Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 14, No. 4, pp. 635–657, 2018

AN IMPROVED SMOOTHING ALGORITHM BASED ON A REGULARIZED CHKS SMOOTHING FUNCTION FOR THE P_0-NCP OVER SYMMETRIC CONES*

Jingyong Tang, Jinchuan Zhou and Liang Fang

Abstract: The smoothing algorithms have been successfully applied to solve the monotone symmetric cone complementarity problem (denoted by SCCP). In this paper, we first introduce a new smoothing function which is a regularized version of the well-known Chen-Harker-Kanzow-Smale (CHKS) smoothing function. Based on this smoothing function, we then propose a smoothing algorithm for solving the P_0-NCP over symmetric cones (denoted by P_0-SCNCP), which includes the monotone SCCP as a special case. Our algorithm reformulates the P_0-SCNCP as a system of smoothing equations based on a non-regulation function. Moreover, it adopts a new non-monotone line search scheme which contains the usual monotone line search rule and some existing non-monotone line search rules as special cases. Under weak assumptions, we prove that the proposed algorithm is globally and locally quadratically convergent. Preliminary numerical results indicate that the proposed algorithm is promising.

Key words: *symmetric cone complementarity problem, smoothing algorithm, Cartesian P_0-property, regulation function*

Mathematics Subject Classification: *90C25, 90C30, 65K05*

1 Introduction

There recently has been much interest in the symmetric cone complementarity problem (SCCP), which is to find a vector $(x, y) \in \mathcal{J} \times \mathcal{J}$ such that

$$x \in \mathcal{K}, \quad y \in \mathcal{K}, \quad y = F(x), \quad \langle x, y \rangle = 0, \tag{1.1}$$

where $\mathcal{J}$ is an n-dimensional real Euclidean space (see Sect. 2 for the definition), $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product, $F : \mathcal{J} \to \mathcal{J}$ is a continuously differentiable mapping, and $\mathcal{K} \subset \mathcal{J}$ is a symmetric cone, i.e., $\mathcal{K}$ is a self-dual closed convex cone with nonempty interior int$\mathcal{K}$, and for any $x, y \in$ int$\mathcal{K}$, there exists an invertible linear transformation $T : \mathcal{J} \to \mathcal{J}$ such that $T(\mathcal{K}) = \mathcal{K}$ and $T(x) = y$. The SCCP (1.1) has wide applications in many fields [1,3] and provides a unified framework for various existing complementarity problems such as the nonlinear complementarity problem (NCP), the second-order cone complementarity problem (SOCCP) and the semi-definite complementarity problem (SDCP).

*This work was supported by National Natural Science Foundation of China (11771255, 11801325) , Shandong Province Natural Science Foundation (ZR2016AM07) and Nanhu Scholars Program for Young Scholars of Xinyang Normal University.

In the last few years, various solution methods have been developed for solving the SCCP (e.g. [7–9,11–14,17,19,23,27,29]), in which the smoothing Newton algorithm is one kind of the most effective algorithms (e.g., [7–9,12–14,17,19,23]). This class of algorithms usually reformulates the SCCP as a system of smoothing equations and then solves the equations approximately by using Newton's method. One example of this kind of smoothing equations is

$$\begin{pmatrix} h(\mu) \\ F(x) - y \\ \phi(\mu, x, y) \end{pmatrix} = 0, \tag{1.2}$$

in which $h(\mu) = \mu$ or $h(\mu) = e^{\mu} - 1$, and $\phi : \mathcal{R} \times \mathcal{J} \times \mathcal{J} \to \mathcal{J}$ is a smoothing function for the SCCP. Recall that a function $\phi : \mathcal{R} \times \mathcal{J} \times \mathcal{J} \to \mathcal{J}$ is called a smoothing function if

(i) $\phi(0, \cdot, \cdot)$ is non-differentiable on $\mathcal{J} \times \mathcal{J}$;
(ii) ϕ is continuously differentiable at any $(\mu, x, y) \in \mathcal{R}_{++} \times \mathcal{J} \times \mathcal{J}$;
(iii) $\phi(0, x, y) = 0 \Longleftrightarrow x \in \mathcal{K},\ y \in \mathcal{K},\ \langle x, y\rangle = 0.$

It is worth pointing out that smoothing Newton algorithms in [7–9,12–14,17,19,23] are all designed for solving the SCCP in which the mapping F is monotone, that is,

$$\langle x - y, F(x) - F(y)\rangle \geq 0, \quad \forall (x, y) \in \mathcal{J} \times \mathcal{J}. \tag{1.3}$$

Lately, Lu and Huang [15] extended a smoothing Newton method to solve a non-monotone symmetric cone linear complementarity problem (SCLCP), the Cartesian P_0-SCLCP, and proved that the algorithm is globally and locally quadratically convergent.

In this paper, we consider the P_0-NCP over symmetric cones (P_0-SCNCP), which is to find a vector $(x, y) \in \mathcal{J} \times \mathcal{J}$ such that

$$x \in \mathcal{K}, \quad y \in \mathcal{K}, \quad y = F(x), \quad \langle x, y\rangle = 0, \tag{1.4}$$

where

$$\mathcal{J} = \mathcal{J}^{n_1} \times \cdots \times \mathcal{J}^{n_m}, \quad \mathcal{K} = \mathcal{K}^{n_1} \times \cdots \times \mathcal{K}^{n_m} \tag{1.5}$$

with $m, n_1, ..., n_m \geq 1$ and $n = \sum_{i=1}^{m} n_i$, in which $\mathcal{J}^{n_i}$ is a *simple* Euclidean Jordan algebra (see Sect. 2 for the definition), $\mathcal{K}^{n_i}$ is a symmetric cone, and $F : \mathcal{J} \to \mathcal{J}$ is a continuously differentiable mapping which has the Cartesian P_0-property. A mapping $F = (F_1, ..., F_m)$ with $F_i : \mathcal{R}^n \to \mathcal{R}^{n_i}$ is said to have the Cartesian P_0-property if for any $x = (x_1, ..., x_m) \in \mathcal{R}^n, y = (y_1, ..., y_m) \in \mathcal{R}^n$ and $x \neq y$, there is an index $v \in \{1, ..., m\}$ such that

$$x_v \neq y_v \quad \text{and} \quad \langle x_v - y_v, F_v(x) - F_v(y)\rangle \geq 0. \tag{1.6}$$

When $m = 1$, the Cartesian P_0-property of F becomes the monotonicity of F, and the P_0-SCNCP becomes the monotone SCCP. In addition, if $F(x) = Lx + q$, where $L : \mathcal{J} \to \mathcal{J}$ is a linear transformation and $q \in \mathcal{J}$, then the P_0-SCNCP becomes the P_0-SCLCP investigated by Lu and Huang [15].

Recently, Li and Wei [14] proved that smoothing Newton algorithms for solving the SCCP have global and local quadratical convergence if $h(\mu)$ in (1.2) is a *regulation function.*

Definition 1.1 ([14, Definition 3.1])**.** The function $h : \mathcal{R}_+ \to \mathcal{R}$ is called a regulation function if it satisfies the following conditions:

(a) $h(\mu)$ is continuously differentiable;
(b) $h(\mu) \geq 0$ and $h(\mu) = 0$ if and only if $\mu = 0$;
(c) $h(\mu) \to \infty$ as $\mu \to \infty$;
(d) $h(\mu) \leq \mu h'(\mu)$ for any $\mu > 0$;
(e) there exist real numbers $a > 0, b > 0$ such that $h'(\mu) \leq a h(\mu) + b$.

Obviously, μ and $e^{\mu}-1$ are regulation functions. More properties and examples of the regulation function can be found in [14]. A crucial question in this respect is whether we can find a non-regulation function which can give rise to an efficient smoothing algorithm for solving the P_0-SCNCP. Moreover, we also ask whether the corresponding algorithm has encouraging convergent properties and numerical results like existing smoothing algorithms designed by the regulation function.

Motivated by these questions, in this paper we first introduce a new smoothing function which is a regularized version of the well-known Chen-Harker-Kanzow-Smale (CHKS) smoothing function. Based on this function, we then design a smoothing algorithm for solving the P_0-SCNCP. Our algorithm solves the smoothing equation (1.2) with $h(\mu)=\ln(1+\mu)$. Since $(1+\mu)\ln(1+\mu)>\mu$ for any $\mu>0$ (see Lemma 4.3 below), we have $h(\mu)>\mu h'(\mu)$ for any $\mu>0$. This shows $h(\mu)=\ln(1+\mu)$ does not satisfy the condition (d) in Definition 1.1 and it is not a regulation function. Since $h(\mu)=\ln(1+\mu)$ is a non-regulation function, our algorithm uses different Newton equation and line search rule to obtain the search direction and step size. Under mild assumptions, we prove that our algorithm has global and local quadratical convergence properties. We also report some numerical results which demonstrate that our algorithm is very effective. In addition, like some non-monotone smoothing algorithms (e.g., [6,7,10,18,20,24–26,32]), our algorithm adopts a non-monotone line search scheme which contains the usual monotone line search rule used in [8,12–15,17,23] and the non-monotone line search rules studied in [10,24] as special cases.

The paper is organized as follows. In Sect. 2, we briefly give some basic results of Euclidean Jordan algebras. In Sect. 3, we introduce a new smoothing function and give its properties. In Sect. 4, we present a smoothing algorithm for solving the P_0-SCNCP. The global and local quadratic convergence of the algorithm are investigated in Sect. 5. Preliminary numerical results are reported in Sect. 6. Some conclusions are given in Sect. 7.

In our notations, $\|\cdot\|$ denotes the Euclidean norm. For any $x,y\in\mathcal{J}$, $x\succeq y$ (respectively, $x\succ y$) denotes $x-y\in\mathcal{K}$ (respectively, $x-y\in\mathrm{int}\mathcal{K}$). We write $x=o(\alpha)$(or $O(\alpha)$) if $\|x\|/|\alpha|$ tends to zero (or uniformly bounded) as $\alpha\to 0$. Let $E:\mathcal{J}\to\mathcal{J}$ be a mapping. If there exists a linear operator $DE(x)$ which satisfies

$$\lim_{\|h\|\to 0}\frac{\|E(x+h)-E(x)-DE(x)h\|}{\|h\|}\to 0,$$

then E is said to be Fréchet differentiable at x and $DE(x)$ is the Fréchet derivative of E at x.

$\boxed{2}$ Some Preliminaries

In this section, we briefly give some results of Euclidean Jordan algebras, which is a basic tool extensively used in this paper. Details on Euclidean Jordan algebras can be found in [2,22].

A Euclidean Jordan algebra (EJA) is a triple $(\mathcal{J},\langle\cdot,\cdot\rangle,\circ)$, where $(\mathcal{J},\langle\cdot,\cdot\rangle)$ is a real n-dimensional inner product space and $(x,y)\to x\circ y:\mathcal{J}\times\mathcal{J}\to\mathcal{J}$ is a bilinear mapping which satisfies the following conditions:

(i) $x\circ y=y\circ x$ for all $x,y\in\mathcal{J}$;

(ii) $x\circ(x^2\circ y)=x^2\circ(x\circ y)$ for all $x,y\in\mathcal{J}$, where $x^2:=x\circ x$;

(iii) $\langle x\circ y,z\rangle=\langle x,y\circ z\rangle$ for all $x,y,z\in\mathcal{J}$.

