1.4.2 反復法 −クリロフ部分空間法−

ガウスの消去法などの直接法は機械的に繰り返せば答えがでるので頑健 (robust) ではあるが, 計算量が $\mathcal{O}(n^3)$ なので n が 10 倍になれば計算時間は 1000 倍になる. これは 1 分で解けたものが 16 時間になるということである. 実用上は非ゼロ要素がごく一部にしかない巨大な行列を解くことが多い. このような非ゼロ要素が一部にしかないという行列には, 帯行列 (band matrix), スカイライン行列 (skyline matrix), 疎行列 (sparse matrix) がある. このような行列ではゼロ要素にまでメモリを割り当てたり, 消去演算をするのは計算資源の無駄であり, このような場合, **反復法** (iterative method) がよく使われる.

反復法は, 反復回数 k に依存しない B と $\boldsymbol{c}$ によって

$$\boldsymbol{x}_k = B\boldsymbol{x}_{k-1} + \boldsymbol{c}$$

と表される**定常的な解法** (stationary methods) と, 各反復ごとに変化する情報を積極的に計算に取り込む**非定常的な解法** (nonstationary methods) の 2 種類に大別される. 定常的な解法には, ヤコビ (Jacobi) 法, ガウス−ザイデル (Gauss-Seidel) 法, 逐次的過剰緩和法 (SOR 法), 対称逐次的過剰緩和法 (SSOR 法) などがある. 非定常的な解法には共役勾配法 (CG 法), 最小残差法 (MINRES 法), 一般化最小残差法 (GMRES 法), 双共役勾配法 (BiCG 法), 疑似最小残差法 (QMR 法), 二乗共役勾配法 (CGS 法), 安定化双共役勾配法 (Bi-CGSTAB 法) などがある. 本書では非定常的な解法の多くが属するクリロフ部分空間法について述べる.

$A = (I - (I - A))$ とすると, $A\boldsymbol{x} = \boldsymbol{b}$ は $\boldsymbol{x} = \boldsymbol{b} + (I - A)\boldsymbol{x}$ と書ける. ここで, 反復式

$$\begin{aligned} \boldsymbol{x}_k &= \boldsymbol{b} + (I - A)\boldsymbol{x}_{k-1} \\ &= (\boldsymbol{b} - A\boldsymbol{x}_{k-1}) + \boldsymbol{x}_{k-1} \\ &= \boldsymbol{r}_{k-1} + \boldsymbol{x}_{k-1} \end{aligned} \tag{1.15}$$

を導入し, 式 (1.15) の繰り返し (反復) によって $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k$ を求める. $\boldsymbol{r}_{k-1}$ は $(k-1)$ 回の反復時の残差ベクトルであり, k 回目の反復解 $\boldsymbol{x}_k$ は $\boldsymbol{r}_{k-1}$ と $\boldsymbol{x}_{k-1}$ から作る.

同様の操作を繰り返すと

$$\boldsymbol{x}_k = \boldsymbol{r}_{k-1} + \boldsymbol{r}_{k-2} + \cdots + \boldsymbol{r}_0 + \boldsymbol{x}_0$$

となり, 反復解 $\boldsymbol{x}_k$ は初期値 $\boldsymbol{x}_0$ と反復ごとの残差 $\boldsymbol{r}_i$ で表される. 式 (1.15) の両辺に左から A をかけて $\boldsymbol{b}$ から引くと, $\boldsymbol{r}_k = \boldsymbol{b} - A\boldsymbol{x}_k$ より

$$\begin{aligned} \boldsymbol{r}_k &= (\boldsymbol{b} - A\boldsymbol{x}_{k-1}) - A\boldsymbol{r}_{k-1} \\ &= \boldsymbol{r}_{k-1} - A\boldsymbol{r}_{k-1} \\ &= (I - A)\boldsymbol{r}_{k-1} \end{aligned}$$

となる. 改めて $\boldsymbol{x}_k$ を 残差ベクトルで表すと

$$\begin{aligned} \boldsymbol{x}_k &= \boldsymbol{r}_{k-1} + \cdots + \boldsymbol{r}_0 + \boldsymbol{x}_0 \\ &= (I - A)\boldsymbol{r}_{k-2} + (I - A)\boldsymbol{r}_{k-3} + \cdots + (I - A)\boldsymbol{r}_0 + \boldsymbol{r}_0 + \boldsymbol{x}_0 \\ &= \left[(I - A)^{k-1} + (I - A)^{k-2} + \cdots + (I - A) + I\right]\boldsymbol{r}_0 + \boldsymbol{x}_0 \end{aligned}$$

