Sl_1QP Based Algorithm with Trust Region Technique for Solving Nonlinear Second-Order Cone Programming Problems

Kohei YASUDA

Feburuary 2013

Abstract In this paper, we propose an algorithm based on Fletcher's Sl_1QP method and the trust region technique for solving Nonlinear Second-Order Cone Programming (NSOCP) problems. The Sl_1QP method was originally developed for nonlinear optimization problems with inequality constraints. It converts a constrained optimization problem into an unconstrained problem by using the l_1 exact penalty function, and then finds an optimum by solving approximate quadratic programming subproblems successively. In order to apply the Sl_1QP method to the NSOCP problem, we introduce an exact penalty function with respect to second-order cone constraints and reformulate the NSOCP problem as an unconstrained optimization problem. However, since each subproblem generated by the Sl_1QP method is not differentiable, we reformulate it as a second-order cone programming problem whose objective function is quadratic and constraint functions are affine. We analyze the convergence property of the proposed algorithm, and show that the generated sequence converge to a stationary point of the NSOCP problem under mild assumptions. We also confirm the efficiency of the algorithm by means of numerical experiments.

Keywords Nonlinear second-order cone programming problem, Trust region method, Exact penalty function

Contents

1 Introduction

In this paper, we focus on the following Nonlinear Second-Order Cone Program (NSOCP):

$$\begin{array}{lll} \text{(NSOCP)} & \min\limits_{x} & f(x) \\ & \text{s.t.} & g(x) \in \mathcal{K}, \\ & & h(x) = 0, \end{array} \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^m$ and $h : \mathbb{R}^n \to \mathbb{R}^l$ are twice continuously differentiable functions, and $\mathcal{K}$ denotes the Cartesian product of several second-order cones (SOCs), i.e., $\mathcal{K} := \mathcal{K}^{m_1} \times \mathcal{K}^{m_2} \times \cdots \times \mathcal{K}^{m_s}$ with $m = m_1 + m_2 + \cdots + m_s$ and

$$\mathcal{K}^{m_i} := \begin{cases} \{x \in \mathbb{R} \mid x \geq 0\} & (m_i = 1), \\ \left\{x \in \mathbb{R}^{m_i} \mid x_1 \geq (x_2^2 + \cdots + x_{m_i}^2)^{1/2}\right\} & (m_i \geq 2). \end{cases}$$

In case of $m_1 = m_2 = \cdots = m_s = 1$, NSOCP (1.1) reduces to the standard nonlinear program (NLP) since we have $\mathcal{K} = \{x \in \mathbb{R}^n \mid x \geq 0\}$. This means that the NLP is a special case of NSOCP (1.1). On the other hand, NSOCP (1.1) is contained in the (nonlinear) semidefinite program (SDP), which is an optimization problem with a matrix variable and semidefinite constraints, since we have

$$x_0 \geq \|\bar{x}\| \iff \begin{pmatrix} x_0 & \bar{x}^T \\ \bar{x} & x_0 I \end{pmatrix} \succeq 0$$

for any $(x_0, \bar{x}) \in \mathbb{R} \times \mathbb{R}^{m_i - 1}$. However, it is not desirable to solve an NSOCP as an SDP since its variable is a matrix.

The NSOCP has a lot of applications such as the convex quadratic programming problem with quadratic constraints [1], finite impulse response filter design problems [8], etc. Among them, the robust optimization problem is one of important applications. Consider the following problem:

$$\begin{array}{ll} \min\limits_{x} & p(x) \\ \text{s.t.} & \inf\limits_{\omega \in W} \omega^T q(x) \geq 0, \end{array} \tag{1.2}$$

where $p : \mathbb{R}^n \to \mathbb{R}$ and $q : \mathbb{R}^n \to \mathbb{R}^m$ are given functions and W is the set defined by

$$W := \{\omega_0 + Qr \in \mathbb{R}^m \mid r \in \mathbb{R}^{m'}, \|r\| \leq 1\}$$

with a given vector $\omega_0 \in \mathbb{R}^m$ and a matrix $Q \in \mathbb{R}^{m \times m'}$. Noticing $\inf_{\omega \in W} \omega^T q(x) = \omega_0^T q(x) - \|Qq(x)\|$, problem (1.2) can be reformulated as

$$\begin{array}{ll} \min\limits_{x} & p(x) \\ \text{s.t.} & \omega_0^T q(x) - \|Qq(x)\| \geq 0, \end{array} \tag{1.3}$$

which can be cast as NSOCP (1.1) by letting

$$g(x) := \begin{pmatrix} \omega_0^T q(x) \\ Qq(x) \end{pmatrix}, \quad \mathcal{K} := \mathcal{K}^{m'+1}.$$

When the functions f, g and h are affine, NSOCP(1.1) is specially called a Linear Second-Order Cone Program (LSOCP):

$$\begin{array}{lll} \text{(LSOCP)} & \min\limits_{x} & c^T x \\ & \text{s.t.} & Ax + b = 0, \\ & & x \in \mathcal{K}, \end{array} \tag{1.4}$$

which has been studied extensively so far [1, 9, 10]. For solving LSOCP, the primal-dual interior point method is known to be effective, and many software packages have been developed [11, 12].