We call $x\circ y$ the *Jordan product* of x and y. If for some element $e\in\mathcal{J}$, $x\circ e=e\circ x=x$ for all

$x \in \mathcal{J}$, then e is called a *unit element.* The unit element, if it exists, is unique. A Euclidean Jordan algebra is called *simple* if it is not the direct sum of two Euclidean Jordan algebras. Every Euclidean Jordan algebra is, in a unique way, a direct sum of simple Euclidean Jordan algebras (see, [2, Proposition III.4.4]).

Given a Euclidean Jordan algebra $(\mathcal{J}, \langle\cdot,\cdot\rangle, \circ)$, we denote the set of squares as

$$\mathcal{K} := \{x^2 | x \in \mathcal{J}\}.$$

Then, by [2, Theorem III.2.1], we know that $\mathcal{K}$ is the *symmetric cone.*

For any $x \in \mathcal{J}$, we define $\mathrm{m}(x) := \min\{k : \{e, x, x^2, ..., x^k\}$ are linearly dependent$\}$. Then $\mathrm{m}(x)$ is said to be the *degree* of x. The *rank* of $(\mathcal{J}, \langle\cdot,\cdot\rangle, \circ)$ is defined as $r := \max\{\mathrm{m}(x) : x \in \mathcal{J}\}$. An element $c \in \mathcal{J}$ is said to be *idempotent* if $c^2 = c$. A *complete system of orthogonal idempotents* is a finite set $\{c_1, ..., c_r\}$ where

$$c_j^2 = c_j, \quad c_i \circ c_j = 0, \quad \forall i \neq j,\ i, j = 1, ..., r, \quad \text{and} \quad \sum_{i=1}^{r} c_i = e.$$

An idempotent is said to be *primitive* if it is nonzero and cannot be written as the sum of two other nonzero idempotents. We call a complete system of orthogonal primitive idempotents a *Jordan frame.* Then, we have the following important *spectral decomposition* theorem.

Theorem 2.1 ([2, Theorem III.1.2])**.** *Let $(\mathcal{J}, \langle\cdot,\cdot\rangle, \circ)$ be a Euclidean Jordan algebra with rank r. Then, for any $x \in \mathcal{J}$, there exist a Jordan frame $\{c_1, ..., c_r\}$ and real numbers $\lambda_1(x), ..., \lambda_r(x)$ such that*

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

The numbers $\lambda_i(x) (i = 1, ..., r)$ (counting multiplicities), which are uniquely determined by x, are called the eigenvalues, and $\sum_{i=1}^{r} \lambda_i(x) c_i$ the spectral decomposition of x.

Let $x \in \mathcal{J}$ and $\lambda_1(x), ..., \lambda_r(x)$ be its eigenvalues. Define

$$\mathrm{Tr}(x) := \sum_{i=1}^{r} \lambda_i(x), \quad \mathrm{Det}(x) := \prod_{i=1}^{r} \lambda_i(x),$$

where $\mathrm{Tr}(x)$ is the trace of x and $\mathrm{Det}(x)$ is the determinant of x. For the identity element e, $\mathrm{Tr}(e) = r$ and $\mathrm{Det}(x) = 1$.

Suppose that $x \in \mathcal{J}$ has the spectral decomposition $x = \sum_{i=1}^{r} \lambda_i(x) c_i$. Let $f : \mathcal{R} \to \mathcal{R}$ be a real-valued function. It is natural to define a vector valued function associated with the Euclidean Jordan algebra $(\mathcal{J}, \langle\cdot,\cdot\rangle, \circ)$ by

$$f(x) := \sum_{i=1}^{r} f(\lambda_i(x)) c_i.$$

The function f is also called a *Löwner operator* and shown to inherit many properties from f. In particular, by letting $t_+ := \max\{0, t\}$, $t_- := \min\{0, t\}$ and noting $|t| = t_+ - t_- (t \in \mathcal{R})$, respectively, we define

$$x_+ := \sum_{i=1}^{r} \lambda_i(x)_+ c_i, \ x_- := \sum_{i=1}^{r} \lambda_i(x)_- c_i, \text{ and } |x| := \sum_{i=1}^{r} |\lambda_i(x)| c_i.$$

It is easy to verify that $x_+ \in \mathcal{K}, x = x_+ + x_-$ and $|x| = x_+ - x_-$. Since $x \in \mathcal{K}$ if and only if $\lambda_i(x) \geq 0, i = 1, ..., r$, by letting $f(t) := \sqrt{t}$ for $t \in \mathcal{R}_+$, we define

$$\sqrt{x} := \sum_{i=1}^{r} \sqrt{\lambda_i(x)} c_i, \ \forall\, x \in \mathcal{K}.$$

For a given $x \in \mathcal{J}$, we define a linear operator $L_x : \mathcal{J} \to \mathcal{J}$ by

$$L_x y := x \circ y, \quad \forall\, y \in \mathcal{J}.$$

We call L_x the corresponding *Lyapunov transformation* of x. From the product structure of $\mathcal{J}$ in (1.5) and the linearity of Lyapunov transformations, for any $x \in \mathcal{J}$, we can immediately give the following matrix representation of L_x,

$$L_x := \begin{pmatrix} L_{x_1} & & \\ & \ddots & \\ & & L_{x_m} \end{pmatrix}.$$

Moreover, for any $x = (x_1, ..., x_m) \in \mathcal{J}, y = (y_1, ..., y_m) \in \mathcal{J}$, it follows that

$$x \circ y = (x_1 \circ y_1, ..., x_m \circ y_m), \quad \text{and} \quad \langle x, y \rangle = \sum_{i=1}^{m} \langle x_i, y_i \rangle.$$

The following lemma gives some properties of Lyapunov transformations, whose proof can be found in [16, Lemma 2.3].

Lemma 2.2. *For any $x, y \in \mathcal{J}$, let L_x and L_y be the corresponding Lyapunov transformations. Then the following results hold.*
(i) *L_{x_i} is self-adjoint for any $i \in \{1, ..., m\}$, i.e., $\langle L_{x_i} y_i, z_i \rangle = \langle y_i, L_{x_i} z_i \rangle$ for all $y_i, z_i \in \mathcal{J}^{n_m}$.*
(ii) *$L_x = \sum_{i=1}^{r} \lambda_i(x) L_{c_i}$, where $x = \sum_{i=1}^{r} \lambda_i(x) c_i$ is the spectral decomposition of x;*
(iii) *if the inverse operator L_x^{-1} exists, then $L_x^{-1} := \mathrm{diag}(L_{x_1}^{-1}, ..., L_{x_m}^{-1})$;*
(iv) *for any $\alpha, \beta \in \mathcal{R}$, it holds that $\alpha L_x + \beta L_y := \mathrm{diag}(\alpha L_{x_1} + \beta L_{y_1}, ..., \alpha L_{x_m} + \beta L_{y_m})$.*

At the end of this section, we give the definition of the Cartesian P_0-property for a matrix $M \in \mathcal{R}^{n \times n}$.

Definition 2.3. A matrix $M \in \mathcal{R}^{n \times n}$ is said to have the Cartesian P_0-property if for any $0 \neq x = (x_1, ..., x_m) \in \mathcal{R}^n$ with $x_i \in \mathcal{R}^{n_i}$, there exists an index $v \in \{1, ..., m\}$ such that $x_v \neq 0$ and $\langle x_v, (Mx)_v \rangle \geq 0$.

As is well-known, if a continuously differentiable mapping $F : \mathcal{R}^n \to \mathcal{R}^n$ has the Cartesian P_0-property, then its Jacobian matrix $F'(x)$ at any $x \in \mathcal{R}^n$ enjoys the Cartesian P_0-property.

3 A New Smoothing Function

As is well-known, smoothing functions play important roles in designing smoothing algorithms. Up to now, many smoothing functions have been proposed. Among them, the Chen-Harker-Kanzow-Smale (CHKS) smoothing function is one of the most prominent smoothing functions, which is defined by

$$\varphi(\mu, a, b) = a + b - \sqrt{(a-b)^2 + 4\mu^2 e}, \quad \forall (\mu, a, b) \in \mathcal{R}_+ \times \mathcal{J}^n \times \mathcal{J}^n.$$

Recently, Huang and Ni [8] proposed a regularized version of the CHKS smoothing function defined by

$$\psi_{\rm HN}(\mu, a, b) = (1+\mu)(a+b) - \sqrt{(1-\mu)^2(a-b)^2 + 4\mu^2 e}, \quad \forall(\mu, a, b) \in \mathcal{R}_+ \times \mathcal{J}^n \times \mathcal{J}^n.$$

Based on ψ, Huang and Ni [8] extended two generic frameworks of smoothing algorithms to solve the SCCP, and Lu and Huang [15] extended a smoothing Newton algorithm to solve the Cartesian P_0-SCLCP.

In this paper, we introduce a new smoothing function as follows:

$$\phi(\mu, a, b) = a + b - \sqrt{(1-2\mu)^2(a-b)^2 + 4\mu^2 e}, \quad \forall(\mu, a, b) \in \mathcal{R}_+ \times \mathcal{J}^n \times \mathcal{J}^n. \tag{3.1}$$

As been shown later, this new function has following favorable properties analogous to what the regularized CHKS function $\psi_{\rm HN}$ has.

(i) ϕ is continuously differentiable on $\mathcal{R}_{++} \times \mathcal{J}^n \times \mathcal{J}^n$ and satisfies

$$\phi(0, a, b) = 0 \Longleftrightarrow a \succeq 0, b \succeq 0, \langle a, b\rangle = 0.$$

Thus, we can use ϕ to reformulate the SCCP or P_0-SCNCP as a family of parameterized smooth equations and then solve the smooth equations approximately by using Newton's method.

(ii) ϕ is coercive under suitable assumptions. This property insures the merit function has coerciveness, which plays an important rule in proving the global convergence of smoothing algorithms.

(iii) ϕ is strongly semi-smooth, which is key to prove the local quadratic convergence of smoothing algorithms.

First, we show that the function ϕ is continuously differentiable on $\mathcal{R}_{++} \times \mathcal{J}^n \times \mathcal{J}^n$.

Lemma 3.1. *Let $a, b, u, v \in \mathcal{J}^n, \mu > 0, \theta \in \mathcal{R}$ and $\phi(\mu, a, b)$ be defined by* (3.1).
(i) *Suppose that the spectral decomposition of $(a-b)^2$ is given by*

$$(a-b)^2 = \sum_{i=1}^{r} \lambda_i c_i, \tag{3.2}$$

where $\{c_1, ..., c_r\}$ is a Jordan frame and the numbers $\lambda_1, ..., \lambda_r$ (with multiplicities) are uniquely determined by $(a-b)^2$. Then

$$D_\mu \phi(\mu, a, b)\theta = \sum_{i=1}^{r} \frac{[(2-4\mu)\lambda_i - 4\mu]\theta}{\sqrt{(1-2\mu)^2\lambda_i + 4\mu^2}} c_i, \tag{3.3}$$

and $D_\mu\phi(\cdot,\cdot,\cdot)$ is continuous on $\mathcal{R}_{++} \times \mathcal{J}^n \times \mathcal{J}^n$.
(ii) *Denote $c := \sqrt{(1-2\mu)^2(a-b)^2 + 4\mu^2 e}$. Then*

$$D_a\phi(\mu, a, b)u = u - L_c^{-1}[(1-2\mu)^2(a-b) \circ u], \tag{3.4}$$

$$D_b\phi(\mu, a, b)v = v - L_c^{-1}[(1-2\mu)^2(a-b) \circ (-v)], \tag{3.5}$$

and $D_a\phi(\cdot,\cdot,\cdot)$ and $D_b\phi(\cdot,\cdot,\cdot)$ are continuous on $\mathcal{R}_{++} \times \mathcal{J}^n \times \mathcal{J}^n$.

