Department of Policy and Planning Sciences

Discussion Paper Series

No.1378

A New Extension of Chubanov's Method to Symmetric Cones

by

Shin-ichi KANOH and Akiko YOSHISE

October 2021

Revised November 2022

Revised April 2023

UNIVERSITY OF TSUKUBA
Tsukuba, Ibaraki 305-8573
JAPAN

A New Extension of Chubanov's Method to Symmetric Cones

Shin-ichi Kanoh*and Akiko Yoshise†

October 2021
Revised November 2022
Revised April 2023

Abstract

We propose a new variant of Chubanov's method for solving the feasibility problem over the symmetric cone by extending Roos's method (2018) of solving the feasibility problem over the non-negative orthant. The proposed method considers a feasibility problem associated with a norm induced by the maximum eigenvalue of an element and uses a rescaling focusing on the upper bound for the sum of eigenvalues of any feasible solution to the problem. Its computational bound is (i) equivalent to that of Roos's original method (2018) and superior to that of Lourenço et al.'s method (2019) when the symmetric cone is the nonnegative orthant, (ii) superior to that of Lourenço et al.'s method (2019) when the symmetric cone is a Cartesian product of second-order cones, (iii) equivalent to that of Lourenço et al.'s method (2019) when the symmetric cone is the simple positive semidefinite cone, and (iv) superior to that of Pena and Soheili's method (2017) for any simple symmetric cones under the feasibility assumption of the problem imposed in Pena and Soheili's method (2017). We also conduct numerical experiments that compare the performance of our method with existing methods by generating strongly (but ill-conditioned) feasible instances. For any of these instances, the proposed method is rather more efficient than the existing methods in terms of accuracy and execution time.

1 Introduction

Recently, Chubanov [2, 3] proposed a new polynomial-time algorithm for solving the problem (P(A)),

$$\mathrm{P}(A) \quad \text{find} \quad x > \mathbf{0} \quad \text{s.t.} \quad Ax = \mathbf{0},$$

where A is a given integer (or rational) matrix and $\mathrm{rank}(A) = m$ and $\mathbf{0}$ is an n-dimensional vector of 0s. The method explores the feasibility of the following problem $\mathrm{P}_{S_1}(A)$, which is equivalent to P(A) and given by

$$\mathrm{P}_{S_1}(A) \quad \text{find} \quad x > \mathbf{0} \quad \text{s.t.} \quad Ax = \mathbf{0}, \quad \mathbf{0} < x \leq \mathbf{1},$$

where $\mathbf{1}$ is an n-dimensional vector of 1s. Chubanov's method consists of two ingredients, the "main algorithm" and the "basic procedure." Note that the alternative problem D(A) of P(A) is given by

$$\mathrm{D}(A) \quad \text{find} \quad y \geq \mathbf{0} \quad \text{s.t.} \quad y \in \mathrm{range}A^\top, \quad y \neq \mathbf{0},$$

---

*Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki 305-8573, and Japan Society for the Promotion of Science, 5-3-1 Kojimachi, Chiyoda-ku, Tokyo 102-0083, Japan. email: s2130104@s.tsukuba.ac.jp

†Corresponding author. Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan. email: yoshise@sk.tsukuba.ac.jp

where range$A^\top$ is the orthogonal complement of ker$A$. The structure of the method is as follows: In the outer iteration, the main algorithm calls the basic procedure, which generates a sequence in $\mathbb{R}^n$ using projection to the set ker$A := \{x \in \mathbb{R}^n \mid Ax = \mathbf{0}\}$. The basic procedure terminates in a finite number of iterations returning one of the following: (i). a solution of problem P(A), (ii). a solution of problem D(A), or (iii). a cut of P(A), i.e., an index $j \in \{1, 2, \ldots, n\}$ for which $0 < x_j \leq \frac{1}{2}$ holds for any feasible solution of problem P$_{S_1}(A)$.

If result (i) or (ii) is returned by the basic procedure, then the feasibility of problem P(A) can be determined and the main procedure stops. If result (iii) is returned, then the main procedure generates a diagonal matrix $D \in \mathbb{R}^{n \times n}$ with a (j, j) element of 2 and all other diagonal elements of 1 and rescales the matrix as AD^{-1}. Then, it calls the basic procedure with the rescaled matrix. Chubanov's method checks the feasibility of P(A) by repeating the above procedures.

For problem P(A), [15] proposed a tighter cut criterion of the basic procedure than the one used in [3]. [3] used the fact that $x_j \leq \frac{\sqrt{n}\|z\|_2}{y_j}$ holds for any $y \in \mathbb{R}^n$ satisfying $\sum_{i=1}^n y_i = 1, y \geq 0$ and $y \notin \mathrm{range}A^T$, $z \in \mathbb{R}^n$ obtained by projecting this y onto ker$A$, and any feasible solution $x \in \mathbb{R}^n$ of P$_{S_1}(A)$, and the basic procedure is terminated if a y is found for which $\frac{\sqrt{n}\|z\|_2}{y_j} \leq \frac{1}{2}$ holds for some index j. On the other hand, [15] showed that for $v = y - z$, $x_j \leq \min(1, \mathbf{1}^T [-v/v_j]^+) \leq \frac{\sqrt{n}\|z\|_2}{y_j}$ holds if $v_j \neq 0$, where $[-v/v_j]^+$ is the projection of $-v/v_j \in \mathbb{R}^n$ onto the nonnegative orthant and $\mathbf{1}$ is the vector of ones, and the basic procedure is terminated if a y is found for which $\mathbf{1}^T [-v/v_j]^+ \leq \frac{1}{2}$ holds.

Chubanov's method has also been extended to include the feasibility problem over the second-order cone [9] and the symmetric cone [13, 10]. The feasibility problem over the symmetric cone is of the form,

$$\mathrm{P}(\mathcal{A}) \qquad \text{find} \quad x \in \mathrm{int}\, \mathcal{K} \qquad \text{s.t.} \quad \mathcal{A}(x) = \mathbf{0},$$

where $\mathcal{A}$ is a linear operator, $\mathcal{K}$ is a symmetric cone, and int $\mathcal{K}$ is the interior of the set $\mathcal{K}$. As proposed in [13, 10], for problem P($\mathcal{A}$), the structure of Chubanov's method remains the same; i.e., the main algorithm calls the basic procedure, and the basic procedure returns one of the following in a finite number of iterations: (i). a solution of problem P($\mathcal{A}$), or (ii). a solution of the alternative problem of problem P($\mathcal{A}$), or (iii). a recommendation of scaling problem P($\mathcal{A}$). If result (i) or (ii) is returned by the basic procedure, then the feasibility of the problem P($\mathcal{A}$) can be determined and the main procedure stops. If result (iii) is returned, the problem is scaled appropriately and the basic procedure is called again.

It should be noted that the purpose of rescaling differs between [10] and [13]. In [13], the authors devised a rescaling method so that the following value becomes larger:

$$\delta(\ker \mathcal{A} \cap \mathcal{K}) := \max_x \left\{ \det(x) \mid x \in \ker \mathcal{A} \cap \mathcal{K}, \|x\|_J^2 = r \right\},$$

where $\ker \mathcal{A} := \{x \mid \mathcal{A}(x) = \mathbf{0}\}$ and $\|x\|_J$ is the norm induced by the inner product $\langle x, y \rangle = \mathrm{trace}(x \circ y)$ defined in section 2.3. They proposed four updating schemes to be employed in the basic procedure and conducted numerical experiments to compare the effect of these schemes when the symmetric cone is the nonnegative orthant [14].

In [10], the authors assumed that the symmetric cone $\mathcal{K}$ is given by the Cartesian product of p simple symmetric cones $\mathcal{K}_1, \ldots, \mathcal{K}_p$, and they investigated the feasibility of the problem (P$_{S_{1,\infty}}(\mathcal{A})$),

$$\mathrm{P}_{S_{1,\infty}}(\mathcal{A}) \qquad \text{find} \quad x \in \mathrm{int}\, \mathcal{K} \qquad \text{s.t.} \quad \mathcal{A}(x) = \mathbf{0}, \quad \|x\|_{1,\infty} \leq 1,$$

where for each $x = (x_1, \ldots, x_p) \in \mathcal{K} = \mathcal{K}_1 \times \cdots \mathcal{K}_p$, $\|x\|_{1,\infty}$ is defined by $\|x\|_{1,\infty} := \max\{\|x_1\|_1, \ldots, \|x_p\|_1\}$, and $\|x\|_1$ is the sum of the absolute values of all eigenvalues of x. Note that if $p = 1$, then problem P$_{S_{1,\infty}}(\mathcal{A})$ turns out to be P$_{S_1}(\mathcal{A})$, which is equivalent to P($\mathcal{A}$):

$$\mathrm{P}_{S_1}(\mathcal{A}) \quad \text{find} \quad x \in \mathrm{int}\, \mathcal{K} \quad \text{s.t.} \quad \mathcal{A}(x) = \mathbf{0}, \quad \|x\|_1 \leq 1.$$

2

The authors focused on the volume of the feasible region of $\mathrm{P}_{S_{1,\infty}}(\mathcal{A})$ and devised a rescaling method so that the volume becomes smaller. Their method will stop when the feasibility of problem $\mathrm{P}_{S_{1,\infty}}(\mathcal{A})$ or the fact that the minimum eigenvalue of any feasible solution of problem $\mathrm{P}_{S_{1,\infty}}(\mathcal{A})$ is less than ε is determined.

The aim of this paper is to devise a new variant of Chubanov's method for solving $\mathrm{P}(\mathcal{A})$ by extending Roos's method [15] to the following feasibility problem ($\mathrm{P}_{S_\infty}(\mathcal{A})$) over the symmetric cone $\mathcal{K}$:

$$\mathrm{P}_{S_\infty}(\mathcal{A}) \qquad \text{find} \quad x \in \mathrm{int}\,\mathcal{K} \qquad \text{s.t.} \quad \mathcal{A}(x) = \mathbf{0}, \quad \|x\|_\infty \leq 1,$$

where $\|x\|_\infty$ is the maximum absolute eigenvalue of x. Throughout this paper, we will assume that $\mathcal{K}$ is the Cartesian product of p simple symmetric cones $\mathcal{K}_1, \ldots, \mathcal{K}_p$, i.e., $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_p$. Here, we should mention an important issue about Lemma 4.2 in [15], which is one of the main results of [15]. The proof of Lemma 4.2 given in the paper [15] is incorrect and a correct proof is provided in the paper [19], while this study derives theoretical results without referring to the lemma. Our method has a feature that the main algorithm works while keeping information about the minimum eigenvalue of any feasible solution of $\mathrm{P}_{S_\infty}(\mathcal{A})$ and, in this sense, it is closely related to Lourenço et al.'s method [10]. Using the norm $\|\cdot\|_\infty$ in problem $\mathrm{P}_{S_\infty}(\mathcal{A})$ makes it possible to

- calculate the upper bound for the minimum eigenvalue of any feasible solution of $\mathrm{P}_{S_\infty}(\mathcal{A})$,

- quantify the feasible region of $\mathrm{P}(\mathcal{A})$, and hence,

- determine whether there exists a feasible solution of $\mathrm{P}(\mathcal{A})$ whose minimum eigenvalue is greater than ε as in [10].

Note that the symmetric cone optimization includes several types of problems (linear, second-order cone, and semi-definite optimization problems) with various settings and the computational bound of an algorithm depends on these settings. As we will describe in section 6, the theoretical computational bound of our method is

- equivalent to that of Roos's original method [15] and superior to that of Lourenço et al.'s method [10] when the symmetric cone is the nonnegative orthant,

- superior to that of Lourenço et al.'s method when the symmetric cone is a Cartesian product of second-order cones, and

- equivalent to that of Lourenço et al.'s method when the symmetric cone is the simple positive semidefinite cone, under the assumption that the costs of computing the spectral decomposition and of the minimum eigenvalue are of the same order for any given symmetric matrix.

- superior to that of Pena and Soheili's method [13] for any simple symmetric cones under the feasibility assumption of the problem imposed in [13].

Another aim of this paper is to give comprehensive numerical comparisons of the existing algorithms and our method. As described in section 7, we generate strongly feasible ill-conditioned instances, i.e., $\ker \mathcal{A} \cap \mathrm{int}\,\mathcal{K} \neq \emptyset$ and $x \in \ker \mathcal{A} \cap \mathrm{int}\,\mathcal{K}$ has positive but small eigenvalues, for the simple positive semidefinite cone $\mathcal{K}$, and conduct numerical experiments.

The paper is organized as follows: Section 2 contains a brief description of Euclidean Jordan algebras and their basic properties. Section 3 gives a collection of propositions which are necessary to extend Roos's method to problem $\mathrm{P}_{S_\infty}(\mathcal{A})$ over the symmetric cone. In sections 4 and 5, we explain the basic procedure and the main algorithm of our variant of Chubanov's method. Section 6 compares the theoretical computational bounds of Lourenço et al.'s method [10], Pena and Soheili's method [13] and our method. In section 7, we conduct numerical experiments comparing our variant with the existing methods. The conclusions are summarized in section 8.

2 Euclidean Jordan algebras and their basic properties

In this section, we briefly introduce Euclidean Jordan algebras and symmetric cones. For more details, see [5]. In particular, the relation between symmetry cones and Euclidean Jordan algebras is given in Chapter III (Koecher and Vinberg theorem) of [5].

2.1 Euclidean Jordan algebras

Let $\mathbb{E}$ be a real-valued vector space equipped with an inner product $\langle \cdot, \cdot \rangle$ and a bilinear operation $\circ :$ $\mathbb{E} \times \mathbb{E} \to \mathbb{E}$, and e be the identity element, i.e., $x \circ e = e \circ x = x$ holds for any $x \in \mathbb{E}$. $(\mathbb{E}, \circ)$ is called a Euclidean Jordan algebra if it satisfies

$$x \circ y = y \circ x, \quad x \circ (x^2 \circ y) = x^2 \circ (x \circ y), \quad \langle x \circ y, z \rangle = \langle y, x \circ z \rangle$$

for all $x, y, z \in \mathbb{E}$ and $x^2 := x \circ x$. We denote $y \in \mathbb{E}$ as x^{-1} if y satisfies $x \circ y = e$. $c \in \mathbb{E}$ is called an idempotent if it satisfies $c \circ c = c$, and an idempotent c is called primitive if it can not be written as a sum of two or more nonzero idempotents. A set of primitive idempotents $c_1, c_2, \ldots c_k$ is called a Jordan frame if $c_1, \ldots c_k$ satisfy

$$c_i \circ c_j = 0 \ (i \neq j), \quad c_i \circ c_i = c_i \ (i = 1, \ldots, k), \quad \sum_{i=1}^{k} c_i = e.$$

For $x \in \mathbb{E}$, the degree of x is the smallest integer d such that the set $\{e, x, x^2, \ldots, x^d\}$ is linearly independent. The rank of $\mathbb{E}$ is the maximum integer r of the degree of x over all $x \in \mathbb{E}$. The following properties are known.

Proposition 2.1 (Spectral theorem (cf. Theorem III.1.2 of [5])). *Let $(\mathbb{E}, \circ)$ be a Euclidean Jordan algebra having rank r. For any $x \in \mathbb{E}$, there exist real numbers $\lambda_1, \ldots, \lambda_r$ and a Jordan frame $c_1, \ldots, c_r$ for which the following holds:*

$$x = \sum_{i=1}^{r} \lambda_i c_i.$$

The numbers $\lambda_1, \ldots, \lambda_r$ are uniquely determined eigenvalues of x (with their multiplicities). Furthermore, $\mathrm{trace}(x) := \sum_{i=1}^{r} \lambda_i$, $\det(x) := \prod_{i=1}^{r} \lambda_i$.

2.2 Symmetric cone

A proper cone is symmetric if it is self-dual and homogeneous. It is known that the set of squares $\mathcal{K} = \{x^2 : x \in \mathbb{E}\}$ is the symmetric cone of $\mathbb{E}$ (cf. Theorems III.2.1 and III.3.1 of [5]). The following properties can be derived from the results in [5], as in Corollary 2.3 of [21]:

Proposition 2.2. *Let $x \in \mathbb{E}$ and let $\sum_{j=1}^{r} \lambda_j c_j$ be a decomposition of x given by Propositoin 2.1. Then*

(i) $x \in \mathcal{K}$ *if and only if* $\lambda_j \geq 0$ $(j = 1, 2, \ldots, r)$,

(ii) $x \in \mathrm{int}\,\mathcal{K}$ *if and only if* $\lambda_j > 0$ $(j = 1, 2, \ldots, r)$.

From Propositions 2.1 and 2.2 for any $x \in \mathbb{E}$, its projection $P_{\mathcal{K}}(x)$ onto $\mathcal{K}$ can be written as an operation to round all negative eigenvalues of x to 0, i.e., $P_{\mathcal{K}}(x) = \sum_{i=1}^{r} [\lambda_i]^+ c_i$, where $[\cdot]^+$ denotes the projection onto the nonnegative orthant. Using $P_{\mathcal{K}}$, we can decompose any $x \in \mathbb{E}$ as follows.

4

Lemma 2.3. *Let $x \in \mathbb{E}$, and $\mathcal{K}$ be the symmetric cone corresponding to $\mathbb{E}$. Then, x can be decomposed into $x = P_{\mathcal{K}}(x) - P_{\mathcal{K}}(-x)$.*

Proof. From Propositoin 2.1, let x be given as $x = \sum_{i=1}^{r} \lambda_i c_i$. Let I_1 be the set of indices such that $\lambda_i \geq 0$ and I_2 be the set of indices such that $\lambda_i < 0$. Then, we have $P_{\mathcal{K}}(x) = \sum_{i \in I_1} \lambda_i c_i$ and $P_{\mathcal{K}}(-x) = \sum_{i \in I_2} -\lambda_i c_i$, which implies that $x = \sum_{i \in I_1} \lambda_i c_i + \sum_{i \in I_2} \lambda_i c_i = P_{\mathcal{K}}(x) - P_{\mathcal{K}}(-x)$. $\square$

A Euclidean Jordan algebra $(\mathbb{E}, \circ)$ is called <u>simple</u> if it cannot be written as any Cartesian product of non-zero Euclidean Jordan algebras. If the Euclidean Jordan algebra $(\mathbb{E}, \circ)$ associated with a symmetric cone $\mathcal{K}$ is simple, then we say that $\mathcal{K}$ is <u>simple</u>. In this paper, we will consider that $\mathcal{K}$ is given by a Cartesian product of p simple symmetric cones $\mathcal{K}_\ell$, $\mathcal{K} := \mathcal{K}_1 \times \cdots \times \mathcal{K}_p$, whose rank and identity element are r_ℓ and e_ℓ $(\ell = 1, \ldots, p)$. The rank r and the identity element of $\mathcal{K}$ are given by

$$r = \sum_{\ell=1}^{p} r_\ell, \quad e = (e_1, \ldots, e_p). \tag{1}$$

In what follows, x_ℓ stands for the ℓ-th block element of $x \in \mathcal{K}$, i.e., $x = (x_1, \ldots, x_p) \in \mathcal{K}_1 \times \cdots \times \mathcal{K}_p$. For each $\ell = 1, \cdots, p$, we define $\lambda_{\min}(x_\ell) := \min\{\lambda_1, \cdots, \lambda_{r_\ell}\}$ where $\lambda_1, \cdots, \lambda_{r_\ell}$ are eigenvalues of x_ℓ. The minimum eigenvalue $\lambda_{\min}(x)$ of $x \in \mathcal{K}$ is given by $\lambda_{\min}(x) = \min\{\lambda_{\min}(x_1), \cdots, \lambda_{\min}(x_p)\}$.

Next, we consider the <u>quadratic representation</u> $Q_v(x)$ defined by $Q_v(x) := 2v \circ (v \circ x) - v^2 \circ x$. For the cone $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_p$, the quadratic representation $Q_v(x)$ of $x \in \mathcal{K}$ is denoted by $Q_v(x) = (Q_{v_1}(x_1), \ldots, Q_{v_p}(x_p))$. Letting I_ℓ be the identity operator of the Euclidean Jordan algebra $(\mathbb{E}_\ell, \circ_\ell)$ associated with the cone $\mathcal{K}_\ell$, we have $Q_{e_\ell} = I_\ell$ for $\ell = 1, \ldots, p$. The following properties can also be retrieved from the results in [5] as in Proposition 3 of [10]:

Proposition 2.4. *For any $v \in \text{int}\mathcal{K}$, $Q_v(\mathcal{K}) = \mathcal{K}$.*

It is also known that the following relations hold for the quadratic representation Q_v and $\det(\cdot)$ [5].

Proposition 2.5 (cf. Proposition II.3.3 and III.4.2-(i), [5])**.** *For any $v, x \in \mathbb{E}$,*

1. $\det Q_v(x) = \det(v)^2 \det(x)$,

2. $Q_{Q_v(x)} = Q_v Q_x Q_v$ (i.e., if $x = e$ then $Q_{v^2} = Q_v Q_v$) .

More detailed descriptions, including concrete examples of symmetric cone optimization, can be found in, e.g., [5, 6, 16, 1]. Here, we will use concrete examples of symmetric cones to explain the biliniear operation, the identity element, the inner product, the eigenvalues, the primitive idempotents, the projection on the symmetric cone and the quadratic representation on the cone.

Example 2.6 ($\mathcal{K}$ is the semidefinite cone $\mathbb{S}^n_+$)**.** Let $\mathbb{S}^n$ be the set of symmetric matrices of $n \times n$. The semidefinite cone $\mathbb{S}^n_+$ is given by $\mathbb{S}^n_+ = \{X \in \mathbb{S}^n : X \succeq O\}$. For any symmetric matrices $X, Y \in \mathbb{S}^n$, define the bilinear operation $\circ$ and inner product as $X \circ Y = \frac{XY + YX}{2}$ and $\langle X, Y \rangle = \text{tr}(XY) = \sum_{i=1}^{n} \sum_{j=1}^{n} X_{ij} Y_{ij}$, respectively. For any $X \in \mathbb{S}^n$, perform the eigenvalue decomposition and let $u_1, \ldots, u_n$ be the corresponding normalized eigenvectors for the eigenvalues $\lambda_1, \ldots, \lambda_n$: $X = \sum_{i=1}^{n} \lambda_i u_i u_i^T$. The eigenvalues of X in the Jordan algebra are $\lambda_1, \ldots, \lambda_n$ and the primitive idempotents are $c_1 = u_1 u_1^T, \ldots, c_n = u_n u_n^T$, which implies that the rank of the semidefinite cone $\mathbb{S}^n_+$ is $r = n$. The identity element is the identity matrix I and the projection onto $\mathbb{S}^n_+$ is given by $P_{\mathbb{S}^n_+}(X) = \sum_{i=1}^{n} [\lambda_i]^+ u_i u_i^T$. The quadratic representation of $V \in \mathbb{S}^n$ is given by $Q_V(X) = VXV$.

