解　説

線形計画法の新解法について†

今　井　　浩††

1. まえがき

線形計画法 (Linear Programming) は，通信ネットワークの効率改善や飛行機の運行計画立案など，各種システムの設計・運用などに役立てられてきた最適化手法である．この線形計画法に関して，従来用いられてきた単体法というアルゴリズムより数十倍速いと主張されるアルゴリズムが，1984年に AT & T Bell 研究所の若手研究者の N. Karmarkar[7] によって提案され，アルゴリズムの研究の分野にとどまらない幅広い分野にわたって注目を浴びている．

たとえば，昨年9月の Business Week 誌[3]は，これにからんだ騒動を記すとともに，AT & T はすでにある程度のソフトウェアを開発し，それを 800,000 変数よりなる自社の長距離電話ネットワークシステムの運用問題に適用し，その巨大システムの効率を1割程度向上させることに成功したと述べている．1割というと相対的には大したことがないと思われるかもしれないが，システムが巨額な費用を投入してできているものだけに，その絶対的な効果は非常に大きいのである．

Karmarkar の論文[7]は，線形計画問題に対する新しい多項式時間のアルゴリズムを与えるという形で述べられており，実用性に関する Karmarkar の主張が正しいとすれば，これは計算の複雑さに関する基礎的な研究の非常に有益な成果といえる．ただし，実用性に関する主張については，関連分野の研究者の間でも完全には認められているわけではなく，疑問をはさむ人もいる．これは，アルゴリズムのインプリメンテーションに関する詳細が，広く公表されているわけではないことにもよる．

本稿では，線形計画問題に対する Karmarkar の新解法について，これまでの線形計画問題の研究の流れを計算の複雑さの理論とからめて述べたのち，Karmarkar 法の1変形のアルゴリズムを簡単に説明し，そのアルゴリズムについて発表されている実際の計算機実験結果について概説し，最近の線形計画法に関する動向を紹介する．

2. 線形計画問題に対する各種アルゴリズムの概観

線形計画問題に定式化される具体的な問題の例として，毎度おなじみの献立問題を取り上げよう．健康維持のため必要な m 種類の栄養素 i ($i=1,\cdots,m$) に着目し，それらの栄養素を含む n 種類の食品 j ($j=1,\cdots,n$) を買いに行くことを考える．このとき，各栄養素 i を必要摂取分以上とるという条件のもとで，できるだけ安く買物をすませる問題は，食品 j の1単位当たりの栄養素 i の量を a_{ij} とし，栄養素 i の必要量を b_i，食品 j の1単位当たりの費用を c_j とすると，次のような線形計画問題となる．

$$\min \sum_{j=1}^{n} c_j x_j$$

$$\text{s.t.} \sum_{j=1}^{n} a_{ij} x_j \geqq b_i \quad (i=1,\cdots,m)$$

$$x_j \geqq 0 \quad (j=1,\cdots,n)$$

ここで，x_j は食品 j の購買量であり，変数である．$m\times n$ の行列 $A=(a_{ij})$，n 次元ベクトル $\boldsymbol{c}=(c_j)$，$\boldsymbol{x}=(x_j)$，m 次元ベクトル $\boldsymbol{b}=(b_i)$ を用いてこの問題を書き直すと次のように表せる．

$$\min \boldsymbol{c}^T\boldsymbol{x}$$

$$\text{s.t.} \ A\boldsymbol{x}\geqq\boldsymbol{b}$$

$$\boldsymbol{x}\geqq\boldsymbol{0}$$

一般の線形計画問題は，変数に関して線形の不等式・等式の制約条件の下で，線形の目的関数を最小化・最大化する問題である．上の問題では，$m+n$ 個のすべての制約式は不等式であり，$\boldsymbol{c}^T\boldsymbol{x}$ が目的関数である．この献立問題の例だけからは想像しにくいかもしれないが，電話ネットワークや生産管理などの多

† On New Algorithms for Linear Programming by Hiroshi IMAI (Department of Computer Science and Communication Engineering, Kyushu University).

