SIAM J. OPTIM.
Vol. 18, No. 1, pp. 65–86

AN INTERIOR-POINT TRUST-REGION ALGORITHM FOR GENERAL SYMMETRIC CONE PROGRAMMING*

YE LU† AND YA-XIANG YUAN‡

Abstract. An interior-point trust-region algorithm is proposed for minimizing a general (nonconvex) quadratic objective function in the intersection of a symmetric cone and an affine subspace. The algorithm uses a trust-region model to ensure descent on a suitable merit function. Global first-order and second-order convergence results are proved. Numerical results are presented.

Key words. interior-point algorithm, trust-region subproblem, symmetric cone

AMS subject classifications. 90C30, 90C51

DOI. 10.1137/040611756

1. Introduction. In the last two decades, interior-point algorithms for convex programming have been developed quite well in both theory and practice. However, research on interior-point algorithms for nonconvex programming is still very active, as nonconvex problems are considerably more difficult. We mention here some of the recent works. For semidefinite relaxations, see Zhang [33] and Ye and Zhang [31]; for line search based algorithms, see Absil and Tits [1] and Bakry et al. [3], Forsgren and Gill [12], Gay, Overton, and Wright [13], Tits et al. [25], Vanderbei and Shanno [27], and Wächter [28]. By contrast, for trust-region-type interior-point algorithms, Ye [29] developed an affine scaling algorithm for indefinite quadratic programming by solving sequential trust-region subproblems. Global first-order and second-order convergence results were proved, and later enhanced by Sun [24] for the convex case. The idea of affine scaling can be traced back to Dikin [8]. An affine-scaling potential-reduction interior-point trust-region algorithm was developed for the indefinite quadratic programming in Ye [30, section 9]. Recently, in Faybusovich and Lu [11], Ye's algorithm has been extended to the minimization of a quadratic function in the intersection of a symmetric cone and an affine subspace. In this paper, we call such a problem symmetric cone programming and develop an affine-scaling primal barrier interior-point trust-region algorithm to solve it. Since the class of symmetric cones contains the positive orthant in R^n, the second-order cone, and the cone of positive semidefinite symmetric matrices, our approach solves a large class of optimization problems. In the trust-region literature, we refer the reader to Conn, Gould, and Toint [6, section 13] for a primal barrier algorithm and Conn et al. [7] for a primal-dual algorithm. Under the theoretical framework of their work, we bring the properties of ϑ-*normal barrier* and symmetric cone into our analysis. By doing so, we show that the primal barrier algorithm developed in Conn, Gould, and Toint [6] can be extended to solve symmetric cone programming. Although our algorithm still provides the mechanism to declare the iteration unsuccessful if feasibility is not achieved, it does not contain an explicit

*Received by the editors July 16, 2004; accepted for publication (in revised form) August 24, 2006; published electronically February 2, 2007.
http://www.siam.org/journals/siopt/18-1/61175.html

†Operations Research Center, Massachusetts Institute of Technology, 77 Mass Ave., Bldg. E40-130, Cambridge, MA 02139 (yelu@mit.edu). This work was done while the author was a graduate student at the University of Notre Dame Mathematics Department.

‡LSEC, ICMSEC, AMSS, Chinese Academy of Sciences, Beijing 100080, China (yyx@lsec.cc.ac.cn). This author's work was partially supported by NSFC grant 10231060.

constraint on the step calculation (see constraint (13.2.1) in Conn, Gould, and Toint [6, p. 499] and the constraint (13.3.1) of the algorithm in Conn, Gould, and Toint [6, p. 505]). This makes our algorithm theoretically somewhat simpler, although its practical merit remains to be investigated. Moreover, we establish inequality (4.22) in section 4 of this paper to explicitly estimate the convergence of the algorithm. This is quite remarkable, since our analysis does not require any convexity assumptions.

This paper is organized as follows. In section 2, we present some concepts and results of the symmetric cone and *ϑ-normal barrier* in the theory of interior-point methods. In section 3, we formulate the first-order and second-order optimality conditions for our optimization problem. In section 4, we present a convergence analysis for our interior-point trust-region algorithm. The techniques of proofs in Lemmas 4.3–4.6 essentially follow from Conn, Gould, and Toint [6, section 13], together with the applications of the properties of the *ϑ-normal barrier*. In section 5, our algorithm is used to solve the large-scale trust-region subproblem. In section 6, we apply our algorithm to a class of quadratic programs and discuss some further implementation issues. Concluding remarks and recommendations are presented in section 7.

2. Symmetric cone and ϑ-normal barrier. In this section we introduce some concepts and relevant results which will be used in the following sections.

Nesterov and Nemirovskii [18] developed the concept of *ϑ-normal barrier*, which has become one of the most important tools for the analysis of interior-point methods. It is also an essential tool in our analysis. We assume that K is a convex cone in a finite-dimensional real vector space E. Let K° be the interior of K. The definition of *ϑ-normal barrier* is given as follows.

DEFINITION 2.1. *Let $F : K^\circ \to R$ be a C^3-smooth strictly convex function such that F is a barrier for K (i.e., $F(x) \to \infty$ as $x \in K^\circ$ approaches the boundary of K), and there exists $\vartheta \geq 1$ such that for each $t > 0$,*

$$F(tx) = F(x) - \vartheta \ln(t) \tag{2.1}$$

and

$$| F^{'''}(x)[h,h,h] | \leq 2 \langle F^{''}(x)h, h\rangle^{3/2} \tag{2.2}$$

for all $x \in K^\circ$ and for all $h \in E$. Then F is called a ϑ-normal barrier for K and ϑ is called barrier parameter of F.

In principle, every convex cone admits a *ϑ-normal barrier* (see Nesterov and Nemirovskii [18, section 4]). But, in this paper we consider only a special kind of convex cone, called a symmetric cone. As a regular convex cone K in a finite-dimensional real vector space E endowed with an inner product $\langle\ \rangle$, the dual of K is defined as

$$K^* = \{y \in E \mid \langle x, y\rangle \geq 0\ \forall x \in E\}.$$

We define $Aut(K)$ to be the set of automorphisms of the convex cone K, that is, $AK = K$ for any $A \in Aut(K)$. The following is the definition of symmetric cone.

DEFINITION 2.2. *A convex cone K is called homogeneous if $Aut(K)$ is transitive on K°; that is, given any pair of points $x, s \in K^\circ$ there exists $A \in Aut(K)$ such that $Ax = s$. The cone K is said to be self-dual if there is an inner product such that $K^* = K$. K is said to be symmetric if it is homogeneous and self-dual.*

The following important cones are special symmetric cones.

The positive orthant. The simplest symmetric cone is the positive orthant

$$R^n_{++} = \{x \mid x > 0, x \in R^n\} = R_{++} \oplus \cdots \oplus R_{++}, \tag{2.3}$$

which is the direct sum of n copies of R_{++}. $F(x) = -\sum_{i=1}^{n} \ln x_i$ is a *ϑ-normal barrier* for R^n_{++} with $\vartheta = n$.

The second-order cone. This is the cone defined by

$$SOC := \left\{ x \in R^n : \sum_{i=1}^{n-1} x_i^2 \le x_n^2 \text{ and } x_n \ge 0 \right\}. \tag{2.4}$$

The function $F(x) = -\ln(x_n^2 - \sum_{i=1}^{n-1} x_i^2)$ is a *ϑ-normal barrier* for the second-order cone SOC with $\vartheta = 2$.

The cone of positive semidefinite matrices. This is the cone of all positive semidefinite matrices

$$S_+^{n\times n} = \{X \mid X \in R^{n\times n}, \quad X \text{ positive semidefinite}\}. \tag{2.5}$$

$F(X) = -\ln\det(X)$ is a *ϑ-normal barrier* for $S_+^{n\times n}$ with $\vartheta = n$.

Let $F''(x)$ denote the Hessian of *ϑ-normal barrier* $F(x)$. The strictly convex assumption of $F(x)$ implies that $F''(x)$ is positive definite for every $x \in K^\circ$. Thus, $\|v\|_x = \langle v, F''(x)v\rangle^{\frac{1}{2}}$ is a norm on E induced by $F''(x)$. Let $B_x(y, r)$ denote the open ball of radius r centered at y, where the radius is measured with respect to $\| \ \|_x$. This ball is called the Dikin ball. The following lemmas are very crucial for the analysis of our algorithm in the next sections.

LEMMA 2.1. *Assume $F(x)$ is a ϑ-normal barrier for K; then for all $x \in K^\circ$ we have $B_x(x, 1) \subseteq K^\circ$.*

LEMMA 2.2. *Assume $F(x)$ is a ϑ-normal barrier for K, $x \in K^\circ$, and $y \in B_x(x, 1)$; then*

$$\left| F(y) - F(x) - \langle F'(x), y - x\rangle - \frac{\langle y - x, F''(x)(y - x)\rangle}{2} \right| \le \frac{\|y - x\|_x^3}{3(1 - \|y - x\|_x)}. \tag{2.6}$$

LEMMA 2.3. *Let F be a ϑ-normal barrier for K; then*

$$F''(x)^{-1}F'(x) = -x, \tag{2.7}$$

$$\langle -F'(x), x\rangle = \vartheta. \tag{2.8}$$

LEMMA 2.4. *If K is a symmetric cone and F is a ϑ-normal barrier for K, then $F''(x)$ is a linear automorphism of K for each $x \in K^\circ$.*

The proofs of the above lemmas can be found in Chapter 2 of Renegar [20].

Lemma 2.1 tells us the ball of radius 1 measured by $\|.\|_x$ is always contained in K°. Lemma 2.2 shows that at least locally, the quadratic approximation is very good for the *ϑ-normal barrier* F. Lemma 2.3 plays an important role in our proof of Lemma 4.1 in section 4. Lemma 2.4 is a special property of the symmetric cone, which is one of the reasons why in this paper we focus on the symmetric cones instead of general cones.

3. Optimality conditions. In this section, we formulate the first-order and second-order optimality conditions of our optimization problem:

We consider the following optimization problem:

$$\min \quad q(x) = \frac{1}{2}\langle x, Qx\rangle + \langle c, x\rangle \tag{3.1}$$

$$\text{subject to} \quad Ax = b, \tag{3.2}$$

$$x \in K. \tag{3.3}$$

Here $Q : E \mapsto E$ is a symmetric linear operator, $c \in E$. $A : E \mapsto R^m$ is a linear operator and $b \in R^m$. K is a symmetric cone. We assume that our feasible set $\digamma_p = \{x \in E | Ax = b, x \in K\}$ is bounded and has relative interior. The following theorem is the first-order optimality condition for our optimization problem. For a proof, see, e.g., Bonnans and Shapiro [5] or Faybusovich and Lu [11].