Example 2.7 ($\mathcal{K}$ is the second-order cone $\mathbb{L}_n$). The second order cone is given by $\mathbb{L}_n = \{(x_1, \bar{\boldsymbol{x}}^\top)^\top \in \mathbb{R}^n : x_1 \geq \|\bar{\boldsymbol{x}}\|_2\}$. For any $x, y \in \mathbb{R}^n$, define the bilinear operation $\circ$ and the inner product as $x \circ y = (x^\top y, (x_1 \bar{\boldsymbol{y}} + y_1 \bar{\boldsymbol{x}})^\top)^\top$ and $\langle x, y \rangle = 2 \sum_{i=1}^n x_i y_i$, respectively. For any $x \in \mathbb{R}^n$, by the decomposition

$$x = (x_1 + \|\bar{\boldsymbol{x}}\|_2) \begin{pmatrix} 1/2 \\ \frac{\bar{\boldsymbol{x}}}{2\|\bar{\boldsymbol{x}}\|_2} \end{pmatrix} + (x_1 - \|\bar{\boldsymbol{x}}\|_2) \begin{pmatrix} 1/2 \\ -\frac{\bar{\boldsymbol{x}}}{2\|\bar{\boldsymbol{x}}\|_2} \end{pmatrix},$$

we obtain the eigenvalues and the primitive idempotents as follows:

$$\lambda_1 = x_1 + \|\bar{\boldsymbol{x}}\|_2 \quad , \quad \lambda_2 = x_1 - \|\bar{\boldsymbol{x}}\|_2,$$

$$c_1 = \begin{cases} \frac{1}{2}(1, \frac{\bar{\boldsymbol{x}}^\top}{\|\bar{\boldsymbol{x}}\|_2})^\top & \|\bar{\boldsymbol{x}}\|_2 \neq 0 \\ \frac{1}{2}(1, z^\top)^\top & \|\bar{\boldsymbol{x}}\|_2 = 0 \end{cases} \quad , \quad c_2 = \begin{cases} \frac{1}{2}(1, -\frac{\bar{\boldsymbol{x}}^\top}{\|\bar{\boldsymbol{x}}\|_2})^\top & \|\bar{\boldsymbol{x}}\|_2 \neq 0 \\ \frac{1}{2}(1, -z^\top)^\top & \|\bar{\boldsymbol{x}}\|_2 = 0 \end{cases}.$$

where $z \in \mathbb{R}^{n-1}$ is an arbitrary vector satisfying $\|z\|_2 = 1$. The above implies that the rank of the second-order cone $\mathbb{L}_n$ is $r = 2$. The identity element is given by $e = (1, \boldsymbol{0}^\top)^\top \in \mathbb{R}^n$. The projection $P_{\mathbb{L}_n}(x)$ onto $\mathbb{L}_n$ is given by

$$P_{\mathbb{L}_n}(x) = [x_1 + \|\bar{\boldsymbol{x}}\|_2]^+ \begin{pmatrix} 1/2 \\ \frac{\bar{\boldsymbol{x}}}{2\|\bar{\boldsymbol{x}}\|_2} \end{pmatrix} + [x_1 - \|\bar{\boldsymbol{x}}\|_2]^+ \begin{pmatrix} 1/2 \\ -\frac{\bar{\boldsymbol{x}}}{2\|\bar{\boldsymbol{x}}\|_2} \end{pmatrix}.$$

Letting I_{n-1} be the identity matrix of order $n-1$, the quadratic representation $Q_v(\cdot)$ of $v \in \mathbb{R}^n$ is as follows:

$$Q_v(x) = \begin{pmatrix} \|v\|_2^2 & 2v_1 \bar{\boldsymbol{v}}^T \\ 2v_1 \bar{\boldsymbol{v}} & \det v I_{n-1} + 2\bar{\boldsymbol{v}}\bar{\boldsymbol{v}}^T \end{pmatrix} x.$$

2.3 Notation

This subsection summarizes the notations used in this paper. For any $x, y \in \mathbb{E}$, we define the inner product $\langle \cdot, \cdot \rangle$ and the norm $\| \cdot \|_J$ as $\langle x, y \rangle := \mathrm{trace}(x \circ y)$ and $\|x\|_J := \sqrt{\langle x, x \rangle}$, respectively. For any $x \in \mathbb{E}$ having decomposition $x = \sum_{i=1}^r \lambda_i c_i$ as in Proposition 2.1, we also define $\|x\|_1 := |\lambda_1| + \cdots + |\lambda_r|$, $\|x\|_\infty := \max\{|\lambda_1|, \ldots, |\lambda_r|\}$. For $x \in \mathcal{K}$, we obtain the following equivalent representations: $\|x\|_1 = \langle e, x \rangle$, $\|x\|_\infty = \lambda_{\max}(x)$. The following is a list of other definitions and frequently used symbols in the paper.

- d: the dimension of the Euclidean space $\mathbb{E}$ corresponding to $\mathcal{K}$,

- $F_{\mathrm{P}_{S_\infty}(\mathcal{A})}$: the feasible region of $\mathrm{P}_{S_\infty}(\mathcal{A})$,

- $P_\mathcal{A}(\cdot)$: the projection map onto $\ker \mathcal{A}$,

- $\mathcal{P}_\mathcal{K}(\cdot)$: the projection map onto $\mathcal{K}$,

- $\lambda(x) \in \mathbb{R}^r$: an r-dimensional vector composed of the eigenvalues of $x \in \mathcal{K}$,

- $\lambda(x_\ell) \in \mathbb{R}^{r_\ell}$: an r_ℓ-dimensional vector composed of the eigenvalues of $x_\ell \in \mathcal{K}_\ell$ ($\ell = 1, \ldots, p$),

- $c(x_\ell)_i \in \mathcal{K}_\ell$: the i-th primitive idempotent of $x_\ell \in \mathbb{E}_\ell$. When $\mathcal{K}$ is simple, it is abbreviated as c_i.

- $[\cdot]^+$: the projection map onto the nonnegative orthant, and

- $\mathcal{A}^*(\cdot)$: the adjoint operator of the linear operator $\mathcal{A}(\cdot)$, i.e., $\langle \mathcal{A}(x), y \rangle = \langle x, \mathcal{A}^*(y) \rangle$ for all $x \in \mathcal{K}$ and $y \in \mathbb{R}^m$.

6

3 Extension of Roos's method to the symmetric cone problem

3.1 Outline of the extended method

We focus on the feasibility of the following problem $\mathrm{P}_{S_\infty}(\mathcal{A})$, which is equivalent to $\mathrm{P}(\mathcal{A})$:

$$\mathrm{P}_{S_\infty}(\mathcal{A}) \qquad \text{find} \quad x \in \operatorname{int}\mathcal{K} \qquad \text{s.t.} \quad \mathcal{A}(x) = \mathbf{0}, \quad \|x\|_\infty \leq 1.$$

The alternative problem $\mathrm{D}(\mathcal{A})$ of $\mathrm{P}(\mathcal{A})$ is

$$\mathrm{D}(\mathcal{A}) \qquad \text{find} \quad y \in \mathcal{K} \qquad \text{s.t.} \quad y \in \operatorname{range}\mathcal{A}^*, \quad y \neq \mathbf{0},$$

where $\operatorname{range}\mathcal{A}^*$ is the orthogonal complement of $\ker\mathcal{A}$. As we mentioned in section 2.2, we assume that $\mathcal{K}$ is given by a Cartesian product of p simple symmetric cones $\mathcal{K}_\ell (\ell = 1, \ldots, p)$, i.e., $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_p$. In our method, the upper bound for the sum of eigenvalues of a feasible solution of $\mathrm{P}_{S_\infty}(\mathcal{A})$ plays a key role, whereas the existing work focuses on the volume of the set of the feasible region [10] or the condition number of a feasible solution [13]. Before describing the theoretical results, let us outline the proposed algorithm when $\mathcal{K}$ is simple. The algorithm repeats two steps: (i). find a cut for $\mathrm{P}_{S_\infty}(\mathcal{A})$, (ii) scale the problem to an isomorphic problem equivalent to $\mathrm{P}_{S_\infty}(\mathcal{A})$ such that the region narrowed by the cut is efficiently explored. Given a feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$ and a constant $0 < \xi < 1$, our method first searches for a Jordan frame $\{c_1, \ldots, c_r\}$ such that the following is satisfied:

$$\langle c_i, x \rangle \leq \xi \ (i \in H), \quad \langle c_i, x \rangle \leq 1 \ (i \notin H),$$

where $H \subseteq \{1, \ldots, r\}$ and $|H| > 0$. In this case, instead of $\mathrm{P}_{S_\infty}(\mathcal{A})$, we may consider $\mathrm{P}_{S_\infty}^{\mathrm{Cut}}(\mathcal{A})$ as follows:

$$\mathrm{P}_{S_\infty}^{\mathrm{Cut}}(\mathcal{A}) \qquad \text{find} \quad x \in \operatorname{int}\mathcal{K} \qquad \text{s.t.} \quad \langle c_i, x \rangle \leq \xi \ (i \in H), \quad \langle c_i, x \rangle \leq 1 \ (i \notin H),$$
$$\mathcal{A}(x) = \mathbf{0}, \ \|x\|_\infty \leq 1.$$

Here, we define the set $SR^{\mathrm{Cut}} = \{x \in \mathbb{E} : x \in \operatorname{int}\mathcal{K}, \ \|x\|_\infty \leq 1, \ \langle c_i, x \rangle \leq \xi \ (i \in H), \ \langle c_i, x \rangle \leq 1 \ i \notin H)\}$ as the search range for the solutions of the problem $\mathrm{P}_{S_\infty}^{\mathrm{Cut}}(\mathcal{A})$. The proposed method then creates a problem equivalent and isomorphic to $\mathrm{P}_{S_\infty}(\mathcal{A})$ such that SR^{Cut}, the region narrowed by the cut, can be searched efficiently. Such a problem is obtained as follows:

$$\mathrm{P}_{S_\infty}(\mathcal{A}Q_g) \qquad \text{find} \quad \bar{x} \in \operatorname{int}\mathcal{K} \qquad \text{s.t.} \quad \mathcal{A}Q_g(\bar{x}) = \mathbf{0}, \quad \|\bar{x}\|_\infty \leq 1,$$

where g is given by $g = \sqrt{\xi}\sum_{i \in H} c_i + \sum_{i \notin H} c_i \in \operatorname{int}\mathcal{K}$ for which $e = Q_{g^{-1}}(u)$ holds for $u = \sum_{i \in H}\xi c_i + \sum_{i \notin H} c_i$.

In the succeeding sections, we explain how the cut for $\mathrm{P}_{S_\infty}(\mathcal{A})$ is obtained from some $v \in \operatorname{range}\mathcal{A}^*$; we also explain the scaling method for the problem in detail. To simplify our discussion, we will assume that $\mathcal{K}$ is simple, i.e., $p = 1$, in section 3.2. Then, in section 3.3, we will generalize our discussion to the case of $p \geq 2$.

3.2 Simple symmetric cone case

Let us consider the case where $\mathcal{K}$ is simple. It is obvious that, for any feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$, the constraint $\|x\|_\infty \leq 1$ implies that $\langle e, x \rangle \leq r$, since $x \in \mathcal{K}$. In Proposition 3.3, we show that this bound may be improved as $\langle e, x \rangle < r$ by using a point $v \in \operatorname{range}\mathcal{A}^* \setminus \{0\}$. To prove Proposition 3.3, we need the following Lemma 3.1 and Proposition 3.2.

Lemma 3.1. *Let* $(\mathbb{E}, \circ)$ *be a Euclidean Jordan algebra with the associated symmetric cone* $\mathcal{K}$. *For any* $y \in \mathbb{E}$, *the following equation holds:*

$$\max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} \langle y, x \rangle = \langle \mathcal{P}_{\mathcal{K}}(y), e \rangle.$$

Proof. Using the decomposition $y = \sum_{i=1}^{r} \lambda_i c_i$ obtained by Proposition 2.1, we see that

$$\max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} \langle y, x \rangle = \max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} \left\langle \sum_{i=1}^{r} \lambda_i c_i, x \right\rangle = \max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} \sum_{i=1}^{r} \lambda_i \langle c_i, x \rangle. \tag{2}$$

Noting that $x \in \mathcal{K}, e - x \in \mathcal{K}$ from $\mathbf{0} \leq \lambda(x) \leq \mathbf{1}$, since $c_i \in \mathcal{K}$ is primitive idempotent, we find that $\langle c_i, x \rangle \geq 0$ and $\langle c_i, e - x \rangle \geq 0$, which implies that $0 \leq \langle c_i, x \rangle \leq 1$. Thus, letting I_1 be the set of indices for which $\lambda_i \leq 0$ and I_2 be the set of indices for which $\lambda_i > 0$, if there exists an x satisfying

$$\langle c_i, x \rangle = \begin{cases} 0 & i \in I_1 \\ 1 & i \in I_2 \end{cases}, \tag{3}$$

then such an x is an optimal solution of (2). In fact, if we define $x^* = \sum_{i \in I_2} c_i$, then by the dedfinition of the Jordan frame, x^* satisfies (3) and $\mathbf{0} \leq \lambda(x) \leq \mathbf{1}$ and becomes an optimal solution of (2). In this case, the optimal value of (2) turns out to be

$$\max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} \sum_{i=1}^{r} \lambda_i \langle c_i, x \rangle = \sum_{i=1}^{r} \lambda_i \langle c_i, x^* \rangle = \sum_{i \in I_2} \lambda_i = \sum_{i=1}^{r} [\lambda_i]^+ = \langle \mathcal{P}_{\mathcal{K}}(y), e \rangle.$$

$\square$

Proposition 3.2. *Let* $(\mathbb{E}, \circ)$ *be a Euclidean Jordan Algebra with the corresponding symmetric cone* $\mathcal{K}$. *For a given* $c \in \mathbb{E}$, *consider the problem*

$$\max \quad \langle c, x \rangle \qquad \text{s.t} \quad \mathcal{A}(x) = \mathbf{0}, \quad \mathbf{0} \leq \lambda(x) \leq \mathbf{1}.$$

The dual problem of the above is

$$\min \quad \langle \mathcal{P}_{\mathcal{K}}(c - u), e \rangle \qquad \text{s.t} \quad u \in \text{range} \mathcal{A}^*.$$

Proof. Define the Lagrangian function $L(x, w)$ as $L(x, w) := \langle c, x \rangle - w^\top \mathcal{A}(x)$ where $w \in \mathbb{R}^m$ is the Lagrange multiplier. Supoose that x^* is an optimal sotution of the primal problem. Then, for any $w \in \mathbb{R}^m$, we have $\langle c, x^* \rangle = L(w, x^*) \leq \max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} L(w, x)$, and hence,

$$\begin{aligned}
\langle c, x^* \rangle &\leq \min_{w} \max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} L(x, w) = \min_{w} \max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} \{\langle c, x \rangle - \langle \mathcal{A}^*(w), x \rangle\} \\
&= \min_{w} \max_{\mathbf{0} \leq \lambda(x) \leq \mathbf{1}} \{\langle c - \mathcal{A}^*(w), x \rangle\} \\
&= \min_{w} \langle \mathcal{P}_{\mathcal{K}}(c - \mathcal{A}^*(w)), e \rangle \quad \text{(by lemma 3.1)} \\
&= \min_{u \in \text{range} \mathcal{A}^*} \langle \mathcal{P}_{\mathcal{K}}(c - u), e \rangle.
\end{aligned}$$

$\square$

Proposition 3.3. *Suppose that* $v \in \text{range} \mathcal{A}^*$ *is given by* $v = \sum_{i=1}^{r} \lambda_i c_i$ *as in Proposition 2.1. For each* $i \in \{1, \ldots, r\}$ *and* $\alpha \in \mathbb{R}$, *define* $q_i(\alpha) := [1 - \alpha \lambda_i]^+ + \sum_{j \neq i}^{r} [-\alpha \lambda_j]^+$. *Then, the following relations hold for any* $x \in F_{\mathrm{P}_{S_\infty}}(\mathcal{A})$ *and* $i \in \{1, \ldots, r\}$:

$$\langle c_i, x \rangle \leq \min_{\alpha \in \mathbb{R}} q_i(\alpha) = \begin{cases} \min\left\{1, \left\langle e, \mathcal{P}_{\mathcal{K}}\left(-\frac{1}{\lambda_i} v\right) \right\rangle\right\} & \text{if } \lambda_i \neq 0, \\ 1 & \text{if } \lambda_i = 0. \end{cases} \tag{4}$$

Proof. For each $i \in \{1, 2, \ldots, r\}$, we have

$$\mathcal{P}_{\mathcal{K}}(c_i - \alpha v) = \mathcal{P}_{\mathcal{K}}\left(c_i - \alpha \sum_{j=1}^{r} \lambda_j c_j\right) = \mathcal{P}_{\mathcal{K}}\left((1 - \alpha \lambda_i) c_i - \sum_{j \neq i}^{r} \alpha \lambda_j c_j\right),$$

8

and hence,

$$\langle \mathcal{P}_{\mathcal{K}} \left(c_i - \alpha v \right), e \rangle = \left\langle \mathcal{P}_{\mathcal{K}} \left((1 - \alpha\lambda_i)c_i - \sum_{j \neq i}^{r} \alpha\lambda_j c_j \right), \sum_{i=1}^{r} c_i \right\rangle = [1 - \alpha\lambda_i]^+ + \sum_{j \neq i}^{r} [-\alpha\lambda_j]^+ = q_i(\alpha). \quad (5)$$

Note that, since $q_i(\alpha)$ is a piece-wise linear convex function, if $\lambda_i = 0$, it attains the minimum at $\alpha = 0$ with $q_i(0) = 1$, and if $\lambda_i \neq 0$, it attains the minimum at $\alpha = 0$ with $q_i(0) = 1$ or at $\alpha = \frac{1}{\lambda_i}$ with

$$q\left(\frac{1}{\lambda_i}\right) = \sum_{j \neq i}^{r} \left[-\frac{\lambda_j}{\lambda_i}\right]^+ = \sum_{j=1}^{r} \left[-\frac{\lambda_j}{\lambda_i}\right]^+ = \left\langle e, \mathcal{P}_{\mathcal{K}}\left(-\frac{1}{\lambda_i}v\right)\right\rangle.$$

Thus, we obtain equivalence in (4). Since $\alpha v \in \text{range}\,\mathcal{A}^*$ for all $\alpha \in \mathbb{R}$, for each $i \in \{1, \ldots, r\}$, Proposition 3.2 and (5) ensure that $\langle c_i, x \rangle \leq \langle \mathcal{P}_{\mathcal{K}} \left(c_i - \alpha v \right), e \rangle = q_i(\alpha)$ for all $\alpha \in \mathbb{R}$, which implies the inequality in (4). $\square$

Since $\sum_{i=1}^{r} c_i = e$ holds, Proposition 3.3 allows us to compute upper bounds for the sum of eigenvalues of x. The following proposition gives us information about indices whose upper bound for $\langle c_i, x \rangle$ in Proposition 3.3 is less than 1.

Proposition 3.4. *Suppose that $v \in \text{range}\,\mathcal{A}^*$ is given by $v = \sum_{i=1}^{r} \lambda_i c_i$ as in Proposition 2.1. If v satisfies $\left\langle e, \mathcal{P}_{\mathcal{K}}\left(-\frac{1}{\lambda_i}v\right)\right\rangle = \xi < 1$ for some $\xi < 1$ and for some $i \in \{1, \ldots, r\}$ for which $\lambda_i \neq 0$ holds, then λ_i has the same sign as $\langle e, v \rangle$.*

Proof. First, we consider the case where $\lambda_i > 0$. Since the assumption implies that $\langle e, \mathcal{P}_{\mathcal{K}}(-v) \rangle = \lambda_i \xi$, we have

$$\langle e, v \rangle = \langle e, \mathcal{P}_{\mathcal{K}}(v) \rangle - \langle e, \mathcal{P}_{\mathcal{K}}(-v) \rangle = \langle e, \mathcal{P}_{\mathcal{K}}(v) \rangle - \lambda_i \xi \geq \lambda_i(1 - \xi) > 0,$$

where the first equality comes from Lemma 2.3.

For the case where $\lambda_i < 0$, since the assumption also implies that $\langle e, \mathcal{P}_{\mathcal{K}}(v) \rangle = -\lambda_i \xi$, we have

$$\langle e, v \rangle = \langle e, \mathcal{P}_{\mathcal{K}}(v) \rangle - \langle e, \mathcal{P}_{\mathcal{K}}(-v) \rangle = -\lambda_i \xi - \langle e, \mathcal{P}_{\mathcal{K}}(-v) \rangle \leq -\lambda_i \xi - (-\lambda_i) = (1 - \xi)\lambda_i < 0.$$

This completes the proof. $\square$

The above two propositions imply that, for any $v \in \text{range}\,\mathcal{A}^*$ with $v = \sum_{i=1}^{r} \lambda_i c_i$, if we compute $\langle c_i, x \rangle$ according to Proposition 3.3 for $i \in \{1, \ldots, r\}$ having the same sign as the one of $\langle e, v \rangle$, we obtain an upper bound for the sum of eigenvalues of x over the set $F_{\mathrm{P}_{S_\infty}(\mathcal{A})}$. The following proposition concerns the scaling method of problem $\mathrm{P}_{S_\infty}(\mathcal{A})$ when we find such a $v \in \text{range}\,\mathcal{A}^*$.

Proposition 3.5. *Let $H \subseteq \{1, \ldots r\}$ be a nonempty set, $c_1, \ldots, c_r$ be a Jordan frame, and ξ be a real number satisfying $0 < \xi < 1$. Let us define $g \in \text{int}\,\mathcal{K}$ as*

$$g := \sqrt{\xi} \sum_{h \in H} c_h + \sum_{h \notin H} c_h \quad i.e., \quad g^{-1} = \frac{1}{\sqrt{\xi}} \sum_{h \in H} c_h + \sum_{h \notin H} c_h. \quad (6)$$

For the two sets $SR^{\mathrm{Cut}} = \{x \in \mathbb{E} : x \in \text{int}\,\mathcal{K}, \ \|x\|_\infty \leq 1, \ \langle c_i, x \rangle \leq \xi \ (i \in H), \ \langle c_i, x \rangle \leq 1 \ (i \notin H)\}$ and, $SR^{\mathrm{Scaled}} = \{\bar{x} \in \mathbb{E} : \bar{x} \in \text{int}\,\mathcal{K}, \ \|\bar{x}\|_\infty \leq 1\}$, $Q_g(SR^{\mathrm{Scaled}}) \subseteq SR^{\mathrm{Cut}}$ holds.

9

Proof. Let $\bar{x}$ be an arbitrary point of SR^{Scaled}. It suffices to show that (i) $Q_g(\bar{x}) \in \mathrm{int}\,\mathcal{K}$, (ii) $\|Q_g(\bar{x})\|_\infty \le 1$, (iii) $\langle c_i, Q_g(\bar{x}) \rangle \le \xi$ $(i \in H)$ and (iv) $\langle c_i, Q_g(\bar{x}) \rangle \le 1$ $(i \notin H)$ hold.

(i): Let us show that $Q_g(\bar{x}) \in \mathrm{int}\,\mathcal{K}$. Since g and $\bar{x}$ lie in the set $\mathrm{int}\,\mathcal{K}$, from Propositions 2.4 and 2.5, we see that

$$Q_g(\bar{x}) \in \mathcal{K}, \quad \det Q_g(\bar{x}) = \det(g)^2 \det(\bar{x}) > 0,$$

which implies that $Q_g(\bar{x}) \in \mathrm{int}\,\mathcal{K}$.

(ii): Next let us show that $\|Q_g(\bar{x})\|_\infty \le 1$. Since $\bar{x} \in SR^{\mathrm{Scaled}}$, we see that $\bar{x} \in \mathrm{int}\,\mathcal{K}$, $\|\bar{x}\|_\infty \le 1$ and hence $e - \bar{x} \in \mathcal{K}$. Since $g \in \mathrm{int}\,\mathcal{K}$, Proposition 2.4 guarantees that

$$Q_g(e - \bar{x}) \in \mathcal{K}. \tag{7}$$

By the definition (6) of g, the following equations hold for $c_1, \ldots, c_r$:

$$\text{For any } i \in H, \quad Q_g(c_i) = 2g \circ (g \circ c_i) - (g \circ g) \circ c_i$$

$$= 2g \circ \sqrt{\xi} c_i - \left(\xi \sum_{h \in H} c_h + \sum_{h \notin H} c_h \right) \circ c_i$$

$$= 2\xi c_i - \xi c_i = \xi c_i.$$

$$\text{For any } i \notin H, \quad Q_g(c_i) = 2g \circ (g \circ c_i) - (g \circ g) \circ c_i$$

$$= 2g \circ c_i - \left(\xi \sum_{h \in H} c_h + \sum_{h \notin H} c_h \right) \circ c_i$$

$$= 2c_i - c_i = c_i.$$

Thus, we obtain $Q_g(e) = \xi \sum_{i \in H} c_i + \sum_{i \notin H} c_i$. Combining this with the facts $c_i \in \mathcal{K}$ and $(1 - \xi) > 0$ and (7), we have

$$\mathcal{K} \ni (1 - \xi) \sum_{i \in H} c_i + Q_g(e - \bar{x}) = (1 - \xi) \sum_{i \in H} c_i + Q_g(e) - Q_g(\bar{x})$$

$$= (1 - \xi) \sum_{i \in H} c_i + \left(\xi \sum_{i \in H} c_i + \sum_{i \notin H} c_i \right) - Q_g(\bar{x})$$

$$= e - Q_g(\bar{x}).$$

Since we have shown that $Q_g(\bar{x}) \in \mathrm{int}\,\mathcal{K}$, we can conclude that $\|Q_g(\bar{x})\|_\infty \le 1$.

(iii) and (iv): Finally, we compute an upper bound for the value $\langle Q_g(\bar{x}), c_i \rangle$ over the set SR^{Scaled}. It follows from $c_i \in \mathcal{K}$ and (7) that $\langle Q_g(e - \bar{x}), c_i \rangle \ge 0$, i.e., $\langle Q_g(e), c_i \rangle \ge \langle Q_g(\bar{x}), c_i \rangle$ holds. Since we have shown that $Q_g(e) = \xi \sum_{i \in H} c_i + \sum_{i \notin H} c_i$, this implies $\langle Q_g(\bar{x}), c_i \rangle \le \xi$ holds if $i \in H$ and $\langle Q_g(\bar{x}), c_i \rangle \le 1$ holds if $i \notin H$. $\square$

Note that Proposition 3.5 implies that if a cut is obtained for $\mathrm{P}_{S_\infty}(\mathcal{A})$ based on Proposition 3.3, we can expect a more efficient search for solutions to problem $\mathrm{P}_{S_\infty}(\mathcal{A}Q_g)$ rather than trying to solve problem $\mathrm{P}_{S_\infty}(\mathcal{A})$.

3.3 Non-simple symmetric cone case

In this section, we consider the case where the symmetric cone is not simple. Propositions 3.6 and 3.7 are extensions of Proposition 3.3 and 3.4, respectively.

