3　シンプレックス法の収束性と初期化

第 2 章では，シンプレックス法の基本的な仕組みを紹介し，実行可能辞書をもつ基準形 LP に対して

- 現在の実行可能基底解が最適解であると認識される，あるいは
- 入力 LP の非有界性が判定される

まで，その辞書を繰り返し更新する方法であることがわかりました．しかし，これだけで LP が解けたとはいえません．技術的に重要な問題点：

(a)　収束性の問題: シンプレックス法は有限回で終了するか？

(b)　初期化の問題: 任意の基準形 LP に対して，どのように実行可能辞書をえるか？

の 2 つがまだ残されています．

この章では，これらの問題の解決策について説明します．まず (a) に関しては有限回で終了しない LP の例を示し，そのような現象 (巡回) を防ぐ 1 つの方法 — Bland のピボット選択規則を紹介します．次に (b) については，与えられた LP に新たな変数を 1 つつけ加え，自明な実行可能辞書をもつ LP を定義し，これを解くことによってもとの問題の実行可能辞書が (存在すれば) 求められることを示します．そして最後に，以上 2 つの結果から第 1 章の基本定理を導きます．

3.1　ピボット選択規則と巡回

シンプレックス法が有限回で終了しない巡回現象をみる前に，ピボット列の選択規則について説明しておきましょう．

前章の algorithm シンプレックス法，ステップ 2 では，

$c_s > 0$ となる添字 s $(1 \leq s \leq n)$ を 1 つ選ぶ;

とのみあり，$c_s > 0$ の s が複数ある場合のピボット列の選択方法が明確には示されていませんでした．シンプレックス法では，このピボット列の選択に多くの自由度があります．普通，用いられるのは Dantzig によって提案された**最大係数規則** (largest coefficient rule) です．

最大係数規則:

- ピボット列を選ぶとき，$c_s > 0$ を満たす列 s が複数あれば，その中で係数 c_s の値の最も大きいものを選ぶ．

つまり，この規則は，対応する非基底変数 $x_{N(s)}$ の値が 1 単位増加したとき，目的関数 z の値が最も大きくなるようにピボット列を選択します．その意味で，ごく自然な選択規則ということができます．

しかし，最大係数規則でピボット列を選んだ場合，実際に得られる目的関数の増加が最大になるとはかぎりません．次の**最大改善規則** (largest improvement rule) にしたがえば，1 回のピボットで得られる目的関数値の増加を最大にすることができます．

最大改善規則:

- ピボット列を選ぶとき，$c_s > 0$ を満たす列 s が複数あれば

$$v_s = \min\{b_i \,/\, a_{ij} \mid a_{ij} > 0,\ \ 1 \le i \le m\}$$

として $c_s \times v_s$ の最も大きなものを選ぶ．

どちらの選択規則が優れているかの議論は後にまわして，ここでは最大係数規則を使って次の基準形 LP を解いてみましょう:

例 3.1.

$$\left|\begin{array}{lrcrcrcrcl} \text{最大化} & 10x_1 & - & 57x_2 & - & 9x_3 & - & 24x_4 & & \\ \text{条　件} & 0.5x_1 & - & 5.5x_2 & - & 2.5x_3 & + & 9x_4 & \le & 0 \\ & 0.5x_1 & - & 1.5x_2 & - & 0.5x_3 & + & x_4 & \le & 0 \\ & x_1 & & & & & & & \le & 1 \\ & & & & & x_1,\ x_2,\ x_3,\ x_4 & & & \ge & 0. \end{array}\right. \tag{3.1}$$

制約条件の右辺 b_i はすべて非負の定数であり，スラック変数 $x_5,\, x_6,\, x_7\ (\geq 0)$ と目的関数値を表す変数 z を導入すれば，直ちに実行可能な辞書

$$\left|\begin{array}{rcrcrcrcr} x_5 &=& 0 &-& 0.5x_1 &+& 5.5x_2 &+& 2.5x_3 &-& 9x_4 \\ x_6 &=& 0 &-& 0.5x_1 &+& 1.5x_2 &+& 0.5x_2 &-& x_4 \\ x_7 &=& 1 &-& x_1 & & & & & & \\ z &=& 0 &+& 10x_1 &-& 57x_2 &-& 9x_3 &-& 24x_4 \end{array}\right. \tag{3.2}$$

