固有値問題ノートの補足

桂田 祐史

2005 年 8 月 21 日, 2008 年 6 月 9 日, 2013 年 8 月 26 日

(大規模作業中)
この文書は今のところ

http://nalab.mind.meiji.ac.jp/~mk/labo/text/eigenvalues-add.pdf

においてある。将来的には

『行列の固有値問題』
http://nalab.mind.meiji.ac.jp/~mk/labo/text/eigenvalues.pdf

にマージする予定である。
なお、一般化固有値問題については別に

『一般化固有値問題』
http://nalab.mind.meiji.ac.jp/~mk/labo/text/generalized-eigenvalue-problem.pdf

を用意してある。

目 次

1 いろは

$A \in \mathbf{C}^{n\times n}$ に対して、

$$Ax = \lambda x, \quad x \neq 0$$

を満たす x が存在するとき、λ を A の固有値 (eigenvalue) と呼び、x を λ に属する固有ベクトル (eigenvector) と呼ぶ。

λ が A の固有値であることと、

$$\ker(\lambda I - A) = \{x \in \mathbf{C}^n; Ax = \lambda x\}$$

が $\{0\}$ でないことは同値である。

λ が A の固有値であるとき、$\ker(\lambda I - A)$ を固有空間 (eigenspace) と呼ぶ。

定理 1.1 (固有値は固有多項式の根) λ が A の固有値であることと、n 次代数方程式

$$\det(\lambda I - A) = 0$$

の根であることは同値である。

定義 1.2 (固有多項式、固有方程式) $p(\lambda) := \det(\lambda I - A)$ を A の固有多項式、

$$p(\lambda) = 0$$

を固有方程式と呼ぶ。

系 1.3 () $A \in \mathbf{C}^{n\times n}$ の固有値の個数は n 個以下である。

A の固有多項式に重根がないときはちょうど n 個 (相異なる) 固有値が存在することになるが、重根がある場合はその重複度分だけ同じものをならべて、$\lambda_1, \cdots, \lambda_n$ と書くことが多い。以下、n 次正方行列 A について、A の固有値は $\lambda_1, \cdots, \lambda_n$ である、と言ったときはそういう意味である。例えば、「対角行列 $D = \mathrm{diag}(d_1, \cdots, d_n)$ の固有値は対角成分 $d_1, \cdots, d_n$ である。」…このことの証明は D の固有多項式についての等式

$$\det(\lambda I - D) = \prod_{i=1}^{n}(\lambda - d_i)$$

による。ちなみにこの事実は三角行列に対して一般化できる。すなわち「A が上三角行列または下三角行列とするとき、A の固有値は対角成分 $a_{11}, \cdots, a_{nn}$ である」。

系 1.4 (固有値は固有多項式の根ということから) (1) 任意の非特異行列 P に対して、A を相似変換した行列 $P^{-1}AP$ と A の固有値は重複度も込めて一致する。

(2) A と A^T の固有値は重複度も込めて一致する。

(3) A の固有値の複素共役は重複度も込めて A^* の固有値に一致する。

(4) $\det A$ は A の固有値 (重根の場合、重複度の分だけ並べる) の積に等しい。特に、

$$A \text{ が非特異} \quad \Leftrightarrow \quad 0 \text{ は } A \text{ の固有値でない.}$$

補題 1.5 A の相異なる固有値 $\lambda_1, \cdots, \lambda_m$ に属する固有ベクトルは 1 次独立である。

証明　ヴァンデルモンドの行列式を使う。 ■

命題 1.6 A が相異なる n 個の固有値を持つとき、すなわち固有方程式が重根を持たないとき、以下の (1), (2) が成り立つ。

(1) A の各固有値 $\lambda_1, \cdots, \lambda_n$ に属する固有ベクトルは全空間の基底をなす。

(2) A の各固有値に属する固有空間は 1 次元である。

この命題の状況のように、A の固有値 $\lambda_1, \cdots, \lambda_n$ に属する固有ベクトル $p_1, \cdots, p_n$ が全空間の基底となるとき、$P = (p_1, \cdots, p_n)$ とおくと、これは非特異で

$$P^{-1}AP = \mathrm{diag}(\lambda_1, \cdots, \lambda_n).$$

このように適当な非特異行列 P に対して、$P^{-1}AP$ が対角行列 $\mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ になるとき、A は対角化可能という。このとき、λ_i は A の固有値で、p_i は λ_i に属する固有ベクトル、p_1, $\cdots$, p_n は全空間の基底となる。

命題 1.7 A の固有値全体を $\lambda_1, \cdots, \lambda_n$ とするとき、以下がなりたつ。いずれの場合も対応する固有値の固有ベクトルは等しい。

(1) $c \in \mathbf{C}$ とするとき、cA の固有値は $c\lambda_1, \cdots, c\lambda_n$ である。

(2) $c \in \mathbf{C}$ とするとき、$cI + A$ の固有値は $c + \lambda_1, \cdots, c + \lambda_n$ である。

(3) A が可逆であるとき、A^{-1} の固有値は ${\lambda_1}^{-1}, \cdots, {\lambda_n}^{-1}$ である。

λ を A の固有値とするとき、$\overline{\lambda}$ は A^* の固有値であるが、その固有ベクトル x は

$$A^*x = \overline{\lambda}x$$

を満たすので、

$$x^*A = \lambda x^*$$

を満たす。$yA = \lambda y$, $y \neq 0$ を満たす y のことを固有値 λ に属する行固有ベクトルとよぶ。

定理 1.8 (Frobenius の定理) A の固有値を $\lambda_1, \cdots, \lambda_n$ とするとき、任意の多項式 $p(z)$ について、$p(A)$ の固有値は $p(\lambda_1), \cdots, p(\lambda_n)$ である。

定義 1.9 (固有値の代数的多重度、幾何学的多重度) $A \in \mathbf{C}^{n\times n}$ の固有値 λ に対して、固有値 λ の代数的多重度とは、固有多項式 $p(z) = \det(zI - A)$ における根 λ の多重度のことと定義する。また、固有値 λ の幾何学的多重度とは、λ に属する固有空間 $\ker(\lambda I - A)$ の次元のことと定義する。

2 概略

対称な問題 (行列が実対称行列や Hermite 行列である場合) と、そうでない問題との間には大きな違いがある。

古典的な Jacobi 法は別にして、次の手順を採る。

1. 与えられた行列を「簡単化」する。具体的には、対称な問題の場合は三重対角化し、非対称な問題の場合には Hessenberg 形にする。直交行列による相似変換が基本である。

2. 簡単化された問題を反復法で解く。冪乗法の系統 (逆反復法, シフト法を含む)、二分法、QR 法など。

この中継地点の発見をしたのは、Givens であるという。戸川 [11] (1971) から引用する。

> それを最初に行ったのはギブンスであった。彼は ① のために、のちにギブンスの方法と呼ばれるようになった有名な算法を考案し、また ② のために、現在ではスツルムの方法とか、バイセクション法と呼ばれている算法を考案した。それらは現在でも有力な手法として使われているが、その後、上記 ①, ② それぞれに対し、新しい方法が次つぎに発表され、どちらもきわめて能率よく計算できるようになった。現在、① の方法としてはハウスホルダー法が決定版といわれており、② の方法としては、小さい固有値を数個求めたいとか、大きい固有値を数個求めたいという場合はスツルム法、すべての固有値を求めたい場合は QR 法が最もよいとされている。問題によっては ① の目的に (改良された) ランチョス法が用いられることがある。

ちなみに Wilkinson がかの “Algebraic Eigenvalue Problem” を著したのは 1965 年である。

3 基本的な実直交行列とそれによる相似変換

超平面に対する対称移動である Householder 変換と、回転である Givens 変換が代表的かつ—きほんて—

3.1 Householder 変換

$\mathbf{R}^n$ の超平面に関する対称移動は明らかに内積を保つので直交変換である。これを **Householder 変換**、鏡映変換、または基本直交変換と呼ぶ。

超平面を $W_u = \{x \in \mathbf{R}^n; (x,u) = 0\}$ $(u \in \mathbf{R}^n \setminus \{0\})$ とおくと、W_u に関する対称移動を表わす行列は

$$H = H(u) = I_n - 2\frac{uu^T}{\|u\|^2}.$$

この形からも、図形的な意味からも、$H^{-1} = H^T = H$ (実直交行列かつ実対称行列) が成り立つことが分かる。

参考　N 次元の任意の直交変換は、N 個以下のベクトルに関する鏡映変換の積に書ける (Cartan)。

補題 3.1 ($\|x\| = \|y\| \neq 0$ なる 2 点を相互に写す鏡映変換) $\|x\| = \|y\|, x \neq y$ なる $x, y \in \mathbf{R}^n$ に対して、

$$u = \frac{x-y}{\|x-y\|}, \quad U = I - 2uu^T$$

とおくと $Ux = y$ かつ $Uy = x$.