Proposition 3.6. *Suppose that, for any $v \in \text{range}\,\mathcal{A}^*$, the ℓ-th block element v_ℓ of $v \in \mathbb{E}$ is decomposed into $v_\ell = \sum_{i=1}^{r_\ell} \lambda(v_\ell)_i c(v_\ell)_i$ as in Proposition 2.1. For each $\ell \in \{1,\ldots,p\}$ and $i \in \{1,\ldots,r_p\}$, define*

$$q_{\ell,i}(\alpha) := [1 - \alpha\lambda(v_\ell)_i]^+ + \sum_{k\neq i}^{r_\ell} [-\alpha\lambda(v_\ell)_k]^+ + \sum_{j\neq\ell}^{p} \sum_{k=1}^{r_j} [-\alpha\lambda(v_j)_k]^+. \tag{8}$$

Then, the following relations hold for any feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$, $\ell \in \{1,\ldots,p\}$ and $i \in \{1,\ldots,r_p\}$.

$$\langle c(v_\ell)_i, x_\ell \rangle \leq \min_{\alpha\in\mathbb{R}} q_{\ell,i}(\alpha) = \begin{cases} \min\left\{1, \left\langle e, \mathcal{P}_\mathcal{K}\left(-\frac{1}{\lambda(v_\ell)_i}v\right)\right\rangle\right\} & \text{if } \lambda(v_\ell)_i \neq 0, \\ 1 & \text{if } \lambda(v_\ell)_i = 0 \end{cases}. \tag{9}$$

Proof. Let $c \in \mathbb{E}$ be an element whose ℓ-th block element is $c_\ell = c(v_\ell)_i$ and other block elements take 0. For any real number $\alpha \in \mathbb{R}$, Proposition 3.2 ensures that

$$\langle c(v_\ell)_i, x_\ell \rangle = \langle c, x \rangle \leq \langle \mathcal{P}_\mathcal{K}(c - \alpha v), e \rangle$$

$$= \langle \mathcal{P}_{\mathcal{K}_\ell}(c(v_\ell)_i - \alpha v_\ell), e_\ell \rangle + \sum_{j\neq\ell}^{p} \langle \mathcal{P}_{\mathcal{K}_j}(-\alpha v_j), e_j \rangle$$

$$= [1 - \alpha\lambda(v_\ell)_i]^+ + \sum_{k\neq i}^{r_\ell} [-\alpha\lambda(v_\ell)_k]^+ + \sum_{j\neq\ell}^{p} \sum_{k=1}^{r_j} [-\alpha\lambda(v_j)_k]^+ = q_{\ell,i}(\alpha). \tag{10}$$

We obtain (9) by following a similar argument to the one used in the proof of Proposition 3.3. $\qquad\square$

The next proposition follows similarly to Proposition 3.4, by noting that $\langle e, \mathcal{P}_\mathcal{K}(-v)\rangle = \lambda(v_\ell)_i\xi$ holds if $\lambda(v_\ell)_i > 0$ and that $\langle e, \mathcal{P}_\mathcal{K}(v)\rangle = -\lambda(v_\ell)_i\xi$ if $\lambda(v_\ell)_i < 0$.

Proposition 3.7. *Suppose that, for any $v \in \text{range}\,\mathcal{A}^*$, each ℓ-th block element v_ℓ of v is decomposed into $v_\ell = \sum_{i=1}^{r_\ell} \lambda(v_\ell)_i c(v_\ell)_i$ as in Proposition 2.1. If v satisfies*

$$\lambda(v_\ell)_i \neq 0 \quad \text{and} \quad \left\langle e, \mathcal{P}_\mathcal{K}\left(-\frac{1}{\lambda(v_\ell)_i}v\right)\right\rangle = \xi_\ell < 1 \tag{11}$$

for some $\xi < 1$, $\ell \in \{1,\ldots,p\}$ and $i \in \{1,\ldots,r_\ell\}$, then $\lambda(v_\ell)_i$ has the same sign as $\langle e, v\rangle$.

From Proposition 3.6, if we obtain $v \in \text{range}\,\mathcal{A}^*$ satisfying (11) for a block $\ell \in \{1,\ldots,p\}$ with an index $i \in \{1,\ldots r_\ell\}$, then the upper bound for the sum of the eigenvalues of any feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$ is reduced by $\langle e, x\rangle \leq r - 1 + \xi_\ell < r$. In this case, as described below, we can find a scaling such that the sum of eigenvalues of any feasible solution of $\mathrm{P}_{S_\infty}(\mathcal{A})$ is bounded by r. Let H_ℓ be the set of indices i satisfying (11) for each block ℓ. According to Proposition 3.5, set $g_\ell = \sqrt{\xi_\ell} \sum_{h\in H_\ell} c(v_\ell)_h + \sum_{h\notin H_\ell} c(v_\ell)_h$ and define the linear operator Q as follows:

$$Q_\ell := \begin{cases} Q_{g_\ell} & \text{if } |H_\ell| \neq 0, \\ I_\ell & \text{otherwise,} \end{cases}$$

$$Q(\mathbb{E}_1,\ldots,\mathbb{E}_p) := (Q_1(\mathbb{E}_1),\ldots,Q_p(\mathbb{E}_p)),$$

11

where I_ℓ is the identity operator of the Euclidean Jordan algebra $\mathbb{E}_\ell$ associated with the symmetric cone $\mathcal{K}_\ell$. From Proposition 3.5 and its proof, we can easily see that

$$Q_{g_\ell^{-1}}(c_i) = \frac{1}{\xi}c_i \ (i \in H_\ell), \quad Q_{g_\ell^{-1}}(c_i) = c_i \ (i \notin H_\ell), \tag{12}$$

and the sum of eigenvalues of any feasible solution of the scaled problem $\mathrm{P}_{S_\infty}(\mathcal{A}Q)$ is bounded by $\langle e, e \rangle = r = \sum_{\ell=1}^{p} r_\ell$.

4   Basic procedure of the extended method

4.1   Outline of the basic procedure

In this section, we describe the details of our basic procedure. First, we introduce our stopping criteria and explain how to update y^k when the the stopping criteria is not satisfied. Next, we show that the stopping criteria is satisfied within a finite number of iterations. Our stopping criteria is new and different from the ones used in [10, 13], while the method of updating y^k is similar to the one used in [10] or in the von Neumann scheme of [13]. Algorithm 1 is a full description of our basic procedure.

4.2   Termination conditions of the basic procedure

For $z^k = P_\mathcal{A}(y^k)$, $v^k = y^k - z^k$ and a given $\xi \in (0, 1)$, our basic procedure terminates when any of the following four cases occurs:

1. $z^k \in \operatorname{int} \mathcal{K}$ meaning that z^k is a solution of $\mathrm{P}(\mathcal{A})$,

2. $z^k = \mathbf{0}$ meaning that y^k is feasible for $\mathrm{D}(\mathcal{A})$,

3. $y^k - z^k \in \mathcal{K}$ and $y^k - z^k \neq \mathbf{0}$ meaning that $y^k - z^k$ is feasible for $\mathrm{D}(\mathcal{A})$, or

4. there exist $\ell \in \{1, \ldots, p\}$ and $i \in \{1, \ldots, r_\ell\}$ for which

$$\lambda(v_\ell^k)_i \neq 0 \quad \text{and} \quad \left\langle e, \mathcal{P}_\mathcal{K}\left(-\frac{1}{\lambda(v_\ell^k)_i}v^k\right)\right\rangle = \xi_\ell \leq \xi < 1, \tag{13}$$

   meaning that $\langle e, x \rangle < r$ holds for any feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$ (see Proposition 3.6).

Cases 1 and 2 are direct extensions of the cases in [3], while case 3 was proposed in [9, 10]. Case 3 helps us to determine the feasibility of $\mathrm{P}(\mathcal{A})$ efficiently, while we have to decompose $y^k - z^k$ for checking it. If the basic procedure ends with case 1, 2, or 3, the basic procedure returns a solution of $\mathrm{P}(\mathcal{A})$ or $\mathrm{D}(\mathcal{A})$ to the main algorithm. If the basic procedure ends with case 4, the basic procedure returns to the main algorithm p index sets $H_1, \ldots, H_p$ each of which consists of indices i satisfying (13) and the set of primitive idempotents $C_\ell = \{c(v_\ell^k)_1, \ldots, c(v_\ell^k)_{r_\ell}\}$ of v_ℓ^k for each ℓ.

4.3   Update of the basic procedure

The basic procedure updates $y^k \in \operatorname{int} \mathcal{K}$ with $\langle y^k, e \rangle = 1$ so as to reduce the value of $\|z^k\|_J$. The following proposition is essentially the same as Proposition 13 in [10], so we will omit its proof.

Proposition 4.1 (cf. Proposition 13, [10]). *For $y^k \in \operatorname{int} \mathcal{K}$ satisfying $\langle y^k, e \rangle = 1$, let $z^k = P_{\mathcal{A}}(y^k)$. If $z^k \notin \operatorname{int} \mathcal{K}$ and $z^k \neq \mathbf{0}$, then the following hold.*

1. *There exists $c \in \mathcal{K}$ such that $\langle c, z^k \rangle = \lambda_{\min}(z^k) \leq 0$, $\langle e, c \rangle = 1$ and $c \in \mathcal{K}$.*

2. *For the above c, suppose that $P_{\mathcal{A}}(c) \neq \mathbf{0}$ and define*

$$\alpha = \langle P_{\mathcal{A}}(c), P_{\mathcal{A}}(c) - z^k \rangle \| z^k - P_{\mathcal{A}}(c) \|_J^{-2}. \tag{14}$$

   Then, $y^{k+1} := \alpha y^k + (1 - \alpha)c$ satisfies $y^{k+1} \in \operatorname{int} \mathcal{K}$, $\|y^{k+1}\|_{1,\infty} \geq 1/p$, $\langle y^{k+1}, e \rangle = 1$, and $z^{k+1} := P_{\mathcal{A}}(y^{k+1})$ satisfies $\|z^{k+1}\|_J^{-2} \geq \|z^k\|_J^{-2} + 1$.

A method of accelerating the update of y^k is provided in [15]. For $\ell \in \{1, 2, \ldots, p\}$, let $I_\ell := \{i \in \{1, 2, \ldots, r_\ell\} \mid \lambda_i(z_\ell^k) \leq 0\}$ and set $N = \sum_{\ell=1}^p |I_\ell|$. Define the ℓ-th block element of $c \in \mathcal{K}$ as $c_\ell = \frac{1}{N} \sum_{i \in I_\ell} c(z_\ell^k)_i$. Using $P_{\mathcal{A}}(c)$, the acceleration method computes α by (14) so as to minimize the norm of z^{k+1} and update y by $y^{k+1} = \alpha y^k + (1 - \alpha)c$. We incorporate this method in the basic procedure of our computational experiment. As described in [13], we can also use the smooth perceptron scheme [17, 18] to update y^k in the basic procedure. As explained in the next section, using the smooth perceptron scheme significantly reduces the maximum number of iterations of the basic procedure. A detailed description of our basic procedure is given in Appendix A.

4.4 Finite termination of the basic procedure

In this section, we show that the basic procedure terminates in a finite number of iterations. To do so, we need to prove Lemma 4.2 and Proposition 4.3.

Lemma 4.2. *Let $(\mathbb{E}, \circ)$ be a Euclidean Jordan algebra with the corresponding symmetric cone $\mathcal{K}$ given by the Cartesian product of p simple symmetric cones. For any $x \in \mathbb{E}$ and $y \in \mathcal{K}$, $[\langle x, y \rangle]^+ \leq \langle \mathcal{P}_{\mathcal{K}}(x), y \rangle$ holds.*

Proof. Let $x \in \mathbb{E}$ and suppose that each ℓ-th block element x_ℓ of x is given by $x_\ell = \sum_{i=1}^{r_\ell} \lambda(x_\ell)_i c(x_\ell)_i$ as in Proposition 2.1. Then, we can see that

$$
\begin{aligned}
[\langle x, y \rangle]^+ &= \left[\sum_{\ell=1}^p \left\langle \sum_{i=1}^{r_\ell} \lambda(x_\ell)_i c(x_\ell)_i, y_\ell \right\rangle \right]^+ \\
&= \left[\sum_{\ell=1}^p \left(\sum_{i=1}^{r_\ell} \lambda(x_\ell)_i \langle c(x_\ell)_i, y_\ell \rangle \right) \right]^+ \\
&\leq \sum_{\ell=1}^p \sum_{i=1}^{r_\ell} [\lambda(x_\ell)_i \langle c(x_\ell)_i, y_\ell \rangle]^+ \\
&= \sum_{\ell=1}^p \sum_{i=1}^{r_\ell} [\lambda(x_\ell)_i]^+ \langle c(x_\ell)_i, y_\ell \rangle = \sum_{\ell=1}^p \left\langle \sum_{i=1}^{r_\ell} [\lambda(x_\ell)_i]^+ c(x_\ell)_i, y_\ell \right\rangle = \langle \mathcal{P}_{\mathcal{K}}(x), y \rangle .
\end{aligned}
$$

where the inequality follows from the fact that $c(x_\ell)_1, \ldots, c(x_\ell)_{r_\ell}$, and y_ℓ lie in $\mathcal{K}_\ell$. $\qquad \square$

Proposition 4.3. *For a given $y \in \mathcal{K}$, define $z = P_{\mathcal{A}}(y)$ and $v = y - z$. Suppose that $v \neq 0$ and each ℓ-th element v_ℓ is given by $v_\ell = \sum_{i=1}^{r_\ell} \lambda(v_\ell)_i c(v_\ell)_i$, as in Proposition 2.1. Then, for any $x \in F_{P_{S_\infty}(\mathcal{A})}$, $\ell \in \{1, \ldots, p\}$ and $i \in \{1, \ldots, r_\ell\}$,*

$$\langle c(v_\ell)_i, x_\ell \rangle \leq \min_\alpha q_{\ell,i}(\alpha) \leq \frac{1}{\langle y_\ell, c(v_\ell)_i \rangle} \|z\|_J \tag{15}$$

hold where $q_{\ell,i}(\alpha)$ is defined in (8).

13

Proof. The first inequality of (15) follows from (10) in the proof of Proposition 3.6. The second inequality is obtained by evaluating $q_{\ell,i}(\alpha)$ at $\alpha = \frac{1}{\langle y_\ell, c(v_\ell)_i\rangle}$, as follows:

$$q_{\ell,i}\left(\frac{1}{\langle y_\ell, c(v_\ell)_i\rangle}\right) = \left[1 - \frac{1}{\langle y_\ell, c(v_\ell)_i\rangle}\lambda(v_\ell)_i\right]^+ + \sum_{k\neq i}^{r_\ell}\left[-\frac{1}{\langle y_\ell, c(v_\ell)_i\rangle}\lambda(v_\ell)_k\right]^+ + \sum_{j\neq\ell}^{p}\sum_{k=1}^{r_j}\left[-\frac{1}{\langle y_\ell, c(v_\ell)_i\rangle}\lambda(v_j)_k\right]^+$$

$$= \left[1 - \frac{\langle y_\ell - z_\ell, c(v_\ell)_i\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+ + \sum_{k\neq i}^{r_\ell}\left[-\frac{\langle y_\ell - z_\ell, c(v_\ell)_k\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+ + \sum_{j\neq\ell}^{p}\sum_{k=1}^{r_j}\left[-\frac{\langle y_j - z_j, c(v_j)_k\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+$$

(since $\lambda(v_\ell)_i = \langle v_\ell, c(v_\ell)_i\rangle$ and $v_\ell = y_\ell - z_\ell$)

$$= \left[\frac{\langle z_\ell, c(v_\ell)_i\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+ + \sum_{k\neq i}^{r_\ell}\left[\frac{\langle z_\ell, c(v_\ell)_k\rangle - \langle y_\ell, c(v_\ell)_k\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+ + \sum_{j\neq\ell}^{p}\sum_{k=1}^{r_j}\left[\frac{\langle z_j, c(v_j)_k\rangle - \langle y_j, c(v_j)_k\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+$$

$$\leq \left[\frac{\langle z_\ell, c(v_\ell)_i\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+ + \sum_{k\neq i}^{r_\ell}\left[\frac{\langle z_\ell, c(v_\ell)_k\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+ + \sum_{j\neq\ell}^{p}\sum_{k=1}^{r_j}\left[\frac{\langle z_j, c(v_j)_k\rangle}{\langle y_\ell, c(v_\ell)_i\rangle}\right]^+$$

(since $y_\ell, c(v_\ell)_i \in \mathcal{K}_\ell$ and then $\langle y_\ell, c(v_\ell)_i\rangle \geq 0$)

$$= \frac{1}{\langle y_\ell, c(v_\ell)_i\rangle}\left(\sum_{k=1}^{r_\ell}[\langle z_\ell, c(v_\ell)_k\rangle]^+ + \sum_{j\neq\ell}^{p}\sum_{k=1}^{r_j}[\langle z_j, c(v_j)_k\rangle]^+\right)$$

$$\leq \frac{1}{\langle y_\ell, c(v_\ell)_i\rangle}\left(\sum_{k=1}^{r_\ell}\langle \mathcal{P}_{\mathcal{K}_\ell}(z_\ell), c(v_\ell)_k\rangle + \sum_{j\neq\ell}^{p}\sum_{k=1}^{r_j}\langle \mathcal{P}_{\mathcal{K}_j}(z_j), c(v_j)_k\rangle\right) \quad \text{(by Lemma 4.2)}$$

$$= \frac{1}{\langle y_\ell, c(v_\ell)_i\rangle}\left(\langle \mathcal{P}_{\mathcal{K}_\ell}(z_\ell), e_\ell\rangle + \sum_{j\neq\ell}^{p}\langle \mathcal{P}_{\mathcal{K}_j}(z_j), e_j\rangle\right)$$

$$= \frac{\langle \mathcal{P}_{\mathcal{K}}(z), e\rangle}{\langle y_\ell, c(v_\ell)_i\rangle} = \frac{\|\mathcal{P}_{\mathcal{K}}(z)\|_1}{\langle y_\ell, c(v_\ell)_i\rangle} \leq \frac{\|\mathcal{P}_{\mathcal{K}}(z)\|_J}{\langle y_\ell, c(v_\ell)_i\rangle} \leq \frac{\|z\|_J}{\langle y_\ell, c(v_\ell)_i\rangle}.$$

$\square$

Proposition 4.4. *Let $r_{\max} = \max\{r_1,\ldots,r_p\}$. The basic procedure (Algorithm 1) terminates in at most $\frac{p^2 r_{\max}^2}{\xi^2}$ iterations.*

Proof. Suppose that y^k is obtained at the k-th iteration of Algorithm 1. Proposition 4.1 implies that $\|y^k\|_{1,\infty} \geq \frac{1}{p}$ and an ℓ-th block element exists for which $\langle y_\ell, e_\ell\rangle \geq \frac{1}{p}$ holds. Thus, by letting $v^k = y^k - z^k$ and the ℓ-th block element v_ℓ^k of v^k be $v_\ell^k = \sum_{i=1}^{r_\ell}\lambda(v_\ell^k)_i c(v_\ell^k)_i$ as in Proposition 2.1, we have

$$\max_{i=1,\ldots,r_\ell}\langle y_\ell^k, c(v_\ell^k)_i\rangle \geq (pr_\ell)^{-1}. \tag{16}$$

Since Proposition 4.1 ensures that $\frac{1}{\|z^k\|_J^2} \geq k$ holds at the k-th iteration, by setting $k = \frac{p^2 r_{\max}^2}{\xi^2}$, we see that $\xi \geq pr_{\max}\|z^k\|_J$, and combining this with (16), we have

$$\xi \geq pr_{\max}\|z^k\|_J \geq pr_\ell\|z^k\|_J \geq \frac{1}{\max_{i=1,\ldots,r_\ell}\langle y_\ell^k, c(v_\ell)_i\rangle}\|z^k\|_J.$$

The above inequality and Proposition 4.3 imply that for any $\ell \in \{1,\ldots,p\}$ and $i \in \{1,\ldots,r_p\}$,

$$\langle c(v_\ell^k)_i, x_\ell\rangle \leq \min_{\alpha} q_{\ell,i}(\alpha) \leq \frac{1}{\langle y_\ell^k, c(v_\ell^k)_i\rangle}\|z^k\|_J \leq \xi.$$

14

From the equivalence in (9) and the setting $\xi \in (0,1)$, we conclude that Algorithm 1 terminates in at most $\frac{p^2 r_{\max}^2}{\xi^2}$ iterations by satisfying (13) in the fourth termination condition at an ℓ-th block and an index i. $\qquad\square$

An upper bound for the number of iterations of Algorithm 5 using smooth perceptoron scheme can be found as follows.

Proposition 4.5. *Let* $r_{\max} = \max\{r_1, \ldots, r_p\}$. *The basic procedure (Algorithm 5) terminates in at most* $\frac{2\sqrt{2}pr_{\max}}{\xi}$ *iterations.*

Proof. From Proposition 6 in [13], after $k \geq 1$ iterations, we obtain the inequality $\|z^k\|_J^2 \leq \frac{8}{(k+1)^2}$. Similarly to the previous proof of Proposition 4.4, if $\xi \geq pr_{max}\|z^k\|_J$ holds, then Algorithm 5 terminates. Thus, $k \leq \frac{2\sqrt{2}pr_{\max}}{\xi}$ holds for a given k satisfying $\left(\frac{\xi}{pr_{max}}\right)^2 \leq \frac{8}{(k+1)^2}$. $\qquad\square$

Here, we discuss the computational cost per iteration of Algorithm 1. At each iteration, the two most expensive operations are computing the spectral decomposition on line 5 and computing $P_{\mathcal{A}}(\cdot)$ on lines 24 and 26. Let C_ℓ^{sd} be the computational cost of the spectral decomposition of an element of $\mathcal{K}_\ell$. For example, $C_\ell^{\mathrm{sd}} = \mathcal{O}(r_\ell^3)$ if $\mathcal{K}_\ell = \mathbb{S}_+^{r_\ell}$ and $C_\ell^{\mathrm{sd}} = \mathcal{O}(r_\ell)$ if $\mathcal{K}_\ell = \mathbb{L}_{r_\ell}$, where $\mathbb{L}_{r_\ell}$ denotes the r_ℓ-dimensional second-order cone. Then, the cost C^{sd} of computing the spectral decomposition of an element of $\mathcal{K}$ is $C^{\mathrm{sd}} = \sum_{\ell=1}^p C_\ell^{\mathrm{sd}}$. Next, let us consider the computational cost of $P_{\mathcal{A}}(\cdot)$. Recall that d is the dimension of the Euclidean space $\mathbb{E}$ corresponding to $\mathcal{K}$. As discussed in [10], we can compute $P_{\mathcal{A}} = I - \mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}$ by using the Cholesky decomposition of $(\mathcal{A}\mathcal{A}^*)^{-1}$. Suppose that $(\mathcal{A}\mathcal{A}^*)^{-1} = LL^*$, where L is an $m \times m$ matrix and we store $L^*\mathcal{A}$ in the main algorithm. Then, we can compute $P_{\mathcal{A}}(\cdot)$ on lines 24 and 26, which costs $\mathcal{O}(md)$. The operation $u_\mu(\cdot) : \mathbb{E} \to \{u \in \mathcal{K} \mid \langle u, e \rangle = 1\}$ in Algorithm 5 can be performed within the cost C^{sd} [18, 13]. From the above discussion and Proposition 4.4, the total costs of Algorithm 1 and Algorithm 5 are given by $\mathcal{O}\left(\frac{p^2 r_{\max}^2}{\xi^2}\max(C^{\mathrm{sd}}, md)\right)$ and $\mathcal{O}\left(\frac{pr_{\max}}{\xi}\max(C^{\mathrm{sd}}, md)\right)$, respectively.

