数理解析研究所講究録
第1676巻 2010年 169-182

線形二次錐計画問題に対する半無限計画変換を用いた単体法的アプローチ

京都大学・情報学研究科　伊藤好彦
京都大学・情報学研究科　林　俊介

1　はじめに

本稿では，次のような線形二次錐計画問題 (Linear Second-Order Cone Program: LSOCP) [1] を考える．

$$\begin{array}{lll} \text{(LSOCP)} & \text{minimize} & c^\top x \\ & \text{subject to} & Ax+b \in K \end{array} \tag{1}$$

ここで，$A \in \mathbb{R}^{n\times m}, b \in \mathbb{R}^n, c \in \mathbb{R}^m$ は与えられた行列およびベクトルであり，$K \subseteq \mathbb{R}^n$ は n_i 次元の二次錐

$$K^{(n_i)} := \begin{cases} \{\, u=(u_1,\overline{u}) \in \mathbb{R}\times\mathbb{R}^{n_i-1} \,|\, u_1 \geq \|\overline{u}\|, u_1 \in \mathbb{R}, \overline{u} \in \mathbb{R}^{n_i-1} \,\} & (n_i \geq 2) \\ \mathbb{R}_+ = \{\, u \in \mathbb{R} \,|\, u \geq 0 \,\} & (n_i = 1) \end{cases}$$

を用いて，

$$K = K^{(n_1)} \times K^{(n_2)} \times \cdots \times K^{(n_p)}$$

と表される集合である．ただし，$n_1+n_2+\cdots+n_p = n$ であり，$\|\cdot\|$ はユークリッドノルムを表す．本稿では n 次元実ベクトル u に対して，その第一成分を u_1，残りの $n-1$ 個の成分を1つのベクトルとみなして $\overline{u}$ と書く．また，$(u_1,\overline{u}) \in \mathbb{R}\times\mathbb{R}^{n-1}$ と $\binom{u_1}{\overline{u}} \in \mathbb{R}^n$ とをしばしば区別せずに書く．

LSOCP は幅広いクラスの問題に応用できることが知られている．たとえば，現実の応用例として，Antenna array weight design や有限インパルス応答フィルタの設計，損失リスクの制約を含むポートフォリオ最適化などが知られている [5]．また，非負象限は1次元の二次錐の直積でもあるので，LSOCP は線形計画問題 (Linear Program: LP) を部分クラスとして含んでいるし，二次計画問題 (Quadraric Program: QP) やある種のロバスト最適化問題も，LSOCP に再定式化することができる．一方，LSOCP を部分クラスとして含むより広いクラスの問題として半正定値計画問題 (Semidefinide Program: SDP) [11] がある．しかし，SDP は行列を変数とした最適化問題であるため，LSOCP として解ける問題を SDP に変換して解くのは，計算量の観点からも好ましいとはいえない．

LSOCP を解くためのアルゴリズムに関して，これまで多くの研究がなされてきた．その中でもっともよく知られているのが内点法 [15] である．特に主双対内点法 [10, 15] は収束性に関して理論面においても，実装面においても非常に有効であり，実際にいくつかのソフトウェアが開発されている [9, 8]．一方で LP に対する単体法 (シンプレックス法) [2] を拡張したアルゴリズム (単体法的アルゴリズム) もこれまでいくつか提案されている．たとえば，Pataki [7] らは，LSOCP より広いクラスの問題である SDP に対して LP に対する単体法を理論的に拡張した．しかし，彼らは具体的な SDP に対してアルゴリズムを実装するにはいたっていない．また，

村松 [6] は，ある特殊な構造をもつ LSOCP に対して，LP の単体法における辞書 (dictionary) と同様のものを定義し，それに基づき実装可能な単体法的アルゴリズムを提案した．さらにその実装報告が栗田 · 村松 [14] によってなされている．

このように，LSOCP の解法は大きく二つに分けて内点法と単体法的アプローチがあるが，一般的には，前者の方が後者に比べて理論的にも実用的にも優れている．実際，村松の単体法的アルゴリズムは LSOCP の一部のクラスの問題にしか適用できないのに対し，主双対内点法はほとんど全てのクラスの LSOCP に対して適用できる．また，単体法的アルゴリズムは，LP に対する単体法が必ずしも多項式時間で収束しないように，一般には多項式時間で収束することが保証されていない．さらに，実行可能領域の端点 (extreme point) をたどって，最適解を見つけるため，解に収束するのが遅い．これに対して，主双対内点法は，多項式時間で収束することが理論的に保証されている．

しかしながら，LSOCP に対して単体法的アルゴリズムを考えることは単に理論的な興味だけでなく，次のような実用的な利点が考えられる．たとえば，アルゴリズムの部分問題のように，複数の LSOCP を順々に解いていかなければならないという状況を考えよう．このような場合では，各 LSOCP は互いによく似た構造を持ち，ある一つの LSOCP の解はその直前に解いた LSOCP の解の近くに存在することが多い．したがって，こういった状況に対しては，単体法的手法の方が，一つ前に解いた問題に対する基底の情報を上手く使うことにより，次の LSOCP をより効率的に解けることが期待できる．

本稿では，LSOCP に対する単体法的アプローチとして，LSOCP を線形半無限計画問題 (Linear Semi-Infinite Program: LSIP) に再定式化し，その LSIP に対して Dual-Simplex Primal-Exchange 法 (DSPE 法) を適用することを考える．ここで LSIP とは，決定変数が有限次元であり，無限個の線形不等式で表されるような制約条件の下で，線形関数を最小化するような問題である．なお，本稿で提案するアプローチは，先に述べた LSOCP に対する既存の単体法的アプローチとは異なる点がいくつかある．実際，一般的な形の LSOCP はすべて LSIP に再定式化することができるので，村松 [6] のように対象とする LSOCP の形を限定する必要がない．また，LSIP に対する DSPE 法は，以下の節でのべるように容易に計算機に実装することができる．

