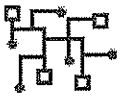

ISE

Industrial and
Systems Engineering

An interior point constraint generation method for semi-infinite linear programming

Mohammad R. Oskoorouchi
California State University San Marcos

Hamid Ghaffari
McMaster University

Tamás Terlaky
Lehigh University

Report: 08T-005

An interior point constraint generation method for semi-infinite linear programming*

Mohammad R. Oskoorouchi †
College of Business Administration
California State University San Marcos
San Marcos, California 92096, USA
Email: moskooro@csusm.edu

Hamid R. Ghaffari
School of Computational Engineering and Science
McMaster University
Hamilton ON, L8P4K1, Canada
Email: ghaffar@univmail.cis.mcmaster.ca

Tamás Terlaky‡
School of Computational Engineering and Science
McMaster University
Hamilton ON, L8P4K1, Canada
Email: terlaky@mcmaster.ca

July 2008

*This research has been completed when the first author was visiting McMaster University

†The research of this author has been support by a research grant from the College of Business Administration, California State University San Marcos, and the University Professional Development Grant.

‡The research of this author has been support by NSERC, CRC, and MITACS.

Abstract

We propose an interior point constraint generation algorithm for semi-infinite linear optimization (SILP) and prove that the algorithm converges to an ε-solution of SILP after a finite number of constraints is generated. We derive a complexity bound on the number of Newton steps needed to approach the updated $\mu-$ center after adding multiple violating constraints, and a complexity bound on the total number of constraints that is required for the overall algorithm to converge.

We implement our algorithm to solve second-order cone optimization (SOCO) problems and compare our numerical results with that of SeDuMi. We show that our interior point constraint generation method outperforms classical primal-dual interior point methods on a class of large-scale SOCO problems.

Keywords: Semi-infinite programming, conic optimization, central path, constraint generation method.

1 Introduction

In this paper we present a constraint generation algorithm for problems with linear objective function and exponentially large, or infinitely many, constraints. This class of problems essentially contains semi-infinite linear programming (SILP) and convex optimization (CO); in particular, semidefinite optimization (SDO) and second-order cone optimization (SOCO).

Although there exist many efficient software packages based on polynomial interior point methods for convex conic optimization (such as SDPT3 [22]), SeDuMi [21], and CSDP [5]), and based on low-rank factorization (such as SDPLR [6]), but we would still like to keep this class of problems within our domain as we develop this algorithm. There are two main reasons for this. First, while today's software packages perform extremely well on small to moderate size convex conic problems, they can't efficiently handle large-scale problems. For example a problem with a few thousand conic constraints of large size, say 10^4, especially when it is dense would be an expensive problem for the classical primal-dual interior point methods, and thus would take a long time to be solved even by today's state-of-the-art software packages.

The second reason lies in a potential application of the algorithm that we develop here. Recently, there has been a growing interest in problems with conic constraints and integer variables. See for instance Cezik and Iyengar [7] and Atamturk and Narayanan [2]. Many such problems arise in finance and

engineering. Solving the conic relaxation of these problems by the classical interior point methods might be disadvantageous because there is no clear strategy for a warm start after an integer cut is introduced in the branch-and-cut algorithm. The use of outer approximation of conic constraints, on the other hand, has been proven to be a successful approach to this class of problems. For instance, Vielma et al. [23] use the polyhedral approximation of Ben-Tal and Nemirovski [3] to take advantage of the warm start in their branch-and-bound algorithm. Or, Bonami et al. [4] and Abhishek et al. [1] use the idea of polyhedral approximation in their integer programming solvers.

Therefore, solving convex conic optimization problems by outer approximation constraint generation methods is not only beneficial in a class of large-scale problems as we explore in this paper, it could also be combined with branch-and-bound, and branch-and-cut methods to develop efficient algorithms for mixed integer conic programming.

We present our algorithm in the context of SILP and prove its theoretical convergence and complexity in this general setting. Consider the following problem:

$$\max \left\{ b^T y : a_\omega^T y \leq c_\omega, \omega \in \Omega \right\}, \tag{1}$$

where $b \in \mathbb{R}^m$, $a_\omega \in \mathbb{R}^m$, and $c_\omega \in \mathbb{R}^1$, for $\omega \in \Omega$, a compact set. Problem (1) is called semi-infinite linear programming. This problem has been well-studied in the literature and has many applications in engineering and management. See Goberna and Lopez [11] for a theoretical survey and Lopez and Still [16] for a more recent survey on this topic.

We propose a build-up technique to solve problem (1), and show that a finite number of iterations are required to obtain an ε-optimal solution. Let

$$\mathcal{F}_\Omega = \{y \in \mathbb{R}^m : a_\omega^T y \leq c_\omega, \omega \in \Omega\},$$

the feasible region of problem (1) be compact. Consider a discretization of problem (1), where the feasible region is an outer approximation of $\mathcal{F}_\Omega$, and is defined by a finite number of constraints:

$$\max \left\{ b^T y : A^T y \leq c \right\}, \tag{2}$$

where $A \in \mathbb{R}^{m \times n}$ is a full row rank matrix composed of column vectors a_i's and $c \in \mathbb{R}^n$ is composed of scalars c_i's. Problem (2) is a relaxation of the

original problem. Let us call it the dual problem. The corresponding primal problem reads

$$\min \left\{ c^T x : Ax = b, x \geq 0 \right\}. \tag{3}$$

The main idea of our algorithm is as follows: We start from problem (2) with only an artificial box constraint whose bounds (RHS) is dynamically updated. Using a point in the vicinity of the central path of problem (2), multiple violated *(deep)* constraints from $\mathcal{F}_\Omega$ are identified. The feasible region of the dual problem is updated by adding the violated constraints and the barrier function is simultaneously updated by reducing the barrier parameter. This is equivalent to adding columns to primal problem (3). Then the strict feasibility for the new feasible region is recovered and the central path is updated. This process continues until the barrier parameter is small enough, i.e., the duality gap approaches to zero.

There are many algorithms in the literature based on cutting plane methods for SILP. See for instance Ferris and Philpott [9], Wu et al. [26], Li et al. [15], and Luo et al. [17].

The method that we describe in this paper is a variant of Luo et al. [17] with main differences both from the theoretical and implementation viewpoints. There are three main theoretical enhancements: first, our algorithm adds violating constraints with no changes to the right hand side. In [17] when a violating constraints is identified, it is relaxed by changing its right hand side to make the current $\mu-$center strictly feasible, which of course results in loss of information. We keep the violated constraints as deep as they are. Second, we extend the analysis to the case where multiple violating constraints are added simultaneously instead of adding one constraint at a time. Finally, at each iteration we update the barrier parameter together with updating the feasible region in the same step. All of these modifications contribute to the efficiency of the method as documented in the implementation section of this paper. We implement our algorithm to solve SOCO problems and show that our method outperforms primal-dual interior point methods on a class of large-scale problems.

We derive two theoretical complexity results. After adding p violated constraints and simultaneously updating the centering parameter μ by $\mu^+ = (1-\eta)\mu$ where $\eta = \frac{1}{9\sqrt{2m}}$, we show that only $O(p\log(p+1))$ Newton steps are required to obtain a point in the vicinity of the new μ^+-center. We also show that the constraint generation algorithm stops with an ε-solution to

the SILP problem after adding at most

$$O(\frac{m^2\hat{p}^2}{\delta^2}e^{3\sqrt{m}/\varepsilon})$$

constraints, where δ is the radius of the largest full dimensional ball contained in $\mathcal{F}_\Omega$, and $\hat{p}$ is the maximum number of constraints added simultaneously.

We test our algorithm by implementing it on two classes of problems: classical SILP and SOCO problems. The classical SILP problems, selected from the literature, are used to illustrate the convergence behavior of our algorithm in terms of the number of iterations that it takes for the upper and lower bounds to approach the optimal value line with a high precision.