Algorithm 1 Basic procedure (von Neumann scheme)
___
1: **Input:** $P_\mathcal{A}$, $y^1 \in \text{int } \mathcal{K}$ such that $\langle y^1, e \rangle = 1$ and a constant ξ such that $0 < \xi < 1$

2: **Output:** (i) a solution to P($\mathcal{A}$) or (ii) D($\mathcal{A}$) or (iii) a certificate that, for any feasible solution x to P$_{S_\infty}$($\mathcal{A}$), $\langle e, x \rangle < r$

3: initialization: $k \leftarrow 1, z^1 \leftarrow P_\mathcal{A}(y^1), v^1 \leftarrow y^1 - z^1, H_1, \dots, H_p = \emptyset$

4: **while** $k \le \frac{p^2 r_{\max}^2}{\xi^2}$ **do**

5:     For every $\ell \in \{1, \dots, p\}$, perform spectral decomposition: $z_\ell^k = \sum_{i=1}^{r_\ell} \lambda(z_\ell^k)_i c(z_\ell^k)_i$ and $v_\ell^k = \sum_{i=1}^{r_\ell} \lambda(v_\ell^k)_i c(v_\ell^k)_i$

6:     **if** $z^k \in \text{int } \mathcal{K}$ **then**

7:         **stop** basic procedure and **return** z^k (Output (i))

8:     **else if** $z^k = 0$ or $v^k \in \mathcal{K} \setminus \{0\}$ **then**

9:         **stop** basic procedure and **return** y^k or v^k (Output (ii))

10:     **end if**

11:     **if** $\langle v^k, e \rangle > 0$ **then**

12:         **for** $\ell \in \{1, \dots, p\}$ **do**

13:             $I_\ell \leftarrow \{i \mid \lambda(v_\ell^k)_i > 0\}$ and then $H_\ell \leftarrow \left\{ i \in I_\ell \mid \left\langle e, \mathcal{P}_\mathcal{K} \left(-\frac{1}{\lambda(v_\ell^k)_i} v^k \right) \right\rangle \le \xi \right\}$

14:         **end for**

15:     **else**

16:         **for** $\ell \in \{1, \dots, p\}$ **do**

17:             $I_\ell \leftarrow \{i \mid \lambda(v_\ell^k)_i < 0\}$ and then $H_\ell \leftarrow \left\{ i \in I_\ell \mid \left\langle e, \mathcal{P}_\mathcal{K} \left(-\frac{1}{\lambda(v_\ell^k)_i} v^k \right) \right\rangle \le \xi \right\}$

18:         **end for**

19:     **end if**

20:     **if** $|H_1| + \dots + |H_p| > 0$ **then**

21:         For every $\ell \in \{1, \dots, p\}$, let C_ℓ be $\{c(v_\ell^k)_1, \dots, c(v_\ell^k)_{r_\ell}\}$.

22:         **stop** basic procedure and **return** $H_1, \dots, H_p$ and $C_1, \dots, C_p$ (Output (iii))

23:     **end if**

24:     Let u be an idempotent such that $\langle e, u \rangle = 1$ and $\langle z^k, u \rangle = \lambda_{\min}(z^k)$

25:     $y^{k+1} \leftarrow \alpha y^k + (1-\alpha)u$, where $\alpha = \frac{\langle P_\mathcal{A}(u), P_\mathcal{A}(u) - z^k \rangle}{\|z^k - P_\mathcal{A}(u)\|_J^2}$

26:     $k \leftarrow k+1$, $z^k \leftarrow P_\mathcal{A}(y^k)$ and $v^k \leftarrow y^k - z^k$

27: **end while**

28: **return** basic procedure error
___

5 Main algorithm of the extended method

5.1 Outline of the main algorithm

In what follows, for a given accuracy $\varepsilon > 0$, we call a feasible solution of P$_{S_\infty}$($\mathcal{A}$) whose minimum eigenvalue is ε or more an <u>ε-feasible solution of P$_{S_\infty}$($\mathcal{A}$)</u>. This section describes the main algorithm. To set the upper bound for the minimum eigenvalue of any feasible solution x of P$_{S_\infty}$($\mathcal{A}$), Algorithm 2 focuses on the product $\det(\bar{x})$ of the eigenvalues of the arbitrary feasible solution $\bar{x}$ of the scaled problem P$_{S_\infty}$($\mathcal{A}^k Q^k$). Algorithm 2 works as follows. First, we calculate the corresponding projection $P_\mathcal{A}$ onto $\ker \mathcal{A}$ and generate an initial point as input to the basic procedure. Next, we call the basic procedure and determine whether to end the algorithm with an ε-feasible solution or to perform problem scaling according to the returned result, as follows:

1. If a feasible solution of P($\mathcal{A}$) or D($\mathcal{A}$) is returned from the basic procedure, the feasibility of P($\mathcal{A}$) can be determined, and we stop the main algorithm.

2. If the basic procedure returns the sets of indices $H_1, \ldots, H_p$ and the sets of primitive idempotents $C_1, \ldots, C_p$ that construct the corresponding Jordan frames, then for the total number of cuts obtaibed in the ℓ-th block num_ℓ,

    (a) if $\mathrm{num}_\ell \geq r_\ell \frac{\log \varepsilon}{\log \xi}$ holds for some $\ell \in \{1, \ldots p\}$, we determine that $\mathrm{P}_{S_\infty}(\mathcal{A})$ has no ε-feasible solution according to Proposition 5.1 and stop the main algorithm,

    (b) if $\mathrm{num}_\ell < r_\ell \frac{\log \varepsilon}{\log \xi}$ holds for any $\ell \in \{1, \ldots p\}$, we rescale the problem and call the basic procedure again.

Note that our main algorithm is similar to Lourenço et al.'s method in the sense that it keeps information about the possible minimum eigenvalue of any feasible solution of the problem. In contrast, Pena and Soheili's method [13] does not keep such information. Algorithm 2 terminates after no more than $-\frac{r}{\log \xi} \log\left(\frac{1}{\varepsilon}\right) - p + 1$ iterations, so our main algorithm can be said to be a polynomial-time algorithm. We will give this proof in section 5.2. We should also mention that step 24 in Algorithm 2 is not a reachable output theoretically. We have added this step in order to consider the influence of the numerical error in practice.

Algorithm 2 Main algorithm

1: **Input:** $\mathcal{A}$, $\mathcal{K}$, ε and a constant ξ such that $0 < \xi < 1$
2: **Output:** a solution to $\mathrm{P}(\mathcal{A})$ or $\mathrm{D}(\mathcal{A})$ or a certificate that there is no ε feasible solution.
3: $k \leftarrow 1$, $\mathcal{A}^1 \leftarrow \mathcal{A}$, $\mathrm{num}_\ell \leftarrow 0$, $\bar{Q}_\ell \leftarrow I_\ell$, $RP_\ell \leftarrow I_\ell$, $RD_\ell \leftarrow I_\ell$ for all $\ell \in \{1, \ldots, p\}$
4: Compute $P_{\mathcal{A}^k}$ and call the basic procedure with $P_{\mathcal{A}^k}$, $\frac{1}{r}e$, ξ
5: **if** basic procedure returns z **then**
6:     **stop** main algorithm and **return** RPz (RPz is a feasible solution of $\mathrm{P}(\mathcal{A})$)
7: **else if** basic procedure returns y or v **then**
8:     **stop** main algorithm and **return** RDy or RDv (RDy or RDv is a feasible solution of $D(\mathcal{A})$)
9: **else if** basic procedure returns $H_1^k, \ldots, H_p^k$ and $C_1^k, \ldots, C_p^k$ **then**
10:     **for** $\ell \in \{1, \ldots, p\}$ **do**
11:       **if** $|H_\ell^k| > 0$ **then**
12:         $g_\ell \leftarrow \sqrt{\xi} \sum_{h \in H_\ell^k} c^k(v_\ell)_h + \sum_{h \notin H_\ell^k}^{r_\ell} c^k(v_\ell)_h$
13:         $Q_\ell \leftarrow Q_{g_\ell}$, $RP_\ell \leftarrow RP_\ell Q_{g_\ell}$, $RD_\ell \leftarrow RD_\ell Q_{g_\ell^{-1}}$
14:         $\mathrm{num}_\ell \leftarrow |H_\ell^k| + \mathrm{num}_\ell$
15:         **if** $\mathrm{num}_\ell \geq r_\ell \frac{\log \varepsilon}{\log \xi}$ **then**
16:           **stop** main algorithm. There is no ε feasible solution.
17:         **end if**
18:         $\bar{Q}_\ell \leftarrow Q_{g_\ell^{-1}} \bar{Q}_\ell$
19:       **else**
20:         $Q_\ell \leftarrow I_\ell$
21:       **end if**
22:     **end for**
23: **else**
24:     **return** basic procedure error
25: **end if**
26: Let $Q^k = (Q_1, \ldots, Q_p)$
27: $\mathcal{A}^{k+1} \leftarrow \mathcal{A}^k Q^k$, $k \leftarrow k + 1$. Go back to line 4.

5.2 Finite termination of the main algorithm

Here, we discuss how many iterations are required until we can determine that the minimum eigenvalue $\lambda_{\min}(x)$ is less than ε for any $x \in F_{\mathrm{P}_{S_\infty}(\mathcal{A})}$. Before going into the proof, we explain the Algorithm 2 in

more detail than in section 5.1. At each iteration of Algorithm 2, it accumulates the number of cuts $|H_\ell^k|$ obtained in the ℓ-th block and stores the value in num_ℓ. Using num_ℓ, we can compute an upper bound for $\lambda_{\min}(x)$ (Proposition 5.1). On line 18, $\bar{Q}_\ell$ is updated to $\bar{Q}_\ell \leftarrow Q_{g_\ell^{-1}}\bar{Q}_\ell$, where $\bar{Q}_\ell$ plays the role of an operator that gives the relation $\bar{x}_\ell = \bar{Q}_\ell(x_\ell)$ for the solution x of the original problem and the solution $\bar{x}$ of the scaled problem. For example, if $|H_\ell^1| > 0$ for $k = 1$ (suppose that the cut was obtained in the ℓ-th block), then the proposed method scales $\mathcal{A}_\ell^1 Q_\ell^1$ and the problem to yield $\bar{x}_\ell = Q_{g_\ell^{-1}}(x_\ell)$ for the feasible solution x of the original problem. And if $|H_\ell^2| > 0$ even for $k = 2$, then the proposed method scales $\bar{x}$ again, so that $\bar{\bar{x}}_\ell = Q_{g_\ell^{-1}}(\bar{x}_\ell) = \bar{Q}_\ell(x_\ell)$ holds. Note that $\bar{Q}_\ell$ is used only for a concise proof of Proposition 5.1, so it is not essential.

Now, let us derive an upper bound for the minimum eigenvalue $\lambda_{\min}(x_\ell)$ of each ℓ-th block of x obtained after the k-th iteration of Algorithm 2. Proposition 5.2 gives an upper bound for the number of iterations of Algorithm 2.

Proposition 5.1. *After k iterations of Algorithm 2, for any feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$ and $\ell \in \{1,\ldots,p\}$, the ℓ-th block element x_ℓ of x satisfies*

$$r_\ell \log\left(\lambda_{\min}(x_\ell)\right) \le \text{num}_\ell \log \xi. \tag{17}$$

Proof. At the end of the k-th iteration, any feasible solution $\bar{x}$ of the scaled problem $\mathrm{P}_{S_\infty}(\mathcal{A}^{k+1}) = \mathrm{P}_{S_\infty}(\mathcal{A}^k Q^k)$ obviously satisfies

$$\det \bar{x}_\ell \le \det e_\ell \quad (\ell = 1, 2, \ldots, p). \tag{18}$$

Note that $\det \bar{x}_\ell$ can be expressed in terms of $\det x_\ell$. For example, if $|H_\ell^1| > 0$ when $k = 1$, then using Proposition 2.5, for any feasible solution $\bar{x}$ of $\mathrm{P}_{S_\infty}(\mathcal{A}^2)$, we find that

$$\det \bar{x}_\ell = \det Q_{g_\ell^{-1}}(x_\ell) = \det(g_\ell^{-1})^2 \det x_\ell = \left(\frac{1}{\sqrt{\xi}}\right)^{2|H_\ell^1|} \det x_\ell = \left(\frac{1}{\xi}\right)^{|H_\ell^1|} \det x_\ell.$$

This means that $\det \bar{x}_\ell$ can be determined from $\det x_\ell$ and the number of cuts obtained so far in the ℓ-th block. In Algorithm 2, the value of num_ℓ is updated only when $|H_\ell^k| > 0$. Since $\bar{x}$ satisfies $\bar{x}_\ell = \bar{Q}_\ell(x_\ell)$ $(\ell = 1, 2, \ldots, p)$ for each feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$, we can see that

$$\det \bar{x}_\ell = \det \bar{Q}_\ell(x_\ell) = \left(\frac{1}{\xi}\right)^{|H_\ell^k|} \times \left(\frac{1}{\xi}\right)^{|H_\ell^{k-1}|} \cdots \times \left(\frac{1}{\xi}\right)^{|H_\ell^1|} \times \det x_\ell = \left(\frac{1}{\xi}\right)^{\text{num}_\ell} \det x_\ell.$$

Therefore, (18) implies $\det x_\ell \le \xi^{\text{num}_\ell} \det e_\ell = \xi^{\text{num}_\ell}$ and the fact $(\lambda_{\min}(x_\ell))^{r_\ell} \le \det x_\ell$ implies $(\lambda_{\min}(x_\ell))^{r_\ell} \le \xi^{\text{num}_\ell}$. By taking the logarithm of both sides of this inequality, we obtain (17). $\square$

Proposition 5.2. *Algorithm 2 terminates after no more than $-\frac{r}{\log \xi} \log\left(\frac{1}{\varepsilon}\right) - p + 1$ iterations.*

Proof. Let us call iteration k of Algorithm 2 <u>good</u> if $|H_\ell^k| > 0$ for some $\ell \in \{1, 2, \ldots, p\}$ at that iteration. Suppose that at least $-\frac{r_\ell}{\log \xi} \log\left(\frac{1}{\varepsilon}\right)$ <u>good</u> iterations are observed for a cone $\mathcal{K}_\ell$. Then, by substituting $-\frac{r_\ell}{\log \xi} \log\left(\frac{1}{\varepsilon}\right)$ into num_ℓ of inequality (17) in Proposition 5.1, we have $\log\left(\lambda_{\min}(x_\ell)\right) \le \log \varepsilon$ and hence, $\lambda_{\min}(x_\ell) \le \varepsilon$. This implies that Algorithm 2 terminates after no more than

$$\sum_{\ell=1}^p \left(-\frac{r_\ell}{\log \xi} \log\left(\frac{1}{\varepsilon}\right) - 1\right) + 1 = -\frac{r}{\log \xi} \log\left(\frac{1}{\varepsilon}\right) - p + 1$$

iterations. $\square$

6 Computational costs of the algorithms

This section compares the computational costs of Algorithm 2, Lourenço et al.'s method [10] and Pena and Soheili's method [13]. Section 6.1 compares the computational costs of Algorithm 2 and Lourenço et al.'s method, and Section 6.2 compares those of Algorithm 2 and Pena and Soheili's method under the assumption that $\ker \mathcal{A} \cap \operatorname{int} \mathcal{K} \neq \emptyset$.

Both the proposed method and the method of Lourenço et al. guarantee finite termination of the main algorithm by termination criteria indicating the nonexistence of an ε-feasible solution, so that it is possible to compare the computational costs of the methods without making any special assumptions. This is because both methods proceed by making cuts to the feasible region using the results obtained from the basic procedure. On the other hand, Pena and Soheili's method cannot be simply compared because the upper bound of the number of iterations of their main algorithm includes an unknown value of $\delta(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K}) := \max_x \left\{ \det(x) \mid x \in \ker \mathcal{A} \cap \operatorname{int} \mathcal{K}, \|x\|_J^2 = r \right\}$. However, by making the assumption $\ker \mathcal{A} \cap \operatorname{int} \mathcal{K} \neq \emptyset$ and deriving a lower bound for $\delta(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K})$, we make it possible to compare Algorithm 2 with Pena and Soheili's method without knowing the specific values of $\delta(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K})$.

6.1 Comparison of Algorithm 2 and Lourenço et al.'s method

Let us consider the computational cost of Algorithm 2. At each iteration, the most expensive operation is computing $P_{\mathcal{A}}$ on line 4. Recall that d is the dimension of the Euclidean space $\mathbb{E}$ corresponding to $\mathcal{K}$. As discussed in [10], by considering $P_{\mathcal{A}}$ to be an $m \times d$ matrix, we find that the computational cost of $P_{\mathcal{A}}$ is $\mathcal{O}(m^3 + m^2 d)$. Therefore, by taking the computational cost of the basic procedure (Algorithm 1) and Proposition 5.2 into consideration, the cost of Algorithm 2 turns out to be

$$\mathcal{O}\left(-\frac{r}{\log \xi} \log\left(\frac{1}{\varepsilon}\right) \left(m^3 + m^2 d + \frac{1}{\xi^2} p^2 r_{\max}^2 \left(\max\left(C^{\mathrm{sd}}, md \right) \right) \right) \right) \tag{19}$$

where C^{sd} is the computational cost of the spectral decomposition of $x \in \mathbb{E}$.

Note that, in [10], the authors showed that the cost of their algorithm is

$$\mathcal{O}\left(\left(\frac{r}{\varphi(\rho)} \log\left(\frac{1}{\varepsilon}\right) - \sum_{i=1}^{p} \frac{r_i \log(r_i)}{\varphi(\rho)} \right) \left(m^3 + m^2 d + \rho^2 p^3 r_{max}^2 \left(\max\left(C^{\mathrm{min}}, md \right) \right) \right) \right) \tag{20}$$

where C^{min} is the cost of computing the minimum eigenvalue of $x \in \mathbb{E}$ with the corresponding idempotent, ρ is an input parameter used in their basic procedure (like ξ in the proposed algorithm) and $\varphi(\rho) = 2 - 1/\rho - \sqrt{3 - 2/\rho}$.

When the symmetric cone is simple, by setting $\xi = 1/2$ and $\rho = 2$, the maximum number of iterations of the basic procedure is bounded by the same value in both algorithms. Accordingly, we will compare the two computational costs (19) and (20) by supposing $\xi = 1/2$ and $\rho = 2$ (hence, $-\log \xi \simeq 0.69$ and $\varphi(\rho) \simeq 0.09$). As we can see below, the cost (19) of our method is smaller than (20) in the cases of linear programming and second-order cone problems and is equivalent to (20) in the case of semidefinite problems. First, let us consider the case where $\mathcal{K}$ is the n-dimensional nonnegative orthant $\mathbb{R}_+^n$. Here, we see that $r = p = d = n$, $r_1 = \cdots = r_p = r_{\max} = 1$, and $\max\left(C^{\mathrm{sd}}, md \right) = \max\left(C^{\mathrm{min}}, md \right) = md$ hold. By substituting these values, the bounds (19) and (20) turn out to be

$$\mathcal{O}\left(\frac{n}{0.69} \log\left(\frac{1}{\varepsilon}\right) \left(m^3 + m^2 n + 4mn^3 \right) \right)$$

and

$$\mathcal{O}\left(\frac{n}{0.09} \log\left(\frac{1}{\varepsilon}\right) \left(m^3 + m^2 n + 4mn^4 \right) \right).$$

19

This implies that for the linear programming case, our method (which is equivalent to Roos's original method [15]) is superior to Lourenço et al.'s method [10] in terms of bounds (19) and (20) .

Next, let us consider the case where $\mathcal{K}$ is composed of p simple second-order cones $\mathbb{L}^{n_i}$ $(i = 1, \ldots, p)$. In this case, we see that $d = \sum_{i=1}^{p} n_i$, $r_1 = \cdots = r_p = r_{\max} = 2$ and $\max\left(C^{\mathrm{sd}}, md\right) = \max\left(C^{\min}, md\right) = md$ hold. By substituting these values, the bounds (19) and (20) turn out to be

$$\mathcal{O}\left(\frac{2p}{0.69} \log\left(\frac{1}{\varepsilon}\right)\left(m^3 + m^2 d + 16p^2 md\right)\right)$$

and

$$\mathcal{O}\left(\frac{2p}{0.09}\left(\log\left(\frac{1}{\varepsilon}\right) - \log 2\right)\left(m^3 + m^2 d + 16p^3 md\right)\right).$$

Note that ε is expected to be very small (10^{-6} or even 10^{-12} in practice) and $\frac{1}{0.69}\log\left(\frac{1}{\varepsilon}\right) \leq \frac{1}{0.09}\left(\log\left(\frac{1}{\varepsilon}\right) - \log 2\right)$ if $\varepsilon \leq 0.451$. Thus, even in this case, we may conclude that our method is superior to Lourenço et al.'s method in terms of the bounds (19) and (20) .

Finally, let us consider the case where $\mathcal{K}$ is a simple $n \times n$ positive semidefinite cone. We see that $p = 1$, $r = n$, and $d = \frac{n(n+1)}{2}$ hold, and upon substituting these values, the bounds (19) and (20) turn out to be

$$\mathcal{O}\left(\frac{n}{0.69} \log\left(\frac{1}{\varepsilon}\right)\left(m^3 + m^2 n^2 + 4n^2 \max\left(C^{\mathrm{sd}}, mn^2\right)\right)\right)$$

and

$$\mathcal{O}\left(\frac{n}{0.09} \log\left(\frac{1}{\varepsilon}\right)\left(m^3 + m^2 n^2 + 4n^2 \max(C^{\min}, mn^2)\right)\right).$$

From the discussion in Section 6.3, we can assume $\mathcal{O}\left(C^{\mathrm{sd}}\right) = \mathcal{O}\left(C^{\min}\right)$, and the computational bounds of two methods are equivalent.

6.2 Comparison of Algorithm 2 and Pena and Soheili's method

In this section, we assume that $\mathcal{K}$ is simple since [13] has presented an algorithm for the simple form. We also assume that $\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K} \neq \emptyset$, because Pena and Soheili's method does not terminate if $\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K} = \emptyset$ and $\mathrm{range}\,\mathcal{A}^* \cap \mathrm{int}\,\mathcal{K} = \emptyset$. Furthermore, for the sake of simplicity, we assume that the main algorithm of Pena and Soheili's method applies only to $\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}$. (Their original method applies the main algorithm to $\mathrm{range}\,\mathcal{A}^* \cap \mathrm{int}\,\mathcal{K}$ as well.)

First, we will briefly explain the idea of deriving an upper bound for the number of iterations required to find $x \in \ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}$ in Pena and Soheili's method. Pena and Soheili derive it by focusing on the indicator $\delta(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}) := \max_x \left\{\det(x) \mid x \in \ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}, \|x\|_J^2 = r\right\}$. If $\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K} \neq \emptyset$, then $\delta(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}) \in (0, 1]$ holds, and if $e \in \ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}$, then $\delta(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}) = 1$ holds. If $e \in \ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}$, then the basic procedure terminates immediately and returns $\frac{1}{r}e$ as a feasible solution. Then, they prove that $\delta(Q_v(\ker\mathcal{A}) \cap \mathrm{int}\,\mathcal{K}) \geq 1.5 \cdot \delta(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K})$ holds if the parameters are appropriately set, and derive an upper bound on the number of scaling steps, i.e., the number of iterations, required to obtain $\delta(Q_v(\ker\mathcal{A}) \cap \mathrm{int}\,\mathcal{K}) = 1$.

In the following, we obtain an upper bound for the number of iterations of Algorithm 2 using the index $\delta^{\mathrm{supposed}}(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}) := \max_x \left\{\det(x) \mid x \in \ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}, \|x\|_J^2 = 1\right\}$. Note that $\delta(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}) = r^{\frac{r}{2}} \cdot \delta^{\mathrm{supposed}}(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K})$. In fact, if x^* is the point giving the maximum value of $\delta^{\mathrm{supposed}}(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K})$, then the point giving the maximum value of $\delta(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K})$ is $\sqrt{r}x^*$. Also, if $\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K} \neq \emptyset$, then $\delta^{\mathrm{supposed}}(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}) \in (0, 1/r^{\frac{r}{2}}]$, and if $\frac{1}{\sqrt{r}}e \in \ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}$, then $\delta^{\mathrm{supposed}}(\ker\mathcal{A} \cap \mathrm{int}\,\mathcal{K}) = 1/r^{\frac{r}{2}}$.

The outline of this section is as follows: First, we show that a lower bound for $\delta^{\text{supposed}}\left(\ker\mathcal{A} \cap \operatorname{int}\mathcal{K}\right)$ can be derived using the index value $\delta^{\text{supposed}}\left(Q_{g^{-1}}\left(\ker\mathcal{A}\right) \cap \operatorname{int}\mathcal{K}\right)$ for the problem after scaling (Proposition 6.5). Then, using this result, we derive an upper bound for the number of operations required to obtain $\delta^{\text{supposed}}\left(Q_{g^{-1}}\left(\ker\mathcal{A}\right) \cap \operatorname{int}\mathcal{K}\right) = 1/r^{\frac{r}{2}}$ (Proposition 6.6). Finally, we compare the proposed method with Pena and Soheili's method. To prove Proposition 6.3 used in the proof of Proposition 6.5, we use the following propositions from [8].

Proposition 6.1 (Theorem 3 of [8]). *Let $c \in \mathbb{E}$ be an idempotent and $N_\lambda(c)$ be the set such that $N_\lambda(c) = \{x \in \mathbb{E} \mid c \circ x = \lambda x\}$. Then $N_\lambda(c)$ is a linear maniforld, but if $\lambda \neq 0, \frac{1}{2}$, and 1, then $N_\lambda(c)$ consists of zero alone. Each $x \in \mathbb{E}$ can be represented in the form*

$$x = u + v + w, \quad u \in N_0(c), \quad v \in N_{\frac{1}{2}}(c), \quad w \in N_1(c),$$

in one and only one way.