Proof. By the definition of ϕ, we have

$$\phi(\mu + t\theta, a, b) - \phi(\mu, a, b)$$

$$= \sqrt{(1-2\mu)^2(a-b)^2+4\mu^2 e} - \sqrt{(1-2(\mu+t\theta))^2(a-b)^2+4(\mu+t\theta)^2 e}.$$

From (3.2) and the fact that $\sum_{i=1}^r c_i = e$, we have

$$(1-2\mu)^2(a-b)^2+4\mu^2 e = \sum_{i=1}^r [(1-2\mu)^2\lambda_i + 4\mu^2]c_i, \tag{3.6}$$

$$(1-2(\mu+t\theta))^2(a-b)^2+4(\mu+t\theta)^2 e = \sum_{i=1}^r [(1-2(\mu+t\theta))^2\lambda_i + 4(\mu+t\theta)^2]c_i. \tag{3.7}$$

Since $(a-b)^2 \succeq 0$, we have $\lambda_i \geq 0$ for all $i=1,...,r$. Also notice that $c_i^2 = c_i$ for all $i = 1,...,r$ since c_i is idempotent. Using these results, we can obtain from (3.6) and (3.7) that

$$\sqrt{(1-2\mu)^2(a-b)^2+4\mu^2 e} = \sum_{i=1}^r \sqrt{(1-2\mu)^2\lambda_i+4\mu^2}c_i,$$

$$\sqrt{(1-2(\mu+t\theta))^2(a-b)^2+4(\mu+t\theta)^2 e} = \sum_{i=1}^r \sqrt{(1-2(\mu+t\theta))^2\lambda_i+4(\mu+t\theta)^2}c_i.$$

Therefore,

$$\begin{aligned}
&\lim_{t\to 0}\frac{\phi(\mu+t\theta,a,b)-\phi(\mu,a,b)}{t}\\
&=\lim_{t\to 0}\frac{\sqrt{(1-2\mu)^2(a-b)^2+4\mu^2 e} - \sqrt{(1-2(\mu+t\theta))^2(a-b)^2+4(\mu+t\theta)^2 e}}{t}\\
&=\sum_{i=1}^r \lim_{t\to 0}\frac{\sqrt{(1-2\mu)^2\lambda_i+4\mu^2} - \sqrt{(1-2(\mu+t\theta))^2\lambda_i+4(\mu+t\theta)^2}}{t}c_i\\
&=\sum_{i=1}^r \lim_{t\to 0}\frac{[(1-2\mu)^2\lambda_i+4\mu^2]-[(1-2(\mu+t\theta))^2\lambda_i+4(\mu+t\theta)^2]}{t[\sqrt{(1-2\mu)^2\lambda_i+4\mu^2} + \sqrt{(1-2(\mu+t\theta))^2\lambda_i+4(\mu+t\theta)^2}]}c_i\\
&=\sum_{i=1}^r \lim_{t\to 0}\frac{[4(1-2\mu)\theta-4t\theta^2]\lambda_i - 4(2\mu\theta+t\theta^2)]}{[\sqrt{(1-2\mu)^2\lambda_i+4\mu^2} + \sqrt{(1-2(\mu+t\theta))^2\lambda_i+4(\mu+t\theta)^2}]}c_i\\
&=\sum_{i=1}^r \frac{[(2-4\mu)\lambda_i-4\mu]\theta}{\sqrt{(1-2\mu)^2\lambda_i+4\mu^2}}c_i,
\end{aligned}$$

which indicates that $\phi(\mu,a,b)$ is differentiable in μ and

$$D_\mu\phi(\mu,a,b)\theta = \sum_{i=1}^r \frac{[(2-4\mu)\lambda_i-4\mu]\theta}{\sqrt{(1-2\mu)^2\lambda_i+4\mu^2}}c_i.$$

Moreover, from the above equality, it is easy to see that $D_\mu\phi(\cdot,\cdot,\cdot)$ is continuous on $\mathcal{R}_{++}\times\mathcal{J}^n\times\mathcal{J}^n$. This completes the proof of the result (i). Now, we prove the result (ii). Since $c \succ 0$, L_c has inverse operator L_c^{-1}. Let

$$d := \sqrt{(1-2\mu)^2(a+u-b)^2+4\mu^2 e},$$

then we have

$$d^2 - c^2 = 2(1-2\mu)^2(a-b)\circ u + (1-2\mu)^2 u^2.$$

Also notice that $d^2 - c^2 = 2(d-c)\circ c + (d-c)^2$. Hence, we have

$$2(d-c) + L_c^{-1}((d-c)^2) = L_c^{-1}(d^2 - c^2) = L_c^{-1}(2(1-2\mu)^2(a-b)\circ u) + L_c^{-1}((1-2\mu)^2u^2).$$

Since L_c^{-1} is continuous, the above equality implies that $d - c = O(\|u\|)$. Therefore,

$$d - c = L_c^{-1}((1-2\mu)^2(a-b)\circ u) + O(\|u\|^2).$$

So that we have

$$\begin{aligned}\phi(\mu, a+u, b) - \phi(\mu, a, b) &= a + u + b - (a+b) - (d-c)\\ &= u - L_c^{-1}((1-2\mu)^2(a-b)\circ u) + O(\|u\|^2),\end{aligned}$$

and hence $D_a\phi(\mu,a,b)u = u - L_c^{-1}((1-2\mu)^2(a-b)\circ u)$. Since $c \succ 0$ and $L_c^{-1}(w)$ is continuous at any $(w,c) \in \mathcal{J}^n \times \mathrm{int}\mathcal{K}$, we further obtain that $D_a\phi(\mu,a,b)$ is continuous on $\mathcal{R}_{++}\times\mathcal{J}^n\times\mathcal{J}^n$. By a similar way, we can prove (3.5) and $D_b\phi(\mu,a,b)$ is continuous on $\mathcal{R}_{++}\times\mathcal{J}^n\times\mathcal{J}^n$. The proof is completed. □

The following lemma establishes the relations between the P_0-SCNCP and the smoothing function ϕ given in (3.1).

Lemma 3.2. *Let ϕ be defined by* (3.1). *Then*
(i) $a \succeq 0, b \succeq 0$ *and* $a\circ b = 0$ *if and only if* $a \succeq 0, b\succeq 0$ *and* $\langle a, b\rangle = 0$.
(ii) $\phi(0,a,b) = 0$ *if and only if* $a\succeq 0, b\succeq 0, a\circ b = 0$.
(iii) $\phi(\mu,a,b) = 0$ *if and only if* $(1-\mu)a + \mu b \succ 0,\ \mu a + (1-\mu)b \succ 0,$ *and* $[(1-\mu)a+\mu b]\circ[\mu a + (1-\mu)b] = \mu^2 e$.

Proof. The results (i) and (ii) hold from [8, Lemma 3.2 (i) and (ii)]. In addition, from [8, Lemma 3.2 (iii)], we get that for any $(\mu, s, t) \in \mathcal{R}\times\mathcal{J}^n\times\mathcal{J}^n$

$$s + t - \sqrt{(s-t)^2 + 4\mu^2 e} = 0 \Longleftrightarrow s \succ 0, \quad t \succ 0 \ \text{ and } s\circ t = \mu^2 e.$$

By noticing that the definition of ϕ can be rewritten as

$$\begin{aligned}\phi(\mu,a,b) &= [(1-\mu)a + \mu b] + [\mu a + (1-\mu)b]\\ &\quad -\sqrt{[((1-\mu)a+\mu b) - (\mu a + (1-\mu)b)]^2 + 4\mu^2 e},\end{aligned}$$

we can immediately obtain the result (iii). □

Now we give the coerciveness and strong semi-smoothness of the function ϕ, respectively.

Lemma 3.3. *Let ϕ be defined by* (3.1), *and $\xi, \zeta \in \mathcal{R}_{++}$ with $\xi < \zeta$. Suppose that $\{(\mu_k, a^k, b^k)\} \subset \mathcal{R}_{++}\times\mathcal{J}^n\times\mathcal{J}^n$ is a sequence satisfying*

(c1) $\mu_k \in [\xi,\zeta], \{(a^k,b^k)\}$ *is unbounded*; *and*

(c2) *there exists a bounded sequence $\{(u^k,v^k)\}$ such that $\{\langle a^k - u^k, b^k - v^k\rangle\}$ is bounded below.*

Then, $\|\phi(\mu_k,a^k,b^k)\| \to \infty$ as $k\to\infty$.

Proof. Notice that the definition of ϕ can be rewritten as

$$\begin{aligned}\phi(\mu,a,b) &= [(1-\mu)a + \mu b] + [\mu a + (1-\mu)b]\\ &\quad -\sqrt{[((1-\mu)a+\mu b) - (\mu a + (1-\mu)b)]^2 + 4\mu^2 e}.\end{aligned}$$

Using this fact, we can prove the result similarly as [8, Theorem 4.1]. We omit it here. □

Lemma 3.4. *The function* ϕ *in* (3.1) *is strongly semi-smooth at any* $(0, a, b) \in \mathcal{R} \times \mathcal{J}^n \times \mathcal{J}^n$.

Proof. By [22, Proposition 3.4], we know that the function $f(\mu, x) := \sqrt{x^2 + \mu^2 e}$ is strongly semi-smooth at any $(0, x) \in \mathcal{R} \times \mathcal{J}^n$. By the definition of ϕ and the fact that the composition of strongly semi-smooth functions is strongly semi-smooth, we can obtain the result. $\square$

4 The Algorithm

By letting $z := (\mu, x, y) \in \mathcal{R}_+ \times \mathcal{J} \times \mathcal{J}$, we define the function $H : \mathcal{R}_+ \times \mathcal{J} \times \mathcal{J} \to \mathcal{R}_+ \times \mathcal{J} \times \mathcal{J}$ as

$$H(z) := \begin{pmatrix} \ln(1+\mu) \\ \Phi(z) \end{pmatrix} \quad \text{with} \quad \Phi(z) := \begin{pmatrix} F(x) - y \\ \phi(\mu, x_1, y_1) \\ \vdots \\ \phi(\mu, x_m, y_m) \end{pmatrix}, \tag{4.1}$$

where $\phi(\cdot, \cdot, \cdot)$ is defined by (3.1). Then it is easy to see that

$$H(z) = 0 \Longleftrightarrow \mu = 0 \text{ and } (x, y) \text{ is the solution of the } P_0\text{-SCNCP}.$$

Moreover, we denote the merit function $\Psi : \mathcal{R}_+ \times \mathcal{J} \times \mathcal{J} \to \mathcal{R}_+$ by

$$\Psi(z) := \|H(z)\|^2. \tag{4.2}$$

Now we give a formal description of our algorithm.

Algorithm 4.1 (A smoothing algorithm for the P_0-SCNCP)**.** *Step 0:* Choose constants $\sigma \in (0, 1/2)$, $\delta \in (0, 1)$ and $0 < \mu_0 < 1$. Choose constants $\theta \in (0, 1]$ and $\tau \in (0, 1]$ such that and $\tau \le \theta$. Choose a constant $\gamma \in (0, 1)$ such that $\gamma \le \mu_0$ and $\mu_0\gamma < 1/2$. Choose a sufficiently small number $c > 0$. Take $h := (\mu_0, 0, 0) \in \mathcal{R} \times \mathcal{J} \times \mathcal{J}$. Let $(x^0, y^0) \in \mathcal{J} \times \mathcal{J}$ be an arbitrary initial point. Let $z^0 := (\mu_0, x^0, y^0)$ and $C_0 := \Psi(z^0)$. Choose a constant $\epsilon_0 \ge 0$. Set $k := 0$.