On the other hand, there have been only a limited number of studies for NSOCP. For example, Kanzow, Ferenczi and Fukushima [6] proposed an algorithm based on the nonsmooth Newton method for solving NSOCP. However, they only focused on the problem with a nonlinear objective function and affine SOC constraints. Yamashita and Yabe [13] proposed a primal-dual interior point algorithm by combining a barrier penalty function and a potential function, and showed that the algorithm possesses the global convergence property. Kato and Fukushima [7] proposed an algorithm based on the sequential quadratic programming (SQP) method, in which an approximate problem with quadratic objective function and affine constraints are solved at each iteration. However, their algorithm has the drawback such that the generated subproblems may not have a feasible solution.

In this paper, we propose an algorithm based on Fletcher's Sl_1QP method for solving NSOCP (1.1). In applying the method, we first introduce an l_1 exact penalty function to transform the NSOCP into an unconstrained optimization problem. Then, we solve it by using the search direction, which is obtained as the optimum of the following SOCP:

$$\begin{array}{ll} \min\limits_{(d,\gamma,\zeta)\in\mathbb{R}^n\times\mathbb{R}^s\times\mathbb{R}^l} & \nabla f(x^k)^T d + \frac{1}{2} d^T W_k d + \rho \left(\sum_{i=1}^{s} \gamma_i + \sum_{j=1}^{l} |\zeta_j| \right) \\ \text{s.t.} & (\Delta_k, d) \in \mathcal{K}^{n+1}, \\ & g^i(x^k) + \nabla g^i(x^k)^T d + \gamma_i e^i \in \mathcal{K}^{m_i} \ (i = 1, \cdots, s), \\ & h^j(x^k) + \nabla h^j(x^k)^T d + \zeta_j = 0 \ (j = 1, \cdots, l), \\ & \gamma_i \geq 0, \end{array} \tag{1.5}$$

where x^k is the k-th iterative point, W_k is an appropriate given matrix, and $e^i := (1, 0, \ldots, 0)^T \in \mathbb{R}^{m_i}$. The SOC constraint $(\Delta, d) \in \mathcal{K}^{n+1}$ in (1.5) implies that

the Euclidean trust region with radius Δ_k is imposed in each iteration. Moreover, we notice that problem (1.5) is always feasible and can be solved by some existing algorithm efficiently when W_k is positive semidefinite. Concerning the convergence property of our algorithm, we show that, if an accumulation point x^* of the generated sequence is feasible to NSOCP (1.1), it must be a stationary point of (1.1). Though there is no guarantee that the generated sequence converges to a feasible point theoretically, we can observe that the accumulation point is feasible in most cases by means of numerical experiments.

This paper is organized as follows. In section 2, we introduce an l_1 exact penalty function for NSOCP, and propose the Sl_1QP algorithm on which the trust region technique is imposed. In section 3, we analyze the convergence property of the algorithm, and show that the generated sequence converges to the KKT point of the NSOCP under mild assumptions. In section 4, we report some numerical results to show the efficiency of the algorithm. In section 5, we conclude the paper with some remarks.

The notation used in this paper is as follows. $\mathbb{S}^n$ denotes the set of $n \times n$ symmetric matrices. For a vector $x \in \mathbb{R}^n$ with $n \geq 2$, x_0 and $\bar{x}$ denote the first component and the subvector consisting of the remaining $n-1$ components, respectively, that is, $x = \binom{x_0}{\bar{x}}$. For any vectors x and y, the vector $(x^T, y^T)^T$ is often written as (x, y) for simplicity. For a vector x, $\|x\|$ denotes the Euclidean norm, i.e., $\|x\| = \sqrt{x^T x}$ and $\|x\|_1$ denotes the l_1 norm, i.e., $\|x\|_1 = \sum_{i=1}^n |x_i|$

2 Sl_1QP Algorithm for Nonlinear Second-Order Cone Programming Problem

The sequential l_1 Quadratic Programming (Sl_1QP) method was originally proposed by Fletcher [4], which is a method for solving nonlinear programs with inequality constraints. In this section, we extend the Sl_1QP algorithm to the NSOCP. To this end, we first transform NSOCP (1.1) into an unconstrained problem by using the l_1 exact penalty function. Then, we solve the unconstrained problem by generating a sequence, which is calculated by using optima of approximate quadratic subproblems with Euclidean trust region constraints.