となる. $\boldsymbol{z}_k = \left[(I - A)^{k-1} + (I - A)^{k-2} + \cdots + (I - A) + I\right]\boldsymbol{r}_0$, つまり $\boldsymbol{z}_k$ は k 次の**クリロフ部分空間** (Krylov subspace) $\left[\boldsymbol{r}_0, A\boldsymbol{r}_0, \cdots, A^{k-1}\boldsymbol{r}_0\right]$ に属するベクトルである. 問題はクリロフ部分空間からどのようにして解の近似ベクトル $\boldsymbol{x}_k$ を求めるかにある.

対称正定値行列 A に対しては $A\boldsymbol{x} = \boldsymbol{b}$ の解が

$$f(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}, A\boldsymbol{x}) - (\boldsymbol{x}, \boldsymbol{b}) \tag{1.16}$$

を最小にするという事実に基づいて作られた **共役勾配法** (conjugate gradient method; CG 法) がベストである. 共役勾配法では $(k+1)$ 回目の反復における近似解を $\boldsymbol{x}_{k+1}$, $\boldsymbol{x}_k$ の修正ベクトルを $\boldsymbol{p}_k$ として

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k \tag{1.17}$$

とする. 式 (1.17) を式 (1.16) に代入すると

$$\begin{aligned} f(\boldsymbol{x}_{k+1}) &= f(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k) \\ &= f(\boldsymbol{x}_k) - \alpha_k(\boldsymbol{p}_k, \boldsymbol{r}_k) + \frac{1}{2}\alpha_k^2(\boldsymbol{p}_k, A\boldsymbol{p}_k) \end{aligned} \tag{1.18}$$

を得る. 式 (1.18) を α_k の関数として最小化するには

$$\alpha_k = \frac{(\boldsymbol{p}_k, \boldsymbol{r}_k)}{(\boldsymbol{p}_k, A\boldsymbol{p}_k)} \tag{1.19}$$

とすればよい. 修正ベクトル $\boldsymbol{p}_{k+1}$ は

$$\boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k \tag{1.20}$$

〈例 1.2〉 対称正定値行列 A を係数とする連立 1 次方程式 $Ax = b$ を CG 法で解く.

[アルゴリズム]

初期ベクトル x_0 を用意

$r_0 = b - Ax_0; p_0 = r_0$

for $k = 0, 1, \cdots$

$\alpha_k = \frac{(r_k, r_k)}{(p_k, Ap_k)}$

$x_{k+1} = x_k + \alpha_k p_k$

$r_{k+1} = r_k - \alpha_k Ap_k$

収束判定

$\beta_k = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}$

$p_{k+1} = r_{k+1} + \beta_k p_k$

のように決め, Ap_k と直交するように β_k を選ぶ.

$$(Ap_k, r_{k+1} + \beta_k p_k) = (Ap_k, r_{k+1}) + \beta_k (Ap_k, p_k) = 0$$

より

$$\beta_k = -\frac{(Ap_k, r_{k+1})}{(Ap_k, p_k)} \tag{1.21}$$

となる. アルゴリズムを例 1.2 に示す (演習問題 5 参照).

このアルゴリズムによって得られる r_k, p_k は

$$(r_i, r_j) = 0; i \neq j \quad \text{(直交性)}$$
$$(p_i, Ap_j) = 0; i \neq j \quad \text{(共役性)}$$

を満足する. 残差ベクトル r_k の直交性より, 数値計算による誤差がなければ高々 n 回の反復で $r_k = o$ となる. したがって CG 法は n 回の反復で必ず厳密解に収束しうる解法である.

一方, 正定値でない対称行列, 非対称行列に対しても多くの解法が提案されているが, 決定版といえる解法はない. ここでは CG 法と同じようなアルゴリズムである

〈例 1.3〉 連立 1 次方程式 $Ax = b$ を BiCG 法で解く.