†† 九州大学工学部情報工学科

くのシステムの最適化問題を，直接この線形計画問題として，あるいは線形計画問題を部分問題として繰り返し解く最適化問題として定式化することができ，したがって，線形計画問題を高速に解くアルゴリズムがたいへん有用なわけである．

2.1 単体法

単体法 (simplex method) は，線形計画問題の研究が始まった1940年代の後半に，現在 Stanford 大学教授の G. B. Dantzig により提案されたアルゴリズムで，このアルゴリズムに基づいたソフトウェアプログラムのあげてきた華々しい成果が，現在，線形計画法が広い分野での大規模システム解析に用いられる礎となっている．Dantzig は1982年来日し，情報処理学会の全国大会でも特別講演[4]し，モデルとしての線形計画法について語っている．

単体法は，問題の制約式を満たす変数ベクトルよりなる領域（実行可能領域）を考えると，目的関数を最適化する変数の中で必ず端点となっているものがあることに着目し，その領域の適当な端点から出発して，各時点でそのときいる端点から隣の端点（たとえばそこでの目的関数値のより小さいもの）に移り，これを繰り返して最適な解にたどりつこうというものである．簡単な例を示す．次の線形計画問題

$$\max\ x+y$$

$$\text{s.t. } y\leqq 2,\ x-y\leqq 2,\ x\geqq 0,\ y\geqq 0$$

に対して，実行可能領域を示すと図-1 のようになる．2変数の場合，この領域は凸多角形になり，この例では端点は(0, 0)，(2, 0)，(0, 2)，(4, 2)の4点で，最適解は明らかに (4, 2) である．単体法はたとえば(0, 0)から出発して，(2, 0)，(4, 2)とたどって最適解に着く．

2変数の場合，この問題はたいへん簡単である．しかし，このアプローチで困るのは，制約式数が m，変数の数が n の線形計画問題では，一般に実行可能領域の端点の数が m, n に関して指数オーダになり，標準的な単体法に対しては，すべての端点をたどってからでないと最適解にたどりつかないような意地の悪い例が作れ，したがって最悪の場合の単体法の手間が m, n の指数オーダになってしまうことである．

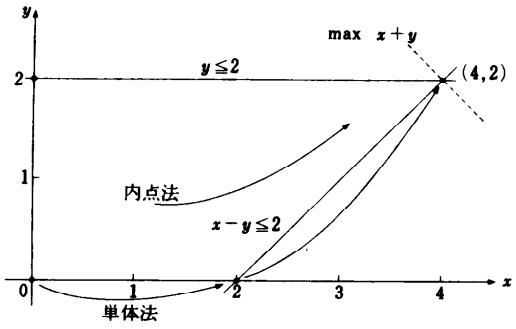

図-1 2次元線形計画問題の例

このようなことから，Dantzig 自身も単体法を提案したものの，当初は単体法がそんなにうまくは線形計画問題を解かないだろうと思っていたとのことである．ところが，実際にプログラムして問題を解いてみると，通常の場合，m, n に関して3乗以下ぐらいの手間で最適解を求めることができることが経験的に観察され，実際的には単体法が結構“良い”アルゴリズムであることが分かった．この点については，理論的に単体法の平均的な場合の振舞いを調べるという研究がなされている．また，プログラミング技法として，制約行列 A のスパースな構造を利用して高速化する手法などが開発され，単体法のソフトウェアとしての完成度はかなりのものになっている．

というわけで，現在ではたいていの大型計算機上に単体法のソフトウェアパッケージがあり，日常的に数百，あるいは数千変数の線形計画問題が解かれている．しかし，1. で述べたような数十万変数の問題に対しては，単体法では実用的時間では太刀打ちできないとされている．