THEOREM 3.1 (first-order optimality condition). *If x^* is a locally minimal solution of* (3.1)–(3.3), *then there exists $s \in K^*(= K)$ such that $Qx^* + c - s \in R(A^*)$ and $\langle x^*, s\rangle = 0$; here $R(A^*)$ is the range of A^* and $A^* : R^m \mapsto E$ is the adjoint of A.*

Assume x is a point in our feasible set $\digamma_p = \{x \in E | Ax = b, x \in K\}$; there must be a unique face $\digamma_x$ of $\digamma_p$ such that x is a relative interior point of $\digamma_x$. We denote $Aff(\digamma_x)$ to be the affine space generated by $\digamma_x$ and V_x to be the vector space such that $Aff(\digamma_x) = V_x + x$. Now we are ready to formulate the second-order optimality condition.

THEOREM 3.2 (second-order optimality condition). *If x^* is a locally minimal solution of* (3.1)–(3.3), *$\digamma_{x^*}$ is the unique face of the feasible set $\digamma_p$ such that x^* is one of its relative interior points, and $V_{x^*} = Aff(\digamma_{x^*}) - x^*$, then Q is positive semidefinite over V_{x^*}.*

Proof. For all $d \in V_{x^*}$, because x^* is a relative interior of $\digamma_{x^*}$, we know $x^* + td \in \digamma_{x^*}$, provided that $|t|$ is sufficiently small. Hence, there exists a $\epsilon > 0$ such that

$$q(x^* + td) - q(x^*) = t\langle Qx^* + c, d\rangle + \frac{t^2}{2}\langle d, Qd\rangle \geq 0 \tag{3.4}$$

as long as $|t| \leq \epsilon$, due to the fact that x^* is a local minimim. The above inequality implies that

$$\langle Qx^* + c, d\rangle = 0, \qquad \langle d, Qd\rangle \geq 0. \tag{3.5}$$

This completes our proof. ∎

If $x \in K^\circ$, it is obvious that $V_x = \{x \in E | Ax = 0\}$. If $x \in \partial K$, the matter becomes much more complicated. But fortunately, we can get some very helpful results in the case of symmetric cones.

If $K = R^n_{++}$ and $x' \in \partial K$, it can be shown that $V_{x'} = \{x \in R^n | Ax = 0, x_j = 0, j \in I\}$, where $I = \{j | x'_j = 0\}$. We know that $F(x) = -\sum_{i=1}^n \ln x_i$ is a *ϑ-normal barrier* for R^n_{++}. Therefore, $F''(x')^{-\frac{1}{2}} = \mathrm{diag}\{x'_1, x'_2, \ldots, x'_n\}$, and consequently $V_{x'} = \{F''(x')^{-\frac{1}{2}} x | AF''(x')^{-\frac{1}{2}} x = 0, , x \in R^n\}$.

If $K = S^{n\times n}_+$, we know $F(X) = -\ln\det(X)$ is a *ϑ-normal barrier* for $S^{n\times n}_+$. Now let $A' \in \partial K$, and $\mathrm{rank}(A') = r < n$. Then $F''(A')^{-\frac{1}{2}} X = A'^{\frac{1}{2}} X A'^{\frac{1}{2}}$. We set $V = \{A'^{\frac{1}{2}} X A'^{\frac{1}{2}} | AA'^{\frac{1}{2}} X A'^{\frac{1}{2}} = 0, X \in S^{n\times n}\}$; just as with the positive orthant case, it holds that $V_{A'} = V$. The following theorem tells us that this property actually holds for all symmetric cones. We will prove this theorem in the appendix.

THEOREM 3.3. *Assume K is a symmetric cone in a finite-dimensional real Euclidean space E and $F(x)$ is the ϑ-normal barrier for K. If $x^* \in K$, then $V_{x^*} = \{F''(x^*)^{-\frac{1}{2}} x | AF''(x^*)^{-\frac{1}{2}} x = 0, x \in E\}$.*

We want to mention that $F''(x^*)^{-\frac{1}{2}}$ is well defined on the boundary of the cone, since it is the quadratic representation of x^* in Jordan algebra. This theorem immediately implies the following corollary, which is extremely important to prove that any limit point of the iterate generated by our algorithm satisfies the second-order optimality condition.

COROLLARY 3.1. *Assume K is a symmetric cone in our optimization problem (3.1)–(3.3); then Q is positive semidefinite on V_{x^*} if and only if the linear operator $F''(x^*)^{-\frac{1}{2}}QF''(x^*)^{-\frac{1}{2}}$ is positive semidefinite on $\{x|AF''(x^*)^{-\frac{1}{2}}x = 0, x \in E\}$.*

4. Interior-point trust-region algorithm. In this section, we present our interior-point trust-region algorithm for solving (3.1)–(3.3). Global first-order and second-order convergence results are proved.

We assume $F(x)$ is the *ϑ-normal barrier* for the symmetric cone K and define the merit function as

$$f_{\eta_k}(x) = q(x) + \frac{1}{\eta_k}F(x). \tag{4.1}$$

In the inner iterations, $f_{\eta_k}(x)$ is decreased for a fixed η_k, while η_k is increased to positive infinity in outer iterations. From Lemma 2.1, for any $x_{k,j} \in K^\circ$ and $d \in E$, we have that $x_{k,j} + d \in K^\circ$, provided that $\|F''(x_{k,j})^{\frac{1}{2}}d\| \le \alpha_{k,j} < 1$. It follows from Lemma 2.2 that

$$\begin{aligned} F(x_{k,j}+d) - F(x_{k,j}) &\le \langle F'(x_{k,j}), d\rangle + \frac{\langle d, F''(x_{k,j})d\rangle}{2} + \frac{\|d\|^3_{x_{k,j}}}{3(1-\|d\|_{x_{k,j}})} \\ &\le \langle F'(x_{k,j}), d\rangle + \frac{\langle d, F''(x_{k,j})d\rangle}{2} + \frac{\alpha^3_{k,j}}{3(1-\alpha_{k,j})}. \end{aligned} \tag{4.2}$$

Therefore, we get

$$\begin{aligned} f_{\eta_k}(x_{k,j}+d) - f_{\eta_k}(x_{k,j}) \le{}& \frac{\langle d, (Q + \frac{1}{\eta_k}F''(x_{k,j}))d\rangle}{2} \\ &+ \left\langle Qx_{k,j} + c + \frac{1}{\eta_k}F'(x_{k,j}), d\right\rangle + \frac{\alpha^3_{k,j}}{3(1-\alpha_{k,j})\eta_k}. \end{aligned} \tag{4.3}$$

From the above relation, it is obvious that in order to decrease $f_{\eta_k}(x)$, we can try to minimize its upper bound given in the right-hand side of the above inequality. This leads to the following subproblem:

$$\min \frac{1}{2}\left\langle d, \left(Q + \frac{1}{\eta_k}F''(x_{k,j})\right)d\right\rangle + \left\langle Qx_{k,j} + c + \frac{1}{\eta_k}F'(x_{k,j}), d\right\rangle = m_{k,j}(d) \tag{4.4}$$

$$\text{subject to} \quad Ad = 0, \tag{4.5}$$

$$\|F''(x_{k,j})^{\frac{1}{2}}d\|^2 \le \alpha^2_{k,j}. \tag{4.6}$$

Define

$$Q_{k,j} = F''(x_{k,j})^{-\frac{1}{2}}QF''(x_{k,j})^{-\frac{1}{2}} + \frac{1}{\eta_k}I, \tag{4.7}$$

$$c_{k,j} = F''(x_{k,j})^{-\frac{1}{2}}\left(Qx_{k,j} + c + \frac{1}{\eta_k}F'(x_{k,j})\right), \tag{4.8}$$

$$A_{k,j} = AF''(x_{k,j})^{-\frac{1}{2}}, \tag{4.9}$$

and using the transformation

$$d' = F''(x_{k,j})^{\frac{1}{2}}d, \tag{4.10}$$

equations (4.4)–(4.6) can be rewritten as

$$\min q'_{k,j}(d') = \frac{1}{2}\langle d', Q_{k,j}d'\rangle + \langle c_{k,j}, d'\rangle \tag{4.11}$$

$$\text{subject to} \qquad A_{k,j}d' = 0, \tag{4.12}$$

$$\|d'\|^2 \le \alpha_{k,j}^2. \tag{4.13}$$

Instead of solving (4.11)–(4.13) exactly, we need only compute an approximate solution $d'_{k,j}$ satisfying the following two inequalities:

$$q'_{k,j}(d'_{k,j}) \le -\theta\|p_{k,j}\| \min\left\{\frac{\|p_{k,j}\|}{\beta_{k,j}}, \alpha_{k,j}\right\} \tag{4.14}$$

and

$$q'_{k,j}(d'_{k,j}) \le \theta\lambda_{k,j}\min\{\lambda_{k,j}^2, \alpha_{k,j}^2\}, \tag{4.15}$$

where $\theta \in (0, \frac{1}{2})$, $\beta_{k,j} = 1 + \|Q_{k,j}\|$, $p_{k,j}$ is the projection of $c_{k,j}$ onto the null space of $A_{k,j}$, and $\lambda_{k,j}$ is the least eigenvalue of $(N_{k,j})^* Q_{k,j} N_{k,j}$, $N_{k,j}$ being an orthonormal basis spanning the null space of $A_{k,j}$. We can see that inequality (4.15) makes sense only when $\lambda_{k,j} < 0$. The two conditions (4.14) and (4.15) are common in trust-region methods. Inequality (4.14) can be obtained at the Cauchy point and inequality (4.15) can be obtained when the negative curvature is exploited. Projected conjugate gradient/Lanczos-like methods are able to produce such a step at a reasonable cost (see Gould et al. [14]).

Once $d'_{k,j}$ is computed, we obtain the trial step

$$d_{k,j} = F''(x_{k,j})^{-\frac{1}{2}} d'_{k,j} \tag{4.16}$$

and define the predicted reduction in the merit function (4.1) by

$$Pred_{k,j} = m_{k,j}(0) - m_{k,j}(d_{k,j}) = -q'_{k,j}(d'_{k,j}). \tag{4.17}$$

The feasible set is denoted by $F_p = \{x \in E | Ax = b, x \in K\}$. Now we are ready to present our algorithm.

ALGORITHM 4.1 (an interior-point trust-region algorithm).