On second-order cone optimization we use randomly generated problems and compare our results with that of SeDuMi to show that our constraint generation algorithm outperforms the classical primal-dual interior point methods on a certain class of large scale optimization problems. These results, together with the computational results of Glineur [10] that shows that the primal-dual interior point methods outperform Ben-Tal and Nemirovski [3] approximation on SOCO, suggest that our constraint generation method could be an alternative to polyhedral approximation in mixed integer conic programming.

The paper is organized as follows: Section 2 presents preliminaries and some technical lemmas that are needed throughout this paper. In Section 3 we describe our constraint generation algorithm in detail. Complexity of recovering the $\mu-$center and complexity and convergence of the algorithm are given in Sections 4 and 5 respectively. Finally, in Section 6 we present our numerical results and application of this method in second-order cone optimization.

2 Preliminaries

We indicate the primal, dual and primal-dual feasible regions of the discretization problem by $\mathcal{F}_p$, $\mathcal{F}_d$, and $\mathcal{F}$ respectively:

$$\mathcal{F}_p = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\},$$

$$\mathcal{F}_d = \{s \in \mathbb{R}^n : A^T y + s = c, s \geq 0\},$$

$$\mathcal{F} = \mathcal{F}_p \times \mathcal{F}_d.$$

Let $\mu > 0$ be the barrier parameter, the corresponding barrier functions read

$$\varphi_p(x,\mu) := \frac{c^T x}{\mu} - \sum_{i=1}^n \log x_i,$$

$$\varphi_d(s,\mu) := \frac{-b^T y}{\mu} - \sum_{i=1}^n \log s_i,$$

$$\varphi(x,s,\mu) := \frac{x^T s}{\mu} - \sum_{i=1}^n \log x_i s_i.$$

Due to the one-to-one correspondence between y and s in $\mathcal{F}_d$, we drop the argument y from the barrier function. The unique minimizer of $\varphi(x,s,\mu)$ over $\mathcal{F}$ denoted by $(x(\mu), s(\mu))$, is a point on the central path and satisfies the primal-dual feasibility and the centering condition $xs = \mu e$, where xs is the Hadamard product of x and s, i.e., it is an n-vector composed of $x_i s_i$, and $e \in \mathbb{R}^n$ is the vector with all its components equal to 1. We also call this point the μ-center. For the μ-center $(x(\mu), s(\mu))$, one has

$$\varphi(x(\mu), s(\mu), \mu) = n - n \log \mu.$$

A θ-approximate μ-center $(\bar{x}, \bar{s})$ is a point in the vicinity of the central path that satisfies

$$A^T \bar{y} + \bar{s} = c, \quad A\bar{x} = b, \quad \left\| \frac{\bar{x}\bar{s}}{\mu} - e \right\| \leq \theta < 1. \tag{4}$$

We now state some technical lemmas that are needed throughout this paper. The proofs of those lemmas that are not given here can be found in interior point methods books, such as Roos et al. [20], Ye [24], and den Hertog [14].

Lemma 1 *Let $z \in \mathbb{R}^n$, and $\|z\| < 1$. Then*

$$\phi(\|z\|) \leq \psi(z) \leq \phi(-\|z\|),$$

where $\psi(z) = e^T z - \sum_{j=1}^n \log(1 + z_j)$, and $\phi(\alpha) = \alpha - \log(1+\alpha)$.

Lemma 2 *If $z \in \mathbb{R}^n$ and $\|z\|_\infty < 1$, then*

$$e^T z - \frac{\|z\|^2}{2(1 - \|z\|_\infty)} \leq \sum_{j=1}^n \log(1 + z_j) \leq e^T z.$$

Lemma 3 *Let $(\bar{x}, \bar{s})$ be a θ-approximate μ-center. Then*

$$n - \theta\sqrt{n} \leq \frac{\bar{x}^T \bar{s}}{\mu} \leq n + \theta\sqrt{n}.$$

Moreover if $\mu^+ = (1-\eta)\mu$ with $0 < \eta < 1$, then

$$\left\| \frac{\bar{x}\bar{s}}{\mu^+} - e \right\| \leq \frac{1}{1-\eta}(\theta + \eta\sqrt{n}).$$

Corollary 4 *For $n \geq 2$, $\eta = \frac{1}{9\sqrt{n}}$, and arbitrary $\theta \leq 1/4$, one has*

$$\left\| \frac{\bar{x}\bar{s}}{\mu^+} - e \right\| \leq 0.40.$$

3 Constraint generation algorithm

In this section we present our constraint generation algorithm for solving problem (1). We make the following assumptions:

Assumption 1 *The index set Ω is compact and the mappings $t \longrightarrow a_t$ and $t \longrightarrow c_t$ are continuous in t.*

Assumption 2 *The feasible region $\mathcal{F}_\Omega$ contains an δ-radius full dimensional ball.*

Assumption 3 *$\mathcal{F}_\Omega$ is contained in the unit cube $[0,1]^m$, and all m-vectors b and a_t are normalized.*

Assumption 1 is made to ensure that the optimal solution of the constraint generation algorithm coincides with that of problem (1) (see Lemma 6). Assumption 2 is needed to establish a bound on the number of constraints, and Assumption 3 is a scaling assumption that will help to keep the complexity bound simple.

We now describe the algorithm. Let $\bar{y}$ be a point in the vicinity of the central path of $\mathcal{F}_d$ (see (4)) and $\bar{a}_j^T y \leq \bar{c}_j$, for $j = 1, \ldots, p$ be p constraints in $\mathcal{F}_\Omega$ such that $\bar{c}_j < \bar{a}_i^T \bar{y}$. The feasible region of the updated discretization therefore reads

$$\mathcal{F}_d^+ = \{s \in I\!R_+^n, r \in I\!R_+^p : A^T y + s = c, \bar{A}^T y + r = \bar{c}\}$$

where $\bar{A} \in \mathbb{R}^{m \times p}$ is composed of the p column vectors $\bar{a}_i$'s and $\bar{c} = (\bar{c}_1; \ldots; \bar{c}_p)$. Let $\mu^+ = (1-\eta)\mu$ be the updated barrier parameter for a later-specified value $0 < \eta < 1$. The task is now to find a point in the vicinity of the central path of the updated discretization, close to the μ^+-center of $\mathcal{F}_d^+$. However, since $\bar{c} < \bar{A}^T\bar{y}$, then $\bar{A}^T y \leq \bar{c}$ are deep constraints for $\mathcal{F}_d$, and thus the current point $\bar{y}$ is not a feasible point of $\mathcal{F}_d^+$. Therefore we first need to derive a strictly feasible point for $\mathcal{F}_d^+$. Let

$$\bar{t} = \arg\min\{\frac{p}{2} t^T V t - \sum_{i=1}^{p} \log t_i\}, \tag{5}$$

where $V = \bar{A}^T(A\bar{X}^2A^T)^{-1}\bar{A}$, where X is a diagonal $n \times n$ matrix made up of vector x. Define

$$\bar{d} = p(\bar{A}^T\bar{y} - \bar{c})\bar{t}. \tag{6}$$

Notice that since $\bar{A}^T\bar{y} - \bar{c} > 0$, and $\bar{t} > 0$, then $\bar{d} > 0$. Let $\alpha < 1 - \theta$ is fixed. We consider two cases:

1. *Moderately deep constraints*: $\bar{d} < \alpha e$. In this case we show that all violating constraints cross the Dikin's ellipsoid around $\bar{y}$ and the dual feasibility can be recovered using the current point $\bar{y}$.

2. *Very deep constraints*: There exists a constraint for which $\bar{d}_i \geq \alpha$. In this case the dual feasibility cannot be recovered. We show that one can recover feasibility in the primal space

   $$\mathcal{F}_p^+ = \{x \in \mathbb{R}_+^n, t \in \mathbb{R}_+^p : Ax + \bar{A}t = b\}$$

   and obtain the new μ^+-center using the primal barrier function.