2.2 楕円体法

計算の複雑さの理論では，一般に，入力のサイズの多項式オーダの手間で問題を解くアルゴリズムを“良い”アルゴリズムと考え，そのようなアルゴリズムを多項式時間アルゴリズムと呼び，一方，最悪の場合指数オーダの手間のかかるアルゴリズムを“悪い”アルゴリズムと考える．この考えは1970年代に確立されたが，それまでの単体法の成功は，当時，一方でこの考えの妥当性について見直しを迫っていたと同時に，他方で線形計画問題に対して多項式時間アルゴリズムが存在するかもしれないことを暗示していたのかもしれない．

1979年にはソヴィエトの Khachian[8] が，ついに線形計画問題に対して多項式時間アルゴリズムが存在することを示し，マスメディアを巻き込んで一大センセーションを引き起こした．Khachian は，線形計画問題より一般的な非線形計画問題に対してソヴィエトでそれまでに独自に研究されていた楕円体法を線形計画問題に適用し，それに計算の複雑さの理論をいくつ

かの独創的なアイデアとともに加味することにより，線形計画問題に対する多項式時間アルゴリズムが構成できることを示した．マスメディアの報道の中には，今にも新しい解法が単体法に取って代わるような記事もあったが，実際には発表後1年ほどの間に，実用的には楕円体法は単体法と比べものにならないくらい遅いことが分かり，騒動は落胆とともに一挙に静まった．

Khachian の成果は，残念ながら理論的興味だけのものに終わったが，理論的には，線形計画問題に対する多項式時間アルゴリズムの存在を示したのみならず，それまで難しいとされていたいくつかの組合せ最適化問題に対しても多項式時間アルゴリズムを構成するのに使われたりされており，大きな成果といえる．Khachian のアルゴリズムとその解析については，すでに教科書[6]に優れた解説がされているので，ここではその詳細は省き，独創的なアイデアのうちの一つで，Karmarkar のアルゴリズムの停止規則でも用いられるアイデアについて述べる．

これを一言でいうと，「ある種の厳密解は近似解法を用いても求めることができる」ということで，次の命題がその中心となる．

命題 1. 最適解を有する線形計画問題の入力である行列 A，ベクトル b, c の入力サイズ（A, b, c のすべての要素を2進数でコーディングしたときのコードの長さ）を L とする．このとき，実行可能領域で，最適解の端点を x^*，最適解でない端点を x とすると，$|c^T x^* - c^T x| \geqq 2^{-2L}$ である．□

すなわち，入力のサイズ L によって定まるある距離だけ最適解に近づけば，最適解とそうでない端点を見分けられるというのである．このことより，Khachian のアルゴリズムの手間は L に依存し，制約式の数と変数の数の和を N とすると，L ビットの数の基本演算を $O(N^4L)$ 回行う手間となる．

2.3 Karmarkar 法と内点法

1984年に Karmarkar[7] は，線形計画問題に対する新たな多項式時間アルゴリズムで実用的にも単体法より数段優れていると主張するものを提案した．その手間は，$O(N^{3.5}L)$ 回の L ビットの数の基本演算を行う手間である．

そのアルゴリズムの第1の特徴は，単体法が実行可能領域の境界を端点から端点へとたどっていって最適解を求めるのと違い，実行可能領域の内部を通っていくことである．このような内部を通っていくというアルゴリズム（一般に内点法と呼ばれる）は，非線形計画法の分野で単体法などと同じぐらい古くから提案されていたが，線形計画法に限れば単体法の華々しい成功のためかこの種のアプローチは実際的には忘れ去られていた．第2の特徴は，射影変換を用いてその時点でいる点を中心とする変換を実行可能領域に適用し，その後変換された空間でごく素朴によりよい内点が存在する方向を求めることである．このアルゴリズムの解説も，すでに教科書[6]に記されている．

