \in \mathscr{S}^n$, $(\mathscr{A}X)_i := A_i \bullet X$ and $\mathscr{A}^T\mathbf{y} := \sum_{i=1}^m y_i A_i$ is the adjoint operator of $\mathscr{A}$ with respect to the standard inner product in $\mathscr{S}^n$ and $\mathbb{R}^m$ and where $X \succeq 0$ means that X is symmetric positive-semidefinite. It is important to note that the problem presented in (1) is slightly more general than a standard linear SDP problem as it allows the presence of the simple quadratic term $\frac{h}{2}X \bullet X$. Clearly, for $h = 0$ a *classical (linear)* SDP problem is recovered. The main motivation to consider such a formulation is that we will deal with a primal–dual *regularized* version of classical SDP problems where the presence of simple quadratic terms of the above form arises naturally, see Sect. 2.1 for more details. It is also important to note, at this stage, that (1) is a convex programming problem when the cone of positive-semidefinite matrices is considered, see [13, Sec. 4].

In classical IPM for SDP, most of the computational effort lies in the solution of a symmetric positive definite system of linear equations that determines the search direction. Indeed, after suitable reformulation of the nonlinear IPM map, see Sect. 2.1 for more details, the computation of the Newton search direction can be reduced to the solution of the following system of linear equations

$$A\Theta A^T\mathbf{v} = \mathbf{q}, \tag{2}$$

where A is a suitable matrix representation of the operator $\mathscr{A}$ and Θ is a symmetric positive definite matrix changing at every IPM iteration. See Sect. 6 more details. Whenever the matrix A is full row rank, (2) is nonsingular and the IPM search direction

is well defined. However, as IPM approaches optimality the scaling matrix Θ gets increasingly ill-conditioned. In practice, even when a direct solver is employed for the solution of (2), the quality of the search direction may be degraded by the conditioning of the matrix $A\Theta A^T$.

The inherent ill-conditioning of IPMs is a consequence of the nature of the matrix Θ which acts as a continuos approximation of the indicator function for the zero eigenvalues of the optimal solution and therefore exhibits a challenging structure of eigenvalues with a subset of them going to infinity and another subset going to zero. Additionally, the conditioning of matrix $A\Theta A^T$ is also affected by the conditioning of the matrix A itself. In particular, if the matrix A does not have full row rank, then (2) becomes singular.

As a result, even when a direct method is used for the solution of such a linear system, substantial numerical instabilities might occur. This issue usually manifests itself in inaccurate search directions leading to either a slow convergence or the lack of it. With developments presented in this paper we intend to remedy such situations and improve the robustness of SDP solvers.

1.2 Contribution and related literature

Primal–dual regularization of IPM for linear and quadratic programming [1, 33] has been introduced in order to alleviate some of the numerical difficulties related to the linear system in (2). In essence, the technique amounts to adding diagonal regularization terms to the augmented system formulation of the linear equations arising in Newton method. Despite the fact that this strategy has proven to be effective in practice, to the best of our knowledge, very few attempts have been made to deliver a better understanding of the theory behind such regularization techniques, specially in the SDP case.

In this work, we will investigate primal–dual regularized IPM based on the Proximal Point Method (PPM), see [32], for the solution of SDP problems.

The scheme here considered is exact in the sense that the PPM framework allows to state the convergence to a primal–dual solution of (1) if such an optimal solution exists, see Sect. 2.1. Every PPM sub-problem is solved using a primal–dual regularized infeasible IPM for which we prove polynomial convergence, see Sect. 7. The results presented here represent a non-trivial generalization of the results available for the quadratic programming case [8, 9] and strongly rely on some surprising properties of the Nesterov-Todd scaling direction for SDP [27, 28], see Sect. 6, and in particular Lemma 7.

An immediate benefit of interpretting primal–dual regularization of IPM as derived from the proximal point method is that the linear system (2) is modified as

$$(A(\Theta^{-1}+\rho I)^{-1}A^T+\delta I)\mathbf{v}=\mathbf{q} \text{ where } \rho>0 \text{ and } \delta>0, \tag{3}$$

and this allows to drop the requirement of A to have the full row rank. Indeed, the presence of the regularization terms ρ and δ ensures the positive-definiteness of the matrix in (3) and guarantees uniform bounds of its eigenvalues. This represents a

remarkable advantage for the linear algebra solvers used for (3). As a result, our tuned-for-robustness general purposes IPM scheme is able to successfully solve SDP instances characterized by an extreme ill-conditioning of matrices A for which state-of-the-art solvers struggle or fail. For standard well-conditioned or moderately ill-conditioned problems, the proposed regularized IPM scheme delivers comparable performance to those of other solvers known from the literature, see Sect. 9.

Despite the fact that in the last few years several works addressed the possible synergies occurring between the PPM and IPM, see e.g., [7–9, 30, 38], to the best of our knowledge, only two works address in detail the primal–dual regularized IPM in the SDP context, namely [17, 30]. In both cases, the starting point of the investigation is related to the application of the PPM to augmented Lagrangian method. The method proposed in this paper differs from these two approaches because we frame our derivation using a saddle reformulation of the problem, see Sect. 2. As a result of these theoretical developments, we are able to prove the convergence of the overall scheme without requiring the regularization parameters ρ, δ to be decreased to zero, which is a novelty w.r.t. to [30]. Instead, when our contribution is compared to [17], we are able to prove the convergence of the method without assuming that the primal–dual iterates $(X, \mathbf{y}, S)$ remain bounded, see [17, Th. 10.5]. Finally, to the best of our knowledge, the proof of polynomial convergence of the *primal–dual regularized infeasible long-step* IPM considered in this paper (see Sect. 3) is completely new in the context of SDP and generalises the techniques presented in [9] by relying strongly on particular properties of the Nesterov-Todd scaling [27] and the related primal–dual regularized IPM matrices, see Sect. 7.

1.3 Notation

The following notation will be used in the remaining part of this work.
$\mathbb{R}^n_+ := \{r \in \mathbb{R}^n \text{ s.t. } r \geq 0\}$, i.e., the set of non-negative real numbers.
$\mathcal{S}^n := \{B \in \mathbb{R}^{n\times n} \text{ s.t. } B = B^T\}$, i.e., the set of symmetric matrices.
$\mathcal{S}^n_+ := \{B \in \mathcal{S}^n \text{ s.t. } B \succeq 0\}$, i.e., the set of symmetric positive semi-definite matrices.
$\mathcal{S}^n_{++} := \{B \in \mathcal{S}^n \text{ s.t. } B \succ 0\}$, i.e., the set of symmetric positive definite matrices.
Given $X, Y \in \mathbb{R}^{n\times n}$, we define $X \bullet Y := \text{tr}(X^T Y)$ and $\|X\| := \sqrt{\text{tr}(X^T X)}$.
Given $X \in \mathbb{R}^{n\times n}$ with real spectrum, we denote by $\lambda(X)$ the set of eigenvalues of X ordered as $\lambda_1(X) \geq \lambda_2(X) \geq \cdots \geq \lambda_n(X)$.
We use the same symbol $\|\cdot\|$ to denote the Frobenius and the Euclidean norm. The context will clarify the meaning.
Given a convex set $D \subset \mathcal{S}^n \times \mathbb{R}^m$, $N_D(X, \mathbf{y})$ denotes the normal cone to $(X, \mathbf{y})$ (see [14, Sec. 2.1] or [10, Sect. 4]). Finally, given $a \in \mathbb{C}$, we denote $\Re(a)$ and $\Im(a)$ respectively, the real and imaginary part of a.

2 Computational framework

In order to use rigorously the PPM framework, let us introduce the Lagrangian function associated with (1)

$$\mathscr{L}(X,\mathbf{y}) := \frac{h}{2}X \bullet X + C \bullet X - \mathbf{y}^T(\mathscr{A}X - \mathbf{b}) + I_D(X,\mathbf{y}), \tag{4}$$

where $I_D(\cdot)$ is the indicator function of the set $D := \mathscr{S}^n_+ \times \mathbb{R}^m$.

We notice that the particular Lagrangian considered in (4) is, somehow, non-standard in the IPM literature due to the presence of the term $I_D(X,\mathbf{y})$. This choice allows us to consider variational formulation of the problem (1) (see (7)) suitable for the application of the proximal point method [32]. In the following Lemma 1, we briefly summarize the properties of the function $I_D(X,\mathbf{y})$ needed to obtain such a variational formulation.

Lemma 1 *Given $X \in \mathscr{S}^n_+$, $X = Q\,\mathrm{diag}(\boldsymbol{\lambda})Q^T$, the following properties are true:*

1.

$$[V,\mathbf{w}] \in N_D(X,\mathbf{y}) \Leftrightarrow \begin{cases} V = Q\,\mathrm{diag}(N_{\mathbb{R}^n_+}(\boldsymbol{\lambda}))Q^T \\ \mathbf{w}_i = 0 \; for\; i = 1,\ldots,m \end{cases}, \tag{5}$$

2. $\partial_X I_D(X,\mathbf{y}) = N_{\mathscr{S}^n_+}(X)$ *and* $\partial_{\mathbf{y}} I_D(X,\mathbf{y}) = \{0\}$.

In our particular case, using Lemma 1, the saddle sub-differential operator related to (4) can be written as

$$T_{\mathscr{L}}(X,\mathbf{y}) := \begin{bmatrix} \partial_X \mathscr{L}(X,\mathbf{y}) \\ \partial_{\mathbf{y}}(-\mathscr{L}(X,\mathbf{y})) \end{bmatrix} = \begin{bmatrix} hX + C - \mathscr{A}^T\mathbf{y} + \partial_X I_D(X,\mathbf{y}) \\ \mathscr{A}X - \mathbf{b} \end{bmatrix}. \tag{6}$$

The proper saddle function $\mathscr{L}(X,\mathbf{y})$ satisfies the hypothesis of [31, Cor. 2, p. 249] and hence the associated saddle operator, namely $T_{\mathscr{L}}$, is maximal monotone. In particular, the solutions $(X^*,\mathbf{y}^*) \in \mathscr{S}^n_+ \times \mathbb{R}^m$ of the problem $0 \in T_{\mathscr{L}}(X,\mathbf{y})$ are just the saddle points of $\mathscr{L}$, if any. It is important to note, at this stage, that since $T_{\mathscr{L}}$ is maximal monotone, $T^{-1}_{\mathscr{L}}(0)$ is closed and convex.

Moreover, the problem of finding $(X^*,\mathbf{y}^*)$ s.t. $0 \in T_{\mathscr{L}}(X^*,\mathbf{y}^*)$ can be alternatively written as finding a solution of the following problem

$$-\begin{bmatrix} h\mathscr{I} & -\mathscr{A}^T \\ \mathscr{A} & 0 \end{bmatrix}\begin{bmatrix} X \\ \mathbf{y} \end{bmatrix} + \begin{bmatrix} -C \\ \mathbf{b} \end{bmatrix} \in N_D(X,\mathbf{y}), \tag{7}$$

which represents the variational formulation of problem (6) (see [14, Sec. 2.1]).

Given $(X^*,\mathbf{y}^*)$ a solution of (7), we can recover a solution $(X^*,\mathbf{y}^*,S^*)$ of (1) defining $S^* := -V^*$ where $(V^*,\mathbf{w}^*) \in N_D(X^*,\mathbf{y}^*)$. Indeed, it is easy to see using a standard Lagrangian and (5), that the matrix S^* corresponds to the standard Lagrange multiplier associated with the constraints corresponding to $\mathscr{S}^n_+$.

Before concluding this section, let us stress the fact that in the remainder of this work we make the following

Assumption 1 The (convex) set of saddle points of $\mathscr{L}$ is non-empty, i.e., $T_{\mathscr{L}}^{-1}(0) \neq \emptyset$, and hence there exists at least one solution $(X^*, \mathbf{y}^*, S^*)$ of the problem (1).

2.1 Proximal point method

In this section we follow closely the developments of [8], specializing our discussion for the operator $T_{\mathscr{L}}$. The proximal point method [32] finds zeros of maximal monotone operators by recursively applying their proximal operator. In particular, starting from an initial guess $[X_0, \mathbf{y}_0]$, a sequence $[X_k, \mathbf{y}_k]$ of primal–dual pairs is generated as follows:

$$\begin{aligned} &(X_{k+1}, \mathbf{y}_{k+1}) = \mathscr{P}(X_k, \mathbf{y}_k), \text{ where } \mathscr{P} = (I + \Sigma^{-1} T_{\mathscr{L}})^{-1} \\ &\text{and } \Sigma := \text{blockdiag}(\rho I_d, \delta I_m), \text{ with } \rho > 0, \ \delta > 0. \end{aligned} \tag{8}$$

Since $\Sigma^{-1} T_{\mathscr{L}}$ is also a maximal monotone operator, $\mathscr{P}$ is single valued, non expansive and the sequence by (8) converges to a solution $[X^*, \mathbf{y}^*] \in T_{\mathscr{L}}^{-1}(0)$ [32].

Evaluating the proximal operator $\mathscr{P}$ is equivalent to finding a solution $(X, \mathbf{y})$ to the problem

$$0 \in T_{\mathscr{L}}(X, \mathbf{y}) + \Sigma((X, \mathbf{y}) - (X_k, \mathbf{y}_k)),$$

which is guaranteed to be unique. In particular, evaluating the proximal operator is equivalent to finding a solution of

$$0 \in \begin{bmatrix} hX + C - \mathscr{A}^T \mathbf{y} + \partial_X I_D(X, \mathbf{y}) + \rho(X - X_k) \\ \mathscr{A} X - \mathbf{b} + \delta(\mathbf{y} - \mathbf{y}_k) \end{bmatrix},$$

which, in turn, corresponds to solving the primal dual regularized problem in (RP-k):

$$\begin{aligned} \min_{X \in \mathbb{R}^{n \times n}, \mathbf{y} \in \mathbb{R}^m} \ & \frac{h}{2} X \bullet X + C \bullet X + \frac{\rho}{2} \|X - X_k\|^2 + \frac{\delta}{2} \|\mathbf{y}\|^2 \\ \text{s.t. } & \mathscr{A} X + \delta(\mathbf{y} - \mathbf{y}_k) = \mathbf{b} \\ & X \succeq 0. \end{aligned} \tag{RP-k}$$

Moreover, the problem (RP-k), can be written in the following variational form:

$$-\begin{bmatrix} (h + \rho)\mathscr{I} & -\mathscr{A}^T \\ \mathscr{A} & \delta \mathscr{I} \end{bmatrix} \begin{bmatrix} X \\ \mathbf{y} \end{bmatrix} + \begin{bmatrix} -C + \rho X_k \\ \mathbf{b} + \delta \mathbf{y}_k \end{bmatrix} \in N_D(X, \mathbf{y}).$$

Let us introduce the Lagrangian for problem (RP-k)

$$\begin{aligned} \mathscr{L}_k(X, \mathbf{y}, S) = & \frac{h + \rho}{2} X \bullet X + \frac{\delta}{2} \|\mathbf{y}\|^2 \\ & + (C - \rho X_k) \bullet X - \mathbf{y}^T (\mathscr{A} X + \delta(\mathbf{y} - \mathbf{y}_k) - \mathbf{b}) - S \bullet X, \end{aligned} \tag{RL}$$

where $S \in \mathscr{S}_n^+$. Using (RL), we write the KKT conditions

$$\begin{bmatrix} (h+\rho)\mathscr{I} & 0 \\ 0 & \delta\mathscr{I} \end{bmatrix} \begin{bmatrix} X \\ \mathbf{y} \end{bmatrix} + \begin{bmatrix} C - \rho X_k \\ 0 \end{bmatrix} - \begin{bmatrix} \mathscr{A}^T\mathbf{y} \\ \delta\mathbf{y} + (\mathscr{A}X + \delta(\mathbf{y} - \mathbf{y}_k) - \mathbf{b}) \end{bmatrix} - \begin{bmatrix} S \\ 0 \end{bmatrix} = 0;$$
$$SX = 0;$$
$$X, S \in \mathscr{S}_+^n.$$

We can then write the dual form of problem (RP-k) as

$$\max_{X,S\in\mathbb{R}^{n\times n},\ \mathbf{y}\in\mathbb{R}^m} \mathbf{y}^T\mathbf{b} - \frac{h}{2}X \bullet X - \frac{\rho}{2}\|X\|^2 - \frac{\delta}{2}\|\mathbf{y} - \mathbf{y}_k\|^2 \qquad \text{(RD-k)}$$
$$\text{s.t. } (h+\rho)X + (C - \rho X_k) - \mathscr{A}^T\mathbf{y} - S = 0$$
$$X, S \in \mathscr{S}_+^n,$$

where we used the fact that $\mathscr{A}X + \delta\mathbf{y} = \mathbf{b} + \delta\mathbf{y}_k$.

