主双対内点法

中田　和秀

線形計画問題に対する主双対内点法は多項式時間アルゴリズムであるという理論的特徴をもつ．さらに，実際に大規模な問題が高速に解けるという実用性もあり，多くのソフトウェアで実装されている．本稿では，主双対内点法の中でもパス追跡法を取り上げ，実用的な内点法アルゴリズムと，多項式時間での収束性の両面について解説する．

キーワード：線形計画法，内点法，主双対アルゴリズム，パス追跡法

1. はじめに

線形計画問題に対する内点法は理論的に計算量が問題サイズの多項式で抑えらることが保証されているだけでなく，実際に大規模な問題が高速に解けるという，理論と実用の両面で重要な手法である．内点法には，主問題を解く内点法，双対問題を解く内点法，主問題と双対問題を同時に解く内点法がある．その中でも本稿では，主問題と双対問題を同時に解く主双対内点法について解説する．主双対内点法は，主問題や双対問題を単独で解くよりも多くの優れた特徴をもち，線形計画問題を内点法で解くソフトウェアの大半で実装されている．なお，主双対内点法の研究は小島政和氏，水野眞治氏，吉瀬章子氏という日本人研究者が主導し，その後の理論展開にも深く関わっていることを強調しておく．

2. 標準形と最適性条件

次の形式の線形計画問題の主問題と双対問題を考える．

$$\begin{aligned} \text{主問題：}\quad & \text{最小化} && \boldsymbol{c}^\top \boldsymbol{x} \\ & \text{制約条件} && A\boldsymbol{x} = \boldsymbol{b}, \\ & && \boldsymbol{x} \geq \boldsymbol{0}. \end{aligned} \tag{1}$$

$$\begin{aligned} \text{双対問題：}\quad & \text{最大化} && \boldsymbol{b}^\top \boldsymbol{y} \\ & \text{制約条件} && A^\top \boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}, \\ & && \boldsymbol{z} \geq \boldsymbol{0}. \end{aligned} \tag{2}$$

ここで，$A \in \mathbb{R}^{m\times n}$，$\boldsymbol{b} \in \mathbb{R}^m$，$\boldsymbol{c} \in \mathbb{R}^n$ は定数，$\boldsymbol{x} \in \mathbb{R}^n$，$\boldsymbol{y} \in \mathbb{R}^m$，$\boldsymbol{z} \in \mathbb{R}^n$ は変数である．一般性を失うことなく，A は行フルランクだと仮定できる．$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ が主問題 (1) と双対問題 (2) の制約条件を満たすとき，それを実行可能解と呼ぶ．さらに条件 $\boldsymbol{x} > \boldsymbol{0}$，$\boldsymbol{z} > \boldsymbol{0}$ も満たすとき，実行可能内点解と呼ぶ．また，$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ が $\boldsymbol{x} > \boldsymbol{0}$，$\boldsymbol{z} > \boldsymbol{0}$ のみを満たすとき，それを（非実行可能）内点解と呼ぶ．これらの主問題と双対問題には，次に述べる弱双対定理と双対定理が成り立つ．

弱双対定理　主問題と双対問題の実行可能解 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ において，次の関係がある．

$$\boldsymbol{c}^\top \boldsymbol{x} \geq \boldsymbol{b}^\top \boldsymbol{y}$$

双対定理　主問題と双対問題が実行可能解をもつならば最適解が存在し，両問題の最適値は一致する．すなわち，最適解 $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{z}^*)$ において次の関係がある．

$$\boldsymbol{c}^\top \boldsymbol{x}^* = \boldsymbol{b}^\top \boldsymbol{y}^*$$

弱双対定理の証明は，$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ が主問題と双対問題の制約条件を満たすことから，

$$\begin{aligned} \boldsymbol{c}^\top \boldsymbol{x} - \boldsymbol{b}^\top \boldsymbol{y} &= (A^\top \boldsymbol{y} + \boldsymbol{z})^\top \boldsymbol{x} - (A\boldsymbol{x})^\top \boldsymbol{y} \\ &= \boldsymbol{z}^\top \boldsymbol{x} \\ &= \sum_{i=1}^{n} x_i z_i \\ &\geq 0 \end{aligned}$$

が成り立つことでわかる．双対定理の証明は森と松井 [1] などを参照してもらいたい．双対定理より，（実行可能であれば）主問題と双対問題の最適値は等しい．逆に弱双対定理より，目的関数値が等しい実行可能解

なかた　かずひで
東京工業大学工学院
〒 152–8552　東京都目黒区大岡山 2–12–1 W9–60
nakata.k.ac@m.titech.ac.jp

は最適解となる．よって，$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ が主問題 (1) と双対問題 (2) の最適解であることの必要十分条件は次のようになる．

$$\begin{cases} A\boldsymbol{x} = \boldsymbol{b}, \\ A^\top \boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}, \\ \boldsymbol{c}^\top \boldsymbol{x} = \boldsymbol{b}^\top \boldsymbol{y}, \\ \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{z} \geq \boldsymbol{0}. \end{cases}$$

