

の任意の1点の情報が、次の瞬間には全領域に伝わるという重要な性質を拡散方程式はもっている。この性質から、陰公式が陽公式より優れていることを以下のように裏付けすることができる。まず、陽公式では U_j^n の情報は次の時刻で $U_{j-1}^{n+1}, U_j^{n+1}, U_{j+1}^{n+1}$ にしか伝わらない。なぜならば(6.13)式で U_j^{n+1} と関係するのは、 $U_{j-1}^n, U_j^n, U_{j+1}^n$ だけであるからである。これに対し、陰公式では U_j^n の情報は連立1次方程式を通じて $j=1, 2, \dots, N-1$ の U_j^{n+1} すべてに影響を及ぼす。すなわち、ある点の情報が次の時刻で空間のすみずみまでゆきわたる。以上のことから、陰公式の方が元の微分方程式の性質を自然に反映していることがわかる。このことが安定性の条件の差となって現われたのである。

6-4 波動方程式

波動方程式の差分法 この節では、(6.5)式の波動方程式の初期値・境界値問題を解くための差分法について説明する。ただし、もうすこし初期条件を一般化して、

$$\left\{ \begin{array}{l} \frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} \\ (0 < x < 1, t > 0) \end{array} \right. \quad (6.32a)$$

$$\left\{ \begin{array}{l} u(x, 0) = \phi(x), \quad \frac{\partial u}{\partial t}(x, 0) = \psi(x) \\ (0 \leq x \leq 1) \end{array} \right. \quad (6.32b)$$

$$u(0, t) = u(1, t) = 0 \quad (t \geq 0) \quad (6.32c)$$

とする。ただし、 ϕ, ψ は任意の関数であり、 $\phi(0) = \phi(1) = 0$ とする。

格子点は、拡散方程式の場合と同じ図 6-7 のものを用いる。前と同様に N , Δt を定め、 $\Delta x = 1/N$ とする。また、 $x_j = j\Delta x$, $t_n = n\Delta t$ とし、微分解 $u(x_j, t_n)$ に対応する差分解を U_j^n とする。

次に、(6.32a)式を次式の差分方程式で近似する。

$$\frac{1}{(\Delta t)^2}(U_j^{n+1} - 2U_j^n + U_j^{n-1}) = \frac{1}{(\Delta x)^2}(U_{j+1}^n - 2U_j^n + U_{j-1}^n) \quad (j=1, 2, \dots, N-1, n=1, 2, \dots) \quad (6.33)$$

この式を書き換えると、

$$U_j^{n+1} = 2U_j^n - U_j^{n-1} + \alpha(U_{j+1}^n - 2U_j^n + U_{j-1}^n) \quad (6.34)$$

となる。ただし、 $\alpha = (\Delta t/\Delta x)^2$ とする。この式は、時刻 t_{n-1}, t_n での U から次の時刻 t_{n+1} での U が計算できるという形をしている。したがって、初期条件として、 U_j^0, U_j^1 を決めたい。

初期条件の導出 まず、 U_j^0 は(6.32b)式の左の式から、

$$U_j^0 = \phi(x_j) \quad (j=0, 1, \dots, N) \quad (6.35)$$

とすればよい。 U_j^1 を決める方法はいくつか考えられるが、ここでは以下のようにする。まず、テイラーの公式を用いると、

$$u(x, \Delta t) = u(x, 0) + \Delta t \frac{\partial u}{\partial t}(x, 0) + \frac{(\Delta t)^2}{2} \frac{\partial^2 u}{\partial t^2}(x, 0) + O((\Delta t)^3) \quad (6.36)$$

となる。(6.32b)式の右の式から $\frac{\partial u}{\partial t}(x, 0) = \psi(x)$ である。さらに、 $t=0$ でも(6.32a)式が成り立つとすると、

$$\frac{\partial^2 u}{\partial t^2}(x, 0) = \frac{\partial^2 u}{\partial x^2}(x, 0) \quad (6.37)$$

となる。この式の右辺は2階差分商を用いて

$$\frac{\partial^2 u}{\partial x^2}(x, 0) \doteq \frac{1}{(\Delta x)^2}\{u(x+\Delta x, 0) - 2u(x, 0) + u(x-\Delta x, 0)\} \quad (6.38)$$

と近似できる。よって(6.36)式は

$$U_j^1 = U_j^0 + \Delta t \psi(x_j) + \frac{\alpha}{2}(U_{j+1}^0 - 2U_j^0 + U_{j-1}^0) \quad (j=1, 2, \dots, N-1) \quad (6.39)$$

という差分方程式で近似できる。さらに、境界条件は(6.32c)式から、

$$U_0^n = U_N^n = 0 \quad (n=0, 1, \dots) \quad (6.40)$$

となる。

波動方程式のアルゴリズム 以上から、数値計算のアルゴリズムは以下のようになる。ただし、解を求める時刻の上限値を T とし、適当な自然数 M を定めて $\Delta t = T/M$ とする。

(1) N, M, T を設定する

$$\Delta x := 1/N, \quad \Delta t := T/M, \quad \alpha := (\Delta t/\Delta x)^2$$

(2) $j := 0, 1, \dots, N$ の順に

$$U_j^0 := \phi(j\Delta x)$$

を繰り返す

 $j := 1, 2, \dots, N-1$ の順に

$$U_j^1 := U_j^0 + \Delta t \psi(j\Delta x) + \frac{\alpha}{2} (U_{j+1}^0 - 2U_j^0 + U_{j-1}^0)$$

を繰り返す

$$U_0^1 := 0, \quad U_N^1 := 0$$

(3) $n := 1, 2, \dots, M-1$ の順に $j := 1, 2, \dots, N-1$ の順に

$$U_j^{n+1} := 2U_j^n - U_{j-1}^{n-1} + \alpha(U_{j+1}^n - 2U_j^n + U_{j-1}^n)$$

を繰り返す

$$U_0^{n+1} := 0, \quad U_N^{n+1} := 0$$

を繰り返す

前節の拡散方程式の場合と同様に、変数を減らしてメモリを節約するためには、
例えば以下のアルゴリズムを使用する。

(1) N, M, T を設定する

$$\Delta x := 1/N, \quad \Delta t := T/M, \quad \alpha := (\Delta t/\Delta x)^2$$

(2) $j := 0, 1, \dots, N$ の順に

$$old_U_j := \phi(j\Delta x)$$

を繰り返す

 $j := 1, 2, \dots, N-1$ の順に

$$cur_U_j := old_U_j + \Delta t \psi(j\Delta x)$$

$$+ \frac{\alpha}{2} (old_U_{j+1} - 2old_U_j + old_U_{j-1})$$

を繰り返す

$$cur_U_0 := 0, \quad cur_U_N := 0$$

$$new_U_0 := 0, \quad new_U_N := 0$$

(3) $n := 1, 2, \dots, M-1$ の順に $j := 1, 2, \dots, N-1$ の順に

$$new_U_j := 2cur_U_j - old_U_j$$

$$+ \alpha(cur_U_{j+1} - 2cur_U_j + cur_U_{j-1})$$

を繰り返す

 $j := 0, 1, \dots, N$ の順に

$$old_U_j := cur_U_j$$

$$cur_U_j := new_U_j$$

を繰り返す

ここで、 $old_U_j, cur_U_j, new_U_j$ はそれぞれ $U_j^{n-1}, U_j^n, U_j^{n+1}$ の役割を果たす。
安定性の条件 次に、安定性の条件を調べる。(6.34)式の特解は、

$$U_j^n = s^n \exp(ikj\Delta x) \quad (6.41)$$

となる。ただし、 s は

$$s^2 - 2\left(1 - 2\alpha \sin^2 \frac{k\Delta x}{2}\right)s + 1 = 0 \quad (6.42)$$

の根であり、 $\beta = \alpha \sin^2 \frac{k\Delta x}{2} (\geq 0)$ とおくと、

$$s = s_1 = 1 - 2\beta + \{4\beta(\beta-1)\}^{1/2} \quad (6.43a)$$

もしくは

$$s = s_2 = 1 - 2\beta - \{4\beta(\beta-1)\}^{1/2} \quad (6.43b)$$

となる。計算が発散しないためには、任意の k に対して $|s_1| \leq 1$ かつ $|s_2| \leq 1$ でなければならない。 $\beta > 1$ の場合は、 s_1, s_2 ともに実数であり、 $s_1 \neq s_2$ 、 $s_1 s_2 = 1$ より $|s_1|$ か $|s_2|$ のどちらかが必ず 1 より大きくなる。 $\beta = 1$ の場合は、重根の場合であり、 $|s_1| = |s_2| = 1$ となる。 $\beta < 1$ の場合は、 s_1, s_2 ともに複素数であり、 $|s_1| = |s_2| = 1$ となる。以上をまとめると、 $\beta \leq 1$ すなわち

$$\alpha \sin^2 \frac{k\Delta x}{2} \leq 1 \quad (6.44)$$

でなければならないことがわかる。さらに、任意の k に対してこの条件が成

立するには、 $\alpha \leq 1$ すなわち

$$\frac{\Delta t}{\Delta x} \leq 1 \quad (6.45)$$

でなければならない。これが安定性の条件である。

波動方程式の計算例 では、実際に数値計算を行なった結果を示す。ここでは、(6.5)式の問題を採用して $\phi(x) = 2x(1-x)$, $\psi(x) = 0$ とする。まず、 $N = 20$ ($\Delta x = 1/20$), $\Delta t = 1/50$ とした結果を図 6-11(a)に示す。この場合は安定性の条件を満たしている。図 6-4 の微分解と比較しても、計算がうまく行なわれていることがわかる。次に、 $N = 20$ ($\Delta x = 1/20$), $\Delta t = 1/10$ とした場合、すなわち、安定性の条件を満たさない場合の結果を図 6-11(b)に示す。時間が進

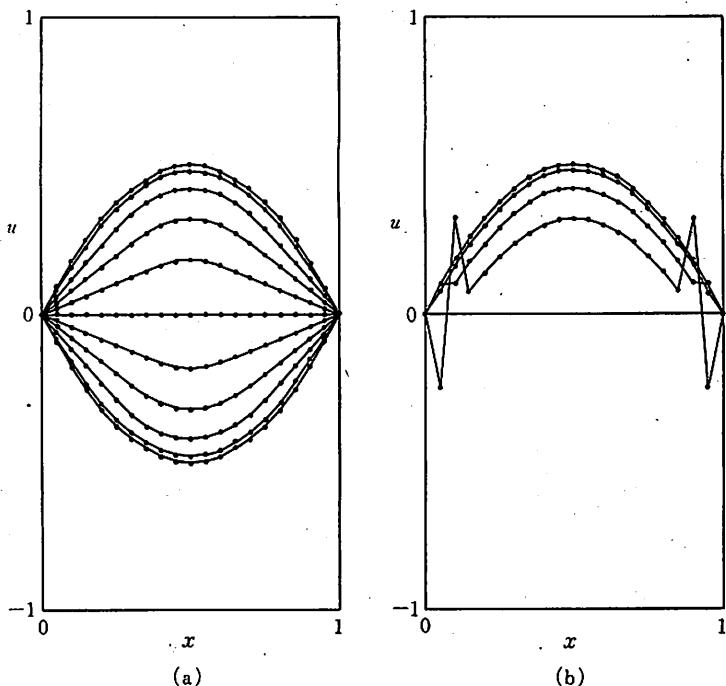


図 6-11 (a) $N=20$ ($\Delta x=1/20$), $\Delta t=1/50$ の場合の差分解。 $n=0, 5, 10, \dots, 50$ でのグラフ。(b) $N=20$ ($\Delta x=1/20$), $\Delta t=1/10$ の場合の差分解。 $n=0, 1, 2, 3$ でのグラフ

むと結果が不安定になり、 $n=3$ で大きく解がずれ始め、その後数時刻で計算が発散する。

波動方程式の性質と安定性の条件 前節の拡散方程式の場合と同様に、安定性の条件は元の微分方程式の性質を反映している。話を簡単にするため、(6.32)式の問題の代わりに空間領域が $-\infty < x < \infty$ の無限領域での問題

$$\left\{ \begin{array}{l} \frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} \\ u(x, 0) = \phi(x), \quad \frac{\partial u}{\partial t}(x, 0) = \psi(x) \end{array} \right. \quad (-\infty < x < \infty, t > 0) \quad (6.46a)$$

$$\left\{ \begin{array}{l} u(x, t) = \frac{1}{2} \{ \phi(x-t) + \phi(x+t) \} + \frac{1}{2} \int_{x-t}^{x+t} \psi(s) ds \end{array} \right. \quad (-\infty < x < \infty) \quad (6.46b)$$

を考えることにする。この問題の解は

$$u(x, t) = \frac{1}{2} \{ \phi(x-t) + \phi(x+t) \} + \frac{1}{2} \int_{x-t}^{x+t} \psi(s) ds \quad (6.47)$$

であり、ダランベール(d'Alembert)の解とよばれる。ある時刻 t_0 、ある位置 x_0 での u の値は $\phi(x_0 - t_0)$, $\phi(x_0 + t_0)$ および $x_0 - t_0 \leq s \leq x_0 + t_0$ の範囲の $\psi(s)$ に依存している。そして、 $\phi(x)$ および $\psi(x)$ は $t=0$ での初期条件で与えられている。初期条件に対する解のこのような依存性を図に示したものが図 6-12 である。3 角形 ABC は直角 2 等辺 3 角形であり、点 A の u の値は線分 BC で表わされる領域の ϕ と ψ の情報から決定される。