証明　明らか。 ■

計算の工夫 **(1)** H と与えられたベクトルとの積

$H = H_u$ を記憶するのではなく、u と $\alpha = \dfrac{2}{\|u\|^2}$ を記憶しておく。

$x \in \mathbf{R}^n$ が与えられたときに $w := Hx$ を計算するには、

$$w := Hx = \left(I_n - 2\frac{uu^T}{\|u\|^2}\right)x = x - \frac{2}{\|u\|^2}uu^Tx = x - \alpha uu^Tx = x - (\alpha u^Tx)u$$

であるから、

$$\beta := \alpha(u^Tx), \quad w = x - \beta u$$

と計算すればよい。加減算 $2n-1$ 回、乗算 $2n+1$ 回で十分である。

計算の工夫 **(2)** H による相似変換 H^TAH

$$\widetilde{A} := H^TAH$$

を計算するには、

$$\alpha = \frac{2}{\|u\|^2}, \quad p = A^Tu, \quad q = Au, \quad v = \alpha u, \quad \beta = p^Tv, \quad \widetilde{A} = A - vp^T - (q - \beta u)v^T.$$

特に A が実対称行列の場合は、

$$\alpha = \frac{2}{\|u\|^2}, \quad p = \alpha(Au), \quad \beta = \frac{\alpha}{2}(p,u), \quad q = p - \beta u, \quad \widetilde{A} = A - uq^T - qu^T.$$

3.2　Givens 変換

$p, q \in \{1, \cdots, n\}$, $p \neq q$, $\theta \in \mathbf{R}$ とするとき、「x_px_q 平面における $-\theta$ の回転」を表わす行列 $G = G(p,q,\theta) = (g_{ij})$ を

$$g_{ij} = \begin{cases} \cos\theta & (i = j \in \{p,q\}) \\ \sin\theta & ((i,j) = (p,q)) \\ -\sin\theta & ((i,j) = (q,p)) \\ 1 & (i = j \notin \{p,q\}) \\ 0 & (\text{その他}) \end{cases}$$

で定義する。これは明らかに実直交行列である。

Givens 変換の行列

```
1  % Givens.m
2  %  G(p,q, θ ) ∈ M(n;R) (p ≠ q, c=cos θ , s=sin θ )
3  function G = Givens(n,p,q,c,s)
4    G = eye(n,n);
5    G(p,p) =   c; G(p,q) = s;
6    G(q,p) = - s; G(q,q) = c;
```

計算の工夫 **(1)** G^T を与えられた行列 A にかける

$$B := G(p,q,\theta)^T A$$

は p, q 行目を除き A と変わらない。変更される行については、

$$\begin{aligned} b_{pj} &= a_{pj}\cos\theta - a_{qj}\sin\theta \quad (j = 1, 2, \cdots, n), \\ b_{qj} &= a_{pj}\sin\theta + a_{qj}\cos\theta \quad (j = 1, 2, \cdots, n). \end{aligned}$$

θ を

$$\cos\theta = \frac{a_{qr}}{\sqrt{{a_{pr}}^2 + {a_{qr}}^2}}, \quad \sin\theta = \frac{a_{pr}}{\sqrt{{a_{pr}}^2 + {a_{qr}}^2}}$$

を満たすように取れば、$b_{pr} = 0$ となる。

計算の工夫 **(2)** 与えられた行列 A を G で相似変換 $(G^T AG)$

$$B := G^T AG, \quad G := G(p,q,\theta)$$

は p, q 行目, p, q 列目を除き A と変わらない。

$$\begin{aligned} b_{ij} &= a_{ij} \quad (i \neq p,q \text{ かつ } j \neq p,q), \\ b_{pj} &= a_{pj}\cos\theta - a_{qj}\sin\theta \quad (j \neq p,q), \\ b_{qj} &= a_{pj}\sin\theta + a_{qj}\cos\theta \quad (j \neq p,q), \\ b_{ip} &= a_{ip}\cos\theta - a_{iq}\sin\theta = b_{pi} \quad (i \neq p,q), \\ b_{iq} &= a_{ip}\sin\theta + a_{iq}\cos\theta = b_{qi} \quad (i \neq p,q), \\ b_{pp} &= a_{pp}\cos^2\theta - 2a_{pq}\cos\theta\sin\theta + a_{qq}\sin^2\theta, \\ b_{qq} &= a_{pp}\sin^2\theta + 2a_{pq}\cos\theta\sin\theta + a_{qq}\cos^2\theta = a_{pp} + a_{qq} - b_{pp}, \\ b_{pq} &= (a_{pp} - a_{qq})\cos\theta\sin\theta + a_{pq}(\cos^2\theta - \sin^2\theta), \\ b_{qp} &= b_{pq}. \end{aligned}$$

4 QR 分解

4.1 QR 分解の定義

$\mathbf{R}^n$ のベクトル列 $a_1, \cdots, a_n$ が 1 次独立であるとき、Gram-Schmidt の直交化法、つまり $k = 1, 2, \cdots, n$ の順に

$$b_k := a_k - \sum_{j=1}^{k-1}(a_k, q_j), \quad q_k := \frac{b_k}{\|b_k\|}$$

と計算することで正規直交基底 $q_1, \cdots, q_n$ が作れる。

$A =: (a_1, \cdots, a_n)$, $Q := (q_1, \cdots, q_n)$, $R := (r_{ij})$, $r_{jk} := (a_k, q_j)$ とおくと、

$$A = QR, \quad Q \text{ は実直交行列}, \quad R \text{ は対角成分が正である上三角行列} \tag{1}$$

と書ける。

この形の分解は一意的である。実際

$$A = Q_1 R_1 = Q_2 R_2$$

とすると、左から ${Q_2}^{-1} = {Q_2}^T$, 右から ${R_1}^{-1}$ (これは対角成分が正の上三角行列である) をかけて、

$$Q_2^T Q_1 = R_2 {R_1}^{-1}.$$

左辺は実直交行列の積として実直交行列 (長さを変えない変換だから) で、右辺は対角成分が正の上三角行列の積として対角成分が正の上三角行列である。簡単な計算で対角成分が正の上三角実直交行列は単位行列に他ならないことが分かる。

以上をまとめて次の命題を得る。

命題 4.1 (QR 分解) n 次正則行列 A に対して、

$$A = QR, \quad Q \text{ は実直交行列}, \quad R \text{ は対角成分が正の上三角行列}$$

を満たす Q, R が一意的に存在し、それは A の列ベクトルに対する Gram-Schmidt の直交化法により得られる。

(1) の形の分解を **QR 分解**と呼ぶ[1]。

[1] これを Schmidt 分解、あるいは Gram-Schmidt の分解と呼ぶことを提案する本もある (シャトラン [1], 齋藤 [9])。以下に見るように (1) の形の分解を得るアルゴリズムには色々あるので、個人的には、有名なアルゴリズムの名前に冠される “Gram-Schmidt” は適切ではないように感じている。

GramSchmidt の直交化による QR 分解

```
% GramSchmidt.m --- GramSchmidt()

% 使用例
% n=3; a=rand(n,n);
% [q r]=GramSchmidt(a)
% a-q*r

% A=(a1 a2 ...,an) の列ベクトルから正規直交基底
% q1, q2,..., qn を並べた Q を求める。
function [q, r] = GramSchmidt(A)
  [n n] = size(A);
  %
  q=zeros(n,n);
  r=zeros(n,n);
  for k=1:n
    b=A(:,k); % 第 k 列 a_k
    for j=1:k-1
      r(j,k)=q(:,j)'*A(:,k); % r_{jk} = a_k と q_j の内積
      b=b-r(j,k)*q(:,j);     % b_k = a_k-(a_k,q_j)q_j
    end
    r(k,k)=norm(b);  % r_{kk} = ||b_k||
    q(:,k)=b/r(k,k); % q_k = b / r_{kk}
  end
```

GramSchmidt.m の実行結果

```
>> n=4;a=rand(n,n)
a =
    0.9827    0.1146    0.5668    0.9616
    0.8066    0.6649    0.8230    0.0589
    0.7036    0.3654    0.6739    0.3603
    0.4850    0.1400    0.9994    0.5485
>> [q r]=GramSchmidt(a)
q =
    0.6415   -0.6491   -0.3412   -0.2251
    0.5266    0.7307   -0.1256   -0.4160
    0.4593    0.1624   -0.0214    0.8730
    0.3166   -0.1355    0.9313   -0.1185
r =
    1.5319    0.6358    1.4229    0.9870
         0    0.4517    0.2074   -0.5971
         0         0    0.6196    0.1676
         0         0         0    0.0086
>> a-q*r
ans =
   1.0e-15 *
         0         0         0    0.1110
         0         0         0   -0.0139
         0         0         0         0
    0.0555         0         0    0.1110
>>
```

注意 **4.2** 上の説明では R として対角成分がすべて正であるものを採用したが、応用上はあまり意味がないので、巷のソフトウェア (例えば MATLAB) の QR 分解用の関数 (手続き) の

仕様では、必ずしもそうはなっていない。 ■

4.2 修正 Gram-Schmidt の直交化法

実は前小節にあげた Gram-Schmidt の直交化法は、浮動小数点演算システムで計算する場合、丸め誤差の観点からは良くない (と言われている[2])。代りに次にあげる修正 Gram-Schmidt の直交化法が利用される。

修正 **Gram-Schmidt** の直交化法による **QR** 分解

```
 1  % ModifiedGramSchmidt.m --- ModifiedGramSchmidt()
 2
 3  % 使用例
 4  % n=3; a=rand(n,n);
 5  % [q r]=ModifiedGramSchmidt(a)
 6  % a-q*r
 7
 8  % A=(a1 a2 ...,an) の列ベクトルから正規直交基底
 9  % q1, q2,..., qn を並べた Q を求める。
10  function [q,r] = ModifiedGramSchmidt(A)
11    [n n] = size(A);
12    %
13    q=zeros(n,n); r=zeros(n,n);
14    for j=1:n
15      r(j,j)=norm(A(:,j)); % 第 k 列 a_k
16      q(:,j)=A(:,j)/r(j,j);
17      for k=j+1:n
18        r(j,k)=q(:,j)'*A(:,k); % r_{jk} = a_k と q_j の内積
19        A(:,k)=A(:,k)-r(j,k)*q(:,j);
20      end
21    end
```

[2] øA.Björk, Solving linear least squares problems by Gram-Schmidt orthogonalization, BIT, Vol.7 (1967), pp.1–21.