本稿の構成を述べる．2 節では，LSOCP と LSIP の具体的な形を示し，LSOCP を LSIP に再定式化する．3 節では LSIP に対する単体法的アルゴリズムである DSPE 法を導入し，それを LSIP に再定式化された LSOCP に対して実装する際の重要な性質について述べる．4 節では，構造が似ている複数の LSOCP を解く際に，3 節で述べた単体法的アルゴリズムと既存の主双対内点法のソルバーとの比較実験およびその考察を行い，単体法的アプローチが有効であることを確認する．5 節では，本稿のまとめと結論を述べる．

2 線形二次錐計画問題と線形半無限計画問題

2.1 線形半無限計画問題とその双対性

本節では，LSIP とその双対問題を具体的な形で示し，その背景について説明する．

LSIP は具体的に以下のように書くことができる．

$$\begin{array}{ll} \text{minimize} & c^{\mathsf{T}} x \\ \text{subject to} & a_t^{\mathsf{T}} x \geq b_t \ (\forall t \in T) \end{array} \tag{2}$$

ここで$c \in \mathbb{R}^m$は与えられた定数ベクトル，Tは任意の添字集合，$a_t = a(t) = (a_1(t), \ldots, a_m(t))^\top$は$T$から$\mathbb{R}^m$への写像，$b_t = b(t)$は$T$から$\mathbb{R}$への写像である．

問題(2)に対する双対問題として，ボレル測度[16]を変数としたものとHaarの双対問題の二つが主に知られている．Tがコンパクトな距離空間であり，制約条件式に含まれる関数a_t, b_tがT上連続であるものとする．このとき，双対問題は以下のように書くことができる．

$$\begin{array}{ll} \text{maximize} & \displaystyle\int_T b(t)\mu(dt) \\ \text{subject to} & \displaystyle\int_T a(t)\mu(dt) = c \\ & \mu \in W,\ \mu \geq 0 \end{array} \tag{3}$$

ここで，WはT上のボレル測度の空間であり，$\mu \geq 0$は任意のボレル集合$T' \subseteq T$に対して$\mu(T') \geq 0$であることを意味する．しかし，この形の双対問題は，一般の添字集合Tに対して定義できず，変数が測度の形で与えられているため計算機で扱うことが困難である．そこで，(3)において，実行可能解μを離散的なもの[1]に限定したHaarの双対問題を考える．具体的には，$\operatorname{supp}\lambda := \{\, t \in T \mid \lambda_t \neq 0 \,\}$が有限集合であるような関数$\lambda : T \to \mathbb{R}_+$を考え，そのような関数を要素とする集合を$\mathbb{R}_+^{(T)} := \{\, \lambda : T \to \mathbb{R}_+ \mid |\operatorname{supp}\lambda| < \infty \,\}$とする．すると，双対問題(3)の積分を有限個の和によって置き換えることができる．それゆえ，主問題(2)に対するHaarの双対問題は次のように書ける．

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{t \in T} \lambda_t b_t \\ \text{subject to} & \displaystyle\sum_{t \in T} \lambda_t a_t = c \\ & \lambda \in \mathbb{R}_+^{(T)} \end{array} \tag{4}$$

ここで，$\sum_{t \in T}$は$t \in \operatorname{supp}\lambda$となるようなすべての$t \in T$に対して総和を取ることを意味する．双対問題(4)は，双対問題(3)の実行可能領域を狭めたものになっているが，一般的な条件で(3)と(4)の最適値が一致することが知られている[4, Chapter 8]．さらに，Tがコンパクトな距離空間でない，より一般的な集合であっても，双対問題(4)を定義することができる．本稿では，これ以降，Haarの双対問題のみをLSIPの双対問題として扱う．

2.2 線形二次錐計画問題の半無限計画変換

本節では，二次錐を無限個の線形不等式制約で書き換えることによりLSOCPをLSIPに再定式化することを考える．次の命題は，k次元の二次錐$K^{(k)}$が無限個の線形不等式を用いて等価に書き換えられることを示している．

命題 2.1. [12, 命題2.2] $k-1$次元の単位球を$\tilde{T} := \left\{\, t \in \mathbb{R}^{k-1} \,\middle|\, \|t\| \leq 1 \,\right\}$とする．このとき，$(u_1, \overline{u}) \in K^{(k)}$であることの必要十分条件は

$$u_1 \geq t^\top \overline{u} \quad (\forall t \in \tilde{T}) \tag{5}$$

が成り立つことである．

[1] 具体的には“有限個のディラック測度の重み和として表されるような離散測度”を指す．

次に，以下のように K の直積構造に応じて以下のように行列 A とベクトル b を分割する．

$$A = \begin{pmatrix} \begin{pmatrix} (a^1)^\top \\ (A^1)^\top \end{pmatrix} \\ \begin{pmatrix} (a^2)^\top \\ (A^2)^\top \end{pmatrix} \\ \vdots \\ \begin{pmatrix} (a^p)^\top \\ (A^p)^\top \end{pmatrix} \end{pmatrix} \qquad b = \begin{pmatrix} \begin{pmatrix} b_1^1 \\ \overline{b}^1 \end{pmatrix} \\ \vdots \\ \begin{pmatrix} b_1^p \\ \overline{b}^p \end{pmatrix} \end{pmatrix}$$

ここで，$a^i \in \mathbb{R}^m$, $A^i \in \mathbb{R}^{m\times(n_i-1)}$, $(b_1^i, \overline{b}^i) \in \mathbb{R} \times \mathbb{R}^{n_i-1}$ $(i = 1,\ldots,p)$ である．このとき，LSOCP (1) の制約は

$$\begin{pmatrix} (a^i)^\top x + b_1^i \\ (A^i)^\top x + \overline{b}^i \end{pmatrix} \in K^{(n_i)} \qquad (i = 1,\ldots,p)$$

と書くことができる．したがって，命題2.1より，LSOCP(1) は次のように LSIP の形で書き換えられる．

$$\begin{array}{ll} \text{minimize} & c^\top x \\ \text{subject to} & (a^i - A^i t^i)^\top x \geq (t^i)^\top \overline{b}^i - b_1^i \ (\forall t^i \in \tilde{T}^i \ i = 1,\ldots,p) \end{array} \tag{6}$$