Step 0 **Initialization.** *An initial point $x_{0,0} \in ri\{F_p\}$, an initial trust-region radius $\alpha_{0,0} \in (0,1)$, and an initial parameter $\eta_0 > 0$ are given. The constants η'_1, η'_2, γ_1, and γ_2 are also given and satisfy $0 < \eta'_1 \le \eta'_2 < 1$ and $0 < \gamma_1 \le \gamma_2 < 1$. Two tolerance numbers $\epsilon_1, \epsilon_2 \in (0,1)$ are given. Set $k = 0$ and $j = 0$.*

Step 1 **Test inner iteration termination.** *If $\eta_k\|p_{k,j}\| < \epsilon_1$ and $\eta_k\lambda_{k,j} > -\epsilon_2$, set $x_{k+1,0} = x_{k,j}$ and go to Step* 5.

Step 2 **Step calculation.** *Solve* (4.11)–(4.13) *to obtain $d'_{k,j}$, which satisfies* (4.14) *and* (4.15)*, and set $d_{k,j}$ by* (4.16).

Step 3 **Acceptance of the trial point.** *If $x_{k,j} + d_{k,j} \notin ri\{F_p\}$, set $\rho_{k,j} = -\infty, x_{k,j+1} = x_{k,j}$ and go to Step* 4*; otherwise compute the ratio*

$$\rho_{k,j} = \frac{f_{\eta_k}(x_{k,j}) - f_{\eta_k}(x_{k,j} + d_{k,j})}{Pred_{k,j}}. \tag{4.18}$$

Let

$$x_{k,j+1} = \begin{cases} x_{k,j} + d_{k,j} & \text{if } \rho_{k,j} \geq \eta_1', \\ x_{k,j} & \text{otherwise.} \end{cases} \tag{4.19}$$

Step 4 **Trust-region radius update.** *If* $\rho_{k,j} \geq \eta_2'$, *set* $\alpha_{k,j+1} \in [\alpha_{k,j}, \infty)$; *if* $\rho_{k,j} \geq \eta_1'$, *set* $\alpha_{k,j+1} \in [\gamma_2 \alpha_{k,j}, \alpha_{k,j}]$; *if* $\rho_{k,j} < \eta_1'$, *set* $\alpha_{k,j+1} \in [\gamma_1 \alpha_{k,j}, \gamma_2 \alpha_{k,j}]$; *increase* j *by* 1 *and go to Step* 1.

Step 5 **Update parameter** η. *Choose* $\eta_{k+1} > \eta_k$ *in such a way as to ensure that* $\eta_k \to +\infty$ *when* $k \to +\infty$. *Increase* k *by* 1 *and go to Step* 1.

We want to mention that from Lemma 2.1, our trail point will always stay inside the feasible set if we keep $\alpha_{k,j}$ less than 1. However, we believe that our current mechanism can make the algorithm more efficient without keeping $\alpha_{k,j}$ less than 1. Although it allows the feasibility to not be achieved, sufficient descent of the merit function can be achieved in a successful step. Just like the usual notation in the trust-region literature, if $\rho_{k,j} \geq \eta_2'$, we call this iteration very successful; if $\rho_{k,j} \geq \eta_1'$, we call this iteration successful; if $\rho_{k,j} < \eta_1'$, we call this iteration a failure. Since $p_{k,j}$ is the projection of $c_{k,j}$ onto the null space of $A_{k,j}$, there exists a vector $y \in R^m$ such that

$$p_{k,j} = c_{k,j} - (A_{k,j})^* y. \tag{4.20}$$

LEMMA 4.1. *Let* $p_{k,j}$ *be given by* (4.20) *and*

$$s_{k,j} = Q x_{k,j} + c - A^* y; \tag{4.21}$$

if $\eta_k \|p_{k,j}\| < 1$, *then* $s_{k,j} \in K^\circ$ *and*

$$\langle x_{k,j}, s_{k,j} \rangle \leq \frac{1}{\eta_k}(\sqrt{\vartheta} + \vartheta). \tag{4.22}$$

Proof. It follows from (4.20) that

$$\begin{aligned} p_{k,j} = c_{k,j} - (A_{k,j})^* y &= F''(x_{k,j})^{-\frac{1}{2}} Q x_{k,j} + F''(x_{k,j})^{-\frac{1}{2}} c \\ &\quad + \frac{1}{\eta_k} F''(x_{k,j})^{-\frac{1}{2}} F'(x_{k,j}) - (A F''(x_{k,j})^{-\frac{1}{2}})^* y \\ &= F''(x_{k,j})^{-\frac{1}{2}} s_{k,j} + \frac{1}{\eta_k} F''(x_{k,j})^{-\frac{1}{2}} F'(x_{k,j}). \end{aligned} \tag{4.23}$$

Therefore, the above relation and our assumption $\eta_k \|p_{k,j}\| < 1$ imply that

$$\begin{aligned} \|F''(x_{k,j})^{-\frac{1}{2}}(\eta_k s_{k,j} + F'(x_{k,j}))\| &= \|F''(x_{k,j})^{\frac{1}{2}}(F''(x_{k,j})^{-1} \eta_k s_{k,j} - x_{k,j})\| \\ &< 1. \end{aligned} \tag{4.24}$$

Here the last equality follows from Lemma 2.3. Then from Lemma 2.1, we know that $F''(x_{k,j})^{-1} \eta_k s_{k,j} \in K^\circ$. It follows from Lemma 2.4 that $\eta_k s_{k,j} \in K^\circ$, and consequently $s_{k,j} \in K^\circ$.

It follows from (4.23) that $s_{k,j} = \frac{1}{\eta_k}(F''(x_{k,j})^{\frac{1}{2}}(\eta_k p_{k,j}) - F'(x_{k,j}))$. Consequently, we have that

$$\begin{aligned}
\langle x_{k,j}, s_{k,j}\rangle &= \frac{1}{\eta_k}(\langle F''(x_{k,j})^{\frac{1}{2}} x_{k,j}, \eta_k p_{k,j}\rangle + \langle x_{k,j}, -F'(x_{k,j})\rangle) \\
&\le \frac{1}{\eta_k}(\|F''(x_{k,j})^{\frac{1}{2}} x_{k,j}\|\|\eta_k p_{k,j}\| + \langle x_{k,j}, -F'(x_{k,j})\rangle) \\
&\le \frac{1}{\eta_k}(\langle x_{k,j}, F''(x_{k,j}) x_{k,j}\rangle^{\frac{1}{2}} + \langle x_{k,j}, -F'(x_{k,j})\rangle) \\
&= \frac{1}{\eta_k}(\langle x_{k,j}, -F'(x_{k,j})\rangle^{\frac{1}{2}} + \langle x_{k,j}, -F'(x_{k,j})\rangle) \\
&= \frac{1}{\eta_k}(\sqrt{\vartheta} + \vartheta). \qquad (4.25)
\end{aligned}$$

The last two equalities follow from Lemma 2.3. □

This convergence estimate is remarkable, considering the problem is nonconvex. We can achieve this mainly due to the special properties of the *ϑ-normal barrier*. From this estimate, we can see that the barrier parameter ϑ determines the complexity of our problem, which coincides with its role in the interior-point algorithm for convex programming.

For the rest of this section, we will show that the stop rule for the inner iterations can be satisfied in finitely many iterations. First, the following two lemmas are indispensable for our analysis.

LEMMA 4.2. (a) *The map $x \mapsto F''(x)^{-\frac{1}{2}}$ is continuous on the feasible set $\mathcal{F}_p$.*

(b) *There is a constant $C > 0$, such that $\|F''(x)^{-\frac{1}{2}}\| \le C$ for any $x \in \mathcal{F}_p$.*

For the positive orthant case, $F''(x)^{-\frac{1}{2}} = X = \mathrm{diag}\{x_1, x_2, \ldots, x_n\}$, and for the cone of semidefinite matrices, $F''(X)^{-\frac{1}{2}}\xi = X^{\frac{1}{2}}\xi X^{\frac{1}{2}}$. Therefore, part (a) is obviously true for these two cases. For the general symmetric cone, it is still true from the Jordan algebra point of view. We will give an explanation in the appendix. Part (b) follows immediately form part (a) and our assumption that $\mathcal{F}_p$ is bounded.

LEMMA 4.3. *There exists a positive constant C' such that if*

$$\alpha_{k,j} \le \min\left\{\frac{(1-\eta_2')\theta\eta_k\|p_{k,j}\|}{1+(1-\eta_2')\theta\eta_k\|p_{k,j}\|}, \frac{\|p_{k,j\|}}{C'}\right\}, \qquad (4.26)$$

then the iteration $\{k, j\}$ is very successful and $\alpha_{k,j+1} \ge \alpha_{k,j}$.

Proof. Let $C' = 1 + C^2\|Q\|$, where C is defined as in Lemma 4.2. It follows from the definition of $\beta_{k,j}$ and the last lemma that

$$\beta_{k,j} = 1 + \|Q_{k,j}\| \le C'. \qquad (4.27)$$

Therefore, when $\alpha_{k,j} \le \frac{\|p_{k,j}\|}{C'}$, the inequality (4.14) becomes

$$q'(d'_{k,j}) \le -\theta\|p_{k,j}\|\alpha_{k,j}. \qquad (4.28)$$

Inequality $\alpha_{k,j} \le \frac{(1-\eta_2')\theta\eta_k\|p_{k,j}\|}{1+(1-\eta_2')\theta\eta_k\|p_{k,j}\|} < 1$ ensures that $x_{k,j} + d_{k,j} \in ri\{\mathcal{F}_p\}$. It follows

from inequalities (4.3) and (4.28) that

$$|\rho_{k,j}-1| = \left|\frac{f_{\eta_k}(x_{k,j}) - f_{\eta_k}(x_{k,j}+d_{k,j}) + m_{k,j}(d_{k,j})}{Pred_{k,j}}\right| \le \frac{\frac{(\alpha_{k,j})^3}{3(1-\alpha_{k,j})}}{\theta\alpha_{k,j}\eta_k\|p_{k,j}\|}$$

$$< \frac{\frac{\alpha_{k,j}}{1-\alpha_{k,j}}}{\theta\eta_k\|p_{k,j}\|} \le 1-\eta_2'. \tag{4.29}$$

Therefore, $-(\rho_{k,j}-1) \le |\rho_{k,j}-1| \le 1-\eta_2'$, and we can see that $\rho_{k,j} \ge \eta_2'$. Consequently, the iteration is very successful and $\alpha_{k,j+1} \ge \alpha_{k,j}$. □

LEMMA 4.4. *If $\eta_k\|p_{k,j}\| \ge \epsilon$ for some constant $\epsilon \in (0, \alpha_{k,0})$ and all j, then*

$$\alpha_{k,j} \ge \min\left\{\frac{\gamma_1(1-\eta_2')\theta\epsilon}{1+(1-\eta_2')\theta\epsilon}, \frac{\gamma_1\epsilon}{C'\eta_k}\right\} \tag{4.30}$$

holds for all j.