3 番目の条件は，弱双対定理の証明中の不等式が等式で成り立つことを意味するため，

$$\begin{aligned} \boldsymbol{c}^\top \boldsymbol{x} = \boldsymbol{b}^\top \boldsymbol{y} \quad &\Longleftrightarrow \quad x_i z_i = 0 \ \ (i = 1, 2, \ldots, n) \\ &\Longleftrightarrow \quad X\boldsymbol{z} = \boldsymbol{0} \end{aligned}$$

と書き換えることができる．なお，X は $\boldsymbol{x}$ の各要素を対角部分に並べた行列である．$x_i z_i = 0\ (i = 1, 2, \ldots, n)$ や $X\boldsymbol{z} = \boldsymbol{0}$ は相補性条件（相補スラック条件）と呼ばれる．以上の議論より，主問題 (1) と双対問題 (2) の最適解であるための必要十分条件は

$$\begin{cases} A\boldsymbol{x} = \boldsymbol{b}, \\ A^\top \boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}, \\ X\boldsymbol{z} = \boldsymbol{0}, \\ \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{z} \geq \boldsymbol{0} \end{cases} \tag{3}$$

だとわかる．

3. 主双対内点法の概要

主双対内点法として，アフィンスケーリング法 [2]，パス追跡法 [3]，プレディクタ・コレクタ法 [4]，ポテンシャル減少法 [5] など，さまざまなタイプの内点法が提案されてきた．これらの内点法は導出過程や多項式時間での収束性の証明においては大きな違いがあるものの，アルゴリズムとしては共通しているところも多い．本稿では，理解しやすく，多くのソフトウェアで実装されているパス追跡法に絞って説明を行う．

3.1 中心パス

パス追跡法では，最適性条件 (3) を満たす $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ を直接求める．つまり，目的関数の最小化・最大化をするのではなく，方程式を解くことになる．このとき，少しずつ方程式の解に近づいていく反復法を考えるが，各反復点は実行可能内点解もしくは（非実行可能）内点解に限定する．これが「内点法」という名前の由来となっている．その際，反復点が内点解の集合 $\{(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \mid \boldsymbol{x} > \boldsymbol{0},\ \boldsymbol{z} > \boldsymbol{0}\}$ の境界に近づくと，そこで停滞してしまう可能性がある．よって，できるだけ境界から遠い場所を通ることが，最適解への収束のスピードを上げるために重要である．そのため，解析的中心という概念を導入する．解析的中心はパラメタ $\mu > 0$ に対し，次の方程式の解として定義される．

$$\begin{cases} A\boldsymbol{x} = \boldsymbol{b}, \\ A^\top \boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}, \\ X\boldsymbol{z} = \mu \boldsymbol{e}, \\ \boldsymbol{x} > \boldsymbol{0},\ \boldsymbol{z} > \boldsymbol{0}. \end{cases}$$

$\boldsymbol{e} \in \mathbb{R}^n$ はすべての要素が 1 のベクトルである．3 番目の等式は，$x_i z_i = \mu\ (i = 1, 2, \ldots, n)$ とも書けるので，これによって x_i や z_i が 0 に近づくのを防いでいることが見て取れる．$\mu > 0$ が一つ定まれば，この方程式を満たす解は一意に定まることがわかっており，その解を $(\boldsymbol{x}(\mu), \boldsymbol{y}(\mu), \boldsymbol{z}(\mu))$ と表記する．解析的中心は，μ をパラメタとしてなめらかな曲線を描き，$\mu \to 0$ において最適解の一つに収束することが知られている．この曲線を中心パスと呼ぶ．中心パスは次のように定義できる．

$$\mathcal{P} = \left\{ (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \;\middle|\; \begin{array}{l} A\boldsymbol{x} = \boldsymbol{b}, \\ A^\top \boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}, \\ {}^{\exists}\mu > 0,\ X\boldsymbol{z} = \mu \boldsymbol{e}, \\ \boldsymbol{x} > \boldsymbol{0},\ \boldsymbol{z} > \boldsymbol{0}. \end{array} \right\}.$$

中心パスは方程式の解の形でしか与えられていないため，具体的な数値がわかっているわけではないことに注意する．パス追跡タイプの内点法では，この中心パス $\mathcal{P}$ に沿って反復点が進み最適解に近づいていく．