Definition 1 Solution of (RP-k)–(RD-k)
Using standard duality theory, we say that $(X_k^*, \mathbf{y}_k^*, S_k^*)$ is a solution of problems (RP-k)–(RD-k) if the following identities hold

$$\mathscr{A}X_k^* + \delta(\mathbf{y}_k^* - \mathbf{y}_k) - \mathbf{b} = 0$$
$$(h+\rho)X_k^* + (C - \rho X_k) - \mathscr{A}^T\mathbf{y}_k^* - S_k^* = 0$$
$$X_k^* \bullet S_k^* = 0 \text{ and } X_k^*, S_k^* \in \mathscr{S}_+^n.$$

3 Primal–Dual IPM for proximal point evaluations

In this Section we present the particular version of the IPM used to solve the primal dual problems (RP-k) - (RD-k), see Algorithm 1. This algorithm is the inner solver used at every PPM iteration and represents the building brick of our proposal, see Sect. 4 for more details.

To this aim, we introduce the following Lagrangian function which uses a logarithmic barrier to take into account the semidefinite constraints

$$\mathscr{L}_k(X, \mathbf{y}, S) = \frac{h+\rho}{2}X \bullet X + \frac{\delta}{2}\|\mathbf{y}\|^2 +$$
$$(C - \rho X_k) \bullet X - \mathbf{y}^T(\mathscr{A}X + \delta(\mathbf{y} - \mathbf{y}_k) - \mathbf{b}) - \mu \ln(\det X).$$

We write the corresponding KKT conditions

$$\nabla_X \mathscr{L}_k(X, \mathbf{y}) = (h+\rho)X - \mathscr{A}^T\mathbf{y} + C - \rho X_k - \mu X^{-1} = 0;$$
$$-\nabla_{\mathbf{y}} \mathscr{L}_k(X, \mathbf{y}) = (\mathscr{A}X + \delta(\mathbf{y} - \mathbf{y}_k) - \mathbf{b}) = 0.$$

Setting $S = \mu X^{-1}$, we can consider the following IPM map

$$F_k^{\mu,\sigma}(X, \mathbf{y}, S) := \begin{bmatrix} (h+\rho)X - \mathscr{A}^T\mathbf{y} + C - \rho X_k - S \\ \mathscr{A}X + \delta(\mathbf{y} - \mathbf{y}_k) - \mathbf{b} \\ XS - \sigma\mu I \end{bmatrix}, \tag{9}$$

where $\sigma \in (0, 1)$ is the barrier reduction parameter and $X \succ 0,\ S \succ 0$. A primal–dual interior-point method applied to problems (RP-k)–(RD-k) involves performing Newton iterations to solve a nonlinear problem of the form

$$F_k^{\mu,\sigma}(X, \mathbf{y}, S) = 0, \quad X \succ 0,\ S \succ 0.$$

In order to maintain the symmetry of the Newton updates, as in [41], we replace the last equation in (9) with $H_P(XS) - \sigma\mu I$, where

$$H_P(B) := \frac{1}{2}(PBP^{-1} + (PBP^{-1})^T)$$

for some invertible matrix P. With such a substitution, the nonlinear map $F_k^{\mu,\sigma}(X, \mathbf{y}, S)$ takes the form

$$F_k^{\mu,\sigma}(X, \mathbf{y}, S) := \begin{bmatrix} (h+\rho)X - \mathscr{A}^T\mathbf{y} + C - \rho X_k - S \\ \mathscr{A}X + \delta(\mathbf{y} - \mathbf{y}_k) - \mathbf{b} \\ H_P(XS) - \sigma\mu I \end{bmatrix}.$$

It is worth mentioning that the previous substitution maintains the symmetry of the Newton's updates without affecting the definition of the central path, as the following proposition shows:

Proposition 1 ([41, Prop. 4.1]) *Given $M \in \mathbb{R}^{n\times n}$ with real spectrum, a nonsingular $P \in \mathbb{R}^{n\times n}$ and a scalar τ,*

$$H_P(M) = \tau I \ \Leftrightarrow\ M = \tau I.$$

Corollary 1 *If $H_p(XS) = \tau I$, then $X \bullet S = n\tau$.*

Proof Using Proposition 1, we have $H_p(XS) = \tau I \Leftrightarrow XS = \tau I$ and hence $X \bullet S = X \bullet \tau X^{-1} = n\tau$. □

The proofs of the statement of Lemma 2 below follows using standard arguments.

Lemma 2 *For all non singular $P \in \mathbb{R}^{n\times n}$ and $M \in \mathbb{R}^{n\times n}$ we have*

- $\|H_P(M)\| \le \|M\|$
- $\sqrt{\sum_{j=1}^{n} |\lambda_j(M)|^2} \le \|M\|$

– *if* $M \in \mathscr{S}^n$

$$|\lambda_n(M)| \leq \|M\|.$$

We are now in the position to define the neighbourhood of the central-path considered in this work:

Definition 2 Neighbourhood of the infeasible P-scaled central path:

$$\begin{aligned}
&\mathscr{N}_k(\bar{\gamma}, \underline{\gamma}, \gamma_p, \gamma_d) := \{(X, \mathbf{y}, S) \in \mathscr{S}^n_{++} \times \mathbb{R}^m \times \mathscr{S}^n_{++} : \\
&\bar{\gamma} X \bullet S/n \geq \lambda(H_P(XS)) \geq \underline{\gamma} X \bullet S/n \ \text{ for } \ i = 1, \ldots, n; \\
&X \bullet S \geq \gamma_p \|\mathscr{A} X + \delta(\mathbf{y} - \mathbf{y}_k) - \mathbf{b}\|; \\
&X \bullet S \geq \gamma_d \|(h + \rho)X - \rho X_k - \mathscr{A}^T \mathbf{y} - S + C\|\},
\end{aligned}$$

where $\bar{\gamma} > 1 > \underline{\gamma} > 0$ and $(\gamma_p, \gamma_d) > 0$.

In the following we consider inexact Newton steps for (12) obtained from the current iterate $(X, \mathbf{y}, S)$ by solving the problem

$$\begin{bmatrix} (h+\rho)\mathscr{I} & -\mathscr{A}^T & -\mathscr{I} \\ \mathscr{A} & \delta\mathscr{I} & 0 \\ \mathscr{E} & 0 & \mathscr{F} \end{bmatrix} \begin{bmatrix} \Delta X \\ \Delta \mathbf{y} \\ \Delta S \end{bmatrix} = -F_k^{\mu,\sigma}(X, \mathbf{y}, S) + \begin{bmatrix} 0 \\ \boldsymbol{\zeta} \\ 0 \end{bmatrix}, \tag{10}$$

where $\mathscr{I}$ denotes the identity operator and

$$\begin{aligned}
\mathscr{E} \quad &:= \nabla_X H_P(XS) = P \odot P^{-T} S, \\
\mathscr{F} \quad &:= \nabla_S H_P(XS) = PX \odot P^{-T} \\
\text{where } \ (G \odot Q)K &:= \tfrac{1}{2}(GKQ^T + QK^TG^T)
\end{aligned} \tag{11}$$

Hence $\mathscr{E}\Delta X = \nabla_X H_P(XS)\Delta X = H_P(\Delta XS)$ and $\mathscr{F}\Delta S = \nabla_S H_P(XS)\Delta S = H_P(X\Delta S)$. Term $\boldsymbol{\zeta}$ in the second block of the right hand side is the inexactness introduced in the computational process (see Sect. 6 for more details).

Definition 3 For the sake of clarity, in the following we will define

$$\begin{bmatrix} \Xi_d \\ \xi_p \\ \Xi_{\mu,\sigma} \end{bmatrix} := -F_k^{\mu,\sigma}(X, \mathbf{y}, S) = \begin{bmatrix} -(h+\rho)X + \mathscr{A}^T\mathbf{y} - C + \rho X_k + S \\ \mathbf{b} - \mathscr{A}X - \delta(\mathbf{y} - \mathbf{y}_k) \\ \sigma\mu I - H_P(XS) \end{bmatrix}, \tag{12}$$

and

$$\begin{bmatrix} X_k^j(\alpha) \\ \mathbf{y}_k^j(\alpha) \\ S_k^j(\alpha) \end{bmatrix} := \begin{bmatrix} X_k^j \\ \mathbf{y}_k^j \\ S_k^j \end{bmatrix} + \begin{bmatrix} \alpha\Delta X_k^j \\ \alpha\Delta \mathbf{y}_k^j \\ \alpha\Delta S_k^j \end{bmatrix}. \tag{13}$$

In Algorithm 1 we report a prototype IPM scheme for the solution of problems (RP-k)–(RD-k). The fundamental steps involved in the algorithm are: computing the Newton direction by solving (10) with a level of inexactness that satisfies (14), see Line 2; finding the largest stepsize that guarantees to remain inside the neighbourhood and to sufficiently reduce the complementarity products, see Line 7 and Eq. (N & C); preparing the quantities to be used in the next iteration, see Lines 8-9.

In Sect. 7 we study the convergence and polynomial complexity of this algorithm. Concerning the notation, let us notice that in (13) the subscript k is related to the iteration count of Algorithm 2 (PPM) whereas the superscript j is related to the iteration of Algorithm 1 (IPM). However, if there is no danger of creating an ambiguity these indices will be omitted to keep the notation simpler.

Input: $\sigma, \bar{\sigma} \in (0,1)$ barrier reduction parameters s.t. $\sigma < \bar{\sigma}$;
$C_{\text{inexact}} \in (0,1)$ inexactness parameter s.t. $\gamma_p C_{\text{inexact}} < \sigma$;
$\varepsilon_{p,k} > 0, \varepsilon_{d,k} > 0, \varepsilon_{c,k} > 0$ optimality tolerances;
Initialization:
Iteration counter $j = 0$;
Primal–dual point $(X_k^0, \mathbf{y}_k^0, S_k^0) \in \mathcal{N}_k(\bar{\gamma}, \underline{\gamma}, \gamma_p, \gamma_d)$
Compute $\mu_k^0 := X_k^0 \bullet S_k^0/n$, $\Xi_{d,k}^0$, $\xi_{p,k}^0$ and $\Xi_{\mu_k^0,\sigma}^0$.

1 **while** `Stopping Criterion in (18) is False` **do**
2 Solve the KKT system (10) using $[\Xi_{d,k}^j, \xi_{p,k}^j, \Xi_{\mu_k^j,\sigma}^j]^T$ with $\|\zeta_k^j\| \le C_{\text{inexact}} X_k^j \bullet S_k^j$ to find $[\Delta X_k^j,\ \Delta \mathbf{y}_k^j,\ \Delta S_k^j]$;
3 Define

$$\alpha_p^{*,j} := \sup\{\alpha \in \mathbb{R} \ :\ \lambda_n(X_k^j(\alpha)) \ge 0\}$$

$$\alpha_d^{*,j} := \sup\{\alpha \in \mathbb{R} \ :\ \lambda_n(S_k^j(\alpha)) \ge 0\}$$

and $\alpha^{*,j} := \min\{\alpha_p^{*,j}, \alpha_d^{*,j}\}$;
4 **if** $(X_k(\alpha^{*,j}), \mathbf{y}_k(\alpha^{*,j}), S_k(\alpha^{*,j}))$ *is a solution of (RP-k) - (RD-k)* **then**
5 Stop
6 **end**
7 Find α_k^j as the maximum for $\alpha \in [0,1]$ s.t.

$$(X_k^j(\alpha), \mathbf{y}_k^j(\alpha), S_k^j(\alpha)) \in \mathcal{N}_k(\bar{\gamma}, \underline{\gamma}, \gamma_p, \gamma_d) \text{ and} \qquad \text{(N \& C)}$$
$$X_k^j(\alpha) \bullet S_k^j(\alpha) \le (1 - (1-\bar{\sigma})\alpha) X_k^j \bullet S_k^j;$$

8 Set $\begin{bmatrix} X_k^{j+1} \\ \mathbf{y}_k^{j+1} \\ S_k^{j+1} \end{bmatrix} = \begin{bmatrix} X_k^j \\ \mathbf{y}_k^j \\ S_k^j \end{bmatrix} + \begin{bmatrix} \alpha_k^j \Delta X_k^j \\ \alpha_k^j \Delta \mathbf{y}_k^j \\ \alpha_k^j \Delta S_k^j \end{bmatrix}$;
9 Compute the infeasibilities $\Xi_{d,k}^{j+1}$, $\xi_{p,k}^{j+1}$, $\Xi_{\mu_k^{j+1},\sigma}^{j+1}$ and barrier parameter $\mu_k^{j+1} := X_k^{j+1} \bullet S_k^{j+1}/n$;
10 Update the iteration counter: $j := j + 1$.
11 **end**

Algorithm 1: Infeasible IPM for problem (1)

Assumption 2 – As outlined in [35], the transformation (10) can be viewed as a linear transformation from $\mathscr{S}^n_{++} \times \mathbb{R}^m \times \mathscr{S}^n_{++}$ to itself and thus, under suitable conditions, it will produce Newton directions $(\Delta X, \Delta \mathbf{y}, \Delta S)$ s.t. $\Delta X,\ \Delta S \in \mathscr{S}^n$. For this reason, in the rest of our discussion, we will suppose that $\Delta X,\ \Delta S$ are symmetric.
– In the remainder of this work, we assume $\mathscr{E}$ to be non singular and $\mathscr{E}^{-T}$ will denote the inverse of the adjoint operator of $\mathscr{E}$ (see Sect. 6 for more details on this assumption).
– Concerning the measure of inexactness $\boldsymbol{\zeta}$, we assume in the following that

$$\|\boldsymbol{\zeta}\| \leq C_{\text{inexact}}\, X \bullet S, \tag{14}$$

where $C_{\text{inexact}} \in (0, 1)$.

The following results provide some extra clarification of details concerning Algorithm 1.

Lemma 3 *The following equalities hold:*

$$\mathscr{F}^T \mathscr{E}^{-T} S = X \quad , \quad \mathscr{E}^{-T} S \bullet \Xi_{\mu,\sigma} = \sigma \mu n - X \bullet S$$

and

$$S \bullet \Delta X + X \bullet \Delta S = (\sigma - 1) X \bullet S. \tag{15}$$

Proof Concerning the first two equalities in the statement, we have, by direct computation and using the definitions in (11),

$$\mathscr{E}^T I = (P^T \odot S P^{-1}) I = S \ \text{ and } \ \mathscr{F}^T I = (X P^T \odot P^{-1}) I = X \ \Rightarrow \ \mathscr{F}^T \mathscr{E}^{-T} S = X.$$

Hence

$$\mathscr{E}^{-T} S \bullet \Xi_{\mu,\sigma} = I \bullet \Xi_{\mu,\sigma} = \sigma \mu n - \text{tr}\, \frac{1}{2}(PXSP^{-1} + (PXSP^{-1})^T) = \sigma \mu n - X \bullet S.$$

The second part of the statement follows using the last equation in (12) and observing that

$$\begin{aligned} &\sigma \mu n - X \bullet S = \mathscr{E}^{-T} S \bullet \Xi_{\mu,\sigma} = \mathscr{E}^{-T} S \bullet \mathscr{E} \Delta X + \mathscr{E}^{-T} S \bullet \mathscr{F} \Delta S \Rightarrow \\ &(\sigma - 1) X \bullet S = S \bullet \Delta X + X \bullet \Delta S, \end{aligned}$$

where the implication used the properties stated at the beginning of the proof. □

As a result, using (15), we can state the following identity:

$$(X^j + \alpha \Delta X^j) \bullet (S^j + \alpha \Delta S^j) = (1 + \alpha(\sigma - 1)) X^j \bullet S^j + \alpha^2 \Delta X^j \bullet \Delta S^j. \tag{16}$$

Finally, using the last block equation in (10), we have

$$\lambda_i(\mathscr{E}\Delta X + \mathscr{F}\Delta S) = \sigma\frac{X \bullet S}{n} - \lambda_i(H_P(XS)),$$

i.e.,

$$\lambda_i(H_P(\Delta X S + X\Delta S)) = \sigma\frac{X \bullet S}{n} - \lambda_i(H_P(XS)) \quad \text{for all } i = 1, \ldots, n. \quad (17)$$

Lemma 4 *For all $\alpha \in [0, \alpha^{*,j}]$ (see Algorithm 1 for the definition of $\alpha^{*,j}$) and for all P invertible, we have*

$$\lambda_n(H_P(X^j(\alpha)S^j(\alpha))) \leq \lambda(X^j(\alpha)S^j(\alpha)) \leq \lambda_1(H_P(X^j(\alpha)S^j(\alpha))).$$

Proof The proof is a result of the following observations:

1. $\lambda(X^j(\alpha)S^j(\alpha)) = \lambda(S^j(\alpha)X^j(\alpha)) \subset \mathbb{R}$ since

$$\lambda(X^j(\alpha)S^j(\alpha)^{1/2}S^j(\alpha)^{1/2}) = \lambda(S^j(\alpha)^{1/2}X^j(\alpha)S^j(\alpha)^{1/2})$$

and $X^j(\alpha) \succeq 0$, $S^j(\alpha) \succeq 0$ by Line 3 of Algorithm 1.