Proposition 6.2 (Theorem 11 of [8]). *$c \in \mathbb{E}$ is a primitive idempotent if and only if $N_1(c) = \{x \in \mathbb{E} \mid c \circ x = x\} = \mathbb{R}c$.*

Proposition 6.3. *Let $c \in \mathbb{E}$ be a primitive idempotent. Then, for any $x \in \mathbb{E}$, $\langle x, Q_c(x)\rangle = \langle x, c\rangle^2$ holds.*

Proof. From Propositions 6.1 and 6.2, for any $x \in \mathbb{E}$, there exist a real number $\lambda \in \mathbb{R}$ and elements $u \in N_0(c)$ and $v \in N_{\frac{1}{2}}(c)$ such that $x = u + v + \lambda c$.

First, we show that $\langle x, c\rangle = \lambda$. For $v \in N_{\frac{1}{2}}(c)$, we see that $\langle v, c\rangle = \langle v, c \circ c\rangle = \langle v \circ c, c\rangle = \langle c \circ v, c\rangle = \frac{1}{2}\langle v, c\rangle$, which implies that $\langle v, c\rangle = 0$. Thus, since $u \in N_0(c)$ and $u \circ c = 0$, $\langle x, c\rangle$ is given by

$$\langle x, c\rangle = \langle u + v + \lambda c, c\rangle = \langle u, c\rangle + \langle v, c\rangle = \lambda\langle c, c\rangle = 0 + 0 + \lambda.$$

On the other hand, by using the facts $x = u + v + \lambda c$, $c^2 = c$, $c \circ u = 0$ and $c \circ v = \frac{1}{2}v$ repeatedly, we have

$$\begin{aligned}
\langle x, Q_c(x)\rangle &= \langle x, 2c \circ (c \circ x) - c^2 \circ x\rangle \\
&= \langle x, 2c \circ (c \circ (u + v + \lambda c)) - c \circ (u + v + \lambda c)\rangle \\
&= \langle x, 2c \circ (\frac{1}{2}v + \lambda c) - (\frac{1}{2}v + \lambda c)\rangle \\
&= \langle x, (\frac{1}{2}v + 2\lambda c) - (\frac{1}{2}v + \lambda c)\rangle = \langle x, \lambda c\rangle = \lambda^2.
\end{aligned}$$

Thus, we have shown that $\langle x, Q_c(x)\rangle = \langle x, c\rangle^2$ holds. $\qquad\square$

Remark 6.4. *It should be noted that the proof of Proposition 3 in [13] is not correct since equation (14) does not necessarily hold. The above Proposition 6.3 also gives a correct proof of Proposition 3 in [13]. See the computation $\langle y, Q_{g^{-2}}(y)\rangle$ in the proof of Proposition 6.5.*

Proposition 6.5. *Suppose that $\ker\mathcal{A} \cap \operatorname{int}\mathcal{K} \neq \emptyset$ and that, for a given nonempty index set $H \subseteq \{1, \ldots r\}$, Jordan frame $c_1, \ldots, c_r$, and $0 < \xi < 1$,*

$$\langle c_i, x\rangle \leq \xi \ (i \in H), \quad \langle c_i, x\rangle \leq 1 \ (i \notin H)$$

holds for any $x \in F_{\mathrm{P}_{S_\infty}(\mathcal{A})}$. Define $g \in \operatorname{int}\mathcal{K}$ as $g := \sqrt{\xi}\sum_{h \in H} c_h + \sum_{h \notin H} c_h$. Then, the following inequality holds:

$$\delta^{\text{supposed}}\left(\ker\mathcal{A} \cap \operatorname{int}\mathcal{K}\right) > \xi \cdot \delta^{\text{supposed}}\left(Q_{g^{-1}}\left(\ker\mathcal{A}\right) \cap \operatorname{int}\mathcal{K}\right).$$

Proof. For simplicity of discussion, let $|H| = 1$, i.e., $H = \{i\}$. Let us define the points x^*, y^*, and $\bar{x}^*$ as follows:

$$\begin{aligned}
x^* &= \arg\max\left\{\det(x) \mid x \in \ker\mathcal{A} \cap \operatorname{int}\mathcal{K}, \|x\|_J^2 = 1\right\}, \\
y^* &= \arg\max\left\{\det(y) \mid y \in \ker\mathcal{A} \cap \operatorname{int}\mathcal{K}, \|Q_{g^{-1}}(y)\|_J^2 = 1\right\}, \\
\bar{x}^* &= \arg\max\left\{\det(\bar{x}) \mid \bar{x} \in Q_{g^{-1}}\left(\ker\mathcal{A}\right) \cap \operatorname{int}\mathcal{K}, \|\bar{x}\|_J^2 = 1\right\}.
\end{aligned}$$

21

Note that the feasible region with respect to y is the set of solutions whose norm is 1 after scaling. First, we show that $\|y\|_J^2 < 1$, and then $\det(x^*) > \det(y^*)$. Proposition 2.5 ensures that $\|Q_{g^{-1}}(y)\|_J^2 = \langle Q_{g^{-1}}(y), Q_{g^{-1}}(y)\rangle = \langle y, Q_{g^{-2}}(y)\rangle$. To expand $Q_{g^{-2}}(y)$, we expand the following equations by letting $a = \frac{1}{\sqrt{\xi}} - 1$:

$$g^{-2} = e + (2a + a^2)c_i,$$
$$g^{-4} = e + \left(2(2a + a^2) + (2a + a^2)^2\right)c_i$$
$$g^{-2} \circ y = y + (2a + a^2)c_i \circ y,$$
$$g^{-2} \circ (g^{-2} \circ y) = y + 2(2a + a^2)c_i \circ y + (2a + a^2)^2 c_i \circ (c_i \circ y),$$
$$g^{-4} \circ y = y + \left(2(2a + a^2) + (2a + a^2)^2\right)c_i \circ y.$$

Thus, $Q_{g^{-2}}(y)$ turns out to be

$$Q_{g^{-2}}(y) = 2g^{-2} \circ (g^{-2} \circ y) - g^{-4} \circ y$$
$$= y + 2(2a + a^2)c_i \circ y + 2(2a + a^2)^2 c_i \circ (c_i \circ y) - (2a + a^2)^2 c_i \circ y$$
$$= y + 2(2a + a^2)c_i \circ y + (2a + a^2)^2 Q_{c_i}(y),$$

and hence, we obtain $\|Q_{g^{-1}}(y)\|_J^2$ as

$$\langle y, Q_{g^{-2}}(y)\rangle = \|y\|_J^2 + 2(2a + a^2)\langle y, c_i \circ y\rangle + (2a + a^2)^2 \langle y, Q_{c_i}(y)\rangle$$
$$= \|y\|_J^2 + 2(2a + a^2)\langle y \circ y, c_i\rangle + (2a + a^2)^2 \left(\langle y, c_i\rangle\right)^2$$

where the second equality follows from Proposition 6.3. Here, $y \in \operatorname{int} \mathcal{K}$ and $c_i \in \mathcal{K}$ imply that $\langle y, c_i\rangle > 0$, and $y \circ y = y^2 \in \operatorname{int} \mathcal{K}$ implies $\langle y \circ y, c_i\rangle > 0$. Noting that $a > 0$ and $\|Q_{g^{-1}}(y)\|_J^2 = 1$, $\|y\|_J^2 < 1$ should hold and hence, $\frac{1}{\|y^*\|_J} > 1$, which implies that $\det\left(\frac{1}{\|y^*\|_J}y^*\right) > \det(y^*)$. Since $\left\|\frac{1}{\|y^*\|_J}y^*\right\|_J^2 = 1$ holds, we find that $\det(x^*) > \det(y^*)$. Next, we describe the lower bound for $\det(y^*)$ using $\det(\bar{x}^*)$. Since the largest eigenvalue of $\bar{x}$ satisfying $\|\bar{x}\|_J^2 = 1$ is less than 1, by Proposition 3.5, we have:

$$\left\{Q_g(\bar{x}) \in \mathbb{E} \mid \bar{x} \in Q_{g^{-1}}(\ker \mathcal{A}) \cap \mathcal{K}, \|\bar{x}\|_J^2 = 1\right\} \subseteq \ker \mathcal{A} \cap \mathcal{K}.$$

This implies $\det(y^*) \geq \det\left(Q_g(\bar{x}^*)\right)$, and by Proposition 2.5, we have $\det(y^*) \geq \det(g)^2 \det(\bar{x}^*) = \xi^{|H|}\det(\bar{x}^*) = \xi \det(\bar{x}^*)$. Thus, $\det(x^*) > \det(y^*) \geq \xi \det(\bar{x}^*)$ holds, and we can conclude that $\delta^{\text{supposed}}\left(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K}\right) > \xi \cdot \delta^{\text{supposed}}\left(Q_{g^{-1}}(\ker \mathcal{A}) \cap \operatorname{int} \mathcal{K}\right)$. $\qquad\square$

Next, using Proposition 6.5, we derive the maximum number of iterations until the proposed method finds $x \in \ker \mathcal{A} \cap \operatorname{int} \mathcal{K}$ by using $\delta(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K})$ as in Pena and Soheili's method.

Proposition 6.6. *Suppose that $\ker \mathcal{A} \cap \operatorname{int} \mathcal{K} \neq \emptyset$ holds. Algorithm 2 returns $x \in \ker \mathcal{A} \cap \operatorname{int} \mathcal{K}$ after at most $\log_\xi \delta(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K})$ iterations.*

Proof. Let $\ker \bar{\mathcal{A}}$ be the linear subspace at the start of k iterations of Algorithm 2 and suppose that $\delta^{\text{supposed}}\left(\ker \bar{\mathcal{A}} \cap \operatorname{int} \mathcal{K}\right) = 1/r^{\frac{r}{2}}$ holds. Then, from Proposition 6.5, we find that $\delta^{\text{supposed}}\left(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K}\right) > \xi^k/r^{\frac{r}{2}}$. This implies that $\delta(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K}) > \xi^k$ since $\delta(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K}) = r^{\frac{r}{2}} \cdot \delta^{\text{supposed}}\left(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K}\right)$ holds. By taking the logarithm base ξ, we obtain $\log_\xi \delta(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K}) > k$. $\qquad\square$

From here on, using the above results, we will compare the computational complexities of the methods in the case that $\mathcal{K}$ is simple and $\ker \mathcal{A} \cap \operatorname{int} \mathcal{K} \neq \emptyset$ holds. Table 1 summarizes the upper bounds on the

number of iterations of the main algorithm (UB#iter) of the two methods and the computational costs required per iteration (CC/iter). As in the previous section, the main algortihm requires $\mathcal{O}(m^3 + m^2 d)$ to compute the projection $\mathcal{P}_\mathcal{A}$. Here, BP shows the computational cost of the basic procedure in each method.

Table 1: Comparison of our method and Pena and Soheili's method in the main algorithm

Method	UB#iter	CC/iter
Proposed method	$\log_\xi \delta \left(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K} \right)$	$m^3 + m^2 d+$ BP
Pena and Soheili's method	$-\log_{1.5} \delta \left(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K} \right)$	$m^3 + m^2 d+$ BP

The upper bound on the number of iterations of Algorithm 2 is given by $\log_\xi \delta \left(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K} \right) = \log_{1.5} \delta \left(\ker \mathcal{A} \cap \operatorname{int} \mathcal{K} \right) / \log_{1.5} \xi$, where we should note that $0 < \xi < 1$. Since $0 < \frac{1}{-\log_{1.5} \xi} \leq 1$ when $\xi \leq 2/3$, if $\xi \leq 2/3$, then the upper bound on the number of iterations of Algorithm 2 is smaller than that of the main algorithm of Pena and Soheili's method.

Next, Table 2 summarizes upper bounds on the number of iterations of basic procedures in the proposed method (UB#iter) and Pena and Soheili's method and the computational cost required per iteration (CC/iter). It shows cases of using the von Neumann scheme and the smooth perceptron in each method (corresponding to Algorithm 1 and Algorithm 5 in the proposed method). As in the previous section, C^{sd} denotes the computational cost required for spectral decomposition, and C^{min} denotes the computational cost required to compute only the minimum eigenvalue and the corresponding primitive idempotent.

Table 2: Comparison of our method and Pena and Soheili's method in the basic procedure

Method	von Neumann scheme		smooth perceptron	
	UB#iter	CC/iter	UB#iter	CC/iter
Proposed method	$\frac{r^2}{\xi^2}$	$\max(C^{\mathrm{sd}}, md)$	$\frac{2\sqrt{2}r}{\xi} - 1$	$\max(C^{\mathrm{sd}}, md)$
Pena and Soheili's method	$16r^4$	$\max(C^{\mathrm{min}}, md)$	$8\sqrt{2}r^2 - 1$	$\max(C^{\mathrm{sd}}, md)$

Note that by setting $\xi = (4r)^{-1}$, the upper bounds on the number of iterations of the basic procedure of the two methods are the same. If $\xi = (4r)^{-1}$, then $\frac{1}{-\log_{1.5} \xi} = \frac{1}{\log_{1.5} 4r} \leq \frac{1}{\log_{1.5} 4} = 0.292$, and the upper bound of the number of iterations of Algorithm 2 is less than 0.3 times the upper bound of the number of iterations of the main algorithm of Pena and Soheili's method, which implies that the larger the value of r is, the smaller the ratio of those bounds becomes. From the discussion in Section 6.3, we can assume $\mathcal{O}(C^{\mathrm{sd}} = \mathcal{C}^{\mathrm{min}})$, and Table 2 shows that the proposed method is superior for finding a point $x \in \ker \mathcal{A} \cap \operatorname{int} \mathcal{K}$.

6.3 Computational costs of C^{sd} and C^{min}

This section discusses the computational cost required for spectral decomposition C^{sd} and the computational cost required to compute only the minimum eigenvalue and the corresponding primitive idempotent C^{min}.

There are so-called direct and iterative methods for eigenvalue calculation algorithms, briefly described

on pp.139-140 of [4]. (Note that it is also written that there is no direct method in the strict sense of an eigenvalue calculation since finding eigenvalues is mathematically equivalent to finding zeros of polynomials).

In general, when using the direct method of $\mathcal{O}(n^3)$, we see that $C^{\mathrm{sd}} = \mathcal{O}(n^3)$ and $C^{\mathrm{min}} = \mathcal{O}(n^3)$. The Lanczos algorithm is a typical iterative algorithm used for sparse matrices. Its cost per iteration of computing the product of a matrix and a vector once is $\mathcal{O}(n^2)$. Suppose the number of iterations at which we obtain a sufficiently accurate solution is constant with respect to the matrix size. In that case, the overall computational cost of the algorithm is $\mathcal{O}(n^2)$. Corollary 10.1.3 in [7] discusses the number of iterations that yields sufficient accuracy. It shows that we can expect fewer iterations if the value of "the difference between the smallest and second smallest eigenvalues / the difference between the second smallest and largest eigenvalue" is larger. However, it is generally difficult to assume that the above value does not depend on the matrix size and is sufficiently large. Thus, even in this case, we cannot take advantage of the condition that we only need the minimum eigenvalue, and we conclude that it is reasonable to consider that $\mathcal{O}(C^{\mathrm{sd}}) = \mathcal{O}(C^{\mathrm{min}})$.

7 Numerical experiments

7.1 Outline of numerical implementation

Numerical experiments were performed using the authors' implementations of the algorithms on a positive semidefinite optimization problem with one positive semidefinite cone $\mathcal{K} = \mathbb{S}^n_+$ of the form

$$\mathrm{P}(\mathcal{A}) \quad \text{find} \quad X \in \mathbb{S}^n_{++} \quad \text{s.t.} \quad \mathcal{A}(X) = \mathbf{0} \in \mathbb{R}^m$$

where $\mathbb{S}^n_{++}$ denotes the interior of $\mathcal{K} = \mathbb{S}^n_+$. We created strongly feasible ill-conditioned instances, i.e., $\ker \mathcal{A} \cap \mathbb{S}^n_{++} \neq \emptyset$ and $X \in \ker \mathcal{A} \cap \mathbb{S}^n_{++}$ has positive but small eigenvalues. We will explain how to make a such instance in section 7.2. In what follows, we refer to Lourenço et al.'s method [10] as Lourenço (2019), and Pena and Soheili's method [13] as Pena (2017). We set the termination parameter as $\xi = 1/4$ in our basic procedure. The reason for setting $\xi = 1/4$ is to prevent the square root of ξ from becoming an infinite decimal, and to prevent the upper bound on the number of iterations of the basic procedure from becoming too large. We also set the accuracy parameter as $\varepsilon = 1\mathrm{e}\text{-}12$, both in our main algorithm and in Lourenço (2019) and determined whether $\mathrm{P}_{S_\infty}(\mathcal{A})$ or $\mathrm{P}_{S_1}(\mathcal{A})$ has a solution whose minimum eigenvalue is greater than or equal to ε. Note that [13] proposed various update methods for the basic procedure. In our numerical experiments, all methods employed the modified von Neumann scheme (Algorithm 4) with the identity matrix as the initial point and the smooth perceptron scheme (Algorithm 5). This implies that the basic procedures used in the three methods differ only in the termination conditions for moving to the main algorithm and that all other steps are the same. All executions were performed using MATLAB R2022a on an Intel (R) Core (TM) i7-6700 CPU @ 3.40GHz machine with 16GB of RAM. Note that we computed the projection $\mathcal{P}_{\mathcal{A}}$ using the MATLAB function for the singular value decomposition. The projection $\mathcal{P}_{\mathcal{A}}$ was given by $\mathcal{P}_{\mathcal{A}} = I - A^\top (AA^\top)^{-1} A$ using the matrix $A \in \mathbb{R}^{m \times d}$ which represents the linear operator $\mathcal{A}(\cdot)$ and the identity matrix I. Here, suppose that the singular value decomposition of a matrix A is given by $A = U\Sigma V^\top = U(\Sigma_m \ O)V^\top$ where $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{d \times d}$ are orthogonal matrices, and $\Sigma_m \in \mathbb{R}^{m \times m}$ is a diagonal matrix with m singular values on the diagonal. Substituting this decomposition into $A^\top (AA^\top)^{-1} A$, we have

$$
\begin{aligned}
A^\top (AA^\top)^{-1} A &= A^\top (U\Sigma\Sigma^\top U^\top)^{-1} A \\
&= A^\top U^{-\top} (\Sigma_m^2)^{-1} U^{-1} A \\
&= V\Sigma^\top \Sigma_m^{-2} \Sigma V^\top = V \begin{pmatrix} I_m & O \\ O & O \end{pmatrix} V^\top = V_{:,1:m} V_{:,1:m}^\top,
\end{aligned}
$$

24

where $V_{:,1:m}$ represents the submatrix from column 1 to column m of V. Thus, for any $x \in \mathbb{E}$, we can compute $\mathcal{P}_{\mathcal{A}}(x) = x - V_{:,1:m}V_{:,1:m}^{\top}x$.

In what follows, $\bar{X} \in \mathbb{S}^n$ denotes the output obtained from the main algorithm and X^* the result scaled as the solution of the original problem P($\mathcal{A}$) multiplied by a real number such that $\lambda_{max}(X^*) = 1$. When X^* was obtained, we defined the residual of the constraints as the value of $\|\mathcal{A}(X^*)\|_2$.

We also solved the following problem with a commercial code, Mosek [12], and compared it with the output of Chubanov's methods:

$$
\begin{array}{llll}
\text{(P)} & \min & 0 & \text{s.t} \quad \mathcal{A}(X) = \mathbf{0}, \quad X \in \mathbb{S}^n_+, \\
\text{(D)} & \max & \mathbf{0}^{\top}y & \text{s.t} \quad -\mathcal{A}^*y \in \mathbb{S}^n_+.
\end{array}
$$

Here, Mosek solves the self-dual embedding model by using a path-following interior-point method, so if we obtain a solution (X^*, y^*), then X^* and $-\mathcal{A}^*y^*$ lie in the (approximate) relative interior of the primal feasible region and the dual feasible region, respectively [20]. That is, X^* obtained by solving a strongly feasible problem with Mosek is in $\mathbb{S}^n_{++}$, X^* obtained by solving a weakly feasible problem is in $\mathbb{S}^n_+ \setminus \mathbb{S}^n_{++}$, and X^* obtained by solving an infeasible problem is $X^* = O$ (i.e., $-\mathcal{A}^*y^* \in \mathbb{S}^n_{++}$). As well as for Chubanov's methods, we computed $\|\mathcal{A}(X^*)\|_2$ for the solution obtained by Mosek after scaling so that $\lambda_{max}(X^*)$ would be 1. Note that (P) and (D) do not simultaneously have feasible interior points. In general, it is difficult to solve such problems stably by using interior point methods, but since strong complementarity exists between (P) and (D), they can be expected to be stably solved. By applying Lemma 3.4 of [11], we can generate a problem in which both the primal and dual problems have feasible interior points in which it can be determined whether (P) has a feasible interior point. However, since there was no big difference between the solution obtained by solving the problem generated by applying Lemma 3.4 of [11] and the solution obtained by solving the above (P) and (D), we showed only the results of solving (P) and (D) above.

7.2 How to generate instances

Here, we describe how the strongly feasible ill-conditioned instances were generated. In what follows, for any natural numbers m, n, rand(n) is a function that returns n-dimensional real vectors whose elements are uniformly distributed in the open segment $(0, 1)$, and rand(m, n) is a function that returns an $m \times n$ real matrix whose elements are uniformly distributed in the open segment $(0, 1)$. Furthermore, for any $x \in \mathbb{R}^n$ and $X \in \mathbb{R}^{m \times n}$, diag($x$) $\in \mathbb{R}^{n \times n}$ is a function that returns a diagonal matrix whose diagonal elements are the elements of x, and vec(X) $\in \mathbb{R}^{mn}$ is a function that returns a vector obtained by stacking the n column vectors of X. The strongly feasible ill-conditioned instances were generated by extending the method of generating ill-conditioned strongly feasible instances proposed in [14] to the symmetric cone case.

Proposition 7.1. *Suppose that $\bar{x} \in \text{int}\mathcal{K}$, $\|\bar{x}\|_{\infty} \leq 1$ and $\bar{u} \in \mathcal{K}, \|\bar{u}\|_1 = r$ satisfy $\langle \bar{x}, \bar{u} \rangle = r$. Define the linear operator $\mathcal{A} : \mathbb{E} \to \mathbb{R}^m$ as $\mathcal{A}(x) = (\langle a_1, x \rangle, \langle a_2, x \rangle, \ldots, \langle a_m, x \rangle)^T$ for which $a_1 = \bar{u} - \bar{x}^{-1}$ and $\langle a_j, \bar{x} \rangle = 0$ hold for any $j = 2, \ldots, m$. Then,*

$$
\bar{x} = \arg \max_x \{\det(x) : x \in \mathcal{K} \cap \ker\mathcal{A}, \|x\|_{\infty} = 1\}. \tag{21}
$$

Proof. First, note that the assertion (21) is equivalent to

$$
\bar{x} = \arg \max_{x \in \mathcal{F}} \{\log \det(x)\} \quad \text{where } \mathcal{F} := \{x \in \mathcal{K} \cap \ker\mathcal{A} : \|x\|_{\infty} \leq 1\}. \tag{22}
$$

From the assumptions, we see that $\bar{x} \in \mathcal{K}$, $\|\bar{x}\|_{\infty} \leq 1$ and $\langle a_1, \bar{x} \rangle = \langle \bar{u} - \bar{x}^{-1}, \bar{x} \rangle = r - r = 0$; thus, $\mathcal{A}(\bar{x}) = 0$ and $\bar{x} \in \mathcal{F}$. Since $\nabla \log \det(x) = x^{-1}$, if $\bar{x}$ satisfies

$$
\langle x - \bar{x}, \bar{x}^{-1} \rangle \leq 0 \text{ for any } x \in \mathcal{F} \tag{23}
$$

we can conclude that (22) holds. In what follows, we show that (23) holds.

For any $x \in \mathcal{F}$, $x \in \ker\mathcal{A}$ and hence, $\langle a_1, x \rangle = \langle \bar{u} - \bar{x}^{-1}, x \rangle = \langle \bar{u} - \bar{x}^{-1}, x \rangle = 0$, i.e., $\langle \bar{u}, x \rangle = \langle \bar{x}^{-1}, x \rangle$. Thus, we obtain

$$
\begin{aligned}
\langle x - \bar{x}, \bar{x}^{-1} \rangle &= \langle \bar{u}, x \rangle - r \\
&\leq \langle \bar{u}, x \rangle - \|\bar{u}\|_1 \|x\|_\infty \qquad (\text{by } \|\bar{u}\|_1 = r \text{ and } \|x\|_\infty \leq 1) \\
&\leq 0 \qquad (\text{by } \langle \bar{u}, x \rangle \leq \|\bar{u}\|_1 \|x\|_\infty)
\end{aligned}
$$

which completes the proof. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad\square$

Proposition 7.1 guarantees that we can generate a linear operator $\mathcal{A}$ satisfying $\ker\mathcal{A} \cap \mathbb{S}_{++}^n \neq \emptyset$ by determining an appropriate value $\mu = \max\limits_{X \in \mathcal{F}} \det(X)$, where $\mathcal{F} = \{X \in \mathbb{S}^n : X \in \mathbb{S}_{++}^n \cap \ker\mathcal{A}, \|X\|_\infty = 1\}$. The details on how to generate the strongly feasible instances are in Algorithm 3. The input consists of the rank of the semidefinite cone n, the number of constraints m, an arbitrary orthogonal matrix P, and the parameter $\tau \in \mathbb{R}_{++}$ which determines the value of μ. We made instances for which the value of μ satisfies $1e - \tau \leq \mu \leq 1e - (\tau - 1)$. In the experiments, we set $\tau \in \{50, 100, 150, 200, 250\}$ so that μ would vary around 1e-50, 1e-100, 1e-150, 1e-200, and 1e-250.