3.2 探索方向

現在の反復点を $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ とする．このとき，μ を $\|X\boldsymbol{z} - \mu \boldsymbol{e}\|$ が最小となるようにとることによって，現在の反復点に（その意味で）一番近い解析的中心 $(\boldsymbol{x}(\mu), \boldsymbol{y}(\mu), \boldsymbol{z}(\mu))$ を求めることができる．なお，$\|\cdot\|$ はユークリッドノルムとする．すると，簡単な計算によって $\mu = \boldsymbol{x}^\top \boldsymbol{z}/n$ だとわかる．$\mu \to 0$ のとき解析的中心は最適解に収束する．そこで，減少率 $\sigma \in [0, 1)$ に対し，解析的中心 $(\boldsymbol{x}(\sigma\mu), \boldsymbol{y}(\sigma\mu), \boldsymbol{z}(\sigma\mu))$ をターゲットとして，その方向に向かうことを試みる．$(\boldsymbol{x}(\sigma\mu), \boldsymbol{y}(\sigma\mu), \boldsymbol{z}(\sigma\mu))$ は次の方程式の解である．

$$\begin{cases} A\boldsymbol{x} = \boldsymbol{b}, \\ A^\top \boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}, \\ X\boldsymbol{z} = \sigma\mu \boldsymbol{e}, \\ \boldsymbol{x} > \boldsymbol{0},\ \boldsymbol{z} > \boldsymbol{0}. \end{cases}$$

ひとまず不等式条件を無視することにすると，これは，関数 $F : \mathbb{R}^{n+m+n} \to \mathbb{R}^{m+n+n}$ を用いた方程式

$$F(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})=\begin{pmatrix} A\boldsymbol{x}-\boldsymbol{b} \\ A^\top\boldsymbol{y}+\boldsymbol{z}-\boldsymbol{c} \\ X\boldsymbol{z}-\sigma\mu\boldsymbol{e}\end{pmatrix}=\begin{pmatrix}\boldsymbol{0}\\ \boldsymbol{0}\\ \boldsymbol{0}\end{pmatrix}$$

の解となる．探索方向 $(d\boldsymbol{x},d\boldsymbol{y},d\boldsymbol{z})$ に対し，次の反復点を $(\boldsymbol{x}+d\boldsymbol{x},\boldsymbol{y}+d\boldsymbol{y},\boldsymbol{z}+d\boldsymbol{z})$ とすると，それはターゲットと一致していてほしい．つまり，

$$F(\boldsymbol{x}+d\boldsymbol{x},\boldsymbol{y}+d\boldsymbol{y},\boldsymbol{z}+d\boldsymbol{z})=\begin{pmatrix}\boldsymbol{0}\\ \boldsymbol{0}\\ \boldsymbol{0}\end{pmatrix} \tag{4}$$

を満たすことが望ましい．しかし F の非線形性から，これを満たす探索方向 $(d\boldsymbol{x},d\boldsymbol{y},d\boldsymbol{z})$ を計算することは難しい．よって，式 (4) の左辺を現在の反復点 $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})$ を中心に 1 次近似して，

$$F(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})+\nabla F(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})\begin{pmatrix}d\boldsymbol{x}\\ d\boldsymbol{y}\\ d\boldsymbol{z}\end{pmatrix}=\begin{pmatrix}\boldsymbol{0}\\ \boldsymbol{0}\\ \boldsymbol{0}\end{pmatrix}$$

を満たす探索方向 $(d\boldsymbol{x},d\boldsymbol{y},d\boldsymbol{z})$ を用いることにする．なお，∇ は微分を意味し，∇F はヤコビ行列である．これは，$(d\boldsymbol{x},d\boldsymbol{y},d\boldsymbol{z})$ に関する次の線形方程式の解となる．

$$\begin{pmatrix}A & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & A^\top & I\\ Z & \boldsymbol{0} & X\end{pmatrix}\begin{pmatrix}d\boldsymbol{x}\\ d\boldsymbol{y}\\ d\boldsymbol{z}\end{pmatrix}=\begin{pmatrix}\boldsymbol{b}-A\boldsymbol{x}\\ \boldsymbol{c}-A^\top\boldsymbol{y}-\boldsymbol{z}\\ \sigma\mu\boldsymbol{e}-X\boldsymbol{z}\end{pmatrix}. \tag{5}$$

ここで，I は単位行列，Z は $\boldsymbol{z}$ の各要素を対角部分に並べた行列である．2 次（以上の）項を無視しているため，このようにして求めた $(\boldsymbol{x}+d\boldsymbol{x},\boldsymbol{y}+d\boldsymbol{y},\boldsymbol{z}+d\boldsymbol{z})$ はターゲットと一致していないが，ターゲットの近くにあることが期待できる．なお，この計算は方程式を 1 次近似することで反復を繰り返す Newton 法の 1 反復に相当するため，この探索方向を Newton 方向とも呼ぶ．

線形方程式はガウスの掃き出し法などにより，行列のサイズ（この場合は $2n+m$）の 3 乗に比例した計算量で解くことができる [6]．しかし，大規模な問題を内点法で解く際，この計算がボトルネックとなる．だが，線形方程式 (5) の係数行列は，9 個の部分行列のうち 4 個がゼロ行列であり，構造的に疎性がある．これを利用することにより，サイズの小さな線形方程式に折りたたむことができる．線形方程式 (5) の右辺は，上から主問題の線形制約の残差，双対問題の線形制約の残差，相補性条件に関連する式の残差を表す定数であり，それらを $\boldsymbol{r}_p,\boldsymbol{r}_d,\boldsymbol{r}_c$ と表記する．そのとき，線形方程式 (5) は，次のように書き換えることができる．

$$A d\boldsymbol{x}=\boldsymbol{r}_p, \tag{6}$$
$$A^\top d\boldsymbol{y}+d\boldsymbol{z}=\boldsymbol{r}_d, \tag{7}$$
$$Z d\boldsymbol{x}+X d\boldsymbol{z}=\boldsymbol{r}_c. \tag{8}$$