Step 1: If $\|H(z^k)\| = 0$, then stop. Else, compute

$$\beta_k := \begin{cases} \gamma \min\{1, \Psi(z^k)\}, & \text{if } k = 0, \\ \gamma \min\{1, \Psi(z^k), \beta_{k-1}\}, & \text{if } k \ge 1. \end{cases} \tag{4.3}$$

Step 2: Compute $\Delta z^k := (\Delta\mu_k, \Delta x^k, \Delta y^k) \in \mathcal{R} \times \mathcal{J} \times \mathcal{J}$ by

$$DH(z^k)\Delta z^k = -H(z^k) + \frac{2\beta_k}{1+\mu_k} h, \tag{4.4}$$

where $DH(z^k)$ denotes the Jacobian of H at z^k.

Step 3: Let α_k be the maximum of the values $1, \delta, \delta^2, \ldots$ such that

$$\Psi(z^k + \alpha_k \Delta z^k) \le \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\alpha_k\right](C_k + \epsilon_k) \tag{4.5}$$

and

$$(1 + \alpha_k)\mu_k < 1. \tag{4.6}$$

Step 4: Set $z^{k+1} := z^k + \alpha_k \Delta z^k$. Set $k := k + 1$.

Step 5: If $\Psi(z^k) < c$, then set

$$C_k := \Psi(z^k), \quad \epsilon_k := 0. \tag{4.7}$$

Otherwise, set

$$C_k := (1-\theta)C_{k-1} + \theta\Psi(z^k), \quad \epsilon_k = (1-\tau)\epsilon_{k-1}. \tag{4.8}$$

Go to Step 1.

Remark 4.2. **(i)** Based on a regulation function $h(\mu)$, existing smoothing algorithms (e.g., [8,12–15,17,18,23]) obtain the search direction Δz^k by solving the following Newton equation

$$DH(z^k)\Delta z^k = -H(z^k) + h'(\mu_k)\beta_k h.$$

Since the function $h(\mu) = \ln(1+\mu)$ is not a regulation function, to ensure Algorithm 4.1 be well-defined, we require the search direction Δz^k satisfy

$$DH(z^k)\Delta z^k = -H(z^k) + 2h'(\mu_k)\beta_k h.$$

(ii) Zhang and Hager [30] proposed a non-monotone line search scheme for the unconstrained optimization problem. They let $Q_0 := 1$ and choose $\eta_{\min}$ and $\eta_{\max}$ such that $0 \leq \eta_{\min} < \eta_{\max} < 1$, and choose $\eta_k \in [\eta_{\min}, \eta_{\max}]$, and then set

$$Q_{k+1} := \eta_k Q_k + 1,$$

$$C_{k+1} := \frac{\eta_k Q_k C_k + \Psi(z^{k+1})}{Q_{k+1}}.$$

Many smoothing algorithms based on the Zhang and Hager's non-monotone line search scheme [30] have been studied for solving various optimization problems, such as the SCCP [7], the NCP [18,32], the support vector machine [20], the system of equalities and inequalities [31], and so on. Other smoothing algorithms based on different non-monotone line search scheme have also been proposed by many authors (e.g., [6,10,24,25]). Motivated by the ideas in these papers, our algorithm adopts a new non-monotone line search in Step 3. Notice that

• if we choose $\theta = 1$, $\tau = 1$ and $\epsilon_0 = 0$, then by (4.8) we have $C_k = \Psi(z^k)$ and $\epsilon_k = 0$ for all $k \geq 0$ and hence (4.5) becomes

$$\Psi(z^k + \alpha_k \Delta z^k) \leq \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\alpha_k\right]\Psi(z^k).$$

In this case, Step 3 of Algorithm 4.1 is the usual monotone line search which has been used extensively in smoothing algorithms;

• if we choose $0 < \theta < 1$, $\tau = 1$ and $\epsilon_0 = 0$, then $C_k = (1-\theta)C_{k-1} + \theta\Psi(z^k)$ and $\epsilon_k = 0$ and hence (4.5) becomes

$$\Psi(z^k + \alpha_k \Delta z^k) \leq \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\alpha_k\right]C_k.$$

In this case, Step 3 of Algorithm 4.1 is a convex combination-type non-monotone line search which has been studied in [24,26];

• if we choose $\theta = 1$, $0 < \tau < 1$ and $\epsilon_0 > 0$, then $C_k = \Psi(z^k)$ and $\epsilon_k = (1-\tau)\epsilon_{k-1}$ and hence (4.5) becomes

$$\Psi(z^k + \alpha_k \Delta z^k) \leq \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\alpha_k\right][\Psi(z^k) + \epsilon_k].$$

In this case, Step 3 of Algorithm 4.1 is the standard non-monotone line search (e.g., [10]).
(iii) Since smoothing algorithms using the monotone line search possess locally fast convergence, following the idea in [6], we let the algorithm perform the monotone line search in Step 3 when $\Psi(z^k) < c$ with a sufficiently small positive number c.
(iv) In Step 3, we need α_k not only satisfy the Armijo line search (4.5) but also satisfy the condition (4.6). As be seen later, the condition (4.6) can insure that the sequence $\{\mu_k\}$ generated by Algorithm 4.1 satisfies $0 < \mu_k < 1$ for all $k \geq 0$. Notice that if $0 < \mu_k < 1$, then (4.6) holds when α_k is sufficiently small.

Lemma 4.3. *For any* $0 < \mu < 1$*, one has*

$$\mu < (1+\mu)\ln(1+\mu) < 2\mu. \tag{4.9}$$

Proof. On one hand, let $\psi(\mu) := (1+\mu)\ln(1+\mu) - \mu$. Then $\psi'(\mu) = \ln(1+\mu) > 0$ for any $\mu > 0$. This shows that $\psi(\mu)$ is monotonically increasing and hence $\psi(\mu) > \psi(0) = 0$, i.e., $(1+\mu)\ln(1+\mu) > \mu$ for any $\mu > 0$. On the other hand, by defining $\varphi(\mu) := \mu - \ln(1+\mu)$, we have $\varphi'(\mu) = 1 - \frac{1}{1+\mu} > 0$ for any $\mu > 0$ and hence $\varphi(\mu) > \varphi(0) = 0$, i.e., $\mu > \ln(1+\mu)$ for any $\mu > 0$. Also notice that $2 > 1+\mu$ for any $0 < \mu < 1$. So, $2\mu > (1+\mu)\ln(1+\mu)$ holds for any $0 < \mu < 1$. We have the desired result. □

Lemma 4.4. *Let* $H(z)$ *be defined by* (4.1)*. If* F *has the Cartesian* P_0*-property, then the Jacobian of* H *in Step* 2 *of Algorithm* 4.1 *is invertible for any* $0 < \mu < 1$.

Proof. For any $0 < \mu < 1$, let $\tilde{z} := (\tilde{\mu}, \tilde{x}, \tilde{y}) \in \mathcal{R} \times \mathcal{J} \times \mathcal{J}$ be a vector in the null space of DH, where $\tilde{x} = (\tilde{x}_1, ..., \tilde{x}_m), y = (\tilde{y}_1, ..., \tilde{y}_m)$ with $\tilde{x}_i, \tilde{y}_i \in \mathcal{J}^{n_i}$, it suffices to show that $\tilde{z} = 0$. By the definition of H in (4.1), we can obtain from $DH(z)\tilde{z} = 0$ that

$$\tilde{\mu} = 0, \tag{4.10}$$

$$DF(x)\tilde{x} - \tilde{y} = 0, \tag{4.11}$$

$$D\phi(\mu, x_i, y_i)(\tilde{\mu}, \tilde{x}_i, \tilde{y}_i) = 0, \quad i = 1, ..., m. \tag{4.12}$$

Now we assume that $\tilde{x} \neq 0$. Since F has the Cartesian P_0-property, DF has the Cartesian P_0-property. Hence, there exists an index $v \in \{1, ..., m\}$ such that

$$\tilde{x}_v \neq 0, \quad \langle \tilde{x}_v, (DF(x)\tilde{x})_v \rangle \geq 0. \tag{4.13}$$

Since $\tilde{\mu} = 0$, it follows from Lemma 3.1 that

$$D\phi(\mu, x_v, y_v)(\tilde{\mu}, \tilde{x}_v, \tilde{y}_v) = \tilde{x}_v + \tilde{y}_v - L_{c_v}^{-1}[(1-2\mu)^2(x_v - y_v) \circ (\tilde{x}_v - \tilde{y}_v)],$$

where $c_v := \sqrt{(1-2\mu)^2(x_v - y_v)^2 + 4\mu^2 e_v}$. Then (4.12) becomes $\tilde{x}_v + \tilde{y}_v - L_{c_v}^{-1}[(1-2\mu)^2(x_v - y_v) \circ (\tilde{x}_v - \tilde{y}_v)] = 0$, i.e.,

$$L_{c_v}(\tilde{x}_v + \tilde{y}_v) - [(1-2\mu)^2(x_v - y_v) \circ (\tilde{x}_v - \tilde{y}_v)] = 0. \tag{4.14}$$

Since $\tilde{x}_v + \tilde{y}_v = [(1-\mu)\tilde{x}_v + \mu\tilde{y}_v] + [\mu\tilde{x}_v + (1-\mu)\tilde{y}_v]$, by (4.14) we have

$$[c_v - (1-2\mu)(x_v - y_v)] \circ [(1-\mu)\tilde{x}_v + \mu\tilde{y}_v] + [c_v + (1-2\mu)(x_v - y_v)] \circ [\mu\tilde{x}_v + (1-\mu)\tilde{y}_v] = 0. \tag{4.15}$$

From (4.11) and (4.13), we can obtain that $\langle \tilde{x}_v, \tilde{y}_v \rangle \geq 0$, which implies that for any $0 < \mu < 1$

$$\langle (1-\mu)\tilde{x}_v + \mu\tilde{y}_v, \mu\tilde{x}_v + (1-\mu)\tilde{y}_v \rangle = (1-\mu)\mu(\|\tilde{x}_v\|^2 + \|\tilde{y}_v\|^2) + ((1-\mu)^2 + \mu^2)\langle \tilde{x}_v, \tilde{y}_v \rangle \geq 0. \tag{4.16}$$

Since $c_v^2 - (1-2\mu)^2(x_v - y_v)^2 = 4\mu^2 e_v \succ 0$, it follows from [4, Proposition 8] that

$$c_v - (1-2\mu)(x_v - y_v) \succ 0 \quad \text{and} \quad c + (1-2\mu)(x_v - y_v) \succ 0. \tag{4.17}$$

In addition,

$$[c_v-(1-2\mu)(x_v-y_v)]\circ[c_v+(1-2\mu)(x_v-y_v)] = c_v^2-(1-2\mu)^2(x_v-y_v)^2 = 4\mu^2 e \succ 0 \tag{4.18}$$

