統計数理(2005)
第 53 巻 第 2 号 349–360
ⓒ2005 統計数理研究所
特集「計算推論—モデリング・数理・アルゴリズム—」
[原著論文]

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

栗田 圭介[1] ・村松 正和[2]

(受付 2005 年 2 月 28 日; 改訂 2005 年 6 月 27 日)

要　　旨

Muramatsu (2003)によって提案された，2 次錐計画のあるサブクラスに対するピボット・アルゴリズムを実装する．またその実装を用いて様々なピボット選択規則に関する数値実験を行なう．その結果，ピボット選択規則がアルゴリズムの実用的な効率に大きな影響を与えることがわかった．また，Muramatsu (2003)によって提案されたピボット演算を一部改良することで，最適解に至る反復回数が改善されることを数値実験により示す．

キーワード： 2 次錐計画，ピボット，単体法．

1. はじめに

2 次錐計画とは，アフィン空間と 2 次錐

$$\mathcal{K}_{r+1} = \left\{ (u_0, \boldsymbol{u}) \in \mathbb{R}^{r+1} \mid u_0 \geq \|\boldsymbol{u}\| \right\}$$

の交わりを許容領域とし，線形目的関数を最大化または最小化する最適化問題である．この問題は線形計画の拡張となっているのみならず，その非線形性の為に，チェビシェフ近似問題，施設配置問題，ロバスト最適化など様々な応用があることが知られており(Ben-Tal and Nemirovski, 2001; Lobo et al., 1998; Muramatsu and Suzuki, 2003; Sasagawa and Tsuchiya, 2003)，重要な最適化問題と位置づけられている．また，この問題は錐線形計画問題(Nesterov and Nemirovski, 1993; Muramatsu, 2002; 村松, 2003; 田村・村松, 2002)の一部とみなすこともでき，解法としては内点法が確立されている(Monteiro and Tsuchiya, 2000; Andersen et al., 2000)．

線形計画に対しては，内点法より以前に Dantzig (1963)により単体法が提案されており，やはり効率の良い解法として知られている．しかし，線形計画の拡張である 2 次錐計画に対しては，単体法に対応するアルゴリズムに関する研究はその非線形性の克服が障害となり，ほとんど行なわれていないのが実情である．2 次錐計画を含む半正定値計画に関して，Pataki (1996)や Lasserre (1996)の研究はあるが，実装可能なものではない．その中で Muramatsu (2003)は，問題のクラスを一般の 2 次錐計画でなく，そのサブクラスに限り，それに属する問題に対して単体法の直接的な拡張としてピボット演算およびそれを用いたアルゴリズム(ピボット・アルゴリズム)を提案し，理論的な解析を行なった．本稿ではそのアルゴリズムを実装する．我々の知

[1] 電気通信大学大学院 情報工学専攻：〒182–8585 東京都調布市調布が丘 1–5–1
[2] 電気通信大学 情報工学科：〒182–8585 東京都調布市調布が丘 1–5–1

る限り，2 次錐計画に対するピボット・アルゴリズムで，実装可能なものは Muramatsu (2003) だけであり，本論文はその最初の実装の報告である．

さて，線形計画に対する単体法では，ピボット演算の際にどのように非基底変数を選ぶか自由度があることが知られている．このとき，特定の非基底変数を選ぶ規則を，本論文ではピボット選択規則と呼ぶ．線形計画の場合には，どのようなピボット選択規則を用いても単体法は(非退化仮定のもとで)大域的収束することが知られている．それに比べ，Muramatsu (2003) では，ある特定のピボット選択規則を用いた場合しか，大域的収束が証明されていない．

本論文では，この 2 次錐計画に対するピボット・アルゴリズムにおいて，様々なピボット選択規則によってどのようにアルゴリズムの挙動が変わるのかを数値的に調べる．その結果，ピボット選択規則とアルゴリズムの効率の間に一定の関係を見て取ることができた．さらにその結果を考察し，Muramatsu (2003)で提案されたピボット演算の改良を提案する．この改良されたピボット演算を用いると，より効率の良い安定したアルゴリズムが得られることがわかった．

この論文の構成は次の通りである．まず第 2 章で，Muramatsu (2003)で提案された 2 次錐計画問題のサブクラスと，それに対する辞書およびピボット演算に関する概略を述べる．第 3 章はそのピボット演算を用い，様々なピボット選択規則を用いた単体法の数値実験を行なう．さらに第 4 章では，ピボット演算を一部改良し，それがアルゴリズムの効率に大きな影響を与えることを数値的に示す．第 5 章で結論と今後の課題を述べる．

2. 2 次錐計画に対するピボット・アルゴリズム

2.1 扱う問題と辞書

本論文では Muramatsu (2003)に従い次の形をした 2 次錐計画問題を考える：

$$\langle P\rangle \begin{cases} \text{minimize} & \boldsymbol{c}^T\boldsymbol{x}+\boldsymbol{d}^T\boldsymbol{u}+u_0 \\ \text{subject to} & A\boldsymbol{x}+R\boldsymbol{u}=\boldsymbol{b}, \\ & \boldsymbol{x}\geq 0, \quad \begin{pmatrix} u_0 \\ \boldsymbol{u}\end{pmatrix}\in\mathcal{K}_{r+1}. \end{cases}$$