ここで，$\tilde{T}^i = \left\{ t^i \in \mathbb{R}^{n_i-1} \,\middle|\, \|t^i\| \leq 1 \right\}$ である．一方，LSIP (6) に対する双対問題は次のように書ける．

$$\begin{array}{ll} \underset{\lambda^1,\lambda^2,\ldots,\lambda^p}{\text{maximize}} & \displaystyle\sum_{i=1}^{p} \sum_{t^i \in \tilde{T}^i} \lambda_{t^i}^i ((t^i)^\top \overline{b}^i - b_1^i) \\ \text{subject to} & \displaystyle\sum_{i=1}^{p} \sum_{t^i \in \tilde{T}^i} \lambda_{t^i}^i (a^i - A^i t^i) = c \\ & \lambda^i \in \mathbb{R}_+^{(\tilde{T}^i)} \ (i = 1,\ldots,p) \end{array} \tag{7}$$

3 単体法的アルゴリズム

前節では，LSOCP が LSIP として等価に再定式化できることを示した．本節では，LSIP に対する代表的な単体法的アルゴリズムである DSPE 法 [4. Chapter 12] を紹介する．さらに DSPE 法を LSOCP から再定式化された LSIP に適用する．

3.1 Dual-Simplex Primal-Exchange 法

DSPE 法は LP における単体法を LSIP に拡張したものであり，その名が示すように，双対問題の空間でピボット操作を行い，それが主問題の空間ではアクティブな制約を交換 (exchange) することに対応している．本節では，一般の LSIP(2) に対する DSPE 法について述べ，その性質についていくつか紹介する．

DSPE 法では，初期点として双対問題 (4) の実行可能端点を与える必要がある．そこで，まず空間 $\mathbb{R}^{(T)}$ 内の凸集合に対する端点の定義と，問題 (4) の実行可能端点の性質について述べる．凸集合 $C \subseteq \mathbb{R}^{(T)}$ に対して $c \in C$ が端点であるとは以下のように定義される[2]．

定義 3.1. $C \subseteq \mathbb{R}^{(T)}$ を凸集合とし，c を集合 C の要素とする．このとき，$c = (1-\mu)c_1 + \mu c_2$ となるような $c_1, c_2 \in C \setminus \{c\}$ および $\mu \in (0,1)$ が存在しないならば，c を集合 C の端点 (extreme point) という．

問題 (4) に対して実行可能領域を $\Lambda := \left\{\lambda \in \mathbb{R}_+^{(T)} \middle| \sum_{t\in T} \lambda_t a_t = c\right\}$ と表す．Λ は凸集合であるので，同様に端点を定義できる．また，Λ の端点について次の二つの性質が重要である．

- $\lambda \in \Lambda$ が端点であることと，a_t $(t \in \operatorname{supp}\lambda)$ が線形独立であるということは同値である [3, Theorem 3.1].
- 端点 $\lambda \in \Lambda$ が $|\operatorname{supp}\lambda| = m$ を満たすとき，その端点は非退化であるという．

よって，DSPE 法の初期点として，$a_t\,(t \in \operatorname{supp}\lambda)$ が線形独立となるようなものを選ぶ．また，後述の定理 3.1 より，DSPE 法において生成される点列 $\{\lambda^r\} \subseteq \mathbb{R}_+^{(T)}$ は常に Λ の端点になることが保証されている．

LP に対する単体法が各反復で基底に対してピボット操作を行うのと同様に，DSPE 法はある反復で生成される点 $\lambda^r \in R^{(T)}$ で定義される基底集合に対してピボット操作を行う．ここで，端点 $\lambda \in \Lambda$ に対して，集合 $S \subseteq T$ が以下の 2 つを満たすとき，S を λ に対する基底集合という．

(i) $S \supseteq \operatorname{supp}\lambda$ である．

(ii) $a_t\,(t \in S)$ が $\mathbb{R}^m$ の基底を成している．

条件 (ii) より明らかに $|S| = m$ であるが，一般に基底集合 S は端点 λ に対して一意に決まるものではない．しかし，条件 (i), (ii) に加え，端点 λ が非退化であるならば，S は一意に決まり，$S = \operatorname{supp}\lambda$ である．実際，LP と同様に DSPE 法は，生成される端点 $\lambda^r\,(r = 1, 2, \ldots)$ が非退化であれば，後ほど紹介する定理 3.1 によって，循環が発生しない，すなわち，各反復ごとに目的関数値が狭義に減少することが保証されている．

以上のことをもとに，DSPE 法の詳細な手順を以下に述べる．ただし，$a_t\,(t \in T)$ によって張る空間の次元は m であり[3]，任意の $r \in \{0, 1, 2, \ldots\}$ に対して λ^r は非退化であるとする．さらに，$c \neq 0_m$ かつ，$\Lambda \neq \emptyset$ であるとする．

Dual-simplex primal-exchange method (DSPE 法)

Step 0 初期実行可能端点 λ^0 を選び，$r := 0$ とする．また，$S_0 \subseteq T$ を λ^0 に対する基底集合とする．

Step 1 方程式系 $a_t^\top x = b_t\,(t \in S_r)$ を解き，その一意解を x_r とする．

Step 2 $a_t^\top x^r - b_t < 0$ であるような $t \in T$ を一つ見つけ，それを t_{in}^r とおく．もし，そのような t_{in}^r が存在しない，すなわち，$\max_{t\in T}(a_t^\top x^r - b_t) \geq 0$ であれば，反復を終了する．

[2] 本定義は $\mathbb{R}^n$ 内の凸集合に対する端点の定義の自然な拡張になっている．
[3] この前提は非退化な端点が存在することに対する必要条件になっている．

Step 3 $\operatorname{supp} g^r \subseteq S_r$ かつ $\sum_{t\in S_r} g_t^r a_t = a_{t_{\text{in}}^r}$ となる $g^r \in \mathbb{R}^{(T)}$ を求める.