2.

$$\Re(\lambda_1(X^j(\alpha)S^j(\alpha))) = \lambda_1(X^j(\alpha)S^j(\alpha)) = \lambda_1(PX^j(\alpha)S^j(\alpha)P^{-1})$$
$$\leq \lambda_1(\frac{1}{2}(PX^j(\alpha)S^j(\alpha)P^{-1} + (PX^j(\alpha)S^j(\alpha)P^{-1})^T)) = \lambda_1(H_P(X^j(\alpha)S^j(\alpha))),$$

where the inequality follows using [22, Ex. 20, page 187];

3.

$$-\lambda_n(X^j(\alpha)S^j(\alpha)) = \lambda_1(-X^j(\alpha)S^j(\alpha)) = \lambda_1(-PX^j(\alpha)S^j(\alpha)P^{-1})$$
$$\leq \lambda_1(-\frac{1}{2}(PX^j(\alpha)S^j(\alpha)P^{-1} - (PX^j(\alpha)S^j(\alpha)P^{-1})^T))$$
$$= \lambda_1(-H_P(X^j(\alpha)S^j(\alpha)))$$
$$= -\lambda_n(H_P(X^j(\alpha)S^j(\alpha))),$$

where we used that for a matrix with a real spectrum $R \in \mathbb{R}^{n\times n}$ it holds $\lambda_1(-R) = -\lambda_n(R)$ and, also in this case, the inequality [22, Ex. 20, page 187]. □

4 Proximal-stabilized interior point method for semidefinite programming (PS-SDP-IPM)

The Proximal-Stabilized Interior Point Method for Semidefinite Programming (PS-SDP-IPM) proposed and analysed in this work is presented in Algorithm 2. It uses two

nested cycles to solve problem (1). The outer loop uses an inexact PPM [24]: the current approximate solution $(X_k, \mathbf{y}_k)$ is used to regularize the original SDP problem, which is then solved using an IPM to find the next approximate solution $(X_{k+1}, \mathbf{y}_{k+1}) \approx (X^*, \mathbf{y}^*)$. And indeed, at the inner loop level, the inexact infeasible IPM presented in Sect. 3 is used to solve the PPM sub-problems, see Algorithm 1. To summarize, in the following we use three acronyms: PPM refers to the outer cycle; IPM refers to the inner cycle; PS-SDP-IPM refers to the overall procedure, combining PPM and IPM.

Input: $tol > 0, \sigma_r \in (0, 1), \tau_1 > 0$.
Initialization: Iteration counter $k = 0$; initial point $(X_0, \mathbf{y}_0)$
1 **while** *Stopping Criterion False* **do**
2 Use the IPM Algorithm 1 with starting point $(X_k^0, \mathbf{y}_k^0) = (X_k, \mathbf{y}_k)$ to find $(X_{k+1}, \mathbf{y}_{k+1})$ s.t.

$$\|r_k(X_{k+1}, \mathbf{y}_{k+1})\| < \frac{(\sigma_r)^k}{\tau_1} \min\{1, \|(X_{k+1}, \mathbf{y}_{k+1}) - (X_k, \mathbf{y}_k)\|\} \tag{18}$$

3 Update the iteration counter: $k := k + 1$.
4 **end**

Algorithm 2: PS-SDP-IPM

Let us highlight that the stopping criteria for Algorithm 2 see Line 1, are not well defined at this stage and irrelevant for the current discussion (see Sect. 9 and in particular Eq. (39), for the ones used in our numerical experiments). Instead, it is important to note that the inner solver, i.e., Algorithm 1, terminates when a required accuracy of the solution of a sub-problem is reached. The criterion involves the following *natural residual* of problems (RP-k)–(RD-k):

Definition 4 *(Natural Residual)*

$$r_k(X, \mathbf{y}) := \begin{bmatrix} X \\ \mathbf{y} \end{bmatrix} - \Pi_D\left(\begin{bmatrix} X \\ \mathbf{y} \end{bmatrix} - \begin{bmatrix} (h+\rho)X - \rho X_k + C - \mathscr{A}^T\mathbf{y} \\ \mathscr{A}X - \mathbf{b} + \delta(\mathbf{y} - \mathbf{y}_k) \end{bmatrix}\right),$$

where

$$D := \mathscr{S}_+^n \times \mathbb{R}^m$$

and where Π_D is the corresponding projection operator. Moreover, it is easy to verify that $(X_k^*, \mathbf{y}_k^*, S_k^*)$ is a solution of problems (RP-k)–(RD-k) if and only if $\mathbf{r}_k(X_k^*, \mathbf{y}_k^*) = 0$, see [14, Sec. 2A]

Remark 1 The use of the natural residual $r_k(X, \mathbf{y})$ in the stopping condition (18) of Algorithm 2 would be fully theoretically justified by the existence of a constant $\tau_1 > 0$ s.t.

$$\|(X, \mathbf{y}) - \mathscr{P}(X_k, \mathbf{y}_k)\| \leq \tau_1 \|r_k(X, \mathbf{y})\|, \tag{19}$$

i.e., that $\|r_k(X, \mathbf{y})\|$ is a global error bound for the SDP problem (RP-k). This is true in general for the QP case, see [8, Sec. 2.3] but might have theoretical limitation in

the SDP context here considered. And indeed, despite that other types of error bounds are available in the literature for the SDP case, see, e.g., [34] and reference therein, to the best of our knowledge, no clear results are given for the natural residual $r_k(X, \mathbf{y})$ here considered. On the other hand, the computational results presented in Sect. 9 will show that the use of such criterion is largely effective in practice as stopping criterion for the inner IPM solver in Algorithm 2, and, for the computational purposes of this work, we postulate that a version of (19) holds. Finally, it is important to note that when the validity of (19) is assumed, the inexact version of the PPM considered in Algorithm 2 is globally convergent, see [24].

As a final observation regarding Algorithm 2, it is important to note that every call to the inner solver, Algorithm 1, is *hot-started*, that is, choosing $(X_k^0, \mathbf{y}_k^0) = (X_k, \mathbf{y}_k)$, i.e., using the previous approximation generated by the PPM method as an initial approximation for the computation of $(X_{k+1}, \mathbf{y}_{k+1})$. This is a well understood computational practice and it is theoretically justified by the fact that, in general, the Proximal Point operator is Lipschitz. This argument was pointed out and extensively used in [8, Sec. 3.1] for the Quadratic Programming case and, for the sake of completeness, we present here the adapted reasoning. Defining η as the Lipschitz constant of the proximal operator, see [23, Theorem 4], we have that

$$\begin{aligned} &\|\mathscr{P}(X_k, \mathbf{y}_k) - (X_k, \mathbf{y}_k)\| \\ &\quad \leq \|\mathscr{P}(X_k, \mathbf{y}_k) - \mathscr{P}(X_{k-1}, \mathbf{y}_{k-1})\| + \|\mathscr{P}(X_{k-1}, \mathbf{y}_{k-1}) - (X_k, \mathbf{y}_k)\| \\ &\quad \leq \eta\|(X_k, \mathbf{y}_k) - (X_{k-1}, \mathbf{y}_{k-1})\| + \|\mathscr{P}(X_{k-1}, \mathbf{y}_{k-1}) - (X_k, \mathbf{y}_k)\|. \end{aligned} \tag{20}$$

Since we are assuming that (19) holds, i.e., that the natural residual is a valid error bound, the inexact PPM considered here is convergent (see Remark 1) and we have that

$$\|(X_k, \mathbf{y}_k) - (X_{k-1}, \mathbf{y}_{k-1})\| \to 0 \text{ and } \|\mathscr{P}(X_{k-1}, \mathbf{y}_{k-1}) - (X_k, \mathbf{y}_k)\| \to 0,$$

and hence $\|\mathscr{P}(X_k, \mathbf{y}_k) - (X_k, \mathbf{y}_k)\| \to 0$. Such observation suggests that, eventually, $\|\mathscr{P}(X_k, \mathbf{y}_k) - (X_k, \mathbf{y}_k)\|$ will become sufficiently small so that the *fast final convergence of IPM* holds immediately and the IPM converges fast. As a result, we expect that each proximal subproblem will need a non-increasing number of IPM iterations to be solved. And indeed, we observe this behaviour in practice (see all the numerical results presented in Sect. 9): typically one IPM iteration per PPM swipe is sufficient to deliver enough of the accuracy required to satisfy condition (18), and hence the convergence of the overall scheme.

5 Preliminary results for convergence

As mentioned in Sect. 4, the particular inexact version of the PPM scheme considered here has been proved to be convergent, see [24]. To prove the overall soundnesses of our proposal, we need then to prove the convergence of Algorithm 1. In this section we

start such analysis proving that the particular infeasible and inexact IPM considered here (see Algorithm 1), used as inner solver in Algorithm 2, is well defined.

Before continuing, let us observe that the PPM iteration counter k is fixed through this section and, for the sake of readability, will be used only in relation to the fixed PPM iteration $(X_k, \mathbf{y}_k, S_k)$ and not in the context of the IPM iterations $(X_k^j, \mathbf{y}_k^j, S_k^j)$.

We start from analysing the progresses made in a single Newton iteration. Using the first two blocks in (10) we obtain

$$\begin{aligned}
&(h+\rho)X^j(\alpha) - \rho X_k - \mathscr{A}^T\mathbf{y}^j(\alpha) - S^j(\alpha) + C \\
&\quad = ((h+\rho)X^j - \rho X_k - \mathscr{A}^T\mathbf{y}^j - S^j + C) + \alpha((\rho+h)\Delta X^j - \mathscr{A}^T\Delta\mathbf{y}^j - \Delta S^j) \\
&\quad = (1-\alpha)((\rho+h)X^j - \rho X_k - \mathscr{A}^T\mathbf{y}^j - S^j + C), \qquad (21) \\
&\mathscr{A}X^j(\alpha) + \delta(\mathbf{y}^j(\alpha) - \mathbf{y}_k) - \mathbf{b} \\
&\quad = (\mathscr{A}X^j + \delta(\mathbf{y}^j - \mathbf{y}_k) - \mathbf{b}) + \alpha(\mathscr{A}\Delta X^j + \delta\Delta\mathbf{y}^j) \\
&\quad = (1-\alpha)(\mathscr{A}X^j + \delta(\mathbf{y}^j - \mathbf{y}_k) - \mathbf{b}) + \alpha\zeta^j. \qquad (22)
\end{aligned}$$

We are now ready to demonstrate that Algorithm 1 is well defined.

Theorem 1 *Suppose that $(X^j, \mathbf{y}^j, S^j) \in \mathcal{N}_k(\bar{\gamma}, \underline{\gamma}, \gamma_p, \gamma_d)$ s.t. $X^j \bullet S^j > 0$ is given. If the stopping conditions at Line 4 of Algorithm 1 are not satisfied, then there exists $0 < \hat{\alpha}^j < \alpha^{*,j}$ such that conditions (N & C) are satisfied for all $\alpha \in [0, \hat{\alpha}^j]$.*

Proof In this proof we omit also the IPM iterate counter j, i.e., $(X^j, \mathbf{y}^j, S^j) \equiv (X, \mathbf{y}, S)$. Let us define the following functions:

$$\begin{aligned}
\underline{f}_n(\alpha) &:= \lambda_n(H_P((X+\alpha\Delta X)(S+\alpha\Delta S))) - \underline{\gamma}(X+\alpha\Delta X)\bullet(S+\alpha\Delta S)/n \\
\bar{f}_1(\alpha) &:= \bar{\gamma}(X+\alpha\Delta X)\bullet(S+\alpha\Delta S)/n - \lambda_1(H_P((X+\alpha\Delta X)(S+\alpha\Delta S))) \\
h(\alpha) &:= (1-(1-\bar{\sigma})\alpha)X\bullet S - (X+\alpha\Delta X)\bullet(S+\alpha\Delta S) \\
g_d(\alpha) &:= (X+\alpha\Delta X)\bullet(S+\alpha\Delta S) \\
&\quad - \gamma_d\|(h+\rho)(X+\alpha\Delta X) - \rho X_k - \mathscr{A}^T(\mathbf{y}+\alpha\Delta\mathbf{y}) - (S+\alpha\Delta S) + C\| \\
g_p(\alpha) &:= (X+\alpha\Delta X)\bullet(S+\alpha\Delta S) \\
&\quad - \gamma_p\|\mathscr{A}(X+\alpha\Delta X) + \delta(\mathbf{y}+\alpha\Delta\mathbf{y} - \mathbf{y}_k) - \mathbf{b}\|.
\end{aligned}$$

Using (21) in the expressions of $g_d(\alpha)$ we have

$$g_d(\alpha) = (X+\alpha\Delta X)\bullet(S+\alpha\Delta S) - \gamma_d(1-\alpha)\|(\rho+h)X - \rho X_k - \mathscr{A}^T\mathbf{y} - S + C\|,$$

whereas, using (22) in the expressions of $g_p(\alpha)$, we have

$$g_p(\alpha) \geq (X+\alpha\Delta X)\bullet(S+\alpha\Delta S) - \gamma_p((1-\alpha)\|\mathscr{A}X + \delta(\mathbf{y}-\mathbf{y}_k) - \mathbf{b}\| + \alpha\|\zeta\|).$$

We start from demonstrating that there exists $\hat{\alpha}^j > 0$ such that

$$\underline{f}_n(\alpha) \geq 0,\ \ \bar{f}_1(\alpha) \geq 0,\ \ h(\alpha) \geq 0,\ \ g_p(\alpha) \geq 0,\ \ g_d(\alpha) \geq 0 \text{ for all } \alpha \in [0, \hat{\alpha}^j].$$

In the following, we will use extensively the identities in (16) and (17) and the fact that $(X, \mathbf{y}, S) \in \mathcal{N}_k(\bar{\gamma}, \underline{\gamma}, \gamma_p, \gamma_d)$. Moreover, due to the symmetry of the operator $H_P(\cdot)$ in (23) and (24) we will use the inequalities (obtained using standard arguments for symmetric matrices):

$$\begin{aligned}\lambda_n(H_P((X+\alpha\Delta X)(S+\alpha\Delta S))) \geq\ & \lambda_n(H_P(XS)) + \alpha\lambda_n(H_P(\Delta XS + X\Delta S)) \\ & +\alpha^2\lambda_n(H_P(\Delta X\Delta S))\end{aligned}$$

and

$$\begin{aligned}\lambda_1(H_P((X+\alpha\Delta X)(S+\alpha\Delta S))) \leq\ & \lambda_1(H_P(XS)) + \alpha\lambda_1(H_P(\Delta XS + X\Delta S)) \\ & +\alpha^2\lambda_1(H_P(\Delta X\Delta S)).\end{aligned}$$

We have

$$\begin{aligned}\underline{f}_n(\alpha) \geq & \underbrace{(1-\alpha)(\lambda_n(H_P(XS)) - \underline{\gamma}\frac{X\bullet S}{n})}_{\geq 0} + \alpha^2(\lambda_n(H_P(\Delta X\Delta S)) \\ & - \underline{\gamma}\frac{\Delta X\bullet\Delta S}{n}) + \alpha\sigma(1-\underline{\gamma})\frac{X\bullet S}{n} \\ \geq & \alpha^2(\lambda_n(H_P(\Delta X\Delta S)) - \underline{\gamma}\frac{\Delta X\bullet\Delta S}{n}) + \alpha\sigma(1-\underline{\gamma})\frac{X\bullet S}{n}.\end{aligned} \tag{23}$$

Since $X \bullet S > 0$, using a simple continuity argument, we can infer the existence of a small enough $\alpha_{\underline{f}_n} > 0$ s.t. $\underline{f}_n(\alpha) \geq 0$ for all $\alpha \in [0, \alpha_{\underline{f}_n}]$. Reasoning analogously, we have:

$$\begin{aligned}\bar{f}_1(\alpha) \geq & \underbrace{(1-\alpha)(\bar{\gamma}\frac{X\bullet S}{n} - \lambda_1(H_P(XS)))}_{\geq 0} \\ & + \alpha^2(\bar{\gamma}\frac{\Delta X\bullet\Delta S}{n} - \lambda_1(H_P(\Delta X\Delta S))) + \alpha\sigma(\bar{\gamma}-1)\frac{X\bullet S}{n} \\ \geq & \alpha^2(\bar{\gamma}\frac{\Delta X\bullet\Delta S}{n} - \lambda_1(H_P(\Delta X\Delta S))) + \alpha\sigma(\bar{\gamma}-1)\frac{X\bullet S}{n},\end{aligned} \tag{24}$$

and hence there exists a small enough $\alpha_{\bar{f}_1} > 0$ s.t. $\bar{f}_1(\alpha) \geq 0$ for all $\alpha \in [0, \alpha_{\bar{f}_1}]$.

Concerning $h(\alpha)$ we have:

$$h(\alpha) = (\bar{\sigma} - \sigma)\alpha X \bullet S - \alpha^2 \Delta X \bullet \Delta S, \tag{25}$$

and, since $(\bar{\sigma} - \sigma)X \bullet S > 0$, there exists $\alpha_{\hat{h}} > 0$ small enough s.t. $h(\alpha) \geq 0$ for all $\alpha \in [0, \alpha_{\hat{h}}]$.