がえられ，シンプレックス法を始めることができます．シンプレックス法のステップ 2 では，z 行の変数 x_1 の係数のみが正であることから第 1 列をピボット列 s として選びます．次のステップ 3 ですが，

$$b_1 \,/\, a_{11} = b_2 \,/\, a_{21} = 0 < b_3 \,/ a_{31} = 1$$

が成り立ち，ピボット行 r としては $b_1 = b_2 = 0$ の第 1 行か第 2 行が選ばれることになります．ここでは，ピボット列選択の最大係数規則に加え，

- ピボット行を選ぶとき，$b_r \,/\, a_{rs} = \min\{b_i \,/\, a_{is} \mid a_{is} > 0,\ \ i = 1,\, \ldots,\, m\}$ を満たす行 r が複数あれば添字 $B(r)$ の最も小さいものを選ぶ

こととし，以後このピボット行選択規則を適用することにします．したがって，第 1 行をピボット行に選び，$(r,\, s) = (1,\, 1)$ を中心ととするピボット演算を行います:

$$\left|\begin{array}{rcrcrcrcr} x_1 &=& 0 &-& 2x_5 &+& 11x_2 &+& 5x_3 &-& 18x_4 \\ x_6 &=& 0 &+& x_5 &-& 4x_2 &-& 2x_3 &+& 8x_4 \\ x_7 &=& 1 &+& 2x_5 &-& 11x_2 &-& 5x_3 &+& 18x_4 \\ z &=& 0 &-& 20x_5 &+& 53x_2 &+& 41x_3 &-& 204x_4 \end{array}\right. \tag{3.3}$$

基底変数の集合は $\{x_5,\, x_6,\, x_7,\, z\}$ から $\{x_1,\, x_6,\, x_7,\, z\}$ に替わりましたが，基底解は

$$(x_1,\, x_2,\, x_3,\, x_4,\, x_5,\, x_6,\, x_7,\, z) = (0,\, 0,\, 0,\, ,0,\, 0,\, 0,\, 1,\, 0)$$

から全く変化していません．

この例のように右辺の定数 b_i がゼロの行で行うピボット演算を**退化している** (degenerate) といいます．簡単に証明できますが，

- ピボット演算が退化している必要十分条件は，基底解が変化しない　(演習問題 3.1)

ことです．また，右辺に値ゼロの定数 b_i が**少なくとも 1 つ**含まれるとき，その辞書は**退化している** (degenerate) といいます．したがって，(3.2), (3.3) はともに退化した辞書です．

さて，もしもシンプレックス法が有限回で終了しないとすれば，それはどのような状況でしょうか．一般に，

- 同じ基底変数の集合をもつ辞書は一意に定まる　(演習問題 3.2)
- シンプレックス法でピボット演算が退化していなければ，目的関数値は必ず増加する

ことを考えあわせると，シンプレックス法が収束しないためには，何回かのピボット演算ののちに再び同じ辞書が現れ，その中で行われたピボット演算はすべて退化していなければなりません．そして，この**巡回** (cycling) とよばれる現象は，最大係数規則に限らず，最大改善規則でも起こりうることが知られています．

