最適化ソフトウェア SDPA

東京工業大学 中田和秀，中央大学 藤澤克樹，東京工業大学 福田光浩，
神奈川大学 山下真，理化学研究所 中田真秀，海上技術安全研究所 小林和博

概要

The optimization software SDPA which has been developed by our group is a solver for symmetric cone programs. The symmetric cone program is a large scheme which includes linear programs, second-order cone programs and semidefinite programs. It has many applications covering various fields such as combinatorial optimization, systems and control theory, robust optimization and quantum chemistry. Primal-dual interior-point methods, which are polynomial-time algorithms, were proposed to solve symmetric cone programs. SDPA is based on the primal-dual interior-point method. In addition, SDPA utilizes sparsity of data in several ways and parallel computation to solve huge size problems efficiently. Using SDPA, we can obtain the solution of symmetric cone programs easily without knowing the details of the algorithm and its implementation techniques. This paper briefly explain the SDPA and its variants. Then outlines an algorithmic framework of the primal-dual interior-point method.

キーワード：最適化, 対称錐計画問題, 内点法, 並列計算, ソフトウェア
optimization, symmetric cone programming, interior-point method, parallel computing, software

1 はじめに

10 年前や 20 年前と比較して，現在では相当大規模な最適化問題を解くことが可能となり，それに伴い応用範囲も広がってきた．これには大きく分けて 2 つの理由がある．1 つはムーアの法則で示されるような指数関数的な計算機性能の向上であり，もう 1 つは最適化アルゴリズムの発展である．有名な最適化ソフトウェアである CPLEX では，最近 15 年の間に計算機の速度が 1000 倍となり，アルゴリズムの効率性が 2000 倍上がったため，全体として 200 万倍の高速化が実現したという [5].

本稿では，筆者等が現在進行形で開発中の最適化ソフトウェア SDPA について述べる．SDPA は対称錐計画問題を主双対内点法によって解く．もともと内点法は，1984 年 Karmarkar によって線形計画問題に対する多項式時間アルゴリズムとして提案された．多項式オーダーの解法という理論的側面だけでなく，実際に大規模な線形計画問題が解け

るという実用性が明らかとなり，その後多くの研究がなされてきた．現状において，線形計画問題を解く最も優れた内点法は，主双対内点法である．最近の研究により，この主双対内点法は，凸２次関数からなる制約や，行列が半正定値であるという制約を含んだ対称錐計画問題にも適用可能であることが明らかとなった．そして，応用上現れるような大規模な対称錐計画問題を，現実的な計算資源でできる限り効率良く解く研究もさかんに行われている．その結果，変数の数が 100 万から 1000 万程度あるような問題を扱うことが可能となってきた．SDPA はこれらの研究の成果を数多く実装している．よって，SDPA を利用することにより，主双対内点法の詳細や高速化技術を知らなくても，対称錐計画問題の最適解を容易に得ることができる．

本稿の構成は以下の通りである．2 章では，SDPA が対象とする対称錐計画問題について説明をする．3 章では，筆者等が開発した SDPA の紹介を行う．4 章では，対称錐計画問題を解くため，SDPA が実装している主双対内点法の概略を述べる．5 章では，まとめを述べる．

2　対称錐計画問題

本章では，最適化ソフトウェア SDPA で解くことが可能な対称錐計画問題について説明をする．対称錐計画問題は非常に大きなクラスの数理計画問題であり，線形計画問題・凸２次計画問題・２次錐計画問題・半正定値計画問題を全て含んでいる（これら個別の問題については，和文解説 [18, 23] で分かりやすく述べられている）．本稿を通して，大文字のボールド体（$\boldsymbol{X}, \boldsymbol{A}$ など）は行列，小文字のボールド体（$\boldsymbol{x}, \boldsymbol{a}$ など）はベクトル，小文字のイタリック体（x, a など）は実数を意味することにする．

対称錐計画問題の標準形は，アルゴリズムを構築する上では便利な形式であるが，応用問題と対応させるという立場からは分かり辛い面もある．そこで，標準形は 4 章で述べることにし，対称錐計画問題として捉えることのできる問題について説明する．一般に最適化問題は，変数を $\boldsymbol{x}$ とすると，次の左のような形で定式化される．

$$
\begin{array}{lll}
\text{minimize} & f(\boldsymbol{x}) & \\
\text{subject to} & g(\boldsymbol{x}) \leq \boldsymbol{0}, & \\
& h(\boldsymbol{x}) = \boldsymbol{0}, & \\
& \boldsymbol{x} \in S. &
\end{array}
\quad \Longleftrightarrow \quad
\begin{array}{lll}
\text{minimize} & t & \\
\text{subject to} & f(\boldsymbol{x}) \leq t, & \\
& g(\boldsymbol{x}) \leq \boldsymbol{0}, & \\
& h(\boldsymbol{x}) = \boldsymbol{0}, & \\
& \boldsymbol{x} \in S. &
\end{array}
$$