Lemma 5 *Let $\mathcal{F}_p$ and $\mathcal{F}_d$ be the primal and dual feasible regions of the discretization problem respectively, μ be the barrier parameter, and $(\bar{x}, \bar{s})$ be a point in the vicinity of the central path that satisfied (4). Let p violating constraints $\bar{A}^T y \leq \bar{c}$ are added to $\mathcal{F}_d$. Then for $\bar{d}_i < \alpha < 1 - \theta$*

$$x^+ = (\bar{x} + \alpha\Delta x; \alpha\bar{t})$$

where $\Delta x = -\bar{X}^2A^T(A\bar{X}^2A^T)^{-1}\bar{A}\bar{t}$, is strictly feasible for $\mathcal{F}_p^+$; and

$$s^+ = (\bar{s} + \alpha\Delta s; \bar{r})$$

where $\Delta s = A^T(A\bar{X}^2A^T)^{-1}\bar{A}\bar{t}$, and

$$\bar{r} = \frac{1}{p}(\alpha e - \bar{d})\bar{t}^{-1}, \tag{7}$$

where the p-vector $\bar{t}^{-1}$ is the component-wise inverse of vector $\bar{t}$, is strictly feasible for $\mathcal{F}_d^+$.

Proof. Goffin and Vial [13] prove a similar lemma for their cutting plane algorithm where $\mu = 1$ and $\bar{d} = 0$. The directions Δx and Δs defined in this lemma are similar to those of [13]. Therefore their proof, to some extend, remains valid here. In particular $A(\Delta x) + \bar{A}\bar{t} = 0$ which is obtained by construction. Also the strict feasibility of the updating directions $\bar{x}+\alpha\Delta x > 0$ and $\bar{s} + \alpha\Delta s > 0$ are obtained by Lemma 7 below and the fact that $\alpha < 1 - \theta$.

We prove that $\bar{A}^T y^+ + \bar{r} = \bar{c}$ and $\bar{r} > 0$. Notice that $A^T(\bar{y}+\Delta y)+\bar{s}+\Delta s = c$. Thus $A^T\Delta y = -\Delta s$ and $\Delta y = -(A\bar{X}^2A^T)^{-1}\bar{A}\bar{t}$. Therefore

$$\bar{A}^T y^+ + \bar{r} = \bar{A}^T\bar{y} + \alpha\bar{A}^T\Delta y + \bar{r} = \bar{A}^T\bar{y} - \alpha V\bar{t} + \bar{r}$$

and from the KKT optimality conditions of problem (5), we have

$$\begin{aligned} \bar{A}^T y^+ + \bar{r} &= \bar{A}^T\bar{y} - \frac{\alpha}{p}\bar{t}^{-1} + \frac{1}{p}(\alpha e - \bar{d})\bar{t}^{-1} \\ &= \bar{A}^T\bar{y} - \frac{1}{p}\bar{d}\bar{t}^{-1} \\ &= \bar{c}. \end{aligned}$$

On the other hand since $\bar{d} < \alpha e$, thus $\bar{r} > 0$. ■

Lemma 5 shows that if the violating constraints are moderately deep then the Newton's method can be initiated from x^+ and s^+ to obtain a point in the vicinity of the new central path. In the next section we derive a bound on the number of Newton steps required to update the μ^+-center.

In case 2, when there is at least one very deep inequality, the dual feasibility cannot be recovered because it is not clear how far the constraint is away from the Dikin's ellipsoid. In this situation one can still recover the primal feasibility using x^+ and the Newton's method can be applied to the primal space to update the μ^+-center. This procedure is repeated until the barrier parameter μ falls within the desired accuracy rate.

The next lemma due to Luo, et al. [17] shows that the constrain generation algorithm reaches the optimal solution of problem (1).

Lemma 6 *Let $\varepsilon > 0$ be given. Under Assumption 1, if $\bar{y} \in \mathcal{F}_\Omega$ is in the vicinity of $\mu < \frac{\varepsilon}{n+\sqrt{n}}$, then $\bar{y}$ is an ε-maximizer of problem (1).*

We now formally present our algorithm.

Algorithm 1 *Initiate* $\mathcal{F}_d^0 = [0,1]^m$, $\mu_0 = 1$, $y^0 = \frac{1}{2}e$, $s^0 = \frac{1}{2}e$, $n_0 = 2m$, *and* $\eta_0 = \frac{1}{9\sqrt{2m}}$, $\theta = 0.25$ *and* $k = 1$.

While $(n_k + \sqrt{n_k})\mu_k \geq \varepsilon$ **do**

Step 1. *Identify* p_k *deep constraints* $(\bar{A}^k)^T y \leq \bar{c}^k$, *in* $\mathcal{F}_\Omega$ *such that* $\bar{c}^k > (\bar{A}^k)^T y^k$.

Step 2. *Update* $n_k = n_{k-1} + p_k$, $\eta_k = \frac{1}{9\sqrt{n_k}}$, $\mu_k = (1 - \eta_k)\mu_{k-1}$

$$A^k = [A^{k-1} \;\; \bar{A}^k], \text{ and } c^k = (c^{k-1}; \bar{c}^k)$$

Step 3. *Compute* $\bar{t}$ *from (5) and* $\bar{d}$ *from (6). If* $\bar{d} < \alpha e$, *then use* s^+ *to start a dual Newton procedure to obtain* s^k *and* $x^k := x(s^k)$ *in the vicinity of the* μ_k*-center of* $\mathcal{F}_d^k$.

Step 3. *Otherwise use* x^+ *to start a primal Newton procedure to obtain* x^k *and* $s^k := s(x^k)$ *in the vicinity of the* μ_k*-center of* $\mathcal{F}_p^k$.

Step 4. *k=k+1.*

End

4 Complexity of recovering the μ-center

In this section we derive a bound on the number of Newton steps that is required to obtain a point in the vicinity of the μ^+-center when all violating constraints are moderately deep.

First this lemma from Goffin and Vial [13].

Lemma 7 *For directions* Δx *and* Δs *in Lemma 5, we have*

$$\|\bar{X}^{-1}\Delta x\| \leq \frac{1}{1-\theta} \quad \text{and} \quad \|\bar{S}^{-1}\Delta s\| \leq \frac{1}{1-\theta}.$$

We also need the following technical lemma:

Lemma 8 *Let* $(\bar{x}, \bar{s})$ *be a* θ*-approximate* μ*-center. Then*

1. $\varphi(\bar{x}, \bar{s}, \mu) \leq \varphi(x(\mu), s(\mu), \mu) + \frac{\theta^2}{2(1-\theta)}$.
2. $\varphi(\bar{x}, \bar{s}, \mu^+) + n\log(1-\eta) \leq \varphi(x(\mu), s(\mu), \mu) + \nu(n, \eta, \theta)$

3. $\varphi(\bar{x},\bar{s},\mu^+) \le \varphi(x(\mu^+),s(\mu^+),\mu^+) + \nu(n,\eta,\theta)$

where

$$\nu(n,\eta,\theta) = n\log(1-\eta) + \frac{\eta(n+\theta\sqrt{n})}{1-\eta} + \frac{\theta^2}{2(1-\theta)}.$$

Proof. In view of Lemma 2, the first inequality is straightforward. Let us prove the second inequality. Since $(\bar{x},\bar{s})$ is in the vicinity of the μ-center, from Lemma 2

$$\begin{aligned}\varphi(\bar{x},\bar{s},\mu^+) &= \frac{\bar{x}^T\bar{s}}{\mu^+} - n\log\mu - \sum_{i=1}^{n}\log\frac{\bar{x}_i\bar{s}_i}{\mu} \\ &\le \frac{\bar{x}^T\bar{s}}{\mu^+} - \frac{\bar{x}^T\bar{s}}{\mu} + n - n\log\mu + \frac{\theta^2}{2(1-\theta)}.\end{aligned}$$

The second inequality follows from Corollary 4 and

$$\frac{\bar{x}^T\bar{s}}{\mu^+} - \frac{\bar{x}^T\bar{s}}{\mu} = \frac{\eta\bar{x}^T\bar{s}}{(1-\eta)\mu} \le \frac{\eta(n+\theta\sqrt{n})}{1-\eta}.$$

The third inequality implies from the second one. ■

Notice that the bounds on the primal-dual barrier function in Lemma 8 are also valid for the primal and the dual barrier functions. The following corollary simplifies these bounds for some given values.