ただし，$\boldsymbol{b}\in\mathbb{R}^m$，$\boldsymbol{c}\in\mathbb{R}^n$，$\boldsymbol{d}\in\mathbb{R}^r$，$A\in\mathbb{R}^{m\times n}$，$R\in\mathbb{R}^{m\times r}$ である．この問題が，通常の 2 次錐計画問題と異なるのは，2 次錐制約が 1 つしかないことと，変数 u_0 が等式制約に現れないことである．現在のところ，この二つの条件が，ピボット・アルゴリズムが動く必須条件である．

よく知られているように，$\langle P\rangle$ の双対問題は

$$\langle D\rangle \begin{cases} \text{maximize} & \boldsymbol{b}^T\boldsymbol{y} \\ \text{subject to} & \boldsymbol{s}+A^T\boldsymbol{y}=\boldsymbol{c}, \\ & \boldsymbol{z}+R^T\boldsymbol{y}=\boldsymbol{d}, \\ & \boldsymbol{s}\geq\boldsymbol{0}, \quad (1,\boldsymbol{z})\in\mathcal{K}_{r+1}. \end{cases}$$

となり，$\langle P\rangle$ の許容解 $(\boldsymbol{x},u_0,\boldsymbol{u})$ と $\langle D\rangle$ の許容解 $(\boldsymbol{y},\boldsymbol{s},\boldsymbol{z})$ との間に次の相補性条件

$$\boldsymbol{x}^T\boldsymbol{s}=0, \quad u_0+\boldsymbol{u}^T\boldsymbol{z}=0 \tag{2.1}$$

が成り立っているとき，これらはそれぞれ $\langle P\rangle$，$\langle D\rangle$ の最適解となる．ここで，$\boldsymbol{x}$，$\boldsymbol{s}$ に関しては，$\boldsymbol{x}\geq\boldsymbol{0}$，$\boldsymbol{s}\geq\boldsymbol{0}$ と (2.1) から，$x_js_j=0$ $(j=1,\ldots,n)$ という条件が得られるが，$(u_0,\boldsymbol{u})$ と $\boldsymbol{z}$ に関しては相補性と 2 次錐制約からだけでは，どちらかの変数が 0 というようなことは言えず，自由度があることを指摘しておく．この観察が後に，辞書を定義するとき重要となる．

さて基底変数として，$B\subseteq\{1,\ldots,n\}$，$B'\subseteq\{1,\ldots,r\}$ を

$$G=(A_B,R_{B'})\in\mathbb{R}^{m\times m}$$

が逆可能となるように取る．ここで A_B は行列 A の変数 B に対応する列を抜き出した行列である．$R_{B'}$ も定義は同様である．G を基底行列と言う．

G^{-1} を $\langle P\rangle$ の等式条件の両辺に左からかけることにより，

$$\begin{pmatrix} \boldsymbol{x}_B \\ \boldsymbol{u}_{B'} \end{pmatrix} = G^{-1}\boldsymbol{b} - G^{-1}A_N\boldsymbol{x}_N - G^{-1}R_{N'}\boldsymbol{u}_{N'}$$

が得られる．ただし，$N=\{1,\ldots,n\}\backslash B$，$N'=\{1,\ldots,r\}\backslash B'$ であり，$\boldsymbol{x}_B$ は $\boldsymbol{x}$ の B に相当する部分からなるベクトル，$\boldsymbol{u}_{B'}$ は $\boldsymbol{u}$ の B' に相当する部分からなるベクトルである．

次に，基底解 $(\tilde{\boldsymbol{x}},\tilde{u}_0,\tilde{\boldsymbol{u}})$ について説明する．Muramatsu (2003)では，前述の $\langle P\rangle$ における相補性から来る自由度を克服するため，あらかじめ基底解の $\tilde{u}$ のうち，N' に対応する部分 $\tilde{\boldsymbol{u}}_{N'}$ は与えられていると考え，辞書を構成した．その手順は以下の通りである．まず $\boldsymbol{u}_{N'}=\tilde{\boldsymbol{u}}_{N'}+\boldsymbol{v}_{N'}$ と変数変換して，$\langle P\rangle$ の等式条件を

$$\begin{pmatrix} \boldsymbol{x}_B \\ \boldsymbol{u}_{B'} \end{pmatrix} = \tilde{\boldsymbol{b}} - G^{-1}A_N\boldsymbol{x}_N - G^{-1}R_{N'}\boldsymbol{v}_{N'} \quad (\tilde{\boldsymbol{b}}=G^{-1}(\boldsymbol{b}-R_{N'}\tilde{\boldsymbol{u}}_{N'})),$$