Note that Algorithm 3 generates instances using $\bar{x}$ that has a natural eigenvalue distribution. For example, let $n - 1 = 3$ and consider two Xs where one has 3 eigenvalues of about 1e-2, and the others have 1 each of 1e-1, 1e-2, and 1e-3. $\det(X) \simeq$ 1e-6 is obtained for both Xs, but the latter is more natural for the distribution of eigenvalues. In our experiment, we generated ill-conditioned instances by using X having a natural eigenvalue distribution as follows:

1. Find an integer s that satisfies $1e\text{-}s \leq l^{\frac{1}{n-1}} \leq u^{\frac{1}{n-1}} \leq 1e\text{-}(s-1)$.

2. Generate $t = 2s - 1$ eigenvalue classes.

3. Decide how many eigenvalues to generate for each class.

For example, when $n = 13$ and $\tau = 30$, Algorithm 3 yields $s = 3$, $t = 5$, $a = 2$ and $b = 2$, and since b is even, we have $num = (2, 3, 2, 3, 2)^\top$. The classes of $t = 5$ eigenvalues are shown in Table 3 below. Note that $(l^{\frac{1}{n-1}} \cdot 10^{s-i}) \cdot (l^{\frac{1}{n-1}} \cdot 10^{s-(t-i+1)}) = l^{\frac{2}{n-1}}$ and $(u^{\frac{1}{n-1}} \cdot 10^{s-i}) \cdot (u^{\frac{1}{n-1}} \cdot 10^{s-(t-i+1)}) = u^{\frac{2}{n-1}}$ hold for the i-th and $t - i + 1$-th classes. This implies that we obtain $1e - \tau \leq \mu = \det(X) \leq 1e - (\tau - 1)$ both when generating $n - 1$ eigenvalues in the sth class and when generating $n - 1$ eigenvalues of X according to num. When $n = 14, \tau = 30$, Algorithm 3 gives $s = 3$, $t = 5$, $a = 2$, and $b = 3$, and since b is an odd number, we have $num = (2, 3, 3, 3, 2)^\top$. Thus, Algorithm 3 generates the instances by controlling the frequency so that the geometric mean of the $n - 1$ eigenvalues of X falls within the s-th class width.

Table 3: Frequency distribution table of eigenvalues of X generated by Algorithm 3 when $n = 13$ or $n = 14$, $\tau = 30$

Class	Class width of eigenvalues of $\bar{x}$		Frequency(num)	
	Lower bound	Upper bound	$n = 13$	$n = 14$
1	$l^{\frac{1}{n-1}} \cdot 10^2$	$u^{\frac{1}{n-1}} \cdot 10^2$	2	2
2	$l^{\frac{1}{n-1}} \cdot 10^1$	$u^{\frac{1}{n-1}} \cdot 10^1$	3	3
3	$l^{\frac{1}{n-1}}$	$u^{\frac{1}{n-1}}$	2	3
4	$l^{\frac{1}{n-1}} \cdot 10^{-1}$	$u^{\frac{1}{n-1}} \cdot 10^{-1}$	3	3
5	$l^{\frac{1}{n-1}} \cdot 10^{-2}$	$u^{\frac{1}{n-1}} \cdot 10^{-2}$	2	2

26

Algorithm 3 Strongly feasible instance

1: **Input:** n, m, τ, P
2: **Output:** A
3: $l \leftarrow 1\text{e} - \tau$, $u \leftarrow 1\text{e} - (\tau - 1)$, $s \leftarrow \lceil \frac{\tau}{n-1} \rceil$ and $t \leftarrow 2s - 1$
4: $b \leftarrow (n-1) \bmod t$, $a \leftarrow \frac{(n-1)-b}{t}$ and $num \leftarrow a \cdot \mathbf{1} \in \mathbb{R}^t$
5: **if** b is odd **then**
6: $\quad \bar{b} \leftarrow \frac{b-1}{2}$ and $num_i \leftarrow num_i + 1$ such that $s - \bar{b} \leq i \leq s + \bar{b}$
7: **else**
8: $\quad \bar{b} \leftarrow \frac{b}{2}$ and $num_i \leftarrow num_i + 1$ such that $s - \bar{b} \leq i < s$ or $s < i \leq s + \bar{b}$
9: **end if**
10: $d_1 \leftarrow 1$ and $k \leftarrow 2$
11: **for** $i = 1$ to t **do**
12: $\quad$ **for** $j = 1$ to num_i **do**
13: $\quad\quad dl \leftarrow l^{\frac{1}{n-1}} \cdot 10^{s-i}$ and $du \leftarrow u^{\frac{1}{n-1}} \cdot 10^{s-i}$
14: $\quad\quad d_k \leftarrow dl + (du - dl)\,\text{rand}\,(1)$
15: $\quad\quad k \leftarrow k + 1$
16: $\quad$ **end for**
17: **end for**
18: $D' \leftarrow \text{diag}(d)$ and then compute $C \leftarrow PD'P^T$ and $c \leftarrow \text{vec}(C)$
19: $u \leftarrow (n, 0_{n-1}^T)^T$ where 0_{n-1} denotes the $n-1$-dimensional vector of zeros
20: $U \leftarrow P(\text{diag}(u) - D'^{-1})P^T$, $A' \leftarrow \text{vec}(U)$ and $R \leftarrow I - \frac{1}{\|c\|_2^2}cc^T$
21: **for** $i = 1$ to $m - 1$ **do**
22: $\quad A'_i \leftarrow \text{rand}(n, n)$ and $A_i \leftarrow \left(A'_i + (A'_i)^T\right)/2$
23: $\quad A' \leftarrow \begin{pmatrix} A' \\ \text{vec}(A_i)^T \end{pmatrix}$
24: **end for**
25: $\bar{A} \leftarrow A'R$
26: $A \leftarrow \begin{pmatrix} \text{vec}(U)^T \\ \bar{A} \end{pmatrix}$

7.3 Numerical results and observations

We set the size of the positive semidefinite matrix to $n = 50$, so that the computational experiments could be performed in a reasonable period of time. To eliminate bias in the experimental results, we generated instances in which the number of constraints m was controlled using the parameter ν for the number $\frac{n(n+1)}{2}$ of variables in the symmetric matrix of order n. Specifically, the number of constraints m on an integer was obtained by rounding the value of $\frac{n(n+1)}{2}\nu$, where $\nu \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. For each $\nu \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, we generated five instances, i.e., 25 instances for each of five strongly feasible cases (corresponding to five patterns of $\mu \simeq 1\text{e-}50, \ldots, \mu \simeq 1\text{e-}250$, see section 7.2 for details). Thus, we generated 125 strongly feasible instances. We set the upper limit of the execution time to 2 hours and compared the performance of our method with those of Lourenço (2019), Pena(2017) and Mosek. When using Mosek, we set the primal feasibility tolerance to 1e-12.

Tables 4 and 5 list the results for the (ill-conditioned) strongly feasible case. The "CO-ratio" column shows the ratio of $|N_2|/|N_1|$ where N_1 is the set of problems for which the algorithm terminated within 2 hours, the upper limit of the execution time, and $N_2 \subseteq N_1$ is the set of problems for which a correct output is obtained, the "times(s)" column shows the average CPU time of the method, the "M-iter" column shows the average iteration number of each main algorithm, the $\|\mathcal{A}(X^*)\|_2$ column shows the residual of the constraints, and the $\lambda_{\min}(X^*)$ column shows the minimum eigenvalue of X^*. The "BP" column shows which scheme (the modified von Neumann (MVN) or the smooth perceptron (SP)) was

27

Table 4: Results for strongly feasible instances (Correct output (CO-) ratio, CPU time and and M-iter)

Instance	BP	Algorithm 2			Lourenço (2019)			Pena (2017)		
		CO-ratio	time(s)	M-iter	CO-ratio	time(s)	M-iter	CO-ratio	time(s)	M-iter
$\mu \simeq$ 1e-50	MVN	25/25	7.81	3.28	25/25	25.94	14.48	25/25	3.60	1.00
	SP	25/25	0.75	1.00	25/25	10.12	14.08	25/25	0.80	1.00
$\mu \simeq$ 1e-100	MVN	25/25	51.62	53.12	25/25	448.05	329.04	1/1	(4513.59)	(2.00)
	SP	25/25	32.11	36.04	25/25	256.24	365.76	25/25	123.65	23.32
$\mu \simeq$ 1e-150	MVN	25/25	99.39	118.12	25/25	888.25	728.68	-	-	-
	SP	25/25	76.98	91.96	25/25	520.73	756.36	25/25	781.88	117.32
$\mu \simeq$ 1e-200	MVN	25/25	144.48	185.40	25/25	1328.68	1145.40	-	-	-
	SP	25/25	118.06	151.44	25/25	789.29	1150.20	25/25	1874.20	236.44
$\mu \simeq$ 1e-250	MVN	25/25	188.11	251.44	25/25	1827.24	1601.20	-	-	-
	SP	25/25	162.67	215.12	25/25	1074.07	1564.80	25/25	3308.35	376.24

Table 5: Results for ill-conditioned strongly feasible instances ($\|\mathcal{A}(X^*)\|_2$ and $\lambda_{\min}(X^*)$)

Instance	BP	Algorithm 2		Lourenço (2019)		Pena (2017)	
		$\|\mathcal{A}(X^*)\|_2$	$\lambda_{\min}(X^*)$	$\|\mathcal{A}(X^*)\|_2$	$\lambda_{\min}(X^*)$	$\|\mathcal{A}(X^*)\|_2$	$\lambda_{\min}(X^*)$
$\mu \simeq$ 1e-50	MVN	1.24e-11	4.42e-4	7.64e-12	3.60e-4	1.27e-11	3.95e-4
	SP	1.23e-11	8.48e-4	8.22e-12	8.10e-4	1.23e-11	8.48e-4
$\mu \simeq$ 1e-100	MVN	9.98e-12	2.73e-6	1.26e-11	3.01e-6	(1.07e-8)	(3.34e-6)
	SP	4.18e-11	3.19e-5	1.10e-11	3.75e-5	5.38e-9	3.39e-6
$\mu \simeq$ 1e-150	MVN	1.96e-10	5.24e-8	4.29e-10	4.10e-8	-	-
	SP	2.21e-10	3.98e-7	5.60e-10	5.54e-7	6.31e-9	3.78e-7
$\mu \simeq$ 1e-200	MVN	1.51e-8	7.86e-10	4.76e-8	1.18e-9	-	-
	SP	1.09e-8	4.87e-9	3.81e-8	5.93e-9	1.72e-8	5.06e-9
$\mu \simeq$ 1e-250	MVN	9.51e-7	8.43e-12	2.58e-6	2.52e-11	-	-
	SP	1.72e-6	5.14e-11	3.35e-6	7.05e-11	1.73e-6	5.40e-11

used in the basic procedure. The values in parentheses () in row $\mu \approx$ 1e-100 are the average values excluding instances for which the method ended up running out of time.

First, we compare the results when using MVN or SP as the basic procedure in each method. From Table 4, we can see that for strongly-feasible problems, using SP as the basic procedure has a shorter average execution time than using MVN. Next, we compare the results of each method. For $\mu \simeq$ 1e-50, there was no significant difference in performance among the three methods. For $\mu \leq$ 1e-100, the results in the rows BP=MVN show that our method and Lourenço (2019) obtained interior feasible solutions for all problems, while Pena (2017) ended up running out of time for 99 instances. This is because Pena (2017) needs to call its basic procedure to find a solution of range$\mathcal{A}^* \cap \mathbb{S}_{++}^n$. Comparing our method with Lourenço (2019), we see that Algorithm 2 is superior in terms of CPU time. Finally, we compare the results for each value of μ. As μ becomes smaller, i.e., as the problem becomes more ill-conditioned, the number of scaling times and the execution time increase, and the accuracy of the obtained solution gets worse.

Table 6 summarizes the results of our experiments using Mosek to solve strongly feasible ill-conditioned instances. Mosek sometimes returned the error message "rescode = 10006" for the $\mu \leq 1e - 200$ instances. This error message means that "the optimizer is terminated due to slow progress." In this

Table 6: Results for ill-conditioned strongly feasible instances with Mosek

Instance	CO-ratio	time(s)	$\|\mathcal{A}(X^*)\|_2$	$\lambda_{\min}(X^*)$
$\mu \simeq$ 1e-50	25/25	1.96	8.73e-13	7.99e-3
$\mu \simeq$ 1e-100	25/25	3.18	1.87e-12	4.51e-5
$\mu \simeq$ 1e-150	25/25	3.72	2.48e-10	4.45e-7
$\mu \simeq$ 1e-200	21/25	6.56	2.58e-7	4.35e-9
$\mu \simeq$ 1e-250	1/25	6.88	2.57e-7	5.37e-11

case, the obtained solution is not guaranteed to be optimal, but it may have sufficient accuracy as a feasible solution. Therefore, we took the CO-ratio when the residual $\|\mathcal{A}(X^*)\|_2$ is less than or equal to 1e-5 to be the correct output. The reason why we set the threshold to 1e-5 is that the maximum value of $\|\mathcal{A}(X^*)\|_2$ was less than 1e-5 among the X^* values obtained for the strongly feasible ill-conditioned instances by the three methods, Algorithm 2, Lourenço (2019) and Pena (2017). On the other hand, for the $\mu \leq 1e-200$ instances, the Chubanov methods had higher CO-ratios. That is, when the problem was quite ill-conditioned, the solution obtained by each of the Chubanov methods had a smaller value of $\|\mathcal{A}(X^*)\|_2$ compared with the solution obtained by Mosek, which implies that the accuracy of the solution obtained by each of the Chubanov methods was higher than that of Mosek.

8   Concluding remarks

In this study, we proposed a new version of Chubanov's method for solving the feasibility problem over the symmetric cone by extending Roos's method [15] for the feasible problem over the nonnegative orthant. Our method has the following features:

- Using the norm $\| \cdot \|_\infty$ in problem $\mathrm{P}_{S_\infty}(\mathcal{A})$ makes it possible to (i) calculate the upper bound for the minimum eigenvalue of any feasible solution of $\mathrm{P}_{S_\infty}(\mathcal{A})$, (ii) quantify the feasible region of $\mathrm{P}(\mathcal{A})$, and hence (iii) determine whether there exists a feasible solution of $\mathrm{P}(\mathcal{A})$ whose minimum eigenvalue is greater than ϵ as in [10].

- In terms of the computational bound, our method is (i) equivalent to Roos's original method [15] and superior to Lourenço et al.'s method [10] when the symmetric cone is the nonnegative orthant, (ii) superior to Lourenço et al.'s when the symmetric cone is a Cartesian product of second-order cones, (iii) equivalent to Lourenço et al.'s when the symmetric cone is the simple positive semidefinite cone, under the assumption that the costs of computing the spectral decomposition and the minimum eigenvalue are of the same order for any given symmetric matrix, and (iv) superior to Pena and Soheili's method [13] for any simple symmetric cones under the assumption that $\mathrm{P}(\mathcal{A})$ is feasible.

We also conducted comprehensive numerical experiments comaring our method with the existing mtehods of Chubanov [10, 13] and Mosek. Our numerical results showed that

- It is considerably faster than the existing methods on ill-conditioned strongly feasible instances.

- Mosek was the better than Chubanov's methods in terms of execution time. On the other hand, in terms of the accuracy of the solution (the value of $\|\mathcal{A}(X^*)\|_2$), we found that all of Chubanov's methods are better than Mosek. In particular, we have seen such results for strongly-feasible (terribly) ill-conditioned ($\mu \simeq 1e-250$) instances.

29

In this paper, we performed computer experiments by setting $\xi = 1/4$ in the basic procedure to avoid inducements for calculation errors, but there is room for further study on how to choose the value of ξ. For example, if the problem size is large, the computation of the projection $\mathcal{P}_{\mathcal{A}}$ is expected to take much more time. In this case, rather than setting $\xi = 1/4$, running the algorithm as $\xi < 1/4$ may reduce the number of scaling steps to be performed before completion. As a result, the algorithm's run time may be shorter than when we set $\xi = 1/4$. More desirable approach may be to choose an appropriate value of ξ at each iteration along to the algorithm's progress.

Acknowledgments

References

[1] Alizadeh F. (2012) . An introduction to formally real Jordan algebras and their applications in optimization. In: Anjos M., Lasserre J. (eds) Handbook on Semidefinite, Conic and Polynomial Optimization. International Series in Operations Research & Management Science, vol 166. Springer, Boston, MA.

[2] Chubanov, S. (2012). A strongly polynomial algorithm for linear systems having a binary solution. Mathematical programming, 134(2), 533-570.

[3] Chubanov, S. (2015). A polynomial projection algorithm for linear feasibility problems. Mathematical Programming, 153(2), 687-713.

[4] Demmel, J. W. (1997). Applied numerical linear algebra. Society for Industrial and Applied Mathematics.

[5] Faraut, J. & Korányi, A. (1994). Analysis on symmetric cones. Oxford University Press, Oxford, UK.

[6] Faybusovich, L. (1997). Euclidean Jordan algebras and interior-point algorithms. Positivity 1, 331–357.

[7] Golub, G. H., & Van Loan, C. F. (2013). Matrix computations. JHU press.

[8] Jordan, P., Neumann, J. v., & Wigner, E. (1934). On an Algebraic Generalization of the Quantum Mechanical Formalism. Annals of Mathematics, 35(1), 29–64.

[9] Kitahara, T., & Tsuchiya, T. (2018). An extension of Chubanov's polynomial-time linear programming algorithm to second-order cone programming. Optimization Methods and Software, 33(1), 1-25.

[10] Lourenço, B. F., Kitahara, T., Muramatsu, M., & Tsuchiya, T. (2019). An extension of Chubanov's algorithm to symmetric cones. Mathematical Programming, 173(1-2), 117-149.

[11] Lourenço, B. F., Muramatsu, M., & Tsuchiya, T. (2021). Solving SDP completely with an interior point oracle. Optimization Methods and Software, 36(2-3), 425-471.

[12] MOSEK, A. Moset optimization toolbox for MATLAB (2019). Release, 9, 98.

[13] Pena, J., & Soheili, N. (2017). Solving conic systems via projection and rescaling. Mathematical Programming, 166(1-2), 87-111.

[14] Pena, J., & Soheili, N. (2019). Computational performance of a projection and rescaling algorithm. Optimization Methods and Software, 1-18.

[15] Roos, K. (2018). An improved version of Chubanov's method for solving a homogeneous feasibility problem. Optimization Methods and Software, 33(1), 26-44.

[16] Schmieta, S., Alizadeh, F. (2003). Extension of primal-dual interior point algorithms to symmetric cones. Mathematical Programming, Series. A 96, 409–438.

[17] Soheili, N., & Pena, J. (2012). A smooth perceptron algorithm. SIAM Journal on Optimization, 22(2), 728-737.

[18] Soheili, N., & Pena, J. (2013). A primal–dual smooth perceptron–von Neumann algorithm. In Discrete Geometry and Optimization (pp. 303-320). Springer, Heidelberg.

[19] Wei, Z., & Roos, K. (2019). Using Nemirovski's Mirror-Prox method as Basic Procedure in Chubanov's method for solving homogeneous feasibility problems. Manuscript (http://www. optimization-online. org/DB HTML/2018/04/6559. html).

[20] Wolkowicz, H., Saigal, R., Vandenberghe, L.: Handbook of Semidefinite Programming: Theory, Algorithms, and Applications, vol. 27. Springer, Berlin (2012)

[21] Yoshise A. (2007) Interior point trajectories and a homogeneous model for nonlinear complementarity problems over symmetric cones. SIAM Journal on Optimization, 17(4), 1129-1153.

A   Basic procedure

Algorithm 4 Basic procedure (Modified von Neumann scheme)

1: Same as lines 1-3 of Algorithm 1
2: **while** $k \leq \frac{p^2 r_{\max}^2}{\xi^2}$ **do**
3:     Same as lines 5-23 of Algorithm 1
4:     **for** $\ell \in \{1, \ldots, p\}$ **do**
5:         $S_\ell \leftarrow \{i \mid \lambda(z_\ell^k)_i \leq 0\}$ and then $u_\ell \leftarrow \sum_{i \in S_\ell} c(z_\ell^k)_i$
6:     **end for**
7:     $u \leftarrow \frac{1}{\sum_{\ell=1}^{p} |S_\ell|} u$ and $y^{k+1} \leftarrow \alpha y^k + (1-\alpha)u$, where $\alpha = \frac{\langle P_{\mathcal{A}}(u), P_{\mathcal{A}}(u) - z^k \rangle}{\|z^k - P_{\mathcal{A}}(u)\|_J^2}$
8:     $k \leftarrow k + 1$, $z^k \leftarrow P_{\mathcal{A}}(y^k)$ and $v^k \leftarrow y^k - z^k$
9: **end while**

Below, we describe the results of updating y^k with the smooth perceptron scheme as described in [13]. Given $\mu > 0$, we define operator $u_\mu(\cdot) : \mathbb{E} \to \{u \in \mathcal{K} \mid \langle u, e \rangle = 1\}$ as $u_\mu(v) := \underset{u \in \mathcal{K}, \langle u, e \rangle = 1}{\arg\min} \left\{ \langle u, v \rangle + \frac{\mu}{2} \|u - \bar{u}\|_J^2 \right\}$.

Algorithm 5 Basic procedure (Smooth perceptron scheme)

1: **Input:** $P_{\mathcal{A}}$ and ξ such that a constant $0 < \xi < 1$

2: **Output:** (i) a solution to P($\mathcal{A}$) or (ii) D($\mathcal{A}$) or (iii) a certificate that, for any feasible solution x to $P_{S_\infty}(\mathcal{A})$, $\langle e, x \rangle < r$

3: initialization : $\bar{u} \leftarrow \frac{1}{r}e$, $\mu^0 \leftarrow 2$, $u^0 \leftarrow \bar{u}$, $k \leftarrow 0, H_1, \ldots, H_p = \emptyset$.

4: compute $y^0 \leftarrow u_{\mu_0}\left(P_{\mathcal{A}}(u^0)\right)$, $z^0 \leftarrow P_{\mathcal{A}}(y^0)$, $v^0 \leftarrow y^0 - z^0$.

5: **while** $k \leq \frac{2\sqrt{2}pr_{\max}}{\xi} - 1$ **do**

6:    Same as lines 5-23 of Algorithm 1

7:    $\theta^k \leftarrow \frac{2}{k+3}$ and $u^{k+1} \leftarrow (1 - \theta^k)(u^k + \theta^k y^k) + (\theta^k)^2 u_{\mu^k}\left(P_{\mathcal{A}}(u^k)\right)$

8:    $\mu^{k+1} \leftarrow (1 - \theta^k)\mu^k$ and $y^{k+1} \leftarrow (1 - \theta^k)y^k + \theta^k u_{\mu^{k+1}}\left(P_{\mathcal{A}}(u^{k+1})\right)$

9:    $k \leftarrow k + 1$, $z^k \leftarrow P_{\mathcal{A}}(y^k)$ and $v^k \leftarrow y^k - z^k$

10: **end while**

B   Another new main algorithm

Here, we introduce another new main algorithm, Algorithm 6, whose computational cost may not be given in polynomials but might better determine ϵ-feasible solutions.

B.1   Outline of Algorithm 6

The procedures of Algorithm 6 are almost identical to Algorithm 2, except for one of the termination criteria (the criterion indicating the non-existence of ε-feasible solutions). Specifically, to set the upper bound for the minimum eigenvalue of any feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$, Algorithm 2 focuses on the product $\det(\bar{x})$ of the eigenvalues of the arbitrary feasible solution $\bar{x}$ of the scaled problem $\mathrm{P}_{S_\infty}(\mathcal{A}^k Q^k)$, while Algorithm 6 focuses on the sum $\langle \bar{x}, e \rangle$ of the eigenvalues. Algorithm 6 works as follows.

1. If a feasible solution of $\mathrm{P}(\mathcal{A})$ or $\mathrm{D}(\mathcal{A})$ is returned from the basic procedure, the feasibility of $\mathrm{P}(\mathcal{A})$ can be determined, and we stop the main algorithm.