Proof. It is easy to see that (4.30) holds for $j=0$ as $\alpha_{k,0} > \epsilon$. Assume that the j is the first integer such that $\alpha_{k,j+1} < \min\{\frac{\gamma_1(1-\eta_2')\epsilon}{1+(1-\eta_2')\epsilon}, \frac{\gamma_1\epsilon}{C'\eta_k}\}$; from the update of the trust-region radius, we know $\gamma_1\alpha_{k,j} \le \alpha_{k,j+1}$, and hence

$$\alpha_{k,j} < \min\left\{\frac{(1-\eta_2')\theta\epsilon}{1+(1-\eta_2')\theta\epsilon}, \frac{\epsilon}{C'\eta_k}\right\}$$

$$< \min\left\{\frac{(1-\eta_2')\theta\eta_k\|p_{k,j}\|}{1+(1-\eta_2')\theta\eta_k\|p_{k,j}\|}, \frac{\|p_{k,j\|}}{C'}\right\}. \tag{4.31}$$

From the above inequality and the last lemma, we have that $\alpha_{k,j+1} \ge \alpha_{k,j}$, which contradicts the assumption that $\alpha_{k,j+1}$ is the first trust-region radius violating (4.30). The contradiction shows that the lemma is true. □

Now we are ready to prove that the first part of the stopping rule, i.e., $\eta_k\|p_{k,j}\| < \epsilon_1$, can be satisfied in finitely many iterations.

LEMMA 4.5. (a) *If there are only finitely many successful iterations in each inner algorithm, then $x_{k,j} = x^*$ and $\|p_{k,j}\| = \|p(x^*)\| = 0$ for all sufficiently large j.*

(b) $\liminf_{j\to\infty} \eta_k\|p_{k,j}\| = 0$.

(c) $\lim_{j\to\infty} \eta_k\|p_{k,j}\| = 0$.

Proof. (a) The mechanism of the algorithm ensures that $x^* = x_{k,j_0} = x_{k,j}$ for all $j > j_0$, where $\{k, j_0\}$ is the index of the last successful iterate. Since all iterations are unsuccessful for sufficiently large j, we know $\alpha_{k,j}$ will converge to zero. If $\|p(x^*)\| = \|p_{k,j_0}\| > 0$, Lemma 4.4 implies that $\alpha_{k,j}$ will be bounded from zero. This contradiction shows that $\|p_{k,j_0}\|$ has to be zero.

(b) For the purpose of deriving contradiction, we assume that for all j, $\eta_k\|p_{k,j}\| \ge \epsilon$ for some $\epsilon > 0$. From Lemma 4.4 we know that $\alpha_{k,j} \ge \min\{\frac{\gamma_1(1-\eta_2)\theta\epsilon}{1+(1-\eta_2)\theta\epsilon}, \frac{\gamma_1\epsilon}{C'\eta_k}\}$ for all j. We consider all successful iterations $\{k, j\}$; then

$$f_{\eta_k}(x_{k,j}) - f_{\eta_k}(x_{k,j}+d_{x_{k,j}}) \ge \eta_1' Pred_{k,j}$$

$$\ge \eta_1'\theta\|p_{k,j}\| \min\left\{\frac{\|p_{k,j}\|}{\beta_{k,j}}, \alpha_{k,j}\right\}; \tag{4.32}$$

here the last inequality follows by inequality (4.14). From the above analysis, we know $\eta_1'\theta\|p_{k,j}\| \min\{\frac{\|p_{k,j}\|}{\beta_{k,j}}, \alpha_{k,j}\} \ge \sigma > 0$; here σ is some positive constant number

that is independent of j. If we have infinitely many successful iterations, the difference between $f_{\eta_k}(x_{k,0})$ and $f_{\eta_k}(x_{k,j})$ will be unbounded when $j \to +\infty$. This contradicts the assumption that $f_{\eta_k}(x)$ is bounded from below on the feasible set. Hence, we conclude that $\liminf_{j\to\infty} \eta_k \|p_{k,j}\| = 0$.

(c) For the purpose of deriving a contradiction, assume there is a subsequence of successful iterations $\{x_{k,j_i}\}$ such that $\eta_k\|p_{k,j_i}\| \geq 2\epsilon$ for some $\epsilon > 0$ and for all $\{j_i\}$. Our part (b) ensures the existence for each $\{j_i\}$ of a first successful iteration $l_i = l(j_i) > j_i$ such that $\eta_k\|p_{k,l_i}\| < \epsilon$. We thus obtain another subsequence of successful iterations $\{l_i\}$ such that $\eta_k\|p_{k,j}\| \geq \epsilon$ for $j_i \leq j < l_i$ and $\eta_k\|p_{k,l_i}\| < \epsilon$. Define $\kappa = \{j \in S | j_i \leq j < l_i\}$; here S indicates the successful iterations. For $j \in \kappa$, from inequality (4.32) we have

$$\begin{aligned} f_{\eta_k}(x_{k,j}) - f_{\eta_k}(x_{k,j+1}) &\geq \eta_1'\theta\|p_{k,j}\| \min\left\{\frac{\|p_{k,j}\|}{\beta_{k,j}}, \alpha_{k,j}\right\} \\ &\geq \eta_1'\theta\frac{\epsilon}{\eta_k}\min\left\{\frac{\epsilon}{\eta_k\beta_{k,j}}, \alpha_{k,j}\right\}. \end{aligned} \tag{4.33}$$

Since the sequence $f_{\eta_k}(x_{k,j})$ is monotonically decreasing and bounded from below, it is convergent. Therefore, the left-hand side of (4.33) must tend to zero when j tends to infinity. This gives that $\lim_{j\to\infty, j\in\kappa} \alpha_{k,j} = 0$. As a consequence, the second term dominates the minimum in (4.33) and we obtain that for $j \in \kappa$ sufficiently large,

$$\alpha_{k,j} \leq \frac{2\eta_k(f_{\eta_k}(x_{k,j}) - f_{\eta_k}(x_{k,j+1}))}{\eta_1'\theta\epsilon}. \tag{4.34}$$

We then deduce from this bound that, for i sufficiently large,

$$\begin{aligned} \|x_{k,j_i} - x_{k,l_i}\| &\leq \sum_{j=j_i, j\in\kappa}^{l_i-1} \|d_{x_{k,j}}\| = \sum_{j=j_i, j\in\kappa}^{l_i-1} \|F''(x_{k,j})^{-\frac{1}{2}} d'_{x_{k,j}}\| \\ &\leq \sum_{j=j_i, j\in\kappa}^{l_i-1} \|F''(x_{k,j})^{-\frac{1}{2}}\| \|d'_{x_{k,j}}\| \leq \sum_{j=j_i, j\in\kappa}^{l_i-1} C\alpha_{k,j} \\ &\leq C \sum_{j=j_i, j\in\kappa}^{l_i-1} \frac{2\eta_k(f_{\eta_k}(x_{k,j}) - f_{\eta_k}(x_{k,j+1}))}{\eta_1'\theta\epsilon} \\ &= \frac{2C\eta_k(f_{\eta_k}(x_{k,j_i}) - f_{\eta_k}(x_{k,l_i}))}{\eta_1'\theta\epsilon}. \end{aligned} \tag{4.35}$$

Here the third inequality follows from part (b) of Lemma 4.2, and the fourth inequality follows from inequality (4.34). Because $f_{\eta_k}(x_{k,j})$ is monotonically decreasing for j and bounded from below, it is convergent. Consequently, $f_{\eta_k}(x_{k,j_i}) - f_{\eta_k}(x_{k,l_i})$ tends to zero when $i \to +\infty$. We therefore obtain that $\|x_{k,j_i} - x_{k,l_i}\|$ tends to zero when $i \to +\infty$. Without loss of generality, we can assume x^* to be the common limit point of sequences $\{x_{k,j_i}\}_{i=1}^{\infty}$ and $\{x_{k,l_i}\}_{i=1}^{\infty}$. Since the sequences of our algorithm make the value of $f_{\eta_k}(x)$ decrease, the limit point x^* must be in the interior of the feasible set F_p. We know

$$\|p_{k,j_i} - p_{k,l_i}\| \leq \|p_{k,j_i} - p(x^*)\| + \|p(x^*) - p_{k,l_i}\|. \tag{4.36}$$

From part (a) of Lemma 4.2, we know that $c_{k,j}$ is continuous for x. Since $p_{k,j}$ is the projection of $c_{k,j}$ over the null space of $A_{k,j}$, $p_{k,j}$ is also continuous for x. Therefore, the right-hand side of inequality (4.36) will converge to zero when i tends to infinity. But, on the other hand, we know $\eta_k\|p_{k,j_i} - p_{k,l_i}\| \geq \eta_k\|p_{k,j_i}\| - \eta_k\|p_{k,l_i}\| \geq \epsilon$. Therefore, we get a contradiction, which means our initial assumption that $\eta_k\|p_{k,j}\|$ does not converge to zero cannot be true. This completes our proof. ∎

Now we prove that the second part of the stop rule, $\eta_k\lambda_{k,j} > -\epsilon_2$, can also be satisfied in finitely many iterations.