このとき，新しい変数 $t \in \Re$ を用いることにより，右の形式に変換することができる．すると，目的関数は変数に対して線形となり，目的関数 $f(\boldsymbol{x})$ の非線形性による複雑さ（難しさ）を，制約条件に移すことができる．よって，以下では制約条件の形式についてのみ考える．対称錐計画問題として捉えることが可能な制約条件の例を以下で列挙した．なお，$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{b}, a, b, c, k$ は定数であり，残りは変数（定数でもよい）である．

$\boldsymbol{Ax} = \boldsymbol{b},\ \boldsymbol{y} = \boldsymbol{Ax} + \boldsymbol{b}$	ベクトルに対する線形制約
$\boldsymbol{AXB} = \boldsymbol{C},\ \boldsymbol{X} = \boldsymbol{AYB} + \boldsymbol{C}$	行列に対する線形制約
$\boldsymbol{x} \geq \boldsymbol{0}$	ベクトルに対する非負制約
$\boldsymbol{X}$ が半正定値	行列に対する半正定値制約
$\boldsymbol{x}^T\boldsymbol{Ax} + \boldsymbol{b}^T\boldsymbol{x} + c \leq 0$　(ただし $\boldsymbol{A}$は半正定値)	ベクトルに対する凸２次制約
$\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{Y} + \boldsymbol{A}$ が半負定値	行列に対する凸２次制約
$y \geq \|\boldsymbol{x}\|_p \equiv (\sum_{i=1}^n \|x_i\|^p)^{1/p}$　$(p = 1, 2, \cdots)$	ベクトルに対する p ノルム制約
$y \geq \|\boldsymbol{x}\|_\infty \equiv \max_k\{\|x_k\|\}$	ベクトルに対する ∞ ノルム制約
$y \geq \|\boldsymbol{X}\|_p \equiv \max_{\|\boldsymbol{v}\|_p=1} \|\boldsymbol{Xv}\|_p$　$(p = 1, 2, \infty)$	行列に対する p ノルム制約
$x^p \leq y^q,\ x, y \geq 0$　$(p, q = 1, 2, \cdots,\ p \geq q)$	
$\boldsymbol{x}^T\boldsymbol{Ax} \leq yz,\ y \geq 0,\ z \geq 0$　(ただし $\boldsymbol{A}$は半正定値)	
$\boldsymbol{A}_0 + \sum_{i=1}^k \boldsymbol{A}_i x_i$ が半負定値	線形行列不等式 (LMI)
$\sum_{i=1}^k \lambda_i(\boldsymbol{X}) \geq y,\ \boldsymbol{X} = \boldsymbol{X}^T$	対称行列の固有値に基づいた制約
$\sum_{i=1}^k \sigma_i(\boldsymbol{X}) \geq y$	行列の特異値に基づいた制約

ここで，$\lambda_k(\boldsymbol{X})$ は対称行列 $\boldsymbol{X}$ の中の（重複を含め）k 番目に小さな固有値，$\sigma_k(\boldsymbol{X})$ は行列 $\boldsymbol{X}$ の中の（重複を含め）k 番目に小さな特異値を意味する．

これらの制約条件以外にも，対称錐計画問題に帰着可能な制約条件が数多く知られている．詳細は [4] などを参照して頂きたい．なお，$\boldsymbol{AX} + \boldsymbol{XA}^T$ が負定値となるような対称行列 $\boldsymbol{X}$ の存在性は，$\boldsymbol{AX} + \boldsymbol{XA}^T - t\boldsymbol{I}$ が半負定値という制約条件の元で，変数 t を最小化する対称錐計画問題を解くことで判断できる．

上記の制約を満たす変数の領域は全て凸集合である．対称錐計画問題は凸計画問題であり，対称錐計画問題のクラスは凸計画問題のクラスより真に小さい．しかしながら，実用上現れるような凸計画問題は対称錐計画問題に帰着できることが多い．

3　SDPA の紹介

現在，対称錐計画問題を解くためのソフトウェアやハードウェア環境が整備されつつある．本章では，対称錐計画問題を解くソフトウェアである SDPA や，その周辺について述べる．

3.1　SDPA について

筆者等のグループは，SDPA という対称錐計画問題を解くソフトウェアを開発している．これは小島政和氏 (東京工業大学) や二方克昌氏 (Virginia 大学) と共同で研究を行っている．全てが完全にオープンソースであり，GNU General Public License の下で配布している．マニュアルも完備しているので，興味ある方は是非利用して欲しい．ソース

コードはホームページ (SDPA Online for your future)

`http://sdpa.indsys.chuo-u.ac.jp/sdpa/`

からダウンロードすることができる．SDPA は主双対内点法を実装しているが，その概要については次章で説明をする．以下に，SDPA の特徴を列挙した（なお，一部の機能は開発中である）.