Corollary 9 *For $n \ge 2$, $\eta = \frac{1}{9\sqrt{n}}$, and arbitrary $\theta \le 1/4$*

1. $\varphi(\bar{x},\bar{s},\mu) \le \varphi(x(\mu),s(\mu),\mu) + 0.05.$
2. $\varphi(\bar{x},\bar{s},\mu^+) + n\log(1-\eta) \le \varphi(x(\mu),s(\mu),\mu) + 0.10,$
3. $\varphi(\bar{x},\bar{s},\mu^+) \le \varphi(x(\mu^+),s(\mu^+),\mu^+) + 0.10.$

Proof. Since $0 < \eta < 1$

$$\begin{aligned}\nu(n,\eta,\theta) &\le -n\eta + \frac{\eta(n+\theta\sqrt{n})}{1-\eta} + \frac{\theta^2}{2(1-\theta)} \\ &= \frac{\theta\eta\sqrt{n} + n\eta^2}{1-\eta} + \frac{\theta^2}{2(1-\theta)}\end{aligned}$$

For $\eta = \frac{1}{9\sqrt{n}}$ and $n \ge 2$

$$\nu(n,\eta,\theta) \le \frac{12}{99}(\theta + 1/9) + \frac{\theta^2}{2(1-\theta)}.$$

The proof follows from $\theta \leq 0.25$. ■

Now we establish an upper bound on the primal barrier function at (x^+, μ^+).

Lemma 10 *Let $(\bar{x}, \bar{s})$ be a θ-approximate μ-center, $\mu^+ = (1 - \frac{1}{9\sqrt{n}})\mu$, and all violating constraints are moderately deep, i.e., $\bar{d} < \alpha e$. Then for $\alpha < 1-\theta$ and $0 < \theta \leq 1/4$*

$$\varphi_p^+(x^+, \mu^+) \leq \varphi_p(\bar{x}, \mu^+) - \alpha - \log(1 - \frac{\alpha}{1-\theta}) - e^T\bar{d} - \sum_{j=1}^{p} \log \alpha\bar{t}_j + 0.40.$$

Proof. The primal barrier function at (x^+, μ^+) reads

$$\varphi_p^+(x^+, \mu^+) = \frac{(c^+)^T x^+}{\mu^+} - \sum_{j=1}^{n} \log \bar{x}_j(1 + \alpha\Delta x_j/\bar{x}_j) - \sum_{j=1}^{p} \log \alpha\bar{t}_j.$$

Since $\|\bar{X}^{-1}\Delta x\| \leq \frac{1}{1-\theta}$ and $\alpha < 1 - \theta$, from Lemma 1

$$\varphi_p^+(x^+, \mu^+) \leq \frac{(c^+)^T x^+}{\mu^+} - \sum_{j=1}^{n} \log \bar{x}_j \tag{8}$$

$$-\alpha e^T\bar{X}^{-1}\Delta x - \frac{\alpha}{1-\theta} - \log(1 - \frac{\alpha}{1-\theta}) - \sum_{j=1}^{p} \log \alpha\bar{t}_j.$$

On the other hand

$$\frac{(c^+)^T x^+}{\mu^+} - \alpha e^T\bar{X}^{-1}\Delta x = \frac{c^T\bar{x}}{\mu^+} + \alpha\left[\frac{c^T\Delta x}{\mu^+} + \frac{\bar{c}^T\bar{t}}{\mu^+} - e^T\bar{X}^{-1}\Delta x\right] \tag{9}$$

In view of (6)

$$\bar{c}^T\bar{t} = \bar{y}^T\bar{A}\bar{t} - \frac{\bar{t}^T\bar{D}\bar{t}^{-1}}{p},$$

where $\bar{D}$ is a diagonal matrix made up of $\bar{d}$. Thus the term in the brackets in (9) reads

$$\frac{\bar{s}^T\Delta x}{\mu^+} - e^T\bar{X}^{-1}\Delta x - \frac{e^T\bar{d}}{\mu^+},$$

or

$$(\frac{\bar{s}\bar{x}}{\mu^+} - e)^T(\bar{X}^{-1}\Delta x) - \frac{e^T\bar{d}}{\mu^+} \tag{10}$$

and using the Cauchy-Schwartz Inequality we have

$$(\frac{\bar{s}\bar{x}}{\mu^+} - e)^T(\bar{X}^{-1}\Delta x) \leq \|\frac{\bar{s}\bar{x}}{\mu^+} - e\|\|\bar{X}^{-1}\Delta x\|.$$

Now from (10), Corollary 4, Lemma 7 and the assumption that $\mu \leq 1$, we have

$$\frac{(c^+)^T x^+}{\mu^+} - \alpha e^T \bar{X}^{-1}\Delta x \leq \frac{c^T\bar{x}}{\mu^+} - e^T\bar{d} + 0.40.$$

The proof now follows from (8). ■

Notice that since $\bar{d} > 0$, the term $e^T\bar{d}$ can be eliminated from the bound in Lemma 10. We now bound the dual barrier function.

Lemma 11 *Let the assumptions of Lemma 10 be satisfied. Then*

$$\varphi_d^+(s^+, \mu^+) \leq \varphi_d(\bar{s}, \mu^+) - \alpha - \log(1 - \frac{\alpha}{1-\theta}) - \sum_{j=1}^{p} \log \bar{r}_j + 0.40.$$

Proof.

$$\begin{aligned}\varphi_d^+(s^+, \mu^+) &= \frac{-b^T y^+}{\mu^+} - \sum_{j=1}^{n+p} \log s_j^+ \\ &= \frac{-b^T\bar{y}}{\mu^+} - \frac{\alpha b^T\Delta y}{\mu^+} - \sum_{j=1}^{n} \log \bar{s}_j(1 + \alpha\Delta s_j/\bar{s}_j) - \sum_{j=1}^{p} \log \bar{r}_j.\end{aligned}$$

Now since $\alpha < 1 - \theta$, in view of Lemma 1 and Lemma 7 we have

$$\varphi_d^+(s^+, \mu^+) \leq$$
$$\varphi_d(\bar{s}, \mu^+) + \frac{\alpha\bar{x}^T\Delta s}{\mu^+} - \alpha e^T\bar{S}^{-1}\Delta s - \frac{\alpha}{1-\theta} - \log(1 - \frac{\alpha}{1-\theta}) - \sum_{j=1}^{p} \log \bar{r}_j.$$

On the other hand

$$\begin{aligned}\frac{\bar{x}^T\Delta s}{\mu^+} - e^T\bar{S}^{-1}\Delta s &\leq |(\frac{\bar{x}\bar{s}}{\mu^+} - e)^T\bar{S}^{-1}\Delta s| \\ &\leq \|\frac{\bar{x}\bar{s}}{\mu^+} - e\|\|\bar{S}^{-1}\Delta s\| \\ &\leq \frac{0.40}{1-\theta}\end{aligned}$$

where the last inequality is due to Corollary 4 and Lemma 7 ■

We now present the main result of this section.