LEMMA 4.6. *For every fixed k, $\limsup_{j\to\infty} \lambda_{k,j} \geq 0$.*

Proof. For the purpose of deriving a contradiction, we assume that $\lambda_{k,j} \leq \lambda_*$ for some $\lambda_* < 0$ and all j. From inequality (4.15) we know that

$$Pred_{k,j} = -q'(d'_{x_{k,j}}) \geq -\theta\lambda_{k,j}\min\{\lambda_{k,j}^2, \alpha_{k,j}^2\} \geq -\theta\lambda_*\min\{\lambda_*^2, \alpha_{k,j}^2\}. \tag{4.37}$$

Therefore, we get

$$|\rho_{k,j} - 1| = \left|\frac{f_{\eta_k}(x_{k,j}) - f_{\eta_k}(x_{k,j} + d_{k,j}) + m_{k,j}(d_{k,j})}{Pred_{k,j}}\right| \leq \frac{\frac{(\alpha_{k,j})^3}{3(1-\alpha_{k,j})}}{-\eta_k\theta\lambda_*\min\{\lambda_*^2, \alpha_{k,j}^2\}}. \tag{4.38}$$

From this inequality, there exists a constant $\delta_1 > 0$ such that if $\alpha_{k,j} < \delta_1$, then $|\rho_{k,j} - 1| \leq 1 - \eta_2'$, that is, $\rho_{k,j} \geq \eta_2'$, which means this iteration is very successful and $\alpha_{k,j+1} \geq \alpha_{k,j}$. Now we assume $\{k, j_0\}$ is the first iteration such that $\alpha_{k,j_0} \leq \delta_1$; then from our above analysis, we know that $\alpha_{k,j} \geq \min\{\gamma_1\delta_1, \alpha_{k,j_0}\} := \delta_2$ for all $j \geq j_0$. Consequently,

$$\begin{aligned} f_{\eta_k}(x_{k,j}) - f_{\eta_k}(x_{k,j} + d_{x_{k,j}}) &\geq \eta_1' Pred_{k,j} \geq -\eta_1'\theta\lambda_*\min\{\lambda_*^2, \alpha_{k,j}^2\} \\ &\geq -\eta_1'\theta\lambda_*\min\{\lambda_*^2, \delta_2^2\} > 0 \end{aligned} \tag{4.39}$$

whenever $\{k, j\}$ is successful. If there are infinitely many successful iterations after $\{k, j_0\}$, (4.39) contradicts the fact that $f_{\eta_k}(x)$ is bounded from below. If there are finitely many successful iterations, the mechanism of our algorithm ensures that $\alpha_{k,j}$ converges to zero. But it again contradicts $\alpha_{k,j} \geq \min\{\gamma_1\delta_1, \alpha_{k,j_0}\} := \delta_2$ for all $j \geq j_0$. Hence our original assumption that there exists $\lambda_* < 0$ such that for all j, $\lambda_{k,j} \leq \lambda_*$ cannot be true. This completes the proof. ∎

From Lemma 4.6 and part (c) of Lemma 4.5, it is obvious that the stopping rule of our inner algorithm can be satisfied in finite many iterations.

THEOREM 4.1. *For every fixed η_k, Step 1 through Step 4 can be terminated in finitely many iterations.*

Finally, we can derive that any limit point of the sequences our algorithm generates satisfies both the first-order and the second-order optimality conditions.

THEOREM 4.2. *Assume x^* is any limit point of the sequences $\{x_{k,0}\}_{k=0}^{\infty}$ our algorithm generates; then x^* satisfies both the first-order and the second-order optimality conditions for our problem* (3.1)–(3.3).

Proof. We assume that $s_{k+1,0}$ is defined by (4.21). From Lemma 4.1, we know

$$\langle x_{k+1,0}, s_{k+1,0}\rangle \leq \frac{1}{\eta_k}(\vartheta + \sqrt{\vartheta}). \tag{4.40}$$

The above inequality implies that $\eta_k \to \infty$ as $k \to \infty$. Hence, any limit point of the sequences $\{x_{k,0}\}_{k=0}^{\infty}$ must satisfy the first-order optimality condition. Moreover, we know that

$$\lambda_{k+1,0} \geq \frac{-\epsilon_2}{\eta_k}, \tag{4.41}$$

which implies that

$$\liminf_{k\to\infty} \lambda_{k,0} \geq 0. \tag{4.42}$$

The above inequality, the definition of $\lambda_{k,j}$, and our continuity assumption show that $F''(x^*)^{-\frac{1}{2}} Q F''(x^*)^{-\frac{1}{2}}$ is positive semidefinite on the vector space $\{x | AF''(x^*)^{-\frac{1}{2}} x = 0, x \in E\}$. From Corollary 3.1, we know x^* satisfies the second-order optimality condition. □

5. Solve the large-scale trust-region subproblem. In this section, we show how to use our algorithm to solve the trust-region subproblem exactly and approximately. Numerical results are presented.

Consider the following standard trust-region subproblem:

$$\min \quad q(x) = \frac{1}{2}\langle x, Qx\rangle + \langle c, x\rangle \tag{5.1}$$

$$\text{subject to} \quad \|x\| \leq \Delta, \tag{5.2}$$

where $\|.\|$ is the ℓ_2-norm. By introducing a new variable x_{n+1}, we can transform this problem into the following nonlinear second-order cone programming:

$$\min \quad q(x) = \frac{1}{2}\langle x, Qx\rangle + \langle c, x\rangle \tag{5.3}$$

$$\text{subject to} \quad x_{n+1} = \Delta, \tag{5.4}$$

$$\sum_{i=1}^{n} x_i^2 \leq x_{n+1}^2. \tag{5.5}$$

Obviously, this is a special symmetric cone programming with $A = (0 \cdots 0, 1)$ and $K = \{x \in R^{n+1} : \sum_{i=1}^{n} x_i^2 \leq x_{n+1}^2 \text{ and } x_{n+1} \geq 0\}$. If we want to use Algorithm 4.1 to solve (5.3)–(5.5), we need to choose a method of solving (4.11)–(4.13). Since we are interested in solving large-scale problems, this motivates us to choose the methods that rely only on matrix-vector product. The first method in this class is the Steihaug–Toint truncated conjugate gradient method, which is due to Toint [26] and Steihaug [23]. And the adaptation to handle additional affine constraints can be found in Gould, Hribar, and Nocedal [15]. Here we give the version of the algorithm for solving (4.11)–(4.13).

ALGORITHM 5.1 (the Steihaug–Toint method with affine constraints).

Step 0 **Initialization.** *For fixed* $\{k, j\}$ *in* (4.11)–(4.13), *let* $d'_0 = 0$, $g_0 = c_{k,j}$, $v_0 = P_{A_{k,j}} c_{k,j}$, *and* $p_0 = -v_0$. *For* $h = 0, 1, \ldots$ *until convergence, perform the iteration.*

Step 1 **Check the negative curvature.** *Set* $\kappa_h = \langle p_h, Q_{k,j} p_h\rangle$. *If* $\kappa_h \leq 0$, *compute* σ_h *as the positive root of* $\|d'_h + \sigma p_h\| = \alpha_{k,j}$, *set* $d'_{h+1} = d'_h + \sigma_h p_h$, *and stop. End if*

Step 2 **Check the boundary constraints.** *Set* $\beta_h = \langle g_h, v_h\rangle/\kappa_h$. *If* $\|d'_h + \beta_h p_h\| \geq \alpha_{k,j}$, *compute* σ_h *as the positive root of* $\|d'_h + \sigma p_h\| = \alpha_{k,j}$, *set* $d'_{h+1} = d'_h + \sigma_h p_h$, *and stop. End if*

Step 3 **Perform the conjugate gradient step.** *Set* $d'_{h+1} = d'_h + \beta_h p_h$, $g_{h+1} = g_h + \beta_h Q_{k,j} p_h$, $v_{h+1} = P_{A_{k,j}} g_{h+1}$, *and* $p_{h+1} = -v_{h+1} + \frac{\langle g_{h+1}, v_{h+1}\rangle}{\langle g_h, v_h\rangle} p_h$.

Here $P_{A_{k,j}}$ is the projection onto the null space of $A_{k,j}$.

There are several advantages of this algorithm:

(1) It requires only matrix-vector product.
(2) It usually terminates very fast.
(3) It is applicable to problems with affine constraints.
(4) If the objective function is convex, the computed approximation solution gives at least half of the optimal reduction (Yuan [32]).

This Steihaug–Toint method is basically unconcerned with the trust region until it blunders into its boundary and stops. This is rather unfortunate, particularly, as considerable experience has shown that this frequently happens during the first few iterations when a negative curvature is present, causing the following disadvantages to the algorithm:

(A) Even if the problem is convex, optimal solution cannot be expected, except when the solution lies interior to the trust region.
(B) If it blunders into the boundary or a negative curvature is present too early, the approximate solution is not very good.
(C) It cannot handle the hard case.
(D) Optimal solution for the nonconvex problem is normally impossible for this algorithm.

Can we remove these disadvantages of Algorithm 5.1 while retaining its advantages? The answer is yes. After transforming (5.1)–(5.2) into (5.3)–(5.5), we use Algorithm 4.1 to solve it. In each iteration we use the Steihaug–Toint conjugate gradient method to solve (4.11)–(4.13). Since we basically repeat using Algorithm 5.1 in each iteration, we can keep all the advantages of Algorithm 5.1 as long as the number of iterations is not too big. It turns out that the number of iterations is very reasonable from the numerical results presented in this section. What can we achieve by doing this? We can get at least a first-order critical point of (5.3)–(5.5). This gives an optimal solution of (5.1)–(5.2) if Q is positive semidefinite. Thus we have removed (A). Algorithm 5.1 sometimes cannot give us a good approximate solution because it hits the boundary too early. By adding a ϑ-norm barrier to the quadratic model, we can prevent the iterates to reach the boundary too soon. This idea can give us a much better approximate solution, which is verified by the numerical results in this section. Therefore, we have removed (B). We know that Algorithm 5.1 cannot handle the hard case. If $c = 0$ and Q is indefinite, the method will terminate at $d' = 0$ with no decrease in the model. This cannot happen in our new framework. For Algorithm 4.1, a first-order critical point is always ensured even if the problem is in the hard case. Therefore, we have removed (C). Moreover, Algorithm 4.1 can be improved to find the optimal solution of (5.1)–(5.2) for all the cases, including the nonconvex case and the hard case. We first need the following lemma, which is well known in the trust-region literature.

LEMMA 5.1. *Any global minimizer* x^* *of* (5.1)–(5.2) *satisfies the equation*

$$(Q + \mu^* I)x^* = -c; \tag{5.6}$$

here $Q + \mu^* I$ *is positive semidefinite,* $\mu^* \geq 0$, *and* $\mu^*(\|x^*\| - \Delta) = 0$.

For a proof, see, e.g., Section 7.2 of Conn, Gould, and Toint [6].

The following algorithm removes (D).

ALGORITHM 5.2 (an algorithm for optimal solution).

Step 0 **Make Q positive semidefinite** *Find s, the smallest eigenvalue of Q and v, its corresponding eigenvector. If $s < 0$, set $Q = Q - sI$, end if.*

Step 1 **Solve new model.** *Use Algorithm* 4.1 *to solve* (5.3)–(5.5) *with Q positive semidefinite to get solution x_0.*

Step 2 **Go back to original model.** *If $s \geq 0$, set $x = x_0$, end if. If $s < 0$ and $\|x_0\| = \Delta$, set $x = x_0$, end if. If $s < 0$ and $\|x_0\| < \Delta$, set $x = x_0 + \sigma v$, σ is chosen so that $\|x_0 + \sigma v\| = \Delta$, end if.*

We claim that x is an optimal solution of (5.1)–(5.2). If $s \geq 0$, it is obvious. If $s < 0$ and $\|x_0\| = \Delta$, it follows from the fact that x_0 is an optimal solution of the new model with Q positive semidefinite and Lemma 5.1. If $s < 0$ and $\|x_0\| < \Delta$, $\mu^* = 0$ in (5.6) of Lemma 5.1 and consequently $Qx_0 = -c$ for the new convex Q. And since $Qv = 0$ for the new Q, $Qx = Q(x_0 + \sigma v) = -c$. From Lemma 5.1, we know that x is an optimal solution of (5.1)–(5.2). Therefore, we have removed all the disadvantages of the Steihaug–Toint method, while our algorithms mainly rely on the conjugate gradient method. For finding the optimal solution of the problem when it is nonconvex, we need to compute the least eigenvalue. However, those eigenvalue-based algorithms like those of Sorensen [22], Rojas, Santos, and Sorensen [21], and Rendl and Wolkowicz [19] require computing sequences of least eigenvalues, while we compute the least eigenvalue only once. As pointed out to us by a referee, Griffin and Gill [16] independently applied a truncated conjugate gradient algorithm to a shifted quadratic function to solve the trust-region subproblem.