[アルゴリズム]

初期ベクトル x_0 と $\tilde{x}_0$ を用意

$r_0 = b - Ax_0; \tilde{r}_0 = \tilde{b} - A^T \tilde{x}_0$

$p_0 = r_0; \tilde{p}_0 = \tilde{r}_0$

for $k = 0, 1, \cdots$

$\alpha_k = \frac{(r_k, \tilde{r}_k)}{(Ap_k, \tilde{p}_k)}$

$x_{k+1} = x_k + \alpha_k p_k$

$r_{k+1} = r_k - \alpha_k Ap_k; \tilde{r}_{k+1} = \tilde{r}_k - \alpha_k A^T \tilde{p}_k$

収束判定

$\beta_k = \frac{(r_{k+1}, \tilde{r}_{k+1})}{(r_k, \tilde{r}_k)}$

$p_{k+1} = r_{k+1} + \beta_k p_k; \tilde{p}_{k+1} = \tilde{r}_{k+1} + \beta_k \tilde{p}_k$

る**双共役勾配法** (bi-conjugate gradient method; BiCG 法) をとりあげる. 双共役勾配法では方程式

$$Ax = b$$

と双対な方程式

$$A^T \tilde{x} = \tilde{b}$$

を組み合わせて計算する. ふつうは $\tilde{b} = b$ とする. アルゴリズムを例 1.3 に示す.

このアルゴリズムから得られる $r_k, \tilde{r}_k, p_k, \tilde{p}_k$ は

$$(r_i, \tilde{r}_j) = 0; i \neq j \quad \text{(双対直交性)}$$
$$(Ap_i, \tilde{p}_j) = 0; i \neq j \quad \text{(双対共役直交性)}$$

を満足する. そのため, α_k や β_k の分母が 0 になってアルゴリズムが破綻 (breakdown) しなければ, BiCG 法は高々 n 回の反復で解に収束する. 破綻を避けるためのアルゴリズムの改良の 1 つには, 破綻が起きたときの $x_k, \tilde{x}_k$ をそれぞれ $x_0, \tilde{x}_0$ として最初からやり直す (restart) という方法がある.

反復法の収束は係数行列 A と右辺 $\boldsymbol{b}$ の性質に依存するため, 反復法では $A\boldsymbol{x}=\boldsymbol{b}$ を同値で性質の良い方程式に変換してから解くのが一般的である. これを**前処理** (preconditioning) という. 前処理行列 (precoditioner) として行列 M および N を用意し, $M^{-1}AN^{-1}\tilde{\boldsymbol{x}}=M^{-1}\boldsymbol{b}$, $\tilde{\boldsymbol{x}}=N\boldsymbol{x}$ とする. $M^{-1}AN^{-1}$ が単位行列に近くなれば早く収束することが期待される ($A=M$, $N=I$ なら解けたことになる). 実際は, あらかじめ変換された方程式を作るのでなく, アルゴリズムを変換された方程式に対するものに書き換える. 反復中では $M^{-1}\boldsymbol{p}$ または $N^{-1}\boldsymbol{p}$ の計算が必要になるため, 効率の良い前処理には

- MN が A をよく近似していること
- $M^{-1}\boldsymbol{p}$, $N^{-1}\boldsymbol{p}$ の計算が容易なこと

という条件が必要である.

一般の行列の場合は $N=I$, M として不完全 LDU 分解

$$A=LDU+R, L\text{は下三角行列}, D\text{は対角行列}, U\text{は上三角行列}$$

の LDU が使われる.R は不完全 LDU 分解の不完全さを表しており,R は問題の構造 (非ゼロ要素がどこにあるか) や性質に応じて決める. $R=0$ なら完全 LDU 分解となり, 反復法を適用するまでもなく問題は解けている.

対称行列の場合は, 不完全コレスキー分解

$$A=U^TU+R, U\text{は上三角行列}$$

を用いて

$$U^{-T}AU^{-1}\tilde{\boldsymbol{x}}=U^{-T}b,\ \tilde{\boldsymbol{x}}=U\boldsymbol{x} \tag{1.22}$$

とする.A が対称正定値行列なら $U^{-T}AU^{-1}$ も対称正定値行列にできる. 式 (1.22) に CG 法を適用したのが, 不完全コレスキー分解前処理つき共役勾配法 (incomplete Cholesky conjugate gradient method, ICCG 法) である.

一般に, 効率的な前処理を作るためには問題に対する知識, 例えば解こうとしている方程式がどのような物理問題から導かれたものか, 解の傾向はどうなっているかなどが必要である. これは初期ベクトル $\boldsymbol{x}_0$ についてもいえる. 反復法の場合, それぞれの解法には適した問題とそうでない問題があり, すべての問題に対してある解法が最適になることはない.