Thus, using (4.15)–(4.18), it follows from [28, Lemma 2.7 (vi)] that $(1-\mu)\tilde{x}_v + \mu\tilde{y}_v = 0$ and $\mu\tilde{x}_v + (1-\mu)\tilde{y}_v = 0$, which, together with $\langle \tilde{x}_v, \tilde{y}_v\rangle \geq 0$ and $0 < \mu < 1$, yields that $\tilde{x}_v = \tilde{y}_v = 0$. This contradicts with $\tilde{x}_v \neq 0$ in (4.13). So, $\tilde{x} = 0$. Furthermore, by (4.11) we have $\tilde{y} = 0$. Thus, $DH(z)$ is invertible. This completes the proof. □

Lemma 4.5. *Suppose that F has the Cartesian P_0-property. If $0 < \mu_k < 1$ and $\mu_k \geq \mu_0\beta_k$, then $z^{k+1} = (\mu_{k+1}, x^{k+1}, y^{k+1})$ can be generated by Algorithm 4.1 with $0 < \mu_{k+1} < 1$ and $\mu_{k+1} \geq \mu_0\beta_{k+1}$.*

Proof. Since $0 < \mu_k < 1$, it follows from Lemma 4.4 that $DH(z^k)$ is nonsingular. Hence, the direction $\Delta z^k = (\Delta\mu_k, \Delta x^k, \Delta y^k)$ can be obtained by Step 2. From the first equation in (4.4) we have

$$\frac{\Delta\mu_k}{1+\mu_k} = -\ln(1+\mu_k) + \frac{2\mu_0\beta_k}{1+\mu_k},$$

which gives

$$\Delta\mu_k = -(1+\mu_k)\ln(1+\mu_k) + 2\mu_0\beta_k.$$

For any $\alpha \in (0, 1/2)$, it follows that

$$\mu_k + \alpha\Delta\mu_k = \mu_k - \alpha(1+\mu_k)\ln(1+\mu_k) + 2\alpha\mu_0\beta_k. \tag{4.19}$$

Since $0 < \mu_k < 1$, by using the right inequality in (4.9), we have $(1+\mu_k)\ln(1+\mu_k) < 2\mu_k$. So, for any $\alpha \in (0, 1/2)$, we can obtain from (4.19) that

$$\mu_k + \alpha\Delta\mu_k \geq (1-2\alpha)\mu_k + 2\alpha\mu_0\beta_k > 0. \tag{4.20}$$

For any $\alpha \in (0, 1/2)$, we denote

$$w_k(\alpha) := \Psi(z^k + \alpha\Delta z^k) - \Psi(z^k) - \alpha D\Psi(z^k)\Delta z^k,$$

then, by using (4.4), for any $\alpha \in (0, 1/2)$,

$$\begin{aligned}
\Psi(z^k + \alpha\Delta z^k) &= \Psi(z^k) + \alpha D\Psi(z^k)\Delta z^k + w_k(\alpha) \\
&= \Psi(z^k) + 2\alpha H(z^k)^T DH(z^k)\Delta z^k + w_k(\alpha) \\
&= \Psi(z^k) + 2\alpha H(z^k)^T\left[-H(z^k) + \frac{2\beta_k}{1+\mu_k}h\right] + w_k(\alpha) \\
&\leq (1-2\alpha)\Psi(z^k) + \frac{4\alpha\mu_0}{1+\mu_k}\beta_k\|H(z^k)\| + w_k(\alpha) \\
&\leq (1-2\alpha)\Psi(z^k) + \frac{4\alpha\mu_0\gamma}{1+\mu_k}\Psi(z^k) + w_k(\alpha) \\
&= \left[1 - 2\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\alpha\right]\Psi(z^k) + w_k(\alpha),
\end{aligned} \tag{4.21}$$

where the second inequality follows from $\beta_k \le \gamma\|H(z^k)\|$ since $\min\{1,\xi^2\} \le \xi$ for all $\xi \ge 0$. Since the function Ψ is continuously differentiable for any $z \in \mathcal{R}_{++} \times \mathcal{J} \times \mathcal{J}$, it follows from the definition of the function $w_k(\alpha)$ and the fact $\mu_k > 0$ that $w_k(\alpha) = o(\alpha)$. This, together with (4.21) and $0 < \mu_0\gamma < 1/2$, implies that there exists a constant $\bar{\alpha} \in (0, 1/2)$ such that

$$\Psi(z^k + \alpha\Delta z^k) \le \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\alpha\right]\Psi(z^k) \tag{4.22}$$

holds for any $\alpha \in (0, \bar{\alpha}]$ and $\sigma \in (0, 1/2)$. Now we prove $\Psi(z^k) \le C_k + \epsilon_k$. If $k = 0$, then $\Psi(z^0) \le \Psi(z^0) + \epsilon_0 = C_0 + \epsilon_0$. If $k \ge 1$, then by (4.8) we have

$$\begin{aligned} C_k &= (1-\theta)C_{k-1} + \theta\Psi(z^k) \\ &\le (1-\theta)C_{k-1} + \theta(C_{k-1} + \epsilon_{k-1}) \\ &= C_{k-1} + \theta\epsilon_{k-1}, \end{aligned} \tag{4.23}$$

where the inequality holds since $\Psi(z^k) \le C_{k-1} + \epsilon_{k-1}$ by (4.5). Hence, we have $C_{k-1} \ge C_k - \theta\epsilon_{k-1}$. Using this result, we can obtain from (4.8) that

$$\begin{aligned} \Psi(z^k) &= \frac{C_k - (1-\theta)C_{k-1}}{\theta} \\ &\le \frac{C_k - (1-\theta)(C_k - \theta\epsilon_{k-1})}{\theta} \\ &= C_k + (1-\theta)\epsilon_{k-1} \\ &\le C_k + (1-\tau)\epsilon_{k-1} \\ &= C_k + \epsilon_k. \end{aligned} \tag{4.24}$$

Hence, from (4.22) and (4.24) we can obtain that

$$\Psi(z^k + \alpha\Delta z^k) \le \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\alpha\right](C_k + \epsilon_k) \tag{4.25}$$

holds for any $\alpha \in (0, \bar{\alpha}]$ and $\sigma \in (0, 1/2)$. Let $\tilde{\alpha} := \min\{\bar{\alpha}, \frac{1-\mu_k}{\mu_k}\}$. It is easy to see that (4.25) and the inequality

$$(1+\alpha)\mu_k < 1$$

hold for any $\alpha \in (0, \tilde{\alpha})$. This demonstrates that Step 3 is well-defined at the kth iteration. Hence, $z^{k+1} = (\mu_{k+1}, x^{k+1}, y^{k+1})$ can be generated by Algorithm 4.1.

Now we prove $0 < \mu_{k+1} < 1$ and $\mu_{k+1} \ge \mu_0\beta_{k+1}$. Since $\alpha_k \in (0, 1/2)$, by (4.20) we know that

$$\mu_{k+1} = \mu_k + \alpha_k\Delta\mu_k \ge (1 - 2\alpha_k)\mu_k + 2\alpha_k\mu_0\beta_k > 0.$$

Moreover, since $0 < \mu_k < 1$, by the left inequality in (4.9), we get $\mu_k < (1+\mu_k)\ln(1+\mu_k)$. Using this fact, we obtain from (4.19) that

$$\mu_{k+1} = \mu_k - \alpha_k(1+\mu_k)\ln(1+\mu_k) + 2\alpha_k\mu_0\beta_k \le (1-\alpha_k)\mu_k + 2\alpha_k\mu_0\beta_k,$$

which, together with $\mu_k \ge \mu_0\beta_k$ and (4.6), yields that

$$\mu_{k+1} \le (1-\alpha_k)\mu_k + 2\alpha_k\mu_0\beta_k \le (1+\alpha_k)\mu_k < 1.$$

Since $\mu_k \ge \mu_0\beta_k$, from (4.20) it follows that

$$\mu_{k+1} \ge (1-2\alpha_k)\mu_k + 2\alpha_k\mu_0\beta_k \ge (1-2\alpha_k)\mu_0\beta_k + 2\alpha_k\mu_0\beta_k = \mu_0\beta_k \ge \mu_0\beta_{k+1},$$

where the last inequality holds since $\{\beta_k\}$ is monotonically decreasing by its definition. The proof is completed. $\square$

Theorem 4.6. *Suppose that F has the Cartesian P_0-property. Then Algorithm* 4.1 *is well-defined and generates an infinite sequence $\{z_k = (\mu_k, x^k, y^k)\}$ with $0 < \mu_k < 1$ and $\mu_k \geq \mu_0\beta_k$ for all $k \geq 0$.*

Proof. By Step 0 of Algorithm 4.1, we have $0 < \mu_0 < 1$. Moreover, by (4.3) we get $\beta_0 = \gamma \min\{1, \Psi(z^0)\} \leq \gamma$ and hence $\mu_0 \geq \mu_0\gamma \geq \mu_0\beta_0$. So, it follows from Lemma 4.5 that $z^1 = (\mu_1, x^1, y^1)$ can be generated by Algorithm 4.1 with $0 < \mu_1 < 1$ and $\mu_1 \geq \mu_0\beta_1$. Then, by repeatedly resorting to Lemma 4.5, we obtain the desired result. The proof is completed. □

5 Convergence Properties of Algorithm 4.1

In this section, we analyze the convergence properties of Algorithm 4.1. First, we establish its global convergence. For this purpose, we need the coerciveness of the function H.

Lemma 5.1. *Suppose that F has the Cartesian P_0-property and that H is defined by* (4.1)*. Then $H(\mu, x, y)$ is coercive in (x, y) for each $\mu > 0$, i.e.,*

$$\lim_{\|(x,y)\|\to\infty} \|H(\mu, x, y)\| = \infty.$$

Proof. Suppose that the result of the lemma doesn't hold. Then there exist $\mu \in [\tilde{\mu}, \bar{\mu}]$ and an unbounded sequence $\{(x^k, y^k)\}$ such that $\{H(\mu, x^k, y^k)\}$ is bounded. Since

$$\|H(\mu, x^k, y^k)\|^2 = (\ln(1+\mu))^2 + \|F(x^k) - y^k\|^2 + \|\Phi(\mu, x^k, y^k)\|^2,$$

it follows that $\{F(x^k) - y^k\}$ is bounded. Denote $g(x^k, y^k) := y^k - F(x^k)$, then $\{g(x^k, y^k)\}$ is bounded and $y^k = F(x^k) + g(x^k, y^k)$. First, we prove that $\{x^k\}$ is unbounded. In fact, if $\{x^k\}$ is bounded, then $\{F(x^k)\}$ is bounded by the continuity of F, and $\{y^k\}$ is unbounded since $\{(x^k, y^k)\}$ is unbounded. Therefore, we obtain that $\{y^k - F(x^k)\}$ is unbounded. A contradiction is derived. Notice that $x^k = (x_1^k, ..., x_m^k)$ with $x_i^k \in \mathcal{K}^{n_i}$ for each k. Define the index set

$$N := \{i \in \{1, ..., m\} | \{x_i^k\} \text{ is unbounded}\}.$$

Since $\{x^k\}$ is unbounded, the index set N is nonempty. Let $\{\bar{x}^k\}$ be a bounded sequence with $\bar{x}^k = (\bar{x}_1^k, ..., \bar{x}_m^k)$ and $\bar{x}_i^k \in \mathcal{K}^{n_i}$ for each k, where $\{\bar{x}_i^k\}$ is defined as follows:

$$\bar{x}_i^k = \begin{cases} 0, & \text{if } i \in N, \\ x_i^k, & \text{otherwise.} \end{cases}$$

Set $\bar{y}^k := F(\bar{x}^k) + g(x^k, y^k)$. Then, $\{\bar{y}^k\}$ is bounded since $\{g(x^k, y^k)\}$ is bounded and $\{F(\bar{x}^k)\}$ is also bounded by the continuity of F. By the Cartesian P_0-property of F, we have $v > 0$ such that

$$\begin{aligned} 0 &\leq \max_{i=1,\ldots,m} \langle x_i^k - \bar{x}_i^k, F_i(x^k) - F_i(\bar{x}^k)\rangle \\ &= \max_{i=1,\ldots,m} \langle x_i^k - \bar{x}_i^k, y_i^k - \bar{y}_i^k\rangle \\ &= \langle x_v^k - \bar{x}_v^k, y_v^k - \bar{y}_v^k\rangle, \end{aligned}$$

where v is an index from $\{1, ..., m\}$ for which the maximum is attained. Clearly, $v \in N$, which means that $\{x_v^k\}$ is unbounded and $\bar{x}_v^k = 0$, and hence $\{(x_v^k, y_v^k)\}$ is unbounded and $\{(\bar{x}_v^k, \bar{y}_v^k)\}$ is bounded. So, by Lemma 3.3 we have $\lim_{k\to\infty} \phi(\mu, x_v^k, y_v^k) = \infty$ and hence $\lim_{k\to\infty} \Phi(\mu, x^k, y^k) = \infty$. This contradicts the boundedness of $\{H(\mu, x^k, y^k)\}$. So, we obtain the desired result. □

Lemma 5.2. *Suppose that the sequence* $\{z^k = (\mu_k, x^k, y^k)\}$ *is generated by Algorithm* 4.1. *Then* $\{C_k\}$ *and* $\{\Psi(z^k)\}$ *are bounded for all* $k \geq 0$.