ModifiedGramSchmidt.m の実行結果

```
>> n=4;a=rand(n,n)
a =
    0.6273    0.6552    0.5947    0.7764
    0.6991    0.8376    0.5657    0.4893
    0.3972    0.3716    0.7165    0.1859
    0.4136    0.4253    0.5113    0.7006
>> [q r]=ModifiedGramSchmidt(a)
q =
    0.5700   -0.2729   -0.7192   -0.2886
    0.6352    0.7226    0.2609   -0.0788
    0.3609   -0.5861    0.6431   -0.3356
    0.3759   -0.2447    0.0323    0.8932
r =
    1.1005    1.1995    1.1492    1.0839
         0    0.1046   -0.2985   -0.1386
         0         0    0.1972   -0.2886
         0         0         0    0.3008
>> a-q*r
ans =
   1.0e-15 *
         0         0   -0.1110         0
         0         0         0         0
         0         0         0   -0.0555
         0         0         0         0
>>
```

(「良くない」のはなぜか？本当にそうなのか、どこかでチェックしたい。)

4.3 Givens 変換による QR 分解

(説明を書く暇がないのでとりあえず算法だけでも)

Givens 変換による QR 分解

```
% QR_Givens.m --- QR_Givens()
% 注意: このプログラムは原理の説明用であり、計算の効率は非常に低い

% 使用例
% n=3; a=rand(n,n); [q r]=QR_Givens(a)
% norm(a-q*r)
% norm(q*q')

% A=(a1 a2 ...,an) の列ベクトルから正規直交基底
% q1, q2,..., qn を並べた Q を求める。
function [Q,R] = QR_Givens(A)
  [n n] = size(A);
  R=A; Q=eye(n,n);
  %
  for q=1:n-1
    for p=q+1:n
      d=sqrt(R(p,q)*R(p,q)+R(q,q)*R(q,q));
      c=R(q,q)/d;
      s=R(p,q)/d;
      G=Givens(n,p,q,c,s); % xp, xq 平面で θ の回転 (c=cos θ , s=sin θ )
      R=G'*R;
      Q=Q*G;
    end
  end
```

QR_Givens.m の実行結果

```
>> n=4;a=rand(n,n)
a =
    0.6273    0.6552    0.5947    0.7764
    0.6991    0.8376    0.5657    0.4893
    0.3972    0.3716    0.7165    0.1859
    0.4136    0.4253    0.5113    0.7006
>> [q r]=QR_Givens(a)
q =
    0.5700   -0.2729   -0.7192   -0.2886
    0.6352    0.7226    0.2609   -0.0788
    0.3609   -0.5861    0.6431   -0.3356
    0.3759   -0.2447    0.0323    0.8932
r =
    1.1005    1.1995    1.1492    1.0839
   -0.0000    0.1046   -0.2985   -0.1386
    0.0000    0.0000    0.1972   -0.2886
    0.0000   -0.0000    0.0000    0.3008
>> a-q*r
ans =
   1.0e-15 *
         0    0.1110    0.1110    0.1110
         0    0.1110    0.1110    0.0555
   -0.0555         0         0   -0.0278
   -0.0555    0.0555    0.1110         0
>>
```

4.4 Householder 変換による QR 分解

算法の基本的な考え方は、Householder 変換による実対称行列の三重対角化と同じである。

(2) $A_0 := A,$

(3) $A_{k+1} := U_k A_k, \quad U_k := I - 2\boldsymbol{u}_k \boldsymbol{u}_k^T \quad (k = 0, 1, \ldots, n-2)$

で $\{A_k\}_{k=0}^{n-1}$ を定めたとき (ただし $\boldsymbol{u}_k$ はこれから後で定める単位ベクトル)、A_k は “k 列だけ上三角”、すなわち

$$A_k = \left(\begin{array}{ccc|c} \alpha_1 & & * & \\ & \ddots & & B_k \\ \mathbf{0} & & \alpha_k & \\ \hline & O & & C_k \end{array}\right)$$

という形にすることを目指す。もしこれができれば

$$UA = R, \quad U := U_{n-2} \cdots U_1 U_0, \quad R := A_{n-1}$$

で、U は実直交行列、R は上三角行列となるので、

$$A = QR, \quad Q := U^T$$

という QR 分解が得られる。

$\boldsymbol{u}_k$ を

$$\boldsymbol{u}_k = \left(\begin{array}{c} \mathbf{0} \\ \boldsymbol{v}_k \end{array}\right), \quad \mathbf{0} \in \mathbf{R}^k, \quad \boldsymbol{v}_k \in \mathbf{R}^{n-k}$$

の形に取ると、

$$U_k = \left(\begin{array}{c|c} I_k & O \\ \hline O & V_k \end{array}\right), \quad V_k := I_{n-k} - 2\boldsymbol{v}_k \boldsymbol{v}_k^T$$

という形なるので、

$$A_{k+1} = U_k A_k = \left(\begin{array}{ccc|c} \alpha_1 & & * & \\ & \ddots & & B_k \\ \mathbf{0} & & \alpha_k & \\ \hline & O & & V_k C_k \end{array}\right).$$

我々の目標を達成するには

$$V_k C_k = \left(\begin{array}{c|c} \alpha_{k+1} & \\ 0 & \\ \vdots & * \\ 0 & \end{array}\right)$$

という形をしていればよい。それには、

$$\boldsymbol{v}_k := \frac{1}{\|\boldsymbol{x}_k - \boldsymbol{y}_k\|}(\boldsymbol{x}_k - \boldsymbol{y}_k), \quad \boldsymbol{x}_k := C_k \text{ の第 1 列}, \quad \boldsymbol{y}_k := \begin{pmatrix} \alpha_{k+1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad \alpha_{k+1} = \pm \|\boldsymbol{x}_k\|$$

と取ればよい。

ここで $\pm$ となっているのは、どちらを選んでもよいが、通常丸め誤差を抑えるために (桁落ちがおきないように)、$\boldsymbol{x}_k$ の第 1 成分の符号と逆になるように取るべし、という意見がある。次のプログラムではそうしてあるが、その結果は MATLAB の組み込み関数である `qr()` の結果と一致した。

Householder 変換による QR 分解のサンプル・プログラム

```
% qrhouse.m --- Householder 法による QR 分解
% 試すには n=5;a=rand(n,n);a=(a+a')/2;qr_house(a)
function [q,r] = qrhouse(a)
  [n,n]=size(a);
  U=eye(n,n);
  for k=0:n-2
    x=a(k+1:n,k+1);
    alpha=-sign(x(1))*norm(x);
    x(1)=x(1)-alpha;
    v=x/norm(x);
    V=eye(n-k,n-k)-2*v*v';
    a(k+1:n,k+1:n)=V*a(k+1:n,k+1:n);
    U(k+1:n,:)=V*U(k+1:n,:);
  end
  q=U';
  r=a;
%end function
```

実行結果

```
>> n=5;a=rand(n,n);a=(a+a')/2
a =
    0.6273    0.7683    0.5569    0.5571    0.3849
    0.7683    0.3716    0.4683    0.7887    0.6153
    0.5569    0.4683    0.7764    0.6480    0.2756
    0.5571    0.7887    0.6480    0.7036    0.3125
    0.3849    0.6153    0.2756    0.3125    0.5668
>> [q r]=qrhouse(a)
q =
   -0.4739    0.2935    0.1766    0.5104   -0.6306
   -0.5804   -0.6970    0.3665   -0.2008    0.0519
   -0.4206   -0.1361   -0.7970    0.3050    0.2764
   -0.4209    0.4579   -0.1876   -0.7491   -0.1295
   -0.2908    0.4470    0.4051    0.2121    0.7117
r =
   -1.3238   -1.2876   -1.2151   -1.3813   -0.9518
    0.0000    0.5390    0.1514   -0.0125    0.0431
    0.0000   -0.0000   -0.3587   -0.1344    0.2448
   -0.0000    0.0000    0.0000   -0.1372    0.0431
    0.0000    0.0000         0   -0.0000    0.2283
>> [q2 r2]=qr(a)
q2 =
   -0.4739    0.2935    0.1766    0.5104   -0.6306
   -0.5804   -0.6970    0.3665   -0.2008    0.0519
   -0.4206   -0.1361   -0.7970    0.3050    0.2764
   -0.4209    0.4579   -0.1876   -0.7491   -0.1295
   -0.2908    0.4470    0.4051    0.2121    0.7117
r2 =
   -1.3238   -1.2876   -1.2151   -1.3813   -0.9518
         0    0.5390    0.1514   -0.0125    0.0431
         0         0   -0.3587   -0.1344    0.2448
         0         0         0   -0.1372    0.0431
         0         0         0         0    0.2283
>> norm(q-q2)
ans =
   3.2522e-15
>> norm(r-r2)
ans =
   1.2993e-15
>>
```

4.5 各方法の比較 (1) 素朴にやってみよう…

文献に書いてあることを理解して要約すべきだと思うが…MATLAB 時代の良いことは試すことは簡単にできるということだ。

compare.m

```
 1  function compare(n)
 2    a=rand(n,n);
 3  %% qr()
 4    fprintf('MATLAB の qr()\n');
 5    tic
 6    [q0 r0]=qr(a);
 7    toc
 8    fprintf('||Q R-A||=%e\n\n', norm(q0*r0-a));
 9  %%
10    fprintf(' 普通の Gram-Schmidt\n');
11    tic
12    [q1 r1]=GramSchmidt(a);
13    toc
14    fprintf('MATLAB の関数の結果との差: %e %e\n', norm(q1-q0), norm(r1-r0));
15    fprintf('||Q R-A||=%e\n\n', norm(q1*r1-a));
16  %%
17    fprintf(' 修正 Gram-Schmidt\n');
18    tic
19    [q2 r2]=ModifiedGramSchmidt(a);
20    toc
21    fprintf('MATLAB の関数の結果との差: %e %e\n', norm(q2-q0), norm(r2-r0));
22    fprintf('||Q R-A||=%e\n\n', norm(q2*r2-a));
23  %%
24    fprintf('Givens 変換による QR 分解\n');
25    tic
26    [q3 r3]=QR_Givens(a);
27    toc
28    fprintf('MATLAB の関数の結果との差: %e %e\n', norm(q3-q0), norm(r3-r0));
29    fprintf('||Q R-A||=%e\n\n', norm(q3*r3-a));
30  %%
31    fprintf('Householde 変換による QR 分解\n');
32    tic
33    [q4 r4]=qrhouse(a);
34    toc
35    fprintf('MATLAB の関数の結果との差: %e %e\n', norm(q4-q0), norm(r4-r0));
36    fprintf('||Q R-A||=%e\n\n', norm(q4*r4-a));
```