1.5 解けない方程式 1 —式の本数が少ない—

$$\begin{cases} x+2y+3z=0 \\ 2x-y-3z=0 \end{cases}$$

のように未知数よりも方程式の本数が少ない場合, この連立方程式は一意には解けない. この場合もどれか 1 つの未知数の項, 例えば z の項を右辺に移してその値を決めると x, y の値は z に応じて一意に決まる.

$$\begin{cases} x+2y=-3z \\ 2x-y=3z \end{cases}$$

〈例 1.4〉 上階段化ガウスの消去法を用いて行列 A のランクを求める.

[アルゴリズム]

```
ir = 0
for k = 1, 2, ···, n
    amax = | a_{k-ir,k} |; ip = k - ir
    for i = k - ir + 1, k - ir + 2, ···, n
        if | a_ik | > amax then
            amax = | a_ik |; ip = k
        end if
    if amax > ε then
        for j = k + 1, k + 2, ···, n
            a_{ip,j} ⇌ a_{k-ir,j}
        for i = k - ir + 1, k - ir + 2, ···, n
            α_ik = -a_ik / a_kk
            for j = k + 1, k + 2, ···, n
                a_ij = a_ij + α_ik · a_kj
    else
        ir = ir + 1
    end if
```

この場合, 解は2つの面の交わる部分なので, 3次元空間内の直線になる. これと同じことは例 1.1 の部分軸選択つきガウスの消去法で $ir \neq 0$ となったときにもいえる. $ir \neq 0$ すなわち $\mathrm{rank}(A) < n$ のとき, 行列 A は正則でないので方程式 $Ax = b$ は一意な解を持たない. しかし, 解空間が空集合 (ϕ) というわけではない. まずは例 1.1 のアルゴリズムを行列 A の rank が数値的に正しく決められるよう修正しよう. 修正後のアルゴリズムを例 1.4 に示す. 例 1.1 の部分軸選択つきガウスの消去法では消去に用いるピボットを対角から下で探索していたのに対し, 例 1.4 ではその時点までに何本の1次独立なベクトルがあったかによって変える. このアルゴリズムを上階段化ガウスの消去法といい, 消去結果を上階段行列という. $\mathrm{rank}(A) = n - ir$ であり, $\mathrm{rank}(A) = n$ の場合は部分軸選択つきガウスの消去法による結果と同一である. さて, 上階段化ガウスの消去法による消去結果が

$$\begin{pmatrix} u_{11} & u_{12} & u_{13} & u_{14} & u_{15} & u_{16} & u_{17} \\ & u_{22} & u_{23} & u_{24} & u_{25} & u_{26} & u_{27} \\ & & & u_{34} & u_{35} & u_{36} & u_{37} \\ & & & & u_{45} & u_{46} & u_{47} \\ & & & & & u_{56} & u_{57} \\ & & \mathbf{0} & & & & 0 \\ & & & & & & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \end{pmatrix}$$

だったとしよう. このとき, 行列 U を縦ベクトル $u_1 \cdots u_7$ が並んだものとしてベクトル形式で書くと

$$u_1x_1 + u_2x_2 + \cdots + u_7x_7 = c$$

となる. この式に解が存在するためには $c_6 = c_7 = 0$ でなければならない. 上階段化ガウスの消去法アルゴリズムから明らかなように U は

u_3はu_1とu_2の1次結合,
u_7はu_1, u_2, u_4, u_5, u_6の1次結合,
u_1, u_2, u_4, u_5, u_6が1次独立

である. 1次従属なベクトル u_3, u_7 を右辺に移項して, すべて0であるベクトル

名取亮編. 数値計算法, オーム社, 1998

の第6要素, 第7要素を取り去った5次元ベクトルで考えれば

$$\begin{pmatrix} u_{11} & u_{12} & u_{14} & u_{15} & u_{16} \\ & u_{22} & u_{24} & u_{25} & u_{26} \\ & & u_{34} & u_{35} & u_{36} \\ & & & u_{45} & u_{46} \\ & & & & u_{56} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{pmatrix} - x_3 \begin{pmatrix} u_{13} \\ u_{23} \\ 0 \\ 0 \\ 0 \end{pmatrix} - x_7 \begin{pmatrix} u_{17} \\ u_{27} \\ u_{37} \\ u_{47} \\ u_{57} \end{pmatrix} \tag{1.23}$$

となる. 左辺の上三角行列はすべての対角要素がゼロでないから正則であり, 式 (1.23) は x_3, x_7 を任意に定めることにより解ける.