例 3.1 の問題 (3.1) は，最大係数規則で巡回が起きるように作られた問題例で，1983 年の Chvátal の教科書に掲載されたものです．実際，辞書 (3.3) にピボット演算を続けると

$$
\left|\begin{array}{lcrrrrr}
x_1 &=& 0 & +0.75x_5 & -2.75x_6 & -0.5x_3 & +4x_4 \\
x_2 &=& 0 & +0.25x_5 & -0.25x_6 & -0.5x_3 & +2x_4 \\
x_7 &=& 1 & -0.75x_5 & +2.75x_6 & +0.5x_3 & -4x_4 \\
z &=& 0 & -6.75x_5 & -13.25x_6 & +14.5x_3 & -98x_4
\end{array}\right|
\rightarrow
\left|\begin{array}{lcrrrrr}
x_3 &=& 0 & +1.5x_5 & -5.5x_6 & -2x_1 & +8x_4 \\
x_2 &=& 0 & -0.5x_5 & +2.5x_6 & +x_1 & -2x_4 \\
x_7 &=& 1 & & & -x_1 & \\
z &=& 0 & +15x_5 & -93x_6 & -29x_1 & +18x_4
\end{array}\right|
$$

$$
\rightarrow
\left|\begin{array}{lcrrrr}
x_3 &=& 0 & -0.5x_5 & +4.5x_6 & +2x_1 & -4x_2 \\
x_4 &=& 0 & -0.25x_5 & +1.25x_6 & +0.5x_1 & -0.5x_2 \\
x_7 &=& 1 & & & -x_1 & \\
z &=& 0 & +10.5x_5 & -70.5x_6 & -20x_1 & -9x_2
\end{array}\right|
$$

となり，次にえられる辞書

$$
\left|\begin{array}{lcrrrrrrrr}
x_5 &=& 0 &-& 2x_3 &+& 9x_6 &+& 4x_1 &-& 8x_2 \\
x_4 &=& 0 &+& 0.5x_3 &-& x_6 &-& 0.5x_1 &+& 1.5x_2 \\
x_7 &=& 1 & & & & & -& x_1 & & \\
z &=& 0 &-& 21x_3 &+& 24x_6 &+& 22x_1 &-& 93x_2
\end{array}\right| \tag{3.4}
$$

で，$(r,\ s) = (2,\ 2)$ を中心とするピボット演算を行えば最初の辞書 (3.2) に戻ります．

3.2 有限収束の保証

第 2 章で記述した algorithm シンプレックス法は，最大係数規則や最大改善規則を用いるかぎり，巡回の起きる可能性があるため，一般には有限収束が保証されません．しかし，別のピボット選択規則を用いれば，常に収束することが知られています．そのようなピボット選択規則はいくつか知られていますが，ここでは最もシンプルでエレガントな**最小添字規則** (smallest subscript rule) を紹介しましょう．

最小添字規則:

- ピボット列を選ぶとき，$c_s > 0$ を満たす列が複数あれば添字 $N(x)$ の最も小さいものを選ぶ．
- ピボット行を選ぶとき，$b_r \,/\, a_{rs} = \min\{b_i \,/\, a_{is} \mid a_{is} > 0,\ \ i = 1,\ \ldots,\ m\}$ を満たす行 r が複数あれば添字 $B(r)$ の最も小さいものを選ぶ

この選択規則に対して次の定理が成り立ちます:

定理 3.1. [Bland, 1977]
最小添字規則を用いれば，シンプレックス法は必ず有限回で終了する．

定理 3.1 の証明は付録に譲ることにして，巡回を起こした問題 (3.1) に最小添字規則を適用してみましょう．この問題例では，辞書 (2.11) までは最大係数規則と同じピボットが選択されますが，辞書 (2.11) では z 行で正の係数をもつ 2 つの非基底変数 $x_6,\, x_1$ のうち，添字の小さい x_1 の列がピボット列 $r = 3$ として選ばれます．ピボット行は，この場合，一意に x_4 の行 $s = 2$ に定まるので，$(r,\ s) = (3,\ 2)$ を中心とするピボット演算を行うと

$$
\left|\begin{array}{rcrcrcrcrcr}
x_5 &=& 0 &+& 2x_3 &+& x_6 &-& 8x_4 &+& 4x_2 \\
x_1 &=& 0 &+& x_3 &-& 2x_6 &-& 2x_4 &+& 3x_2 \\
x_7 &=& 1 &-& x_3 &+& 2x_6 &+& 2x_4 &-& 3x_2 \\
z &=& 0 &+& x_3 &-& 20x_6 &-& 44x_4 &-& 27x_2
\end{array}\right.
$$