ここでは手抜きをして、誤差と言うよりも残差を計算している。そのうち誤差を計算できるようにしてみたいが、分解の一意性がないのでこまったな…

n	qr()	Gram-Schmidt	修正 Gram-Schmidt	Givens	Householder
10	2.65×10^{-15}	3.40×10^{-16}	3.36×10^{-16}	1.50×10^{-15}	3.19×10^{-15}
20	5.46×10^{-15}	9.26×10^{-16}	8.84×10^{-16}	3.85×10^{-15}	4.13×10^{-15}
40	1.10×10^{-14}	1.69×10^{-15}	1.73×10^{-15}	1.32×10^{-14}	1.41×10^{-14}
80	1.65×10^{-14}	3.30×10^{-15}	3.20×10^{-15}	2.63×10^{-14}	1.77×10^{-14}
160	4.10×10^{-14}	6.75×10^{-15}	6.37×10^{-15}	8.32×10^{-14}	1.18×10^{-13}
320	1.66×10^{-13}	1.34×10^{-14}	1.34×10^{-14}	$?? \times 10^{-14}$	$?? \times 10^{-13}$

少なくともこの実験結果だけからは、Gram-Schmidt の直交化法、修正 Gram-Schmidt の直交化法による結果は他の方法による結果よりも精度が良さそうである。MATLAB の qr() による結果はそれよりも悪い。MATLAB の qr() による結果は Householder 法による結果に近い。

Gram-Schmidt の直交化法と修正 Gram-Schmidt の直交化法による結果の差は少ない。その二つの差よりも、Givens 変換や Householder 変換を用いた方法との差の方がずっと大きい。普通の Gram-Schmidt の直交化法ではうまく解けないような条件の悪い問題を作って、それについて実験しないといけないのか？考え込まされる結果である。

4.6 各方法の比較 (2) お勉強してみよう

さて、自分でやってよく分からないということで、少し本をひもといてみることにしたら、最初にピンポンで、Trefethen and Bau III [12] に色々載っていました。それからぐぐってみたら、この本に書いてあることは今では世界の常識みたい[3]。この手のことを宿題でやらされている人がたくさんいるんだ…

今忙しいのでできないけれど... I shall return.

5 Jacobi 法

誰だったか忘れたが (戸川先生?)、Jacobi 法をモグラ叩きに例えた人がいる。これはなかなかイメージが湧いてきて良いと思う。

補題 5.1 (実行列の要素の平方和) $A = (a_{ij}) \in \mathbf{R}^{n\times n}$ について、

$$\sum_{i,j=1}^{n} {a_{ij}}^2 = \mathrm{tr}(A^T A).$$

証明

$$\begin{aligned}
\mathrm{tr}(A^T A) &= \sum_{i=1}^{n} A^T A \text{ の } (i,i) \text{ 要素} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} A^T \text{ の } (i,j) \text{ 要素 } A^T \text{ の } (j,i) \text{ 要素} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ji}a_{ij} \\
&= \sum_{i,j=1}^{n} {a_{ji}}^2 . \blacksquare
\end{aligned}$$

実行列 A を実直交行列 U で相似変換した $B = U^T AU$ について、

$$\mathrm{tr}(B^T B) = \mathrm{tr}(U^T A^T U \cdot U^T AU) = (U^T A^T AU)$$

[3] Trefethen and Bau III の本の実験の `mgs.m`, `clgs.m` という名前のプログラムが世の中には一杯あるみたい。

となるが、良く知られた公式 $\mathrm{tr}(AB) = \mathrm{tr}(BA)$ より [4]、

$$\mathrm{tr}(B^T B) = (A^T A U U^T) = \mathrm{tr}(A^T A)$$

であるから、

$$B \text{ の要素の平方和} = A \text{ の要素の平方和},$$

つまり

実直交行列による相似変換で要素の平方和は不変である。

補題 5.2 (Jacobi のもぐらたたき一発分の効果) $n \in \mathbf{N}$, $n \geq 2$, $A = (a_{ij})$ は n 次実対称行列、$p, q \in \{1, 2, \ldots, n\}$, $p \neq q$, $\theta \in \mathbf{R}$ とするとき、$U := G(n, p, q, \theta)$, $B = (b_{ij}) := U^T A U$ とおくと、次の (1), (2) が成り立つ。

(1) (Givens 変換した行列の成分と平方和の性質)

$$\begin{aligned}
b_{ij} &= a_{ij} \quad (i \neq p, q \text{ かつ } j \neq p, q) \\
b_{pj} &= a_{pj}\cos\theta - a_{qj}\sin\theta \quad (j \neq p, q) \\
b_{qj} &= a_{pj}\sin\theta + a_{qj}\cos\theta \quad (j \neq p, q) \\
b_{ip} &= a_{ip}\cos\theta - a_{iq}\sin\theta = b_{pi} \quad (i \neq p, q) \\
b_{iq} &= a_{ip}\sin\theta + a_{iq}\cos\theta = b_{qi} \quad (i \neq p, q) \\
b_{pp} &= a_{pp}\cos^2\theta - 2a_{pq}\cos\theta\sin\theta + a_{qq}\sin^2\theta \\
b_{qq} &= a_{pp}\sin^2\theta + 2a_{pq}\cos\theta\sin\theta + a_{qq}\cos^2\theta = a_{pp} + a_{qq} - b_{pp} \\
b_{pq} &= (a_{pp} - a_{qq})\cos\theta\sin\theta + a_{pq}(\cos^2\theta - \sin^2\theta) \\
b_{qp} &= b_{pq}
\end{aligned}$$

となるが、

$$\sum_{i,j=1}^{n} {a_{ij}}^2 = \sum_{i,j=1}^{n} {b_{ij}}^2 \quad (\text{これはもっと一般に成り立つ}),$$
$$\sum_{i \neq j} {a_{ij}}^2 - \sum_{i \neq j} {b_{ij}}^2 = 2({a_{pq}}^2 - {b_{pq}}^2).$$

(2) 特に

$$\theta = \frac{1}{2}\,\mathrm{Arctan}\left(\frac{-2a_{pq}}{a_{pp} - a_{qq}}\right)$$

と取ると、$b_{pq} = b_{qp} = 0$ かつ

$$\sum_{i \neq j} {a_{ii}}^2 = \sum_{i \neq j} {b_{ii}}^2 + 2{a_{pq}}^2.$$

[4] これから、相似変換で trace が不変であることが導かれる。今必要なのはそれではないけれど。

証明　単純な計算なので略する。■

定理 5.3 (古典 Jacobi 法の原理) A を実対称行列とする。$A = A_0$ とおき、以下次の手順で $A_1, A_2, \cdots$ を計算する。$A_k = (a_{ij}^{(k)}) = (a_{ij})$ の非対角要素のうちで、絶対値が最大の要素 a_{pq} に対して (絶対値最大の要素が二つ以上ある場合、任意の一つを選ぶ)、() で回転行列 U_{k+1} を定め、

$$A_{k+1} = {U_{k+1}}^T A_k U_{k+1}$$

とおく。すると、

$$\lim_{k\to\infty} A_k = D \quad (\text{対角行列})$$

となり、D の対角成分が A の固有値、

$$V \overset{\text{def.}}{=} \lim_{k\to\infty} V_k, \quad V_k \overset{\text{def.}}{=} U_1 U_2 \cdots U_k$$

の各列が固有ベクトルを与える。

この定理に基づき、A_k が十分対角行列に近づいたときに計算を打ち切り、A の近似固有値として ${a_{11}}^{(k)}, \cdots, {a_{nn}}^{(k)}$, それらに属する近似固有ベクトルとして V_k の各列を採用する。これを古典 **Jacobi** 法と呼ぶ。

有限ステップで停止したときの誤差評価として、次の定理がある。

定理 5.4 (近似固有値を非対角要素の平方和で評価) n 次実対称行列について、古典 Jacobi 法を適用して作った行列 $A_1, A_2, \cdots$ があるとき、$A_k = (a_{ij}^{(k)})$ の非対角要素の平方和を $F^{(k)}$ とすると、次の評価が成り立つ。

(1) A の固有値を適当に並べたものを $\lambda_1, \cdots, \lambda_n$ とするとき、

$$\left| a_{ii}^{(k)} - \lambda_i \right| \le \sqrt{F^{(k)}} \quad (i = 1, 2, \cdots, n).$$

(2) $N = n(n-1)/2$ とおくとき、

$$F^{(k+1)} \le \left(1 - \frac{1}{N}\right) F^{(k)} \quad (k = 1, 2, \cdots).$$

(3) ある定数 C が存在して、十分大きな k に対して、

$$F^{(k+N)} \le C (F^{(k)})^2.$$

証明　(1) は摂動定理を使うだけ。(2) は、まず (p, q) の選び方より

$$(a_{pq}^{(k)})^2 \ge \frac{F^{(k)}}{n(n-1)}$$

であり、

$$F^{(k)} - F^{(k+1)} = 2(a_{pq}^{(k)})^2 \ge \frac{2}{n(n-1)} F^{(k)} = \frac{1}{N} F^{(k)}.$$

(3) については省略。 ■

絶対値最大の要素の探索はかなり手間がかかる (計算量が大きくなる) ので、色々な変種が考えられている。ある「閾値」$\varepsilon_1 \geq \varepsilon_2 \geq \cdots$ と選び、$|a_{pq}^{(k)}| > \varepsilon_k$ なる $a_{pq}^{(k)}$ に対して回転を施す閾値 **Jacobi** 法や、

```
for p:=1 to n-1 do
  for q:=p+1 to n
    a_pq を叩く
```