2.1 l_1 Exact Penalty Function

In this subsection, we first study the l_1 penalty function for the general nonlinear program with inequality constraints and then extend it to NSOCP (1.1).

Consider the following nonlinear program:

$$\begin{array}{ll} \min\limits_{x} & f(x) \\ \text{s.t.} & q(x) \leq 0, \\ & r(x) = 0. \end{array} \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $q : \mathbb{R}^n \to \mathbb{R}^m$ and $r : \mathbb{R}^n \to \mathbb{R}^l$ are given twice continuously differentiable functions. Then, the l_1 exact penalty function $f_\rho : \mathbb{R}^n \to \mathbb{R}$ for problem (2.1) is defined as

$$f_\rho(x) := f(x) + \rho \left(\sum_{i=1}^{m} \max(0, q_i(x)) + \|r(x)\|_1 \right),$$

where $\rho > 0$ denotes the penalty parameter. Then, we can reformulate NLP (2.1) as the following unconstrained problem:

$$\min_{x} \quad f_\rho(x). \tag{2.2}$$

Let S and S_ρ be the sets of local minima of problem (2.1) and problem (2.2), respectively. Let $\mathcal{F}$ be the feasible set of (2.1). Then, for a sufficient large ρ, it is known that $S \supseteq S_\rho \cap \mathcal{F}$ [14, Proposition 4.1]. Hence, if we choose a sufficient large ρ, then we may obtain a local optimum of problem (2.1) by solving (2.2). However, we notice that S_ρ may contain an infeasible point of problem (2.1), and thus we may not find even a feasible solution if we apply some descent algorithm to problem (2.2). Moreover, since problem (2.2) is nondifferentiable, we have to introduce a certain technique for nondifferentiable optimization.

Now, we define the l_1 exact penalty function for NSOCP (1.1). Let the function $\phi : \mathbb{R}^{m_i} \to \mathbb{R}$ be defined as

$$\phi(z) := z_0 - \|\bar{z}\|, \tag{2.3}$$

where $z = (z_0, \bar{z}) \in \mathbb{R} \times \mathbb{R}^{m_i - 1}$. Since $z \in \mathcal{K}$ if and only if $\phi(z) \geq 0$, we can reformulate NSOCP (1.1) as follows:

$$\begin{array}{lll} \text{(NSOCP)} & \min\limits_{x} & f(x) \\ & \text{s.t.} & \phi(g^i(x)) \geq 0 \quad (i = 1, \ldots, s), \\ & & h(x) = 0, \end{array} \tag{2.4}$$

where $g^i(x) \in \mathbb{R}^{m_i}$ denotes the i-th sub vector of $g(x) \in \mathbb{R}^m$ corresponding to the Cartesian structure of $\mathcal{K}$, that is,

$$g(x) = \left(g^1(x), g^2(x), \ldots, g^s(x)\right) \in \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \times \cdots \times \mathbb{R}^{m_s}.$$

Thus, by using the l_1 exact penalty function, we can rewrite problem (2.4) as

$$\min_x \quad f_\rho(x), \tag{2.5}$$

where

$$f_\rho(x) := f(x) + \rho \left(\sum_{i=1}^{m} \max[0, -\phi(g^i(x))] + \|h(x)\|_1 \right). \tag{2.6}$$

2.2 Sl_1QP Method with Trust Region Technique

In this subsection, we provide an approach for solving the unconstrained problem (2.5) by incorporating a trust region technique into the Sl_1QP method. To this end, we introduce the following function:

$$\begin{aligned} q_k(d) := \quad & \nabla f(x^k)^T d + \frac{1}{2} d^T W_k d \\ & + \rho \left(\sum_{i=1}^{s} \max\left(0, -\phi\left(g^i(x^k) + \nabla g^i(x^k)^T d\right)\right) + \|h(x^k) + \nabla h(x^k)^T d\|_1 \right), \end{aligned} \tag{2.7}$$

where x_k and W_k denote the k-th iteration point, and an approximation of the Hessian of the Lagrangian of f, respectively. Notice that the above function is an approximation of f_ρ defined by (2.6) at x^k. Especially, the first two terms of $q_k(d)$ represent the second-order approximation of the objective function f, and the last term is the first-order approximation of the penalty term in $f_\rho(x)$. Adding a trust region constraint, we obtain the following problem:

$$\begin{aligned} \min_d \quad & q_k(d) \\ \text{s.t.} \quad & \|d\| \le \Delta_k. \end{aligned} \tag{2.8}$$