次のピボットの中心は $(r,\ s) = (3,\ 1)$ に一意に定まり，

$$\left|\begin{array}{rcrcrcrcr} x_5 &=& 2 &-& 2x_7 &+& 5x_6 &-& 4x_4 &-& 2x_2 \\ x_1 &=& 1 &-& x_7 & & & & & & \\ x_3 &=& 1 &-& x_7 &+& 2x_6 &+& 2x_4 &-& 3x_2 \\ z &=& 1 &-& x_7 &-& 18x_6 &-& 42x_4 &-& 30x_2 \end{array}\right.$$

となって，退化から脱出すると同時に最適性の条件も満たされました．ピボットの中心は，あくまで変数の添字の大小で選択され，行番号・列番号の大小には無関係ですから間違えないように注意してください．

演習問題

3.1 ピボット演算が退化している必要十分条件は，基底解が変化しないことを示せ．

3.2 基底変数，非基底変数の集合が等しい２つの辞書は，同一の辞書であることを証明せよ．

[ヒント: ２つの辞書が等価な線形等式系 (つまり，同じ解集合をもつ) ことを使って両者の各係数が等しいことを示す．]

3.3 次の問題 (Beale, 1955) に，最大係数規則，最大改善規則，最小添字規則のそれぞれを用いたシンプレックス法を適用せよ:

$$\left|\begin{array}{lrcrcrcrcl} \text{最大化} & 3/4x_1 &-& 150x_2 &+& 1/50x_3 &-& 6x_4 & & \\ \text{条　件} & 1/4x_1 &-& 60x_2 &-& 1/25x_3 &+& 9x_4 &\leq& 0 \\ & 1/2x_1 &-& 90x_2 &-& 1/50x_3 &+& 3x_4 &\leq& 0 \\ & & & & & x_3 & & &\leq& 1 \\ & & & & & x_1,\ x_2,\ x_3,\ x_4 & & &\leq& 0. \end{array}\right.$$

3.3 初期化と 2 段階シンプレックス法

すでに述べたように，任意の基準形 LP の実行可能解を求めることもシンプレックス法で実現できます．次の例題の実行可能辞書を求めてみましょう:

例 3.2.

$$
\left|\begin{array}{lrcrcrcl}
\text{最大化} & x_1 & + & x_2 & - & 2x_3 & & \\
\text{条　件} & 2x_1 & - & 2x_2 & + & x_3 & \leq & -4 \\
 & & - & 2x_2 & - & 2x_3 & \leq & -3 \\
 & 2x_1 & - & x_2 & + & x_3 & \leq & -2 \\
 & & & x_1, & x_2, & x_3 & \geq & 0.
\end{array}\right. \tag{3.5}
$$

前処理　問題 (3.5) の辞書は

$$
\left|\begin{array}{lcrcrcrcr}
x_4 & = & -4 & - & 2x_1 & + & 2x_2 & - & x_3 \\
x_5 & = & -3 & & & + & 2x_2 & + & 2x_3 \\
x_6 & = & -2 & - & 2x_1 & + & x_2 & - & x_3 \\
z & = & 0 & + & x_1 & + & x_2 & - & 2x_3
\end{array}\right.
$$

であり，明らかに実行可能ではありません．このような場合には，新たに**人工変数** (artificial variable) x_a を導入し，**補助問題** (auxiliary problem):

$$
\left|\begin{array}{lcrcrcrcrcr}
\text{最大化} & & & & - & x_a & & & & & \\
\text{条　件} & x_4 & = & -4 & + & x_a & - & 2x_1 & + & 2x_2 & - & x_3 \\
 & x_5 & = & -3 & + & x_a & & & + & 2x_2 & + & 2x_3 \\
 & x_6 & = & -2 & + & x_a & - & 2x_1 & + & x_2 & - & x_3 \\
 & & & & & & & x_1, \ldots, x_6 \geq 0, & & x_a \geq 0 & &
\end{array}\right. \tag{3.6}
$$