一方、差分解は以下に示すような初期条件の依存性を有している。ここで、(6.46)式と問題設定を合わせるために(6.40)式の境界条件は考慮しない。空間

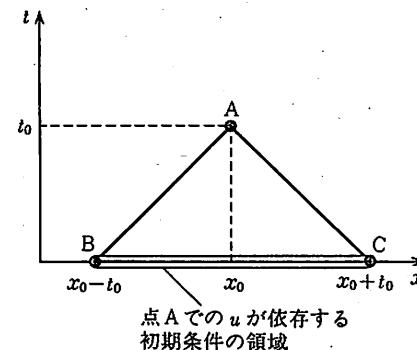


図 6-12 微分解の初期条件
に対する依存性

方向の格子点の範囲を $-\infty < j < \infty$ に拡げ、(6.35), (6.39)式の初期条件の下で(6.34)式を解き、 U_j^n を n の小さい方から順に求めるとする。ある格子点 (x_{j_0}, t_{n_0}) における差分解 $U_{j_0}^{n_0}$ は、(6.34)式より $U_{j_0-1}^{n_0-1}, U_{j_0}^{n_0-1}, U_{j_0+1}^{n_0-1}, U_{j_0}^{n_0-2}$ の 4 つの U の値から決定される。さらに、例えば $U_{j_0-1}^{n_0-1}$ は $U_{j_0-2}^{n_0-2}, U_{j_0-1}^{n_0-2}, U_{j_0}^{n_0-2}, U_{j_0-1}^{n_0-3}$ から決定される。こうして時刻をさかのぼっていき、初期条件を考慮すると、結局 $U_{j_0}^{n_0}$ は、 $j_0 - n_0 \leq j \leq j_0 + n_0$ の範囲の $\phi(x_j)$ と、 $j_0 - n_0 + 1 \leq j \leq j_0 + n_0 - 1$ の範囲の $\psi(x_j)$ とから決定されることがわかる。

(a) $\Delta t / \Delta x > 1$, (b) $\Delta t / \Delta x \leq 1$ の 2 通りの場合に、差分解の依存性と微分解の依存性を重ねて描いたものが図 6-13 である。 $\Delta t / \Delta x > 1$ の場合は、微分解の依存領域を差分解の方が十分にカバーしていない。ということは、差分解に微分解を反映させるための情報が十分でないことになる。一方、 $\Delta t / \Delta x \leq 1$ の場合は差分解の依存領域が微分解の依存領域をカバーしている。以上のことから $\Delta t / \Delta x \leq 1$ という安定性の条件に反映したと考えられるのである。

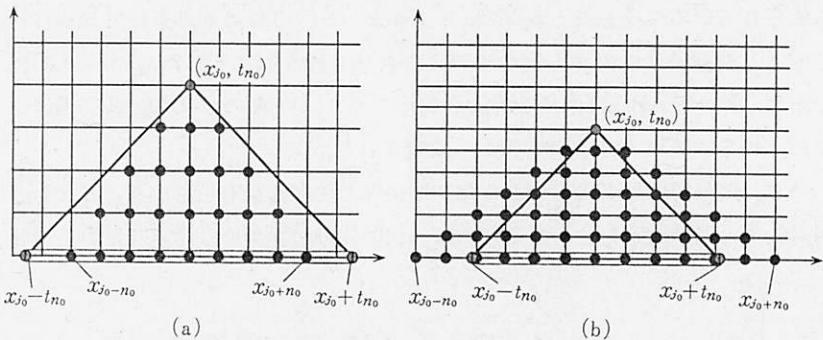


図 6-13 差分解の初期条件に対する依存性。
(a) $\Delta t / \Delta x > 1$ の場合, (b) $\Delta t / \Delta x \leq 1$ の場合

6-5 ラプラス方程式

ラプラス方程式の差分法 この節では、(6.7)式のラプラス方程式の境界値問題を考える。ただし、境界条件を一般化して、

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \quad (0 < x < 1, 0 < y < 1) \quad (6.48a)$$

$$\begin{cases} u(x, 0) = \phi_1(x), & u(x, 1) = \phi_2(x) \\ u(0, y) = \psi_1(y), & u(1, y) = \psi_2(y) \end{cases} \quad (0 \leq x \leq 1, 0 \leq y \leq 1) \quad (6.48b)$$

とする。また、境界条件が整合するために、 $\phi_1(0) = \psi_1(0)$, $\phi_1(1) = \psi_2(0)$, $\phi_2(0) = \psi_1(1)$, $\phi_2(1) = \psi_2(1)$ とする。

まず、格子点を用意する。今回は独立変数 x, y の範囲が限られているので、図 6-14 のような格子点を用いる。ただし、 $Nh = 1$ とする。 x 方向と y 方向の格子点の数と同じにしたが、異なる数の格子点を用いても構わない。また、 $x_i = ih$, $y_j = jh$ とし、微分解 $u(x_i, y_j)$ に対応する差分解を $U_{i,j}$ とする。

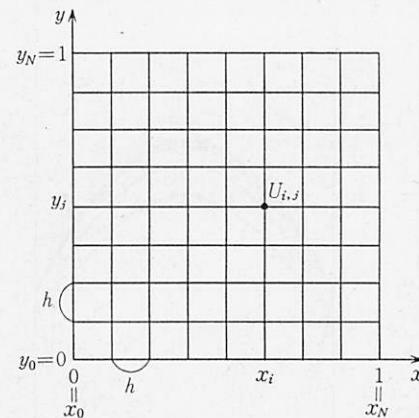


図 6-14 ラプラス方程式の格子点

次に(6.48a)式を次式の差分方程式で置き換える。

$$\frac{1}{h^2}(U_{i+1,j} - 2U_{i,j} + U_{i-1,j}) + \frac{1}{h^2}(U_{i,j+1} - 2U_{i,j} + U_{i,j-1}) = 0 \quad (i=1, 2, \dots, N-1, j=1, 2, \dots, N-1) \quad (6.49)$$

この式を整理すると、

$$4U_{i,j} - (U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1}) = 0 \quad (6.50)$$

となる。

差分方程式と最大値の原理 (6.50)式は、図 6-15 に示すように「口での

U の値は、その周囲の 4 つの ● での U の値の平均値に等しい」という意味にも解釈できる。ということは、□、●の合計 5 点での U のうち、最大値および最小値をとるものは必ず周囲の ● の点のどれかに存在する。一方、元のラプラス方程式(6.48a)式の微分解 u は最大値の原理(maximum principle)とよばれる性質を満たしている。最大値の原理をわかりやすく表現すれば、次のようになる。

ある領域でラプラス方程式を満たす解 u を考える。そして、その領域全体における u の最大値および最小値を考える。 u が定数でない限り、領域の内部の点では最大値も最小値もとらない。このことから上の差分方程式が微分方程式の性質をうまく反映していることがわかる。

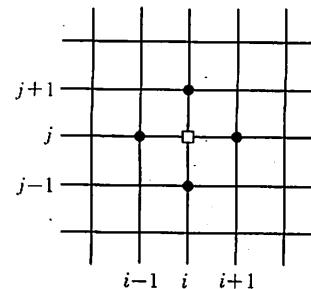


図 6-15 (6.50) 式で表わされる $U_{i,j}$ の関係

連立 1 次方程式の導出 さて、境界条件(6.48b)式から

$$\begin{cases} U_{i,0} = \phi_1(x_i), & U_{i,N} = \phi_2(x_i) \quad (i=0,1,\dots,N) \\ U_{0,j} = \phi_1(y_j), & U_{N,j} = \phi_2(y_j) \quad (j=0,1,\dots,N) \end{cases} \quad (6.51)$$

となる。(6.51)式の条件の下で(6.50)式を解き、未知変数 $U_{i,j}$ ($i=1,2,\dots,N-1, j=1,2,\dots,N-1$) の値を求めれば、差分解が得られる。しかし、(6.50)式は $U_{i,j}$ の値を端から順に計算できるという形をしていない。(6.50)式を各 i, j について並べると、 $U_{i,j}$ に関する連立 1 次方程式になっていることがわかる。例えば $N=4$ のとき、行列の形で表わすと、

$$\left(\begin{array}{ccccccccc} 4 & -1 & 0 & -1 & & & & & \\ -1 & 4 & -1 & 0 & -1 & & & & \\ 0 & -1 & 4 & 0 & 0 & -1 & & & \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & & \\ & -1 & 0 & -1 & 4 & -1 & 0 & -1 & \\ & & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ & & & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & & & & -1 & 0 & -1 & 4 & -1 \\ & & & & & -1 & 0 & -1 & 4 \end{array} \right) \left(\begin{array}{c} U_{1,1} \\ U_{2,1} \\ U_{3,1} \\ U_{4,1} \\ U_{1,2} \\ U_{2,2} \\ U_{3,2} \\ U_{4,2} \\ U_{1,3} \\ U_{2,3} \\ U_{3,3} \end{array} \right) = \left(\begin{array}{c} U_{0,1} + U_{1,0} \\ U_{2,0} \\ U_{4,1} + U_{3,0} \\ U_{0,2} \\ 0 \\ U_{4,2} \\ U_{0,3} + U_{1,4} \\ U_{2,4} \\ U_{4,3} + U_{3,4} \end{array} \right) \quad (6.52)$$

となる。なお、 $U_{i,j}$ のうち(6.51)式により値が与えられているものを右辺に、未知のものを左辺にまとめた。

上の行列の要素には規則性がある。一般の N に対する連立 1 次方程式は次式のようになる。

$$\left(\begin{array}{cccccc} A & B & & 0 & & \\ B & A & B & & & \\ & \ddots & \ddots & \ddots & & \\ & & B & A & B & \\ 0 & & & B & A & \\ & & & & B & A \end{array} \right) \left(\begin{array}{c} U_1 \\ U_2 \\ \vdots \\ U_{N-2} \\ U_{N-1} \end{array} \right) = \left(\begin{array}{c} f_1 \\ f_2 \\ \vdots \\ f_{N-2} \\ f_{N-1} \end{array} \right) \quad (6.53)$$

ここで、 A, B は $(N-1) \times (N-1)$ 小行列であり、それぞれ

$$A = \begin{pmatrix} 4 & -1 & & 0 \\ -1 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ 0 & & -1 & 4 & -1 \\ & & & -1 & 4 \end{pmatrix}, \quad B = \begin{pmatrix} -1 & & & 0 & \\ & -1 & & & \\ & & \ddots & & \\ 0 & & & -1 & \\ & & & & -1 \end{pmatrix} \quad (6.54)$$

である。また、 U_j, f_j は、それぞれ

$$\begin{cases} U_j = (U_{1,j}, U_{2,j}, \dots, U_{N-2,j}, U_{N-1,j})^T & (j=1,2,\dots,N-1) \\ f_1 = (U_{0,1} + U_{1,0}, U_{2,0}, \dots, U_{N-2,0}, U_{N,1} + U_{N-1,0})^T \\ f_j = (U_{0,j}, 0, \dots, 0, U_{N,j})^T & (j=2,3,\dots,N-2) \\ f_{N-1} = (U_{0,N-1} + U_{1,N}, U_{2,N}, \dots, U_{N-2,N}, U_{N,N-1} + U_{N-1,N})^T \end{cases} \quad (6.55)$$

で定義されるベクトルである。 T はベクトルや行列の転置を表わす。

(6.53)式の左辺の行列は、小行列が 3 重の帯の形で対角に並んでいる。このような形の行列をブロック 3 重対角行列(block tridiagonal matrix)といいう。

この行列のはとんどの要素は0であり、(6.53)式を効率よく解く方法がいくつか存在する。そのような方法のひとつを7-4節でくわしく説明する(第7章演習問題[5])。ここでは、(6.53)式は解くことができるとだけしておく。

ラプラス方程式の計算例 (6.48)式の問題に(6.7b)式と同じ境界条件、すなわち、

$$\begin{cases} u(x, 0) = \sin \pi x, & u(x, 1) = 0 \\ u(0, y) = u(1, y) = 0 \end{cases} \quad (0 \leq x \leq 1, 0 \leq y \leq 1) \quad (6.56)$$

を課した場合について数値計算を行なう。 $N=20$ とし、そのときの差分解の結果を等高線図として図6-16に示す。ここには示していないが、図6-6(b)の微分解の等高線を重ねてプロットすると、線の違いが見えるか見えないかの程度まで一致している。なお、 $N \rightarrow \infty$ すなわち $h \rightarrow 0$ の極限をとると、差分解が微分解に収束することがわかっている。このことから、拡散方程式や波動方程式のように安定性の条件を気にする必要がない。なぜそうなるかの直観的な説明は以下のとおりである。前に示したように、(6.50)式は最大値の原理を反映した性質をもっている。したがって、すべての $U_{i,j}$ の中で最大値および最小値をとるものは領域の境界上の格子点に存在する。このことから、領域内部の $U_{i,j}$ の値はつねに発散することがない。

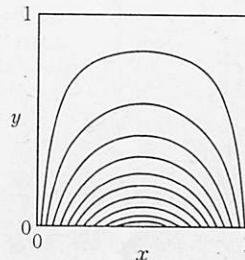


図6-16 $N=20$ の場合の差分解の等高線図。各等高線の高さは図6-6(b)と同じ

第6章 演習問題

[1] 次の拡散方程式の初期値・境界値問題を差分法で解く。

$$\begin{cases} \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} & (0 < x < 1) \\ u(x, 0) = \begin{cases} x & (0 \leq x \leq 1/2) \\ 1-x & (1/2 < x \leq 1) \end{cases} \\ u(0, t) = u(1, t) = 0 & (t \geq 0) \end{cases}$$