We need to mention two implementation techniques when we use Algorithm 5.1 to solve (4.11)–(4.13) in each iteration.

We can see that the main computation in this algorithm is the product of the matrix $Q_{k,j}$ with a vector. In practice, we do not form $Q_{k,j}$ explicitly because it is expensive and destroys the sparse structure of Q. Since $Q_{k,j} = F''(x_{k,j})^{-\frac{1}{2}} Q F''(x_{k,j})^{-\frac{1}{2}} + \frac{1}{\eta_k} I$, we need to compute the product of the matrix $F''(x_{k,j})^{-\frac{1}{2}}$ with a vector, which can be done efficiently. For $F(x) = -\ln(x_{n+1}^2 - \sum_{i=1}^n x_i^2)$, $F''(x)^{-1}$ and $F''(x)^{-\frac{1}{2}}$ have the following explicit forms:

$$F''(x)^{-1} = \frac{1}{2}\begin{bmatrix} (x_{n+1}^2 - y^T y)I + 2yy^T & 2x_{n+1}y \\ 2x_{n+1}y^T & x_{n+1}^2 + y^T y \end{bmatrix}, \tag{5.7}$$

$$F''(x)^{-\frac{1}{2}} = \frac{\sqrt{2}}{2}\begin{bmatrix} \sqrt{x_{n+1}^2 - y^T y}I + \frac{yy^T}{\sqrt{x_{n+1}^2 - y^T y} + x_{n+1}} & y \\ y^T & x_{n+1} \end{bmatrix}, \tag{5.8}$$

where $y = (x_1 \dots x_n)^T$. For more details about the second-order cone and its barrier, see, e.g., Alizadeh and Goldfarb [2] or Faybusovich and Tsuchiya [10].

The first technique is that we do not have to formulate $F''(x)^{-\frac{1}{2}}$ explicitly for computing the product of the matrix $F''(x_{k,j})^{-\frac{1}{2}}$ with a vector. Since $yy^T u = \langle y, u\rangle y$ for any vector $u \in R^n$, the computation needs only $O(n)$ arithmetic operations.

The second technique is for the projection $P_{A_{k,j}}$. For any vector $u \in R^{n+1}$,

$$\begin{aligned} P_{A_{k,j}} u &= u - A_{k,j}^T (A_{k,j} A_{k,j}^T)^{-1} A_{k,j} u \\ &= u - F''(x_{k,j})^{-\frac{1}{2}} A^T (A F''(x_{k,j})^{-1} A^T)^{-1} A F''(x_{k,j})^{-\frac{1}{2}} u. \end{aligned} \tag{5.9}$$

Because $A = (0 \dots 0, 1)$, $AF''(x_{k,j})^{-1}A^T$ is just the $(n+1, n+1)$ entry of $F''(x_{k,j})^{-1}$. The only thing left is to compute the product of $F''(x_{k,j})^{-\frac{1}{2}}$ with a vector, which has been taken care of by $O(n)$ arithmetic operations.

By the above two techniques, each iteration of our algorithm takes hardly more extra work than the Steihaug–Toint truncated conjugate gradient method.

When applying Algorithm 4.1 to (5.3)–(5.5), we change the stopping rule for the inner iterations to make the algorithm more efficient. We remind the reader that there are two conditions of our stopping rule: (a) $\eta_k \|p_{k,j}\| < \epsilon_1$ and (b) $\eta_k \lambda_{k,j} > -\epsilon_2$ for some $\epsilon_1, \epsilon_2 \in (0, 1)$. We ignore condition (b), since a first-order critical point is good enough for our purpose. For condition (a), we need to change it a little because it is independent of whether or not optimality is nearly achieved. In practice, we directly follow the definition of the first-order optimality condition. If $x_{k,j}$ is a first-order critical point, $s_{k,j} = Qx_{k,j} + c - A^*y$ should be inside the second-order cone for some y. In our case $A = (0 \dots 0, 1)$ and all of the entries in the last row of Q are zeros. Therefore, we set $y = -\|Qx_{k,j} + c\|$ such that $s_{k,j}$ is inside the second-order cone. Then we stop the inner iteration as soon as we find $x_{k,j}$ such that $\langle x_{k,j}, s_{k,j} \rangle \le \frac{\epsilon}{\eta_k}$ for some constant ϵ. Lemma 4.1 suggests that $\epsilon = \sqrt{\vartheta} + \vartheta$ is a good choice. For the second-order cone, $\vartheta = 2$. From our practical experience, it works very well for convex problems. In the nonconvex case, it seems that this stopping rule works for some problems but not all of them, which remains to be investigated. The algorithm is halted as soon as $x_{k,j}$ is found such that $\langle x_{k,j}, s_{k,j} \rangle \le 10^{-4}$. In Algorithm 4.1, $\eta_1' = 0.05$ and $\eta_2' = 0.9$ are used and the trust region is updated according to the usual rule. If $\rho_{k,j} \ge \eta_2'$, set $\alpha_{k,j+1} = \max(\alpha_{k,j}, 2\|d_{k,j}'\|)$; if $\rho_{k,j} \in [\eta_1', \eta_2')$, set $\alpha_{k,j+1} = \alpha_{k,j}$; if $\rho_{k,j} < \eta_1'$, set $\alpha_{k,j+1} = \frac{1}{2}\alpha_{k,j}$. The initial value of parameter η is set to be $\frac{1}{\Delta}$ and is updated by $\eta_{k+1} = 10\eta_k$. In each iteration, the Steihaug–Toint conjugate gradient method (Algorithm 5.1) is stopped as soon as $\|v_h\| \le 10^{-\frac{3}{2}}\|v_0\|$ if it does not hit the boundary and negative curvature is not present before that. We decide not to put an upper bound on the number of Steihaug–Toint iterations, which is denoted by h in Algorithm 5.1. In this way, for the convex problems, we will be able to know the number of iterations our algorithm needs to get an optimal solution if we solve each trust-region subproblem approximately, which is measured by $\|v_h\| \le 10^{-\frac{3}{2}}\|v_0\|$.

The algorithms are tested in MATLAB 7.0 on a Linux system. We run our experiments on a Gateway computer with a Pentium IV 3.2G processor and 1G RAM. We compare our results with a software called "Newtrust4b" based on Rendel and Wolkowicz [19]. We choose "Newtrust4b" because it is one of the best software packages for finding the optimal solution of the trust-region subproblem and because it is also implemented in MATLAB code. For the test problems, Q and c are randomly generated with entries uniformly distributed on (0,1). We set the trust-region radius $\Delta = 1$. For different radiuses, the computation time and the number of iterations may vary, but they vary reasonably. In Tables 1–5, n is the dimension of the problems, d is the density of Q; i.e., Q has $d * n^2$ nonzero entries. The data in those columns under the algorithms' names is the computational time by seconds. Sometimes, the MATLAB timing is dependent on the CPU load. Our timings have been averaged to eliminate this dependency. And *its* is the number of iterations of our algorithm. ST *its* is the total number of Steihaug–Toint iterations used during the whole computation. From Algorithm 5.1, we can see that the dominating cost of each Steihaug–Toint iteration is the product of $Q_{k,j}$ and p_h. Therefore, ST *its* gives us the total number of the matrix-vector products used by our algorithm. We first test some convex problems. To make the problem convex, we let $Q = Q - sI$ if s, the least eigenvalue of

TABLE 1
Convex singular problems.

n	d	Newtrust4b	Algorithm 4.1	its	ST its
2500	1	57	3	10	66
5000	0.5	1328	67	28	156
10000	0.05	284	16	12	42
20000	0.01	240	10	9	19
100000	0.0001	181	5	8	15
200000	101 band	248	17	7	11

TABLE 2
Q is sparse with $d = 0.03$.

n	Newtrust4b	Algorithm 5.2	Eigenvalue time	its	ST its
4000	8	6	4	10	31
8000	74	68	58	22	53
12000	156	143	121	21	53
16000	303	278	239	21	54
20000	356	312	253	20	48

Q, is negative. In this way, the problem is convex and nearly singular. To show that our algorithm can handle the singular problems, we set $Q = \left[\begin{smallmatrix} Q & 0 \\ 0 & 0 \end{smallmatrix}\right]$ to make it even more singular. Now the dimension of Q is $n + 1$, and so is c. Since the problems are convex, Algorithm 4.1 gives us optimal solutions.

We can see that Algorithm 4.1 outperforms Newtrust4b for the convex singular problems. The success of Algorithm 4.1 in this case is a good basis for Algorithm 5.2. If the problem is nonconvex or if we do not know whether or not our problem is convex, we have to use Algorithm 5.2 to get an optimal solution. The following two groups of results show us the performance of Algorithm 5.2.

We can see that Algorithm 5.2 is competitive with Newtrust4b in both sparse and dense cases. The eigenvalue time is the computational time cost by computing the least eigenvalue of Q in Algorithm 5.2, which becomes dominant when the problem becomes large. Fortunately, we have reduced this part to the minimum level in Algorithm 5.2 (we need only compute the least eigenvalue once for all). Moreover, the number of iterations and ST iterations of our algorithm is independent of the dimension of the problems.