Karmarkar は，OR 学会や国際数理計画法シンポジウムでたびたび単体法と自分の方法の計算機実験による比較について発表し，新解法が数十倍標準的な単体法のプログラムより優れていると主張した．しかしながら，そのような場での詳細な説明を拒絶し（Karmarkar 自身は，アルゴリズムの発表の後インプリメンテーションについて一度発表したとしているが，その内容はほとんど知られていない），Bell 研も詳細を発表しない方向で対処したため，計算機実験のため Bell 研で開発されたプログラムの性能についてはあまり分かっていない．他の研究者による実験では，単体法と同程度あるいは数倍優れている程度の内点法系統のプログラムは開発されているが，Karmarkar のいう数十倍のスピードアップを実現しているところはないようである．

Karmarkar のアルゴリズム（Karmarkar 法）は，線形計画問題に対する内点法の再考を促し，以後 Karmarkar 法の改良，拡張が数多くなされている．上で述べた他の研究者の実験では，それらのさまざまな拡張アルゴリズムも実験されている．以下では，その中で考え方が単純であり，しかもその計算機実験の結果が公表されているアフィンスケーリング法について述べる．

3. アフィンスケーリング法

このアルゴリズムは，Karmarkar 法において射影変換の代わりにアフィン変換を用いたものといえ，Karmarkar の論文が発表された後，多くの人々によって提案されたものである[2),11)]．Karmarkar 自身も California 大学 Berkeley 校で行った共同研究の中で発表している[1)]．このアルゴリズムは，考え方が簡単であり，性質がよい問題では Karmarkar 法と同じような振舞いを示し，アルゴリズムの各ステップで必ず線形目的関数値が減少するというよい性質がある．反面，元の射影変換を用いた Karmarkar 法と違い，

多項式オーダの手間で線形計画問題を解くかどうかは分かっておらず，実際にアルゴリズムの振舞いが非常に悪い場合もある．以下，このアルゴリズムについて解説する．

3.1 アルゴリズムの概説

以下では次の型の線形計画問題を考える．

$$\min\ \boldsymbol{c}^T\boldsymbol{x}$$
$$\text{s.t. } A\boldsymbol{x}\geqq\boldsymbol{b}$$

A は $m\times n$ の行列で，rank $A=n$ とする．実行可能領域を X で表し：

$$X=\{\boldsymbol{x}\mid A\boldsymbol{x}\geqq\boldsymbol{b}\}$$

X に関して $A\boldsymbol{x}'>\boldsymbol{b}$ であるような真の内点 $\boldsymbol{x}'$ が存在するとする．また，ここでは簡単のためそのような内点 $\boldsymbol{x}'$ が最初に与えられているとする．

$\boldsymbol{x}'$ から領域 X の中を通って，より線形目的関数 $\boldsymbol{c}^T\boldsymbol{x}$ を減らす点に移ることを考えると，まずだれでも最初に思いつくのは，$\boldsymbol{x}'$ から $-\boldsymbol{c}$ の方向に進むことである．こうすると必ず目的関数値は減るものの，それでは $\boldsymbol{x}'$ がどこにいても同じ方向へしか動けず，これだけでは最適解にたどりつけないのは明らかである．

$\boldsymbol{x}'$ からのよい方向を求めるには，やはり実行可能領域 X の，特に $\boldsymbol{x}'$ の近傍での領域の情報をなんらかの形で使うべきである．アフィンスケーリング法は，$\boldsymbol{x}'$ を中心とし，実行可能領域 X に含まれる楕円を以下のように考え，その楕円が $\boldsymbol{x}'$ の近傍の実行可能領域を近似しているものとみなして，楕円上で $\boldsymbol{c}^T\boldsymbol{x}$ を最小にする点の方向に $\boldsymbol{x}'$ から進んで行こうというものである．