を定義します．このとき，

- 問題 (3.5) が実行可能である $\Longleftrightarrow$ 問題 (3.6) の最適な目的関数値がゼロ

であることが言えます (演習問題 3.4)．つまり，(3.6) を解くことによって，(存在すれば) もとの問題 (3.5) の実行可能解が求められます．

補助問題 (3.5) を解くために，まず辞書を作ります:

$$
\left|\begin{array}{rcrcrcrcrcr}
 & x_4 & = & -4 & + & x_a & - & 2x_1 & + & 2x_2 & - & x_3 \\
 & x_5 & = & -3 & + & x_a & & & + & 2x_2 & + & 2x_3 \\
 & x_6 & = & -2 & + & x_a & - & 2x_1 & + & x_2 & - & x_3 \\
(& z & = & 0 & & & + & x_1 & + & x_2 & - & 2x_3 &) \\
 & w & = & 0 & - & x_a & & & & & & &
\end{array}\right| .
$$

ここで最大化する目的関数は $w = -x_a$ であり，カッコ内の変数 z はもとの問題の目的関数値を参照しているだけです．この辞書はまだ実行可能ではありませんが，人工変数 x_a の列でピボット演算を行うことにより，常に実行可能な辞書に書き換えることができます．

つまり，他の非基底変数の値をゼロに固定したまま，x_a の値だけを基底変数の値がすべて非負となるまで増加させます．最後に非負となった基底変数と人工変数 x_a を入れ替えるピボット演算を行えば実行可能な辞書がえられます．この例で x_a と入れ替わる基底変数は x_4 で，その結果，実行可能な辞書

$$
\left|\begin{array}{rcrcrcrcrcr}
 & x_a & = & 4 & + & x_4 & + & 2x_1 & - & 2x_2 & + & x_3 \\
 & x_5 & = & 1 & + & x_4 & + & 2x_1 & & & + & 3x_3 \\
 & x_6 & = & 2 & + & x_4 & + & & - & x_2 & & \\
(& z & = & 0 & & & + & x_1 & + & x_2 & - & 2x_3 &) \\
 & w & = & -4 & - & x_4 & - & 2x_1 & + & 2x_2 & - & x_3 &
\end{array}\right| \tag{3.7}
$$

がえられ，algorithm シンプレックス法に入力することができます．

ケース 1　前処理でえられた補助問題の実行可能辞書をシンプレックス法に入力すれば，その目的関数値には

$$w = -x_a \leq 0$$

という明らかな上界が存在するので，非有界となって終了することはありません．実際，辞書 (3.7) で $(r,\ s) = (1,\ 3)$ を中心にピボット演算を行えば，

$$
\left|\begin{array}{rcrcrcrcrcr}
 & x_2 & = & 2 & + & 1/2x_4 & + & x_1 & - & 1/2x_a & + & 1/2x_3 \\
 & x_5 & = & 1 & + & x_4 & + & 2x_1 & & & + & 3x_3 \\
 & x_6 & = & 0 & + & 1/2x_4 & - & x_1 & + & 1/2x_a & - & 1/2x_3 \\
(& z & = & 2 & + & 1/2x_4 & + & 2x_1 & - & 1/2x_a & - & 3/2x_3 \quad) \\
 & w & = & 0 & & & & & - & x_a &
\end{array}\right.
$$

となり，最適性が満たされます．この場合，人工変数 x_a が非基底変数となって値はゼロになりましたので，もとの問題 (3.5) の実行可能解がえられたことになります.「人工変数 x_a が非基底変数」となっていますので，その値を恒等的にゼロとおき，w の等式を無視することで直ちに (3.5) の実行可能辞書:

$$
\left|\begin{array}{rcrcrcrcr}
x_2 & = & 2 & + & 1/2x_4 & + & x_1 & + & 1/2x_3 \\
x_5 & = & 1 & + & x_4 & + & 2x_1 & + & 3x_3 \\
x_6 & = & 0 & + & 1/2x_4 & - & x_1 & - & 1/2x_3 \\
z & = & 2 & + & 1/2x_4 & + & 2x_1 & - & 3/2x_3
\end{array}\right.
$$

がえられます．これを入力として再び algorithm シンプレックス法を実行すれば，もとの問題 (3.5) の最適解が求められるか，あるいは非有界であることが判明します.