Concerning $g_d(\alpha)$ we have:

$$\begin{aligned} g_d(\alpha) = &\underbrace{(1-\alpha)\big(X \bullet S - \gamma_d \|(h+\rho)X - \rho X_k - \mathscr{A}^T\mathbf{y} - S + C\|\big)}_{\geq 0} \\ &+ \alpha\sigma X \bullet S + \alpha^2 \Delta X \bullet \Delta S \geq \alpha\sigma X \bullet S + \alpha^2 \Delta X \bullet \Delta S, \end{aligned} \tag{26}$$

and hence there exists $\alpha_{\hat{g}_d} > 0$ small enough s.t. $g_d(\alpha) \geq 0$ for all $\alpha \in [0, \alpha_{\hat{g}_d}]$.

Finally, concerning $g_p(\alpha)$ we have:

$$\begin{aligned} g_p(\alpha) \geq &\underbrace{(1-\alpha)(X \bullet S - \gamma_p \|\mathscr{A}X + \delta(\mathbf{y} - \mathbf{y}_k) - \mathbf{b}\|)}_{\geq 0} \\ &+ \alpha\sigma X \bullet S + \alpha^2 \Delta X \bullet \Delta S - \alpha\gamma_p \|\zeta\| \\ \geq &\alpha(\sigma - \gamma_p C_{\text{inexact}}) X \bullet S + \alpha^2 \Delta X \bullet \Delta S, \end{aligned} \tag{27}$$

and hence there exists $\hat{g}_p > 0$ small enough s.t. $g_p(\alpha) \geq 0$ for all $\alpha \in [0, \alpha_{\hat{g}_p}]$.

Let us define

$$\hat{\alpha}^j = \min\{\alpha_{\underline{f}_n},\ \alpha_{\bar{f}_1},\ \alpha_{\hat{h}},\ \alpha_{\hat{g}_d},\ \alpha_{\hat{g}_p},\ 1\} > 0.$$

To prove the thesis, it remains to show that $\alpha^{*,j} > \hat{\alpha}^j$, i.e., that

$$(X(\alpha), \mathbf{y}(\alpha), S(\alpha)) \in \mathscr{S}^n_{++} \times \mathbb{R}^m \times \mathscr{S}^n_{++} \text{ for all } \alpha \in [0, \hat{\alpha}^j].$$

To this aim, let us suppose, reasoning by contradiction, that $\alpha^{*,j} \leq \hat{\alpha}^j$. By definition of $\alpha^{*,j}$, we have $\lambda_n((X + \alpha^{*,j}\Delta X)(S + \alpha^{*,j}\Delta S)) = 0$. Using Lemma 4, we have $\lambda_n(H_P((X + \alpha^{*,j}\Delta X)(S + \alpha^{*,j}\Delta S))) = 0$ and hence

$$\underline{f}_n(\alpha^{*,j}) = -\underline{\gamma} X(\alpha^{*,j}) \bullet S(\alpha^{*,j})/n \geq 0 \quad \Rightarrow \quad X(\alpha^{*,j}) \bullet S(\alpha^{*,j}) = 0.$$

This last implied result used together with the definition of $g_d(\alpha^{*,j})$ and $g_p(\alpha^{*,j})$ yields

$$\begin{aligned} &\mathscr{A}X(\alpha^{*,j}) + \delta(\mathbf{y}(\alpha^{*,j}) - \mathbf{y}_k) - \mathbf{b} = 0 \\ &(h+\rho)X(\alpha^{*,j}) + (C - \rho X_k) - \mathscr{A}^T\mathbf{y}(\alpha^{*,j}) - S(\alpha^{*,j}) = 0, \end{aligned}$$

which means that $(X(\alpha^{*,j}), \mathbf{y}(\alpha^{*,j}), S(\alpha^{*,j}))$ is a solution of problems (RP-k) - (RD-k). Therefore, we have obtained a contradiction because we have assumed that Algorithm 1 did not stop at Line 4. □

Corollary 2 *The right-hand sides of the Newton systems are uniformly bounded.*

Proof As a consequence of Theorem 1, there exists of a sequence of iterates $\{(X^j, \mathbf{y}^j, S^j)\}_{j\in\mathbb{N}}$ produced by Algorithm 1 s.t.

$$(X^j, \mathbf{y}^j, S^j) \in \mathcal{N}_k(\bar{\gamma}, \underline{\gamma}, \gamma_p, \gamma_d).$$

Since, by construction $X^j \bullet S^j \leq X^0 \bullet S^0$, we have

$$\begin{aligned}
&\|\mathcal{A}X^j + \delta(\mathbf{y}^j - \mathbf{y}_k) - \mathbf{b}\| \leq X^0 \bullet S^0/\gamma_p,\\
&\|(\rho + h)X^j - \rho X_k - \mathcal{A}^T\mathbf{y}^j - S^j + C\| \leq X^0 \bullet S^0/\gamma_d.
\end{aligned}$$

Moreover, we have,

$$\begin{aligned}
&\|H_P(X^jS^j) - \sigma\mu^j I\| \leq \|H_P(X^jS^j)\| + \sigma\mu^j\|I\|\\
&= \sqrt{\mathrm{tr}(H_P(X^jS^j)^2)} + \sigma\mu^j\sqrt{n} \leq \sqrt{n}\lambda_1(H_P(X^jS^j)) + \sigma\mu^j\sqrt{n}\\
&\leq \frac{\bar{\gamma}}{\sqrt{n}}X^j \bullet S^j + \frac{\sigma}{\sqrt{n}}X^j \bullet S^j \leq \frac{\bar{\gamma} + \sigma}{\sqrt{n}}X^0 \bullet S^0.
\end{aligned}$$

□

6 Vectorization and the Nesterov–Todd (NT) direction

Through this section we will assume that $X, S \in \mathcal{S}^n_{++}$. To compute the Newton step $(\Delta X, \Delta\mathbf{y}, \Delta S)$, it is easier to express the linear systems of Eq. (10) in the standard matrix–vector form by using the *symmetrized Kronecker products*, see [35, Appendix].

Definition 5 We start defining an operator transforming symmetric matrices into vectors: if $U \in \mathcal{S}^n$, $\mathrm{svec}(U)$ is defined by

$$\begin{aligned}
&\mathrm{svec}: \ \mathcal{S}^n \to \mathbb{R}^{n(n+1)/2}\\
&\mathrm{svec}(U) := (u_{11}, \sqrt{2}u_{21}, \ldots, \sqrt{2}u_{n1}, u_{22}, \sqrt{2}u_{32}, \ldots, \sqrt{2}u_{n2}, \ldots, u_{nn})^T.
\end{aligned}$$

It is important to note that svec is an isometry, i.e., $\mathrm{svec}(U)^T\,\mathrm{svec}(V) = U \bullet V$, see [35, Appendix: Property 10]. We denote by smat the inverse map of svec.

We also define the symmetrized Kronecker product of the matrices G, K as the square matrix of order $n(n+1)/2$, the action of which on a vector $\mathrm{svec}(U)$ for $U \in \mathcal{S}^n$ is given by

$$(G \otimes_S K)\,\mathrm{svec}(U) = \frac{1}{2}\,\mathrm{svec}(KUG^T + GUK^T)$$

Finally, defining $A^T := [\text{svec}(A_1), \ldots, \text{svec}(A_m)]$, we can write the system of Eq. (10) as

$$\begin{bmatrix} (h+\rho)I & -A^T & -I \\ A & \delta I & 0 \\ E & 0 & F \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\mathbf{s} \end{bmatrix} = \begin{bmatrix} \xi_d \\ \xi_p + \boldsymbol{\zeta} \\ \xi_{\mu,\sigma} \end{bmatrix}, \tag{28}$$

where $\Delta\mathbf{x} := \text{svec}(\Delta X)$, $\Delta\mathbf{s} := \text{svec}(\Delta S)$, $\xi_d := \text{svec}(\Xi_d)$, $\xi_{\mu,\sigma} = \text{svec}(\Xi_{\mu,\sigma})$ and

$$E := P \otimes_S P^{-T} S \quad \text{and} \quad F := PX \otimes_S P^{-T}.$$

The presence of the error ζ in (28) is justified by the following computational procedure used to solve such linear system (generally know as reduction to the normal equation):

$$\begin{aligned} &\left(A(\Theta^{-1} + (h+\rho)I)^{-1}A^T + \delta I\right)\Delta\mathbf{y} \\ &\quad = \zeta + \boldsymbol{\xi}_p - A(\Theta^{-1} + (h+\rho)I)^{-1}(F^{-1}\boldsymbol{\xi}_{\mu,\sigma} + \boldsymbol{\xi}_d) \qquad (29) \\ &(\Theta^{-1} + (h+\rho)I)\Delta\mathbf{x} = A^T\Delta\mathbf{y} + \boldsymbol{\xi}_d + F^{-1}\boldsymbol{\xi}_{\mu,\sigma} \qquad (30) \\ &F\Delta\mathbf{s} = (\boldsymbol{\xi}_{\mu,\sigma} - E\Delta\mathbf{x}). \qquad (31) \end{aligned}$$

where $\Theta := E^{-1}F$. It is important to note that the form of the right hand side in (28) originates from our assumption that an inexactness ζ is present only in the computation of (29). Hence, we assume in the following, that (30) and (31) are satisfied exactly. Before continuing, let us give basic definitions used in the remainder of this work:

Definition 6 Normal Matrix:

$$S_{\rho,\delta} := A(\Theta^{-1} + (h+\rho)I)^{-1}A^T + \delta I. \tag{32}$$

Moreover we define

$$\bar{\boldsymbol{\xi}}_p := \boldsymbol{\xi}_p - A(\Theta^{-1} + (h+\rho)I)^{-1}(F^{-1}\boldsymbol{\xi}_{\mu,\sigma} + \boldsymbol{\xi}_d).$$

Using (29) and (32), we have

$$\|A\Delta\mathbf{x} + \delta\Delta\mathbf{y} - \boldsymbol{\xi}_p\| = \|S_{\rho,\delta}\Delta\mathbf{y} - \bar{\boldsymbol{\xi}}_p\| = \|\boldsymbol{\zeta}\|,$$

which then clarifies the particular form of (28).

In our convergence analysis we will focus on the Nesterov-Todd direction [27, 28]. For the sake of readability and completeness, in the remainder of this section, we will introduce and review some of the useful basic definitions and properties. All the material presented here is borrowed from [35].

Let us consider the metric-geometric mean of the matrices X and S^{-1}, defined as:

$$W := X^{\frac{1}{2}}(X^{\frac{1}{2}}SX^{\frac{1}{2}})^{-\frac{1}{2}}X^{\frac{1}{2}} = S^{-\frac{1}{2}}(S^{\frac{1}{2}}XS^{\frac{1}{2}})^{\frac{1}{2}}S^{-\frac{1}{2}} \in \mathscr{S}^n_+$$

Lemma 5 *W satisfies the following identities:*

- $W^{-1}XW^{-1} = S \quad and \quad WSW = X;$
- $W^{-1/2}XW^{-1/2} = W^{1/2}SW^{1/2} \in \mathscr{S}^n_+ \Rightarrow W^{-1/2}XSW^{1/2} = W^{1/2}SXW^{-1/2}.$

Definition 7 The Nesterov-Todd (NT) direction is obtained when $P = W^{-1/2}$.

Lemma 6 summarizes some of the properties of the NT direction/scaling needed in the analysis later.

Lemma 6 *The following statements are true:*

1. $W \quad and \quad P = W^{-1/2} \in \mathscr{S}^n_{++};$
2. E, F *are invertible;*
3. $E^{-1}F = W \otimes_S W \in \mathscr{S}^n_{++};$
4. $(F^{-1}E)^{1/2} = W^{-1/2} \otimes_S W^{-1/2}.$

Proof The proof of Item 1. follows observing that $X, S \in \mathscr{S}^n_+$. Item 2. follows observing that (see [35, Sec. 3.3])

$$E = (I \otimes_S W^{1/2}SW^{1/2})(W^{-1/2} \otimes_S W^{-1/2})$$
$$F = (W^{-1/2}XW^{-1/2} \otimes_S I)(W^{1/2} \otimes_S W^{1/2})$$

and that, in both cases, the factors are invertible (see properties of *symmetrized Kronecker products* [35, Appendix: Property 11]). To prove Item 3. let us observe that

$$E(W \otimes_S W) = (W^{-1/2} \otimes_S W^{1/2}S)(W \otimes_S W) = W^{1/2} \otimes_S W^{1/2}SW$$
$$= W^{1/2}SW \otimes_S W^{1/2} = W^{-1/2}XW^{-1/2}W^{1/2} \otimes_S W^{1/2} = F,$$

where in the third equality we used [35, Appendix: Property 2] whereas in the fourth equality we used Lemma 5. The proof of Item 4. follows from Item 3. observing that $F^{-1}E = W^{-1} \otimes_S W^{-1}$ (see [35, Appendix: Property 5]) and from [35, Appendix: Property 7]. □

Finally, we state the following Lemma 7 which will be crucial in the proof of polynomial complexity:

Lemma 7 *The following equalities hold:*

1. $H_{W^{-1/2}}(XS) = W^{-1/2}XSW^{1/2} \quad and \quad H_{W^{-1/2}}(XS)^{1/2} = W^{-1/2}XW^{-1/2} = W^{1/2}SW^{1/2}$
2. $\mathrm{svec}(\sigma\mu I - H_{W^{-1/2}}(XS)) = ((W^{-1/2}XW^{-1/2}) \otimes_S I)\,\mathrm{svec}(\sigma\mu W^{1/2}X^{-1}W^{1/2} - W^{1/2}SW^{1/2})$
3. $W^{-1/2} \otimes_S W^{-1/2} = F^{-1}(W^{-1/2}XW^{-1/2}) \otimes_S I, \quad i.e. \quad F(F^{-1}E)^{1/2} = (W^{-1/2}XW^{-1/2}) \otimes_S I$

Proof 1. Follows from observing that

$$H_{W^{-1/2}}(XS) = \frac{1}{2}(W^{-1/2}XSW^{1/2} + W^{1/2}SXW^{-1/2}) = W^{-1/2}XSW^{1/2}$$

$$= (W^{-1/2} X W^{-1/2})^2 = (W^{1/2} S W^{1/2})^2,$$

where in the second, third and fourth equalities we used Lemma 5.

2. Follows from observing that

$$\begin{aligned}
&\operatorname{svec}(\sigma\mu I - W^{-1/2} X W^{-1/2} W^{1/2} S W^{1/2}) \\
&= \operatorname{svec}(\frac{1}{2} W^{-1/2} X W^{-1/2} (\sigma\mu W^{1/2} X^{-1} W^{1/2} - W^{1/2} S W^{1/2}) \\
&\quad + \frac{1}{2} (\sigma\mu W^{1/2} X^{-1} W^{1/2} - W^{1/2} S W^{1/2}) W^{-1/2} X W^{-1/2}) \\
&= ((W^{-1/2} X W^{-1/2}) \otimes_S I) \operatorname{svec}(\sigma\mu W^{1/2} X^{-1} W^{1/2} - W^{1/2} S W^{1/2}),
\end{aligned}$$

where in the second equality we used the commutativity of the matrices

$$W^{1/2} S W^{1/2} = W^{-1/2} X W^{-1/2}.$$

3. Follows from observing that

$$\begin{aligned}
F W^{-1/2} \otimes_S W^{-1/2} &= (W^{-1/2} X W^{-1/2} \otimes_S I)(W^{1/2} \otimes_S W^{1/2})(W^{-1/2} \otimes_S W^{-1/2}) \\
&= (W^{-1/2} X W^{-1/2} \otimes_S I),
\end{aligned}$$

where, in the second equality we used [35, Appendix: Property 7]. □

7 Convergence and polynomial complexity

In this section we show that Algorithm 1 converges to an ε-accurate solution in a polynomial number of iterations. As knowledgeably pointed out by an anonymous referee, the complexity analysis of feasible IPM schemes for linear and quadratic SDP could be obtained, in principle, adapting arguments from [16]. On the other hand, we consider here an infeasible-inexact version of the interior point framework. The implant of the proof presented here generalizes the one in [9] for the SDP case and relies heavily on the fact that when primal–dual regularization is introduced, it is possible to exploit the particular form of the inverses of the Newton matrices and some properties of the Nesterov-Todd scaling, to give explicit bounds on the norm of the generated Newton directions in terms of the complementarity product μ. In the following we will omit the index j when this does not lead to ambiguities.

As it is customary in IPM literature, in this section we make the following assumption:

Assumption 3 The norm of A, i.e., $\|A\|$, is independent of n.