- プログラミング言語 C++ で実装しており，MATLAB などの有料パッケージを必要としない
- Mehrotra の予測子修正子法を用いたパス追跡タイプの主双対内点法で解く
- ライブラリ関数として呼ぶことも可能である
- FreeBSD 上では、ports 化されているので導入が簡単
- MATLAB インターフェイスもあり，MATLAB 上からも利用できる
- MPI と ScaLAPACK を利用して，並列計算も可能である
- 必要に応じて，GMP(The GNU Multiple Precision Arithmetic Library) を用いた多倍長演算も可能である
- 重み付き対数行列式問題にも対応している
- マルチスレッド対応である

ソフトウェアは，基本版となる SDPA を始め，その派生版である SDPA-C, SDPARA, SDPARA-C, SDPA-M, SDPA-GMP などがある．複数のソフトウェアがある理由は，解く SDP 問題の性質や使用する計算環境あるいは必要とする情報によって，ベストなソフトウェアが異なってくるためである.

- SDPA（SemiDefinite Programming Algorithm)
  基本版であり，多くの問題に対して有効 [35]
- SDPA-C（SDPA - Completion method)
  行列補完理論を利用し，特殊な構造を持った問題を効率良く解く [13, 24]
- SDPARA（SDPA paRAllel version)
  SDPA の並列計算版 [36]
- SDPARA-C（SDPA paRAllel version - Completion method)
  SDPA-C の並列計算版 [25]
- SDPA-M（SDPA - Matlab interface)
  SDPA を MATLAB 上から実行できるようにしたもの
- SDPA-GMP（SDPA - GMP)
  GMP を利用した多倍長計算により，高精度な解を求めることができる

基本版の SDPA は比較的安定して効率的な計算を行うため，多くの問題に適している．だが，大規模問題を解くには時間がかかることも多い．その場合には，問題の性質を調べた上で，適切な派生版の使用を検討するとよい.

3.2 大規模問題を解くために

SDPA は対称錐計画問題を主双対内点法によって解く．しかし，主双対内点法をそのまま実装しただけでは，変数の次元や制約の数が数千を越える応用問題を，現実的な計算時間内で解くことはできない．このような問題を解くためには，ソフトウェアとハードウェアの両面で様々な工夫が必要となる．

まず最初に，ソフトウェア（アルゴリズム）面での高速化について述べる．応用上現れるような大規模な対称錐計画問題の多くは，疎な性質を持っている．これは，各変数間に不完全な独立性があることを意味する．このとき，影響を与えない他の変数を一時的に無視するようなアルゴリズムとデータ構造を実装することが不可欠となる．ただし，主双対内点法のアルゴリズム全体では，全ての変数が影響しあっているため，なるべく変数間の独立性を保ちつつ疎性を悪化させないように計算を進めていくには，様々な工夫が必要である．これらに関しては多くの研究があり [12, 13, 24] などを参照して頂きたい．

参考までに，ベンチマーク問題集 SDPLIB [7] にある大規模問題において，対称錐計画問題の疎性（行列の中の非ゼロ要素の割合）を表 1 で示した．ほとんどの要素がゼロであることが確認できる．

表 1　SDPLIB の問題の疎性

非ゼロの割合	$10^{-2} \sim 10^{-3} \sim 10^{-4} \sim 10^{-5} \sim 10^{-6} \sim 10^{-7} \sim 10^{-8}$	計
問題数	4 問　3 問　5 問　6 問　2 問　1 問	21 問

一方，ハードウェアの面で有効なのは，PC クラスタなどを用いた並列計算である．しかしながら，ある程度の規模の並列計算環境を準備することは，一般ユーザに取って大変困難である．このため，筆者等のグループは，SDPA Online Solver という対称錐計画問題を解くためのハードウェア環境を提供している．これは，ユーザがインターネットを介して解きたい対称錐計画問題を送ると，システム側が備えている計算サーバ上で解き，その計算結果をユーザに知らせてくれるシステムである．今すぐに問題を解きたい，ソースコードをコンパイルする手間を省きたい，MATLAB が使えない等のユーザにもお薦めである．

SDPA Online Solver は，グリッド計算技術を用いた 3 層のクライアント・サーバシステムである．グリッド技術を用いているため，地理的に分散した計算リソースを動的に制御することができ，拡張性に富むという運用面からみた特徴も持っている．SDPA Online Solver のホームページの URL は次の通りである．

`http://sdpa.indsys.chuo-u.ac.jp/portal/`

MPI が動作する並列計算環境が用意できる方は，SDPA のホームページで公開されて

いる並列計算用ソフトウェア SDPARA や SDPARA-C のソースコードをダウンロードし，自前の環境で大規模な対称錐計画問題を解くことも可能である．これらのソフトウェアでは，効率よく並列計算させるために，各計算機になるべく均等に計算を割り振り，かつ計算機間の通信量を少なくするような設計がなされている．その結果，並列計算において高水準のスケーラビリィティが達成されている ([25, 36])．また，SDPA に数値演算ライブラリ GotoBLAS [19] をリンクさせることにより，マルチコア・プロセッサ上でマルチスレッドによる並列計算を行うこともできる．