In our algorithm, we solve (2.8) to find the optimum d^k in the k-th iteration, and set the next iteration point by $x^{k+1} := x^k + d^k$. Since $q_k(d)$ involves the l_1 norm and the max function, it is not differentiable. However, we can reformulate problem (2.8) as the SOCP by using slack variables γ and ζ.

$$\begin{aligned} \min_{(d,\gamma,\zeta)\in\mathbb{R}^n\times\mathbb{R}^s\times\mathbb{R}^l} \quad & \nabla f(x^k)^T d + \frac{1}{2} d^T W_k d + \rho \left(\sum_{i=1}^{s} \gamma_i + \sum_{j=1}^{l} \zeta_l \right) \\ \text{s.t.} \quad & \|d\| \le \Delta_k, \\ & -\gamma_i \le \phi(g^i(x^k) + \nabla g^i(x^k)^T d) \quad (i = 1, \ldots, s), \\ & -\zeta \le h(x^k) + \nabla h(x^k)^T d \le \zeta, \\ & \gamma \ge 0, \end{aligned} \tag{2.9}$$

where $\gamma := (\gamma_1, \ldots, \gamma_s) \in \mathbb{R}^s$ and $\zeta := (\zeta_1, \ldots, \zeta_l) \in \mathbb{R}^l$. Let $e^i := (1, 0, \ldots, 0)^T \in \mathbb{R}^{m_i}$ $(i = 1, \ldots, s)$. Note that $\phi(x) \geq -\gamma$ if and only if $x + (\gamma_1 e^1, \ldots, \gamma_s e^s) \in \mathcal{K}$, we can rewrite (2.9) as

$$\begin{array}{ll} \min\limits_{(d,\gamma,\zeta)\in\mathbb{R}^n\times\mathbb{R}^s\times\mathbb{R}^l} & \nabla f(x^k)^T d + \dfrac{1}{2} d^T W_k d + \rho \left(\displaystyle\sum_{i=1}^{s} \gamma_i + \sum_{j=1}^{l} \zeta_l \right) \\ \text{s.t.} & (\Delta_k, d) \in \mathcal{K}^{n+1}, \\ & g^i(x^k) + \nabla g^i(x^k)^T d + \gamma_i e^i \in \mathcal{K}^{m_j} \ (i = 1, \cdots, s), \\ & -\zeta \leq h(x^k) + \nabla h(x^k)^T d \leq \zeta, \\ & \gamma \geq 0. \end{array} \tag{2.10}$$

We can solve (2.10) by using an existing method such as [5, 12].

2.3 Algorithm

Summarizing the above arguments, we state the algorithm.

Algorithm 1

Step 0: Choose $x_0 \in \mathbb{R}^n$, $\rho > 0$, $W_0 \in \mathbb{S}^n$, $0 < \eta_1 < \eta_2 < 1$, $0 < \gamma_1 < 1 < \gamma_2$, and $\Delta_0 > 0$. Set $k = 0$.

Step 1: Solve subproblem (2.10) to obtain the optimum (d^k, γ^k, ζ^k). If a certain stopping criterion is satisfied, then STOP. Otherwise, go to Step 2.

Step 2: Let

$$r_k := \frac{f_\rho(x^k) - f_\rho(x^k + d^k)}{q_k(0) - q_k(d^k)}.$$

Step 3: Calculate x^{k+1} and Δ_{k+1} as follows:

(i) if $r_k \leq \eta_1$, then set $x^{k+1} := x^k$ and $\Delta_{k+1} := \gamma_1 \Delta_k$;

(ii) if $\eta_1 < r_k < \eta_2$, then set $x^{k+1} := x^k + d_k$ and $\Delta_{k+1} := \Delta_k$;

(iii) if $r_k \geq \eta_2$, then set $x^{k+1} := x^k + d^k$ and $\Delta_{k+1} := \gamma_2 \Delta_k$.

Step 4: Calculate W_{k+1} in an appropriate manner. Set $k := k+1$ and go to Step 1.

3 Global Convergence

Dennis, Li and Tapia [2] analyzed the conditions under which the trust region method for a nonsmooth unconstrained optimization problem has the global convergence. Utilizing their results, we prove the global convergence of Algorithm 1.

3.1 Directional Derivative and Regularity

We first introduce some concepts on the directional derivative and the regularity, which play an important role in showing the global convergence.

Definition 1. *A function $F : \mathbb{R}^n \to \mathbb{R}$ is said to be locally Lipschitz around x if there exist some constants $K > 0$ and $\epsilon > 0$ such that*

$$|F(x_1) - F(x_2)| \leq K\|x_1 - x_2\|$$

for any $(x_1, x_2) \in N_\epsilon(x) \times N_\epsilon(x)$, where $N_\epsilon(x) := \{y \mid \|y - x\| < \epsilon\}$. Moreover, the function F is said to be locally Lipschitz on a set U if it is locally Lipschitz around any point $x \in U$.