Proof. On one hand, by (4.8) we have $\epsilon_k = (1-\tau)\epsilon_{k-1} = (1-\tau)^k \epsilon_0$. This implies that the sequence $\{\epsilon_k\}$ is bounded. On the other hand, by (4.23) we can conclude that

$$C_k \leq C_{k-1} + \theta\epsilon_{k-1} \leq C_{k-2} + \theta\epsilon_{k-2} + \theta\epsilon_{k-1} \leq \cdots \leq C_0 + \theta\Sigma_{i=0}^{k-1}\epsilon_i.$$

Since $\Sigma_{i=0}^{k-1}\epsilon_i = \Sigma_{i=0}^{k-1}(1-\tau)^i\epsilon_0 < \infty$, we get that $\{C_k\}$ is bounded. From (4.24), we know that $\Psi(z^k) \leq C_k + \epsilon_k$ for all $k \geq 0$. Hence, $\{\Psi(z^k)\}$ is bounded. This completes the proof. $\square$

Theorem 5.3. *Suppose that F has the Cartesian P_0-property and that $\{z^k\}$ is the iteration sequence generated by Algorithm* 4.1. *Then the following results hold.*
(i) *$\{\Psi(z^k)\}$ converges to zero and hence any accumulation point of $\{z^k\}$ is a solution of $H(z) = 0$.*
(ii) *If the solution set of the* P_0-SCNCP *is nonempty and bounded, then $\{z^k\}$ is bounded.*

Proof. Since $\{\beta_k\}$ is monotonically decreasing and bounded from below by zero, it is convergent. Thus, there exists $\beta^* \geq 0$ such that $\lim_{k\to\infty}\beta_k = \beta^*$. We now assume that $\beta^* > 0$ and derive a contradiction. Since $0 < \mu_k < 1$ and $\mu_k \geq \mu_0\beta_k$ for all $k \geq 0$ by Theorem 4.6, we obtain that for all $k \geq 0$

$$0 < \mu_0\beta^* \leq \mu_0\beta_k \leq \mu_k < 1$$

This implies that $\{z^k\}$ is bounded because otherwise $\{\Psi(z^k)\}$ must be unbounded by Lemma 5.1 which contradicts with Lemma 5.2. Hence, $\{z^k\}$ has at least one accumulation point $z^* := (\mu^*, x^*, y^*)$. Without loss of generality, we assume that $\lim\limits_{k\to\infty} z^k = z^*$. From Lemma 5.2, we know that the sequence $\{C_k\}$ is bounded and hence it has a convergent subsequence, denoted by $\{C_k\}_{k\in\mathcal{I}}$ where $\mathcal{I} \subset \{0, 1, 2, ...\}$. Then there exists $C^* \geq 0$ such that $\lim\limits_{\mathcal{I}\ni k\to\infty} C_k = C^*$. Thus, from (4.8) and the continuity of Ψ, we can obtain that

$$C^* = \lim_{\mathcal{I}\ni k\to\infty} \frac{C_k - (1-\theta)C_{k-1}}{\theta} = \lim_{\mathcal{I}\ni k\to\infty} \Psi(z^k) = \Psi(z^*).$$

Since $\beta^* > 0$, by the definition of β_k in (4.3), we have $C^* = \Psi(z^*) > 0$. Since $\mu_0\beta_k \leq \mu_k < 1$ by Theorem 4.6, we have

$$0 < \mu_0\beta^* = \lim_{k\to\infty} \mu_0\beta_k \leq \lim_{k\to\infty} \mu_k = \mu^* < 1.$$

Then, it follows from Lemma 4.4 that $DH(z^*)$ exists and is invertible. Hence, there exists a closed neighborhood $N(z^*)$ of z^* such that for any $z \in N(z^*)$ we have $0 < \mu < 1$ and $DH(z)$ is invertible. Then, for all sufficiently large $k \in \mathcal{I}$, we have $z^k \in N(z^*)$ since $\lim_{k\to\infty} z^k = z^*$ and hence $0 < \mu_k < 1$ and $DH(z^k)$ is invertible. Let Δz^k be the unique solution to the system of equations

$$DH(z^k)\Delta z^k = -H(z^k) + \frac{2\beta_k}{1+\mu_k}h, \quad k \in \mathcal{I}.$$

Similarly to the proof of Lemma 4.5, for all sufficiently large $k \in \mathcal{I}$, there exists a nonnegative integer $\bar{l}$ such that $\delta^{\bar{l}} \in (0, \tilde{\alpha})$ and

$$\Psi(z^k + \delta^{\bar{l}}\Delta z^k) \leq \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\delta^{\bar{l}}\right](C_k + \epsilon_k), \quad k \in \mathcal{I}$$

and

$$(1+\delta^{\bar{l}})\mu_k < 1, \quad k \in \mathcal{I}.$$

For all sufficiently large $k \in \mathcal{I}$, since $\alpha_k \geq \delta^{\bar{l}}$, it follows from Steps 3 and 4 in Algorithm 4.1 that

$$\Psi(z^{k+1}) \leq \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\alpha_k\right]\Psi(z^k) \leq \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu_k}\right)\delta^{\bar{l}}\right](C_k + \epsilon_k), \quad k \in \mathcal{I}.$$

Notice that $\lim\limits_{k\to\infty} \epsilon_k = 0$. Taking limits on both sides of the above inequality, we obtain that

$$\Psi(z^*) \leq \left[1 - 2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu^*}\right)\delta^{\bar{l}}\right]C^*,$$

which together with $C^* = \Psi(z^*) > 0$ implies that $2\sigma\left(1 - \frac{2\mu_0\gamma}{1+\mu^*}\right)\delta^{\bar{l}} \leq 0$. This contradicts the fact that $0 < \sigma < 1/2$ and $0 < \mu_0\gamma < 1/2$. Hence, we have $\beta^* = 0$. Furthermore, from the definition of β_k given in (4.3), it follows that there exists a subsequence $\{z^{k_n}\}$ of $\{z^k\}$ such that $\lim\limits_{k_n\to\infty} \Psi(z^{k_n}) = 0$. Since Algorithm 4.1 performs the monotone line search when $\Psi(z^k)$ is sufficiently small, we have $\Psi(z^{k+1}) \leq \Psi(z^k)$ when k is sufficiently large. Thus, we may obtain that $\lim\limits_{k\to\infty} \Psi(z^k) = 0$. Let z^* be an arbitrary accumulation point of $\{z^k\}$. Then, there exists a subsequence $\{z^{k_j}\} \subseteq \{z^k\}$ such that $\{z^{k_j}\}$ converges to z^* as $k_j \to \infty$. Then, it follows from the continuity of Ψ that $\Psi(z^*) = \lim\limits_{k_j\to\infty} \Psi(z^{k_j}) = 0$ and hence $H(z^*) = 0$.

Next, we prove the result (ii). Since $\lim_{k\to\infty} \Psi(z^k) = 0$, we have $\lim_{k\to\infty} \mu_k = 0$ and $\lim_{k\to\infty} \|\Phi(z^k)\| = 0$. Thus, by the famous mountain pass theorem (see, [21, Theorem 9.2.7]) and by following the similar proof lines of [25, Theorem 5.2], we can prove that $\{(x^k, y^k)\}$ is bounded and hence $\{z^k\}$ is bounded. So, we complete the proof. □

Now we analyze the rate of convergence for Algorithm 4.1. For this purpose, we need the strong semi-smoothness of the function H which can be obtained by Lemma 3.4.

Theorem 5.4. *Suppose that F has the Cartesian P_0-property and that DF is Lipschitz continuous on $\mathcal{J}$. Let z^* be an accumulation point of the iteration sequence $\{z^k\}$ generated by Algorithm 4.1. If all $V \in \partial H(z^*)$ are nonsingular, then $\|z^{k+1} - z^*\| = O(\|z^k - z^*\|^2)$.*

Proof. The proof is similar to [25, Theorem 5.3]. For brevity, we omit the details here. □

6 Numerical Experiments

In this section, we report some numerical results of Algorithm 4.1 for solving the following second-order cone complementarity problem (SOCCP):

$$\text{Find } (x,y) \in \mathcal{R}^n \times \mathcal{R}^n \quad \text{such that } x \in \mathcal{K},\ \ y \in \mathcal{K},\ \ y = F(x),\ \ \langle x, y\rangle = 0, \tag{6.1}$$

where $\mathcal{K} \subset \mathcal{R}^n$ is the Cartesian product of second-order cones, i.e., $\mathcal{K} = \mathcal{K}^{n_1} \times \cdots \times \mathcal{K}^{n_m}$, and the n_i-dimensional second-order cone (SOC) $\mathcal{K}^{n_i} \subset \mathcal{R}^{n_i}$ is defined by

$$\mathcal{K}^{n_i} := \{(x_1, x_2^T)^T \in \mathcal{R} \times \mathcal{R}^{n_i - 1} : x_1 \geq \|x_2\|\}.$$

All experiments are performed on a Intel(R) Core(TM) i7-4790 CPU 3.60GHz personal computer with 8.00GB memory. The program codes are written in MATLAB and run in MATLAB R2012b environment. The parameters used in Algorithm 4.1 are chosen as $\mu_0 = 10^{-2},\ \sigma = 0.2,\ \delta = 0.8,\ \gamma = 10^{-4},\ \ c = 10^{-6}$. Moreover, we use $\|H(z^k)\| \leq 10^{-8}$ as the stopping criterion.