3.3 モデリング言語

SDPA をライブラリでなく単体で使用する場合，まず解きたい問題を SDPA の入力形式で記述する必要がある．しかし，入力データを作成することはかなりの負担である．もし，MATLAB が使える環境であれば，YALMIP [20] というモデリング言語を使用すると大変便利である．例えば，次のような問題を考えよう．変数は $n \times n$ の対称行列 $\boldsymbol{X}$ と $\boldsymbol{Y}$ で，定数は $n \times n$ の対称行列 $\boldsymbol{A}$ と $\boldsymbol{B}$，並びに n 次元の単位行列 $\boldsymbol{I}$ である．問題中の $\boldsymbol{A} \succeq \boldsymbol{B}$ は $\boldsymbol{A} - \boldsymbol{B}$ が半正定値行列であることを意味し，$\boldsymbol{A} \geq \boldsymbol{O}$ は $\boldsymbol{A}$ の各成分が非負であることを意味する．

$$
\begin{array}{ll}
\text{maximize} & \mathrm{Tr}(\boldsymbol{Y}) \\
\text{subject to} & \boldsymbol{X} \succeq \boldsymbol{O}, \ \ \boldsymbol{Y} \succeq \boldsymbol{O}, \\
& \boldsymbol{X} + \boldsymbol{Y} = \boldsymbol{I}, \ \ \boldsymbol{X} \geq \boldsymbol{O}, \\
& (\boldsymbol{A}\boldsymbol{X} + \boldsymbol{X}\boldsymbol{A}) \preceq \boldsymbol{B}.
\end{array}
$$

例えば $n = 50$ の場合，この問題を標準形に変換し，SDPA 形式の入力データを作成すると，データファイルは 5 万行を越える．すなわち，入力データを生成するために別のプログラムが必要となる．しかし，YALMIP を使うと，次のような入力だけで解くことができる．

```
X = sdpvar(n,n);
Y = sdpvar(n,n);
I = eye(n);
F1 = set(X >= 0)  + set(Y >= 0);
F2 = set(X+Y == I) + set(X(:) >= 0);
F3 =  set(A*X + X*A <= B);
solvesdp(F1+F2+F3,trace(Y));
```

最初の 6 行で問題を記述しており，最後の 1 行によって前節で述べたソフトウェアを用いて問題を解いている．MATLAB をご存じの方は，おおよその意味を理解できるのではないだろうか．１つ注意しておくと，問題の記述が容易になっても，対称錐計画問題を解くことが容易になるわけではない．上記の問題を解くのに必要な時間は，行列変数の次元 n

に対して $\mathcal{O}(n^6)$ 程度になることが予想される．現在の PC では，n が 50 から 100 程度の問題を解くのが限界であろう．

3.4 他のソフトウェア

SDPA の他にも対称錐計画問題を解くソフトウェアは，数多く開発されてきた．その中で，筆者が知っているものを以下に記した．これらは，無償／有償，公開／非公開，ソースコードの配布／バイナリでの配布，現在も開発中／開発終了など様々である．

SDPA [35], SDPARA [36], SDPA-C [13, 24] SDPARA-C [25] SeDuMi [29], SDPT3 [31], CSDP [6], DSDP [3], PDSDP [2], PENBMI, PENSDP [16], SDPLR [9], SBmethod [14], SP [32], SDPSOL [34], SDPHA [8], SDPPack [1]

これらのソフトウェアの優劣を，一概に言うことはできない．１つの参考材料としては，H. D. Mittelmann のホームページの情報が利用できる．

`http://plato.la.asu.edu/bench.html`

このホームページには，幾つかのソフトウェアで，標準ベンチマーク集である SDPLIB [7] の問題などを解いた包括的な結果が掲載されている．それを見る限り，各ソフトウェアはそれぞれが得意としている問題群があるようである．

なお，前節で述べた YALMIP は CSDP, DSDP, PENSDP, SDPA, SDPLR, SDPT3, SeDuMi に対応している．

4 主双対内点法

本章では，SDPA が実装している主双対内点法の概略を述べる．

4.1 対称錐計画問題の標準形

対称錐計画問題は，[11, 26, 27] で提案された数理計画問題である．まず，対称錐（等質自己双対錐とも呼ばれる）を定義しよう．有限次元計量実線形空間 V に対し，部分集合 $\mathcal{K}$ が次の性質を満たすとき，対称錐と呼ぶ．

- $\forall \boldsymbol{x} \in \mathcal{K},\ \forall \alpha \geq 0,\ \ \alpha \boldsymbol{x} \in \mathcal{K}$
- $\mathcal{K}^* \equiv \{\boldsymbol{x} \in V \mid \forall \boldsymbol{y} \in \mathcal{K}, \langle \boldsymbol{x}, \boldsymbol{y} \rangle \geq 0\} = \mathcal{K}$
- $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathrm{int}(\mathcal{K}),\ \exists g \in GL(V),\ g\mathcal{K} = \mathcal{K},\ g\boldsymbol{x} = \boldsymbol{y}$