Step 4 もし $-g^r \in \mathbb{R}_+^{(T)}$ ならば目的関数が非有界なので反復を終了する.
そうでなければ,

$$\mu_r := \min_{t\in S_r, g_t^r>0} \{\lambda_t^r / g_t^r\}$$
$$t_{\text{out}}^r := \operatorname*{argmin}_{t\in S_r, g_t^r>0} \{\lambda_t^r / g_t^r\}$$

とおく.

Step 5 λ^{r+1} および S_{r+1} を

$$\begin{cases} \lambda_t^{r+1} := \lambda_t^r - \mu_r g_t^r & (t \in S_r,\ t \neq t_{\text{out}}^r) \\ \lambda_{t_{\text{out}}^r}^{r+1} := 0 & \\ \lambda_t^{r+1} := 0 & (t \in T \setminus S_r,\ t \neq t_{\text{in}}^r) \\ \lambda_{t_{\text{in}}^r}^{r+1} := \mu_r & \end{cases}$$

$$S_{r+1} := S_r \cup \{t_{\text{in}}^r\} \setminus \{t_{\text{out}}^r\}$$

で定義する. $r := r+1$ として, Step 1 にもどる.

DSPE 法について以下の定理が成り立つ.

定理 3.1. [4. Theorem 12.2] LSIP に対する DSPE 法によって生成される点列を $\{(x^r, \lambda^r)\}$ とする. このとき r 回目の反復で以下の 3 つの場合のいずれかが起こる.

(i) Step 2 で反復が終了すると, x^r, λ^r はそれぞれ主問題 (2), 双対問題 (4) の最適解である.

(ii) Step 4 で反復が終了すると, 双対問題 (4) は非有界であり, 主問題 (2) は実行可能解をもたない.

(iii) 反復が終了しないならば, λ^{r+1} は Λ の端点であり,

$$\sum_{t\in T} \lambda_t^r b_t < \sum_{t\in T} \lambda_t^{r+1} b_t$$

が成り立つ.

このアルゴリズムについていくつか注意すべき点を述べる. Step 2 では, x^r において違反している制約があるならば, それを一つ求め, x^r がすべての制約を満たしているならば, x^r を最適解として出力する. ここで, 毎回の反復で無限個の不等式制約の中でもっとも違反しているものを見つけることができれば, すなわち, $t_{\text{in}}^r = \operatorname{argmin}_{t\in T}\{a_t^\top x^r - b_t\}$ とすることができれば, 全体の収束が早くなることが期待できる. 一般的にこのような t_{in}^r を見つけることは容易ではない. しかし, 後述するように LSOCP を再定式化した LSIP ではこのような t_{in}^r を陽に求めることができる. Step 4 では次の反復で基底集合から出る S_r の要素を選択している. さらに, Step 1, Step 3 ではそれぞれ x_r, g^r を求めているが, これは非退化仮定の下では n 次元連立方程式を解くことで, 陽に求めることができる.

3.2 LSOCPと等価なLSIPに対するDSPE法の適用

本節ではLSIP(6)に対してDSPE法を適用する．DSPE法は，各反復においてStep 1で求めるx^rに対してStep 2で$\overline{t}^r = \mathrm{argmin}_{t\in T}\{a_t^\top x^r - b_t\}$なる$\overline{t}^r \in T$を見つけることができれば収束が早くなることが期待できるが，一般にそのような$\overline{t}^r$を求めることは容易ではない．しかし，LSIP(6)に対しては，このような$\overline{t}^r$を陽に求めることができる．

まず，LSIP(2)のTに対応するものは，LSIP(6)では$\tilde{T}^i\ (i=1,\ldots,p)$の直和，すなわち，

$$T = \tilde{T}^i \oplus \tilde{T}^2 \oplus \cdots \oplus \tilde{T}^m$$

であることに注意する．したがって，Tの任意の要素は$i \in \{1,\ldots,p\}$と$t^i \in \tilde{T}^i$を用いて，

$$t = (i, t^i)$$

と表すことができる．次に，r番目の反復点x^rおよび集合$\tilde{T}^i = \left\{t^i \in \mathbb{R}^{n_i-1} \middle| \|t^i\| \le 1\right\}$を用いて，$\overline{t}^{r,i}$と$s^{r,i}\ (i=1,\ldots,p)$を以下のように定義する．

$$\begin{aligned}
\overline{t}^{r,i} &:= \underset{t^i\in\tilde{T}^i}{\mathrm{argmin}}\{(a^i - A^i t^i)^\top x^r - (t^i)^\top \overline{b}^i + b_1^i\} \\
&= \underset{t^i\in\tilde{T}^i}{\mathrm{argmin}}\{(t^i)^\top(-(A^i)^\top x^r - \overline{b}^i) + (a^i)^\top x^r + b_1^i\} \\
&= \frac{(A^i)^\top x^r + \overline{b}^i}{\|(A^i)^\top x^r + \overline{b}^i\|} \\
s^{r,i} &:= \min_{t^i\in\tilde{T}^i}\{(a^i - A^i t^i)^\top x^r - (t^i)^\top \overline{b}^i + b_1^i\} \\
&= -\|(A^i)^\top x^r + \overline{b}^i\| + (a^i)^\top x^r + b_1^i
\end{aligned}$$

ここで，$\overline{t}^{r,i}$および，$s^{r,i}$が陽に表せていることに注意する．さらに，

$$\bar{i}_r := \underset{i=1,\ldots,p}{\mathrm{argmin}}\, s^{r,i} \tag{8}$$

$$\overline{t}^r := (\bar{i}_r, \overline{t}^{r,\bar{i}_r}) \tag{9}$$

とすることにより，$\overline{t}_r$が陽に計算できる．なお，(8)において$\bar{i}_r$の候補が複数ある時は，iが小さいものから選択する．また，

$$s^r := \min_{i=1,\ldots,p} s^{r,i} \tag{10}$$

について，$s_r \ge 0$が成り立つとき，x^rはLSIP(2)の最適解になっているので反復を終了する．

以上の議論をもとにDSPE法をLSIP(6)に適用する．ただし，DSPE法において適用される仮定はすべて満たされているとする．なお，簡単のため以下の表記を導入する．反復r回目における基底集合を$S_r \subseteq T$とする．また，任意の$t = (i, t^i) \in T$に対して

$$\begin{aligned}
\tilde{a}_t &:= a^i - A^i t^i \\
\tilde{b}_t &:= (t^i)^\top \overline{b}^i - b_1^i
\end{aligned} \tag{11}$$