一般に $n \times n$ の行列 A が $\mathrm{rank}(A) = r$ で $b \in \mathrm{span}(A)$ だったとすると, A のゼロ空間 $\mathrm{zero}(A) = \{x | Ax = o\}$ は $(n-r)$ 本の1次独立なベクトル $z_1, \cdots, z_{n-r}$ から形成される. すなわち

$$\mathrm{zero}(A) = \mathrm{span}(z_1, \cdots, z_{n-r})$$

である. $Ax = b$ の解は

$$x = x_0 + \sum_{j=1}^{n-r} \alpha_j z_j$$

となる. x_0 は $Ax = b$ の特解で, $\mathrm{rank}(A) = n$ のとき $\mathrm{zero}(A)$ は $\{o\}$ である.

1.6 解けない方程式2 —式の本数が多い—

方程式の本数が未知数の数より多い場合も, 一部の例外を除いて, 解は存在しない. 例えば

$$\begin{cases} x + 2y = 3 \\ 2x - y = 4 \\ ax + y = 7 \end{cases}$$

という3本の式があったとき, $a = 3$ の場合に限り解は存在するが, $a \neq 3$ の場合はどの2本の式を連立させるかによって x, y の値が変わってくる. ところが, モデル $ax + by$ (a, b は未知の定数) に従う物理現象をいくつかの実測値から予測しようという場合などのように, 未知数の数よりも方程式の本数が多くなるという応用は少なくない. このとき厳密な意味での解は存在しないので, これらの条件をできる

だけ満足するよう x, y を定めようというのが**最小二乗法** (least squares method) の考え方で, このときの解が最小二乗解

$$min||\boldsymbol{b} - A\boldsymbol{x}||^2 \tag{1.24}$$

である. $\boldsymbol{b} - A\boldsymbol{x}$ は残差であり, 残差が $\boldsymbol{o}$ なら厳密解になる.

A は $m \times n$ の行列, $\boldsymbol{b}$ は m 次元ベクトル, $\boldsymbol{x}$ は n 次元ベクトル ($m \geq n$) とする. 式 (1.24) を最小化するには, 同値な式

$$\begin{aligned}&(\boldsymbol{b} - A\boldsymbol{x}, \boldsymbol{b} - A\boldsymbol{x})\\ &= (A^TA\boldsymbol{x}, \boldsymbol{x}) - 2(A^T\boldsymbol{b}, \boldsymbol{x}) + (\boldsymbol{b}, \boldsymbol{b})\end{aligned}$$

を最小化すればよい. この式は

$$A^TA\boldsymbol{x} = A^T\boldsymbol{b} \tag{1.25}$$

のときに最小値をとる. 式 (1.25) を**正規方程式** (normal equation) という. 正規方程式は $n \times n$ の対称行列 A^TA を係数に持つ方程式なのでガウスの消去法でも解けるが, A^TA の条件数は A の条件数の 2 乗なので, 計算誤差の問題から式 (1.25) をそのまま解くのは好ましくない. そこで A を $m \times n$ の直交行列 Q ($Q^TQ = I$) と, $n \times n$ の上三角行列 R を用いて

$$A = QR$$

と分解する. この分解を ***QR*分解**(QR decomposition) と呼ぶ.

QR 分解の結果を用いると, 式 (1.25) は

$$\begin{aligned}&A^TA\boldsymbol{x} = (QR)^T(QR)\boldsymbol{x} = R^TQ^TQR\boldsymbol{x} = R^TR\boldsymbol{x},\\ &A^T\boldsymbol{b} = (QR)^T\boldsymbol{b} = R^TQ^T\boldsymbol{b}\end{aligned}$$

となり, 両辺に R^T の逆行列を作用させれば, 上三角行列 R を係数とする方程式

$$R\boldsymbol{x} = Q^T\boldsymbol{b}$$

が得られる. 正方行列 R に逆行列は存在しうるが, $m \times n$ $(m \geq n)$ の行列 A には逆行列が存在しないことに注意しよう. QR 分解の代表的な方法には修正グラム・シュミット (Gram-Schmidt) 法とハウスホルダー変換を用いる方法がある. ここでは修正グラム・シュミット法の概略だけを述べ, ハウスホルダー変換や QR 分解の詳細については 2.5 節で述べる.