ここで，$\mathrm{int}(\mathcal{K})$ は対称錐 $\mathcal{K}$ の内部を意味する．$GL(V)$ は V 上での一般一次変換群である．対称錐計画問題の標準形は，（適切な内積の元で）対称錐 $\mathcal{K}$ を用いた次のような形式で表すことができる．

$$
\begin{array}{lll}
\textbf{主問題} & & \\
\text{maximize} & \langle \boldsymbol{c}, \boldsymbol{x} \rangle & \\
\text{subject to} & \mathcal{A}\boldsymbol{x} = \boldsymbol{b}, & \\
& \boldsymbol{x} \in \mathcal{K}. &
\end{array}
\qquad
\begin{array}{lll}
\textbf{双対問題} & & \\
\text{minimize} & \langle \boldsymbol{b}, \boldsymbol{y} \rangle & \\
\text{subject to} & \boldsymbol{z} = \mathcal{A}^* \boldsymbol{y} - \boldsymbol{c}, & \\
& \boldsymbol{z} \in \mathcal{K}. &
\end{array}
\tag{1}
$$

ここで，変数は，$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ であるが，$\boldsymbol{x}, \boldsymbol{z}$ は $\mathcal{K}$ の存在する V の要素であり，$\boldsymbol{y}$ は別の有限次元計量空間 W に属する．また，$\mathcal{A}$ は V から W への線形写像，$\mathcal{A}^*$ は $\mathcal{A}$ の随伴で，W から V への線形写像である．2 章では様々な制約を挙げたが，数理計画問題としての難しさを生じさせる非線形性や不等式条件は，対称錐 $\mathcal{K}$ の部分で吸収される枠組みであることを注意しておく．

(1) の両問題の双対性により，次に示すような弱双対定理と（強）双対定理が成り立つ．$\tilde{\boldsymbol{x}}$ を主問題の実行可能解 $(\mathcal{A}\tilde{\boldsymbol{x}} = \boldsymbol{b},\ \tilde{\boldsymbol{x}} \in \mathcal{K})$，$\tilde{\boldsymbol{y}}, \tilde{\boldsymbol{z}}$ を双対問題の実行可能解 $(\tilde{\boldsymbol{z}} = \mathcal{A}^*\tilde{\boldsymbol{y}} - \boldsymbol{c}, \tilde{\boldsymbol{z}} \in \mathcal{K})$ としよう．このとき，弱双対定理は

$$
\langle \boldsymbol{c}, \tilde{\boldsymbol{x}} \rangle \leq \langle \boldsymbol{b}, \tilde{\boldsymbol{y}} \rangle
$$

という不等式が成立することを述べている．本稿では以下，実行可能内点 ($\boldsymbol{z} = \mathcal{A}^*\boldsymbol{y} - \boldsymbol{c},\ \mathcal{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \in \mathrm{int}(\mathcal{K}),\ \boldsymbol{z} \in \mathrm{int}(\mathcal{K})$) が存在するという Slater の想定条件を仮定する．すると（強）双対定理より，主問題の最適解を $\boldsymbol{x}'$，双対問題の最適解を $\boldsymbol{y}', \boldsymbol{z}'$ としたとき，

$$
\langle \boldsymbol{c}, \boldsymbol{x}' \rangle = \langle \boldsymbol{b}, \boldsymbol{y}' \rangle
$$

という関係が成り立つ（詳細については [30] の 4 章を参照してもらいたい）．従って，対称錐計画問題 (1) を解くことは，

$$
\begin{cases}
\boldsymbol{z} = \mathcal{A}^*\boldsymbol{y} - \boldsymbol{c} \\
\mathcal{A}\boldsymbol{x} = \boldsymbol{b} \\
\langle \boldsymbol{c}, \boldsymbol{x} \rangle = \langle \boldsymbol{b}, \boldsymbol{y} \rangle \\
\boldsymbol{x} \in \mathcal{K},\ \boldsymbol{z} \in \mathcal{K}
\end{cases}
\tag{2}
$$

という方程式を満たす $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ を見つけることと等価である．

4.2 Euclid 的 Jordan 代数

次節で主双対内点法を説明する道具として，Euclid 的 Jordan 代数を導入する．Euclid 的 Jordan 代数 $(V, \circ)$ により，対称錐計画問題を線形計画問題の拡張として統一的に扱うことが可能となる．ここで，V は有限次元計量実線形空間，$\circ$ は以下を満たす $V \times V \to V$ の 2 項演算である。

- $\boldsymbol{x} \circ \boldsymbol{y} = \boldsymbol{y} \circ \boldsymbol{x}$
- $\forall \alpha, \beta \in \Re,\ (\alpha\boldsymbol{x}_1 + \beta\boldsymbol{x}_2) \circ \boldsymbol{y} = \alpha\boldsymbol{x}_1 \circ \boldsymbol{y} + \beta\boldsymbol{x}_2 \circ \boldsymbol{y}$
- $\boldsymbol{x} \circ ((\boldsymbol{x} \circ \boldsymbol{x}) \circ \boldsymbol{y}) = (\boldsymbol{x} \circ \boldsymbol{x}) \circ (\boldsymbol{x} \circ \boldsymbol{y})$
- $\boldsymbol{x} \circ \boldsymbol{x} + \boldsymbol{y} \circ \boldsymbol{y} = \boldsymbol{0} \implies \boldsymbol{x} = \boldsymbol{y} = \boldsymbol{0}$