Theorem 12 *Let $(\bar{x}, \bar{s})$ be a θ-approximate μ-center, $\mu^+ = (1 - \frac{1}{9\sqrt{n}})\mu$, and all violating constraints are moderately deep. Moreover let $\bar{d} < (\alpha/2)e$. Then for $\alpha < 1 - \theta$ and $0 < \theta \leq 1/4$*

$$\varphi^+(x^+, s^+, \mu^+) - \varphi^+(x(\mu^+), s(\mu^+), \mu^+) \leq p \log p + \xi(p, \theta, \alpha),$$

where

$$\xi(p, \theta, \alpha) = 1.0 - 2\alpha - 2\log(1 - \frac{\alpha}{1-\theta}) - p(1 + \log\frac{\alpha^2}{2}).$$

Proof. Adding inequalities in Lemma 10 and Lemma 11 gives

$$\begin{aligned}\varphi^+(x^+, s^+, \mu^+) \leq \varphi(\bar{x}, \bar{s}, \mu^+)+ \\ +0.80 - 2\alpha - 2\log(1 - \frac{\alpha}{1-\theta}) - \sum_{j=1}^{p} \log \alpha \bar{t}_j \bar{r}_j.\end{aligned}$$

Now in view of (7)

$$\begin{aligned}\sum_{j=1}^{p} \log \alpha \bar{t}_j \bar{r}_j &= \sum_{j=1}^{p} \log \frac{\alpha}{p}(\alpha - \bar{d}_j) \\ &= p \log \alpha/p + \sum \log(\alpha - \bar{d}_j) \\ &\geq p \log \frac{\alpha^2}{2p}\end{aligned}$$

where the inequality is valid because $\bar{d}_j < \alpha/2$, for $j = 1, \ldots, p$. Thus

$$\begin{aligned}\varphi^+(x^+, s^+, \mu^+) \leq \varphi(\bar{x}, \bar{s}, \mu^+)+ \\ +0.80 - 2\alpha - 2\log(1 - \frac{\alpha}{1-\theta}) + p \log \frac{2p}{\alpha^2}\end{aligned}$$

and from Lemma 8 and Corollary 9

$$\begin{aligned}\varphi^+(x^+, s^+, \mu^+) \leq \varphi(x(\mu^+), s(\mu^+), \mu^+)+ \\ +1.0 - 2\alpha - 2\log(1 - \frac{\alpha}{1-\theta}) + p \log \frac{2p}{\alpha^2}.\end{aligned} \tag{11}$$

On the other hand

$$\begin{aligned}\varphi(x(\mu^+), s(\mu^+), \mu^+) &= n - n \log \mu^+ \\ &= n + p - (n + p) \log \mu^+ - (p - p \log \mu^+) \\ &= \varphi^+(x(\mu^+), s(\mu^+), \mu^+) - p + p \log \mu^+ \\ &\leq \varphi^+(x(\mu^+), s(\mu^+), \mu^+) - p.\end{aligned}$$

The proof follows from (11) now. ■

Theorem 12 shows that after adding p moderately deep constraints and simultaneously updating μ, only $O(p\log(p+1))$, Newton steps are required to obtain a point in the vicinity of the new μ^+-center. We remark that the assumption $\mu \leq 1$ has been made only to simplify this bound. If $\mu > 1$, the complexity changes to $O(p\log(\mu p+1))$.

5 Complexity analysis and convergence

The complexity analysis and convergence of Algorithm 1 can be done in general case. Let $\bar{A}^T y \leq \bar{c}$ be the p violating constraints such that $\bar{c} < \bar{A}^T \bar{y}$. In this section, we do not differentiate between moderate and very deep constraints. All constraints are deep.

Lemma 13 *For $n \geq 2$, $\eta = \frac{1}{9\sqrt{n}}$, $\theta = 0.25$ and $\alpha = 0.50$*

$$\varphi_d^+(s(\mu^+,\mu^+) \geq \varphi_d(s(\mu),\mu) - p\log p - \sum_{i=1}^{p} \log v_i^{1/2}, \tag{12}$$

where $v \in \mathbb{R}^p$ is composed of the diagonal elements of matrix V defined in Section 3.

Proof. First observe that

$$\begin{aligned} \varphi_d^+(s(\mu^+),\mu^+) &= n+p-(n+p)\log\mu^+ - \varphi_p^+(x(\mu^+),\mu^+) \\ &\geq n+p-(n+p)\log\mu^+ - \varphi_p^+(x^+,\mu^+), \end{aligned}$$

and from Lemma 10

$$\begin{aligned} \varphi_d^+(s(\mu^+),\mu^+) &\geq n - n\log\mu - n\log(1-\eta) - \varphi_p(\bar{x},\mu^+) \\ &+p - p\log\mu^+ + \alpha + \log(1-\frac{\alpha}{1-\theta}) + e^T\bar{d} + \sum_{j=1}^{p}\log\alpha\bar{t}_j - 0.40 \end{aligned}$$

Now from Corollary 9

$$\begin{aligned} \varphi_d^+(s(\mu^+),\mu^+) &\geq n - n\log\mu - \varphi_p(x(\mu),\mu) \\ &+p - p\log\mu^+ + \alpha + \log(1-\frac{\alpha}{1-\theta}) + e^T\bar{d} + \sum_{j=1}^{p}\log\alpha\bar{t}_j - 0.50 \end{aligned}$$

Thus

$$\varphi_d^+(s(\mu^+),\mu^+) \geq \varphi_d(s(\mu),\mu) + \sum_{j=1}^{p} \log \bar{t}_j$$
$$+p + \alpha + \log(1 - \frac{\alpha}{1-\theta}) + e^T\bar{d} + p\log\alpha - 0.50.$$

On the other hand Goffin and Vial [13] prove that

$$\sum_{j=1}^{p} \log \bar{t}_j \geq -p\log p - \sum_{j=1}^{p} \log v_j^{1/2}.$$

Therefore

$$\varphi_d^+(s(\mu^+),\mu^+) \geq \varphi_d(s(\mu),\mu) - p\log p - \sum_{j=1}^{p} \log v_j^{1/2}$$
$$+p + \alpha + \log(1 - \frac{\alpha}{1-\theta}) + e^T\bar{d} + p\log\alpha - 0.50.$$

The proof follows by replacing $\theta = 0.25$ and $\alpha = 0.50$. ■

Lemma 13 establishes a bound on the optimal value of the updated dual barrier function after adding p deep constraints and updating μ. Notice that Inequality (12) derived in this lemma is the exact same inequality for central cuts (see Ye [25] and Goffin and Vial [13]). The reason is that here we simply ignore $e^T\bar{d} > 0$ from the bound because we do not have any information on the depth of the cut. However, in practice having deep constraints are beneficial in the sense that a feasible solution to the original problem is reached faster when constraints are added with no changes to their right hand side.

Lemma 14 *Let at the kth iteration of the algorithm,* $\mu_k \leq \mu_0 := 1$, $n_k := n_0 + n_p := 2m + \sum_{i=1}^{k} p_i$, *and*

$$p = \max\{p_i, i = 1, \ldots, k\}$$

where p_i *are the number of deep constraints added at iteration* i. *Then*

$$-\frac{\sqrt{m}}{\mu_k} + n_k \log\delta \leq -\varphi_d^k(s(\mu_k),\mu_k) \leq \frac{\sqrt{m}}{2} + 2m\log\frac{1}{2} + n_p\log(p+1) + \sum_{i=1}^{n_p} \log v_i^{1/2}$$

where δ *is the radius of the full dimensional ball contained in* $\mathcal{F}_\Omega$.

Proof. The right hand side inequality follows from Lemma 13

$$\varphi_d^k(s(\mu_k),\mu_k) \geq \varphi_d^0(s(\mu_0),\mu_0) - \sum_{i=1}^{n_p} p_i \log(p+1) - \sum_{i=1}^{n_p} \log v_i^{1/2}$$

and the fact that

$$\begin{aligned}
\varphi_d^0(s(\mu_0),\mu_0) &= \frac{-b^T y}{\mu_0} - \sum_{j=1}^{2m} \log s_j(\mu_0) \\
&= \frac{-b^T e}{2} - 2m \log \frac{1}{2} \\
&\geq \frac{-\sqrt{m}}{2} - 2m \log \frac{1}{2}
\end{aligned}$$

where the inequality is due to Assumption 3 that $\|b\| \leq 1$.

To prove the left hand side inequality, let (y^c, s^c) be the center of the δ-ball. Then $s_i^c = c_i - a_i^T y^c \geq \delta$, for all $i = 1, \ldots, 2m + n_p$. Also from Assumption 2 since $\mathcal{F}_\Omega$ is contained in the unit cube, $\|y\|_\infty \leq 1$. Therefore at the kth iteration

$$\varphi_d^k(s^c,\mu_k) = \frac{-b^T y^c}{\mu_k} - \sum_{i=1}^{n} \log s_i^c \leq \frac{\sqrt{m}}{\mu_k} - (n_p + 2m) \log \delta.$$

The lemma now follows from $\varphi_d^k(s(\mu_k),\mu_k) \leq \varphi_d^k(s^c,\mu_k)$. ■

The following lemma is due to Ye [25]:

Lemma 15 *Let $p \leq m$. Then*

$$\sum_{i=1}^{n_p} \log v_i \leq 2m^2 \log(1 + \frac{n_p}{8m^2}).$$

We now present the main result of this paper:

Theorem 16 *Let $1 \leq p_i \leq p \leq m$, for all i. Then the constraint generation algorithm stops with an ε-solution to the semi-infinite linear programming after adding at most*

$$O(\frac{m^2 p^2}{\delta^2} e^{3\sqrt{m}/\varepsilon})$$

constraints.