Definition 2. *Let $F : \mathbb{R}^n \to \mathbb{R}$ be a given function. Then,*

(a) *the one-sided directional derivative of F at x along the direction $d \in \mathbb{R}^n$ is defined as*

$$F'(x; d) := \lim_{t \downarrow 0} \frac{F(x + td) - F(x)}{t};$$

(b) *the generalized directional derivative of F at x along the direction $d \in \mathbb{R}^n$ is defined as*

$$F^\circ(x; d) := \limsup_{\substack{y \to x \\ t \downarrow 0}} \frac{F(y + td) - F(y)}{t}.$$

By using the above two types of directional derivatives, we define the regularity for a scalar-valued function.

Definition 3. *A function $F : \mathbb{R}^n \to \mathbb{R}$ that is locally Lipschitz around x is said to be regular at x if $F'(x; d)$ exists for all directions $d \in \mathbb{R}^n$ and*

$$F'(x; d) = F^\circ(x; d).$$

Moreover, the function F is said to be regular on a set U if it is regular at every point of the set U.

The following two propositions are useful in showing the global convergence of our algorithm.

Proposition 1. *[2, Theorem 2.8] Let U be an open convex set in $\mathbb{R}^n$ and let F be a convex function on U. Then F is Lipschitz around any point x in U and $F^\circ(x,d)$ coincides with $f'(x;d)$ for all d in $\mathbb{R}^n$. Thus F is regular on U.*

Proposition 2. *[2, Corollary 2.9] Let U be an open convex set in $\mathbb{R}^n$ and let H be a convex function on U. Let $G : V \subset \mathbb{R}^m \to U \subset \mathbb{R}^n$ be continuously differentiable on V. Then the composite function $C(x) = H(G(x))$ is Lipschitz around any point x in V and $C^\circ(x;d)$ coincides with $C'(x;d)$ for all d in $\mathbb{R}^n$. Thus composite function C is regular on V.*

3.2 Convergence Result

Next we show the global convergence of Algorithm 1. Dennis et al. [2] showed that a certain trust region algorithm is globally convergent under some assumptions. Applying their result directly to Algorithm 1, we can readily obtain the convergence result as follows.

Assumption 1. *Let $m(x,W)(d)$ be defined by*

$$\begin{aligned} m(x,W)(d) := \quad & f(x) + \nabla f(x)^T d + \frac{1}{2} d^T W d \\ & + \rho\Big(\sum_{i=1}^{s} \max(0, -\phi\left(\big(g_i(x) + \nabla g_i(x)^T d\big)\right) + \|h(x) + \nabla h(x)^T d\|_1 \Big). \end{aligned}$$

Let $X_0 := \{x \mid f_\rho(x) \le f_\rho(x_0)\}$, where f_ρ is defined by (2.6) and x^0 is the initial point chosen in Step 0 of Algorithm 1. Let $P \subset \mathbb{S}^n$ be a certain bounded set satisfying $\{W_k\} \subset P$. The following five conditions hold.

(a) *P is bounded.*

(b) *The function f_ρ is regular on X_0.*

(c) *For any $d \in \mathbb{R}^n$, the local model $m(x,W)(d)$ is continuous with respect to (x,W).*

(d) *For any $(x,W) \in X_0 \times P$, the local model $m(x,W)(d)$ is regular with respect to $d \in \mathbb{R}^n$.*

(e) *For any $(x,W) \in X_0 \times P$, the local model $m(x,W)(d)$ satisfies*

$$m(x,W)(0) = f_\rho(x)$$

and

$$m(x,W)^\circ(0;d) = f_\rho^\circ(x;d).$$

Theorem 1. *[2, Theorem 4.3] Suppose that Assumption 1 holds. Then, any accumulation point of $\{x^k\}$ generated by Algorithm 1 is a stationary point of f_ρ.*

However, it is not evident whether or not Assumption 1 holds, and hence it is important to derive the condition under which Assumption 1 holds. In what follows, we show that all conditions (a)–(e) in Assumption 1 automatically hold, provided that the generated sequences $\{x^k\}$ and $\{W_k\}$ are bounded.

In fact, it is easily seen that Assumptions 1(a)–(d) hold.

Theorem 2. *Suppose that the sequence $\{x^k\}$ and $\{W_k\}$ generated by Algorithm 1 are bounded. Then, Assumptions 1(a)–(d) hold.*

Proof. Since we readily have (a) and (c) from the boundedness of $\{W_k\}$ and the definition of $m(x, W)(d)$, respectively, we only show (b) and (d).