このとき，以下の関係を満たすベクトルとして，単位元 $\boldsymbol{e}$ と（存在すれば）逆元 $\boldsymbol{x}^{-1}$ を定義することができる．

$$\boldsymbol{x} \circ \boldsymbol{e} = \boldsymbol{x}, \quad \boldsymbol{e} \circ \boldsymbol{x} = \boldsymbol{x}, \quad \boldsymbol{x} \circ \boldsymbol{x}^{-1} = \boldsymbol{x}^{-1} \circ \boldsymbol{x} = \boldsymbol{e}.$$

また $\boldsymbol{x} \in V$ は，$\boldsymbol{x} = \sum_{k=1}^{r} \lambda_k \boldsymbol{c}_k$ という形式でスペクトル分解できることができる．このとき，$\boldsymbol{x}$ と $\boldsymbol{y}$ の内積 $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ を，$\langle \boldsymbol{x}, \boldsymbol{y} \rangle \equiv \sum_{k=1}^{r} \lambda_k (\boldsymbol{x} \circ \boldsymbol{y})$ と定義する．

１つの対称錐 $\mathcal{K}$ に対し，$\mathcal{K} = \{\boldsymbol{x} \circ \boldsymbol{x} \mid \forall \boldsymbol{x} \in V\}$ となるような Euclid 的 Jordan 代数 $(V, \circ)$ が存在する．逆に，ある Euclid 的 Jordan 代数 $(V, \circ)$ に対し，$\mathcal{K} = \{\boldsymbol{x} \circ \boldsymbol{x} \mid \forall \boldsymbol{x} \in V\}$ は対称錐となることが知られている．Euclid 的 Jordan 代数に関して多くの研究成果が知られているが，本稿では深入りをしない．詳細に興味がある方は，[10, 21] などをご覧頂きたい．

4.3 主双対内点法の概要

Euclid 的 Jordan 代数を用いると，対称錐計画問題 (1) の最適性条件 (2) は，次の方程式に書き換えることができる．

$$\begin{cases} \boldsymbol{z} = \mathcal{A}^* \boldsymbol{y} - \boldsymbol{c} \\ \mathcal{A}\boldsymbol{x} = \boldsymbol{b} \\ \boldsymbol{x} \circ \boldsymbol{z} = \boldsymbol{0} \\ \boldsymbol{x} \in \mathcal{K}, \ \boldsymbol{z} \in \mathcal{K}. \end{cases}$$

最初の 3 つの等式に着目すると，等式の数は $2 \dim(V) + \dim(W)$ であり，これは変数 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ の次元と等しい．ただし，４番目の $\boldsymbol{x} \in \mathcal{K}, \ \boldsymbol{z} \in \mathcal{K}$ の条件のため，最初の 3 つの等式からなる方程式系に単純に Newton 法を適用するだけでは，一般には最適解は得られない．最適解においては，その方程式系の Hesse 行列には逆行列も存在しない．

そのため，最適解に収束するような曲線（中心パス）を構築し，その曲線に沿って進みながら最適解に近づこうとするのが，以下で説明するパス追跡タイプの主双対内点法である．任意の $\mu > 0$ に対し，

$$\begin{cases} \boldsymbol{z} = \mathcal{A}^* \boldsymbol{y} - \boldsymbol{c} \\ \mathcal{A}\boldsymbol{x} = \boldsymbol{b} \\ \boldsymbol{x} \circ \boldsymbol{z} = \mu \boldsymbol{e} \\ \boldsymbol{x} \in \mathrm{int}(\mathcal{K}), \ \boldsymbol{z} \in \mathrm{int}(\mathcal{K}) \end{cases}$$

の解は一意であることが証明できるため，それを $(\boldsymbol{x}(\mu), \boldsymbol{y}(\mu), \boldsymbol{z}(\mu))$ とする．このとき，中心パスは次のように定義される．

$$\{(\boldsymbol{x}(\mu), \boldsymbol{y}(\mu), \boldsymbol{z}(\mu)) \in V \times W \times V \ \mid \ \mu > 0\}$$

この曲線は滑らかであり，$\mu \to 0$ のとき，最適解（の 1 つ）に収束することが知られている．よって，中心パスの近傍を μ が 0 となる方向に進めば，最適解に近づいていくことが期待できる．アルゴリズムの概略を以下で示した（図 1 も参照）．

パス追跡タイプの主双対内点法

Step 0: 初期反復点 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \in \mathrm{int}(\mathcal{K}) \times W \times \mathrm{int}(\mathcal{K})$ と, パラメータ $\mu > 0$, 定数 $0 < \sigma < 1$ を選ぶ.

Step 1: もし, 現在の反復点 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ が終了条件を満たしていたならば, 反復を終了する.