So far, we have been focusing on finding the optimal solution of the trust-region subproblem. But if the problem is nonconvex, Algorithm 4.1 can deliver us only an approximate solution. How good is Algorithm 4.1 for nonconvex problems? From our practical experience, we have to say that the performance of Algorithm 4.1 on finding an approximate solution for nonconvex problems is not as stable as its performance on finding exact solution for convex problems. This is reflected by the fact that the convergence is sensitive to the inner iteration stopping rule. The inner iteration stopping rule, $\langle x_{k,j}, s_{k,j}\rangle \le \frac{\sqrt{\vartheta}+\vartheta}{\eta_k}$, works for some of our testing problems but not for all of them. For those test problems where Algorithm 4.1 has good performance, the number of iterations is around 40. For those test problems where Algorithm 4.1 has bad performance, the number of iterations can reach over 100. The change of stopping rule of inner iteration (like from $\langle x_{k,j}, s_{k,j}\rangle \le \frac{\sqrt{\vartheta}+\vartheta}{\eta_k}$ to $\langle x_{k,j}, s_{k,j}\rangle \le \frac{1}{\eta_k}$) can significantly affect the performance of the algorithm on the same test problem. This phenomenon remains to be investigated. On the positive side, even for the problems where Algorithm 4.1 has bad performance, the convergence slows down only when η_k

TABLE 3
Q is dense with d = 1.

n	Newtrust4b	Algorithm 5.2	Eigenvalue time	its	ST its
1000	9	3	2	15	144
2000	14	7	4	22	141
3000	36	18	11	20	154
4000	147	71	56	22	144
5000	323	145	127	19	164

TABLE 4
Nonconvex singular problems.

n	d	Algorithm 4.1	Accuracy	its	ST its
3000	1	1	90%	5	10
6000	1	5	90%	5	10
10000	0.03	3	95%	3	6
20000	0.03	11	95%	3	6
100000	0.0001	2	99%	3	4
200000	101 band	8	99%	3	4

becomes large and our solution is close to the optimal solution. Here we give a group of examples to show that Algorithm 4.1 can deliver us a good approximate solution at a relatively low cost. For each of these problems, we first use Algorithm 5.2 to get an optimal solution and consequently the best possible reduction. Then we use Algorithm 4.1 to solve it and stop the algorithm when 90% of the optimal reduction is achieved. Q is randomly generated with entries uniformly distributed on (0,1). We have check that Q is indefinite. To show the performance of our algorithm on the singular problems, we set $Q = \left[\begin{smallmatrix} Q & 0 \\ 0 & 0 \end{smallmatrix}\right]$. Therefore, the actual dimension of Q in Table 4 is $n + 1$.

We can see that after a few Steihaug–Toint iterations, Algorithm 4.1 can deliver us a good approximate solution. Therefore, we have achieved our goal of improving the solution quality of the Steihaug–Toint method while keeping its computational advantages.

In summary, techniques developed in this section give us two algorithms for solving the trust-region subproblem. Algorithm 5.2 gives us an optimal solution for both convex and nonconvex problems. Algorithm 4.1 gives us a good approximate solution for nonconvex problems and an optimal solution for convex problems.

6. Further numerical results and implementation issues. In this section, we discuss some implementation issues for solving general symmetric cone programming. We also present some numerical results of solving a class of quadratic programming.

To solve the general symmetric cone programming, we have to handle three basic implementation issues. The first issue is to find a starting point in our feasible set. This feasible set has been well studied in the interior-point algorithm literature. We can use the same technique to find a feasible starting point for our problem. The second issue is to handle affine constraints. This requires us either to solve the normal equations or to project iterates onto the null space of $A_{k,j}$. Gould, Hribar, and Nocedal [15] is a good reference for handling this issue. The third issue is about preconditioning. We recall that $Q_{k,j} = F''(x_{k,j})^{-\frac{1}{2}} Q F''(x_{k,j})^{-\frac{1}{2}} + \frac{1}{\eta_k} I$. When we are getting close to the optimal solution, η_k is getting large, and consequently the right part $\frac{1}{\eta_k} I$ is about to disappear. At the same time, the iterate $x_{k,j}$ is getting close

TABLE 5
Q is positive definite with density d = 1.

n	its	ST its	Algorithm 4.1
1000	67	529	4
2000	75	799	17
3000	81	777	40
4000	71	630	75
5000	70	671	109

to the boundary. Therefore, $F''(x_{k,j})^{-\frac{1}{2}}$ becomes nearly singular, which can make the condition number of $Q_{k,j}$ large. As we know, the convergence behavior of the conjugate gradient method is strongly dependent on the conditioning of $Q_{k,j}$. Therefore, the appropriate preconditioning technique is necessary to make the algorithm efficient.

Handling these implementation issues is beyond the scope of this paper. However, to see how our algorithm performs on solving problems other than the trust-region subproblem, we present some numerical results for a class of quadratic programming, which is minimization of a strictly convex quadratic objective function over the positive orthant. This problem is bounded from below. We use the vector e with every entry 1 as our starting point. Q is randomly generated with entries uniformly distributed on (0,1). We make the problem convex by letting $Q = Q + (-s + 1)I$ if s, the least eigenvalue of Q, is negative. c is randomly generated with entries uniformly distributed on $(-1, 0)$. In this way, the problem will have nontrivial solution. In each step, the conjugate gradient method is stopped if the iterate hits the boundary or $\|g_h\| \leq 10^{-3/2}\|g_0\|$. The inner iteration is stopped when we find $x_{k,j}$ such that $s_{k,j} = Qx_{k,j} + c$ belongs to positive orthant and $\langle x_{k,j}, s_{k,j}\rangle \leq \frac{n+\sqrt{n}}{\eta_k}$. The algorithm is halted as soon as $x_{k,j}$ is found such that $\langle x_{k,j}, s_{k,j}\rangle \leq 10^{-4}$. All other implementation techniques are similar to those we discussed in section 5. The following is a group of results.

The number of iterations as well as the total number of Steihaug–Toint iterations are independent of the dimension of problems, which makes our algorithm have practical potential for solving large-scale problems. One practical observation we want to mention is that the computational time of reducing $\langle x_{k,j}, s_{k,j}\rangle$ from 10^{-3} to 10^{-4} is even more than the computational time of reducing it from the starting value to 10^{-3}. This is caused by the fact that the convergence of the conjugate gradient algorithm considerably slows down when the iterate is close to optimal solution and consequently the boundary, which agrees with our theoretical analysis above. Therefore, appropriate preconditioning is indispensable to make the algorithm more efficient. Since it has been shown in section 5 that Algorithm 4.1 works well for solving the singular problems, an alternative way is to use it to solve the trust-region subproblem when the iterate is close to the boundary of the positive orthant. Which way is better remains to be investigated.

7. Concluding remarks. In this paper, we have shown that combining the techniques developed in trust-region literature (especially Conn, Gould, and Toint [6]) with those techniques in interior-point method literature can be very powerful both in theoretical analysis and practical implementation. For further theoretical research, Lu and Yuan [17] have recently proved that the complexity of an interior-point trust-region algorithm for convex programming is polynomial time. On the practical side, the numerical results presented in this paper show that our algorithm

has practical potential. But a lot more work needs to be done to turn this method into a practical software package for solving general symmetric cone programming.

Appendix. In this appendix, we describe the face $V_{A'}$ for the semidefinite case. We use the Jordan algebra technique to prove Theorem 3.3 and give an explanation why Lemma 4.2 holds for general symmetric cone.

If $K = S_+^{n\times n}$, we know $F(X) = -\ln\det(X)$ is a *ϑ-normal barrier* for $S_+^{n\times n}$. Let $A' \in \partial K$, and $\operatorname{rank}(A') = r < n$; then $F''(A')^{-\frac{1}{2}}X = A'^{\frac{1}{2}}XA'^{\frac{1}{2}}$. We set $V = \{A'^{\frac{1}{2}}XA'^{\frac{1}{2}} | AA'^{\frac{1}{2}}XA'^{\frac{1}{2}} = 0, X \in S^{n\times n}\}$. In section 3, we have claimed that $V_{A'} = V$. Now we give a proof.

Proof. First, we need to characterize $V_{A'}$. Since A' is a semidefinite matrix, then we can find an orthogonal matrix U, such that $U^{-1}A'U = D$, where $D = \operatorname{diag}\{\lambda_1, \ldots, \lambda_r, 0, \ldots, 0\}$ with $\lambda_i > 0$, $i = 1, \ldots, r$.

Let $C = \operatorname{diag}\{0, \ldots, 0, 1, \ldots, 1\}$ be the matrix whose first r diagonal entries are 0 and the last $n - r$ diagonal entries are 1, and let $Q' = UCU^{-1}$. Then Q' is a nonzero positive semidefinite matrix and $\langle Q', A'\rangle = \langle UCU^{-1}, UDU^{-1}\rangle = \langle C, D\rangle = 0$. Furthermore, for any positive semidefinite $n \times n$ matrix X, $\langle Q', X\rangle \geq 0$. Therefore, the hyperplane $H = \{X \in S^{n\times n} | \langle Q', X\rangle = 0\}$ isolates $S_+^{n\times n}$ and contains A'. We claim that

$$\digamma_{A'} = \{X \in S_+^{n\times n} | AX = b, \langle Q', X\rangle = 0\}.$$

To see that A' is a relative interior point of $\digamma_{A'}$, we need only show that A' is an interior point of $\digamma = \{X \in S_+^{n\times n} | \langle Q', X\rangle = 0\}$. The map $X \mapsto Y = U^{-1}XU$ is a nondegenerate linear transform which maps $S_+^{n\times n}$ onto itself, maps Q' onto C, and maps A' onto D. Then $\digamma$ is mapped onto $\digamma' = \{Y \in S_+^{n\times n} | \langle C, Y\rangle = 0\}$. Clearly, Y must have the last $n - r$ rows and last $n - r$ columns be 0. The upper left $r \times r$ submatrix Y' of Y can be arbitrary positive semidefinite matrix, $Y = \left(\begin{smallmatrix} Y' & 0 \\ 0 & 0 \end{smallmatrix}\right)$. It is easy to see that $\digamma'$ contains D in its interior. Since $Y \mapsto X = UYU^{-1}$ is a nondegenerate linear transform, which maps D onto A' and $\digamma'$ onto $\digamma$, then A' is an interior point of $\digamma$. Therefore A' is a relative interior point of $\digamma_{A'}$. Then

$$V_{A'} = \{X \in S^{n\times n} | AX = 0, \langle Q', X\rangle = 0\}.$$

To show that $V_{A'} = V$, we need only verify that

$$V' = \{A'^{\frac{1}{2}}XA^{\frac{1}{2}} | X \in S_+^{n\times n}\} = \digamma = \{X \in S_+^{n\times n} | \langle Q', X\rangle = 0\}.$$

For all $F \in \digamma$, since $\digamma = U\digamma' U^{-1}$,

$$F = U \begin{pmatrix} Y' & 0 \\ 0 & 0 \end{pmatrix} U^{-1}$$

for some $r \times r$ positive semidefinite matrix Y'. Let

$$X = U \begin{pmatrix} D'Y'D' & 0 \\ 0 & 0 \end{pmatrix} U^{-1};$$

here $D' = \operatorname{diag}\{\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_r}\}^{-1}, \lambda_i > 0, i = 1, \ldots, r$ are the eigenvalues of A'. Then

$$A'^{\frac{1}{2}}XA'^{\frac{1}{2}} = \left(U\operatorname{diag}\{\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_r}, 0, \ldots, 0\}U^{-1}\right)\left(U\begin{pmatrix} D'Y'D' & 0 \\ 0 & 0 \end{pmatrix}U^{-1}\right),$$

$$\left(U\mathrm{diag}\{\sqrt{\lambda_1},\ldots,\sqrt{\lambda_r},0,\ldots,0\}U^{-1}\right) = U\begin{pmatrix} Y^{'} & 0 \\ 0 & 0 \end{pmatrix}U^{-1} = F.$$

We conclude that $\digamma \subset V'$. It is easy to see that $\dim(\digamma) = \dim(V') = \frac{r(r+1)}{2}$, and we get $\digamma = V'$. Therefore,

$$\{X \in S^{n\times n} | \langle Q', X\rangle = 0\} = Aff(\digamma) = Aff(V') = \{A'^{\frac{1}{2}} X A'^{\frac{1}{2}} | X \in S^{n\times n}\},$$

which implies that

$$V_{A'} = V = \{F''(A')^{-\frac{1}{2}} X | AF''(A')^{-\frac{1}{2}} X = 0, X \in S^{n\times n}\}.$$

Before we prove Theorem 3.3, we introduce some notation of Jordan algebra. Since every symmetric cone K can be realized as a cone of squares in an appropriated Euclidean algebra (see Faraut and Koranyi [9] for details), we can use the Jordan algebra technique to prove Theorem 3.3.