2. If the basic procedure returns the sets of indices $H_1, \ldots, H_p$ and the sets of primitive idempotents $C_1, \ldots, C_p$ that construct the corresponding Jordan frames, then

   in Algorithm 6:

   (a) if $\frac{r_\ell}{\left(r_\ell + \left(\frac{1}{\xi} - 1\right) m_\ell\right)} < \varepsilon$ holds for some $\ell \in \{1, \ldots p\}$, we determine that $\mathrm{P}_{S_\infty}(\mathcal{A})$ has no ε-feasible solution according to Proposition B.1 and stop the main algorithm,

   (b) if $\frac{r_\ell}{\left(r_\ell + \left(\frac{1}{\xi} - 1\right) m_\ell\right)} \geq \varepsilon$ holds for any $\ell \in \{1, \ldots p\}$, we rescale the problem and call the basic procedure again.

Table 7: Upper bounds on the number of iterations of the main algorithms (cf. proposition 5.2 and proposition B.2)

Main Algorithm	Upper bound on # of iterations
Algorithm 2	$-\frac{r}{\log \xi} \log \left(\frac{1}{\varepsilon}\right) - p + 1$
Algorithm 6	$\frac{\xi}{1-\xi} \left(\frac{1}{\varepsilon} - 1\right) r - p + 1$

Table 7 lists upper bounds on the numbers of iterations required by Algorithms 2 and 6. As shown in the table, Algorithm 2 can be said to be a polynomial-time algorithm, but Algorithm 6 is not. On the other hand, the results of the numerical experiments in Appendix C.2 show that Algorithm 6 is superior to Algorithm 2 at detecting ε-feasibility for the generated instances.

Algorithm 6 Main algorithm using another criteria for ε-feasibility

---
1: **Input:** $\mathcal{A}$, $\mathcal{K}$, ε and a constant ξ such that $0 < \xi < 1$
2: **Output:** a solution to P($\mathcal{A}$) or D($\mathcal{A}$) or a certificate that there is no ε feasible solution.
3: $k \leftarrow 1$, $\mathcal{A}^1 \leftarrow \mathcal{A}$, $m_\ell \leftarrow 0$, $\bar{Q}_\ell \leftarrow I_\ell$, $RP_\ell \leftarrow I_\ell$, $RD_\ell \leftarrow I_\ell$ for all $\ell \in \{1, \ldots, p\}$
4: Compute $P_{\mathcal{A}^k}$ and call the basic procedure with $P_{\mathcal{A}^k}$, $\frac{1}{r}e$, ξ
5: **if** basic procedure returns z **then**
6:     **stop** main algorithm and **return** RPz (RPz is a feasible solution of P($\mathcal{A}$))
7: **else if** basic procedure returns y or v **then**
8:     **stop** main algorithm and **return** RDy or RDv (RDy or RDv is a feasible solution of $D(\mathcal{A})$)
9: **else if** basic procedure returns $H_1^k, \ldots, H_p^k$ and $C_1^k, \ldots, C_p^k$ **then**
10:     **for** $\ell \in \{1, \ldots, p\}$ **do**
11:         **if** $|H_\ell^k| > 0$ **then**
12:             $g_\ell \leftarrow \sqrt{\xi} \sum_{h \in H_\ell^k} c^k(v_\ell)_h + \sum_{h \notin H_\ell^k}^{r_\ell} c^k(v_\ell)_h$
13:             $Q_\ell \leftarrow Q_{g_\ell}$, $RP_\ell \leftarrow RP_\ell Q_{g_\ell}$, $RD_\ell \leftarrow RD_\ell Q_{g_\ell^{-1}}$
14:             $m_\ell \leftarrow \left\langle \bar{Q}_\ell \left(\sum_{h \in H_\ell^k} c^k(v_\ell)_h \right), e_\ell \right\rangle + m_\ell$
15:             **if** $\frac{r_\ell}{\left(r_\ell + (\frac{1}{\xi} - 1)m_\ell\right)} \leq \varepsilon$ **then**
16:                 **stop** main algorithm. There is no ε feasible solution.
17:             **end if**
18:             $\bar{Q}_\ell \leftarrow \bar{Q}_\ell Q_{g_\ell^{-1}}$
19:         **else**
20:             $Q_\ell \leftarrow I_\ell$
21:         **end if**
22:     **end for**
23: **else**
24:     **return** basic procedure error
25: **end if**
26: Let $Q^k = (Q_1, \ldots, Q_p)$
27: $\mathcal{A}^{k+1} \leftarrow \mathcal{A}^k Q^k$, $k \leftarrow k + 1$. Go back to line 4.

B.2 Finite termination of Algorithm 6

Here, we prove finite termination of Algorithm 6. Before going into the proof, we explain Algorithm 6 in more detail than in Appendix B.1. In Algorithm 6, when a cut is obtained in the ℓ-th block, it computes the value of $\left\langle \bar{Q}_\ell \left(\sum_{h \in H_\ell^k} c^k(v_\ell)_h \right), e_\ell \right\rangle$ and stores its cumulative value in m_ℓ. In fact, using this m_ℓ, we can compute an upper bound for $\lambda_{\min}(x)$ (Proposition B.1). On line 18, $\bar{Q}_\ell$ is updated as $\bar{Q}_\ell \leftarrow \bar{Q}_\ell Q_{g_\ell^{-1}}$, and $\bar{Q}_\ell$ of Algorithm 6 plays the role of an operator that gives the relation $\langle \bar{x}_\ell, a_\ell \rangle = \langle x_\ell, \bar{Q}_\ell(a_\ell) \rangle$ for the solution x of the original problem, the solution $\bar{x}$ of the scaled problem, and any $a \in \mathbb{E}$. For example, as before, if $|H_\ell^1| > 0$ for $k = 1$, then $\langle \bar{x}_\ell, a_\ell \rangle = \langle Q_{g_\ell^{-1}}(x_\ell), a_\ell \rangle = \langle x_\ell, Q_{g_\ell^{-1}}(a_\ell) \rangle$ is valid. And if $|H_\ell^2| > 0$ even for $k = 2$, then the proposed method scales $\bar{x}$ again, so that $\langle \bar{\bar{x}}_\ell, a_\ell \rangle = \langle \bar{x}_\ell, Q_{g_\ell^{-1}}(a_\ell) \rangle = \langle x_\ell, \bar{Q}_\ell(a_\ell) \rangle$ holds.

Proposition 5.1 guarantees that ε-feasibility of the problem P($\mathcal{A}$) can be detected by computing $\det(\bar{x})$ of any feasible solution of $P_{S_\infty}(\mathcal{A}^k Q^k)$. The following proposition ensures that we may use the value $\langle \bar{x}, e \rangle$ of any feasible solution of $P_{S_\infty}(\mathcal{A}^k Q^k)$ to detect the ε-feasibility of problem P($\mathcal{A}$), instead of $\det(\bar{x})$. While the analysis of the computational complexity in section 5.2 does not hold for it, the new criteria is better able to detect ε-feasibility in the numerical experiments presented in Appendix C.2 .

Proposition B.1. *After k iterations of Algorithm 6, for any feasible solution x of $P_{S_\infty}(\mathcal{A})$ and $\ell \in$*

$\{1, \ldots, p\}$, *the ℓ-th block element x_ℓ of x satisfies*

$$\lambda_{\min}(x_\ell) \leq \frac{r_\ell}{\left(r_\ell + \left(\frac{1}{\xi} - 1\right) m_\ell\right)}. \tag{24}$$

Proof. In Algorithm 6, m_ℓ is updated only when $|H_\ell^k| > 0$. Suppose that, at the end of the k-th iteration of Algorithm 6, the last update of m_ℓ had been at the $k'(\leq k)$-th iteration. Then, the stopping criteria of the basic procedure guarantees that at the beginning of the k'-th iteration, $\bar{Q}_\ell$ satisfies

$$\langle x, \bar{Q}_\ell(c^{k'}(v_\ell)_i)\rangle \leq \begin{cases} \xi & i \in H_\ell^{k'} \\ 1 & i \notin H_\ell^{k'} \end{cases}. \tag{25}$$

This gives a lower bound for $|H_\ell^{k'}|$:

$$\frac{1}{\xi}\left\langle x, \bar{Q}_\ell\left(\sum_{i \in H_\ell^{k'}} c^{k'}(v_\ell)_i\right)\right\rangle \leq |H_\ell^{k'}|. \tag{26}$$

Using the fact that $x_\ell - \lambda_{\min}(x_\ell)e_\ell \in \mathcal{K}_\ell$, we obtain

$$\lambda_{\min}(x_\ell)\langle e_\ell, \bar{Q}_\ell(e_\ell)\rangle \leq \langle x_\ell, \bar{Q}_\ell(e_\ell)\rangle$$

$$= \left\langle x_\ell, \bar{Q}_\ell\left(\sum_{j \notin H_\ell^{k'}} c^{k'}(v_\ell)_j\right)\right\rangle + \left\langle x_\ell, \bar{Q}_\ell\left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j\right)\right\rangle$$

$$\leq r_\ell - |H_\ell^{k'}| + \left\langle x_\ell, \bar{Q}_\ell\left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j\right)\right\rangle \quad \text{(by (25))}$$

$$\leq r_\ell - \left(\frac{1}{\xi} - 1\right)\left\langle x_\ell, \bar{Q}_\ell\left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j\right)\right\rangle \quad \text{(by (26))}$$

$$\leq r_\ell - \left(\frac{1}{\xi} - 1\right)\lambda_{\min}(x_\ell)\left\langle e_\ell, \bar{Q}_\ell\left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j\right)\right\rangle,$$

and hence,

$$\lambda_{\min}(x_\ell)\left(\langle e_\ell, \bar{Q}_\ell(e_\ell)\rangle + \left(\frac{1}{\xi} - 1\right)\left\langle e_\ell, \bar{Q}_\ell\left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j\right)\right\rangle\right) \leq r_\ell. \tag{27}$$

Next, suppose that, at the beginning of the k'-th iteration of Algorithm 6, the last update of m_ℓ had been performed at the $i(< k')$-th iteration.

Let $\bar{Q}_\ell^{\text{pre}}$ be $\bar{Q}_\ell$ obtained at the beginning of the i-th iteration of Algorithm 6, and let $Q_{g_\ell}^{\text{pre}}$ and m_ℓ^{pre} be Q_ℓ and m_ℓ obtained after the update at the i-th iteration. Note that $\bar{Q}_\ell$ at the beginning of the k'-th

iteration of Algorithm 6 can be represented by $\bar{Q}_\ell = \bar{Q}_\ell^{\text{pre}} Q_{g_\ell^{-1}}^{\text{pre}}$. Thus, from (12), we see that

$$Q_{g_\ell^{-1}}^{\text{pre}}(e_\ell) = Q_{g_\ell^{-1}}^{\text{pre}}\left(\sum_{j=1}^{r_\ell} c^i(v_\ell)_j\right)$$

$$= Q_{g_\ell^{-1}}^{\text{pre}}\left(\sum_{j \in H_\ell^{k'}} c^i(v_\ell)_j\right) + Q_{g_\ell^{-1}}^{\text{pre}}\left(\sum_{j \notin H_\ell^{k'}} c^i(v_\ell)_j\right)$$

$$= \frac{1}{\xi}\sum_{j \in H_\ell^{k'}} c^i(v_\ell)_j + \sum_{j \notin H_\ell^{k'}} c^i(v_\ell)_j$$

$$= e_\ell + \left(\frac{1}{\xi} - 1\right)\sum_{j \in H_\ell^i} c^i(v_\ell)_j$$

and hence,

$$\bar{Q}_\ell(e_\ell) = \bar{Q}_\ell^{\text{pre}} Q_{g_\ell^{-1}}^{\text{pre}}(e_\ell) = \bar{Q}_\ell^{\text{pre}}\left(e_\ell + \left(\frac{1}{\xi} - 1\right)\sum_{j \in H_\ell^i} c^i(v_\ell)_j\right)$$

$$= \bar{Q}_\ell^{\text{pre}}(e_\ell) + \left(\frac{1}{\xi} - 1\right)\bar{Q}_\ell^{\text{pre}}\left(\sum_{j \in H_\ell^i} c^i(v_\ell)_j\right). \tag{28}$$

By recursively applying (28) to $\bar{Q}_\ell^{\text{pre}}(e_\ell)$, we finally obtain

$$\left\langle e_\ell, \bar{Q}_\ell(e_\ell)\right\rangle = r_\ell + \left(\frac{1}{\xi} - 1\right)m_\ell^{\text{pre}}.$$

Let $m_\ell^{k'}$ be the value of m_ℓ obtained after the update at the k'-th iteration. Then,

$$m_\ell^{k'} = m_\ell^{\text{pre}} + \left\langle e_\ell, \bar{Q}_\ell\left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j\right)\right\rangle \tag{29}$$

and, by (27), we obtain

$$\lambda_{\min}(x_\ell) \leq \frac{r_\ell}{\left(r_\ell + \left(\frac{1}{\xi} - 1\right)m_\ell^{\text{pre}} + \left(\frac{1}{\xi} - 1\right)\left\langle e_\ell, \bar{Q}_\ell\left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j\right)\right\rangle\right)}$$

$$= \frac{r_\ell}{\left(r_\ell + \left(\frac{1}{\xi} - 1\right)m_\ell^{k'}\right)}.$$

Since, at the end of the k-th iteration of Algorithm 6, the last update of m_ℓ was at the k'-th iteration, we see that $m_\ell = m_\ell^{k'}$, and hence (24) holds after k iterations of Algorithm 6. $\qquad\square$

Using Proposition B.1, we find an upper bound on the number of iterations of Algorithm 6.

Proposition B.2. *Algorithm 6 terminates after no more than $\frac{\xi}{1-\xi}\left(\frac{1}{\varepsilon} - 1\right)r - p + 1$ iterations.*

Proof. When $|H_\ell^k| > 0$ for $\ell \in \{1, \ldots, p\}$ at the k-th iteration of Algorithm 6, we say that the iteration is "good" for the ℓ-th block. From Proposition B.1, since the (meaningful) upper bound of the minimum eigenvalue $\lambda_{\min}(x_\ell)$ of x_ℓ of the ℓ-th block of any feasible solution x of $\mathrm{P}_{S_\infty}(\mathcal{A})$ depends on the value of m_ℓ, we first calculate a lower bound for the increment of m_ℓ per good iteration in the ℓ-th block. Similar to the proof of Proposition B.1, suppose that the k'-th iteration is a good iteration for the ℓ-th block. As shown in equation (29), the value of m_ℓ is increased at this time by $\left\langle e_\ell, \bar{Q}_\ell \left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j \right) \right\rangle$ using $\bar{Q}_\ell$ at the beginning of the k'-th iteration. Let us express $Q_{g_\ell^{-1}}$ using g_ℓ obtained at the k-th iteration as $Q_{g_\ell^{-1}}^k$, i.e., $\bar{Q}_\ell = Q_{g_\ell^{-1}}^1 Q_{g_\ell^{-1}}^2 \ldots Q_{g_\ell^{-1}}^{k'-1}$. Then, the increment of m_ℓ at the k'-th iteration is as follows:

$$\left\langle e_\ell, \bar{Q}_\ell \left(\sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j \right) \right\rangle = \left\langle Q_{g_\ell^{-1}}^{k'-1} \ldots Q_{g_\ell^{-1}}^1 (e_\ell), \sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j \right\rangle. \tag{30}$$

Note that $Q_{g_\ell^{-1}}^{k'-1} \ldots Q_{g_\ell^{-1}}^1 (e_\ell) - e_\ell \in \mathcal{K}_\ell$ holds, as we will prove below using induction. First, if the first iteration is a good one for the ℓ block, then $Q_{g_\ell^{-1}}^1 (e_\ell) = \frac{1}{\xi} \sum_{i \in H_\ell^1} c^1(v_\ell)_i + \sum_{j \notin H_\ell^1} c^1(v_\ell)_j = e_\ell + \left(\frac{1}{\xi} - 1 \right) \sum_{i \in H_\ell^1} c^1(v_\ell)_i$, and if it is not a good iteration, then $Q_{g_\ell^{-1}}^1 (e_\ell) = e_\ell$. Thus, $Q_{g_\ell^{-1}}^1 (e_\ell) - e_\ell \in \mathcal{K}_\ell$ holds. Next, when $Q_{g_\ell^{-1}}^i \ldots Q_{g_\ell^{-1}}^1 (e_\ell) - e_\ell \in \mathcal{K}$ holds, by Proposition 2.4, $Q_{g_\ell^{-1}}^{i+1} \left(Q_{g_\ell^{-1}}^i \ldots Q_{g_\ell^{-1}}^1 (e_\ell) - e_\ell \right) \in \mathcal{K}_\ell$ holds. Furthermore, the same calculation as in the first iteration yields $Q_{g_\ell^{-1}}^{i+1} (e_\ell) - e_\ell \in \mathcal{K}_\ell$, and we see that

$$Q_{g_\ell^{-1}}^{i+1} \left(Q_{g_\ell^{-1}}^i \ldots Q_{g_\ell^{-1}}^1 (e_\ell) - e_\ell \right) \in \mathcal{K}_\ell \Leftrightarrow Q_{g_\ell^{-1}}^{i+1} Q_{g_\ell^{-1}}^i \ldots Q_{g_\ell^{-1}}^1 (e_\ell) - Q_{g_\ell^{-1}}^{i+1} (e_\ell) \in \mathcal{K}_\ell$$
$$\Rightarrow Q_{g_\ell^{-1}}^{i+1} Q_{g_\ell^{-1}}^i \ldots Q_{g_\ell^{-1}}^1 (e_\ell) - e_\ell \in \mathcal{K}_\ell.$$

Thus, from (30), we obtain a lower bound for the increment of m_ℓ as

$$\left\langle Q_{g_\ell^{-1}}^{k'-1} \ldots Q_{g_\ell^{-1}}^1 (e_\ell), \sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j \right\rangle \geq \left\langle e_\ell, \sum_{j \in H_\ell^{k'}} c^{k'}(v_\ell)_j \right\rangle$$
$$= |H_\ell^{k'}| \geq 1,$$

which means that the value of m_ℓ increases by at least 1 per good iteration. Therefore, if the number of good iterations for the ℓ-th block is $\left(\frac{r_\ell}{\varepsilon} - r_\ell \right) \left(\frac{\xi}{1-\xi} \right)$ or more, then from Proposition B.1, we can conclude that $\lambda_{min}(x_\ell) \leq \varepsilon$ holds; i.e., we obtain an upper bound for the number of iterations of Algorithm 6 as follows:

$$\sum_{\ell=1}^{p} \left(\left(\frac{r_\ell}{\varepsilon} - r_\ell \right) \left(\frac{\xi}{1-\xi} \right) - 1 \right) + 1 = \frac{\xi}{1-\xi} \left(\frac{1}{\varepsilon} - 1 \right) r - p + 1.$$

$\square$

It should be noted that the number of iterations required by Algorithm 6 to detect the non-existence of ε-feasible solutions is actually likely to be much smaller than the value given in Proposition B.2. This is because Proposition B.2 calculates the lower bound for the increment of m_ℓ for one good iteration as 1. The increment of m_ℓ can be calculated using $\bar{Q}_\ell$, but it is difficult to calculate the exact increment of m_ℓ because $\bar{Q}_\ell$ depends on the results returned by the previous basic procedure. Suppose that both the first and second iterations are good for the ℓ-th block. Then, the increment of m_ℓ at the second iteration is

$$\left\langle Q_{g_\ell^{-1}}^1 (e_\ell), \sum_{j \in H_\ell^2} c^2(v_\ell)_j \right\rangle = \left\langle e_\ell + \left(\frac{1}{\xi} - 1 \right) \sum_{i \in H_\ell^1} c^1(v_\ell)_i, \sum_{j \in H_\ell^2} c^2(v_\ell)_j \right\rangle,$$

but it is difficult to find a lower bound greater than 0 for $\left\langle \sum_{i \in H_\ell^1} c^1(v_\ell)_i, \sum_{j \in H_\ell^2} c^2(v_\ell)_j \right\rangle$.

37

C Additional numerical experiments

In addition to the strongly feasible instances described in section 7.2, we generated the following two types of instances and conducted numerical experiments.

- Weakly feasible instances, i.e., $\ker \mathcal{A} \cap \mathbb{S}_{++}^n = \emptyset$, but $\ker \mathcal{A} \cap \mathbb{S}_+^n \setminus \{O\} \neq \emptyset$.

- Infeasible instances, i.e., $\ker \mathcal{A} \cap \mathbb{S}_+^n = O$.

C.1 How to generate instances used in the additional experiments

Here, we describe how the weakly feasible instances and infeasible instances were generated. Note that, due to the rounding error of the numerical computation, the weakly (ill-conditioned strongly) feasible instances generated in this experiment may not have been weakly (ill-conditioned strongly) feasible, and could be infeasible or interior feasible. Thus, the term "fragilely" may not be appropriate, but it would be better to call them "fragilely weakly feasible (fragilely ill-conditioned strongly)."

C.1.1 Weakly feasible instances

The weakly feasible instances were generated by Algorithm 7.

Algorithm 7 Weakly feasible instance

1: **Input:** $n, m, A' = [\ ]$
2: **Output:** A
3: $B \leftarrow \mathrm{rand}(n, n)$ // B must not be O
4: $C \leftarrow \frac{B + B^T}{2}$ // $C \neq O$ must not be $C \succeq O$ or $C \preceq O$
5: $C_+ \leftarrow \mathcal{P}_{\mathbb{S}_+^n}(C)$ // $C_+ \neq O$ since $C \neq O$ is not negative semidefinite.
6: $C_- \leftarrow -\mathcal{P}_{\mathbb{S}_+^n}(-C)$ // $C_- \neq O$ since $C \neq O$ is not positive semidefinite.
7: $c_+ \leftarrow \mathrm{vec}(C_+)$ and $R \leftarrow I - \frac{1}{\|c_+\|_2^2} c_+ c_+^T$
8: **for** $i = 1$ to $m - 1$ **do**
9:      $A_i' \leftarrow \mathrm{rand}(n, n)$ and $A_i \leftarrow \left(A_i' + (A_i')^T\right)/2$
10:      $A' \leftarrow \begin{pmatrix} A' \\ \mathrm{vec}(A_i)^T \end{pmatrix}$
11: **end for**
12: $A \leftarrow \begin{pmatrix} \mathrm{vec}(C_-)^T \\ A'R \end{pmatrix}$

Proposition C.1. *For any $A \in \mathbb{R}^{m \times n^2}$ returned by Algorithm 7, no $X \in \mathbb{S}_{++}^n$ exists that satisfies $A(\mathrm{vec}(X)) = \mathbf{0}$, but an $X \in \mathbb{S}_+^n \setminus \{O\}$ exists that satisfies $A(\mathrm{vec}(X)) = \mathbf{0}$.*

Proof. First, we show that an $X \in \mathbb{S}_+^n \setminus \{O\}$ exists that satisfies $A(\mathrm{vec}(X)) = \mathbf{0}$. For the matrix $C_+ \in \mathbb{S}_+^n$ computed on line 5 of Algorithm 7, we see that $C_+ \neq O$ and the following holds:

$$A(\mathrm{vec}(C_+)) = Ac_+ = \begin{pmatrix} \mathrm{vec}(C_-)^T \\ A'R \end{pmatrix} c_+ = \begin{pmatrix} \mathrm{vec}(C_-)^T c_+ \\ A'Rc_+ \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ A'(c_+ - c_+) \end{pmatrix} = \mathbf{0}.$$

Next, we show by contradiction that no $X \in \mathbb{S}_{++}^n$ exists that satisfies $A(\mathrm{vec}(X)) = \mathbf{0}$. Suppose that an $X \in \mathbb{S}_{++}^n$ satisfies $A(\mathrm{vec}(X)) = \mathbf{0}$. Since the first row of A is $\mathrm{vec}(C_-)^T$, if $A(\mathrm{vec}(X)) = \mathbf{0}$ holds, then

$\mathrm{vec}(C_-)^T\mathrm{vec}(X) = 0$, i.e.,

$$\mathrm{vec}(C_-)^T\mathrm{vec}(X) = \langle C_-, X \rangle = \langle PDP^T, QEQ^T \rangle$$

$$= \langle D, P^T QEQ^T P \rangle = \sum_{i=1}^{n} D_{ii} \left(P^T QEQ^T P \right)_{ii} = 0$$

where $C_- = PDP^T$, $X = QEQ^T$, P are Q orthogonal matrices, and D and E are diagonal matrices. Here, $X \in \mathbb{S}^n_{++}$ implies $\left(P^T QEQ^T P \right)_{ii} > 0$ for any $i \in \{1, \dots, n\}$ and hence, D should be O, but this contradicts $C_- \neq O$. Thus, no $X \in \mathbb{S}^n_{++}$ exists satisfying $A\left(\mathrm{vec}(X)\right) = \mathbf{0}$. $\qquad\square$

C.1.2   Infeasible instances

The infeasible instances were generated by Algorithm 8. If we define the linear operator $\mathcal{A} : \mathbb{S}^n \to \mathbb{R}^m$ as $\mathcal{A}(X) = (\langle A_1, X \rangle, \dots, \langle A_m, X \rangle)^T$, then by choosing $A_1 \in \mathbb{S}^n_{++}$, we obtain $\mathcal{A}$ such that $\ker \mathcal{A} \cap \mathbb{S}^n_+ = \{O\}$. On the basis of this observation, by introducing a parameter $\alpha > 0$, we generated a positive definite matrix A_1 whose minimum eigenvalue is a uniformly distributed random number in $(0, \alpha)$. We chose $\alpha \in \{1e-1, 1e-2, 1e-3, 1e-4, 1e-5\}$. The input of Algorithm 8 consisted of the rank of the semidefinite cone n, the number of constraints m, an arbitrary orthogonal matrix P, and the parameter $\alpha > 0$.