このとき，式 (7) から $d\boldsymbol{z}$ について解くことができ，

$$d\boldsymbol{z}=\boldsymbol{r}_d-A^\top d\boldsymbol{y}$$

だとわかる．これを式 (8) に代入すると，$d\boldsymbol{x}$ について解くことができ

$$d\boldsymbol{x}=Z^{-1}(\boldsymbol{r}_c-X(\boldsymbol{r}_d-A^\top d\boldsymbol{y}))$$

が得られる．なお，Z は対角行列なので，Z^{-1} はその対角成分の逆数を取ったものである．これを式 (6) に代入すると，

$$AZ^{-1}XA^\top d\boldsymbol{y}=\boldsymbol{r}_p-Z^{-1}(\boldsymbol{r}_c-X\boldsymbol{r}_d)$$

という $d\boldsymbol{y}$ に関する線形方程式が導ける．この線形方程式のサイズは m となり，元の方程式 (5) のサイズ $2n+m$ より小さい．$d\boldsymbol{y}$ が計算できたら，次の順で $d\boldsymbol{z}$, $d\boldsymbol{x}$ を計算することができる．

$$d\boldsymbol{z}=\boldsymbol{r}_d-A^\top d\boldsymbol{y},$$
$$d\boldsymbol{x}=Z^{-1}(\boldsymbol{r}_c-X d\boldsymbol{z}).$$

新しい計算法におけるボトルネックは，係数行列 $AZ^{-1}XA^\top$ の計算になるが，X や Z^{-1} が対角行列であることを考慮すると，$\mathrm{O}(nm^2)$ の計算量で十分である．

3.3 ステップサイズ

次の反復点 $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})+(d\boldsymbol{x},d\boldsymbol{y},d\boldsymbol{z})$ は内点解でなくなる可能性がある．このため，ステップサイズ α を導入し，次の反復点として $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})+\alpha(d\boldsymbol{x},d\boldsymbol{y},d\boldsymbol{z})$ を用いることにする．$\boldsymbol{x}+\alpha d\boldsymbol{x}>\boldsymbol{0}$ を満たすためのステップサイズ $\alpha\geq 0$ の必要十分条件は，$\alpha_p=\min\limits_{dx_i<0}\left\{-\dfrac{x_i}{dx_i}\right\}$ としたとき，$\alpha\in[0,\alpha_p)$ である．同様に $\boldsymbol{z}+\alpha d\boldsymbol{z}>\boldsymbol{0}$ を満たすためのステップサイズ $\alpha\geq 0$ の必要十分条件は，$\alpha_d=\min\limits_{dz_i<0}\left\{-\dfrac{z_i}{dz_i}\right\}$ としたとき，$\alpha\in[0,\alpha_d)$ である．ステップサイズ α は 1 を超えないほうが安全であることも考慮して，定数 $\tau\in(0,1)$ を用いて次のようにステップサイズを決める．

$$\alpha=\min\{\tau\alpha_p,\tau\alpha_d,1\}$$

3.4 アルゴリズム

以上をまとめると，一番シンプルなパス追跡タイプの主双対内点法は次のアルゴリズムとなる．なお，アルゴリズム中の $:=$ は右辺の値を左辺に代入することを意味する．

シンプルな主双対パス追跡法

Step 0： 初期点の設定

$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ に内点解を設定

$\mu := \boldsymbol{x}^\top \boldsymbol{z}/n$

Step 1： 残差 $(\boldsymbol{r}_p, \boldsymbol{r}_d, \boldsymbol{r}_c)$ を計算

$\boldsymbol{r}_p := \boldsymbol{b} - A\boldsymbol{x}$

$\boldsymbol{r}_d := \boldsymbol{c} - A^\top \boldsymbol{y} - \boldsymbol{z}$

$\boldsymbol{r}_c := \sigma\mu\boldsymbol{e} - X\boldsymbol{z}$

Step 2： 探索方向 $(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z})$ を計算

1. $B := AZ^{-1}XA^\top$,
   $s := \boldsymbol{r}_p - Z^{-1}(\boldsymbol{r}_c - X\boldsymbol{r}_d)$
2. 線形方程式 $Bd\boldsymbol{y} = \boldsymbol{s}$ を解く
3. $d\boldsymbol{z} := \boldsymbol{r}_d - A^\top d\boldsymbol{y}$
4. $d\boldsymbol{x} := Z^{-1}(\boldsymbol{r}_c - Xd\boldsymbol{z})$