6.1 Linear SOCCPs

Proposition 6.1. Consider the SOCCP (6.1), in which $\mathcal{K} = \mathcal{K}^2 \times \mathcal{K}^2$ and $y = Mx + q$ with

$$M = \begin{pmatrix} 0 & 0 & 0 & \alpha \\ 0 & 0 & 0 & \alpha \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \beta \end{pmatrix} \quad \text{with} \quad \alpha, \beta > 0, \quad \text{and} \quad q = \begin{pmatrix} 10 \\ 1 \\ 2 \\ 3 \end{pmatrix}.$$

This problem was proposed by Lu and Huang [15]. From Proposition 2.1 in [15], we know that Problem 6.1 is a class of P_0-SCLCPs. We test several specific P_0-SCLCPs by choosing α, β as follows:

(P1) $\alpha = 5,\ \beta = 10$; (P2) $\alpha = 10,\ \beta = 5$; (P3) $\alpha = 10,\ \beta = 20$; (P4) $\alpha = 20,\ \beta = 10$; (P5) $\alpha = 20,\ \beta = 25$; (P6) $\alpha = 10,\ \beta = 50$.

For our algorithm, we let $\theta = 0.8$, $\tau = 0.5$ and $\epsilon_0 = 10$, and choose $x^0 = (1, ..., 1)^T, y^0 = Mx^0 + q$ as the starting point. Table 1 lists the results, in which **P** denotes the tested problem, **IT** and **CPU** denote the value of the number of iterations and the CPU time in seconds, and **SOLU** denotes the solution of the concerned problem obtained by our algorithm. From Table 1, it can be seen that, for every case we tested, our algorithm may find a solution of the concerned problem meeting the desired accuracy in very few iterations and in short CPU time. Although the reported numerical results are preliminary, they demonstrate that the new non-monotone smoothing algorithm is promising for solving the P_0-SCLCPs.

Table 1 Numerical results for different cases of Problem 6.1 by Algorithm 4.1

P	IT	CPU	SOLU	
(P1)	3	0.008	$x^* = (0, 0, 0.1, -0.1)^T$,	$y^* = (9.5, 0.5, 2, 2)^T$
(P2)	3	0.008	$x^* = (0, 0, 0.2, -0.2)^T$,	$y^* = (8, -1, 2, 2)^T$
(P3)	3	0.010	$x^* = (0, 0, 0.05, -0.05)^T$,	$y^* = (9.5, 0.5, 2, 2)^T$
(P4)	3	0.010	$x^* = (0, 0, 0.1, -0.1)^T$,	$y^* = (8, -1, 2, 2)^T$
(P5)	3	0.012	$x^* = (0, 0, 0.04, -0.04)^T$,	$y^* = (9.2, 0.2, 2, 2)^T$
(P6)	3	0.012	$x^* = (0, 0, 0.02, -0.02)^T$,	$y^* = (9.8, 0.8, 2, 2)^T$

Proposition 6.2. Consider the SOCCP (6.1), in which $\mathcal{K} = \mathcal{K}^n$ and $y = Mx + q$, where $q \in \mathcal{R}^n$ and $M \in \mathcal{R}^{n \times n}$ is a rank-deficient positive semi-definite matrix.

In the experiments, we let the rank l of M be an integer randomly chosen from $[0.5n, n-1]$. In order to obtain a positive semi-definite matrix M with $l < n$, we let $M = nBB^T / \|BB^T\|$, where $B = \text{rand}(n, l)$. Furthermore, we let $q := n^{1/2}e - Me$, where $e = (1, 0, ..., 0)^T$ is the unit element in $\mathcal{K}^n$. Then, Problem 6.2 has a solution since M is positive semi-definite and there exist $\bar{x} := e \in \text{int}\mathcal{K}^n$ and $\bar{y} := n^{1/2}e \in \text{int}\mathcal{K}^n$ such that $\bar{y} = M\bar{x} + q$.

We generate 100 problem instances for each size of $n = 200, 400, ..., 1200$, and test these problems by using the starting points: (1) $x^0 = y^0 = e$; (2) $x^0 = e$, $y^0 = Mx_0 + q$. In the following, we compare three algorithms:

(i) the new non-monotone algorithm, denoted by **Non (i)**, corresponding to $\theta = 0.8$, $\tau = 0.5$ and $\epsilon_0 = 10$ in Algorithm 4.1;

(ii) the non-monotone algorithm based on the non-monotone line search scheme introduce by Zhang and Hager [30] with $\eta_k = 0.8$, denoted by **Non (ii)**;

(iii) the monotone line search algorithm, denoted by **Mon**, corresponding to $\theta = 1$, $\tau = 1$ and $\epsilon_0 = 0$ in Algorithm 4.1.

Figure 1 shows the convergence behavior of **Non (i)** for one of the test problems with $n = 1000$. From Figure 1, we may find that the sequence $\{\|H(z^k)\|\}$ is non-monotonically

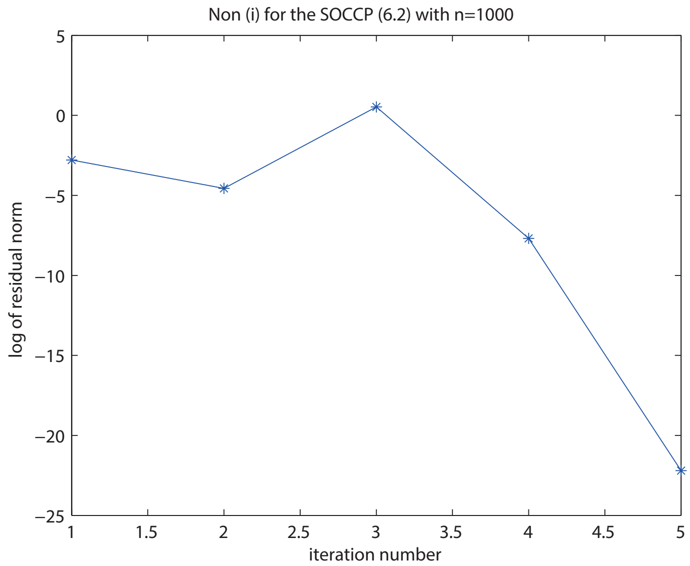

Fig.1: The logarithm of residual norm $\|H(z)\|$ by iterations

decreasing and converges to zero. Table 2 lists the numerical results, in which **SP** denotes the starting point, n denotes the size of the problem, **AIT** and **ACPU** denote the average value of the number of iterations and the CPU time in seconds when the algorithm terminates among the 100 testing. From table 2, we may find that either of the non-monotone algorithms is superior to the monotone algorithm.

Table 2 Numerical comparisons of algorithms for Problem 6.2

		Non(i)		Non (ii)		Mon	
SP	n	AIT	ACPU	AIT	ACPU	AIT	ACPU
(1)	200	5.65	0.07	5.65	0.07	5.39	0.07
	400	5.17	0.34	5.24	0.36	5.68	0.38
	600	5.09	0.89	5.18	0.97	6.10	1.09
	800	5.02	1.87	5.12	1.91	6.10	2.27
	1000	4.99	3.35	5.08	3.36	6.20	4.02
	1200	5.01	5.71	5.07	5.78	6.31	6.79
(2)	200	4.84	0.06	4.61	0.06	5.18	0.07
	400	4.65	0.31	4.76	0.31	5.08	0.33
	600	4.80	0.89	4.98	0.90	5.20	0.97
	800	4.90	1.87	5.02	1.89	5.20	2.05
	1000	5.01	3.11	5.07	3.32	5.25	3.64
	1200	5.06	5.26	5.14	5.85	5.32	5.82

Proposition 6.3. Consider the SOCCP (6.1), in which $\mathcal{K} = \mathcal{K}^{n_1} \times \cdots \times \mathcal{K}^{n_m}$, $x = (x_1, ..., x_m) \in \mathcal{R}^n$ with $x_i \in \mathcal{R}^{n_i}$ and $n = \sum_{i=1}^m n_i$, and $y = Mx + q$.

The matrix $M \in \mathcal{R}^{n\times n}$ and the vector $q = (q_1, ..., q_m) \in \mathcal{R}^n$ are generated by the following procedure. We choose $N_i = \text{rand}(n_i, n_i)$ for $i = 1, ..., m$ and then let M be the block diagonal matrix with $N_1^T N_1, ..., N_m^T N_m$ as block diagonals, i.e., $M = \text{diag}\{N_i^T N_i\}_{i=1}^m$. Moreover, by choosing $q_i = (q_{i1}, q_{i2}^T)^T \in \mathcal{R}^{n_i}$ with $q_{i2} = \text{rand}(n_i - 1, 1)$ and $q_{i1} = \|q_{i2}\| + 1$, we let $q = (q_1, ..., q_m) \in \mathcal{R}^n$. It is easy to verify that the function $y = Mx + q$ generated by such way has the Cartesian P_0-property.

In the experiments, we let $m = 4$ and $n_i = \frac{n}{4}$ for any $i = 1, ..., m$. We use $x^0 = e$ and $y^0 = e$ as the starting point. We generate 100 problem instances for each size of $n = 100, ..., 800$. For comparison purposes, we apply Algorithm 4.1 ($\theta = 0.8$, $\tau = 0.5$ and $\epsilon_0 = 10$) and the smoothing Newton algorithm studied by Lu and Huang [15] to solve these tested problems, respectively. Table 3 lists the numerical results, in which **AGAP** denotes the average value of $|\langle x^k, y^k\rangle|$ when the algorithm terminates among the 100 testing. From Table 3, we may find that our algorithm has some advantages over the algorithm in [15].

Table 3 Numerical comparisons of algorithms for Problem 6.3

	Algorithm 4.1			Algorithm in [15]		
n	AIT	ACPU	AGAP	AIT	ACPU	AGAP
100	6.97	0.03	2.8488×10^{-11}	8.08	0.03	2.0965×10^{-8}
200	8.47	0.09	2.9975×10^{-11}	9.76	0.09	9.6743×10^{-8}
300	9.30	0.21	1.1793×10^{-10}	11.08	0.24	9.0962×10^{-8}
400	9.74	0.38	9.0609×10^{-11}	12.11	0.52	9.2606×10^{-8}
500	10.15	0.72	1.2751×10^{-10}	12.80	0.93	8.7162×10^{-8}
600	10.13	1.15	2.8607×10^{-10}	13.55	1.52	6.9206×10^{-8}
700	10.45	1.83	4.9905×10^{-10}	14.40	2.41	5.8967×10^{-8}
800	11.15	2.68	2.1310×10^{-10}	15.01	3.52	5.1880×10^{-8}

6.2 Nonlinear SOCCPs

Proposition 6.4. Consider the SOCCP (6.1), where $\mathcal{K} = \mathcal{K}^3$ and $F : \mathcal{R}^3 \to \mathcal{R}^3$ is given by

$$F(x) = \begin{pmatrix} 0.07x_1^3 - 4 \\ 0.04x_2^3 - 3.93 \\ 0.03x_3^3 - 5.72 \end{pmatrix}.$$

By Algorithm 4.1, we obtain one solution $x^* = (5, 3, 4)^T$. Since the Jacobian $F'(x) = \text{diag}\{0.21x_1^2, 0.12x_2^2, 0.09x_3^2\}$ is positive semidefinite, F is monotone. We test this problem by using the starting point $x^0 = y^0$: (1) $(1, ..., 1)^T$; (2) $(-1, ..., -1)^T$; (3) $(10, ..., 10)^T$; (4) $(50, ..., 50)^T$; (5) $(100, ..., 100)^T$; (6) $(200, ..., 200)^T$. Numerical results are listed in Table 4, where **SP** denotes the starting point and **GAP** denotes the value of $|\langle x^k, y^k\rangle|$ when the algorithm terminates.