と等価な条件に変換する．さらに

$$\begin{pmatrix} \tilde{\boldsymbol{x}}_B \\ \tilde{\boldsymbol{u}}_{B'} \end{pmatrix} = \tilde{\boldsymbol{b}}, \quad \begin{pmatrix} D_{BN} \\ D_{B'N} \end{pmatrix} = G^{-1}A_N, \quad \begin{pmatrix} D_{BN'} \\ D_{B'N'} \end{pmatrix} = G^{-1}R_{N'},$$

とおけば，$\langle P\rangle$ の等式条件は

$$\begin{aligned} \boldsymbol{x}_B &= \tilde{\boldsymbol{x}}_B - D_{BN}\boldsymbol{x}_N - D_{BN'}\boldsymbol{v}_{N'} \\ \boldsymbol{u}_{B'} &= \tilde{\boldsymbol{u}}_{B'} - D_{B'N}\boldsymbol{x}_N - D_{B'N'}\boldsymbol{v}_{N'} \end{aligned}$$

と書き表すことができる．これらの変数を用いて目的関数を書き直すと，

$$\theta = \tilde{\theta} + \tilde{\boldsymbol{s}}_N^T\boldsymbol{x}_N + \tilde{\boldsymbol{z}}_{N'}^T\boldsymbol{v}_{N'} + u_0$$

となる．ただし，

$$\begin{aligned} \tilde{\theta} &= \boldsymbol{c}_B^T\tilde{\boldsymbol{x}}_B + \boldsymbol{d}_{B'}^T\tilde{\boldsymbol{u}}_{B'} + \boldsymbol{d}_{N'}^T\tilde{\boldsymbol{u}}_{N'} \\ \tilde{\boldsymbol{s}}_N &= \boldsymbol{c}_N - D_{BN}^T\boldsymbol{c}_B - D_{B'N}^T\boldsymbol{d}_{B'} \\ \tilde{\boldsymbol{z}}_{N'} &= \boldsymbol{d}_{N'} - D_{BN'}^T\boldsymbol{c}_B - D_{B'N'}^T\boldsymbol{d}_{B'} \end{aligned}$$

である．これらの関係から，$\langle P\rangle$ に対し，B，B'，$\tilde{\boldsymbol{u}}_{N'}$ が与えられたときの辞書 $\mathcal{D}(B,B';\tilde{\boldsymbol{u}}_{N'})$ を次のように定義する：

$$\mathcal{D}(B,B';\tilde{\boldsymbol{u}}_{N'}) \quad \begin{cases} \theta = \tilde{\theta} & +\tilde{\boldsymbol{s}}_N^T\boldsymbol{x}_N & +\tilde{\boldsymbol{z}}_{N'}^T\boldsymbol{v}_{N'} + u_0 \\ \boldsymbol{x}_B = \tilde{\boldsymbol{x}}_B & -D_{BN}\boldsymbol{x}_N & -D_{BN'}\boldsymbol{v}_{N'} \\ \boldsymbol{u}_{B'} = \tilde{\boldsymbol{u}}_{B'} & -D_{B'N}\boldsymbol{x}_N & -D_{B'N'}\boldsymbol{v}_{N'} \\ \boldsymbol{u}_{N'} = \tilde{\boldsymbol{u}}_{N'} & & +\boldsymbol{v}_{N'}. \end{cases} \tag{2.2}$$

線形計画では，基底変数(非基底変数)を決定すれば自動的に辞書が定まっていたが，問題 $\langle P\rangle$ に対する辞書ではそれ以外に基底解の N' に対応する部分，$\tilde{u}_{N'}$ が必要になっていることに注意する．

さて，$\tilde{u}_0$ はなるべく小さい方が目的関数値は小さいのだから，常に $\tilde{u}_0=\|\tilde{\boldsymbol{u}}\|$ であると考えて差し支えない．これより，辞書 $\mathcal{D}(B,B';\tilde{\boldsymbol{u}}_{N'})$ に対応する基底解は，$\boldsymbol{x}_N=\boldsymbol{0}$，$\boldsymbol{v}_{N'}=\boldsymbol{0}$，$u_0=\tilde{u}_0=\|\tilde{\boldsymbol{u}}\|$ と置いて次のように定義される：

$$(\boldsymbol{x}_B,\boldsymbol{x}_N,u_0,\boldsymbol{u}_{B'},\boldsymbol{u}_{N'}) = (\tilde{\boldsymbol{x}}_B,\boldsymbol{0},\tilde{u}_0,\tilde{\boldsymbol{u}}_{B'},\tilde{\boldsymbol{u}}_{N'}). \tag{2.3}$$

基底解が許容であることは，$\tilde{\boldsymbol{x}}_B \geq \mathbf{0}$ と同値である．

2.2 部分問題とピボット演算

Muramatsu (2003)では 3 種類の部分問題を解くことで，ピボット演算を行なうことを考えている．最初の部分問題は，基底解から x_i ($i \in N$)を 0 から増やすことにより，目的関数を下げることができるかどうかをチェックする問題である．すなわち，各 $i \in N$ に対し，

$$\langle S_i \rangle \begin{cases} \text{minimize}_{\ (x_i, u_0)} & \tilde{s}_i x_i + u_0 \\ \text{subject to} & \tilde{\boldsymbol{x}}_B - x_i D_{Bi} \geq \mathbf{0}, \quad x_i \geq 0, \\ & \begin{pmatrix} u_0 \\ \tilde{\boldsymbol{u}}_{B'} - x_i D_{B'i} \\ \tilde{\boldsymbol{u}}_{N'} \end{pmatrix} \in \mathcal{K}_{r+1}, \end{cases}$$