Algorithm 8 Infeasible instance

1: **Input:** $n, m, \alpha, P, A' = [\ ]$
2: **Output:** A
3: $B \leftarrow \mathrm{rand}(n, n)$
4: $B' \leftarrow \frac{B+B^T}{2}$ and then compute an orthogonal matrix Q and diagonal matrix E such that $B' = QDQ^T$
5: $E_+ = \mathrm{rand}(1) \times \alpha I + \mathcal{P}_{\mathbb{S}^n_+}(E)$
6: $d \leftarrow \mathrm{rand}(n)$ and $D \leftarrow \mathrm{diag}(d)$
7: $B_+ \leftarrow QE_+Q^T$ and $C \leftarrow PDP^T$
8: $c = \mathrm{vec}(C)$ and $R \leftarrow I - \frac{1}{\|c\|_2^2} cc^T$
9: **for** $i = 1$ to $m - 1$ **do**
10: $\quad A'_i \leftarrow \mathrm{rand}(n, n)$ and $A_i \leftarrow \left(A'_i + (A'_i)^T \right)/2$
11: $\quad A' \leftarrow \begin{pmatrix} A' \\ \mathrm{vec}(A_i)^T \end{pmatrix}$
12: **end for**
13: $A \leftarrow \begin{pmatrix} \mathrm{vec}(B_+)^T \\ A'R \end{pmatrix}$

Note that the first row of the matrix A returned by Algorithm 8 is $\mathrm{vec}(B_+)^T$. Since $B_+ \in \mathbb{S}^n_{++}$, we see that $\mathrm{vec}(B_+)^T\mathrm{vec}(X) > 0$ for any positive definite matrix $X \in \mathbb{S}^n_{++}$. Thus, there is no $X \in \mathbb{S}^n_{++}$ satisfying $A\left(\mathrm{vec}(X)\right) = \mathbf{0}$, which implies that the generated instance is infeasible.

C.2   Additional numerical results and observations

As in section 7, we set the size of the semidefinite matrix to $n = 50$ and the number of constraints m using $\nu \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. For each $\nu \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, we generated five instances, i.e., 25 instances for a weakly feasible case and 25 instances for each of five infeasible cases (corresponding to five patterns of $\alpha = 1e\text{-}1, \dots, \alpha = 1e\text{-}5$, see section C.1.2 for details). Thus, we generated 25 weakly feasible instances and 125 infeasible instances. We set the upper limit of the execution time to 2 hours and compared the performance of our method (Algorithm 2, 6) with those of Lourenço (2019) and Pena(2017).

We classified the output-results into five types: A: an interior feasible solution is found; B: no interior feasible solution is found (ver.1); C: no ε-feasible solution is found (only for Lorenço (2019) and our method); D: no interior feasible solution is found (ver.2; only for Pena (2017)); E: Out-of-time. In what follows, we briefly explain how output-result type D for Pena (2017) differs from output-result type B.

[13] pointed out that if $P(\mathcal{A})$ has no interior feasible solution, meaning that if the main algorithm of Pena (2017) is applied to only $P(\mathcal{A})$, it does not stop within a finite number of iterations. To overcome this problem, Pena et al. constructed the main algorithm in a way that it applies not only to $P(\mathcal{A})$ but also to problem $Q(\mathcal{A})$:

$$Q(\mathcal{A}) \quad \text{find} \quad X \in \mathbb{S}_{++}^n \quad \text{s.t.} \quad X \in \text{range}\,\mathcal{A}^*.$$

Accordingly, we defined output-result type B as the case where a feasible solution of $D(\mathcal{A})$ is obtained by applying the main algorithm to $P(\mathcal{A})$ and defined output-result type D as the case where a feasible solution of $Q(\mathcal{A})$ is obtained by applying the main algorithm to $Q(\mathcal{A})$.

Table 8 summarizes the results for infeasible instances. Similarly to Table 4, the "CO-ratio and "times(s)" columns respectively show the ratio of correct outputs and the average CPU time of each method (the values in parentheses () in rows $\alpha = $ 1e-4 and $\alpha = $ 1e-5 are the average CPU times of each method excluding the instances for which the method ended up running out of time). When using MVN as the basic procedure, whereas our method and Lourenço (2019) found an element of $\text{range}\,\mathcal{A}^* \cap \mathbb{S}_+^n$ for all instances, Pena (2017) ended up running out of time for one instance for $\alpha = $ 1e-4 and $\alpha = $ 1e-5.

From the results for infeasible instances, we can observe the following three points. First, our method obtained correct outputs for every instance in a short execution time. This would be because it employed an efficient scaling and found an element of $\text{range}\,\mathcal{A}^* \cap \mathbb{S}_+^n$. Second, the method of Pena (2017) obtained better results when SP was used as the basic procedure. As shown in Table 8, the method of Pena (2017) using SP as the basic procedure solved all problems and had shorter execution times than the method using MVN. Since Pena's (2017) method calls the basic procedure not only to find points in $\ker\mathcal{A} \cap \mathbb{S}_{++}^n$ but also to find points in $\text{range}\,\mathcal{A}^* \cap \mathbb{S}_{++}^n$, using SP, which can update basic procedures efficiently, is better than using MVN in terms of execution time. Third, it is not always possible to detect infeasibility (i.e., to find a point in $\text{range}\,\mathcal{A}^* \cap \mathbb{S}_+^n$) in a shorter execution time when using SP than when using MVN. In fact, according to Lourenço (2019), the execution time is shorter when using MVN as the basic procedure than when using SP. SP is a more efficient update method than MVN in terms of satisfying a termination criterion (the criterion for moving to scaling) of the basic procedure. On the other hand, from the point of view of finding points in $\text{range}\,\mathcal{A}^* \cap \mathbb{S}_+^n$, it is not possible to determine whether SP or MVN is more suitable. Pena (2017) used SP to significantly reduce the execution time, which is the result of updating the basic procedure for finding points in $\text{range}\,\mathcal{A}^* \cap \mathbb{S}_{++}^n$ more efficiently than MVN. Mosek obtained a point in $\text{range}\,\mathcal{A}^* \cap \mathbb{S}_{++}^n$ as a feasible solution to the dual problem for all instances. From the viewpoint of execution time, Mosek was superior to the other methods.

For the weakly feasible instances, we compared our method (Algorithm 2), a modified version with another criteria for ε-feasibility (Algorithm 6), Lourenço (2019), and Pena (2017). The results are summarized in Table 9. As described above, we classified the output-results into type A: an interior feasible solution is found; type B: no interior feasible solution is found (ver.1); type C: no ε-feasible solution is found (only for Lorenço (2019) and our methods); type D: no interior feasible solution is found (ver.2; only for Pena (2017)); type E: Out-of-time. Note that B^* indicates that the output was B, but when we converted the obtained solution to a solution of $D(\mathcal{A})$, it contained a negative eigenvalue and violated the SDP constraint. Note that, due to rounding errors, the true state of each generated weakly feasible instance is unknown, and it is impossible to determine whether the results obtained by the algorithms are correct or incorrect. Thus, Table 9 lists the output types and average execution time without noting which are correct.

Table 8: Results for infeasible instances

Instance	BP	Algorithm 2		Lourenço (2019)		Pena (2017)		Mosek	
		CO-ratio	time(s)	CO-ratio	time(s)	CO-ratio	time(s)	CO-ratio	time(s)
$\alpha = $ 1e-1	MVN	25/25	1.23	25/25	2.37	25/25	0.79	25/25	1.22
	SP	25/25	1.01	25/25	21.46	25/25	0.61		
$\alpha = $ 1e-2	MVN	25/25	4.39	25/25	37.93	25/25	25.99	25/25	1.25
	SP	25/25	3.87	25/25	62.92	25/25	1.05		
$\alpha = $ 1e-3	MVN	25/25	5.38	25/25	61.61	25/25	61.55	25/25	1.25
	SP	25/25	5.34	25/25	84.08	25/25	2.08		
$\alpha = $ 1e-4	MVN	25/25	7.81	25/25	88.32	24/24	(20.80)	25/25	1.24
	SP	25/25	7.40	25/25	98.79	25/25	33.48		
$\alpha = $ 1e-5	MVN	25/25	9.08	25/25	76.17	24/24	(9.47)	25/25	1.24
	SP	25/25	8.00	25/25	91.88	25/25	55.42		

Table 9: Output types for weakly feasible instances

Method	BP	$\nu = 0.1$	$\nu = 0.3$	$\nu = 0.5$	$\nu = 0.7$	$\nu = 0.9$	time(s)
Algorithm 2	MVN	AAAAA	AAAAA	AAAAA	AAAAA	BBBBB	414.42
	SP	AAAAA	AAAAA	AAAAA	AAAAA	ABABB	226.25
Algorithm 6	MVN	CCCCC	CCCCC	CCCCC	CCCCC	CCCCC	301.97
	SP	CCCCC	CCCCC	CCCCC	CCCCC	CCCCC	179.72
Lourenço (2019)	MVN	AAAAA	BB*B*B*B	ABAAA	ABAB*B*	BBBBB	3512.78
	SP	AAAAA	AAAAA	AAAAA	AAAAA	BBBBB	1550.76
Pena (2017)	MVN	EEEEE	EEEEE	EEEEE	EEEEE	EEEEE	
	SP	AAAAA	DAAAD	AAAAA	AAAAA	DDDDD	3239.12

From Table 9, we can observe the following:

- For all the methods, the average execution time was shorter when SP was used as the basic procedure than when MVN was used.

- All methods except Algorithm 6 sometimes obtained output type A (an interior feasible solution is found), and Pena(2017) returned output-result D, while the obtained solution had $0 \sim 5$ negative eigenvalues (about -1e-16) and more than 20 positive eigenvalues (less than 1e-12) when we converted it into a solution of $P(\mathcal{A})$.

- Lourenço (2019) obtained output type B* (no interior feasible solution is found) but when we converted the obtained solution into a solution of $D(\mathcal{A})$, it contained a negative eigenvalue and violated the SDP constraint). The obtained solution had $1 \sim 3$ negative eigenvalues (about -1e-6) and violated the SDP constraint when we converted it into a solution of $D(\mathcal{A})$.

- Our modified method (Algorithm 6) was able to determine the existence of an ε-feasible solution for all instances. This implies that, at least for this specific set of weakly feasible instances, the criteria focusing on the total value of the eigenvalues used in Algorithm 6 is more suitable than the criteria focusing on the product of all the eigenvalues.

Table 10 summarizes the results obtained by Mosek. The error message "rescode = 10006" was obtained

Table 10: Results for weakly feasible instances with Mosek

Instance	time(s)	$\|\mathcal{A}(X^*)\|_2$
weakly feasible	5.42	6.86e-9

Table 11: Average of the constraint residuals $\|\mathcal{A}(X^*)\|_2$ of the solution X^* obtained for the weakly feasible instances

Value of ν	Algorithm 2	Lourenço (2019)	Mosek
$\nu = 0.1$	1.28e-13	5.51e-14	1.45e-12
$\nu = 0.3$	1.56e-13	7.04e-14	2.53e-10
$\nu = 0.5$	1.40e-13	1.05e-13	1.29e-9
$\nu = 0.7$	3.44e-13	1.09e-13	3.75e-9

for 22 instances, similar to the results for the strongly feasible ill-conditioned instances. Note that we assumed that feasible solutions were obtained for all instances since the constraint residual $\|\mathcal{A}(X^*)\|_2$ was as small as 1.1e-7 or less for all obtained solutions.

Note that for all problems with $0.1 \leq \nu \leq 0.7$, Algorithm 2 and Lourenço (2019) using SP for the basic procedure returned output A, i.e., a feasible solution to the original problem. Table 11 summarizes the accuracies of the solutions obtained with Algorithm 2, Lourenço (2019), and Mosek for all instances with $0.1 \leq \nu \leq 0.7$. Chubanov's methods sometimes returned output-result type A for weakly feasible instances, but Table 11 shows that the average accuracy of feasible solutions obtained by Chubanov's methods was better than that of Mosek.

D   More comparisons of the basic procedures

In section 6.1, we showed that the bound of the computational cost of our method is lower than that of Lourenço et al. when $\mathcal{K}$ is the n-dimensional nonnegative orthant $\mathbb{R}^n_+$ or a Cartesian product of simple second-order cones, and that their bounds on their costs are equivalent when $\mathcal{K}$ is a simple positive semidefinite cone under the assumption that the costs of computing the spectral decomposition and the minimum eigenvalue are the same for an $n \times n$ symmetric matrix. In this section, we make more detailed comparisons of these algorithms in terms of the performance of the cut obtained from the basic procedure and the detectability of an ε-feasible solution. Similarly to section 7, we will refer to Lourenço et al.'s method [10] as Lourenço (2019) throughout this section.

D.1   Performance comparison of the two basic procedures for the simple case

Here, for the sake of simplicity, we will focus on the case where the symmetric cone is simple, i.e., $p = 1$. Let $\mathbb{E}$ be the Euclidean space corresponding to the symmetric cone $\mathcal{K}$. For any given $w, v \in \mathbb{E}$, Lourenço et al. [10] defined $\text{vol}(w, v)$ as the volume of the intersection $H(w, v) \cap \mathcal{K}$, where $H(w, v)$ is the half space given by $H(w, v) = \{x \in \mathbb{E} \mid \langle w, x \rangle \leq \langle w, v \rangle\}$.

In this section, we first identify the half-space $H(w, v)$ that will be transferred to the half-space $H(e, e/r)$

Table 12: Comparison of reduction rates of the two algorithms: Theoretical results

Basic procedure	UB#iter	Reduction rate
Lourenço (2019)	$\rho^2 r_{\max}^2$	$\mathrm{vol}(w,v) = \left(\frac{r^r}{\det w}\right)^{\frac{d}{r}} \mathrm{vol}(e,e/r) \leq \left(e^{-\varphi(\rho)}\right)^{\frac{d}{r}} \mathrm{vol}(e,e/r)$
Algorithm 1	$r_{\max}^2/\xi^2$	$\mathrm{vol}(w,v) = \left(\xi^N\right)^{\frac{d}{r}} \mathrm{vol}(e,e/r)$

after scaling and then find the constant **rate** $\in \mathbb{R}$ that satisfies $\mathrm{vol}(w,v) \leq \mathbf{rate} \times \mathrm{vol}(e,e/r)$, so that we can compare the proposed method and Lourenço (2019). The proposed method and Lourenço (2019) use the basic procedure results to narrow down the original problem's feasible region. It can be interpreted that the algorithm becomes more efficient as the constant **rate** $\in \mathbb{R}$ (indicating how much $\mathrm{vol}(w,v)$ is reduced compared with $\mathrm{vol}(e,e/r)$) gets smaller. In what follows, we call the constant **rate** $\in \mathbb{R}$ the reduction rate.

Section D.1.1 derives the reduction rate of the proposed method and section D.1.2 that of Lourenço (2019). The results in these sections are summarized in Table 12, where the "UB#iter" column shows the upper bound on the number of iterations required in the basic procedure. The "UB#iter" of Lourenço (2019) comes from Proposition 14 of [10] (where the authors showed their result by substituting $\rho = 2$), whereas that of Algorithm 1 comes from Proposition 4.4 with $\ell = 1$. The "Reduction rate" of Lourenço (2019) comes from Theorem D.2, whereas that of Algorithm 1 comes from (33) with $(w,v) = (Q_{g^{-1}}(e), Q_g(e)/r)$. By setting $\rho = 2$ and $\xi = 1/2$, the two bounds in "UB#iter" have the same value; in this case, the reduction rates turn out to be

$$\text{Lourenço (2019):} \left(\frac{r^r}{\det w}\right)^{\frac{d}{r}} \leq \left(e^{-\varphi(2)}\right)^{\frac{d}{r}} \simeq (0.918)^{\frac{d}{r}}, \qquad \text{Algorithm 1:} \left(\xi^N\right)^{\frac{d}{r}} \leq \left(\frac{1}{2}\right)^{\frac{d}{r}}.$$

The above comparison indicates that Algorithm 1 is superior to the basic procedure in [10] in terms of the reduction rate of the feasible region.

D.1.1   Theoretical reduction rate of Algorithm 1

Suppose that Algorithm 1 returns a result such that there exists a nonempty index set $I \subseteq \{1,\ldots,r\}$ with $|I| = N$ for which

$$\langle c_i, x \rangle \leq \begin{cases} \xi & i \in I \\ 1 & i \notin I \end{cases} \tag{31}$$

holds for any feasible solution x of $\mathrm{P}_{S_\infty(A)}$, where $\{c_1,\ldots,c_r\}$ are primitive idempotents that make up a Jordan frame. Note that Algorithm 1 employs the scaling $\bar{x} = Q_{g^{-1}}(x)$ with $g^{-1} = \frac{1}{\sqrt{\xi}}\sum_{i \in I} c_i + \sum_{i \notin I} c_i$. Let us find $w, v \in \mathbb{E}$ which satisfy

$$H(e, e/r) = Q_{g^{-1}}\left(H(w,v)\right). \tag{32}$$

Since (32) and the scaling $\bar{x} = Q_{g^{-1}}(x)$ imply that

$$\begin{aligned} H(w,v) &= Q_g\left(H(e,e/r)\right) \\ &= \{Q_g(\bar{x}) \in \mathbb{E} \ | \ \langle \bar{x}, e \rangle \leq 1\} \\ &= \{Q_g(\bar{x}) \in \mathbb{E} \ | \ \langle Q_g(\bar{x}), Q_{g^{-1}}(e) \rangle \leq 1\} \\ &= \{x \in \mathbb{E} \ | \ \langle x, Q_{g^{-1}}(e) \rangle \leq \langle Q_{g^{-1}}(e), Q_g(e)/r \rangle = 1\}, \end{aligned}$$

by setting $w = Q_{g^{-1}}(e)$ and $v = Q_g(e)/r$, we find that the half space $H(w,v)$ is transformed to $H(e,e/r)$ after the scaling. Since $Q_{g^{-1}}(e) \in \mathrm{int}\,\mathcal{K}$, we can apply the following proposition to $w = Q_{g^{-1}}(e)$.

43

Proposition D.1 (Proposition 6 of [10]). *Suppose that $w \in \text{int}\,\mathcal{K}$. Then,*

$$Q_{w^{-1/2}\sqrt{\langle w,v \rangle}}\left(H(e,e/r)\right) = H(w,v),$$

$$\text{vol}(w,v) = \left(\frac{\langle w,v \rangle}{\sqrt[r]{\det w}}\right)^d \text{vol}(e,e/r).$$

Using the above proposition and the assumption $|I| = N$ for the set I in (31), we can see how the volume $\text{vol}(Q_{g^{-1}}(e), Q_g(e)/r)$ of $H(Q_{g^{-1}}(e), Q_g(e)/r) \cap \mathcal{K}$ decreases compared with $\text{vol}(e,e/r)$:

$$\text{vol}(Q_{g^{-1}}(e), Q_g(e)/r) = \left(\frac{1}{\sqrt[r]{\det Q_{g^{-1}}(e)}}\right)^d \text{vol}(e,e/r)$$

$$= \left(\frac{1}{\sqrt[r]{\frac{1}{\xi^N}}}\right)^d \text{vol}(e,e/r) = \left(\xi^N\right)^{\frac{d}{r}} \text{vol}(e,e/r). \tag{33}$$

D.1.2   Theoretical reduction rate of the basic procedure of Lourenço (2019)

The following theorem gives the reduction rate of the basic procedure of Lourenço (2019).

Theorem D.2 (Theorem 10 of [10]). *Let $\rho > 1$ and $y \in \mathcal{K} \setminus \{0\}$ be such that $F_{\text{P}_{S_1}(A)} \subseteq H(y, e/\rho r)$. Let $\beta = r - \left(\frac{1}{\rho} - \frac{1}{\sqrt{\rho(3\rho-2)}}\right), w = \frac{r-\beta}{\langle y,e \rangle}\rho r y + \beta e, v = w^{-1}$. Then, the following hold:*

1. $F_{\text{P}_S(A)} \subseteq H(y, e/\rho r) \cap H(e, e/r) \subseteq H(w, v)$

2. $Q_{\sqrt{r}w^{-1/2}}\left(H(e, e/r)\right) = H(w, v)$

3.
$$\text{vol}(w,v) = \left(\frac{r^r}{\det w}\right)^{\frac{d}{r}} \text{vol}(e, e/r) \leq \left(\exp\left(-\varphi(\rho)\right)\right)^{\frac{d}{r}} \text{vol}(e, e/r)$$

where $\varphi(\rho) = 2 - \frac{1}{\rho} - \sqrt{3 - \frac{2}{\rho}}$. In particular, if $\rho \geq 2$, we have $\text{vol}(w,v) < (0.918)^{\frac{d}{r}}\text{vol}(e, e/r)$.

D.1.3   Comparison of reduction rates of the two algorithms in numerical experiments

To confirm whether similar reduction rates are observed numerically, we conducted an experiment where we used our method (Algorithms 4 and 6) with $\xi = 1/2$ and Lourenço (2019) with modified von Neumann scheme to solve a weakly feasible instance with $\nu = 0.1$. At each iteration of the main algorithms, we recorded the value of $\frac{r^r}{\det w}$ of Lourenço (2019) and the value of ξ^N of our method and computed the reduction rates of the search region. The results are summarized in Table 13.

The "#iter of M-A" column shows the number of iterations of the main algorithm. The "Average reduction rate" column shows the average value of $\frac{r^r}{\det w}$ for Lourenço (2019) and the average value of ξ^N for our method (Algorithms 4 and 6). The "Final reduction rate" column shows the value

$$\frac{r^{kr}}{\det w(1) \times \det w(2) \times \cdots \times \det w(k)}$$

44

Table 13: Comparison of reduction rates of the two algorithms: Numerical results

Algorithm	#iter of M-A	Output	Average reduction rate	Final reduction rate
Lourenço (2019) : BP = MVN	3060	A	0.864	3.86e-195
Algorithms 4 and 6	618	C	0.357	9.11e-305

for Lourenço (2019), where $w(k)$ denotes w computed from the result of the basic procedure at the k-th iteration of the main algorithm, or the value

$$\xi^{N_1 + \cdots + N_k}.$$

for our method (Algorithms 4 and 6), where N_k denotes the number of cuts obtained from the basic procedure at the k-th iteration of the main algorithm.

Here, we observed that our method (Algorithms 4 and 6) terminated at the 618-th iteration of the main algorithm with a reduction rate of 9.11e-305, while Lourenço (2019) attained a reduction rate of 5.88e-40 at the same iteration of the main algorithm.

D.2   Detection of an ε-feasible solution

Here, we discuss the capabilities of our method and Lourenço (2019) at detecting an ε-feasible solution. Both methods terminate their main algorithms by detecting the existence of an ε-feasible solution. We compared them by computing the reduction in $\log(\lambda_{\min}(x_\ell))$ per iteration for parameter settings in which the maximum numbers of iterations of the basic procedures would be the same (i.e., $\rho = 2$ in Lourenço (2019) and $\xi = 1/2$ in our method).

In [10], for each block ℓ, Lemma 16 ensures that $\log(\lambda_{\min}(x_\ell))$ is bounded from above by ϵ_ℓ, and Theorem 17 ensures that ϵ_ℓ decreases at least $\frac{\varphi(\rho)}{r_\ell} > 0$ if a *good* iteration is obtained for the block ℓ. For our method, Proposition 5.1 ensures that $\log(\lambda_{\min}(x_\ell))$ is bounded from above by $\frac{\mathrm{num}_\ell}{r_\ell} \log \xi$ and Proposition 5.2 ensures that $\frac{\mathrm{num}_\ell}{r_\ell} \log \xi$ decreases $-\frac{1}{r_\ell} \log \xi > 0$ in the same situation. By substituting $\rho = 2$ and $\xi = 1/2$ into $\varphi(\rho)$ and $-\log \xi$ so that the upper bounds for the numbers of iterations of the basic procedures are the same, we obtain

$$\varphi(2) = 2 - \tfrac{1}{2} - \sqrt{2} \simeq 0.085786, \quad -\log \tfrac{1}{2} = \log 2 \simeq 0.693147$$

which implies that the rate of reduction in the upper bound $\log(\lambda_{\min}(x_\ell))$ of our method is greater than that of Lourenço (2019).