とおく．このとき，問題(6)に対するDSPE法は以下の通りである．

LSOCP と等価な LSIP に対する DSPE 法

Step 0 初期実行可能端点 λ^0 を選ぶ．$r := 0$ とおく．基底集合を $S_0 := \operatorname{supp}\lambda^0$ とする．

Step 1 連立方程式 $\tilde{a}_t^\top x = \tilde{b}^r$ を解き，その解を x^r とする．

Step 2 $\bar{t}^r$ を (9) 式で計算し，$t^r_{\text{in}} := \bar{t}^r$ とおく．もし (10) によって求めた s_r が非負ならば，反復を終了する．そうでないなら，Step 3 へすすむ．

Step 3 $\operatorname{supp} g^r \subseteq S_r$ かつ，$\sum_{t\in S_r} g^r_t \tilde{a}_t = \tilde{a}_{t^r_{\text{in}}}$ を満たす $g^r \in \mathbb{R}^{(T)}$ を求める．

Step 4 そうでなければ，

$$\mu_r := \min_{t\in S_r, g^r_t>0} \{\lambda^r_t / g^r_t\}$$
$$t^r_{\text{out}} := \operatorname*{argmin}_{t\in S_r, g^r_t>0} \{\lambda^r_t / g^r_t\}$$

とおく．

Step 5 λ^{r+1} および S_{r+1} を

$$\begin{cases} \lambda^{r+1}_t := \lambda^r_t - \mu_r g^r_t & (t \in S_r,\ t \neq t^r_{\text{out}}) \\ \lambda^{r+1}_{t^r_{\text{out}}} := 0 & \\ \lambda^{r+1}_t := 0 & (t \in T \setminus S_r,\ t \neq t^r_{\text{in}}) \\ \lambda^{r+1}_{t^r_{\text{in}}} := \mu_r & \end{cases}$$

$$S_{r+1} := S_r \cup \{t^r_{\text{in}}\} \setminus \{t^r_{\text{out}}\}$$

で定義する．$r := r+1$ として，Step 1 にもどる．

3.3 二段階法

LP の単体法では，初期実行可能解をどのように見つけるかは必ずしも自明ではない．そのため，二段階法と呼ばれる手法を用いて初期実行可能解を求めるのが一般的である．これは，解きたい LP に対して補助問題を生成し，その補助問題を単体法を用いて解き，元の問題の実行可能基底解を得るのを第一段階とし，得られた実行可能基底解を出発点として元の問題を解くのを第二段階とする手法である [13]．本稿では LP における単体法の二段階法を LSIP(6) に対する DSPE 法に拡張する．問題 (7) に対して次の補助問題 [3] を考える．

$$\begin{array}{ll} \min & \tilde{\lambda}_1 + \tilde{\lambda}_2 + \cdots + \tilde{\lambda}_m \\ \text{subject to} & \displaystyle\sum_{i=1}^{p} \sum_{t^i\in\tilde{T}^i} \lambda^i_{t^i}(a^i - A^i t^i) + \sum_{i=k}^{m} \tilde{\lambda}_k e_k = c \\ & \lambda^i \in \mathbb{R}^{(T_i)}_+ \ (i = 1,\ldots,p),\ \tilde{\lambda}_k \geq 0 \ (k = 1,\ldots,m) \end{array} \tag{12}$$

ただし，$c \geq 0$ であるものとする[4]．ここで，$\tilde{\lambda}_k \in \mathbb{R}\,(k = 1, \ldots, m)$ は元の問題 (7) の制約条件式にそれぞれ対応して導入された変数であり，人為変数と呼ばれる．補助問題について次の定理が成り立つ．

定理 3.2. [3, Theorem 6.1] 問題 (7) およびその補助問題 (12) に対して以下の定理が成り立つ．ただし，問題 (12) の最適値を v_a と表す．

- 補助問題 (12) が実行可能ならば，$v_a \geq 0$ である．
- もし $v_a > 0$ ならば元の問題 (7) は実行不可能である．
- $v_a = 0$ であるとき，最適解に対応する基底集合は，元の問題 (7) の基底集合となる．

以上の第 1 段階により求まった λ を初期実行可能端点として DSPE 法を問題 (6) に適用する．まとめると，2 段階法は以下のようになる．

二段階 DSPE 法

Step 1 与えられた問題 (7) に対して補助問題 (12) を生成し，それを DSPE 法を用いて解く．そのとき，最適値が 0 ならば，Step 2 へすすむ．最適値が 0 でなければ，元の問題 (7) は実行不可能なのでアルゴリズムを終了する．

Step 2 Step 1 で生成した補助問題の解，および基底を初期値として元の問題 (7) の解を DSPE 法を用いて求める．

Step 1 が終了したとき，元の問題 (7) の初期実行可能端点が退化していることがある．このようなときに人為変数に対応する添字が基底集合の中に残っていることがある．そのような場合でも，基底集合に残っている人為変数に対応する添字を強制的に基底集合から追い出すようなピボット操作を行えばよい．すると，最終的には元の問題 (7) の変数に対応する要素だけを基底集合に含む実行可能端点を求めることができる．