となる．ここで，D_{Bi} は D_{BN} の i に対応する列である．

2 番目の部分問題は，$\tilde{u}_0 > 0$ のとき，v_i ($i \in N'$)を動かして目的関数を下げることを考えるものである．すなわち，$i \in N'$ に対する次の問題である：

$$\langle Z_i \rangle \begin{cases} \text{minimize}_{\ (v_i, u_0)} & \tilde{z}_i v_i + u_0 \\ \text{subject to} & \tilde{\boldsymbol{x}}_B - v_i D_{Bi} \geq \mathbf{0}, \\ & \begin{pmatrix} u_0 \\ \tilde{\boldsymbol{u}}_{B'} - v_i D_{B'i} \\ \tilde{\boldsymbol{u}}_{N'} + v_i \boldsymbol{e}_i \end{pmatrix} \in \mathcal{K}_{r+1}. \end{cases}$$

ここでは v_i は非負制約を持たないことに注意する．

最後に 3 番目の部分問題として $\tilde{u}_0 = 0$ の場合を考える．このとき，辞書 $\mathcal{D}(B, B'; \tilde{\boldsymbol{u}}_{N'})$ に対応する基底解が次の意味で非退化と仮定する．

仮定 1. 非退化仮定

(1) $\tilde{\boldsymbol{x}}_B > \mathbf{0}$
(2) $B' = \emptyset$.

この非退化仮定は，Faybusovich (1997)や Alizadeh and Goldfarb (2003)で提案されている主非退化仮定をこの場合に当てはめたものである．2 番目の条件より，$N' = \{1, \ldots, r\}$ であることに注意する．

さて，この場合の部分問題は，基底解から $-\tilde{z}$ 方向に移動することにより目的関数を下げようとするもので，

$$\langle Z_{N'} \rangle \begin{cases} \text{minimize}_{\ (\lambda, u_0)} & -\lambda \|\tilde{\boldsymbol{z}}_{N'}\|^2 + u_0 \\ \text{subject to} & \tilde{\boldsymbol{x}}_B + \lambda D_{BN'} \tilde{\boldsymbol{z}}_{N'} \geq \mathbf{0}, \quad \lambda \geq 0, \\ & \begin{pmatrix} u_0 \\ -\lambda \tilde{\boldsymbol{z}}_{N'} \end{pmatrix} \in \mathcal{K}_{r+1}. \end{cases}$$

と定式化される．

Muramatsu (2003)では次のことが証明されている

定理 2. (Muramatsu, 2003, Theorem 5.1 と Theorem 6.1) 許容辞書 $\mathcal{D}(B, B'; \tilde{\boldsymbol{u}}_{N'})$, ($\tilde{\boldsymbol{x}}_B > \mathbf{0}$) があるとする．全ての $\langle S_i \rangle$ ($i \in N$) について $x_i = 0$ が最適解であり，さらに

(1)　$\tilde{u}_0 > 0$ の場合，全ての $\langle Z_i \rangle$ $(i \in N')$について $v_i = 0$ が最適解である．
(2)　$\tilde{u}_0 = 0$ の場合，非退化仮定が成り立っており，$\lambda = 0$ が $\langle Z_{N'} \rangle$ の最適解である．

が成り立っていれば，基底解は $\langle P \rangle$ の最適解であり，$\langle D \rangle$ の最適解も行列演算により計算できる．

もし，現在の基底解が最適解でないとすれば，部分問題のうちのどれかが 0 でない最適解を持つ．そのとき，基底変数と非基底変数を入れ替えるピボット演算を行なうことができる．そのピボット演算は，基底変数と非基底変数の種類により次の 4 つに分類される：

(1)　BN ピボット：ある $i \in N$ について $\langle S_i \rangle$ が $\hat{x}_i^* > 0$ なる最適解を持ち，さらにある $j \in B$ について $\tilde{x}_j - D_{ji}\hat{x}_i^* = 0$ が成り立つとき
⇒ $i \in N$ が基底に入り，$j \in B$ が基底から出る．
(2)　$B'N$ ピボット：ある $i \in N$ について $\langle S_i \rangle$ が $\hat{x}_i^* > 0$ なる最適解を持ち，さらに全ての $j \in B$ について $\tilde{x}_j - D_{ji}\hat{x}_i^* > 0$ が成り立つとき
⇒ $i \in N$ が基底に入り，$D_{ji} \neq 0$ なる $j \in B'$ が基底から出る．（そのような j が存在することはやはり Muramatsu, 2003, Lemma 3.2 (ii)で証明されている．）
(3)　BN' ピボット：ある $i \in N'$ について $\langle Z_i \rangle$ が $\hat{v}_i^* \neq 0$ なる最適解を持ち，さらにある $j \in B$ について $\tilde{x}_j - D_{ji}\hat{v}_i^* = 0$ が成り立つとき
⇒ $i \in N'$ が基底に入り，$j \in B$ が基底から出る．
(4)　N' ピボット：ある $i \in N'$ について $\langle Z_i \rangle$ が $\hat{v}_i^* \neq 0$ なる最適解を持ち，さらに全ての $j \in B$ について $\tilde{x}_j - D_{ji}\hat{v}_i^* > 0$ が成り立つとき
⇒ 基底／非基底は変化しない．（$\tilde{\boldsymbol{u}}_{N'}$ のみが変化する．）