Table 4 Numerical results of Problem 6.4 by Algorithm 4.1

SP	IT	CPU	GAP
(1)	6	0.01	4.0461×10^{-13}
(2)	6	0.01	1.3517×10^{-10}
(3)	6	0.01	2.6098×10^{-11}
(4)	10	0.02	2.7881×10^{-11}
(5)	12	0.02	8.6730×10^{-10}
(6)	14	0.02	1.0569×10^{-14}

Proposition 6.5. Consider the SOCCP (6.1), where $\mathcal{K} = \mathcal{K}^3 \times \mathcal{K}^2$ and $F : \mathcal{R}^5 \to \mathcal{R}^5$ is given by

$$F(x) = \begin{pmatrix} 24(2x_1 - x_2)^3 + \exp(x_1 - x_3) - 4x_4 + x_5 \\ -12(2x_1 - x_2)^3 + 3(3x_2 + 5x_3)/\sqrt{1 + (3x_2 + 5x_3)^2} - 6x_4 - 7x_5 \\ -\exp(x_1 - x_3) + 5(3x_2 + 5x_3)/\sqrt{1 + (3x_2 + 5x_3)^2} - 3x_4 + 5x_5 \\ 4x_1 + 6x_2 + 3x_3 - 1 \\ -x_1 + 7x_2 - 5x_3 + 2 \end{pmatrix}.$$

From [5] we know that F is monotone. By Algorithm 4.1, we obtain one solution $x^* \approx (0.2324, -0.0731, 0.2206, 0.5339, -0.5339)^T$. We test this problem by using the starting point $x^0 = y^0$: (1) $(0, ..., 0)^T$; (2) $(1, ..., 1)^T$; (3) $(-1, ..., -1)^T$; (4) $(10, ..., 10)^T$; (5) $(-10, ..., -10)^T$; (6) $(50, ..., 50)^T$. Numerical results are listed in Table 5.

Table 5 Numerical results of Problem 6.5 by Algorithm 4.1

SP	IT	CPU	GAP
(1)	6	0.02	2.8403×10^{-10}
(2)	6	0.02	2.1434×10^{-10}
(3)	12	0.03	2.4774×10^{-15}
(4)	17	0.03	3.9733×10^{-16}
(5)	13	0.03	1.6440×10^{-10}
(6)	14	0.03	1.9601×10^{-10}

Proposition 6.6. Consider the SOCCP (6.1), where $\mathcal{K} = \mathcal{K}^4$ and $F : \mathcal{R}^4 \to \mathcal{R}^4$ is given by

$$F(x) = \begin{pmatrix} e^{x_1} + x_1^2 \\ e^{x_2} + x_2^2 \\ e^{x_3} + x_3^2 \\ e^{x_4} + x_4^2 \end{pmatrix}.$$

By Algorithm 4.1, we obtain one solution $x^* \approx (0.3278, -0.1893, -0.1893, -0.1893)^T$. We test this problem by using the starting point $x^0 = y^0$: (1) $(1, ..., 1)^T$; (2) $(-1, ..., -1)^T$; (3) $(5, ..., 5)^T$; (4) $(-5, ..., -5)^T$; (5) $(10, ..., 10)^T$; (6) $(-10, ..., -10)^T$. Numerical results are listed in Table 6.

Table 6 Numerical results of Problem 6.6 by Algorithm 4.1

SP	IT	CPU	GAP
(1)	8	0.01	1.5701×10^{-16}
(2)	10	0.01	3.9503×10^{-16}
(3)	33	0.02	3.7616×10^{-16}
(4)	11	0.01	7.7079×10^{-16}
(5)	24	0.02	2.8401×10^{-16}
(6)	11	0.01	4.7502×10^{-16}

7 Conclusions

In this paper, based on the new smoothing function ϕ in (3.1) and the non-regulation function $h(\mu) = \ln(1 + \mu)$, we study a non-monotone smoothing algorithm for the P_0-SCNCP, which includes the monotone SCCP as a special cases. Under wake conditions, we prove that the algorithm is globally and locally quadratically convergent. The preliminary numerical results demonstrate that our algorithm is promising for solving the P_0-SCNCP. Although $h(\mu) = \ln(1+\mu)$ does not satisfy the property $h(\mu) \le \mu h'(\mu)$ for any $\mu > 0$, from Lemma 4.3 it satisfies

$$h(\mu) \le 2\mu h'(\mu) \quad \text{for any } \mu \in (0, 1).$$

Hence, it is worthy to analyze smoothing algorithms under a more generalized regulation function $h(\mu)$, which satisfies properties (a)–(c) and (e) in Definition 1.1 and

(d') there exist constants $c_1 > 0$ and $c_2 > 0$ such that $h(\mu) \le c_1 \mu h'(\mu)$ for all $\mu \in (0, c_2)$.

Acknowledgement

The authors are grateful to the referees for their careful reading and helpful suggestions that improved the paper greatly.

References

[1] F. Facchinei and J.-S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problems*, I and II, Springer-Verlag, New York, 2003.

[2] J. Faraut and A. Korányi, *Analysis on Symmetric Cones*, Oxford Mathematical Monographs, Oxford University Press, New York, 1994.

[3] M.C. Ferris and J.-S. Pang, Engineering and economic applications of complementarity problems, *SIAM Rev.* 39 (1997) 669–713.

[4] M.S. Gowda, R. Sznajder and J. Tao, Some $P-$properties for linear transformations on Euclidean Jordan algebras, *Linear Algebra Appl.* 393 (2004) 203–232.

[5] S. Hayashi, N. Yamashita and M. Fukushima, A combined smoothing and regularized method for monotone second-order cone complementarity problems, *SIAM J. Optim.* 15 (2005) 593–615.

[6] S.L. Hu, Z.H. Huang and P. Wang, A non-monotone smoothing Newton algorithm for solving nonlinear complementarity problems, *Optim. Methods Softw.* 24 (2009) 447–460.

[7] Z.H. Huang, S.L. Hu and J.Y. Han, Convergence of a smoothing algorithm for symmetric cone complementarity problems with a nonmonotone line search, *Science in China Series A: Math.* 52 (2009) 833–848.

[8] Z.H. Huang and T. Ni, Smoothing algorithms for complementarity problems over symmetric cones, *Comput. Optim. Appl.* 45 (2010) 557–579.

[9] L. Kong, J. Sun and N. Xiu, A regularized smoothing Newton method for symmetric cone complementarity problems, *SIAM J. Optim.* 19 (2008) 1028–1047.

[10] N. Krejić and S. Rapajić, Globally convergent Jacobian smoothing inexact Newton methods for NCP, *Comput. Optim. Appl.* 41 (2008) 243–261.

[11] G. Lesaja, G.Q. Wang and D.T. Zhu, Interior-point methods for cartesian $p_*(k)$-linear complementarity problems over symmetric cone based on the eligible kernel functions, *Optim. Methods Softw.* 27 (2012) 827–843.

[12] L.X. Liu, S.Y. Liu and H.W. Liu, A predictor-corrector smoothing newton method for symmetric cone complementarity problems, *Appl. Math. Comput.* 217 (2010) 2989–2999.

[13] Y.M. Li, X.T. Wang and D.Y. Wei, Improved smoothing Newton methods for symmetric cone complementarity problems, *Optim. Lett.* 6 (2012) 471–487.

[14] Y.M. Li and D.Y. Wei, A generalized smoothing Newton method for the symmetric cone complementarity problem, *Appl. Math. Comput.* 264 (2015) 335–345.

[15] N. Lu and Z.H. Huang, A smoothing Newton algorithm for a class of non-monotonic symmetric cone linear complementarity problems, *J. Optim. Theory Appl.* 161 (2014) 446–464.

[16] Z.Y. Luo and N.H. Xiu, Path-following interior point algorithms for the Cartesian $P_*(\kappa)$-LCP over symmetric cones, *Science in China Series A: Math.* 52 (2009) 1769–1784.

[17] C.F. Ma, A regularized smoothing Newton method for solving the symmetric cone complementarity problem, *Math. Comput. Modell.* 54 (2011) 2515–2527.

[18] T. Ni and P. Wang, A smoothing-type algorithm for solving nonlinear complementarity problems with a non-monotone line search, A*ppl. Math. Comput.* 216 (2010) 2207–2214.

[19] T. Ni and W.Z. Gu, Smoothing Newton algorithm for symmetric cone complementarity problems based on a one-parametric class of smoothing functions, *J. Appl. Math. Comput.* 35 (2011) 73–92.

[20] T. Ni, W.Z. Gu and J. Zhai, An inexact smoothing-type algorithm for support vector machines, *Neurocomput.* 129 (2014) 127–135.

[21] R.S. Palais and C.L. Terng, *Critical Point Theory and Submanifold Geometry.* Lecture Notes in Mathematics, vol. 1353. Springer, Berlin, 1988.

[22] D. Sun and J. Sun, Löwner's operator and spectral functions in Euclidean Jordan algebras, *Math. Oper. Res.* 33 (2008) 421–445.

[23] J.Y. Tang, L. Dong, L. Fang and J.C. Zhou, The convergence of a modified smoothing-type algorithm for the symmetric cone complementarity problem, *J. Appl. Math. Comput.* 43 (2013) 307–328.

[24] J.Y. Tang, L. Dong, L. Fang and J.C. Zhou, Smoothing Newton algorithm for the second-order cone programming with a nonmonotone line search, *Optim. Lett.* 8 (2014) 1753–1771.

[25] J.Y. Tang, L. Dong, J.C. Zhou and L. Sun, A smoothing-type algorithm for the second order cone complementarity problem with a new nonmonotone line search, *Optim.* 64 (2015) 1935–1955.

[26] J.Y. Tang, J.C. Zhou and L. Fang, A non-monotone regularization Newton method for the second-order cone complementarity problem, *Appl. Math. Comput.* 271 (2015) 743–756.

[27] G.Q. Wang and Y.Q. Bai, A class of polynomial interior-point algorithms for the cartesian p-matrix linear complementarity problem over symmetric cones, *J. Optim. Theory Appl.* 152 (2012) 739–772.

[28] A.Yoshise, Interior point trajectories and a homogeneous model for nonlinear complementarity problems over symmetric cones, *SIAM J. Optim.* 17 (2006) 1129–1153.

[29] A.Yoshise, Homogeneous algorithms for monotone complementarity problems over symmetric cones, *Pac. J. Optim.* 5 (2009) 313–337.

[30] H.C. Zhang and W.W. Hager, A nonmonotone line search technique and its application to unconstrained optimization, *SIAM J. Optim.* 14 (2004) 1043–1056.

[31] Y. Zhang and Z.H. Huang, A nonmonotone smoothing-type algorithm for solving a system of equalities and inequalities, *J. Comput. Appl. Math.* 233 (2010) 2312–2321.

[32] J.G. Zhu, H.W. Liu and C.H. Liu, A family of new smoothing functions and a nonmonotone smoothing Newton method for the nonlinear complementarity problems, *J. Appl. Math. Comput.* 37 (2011) 647–662.

Manuscript received 27 December 2015
revised 16 February 2017
accepted for publication 30 May 2017

Jingyong Tang
College of Mathematics and Information Science
Xinyang Normal University, Xinyang
464000, China
E-mail address: jingyongtang@163.com

Jinchuan Zhou
Department of Mathematics, School of Science
Shandong University of Technology
Zibo, 255049, China
E-mail address: jinchuanzhou@163.com

Liang Fang
College of Mathematics and System Science
Taishan University, Tai'an 271021, China
E-mail address: fangliang3@163.com