- (1) 陽公式を用いて、 $(\Delta x, \Delta t) = (1/6, 1/100), (1/10, 1/100), (1/10, 1/500)$ のそれぞれの場合について $t=0.02, 0.04, 0.06, 0.08$ での $x=0.5$ における値を答えよ。計算が不安定になっているのはどの場合か。
- (2) 陰公式を用いて、 $(\Delta x, \Delta t) = (1/50, 1/100)$ の場合に $t=0.02, 0.04, 0.06, 0.08$ での $x=0.5$ における値を答えよ。

[2] 拡散方程式 $\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$ を近似する差分方程式として、クランク-ニコルソン(Crank-Nicolson)の公式

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = \frac{1}{2} \frac{U_{j+1}^{n+1} - 2U_j^{n+1} + U_{j-1}^{n+1}}{(\Delta x)^2} + \frac{1}{2} \frac{U_{j+1}^n - 2U_j^n + U_{j-1}^n}{(\Delta x)^2}$$

がある。この式は無条件安定であることを示せ。

[3] 次の波動方程式の初期値・境界値問題を差分法で解け。

$$\begin{cases} \frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} & (0 < x < 1, t > 0) \\ u(x, 0) = x(2-x), & \frac{\partial u}{\partial t}(x, 0) = 0 \quad (0 \leq x \leq 1) \\ u(0, t) = 0, & \frac{\partial u}{\partial x}(1, t) = 0 \quad (t \geq 0) \end{cases}$$

ただし $\Delta x = 1/20$, $\Delta t = 1/50$ とし、0から1まで0.1刻みの時刻で $x=0.5$ における値を調べよ。なお、(6.40)式を修正して $x=1$ での境界条件を以下のように与えるとする。

$$\frac{U_N^n - U_{N-1}^n}{\Delta x} = 0 \quad \text{すなわち} \quad U_N^n = U_{N-1}^n \quad (n > 0)$$

[4] 次式は波動方程式と関連の深い偏微分方程式である。

$$\frac{\partial u}{\partial t} = \frac{\partial u}{\partial x}$$

この方程式を近似する以下の差分方程式の安定性の条件を調べよ。

$$(1) \frac{U_j^{n+1} - U_j^n}{\Delta t} = \frac{U_{j+1}^n - U_{j-1}^n}{2\Delta x}$$

$$(2) \frac{U_j^{n+1} - U_j^n}{\Delta t} = \frac{U_{j+1}^n - U_j^n}{\Delta x}$$

[5] ラプラス方程式の境界値問題(6.48)式で

$$\begin{cases} \phi_1(x) = \sin \pi x, & \phi_2(x) = 0 \\ \phi_1(y) = 2y(1-y), & \phi_2(y) = 0 \end{cases}$$

とする。差分法を用いて $N=20$ の場合に解き、 $(x, y) = (0.5, 0.5), (0.25, 0.25), (0.25, 0.75), (0.75, 0.25), (0.75, 0.75)$ での値を求めよ。

[6]

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -f(x, y)$$

はポアソン(Poisson)方程式とよばれる。ただし、関数 $f(x, y)$ は与えられているとする。この方程式を $0 < x < 1, 0 < y < 1$ の領域で(6.48b)式の境界条件の下で解くことを考える。 $N=4$ の場合、(6.52)式の連立1次方程式をどのように修正すればよいか。

7 連立1次方程式

連立1次方程式は、方程式の構造としては単純な部類に属している。ところが、いざ手計算で解こうとすると、元の数が数個から十数個程度のものが限界であろう。一方、数値計算で対象とする連立1次方程式の元の数は、それとは比べものにならないくらいに多いことがしばしばである。そこでこの章では、計算量を考慮に入れながら、連立1次方程式を解くための基本的な方法について説明する。

7-1 連立1次方程式

連立1次方程式とは 鶴亀算とよばれる、よく知られた問題がある。1例を挙げると

鶴と亀が合計 10 匹いる。足の数の合計は 32 本である。鶴と亀はそれぞれ何匹いるか

である。鶴と亀の匹数をそれぞれ x, y とすると、この問題は次の 2 つの方程式に翻訳できる。

$$\begin{cases} x + y = 10 & \text{①} \\ 2x + 4y = 32 & \text{②} \end{cases} \quad (7.1)$$

この問題は以下のようにして解くことができる。まず、②式から①式の 2 倍

を引き去る。すると、

$$\begin{cases} x + y = 10 & \text{①} \\ 2y = 12 & \text{②}' \leftarrow \text{②} - 2 \times \text{①} \end{cases} \quad (7.2)$$

となる。②'式より $y=6$ となり、その値を①式に代入して $x=4$ となる。

上の問題のように、いくつかの未知数に対する1次の代数方程式を複数個連立させたものを連立1次方程式(system of linear equations)といい、その一般形は

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,N}x_N = y_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,N}x_N = y_2 \\ \cdots \\ a_{M,1}x_1 + a_{M,2}x_2 + \cdots + a_{M,N}x_N = y_M \end{cases} \quad (7.3)$$

と表わされる。ここで N は未知変数 x_j ($j=1, 2, \dots, N$) の個数、 M は式の個数である。また、係数 $a_{i,j}$ ($i=1, 2, \dots, M$, $j=1, 2, \dots, N$) および右辺の y_i ($i=1, 2, \dots, M$) はすべて与えられているとする。さらに本書では、次式で表わされるように、式の個数 M が未知変数の個数 N に等しい場合だけを考えることにする。

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,N}x_N = y_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,N}x_N = y_2 \\ \cdots \\ a_{N,1}x_1 + a_{N,2}x_2 + \cdots + a_{N,N}x_N = y_N \end{cases} \quad (7.4)$$

この式を行列で表現すると、

$$\underbrace{\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,N} \\ \cdots & & & \\ a_{N,1} & a_{N,2} & \cdots & a_{N,N} \end{pmatrix}}_{\text{係数行列 } A} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}}_{\text{ベクトル } \mathbf{x}} = \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}}_{\text{ベクトル } \mathbf{y}} \quad (7.5)$$

すなわち

$$A\mathbf{x} = \mathbf{y}$$

となる。ここで A は係数行列(coefficient matrix)とよばれ、 $N \times N$ 行列すな

わち N 次の正方形である。

連立1次方程式を解く問題は理工学のいろいろな場面で登場する。本書でも第3章の(3.32), (3.52)式、第5章の(5.73)式、第6章の(6.28), (6.53)式に連立1次方程式が登場している。

数値計算の必要性 連立1次方程式は一見単純であるが、じつは相当に奥が深い。数値計算においても連立1次方程式の解法について多くの研究がなされている。特に問題となるのは計算量の問題である。実用的な数値計算では N の値が100万近い大規模な問題がざらにある。そして、あらゆる連立1次方程式に対して計算効率がよいオールマイティな解法は存在しない。ところが、解こうとする問題に現われる連立1次方程式の形、特に、(7.5)式の係数行列 A の形に特定のパターンが現われることが多い。そこで、そのパターンに応じて解法を選んで解くというのが実情である。以降の節ではそれら解法のうちから基本的なものを3つ紹介する。

一意解の存在 (7.4)式の連立1次方程式は、解 $\mathbf{x}=(x_1, x_2, \dots, x_N)^T$ がただひとつに定まる場合もあれば、解が無数に存在する場合や解がない場合もある。例えば、

$$\begin{cases} x_1 - x_2 + 2x_3 = 5 \\ -x_1 + 2x_2 - 3x_3 = -6 \\ 3x_1 + x_2 + x_3 = 8 \end{cases} \quad (7.6)$$

の場合、解は $x_1=1$, $x_2=2$, $x_3=3$ と一意に定まる。ところが、

$$\begin{cases} x_1 - x_2 + 2x_3 = 5 \\ -x_1 + 2x_2 - 3x_3 = -6 \\ x_1 - 3x_2 + 4x_3 = 7 \end{cases} \quad (7.7)$$

の場合、 t をパラメータとして $x_1=-t+4$, $x_2=t-1$, $x_3=t$ となり、解は無数に存在する。さらに、

$$\begin{cases} x_1 - x_2 + 2x_3 = 5 \\ -x_1 + 2x_2 - 3x_3 = -6 \\ x_1 - 3x_2 + 4x_3 = 4 \end{cases} \quad (7.8)$$

の場合は解が存在しない。

このように解が無数に存在する場合や、解がない場合をも想定した数値計算法を考えることも可能である。しかし、その話題は本書の範囲を超えるので、ここで対象とする連立1次方程式は、解が一意に存在することが保証されないと仮定する。これでも十分広い範囲の実用的な計算に有効である。また、前章までに扱った連立1次方程式は、どれも一意に解が定まる。なお、解が一意に存在するための必要十分条件は(7.5)式の係数行列 A の行列式 $\det(A)$ の値が0でないことである。このとき、行列 A は正則(regular)であるという。

7-2 ガウスの消去法

ガウスの消去法の具体例 最初に、ガウスの消去法(Gaussian elimination)とよばれる解法を説明しよう。ガウスの消去法は、一般の連立1次方程式に適用可能な汎用性のあるものである。まず、3元連立1次方程式にガウスの消去法を適用した具体例を以下に示すことから始めよう。前節でも登場した

$$\begin{cases} x_1 - x_2 + 2x_3 = 5 & \text{(1)} \\ -x_1 + 2x_2 - 3x_3 = -6 & \text{(2)} \\ 3x_1 + x_2 + x_3 = 8 & \text{(3)} \end{cases} \quad (7.9)$$

について考える。ガウスの消去法の第1段階は変数を消去していく手続きである。 (1) 式を用いて $\text{(2)}, \text{(3)}$ 式の変数 x_1 を消去しよう。各式の x_1 の係数を見比べると、 $\text{(2)} + \text{(1)}$, $\text{(3)} - 3 \times \text{(1)}$ を計算すればよいことがわかる。結果は

$$\begin{cases} x_1 - x_2 + 2x_3 = 5 & \text{(1)} \\ x_2 - x_3 = -1 & \text{(2')} \leftarrow \text{(2)} + \text{(1)} \\ 4x_2 - 5x_3 = -7 & \text{(3')} \leftarrow \text{(3)} - 3 \times \text{(1)} \end{cases} \quad (7.10)$$

となる。次に $\text{(3)}' - 4 \times \text{(2)'}$ を計算して (3)' 式から変数 x_2 を消去する。結果は

$$\begin{cases} x_1 - x_2 + 2x_3 = 5 & \text{(1)} \\ x_2 - x_3 = -1 & \text{(2}'} \\ -x_3 = -3 & \text{(3}'' \leftarrow \text{(3}'} - 4 \times \text{(2}'} \end{cases} \quad (7.11)$$

となる。これで第1段階は終了である。

第2段階は解 (x_1, x_2, x_3) を算出する手続きである。上の最後の連立1次方

程式を眺めると、第3式には変数 x_3 のみが残り、第2式には x_2 と x_3 が、第1式には x_1, x_2, x_3 が残っている。ということは、下の式から順に解いて x_3, x_2, x_1 の順に値を求めていけばよい。こうして、

$$\begin{cases} x_3 = (-3)/(-1) = 3 \\ x_2 = x_3 - 1 = 2 \\ x_1 = x_2 - 2x_3 + 5 = 1 \end{cases} \quad (7.12)$$

となって解 (x_1, x_2, x_3) が求められた。

ガウスの消去法は、以上のように2つの手続きを経て解を求める方法である。第1の手続きは、上方の式を用いて下方の式の変数を次々と消去していくので前進消去(forward elimination)とよばれる。そして第2の手続きは、下方の式から変数の値を確定し、その値をひとつ上の式に代入してまた別の変数の値を確定していくので後退代入(backward substitution)とよばれる。

ガウスの消去法の一般形 上の2つの手続きを N 元連立1次方程式に対して一般化することはそれほど難しくない。まず、連立1次方程式

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + \cdots + a_{1,N}x_N = y_1 & \text{(1)} \\ a_{2,1}x_1 + a_{2,2}x_2 + a_{2,3}x_3 + \cdots + a_{2,N}x_N = y_2 & \text{(2)} \\ a_{3,1}x_1 + a_{3,2}x_2 + a_{3,3}x_3 + \cdots + a_{3,N}x_N = y_3 & \text{(3)} \\ \dots & \vdots \\ a_{N,1}x_1 + a_{N,2}x_2 + a_{N,3}x_3 + \cdots + a_{N,N}x_N = y_N & \text{(N)} \end{cases} \quad (7.13)$$

が与えられたとしよう。この方程式に前進消去の手続きを施す。最初に (1) 式を用いて $\text{(2)} \sim \text{(N)}$ 式の変数 x_1 を消去する。

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + \cdots + a_{1,N}x_N = y_1 & \text{(1)} \\ a'_{2,2}x_2 + a'_{2,3}x_3 + \cdots + a'_{2,N}x_N = y'_2 & \text{(2}'} \leftarrow \text{(2)} - (a_{2,1}/a_{1,1}) \times \text{(1)} \\ a'_{3,2}x_2 + a'_{3,3}x_3 + \cdots + a'_{3,N}x_N = y'_3 & \text{(3}'} \leftarrow \text{(3)} - (a_{3,1}/a_{1,1}) \times \text{(1)} \\ \dots & \vdots \\ a'_{N,2}x_2 + a'_{N,3}x_3 + \cdots + a'_{N,N}x_N = y'_N & \text{(N}'} \leftarrow \text{(N)} - (a_{N,1}/a_{1,1}) \times \text{(1)} \end{cases} \quad (7.14)$$

2番目以降の式の係数 $a_{i,j}$ および右辺の y_i は一般にもとのものと異なる値になるので、 $a'_{2,2}$ のようにダッシュを付けることにする。次に $\text{(2}'}$ 式を用いて $\text{(3}'} \sim$