4 数値実験

本節では，互いによく似た構造を持つ複数の LSOCP(1) を順々に解く場合を考える．具体的には二次錐の直積 K は各問題において同じ構造を持ち，b, c のいずれかを順々に変化させて生成される問題列を考える．そして，そのように生成される問題列に対して，既存の主双対内点法ソルバー SDPT3 [9] および 3 節で提案した二段階 DSPE 法を適用し，それぞれの計算時間の比較を行う．特に，DSPE 法を用いて，問題列中のある一つの問題の解を求めるとき，その直前に解いた問題の解やその基底集合の情報をうまく使って初期実行可能端点を生成する．そうすることにより，ある条件のもとで，問題列の第二番目以降の計算時間が 1 問目の計算時間に比べて小さくなり，問題によっては，DSPE 法の方が SDPT3 よりも早く解を求めることがあることを確認する．

[4] もし，c の成分で負のものがあれば，補助問題を生成する前に元の問題 (7) の制約条件式で，c の成分が負になっている式の両辺に -1 をかけておけばよい．

本稿では，実行可能解を持つようなLSOCPを次のように生成する．まず，LSOCP(1)において各々の二次錐K^{n_i} $(i=1,\ldots,p)$に対応するbの部分ベクトルを$b^i=(b_1^i,\overline{b}^i)\in\mathbb{R}\times\mathbb{R}^{n_i-1}$としたとき，まず$\overline{b}^i$の各成分を$(-1,1)$の一様分布となるように選ぶ．次に，$r$を$(0,1)$の一様分布となるように選び，$b_1^i:=(1+r)\|\overline{b}^i\|$とする．さらに，$A,c$の各成分を$(-1,1)$の一様分布から生成する．このように$A,b,c$を決めると，$b_1^i\geq\|\overline{b}^i\|$が成り立つので，LSOCP(1)は少なくとも$x=0$を実行可能解としてもつ．しかし，有界性や，LSOCPについて強双対性が成り立つ十分条件である実行可能内点の存在については必ずしも保証されないので，生成した問題が有界性や内点の存在性を満たさない場合は，その問題を破棄するものとする．また，数値実験ではDSPE法の終了条件としてStep 2において$s_r\geq-10^{-8}$を採用した．なお，今回の実験では，CPUをPentium4 (3.2GHz × 2)と2GBのメモリを持つ計算機上で行い，アルゴリズムはMATLAB 7.4を用いて実装した．

4.1 bを順々に変化させる場合

LSOCP(1)に対してA,cを固定し，bのみを$b^1,b^2,\ldots,b^{10}$と順々に10通り変化させて生成される問題列

$$\mathcal{L}=\{\mathrm{LSOCP}(b^k)\}_{k=1}^{10}$$

を考える．ただし，$\{b^k\}_{k=1}^{10}$は具体的には次のように生成する．まず，b^1はこの節の初めで述べたように乱数を元に生成する．さらにb^k $(k=2,\ldots,10)$は以下のように生成する．

$$b^k:=b^{k-1}+\frac{\delta\|b^{k-1}\|}{\sqrt{n}}u^n \tag{13}$$

ただし，$\delta>0$を与えられた正の定数であり，$u^n\in\mathbb{R}^n$は各成分が$(-1,1)$の一様乱数となるように生成した乱数ベクトルである．このようにb^kを順々に変化させて生成される問題列に対して，SDPT3を用いて解を求めた場合とDSPE法を用いて解を求めた場合で計算時間を比較する．ここで，問題列$\mathcal{L}=\{\mathrm{LSOCP}(b^k)\}_{k=1}^{10}$に対してDSPE法を適用する際，LSOCP$(b^1)$に対しては，二段階法を用いて初期実行可能端点を求めるが，LSOCP(b^k) $(k=2,\ldots,10)$を解く際には，初期実行可能端点をLSOCP(b^{k-1})の双対最適解[5]とし，初期基底集合をLSOCP(b^{k-1})の最適基底集合とする．なお，問題(7)において，bが変化しても，実行可能領域は変わらないので，前述のように初期点をおいても問題ないことに注意する．

また，本実験では，変数の次元mと二次錐の直積のKの組み合わせ(m,K)として8通り，式(13)におけるδの値として，$10^{-6},10^{-5}$の2通りを考え，各(m,K,δ)に対して10個の問題列$\mathcal{L}_j=\{\mathrm{LSOCP}(b^{k,j})\}_{k=1}^{10}$ $(j=1,\ldots,10)$，すなわち，100個の問題を解くものとする．

得られた結果を表1,2に示す．ここで，各(m,K)に対して，$\mathrm{CPU}_{\mathrm{total}}$は1つの問題列$\mathcal{L}_j$，すなわち10個の問題LSOCP$(b^{1,j})$ $(k=1,2,\ldots,10)$をすべて解くのに要した合計CPU時間を示しており，$\mathrm{CPU}_{\mathrm{first}}$は最初の問題LSOCP$(b^{1,j})$を解くのに要したCPU時間を示している．ただし，表の値は各(m,K)に対する10回の試行$(j=1,2,\ldots,10)$の平均値を示している．また，表2,1はそれぞれ$\delta=10^{-6},10^{-5}$に対応している．表からみても分かるように，DSPE法では，問題列$\mathcal{L}$の1個目の問題の計算時間に対して2個目以降の計算時間がかなり小さく抑えられてい

[5]本節以降で，LSOCP(b^k),LSOCP(c^k),LSOCP(A_k)の双対最適解といった場合，等価なLSIPに対する双対問題(7)の最適解λ^*を表すものとする

表 1: bを変化させたときの $\mathrm{CPU}_{\mathrm{total}}$ と $\mathrm{CPU}_{\mathrm{first}}$ の比較結果 ($\delta = 10^{-6}$)

m	K	$\mathrm{CPU}_{\mathrm{total}}$		$\mathrm{CPU}_{\mathrm{first}}$		$\mathrm{CPU}_{\mathrm{first}}/\mathrm{CPU}_{\mathrm{total}}$	
		SDPT3	DSPE	SDPT3	DSPE	SDPT3	DSPE
10	$K^{(20)}$	1.444	0.185	0.215	0.157	0.149	0.849
10	$(K^{(20)})^{10}$	14.254	0.223	1.511	0.210	0.106	0.942
10	$K^{(100)}$	1.480	0.217	0.146	0.173	0.099	0.797
10	$(K^{(100)})^{10}$	17.869	0.432	1.773	0.410	0.099	0.949
20	$K^{(40)}$	1.852	1.849	0.188	1.804	0.102	0.976
20	$(K^{(40)})^{10}$	19.906	2.480	1.983	2.453	0.100	0.989
20	$K^{(100)}$	1.951	1.937	0.198	1.906	0.101	0.984
20	$(K^{(100)})^{10}$	22.181	3.659	2.223	3.625	0.100	0.991