Proof. From Lemma 14

$$-\frac{\sqrt{m}}{2} - \frac{\sqrt{m}}{\mu_k} + n_k \log \delta - n_p \log(p+1) \leq 2m \log \frac{1}{2} + \sum_{i=1}^{n_p} \log v_i^{1/2}$$

since $p \geq 1$ and $n_k \geq n_p$

$$\begin{aligned} -\frac{3\sqrt{m}}{2n_k\mu_k} + \log(\frac{\delta}{p+1}) &\leq \frac{1}{2n_k}(2m \log \frac{1}{4} + \sum_{i=1}^{n_p} \log v_i) \\ &\leq \frac{1}{2} \log \frac{\frac{m}{2} + \sum_{i=1}^{n_p} v_i}{n_k} \qquad (13) \\ &\leq \frac{1}{2} \log \frac{\frac{m}{2} + 2m^2 \log(1 + \frac{n_p}{8m^2})}{n_k} \qquad (14) \end{aligned}$$

where (13) is due to the Geometric Mean Inequality and (14) is due to Lemma 15. Notice that Inequality (14) is valid at each iterations of the constraint generation algorithm. Therefore a feasible solution in the δ-ball is obtained when this inequality is violated. That is

$$\log \frac{\frac{m}{2} + 2m^2 \log(1 + \frac{n_p}{8m^2})}{n_k} \leq -\frac{3\sqrt{m}}{n_k\mu_k} + \log(\frac{\delta}{p+1})^2.$$

On the other hand from Lemma 6, an ε-solution to the original problem is reached when $\mu_k \leq \frac{\varepsilon}{n_k + \sqrt{n_k}}$. Therefore an ε-solution is achieved when

$$\log \frac{\frac{m}{2} + 2m^2 \log(1 + \frac{n_p}{8m^2})}{n_k} \leq -\frac{3\sqrt{m}}{\varepsilon} + \log(\frac{\delta}{p+1})^2$$

or

$$\frac{\frac{m}{2} + 2m^2 \log(1 + \frac{n_p}{8m^2})}{n_k} \leq \frac{e^{-3\sqrt{m}/\varepsilon^2}\delta^2}{(p+1)^2}.$$

The proof now follows from the above inequality. ■

6 Computational results

In this section we test our algorithm on two sets of problems. First we show the convergence behavior of the algorithm on some classical SILP problems selected from Coope and Watson [8], and then we show the power of our algorithm on a class of SOCO problems using randomly generated data.

All the test problems were done on a desktop computer using Intel(R) Core(TM)2 Quad CPU 2.66 GHz processor with 4 GB RAM.

In the following examples we solve an optimization problem of the form

$$\max \left\{ b^T y : g(y, \omega) \leq 0, \omega \in \Omega \right\}, \tag{15}$$

where $g(y, \omega)$ is a linear function of y for a given ω in the compact set Ω.

To solve this form of problems by our constraint generation algorithm, we need to convert problem (15) to the form of problem (1). To do this, at each iteration an oracle is used to discretize Ω and identify multiple violated constraints using a random search. The violated constraints are then added to the relaxation problem as new constraints and the μ^+-center is updated.

At each iteration of the algorithm, therefore, we deal with a relaxation problem (2), and its corresponding primal problem (3), which is a restricted form of the primal of the original problem.

Example 1 *Let* $b = (-1, -1/2, -1/3)^T$, $g(y, \omega) = \tan(\omega) - \sum_{i=1}^{m} y_i \omega^{i-1}$, *and* $\Omega = [0, 1]$.

Solving this problem using our interior point constraint generation algorithm yields the optimal solution $y^* = (0.089073; 0.423147; 1.0450756)$, and the optimal objective value $b^T y^* = -0.6490412$.

Figure 1 shows the convergence behavior of our algorithm for Example 1. In this figure we plot the objective values of the relaxed dual and restricted primal problems at the current $\mu-$center (y-axis) in each iteration (x-axis). The upper (lower) curve comes from evaluating $c^T x$ ($b^T y$), the objective function of the restricted primal (relaxed dual) problem, at the current $\mu-$center.

This figure illustrates that Algorithm 1 quickly approaches the optimal value with a reasonable duality gap. Observe that a good approximation of the optimal solution is achieved in less than 40 iterations. However, to get a high precision (10^{-8}) we let the algorithm run for about 90 iterations.

Notice that since the feasible region of problem (3) is also feasible for the primal of the original problem, therefore $c^T x$ at the $\mu-$center always gives an upper bound on the optimal objective value, that is, the upper curve never crosses the optimal value line. But this is not true for the lower curve. This curve is obtained by evaluating $b^T y$, the objective value of the dual problem at a feasible point of the relaxation. Therefore this point is not necessarily feasible for the original problem, which is why the lower curve may cross the optimal value line in the early iterates.

Example 2 $b = (-1, -1/2, -1/2, -1/3, -1/4, -1/3)^T$, $g(y, \omega) = e^{\omega_1^2 + \omega_2^2} - (y_1 + \omega_1 y_2 + \omega_2 y_3 + \omega_1^2 y_4 + \omega_1 \omega_2 y_5 + \omega_2^2 y_6)$, *and* $\Omega = [0, 1] \times [0, 1]$.

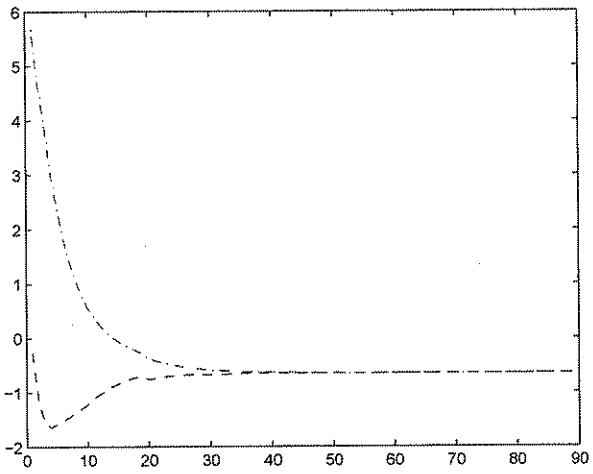

Figure 1: Convergence behavior of Algorithm 1 on Example 1 with 10^{-8} precision

The optimal solution of this problem is

$$y^* = (2.5782999, -4.106585, -4.0981235, 4.2450596, 4.5222404, 4.2370932)^T$$

and the optimal objective value is $b^T y^* = -2.4338899$. The convergence behavior of this problem is shown in Figure 2.

Example 3 $b = (-2, -4, -3)^T$, $g(y, \omega) = \sum_{i=1}^{3}(1 - y_i)h_i(\omega_1, \omega_2) - \frac{1}{2}$, $\Omega = [-1, 4] \times [-1, 4]$, *and*

$$\begin{array}{ll} h_1(\omega_1, \omega_2) = (1/\omega_1)(\exp((-1/\omega_1)(1 + (\omega_2 - 1)^2))) & \omega_1 > 0, \\ h_2(\omega_1, \omega_2) = (1/\omega_1)(\exp((-1/\omega_1)(2 + \omega_2^2/4)) & \omega_1 > 0, \\ h_3(\omega_1, \omega_2) = (1/(\omega_1 - 2)(\exp((-1/(\omega_1 - 2))(1 + (\omega_2 + 1)^2)) & \omega_1 > 2, \\ h_{1,2,3}(\omega_1, \omega_2) = 0 & \textit{elsewhere}\ . \end{array}$$

The optimal solution for this problem is

$$y^* = (1.5425641, -2.1014821, 0.9345579)^T$$

and the optimal objective value is $b^T y^* = 4.3862422$. The convergence behavior of this problem is illustrated in Figure 3.