行列 A の第 i 行ベクトルを $\boldsymbol{a}_i^T$ とし，$\boldsymbol{b}=(b_i)$ とする．$\boldsymbol{x}'$ に対して，第 i 対角要素が $1/(\boldsymbol{a}_i^T\boldsymbol{x}'-b_i)$ (>0) であるような $m\times m$ 対角行列を D とする：

$$D=\begin{pmatrix}\dfrac{1}{\boldsymbol{a}_1^T\boldsymbol{x}'-b_1} & & 0\\ & \ddots & \\ 0 & & \dfrac{1}{\boldsymbol{a}_m^T\boldsymbol{x}'-b_m}\end{pmatrix}$$

このとき m 次元ベクトル $\boldsymbol{e}=(1,1,\cdots,1)^T$ とすると，

$$D(A\boldsymbol{x}'-\boldsymbol{b})=\boldsymbol{e}$$

である．$\boldsymbol{x}'$ におけるこの D を用いて，次のような楕円 $E(\boldsymbol{x}')$ を考える：

$$E(\boldsymbol{x}')=\{\boldsymbol{x}\mid(\boldsymbol{x}-\boldsymbol{x}')^TA^TD^2A(\boldsymbol{x}-\boldsymbol{x}')\leqq1\}$$

ここで

$$\boldsymbol{y}=D(A\boldsymbol{x}-\boldsymbol{b})$$

とすると，

$$DA(\boldsymbol{x}-\boldsymbol{x}')=D(A\boldsymbol{x}-\boldsymbol{b})-D(A\boldsymbol{x}'-\boldsymbol{b})=\boldsymbol{y}-\boldsymbol{e}$$

となる．$(\boldsymbol{y}-\boldsymbol{e})^T(\boldsymbol{y}-\boldsymbol{e})\leqq1$ は m 次元空間での $\boldsymbol{e}$ を中心とし半径 1 の球であるから，この球では必ず $\boldsymbol{y}\geqq0$ すなわち $A\boldsymbol{x}-\boldsymbol{b}\geqq0$ である（対角行列 D の対角要素はすべて正であるから）．したがって，次の命題を得る．

命題 2. $E(\boldsymbol{x}')$ は実行可能領域 X に含まれる．□

以上の簡単な議論により，実行可能領域の任意の内点 $\boldsymbol{x}'$ が与えられたとき，その $\boldsymbol{x}'$ に関する情報だけで $\boldsymbol{x}'$ を中心とし実行可能領域に含まれる楕円が構成できることが分かった．図-2 に図-1 と同じ 2 次元の線形計画問題での例を示す．

この図からも分かるように，この楕円は $\boldsymbol{x}'$ の周辺での実行可能領域を楕円で近似していると考えられ，実行可能領域の中を通って $\boldsymbol{x}'$ からよりよい内点に行こうとするとき，この楕円の上で $\boldsymbol{c}^T\boldsymbol{x}$ を最小にする方向に進むことが自然と考えられる．もともとの実行可能領域は角張っていたため $\boldsymbol{c}^T\boldsymbol{x}$ が最小になる点を見つけにくかったわけだが，楕円は丸まっているため容易にその中で $\boldsymbol{c}^T\boldsymbol{x}$ が最小になる点を見つけることができる．すなわち，

$$\min\{\boldsymbol{c}^T\boldsymbol{x}\mid\boldsymbol{x}\varepsilon E(\boldsymbol{x}')\}$$

の最適解は

$$\boldsymbol{x}'+t\boldsymbol{d}$$

ただし，$\boldsymbol{d}=-(A^TD^2A)^{-1}\boldsymbol{c}$

$$t=1/\sqrt{\boldsymbol{d}^T(A^TD^2A)\boldsymbol{d}}$$

である．図-2 では，$\boldsymbol{x}'=(1,1)^T$ での楕円についての最適解を示している．このとき，楕円 $E(\boldsymbol{x}')$ が実行可能領域に含まれていることより，$\boldsymbol{x}'+t\boldsymbol{d}$ は必ず実行可能領域内にあり $\boldsymbol{x}'+t\boldsymbol{d}$ での目的関数値は，$\boldsymbol{x}'$ での目的関数値より必ず小さくなっている．