Step 3： ステップサイズ α を計算

1. $\alpha_p := \min\limits_{dx_i<0}\left\{-\dfrac{x_i}{dx_i}\right\}$,
   $\alpha_d := \min\limits_{dz_i<0}\left\{-\dfrac{z_i}{dz_i}\right\}$
2. $\alpha := \min\{\tau\alpha_p, \tau\alpha_d, 1\}$

Step 4： 反復点を更新

$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) := (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) + \alpha(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z})$,

$\mu := \boldsymbol{x}^\top \boldsymbol{z}/n$

Step 1 に戻る

このシンプルなアルゴリズムだけでは，理論的な大域的収束性は保証されない．しかしながら，σ として 0.1 や 0.01，τ として 0.95 や 0.99 などを取れば，おおむね最適解に収束する．また反復回数も問題サイズにあまり依存せず，高々十数回であることが多い．ただし，初期点は収束性に影響を与え，少し大きめの内点解を取る必要がある．自明な実行可能内点解がわからない場合，通常 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = (s\boldsymbol{e}, \boldsymbol{0}, s\boldsymbol{e})$ という形式の内点解を使うが，s として 1000 程度は必要なときもある（もちろん問題に依存する）．

このアルゴリズムは，基本的に行列・ベクトル演算のみなので，MATLAB, Octave, Python+NumPy などを使うと，メインルーチンは十数行でプログラムすることができる．興味があれば実装し，反復点の最適解への収束状況を確認してみると内点法の理解が深まる．

4. 多項式時間アルゴリズム

3 節で説明した主双対パス追跡法において，近傍という概念を導入することによって，問題サイズに対し多項式時間アルゴリズムが構築できる．4.1 節では，理論的な解析が比較的容易で多くの論文で研究されている実行可能内点法について紹介し，4.2 節では，解析は困難となるが実用的な手法である非実行可能内点法について紹介をする．

4.1 実行可能内点法

実行可能内点法では，初期点として中心パスに近い実行可能内点解が利用できるという仮定をおく．まず，この仮定のもとでは内点法のすべての反復点が実行可能内点解となることを示す．それには，現在の反復点 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ が実行可能内点解であるとき，次の反復点 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) + \alpha(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z})$ も実行可能内点解となることを示せば十分である．現在の反復点 $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ が実行可能解であるため，$\boldsymbol{r}_p = \boldsymbol{0}$, $\boldsymbol{r}_d = \boldsymbol{0}$ となる．このとき，任意の α に対して

$$\begin{aligned} A(\boldsymbol{x} + \alpha d\boldsymbol{x}) &= A\boldsymbol{x} + \alpha A d\boldsymbol{x} \\ &= \boldsymbol{b} + \alpha\boldsymbol{r}_p = \boldsymbol{b}, \end{aligned}$$

$$\begin{aligned} &A^\top(\boldsymbol{y} + \alpha d\boldsymbol{y}) + (\boldsymbol{z} + \alpha d\boldsymbol{z}) \\ &\quad = (A^\top \boldsymbol{y} + \boldsymbol{z}) + \alpha(Ad\boldsymbol{y} + d\boldsymbol{z}) \\ &\quad = \boldsymbol{c} + \alpha\boldsymbol{r}_d = \boldsymbol{c} \end{aligned}$$

が成り立つため，次の反復点も実行可能解であることがわかる．また，次の反復点が内点解となるようにステップサイズ α を決める．よって，次の反復点は実行可能内点解であるといえる．

$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ が実行可能解のとき，主問題と双対問題の目的関数値の差は

$$\boldsymbol{c}^\top \boldsymbol{x} - \boldsymbol{b}^\top \boldsymbol{y} = \boldsymbol{z}^\top \boldsymbol{x} = n\mu$$

である．最適値は $\boldsymbol{c}^\top \boldsymbol{x}$ と $\boldsymbol{b}^\top \boldsymbol{y}$ の間にあるため，$n\mu$ は最適値との差の上界ともなっている．そして，3 節で説明したアルゴリズムでは，次の反復点に進むことによって μ は $1 - \alpha(1 - \sigma)$ 倍されることが証明できる．各反復での $1 - \alpha(1 - \sigma)$ が定数 $c(< 1)$ 以下であったとしよう．このとき，μ は k 反復によって c^k 倍以下となり，指数的に減少する．そして，最適値との差も指数的に 0 に近づく．この事実がパス追跡法が多項式

時間アルゴリズムとなることに本質的な働きをする．

$1-\alpha(1-\sigma)$ は，減少率 σ とステップサイズ α によって定まる値である．減少率 $\sigma \in [0,1)$ を固定した場合，各反復で $1-\alpha(1-\sigma) \leq c < 1$ となるためには，ステップサイズ α が 0 に近づかないことを担保する必要がある．そのためには，各反復点が中心パスの近くに位置していればよい．そこで，まず中心パスに対し近傍 $\mathcal{N}$ を導入する．多項式時間アルゴリズムとなるパス追跡法では，この近傍 $\mathcal{N}$ から出ないように反復を繰り返す．そのため，3 節で説明した主双対パス追跡法 Step 3 のステップサイズの計算法を修正して，

$$\alpha = \max\{\bar{\alpha} \in [0,1] \mid (\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) + \bar{\alpha}(d\boldsymbol{x}, d\boldsymbol{y}, d\boldsymbol{z}) \in \mathcal{N}\}$$