It is important to note that the linear system in (28) can be written in an alternative form as follows:

$$\underbrace{\begin{bmatrix} (h+\rho)I & A^T & -I \\ A & -\delta I & 0 \\ E & 0 & F \end{bmatrix}}_{=:J} \begin{bmatrix} \Delta \mathbf{x} \\ -\Delta \mathbf{y} \\ \Delta \mathbf{s} \end{bmatrix} = \begin{bmatrix} \xi_d \\ \xi_p + \boldsymbol{\zeta} \\ \xi_{\mu,\sigma} \end{bmatrix}.$$

Let us define $\Theta = E^{-1}F$. By direct computation (see also [3, Remark 1]), we have that

$$J^{-1} = \begin{bmatrix} H^{-1} & \frac{1}{\delta} H^{-1} A^T & H^{-1} F^{-1} \\ \frac{1}{\delta} A H^{-1} & \frac{1}{\delta^2} A H^{-1} A^T - \frac{1}{\delta} I & \frac{1}{\delta} A H^{-1} F^{-1} \\ -\Theta^{-1} H^{-1} & -\frac{1}{\delta} \Theta^{-1} H^{-1} A^T & (I - \Theta^{-1} H^{-1}) F^{-1} \end{bmatrix}, \tag{33}$$

where $H := (h+\rho)I + \Theta^{-1} + \frac{1}{\delta} A^T A$.

Remark 2 $H \in \mathscr{S}_+^n$ since $\Theta \in \mathscr{S}_+^n$ (see Lemma 6).

To prove polynomial complexity, we start by bounding the terms that appear in the expression (33). The next two technical results are useful in this sense.

Lemma 8 *We have that*

$$\|H^{-1}\| \in O(\|A\| \|A^T\|).$$

Proof Using the Sherman–Morrison–Woodbury formula, we get

$$\begin{aligned} H^{-1} = &((\rho+h)I + \Theta^{-1})^{-1} - \frac{1}{\delta}((h+\rho)I + \Theta^{-1})^{-1} A^T \\ &(I + \frac{1}{\delta} A((\rho+h)I + \Theta^{-1})^{-1} A^T)^{-1} A((\rho+h)I + \Theta^{-1})^{-1}. \end{aligned} \tag{34}$$

Since $\Theta \in \mathscr{S}_n^+$ (see Lemma 6), we can write $\Theta = V \Lambda V^T$ with $\Lambda \in \mathscr{S}_+^n$ diagonal and $V^T V = V V^T = I$. Then we have

$$\begin{aligned} \Theta^{-1} + (h+\rho)I &= V(\Lambda^{-1} + (h+\rho)I)V^T \Rightarrow (\Theta^{-1} + (h+\rho)I)^{-1} \\ &= V(\Lambda^{-1} + (h+\rho)I)^{-1} V^T \end{aligned}$$

and

$$(\Lambda^{-1} + (h+\rho)I)^{-1}_{ii} = \frac{\lambda_i(\Theta)}{(h+\rho)\lambda_i(\Theta) + 1} = \frac{(h+\rho)\lambda_i(\Theta)}{(h+\rho)\lambda_i(\Theta) + 1} \frac{1}{(h+\rho)} < \frac{1}{h+\rho} \tag{35}$$

and hence

$$\begin{aligned}\|H^{-1}\| &\leq \|((h+\rho)I+\Theta^{-1})^{-1}\|\big(1+\frac{1}{\delta}\|A^T\|\|A\|\|(I+\frac{1}{\delta}A((h+\rho)I\\ &\quad+\Theta^{-1})^{-1}A^T)^{-1}\|\|((h+\rho)I+\Theta^{-1})^{-1}\|\big)\\ &\leq \frac{1}{(h+\rho)}(1+\frac{1}{\delta(h+\rho)}\|A^T\|\|A\|),\end{aligned}$$

where we used that $\|(I+\frac{1}{\delta}A((h+\rho)I+\Theta^{-1})^{-1}A^T)^{-1}\| \leq 1$. □

Corollary 3 *We have that*

$$\|\Theta^{-1}H^{-1}\| \in O(\|A\|\|A^T\|)$$

Proof Using (35), we observe that

$$\lambda_i(\Theta^{-1}((h+\rho)I+\Theta^{-1})^{-1}) = \Lambda_{ii}^{-1}(\Lambda^{-1}+(h+\rho)I)_{ii}^{-1} = \frac{1}{(h+\rho)\lambda_i(\Theta)+1} \leq 1$$

and hence, using (34), we have

$$\begin{aligned}\|\Theta^{-1}H^{-1}\| &\leq \|\Theta^{-1}((h+\rho)I+\Theta^{-1})^{-1}\|\cdot\\ &\quad\big(1+\frac{1}{\delta}\|A^T\|\|A\|\|\Big(1+\frac{1}{\delta}A(\rho I+\Theta^{-1})^{-1}A^T\Big)^{-1}\|\|(\rho I+\Theta^{-1})^{-1}\|\big)\\ &\leq 1+\frac{1}{\delta(h+\rho)}\|A^T\|\|A\|,\end{aligned}$$

where we used, again, (35). □

Before continuing, let us observe that in the following we define

$$C_3 := \max\{C_1, C_2\} \text{ where } \|H^{-1}\| \leq C_1 \text{ and } \|\Theta^{-1}H^{-1}\| \leq C_2.$$

Theorem 2 *Let us assume that an iterate j of IPM Algorithm 1 belongs to the wide infeasible neighbourhood (see Definition 3). There exists a polynomial $p_1(n)$ of degree at most one s.t.*

$$\|\Delta\mathbf{x}^j\|,\ \|\Delta\mathbf{y}^j\|,\ \ \|\Delta\mathbf{s}^j\| \leq p_1(n)\sqrt{\mu^j} \text{ for all } j \in \mathbb{N}.$$

Proof Using (33), we have

$$\begin{aligned}\Delta\mathbf{x}^j &= (H^j)^{-1}\xi_d^j + \frac{1}{\delta}(H^j)^{-1}A^T(\xi_p^j+\zeta^j) + (H^j)^{-1}(F^j)^{-1}\xi_{\sigma,\mu^j}^j\\ -\Delta\mathbf{y}^j &= \frac{1}{\delta}A(H^j)^{-1}\xi_d^j + (\frac{1}{\delta^2}A(H^j)^{-1}A^T - \frac{1}{\delta}I)(\xi_p^j+\zeta^j)\\ &\quad+\frac{1}{\delta}A(H^j)^{-1}(F^j)^{-1}\xi_{\sigma,\mu^j}^j,\end{aligned}$$

and hence,

$$\begin{aligned}
\|\Delta \mathbf{x}^j\| &\leq \|(H^j)^{-1}\|\|\xi_d^j\| + \frac{1}{\delta}\|(H^j)^{-1}\|\|A^T\|\|(\xi_p^j + \zeta^j)\| \\
&\quad + \|(H^j)^{-1/2}\|\|(H^j)^{-1/2}(F^j)^{-1}\,\mathrm{svec}(\sigma\mu^j I - H_{W^{-1/2}}(X^j S^j))\| \\
&\leq \|(H^j)^{-1}\|\|\xi_d^j\| + \frac{1}{\delta}\|(H^j)^{-1}\|\|A^T\|\|(\xi_p^j + \zeta^j)\| \\
&\quad + \|(H^j)^{-1/2}\|\|(H^j)^{-1/2}(F^j)^{-1}(W^{-1/2}XW^{-1/2}\otimes_S I)\| \\
&\quad \|\,\mathrm{svec}(\sigma\mu^j W^{1/2}X^{-1}W^{1/2} - W^{1/2}SW^{1/2})\| \\
&\leq \frac{C_3 n\mu^j}{\gamma_d} + \frac{C_3\|A^T\|(n(1+\gamma_p C_{\text{inexact}}))\mu^j}{\delta\gamma_p} + C_3(\sqrt{\bar{\gamma}n} + \sigma\frac{\sqrt{n}}{\sqrt{\underline{\gamma}}})\sqrt{\mu^j} \\
&\leq p_{\Delta\mathbf{x}}(n)\sqrt{\mu^j},
\end{aligned} \tag{36}$$

where in the second inequality we used Lemma 7, whereas in the third inequality we used the fact that svec is an isometry and

$$\begin{aligned}
&\|\,\mathrm{svec}(\sigma\mu^j W^{1/2}X^{-1}W^{1/2} - W^{1/2}SW^{1/2})\| \\
&\quad \leq \|\,\mathrm{svec}(\sigma\mu^j W^{1/2}X^{-1}W^{1/2})\| + \|\,\mathrm{svec}(W^{1/2}SW^{1/2})\| \\
&\quad \leq \sigma\mu^j\|W^{1/2}X^{-1}W^{1/2}\| + \|W^{1/2}SW^{1/2}\| \\
&\quad \leq \sigma\mu^j\sqrt{\mathrm{tr}((W^{1/2}X^{-1}W^{1/2})^2)} + \sqrt{\mathrm{tr}((W^{1/2}SW^{1/2})^2)} \\
&\quad \leq \sigma\mu^j\sqrt{\mathrm{tr}(H_{W^{-1/2}}(XS)^{-1})} + \sqrt{\mathrm{tr}((W^{1/2}SW^{1/2})^2)} \\
&\quad \leq \sigma\mu^j\sqrt{\frac{n}{\lambda_n(H_{W^{-1/2}}(XS))}} + \sqrt{n\lambda_1(H_{W^{-1/2}}(XS))} \\
&\quad \leq \sigma\mu^j\sqrt{\frac{n^2}{\underline{\gamma}X\bullet S}} + \sqrt{\bar{\gamma}X\bullet S} \\
&\quad = \sigma\frac{\sqrt{\mu^j}}{\sqrt{\underline{\gamma}}}\sqrt{n} + \sqrt{\bar{\gamma}n}\sqrt{\mu^j}.
\end{aligned}$$

Finally, the fourth inequality in 36, follows observing that $\mu^j \leq \sqrt{\mu^j}\sqrt{\mu^j} \leq \sqrt{\mu^0}\sqrt{\mu^j}$ and defining $p_{\Delta\mathbf{x}}$ as a suitable polynomial of degree one.

Analogously,

$$\begin{aligned}
\|\Delta \mathbf{y}^j\| &\leq \frac{1}{\delta}\|A\|\|(H^j)^{-1}\|\|\xi_d^j\| + (\frac{1}{\delta^2}\|(H^j)^{-1}\|\|A\|\|A^T\| + \frac{1}{\delta})\|(\xi_p^j + \zeta^j)\| \\
&\quad + \frac{1}{\delta}\|A\|\|(H^j)^{-1/2}\|\|(H^j)^{-1/2}(F^j)^{-1}\,\mathrm{svec}(\sigma\mu^j I - H_P(X^j S^j))\| \\
&\leq \frac{\|A\|C_3 n\mu^j}{\delta\gamma_d} + \frac{(C_3\|A^T\|\|A\| + \delta)(n(1+\gamma_p C_{\text{inexact}}))\mu^j}{\delta^2\gamma_p}
\end{aligned}$$

$$+\frac{C_3\|A\|}{\delta}(\sqrt{\bar{\gamma}n}+\sigma\frac{\sqrt{n}}{\sqrt{\underline{\gamma}}})\sqrt{\mu^j}$$
$$\leq p_{\Delta\mathbf{y}}(n)\sqrt{\mu^j}, \tag{37}$$

where we used similar reasoning as in the previous case and where $p_{\Delta\mathbf{y}}$ is a suitable polynomial of degree one. Finally, using the fact that $\Delta\mathbf{s} = (h+\rho)\Delta\mathbf{x} - A^T\Delta\mathbf{y} - \xi_d$ and using (36), (37) and the definition of $\mathcal{N}_k(\bar{\gamma}, \underline{\gamma}, \gamma_p, \gamma_d)$, we have

$$\begin{aligned}\|\Delta\mathbf{s}\| &= (h+\rho)\|\Delta\mathbf{x}\| + \|A^T\|\|\Delta\mathbf{y}\| + \|\xi_d\| \\ &\leq (h+\rho)p_{\Delta\mathbf{x}}(n)\sqrt{\mu^j} + \|A^T\|p_{\Delta\mathbf{y}}(n)\sqrt{\mu^j} + \frac{n\mu^j}{\gamma_d} \\ &\leq p_{\Delta\mathbf{s}}(n)\sqrt{\mu^j},\end{aligned}$$

where, also in this case, $p_{\Delta\mathbf{s}}$ is a suitable polynomial of degree one. □

Corollary 4 *There exists a polynomial $p_2(n)$ of degree at most two s.t.*

$$\begin{aligned}&|\lambda_n(H_{W^{-1/2}}(\Delta X\Delta S)) - \underline{\gamma}\frac{\Delta X\bullet\Delta S}{n}| \leq p_2(n)\mu \\ &|\bar{\gamma}\frac{\Delta X\bullet\Delta S}{n} - \lambda_1(H_{W^{-1/2}}(\Delta X\Delta S))| \leq p_2(n)\mu \\ &|\Delta X\bullet\Delta S| \leq p_2(n)\mu.\end{aligned} \tag{38}$$

Proof We have

$$\begin{aligned}&|\lambda_n(H_{W^{-1/2}}(\Delta X\Delta S)) - \underline{\gamma}\frac{\Delta X\bullet\Delta S}{n}| \leq |\lambda_n(H_{W^{-1/2}}(\Delta X\Delta S))| + |\underline{\gamma}\frac{\Delta X\bullet\Delta S}{n}| \\ &\leq \|H_{W^{-1/2}}(\Delta X\Delta S)\| + \frac{\gamma}{n}\|\Delta X\|\|\Delta S\| \leq \|\Delta X\Delta S\| + \frac{\gamma}{n}\|\Delta X\|\|\Delta S\| \\ &\leq \|\Delta X\|\|\Delta S\| + \frac{\gamma}{n}\|\Delta X\|\|\Delta S\| \leq p_2(n)\mu\end{aligned}$$

where in the third inequality we used Lemma 2. The remaining part of the statement follows from similar arguments. □

We can now apply the previous results and obtain a lower bound on the step-size that depends polynomially on the size of the problem n. This is the last fundamental step before the polynomial complexity result will be stated.

Theorem 3 *There exists α^* s.t. the stepsize α^j in Algorithm 1 satisfies*

$$\alpha^j \geq \alpha^* \geq 1/q(n),$$

where $q(n)$ is a polynomial of degree at most two.

Proof Using (38) in Eqs. (23), (24), (25), (26), (27) we have:

$$\begin{aligned}
\underline{f}_n(\alpha) \geq & \alpha^2(\lambda_n(H_{W^{-1/2}}(\Delta X \Delta S)) - \underline{\gamma}\frac{\Delta X \bullet \Delta S}{n}) + \alpha\sigma(1-\underline{\gamma})\frac{X \bullet S}{n} \\
\geq & -p_2(n)\mu\alpha^2 + \alpha\sigma(1-\underline{\gamma})\mu; \\
\bar{f}_1(\alpha) \geq & \alpha^2(\bar{\gamma}\frac{\Delta X \bullet \Delta S}{n} - \lambda_1(H_{W^{-1/2}}(\Delta X \Delta S))) + \alpha\sigma(\bar{\gamma}-1)\frac{X \bullet S}{n} \\
\geq & -p_2(n)\mu\alpha^2 + \alpha\sigma(\bar{\gamma}-1)\mu; \\
h(\alpha) = & (\bar{\sigma}-\sigma)\alpha X \bullet S - \alpha^2 \Delta X \bullet \Delta S, \\
\geq & n\mu(\bar{\sigma}-\sigma)\alpha - p_2(n)\mu\alpha^2 \\
g_d(\alpha) \geq & \alpha\sigma X \bullet S + \alpha^2 \Delta X \bullet \Delta S \geq n\mu\sigma\alpha - p_2(n)\mu\alpha^2 \\
g_p(\alpha) \geq & \alpha(\sigma - \gamma_p C_{\text{inexact}})X \bullet S + \alpha^2 \Delta X \bullet \Delta S \geq n\mu(\sigma - \gamma_p C_{\text{inexact}})\alpha - p_2(n)\mu\alpha^2.
\end{aligned}$$

Hence, defining

$$\alpha^* \geq \min\{\frac{\sigma(1-\underline{\gamma})}{p_2(n)}, \frac{\sigma(\bar{\gamma}-1)}{p_2(n)}, \frac{(\bar{\sigma}-\sigma)n}{p_2(n)}, \frac{\sigma n}{p_2(n)}, \frac{(\sigma - \gamma_p C_{\text{inexact}})n}{p_2(n)}\},$$

thesis follows from the definition of α^j. □

Theorem 4 *Algorithm 1 has polynomial complexity, i.e., given $\varepsilon > 0$ there exists $K \in O(n^2 \ln(\frac{1}{\varepsilon}))$ s.t. $\mu^j \leq \varepsilon$ for all $j \geq K$.*

Proof Thesis follows from observing that

$$X^j \bullet S^j \leq (1-(1-\bar{\sigma})\alpha^*)^j X^0 \bullet S^0 \leq (1-(1-\bar{\sigma})\frac{1}{q(n)})^j X^0 \bullet S^0.$$

□

8 On the computation of the NT direction

As proved in [35, Th. 3.5], the NT direction can be computed using as P any matrix such that $P^T P = W^{-1}$ and indeed, the choice made in the previous sections, i.e., $P = W^{-1/2}$ is only one of the many possible. From the computational point of view other choices might show some advantages and, in this section, we will briefly review the results obtained in [35, Sec. 3.5] for the practical computation of W and briefly outline the procedures we used to assemble the related normal Eq. (29). To this aim, consider the following Cholesky and SVD decompositions:

$$X = LL^T, \quad S = RR^T \quad \text{and} \quad R^T L = UDV^T.$$

Defining $Q := L^{-1}X^{1/2}$, we have that Q is unitary since $X^{1/2}L^{-T}L^{-1}X^{1/2} = I$.