表 2: bを変化させたときの $\mathrm{CPU}_{\mathrm{total}}$ と $\mathrm{CPU}_{\mathrm{first}}$ の比較結果 ($\delta = 10^{-5}$)

m	K	$\mathrm{CPU}_{\mathrm{total}}$		$\mathrm{CPU}_{\mathrm{first}}$		$\mathrm{CPU}_{\mathrm{first}}/\mathrm{CPU}_{\mathrm{total}}$	
		SDPT3	DSPE	SDPT3	DSPE	SDPT3	DSPE
10	$K^{(20)}$	1.812	0.233	0.181	0.196	0.100	0.841
10	$(K^{(20)})^{10}$	18.178	0.364	1.806	0.308	0.099	0.846
10	$K^{(100)}$	1.887	0.293	0.190	0.212	0.101	0.724
10	$(K^{(100)})^{10}$	19.948	0.592	1.996	0.471	0.100	0.796
20	$K^{(40)}$	2.321	2.043	0.229	2.007	0.099	0.982
20	$(K^{(40)})^{10}$	20.771	3.017	2.069	2.677	0.100	0.887
20	$K^{(100)}$	2.094	2.241	0.214	1.946	0.102	0.868
20	$(K^{(100)})^{10}$	23.145	4.253	2.303	3.534	0.100	0.831

る．実際，SDPT3では，各問題 LSOCP($b^{k,j}$) ($k = 1, 2, \ldots, 10$) での計算時間がほぼ同じになるため，$\mathrm{CPU}_{\mathrm{first}}/\mathrm{CPU}_{\mathrm{total}}$ の値は0.1程度になるが，DSPE法ではその値が0.1よりずっと大きくなっていることが分かる．すなわち，2問目以降の計算時間がほとんどかかっていないことが表1,2より見て取れる．

4.2 cを順々に変化させる場合

LSOCP(1)について A, bを固定し，cのみを順々に変化させていく場合を考える．4.1節の実験と同様に，ベクトル c^k を順々に変化させて次のように問題列 $\mathcal{L}$ を生成する．

$$
\begin{aligned}
c^1 &:= (\text{各成分が } (-1,1) \text{ の一様乱数となるように生成}) \\
c^k &:= c^{k-1} + \frac{\delta \|c^{k-1}\|}{\sqrt{m}} u^m \quad (k = 2, 3, \ldots, 10) \\
\mathcal{L} &= \{\mathrm{LSOCP}(c^k)\}_{k=1}^{10}
\end{aligned}
$$

ここで，$\delta > 0$ は与えられた正の定数であり，$u^m \in \mathbb{R}^m$ は各成分が $(-1,1)$ の一様乱数となるように生成した乱数ベクトルである．このように c^k を順々に変化させて生成される問題列

表 3: c を変化させたときの $\mathrm{CPU}_{\mathrm{total}}$ と $\mathrm{CPU}_{\mathrm{first}}$ の比較結果 ($\delta = 10^{-6}$)

m	K	$\mathrm{CPU}_{\mathrm{total}}$		hot start	$\mathrm{CPU}_{\mathrm{first}}$		$\mathrm{CPU}_{\mathrm{first}}/\mathrm{CPU}_{\mathrm{total}}$	
		SDPT3	DSPE	DSPE	SDPT3	DSPE	SDPT3	DSPE
10	$K^{(20)}$	1.458	0.177	9.0	0.151	0.162	0.104	0.915
10	$(K^{(20)})^{10}$	13.469	0.204	9.0	1.352	0.185	0.100	0.907
10	$K^{(100)}$	1.451	0.233	8.7	0.144	0.172	0.099	0.738
10	$(K^{(100)})^{10}$	16.483	0.443	8.7	1.636	0.325	0.099	0.734
20	$K^{(40)}$	1.984	8.370	6.3	0.201	2.298	0.101	0.275
20	$(K^{(40)})^{10}$	19.654	4.458	8.0	1.877	2.078	0.096	0.466
20	$K^{(100)}$	1.902	7.638	6.8	0.190	2.393	0.100	0.313
20	$(K^{(100)})^{10}$	21.832	7.655	7.7	2.182	3.241	0.100	0.423

$\mathcal{L} = \{\mathrm{LSOCP}(c^k)\}_{k=1}^{10}$ に対して DSPE 法を適用し解を求める場合，4.1 節で行った実験と違い，問題 $\mathrm{LSOCP}(c^k)$ を解くときに $\mathrm{LSOCP}(c^{k-1})$ に対する (問題 (7) の形の) 双対最適解を必ずしもそのまま初期実行可能端点として用いることが出来ない．なぜなら，$\mathrm{LSOCP}(c^{k-1})$ の双対実行可能解[6]が $\mathrm{LSOCP}(c^k)$ の双対実行可能解である保証が無いからである．そこで，$\mathrm{LSOCP}(c^k)$ を解く際の実行可能端点を次のように生成する．

初期実行可能端点の生成法 $\mathrm{LSOCP}(c^{k-1})$ を解いた際に得られた最適基底集合を S_{k-1}^* とする．$c = c^k$ としたときの双対問題 (7) の等式制約を満たし，かつ，$\mathrm{supp}\,\tilde{\lambda} = S_{k-1}^*$ であるような $\tilde{\lambda} \in \mathbb{R}^{(T)}$ を計算する．(この計算は m 次連立方程式を解けば良い．) もし非負条件 $\tilde{\lambda} \in \mathbb{R}_+^{(T)}$ を満たせば，その $\tilde{\lambda}$ を $\mathrm{LSOCP}(c^k)$ を解く際の初期実行可能端点とする．(このとき S_{k-1}^* が初期基底集合となる．) もし，$\lambda \notin \mathbb{R}_+^{(T)}$ ならば，二段階法を用いて初期実行可能端点を求める．