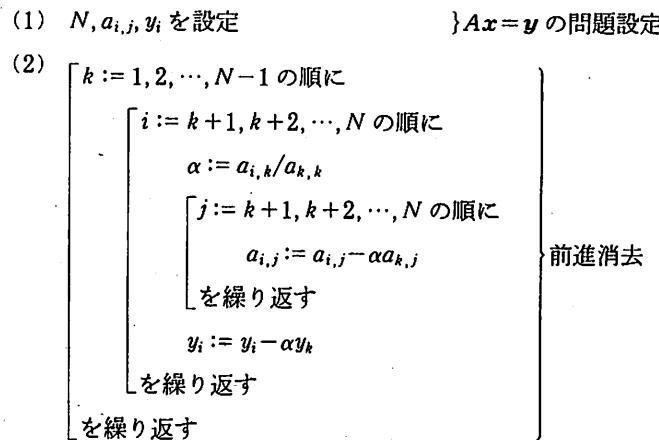
⑩'式の変数 x_2 を消去する。

$$\left\{ \begin{array}{l} a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + \cdots + a_{1,N}x_N = y_1 \\ a'_{2,2}x_2 + a'_{2,3}x_3 + \cdots + a'_{2,N}x_N = y'_2 \\ a''_{3,3}x_3 + \cdots + a''_{3,N}x_N = y''_3 \\ \vdots \\ a''_{N,3}x_3 + \cdots + a''_{N,N}x_N = y''_N \end{array} \right. \quad \begin{array}{l} ① \\ ②' \\ ③'' \leftarrow ③' - (a'_{3,2}/a'_{2,2}) \times ②' \\ \vdots \\ ⑩'' \leftarrow ⑩' - (a'_{N,2}/a'_{2,2}) \times ②' \end{array}$$

(7.15)

以下、同様の手続きを繰り返す。

前進消去のアルゴリズム 連立1次方程式の問題が与えられてから前進消去を終えるまでの手続き全体のアルゴリズムは次のようになる。



前の説明では、変数を x_1, x_2, \dots の順に消去していくと、例えば係数 $a_{2,2}$ が $a'_{2,2}$ に、 $a_{3,3}$ が $a''_{3,3}$ に置き換わっていった。アルゴリズムでは、新しい係数を設ける代わりに、代入の性質を利用して各 $a_{i,j}, y_i$ を新たに定義し直している。なお、このアルゴリズムを終了しても $i > j$ を満たす $a_{i,j}$ は一般に 0 にならない。というのは、次の後退代入の手続きでは「消去」されているはずの $a_{i,j}$ を使用しないので、わざわざ 0 にする必要がないからである。

後退代入のアルゴリズム 前進消去のアルゴリズムを終了すると、最終的に求められた $a_{i,j}$ ($i=1, 2, \dots, N, j=i, i+1, \dots, N$) および y_i ($i=1, 2, \dots, N$)

の値によって

$$\left\{ \begin{array}{l} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,N}x_N = y_1 \\ a_{2,2}x_2 + \cdots + a_{2,N}x_N = y_2 \\ \cdots \\ a_{N-1,N-1}x_{N-1} + a_{N-1,N}x_N = y_{N-1} \\ a_{N,N}x_N = y_N \end{array} \right. \quad (7.16)$$

という形の連立1次方程式が得られる。この式を用いれば、最下行の式から x_N が計算でき、その値をひとつ上の式に代入して x_{N-1} が計算できる。この手続きを繰り返していくと、 $x_N \rightarrow x_{N-1} \rightarrow \cdots \rightarrow x_1$ の順に x_i を求めることができる。これが後退代入の手続きであり、そのアルゴリズムは以下のようになる。

$$\left. \begin{array}{l} (3) \quad x_N := y_N/a_{N,N} \\ \quad i := N-1, N-2, \dots, 1 \text{ の順に} \\ \quad x_i := \left(y_i - \sum_{k=i+1}^N a_{i,k}x_k \right) / a_{i,i} \\ \quad \text{を繰り返す。} \end{array} \right| \text{後退代入}$$

こうして求める解 x が得られた。

ガウスの消去法の計算量 ガウスの消去法の計算量を四則演算の回数で見積もることにしよう。前進消去のアルゴリズムで必要な乗除算の回数は

$$\sum_{k=1}^{N-1} (N-k)(N-k+2) = \frac{N^3}{3} + \frac{N^2}{2} - \frac{5N}{6} \quad (7.17)$$

となるので、 N が十分大きいと約 $N^3/3$ 回と見積もることができる。同様に加減算も約 $N^3/3$ 回必要となる。後退代入の過程では乗除算、加減算ともに約 $N^2/2$ 回必要となる。したがってガウスの消去法のアルゴリズム全体で乗除算、加減算ともに約 $N^3/3$ 回を要する。これは N が大きくなるにつれて急速に増大する計算量である。例えば、 $N=30$ のときは約 9000 回の乗除算、加減算ですむところを、 $N=300$ では約 900 万回もかかるのである。いかに計算の速い計算機といえども、解くことのできる問題の規模は計算時間の制約によって限定される。

アルゴリズムの問題点 上で示したアルゴリズムにはすこし問題点がある。

ただし、この問題点は与えられた連立1次方程式の形によっては考慮する必要がないこともあるが、ここではそのことに立ち入らない。以下に問題点とそれに対するアルゴリズムの改良を示す。

まず、前のアルゴリズムでは前進消去の過程を与えられた式の順序通りになつていった。ところが、例えば

$$\begin{cases} x_1 + x_2 + x_3 = 6 & \text{①} \\ 2x_1 + 2x_2 - x_3 = 3 & \text{②} \\ -x_1 + 3x_2 + x_3 = 8 & \text{③} \end{cases} \quad (7.18)$$

という問題では、最初に②, ③式から変数 x_1 を消去した段階で

$$\begin{cases} x_1 + x_2 + x_3 = 6 & \text{①} \\ -3x_3 = -9 & \text{②}' \leftarrow \text{②} - 2 \times \text{①} \\ 4x_2 + 2x_3 = 14 & \text{③}' \leftarrow \text{③} + \text{①} \end{cases} \quad (7.19)$$

となる。すると、アルゴリズムに従って②'式を用いて③'式から変数 x_2 を消去しようにも、②'式の x_2 の係数がもはや 0 であるので不可能である。無理にアルゴリズムに従って計算を進めると、0 で割るという不可能な操作が生じてしまう。これを避けるためには②'式と③'式の順序を入れ替えればよい。式の順序を入れ替えても連立1次方程式は等価であるからである。

さらに、係数が完全に 0 にならなくとも丸めの誤差による問題が生じことがある。例えば

$$\begin{cases} 0.0003x_1 + x_2 + x_3 = 5 & \text{①} \\ 2x_1 - x_2 + 2x_3 = 6 & \text{②} \\ -x_1 + 3x_2 + x_3 = 8 & \text{③} \end{cases} \quad (7.20)$$

という連立1次方程式を考える。正しい答は $x_1 = 10000/9979 = 1.0021\cdots$, $x_2 = 19970/9979 = 2.0012\cdots$, $x_3 = 29922/9979 = 2.9984\cdots$ である。ガウスの消去法を用いてこの方程式を有効数字5桁で解くと、

$$\begin{cases} 0.0003x_1 + x_2 + x_3 = 5 & \text{①} \\ -6667.7x_2 - 6664.7x_3 = -33328 & \text{②}' \leftarrow \text{②} - 6666.7 \times \text{①} \\ 3336.3x_2 + 3334.3x_3 = 16675 & \text{③}' \leftarrow \text{③} + 3333.3 \times \text{①} \end{cases} \quad (7.21)$$

$$\begin{cases} 0.0003x_1 + x_2 + x_3 = 5 & \text{①} \\ -6667.7x_2 - 6664.7x_3 = -33328 & \text{②}' \\ -0.50000x_3 = -1.0000 & \text{③}'' \leftarrow \text{③}' + 0.50037 \times \text{②}' \end{cases} \quad (7.22)$$

となる。これより $x_1 = 2.0000$, $x_2 = 2.9994$, $x_3 = 2.0000$ となり、正しい答とは全く異なってしまう。

上の現象は、最初に①式を用いて②, ③式から変数 x_1 を消去した段階に端を発する。このとき、 $\text{②}' - 6666.7 \times \text{①}$, $\text{③}' + 3333.3 \times \text{①}$ という式変形を行なうと、①式に大きな数をかけて足したり引いたりするため、②, ③式の係数や右辺の値が②', ③'式において有効数字の下位の桁に押し込められてしまう。このため、③''式に大きな誤差が生じるのである。

これを避けるためには、最初から①式と②式の順序を入れ替えればよい。すなわち、

$$\begin{cases} 2x_1 - x_2 + 2x_3 = 6 & \text{②} \\ 0.0003x_1 + x_2 + x_3 = 5 & \text{①} \\ -x_1 + 3x_2 + x_3 = 8 & \text{③} \end{cases} \quad (7.23)$$

とすればよい。この式の順序で前のガウスの消去法のアルゴリズムを適用すると、有効数字5桁の計算でも $x_1 = 1.0040$, $x_2 = 2.0024$, $x_3 = 2.9972$ となり、各変数とも正しい答と有効数字3桁で一致している。

部分ピボット選択による改良 上の2つの具体例で示したように、前進消去のアルゴリズム中で $a_{k,k}$ の値が 0 あるいは 0 に近いと、前のアルゴリズムはうまく働かない。そして、その問題点は、途中で式の順序を入れ替えることによってほぼ解決することができる。この入れ替えは以下の手続きに従って行なう。

いま、前進消去の手続きの途中で変数 x_{k-1} まで消去されたとする。このとき方程式は

$$\left\{ \begin{array}{l} a_{1,1}x_1 + a_{1,2}x_2 + \dots + a_{1,N}x_N = y_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \dots + a_{2,N}x_N = y_2 \\ \dots \\ a_{k,1}x_1 + \dots + a_{k,N}x_N = y_k \\ a_{k+1,1}x_1 + \dots + a_{k+1,N}x_N = y_{k+1} \\ \dots \\ a_{N,1}x_1 + \dots + a_{N,N}x_N = y_N \end{array} \right. \quad (7.24)$$

の形に変形されている。⑥式から⑩式までは、どのように式の順序を入れ替えてでも連立1次方程式としては同じである。そこで、これらの式のうちいちばん上にもってくる式の候補としては、式中の x_k の係数が 0 からなるべく離れているもの、すなわち、絶対値が最大のものを選択する。例えば $a_{i,k}$ ($i=k, k+1, \dots, N$, 式中の網掛け部分) のうちで絶対値が最大のものは $a_{l,k}$ であるとしよう。すると、上から k 番目の式と l 番目の式を入れ替えて、(7.24)式を

と変形するのである.

このような式の入れ替え操作を各変数の消去で毎回行なう。変数 x_k の消去を行なうための第 k 行の式の x_k の係数をピボット(pivot, かなめの意)とよぶ。そして、上のようにピボットを選ぶ操作を部分ピボット選択(partial pivoting)という。部分という語が付いているのは、このピボット選択よりも計算の精度を高めた完全ピボット選択とよばれる手続きがあるからである。ただし、本書では完全ピボット選択には触れない。

改良されたアルゴリズム 前のアルゴリズムに部分ピボット選択による改良を加えたアルゴリズムは以下のようになる。

- (1) $N, a_{i,j}, y_i$ を設定 } $Ax = y$ の問題設定

- (2) 「 $k := 1, 2, \dots, N-1$ の順に

「(部分)ピボット選択」

$|a_{i,k}|$ ($i=k, k+1, \dots, N$) のうち最大のものが $|a_{l,k}|$ であったとする。 $l \neq k$ ならば、 $a_{k,k}, a_{k,k+1}, \dots, a_{k,N}, y_k$ の値をそれぞれ $a_{l,k}, a_{l,k+1}, \dots, a_{l,N}, y_l$ の値と入れ替える

「 $i := k+1, k+2, \dots, N$ の順に

$$\alpha := a_{i,k}/a_{k,k}$$

「 $j := k+1, k+2, \dots, N$ の順に

$$a_{i,j} := a_{i,j} - \alpha a_{k,j}$$

〔を繰り返す

$$y_i := y$$

左繩

$$(3) \quad x_N := y_N/a_N$$

$\lceil i := N - 1 \rfloor$

$$x_i := \left(y_i - \sum_{k=i+1}^N a_{i,k} x_k \right) / a_{i,i}$$

を繰り返す

7-3 LU 分解

3重対角行列と LU 分解 (3.32)式, (5.73)式, (6.28)式の連立 1 次方程
式は, どれも

$$\underbrace{\begin{pmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & \ddots & \ddots & \ddots & 0 \\ 0 & & b_{N-1} & a_{N-1} & c_{N-1} \\ & & b_N & a_N & \end{pmatrix}}_{\text{係数行列 } A} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ x_N \end{pmatrix}}_{\text{ベクトル } \mathbf{x}} = \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-1} \\ y_N \end{pmatrix}}_{\text{ベクトル } \mathbf{y}} \quad (7.26)$$

という形をしている。係数行列 A は、0でない成分が対角に3重の帯の形で並んでいるので、3重対角行列と呼ばれる。前節で説明したガウスの消去法で(7.26)式を解くことはもちろん可能である。しかしながら、3重対角であるという特殊性を利用すれば、もっと効率よくこれを解くことができる。

その際、ガウスの消去法のアルゴリズムを修正することによって効率よく解くことも可能であるが、その代わりにガウスの消去法の兄弟とでもいべきLU分解の方法を適用することにする。ただし、LU分解の方法は(7.26)式のような特殊な形の連立1次方程式だけに有効なのではなく、ガウスの消去法と同様に、一般的の連立1次方程式にも適用可能であることをあらかじめおことわりしておく。