We have then

$$X^{1/2}SX^{1/2} = Q^T L^T RR^T LQ = Q^T VD^2V^T Q.$$

Since $Q^T V$ is orthogonal, we have

$$(X^{1/2}SX^{1/2})^{-1/2} = (Q^T V)D^{-1}(V^T Q),$$

and hence

$$W := X^{\frac{1}{2}}(X^{\frac{1}{2}}SX^{\frac{1}{2}})^{-\frac{1}{2}}X^{\frac{1}{2}} = LQ(Q^T V)D^{-1}(V^T Q)Q^T L^T = GG^T,$$

where $G := LVD^{-1/2}$. We observe that $G^{-T}G^{-1} = W^{-1}$, and hence $P := G^{-1}$ satisfies $P^T P = W^{-1}$. It is important to note also that

$$G^T SG = G^{-1}XG^{-T} = D,$$

i.e., G scales X and S to the same diagonal matrix and we have

$$H_{G^{-1}}(XS) = \frac{1}{2}(G^{-1}XSG + G^T SXG^{-T}) = D^2.$$

Finally, let us observe that

$$\begin{aligned} E^{-1}F &= W \otimes_S W = GG^T \otimes_S GG^T = (G \otimes_S G)(G^T \otimes_S G^T) \\ &= (G \otimes_S G)(G \otimes_S G)^T \end{aligned}$$

and, analogously,

$$F^{-1}E = W^{-1} \otimes_S W^{-1} = G^{-T}G^{-1} \otimes_S G^{-T}G^{-1} = (G^{-T} \otimes_S G^{-T})(G^{-1} \otimes_S G^{-1}).$$

We are ready now to highlight the computational procedure used to compute the Schur complement arising in (29), i.e.,

$$A(\Theta^{-1} + (h+\rho)I)^{-1}A^T + \delta I.$$

Let us define $W =: Q_W \Lambda_W Q_W^T$. Using [35, Appendix: Property 7]), we have then

$$\begin{aligned} \Theta^{-1} &= W^{-1} \otimes_S W^{-1} = (Q_W \Lambda_W^{-1} Q_W^T) \otimes_S (Q_W \Lambda_W^{-1} Q_W^T) \\ &= (Q_W \otimes_S Q_W)(\Lambda_W^{-1} \otimes_S \Lambda_W^{-1})(Q_W^T \otimes_S Q_W^T), \end{aligned}$$

where $(Q_W \otimes_S Q_W)(Q_W^T \otimes_S Q_W^T) = I \otimes_S I = I$. We have, hence,

$$\begin{aligned} A(\Theta^{-1} + (h+\rho)I)^{-1}A^T = A(Q_W \otimes_S Q_W)((\Lambda_W^{-1} \otimes_S \Lambda_W^{-1}) \\ +(h+\rho)I)^{-1}(Q_W^T \otimes_S Q_W^T)A^T. \end{aligned}$$

Finally, it is worth noticing that the computation of the Newton directions $\Delta\mathbf{x}$, $\Delta\mathbf{y}$, $\Delta\mathbf{s}$, see (28), requires also the computation of

$$\begin{aligned} F^{-1}\xi_{\mu,\sigma} &= F^{-1}E(E^{-1}\xi_{\mu,\sigma}) = W^{-1}\otimes_S W^{-1}\,\mathrm{svec}(\sigma\mu S^{-1} - X) \\ &= (G^{-T}\otimes_S G^{-T})(G^{-1}\otimes_S G^{-1})\,\mathrm{svec}(\sigma\mu S^{-1} - X) \\ &= (G^{-T}\otimes_S G^{-T})\,\mathrm{svec}(\sigma\mu G^{-1}S^{-1}G^{-T} - G^{-1}XG^{-T}) \\ &= (G^{-T}\otimes_S G^{-T})\,\mathrm{svec}(\sigma\mu D^{-1} - D), \end{aligned}$$

where, in the second equality, we used [35, eq. (37)].

9 Numerical results

All the computational tests discussed in this section are performed using a Dell PowerEdge R740 running Scientific Linux 7 with 4× Intel Gold 6234 3.3G, 8C/16T, 10.4GT/s, 24.75M Cache, Turbo, HT (130W) DDR4-2933, with 500GB of memory. The PS-SDP-IPM implementation closely follows the one from [8] and is written in Matlab® R2022a. All the code is available at `https://github.com/StefanoCipolla/PS-SDP-IPM`.

The proposed method is compared with SDPT3 [36, 37], which represents, to the best of our knowledge, one of the most efficient and robust IPM-based solvers for the solution of SDPs. And indeed, in our implementation we use many of the SDPT3 features such as the choice of the initial guess X_0 and S_0, the same stopping criteria and an analogous predictor-corrector technique, see [37, Sec. 2]. See, respectively, Sects. 9.1, 9.2 and 9.3 for more details.

At the same time, our implementation differs form SDPT3 in many aspects. Among the most noticeable ones: 1] we do not exploit any block-structures present in the SDP problems; 2] we do not use highly optimized `mex` routines to assemble the normal Eq. (29); 3] in order to simplify the implementation, in a similar way as [19], we use the standard Kronecker product "$\otimes$" rather than the symmetrised version "$\otimes_S$"; 4] we do not perform any form of preprocessing. In particular, our implementation does not exploit any form of block-diagonal structure in the data and does not perform any kind of analysis to detect the presence of (nearly) dependent constraints.

Before we present the details of our implementation and related numerical experiments, we should emphasize that our purpose here is to demonstrate that the proposed framework for solving (1) (based on the PPM, see Algorithm 2) is sound and it is more robust than SDPT3 when applied to solve numerically challenging problems, and that this extra stability comes without adversely affecting the overall number of interior point iterations. Indeed, our implementation aims mainly at showcasing the developed theory and demonstrating that the proposed framework may be adopted as prototype based on which other, more sophisticated and tailor-made algorithms can be designed for solving SDPs originating from challenging real-world applications.

9.1 Initial point

For the choice of the infeasible initial point, we use a similar strategy to the one implemented in SDPT3, see [36]. In particular, we set:

$$X_0 = \psi I, \quad \mathbf{y}_0 = 0, \quad \text{and} \quad S_0 = \eta I,$$

where

$$\psi = \max\{10, \sqrt{n}, n \max_{k=1,\ldots,m}\{\frac{1+|\mathbf{b}_k|}{\|A(k,:)\|}\}\}$$

and

$$\eta = \max\{10, \sqrt{n}, \max\{\|C\|, \ \|A(k:,)\|, \ k = 1, \ldots, m\}\}.$$

9.2 Stopping criteria

Algorithm 2 is stopped when

$$\max\{\texttt{relgap}, \texttt{pinfeas}, \texttt{dinfeas}\} \leq \texttt{toll}, \tag{39}$$

where `toll` is a fixed number and

- the normalized violation of the linear constraints are:

$$\texttt{pinfeas} := \frac{\|\mathscr{A}X - \mathbf{b}\|}{1 + \|\mathbf{b}\|}, \quad \texttt{dinfeas} := \frac{\|hX + C - \mathscr{A}^T\mathbf{y} - S\|}{1 + \|C\|};$$

- the normalized duality gap is:

$$\texttt{relgap} := \frac{X \bullet S}{1 + |\frac{h}{2}X \bullet X + X \bullet C| + |\mathbf{b}^T\mathbf{y} - \frac{h}{2}X \bullet X|}.$$

  It is important to note that when $h = 0$, the above stopping criterion is the same as used in SDPT3, see [36].

9.3 Predictor corrector

In our implementation, as in SDPT3 [36], we use a Predictor-Corrector strategy. In particular, when the second order correction proposed in [35, Sec. 4.3] is used, the last equation in the linear system (28) becomes

$$E\Delta\mathbf{x} + F\Delta\mathbf{s} = \text{svec}(\sigma\mu I - H_{G^{-1}}(XS) - H_{G^{-1}}(\delta X \delta S))$$

where $\delta X \delta S$ is an approximation of the search direction $\Delta X \Delta S$. The normal equations in (29), (30) and (31) then become

$$\begin{aligned}
&\big(A(\Theta^{-1} + (h+\rho)I)^{-1}A^T + \delta I\big)\Delta \mathbf{y} \\
&\quad = \boldsymbol{\xi}_p + \zeta - A(\Theta^{-1} + (h+\rho)I)^{-1}(F^{-1}\big(\boldsymbol{\xi}_{\mu,\sigma} - \operatorname{svec}(H_{G^{-1}}(\delta X \delta S))\big) + \boldsymbol{\xi}_d) \\
&(\Theta^{-1} + (h+\rho)I)\Delta \mathbf{x} = A^T \Delta \mathbf{y} + \boldsymbol{\xi}_d + F^{-1}\big(\boldsymbol{\xi}_{\mu,\sigma} - \operatorname{svec}(H_{G^{-1}}(\delta X \delta S))\big) \\
&F\Delta \mathbf{s} = \big(\boldsymbol{\xi}_{\mu,\sigma} - \operatorname{svec}(H_{G^{-1}}(\delta X \delta S))\big) - E\Delta \mathbf{x}.
\end{aligned}$$

For the sake of giving the full details of the implementation, it is important to note that in the previous formulae the following equalities hold

$$\begin{aligned}
&F^{-1}\operatorname{svec}(H_{G^{-1}}(\delta X \delta S)) = F^{-1}E(E^{-1}\operatorname{svec}(H_{G^{-1}}(\delta X \delta S))) \\
&= (G^{-T} \otimes_S G^{-T})(G^{-1} \otimes_S G^{-1})(G \otimes_S G)\operatorname{svec}(R_{NT}) \\
&= (G^{-T} \otimes_S G^{-T})\operatorname{svec}(R_{NT}),
\end{aligned}$$

where, in the third equality we used [35, eq. 45] and where

$$R_{NT} := H_{G^{-1}}(\delta X \delta S)./(\mathbf{d}\mathbf{e}^T + \mathbf{e}\mathbf{d}^T),$$

where $\mathbf{d} := \operatorname{diag}(D)$ and "./" denotes the element-wise division.

9.4 Dataset

To evaluate the performance of our Proximal-Stabilised Interior-Point Method, for the case $h = 0$, we consider representative problems from the following classes of standard test sets:

(D1) SDPLIB Collection, see [6];
(D2) DIMACS Challenge test problems [29];
(D3) de Klerk & Sotirov Library [12];
(D4) Atomic Structure Problems [42].

Moreover, in order to showcase the advantages of the introduced primal–dual regularization, we also consider ill-conditioned problems. In particular, we generate ill-conditioned feasible SDP problems adapting a routine from F. Jarre. The Matlab code displayed in Function 1, extracted from our routine for randomly generating test problems, is presented to clarify the details of the procedure.

In Lines 12–28 we generate two matrices X, S s.t. $XS = 0$ and s.t. $\frac{\lambda_1(X)}{\lambda_n(X)} = x_{cond}$ and $\frac{\lambda_1(S)}{\lambda_n(S)} = s_{cond}$ see, in particular, Lines 16 and 23. Subsequently, the common eigenvector basis of the matrices X and S is obtained using a random unitary matrix, see Line 26. Lines 31–43 are coded to generate the ill-conditioned matrix A. And indeed, our main modification of Jarre's routine is reported in Line 34 of Function 1: aiming at generating non-trivially ill-conditioned matrices A, we force the singular values of A to be logarithmically uniformly distributed in the interval $[10^{-2}, 10^{2}]$

(Moderately ill-conditioned) or $[10^{-4}, 10^{3}]$ (Highly ill-conditioned). Indeed, it is expected that the robustness of IPM-type scheme is related to the conditioning of the normal equations matrix AA^T. Understandably, the accuracy of Newton directions deteriorates when AA^T is ill-conditioned, see Eq. (2) and related comments. Finally, Lines 41–49 are used to ensure that, once X, S, A are generated as above, the remainder data for the SDP problem form a primal–dual feasible instance, i.e., $\mathbf{b}$, $\mathbf{y}$ and C are s.t. $A\mathbf{x} = \mathbf{b}$ and that $A^T\mathbf{y} - S + C = 0$. The full version of Function 1 can be obtained at the link indicated at the beginning of Sect. 9.

Function 1 Randomly Generated SDP Problems

```
1  function [A,b,C,X,S,y] = sdp_generator(n,dx,ds,x_cond,s_cond,fname)
2  % Generate a random SDP-problem with the following input:
3  % n  -- dimension of X, S
4  % dx -- number of positive eigenvalues of X with condition x_cond,
5  % ds -- number of positive eigenvalues of S with condition s_cond,
6  % fname  -- name of output file for SDPA-format, in quotes
7  % Returned data: A, b, C
8  %  X : optimal X
9  %  S : complementary to X.
10 %  y : some vector so that A^T*y+C-S=0
11 ...
12 d = n-dx-ds; % lack of strict complementarity when d > 0
13 xs       = abs(rand(n,1)); % data for x and s
14 x        = zeros(n,1);
15 tmp      = xs(1:dx); % positive part of x
16 dtmp     = max(tmp)/x_cond-min(tmp); % to be added to positive part of x
17 tmp      = tmp + dtmp; % with condition about x_cond
18 x(1:dx)  = tmp;
19 s        = zeros(n,1); % likewise for s
20 tmp      = xs(dx+1:dx+ds);
21 dtmp     = max(tmp)/s_cond-min(tmp);
22 tmp      = tmp + dtmp;
23 s(dx+1:dx+ds) = tmp;
24 X = diag(x);
25 S = diag(s);
26 a = randn(n); [U,R]=qr(a); % this gives some random unitary matrix U
27 X = U*X*U.'; X = 0.5*(X+X');
28 S = U*S*U.'; S = 0.5*(S+S');
29 p1 = dx*(dx+3)/2;
30 m  = p1;
31 A  = sprand(n*n,m,0.1,1e-5);
32 A  = full(A);
33 [UU,~,VV]  = svd(A,"econ");
34 exp1       =  3;
35 exp2       = -4;
36 yy         = logspace(exp1,exp2,m);  % Controls the conditioning of A
37 SS         = spdiags(yy.',0,m,m);
38 A          = UU*SS*VV.';
39 for i = 1:m% slightly overdetermined optimal part of x
40 tmp     = (reshape(A(:,i),n,n) + reshape(A(:,i),n,n).' )./2.0 ;
41 A(:,i) = reshape(tmp,n*n,1);
42 end
43 A    = A.';
44 a    = randn(m);
45 A    = a*A;
```

```
46      b    = A*X(:);
47      y    = randn(m,1);
48      Asy  = A.'*y;
49      Asy  = reshape(Asy,n,n);
50      Asy  = 0.5*(Asy+Asy');
51      C    = Asy+S;
52      ...
53      end
```

Finally, in order to showcase the performance of our proposal for the case $h \neq 0$ we consider:

(SD1) the Invalid Nearest Correlation Matrix (INCM) dataset [21]. Indeed, given a symmetric matrix C, the Nearest Correlation Matrix (NCM) problem [20] has form:

$$\begin{aligned} \min_{X \in \mathcal{S}^n} \ & \frac{1}{2}\|X - C\|^2 \\ \text{s.t.} \ & \operatorname{diag}(X) = \mathbf{e} \\ & X \succeq 0, \end{aligned}$$

which can be easily reformulated as an SDP problem of the form (1);

(SD2) randomly generated problems obtained by modifying Line 49 of Function 1 as

Function 2 Randomly Generated Quadratic SDP Problems

```
1       ...
2       h    = 1;
3       C    = Asy+S-h*X;
4       ...
```

As already mentioned, we do not perform any kind of preprocessing on the constraint matrix A for test problems presented above. Finally, it is important to note that, since our algorithm is a primal–dual method storing the primal-dual iterates X and S, we are unable to solve some of the largest problems in all the above mentioned test classes due to their excessive memory requirements, see e.g., [4, 5] for IPM-type approaches able to avoid the storage of both the primal-dual iterates.