を繰り返す特別巡回 **Jacobi** 法などがある。

注意 5.5 (Jacobi 法の評判) Jacobi 法の評判をいくつか引用しておく。

戸川 [11] (1971) から、

> 最近の十数年間に数値計算の “常識” がずいぶん変わった。ある時期まで “最良の方法” と考えられていたものが後退し、それに代わって新しい解法が登場した。その中でも、固有値計算の分野における世代交代は特に著しいものがあった。たとえば、ヤコビの方法は対称行列の固有値計算法の決定版といわれ、どこの計算センターでも標準的なサブルーチンとして使っていたが、最近の文献では “ヤコビの方法は、公式が比較的簡単であるという小さな利点を除けば、もはや歴史的価値しかない” と極論されるところまで転落した。

森・杉原・室田 [6] (1994) から、

> 実対称行列の固有値 (と固有ベクトル) を求める古典的な方法に **Jacobi** 法がある。20 年以上前には決定版のようにもてはやされたが、大規模行列は苦手という理由で、現在では (Rayleigh-Ritz の技法 (→ §6.5) と組み合わせて小規模な行列の固有値を求める場合を除いて) 実用的価値はあまりないと言われるようになってしまった。しかし、行列の次数が小さいなら十分実用であるし、固有値計算法の古典としての意義は大きい。

山本 [7] (2003) から、

> 古典 Jacobi 法と特別巡回 Jacobi 法は最終的には 2 次収束することが知られている。しかし、30×30 程度の行列でも、数千回の回転を要するのが普通で、次数の大きい行列に対しては、計算時間および精度の点で、次節以下に述べる Householder–Givens 法, QR 法の法がすぐれている (表 7.1 参照)。しかし、固有値・固有ベクトルがすべて同時に求まる点で、この方法は捨て難い。特に、固有ベクトルの直交性はよく保たれる。実際、近年、国外で Jacobi 法の安定性が再評価されはじめているのは興味深い。

5.1 Rutishauser の計算式

θ を `atan()` を用いて求めたりする必要はない。

$$
\begin{aligned}
z &= \frac{a_{qq}-a_{pp}}{2a_{pq}} \quad (=\cot 2\theta),\\
t &= \frac{\mathrm{sign}\,(z)}{|z|+\sqrt{1+z^2}} \quad (=\tan\theta),\\
c &= \frac{1}{\sqrt{1+t^2}} \quad (=\cos\theta),\\
s &= ct \quad (=\sin\theta),\\
u &= \frac{s}{1+c} \quad (=\tan\theta/2),\\
b_{pp} &= a_{pp}-ta_{pq},\\
b_{qq} &= a_{qq}+ta_{pq},\\
b_{pq} &= b_{qp}=0,\\
b_{pj} &= b_{jp}=a_{pj}-s(a_{qj}+ua_{pj}) \quad (j\neq p,q),\\
b_{qj} &= b_{jq}=a_{qj}+s(a_{pj}-ua_{qj}) \quad (j\neq p,q).
\end{aligned}
$$

5.2 MATLAB での実験

まず補題 5.2 (1) にあるように、与えられた (n 次) 実対称行列 A と $p,\ q\in\{1,2,\ldots,n\}$, $p\neq q$ に対して、$B:=U^TAU$, $U:=G(p,q,\theta)$ を計算する関数は次のようになる (θ を与える代りに $\cos\theta$, $\sin\theta$ を与えるようにしてある)。

jacobi0.m

```
% jacobi0.m
% Jacobi 法の基礎ステップ --- x_p,x_q 平面の角 θ の回転 (Givens 変換)
%  c=cos θ , s=sin θ
function b=jacobi0(a,p,q,c,s)
  b=a;
  b(p,:)=a(p,:)*c-a(q,:)*s;
  b(q,:)=a(p,:)*s+a(q,:)*c;
  b(:,p)=b(p,:)';
  b(:,q)=b(q,:)';
  b(p,p)=a(p,p)*c*c-2*a(p,q)*s*c+a(q,q)*s*s;
  b(q,q)=a(p,p)+a(q,q)-b(p,p);
  b(p,q)=(a(p,p)-a(q,q))*c*s+a(p,q)*(c*c-s*s);
  b(q,p)=b(p,q);
```

$b_{pq}=b_{qp}=0$ とするためには、補題 5.2 (2) にあるように θ を選ばねばならないが、これについては Rutishauser の計算式を一部利用すると、次のような関数を得る。

jacobi1.m

```
 1  % jacobi1.m
 2  % Jacobi 法の1ステップ --- (p,q)成分を叩く (0 になるよう Given 変換をする)
 3  function b=jacobi1(a,p,q)
 4    % cos θ ,sin θ を得る --- Rutishauser の計算式
 5    z=(a(q,q)-a(p,p))/(2*a(p,q));   % cot 2 θ
 6    t=sign(z)/(abs(z)+sqrt(1+z^2)); % tan θ
 7    c=1/sqrt(1+t^2);                % cos θ
 8    s=c*t;                          % sin θ
 9    % u=s/(1+c); % tan(θ/2)
10    %
11    b=jacobi0(a,p,q,c,s);
12    % 丸め誤差により 0 になっていないのを強制的にクリア
13    b(p,q)=0;
14    b(q,p)=0;
```

特別巡回 Jacobi 法は次のようになる。

jacobi.m

```
 1  % jacobi.m
 2  % 巡回 Jacobi 法
 3  function lambda=jacobi(a,iter)
 4    [n,n]=size(a);
 5    for k=1:iter
 6      for p=1:n-1
 7        for q=p+1:n
 8          a=jacobi1(a,p,q);
 9        end
10      end
11      %a
12    end
13    lambda=sort(diag(a));
```

次の実験プログラムは、与えられた自然数 n に対して、n 次実対称行列 A を作り、繰り返しの数を変えながら巡回 Jacobi 法による近似固有値を求めて、それを信頼できる値 (MATLAB の eig の結果) と比較している。

jacobiexp.m

```
 1  % jacobiexp.m
 2  %
 3  function jacobiexp(n)
 4    % n 次実対称行列を生成
 5    a=rand(n,n);
 6    a=(a+a')/2;
 7    % チェック用に MATLAB の関数で固有値を求めておく
 8    eigen=eig(a);
 9    % タイマーをリセット
10    tic
11    for i=1:100
12      % jacobi() で i 回反復したときの近似固有値の精度を求める
13      error=norm(jacobi(a,i)-eigen);
14      % 結果を表示
15      fprintf('%d %e\n', i, error);
16      % 精度が十分ならば繰り返しをやめる
17      if (error < 1e-13)
18        break;
19      end
20    end
21    % 経過時間を表示
22    toc
```

実行結果

```
>> jacobiexp(5)
1 1.741441e-01
2 7.220988e-02
3 3.632924e-04
4 2.167315e-13
5 2.914633e-15
elapsed_time =
    0.1253
>> jacobiexp(10)
1 2.753837e-01
2 5.652731e-02
3 8.028047e-05
4 3.017277e-09
5 8.183436e-15
elapsed_time =
    0.5129
>> jacobiexp(20)
1 7.215286e-01
2 1.455863e-01
3 1.153771e-02
4 2.517012e-05
5 7.633347e-11
6 3.977614e-14
elapsed_time =
    6.6447
>>
```

6 Lanczos アルゴリズム

6.1 はじめに

与えられた実対称行列を三重対角行列に相似変換するための Lanczos アルゴリズムを理解したい。

ゆくゆくは CG 法や直交関数系の三項漸化式の話まで込めて議論したい。

6.2 クリロフ部分空間

定義 6.1 $A \in \mathbf{R}^{n\times n}$, $x \in \mathbf{R}^n$, $k \in \mathbf{N}$ に対して

$$\mathcal{K}_k(A,x) := \mathrm{Span}\{x, Ax, A^2x, \ldots, A^{k-1}x\}$$

とおき、A の k 次 **Krylov** 部分空間とよぶ。

混乱が生じないときは (A, x を固定して考えることが多いので) $\mathcal{K}_k(A,x)$ を $\mathcal{K}_k$ と略記する。

まず定義から容易に分かることをまとめておく。

命題 6.2 $A \in \mathbf{R}^{n\times n}$, $x \in \mathbf{R}^n$ として、任意の $k \in \mathbf{N}$ に対して $\mathcal{K}_k := \mathcal{K}_k(A,x)$ とおくとき、次の (1)–(4) が成り立つ。

(1) 任意の $k \in \mathbf{N}$ に対して $A\mathcal{K}_k \subset \mathcal{K}_{k+1}$.

(2) 任意の $k \in \mathbf{N}$ に対して $\mathcal{K}_k = \{f(A)x; f(x) \in \mathbf{R}[x],\ \deg f(x) \le k-1\}$.