まず、行列 A を次式のように2つの行列 L と U の積で表わす。

$$\underbrace{\begin{pmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ b_3 & a_3 & c_3 & & \\ & \ddots & \ddots & \ddots & 0 \\ 0 & & b_{N-1} & a_{N-1} & c_{N-1} \\ & & b_N & a_N & \end{pmatrix}}_A = \underbrace{\begin{pmatrix} 1 & & & & \\ l_2 & 1 & & & 0 \\ l_3 & 1 & & & \\ & \ddots & \ddots & \ddots & 0 \\ 0 & l_{N-1} & 1 & & \\ & & l_N & 1 & \end{pmatrix}}_L \underbrace{\begin{pmatrix} d_1 & c_1 & & & \\ d_2 & c_2 & & & 0 \\ d_3 & c_3 & & & \\ & \ddots & \ddots & \ddots & 0 \\ 0 & & d_{N-1} & c_{N-1} & \\ & & & & d_N \end{pmatrix}}_U \quad (7.27)$$

L は対角成分より上の成分がすべて0であるので下3角行列(lower triangular matrix)

とよばれる。ただし、対角成分より下の成分も0が多いので、下3角行列のなかでも特別な形をしている。一方、 U は上3角行列(upper triangular matrix)とよばれる。

L と U が上式から一意に定まることは、実際に l_i, d_i を求める手順を書き下せば明らかである。まず、 L と U の積をとり、 A と比較すると、

$$\underbrace{\begin{pmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ b_3 & a_3 & c_3 & & \\ & \ddots & \ddots & \ddots & 0 \\ 0 & & b_{N-1} & a_{N-1} & c_{N-1} \\ & & b_N & a_N & \end{pmatrix}}_A = \underbrace{\begin{pmatrix} d_1 & c_1 & & & & & \\ l_2 d_1 & d_2 + l_2 c_1 & c_2 & & & & 0 \\ l_3 d_2 & d_3 + l_3 c_2 & c_3 & & & & \\ & \ddots & \ddots & \ddots & \ddots & & \\ 0 & & l_{N-1} d_{N-2} + l_{N-1} c_{N-2} & d_{N-1} + l_{N-1} c_{N-1} & c_{N-1} & & \\ & & l_N d_{N-1} & d_N + l_N c_{N-1} & & & \end{pmatrix}}_U \quad (7.28)$$

となる。左辺と右辺の各成分を比較すると、まず $c_1 \sim c_{N-1}$ の部分と成分が0の部分は左右辺で矛盾がないことがわかる。そして、その他の成分は

$$d_1 = a_1, \quad l_i = b_i / d_{i-1}, \quad d_i = a_i - l_i c_{i-1} \quad (i=2, 3, \dots, N) \quad (7.29)$$

を満たさなければならない。この式を用いると、 a_i, b_i, c_i の値から $d_1, l_2, d_2, l_3, d_3, \dots, l_N, d_N$ の順に値を計算することができる。これで行列 A から L と U が一意に定まった。このように A を L と U の積で表わす過程を LU分解(LU decomposition)といいう。

前進代入と後退代入 LU分解により、(7.26)式は A を LU に置き換えて

$$LUx = y \quad (7.30)$$

となる。次に、新たにベクトル $z = (z_1, z_2, \dots, z_N)^T$ を用意し、 $z = Ux$ と定義する。もちろん x がまだ求められていないので、 z もいまのところ未知である。

(7.30)式の Ux を z で置き換えると、 $Lz = y$ となる。すると、(7.30)式は、

$$Lz = y, \quad Ux = z$$

すなわち

$$\underbrace{\begin{pmatrix} 1 & & & & \\ l_2 & 1 & & & 0 \\ l_3 & & 1 & & \\ & \ddots & & \ddots & \\ 0 & & & l_{N-1} & 1 \\ & & & & l_N & 1 \end{pmatrix}}_L \underbrace{\begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{N-1} \\ z_N \end{pmatrix}}_z = \underbrace{\begin{pmatrix} z_1 \\ z_2 + l_2 z_1 \\ z_3 + l_3 z_2 \\ \vdots \\ z_{N-1} + l_{N-1} z_{N-2} \\ z_N + l_N z_{N-1} \end{pmatrix}}_y = \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{N-1} \\ y_N \end{pmatrix}}_y \quad (7.31a)$$

$$\underbrace{\begin{pmatrix} d_1 & c_1 & & & 0 \\ d_2 & c_2 & & & \\ d_3 & c_3 & & & \\ & \ddots & & \ddots & \\ 0 & & & d_{N-1} & c_{N-1} \\ & & & & d_N \end{pmatrix}}_U \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{N-1} \\ x_N \end{pmatrix}}_x = \underbrace{\begin{pmatrix} d_1 x_1 + c_1 x_2 \\ d_2 x_2 + c_2 x_3 \\ d_3 x_3 + c_3 x_4 \\ \vdots \\ d_{N-1} x_{N-1} + c_{N-1} x_N \\ d_N x_N \end{pmatrix}}_z = \underbrace{\begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{N-1} \\ z_N \end{pmatrix}}_z \quad (7.31b)$$

という連立の式に置き換わる。こうすれば、(7.31a)式を解いて z を求め、その z を用いて(7.31b)式を解いて x を求めればよい。手続きを具体的に書き下すと、まず(7.31a)式から

$$z_1 = y_1, \quad z_i = y_i - l_i z_{i-1} \quad (i=2, 3, \dots, N) \quad (7.32)$$

となるので、 z_1, z_2, \dots, z_N の順に z_i が定まる。この手続きを前進代入(forward substitution)という。そして(7.31b)式から、

$$x_i = (z_i - c_i x_{i+1})/d_i \quad (i=1, 2, \dots, N-1), \quad x_N = z_N/d_N \quad (7.33)$$

となるので、 x_N, x_{N-1}, \dots, x_1 の順に x_i が定まる。この手続きを後退代入という。以上のようにして、最終的に求めたかった x_i ($i=1, 2, \dots, N$) が求められる。

3重対角行列のLU分解の方法のアルゴリズム 上に説明したすべての手続きをアルゴリズムの形でまとめる。

(1) N, a_i, b_i, c_i, y_i を設定 } $Ax = y$ の問題設定

- | | |
|---|---------------------|
| <p>(2) $d_1 := a_1$</p> <p>$i := 2, 3, \dots, N$ の順に</p> <p style="margin-left: 2em;">$l_i := b_i/d_{i-1}$</p> <p style="margin-left: 2em;">$d_i := a_i - l_i c_{i-1}$</p> <p style="margin-left: 2em;">を繰り返す</p> | $A = LU$ と分解 (LU分解) |
| <p>(3) $z_1 := y_1$</p> <p>$i := 2, 3, \dots, N$ の順に</p> <p style="margin-left: 2em;">$z_i := y_i - l_i z_{i-1}$</p> <p style="margin-left: 2em;">を繰り返す</p> | $Lz = y$ を解く (前進代入) |
| <p>(4) $x_N := z_N/d_N$</p> <p>$i := N-1, N-2, \dots, 1$ の順に</p> <p style="margin-left: 2em;">$x_i := (z_i - c_i x_{i+1})/d_i$</p> <p style="margin-left: 2em;">を繰り返す</p> | $Ux = z$ を解く (後退代入) |

3重対角行列のLU分解の方法の計算量 このアルゴリズムで必要となる計算量を、四則演算の回数とメモリの使用量で見積もる。乗除算はLU分解の段階で $2N-2$ 回、前進代入の段階で $N-1$ 回、後退代入の段階で $2N-1$ 回必要となる。ゆえに N が十分大きいと全体で約 $5N$ 回要する。同様に加減算は約 $3N$ 回必要となる。結局、四則演算の回数は N が大きくなても $O(N)$ の回数ですむ。

また、必要となるメモリの量は、連立1次方程式の係数 a_i, b_i, c_i 、右辺 y_i 、解 x_i 、アルゴリズム中で用いる変数 d_i, l_i, z_i を記憶するだけの分である。アルゴリズムを工夫すれば、例えば変数 x_i, y_i, z_i はひとつの変数で共用できるので、メモリ量を減らすことができる。しかしどにかく、 $O(N)$ 個分のメモリだけですんでいることは確かである。

ガウスの消去法との関係 一方、ガウスの消去法をまともに(7.26)式に適用すると、加減算、乗除算ともに約 $N^3/3$ 回行なわなければならない。また、(7.26)式左辺の係数行列の、値が 0 である成分も含めて合計 N^2 個分の成分を記憶するためのメモリが少なくとも必要になる。四則演算の回数およびメモリの使用量のどちらをとっても LU 分解の方法の方が優れていることになる。と

ころが、じつは LU 分解の方法とガウスの消去法は本質的に同じ計算を行なっている。計算量に大きな差が生じたのは、ガウスの消去法をまともに適用するせいである。これでは、係数行列のほとんどを占めている 0 の成分を記憶してそれらに演算を行なってしまう。0 を何倍しても 0 であり、0 を足したり引いたりしても結果は変わらない。そこで、ガウスの消去法でも 0 の成分を記憶せず、それらの成分が関わる計算の部分をあらかじめアルゴリズムから除外しておけば、計算量は同じになる。

では、LU 分解の方法でガウスの消去法と同じ計算が行なわれていることを示す。連立1次方程式 $Ax = y$ は、LU 分解、前進代入の手続きによって $Ux = z$ という形に変形される。この変形された方程式は、ガウスの消去法で前進消去を行なって得られた方程式と同じである。例えば $N=4$ の場合の連立1次方程式(7.26)式は、

$$\begin{aligned} a_1x_1 + c_1x_2 &= y_1 \\ b_2x_1 + a_2x_2 + c_2x_3 &= y_2 \\ b_3x_2 + a_3x_3 + c_3x_4 &= y_3 \\ b_4x_3 + a_4x_4 &= y_4 \end{aligned} \quad (7.34)$$

である。この式にガウスの消去法の前進消去の操作を施すと、前節で用いた記号と異なるが、

$$\begin{aligned} d_1x_1 + c_1x_2 &= z_1 \\ d_2x_2 + c_2x_3 &= z_2 \\ d_3x_3 + c_3x_4 &= z_3 \\ d_4x_4 &= z_4 \end{aligned} \quad (7.35)$$

と $Ux = z$ の形に変形されることがわかる。さらに、LU 分解の方法では、後退代入によりこの式を下の方から解いて x_4, x_3, x_2, x_1 の順に値を求める。この操作はガウスの消去法の後退代入の操作と同じである。

なお、ガウスの消去法のピボット選択に相当する手続きを、LU 分解の方法で行なわなければならない場合がある。しかし、係数行列が3重対角の連立1次方程式が現実の問題で現われた場合は、ピボット選択が不要であることが多い。そこで、上の議論でもピボット選択については考慮していない。

一般的係数行列の場合 係数行列 A の形が3重対角以外の場合でも LU 分解を適用できる。例えば、行列 A の成分のほとんどが 0 でない場合、行列 A を

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,N} \\ \cdots & \cdots & \cdots & \cdots \\ a_{N,1} & a_{N,2} & \cdots & a_{N,N} \end{pmatrix}$$

A

$$= \begin{pmatrix} 1 & & & & & u_{1,1} & u_{1,2} & u_{1,3} & \cdots & u_{1,N} \\ l_{2,1} & 1 & & & & u_{2,2} & u_{2,3} & \cdots & u_{2,N} \\ l_{3,1} & l_{3,2} & 1 & & & \cdots & \cdots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ l_{N,1} & l_{N,2} & \cdots & l_{N,N-1} & 1 & 0 & u_{N-1,N-1} & u_{N-1,N} & u_{N,N} \end{pmatrix} \begin{pmatrix} & & & & & 0 & & & & \\ & & & & & & 0 & & & \\ & & & & & & & 0 & & \\ & & & & & & & & \ddots & \\ & & & & & & & & & \ddots \end{pmatrix} U$$

(7.36)

という具合に下3角行列 L と上3角行列 U の積に分解する。そして、 $Lz = y$, $Ux = z$ を順に解けばよい。この場合もガウスの消去法と同じ操作を見方を変えて行なっているだけである。 A がこのように一般の行列の場合の、LU 分解の方法のアルゴリズムを以下に示す。

(1) $N, a_{i,j}, y_i$ を設定 $\} Ax = y$ の問題設定(2) $j := 1, 2, \dots, N$ の順に $i := 1, 2, \dots, j$ の順に

$$u_{i,j} := a_{i,j} - \sum_{k=1}^{i-1} l_{i,k} u_{k,j}$$

を繰り返す

 $i := j+1, j+2, \dots, N$ の順に

$$l_{i,j} := \frac{1}{u_{j,j}} \left(a_{i,j} - \sum_{k=1}^{j-1} l_{i,k} u_{k,j} \right)$$

を繰り返す

を繰り返す

LU 分解

$$(3) \quad z_1 := y_1$$

$$\left[\begin{array}{l} i := 2, 3, \dots, N \text{ の順に} \\ z_i := y_i - \sum_{k=1}^{i-1} l_{i,k} z_k \\ \text{を繰り返す} \end{array} \right] \quad \text{前進代入}$$

$$(4) \quad x_N := z_N / u_{N,N}$$

$$\left[\begin{array}{l} i := N-1, N-2, \dots, 1 \text{ の順に} \\ x_i := \left(z_i - \sum_{k=i+1}^N u_{i,k} x_k \right) / u_{i,i} \\ \text{を繰り返す} \end{array} \right] \quad \text{後退代入}$$

ただし、ガウスの消去法と同様にピボット選択が一般に必要であるが、ここでは省略する。また、アルゴリズムで必要となる四則演算の回数は、当然ガウスの消去法と同じである。