9.5 Numerical results

In all the following numerical results we use `toll` $= 10^{-5}$. Concerning the regularization parameters we use $\rho = \delta = 10^{-7}$ for datasets (D1), (D2), (D3) and (D4) whereas $\rho = \delta = 10^{-4}$ for the randomly generated test problems (see Sect. 9.4 for more details about the datasets). It is important to note that, for particular problems, different values of the regularization parameters might lead to better performance in terms of IPM iterations, but we prefer to use the same parameters consistently across different datasets to show the robustness of the method presented in this paper. Concerning the parameters used in (18), we set $\sigma_r = 0.7$ and $\tau_1 = 10^{-5}$, i.e.,

$$\|r_k(X_{k+1}, \mathbf{y}_{k+1})\| < 10^5 (0.7)^k \min\{1, \|(X_{k+1}, \mathbf{y}_{k+1}) - (X_k, \mathbf{y}_k)\|\}.$$

Table 1 PS-SDP-IPM vs SDPT3. Datasets (D1), (D2), (D3)

Problem	n	m	$cond(AA^T)$	PS-SDP-IPM			SDPT3	
				PPM It	IPM It	Obj. Val	IPM It	Obj. Val
theta1	50	104	1.000000e+02	11	11	−2.299989e+01	9	−2.299980e+01
theta2	100	498	2.000000e+02	11	11	−3.287881e+01	10	−3.287908e+01
theta3	150	1106	3.000000e+02	11	11	−4.216636e+01	10	−4.216686e+01
theta4	200	1949	4.000000e+02	12	12	−5.032099e+01	10	−5.032077e+01
theta5	250	3028	5.000000e+02	12	12	−5.723210e+01	10	−5.723194e+01
theta6	300	4375	6.000000e+02	12	12	−6.347678e+01	10	−6.347647e+01
thetaG11	801	2401	5.808290e+05	18	18	−3.999991e+02	18	−3.999937e+02
thetaG51	1001	6910	3.259908e+06	29	29	−3.489993e+02	27	−3.489954e+02
truss1	13	6	2.099999e+01	10	10	9.000006e+00	7	9.000025e+00
truss2	133	58	3.059087e+02	14	14	1.233808e+02	10	1.233807e+02
truss3	31	27	3.599998e+01	11	11	9.110007e+00	11	9.110001e+00
truss4	19	12	3.599998e+01	9	9	9.010041e+00	7	9.010053e+00
truss5	331	208	5.608327e+02	15	15	1.326358e+02	13	1.326358e+02

Table 1 continued

Problem	n	m	$cond(AA^T)$	PS-SDP-IPM			SDPT3	
				PPM It	IPM It	Obj. Val	IPM It	Obj. Val
truss6	451	172	5.044709e+04	23	23	9.010035e+02	18	9.010107e+02
truss7	301	86	2.539170e+04	22	22	9.000036e+02	17	9.000053e+02
truss8	628	496	5.608327e+02	17	17	1.331146e+02	13	1.331150e+02
hamming_7_5_6	128	1793	6.400000e+01	10	10	−4.266645e+01	7	−4.266663e+01
hamming_8_3_4	256	16,129	1.280000e+02	10	10	−2.559982e+01	8	−2.559999e+01
hamming_9_8	512	2305	2.560000e+02	11	11	−2.239976e+02	8	−2.239999e+02
torusg3-8	512	512	1.000000e+00	11	11	−4.834076e+07	12	−4.834085e+07
toruspm3-8-50	512	512	1.000000e+00	10	10	−5.278015e+02	12	−5.278067e+02
Laurent_A(19,6)	668	157	2.412383e+09	27	27	2.544804e−03	28	2.444682e−03
Laurent_A(26,10)	1045	228	1.877236e+13	28	30	3.657611e−05	37	4.262735e−05
QAP_Esc16e_part_red	351	90	3.485606e+03	14	14	−2.633675e+01	22	−2.633678e+01
Schrijver_A(19,6)	632	156	1.344795e+09	20	23	1.285946e+03	24	1.279046e+03
crossing_K_7n	135	56	9.534946e+02	15	15	−4.359297e+00	17	−4.359268e+00
crossing_K_8n	620	239	2.220446e+03	26	26	− 5.859947e+00	25	−5.859958e+00
kissing_3_5_5	220	297	4.480325e+04	23	23	1.187214e+01	18	1.187212e+01
kissing_4_7_7	488	695	1.575357e+06	27	27	2.358016e+01	22	2.357986e+01

Table 2 PS-SDP-IPM vs SDPT3. Dataset (D4)

Problem	n	m	$cond(AA^T)$	PS-SDP-IPM			SDPT3	
				PPM It	IPM It	Obj. Val	IPM It	Obj. Val
BH+_2Sigma+_STO-6GN5r12g1T2	1406	948	5.349733e+02	20	20	2.698355e+01	20	2.697984e+01
BH2_2A1_STO-6GN7r14g1T2	2166	1743	6.705349e+02	26	26	3.045058e+01	22	3.043063e+01
BH_1Sigma+_STO-6GN6r12g1T2	1406	948	5.477018e+02	24	24	2.721311e+01	22	2.720655e+01
BeH_2Sigma+_STO-6GN5r12g1T2	1406	948	5.349733e+02	21	21	1.669637e+01	20	1.669374e+01
CH+_1Sigma+_STO-6GN6r12g1T2	1406	948	5.344317e+02	23	23	4.070203e+01	21	4.069334e+01
CH-_3Sigma-_STO-6GN8r12g1T2	1406	948	5.450927e+02	24	24	4.096314e+01	19	4.090723e+01
CH2_1A1_STO-6GN8r14g1T2	2166	1743	6.534970e+02	24	24	4.486575e+01	22	4.485427e+01
CH2_3B1_STO-6GN8r14g1T2	2166	1743	6.534970e+02	25	25	4.508414e+01	21	4.502968e+01
CH_2Pi_STO-6GN7r12g1T2	1406	948	5.464375e+02	26	26	4.107570e+01	21	4.102274e+01
H2O+_2B1_STO-6GN9r14g1T2	2166	1743	6.685036e+02	25	25	8.428921e+01	22	8.421711e+01
H2O_1A1_STO-6GN10r14g1T2	2166	1743	6.674951e+02	23	23	8.495406e+01	21	8.492460e+01
HF+_2Pi_STO-6GN9r12g1T2	1406	948	5.438445e+02	23	23	1.040931e+02	18	1.038861e+02
HF_1Sigma+_STO-6GN10r12g1T2_5	1406	948	5.427331e+02	21	21	1.103919e+02	18	1.047210e+02
LiH_1Sigma+_STO-6GN4r12g1T2	1406	948	5.455722e+02	20	20	8.968791e+00	20	8.967324e+00
NH+_2Pi_STO-6GN7r12g1T2	1406	948	5.464375e+02	25	25	5.793060e+01	21	5.786029e+01
NH-_2Pi_STO-6GN9r12g1T2	1406	948	5.438445e+02	24	24	5.818209e+01	19	5.805495e+01
NH2_2B1_STO-6GN9r14g1T2	2166	1743	6.685036e+02	25	25	6.303661e+01	22	6.298052e+01
NH_3Sigma-_STO-6GN8r12g1T2	1406	948	5.450927e+02	24	24	5.846579e+01	18	5.839148e+01
OH+_3Sigma-_STO-6GN8r12g1T2	1406	948	5.318634e+02	24	24	7.898288e+01	18	7.888714e+01
OH-_1Sigma+_STO-6GN10r12g1T2_5	1406	948	5.427331e+02	22	22	8.340227e+01	19	7.916859e+01
OH_2Pi_STO-6GN9r12g1T2	1406	948	5.306376e+02	24	24	7.963176e+01	18	7.946775e+01

Table 3 PS-SDP-IPM vs SDPT3. Moderately ill-conditioned randomly generated problems. *: maximum number of IPM reached without reaching the prescribed accuracy

			PS-SDP-IPM				SDPT3	
Problem	n	m	$cond(A^T A)$	PPM It	IPM It	Obj. Val	IPM It	Obj. Val
RanSDP50_t1	50	350	3.280847e+10	12	12	6.049342e+00	14	6.049331e+00
RanSDP50_t10	50	350	5.952704e+09	10	10	1.130083e+02	12	1.130079e+02
RanSDP50_t2	50	350	3.598961e+09	11	11	2.974020e+01	12	2.974014e+01
RanSDP50_t3	50	350	9.761829e+11	10	10	6.775553e+01	11	6.775552e+01
RanSDP50_t4	50	350	7.736374e+10	11	11	−1.188373e+01	13	−1.188381e+01
RanSDP50_t5	50	350	1.393905e+09	10	10	6.583660e+01	12	6.583648e+01
RanSDP50_t6	50	350	1.775191e+09	10	10	−4.681987e+01	11	−4.681998e+01
RanSDP50_t7	50	350	3.537864e+09	12	12	−9.402456e+00	13	−9.402449e+00
RanSDP50_t8	50	350	6.617739e+11	12	12	8.097101e+00	13	8.097090e+00
RanSDP50_t9	50	350	8.428466e+09	11	11	4.422563e+01	12	4.422562e+01
RanSDP100_t1	100	1325	1.615491e+10	11	11	−2.917348e+02	12	−2.917354e+02
RanSDP100_t10	100	1325	2.428105e+13	13	13	2.512144e+01	13	2.512155e+01
RanSDP100_t2	100	1325	1.897962e+11	14	14	−1.890518e+01	13	−1.890509e+01
RanSDP100_t3	100	1325	1.672407e+11	12	12	−1.737975e+02	12	−1.737983e+02
RanSDP100_t4	100	1325	3.705897e+10	12	12	−1.047905e+02	12	−1.047904e+02
RanSDP100_t5	100	1325	2.929156e+12	12	12	5.094388e+01	13	5.094319e+01
RanSDP100_t6	100	1325	7.799953e+10	12	12	1.001256e+02	12	1.001259e+02
RanSDP100_t7	100	1325	4.931540e+10	12	12	−7.071981e+01	12	−7.071956e+01
RanSDP100_t8	100	1325	1.105126e+19	12	12	−2.143418e+02	12	−2.143417e+02
RanSDP100_t9	100	1325	1.332517e+11	12	12	−1.419996e+02	12	−1.419994e+02

Table 3 continued

Problem	n	m	PS-SDP-IPM				SDPT3	
			$cond(A^T A)$	PPM It	IPM It	Obj. Val	IPM It	Obj. Val
RanSDP150_t1	150	2925	2.512840e+13	12	12	1.491274e+02	13	1.491269e+02
RanSDP150_t10	150	2925	4.541185e+12	12	12	−4.473989e+02	12	−4.473980e+02
RanSDP150_t2	150	2925	4.654924e+11	13	13	−8.921868e+01	13	−8.921835e+01
RanSDP150_t3	150	2925	4.404052e+13	12	12	−3.012449e+02	12	−3.012447e+02
RanSDP150_t4	150	2925	4.122406e+11	12	12	1.896413e+02	13	1.896405e+02
RanSDP150_t5	150	2925	2.185439e+12	13	13	1.080080e+02	13	1.080077e+02
RanSDP150_t6	150	2925	6.744148e+12	14	14	3.090196e+01	13	3.090130e+01
RanSDP150_t7	150	2925	9.811852e+10	13	13	4.313145e+01	13	4.313159e+01
RanSDP150_t8	150	2925	2.384002e+13	14	14	−4.968978e+01	13	−4.968992e+01
RanSDP150_t9	150	2925	2.123393e+13	12	12	−1.626499e+02	12	−1.626498e+02
RanSDP200_t1	200	5150	3.100688e+15	13	13	3.713280e+02	13	3.713280e+02
RanSDP200_t10	200	5150	4.812519e+13	12	12	−9.752460e+02	12	−9.752457e+02
RanSDP200_t2	200	5150	2.731686e+12	15	15	−4.425393e+01	14	−4.425394e+01
RanSDP200_t3	200	5150	4.262595e+11	13	13	−2.072399e+02	13	−2.072399e+02
RanSDP200_t4	200	5150	2.331255e+12	12	12	−6.189914e+02	12	−6.189904e+02
RanSDP200_t5	200	5150	2.077803e+14	12	12	6.107889e+02	12	6.107895e+02
RanSDP200_t6	200	5150	6.281161e+12	13	13	−1.660900e+02	13	−1.660902e+02
RanSDP200_t7	200	5150	1.921563e+12	12	12	3.689154e+02	12	3.689152e+02
RanSDP200_t8	200	5150	2.031691e+12	13	13	−3.973195e+02	13	−3.973199e+02
RanSDP200_t9	200	5150	2.803209e+12	16	16	7.043534e+01	13	7.043563e+01

Table 4 PS-SDP-IPM vs SDPT3. Highly ill-conditioned randomly generated problems

			PS-SDP-IPM				SDPT3	
Problem	n	m	$cond(A^T A)$	PPM It	IPM It	Obj. Val	IPM It	Obj. Val
RanSDP50_t1	50	350	2.240444e+18	10	10	9.044235e+02	13	9.046814e+02
RanSDP50_t10	50	350	4.692743e+18	9	9	8.335653e+03	16	8.335718e+03
RanSDP50_t2	50	350	7.224367e+18	11	11	5.849702e+02	100*	2.093354e+06
RanSDP50_t3	50	350	2.072869e+18	10	10	1.602352e+03	12	1.602492e+03
RanSDP50_t4	50	350	6.812793e+18	10	10	8.455833e+02	14	8.456725e+02
RanSDP50_t5	50	350	3.274962e+18	9	9	2.872198e+03	13	2.872409e+03
RanSDP50_t6	50	350	2.785605e+18	9	9	−2.023331e+03	47	−2.022288e+03
RanSDP50_t7	50	350	1.882660e+19	10	10	1.776746e+03	100*	2.296894e+03
RanSDP50_t8	50	350	3.668648e+18	10	10	1.488678e+03	14	1.488848e+03
RanSDP50_t9	50	350	3.875649e+18	10	10	1.413380e+03	35	1.419778e+03
RanSDP100_t1	100	1325	1.830921e+19	9	9	−1.381497e+04	100*	−1.206089e+04
RanSDP100_t10	100	1325	2.785560e+19	12	12	−1.589269e+03	100*	1.088038e+03
RanSDP100_t2	100	1325	7.159870e+19	10	10	6.913746e+03	100*	8.460635e+03
RanSDP100_t3	100	1325	6.631253e+19	11	11	2.446307e+03	100*	5.220433e+03
RanSDP100_t4	100	1325	1.588225e+20	10	10	−7.613467e+03	100*	−6.861255e+03
RanSDP100_t5	100	1325	1.330809e+19	11	11	−1.512675e+03	100*	2.174132e+04
RanSDP100_t6	100	1325	1.501702e+19	13	13	2.562124e+02	100*	1.436820e+05
RanSDP100_t7	100	1325	1.439586e+19	11	11	−2.970304e+03	100*	2.780861e+05
RanSDP100_t8	100	1325	8.080943e+18	10	10	−1.254682e+04	100*	−1.068606e+04

Table 4 continued

Problem	n	m	PS-SDP-IPM				SDPT3	
			$cond(A^T A)$	PPM It	IPM It	Obj. Val	IPM It	Obj. Val
RanSDP100_t9	100	1325	7.536949e+20	11	11	1.663361e+03	100*	2.847624e+03
RanSDP150_t1	150	2925	2.537410e+20	9	10	7.908858e+03	100*	9.472064e+03
RanSDP150_t10	150	2925	5.389532e+20	9	10	−9.166515e+03	100*	−5.661504e+03
RanSDP150_t2	150	2925	1.662807e+19	11	12	8.378111e+02	100*	−1.707747e+05
RanSDP150_t3	150	2925	7.709799e+19	10	11	−4.082931e+03	100*	−3.489327e+02
RanSDP150_t4	150	2925	9.890038e+19	9	10	8.546068e+03	100*	1.148628e+04
RanSDP150_t5	150	2925	6.422303e+19	9	10	1.024258e+04	100*	−2.487295e+05
RanSDP150_t6	150	2925	1.629480e+20	9	10	1.693371e+04	100*	2.150013e+04
RanSDP150_t7	150	2925	6.708934e+19	9	10	−1.212905e+04	100*	−8.933214e+03
RanSDP150_t8	150	2925	4.298707e+19	10	11	−4.217620e+03	100*	4.548903e+02
RanSDP150_t9	150	2925	5.127715e+19	10	11	−3.435618e+03	100*	5.056300e+03
RanSDP200_t1	200	5150	1.110233e+20	10	11	1.169101e+04	100*	1.552114e+04
RanSDP200_t10	200	5150	1.395910e+21	8	9	−5.626161e+04	100*	−5.240753e+04
RanSDP200_t2	200	5150	8.986721e+19	10	11	−7.059157e+03	100*	−6.413607e+02
RanSDP200_t3	200	5150	1.975022e+20	11	12	3.959134e+03	100*	7.972133e+03
RanSDP200_t4	200	5150	5.010876e+20	10	11	−1.259077e+04	100*	−1.015486e+04
RanSDP200_t5	200	5150	1.389150e+21	9	10	3.608357e+04	100*	4.215830e+04
RanSDP200_t6	200	5150	1.631263e+20	10	11	8.276966e+03	100*	1.102610e+04
RanSDP200_t7	200	5150	2.642089e+20	10	11	1.342202e+04	100*	1.779850e+04
RanSDP200_t8	200	5150	1.616944e+20	10	11	−1.477352e+04	100*	−1.046309e+04
RanSDP200_t9	200	5150	2.627717e+20	9	10	2.538713e+04	100*	2.921644e+04