とする．

よく使われる近傍として，$\beta \in (0,1)$ をパラメタとした次の二つの近傍 $\mathcal{N}_2(\beta)$, $\mathcal{N}_{-\infty}(\beta)$ がある．近傍として $\mathcal{N}_2(\beta)$ を使うときショートステップ，$\mathcal{N}_{-\infty}(\beta)$ を使うときロングステップと呼ぶ．

・ショートステップ：

$$\mathcal{N}_2(\beta) = \left\{ (\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \;\middle|\; \begin{array}{l} A\boldsymbol{x} = \boldsymbol{b}, \\ A^\top \boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}, \\ \|X\boldsymbol{z} - \mu\boldsymbol{e}\| \leq \beta\mu, \\ \mu = \boldsymbol{x}^\top \boldsymbol{z}/n, \\ \boldsymbol{x} > \boldsymbol{0},\ \boldsymbol{z} > \boldsymbol{0}. \end{array} \right\}$$

・ロングステップ：

$$\mathcal{N}_{-\infty}(\beta) = \left\{ (\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \;\middle|\; \begin{array}{l} A\boldsymbol{x} = \boldsymbol{b}, \\ A^\top \boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}, \\ X\boldsymbol{z} \geq (1-\beta)\mu\boldsymbol{e}, \\ \mu = \boldsymbol{x}^\top \boldsymbol{z}/n, \\ \boldsymbol{x} > \boldsymbol{0},\ \boldsymbol{z} > \boldsymbol{0}. \end{array} \right\}$$

どちらの近傍においても $\beta = 0$ のとき中心パス自身となり，β が大きくなるにつれて近傍は広くなる．そして，$\mathcal{N}_{-\infty}(1)$ は実行可能内点解全体となる．$\mathcal{N}_2(\beta) \subset \mathcal{N}_{-\infty}(\beta)$ であるため，次の反復点に移るとき後者のほうが大きく移動できることが多い．特に，n が大きな場合に差が顕著になる．それが，近傍として $\mathcal{N}_2(\beta)$ を使うときショートステップ，$\mathcal{N}_{-\infty}(\beta)$ を使うときロングステップと呼ぶ理由である．

適切にパラメタ β,σ を選ぶことによって，ステップサイズ α がある程度大きく取れ，パス追跡法は多項式時間アルゴリズムとなる．ショートステップの場合，たとえば $\beta = 0.4$, $\sigma = 1 - 1/\sqrt{n}$ と取ると，ステップサイズ α は 1 に固定しても近傍 $\mathcal{N}_2(\beta)$ 内を探索することができる．このとき，$c = 1 - 1/\sqrt{n}$ であり，線形計画問題の入力ビット数を L とすれば，理論的な反復回数の上界は $\mathrm{O}(\sqrt{n}L)$ となる．また，ロングステップでは，たとえば $\beta = 0.5$, $\sigma = 0.5$ と取ることにより，ステップサイズ α として $2/n$ 以上を取って近傍 $\mathcal{N}_{-\infty}(\beta)$ 内を探索することが保証できる．このとき，$c = 1 - 1/n$ とおけ，理論的な反復回数の上界は $\mathrm{O}(nL)$ となる．なお，一般に $c = 1 - 1/f(n)$ のとき，理論的な反復回数の上界は $\mathrm{O}(f(n)L)$ となる．これらのオーダーの証明については，Wright [7] がわかりやすい．理論的な解析では，ロングステップよりショートステップのほうが優れている．ただし，これは最悪の状況に基づいた計算量である．実際に計算してみると，ロングステップのほうが反復点が大きく進み，その結果最適解に速く収束することが大半である．

現実には中心パスに近い実行可能内点解がわからないことが多い．そのとき，Big M を用いることにより，自明な解析的中心をもつ人工問題を作ることができ，それを実行可能内点法で解くことにより，元の問題の最適解を求めることが可能となる．しかし，理論上保証される Big M の値は非常に大きくなり，多項式時間での収束性は保たれるが，実用的な解法ではなくなる．このため，実用上は次節で説明する非実行可能内点法を採用する．

4.2 非実行可能内点法

非実行可能内点法は，実行可能とは限らない内点解からスタートする内点法である．多くの場合，初期点 $(\boldsymbol{x}_0,\boldsymbol{y}_0,\boldsymbol{z}_0)$ として，ある正の数 s に対し $(s\boldsymbol{e},\boldsymbol{0},s\boldsymbol{e})$ を取る．このとき，よく使われる中心パスの近傍として，パラメタ $\beta \in (0,1)$ と $\eta \in [1,\infty)$ を用いた次の近傍 $\mathcal{N}_2(\beta,\eta)$, $\mathcal{N}_{-\infty}(\beta,\eta)$ がある．