(3) 任意の $k \in \mathbf{N}$ に対して $\mathcal{K}_k$ は $\mathbf{R}^n$ の線形部分空間である。また $\mathcal{K}_k$ は k につき単調増加である:

$$\mathcal{K}_1 \subset \mathcal{K}_2 \subset \cdots \mathcal{K}_j \subset \mathcal{K}_{j+1} \subset \cdots \subset \mathbf{R}^n.$$

(4) 任意の $k \in \mathbf{N}$ に対して

$$\dim \mathcal{K}_{k+1} = \begin{cases} \dim \mathcal{K}_k + 1 & (A^k x \notin \mathcal{K}_k \text{ のとき}) \\ \dim \mathcal{K}_k & (A^k x \in \mathcal{K}_k \text{ のとき}). \end{cases}$$

証明

(1)

$$\begin{aligned} A\mathcal{K}_k &= A\,\mathrm{Span}\{x, Ax, A^2x, \cdots, A^{k-1}x\} = \mathrm{Span}\{Ax, A^2x, A^3x, \cdots, A^kx\} \\ &\subset \mathrm{Span}\{x, Ax, A^2x, A^3x, \cdots, A^kx\} = \mathcal{K}_{k+1}. \end{aligned}$$

(2)

$$\begin{aligned}\mathcal{K}_k &= \{c_1x + c_2Ax + c_3A^2x + \cdots + c_kA^{k-1}x; (c_j) \in \mathbf{R}^k\} \\ &= \{(c_1I + c_2A + c_3A^2 + \cdots + c_kA^{k-1})x; (c_j) \in \mathbf{R}^k\} \\ &= \{f(A)x; f(x) \in \mathbf{R}[x],\ \deg f(x) \leq k-1\}.\end{aligned}$$

(3) Span であるから $\mathbf{R}^n$ の線形部分空間であることは明らかである。また $\mathcal{K}_k$ を生成する集合 $\{x, Ax, \ldots, A^{k-1}x\}$ 自体が k について単調増加であることから、$\mathcal{K}_k$ が k について単調増加であることも明らかである。

(4) 前項より $\dim \mathcal{K}_k \leq \dim \mathcal{K}_{k+1}$. $\mathcal{K}_{k+1} = \mathrm{Span}\{\mathcal{K}_k, A^kx\}$ であるから、$A^kx \in \mathcal{K}_k$ であれば $\mathcal{K}_{k+1} = \mathcal{K}_k$. $A^kx \notin \mathcal{K}_k$ であれば $\dim \mathcal{K}_{k+1} = \dim \mathcal{K}_k + 1$. ■

命題 6.3 $A \in \mathbf{R}^{n\times n}$, $x \in \mathbf{R}^n$, $x \neq 0$ とするとき、$\exists m \in \{1, 2, \ldots, n\}$ s.t.

$$\dim \mathcal{K}_j = j \quad (1 \leq j \leq m), \qquad \mathcal{K}_j = \mathcal{K}_m \quad (j > m).$$

証明　まず $\mathcal{K}_1 = \mathrm{Span}\{x\}$ であるから $\dim \mathcal{K}_1 = 1$. 命題 6.2 から $\exists m \in \mathbf{N}$ s.t.

$$\mathcal{K}_1 \subsetneq \mathcal{K}_2 \subsetneq \cdots \subsetneq \mathcal{K}_m = \mathcal{K}_{m+1}.$$

$A^mx \in \mathcal{K}_{m+1} = \mathcal{K}_m = \mathrm{Span}\{x, Ax, A^2x, \ldots, A^{m-1}x\}$. これから容易に $A^jx \in \mathcal{K}_m$ $(j \geq m)$ が導かれる。ゆえに $\forall j \geq m$ に対して $\mathcal{K}_j = \mathrm{Span}\{x, Ax, \ldots, A^{j-1}x\} \subset \mathcal{K}_m$. もちろん $\mathcal{K}_j \supset \mathcal{K}_m$ であるから $\mathcal{K}_j = \mathcal{K}_m$. ■

6.3 Lanczos 原理

以下、まずは $m = n$, すなわち

(4)
$$\mathcal{K}_n = \mathbf{R}^n$$

の場合を考える。

定理 6.4 $A \in \mathbf{R}^{n\times n}$, $x \in \mathbf{R}^n$ に対して $\mathcal{K}_n(A,x) = \mathbf{R}^n$ が成り立つとするとき、$\mathcal{K}_0 = \{0\}$ とおき、$k = 1, 2, \ldots, n$ の順に $\boldsymbol{v}_k \in \mathcal{K}_k \setminus \mathcal{K}_{k-1}$ を任意に取り、

$$\boldsymbol{w}_k := \boldsymbol{v}_k - \sum_{j=1}^{k-1} (\boldsymbol{v}_k, \boldsymbol{u}_j)\boldsymbol{u}_j,$$
$$\boldsymbol{u}_k := \frac{\boldsymbol{w}_k}{\|\boldsymbol{w}_k\|}$$

とおくと (要するに Gram-Schmidt の直交化法)、$\{\boldsymbol{u}_j\}_{j=1}^n$ は以下の (i), (ii), (iii) をみたす。

(i) $(\boldsymbol{u}_j, \boldsymbol{u}_k) = \delta_{jk}$ $(1 \le j, k \le n)$. ここで δ_{jk} は Kronecker のデルタである。

(ii) 任意の $k \in \{1, \ldots, n\}$ に対して $\{\boldsymbol{u}_j\}_{j=1}^k$ は $\mathcal{K}_k(A,x)$ の基底である。

(iii) $U := (\boldsymbol{u}_1\ \boldsymbol{u}_2\ \cdots\ \boldsymbol{u}_n)$ とおくと $U^TAU =: B$ は Hessenberg 行列で[a]、特に A が実対称行列である場合は三重対角行列になる。すなわち $\{\alpha_i\}_{i=1}^n$, $\{\beta_i\}_{i=1}^{n-1}$ が存在して、

$$U^TAU = \begin{pmatrix} \alpha_1 & \beta_1 & & & & & \\ \beta_1 & \alpha_2 & \beta_2 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \beta_{i-1} & \alpha_i & \beta_i & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \beta_{n-2} & \alpha_{n-1} & \beta_{n-1} \\ & & & & & \beta_{n-1} & \alpha_n \end{pmatrix}.$$

[a] $B = (\beta_{ij})$ が $j > i+1 \Longrightarrow \beta_{ij} = 0$ をみたすとき、B は Hessenberg 行列という。

証明　仮定 $\mathcal{K}_n(A,x) = \mathbf{R}^n$ より、$\dim \mathcal{K}_j(A,x) = j$ $(j = 1, 2, \ldots, n)$ が成り立つので、$\mathcal{K}_j \setminus \mathcal{K}_{j-1}$ は空でなく、Gram-Schmidt の直交化法により $\mathbf{R}^n$ の正規直交基底が得られる。また作り方から明らかに任意の k に対して $\boldsymbol{u}_k \in \mathcal{K}_k$, $\dim \mathcal{K}_k = k$ であるから、$\{\boldsymbol{u}_j\}_{j=1}^k$ は $\mathcal{K}_k$ の基底になっている。

さて、任意の j に対して、$\boldsymbol{u}_j \in \mathcal{K}_j$ であるので、$A\boldsymbol{u}_j \in \mathcal{K}_{j+1}$ であるから、$A\boldsymbol{u}_j$ は $\boldsymbol{u}_1$, $\boldsymbol{u}_2$, $\ldots$, $\boldsymbol{u}_{\min\{j+1,n\}}$ の線形結合になる。すなわち

$$A\boldsymbol{u}_j = \sum_{k=1}^{\min\{j+1,n\}} \beta_{jk}\boldsymbol{u}_k$$

をみたす β_{jk} $(k = 1, 2, \ldots, \min\{j+1, n\})$ が存在する。$k > j+1$ に対して $\beta_{jk} := 0$ とおいて、行列 $B := (\beta_{jk})$ を定めると、

$$AU = UB.$$

$\{\boldsymbol{u}_j\}$ が正規直交基底であることから U は実直交行列で $U^T = U^{-1}$. これを上の等式の両辺に左からかけて $U^TAU = B$. もしも A が実対称であれば、

$$B^T = (U^TAU)^T = U^TA^T(U^T)^T = U^TAU = B.$$

すなわち B も対称である。対称な Hessenberg 行列は三重対角行列に他ならない。 ■

注意 6.5 この定理の $\{\boldsymbol{u}_j\}$ は次の意味で「符号をのぞき一意的」である。すなわち $\boldsymbol{v}_j$ の取り方に任意性があるようだが、$\mathcal{K}_j$ の中で $\mathcal{K}_{j-1}$ に直交する方向は 1 次元しかないので、$\boldsymbol{v}_j$ をどう取り直しても結果として得られるものは $\boldsymbol{u}_j$, $-\boldsymbol{u}_j$ のいずれか一方である。 ■

6.4 都合が「よい」状況下での Lanczos 法

A を n 次の実対称行列で、$x \in \mathbf{R}^n$ に対して、「都合よく」

$$\mathcal{K}_n(A, x) = \mathbf{R}^n$$

が成り立つと仮定する。前節の定理から、

$$U^T A U = \begin{pmatrix} \alpha_1 & \beta_1 & & & & & \\ \beta_1 & \alpha_2 & \beta_2 & & & & 0 \\ & \ddots & \ddots & \ddots & & & \\ & & \beta_{i-1} & \alpha_i & \beta_i & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \beta_{n-2} & \alpha_{n-1} & \beta_{n-1} \\ 0 & & & & & \beta_{n-1} & \alpha_n \end{pmatrix}$$

をみたす実直交行列 $U = (\boldsymbol{u}_1\ \boldsymbol{u}_2\ \cdots\ \boldsymbol{u}_n)$ が存在する。これから

$$\begin{aligned} A\boldsymbol{u}_1 &= \alpha_1\boldsymbol{u}_1 + \beta_1\boldsymbol{u}_2, \\ A\boldsymbol{u}_2 &= \beta_1\boldsymbol{u}_1 + \alpha_2\boldsymbol{u}_2 + \beta_2\boldsymbol{u}_3, \\ &\vdots \\ A\boldsymbol{u}_i &= \beta_{i-1}\boldsymbol{u}_{i-1} + \alpha_i\boldsymbol{u}_i + \beta_i\boldsymbol{u}_{i+1}, \\ &\vdots \\ A\boldsymbol{u}_{n-1} &= \beta_{n-2}\boldsymbol{u}_{n-2} + \alpha_{n-1}\boldsymbol{u}_{n-1} + \beta_{n-1}\boldsymbol{u}_n, \\ A\boldsymbol{u}_n &= \beta_{n-1}\boldsymbol{u}_{n-1} + \alpha_n\boldsymbol{u}_n \end{aligned} \tag{5}$$