for $i = 1, 2, \cdots, n$

ノーマル
$r_{ii} = \|\vec{a}_i\|$; $\vec{q}_i = \vec{a}_i / r_{ii}$

for $j = i+1, i+2, \cdots, n$

$r_{i,j} = (\vec{a}_j, \vec{q}_i)$

$\vec{a}_j = \vec{a}_j - r_{ij}\vec{q}_i$

修正グラム・シュミット法では 1 次独立なベクトル $\boldsymbol{a}_1$ を正規化して $\boldsymbol{q}_1$ とし, $\boldsymbol{q}_1$ 成分を $\boldsymbol{a}_2, \cdots, \boldsymbol{a}_n$ から抜き去る. この結果, $\boldsymbol{a}_2$ は $\boldsymbol{q}_1$ と直交するので, 正規化して $\boldsymbol{q}_2$ とし, $\boldsymbol{q}_2$ 成分を $\boldsymbol{a}_3, \cdots, \boldsymbol{a}_n$ から抜き去る. この操作を n まで繰り返せば正規直交ベクトル $\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_n$ が作られる. アルゴリズムは例 1.5 のようになる.

1.7 固有値・固有ベクトル

線形代数の重要なトピックスに固有値, 固有ベクトルがある. 正方行列 A の**固有値** (eigenvalue) とは

$$A\boldsymbol{v} = \lambda\boldsymbol{v} \tag{1.26}$$

を満足する値 λ のことであり, このときの $\boldsymbol{o}$ でないベクトル $\boldsymbol{v}$ を**固有ベクトル** (eigenvector) という. この方程式は

$$(A - \lambda I)\boldsymbol{v} = \boldsymbol{o}$$

とも書ける. 簡単そうな連立 1 次方程式にみえるが未知数が λ を含めて $n+1$ で, 式の本数が n , しかも λ と v_i の積があるなど, これだけでは解くことができない. そこで $\boldsymbol{v}$ がゼロベクトルでないのに $(A - \lambda I)\boldsymbol{v} = \boldsymbol{o}$ となることを利用して, $A - \lambda I$ が非正則あるいは特異となるような λ をまず求め, そのような λ に対して 式 (1.26) を満足する固有ベクトル $\boldsymbol{v} \neq \boldsymbol{o}$ を求める. ただし $\boldsymbol{v}$ が固有ベクトルなら $k\boldsymbol{v}$ (k はス

カラー) も固有ベクトルになることから, $\boldsymbol{v}$ にはスカラー倍の任意性がある. $A-\lambda I$ が特異となるような λ を求めるには, 手計算だと $\det(A-\lambda I)=0$ を λ に関する n 次方程式と思って計算するのが一般的であるが, コンピュータを使うような場合に, $n\times n$ の行列から固有多項式 (n 次方程式) の係数を計算し, そのすべての解をニュートン法などで求めるのは計算の手間からも現実的でない.

行列が非対称になると, 実数行列からも複素固有値が出現するなど, 問題は格段に難しくなるので, 本書においては対称行列だけを扱う.

数値計算でよく使われるのは, 行列固有の性質

- 対角行列 A の固有値は対角要素の値 λ_i
- 三角行列 U の固有値は対角要素の値 u_{ii}
- **相似変換** $S^{-1}AS$ によって固有値は変わらない

を利用して固有値を求める方法である.

まずは**ギブンス (Givens) の面内回転行列**

$$R_{ij}=\begin{array}{c} \\ i\text{ 行} \\ \\ j\text{ 行} \\ \\ \end{array}\begin{array}{c} \begin{array}{ccccccc} & & i\text{ 列} & & j\text{ 列} & & \end{array} \\ \begin{pmatrix} 1 & & \vdots & & \vdots & & \\ & \ddots & \vdots & & \vdots & & \\ \cdots & \cdots & \cos\theta & \cdots & -\sin\theta & \cdots & \cdots \\ & & \vdots & \ddots & \vdots & & \\ \cdots & \cdots & \sin\theta & \cdots & \cos\theta & \cdots & \cdots \\ & & \vdots & & \vdots & \ddots & \\ & & \vdots & & \vdots & & 1 \end{pmatrix} \end{array}$$

を考えよう. R_{ij} と単位行列とは $r_{ii}, r_{ij}, r_{ji}, r_{jj}$ だけが異なり, $R_{ij}^T=R_{ij}^{-1}$ なので, $R_{ij}^TAR_{ij}$ は相似変換となる.