ピボット演算の手順自体は，等式条件を用いて基底変数と非基底変数を交換するもので，線形計画の場合と同様である．詳細については，Muramatsu (2003)を参照されたい．

退化していない線形計画の場合，「最適解でなければピボット演算できて目的関数が減る」ことを証明すれば，辞書は有限個しかないのでそれは自動的にアルゴリズムの大域的収束性の証明となっている．しかし，$\langle P \rangle$ は非線形計画問題であり，辞書は B と B' のみならず，$\tilde{\boldsymbol{u}}_{N'}$ にも依存する．そのため，定理 2 だけでは大域的収束性は保証できない．実際，Muramatsu (2003)では，一種の大域的収束性を証明しているが，その際には MDS（後述）と呼ばれる特定のピボット選択規則を用いる必要があった．

2.3　ピボット選択規則

前章で紹介したピボット演算を用いた $\langle P \rangle$ を解く単体法的アルゴリズムの概要は以下のようになる．

- ステップ 1：許容で非退化な辞書 $\mathcal{D}(B, B'; \tilde{\boldsymbol{u}}_{N'})$ が与えられる．
- ステップ 2：部分問題 $\langle S_i \rangle$ $(i \in N)$，$\langle Z_i \rangle$ $(i \in N')$，または $\langle Z_{N'} \rangle$ に関して，次を行なう．
  (1)　その部分問題を解く．（これは有限ステップで可能なことが Muramatsu (2003)に示されている．）
  (2)　部分問題が非有界であれば終了．$\langle P \rangle$ は非有界．
  (3)　ある部分問題に対して 0 でない最適解が見つかれば，ピボット演算を行い，新たな辞書を得て 1 へ戻る．
- ステップ 3：全ての部分問題が 0 を最適解に持つとき，定理 2 に基づき $\langle P \rangle$ および $\langle D \rangle$ の最適解を計算する．

ここで，ステップ 2 はどのような順番で部分問題を解くのかについて言及していないが，これを規定するのが次に述べるピボット選択規則である．一般に，基底に入ることのできる非基底変数は複数あるので，その選び方がアルゴリズムの効率に影響を与えることが考えられる．今回実験に使用されるピボット選択規則は以下の 3 種類である．

- **辞書順選択規則**(Lexicographic Order Strategy，LOS)
  ステップ 2 で部分問題を変数の添字の小さいものから順に解いていき，最初に目的関数値を減らすことがわかったものを基底に入れる変数とする．
- **ランダム選択規則**(Random Order Strategy，ROS)
  ステップ 2 で反復毎に乱数を振り，部分問題を解いていく順序を決める．最初に目的関数値を減らすことがわかったものを基底に入れる変数とする．
- **最大目的関数減少規則**(Maximum Decrease Strategy，MDS)
  ステップ 2 で，全ての部分問題を解き，ピボット演算後の目的関数値を計算し，目的関数値が最も減少する変数を選ぶ．

退化していない線形計画の場合には，上記のピボット選択規則のうちどれを使っても単体法は大域的収束をすることが知られているが，2 次錐計画 $\langle P \rangle$ に対しては，MDS 以外の大域的収束は示されていない．今回の実験では，$\langle P \rangle$ に対してピボット選択規則が収束にどのような影響を与えるかを調べる．

厳密に言えば，途中で退化した辞書が現れると，巡回したり，また $\langle Z_{N'} \rangle$ をうまく定義できなくなりアルゴリズムが破綻したりする可能性がある．しかし，今回乱数を用いて発生させた問題では，そのような現象は起きなかった．

3. 数値実験 I：ピボット選択規則の比較

MATLAB を用いてピボット・アルゴリズムの実装を行い，これを用いて，上記の 3 種類のピボット選択規則の比較を行なった．なお，コンピュータは Intel Xeon 2.4 GHz プロセッサ，メモリ 2 GB を持つ RedHat 9 を用いている．

各ピボット選択規則に対して 10 段階に渡って問題の大きさを変えながらそれぞれの場合につき 10 問ずつ，計 100 問の 2 次錐計画問題を解いた．表中の計算時間，反復回数は 10 問のうち，成功したものの平均である．ここでは 2000 回の反復を行っても最適解に収束しない場合に失敗とみなした．