LU分解の方法の長所 最後に、LU分解の方法がガウスの消去法より優れている点をひとつ挙げる。係数行列が一般的の形の場合にLU分解の手続きに要する四則演算の回数は $O(N^3)$ である。それに比べて前進代入・後退代入の手続きでは $O(N^2)$ ですむ。

そこで、元の連立1次方程式 $Ax = y$ の係数行列 A を変えずに、いろいろな y に対して解 x を次つぎに求める場合を考える。LU分解の方法では最初に1回だけ $A = LU$ と分解しておけば、 L と U は変わらないので後は $O(N^2)$ の回数の四則演算で解 x を次つぎと計算できる。これに対しガウスの消去法では、前進消去の手続きで y も同時に変形しなければならない。ゆえに、係数行列 A が変わらなくても異なる y を与えた場合には毎回前進消去から始めなければならず、つねに $O(N^3)$ の回数の四則演算を必要とする。この点でLU分解の方法はガウスの消去法より優れており、この長所が実際の数値計算で役立つことが多い。

7-4 SOR法

この節では連立1次方程式を解くためのもう1つの解法、SOR法(succesive overrelaxation method、逐次過緩和法)について説明する。この解法は今までに説明したガウスの消去法、LU分解の方法とは異なる特徴をもっている。この特徴のために、連立1次方程式の形によってはSOR法の方が優れている場合がある。

ヤコビ法 まず、2元連立1次方程式

$$\begin{cases} 5x_1 + 4x_2 = 13 \\ 2x_1 + 3x_2 = 8 \end{cases} \quad (7.37)$$

という具体的な問題にSOR法を適用し、そのアルゴリズムを示す。ただし、SOR法の基礎となる解法が2つあるのでそれらを先に説明する。なお、(7.37)式の解は $(x_1, x_2) = (1, 2)$ である。

(7.37)式を変形すると、

$$\begin{cases} x_1 = \frac{1}{5}(13 - 4x_2) = f_1(x_2) \\ x_2 = \frac{1}{3}(8 - 2x_1) = f_2(x_1) \end{cases} \quad (7.38)$$

を得る。新たな変数 $x_1^{(k)}, x_2^{(k)}$ ($k = 0, 1, \dots$) を用意し、(7.38)式をもとに $x_1^{(k)}, x_2^{(k)}$ に関する以下の漸化式を作る。

$$\begin{cases} x_1^{(k+1)} = \frac{1}{5}(13 - 4x_2^{(k)}) = f_1(x_2^{(k)}) \\ x_2^{(k+1)} = \frac{1}{3}(8 - 2x_1^{(k)}) = f_2(x_1^{(k)}) \end{cases} \quad (7.39)$$

初期値 $x_1^{(0)}, x_2^{(0)}$ を適当に定めると、(7.39)式に $k=0$ を代入して $x_1^{(1)}, x_2^{(1)}$ を計算できる。さらに、その値から(7.39)式に $k=1$ を代入して $x_1^{(2)}, x_2^{(2)}$ を計算できる。以下同様にして、 $x_1^{(k)}, x_2^{(k)}$ の値を k の小さいものから順に次つぎと求めることができる。いま仮に $k \rightarrow \infty$ で $x_1^{(k)}, x_2^{(k)}$ の値がそれぞれ α_1, α_2 に収束

したとしよう。すると $(x_1, x_2) = (\alpha_1, \alpha_2)$ は(7.38)式、すなわち、(7.37)式を満たすので解である。また、 k を大きくしていけば $x_1^{(k)}, x_2^{(k)}$ は収束値に近づいていくので、十分大きな k に対する $x_1^{(k)}, x_2^{(k)}$ は解の近似値となり得る。

ためしに $x_1^{(0)} = x_2^{(0)} = 0$ として、(7.39)式から $x_1^{(k)}, x_2^{(k)}$ を計算した結果を表 7-1 に示す。この表から、 $k=41$ のときに真の解とほぼ 6 行一致していることがわかる。以上のように、適当な初期値から出発し、(7.39)式のような漸化式による反復計算により解の近似値を求め

る方法を反復法(iterative method)という。これに対し、ガウスの消去法、LU 分解の方法などは直接法(direct method)とよばれる。また、上で説明した反復法は特にヤコビ法(Jacobi's method)とよばれる。

解への収束 それでは、任意の初期値から出発して必ず解に収束するのであろうか。そのことを考えるために、(7.39)式をベクトル $(x_1^{(k)}, x_2^{(k)})^T$ に関する以下の漸化式の形に書き換える。

$$\begin{pmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & -4/5 \\ -2/3 & 0 \end{pmatrix}}_{\text{行列 } M} \begin{pmatrix} x_1^{(k)} \\ x_2^{(k)} \end{pmatrix} + \begin{pmatrix} 13/5 \\ 8/3 \end{pmatrix} \quad (7.40)$$

解 $(x_1, x_2) = (1, 2)$ は

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} = M \begin{pmatrix} 1 \\ 2 \end{pmatrix} + \begin{pmatrix} 13/5 \\ 8/3 \end{pmatrix} \quad (7.41)$$

を満足するので、これら 2 つの式の差をとって

$$\begin{pmatrix} x_1^{(k+1)} - 1 \\ x_2^{(k+1)} - 2 \end{pmatrix} = M \begin{pmatrix} x_1^{(k)} - 1 \\ x_2^{(k)} - 2 \end{pmatrix} \quad (7.42)$$

を導くことができる。この式の左辺、右辺のベクトルは、それぞれ $k+1, k$ 回

表 7-1 ヤコビ法による結果

k	$x_1^{(k)}$	$x_2^{(k)}$
0	0	0
1	2.60000000	2.66666667
2	0.46666667	0.93333333
3	1.85333333	2.35555556
4	0.71555556	1.43111111
5	1.45511111	2.18962963
6	0.84829630	1.69659259
7	1.24272593	2.10113580
8	0.91909136	1.83818272
9	1.12945383	2.05393909
:	:	:
38	0.99999350	1.99998700
39	1.00001040	2.00000433
40	0.99999653	1.99999307
41	1.00000555	2.00000231
真の解 $(x_1, x_2) = (1, 2)$		

目の反復値と真の解との誤差である。したがって、この式は反復値の誤差の漸化式になっている。さらに、

$$\begin{pmatrix} x_1^{(k)} - 1 \\ x_2^{(k)} - 2 \end{pmatrix} = M^k \begin{pmatrix} x_1^{(0)} - 1 \\ x_2^{(0)} - 2 \end{pmatrix} \quad (7.43)$$

となる。

次に行列 M の固有値、固有ベクトルを調べてみよう。 M の固有方程式

$$\det \begin{pmatrix} -\lambda & -4/5 \\ -2/3 & -\lambda \end{pmatrix} = 0 \quad (7.44)$$

を解くと、2つの固有値 $\lambda_1 = \sqrt{\frac{8}{15}}$, $\lambda_2 = -\sqrt{\frac{8}{15}}$ が導かれ、対応する固有ベクトルはそれぞれ $e_1 = (\sqrt{6}, -\sqrt{5})^T$, $e_2 = (\sqrt{6}, \sqrt{5})^T$ となる。すると、(7.43)式の右辺のベクトルは、任意の $x_1^{(0)}, x_2^{(0)}$ に対して

$$\begin{pmatrix} x_1^{(0)} - 1 \\ x_2^{(0)} - 2 \end{pmatrix} = c_1 e_1 + c_2 e_2 \quad (7.45)$$

というようにベクトル e_1 と e_2 の重ね合わせで表現できる。これから(7.43)式は

$$\begin{aligned} \begin{pmatrix} x_1^{(k)} - 1 \\ x_2^{(k)} - 2 \end{pmatrix} &= M^k (c_1 e_1 + c_2 e_2) \\ &= c_1 \lambda_1^k e_1 + c_2 \lambda_2^k e_2 \end{aligned} \quad (7.46)$$

となる。さらに、 $|\lambda_1| < 1$, $|\lambda_2| < 1$ ので

$$\lim_{k \rightarrow \infty} \begin{pmatrix} x_1^{(k)} - 1 \\ x_2^{(k)} - 2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (7.47)$$

となる。よって任意の初期値に対して $x_1^{(k)}, x_2^{(k)}$ が解に収束する。収束するためのポイントは、行列 M の固有値の絶対値がすべて 1 より小さいことである。

ガウス-ザイデル法 次にガウス-ザイデル法(Gauss-Seidel method)を説明する。ガウス-ザイデル法では、(7.39)式をすこし修正して

$$\begin{cases} x_1^{(k+1)} = \frac{1}{5}(13 - 4x_2^{(k)}) = f_1(x_2^{(k)}) \\ x_2^{(k+1)} = \frac{1}{3}(8 - 2x_1^{(k+1)}) = f_2(x_1^{(k+1)}) \end{cases} \quad (7.48)$$

とする。 (7.39)式の第 2 式の右辺の $x_1^{(k)}$ が (7.48)式では $x_1^{(k+1)}$ に置き換わっている。この修正の意図を言葉で表わすなら、「第 1 式で $x_1^{(k)}$ よりも解に近いであろう $x_1^{(k+1)}$ を得るのだから、第 2 式で直ちにそれを使う」といったところである。この修正とともに、反復計算の順序に制限が加わる。まず、初期値 $x_2^{(0)}$ を与える。すると (7.48)式の第 1 式から $x_1^{(1)}$ が計算され、次に第 2 式から $x_2^{(1)}$ が計算される。この手続きを繰り返して、 $x_2^{(0)} \rightarrow x_1^{(1)} \rightarrow x_2^{(1)} \rightarrow x_1^{(2)} \rightarrow x_2^{(2)} \rightarrow \dots$ の順に $x_1^{(k)}, x_2^{(k)}$ が計算される。また、 $x_1^{(0)}$ は必要ない。

$x_2^{(0)} = 0$ を初期値にして計算した結果を表 7-2 に示す。 $k=21$ で真の解とほぼ 6 行一致している。ヤコビ法では $k=41$ のときに同等の結果が得られたので、この例ではガウス-ザイデル法がヤコビ法より解への収束がほぼ 2 倍速い、すなわち、同じ精度の結果を得るために計算の時間が半分ですむ。

じつは、2 元連立 1 次方程式の場合、これは当たり前である。その理由を以下に示す。まず、ガウス-ザイデル法では $x_1^{(k+1)} = f_1(x_2^{(k)}) = f_1(f_2(x_1^{(k)}))$ となる。

一方、ヤコビ法では $x_1^{(k+2)} = f_1(x_2^{(k+1)}) = f_1(f_2(x_1^{(k)}))$ となる。同じ $x_1^{(k)}$ の値に対してガウス-ザイデル法の $x_1^{(k+1)}$ とヤコビ法の $x_1^{(k+2)}$ の値が等しい。 $x_2^{(k)}$ についても同様である。つまり、表 7-1 の $x_1^{(k)}, x_2^{(k)}$ の 1 つおきの数値が表 7-2 の $x_1^{(k)}, x_2^{(k)}$ になっている。したがってガウス-ザイデル法の方が 2 倍速い。

SOR 法 それでは、いよいよ SOR 法について説明する。あるパラメータ ω を用いて (7.48)式を次式のように修正する。

$$\begin{cases} x_1^{(k+1)} = \frac{\omega}{5}(13 - 4x_2^{(k)}) + (1-\omega)x_1^{(k)} = \omega f_1(x_2^{(k)}) + (1-\omega)x_1^{(k)} \\ x_2^{(k+1)} = \frac{\omega}{3}(8 - 2x_1^{(k+1)}) + (1-\omega)x_2^{(k)} = \omega f_2(x_1^{(k+1)}) + (1-\omega)x_2^{(k)} \end{cases} \quad (7.49)$$

ω および $x_1^{(0)}, x_2^{(0)}$ を定めれば、(7.49)式から $x_1^{(1)} \rightarrow x_2^{(1)} \rightarrow x_1^{(2)} \rightarrow x_2^{(2)} \rightarrow \dots$ の順に $x_1^{(k)}, x_2^{(k)}$ が定まる。 $k \rightarrow \infty$ で $x_1^{(k)}, x_2^{(k)}$ がそれぞれある値に収束するならば、それが解であることは明らかである。また、 $\omega=1$ のとき (7.49)式はガウス-ザイデル法に一致する。なお、パラメータ ω は加速係数 (acceleration parameter) とよばれる。

SOR 法の意図を理解するため (7.49)式を

$$\begin{cases} x_1^{(k+1)} - x_1^{(k)} = \omega \{f_1(x_2^{(k)}) - x_1^{(k)}\} \\ x_2^{(k+1)} - x_2^{(k)} = \omega \{f_2(x_1^{(k+1)}) - x_2^{(k)}\} \end{cases} \quad (7.50)$$

と書き直す。この式は $\omega=1$ の場合にガウス-ザイデル法に帰着して、次式のようになる。

$$\begin{cases} x_1^{(k+1)} - x_1^{(k)} = f_1(x_2^{(k)}) - x_1^{(k)} \\ x_2^{(k+1)} - x_2^{(k)} = f_2(x_1^{(k+1)}) - x_2^{(k)} \end{cases} \quad (7.51)$$

(7.51)式の左辺は、 x_1, x_2 の k 回目の反復値から $k+1$ 回目の反復値への変化量を意味し、右辺によってその変化量が計算される。ということは、(7.50)式ではガウス-ザイデル法の流儀で計算される変化量 ((7.50)式の下線部) に ω という数を掛けることによって、変化を「加速」しているのである。ただし、(7.50)式と (7.51)式は $\omega \neq 1$ ならばもはや異なる漸化式であるので、(7.50)式の下線部と (7.51)式の右辺は実際の計算では同じ値にならない。ここでの説明はあくまで SOR 法の意図を感覚的に表わしたものと思っていただきたい。