という条件が導かれる。また

$$\boldsymbol{u}_1 = \pm\frac{1}{\|x\|}x \tag{6}$$

である。実は $\{\alpha_i\}$, $\{\beta_i\}$, $\{\boldsymbol{u}_j\}$ はこの条件 (5), (6) だけで定めることができる。

最初の式と $\boldsymbol{u}_1$ との内積を取ると

$$\alpha_1 = \boldsymbol{u}_1^T A\boldsymbol{u}_1.$$

$\boldsymbol{v}_2 := A\boldsymbol{u}_1 - \alpha_1\boldsymbol{u}_1$ とおくと、$\beta_1\boldsymbol{u}_2 = \boldsymbol{v}_2$ であるから、

$$|\beta_1| = \|\boldsymbol{v}_2\|.$$

ゆえに

$$\beta_1 = \pm\|\boldsymbol{v}_2\|, \quad \boldsymbol{u}_2 := \frac{1}{\beta_1}\boldsymbol{v}_2 = \pm\frac{1}{\|\boldsymbol{v}_2\|}\boldsymbol{v}_2 \quad \text{(複号同順)}.$$

(5) の第 2 式と $\boldsymbol{u}_2$ との内積を取ると、

$$\alpha_2 = \boldsymbol{u}_2^T A\boldsymbol{u}_2.$$

$\boldsymbol{v}_3 := A\boldsymbol{u}_2 - \beta_1\boldsymbol{u}_1 - \alpha_2\boldsymbol{u}_2$ とおくと、$\beta_2\boldsymbol{u}_3 = \boldsymbol{v}_3$ であるから、

$$|\beta_2| = \|\boldsymbol{v}_3\|.$$

ゆえに

$$\beta_2 = \pm\|\boldsymbol{v}_3\|, \quad \boldsymbol{u}_3 := \frac{1}{\beta_2}\boldsymbol{v}_3 = \pm\frac{1}{\|\boldsymbol{v}_3\|}\boldsymbol{v}_3 \quad \text{(複号同順)}.$$

以下同様にして...

6.5 都合が「よくない」場合の Lanczos 法

A を n 次実対称行列とする。前節では

$$\mathcal{K}_n(A, x) = \mathbf{R}^n$$

を満たす $x \in \mathbf{R}^n$ が得られている場合に A の三重対角化を行ったが、この条件はそれほど安易に成り立つものではない。

任意の x に対して $\mathcal{K}_n(A, x) \subsetneq \mathbf{R}^n$ となる場合　A の最小多項式 $\varphi(x) = x^r + \cdots$ の次数 r が n より小さい場合は、

$$A^r = \sum_{j=1}^{r} c_j A^{j-1}$$

をみたす $\{c_j\}$ が存在する。すると任意の $x \in \mathbf{R}^n$ に対して

$$A^r x = \sum_{j=1}^{r} c_j A^{j-1} x \in \mathcal{K}_r(A, x).$$

したがって $\mathcal{K}_{r+1}(A, x) = \mathcal{K}_r(A, x)$. ゆえに $\dim \mathcal{K}_n(A, x) \leq r < n$ となる。従って $\mathcal{K}_n(A, x) = \mathbf{R}^n$ をみたす x は存在しない。

うまく x を取ると $\mathcal{K}_n(A, x) = \mathbf{R}^n$ となる場合であっても　一方 A の最小多項式の次数 r が n に等しい場合は、$\mathcal{K}_n(A, x) = \mathbf{R}^n$ をみたす x が存在することがわかるが (証明？)、実際にどうやってそのような x を見つけるか、あまり簡単ではない。極端な話、x を A の固有ベクトルに選んでしまった場合は $\dim \mathcal{K}_n(A, x) = 1$ であり、Lanczos 法の反復は 1 ステップで止まってしまう。もっと極端な話、$A = \lambda I$ の場合、いかなる $x \neq 0$ も A の固有ベクトルになってしまう。おそらく最小多項式の次数が n である場合は、generic に $\mathcal{K}_n(A, x) = \mathbf{R}^n$ となるのであろうが…

一発でうまくやることはあきらめて　ここでは A の最小多項式の問題には触れないことにする。最初に選んだ $x \neq 0$ に対して $\mathcal{K}_n(A,x) \neq \mathbf{R}^n$ と仮定する。$m := \dim \mathcal{K}_n(A,x)$ とおくと、$\mathcal{K}_m(A,x) = \mathcal{K}_n(A,x)$ である。$\mathcal{K}_m(A,x)$ のことを単に $\mathcal{K}_m$ と書くことにする。

明らかに $\mathcal{K}_m$ は A 不変、すなわち

$$A\mathcal{K}_m \subset \mathcal{K}_m$$

をみたすが、$\mathcal{K}_m$ の直交補空間

$$\mathcal{K}_m{}^{\perp} := \{y \in \mathbf{R}^n; \forall z \in \mathcal{K}_m \ z^T y = 0\}$$

も A 不変である。実際 $y \in \mathcal{K}_m^{\perp}$ とするとき、任意の $z \in \mathcal{K}_m$ に対して、

$$(Ay, z) = (y, Az) = 0 \quad (\mathcal{K}_m \text{ の } A \text{ 不変性より } Az \in \mathcal{K}_m \text{ なので})$$

がなりたつので $Ay \in \mathcal{K}_m^{\perp}$ となり、$\mathcal{K}_m^{\perp}$ が A 不変であることが分かる。

老婆心: $y \in \mathcal{K}_n(A,x)^{\perp}$ とするとき、任意の $j \in \mathbf{N}$ に対して $A^j y \in \mathcal{K}_n(A,x)^{\perp}$ であるから、$\mathcal{K}_n(A,y) \subset \mathcal{K}_n(A,x)^{\perp}$.

そこで最初に選んだ $x_1 \neq 0$ に対して、$m := \dim \mathcal{K}_n(A,x_1) < n$ であった場合は、まず $\mathcal{K}_n(A,x_1)$ の正規直交基底を取り、その後で $x_2 \in \mathcal{K}_n(A,x_1)^{\perp}$, $x_2 \neq 0$ から始めて、$\mathcal{K}_n(A,x_2) \subset \mathcal{K}_n(A,x_1)^{\perp}$ の正規直交基底を取り、以下この操作を続けていけば、最後には

$$\mathbf{R}^n = \mathcal{K}_n(A,x_1) \oplus \mathcal{K}_n(A,x_2) \oplus \cdots \oplus \mathcal{K}_n(A,x_r)$$

とできる。それと並行して、$\mathcal{K}_n(A,x_j)$ の正規直交基底 $\boldsymbol{u}_k^j$ $(k = 1,2,\ldots,m_j)$ が得られる。この正規直交基底を

$$\boldsymbol{u}_1^1, \boldsymbol{u}_2^1, \cdots, \boldsymbol{u}_{m_1}^1, \boldsymbol{u}_1^2, \boldsymbol{u}_2^2, \cdots, \boldsymbol{u}_{m_2}^2, \cdots, \boldsymbol{u}_1^r, \boldsymbol{u}_2^r, \cdots, \boldsymbol{u}_{m_r}^r$$

と並べたものを $\{\boldsymbol{u}_j\}_{j=1}^n$ として、$U := (\boldsymbol{u}_1\ \boldsymbol{u}_2\ \cdots\ \boldsymbol{u}_n)$ とおくと、

$$U^T A U = B = \begin{pmatrix} B_1 & & & \\ & B_2 & & \\ & & \ddots & \\ & & & B_r \end{pmatrix}, \quad B_j = \begin{pmatrix} \alpha_1^j & \beta_1^j & & \\ \beta_1^j & \alpha_2^j & \beta_2^j & \\ & \ddots & \ddots & \ddots \\ & & \beta_{m_j-1}^j & \alpha_{m_j}^j \end{pmatrix}.$$

こうして得られた B は確かに三重対角行列である。

余談 **6.1** 三重対角行列

$$T = \begin{pmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2} & a_{n-1} & b_{n-1} \\ & & & b_{n-1} & a_n \end{pmatrix}$$

の固有値の計算法の一つである二分法の適用には、条件 $b_j \neq 0$ $(j = 1,2,\ldots,n-1)$ が必要であり、$b_j = 0$ となる要素がある場合は行列をブロック分けして処理する、ということになっているが、Lanczos 法では b_j が 0 になるかどうかは三重対角化の際にチェックできること、というかチェックしなければならないわけである。 ■

素朴な疑問をぶつぶつと　とはいえ、ここで説明した方法を実行するには、$\{\boldsymbol{u}_k\}$ を記憶しておく必要がある。本当に Lanczos 法が必要とされる大規模な計算でそれが実行できるものなのだろうか？どうもうまく行かなければ $\boldsymbol{u}_1$ を取り直して再計算と考えている人達がいるような気がするが、それは $\dim \mathcal{K}_n = \mathbf{R}^n$ が成り立つような問題ばかりを相手にしているからか？■

6.6　そのほか

7　摂動定理と分離定理

森・杉原・室田 [6] の演習問題から

定理 7.1 () $A = (a_{ij})$, $B = (b_{ij})$ を実対称行列とし、$C = A - B$ とおく。それぞれ固有値の大きい方から r 番目のものを $\lambda(\cdot)$ とするとき、次が成立する。

(1) $\lambda_r(B) + \lambda_n(C) \leq \lambda_r(A) \leq \lambda_r(B) + \lambda_1(C)$.

(2) $|\lambda_r(A) - \lambda_r(B)| \leq \left(\sum_{i=1}^{n} \sum_{j=1}^{n} (a_{ij} - b_{ij})^2 \right)^{1/2}$.