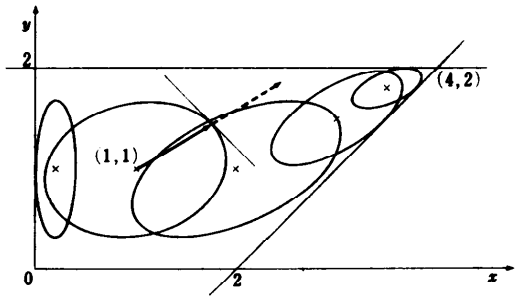

図-2 アフィンスケーリング法で考えられる楕円 (0.2, 1), (1, 1), (2, 1), (3, 1.5), (3.5, 1.8) をおのおの中心とする楕円)

ここで，より全域的な立場から考えると，x' の次の内点として何も楕円上の点 $x'+td$ にこだわることなく，x' から d の方向に進めば目的関数値が減少するのであるから，楕円を越えてさらに進むことが考えられる（図-2 の破線）．その場合，実行可能領域の外に出ては困るので，x' から d の方向に t' だけ進んだとき最初に制約式にぶつかるような t_{max} を求め，あらかじめ決めておいた割合 $\alpha(0<\alpha<1)$ に対して αt_{max} だけ進むということが考えられる．実際に最初にぶつかる制約式を求めるには次のようにすればよい．

$$t_{max}=\min\{(b_i-a_i^Tx')/(a_i^Td)\,|\,a_i^Td<0,\ i=1,\cdots,m\}$$

$a_i^Td<0$ となる i がないときは，その線形計画問題の目的関数はいくらでも小さくできる．

以上の考え方に基づき次のようなアルゴリズムが考えられる．ここで，内点 x' はあらかじめ与えられているものとする．

while 停止条件が満たされていない **do**

$d:=-(A^TD^2A)^{-1}c$;

$t_{max}:=\min\{(b_i-a_i^Tx')/(a_i^Td)\,|\,a_i^Td<0\}$;

$x':=x'+\alpha t_{max}d$;

x' に対する停止条件としては，理論的には命題 1 に基づいたものにすればよく，実際にはより現実的なものを用いればよい．α としては，たとえば最初のうちは 0.99 とし，途中から 0.9 にするなどということが考えられている．初期の内点は，与えられた線形計画問題を少し変形することにより，容易に求めることができる．

3.2 公表されている計算機実験の結果

アフィンスケーリング法については，実際の計算機実験の結果も発表され，標準的な単体法との比較もされている．以下では，その結果を紹介する．

このような実験においては，アルゴリズムのインプリメンテーションがたいへん重要になる．前節で述べたのはアルゴリズムの単なる骨格であり，そのまま素直にプログラム化しただけでは，とても単体法と対等なぐらいの高速性を達成することはできない．実際のインプリメンテーションにおいては，さまざまな高速化技法が必要であり，またこの部分が今後の実際的な研究での中心課題になっていくと思われる．ただし，ここではインプリメンテーションの詳細は省く．

アフィンスケーリング法に関する実験は，二つの論文[1),9)]で発表されている．ともに比較の相手としては，

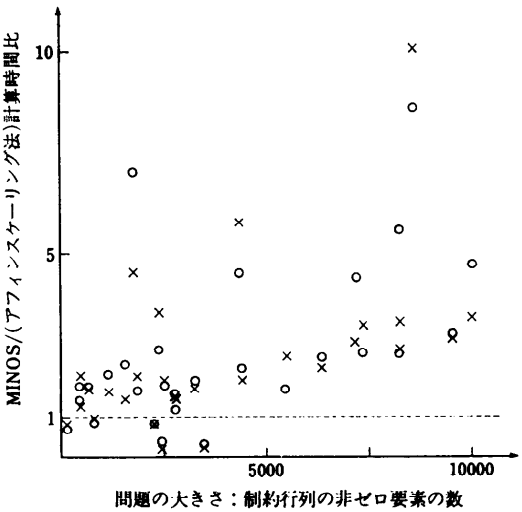

図-3 アフィンスケーリング法と MINOS 単体法との計算時間比[1),9)]
○: Adler et al.[1)], ×: Monma, Morton[9)]