We first show (b). Let

$$G(x) := \begin{pmatrix} f(x) \\ g^1(x) \\ \vdots \\ g^m(x) \\ h(x) \end{pmatrix}, \tag{3.1}$$

$$H(z_0, z_1, \ldots, z_s, z_{s+1}) := z_0 + \rho \sum_{i=1}^{s} \max(0, -\phi(z_i)) + \rho \|z_{s+1}\|_1, \tag{3.2}$$

where $z_0 \in \mathbb{R}$, $z_1 \in \mathbb{R}^{m_1}, \ldots, z_s \in \mathbb{R}^{m_s}$, and $z_{s+1} \in \mathbb{R}^l$. Then $f_\rho(x)$ is represented as $f_\rho(x) = H(G(x))$. Since H is convex and G is continuously differentiable, $f_\rho(x)$ is regular on $\mathbb{R}^n$ by Proposition 2. We thus have (b).

We next show (d). Notice that the local model function $m(x, W)(d) = H(G(x) + \nabla G(x)^T d) + \frac{1}{2} d^T W d$ is convex with respect to d for any fixed (x, W). Hence, by Proposition 1, $m(x, W)(d)$ is regular with respect to d, i.e., (d) holds. □

In order to show that Assumption 1(e) also holds, we define the function θ as follows:

$$\theta(x, W)(d) := \frac{F(x + d) - m(x, W)(d)}{\|d\|}. \tag{3.3}$$

When functions f_ρ and $m(x, W)(\cdot)$ are regular, i.e., Assumptions 1(b) and (d) hold, we can see whether Assumption 1(e) holds or not by checking the limit of $\theta(x, W)(d)$ as $\|d\|$ tends to zero.

Lemma 1. *[2, Lemma 2.13] Suppose that Assumptions 1(b) and (d) hold. Then, for any $(x, W) \in X_0 \times \mathbb{S}^n$, the following three conditions are equivalent:*

(i) $m(x, W)(0) = f_\rho(x)$ *and* $m(x, W)^\circ(0; d) = f_\rho^\circ(x; d)$ *for all* $d \neq 0$ *in* $\mathbb{R}^n$;

(ii) $\lim_{\|d\| \to 0} \theta(x, W)(d) = 0$;

(iii) $\lim_{t \downarrow 0} \theta(x, W)(td) = 0$ *for all* $d \neq 0$ *in* $\mathbb{R}^n$.

By using this lemma, we finally prove the validity of Assumption 1(e).

Theorem 3. *Suppose that the sequences $\{x^k\}$ and $\{W_k\}$ generated by Algorithm 1 are bounded. Then, Assumption 1(e) holds.*

Proof. Since we can easily see that $m(x, W)(0) = f_\rho(x)$, we only show $m(x, W)^\circ(0; d) = f_\rho^\circ(x; d)$. Let functions G, H and θ be defined by (3.1), (3.2) and (3.3), respectively. From the Fréchet differentiability of G, we have

$$\lim_{\|d\| \to 0} \|\sigma(x, d)\| = 0$$

for any $x \in \mathbb{R}^n$, where

$$\sigma(x, d) := \frac{G(x + d) - G(x) - \nabla G(x)^T d}{\|d\|}.$$

Now, notice that

$$\begin{aligned} f_\rho(x + d) &= H(G(x + d)) \\ &= H(G(x) + \nabla G(x)^T d + \sigma(x, d)\|d\|) \end{aligned} \tag{3.4}$$

and

$$m(x, W)(d) = -\frac{1}{2} d^T W d + H(G(x) + \nabla G(x)^T d), \tag{3.5}$$

since

$$G(x) + \nabla G(x)^T d = \begin{pmatrix} f(x) + \nabla f(x)^T d \\ g^1(x) + \nabla g^1(x)^T d \\ \vdots \\ g^s(x) + \nabla g^s(x)^T d \\ h(x) + \nabla h(x)^T d \end{pmatrix}.$$

We thus have

$$\begin{aligned}\|d\||\theta(x,W)(d)| &= |f_\rho(x+d) - m(x,W)(d)| \\ &\le |H(G(x)+\nabla G(x)^T d + \sigma(x,d)\|d\|) - H(G(x)+\nabla G(x)^T d)| + \frac{1}{2}\|W\|\|d\|^2 \\ &\le K\|\sigma(x,d)\|\|d\| + \frac{1}{2}\|W\|\|d\|^2,\end{aligned}$$

where the first inequality is due to (3.4), (3.5) and the triangle inequality, and the last inequality follows since Proposition 1 together with the convexity of H implies that H is Lipschitz around $G(x) + G'(x)d$ with some constant K. Therefore

$$\lim_{\|d\|\to 0} \theta(x,W)(d) = 0$$

for every $(x, W) \in \mathbb{R}^n \times \mathbb{S}^n$. By Lemma 1, Assumption 1(e) holds. □

By Theorems 1–3, we readily have the following result on the global convergence of Algorithm 1.