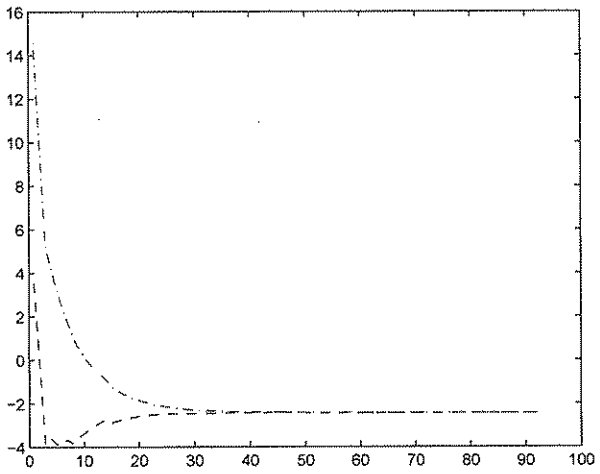

Figure 2: Convergence behavior of Algorithm 1 on Example 2 with 10^{-8} precision

Notice that in Figures 1–3 the lower and upper curves are not monotonically approaching each other when the current iterate is far from the optimal solution. This phenomenon is due to the fact that these bounds are computed by evaluating the objective functions of the relaxed dual problem and its corresponding primal problem at the current $\mu-$center. When violating constraints are identified, the feasible region of the relaxed dual is updated by adding new constraints. Since this problem is not solved to optimality, but evaluated at the μ^+-center, the value of the objective function is unpredictable at this point. However, as we get closer to the optimal solution of the original problem, these fluctuations reduce and the lower and upper curves become lower and upper bounds on the optimal objective value and monotonically approach the optimum value.

As the second, large-scale, test set, we choose to implement our algorithm to solve SOCO problems using randomly generated data. Consider the following SOCO:

$$\max_{y\in\mathbb{R}^m} \left\{ b^T y \,:\, l_b \leq y \leq u_b, (c_j - A_j^T y) \in \mathcal{L}^{n_j},\ j = 1, 2, \ldots, k \right\} \qquad (16)$$

where b is a non-zero vector in $\mathbb{R}^m$, $l_b < u_b$ are real vectors indicating lower

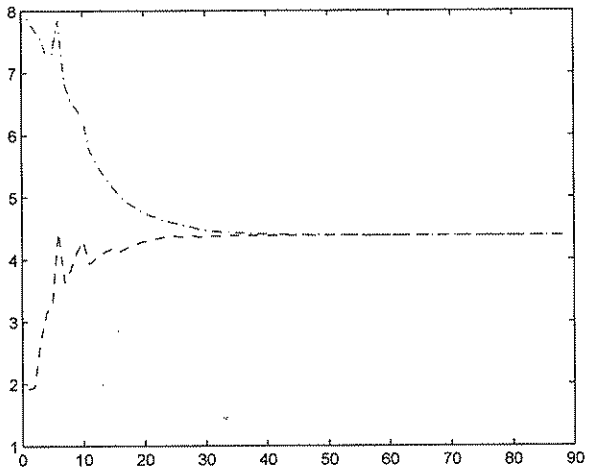

Figure 3: Convergence behavior of Algorithm 1 on Example 3 with 10^{-8} precision

and upper bounds of y respectively, and $\mathcal{L}^n$ is an $n-$dimensional *Lorentz Cone*, defined by:

$$\mathcal{L}^n = \left\{ s \in \mathbb{R}^n \, : \, \sqrt{s_2 + \cdots + s_n} \leq s_1 \right\}. \tag{17}$$

The bound constraints $l_b \leq y \leq u_b$ are added to ensure bounded solution. We use MATLAB function `randn.m` to generate data for matrices A_j and vectors c_j from a normal distribution with mean zero and standard deviation one. We let

$$c_{j1} = 2\sqrt{c_{j2}^2 + \cdots + c_{jn_j}^2}, \quad j = 1, \ldots, k$$

to ensure feasibility, and $n_j = \bar{n}$ for all j, where $n = \sum_{j=1}^{k} \bar{n}_j + 2m$.

At each iteration of the constraint generation algorithm an oracle is called to return an outer approximation of the violating second-order cone constraints. This is obtained by computing the gradient of the constraint functions at the current $\mu-$center. If no violating constraint is detected the algorithm is continued by updating the centering parameter. Our oracle uses a random search for identifying violated constraints. This technique works well when the number of cones (k) is relatively small. A more efficient

			Optimum value		CPU time (in second)	
k	$\bar{n}$	cuts	SILP	SeDuMi	SILP/Oracle	SeDuMi
3	1E+6	30	2.9913802	2.9913802	19/18	99
9	5E+5	31	2.9878005	2.9878005	26/24	260
27	1E+5	30	2.9751433	2.9751433	11/10	105
81	5E+4	31	2.9492360	2.9492360	14/13	147
243	1E+4	26	2.8790904	2.8790904	10/9	86
729	5E+3	32	2.8021491	2.8021491	13/11	135
2187	1E+3	31	2.5023178	2.5023178	11/10	158
6561	5E+2	28	2.3104156	2.3104156	31/27	248
19683	1E+2	33	1.7423363	1.7423363	110/101	115
59049	5E+1	29	1.2555162	1.2555162	861/854	1120

Table 1: Comparison of CPU time of SeDuMi and SILP implemented on randomly generated SOCO with $m = 3$ and different values of $\bar{n}$ and k.

technique is needed to detect violating constraints for problems with large number of conic constraints.

Tables 1 and 2 show the numerical results of this implementation. Each row shows a different random problem with characteristics given in the first two columns: k, the number of second-order cone constraints in problem (16) and $\bar{n}$, the size of each cone, respectively. The column under "cuts" illustrates the number of gradient inequalities needed to add until the optimal solution is reached.

The next pair of columns in Table 1 compare the optimal objective values obtained by solving SOCO by our constraint generation algorithm and SeDuMi. The corresponding columns in Table 2 illustrate the duality gap at the final solution. The CPU times taken to achieve these values by SILP and SeDuMi, rounded to the nearest integer in seconds, are reported in the last two columns of these tables. For our algorithm we report SILP/Oracle to report the cpu time of the whole algorithm and the cpu time of the oracle.

A close study of these results reveals that our constraint generation algorithm outperforms the classical interior point methods when we deal with problems with large number of conic constraints of large size, when m, the dimension of y is relatively small. Except for the last two instances of Table 2, our algorithm outperforms SeDuMi in terms of cpu time. However, it should be mentioned that primal-dual interior point methods, and in particular SeDuMi, is superior to our constraint generation algorithm for problems with small to moderate values of k, n, and m.

			gap		CPU time in second	
k	$\bar{n}$	cuts	SILP	SeDuMi	SILP/Oracle	SeDuMi
2	1E+6	44	5.74E-03	-	139/134	-
4	5E+5	44	5.94E-03	-	91/87	-
8	1E+5	46	6.05E-03	6.93E-03	25/19	154
16	5E+4	44	5.72E-03	8.34E-03	14/7	166
32	1E+4	46	5.83E-03	2.06E-03	10/4	87
64	5E+3	49	6.13E-03	8.25E-03	9/3	72
128	1E+3	49	5.95E-03	4.06E-03	9/3	18
256	5E+2	53	6.17E-03	3.50E-03	7/2	15
512	1E+2	59	6.26E-03	1.71E-03	6/1	8
1024	5E+1	61	6.59E-03	2.45E-03	5/1	5
2048	1E+1	63	6.58E-03	1.85E-03	5/1	2

Table 2: Comparison of CPU time of SeDuMi and SILP implemented on randomly generated SOCO with $m = 30$ and different values of $\bar{n}$ and k.