AR_{ij} と A は式 (1.27) のように i 列と j 列のみが異なり, R_{ij}^TA と A は式 (1.28) のように i 行と j 行のみが異なる. したがって $A'=R_{ij}^TAR_{ij}$ では i 行, j 行, i 列, j 列の値が変化する.

$$\begin{array}{c} \\ AR_{ij}= \end{array}\begin{array}{c} \begin{array}{cc} i\text{ 列} & j\text{ 列} \end{array} \\ \begin{pmatrix} a_{ki}\cos\theta+a_{kj}\sin\theta & -a_{ki}\sin\theta+a_{kj}\cos\theta \end{pmatrix} \end{array} \qquad k=1,\ \cdots,\ n \tag{1.27}$$

$$R_{ij}^TA=\begin{array}{c} \\ i\text{ 行} \\ \\ j\text{ 行} \\ \\ \end{array}\begin{pmatrix} \cos\theta a_{ik}+\sin\theta a_{jk} \\ -\sin\theta a_{ik}+\cos\theta a_{jk} \end{pmatrix} \qquad k=1,\ \cdots,\ n \tag{1.28}$$

$A'=R_{ij}^TAR_{ij}$ の i 行 (j 行) と i 列 (j 列) が交差する場所の値は

$$\begin{aligned} a'_{ii} &= a_{ii}\cos^2\theta+a_{jj}\sin^2\theta+2a_{ij}\cos\theta\sin\theta \\ a'_{ij} &= \frac{1}{2}(a_{jj}-a_{ii})\sin 2\theta+a_{ij}\cos 2\theta \\ &= a'_{ji} \\ a'_{jj} &= a_{ii}\sin^2\theta+a_{jj}\cos^2\theta-2a_{ij}\cos\theta\sin\theta \end{aligned}$$

となる. このとき

$$\frac{a_{ii}-a_{jj}}{2a_{ij}}=(\tan 2\theta)^{-1}$$

となるように θ を選べば $a'_{ij}=a'_{ji}=0$ にできる. 回転行列に必要なのは $\sin\theta$, $\cos\theta$ であって θ ではないことに注目し, θ を求めず高速化を図ったアルゴリズムに高速ギブンス変換 (fast Givens transform) がある.

この操作を用いて対称行列 A を三重対角行列に変換するのが**ギブンス法**で, 非対角要素の値が十分に小さくなるまで繰り返して対角行列 A に変換するのが**ヤコビ (Jacobi) 法**である. ギブンス法は有限回の面内回転で三重対角化できる. 一般に三重対角化にはハウスホルダー変換を用いることが多い.

対称行列の固有ベクトルが直交基底を形成することを利用して固有値, 固有ベクトルを求めるのがべき乗法と逆反復法である.

固有値が $|\lambda_1|>|\lambda_2|\geq\cdots\geq|\lambda_n|$ のように並んでいるとしよう. 任意のベクトル $\boldsymbol{x}$ を固有ベクトル $\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_n$ を使って

$$\boldsymbol{x}=a_1\boldsymbol{v}_1+a_2\boldsymbol{v}_2+\cdots+a_n\boldsymbol{v}_n;\quad \boldsymbol{v}_i\text{は}\lambda_i\text{に対応する固有ベクトル} \tag{1.29}$$

と表す. 式 (1.29) に左から行列 A を作用させると

$$\begin{aligned} A\boldsymbol{x} &= a_1A\boldsymbol{v}_1+a_2A\boldsymbol{v}_2+\cdots+a_nA\boldsymbol{v}_n \\ &= a_1\lambda_1\boldsymbol{v}_1+a_2\lambda_2\boldsymbol{v}_2+\cdots+a_n\lambda_n\boldsymbol{v}_n \end{aligned}$$

となる. この操作を繰り返すと

$$\begin{aligned} A^k\boldsymbol{x} &= a_1\lambda_1^k\boldsymbol{v}_1+a_2\lambda_2^k\boldsymbol{v}_2+\cdots+a_n\lambda_n^k\boldsymbol{v}_n \\ &= a_1\lambda_1^k\left\{\boldsymbol{v}_1+\frac{a_2}{a_1}\left(\frac{\lambda_2}{\lambda_1}\right)^k\boldsymbol{v}_2+\cdots+\frac{a_n}{a_1}\left(\frac{\lambda_n}{\lambda_1}\right)^k\boldsymbol{v}_n\right\} \end{aligned}$$

〈例 1.6〉 べき乗法を用いて対称行列 A の最大固有値 λ を求める.
[アルゴリズム]