Corollary 1. *Suppose that the sequences $\{x^k\}$ and $\{W_k\}$ generated by Algorithm 1 are bounded. Then, any accumulation point of $\{x^k\}$ is a stationary point of f_ρ.*

4 Numerical Experiments

In this section, we report some numerical results. We implement Algorithm 1 on MATLAB (Version R2012b) in which we apply the the solver [5] for the Second-Order Cone Complementarity Problem to each subproblem (2.10). In Step 0 of Algorithm 1, we set the parameters as $\rho = 10$, $W_0 = I_n$, $\eta_1 = 0.5$, $\eta_2 = 0.8$, $\gamma_1 = 0.5$, $\gamma_2 = 1.1$ and $\Delta_0 = 1.0$, where $I_n \in \mathbb{S}^n$ denotes the identity matrix. For the stopping criterion, we use $\|d^k\| < 10^{-6}$.

We update W_k by using the following modified BFGS formula:

Modified BFGS formula

$$W_{k+1} := W_k - \frac{W_k s^k (s^k)^T W_k}{(s_k)^T W_k s^k} + \frac{w^k (w^k)^T}{(s^k)^T w^k},$$

where

$$
\begin{aligned}
s^k &:= x^{k+1} - x^k, \\
y^k &:= \nabla L(x^{k+1}, \lambda^{k+1}, \mu^{k+1}) - \nabla L(x^k, \lambda^{k+1}, \mu^{k+1}), \\
w^k &:= \theta_k y^k + (1 - \theta_k) W_k y^k, \\
\theta_k &:= \begin{cases} 1 & \text{if } (s^k)^T y^k \geq 0.2 (s^k)^T W_k s^k \\ \dfrac{0.8 (s^k)^T M_k s^k}{(s^k)^T (M_k s^k - y^k)} & \text{otherwise} \end{cases},
\end{aligned}
$$

with the Lagrange function L of f_ρ and the corresponding Lagrange multipliers λ^{k+1} and μ^{k+1} for the SOC constraints and the equality constraints, respectively. By using this formula, we can generate positive definite matrices successively if the initial matrix W_0 is positive definite.

In this experiment, we solve the following NSOCP.

$$
\begin{aligned}
\min_x \quad & \exp(x_1 - x_2) + (x_1 - x_5)^4 + \frac{1}{2}\|x\|^2 - \sum_{i=1}^{n} x_i \\
\text{s.t.} \quad & \begin{pmatrix} x^T M_i x + c_i^T x + m_i \\ A_i x - b_i \end{pmatrix} \in \mathcal{K}^{m_i} \quad (i = 1, \ldots, s)
\end{aligned}
\tag{4.1}
$$

where $M_i \in \mathbb{S}^n$, $A_i \in \mathbb{R}^{(m_i - 1) \times n}$, $c_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}^{m_i - 1}$ are given matrices and vectors. Each component of M_i, A_i, c_i and b_i are randomly chosen from the interval $[-1, 1]$. Note that the objective function is convex, but the feasible set is nonconvex. Moreover, this problem always has a feasible point $x = 0$.

We choose 9 different types of Cartesian structure $\mathcal{K}$, and randomly generate 50 problem instances for each $\mathcal{K}$, i.e., 450 instances in total. To each problem, we apply Algorithm 1 with an initial point $x^0 \in \mathbb{R}^n$ whose components are randomly chosen from the interval $[-1, 1]$. When the number of iterations exceeds 500, then we stop the iteration and regard it as a failure.

The obtained results are summarized in Table 1, in which $k_{\min}$, $k_{\max}$ and k_{ave} denote the minimum, the maximum, and the average value of the number of times that subproblem (2.10) was solved, respectively. Moreover, "infeasible" means the number of times that the sequence $\{x^k\}$ generated by Algorithm 1 converges to an infeasible point of (4.1), and "fail" means the number of times that the algorithm does not terminate within 500 iterations. In the column of $\mathcal{K}$, $(\mathcal{K}^5)^2 \times (\mathcal{K}^{10})^2$ denotes $\mathcal{K}^5 \times \mathcal{K}^5 \times \mathcal{K}^{10} \times \mathcal{K}^{10}$ for example.