実験に使用した 2 次錐計画問題は，乱数を使って生成したものである．各ピボット選択規則に対して実験を行なう際に，毎回同じ乱数の種を与えて問題を生成し，同一の問題群を解かせている．実験結果を表 1–表 3 に示す．

まず表 1 を見ると，LOS は変数の数の多少にかかわらず，ほとんど収束に至っていない．これに比べ，ROS (表 2)では，多少収束しない場合があるものの，変数の数が多くなっても比較的安定して解けている．MDS (表 3)は変数の数が多くなると収束に至らない場合が増えてくることがわかる．

反復回数に関しては，毎回のピボット演算における減少量では MDS が有利である半面，MDS では 1 回のピボット演算にかかる時間が多くなる．全体の計算時間としては，ROS，MDS の両者に大きな差は生じない結果となった．安定性では ROS が MDS を上回っており，より優れているといえる．

MDS は Muramatsu (2003)でその大域的収束性が証明されているにも関わらず，収束しない例が見られる．MDS で解けなかった典型的な問題の一つにおける目的関数値の減少する様子を

表 1. 実験結果(LOS).

変数の数			計算結果			
n	m	r	成功	計算時間 [s]	反復回数	時間/反復回数 [ms]
30	30	20	1	12.6300	1647	7.6685
60	60	40	0	-	-	-
90	90	60	0	-	-	-
120	120	80	0	-	-	-
150	150	100	0	-	-	-
180	180	120	0	-	-	-
210	210	140	0	-	-	-
240	240	160	0	-	-	-
270	270	180	0	-	-	-
300	300	200	0	-	-	-

表 2. 実験結果(ROS).

変数の数			計算結果			
n	m	r	成功	計算時間 [s]	反復回数	時間/反復回数 [ms]
30	30	20	10	0.8010	152	5.2628
60	60	40	9	1.1800	165	7.1132
90	90	60	10	3.7230	311	11.9634
120	120	80	10	10.6760	593	17.9973
150	150	100	9	17.9589	683	26.2899
180	180	120	10	27.5790	763	36.1076
210	210	140	9	60.4211	1162	51.9975
240	240	160	10	95.9760	1203	79.7408
270	270	180	9	155.8467	1514	102.9219
300	300	200	8	192.6712	1691	113.8803

表 3. 実験結果(MDS).

変数の数			計算結果			
n	m	r	成功	計算時間 [s]	反復回数	時間/反復回数 [ms]
30	30	20	9	2.6244	183	14.2978
60	60	40	9	3.9300	211	18.5962
90	90	60	9	16.2811	446	36.4321
120	120	80	6	23.3367	510	45.7432
150	150	100	4	14.5225	249	58.2648
180	180	120	6	28.0733	444	63.2045
210	210	140	7	58.3986	685	85.1646
240	240	160	6	78.7283	735	107.0891
270	270	180	6	112.6667	908	124.0139
300	300	200	3	220.1367	1380	159.5193

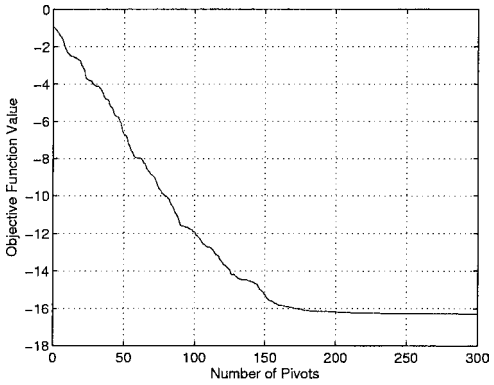

図 1. N' ピボットを用いたときの収束の様子(1).

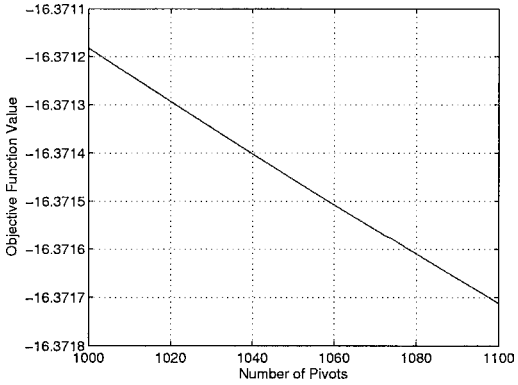

図 2. N' ピボットを用いたときの収束の様子(2).