ケース 2　それでは，補助問題の最適目的関数値がゼロとなっても「人工変数が基底変数」となっている場合，どのように処理すればよいのでしょうか.

例えば，

$$
\left|\begin{array}{lrcrcrcr}
\text{最大化} & & x_1 & + & 2x_2 & & \\
\text{条　件} & & x_1 & + & x_2 & \leq & 1 \\
 & - & x_1 & - & x_2 & \leq & -1 \\
 & & & & x_1,\ x_2 & \geq & 0
\end{array}\right. \tag{3.8}
$$

に対して補助問題を解くと以下のようになります:

$$
\left|\begin{array}{rcrcrcrcr}
 & x_3 & = & 1 & + & x_a & - & x_1 & - & x_2 \\
 & x_4 & = & -1 & + & x_a & + & x_1 & + & x_2 \\
(& z & = & 0 & & & + & x_1 & + & 2x_2 \ ) \\
 & w & = & 0 & - & x_a & & & &
\end{array}\right.
$$

$$
\rightarrow
\left|\begin{array}{rcrcrcrcr}
 & x_3 & = & 2 & + & x_4 & - & 2x_1 & - & 2x_2 \\
 & x_a & = & 1 & + & x_4 & - & x_1 & - & x_2 \\
(& z & = & 0 & & & + & x_1 & + & 2x_2 \ ) \\
 & w & = & -1 & - & x_4 & + & x_1 & + & x_2
\end{array}\right.
$$

$$
\rightarrow
\left|\begin{array}{rcrcrcrcr}
 & x_1 & = & 1 & + & 1/2x_4 & - & 1/2x_3 & - & x_2 \\
 & x_a & = & 0 & + & 1/2x_4 & + & 1/2x_3 & & \\
(& z & = & 1 & + & 1/2x_4 & - & 1/2x_3 & + & x_2 \ ) \\
 & w & = & 0 & - & 1/2x_4 & - & 1/2x_3 & &
\end{array}\right.
$$

ここで最適性の条件が満たされ，補助問題は解けたことになります．ところが，人工変数 x_a は基底変数であり，基底変数が非基底変数によって値の定まる 1 次関数であることを考えると安易にゼロに固定することはできず，もとの問題 (3.8) の実行可能辞書はえられません．

この場合，x_a を含む等式で，その右辺に現れている (言い換えると，係数がゼロでない) 任意の変数を中心にピボット演算を行えば，実行可能性を保ったまま人工変数 x_a を非基底変数にすることができます．例えば，変数 x_3 の列をピボット列として $(r,\ s) = (2,\ 2)$ を中心とするピボット演算を行えば，x_a を非基底変数とする実行可能辞書:

$$
\left|\begin{array}{rcrcrcrcr}
 & x_1 & = & 1 & + & x_4 & - & x_a & - & x_2 \\
 & x_3 & = & 0 & - & x_4 & + & 2x_a & & \\
(& z & = & 1 & + & x_4 & - & x_a & + & x_2 \ ) \\
 & w & = & 0 & & & - & x_a & &
\end{array}\right.
$$

がえられます (この操作で実行可能性が保たれるのはなぜか？　演習問題 3.6).