From Table 1, we can see that Algorithm 1 terminates successfully in most cases. Moreover, we can observe that, when the dimension m of $\mathcal{K}$ is fixed, the number of iteration tends to become larger as the number of SOCs in $\mathcal{K}$ increases. On the other hand, it is not affected by the dimension n of the decision variable

n	m	$\mathcal{K}$	$k_{\min}$	$k_{\max}$	k_{ave}	infeasible	fail
10	10	$(\mathcal{K}^5)^2$	18	187	37.74	0	0
20	15	$(\mathcal{K}^5)^3$	21	271	56.43	0	1
20	20	$(\mathcal{K}^5)^4$	21	273	83.30	0	0
20	20	$(\mathcal{K}^{10})^2$	20	389	49.06	0	0
40	30	$(\mathcal{K}^5)^2 \times (\mathcal{K}^{10})^2$	24	356	67.36	0	0
40	40	$(\mathcal{K}^5)^8$	27	473	113.78	0	1
40	40	$(\mathcal{K}^5)^4 \times (\mathcal{K}^{10})^2$	24	372	88.30	0	0
40	40	$(\mathcal{K}^{10})^4$	24	452	63.70	0	0
40	40	$(\mathcal{K}^{20})^2$	25	212	38.78	0	0

Table 1: Numerical results for (4.1) with various choices of $\mathcal{K}$

x so much. We also note that the generated sequence never converges to an infeasible point of (4.1), though it may happen theoretically. This implies that our algorithm finds the stationary point of NSOCP (4.1) in most cases.

5 Concluding Remarks

In this paper, we proposed an algorithm based on Fletcher's Sl_1QP method for solving nonlinear second-order cone programming problems and show that the algorithm has the global convergence property. We also confirm the efficiency of the algorithm by means of numerical experiments. However, our algorithm still admit much improvement. For example, it is an important issue to consider how to update matrix W_k and determine the penalty parameter ρ. In addition, the idea of second order correction proposed by Fletcher [3] may improve our algorithm.

References

[1] F. Alizadeh, D. Goldfarb, *Second-order cone programming*, Mathematical Programming Ser. B, Vol. 95, pp. 3–51 (2003).

[2] J. E. Dennis, S. B. B. Li, R. A. Tapia, *A unified approach to global convergence of trust region methods for nonsmooth optimization*, Mathematical Programming, Vol. 68, pp. 319–346 (1995).

[3] R. Fletcher, *Second order corrections for non-differentiable optimization*, Lecture Notes in Mathematics, Vol. 912, pp. 85–114 (1982).

[4] R. Fletcher, Practical Methods of Optimization, 2nd Edition, John Wiley & Sons, 1987.

[5] S. Hayashi, N. Yamashita, M. Fukushima, *A combined smoothing and regularization method for monotone second-order cone complementarity problems*, SIAM Journal on Optimization, Vol. 15, pp. 593–615 (2005).

[6] C. Kanzow, I. Ferenczi, M. Fukushima, *On the local convergence of semismooth Newton methods for linear and nonlinear second-order cone programs without strict complementarity*, SIAM Journal on Optimization, Vol. 20, pp. 297–320 (2009).

[7] H. Kato, M. Fukushima, *An SQP-type algorithm for nonlinear second-order cone programs*, Optimization Letters, Vol. 1, pp. 129–144, (2007).

[8] M. S. Lobo, L. Vandenberghe, S. Boyd, H. Lebret, *Applications of second-order cone programming*, Linear Algebra and Its Applications, Vol. 284, pp. 193–228, (1998).

[9] R. D. C. Monteiro, T. Tsuchiya, *Polynomial convergence of primal-dual algorithms for the second-order cone program based on the MZ-family of directions*, Mathematical Programming Ser. A Vol. 88, pp. 61–83 (2000).

[10] M. Muramatsu, *A pivoting procedure for a class of second-order cone programming*, Optimization Methods and Software, Vol. 21, pp. 295–315 (2005).

[11] J. F. Sturm, *Using SeDuMi 1.02, A Matlab toolbox for optimization over symmetric cones*, Optimization Methods and Software, Vol. 11, pp. 625–653, (1999).

[12] K. C. Toh, R. H. Tütüncü, and M. J. Todd. *SDPT3 version 4.0 – a matlab software for semidefinite-quadratic-linear programming*, http://www.math.nus.edu.sg/~mattohkc/sdpt3.html, 2006.

[13] H. Yamashita, H. Yabe, *A primal-dual interior point method for nonlinear optimization over second-order cones*, Optimization Methods and Software, Vol. 24, pp. 407–426 (2009).

[14] Y. Wang, L. Zhang, *Properties of equation reformulation of the Karush-Kuhn-Tucker condition for nonlinear second order cone optimization problems*, Mathematical Methods of Operations Research, Vol. 70, pp. 195–218 (2009).