*: maximum number of IPM reached without reaching the prescribed accuracy

Table 5 PS-SDP-IPM. Quadratic SDP for the INCM problems

Problem	n	m	$cond(AA^T)$	PS-SDP-IPM		
				PPM It	IPM It	Obj. Val
bccd16	3250	3250	1.00e+00	14	14	4.221402e+02
beyu11	12	12	1.00e+00	9	9	4.615181e−05
bhwi01	5	5	1.00e+00	6	6	1.133672e−02
cor1399	1399	1399	1.00e+00	12	12	1.424832e+02
cor3120	3120	3120	1.00e+00	12	12	7.852472e+00
fing97	7	7	1.00e+00	7	7	1.204831e−03
high02	3	3	1.00e+00	6	6	1.392826e−01
mmb13	6	6	1.00e+00	6	6	8.481240e−02
tec03	4	4	1.00e+00	7	7	7.006153e−04
tyda99r1	8	8	1.00e+00	6	6	9.863876e−01
tyda99r2	8	8	1.00e+00	6	6	3.000524e−01
tyda99r3	8	8	1.00e+00	6	6	2.259734e−01

Table 6 PS-SDP-IPM. Randomly generated ill-conditioned quadratic SDP

Problem	n	m	$cond(AA^T)$	PS-SDP-IPM		
				PPM It	IPM It	Obj. Val
QRanSDP50_t1	50	350	1.763103e+19	11	11	8.545588e+03
QRanSDP50_t10	50	350	6.756846e+19	8	9	8.350068e+04
QRanSDP50_t2	50	350	1.803095e+19	12	13	2.940637e+03
QRanSDP50_t3	50	350	1.629658e+19	11	11	1.417986e+04
QRanSDP50_t4	50	350	5.409830e+18	9	10	8.735685e+03
QRanSDP50_t5	50	350	7.504560e+19	9	10	2.561732e+04
QRanSDP50_t6	50	350	3.095987e+19	9	10	−1.742617e+04
QRanSDP50_t7	50	350	2.898951e+19	10	11	1.689618e+04
QRanSDP50_t8	50	350	2.079713e+19	9	10	1.489532e+04
QRanSDP50_t9	50	350	9.792608e+18	9	10	1.390878e+04
QRanSDP100_t1	100	1325	1.981206e+22	9	10	−1.218284e+05
QRanSDP100_t10	100	1325	7.203900e+19	12	13	−1.375624e+04
QRanSDP100_t2	100	1325	1.066704e+20	11	12	6.749437e+04
QRanSDP100_t3	100	1325	6.560960e+20	11	12	3.542514e+04
QRanSDP100_t4	100	1325	6.153031e+19	10	11	−7.398956e+04
QRanSDP100_t5	100	1325	1.234334e+20	11	12	−2.191720e+04
QRanSDP100_t6	100	1325	4.479179e+20	14	15	−2.043313e+03
QRanSDP100_t7	100	1325	4.302334e+20	12	13	−2.795626e+04
QRanSDP100_t8	100	1325	1.290912e+20	9	10	−1.176504e+05
QRanSDP100_t9	100	1325	4.718468e+19	11	12	2.302458e+04

Table 6 continued

Problem	n	m	$cond(AA^T)$	PS-SDP-IPM		
				PPM It	IPM It	Obj. Val
QRanSDP150_t1	150	2925	2.642363e+20	11	12	7.790669e+04
QRanSDP150_t10	150	2925	1.313787e+20	12	13	−7.236711e+04
QRanSDP150_t2	150	2925	3.678462e+20	11	13	1.994142e+04
QRanSDP150_t3	150	2925	2.482740e+21	12	13	−3.281267e+04
QRanSDP150_t4	150	2925	7.906075e+20	11	12	8.326368e+04
QRanSDP150_t5	150	2925	2.544165e+21	11	12	1.019804e+05
QRanSDP150_t6	150	2925	8.523443e+20	11	12	1.712954e+05
QRanSDP150_t7	150	2925	1.288282e+20	11	12	−1.326542e+05
QRanSDP150_t8	150	2925	2.149135e+20	10	12	−4.143799e+04
QRanSDP150_t9	150	2925	4.223481e+20	12	13	−2.870646e+04
QRanSDP200_t1	200	5150	4.951223e+21	10	12	1.039708e+05
QRanSDP200_t10	200	5150	1.284591e+20	9	11	−5.189842e+05
QRanSDP200_t2	200	5150	4.932134e+21	11	13	−6.275959e+04
QRanSDP200_t3	200	5150	9.532349e+20	11	13	4.476215e+04
QRanSDP200_t4	200	5150	4.560851e+20	10	12	−7.484321e+04
QRanSDP200_t5	200	5150	1.100976e+21	9	11	3.427802e+05
QRanSDP200_t6	200	5150	3.313887e+20	10	12	8.971560e+04
QRanSDP200_t7	200	5150	1.857750e+20	10	12	1.203676e+05
QRanSDP200_t8	200	5150	1.129208e+21	10	12	−1.447227e+05
QRanSDP200_t9	200	5150	4.793225e+20	9	11	2.611253e+05

Indeed, in our computational experience, we have found that driving the IPM solver to a high accuracy in the initial PPM iterations is unnecessary and, usually, leads to a significant deterioration of the overall performance.

Finally, the linear systems which are solved for the predictor and corrector direction terms and involve the normal Eq. (32), see Sect. 9.3 for more details, use a Cholesky factorization computed resorting on Matlab's `chol` function.

In Tables 1 and 2 we report the comparison of PS-SDP-IPM vs SDPT3 for datasets (D1), (D2),(D3) and (D4), respectively. In Table 4 we report similar comparison but for randomly generated ill-conditioned SDP problems. The analysis of results presented in Tables 1 and 2 confirms that the number of IPM iterations recorded for PS-SDP-IPM is comparable to the number of IPM iterations of SDPT3 on well and moderately conditioned problems. On the other hand, already from the results there reported, it is possible to trace a very interesting feature of PS-SDP-IPM: for ill-conditioned problems, the number of IPM iterations needed to reach the required accuracy is, in general, smaller than the number of IPM iterations needed by SDPT3, see, e.g., the problems `Laurent_A(26,10), Schrijver_A(19,6)`. The existence of such trend is further confirmed by the results presented in Tables 3 and 4 concerning, respectively, Moderately and Highly ill-conditioned randomly generated SDP examples. Whilst SDPT3 is able to reach the desired accuracy for all the problems in Table 3, it fails

to solve the majority of the problems presented in Table 4. In contrast, PS-SDP-IPM solves all the problems from these test sets.

Finally, in Tables 5 and 6 we report the results of preliminary experiments which illustrate the behaviour of PS-SDP-IPM on quadratic SDP problems when $h = 1$. The regularized IPM proposed in this paper is able to solve all the problems in the selected dataset including the consistently ill-conditioned ones, see Table 6. Despite the fact that the experiments reported here are merely meant to showcase the effectiveness of our proposal in solving the particular quadratic SDP problems, it is important to note that, analogously to what was observed for the pure SDP case (cfr. the conditioning and the total number of IPM iterations in Tables 3 and 4), also in this case the total number of IPM steps needed to achieve the prescribed accuracy depends very little on the conditioning of $A^T A$ (cfr. Tables 5 and 6). As a concluding comment regarding all the experiments presented in this section, we would like to mention the fact that accordingly to what was theoretically outlined at the end of Sect. 4, one or two IPM iterations per PPM step are always sufficient to deliver enough of the accuracy prescribed by the inexactness criterion used in Line 2 of Algorithm 2 (cfr. the columns `PPM It.` and `IPM It.` in all the tables in this section).

10 Conclusions

In this work a tuned-for-robustness version of interior point solver for semidefinite programming problems has been introduced. The computational framework of the proposed algorithm relies on the proximal point method which uses a primal–dual *regularized* interior point method as proximal solver. The proposed technique is sound: a polynomial convergence guarantee has been established for the inner inexact regularized interior point solver when the Nesterov–Todd scaling is employed. Extensive computational experience confirms that the proposed method is significantly more robust than the state-of-the-art solver when applied to challenging SDPs with ill-conditioned constraint matrices.

Acknowledgements The authors are grateful to Dr. Filippo Zanetti for the insightful discussions regarding the efficient computation of the normal equations arising in interior point methods when solving SDP problems.

Data availability statement The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest This study does not have any conflicts to disclose.

References

1. Altman, A., Gondzio, J.: Regularized symmetric indefinite systems in interior point methods for linear and quadratic optimization. Optim. Methods Softw. **11/12**(1–4), 275–302 (1999). https://doi.org/10.1080/10556789908805754
2. Anjos, M.F., Lasserre, J.B.: Handbook on Semidefinite, Conic and Polynomial Optimization, vol. 166. Springer, Berlin (2011)
3. Armand, P., Benoist, J.: Uniform boundedness of the inverse of a Jacobian matrix arising in regularized interior-point methods. Math. Program. **137**(1–2 Ser. A), 587–592 (2013). https://doi.org/10.1007/s10107-011-0498-3
4. Bellavia, S., Gondzio, J., Porcelli, M.: An inexact dual logarithmic barrier method for solving sparse semidefinite programs. Math. Program. **178**(1–2), 109–143 (2019). https://doi.org/10.1007/s10107-018-1281-5
5. Bellavia, S., Gondzio, J., Porcelli, M.: A relaxed interior point method for low-rank semidefinite programming problems with applications to matrix completion. J. Sci. Comput. **89**(2), Paper No. 46, 36 (2021). https://doi.org/10.1007/s10915-021-01654-1
6. Borchers, B.: SDPLIB 1.2, library of semidefinite programming test problems. Interior point methods. pp. 683–690 (1999). https://doi.org/10.1080/10556789908805769
7. Chouzenoux, E., Corbineau, M.C., Pesquet, J.C.: A proximal interior point algorithm with applications to image processing. J. Math. Imaging Vision **62**(6–7), 919–940 (2020). https://doi.org/10.1007/s10851-019-00916-w
8. Cipolla, S., Gondzio, J.: Proximal stabilized interior point methods and low-frequency-update preconditioning techniques. J. Optim. Theory Appl. **197**(3), 1061–1103 (2023)
9. Cipolla, S., Gondzio, J., Zanetti, F.: A regularized interior point method for sparse optimal transport on graphs. Eur. J. Oper. Res. 319: 413-426 (2024)
10. Clason, C., Valkonen, T.: Introduction to nonsmooth analysis and optimization. arXiv (2020). https://doi.org/10.48550/ARXIV.2001.00216
11. D'Apuzzo, M., De Simone, V., di Serafino, D.: On mutual impact of numerical linear algebra and large-scale optimization with focus on interior point methods. Comput. Optim. Appl. **45**(2), 283–310 (2010)
12. de Klerk, E., Sotirov, R.: A new library of structured semidefinite programming instances. Optim. Methods Softw. **24**(6), 959–971 (2009). https://doi.org/10.1080/10556780902896608
13. Dehghani, A., Goffin, J.L., Orban, D.: A primal-dual regularized interior-point method for semidefinite programming. Optim. Methods Softw. **32**(1), 193–219 (2017)
14. Dontchev, A.L., Rockafellar, R.T.: Implicit Functions and Solution Mappings. Springer Monographs in Mathematics. Springer, Dordrecht (2009). https://doi.org/10.1007/978-0-387-87821-8
15. Fawzi, H., Fawzi, O.: Efficient optimization of the quantum relative entropy. J. Phys. A: Math. Theor. **51**(15), 154003 (2018)
16. Faybusovich, L.: Euclidean Jordan algebras and interior-point algorithms. Positivity **1**(4), 331–357 (1997). https://doi.org/10.1023/A:1009701824047
17. Friedlander, M.P., Orban, D.: A primal–dual regularized interior-point method for convex quadratic programs. Math. Program. Comput. **4**(1), 71–107 (2012). https://doi.org/10.1007/s12532-012-0035-2
18. Gondzio, J.: Interior point methods 25 years later. Eur. J. Oper. Res. **218**(3), 587–601 (2012). https://doi.org/10.1016/j.ejor.2011.09.017
19. Habibi, S., Kočvara, M., Stingl, M.: Loraine: an interior-point solver for low-rank semidefinite programming (2023) (preprint hal-04076509)
20. Higham, N.J.: Computing the nearest correlation matrix—a problem from finance. IMA J. Numer. Anal. **22**(3), 329–343 (2002). https://doi.org/10.1093/imanum/22.3.329
21. Higham, N.J., Strabić, N.: Bounds for the distance to the nearest correlation matrix. SIAM J. Matrix Anal. Appl. **37**(3), 1088–1102 (2016). https://doi.org/10.1137/15M1052007

22. Horn, R.A., Johnson, C.R.: Topics in Matrix Analysis. Cambridge University Press, Cambridge (1991). https://doi.org/10.1017/CBO9780511840371
23. Liao-McPherson, D., Kolmanovsky, I.: FBstab: a proximally stabilized semismooth algorithm for convex quadratic programming. Automatica J. IFAC (2020). https://doi.org/10.1016/j.automatica.2019.108801
24. Luque, F.J.: Asymptotic convergence analysis of the proximal point algorithm. SIAM J. Control Optim. **22**(2), 277–293 (1984). https://doi.org/10.1137/0322019
25. Mironowicz, P.: Semi-definite programming and quantum information (2023) arXiv:2306.16560
26. Nesterov, Y., Nemirovskii, A.: Interior-point Polynomial Algorithms in Convex Programming. SIAM, Philadelphia (1994)
27. Nesterov, Y.E., Todd, M.J.: Self-scaled barriers and interior-point methods for convex programming. Math. Oper. Res. **22**(1), 1–42 (1997). https://doi.org/10.1287/moor.22.1.1
28. Nesterov, Y.E., Todd, M.J.: Primal-dual interior-point methods for self-scaled cones. SIAM J. Optim. **8**(2), 324–364 (1998). https://doi.org/10.1137/S1052623495290209
29. Pataki, G., and Schmieta, S.: The DIMACS library of mixed semidefinite-quadratic-linear programs. http://dimacs.rutgers.edu/Challenges/Seventh/Instances
30. Pougkakiotis, S., Gondzio, J.: An interior point-proximal method of multipliers for linear positive semi-definite programming. J. Optim. Theory Appl. **192**(1), 97–129 (2022). https://doi.org/10.1007/s10957-021-01954-4
31. Rockafellar, R.T.: Monotone operators associated with saddle-functions and minimax problems. In: Nonlinear Functional Analysis (Proceedings of Symposia in Pure Mathematics, Vol. XVIII, Part 1, Chicago, Ill., 1968), pp. 241–250. American Mathematical Society, Providence, R.I. (1970)
32. Rockafellar, R.T.: Monotone operators and the proximal point algorithm. SIAM J. Control Optim. **14**(5), 877–898 (1976). https://doi.org/10.1137/0314056
33. Saunders, M., Tomlin, J.A.: Solving regularized linear programs using barrier methods and KKT systems. Technical Report SOL 96-4, Systems Optimization Laboratory, Dept. of Operations Research, Stanford University, Stanford, CA 94305, USA (1996)
34. Sremac, S., Woerdeman, H.J., Wolkowicz, H.: Error bounds and singularity degree in semidefinite programming. SIAM J. Optim. **31**(1), 812–836 (2021). https://doi.org/10.1137/19M1289327
35. Todd, M.J., Toh, K.C., Tütüncü, R.H.: On the Nesterov–Todd direction in semidefinite programming. SIAM J. Optim. **8**(3), 769–796 (1998). https://doi.org/10.1137/S105262349630060X
36. Toh, K.C., Todd, M.J., Tütüncü, R.H.: SDPT3—a MATLAB software package for semidefinite programming, version 1.3. Interior point methods. pp. 545–581 (1999). https://doi.org/10.1080/10556789908805762
37. Tütüncü, R.H., Toh, K.C., Todd, M.J.: Solving semidefinite-quadratic-linear programs using SDPT3. Computational semidefinite and second order cone programming: the state of the art, pp. 189–217 (2003). https://doi.org/10.1007/s10107-002-0347-5
38. Valkonen, T.: Interior-proximal primal–dual methods. Appl. Anal. Optim. **3**(1), 1–28 (2019)
39. Vandenberghe, L., Boyd, S.: Applications of semidefinite programming. Appl. Numer. Math. **29**(3), 283–299 (1999)
40. Wolkowicz, H., Saigal, R., Vandenberghe, L.: Handbook of Semidefinite Programming: Theory, Algorithms, and Applications, vol. 27. Springer, Berlin (2012)
41. Zhang, Y.: On extending some primal–dual interior-point algorithms from linear programming to semidefinite programming. SIAM J. Optim. **8**(2), 365–386 (1998). https://doi.org/10.1137/S1052623495296115
42. Zhao, Z., Braams, B.J., Fukuda, M., Overton, M.L., Percus, J.K.: The reduced density matrix method for electronic structure calculations and the role of three-index representability conditions. J. Chem. Phys. **120**(5), 2095–2104 (2004)