図 1 に示す．反復回数が 200 回を越えた辺りから収束が極端ににぶくなっていることがわかる．

さらに図 2 は，図 1 の続きで反復回数が 1000 回を越えた部分を観察したものである．減少する量が少ないものの，目的関数値は反復毎にほぼ定数ずつ下がっていることがわかる．すなわち，MDS は収束速度が著しく遅く，前章の実験での限界 2000 回の反復では収束に至らなかったものと考えられる．

このことから，数値実験 I における MDS で収束しなかったいくつかの問題に対して，反復回数の上限を増やして解かせてみたところ，いずれも数千回の反復で最適解に収束した．

図 2 におけるように目的関数が定数ずつ下がっていく状態のときには，実は 4 種類あるピボットのうち，N' ピボットを繰り返している．すなわち，N' ピボットがボトルネックになっていると考えられる．これを改善することが，次章の目的である．

なお，LOS についても，同様にいくつかの問題に対して反復回数を増やして実験を行なった．しかしこの場合には，10 万回を超えてもなお最適値とは遠い状態であった．目的関数値は下がり続けており，最終的には最適解へ収束するのかもしれないが，ROS や MDS に比べると効率が悪すぎると結論づけられる．

4. 数値実験 II：N' ピボットの改良

Muramatsu (2003) では N' ピボットにおいては，基底変数と非基底変数を交換しない，とされている．しかし，この場合，なんらかの変数を B' から見つけてきて基底変数と非基底変数を入れ替えることが可能な場合がある．すなわち，非基底変数として $i \in N'$ を選んだ後，もしある $j \in B'$ が存在して $D_{ji} \neq 0$ となるならば，i と j を交換するのである．これは，u_j $(j \in B')$ と u_i $(i \in N')$ の交換になる．このようにしても，次の基底解は変わらないが，変数を入れ替えることにより，それ以降の辞書が変わり，アルゴリズムの挙動が変わることになる．

この章では上記のように，可能な場合には u_j $(j \in B')$ と u_i $(i \in N')$ の交換を行なうことを考える．(可能でない場合には N' ピボットを行なう.) これを $B'N'$ ピボットと呼ぶことにする．

$B'N'$ ピボットを採用した実装を用いて，先程と同様の実験を再び行なった．その実験結果を表 4–表 6 に示す．

この結果から，$B'N'$ ピボットを採用することによって ROS，MDS 共に 100 % 最適解に収束することがわかる．また，LOS においても前の結果に比べ，若干良い結果を得ることができ

表 4. 実験結果(LOS, $B'N'$ ピボット).

変数の数			計算結果			
n	m	r	成功	計算時間 [s]	反復回数	時間/反復回数 [ms]
30	30	20	4	5.8800	642	9.1518
60	60	40	0	-	-	-
90	90	60	0	-	-	-
120	120	80	0	-	-	-
150	150	100	0	-	-	-
180	180	120	0	-	-	-
210	210	140	0	-	-	-
240	240	160	0	-	-	-
270	270	180	0	-	-	-
300	300	200	0	-	-	-

表 5. 実験結果(ROS, $B'N'$ ピボット).

変数の数			計算結果			
n	m	r	成功	計算時間 [s]	反復回数	時間/反復回数 [ms]
30	30	20	10	0.2910	56	5.1142
60	60	40	10	1.1100	144	7.6710
90	90	60	10	3.7720	296	12.7389
120	120	80	10	9.0990	451	20.1350
150	150	100	10	20.0820	663	30.2485
180	180	120	10	29.3970	776	37.8730
210	210	140	10	53.6920	1014	52.9037
240	240	160	10	94.8220	1202	78.8409
270	270	180	10	152.1290	1346	112.9895
300	300	200	10	211.6010	1606	131.7073

表 6. 実験結果(MDS, $B'N'$ ピボット).

変数の数			計算結果			
n	m	r	成功	計算時間 [s]	反復回数	時間/反復回数 [ms]
30	30	20	10	0.2470	20	12.3500
60	60	40	10	0.9850	46	21.3203
90	90	60	10	2.6530	80	33.0386
120	120	80	10	5.8530	119	48.8564
150	150	100	10	10.7660	162	66.4158
180	180	120	10	20.9440	182	114.8875
210	210	140	10	31.7600	255	124.2081
240	240	160	10	34.5130	283	121.9541
270	270	180	10	59.2440	337	175.4858
300	300	200	10	69.5933	382	181.8113

た．$B'N'$ ピボットは，少なくとも実用的には N' ピボットに勝るものと思われる．

ROS，MDS はどちらも失敗せずに問題を解いているが，注目すべきは，計算時間である．全ての部分問題を解いてからピボット演算を行なう MDS は，1 回の反復に時間がかかることが容易に推測でき，また実際そうなっている．一方，MDS は毎回最も大きく目的関数値を下げ，その結果として収束に至るまでのピボット回数が ROS に比べてはるかに少なくなっている．全体の計算時間は MDS が ROS を下回る結果となった．

なお，N' ピボットの代わりに $B'N'$ ピボットを使っても，Muramatsu (2003) の第 8 章の解析はそのまま適用でき，同論文の Theorem 8.1 はそのまま成立する．すなわち，(一定の仮定のもとで) MDS を使った場合のアルゴリズムの大域的収束を証明できる．これは，Theorem 8.1 の証明が，ピボット演算により目的関数値が減ることと，辞書の構造だけに依存しているからである．詳細は省略する．