では、初期値を $x_1^{(0)} = x_2^{(0)} = 0$ とし、 $\omega=1.2$ および 1.5 としたときの計算結果を表 7-3 に示す。 $\omega=1.2$ の場合は $k=10$ 付近で真の解と 6 行一致している。この収束の速さはガウス-ザイデル法の約 2 倍である。 $\omega=1.5$ の場合は、収束の速さはガウス-ザイデル法とほとんど変わらない。

この例でわかるように、パラメータ ω の値によって収束の速さが変わり、収束の速さを最大にする最適の ω の値が存在するのである。いまの連立 1 次方程式の場合は、計算を省くが $\omega=1.18\dots$ であり、 $\omega=1.2$ はこれに近いので収束が速かったのである。

一般の連立 1 次方程式の場合 具体例に沿った説明はここで打ち切り、次式の一般の連立 1 次方程式について各解法を順に説明する。

表 7-3 SOR 法による結果

k	(a) $\omega=1.2$ の場合		(b) $\omega=1.5$ の場合	
	$x_1^{(k)}$	$x_2^{(k)}$	$x_1^{(k)}$	$x_2^{(k)}$
0	0	0	0	0
1	3.12000000	0.70400000	1	3.90000000
2	1.82016000	1.60307200	2	1.83000000
3	1.21701888	1.90577050	3	0.44100000
4	1.04705655	1.98120066	4	0.68070000
5	1.00863605	1.99685102	5	1.07589000
6	1.00129581	1.99959315	6	1.09500300
7	1.00013141	1.99997624	7	1.00002810
8	0.99999653	2.00000753	8	0.97625487
9	0.99999347	2.00000372	9	0.99524395
10	0.99999773	2.00000107	⋮	⋮
11	0.99999943	2.00000024	18	0.99998578
12	0.99999988	2.00000005	19	1.00000785
13	0.99999998	2.00000001	20	1.00000513

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,N}x_N = y_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,N}x_N = y_2 \\ \cdots \\ a_{N,1}x_1 + a_{N,2}x_2 + \cdots + a_{N,N}x_N = y_N \end{cases} \quad (7.52)$$

まず、(7.52)式を変形して

$$\begin{cases} x_1 = \frac{1}{a_{1,1}}\{y_1 - (a_{1,2}x_2 + a_{1,3}x_3 + \cdots + a_{1,N}x_N)\} \\ x_2 = \frac{1}{a_{2,2}}\{y_2 - (a_{2,1}x_1 + a_{2,3}x_3 + \cdots + a_{2,N}x_N)\} \\ \cdots \\ x_i = \frac{1}{a_{i,i}}\{y_i - (a_{i,1}x_1 + a_{i,2}x_2 + \cdots + a_{i,i-1}x_{i-1} + a_{i,i+1}x_{i+1} + \cdots + a_{i,N}x_N)\} \\ \cdots \\ x_N = \frac{1}{a_{N,N}}\{y_N - (a_{N,1}x_1 + a_{N,2}x_2 + \cdots + a_{N,N-1}x_{N-1})\} \end{cases} \quad (7.53)$$

とする。この式を利用して次の漸化式を作る。

$$\begin{cases} x_1^{(k+1)} = \frac{1}{a_{1,1}}\{y_1 - (a_{1,2}x_2^{(k)} + a_{1,3}x_3^{(k)} + \cdots + a_{1,N}x_N^{(k)})\} \\ x_2^{(k+1)} = \frac{1}{a_{2,2}}\{y_2 - (a_{2,1}x_1^{(k)} + a_{2,3}x_3^{(k)} + \cdots + a_{2,N}x_N^{(k)})\} \\ \cdots \\ x_i^{(k+1)} = \frac{1}{a_{i,i}}\{y_i - (a_{i,1}x_1^{(k)} + a_{i,2}x_2^{(k)} + \cdots + a_{i,i-1}x_{i-1}^{(k)} \\ + a_{i,i+1}x_{i+1}^{(k)} + \cdots + a_{i,N}x_N^{(k)})\} \\ \cdots \\ x_N^{(k+1)} = \frac{1}{a_{N,N}}\{y_N - (a_{N,1}x_1^{(k)} + a_{N,2}x_2^{(k)} + a_{N,3}x_3^{(k)} + \cdots + a_{N,N-1}x_{N-1}^{(k)})\} \end{cases} \quad (7.54)$$

これがヤコビ法の漸化式である。いま、ベクトル $\mathbf{x}^{(k)}$ を $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, \dots, x_N^{(k)})^T$ と定義しよう。すると、適当な初期値 $\mathbf{x}^{(0)}$ から出発して(7.54)式を用いて $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots$ を次つぎと計算できる。 $k \rightarrow \infty$ で $\mathbf{x}^{(k)}$ がある定ベクトル $\mathbf{x}^{(\infty)}$ に収束すれば、それが解であることは明らかである。そこで、この節の後で述べるような収束判定条件で計算を途中で打ち切り、そのときの $\mathbf{x}^{(k)}$ の値を解の近似値とするのである。初期値 $\mathbf{x}^{(0)}$ は、特に候補がなければ $\mathbf{x}^{(0)} = (0, 0, \dots, 0)^T$ などのように適当に設定する。

ガウス-ザイデル法の漸化式は、(7.54)式をさらに修正して、

$$\begin{cases} x_1^{(k+1)} = \frac{1}{a_{1,1}}\{y_1 - (a_{1,2}x_2^{(k)} + a_{1,3}x_3^{(k)} + a_{1,4}x_4^{(k)} + \cdots + a_{1,N}x_N^{(k)})\} \\ x_2^{(k+1)} = \frac{1}{a_{2,2}}\{y_2 - (a_{2,1}x_1^{(k+1)} + a_{2,3}x_3^{(k)} + a_{2,4}x_4^{(k)} + \cdots + a_{2,N}x_N^{(k)})\} \\ x_3^{(k+1)} = \frac{1}{a_{3,3}}\{y_3 - (a_{3,1}x_1^{(k+1)} + a_{3,2}x_2^{(k+1)} + a_{3,4}x_4^{(k)} + \cdots + a_{3,N}x_N^{(k)})\} \\ \cdots \\ x_i^{(k+1)} = \frac{1}{a_{i,i}}\{y_i - (a_{i,1}x_1^{(k+1)} + a_{i,2}x_2^{(k+1)} + a_{i,3}x_3^{(k+1)} + \cdots + a_{i,i-1}x_{i-1}^{(k+1)} \\ + a_{i,i+1}x_{i+1}^{(k)} + \cdots + a_{i,N}x_N^{(k)})\} \\ \cdots \\ x_N^{(k+1)} = \frac{1}{a_{N,N}}\{y_N - (a_{N,1}x_1^{(k+1)} + a_{N,2}x_2^{(k+1)} + a_{N,3}x_3^{(k+1)} + \cdots + a_{N,N-1}x_{N-1}^{(k+1)})\} \end{cases} \quad (7.55)$$

となる。網掛けで示した部分がヤコビ法と異なる部分である。1番目の式で $x_1^{(k+1)}$ を計算したら直ちに2番目の式の $x_2^{(k+1)}$ の計算にその値を使用し、さらに3番目の式で $x_3^{(k+1)}$ の計算に $x_1^{(k+1)}, x_2^{(k+1)}$ の値を使用する。以下同様にして i 番目の式で $x_i^{(k+1)}$ を得れば、 $i+1$ 番目以降の式では $x_i^{(k)}$ の代わりに $x_i^{(k+1)}$ を右辺で使用する。このため漸化式は上から順に計算しなければならない。こうしてガウス-ザイデル法では、「新たな近似値を得れば直ちにそれを使用する」点がヤコビ法と異なる。

次に、SOR法の漸化式は(7.55)式をさらに修正して

$$\begin{aligned} x_1^{(k+1)} &= \frac{\omega}{a_{1,1}} \{y_1 - (a_{1,2}x_2^{(k)} + a_{1,3}x_3^{(k)} + a_{1,4}x_4^{(k)} + \dots + a_{1,N}x_N^{(k)})\} + (1-\omega)x_1^{(k)} \\ x_2^{(k+1)} &= \frac{\omega}{a_{2,2}} \{y_2 - (a_{2,1}x_1^{(k+1)} + a_{2,3}x_3^{(k)} + a_{2,4}x_4^{(k)} + \dots + a_{2,N}x_N^{(k)})\} + (1-\omega)x_2^{(k)} \\ x_3^{(k+1)} &= \frac{\omega}{a_{3,3}} \{y_3 - (a_{3,1}x_1^{(k+1)} + a_{3,2}x_2^{(k+1)} + a_{3,4}x_4^{(k)} + \dots + a_{3,N}x_N^{(k)})\} + (1-\omega)x_3^{(k)} \\ &\dots \\ x_i^{(k+1)} &= \frac{\omega}{a_{i,i}} \{y_i - (a_{i,1}x_1^{(k+1)} + a_{i,2}x_2^{(k+1)} + a_{i,3}x_3^{(k+1)} + \dots + a_{i,i-1}x_{i-1}^{(k+1)} \\ &\quad + a_{i,i+1}x_{i+1}^{(k)} + \dots + a_{i,N}x_N^{(k)})\} + (1-\omega)x_i^{(k)} \\ &\dots \\ x_N^{(k+1)} &= \frac{\omega}{a_{N,N}} \{y_N - (a_{N,1}x_1^{(k+1)} + a_{N,2}x_2^{(k+1)} + a_{N,3}x_3^{(k+1)} + \dots + a_{N,N-1}x_{N-1}^{(k+1)})\} \\ &\quad + (1-\omega)x_N^{(k)} \end{aligned} \quad (7.56)$$

とする。ここで、 ω はすべての式に共通な定数である。 $\omega=1$ で SOR 法はガウス-ザイデル法に帰着する。前の2元連立1次方程式の例で述べたように、SOR 法は「ガウス-ザイデル法の流儀による $x^{(k+1)}$ の予測値を ω を用いて加速する」方法であるといえる。

なお、SOR 法で $x^{(k)}$ が解に収束するためには、少なくとも $0 < \omega < 2$ でなければならないことがわかっている。また、連立1次方程式がある条件を満たすとき、収束の速さを最大にする最適の ω が $1 < \omega < 2$ の範囲に存在し、このときヤコビ法、ガウス-ザイデル法よりも速く収束する。

収束判定条件 ヤコビ法、ガウス-ザイデル法、SOR 法の漸化式を列挙したが、どれも反復法であり、ある条件を満たせば反復計算を途中で打ち切るという形にしておかなければ計算が終了しない。計算を打ち切るための条件としては、 $x^{(k)}$ の値が真の解に十分近づいたことを判定する収束判定条件を用いる。収束の判定は難しい問題であり、いろいろな条件が工夫されている。ここでは以下の条件を採用することにする。

ϵ を、ある小さな正数であるとする。そして $x^{(k)}$ に関する漸化式を $k = 0, 1, \dots$ の順に計算していく、

$$\sum_{i=1}^N |x_i^{(k+1)} - x_i^{(k)}| < \epsilon \sum_{i=1}^N |x_i^{(k+1)}| \quad (7.57)$$

となれば計算を打ち切って $x^{(k+1)}$ を答とする。

この条件の意味は以下のとおりである。まず、上式の両辺を N で割って

$$\frac{1}{N} \sum_{i=1}^N |x_i^{(k+1)} - x_i^{(k)}| < \epsilon \frac{1}{N} \sum_{i=1}^N |x_i^{(k+1)}| \quad (7.58)$$

とすると理解しやすい。左辺は $x^{(k+1)}$ と $x^{(k)}$ の各成分の差の絶対値を平均したものである。また、右辺の $\frac{1}{N} \sum_{i=1}^N |x_i^{(k+1)}|$ は $x^{(k+1)}$ の各成分の絶対値の平均値である。すなわち収束判定条件は、「成分で考えて平均すると、 $x^{(k+1)}$ と $x^{(k)}$ の差が、 $x^{(k+1)}$ の大きさの ϵ 倍未満に小さくなる」という条件なのである。なお、他の収束判定条件として、例えば次のようなものがある。

$$\max_{1 \leq i \leq N} |x_i^{(k+1)} - x_i^{(k)}| < \epsilon \times \max_{1 \leq i \leq N} |x_i^{(k+1)}| \quad (7.59)$$

ヤコビ法のアルゴリズム ここで、各解法について初期値設定から収束判定条件による計算終了までの手続きをアルゴリズムにまとめる。まず、ヤコビ法では(7.54)式を用いて $x^{(k)}$ から $x^{(k+1)}$ を計算してしまうと、次の $x^{(k+2)}$ の計算には $x^{(k)}$ が不要になる。そこで、変数を記憶するメモリを節約した以下のようなアルゴリズムが考えられる。

(1) $N, a_{i,j}, y_i$ が与えられる(問題設定)

初期値 $x = (x_1, x_2, \dots, x_N)^T$ を設定する

ϵ を設定する

(2)

以下

$$\text{sum} := 0, \quad \text{error} := 0$$

 $i := 1, 2, \dots, N$ の順に

$$z_i := \frac{1}{a_{i,i}} \left(y_i - \sum_{j=1}^{i-1} a_{i,j} x_j - \sum_{j=i+1}^N a_{i,j} x_j \right)$$

$$\text{sum} := \text{sum} + |z_i|$$

$$\text{error} := \text{error} + |z_i - x_i|$$

を繰り返す

もし $\text{error} < \epsilon \cdot \text{sum}$ ならばステップ(3)に移る $i := 1, 2, \dots, N$ の順に

$$x_i := z_i$$

を繰り返す

を繰り返す

(3) $z = (z_1, z_2, \dots, z_N)^T$ を答とする。

上のアルゴリズムの x, z はそれぞれ $x^{(k)}, x^{(k+1)}$ の役割を果たしている。また、変数 sum, error はそれぞれ $\sum_{i=1}^N |x_i^{(k+1)}|$, $\sum_{i=1}^N |x_i^{(k+1)} - x_i^{(k)}|$ を計算するための変数である。