以上をまとめれば，任意の基準形 LP に対する実行可能辞書の求め方は次のように記述できます:

procedure 初期化 (基準形 LP)

begin

スラック変数を導入して LP の辞書 D を作る;

if 辞書 D が実行不可能 then

begin

人工変数 x_a を辞書 D に導入し，$w = -x_a$ を最大化する補助問題を作る;

補助問題の辞書 D' を作り，algorithm シンプレックス法に入力する;

{ ただし，もとの目的関数の等式も D' に加え，これにもピボット演算を施す }

if 補助問題の最適目的関数値がゼロ then

begin

補助問題の最適辞書をあらためて D' とおく;

if 辞書 D' で人工変数 x_a が基底 then { ケース 2}

x_a の等式で，右辺に現れている任意の変数と x_a を入れ替えるピボット演算を行い，えられた辞書をあらためて D' とおく;

辞書 D' で $x_a := 0$ とし，w の等式を削除してえられるもとの LP の実行可能辞書をあらためて D とおく

end

end

end;

この procedure 初期化と 2 章の algorothm シンプレックス法を組み合わせて LP を解く方法を，**2 段階シンプレックス法** (two-phase simplex method) とよびます:

algorithm 2段階シンプレックス法

入力: 任意の基準形 LP

begin { 第 1 段階 }

procedure 初期化 (LP) を呼んで LP の辞書 D を求める;

if 辞書 D が実行不可能 then 終了 { 入力 LP は実行不可能 }

else

辞書 D を入力として algorithm シンプレックス法を実行する

end

end;

第 1，第 2 段階ともに 2 章の algorithm シンプレックス法が中心的な役割を果たしますが，そのピボット選択に最小添字規則を用いることで，algorithm 2段階シンプレックス法も有限回での収束が保証されます.

3.4 基本定理の証明

第 1 章であげた LP の基本定理 – 定理 1.1 の主張は，

- 実行可能で有界な LP は最適解をもつ

でした．2段階シンプレックス法を使って，これを証明しましょう:

証明: まず，任意に実行可能で有界な LP 問題 P を選ぶ．第 2 章で説明したように LP は同値な基準形の問題に書き換えられるので，一般性を失うことなく，この問題 P も基準形であると仮定できる．問題 P の実行可能性から，2段階シンプレックス法の第 1 段階で algorithm シンプレックス法を最小添字規則とともに用いて実行可能辞書 D がえられる．さらに，P の有界性から，第 2 段階で D に algorithm シンプレックス法を再び最小添字規則とともに適用すれば P の最適解がえられる．これで基本定理は証明された． ■

演習問題

3.4 LP が実行可能であることと，その補助問題の最適目的関数値がゼロであることが同値なことを示せ.

3.5 次の LP を 2 段階シンプレックス法で解け:

(a)

$$
\begin{array}{lrrrrrr}
\text{最大化} & & 3x_1 & + & 2x_2 & & \\
\text{条　件} & - & 2x_1 & + & x_2 & \leq & 1 \\
 & & x_1 & - & 2x_2 & \leq & -4 \\
 & & x_1 & + & x_2 & \leq & 2 \\
 & & & & x_1,\ x_2 & \geq & 0
\end{array}
$$

(b)

$$
\begin{array}{lrrrrrr}
\text{最大化} & & 3x_1 & + & 2x_2 & & \\
\text{条　件} & - & 2x_1 & + & x_2 & \leq & 1 \\
 & & x_1 & - & 2x_2 & \leq & 0 \\
 & - & x_1 & - & x_2 & \leq & -2 \\
 & & & & x_1,\ x_2 & \geq & 0
\end{array}
$$

3.6 2 段階シンプレックス法の第 1 段階 procedure 初期化のケース 2 において，辞書 D' の実行可能性が保たれることを証明せよ.

3.7 algorithm 2 段階シンプレックス法の記述をもとに，実際に 2 段階シンプレックス法のプログラムを組み，問題 2.2 (a), (b), (c)，問題 3.3，問題 3.5 (a), (b) などを解いてみよ.

[オリジナルのプログラムと計算結果，および考察をレポートにまとめれば期末試験に代えることも可！　その場合，締め切りは試験日から 1 週間後．]