5. 終わりに

本研究では，Muramatsu (2003) のアルゴリズムを実装し，$\langle P \rangle$ を解く数値実験を行なった．2 次錐計画に対する単体法的アルゴリズムはいくつか提案されているが，辞書を用いるもので実装可能なものは Muramatsu (2003) だけであり，この論文が我々が知る限りにおいては最初の実装である．様々なピボット選択規則に関して比較検討を行ない，また，$B'N'$ ピボットという新しいピボット演算の方法を提案し，MDS を $B'N'$ ピボットで使うのが最も有効であることを数値的に検証した．このアルゴリズムは，理論的にも N' ピボットを用いたときと変わらない大域的収束の性質を持つ．

$B'N'$ ピボットが N' ピボットより数値的に優れている理由ははっきりしない．どちらを選んでも，次の基底解は変わらず，それ以降の挙動が変化する．数値実験 I で見たように，収束が長引くときには N' ピボットが繰り返されることになるが，このとき，基底・非基底は変化しないため，ずっと同じような構造をした部分問題群を解くことになる．(辞書において，基底・非基底が変化しなければ，D の部分は変化しない．) これにくらべ，$B'N'$ ピボットでは基底・非基底が毎回変化するため，部分問題の構造も係数行列 D の部分を含めて変化する．この多様な変化のおかげで，より早く目的関数を大きく下げるピボットを見つけることができているのかもしれない．詳しい解析は，今後の課題である．

2 次錐計画に対するピボット・アルゴリズムの研究は，理論的にも実用的にもまだまだ未熟である．特に $\langle P \rangle$ に対する双対単体法，2 次錐が複数の場合のピボット演算などは興味深い題材である．今後の展開が期待される．

参 考 文 献

Alizadeh, F. and Goldfarb, D. (2003). Second-order cone programming, *Mathematical Programming Series B*, **95**, 3–51.

Andersen, E. D., Roos, D. and Terlaky, T. (2000). On implementing a primal-dual interior-point method for conic quadratic optimization, Working papers W-274, Helsinki School of Economics and Business Administration.

Ben-Tal, A. and Nemirovski, A. (2001). *Lectures on Modern Convex Optimization*, SIAM, Philadelphia, Pennsylvania.

Dantzig, G. (1963). *Linear Programming and Extensions*, Princeton University Press, Princeton, New Jersey.

Faybusovich, L. (1997). Linear systems in Jordan algebras and primal-dual interior point algorithms, *Journal of Computational and Applied Mathematics*, **86**, 149–175.

Lasserre, J. B. (1996). Linear programming with positive semidefinite matrices, *Mathematical Problems in Engineering*, **2**, 499–522.

Lobo, M. S., Vandenberghe, L., Boyd, S. and Lebret, H. (1998). Second-order cone programming, *Linear Algebra and Applications*, **284**, 193–228.

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

Muramatsu, M. (2002). On a commutative class of search directions for linear programming over symmetric cones, *Journal of Optimization Theory and Applications*, **112**, 595–625.

村松正和(2003). **錐線形計画への招待，システム／制御／情報，47**, 223–228.

Muramatsu, M. (2003). A pivoting procedure for a class of second-order cone programming, Tech. Report CS/00/03, Department of Computer Science, The University of Electro-Communications, Tokyo (to appear in *Optimization Methods and Software*).

Muramatsu, M. and Suzuki, T. (2003). A new second-order cone programming relaxation for MAX-CUT problems, *Journal of Operations Research of Japan*, **46**, 164–177.

Nesterov, Y. and Nemirovski, A. (1993). *Interior Point Polynomial Methods in Convex Programming*: *Theory and Algorithms*, SIAM Publications, Philadelphia, Pennsylvania.

Pataki, G. (1996). Cone-LP's and semidefinite programs: Geometry and a simplex-type method, *Proceedings of the Fifth IPCO Conference on Integer Programming and Combinatorial Optimization*, 161–174, Springer-Verlag, Vancouver.

Sasakawa, T. and Tsuchiya, T. (2003). Optimal magnetic shield design with second-order cone programming, *SIAM Journal on Scientific Computing*, **24**, 1930–1950.

田村明久，村松正和（2002）.**『最適化法』，共立出版，東京.**

360 Proceedings of the Institute of Statistical Mathematics Vol. 53, No. 2, 349–360 (2005)

On Pivot Selection Rules of the Pivoting Method for a Class of Second-order Cone Programming

Keisuke Kurita[1] and Masakazu Muramatsu[2]

[1]Department of Computer Science, Graduate School of Electro-Communications, The University of Electro-Communications
[2]Department of Computer Science, The University of Electro-Communications

A pivoting algorithm for solving a class of second-order cone programming (SOCP) proposed by Muramatsu (2003) is implemented for the first time. Numerical experiments show that the performance of the algorithm heavily depends on the choice of pivot selection rules. Furthermore, we propose an improvement of the pivoting procedure proposed in Muramatsu (2003), which enhances efficiency of the pivoting algorithm. The algorithm performs best with the *maximum decrease strategy* pivoting rule proposed by Muramatsu (2003) and the improved pivoting procedure.

Key words: Second-order cone programming problems, pivoting method.