SOR 法のアルゴリズム 次に SOR 法のアルゴリズムを示す。ガウス-ザイデル法は SOR 法で $\omega=1$ の場合に相当するので、ここでは省略する。SOR 法では(7.56)式の上から i 番目の式までは $x_i^{(k)}$ が登場し、 i 番目の式で $x_i^{(k+1)}$ を計算した後は二度と $x_i^{(k)}$ が現われない。

そこで、ヤコビ法よりもさらにメモリを節約した以下のアルゴリズムを考えられる。

(1) $N, a_{i,j}, y_i$ が与えられる(問題設定)初期値 $x = (x_1, x_2, \dots, x_N)^T$ を設定する ϵ, ω を設定する

(2)

以下

$$\text{sum} := 0, \quad \text{error} := 0$$

 $i := 1, 2, \dots, N$ の順に

$$\text{new_}x := \frac{\omega}{a_{i,i}} \left(y_i - \sum_{j=1}^{i-1} a_{i,j} x_j - \sum_{j=i+1}^N a_{i,j} x_j \right) + (1-\omega)x_i$$

$$\text{sum} := \text{sum} + |\text{new_}x|$$

$$\text{error} := \text{error} + |\text{new_}x - x_i|$$

$$x_i := \text{new_}x$$

を繰り返す

もし $\text{error} < \epsilon \cdot \text{sum}$ ならばステップ(3)に移る

を繰り返す

(3) x を答とする。

このアルゴリズムでは、 $x^{(k)}$ のための変数として $x = (x_1, x_2, \dots, x_N)^T$ と補助的な変数 $\text{new_}x$ だけですんでいる。

各解法の特徴 最後に、補足事項をいくつか述べる。ただし、数学的に厳密な議論は複雑で、本書の範囲を超える。詳しい説明は巻末の参考書を参照していただきたい。

解への収束 すべての解法で漸化式を $x^{(k+1)} = Mx^{(k)} + c$ の形に表わすことができる。ここで M は N 次の正方行列で c は定ベクトルである。任意の初期値 $x^{(0)}$ から出発して $k \rightarrow \infty$ で $x^{(k)}$ が解のベクトルに収束するためには、 M のすべての固有値の絶対値が 1 より小さいことが必要十分条件となる。ところが、連立 1 次方程式の規模が大きくなると固有値を調べること自体が難しい。そこで、元の連立 1 次方程式 $Ax = y$ の係数行列 A がどのような形であれば収束するかという十分条件がいろいろ調べられている。

しかしながら、収束しない例も簡単に作ることができる。ここでは、とりあえず反復法を試してみて、うまく行かなければ考え直すという程度にとどめておく。

ω の決定 SOR 法の ω を最適の値に選ぶと、 $\omega=1$ のガウス-ザイデル法と比べて解への収束が数倍から数十倍速くなることがよくある。しかし、与えら

れた連立1次方程式に対して最適の ω を計算することはなかなか難しい。連立1次方程式の係数行列がある条件を満たすときに、最適である ω を計算する方法がいくつか存在する程度である。もし、ある連立1次方程式を1回解くだけで、しかも最適の ω の値を全く予想できないならば、とりあえずガウス-ザイデル法を使用することをおすすめする。

一方、実際の問題では同じ係数行列の連立1次方程式を右辺の値を変えながら何回も解くことがある。この場合はガウス-ザイデル法の2倍の速さになるだけでもメリットが出てくる。そこで、予備的な問題で ω の値をいろいろ変えてみて実験し、なるべく最適の ω に近づけておくという単純な方法も有効となる。

直接法 vs. 反復法 どちらの系統の方法がよいとは一概にいえない。連立1次方程式の係数行列の形、問題の規模、解に求められる精度、使用できるメモリ量、使用する計算機の種類などに応じて解法を選択する必要がある。そこで本書で触れた解法に限って直接法と反復法のそれぞれの長所をまとめる。

まず、直接法では有限回の手続きで必ず解を得ることができる。原理的にはあらゆる連立1次方程式に適用可能である。

一方、反復法では解へ収束することが前提になるので、適用できる連立1次方程式に制限のあるのが難点であるが、反復法向けの問題ではメモリ使用量が非常に少なくてすむ。例えば(6.53)式にSOR法を適用すると、必要なメモリは右辺のベクトルと解の反復値のベクトルを記憶する分だけではぼすむ(第7章演習問題[5])。

なお、本書では触れることができなかったが、不完全コレスキー分解と共に傾斜法を組み合わせたICCG法とよばれる有力な解法が存在する。この解法は大規模な問題を高速に解く方法として知られている。機会があれば巻末の参考書などで勉強されることをおすすめする。

第7章 演習問題

[1] ガウスの消去法を用いて以下の連立1次方程式を解け。

$$(1) \begin{pmatrix} 1 & -2 & 3 \\ 2 & 1 & 0 \\ 1 & 2 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 5 \\ 5 \end{pmatrix}$$

$$(2) \begin{pmatrix} 2 & -2 & 1 \\ 1 & -1 & 2 \\ -1 & 3 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -9 \\ -3 \\ 12 \end{pmatrix}$$

$$(3) \begin{pmatrix} 1 & -2 & 3 & 1 & 2 \\ 2 & 1 & -1 & 2 & 4 \\ 3 & -1 & -2 & 1 & -1 \\ 1 & 3 & 1 & -4 & -2 \\ 4 & -2 & 1 & 3 & -3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 13 \\ 7 \\ -5 \\ 2 \\ -7 \end{pmatrix}$$

[2] LU分解の方法を用いて以下の連立1次方程式を解け。

$$(1) \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}$$

$$(2) \begin{pmatrix} 4 & 1 & 0 & 0 & 0 \\ 2 & 4 & 1 & 0 & 0 \\ 0 & 2 & 4 & 1 & 0 \\ 0 & 0 & 2 & 4 & 1 \\ 0 & 0 & 0 & 2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 2 \\ -1 \\ -1 \\ 0 \\ 3 \end{pmatrix}$$

[3] 係数行列が5重対角の形の以下の連立1次方程式をLU分解の方法で解く。

$$\underbrace{\begin{pmatrix} a_1 & c_1 & f_1 & & & 0 & & \\ b_2 & a_2 & c_2 & f_2 & & & & \\ e_3 & b_3 & a_3 & c_3 & f_3 & & & \\ \dots & \dots & \dots & \dots & \dots & \dots & & \\ e_{N-2} & b_{N-2} & a_{N-2} & c_{N-2} & f_{N-2} & & & \\ 0 & e_{N-1} & b_{N-1} & a_{N-1} & c_{N-1} & f_{N-1} & & \\ & e_N & b_N & a_N & c_N & f_N & & \end{pmatrix}}_{\text{係数行列 } A} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{N-2} \\ x_{N-1} \\ x_N \end{pmatrix}}_x \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{N-2} \\ y_{N-1} \\ y_N \end{pmatrix}}_y$$

(1) L, U をそれぞれ

$$L = \begin{pmatrix} 1 & & & & \\ l_2 & 1 & & & 0 \\ m_3 & l_3 & 1 & & \\ \dots & \dots & \dots & \dots & \\ 0 & m_N & l_N & 1 & \end{pmatrix}, \quad U = \begin{pmatrix} d_1 & u_1 & f_1 & & 0 & & \\ & \dots & \dots & \dots & & & \\ & & & d_{N-2} & u_{N-2} & f_{N-2} & \\ & & & 0 & d_{N-1} & u_{N-1} & \\ & & & & & d_N & \end{pmatrix}$$

さらに勉強するために

本書では数値計算におけるいくつかのテーマを取り上げ、初学者向けにていねいに解説した。このため、数学的に厳密な導出・証明や、より高度な数値計算法に関する記述を割愛せざるを得なかった。また、本書で触れることのできなかったテーマも存在する。このため、数値計算に興味をもたれた読者が、より本格的に勉強するための参考書を以下に挙げる。なお、本書を執筆する際に、筆者もこれらの書物を大いに参考にさせていただいた。

まず、数値計算全般に関する参考書としては以下のものをおすすめする。

- [1] 洲之内治男：『数値計算』、サイエンス社(1978)
- [2] 森正武：『FORTRAN 77 数値計算プログラミング増補版』、岩波書店(1987)
- [3] 森口繁一：『数値計算工学』、岩波書店(1989)
- [4] 山本哲朗：『数値解析入門』、サイエンス社(1976)

[1]は、基本的な数値計算についてコンパクトにまとめている。[2]は実用的な解法を中心にプログラミングの技術も含めて解説している。[3]は豊富な計算例によって理論の検証をていねいに行なっている。[4]は数値計算全般にわたって理論をまとめている。

実際に数値計算を行なう場合に、どの解法を用いるのがよいか、どのような点に注意すべきかなど迷うことが多い。この際の指針を与えてくれる参考書として以下の2つをおすすめする。

- [5] W. Press, B. Flannery, S. Teukolsky and W. Vetterling:『Numerical Recipes in C 日本語版』(丹慶勝市・奥村晴彦・佐藤俊郎・小林誠訳)、技術評論社(1993)
 - [6] 伊理正夫・藤野和建：『数値計算の常識』、共立出版(1985)
- [5]は数値計算の各テーマについて解法を網羅し、どのような場合にどの解法

が適しているかを根拠まで含めて解説している。さらに、C言語のプログラムが提供されている。[6]は数値計算を行なう際の考え方や注意点を書き連ねたユニークな心覚え集である。

本書の第5章～第7章で解説した微分方程式や連立1次方程式の数値計算に関しては、そのテーマだけで1冊の参考書として書かれているものも多い。以下にいくつか挙げる。

- [7] 菊地文雄・山本昌宏：『微分方程式と計算機演習』、山海堂(1991)
- [8] 戸川隼人：『微分方程式の数値計算』、オーム社(1973)
- [9] 高見穎郎・河村哲也：『偏微分方程式の差分解法』、東京大学出版会(1994)
- [10] 菊地文雄：『有限要素法概説』、サイエンス社(1980)
- [11] 戸川隼人：『マトリクスの数値計算』、オーム社(1971)
- [12] 村田健郎・名取亮・唐木幸比古：『大型数値シミュレーション』、岩波書店(1990)

[7]は主に微分方程式に関して解説と演習問題・解答をバランスよく配置している。[8]は微分方程式に関して、有限要素法と差分法を中心にして理論と解法をまとめたハンドブックである。[9]は前半で偏微分方程式の差分法の基礎を解説し、後半で流体力学への応用とそのためのより高度なテクニックを解説している。なお、筆者は学生時代に偏微分方程式の数値計算に関して高見、河村両先生に手ほどきを受け、その経験が本書でも生きている。[10]は微分方程式の有限要素法に関する初学者向けの参考書である。[11]は連立1次方程式、固有値問題など行列に関する話題を中心に、理論と解法をまとめたハンドブックである。[12]は行列が関わる問題について、特に大型行列の数値計算に重点を置いて理論と応用を解説している。

演習問題略解

演習問題の略解を以下に示す。なお、計算結果の数値は下位の桁を4捨5入や切り捨てなどによって丸めている。また、使用する計算機環境が異なると、計算結果は微妙に変わりうる。したがって、いつも同じ答が得られるとは限らないので注意して欲しい。

第1章

[1] (1)足し算が n 回、掛け算が $\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}$ 回の合計 $n(n+1)/2$ 回。 $n(n+1)/2 = O(n^2)$ 。 (2)55秒、1時間24分10秒、5日19時間1分40秒。 (3)引き算が2回、割り算が1回、掛け算が n 回の合計 $n+3$ 回。13秒、1分43秒、16分43秒。

[2] (1)1.23、(2)136、(3)0.00543。

[3] (1)9.875、相対誤差は約 -1.56×10^{-4} 。 (2)9.877、相対誤差は約 4.66×10^{-5} 。 (3)10.00、相対誤差は約 1.25×10^{-2} 。

[4] (1.15)式の h を $-2h$ で置き換えて

$$f(x-2h) = \sum_{j=0}^{n-1} \frac{(-2h)^j}{j!} f^{(j)}(x) + \frac{(-2h)^n}{n!} f^{(n)}(\xi)$$

ここで ξ は $x-2h$ と x の間の数。例えば $n=3$ のとき

$$f(x-2h) = f(x) - 2hf'(x) + 2h^2f''(x) - \frac{4}{3}h^3f'''(\xi)$$

[5] (1.16b)式より $(x+h)^3 = x^3 + 3hx^2 + 3h^2\xi$ 。両辺を比較して $\xi = x + \frac{h}{3}$ 。よって ξ は h の正負を問わず x と $x+h$ の間にある。

[6] (i)55、(ii)27、(iii)100。

第2章

[1] (1)0.824132…、(2)0.680598…。

[2] (1)2。[1.5と2.5の間には $x=2$ の根しか存在しない。] (2)1。[-2と5の間には、1, 2, 3の3つの根が存在している。しかし、1回目の反復で新たな (a, b) は $(-2, 1.5)$ となり、 a, b の間には $x=1$ の根しか存在しない。]

[3] (1)1、(2)3。[両方ともグラフを描いて考えれば明らか。]

[4] 初期値 x_0 が 1.53, 1.54, 1.55, 1.56 のとき、得られる根はそれぞれ 3, 1, 3, 2 となる。例えば $x_0=1.54$ の場合、 $x_1=2.53$, $x_2=-0.053$ となり、 n が小さいうちは x_n が3つの