・ショートステップ：$\mathcal{N}_2(\beta,\eta) =$

$$\left\{ (\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \;\middle|\; \begin{array}{l} \|A\boldsymbol{x} - \boldsymbol{b}\| \leq \eta\frac{\mu}{\mu_0}\|A\boldsymbol{x}_0 - \boldsymbol{b}\|, \\ \|A^\top \boldsymbol{y} + \boldsymbol{z} - \boldsymbol{c}\| \\ \qquad \leq \eta\frac{\mu}{\mu_0}\|A^\top \boldsymbol{y}_0 + \boldsymbol{z}_0 - \boldsymbol{c}\|, \\ \|X\boldsymbol{z} - \mu\boldsymbol{e}\| \leq \beta\mu, \\ \mu = \boldsymbol{x}^\top \boldsymbol{z}/n,\ \mu_0 = \boldsymbol{x}_0^\top \boldsymbol{z}_0/n, \\ \boldsymbol{x} > \boldsymbol{0},\ \boldsymbol{z} > \boldsymbol{0}. \end{array} \right\}$$

表 1　パス追跡法の反復回数のオーダー

	ショートステップ	ロングステップ
実行可能内点法	$\mathrm{O}(\sqrt{n}L)$	$\mathrm{O}(nL)$
非実行可能内点法	$\mathrm{O}(nL)$	$\mathrm{O}(n^2L)$

・ロングステップ：$\mathcal{N}_{-\infty}(\beta,\eta) =$

$$\left\{ (\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \;\middle|\; \begin{array}{l} \|A\boldsymbol{x}-\boldsymbol{b}\| \le \eta\frac{\mu}{\mu_0}\|A\boldsymbol{x}_0-\boldsymbol{b}\|, \\ \|A^\top\boldsymbol{y}+\boldsymbol{z}-\boldsymbol{c}\| \\ \qquad \le \eta\frac{\mu}{\mu_0}\|A^\top\boldsymbol{y}_0+\boldsymbol{z}_0-\boldsymbol{c}\|, \\ X\boldsymbol{z} \ge (1-\beta)\mu\boldsymbol{e}, \\ \mu = \boldsymbol{x}^\top\boldsymbol{z}/n,\ \mu_0 = \boldsymbol{x}_0^\top\boldsymbol{z}_0/n, \\ \boldsymbol{x}>\boldsymbol{0},\ \boldsymbol{z}>\boldsymbol{0}. \end{array} \right\}$$

これらは，前節で定義した近傍に対して，主問題と双対問題の線形制約条件を緩めたものとなっている．初期点 $(\boldsymbol{x}_0,\boldsymbol{y}_0,\boldsymbol{z}_0)$ は $\mu=\mu_0$ となるので，必ず近傍に含まれる．また，最適解に近づくにつれ $\mu\to 0$ となることから，線形制約条件は満たされるようになる．実行可能内点法のときと同様に，$\mathcal{N}_2(\beta,\eta)\subset\mathcal{N}_{-\infty}(\beta,\eta)$ が成り立つ．後者のほうが反復点は大きく進めることが多く，近傍として $\mathcal{N}_2(\beta,\eta)$ を使うときショートステップ，$\mathcal{N}_{-\infty}(\beta,\eta)$ を使うときロングステップと呼ぶ．非実行可能内点法は，これらの近傍内を通って反復を繰り返す方法である．ただし，多項式時間での収束性を証明するためには，実行可能内点法と同様のステップサイズのコントロールだけでは不十分で，さらに巧みにステップサイズ α や減少率 σ をコントロールする必要がある．証明もかなり複雑になるが，結果としてショートステップでは理論的な反復回数の上界は $\mathrm{O}(nL)$ となり [8]，ロングステップでは $\mathrm{O}(n^2L)$ となる [9]．ただし実際の計算では，多くの場合でロングステップのほうが最適解に速く収束する．まとめると，パス追跡法の計算量のオーダーは表 1 のようになる．非実行可能内点法の計算オーダーは実行可能内点法と比べ悪化していることがわかる．

5.　実際の計算

主双対内点法で高速に線形計画問題の最適解を求めるためには，理論的な計算量の分析とは別のさまざまな実用的な工夫が必要である．本節ではそのいくつかを紹介する．

・Mehrotra のプレディクタ・コレクタ法

この方法は先に仮の探索方向を計算しておき，そこから得られる情報を使って，よりよい探索方向を計算する方法である [10]．実際の計算において，適用しない場合に比べほぼ確実に収束を速める効果があるため，多くのソフトウェアで実装されている．