Step 2: 最適解に近い中心パス上の点（ターゲット）に向かう Newton 方向 $(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z}) \in V \times W \times V$ を計算する.

Step 3: 次の反復点 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) + \alpha(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z})$ が, 領域 $\mathrm{int}(\mathcal{K}) \times W \times \mathrm{int}(\mathcal{K})$ に留まるようなステップサイズ α を計算する.

Step 4: 反復点 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ と, パラメータ μ を更新し, Step 1 に戻る.
$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) := (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) + \alpha(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z}), \quad \mu := \sigma\mu.$

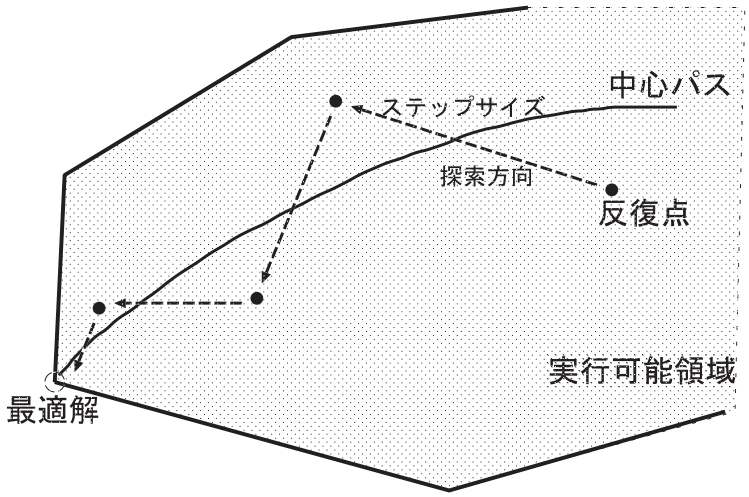

図 1　主双対内点法の概略

このアルゴリズムにおいて, 反復点の $\boldsymbol{x}$ と $\boldsymbol{z}$ は必ず対称錐 $\mathcal{K}$ の内点となっていることに注目してもらいたい. これが, この解法が内点法と呼ばれる理由である. 多項式時間での収束性を証明する際は, 探索方向 $(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z})$, ステップサイズ α, 定数 σ の 3 点をどう決定するかが重要なポイントとなる. しかしながら, 後者 2 つに関しては, ステップサイズ α は最大に取れる値の 0.9 倍程度, 定数 σ は 0.01 から 0.1 程度とした方が, 最適解に速く収束することが多い. よって, 実装するときは, そのような値に設定するとよい. また, 探索方向 $(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z})$ の計算は, 主双対内点法の計算時間の大部分を占める. そのため, 少ない計算量で求めることが可能な HKM 方向 [15, 17, 22] や NT 方向 [26, 27] がよく利用されている. 本章での説明を終えるにあたり, 内点法の詳細を知りたい方に対し, それを専門的に論じている書籍として [4, 11, 18, 28, 33] を紹介しておく.

5 おわりに

本稿では，対称錐計画問題を解くためのソフトウェア SDPA の紹介を行った．また，SDPA が実装している主双対内点法の概略について説明をした．対称錐計画問題のソルバーは世界中で数多く開発されているが，その中でも SDPA は大規模な問題をより高速に解くことが実証されている．対称錐計画問題を解くための敷居が低くなるにつれ，制御工学・構造解析・統計・量子化学・量子情報・金融工学・データマイニングなど，その利用範囲が広がってきた．筆者等が提供しているソフトウェアやハードウェア環境が，さらに多くの応用問題の解決に役立つことを期待している．

参考文献

[1] F. Alizadeh, J.-P.A. Haeberly, M.V. Nayakkankuppam, M.L. Overton and S. Schmieta, SDPPACK user's guide, Computer Science Department, NYU, New York, Technical Report, 1997.
Available at http://cs.nyu.edu/overton/software/sdppack/sdppack.html

[2] S.J. Benson, Parallel computing on semidefinite programs, Preprint ANL/MCS-P939-0302, 2002.
Available at http://www.msc.anl.gov/~benson/dsdp/pdsdp.ps

[3] S.J. Benson, Y. Ye and X. Zhang, Solving large-scale sparse semidefinite programs for combinatorial optimization, *SIAM Journal on Optimization*, 10 (2000) 443–461.

[4] A. Ben-Tal and A. Nemirovskii, Lectures on modern convex optimization: analysis, algorithms, and engineering applications, *SIAM*, Philadelphia (2001).

[5] R. Bixby, Real-world linear programs: a decade and more of progress, *Operations Research*, 50 (2002) 3 – 15.

[6] B. Borchers, CSDP 2.3 user's guide, *Optimization Methods and Software*, 11 & 12 (1999) 597–611. Available at http://www.nmt.edu/~borchers/csdp.html.

[7] B. Borchers, SDPLIB 1.2, a library of semidefinite programming test problems, *Optimization Methods and Software*, 11 & 12 (1999) 683-690.