Let V be an Euclidean Jordan algebra and Ω be a cone of invertible squares in V. We define $\langle x, y\rangle = \mathrm{tr}(x \circ y)$ as the canonical scalar product in V. Let $F(x) = -\log\det(x)$, $x \in \Omega$. Then $F''(x) = P(x)^{-1}$; here $F''(x)$ is the Hession of F evaluated at $x \in \Omega$ with respect to the canonical scalar product $\langle , \rangle$. $P(x)$ is the quadratic representation of x. We assume $\mathrm{rank}(V) = r$. When x is on the boundary $\partial\Omega$ of Ω, $\mathrm{rank}(x) = j < r$.

In the following, we fix a Jordan frame $c_1, \ldots, c_r$ and denote $e_j = c_1 + \cdots + c_j$, $V^{(j)} = V(e_j, 1)$. We denote by Ω^j the symmetric cone associated with the subspace $V^{(j)}$, i.e., the interior relative to $V^{(j)}$. Then $\Omega_j \subset \partial\Omega$. The following lemma characterizes the boundary of symmetric cone. For a proof, see Proposition IV.3.1 in Faraut and Koranyi [9].

LEMMA A.1. *For x in $\overline{\Omega}$ the following properties are equivalent:*

(a) *The rank of x is j.*
(b) *$x \in k\Omega^j$ for some k in $K = G \cap O(V)$; here G is the connected component of the identity in $G(\Omega)$ and $G(\Omega)$ denotes the set of automorphisms of Ω.*
(c) *The rank of $P(x)$ is equal to the dimension of $V^{(j)}$.*

Now we assume $x^* \in \overline{\Omega}$ and $\mathrm{rank}(x^*) = j$. From Lemma A.1, we know that $x^* \in k\Omega^j$ for some k in K. It can be verified that $V_{x^*} = kV^{(j)}$. Now we are ready to prove Theorem 3.3.

Proof of Theorem 3.3. From the above analysis, we need only prove $P(x^*)^{\frac{1}{2}} V = kV^{(j)}$. Since $P(x^*)$ is a positive semidefinite linear operator, $P(x^*)^{\frac{1}{2}} V = P(x^*)V$. Therefore we need only prove $P(x^*)V = kV^{(j)}$. From part (b) of Lemma A.1, we know $x^* = k\sum_{i=1}^{j} \lambda_i c_i = kP(a)e_j$, with $a = \sum_{i=1}^{j} \sqrt{\lambda_i} c_i + \sum_{i=j+1}^{r} c_i$. Then $P(x^*) = p(kP(a)e_j) = kP(P(a)e_j)k^* = kP(a)P(e_j)P(a)k^*$; here the second equality follows by Proposition III.5.2 in Faraut and Koranyi [9] and the last equality follows by Proposition II.3.3 in Faraut and Koranyi [9]. Since $P(e_j)$ is the orthogonal projection onto $V^{(j)}$ and $P(a)$ maps $V^{(j)}$ onto $V^{(j)}$, it is easy to see that $P(x^*)V \subset kV^{(j)}$. Since from part (c) of Lemma A.1, we know $\mathrm{rank}(P(x^*)) = \mathrm{rank}(V^{(j)}) = \mathrm{rank}(kV^{(j)})$, we conclude that $P(x^*)V = kV^{(j)}$. We complete the proof. □

Part (a) of Lemma 4.2 holds only because $F''(x^*)^{-\frac{1}{2}} = P(x^*)^{\frac{1}{2}}$ and $P(x^*)$ is the quadratic representation of x^*.

Acknowledgments. The first author would like to thank his advisor, Andrew Sommese, at Notre Dame for his help and support, and Professor Faybusovich for his guidance in the interior-point methods. Professor Yinyu Ye read this paper carefully

and gave some sincere advice; we appreciate his support and encouragement. We would like to thank Mr. Dong Yang for his help in the implementation of our algorithm. We are grateful to the associate editor and the referees for their precious comments.

REFERENCES

[1] P. A. Absil and A. L. Tits, *Newton-KKT interior-point methods for indefinite quadratic programming*, Comput. Optim. Appl., to appear.
[2] F. Alizadeh and D. Goldfarb, *Second-order cone programming*, Math. Program., 95 (2003), pp. 3–51.
[3] A. S. El Bakry, R. A. Tapia, T. Tsuchiya, and Y. Zhang, *On the formulation and theory of Newton interior point methods for nonlinear programming*, J. Optim. Theory Appl., 89 (1996), pp. 507–541.
[4] A. Barvinok, *A Course in Convexity*, AMS, Providence, RI, 2002.
[5] J. F. Bonnans and A. Shapiro, *Perturbation Analysis of Optimization Problems*, Springer-Verlag, New York, 2000.
[6] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, *Trust-Region Methods*, SIAM, Philadelphia, 2000.
[7] A. R. Conn, N. I. M. Gould, D. Orban, and Ph. L. Toint, *A primal-dual trust-region algorithm for nonconvex nonlinear programming*, Math. Program., 87 (2000), pp. 215–249.
[8] I. I. Dikin, *Iterative solution of problems of linear and quadratic programming*, Soviet Math. Dokl., 8 (1967), pp. 674–675.
[9] J. Faraut and A. Koranyi, *Analysis on Symmetric Cones*, Oxford University Press, New York, 1996.
[10] L. Faybusovich and T. Tsuchiya, *Primal-dual algorithms and infinite-dimensional Jordan algebras of finite rank*, Math. Program., 97 (2003), pp. 471–493.
[11] L. Faybusovich and Y. Lu, *Jordan-algebraic aspects of nonconvex optimization over symmetric cone*, Appl. Math. Optim., 53 (2006), pp. 67–77.
[12] A. Forsgren and P. E. Gill, *Primal-dual interior methods for nonconvex nonlinear programming*, SIAM J. Optim., 8 (1998), pp. 1132–1152.
[13] D. M. Gay, M. L. Overton, and M. H. Wright, *A primal-dual interior method for nonconvex nonlinear programming*, in Advances in Nonlinear Programming, Y. Yuan, ed., Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998, pp. 31–56.
[14] N. I. M. Gould, S. Lucidi, M. Roma, and Ph. L. Toint, *Solving the trust-region subproblem using the Lanczos method*, SIAM J. Optim., 9 (1999), pp. 504–525.
[15] N. I. M. Gould, M. E. Hribar, and J. Nocedal, *On the solution of equality constrained quadratic programming problems arising in optimization*, SIAM J. Sci. Comput., 23 (2001), pp. 1376–1395.
[16] J. Griffin and P. E. Gill, *Trust-region interior methods for large-scale optimization*, in Proceedings of the Eighth SIAM Conference on Optimization, Stockholm, Sweden, 2005.
[17] Y. Lu and Y. Yuan, *An Interior-Point Trust-Region Polynomial Algorithm for Convex Programming*, Technical report, Department of Mathematics, University of Notre Dame, Notre Dame, IN, 2006.
[18] Y. E. Nesterov and A. S. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*, SIAM, Philadelphia, 1994.
[19] F. Rendel and H. Wolkowicz, *A semidefinite framework for trust region subproblem with applications to large scale minimization*, Math. Programming, 77 (1997), pp. 273–299.
[20] J. Renegar, *A Mathematical View of Interior-Point Methods in Convex Programming*, SIAM, Philadelphia, 2001.
[21] M. Rojas, S. A. Santos, and D. C. Sorensen, *A new matrix-free algorithm for the large-scale trust-region subproblem*, SIAM J. Optim., 11 (2000), pp. 611–646.
[22] D. C. Sorensen, *Minimization of a large-scale quadratic function subject to a spherical constraint*, SIAM J. Optim., 7 (1997), pp. 141–161.
[23] T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.
[24] J. Sun, *A convergence proof for an affine-scaling method for convex quadratic programming without nondegeneracy assumptions*, Math. Programming, 60 (1993), pp. 69–79.
[25] A. L. Tits, A. Wächter, S. Bakhtiari, T. J. Urban, and C. T. Lawrence, *A primal-dual interior-point method for nonlinear programming with strong global and local convergence properties*, SIAM J. Optim., 14 (2003), pp. 173–199.
[26] Ph. L. Toint, *Towards an efficient sparsity exploiting Newton method for minimization*, in

Sparse Matrices and Their Uses, I. S. Duff, ed., Academic Press, London, 1981, pp. 57–88.
[27] R. J. Vanderbei and D. F. Shanno, *An interior-point algorithm for nonconvex nonlinear programming*, Comput. Optim. Appl., 13 (1999), pp. 231–252.
[28] A. Wächter, *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, 2002.
[29] Y. Ye, *On an affine scaling algorithm for non-convex quadratic programming*, Math. Programming, 52 (1992), pp. 285–300.
[30] Y. Ye, *Interior Point Algorithms: Theory and Analysis*, John Wiley and Sons, New York, 1997.
[31] Y. Ye and S. Zhang, *New results on quadratic minimization*, SIAM J. Optim., 14 (2003), pp. 245–267.
[32] Y. Yuan, *On the truncated conjugate gradient method*, Math. Program., 87 (2000), pp. 561–571.
[33] S. Zhang, *Quadratic maximization and semidefinite relaxation*, Math. Program., 87 (2000), pp. 453–465.