Stanford 大学で開発された非線形計画法プログラムパッケージである MINOS の単体法の FORTRAN プログラムを用いており（版は一つ違う），また実験での線形計画問題のデータは，コンピュータネットワークで配布されている LP/data[5)] の問題群を用いている．そこで，ここでは計算機環境の相違を単に同一問題での MINOS による計算時間のみで換算し，LP/data の問題に対する結果を図-3 に示す．図-3 で問題の大きさとしては，制約行列 A の非ゼロ要素の数を用いた．

図から分かるように，これらの実験では単体法より平均的には数倍速い結果が得られている．ただ，この結果から内点法系統のアルゴリズムが単体法より優れているなどという結論を導き出せるわけではない．以下問題点を二つあげると

（1） 比較する対象として同じプログラミング言語 FORTRAN で書かれ，コードの入手も割合い容易な MINOS の単体法プログラムが用いられている．しかし，MINOS 自体は非線形計画法全体に対するパッケージであり，その中の一部のプログラムを取り上げてあたかもそれが線形計画問題に対する単体法の代表的なプログラムとして扱うのは疑問が残るかもしれない．また，実際の商用の単体法ソフトウェアでは，アセンブラ言語を用いての高速化や他の単体法の数十年の研究での諸成果を組み込んでおり，MINOS の単体法より通常速いと言われている．もちろん，この点に関しては，内点法系統のアルゴリズムでもアセンブラ

言語を用いて高速化することが可能であり，また他の計算技法についても，単体法の場合と違い，これから研究されていくのであり，高速化の余地が残っているといえる．現段階で対等ぐらいであるということは，将来的にかなり期待できそうだともいえる．

（2） 実験データとして用いられている LP/data の諸問題であるが，これらの問題はなんらかの指針をもって集められたというわけでなく，単に現実に線形計画問題を解いている人でそれらの問題を提供してくれる人から集めたものである．中には，内点法にとって非常に解くのが難しい問題も含まれており，比較にはその点よいかもしれない．ただ，実験データの妥当性を議論するのは非常に難しく，またこのようなデータが広く使われるとこれらの問題にのみ有効な手法を用いて高速化を図るのではといった悪い面も指摘されており，今後の解決すべき問題である．

これらの実験から客観的にいえることは，これまで単体法の成功の影に隠れていた線形計画問題に対する内点法系統のアルゴリズムが，十分単体法と対等（以上）の高速性を有することが確認され，今後の研究への期待が高まっているということである．

3.3 アフィン版アルゴリズムの問題点

本稿では，説明が簡単でしかも単体法に匹敵する計算機実験結果が公表されているということで，アフィンスケーリング法について説明してきたが，この方法はある程度 Karmarkar のもともとのアルゴリズムと似た振舞いは示すものの，Karmarkar 法のような多項式時間アルゴリズムであるかどうかは示されておらず，だいたい問題が退化している場合については収束性の厳密な証明もなされていない．

特に，アフィンスケーリング法は，初期解として実行可能領域の境界に近い点から出発した場合，以降ずっと境界に沿って進んで行くことがある程度理論的にも分かっており，内点法でありながら単体法とある意味で似かよった振舞いを示す．このことは，実験的にも確かめられている[10]．Karmarkar 法の場合，いったん境界の近くによっても次の点を領域の中の方にもっていく力が作用し，その点アフィンスケーリング法と本質的に異なっている．