・疎性

主双対内点法の 1 反復の計算で一番時間がかかるのは，探索方向を求める際の係数行列 $B=AZ^{-1}XA^\top$ の計算と線形方程式を解く部分である．現実の大規模な線形計画問題を解く場合，係数行列 A は疎行列である場合が大半である．このとき，その疎性を活かして係数行列 $B=AZ^{-1}XA^\top$ を計算することができる．さらに係数行列 B も疎行列となることが多いため，その疎性を活かして高速に線形方程式が解ける [7]．理想的な状況では，1 反復当たりの計算量は $\mathrm{O}(nm^2)$ から $\mathrm{O}(n)$ 程度まで減らせる．疎性を利用した計算をすることにより，非常に大規模で疎な線形計画問題において，1 反復当たりの計算時間を 1/1000 に減らすといった劇的な効果がもたらされることもある．

・数値誤差

コンピュータでの計算は数値誤差を伴うため，数学的には同値な計算であっても，計算式が変われば計算結果が変わる可能性がある．このとき，誤差が蓄積していくタイプの計算ではなく，少し計算量が増えても誤差が蓄積しづらい計算式で計算したほうが，内点法の収束が安定するということが起こる．理論的な解析は難しいが，最適解付近で内点法の挙動が不安定なとき数値誤差には注意する必要がある．

内点法を実装したソフトウェアでは，ロングステップ，もしくは多項式時間での収束性は気にせず 3 節で説明したステップサイズのコントロール法を使っている．そのとき，問題の規模にあまり影響されず，高々十数回の反復で実用的な解が得られることが多い．この実際の反復回数の少なさが，主双対内点法の性能の高さに直結している．1979 年に Khachian [11] によって提案された楕円体法は多項式時間アルゴリズムであったものの，反復回数が非常に多くなり実用的でなかったことと対照的である．本節で述べたような工夫を加えた内点法の実装は線形計画問題を高速で解き，特に大規模な問題で単体法を凌ぐと言われている．ただし，分枝限定法や分枝切除法の子問題として線形計画問題を解く場合，ホットスタートのできる単体法のほうが向いている．よって，単体法と内点法は適材適所で使い分けることも重要である．

6.　おわりに

本稿では主双対内点法の中でもパス追跡法について説明を行った．ほかの主双対内点法であるアフィンス

ケーリング法，プレディクタ・コレクタ法，ポテンシャル減少法は，3節で示したアルゴリズムにおいて，減少率 σ とステップサイズ α のコントロール法を変えたものと解釈することができる．それらを含む多種多様な内点法の詳細について学びたい場合，内点法研究の第一人者である水野眞治氏（本特集の筆者の一人でもある）のウェブサイトに充実した解説（合計201ページ！のpdf）があるのでぜひご覧いただきたい（http://www.me.titech.ac.jp/~mizu_lab/text.html）．1冊の書籍に匹敵する情報をインターネット上で無料で公開していることに感謝するばかりである．

当初，線形計画問題の解法として提案された主双対内点法は，その後の研究で2次計画問題，半正定値計画問題，2次錐計画問題，対称錐計画問題へと拡張され，問題サイズの多項式時間で解ける問題クラスが広がっている．それらについて興味のある方は，小島ら [12] が参考になる．線形計画問題に対する内点法を含め，内点法に関する多岐にわたる話題が説明されている．

謝辞　本稿に対し有益な情報と適切なコメントをいただいた東京工業大学の水野眞治氏，福田光浩氏，山下真氏，土橋諒太氏に深く感謝いたします．

参考文献

[1] 森雅夫，松井知己，『オペレーションズ・リサーチ』，朝倉書店，2004.
[2] R. D. C. Monteiro, I. Adler and M. G. C. Resende, "A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series extension," *Mathematics of Operations Research*, **15**, pp. 191–214, 1990.
[3] M. Kojima, S. Mizuno and A. Yoshise, "A primal-dual interior-point algorithm for linear programming," *Progress in Mathematical Programming, Interior Point and Related Methods*, N. Meggiddo (ed.), Springer, pp. 29–47, 1989.
[4] S. Mizuno, M. J. Todd and Y. Ye, "On adaptive-step primal-dual interior-point algorithms for linear programming," *Mathematics of Operations Research*, **18**, pp. 964–981, 1993.
[5] M. Kojima, S. Mizuno and A. Yoshise, "An O($\sqrt{n}L$) iteration potential reduction algorithm for linear complementarity problems," *Mathematical Programming*, **50**, pp. 331–342, 1991.
[6] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins, 1983.
[7] S. J. Wright, *Primal-Dual Interior-Point Methods*, Society for Industrial and Applied Mathematics, 1997.
[8] S. Mizuno, "Polynomiality of infeasible-interior-point algorithms for linear programming," *Mathematical Programming*, **67**, pp. 109–119, 1994.
[9] Y. Zhang, "On the Convergence of a class of infeasible interior-point methods for the horizontal linear complementarity problem," *SIAM Journal on Optimization*, **4**, pp. 208–227, 1994.
[10] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM Journal on Optimization*, **2**, pp.575–601, 1992.
[11] L. G. Khachian, "A polynomial algorithm in linear programming," *Combinatorica*, **4**, pp. 1093–1096, 1984.
[12] 小島政和，土谷隆，水野眞治，矢部博，『内点法』，朝倉書店，2001.