また，4.1 節の実験と同様，各 (m, K, δ) に対して 10 個の問題列 $\mathcal{L}_j = \{\mathrm{LSOCP}(b^{k,j})\}_{k=1}^{10}$ $(j = 1, \ldots, 10)$ を生成するものとする．

実験で得られた結果を表 3,4 に示す．$\mathrm{CPU}_{\mathrm{total}}$ と $\mathrm{CPU}_{\mathrm{first}}$ は前の実験と同様である．また，hot start は問題列 $\mathrm{LSOCP}(c^k)$ の初期実行可能端点を前の問題の最適基底集合から計算できた，すなわち，"基底の引継ぎ" ができた回数を表す[7]．いずれの値も，10 個の問題列 $\mathcal{L}_j$ $(j = 1, \ldots, 10)$ に対する平均値である．また，表 3.4 はそれぞれ $\delta = 10^{-6}, \delta = 10^{-5}$ に対応している．

表が示すように δ が 10^{-6} 程度ならば，基底を引き継げる回数が多いため，SDPT3 に比べて合計計算時間が短くなることが多い．特に二次錐 K が直積構造をもつときその傾向が見られる．しかし，10^{-5} では基底を引き継げる頻度が低くなり，そのため問題列中の各問題の計算時間を減少させる効果が小さくなっていることが分かる．

[6]本節以降で，$\mathrm{LSOCP}(c^k)$, $\mathrm{LSOCP}(A_k)$ の実行可能解といった場合，等価な LSIP に対する双対問題 (7) の実行可能解 λ を表すものとする

[7]$\mathrm{LSOCP}(c^1)$ を解くときには必ず二段階法を用いるので，hot start の値が 9 であれば，すべての場合で基底の引継ぎができたことを意味する．

表 4: cを変化させたときの$\mathrm{CPU}_{\mathrm{total}}$と$\mathrm{CPU}_{\mathrm{first}}$の比較結果 ($\delta = 10^{-5}$)

m	K	$\mathrm{CPU}_{\mathrm{total}}$		hot start	$\mathrm{CPU}_{\mathrm{first}}$		$\mathrm{CPU}_{\mathrm{first}}/\mathrm{CPU}_{\mathrm{total}}$	
		SDPT3	DSPE	DSPE	SDPT3	DSPE	SDPT3	DSPE
10	$K^{(20)}$	2.074	1.792	2.9	0.208	0.260	0.100	0.145
10	$(K^{(20)})^{10}$	17.571	0.816	6.8	1.748	0.261	0.099	0.320
10	$K^{(100)}$	1.933	1.254	5.1	0.200	0.255	0.103	0.203
10	$(K^{(100)})^{10}$	20.593	1.277	7.0	2.057	0.410	0.100	0.321
20	$K^{(40)}$	1.958	24.889	0.0	0.196	2.476	0.100	0.099
20	$(K^{(40)})^{10}$	21.626	18.019	2.4	2.164	2.298	0.100	0.128
20	$K^{(100)}$	2.054	25.611	0.0	0.212	2.623	0.103	0.102
20	$(K^{(100)})^{10}$	22.934	31.623	0.8	2.279	3.407	0.099	0.108

5 結論

本稿では，LSOCPを等価なLSIPに再定式化し，その等価なLSIPに対する最適性条件と元のLSOCPに対する最適性条件との関係を示した．また，LSOCPに対する単体法的アプローチとして，等価に再定式化されたLSIPに対してDSPE法を適用することを提案した．また，提案した単体法的アプローチと既存の内点法のソルバーであるSDPT3との比較実験を行った．その結果，構造が似ている複数のLSOCPを解く場合には単体法的アルゴリズムが有効であることを確認した．

参考文献

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

[2] G. B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, 1963.

[3] M. A. Gorberna and V. Jornet. Geometric fundamentals of the simplex method in semi-infinite programming. *OR Spektrum*, Vol. 10, pp. 145–152, 1988.

[4] M. A. Gorberna and M. A. López. *Linear Semi-infinite Optimization.* John Wiley and Sons Ltd, 1998.

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

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

[7] G. Pataki. Cone-LP's and semidefinite programs: geometry and a simplex-type method. In *Integer Programming and Combinatorial Optimization*, Vol. 1084 of *Lecture Notes in Computer Science*, pp. 162–174. Springer Berlin, Heidelberg, 1996.

[8] J. F. Sturm. *Using SeDuMi 1.02, a matlab toolbox for optimization over symmetric cones (Updated for Version 1.05).* `http://sedumi.ie.lehigh.edu/`, 2001.

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

[10] T. Tsuchiya. A convergence analysis of the scaling-invariant primal-dual path-following algorithms for second-order cone programming. *Optimization Methods and Software*, Vol. 11&12, pp. 141–182, 1999.

[11] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, Vol. 38, pp. 49–95, 1996.

[12] 伊藤好彦. 線形二次錐計画問題に対する半無限計画変換を用いた単体法的アプローチ. 特別研究報告書, 京都大学工学部情報学科数理工学コース, 2009.

[13] 福島雅夫. 数理計画入門. システム制御ライブラリー. 朝倉書店, 2004.

[14] 栗松圭介, 村松正和. 2次錐計画のサブクラスに対する単体法的アルゴリズムにおけるピボット選択規則について. 統計数理, 第53巻, 第2号, pp. 349–360, 2005.

[15] 小島正和, 土谷隆, 水野眞治, 矢部博. 内点法. 経営科学のニューフロンティア 9. 朝倉書店, 2001.

[16] 谷島賢. ルベーグ積分と関数解析. 講座 数学の考え方 13. 朝倉書店, 東京, 2002.