初期値 $\tilde{\boldsymbol{x}}^{(0)}$ を用意
for $k = 1, 2, \cdots$
$\boldsymbol{x}^{(k)} = \tilde{\boldsymbol{x}}^{(k)} / ||\tilde{\boldsymbol{x}}^{(k-1)}||$
$\tilde{\boldsymbol{x}}^{(k)} = A\boldsymbol{x}^{(k)}$
$\lambda^{(k)} = (\boldsymbol{x}^{(k)}, \tilde{\boldsymbol{x}}^{(k)})$
収束判定

となる. $\left|\frac{\lambda_k}{\lambda_1}\right| < 1$,$2 \le k \le n$ であるから, $A^k\boldsymbol{x}$ は λ_1 に対応する固有ベクトル $\boldsymbol{v}_1$ の方向へ方向収束する. オーバーフローしないよう注意しながらこの操作を繰り返せば, 絶対値が最大の固有値 λ_1 に対応する固有ベクトル $\boldsymbol{v}_1$ が求められる.

固有ベクトル $\boldsymbol{v}_1$ が求まったら, **レーリー商** (Rayleigh quotient) を用いて, 固有値

$$\lambda_1 = \frac{(\boldsymbol{v}_1, A\boldsymbol{v}_1)}{(\boldsymbol{v}_1, \boldsymbol{v}_1)}$$

を求めればよい. これが**べき乗法** (power method) である. 例 1.6 にアルゴリズムを示す. $\lambda^{(k)}$ が A の最大固有値の近似値, $\boldsymbol{x}^{(k)}$ が最大固有値に対する固有ベクトルの近似ベクトルである.

固有値, 固有ベクトルの関係式 (1.26) は

$$A^{-1}\boldsymbol{v} = \frac{1}{\lambda}\boldsymbol{v} \tag{1.30}$$

とも書ける. 式 (1.30) は, 逆行列 A^{-1} の固有値は A の固有値の逆数であることと, A^{-1} の固有ベクトルは A と同じであることを示している. したがって A^{-1} に対するべき乗法を行えば, A^{-1} の絶対値最大の固有値と固有ベクトル, すなわち A の絶対値最小の固有値 λ_n と固有ベクトル $\boldsymbol{v}_n$ が求められる. この原理を応用したのが**逆反復法** (inverse iteration method) である. 例 1.7 にアルゴリズムを示す. $\mu^{(k)}$ が A^{-1} の最大固有値の近似値 (A の最小固有値の逆数) , $\boldsymbol{x}^{(k)}$ が A^{-1} の最大固有値に対する固有ベクトルの近似ベクトルである. 実際は逆行列 A^{-1} を作らず, ガウスの消去法を使って $A\tilde{\boldsymbol{x}}^{(k)} = \boldsymbol{x}^{(k)}$ を解く. 固有値の近似値 α がわかっていればシフト付きの逆反復法によって固有値・固有ベクトルが計算できる.

〈例 1.7〉 逆反復法を用いて対称行列 A の最小固有値 $\frac{1}{\mu}$ を求める.
[アルゴリズム]

初期値 $\tilde{\boldsymbol{x}}^{(0)}$ を用意
for $k = 1, 2, \cdots$
$\boldsymbol{x}^{(k)} = \tilde{\boldsymbol{x}}^{(k)} / ||\tilde{\boldsymbol{x}}^{(k-1)}||$
$A\tilde{\boldsymbol{x}}^{(k)} = \boldsymbol{x}^{(k)}$ を解く
$\mu^{(k)} = (\boldsymbol{x}^{(k)}, \tilde{\boldsymbol{x}}^{(k)})$
収束判定

$$\begin{aligned}(A - \alpha I)\boldsymbol{v} &= \lambda\boldsymbol{v} - \alpha\boldsymbol{v} \\ &= (\lambda - \alpha)\boldsymbol{v}\end{aligned}$$

より $(A - \alpha I)$ の絶対値最小の固有値, すなわち α に最も近い A の固有値と対応する固有ベクトルが求められる.

固有値の大まかな分布がわかれば, 逆反復法を利用して固有値・固有ベクトルが正確に求められるので, プログラムライブラリなどでは実対称行列の固有値・固有ベクトルを以下のような手順で求めている.

(1) ハウスホルダー変換 (またはギブンス法) を用いて三重対角行列に変換する
(2) スツルムの定理を利用した二分法によって固有値の大まかな分布を調べる
(3) 逆反復法によって固有値と三重対角行列の固有ベクトルを求める
(4) ハウスホルダー逆変換 (ギブンス逆変換) を用いて固有ベクトルを変換する