このように，すべての制約式からできるだけ離れながら，実行可能領域の中の最適解へ通じる“中心”の道の近くを通って最適解にたどりつこうという考え方は，最近よく研究されているが，その系統のアルゴリズムの実用性に関する研究は今後の課題である．

以上述べたように，内点法系統のアルゴリズムとしてアフィンスケーリング法を実際に用いる場合，それのみからなるアルゴリズムは極端に悪くなりうることがあるので，他の手法と組み合わせて用いることが必要である．

4. 内点法は単体法を凌駕できるか？

現在までのところ，内点法が単体法よりかなり少ない反復回数で問題を解くということについては，広く認められるようになった．しかし，1反復当たりの計算量は内点法の方が多く，何も工夫しないと単体法より1反復当たり大ざっぱにいって1オーダ高い．

線形計画問題に対する内点法の将来については，楽観論と悲観論両方が見受けられる．楽観論では，反復回数そのものは数千変数の問題でも数十回程度なのであるから，1反復当たりの計算での高速化をこれから集中的に研究していけば，単体法を凌駕する内点法のソフトウェアを開発できるというものである．この場合，並列計算技法の導入も重要であると思われる．ただ，並列計算技法そのものは単体法にも適用可能なわけで，お互いの高速化に役にたつものではある．

悲観論では，内点法はもともと線形の世界の問題であった線形計画問題を非線形の世界に持ち込んで解くため，問題の大規模化にともなって，非線形性による計算効率の低下が起こるのではないかというものである．

これら両論のどちらが正しいにせよ，今後の研究をとおして線形計画問題に対するより高速なアルゴリズムが開発されると思われる．

5. むすび

線形計画法に対する新解法について解説した．線形計画法に関して現在精力的に行われている研究の進展については，本年8月29日より9月2日にわたって東京で開かれる第13回国際数理計画法シンポジウムにおいて，Karmarkar 自らの発表も含めて，活発に発表・議論されるものと期待されている．

謝辞 図の作成などご協力をいただいた本学大学院生有松正友氏に感謝します．

参考文献

1) Adler, I., Karmarkar, N., Resende, M. G. C. and Veiga, G.: An Implementation of Karmarkar's Algorithm for Linear Programming, University of California, Berkeley (1986).

2) Barnes, E. R.: A Variation on Karmarkar's Algorithm for Solving Linear Programming Problems, Math. Prog., Vol. 36, No. 2, pp. 174-182 (1986).
3) Business Week, September 21, pp. 35-36 (1987).
4) Dantzig, G. B.: Mathematical Programming and Decision Making (今野 浩訳), 情報処理, Vol. 24, No. 5, pp. 604-609 (1983).
5) Dongarra, J. J. and Grosse, E.: Distribution of Mathematical Software via Electronic Mail, Comm. ACM, Vol. 30, No. 5, pp. 403-407 (1987).
6) 伊理正夫：線形計画法, 共立出版, 東京 (1986).
7) Karmarkar, N.: A New Polynomial-Time Algorithm for Linear Programming, Combinatorica, Vol. 4, pp. 375-395 (1984).
8) Khachian, L. G.: Polynomial Algorithms in Linear Programming, Zh. Vychisl. Mat. i Mat. Fiz., Vol. 20, pp. 51-68 (1980; in Russian); English translation in U. S. S. R. Comput. Math. and Math. Phys., Vol. 20, pp. 53-72 (1980).
9) Monma, C. L. and Morton, A. J.: Computational Experience with a Dual Affine Variant of Karmarkar's Method for Linear Programming, Bell Communications Research (1987).
10) 笹川 卓：線形計画法における内点法の研究, 東京大学大学院工学系研究科計数工学専門課程修士論文 (1988).
11) Vanderbei, R. J., Meketon, M. J. and Freedman, B. A.: A Modification of Karmarkar's Linear Programming Algorithm, AT & T Bell Laboratories, Holmdel (1985).

(昭和63年2月10日受付)