[8] N. Brixius, F.A. Potra and R. Sheng, SDPHA a MATLAB implementation of homogeneous interior-point algorithms for semidefinite programming, *Optimization Methods and Software*, 11 & 12, (1999) 583–596.
Available at http://www.cs.uiowa.edu/~brixius/SDPHA/.

[9] S. Burer and R. D. C. Monteiro, A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization, *Mathematical Programming*, 95

(2003) 329–357.

[10] J. Faraut and A. Koranyi, "Analysis on Symmetric Cones," *Clarendon Press*, Oxford (1994).

[11] L. Faybusovich, "Linear systems in Jordan algebras and primal-dual interior-point algorithms," Journal of computational and applied mathematics, 86 (1997) 149–175.

[12] K. Fujisawa, M. Kojima and K. Nakata, Exploiting sparsity in primal-dual interior-point methods for semidefinite programming, *Mathematical Programming*, 79 (1997) 235–253.

[13] M. Fukuda, M. Kojima, K. Murota and K. Nakata, Exploiting sparsity in semidefinite programming via matrix completion I: General framework, *SIAM Journal on Optimization*, 11 (2000) 647–674.

[14] C. Helmberg and F. Rendl, A spectral bundle method for semidefinite programming, *SIAM Journal on Optimization*, 10 (2000) 673–696.

[15] C. Helmberg, F. Rendl, R. J. Vanderbei and H. Wolkowicz, An interior-point method for semidefinite programming, *SIAM Journal on Optimization*, 6 (1996) 342–361.

[16] M. Kočvara and M. Stingl, PENNON - a code for convex nonlinear and semidefinite programming, *Optimization Methods and Software*, 18 (2003) 317–333.

[17] M. Kojima, S. Shindoh and S. Hara, Interior-point methods for the monotone semidefinite linear complementarity problem in symmetric matrices, *SIAM Journal on Optimization*, 7 (1997) 86–125.

[18] 小島正和, 土谷隆, 水野眞治, 矢部博, 内点法, 朝倉書店 (2001).

[19] Kazushige Goto, GotoBLAS, Available at http://www.tacc.utexas.edu/resources/software/.

[20] J. Löfberg, YALMIP : a toolbox for modeling and optimization in MATLAB, *Proceedings of the CACSD Conference* (2004),
Available at http://control.ee.ethz.ch/ joloef/yalmip.php.

[21] K. McCrimmon, A Taste of Jordan Algebras, Springer, New York (2000).

[22] R. D. C. Monteiro, Primal-dual path-following algorithms for semidefinite programming, *SIAM Journal on Optimization*, 7 (1997) 663–678.

[23] 村松正和, 2次錐計画問題とその応用, 第 15 回 RAMP シンポジウム論文集, (2003) 8–21.

[24] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima and K. Murota, Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results, *Mathematical Programming*, 95 (2003) 303–327.

[25] K. Nakata, M. Yamashita, K. Fujisawa and M. Kojima, A parallel primal-dual interior-point method for semidefinite programs using positive definite matrix

completion, *Parallel Computing*, to appear.

[26] Yu. E. Nesterov and M. J. Todd, Self-scaled barriers and interior-point methods for convex programming, *Mathematics of Operations Research*, 22 (1997) 1–42.

[27] Yu. E. Nesterov and M. J. Todd, Primal-dual interior-point methods for self-scaled cones, *SIAM Journal on Optimization*, 8 (1998) 324–364.

[28] J. Renegar, A Mathematical View of Interior-Point Methods in Convex Optimization, SIAM, Philadelphia, USA, (2001).

[29] J. F. Sturm, Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones, *Optimization Methods and Software*, 11 & 12 (1999) 625–653.

[30] 田村 明久, 村松 正和, 最適化法, 共立出版株式会社 (2002).

[31] K. C. Toh, M. J. Todd and R. H. Tütüncü, SDPT3 — a MATLAB software package for semidefinite programming, version 1.3, *Optimization Methods and Software*, 11 & 12 (1999) 545–581.
Available at http://www.math.nus.edu.sg/~mattohkc/.

[32] L. Vandenberghe and S. Boyd. sp: Software for Semidefinite Programming. User's Guide, Beta Version. Stanford University, October 1994.

[33] H. Wolkowicz, R. Saigal and L. Vandenberghe, eds., Handbook of Semidefinite Programming: Theory, Algorithms, and Applications, *Kluwer Academic Publishers*, Massachusetts (2000).

[34] S.-P. Wu and S. Boyd, sdpsol: A parser/solver for semidefinite programs with matrix structure, In Recent Advances in LMI Methods for Control (2000) 79–91.

[35] M. Yamashita, K. Fujisawa and M. Kojima, Implementation and evaluation of SDPA6.0 (SemiDefinite Programming Algorithm 6.0), *Optimization Methods and Software*, 18 (2003) 491–505.

[36] M. Yamashita, K. Fujisawa and M. Kojima, SDPARA: SemiDefinte Programming Algorithm PARAllel version, *Parallel Computing*, 29 (2003) 1053–1067.