Table 1 reveals an interesting information. When m is small, a duality gap of 10^{-8} is achieved quickly in all of the test problems. This is not a typical behavior of cutting plane methods. These techniques are known to have difficulties near the optimal solution (see Oskoorouchi and Goffin [19] and Oskoorouchi and Mitchell [18]). As m increases the algorithm returns to its traditional performance. This is the reason that in Table 2 we run the test problems to only three digits of accuracy. In this table, we show that our algorithm can reach an approximate solution with reasonable precision faster than SeDuMi.

A disadvantage of our algorithm is that it requires the value of m to be relatively small. When the dimension of this space is large the constraint generation algorithm requires to add too many constraints before the desired accuracy is reached. Also SILP/Oracle shows that a substantial portion of CPU times is consumed by the oracle in the random search. Clearly a more efficient search could reduce this time and consequently enhance the performance of our algorithm.

7 Conclusions

We presented an interior point constraint generation algorithm for semi-infinite linear programming and showed that the algorithm converges to an ε-solution after a finite number of iterations. We derived two theoretical

complexity results. The number of Newton steps needed to updated the $\mu-$ center after adding p new violating constraints is bounded by $O(p\log(p+1))$, and the overall algorithm stops with an ε-solution to the SILP problems after adding at most

$$O(\frac{m^2\hat{p}^2}{\delta^2}e^{3\sqrt{m}/\varepsilon})$$

constraints, where δ is the radius of the largest full dimensional ball contained in $\mathcal{F}_\Omega$, and $\hat{p}$ is the maximum number of constraints added simultaneously.

We illustrated the convergence behavior of our algorithm on some classical SILP and reported numerical results by implementing it on SOCO problems. We showed that our interior point constraint generation method outperforms classical primal-dual interior point methods on problems with large number of conic constraint of large size, when m, the dimension of y is small.

The algorithm that we described in this paper has the potentials to be combined with branch-and-cut algorithms and implemented to solve mixed integer conic programming problems. An efficient technique for problems of this hand is the use of outer approximation of the second-order cone constraints. See for instance Bonami et al. [4] and Abhishek et al. [1]. The main reason to use polyhedral approximation is the opportunity to have a warm start in the branch-and-bound algorithm after adding an integer cut. Ben-Tal and Nemirovski [3] develop a polyhedral approximation for second-order cone optimization that is used by Vielma et al. [23] in their mixed integer conic programming.

The advantage of this approximation is that it is computed once and used at every relaxation node. However, this approximation, although tight, yields an LP with exponentially large number of constraints and many variables. For example the polyhedral approximation of a single second-order cone of dimension 4, create an LP with over 10,000 variables and 22,000 constraints. This could be costly for CPLEX, especially when the number of cones and their dimensions are large.

We see a potential advantage of our algorithm in solving a class of mixed conic integer programming problems. At each node of branch-and-cut algorithm, a conic optimization relaxation is formed. In this paper we showed that we can efficiently solve a class of SOCO problems using an outer approximation. Given that the algorithm works by gradually generating constraints, an integer cut could be treated as a new added constraint in the

process of algorithm. This could potentially produce an efficient process for the branch-and-cut algorithm. We intend to explore this in future research.

References

[1] K. Abhishek, S. Leyffer, and J. T. Linderoth, *Filmint: an outer-approximation-based solver for nonlinear mixed integer programs*, Preprint ANL/MCS-P1374-0906, Argonne National Laboratory, Mathematics and Computer Science Division, 2008.

[2] A. Atamturk and V. Narayanan, *Conic Mixed-Integer Rounding Cuts*, BCOL RESEARCH REPORT 06.03, Industrial Engineering and Operations Research, University of California, Berkeley, CA, 2008.

[3] A. Ben-Tal and A. Nemirovski, *On polyhedral approximations of second-order cone*, Mathematics of Operations Research, 26 (2001), pp. 193–205.

[4] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuejols, I. E. Grossmann, C. D. Laird, J. L. Lodi, F. Margot, N. Sawaya, and A. Wachter, *An algorithmic framework for convex mixed integer nonlinear programs*, Discrete Optimization, 2008, to appear.

[5] B. Borchers, *CSDP, a C library for semidefinite programming*, Optimization Methods and Software, 11 (1999), pp. 613–623.

[6] S. Burer and R.D.C. Monteiro, *A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization*, Mathematical Programming (series B), 95(2) (2003), pp. 329-357.

[7] M. T. Cezik and G. Iyengar, *uts for mixed 0-1 conic programming*, Mathematical Programming, 104 (2005), pp. 179–202.

[8] I. D. Coope and G. A. Watson, *A projected lagranian algorithm for semi-infinite programming*, Mathematical Programming, 32 (1985), pp. 337-356.

[9] M. C. Ferris and A. B. Philpott, *Ann interior point algorithm for semi-definite programming*, Mathematical Programming, 43 (1989), pp. 257–276.

[10] F. Glineur, *Computational experiments with a linear approximation of second-order cone optimization*, Technical Report 0001, Service de

Mathematique et de Reserche Operationnelle, Faculte Polytechnique de Mons, Mons, Belgium, 2001.

[11] M. A. GOBERNA AND M. A. LOPEZ, *Linear semi-infinite programming theory: an updated survey*, Europian Journal of Operations Research, 143 (2002), pp. 390–405.

[12] J.-L. GOFFIN AND J.-P. VIAL, *Shallow, deep and very deep cuts in the analytic center cutting plane method*, Mathematical Programming, 84 (1999), pp. 89–103.

[13] J.-L. GOFFIN AND J.-P. VIAL, *Multiple cuts in the analytic center cutting plane methods*, SIAM Journal on Optimization, 11 (2000), pp. 266–288.

[14] D. DEN HERTOG, *Interior Point Approach to Linear, Quadratic, and Convex Programming*, Kluwer Academic Publishers, Dordrecht, Boston, London, (1994).

[15] S. J. LI, S. Y. WU, X. Q. YANG, AND K. L. TEO, *A relaxed cutting plane method for semi-definite programming*, Computational Optimization and Applications, 196 (2) (2006), pp. 459–473.

[16] M. LOPEZ AND G. STILL, *Semi-infinite programming*, Europian Journal of Operations Research, 180 (2007), pp. 491–518.

[17] Z.-Q. LUO, C. ROOS, AND T. TERLAKY, *Complexity analysis of logarithmic barrier decomposition method for semi-infinite linear programming*, Applied Numerical Mathematics, 29 (1999) pp. 379–394.

[18] M. R. OSKOOROUCHI AND J. E. MITCHELL, *A second-order cone cutting surface method: complexity and applications*, Computational Optimization and Applications, forthcoming.

[19] M. R. OSKOOROUCHI AND J. L. GOFFIN, *A matrix generation approach for eigenvalue optimization*, Mathematical Programming, Ser. A 109 (2007), pp. 155–179.

[20] C. ROOS, T. TERLAKY, AND J.-P. VIAL, *Theory and Algorithms for Linear Optimization*, John Wiley and Sons Ltd., Baffins Lane, Chichester, England, (1997).

[21] J. F. STURM, *Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones*, Optimization Methods and Software, Vol. 11-12 (1999), pp. 625–653.

[22] K. C. Toh, M. J. Todd, and R. H. Tutuncu, *SDPT3–a MATLAB software package for semidefinite programming, version 2.1*, Optimization Methods and Software, 11 (1999), pp. 545–581.

[23] J. P. Vielma, S. Ahmed, and G. L. Nemhauser, *A lifted linear programming branch-and-bound algorithm for mixed integer conic quadratic programs*, INFORMS Journal of Computing, 20, (2008), pp. 438–450.

[24] Y. Ye, *Interior Point Algorithms, Theory and Analysis* John Wiley Inc., New York, NY, (1997).

[25] Y. Ye, *Complexity analysis of the analytic center cutting plane method that uses multiple cuts*, Mathematical Programming, 78 (1997), pp. 85–104.

[26] S. Y. Wu, S. C. Fang, and C. J. Lin, *Relaxed cutting plane method for solving linear semi-definite programming problems*, Computational Optimization and Applications, 99 (1998), pp. 759–779.