山本 [7] から

定理 7.2 (摂動定理) $A = (a_{ij})$, $B = (b_{ij})$ は n 次実対称行列で、それぞれ固有値 $\lambda_1 \leq \cdots \leq \lambda_n$, $\mu_1 \leq \cdots \leq \mu_n$ を持つものとすれば、

$$|\lambda_i - \mu_i| \leq \|A - B\|_F \leq \sqrt{\sum_{i,j=1}^{n} (a_{ij} - b_{ij})^2} \quad (i = 1, 2, \cdots, n).$$

この定理の一般化に Hoffman-Wielandt (1953)

$$\sum_{i=1}^{n} |\lambda_i - \mu_i|^2 \leq \|A - B\|_E^2 = \sum_{i=1}^{n} \nu_i^2 \quad (\nu_1, \cdots, \nu_n \text{ は } A - B \text{ の固有値}).$$

定理 7.3 (分離定理) A を n 次実対称行列, B を A の第 n 行, 第 n 列を除いて得られる $n-1$ 次行列とする ($n > 1$ と仮定する)[a]。 A, B の固有値をそれぞれ $\lambda_1 \leq \cdots \leq \lambda_n$, $\mu_1 \leq \cdots \leq \mu_n$ とすれば、

$$\lambda_1 \leq \mu_1 \leq \lambda_2 \leq \mu_2 \leq \cdots \leq \lambda_{n-1} \leq \mu_{n-1} \leq \lambda_n.$$

[a] MATLAB 風に書けば、$B = A(1 : n-1, 1 : n-1)$.

8 min-max principle

これは結構覚えにくい上に、本によって色々な形があって、混乱しがちである。まとめてみた (結局は伊理先生 [2] のまとめ方に近い)。

人名は本に載っていたものだが、根拠のあるものかどうかは分からない (出典がばらばらなので)。

定理 8.1 (min-max principle) A を n 次実対称行列、その n 個の固有値 (重複度の分だけ並べる) を $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ として、$R_A(x) := \dfrac{x^T Ax}{x^T x}$ とおくとき、次の (1)-(4) が成り立つ。

(1) (Fischer-Poincaré)

$$\min_{\dim V = k} \max_{x \in V \setminus \{0\}} R_A(x) = \text{小さい方から } k \text{ 番目の固有値} = \lambda_{n-k+1} \quad (1 \leq k \leq n).$$

(2)

$$\max_{\dim V = k} \min_{x \in V \setminus \{0\}} R_A(x) = \text{大きい方から } k \text{ 番目の固有値} = \lambda_k \quad (1 \leq k \leq n).$$

(3) (Courant-Fischer)

$$\min_{\dim S = j-1} \max_{x \in S^\perp \setminus \{0\}} R_A(x) = \text{大きい方から } j \text{ 番目の固有値} = \lambda_j \quad (1 \leq j \leq n).$$

(4) (Courant-Weyl)

$$\max_{\dim S = j-1} \min_{x \in S^\perp \setminus \{0\}} R_A(x) = \text{小さい方から } j \text{ 番目の固有値} = \lambda_{n-j+1} \quad (1 \leq j \leq n).$$

9 一般化固有値問題

別に、桂田 [13] というノートを用意してある。

9.1 正則な行列束に関する一般化固有値問題の定義

$A, B \in \mathbf{C}^{n \times m}$ とするとき、

$$\{A - \lambda B; \lambda \in \mathbf{C}\}$$

を行列束と呼ぶ。

行列束 $\{A - \lambda B\}$ が正則であるとは、A と B が同じ次数の正方行列であり、かつ

$$\exists \lambda \in \mathbf{C} \quad \text{s.t.} \quad \det(A - \lambda B) \neq 0$$

が成り立つことと定義する。そうでないとき、行列束は特異であると定義する。

正則な行列束 $\{A-\lambda B\}$ に対して、

$$Ax=\lambda Bx,\quad x\neq 0$$

を満たす x が存在するような λ の値と、そのときの x を求める問題を一般化された固有値問題と呼ぶ。

$$\mathrm{sp}[A,B]\overset{\text{def.}}{=}\{\lambda\in\mathbf{C};\det(A-\lambda B)=0\}.$$

とおき、sp $[A,B]$ の元を、行列束の固有値と呼ぶ。

応用上は A, B が対称で、特に B が半正定符号であることが多い。

9.2 Courant-Fischer の min-max principle

定理 9.1 A, B を n 次 Hermite 行列、B は正定値とする。(A,B) の固有値の大きいものから r 番目のものを $\lambda_r(A,B)$ とするとき、

$$\lambda_r(A,B)=\max_{\dim S=r}\min_{x\in S\setminus\{0\}}\frac{x^*Ax}{x^*Bx}=\max_{\dim S=n-r+1}\max_{u\in S\setminus\{0\}}\frac{u^*Au}{u^*Bu}.$$

9.3 2 分法の一般化固有値問題への拡張

A を実対称, B を正値対称行列とするとき、

$$Ax=\lambda Bx,\quad x\neq 0$$

を満たす λ, x を求めよ、という一般化固有値問題を考える。

(1) 固有値 λ はすべて実数である。

(2) (A,B) の固有値の正のものの個数 $\widehat{\pi}(A,B)$ は $\pi(A)$ に等しい。

(3) 任意に与えた実数 σ に対して、$A-\sigma B=LDL^T$ と LDL^T 分解すれば、D の対角要素のうち正, 0, 負のものの個数が、σ より大きい、等しい、小さい固有値 λ の個数を与える。

10 Shur 分解

任意の $A\in\mathbf{C}^{n\times n}$ に対して、unitary 行列 U と上三角行列 S が存在して、

$$U^HAU=S.$$

ここで U^H は U の Hermite 共役 (共役転置) $\overline{U}^T$ を表わす。これを A の **Schur** 分解という。

(事実自体は、線形代数の教科書、例えば佐武 [10] IV§3 にも書いてあるが、Schur 分解という名前は冠されていないことが多いようである。)

$U = (u_1, u_2, \cdots, u_n)$ の列ベクトル u_j を **Schur** ベクトルと呼ぶ。

A が Hermite 行列ならば、固有ベクトルの直交性により、S が対角行列になり、u_j は A の固有ベクトルになる。

実行列 A に対しては、U を実直交行列 Q に制限し、S を対角ブロックの大きさが 2 以下のブロック上三角行列[5] に緩和した形の実 Schur 分解

$$Q^T A Q = S$$

が存在する (実 Schur 分解)。S の対角ブロックの固有値が A の固有値である。

(実) Schur 分解の存在は数学的に保証されているが、それを有限回の四則と開平演算で計算することはできない

Hermite 対称でもないのに、固有ベクトルではなく Schur ベクトルを計算するのは、これが数値的に安定に計算できるからだという。

MATLAB では `schur()` という関数が用意されている。`[u,t]=schur(a)` として `u*t*u'-a` を計算すると…

11 Sylvester の慣性律, 2 次形式の対角化

(この節は下書きです。)

数値線形代数というよりは、線形代数の話題かもしれないが…

11.1 本題に入る前に

自分自身が最初のうち景色が見えなかったので、色々書いてみる。

Sylvester の慣性律とは、2 次形式を平方完成したときの係数の符号が、平方完成のやり方 (それはたくさんある) によらずに決まる、という事実を定式化したものである。

平方完成のことを「2 次形式の対角化」ともいう。それは議論を行列の言葉で書くことが出来て、与えられた対称行列 (2 次形式の係数行列) A に対して、適当な正則行列 P を見つけて、$P^T AP$ を対角行列にする操作に相当しているからである。

この変換 $A \mapsto P^T AP$ は、相似変換 $A \mapsto P^{-1} AP$ と似ているが、一応は別物であることに注意する (前者は正則な線形変数変換によるもので、後者は基底の取り替えによるものである)。ややこしいのは、対称行列の固有値問題では、直交行列 ($P^T = P^{-1}$ を満たす行列) で対角化するため、両者が重なってしまうことである (個人的にはかなり混乱したところ)。

何に使われるかについても、あらかじめ、少しは知っておいた方が良いかもしれない。

大学 1,2 年次に遭遇する適用例としては、多変数関数の極値問題を微分法で扱う際に、2 階までの Taylor 展開をしたとき、2 次の項が 2 次形式になっている、というのがある (大抵の微積のテキストには書いてあるが、例えば桂田 [4])。f が n 変数関数で、内点 a で極値を取るには、$f'(a) = 0$ が必要で、そのとき

$$f(a+h) = f(a) + h \text{ の 2 次形式} + O(\|h\|^2) \quad (h \to 0).$$

[5] 最初にこれを聞いたとき、Hessenberg 行列のことかと思ったが、違う!

そこで2次形式の符号に興味が出て来るわけである。どういう場合があって、どうやれば判定できるか。他にも Morse の Lemma などで、関数の「様子」を理解するために使われる。

数値線形代数においては、固有値を求めるための2分法というアルゴリズムが、Sylvester の慣性律で理解出来る、という応用もある。

もう少し式を使って具体的な話として書いてみる。

「2次形式の対角化」は、次の二つの同値な問題である。

(a) (平方完成) 2次形式 $A[x] = \sum_{i,j=1}^{n} a_{ij} x_i x_j$ が与えられたとき、

$$A[x] = \sum_{j=1}^{p} \alpha_j y_j^2 - \sum_{k=1}^{q} \alpha_{p+k} y_{p+k}^2 \quad (\text{ここで } \alpha_j > 0)$$

となる正則線形変換 $x = Py$ を求める問題

(b) (行列の「対角化」) 実対称行列 A が与えられたとき、

$$P^T A P = D,$$
$$D := \mathrm{diag}(\alpha_1, \cdots, \alpha_p, -\alpha_{p+1}, \cdots, -\alpha_{p+q}, 0, \cdots, 0) \quad (\text{ここで } \alpha_j > 0)$$

を満たす正則行列 P を求める問題

p, q は P によらず A だけで定まる。さらに実は $p + q = \mathrm{rank}\, A$ である。

P として一般の正則変換を許せば (つまり $P \in GL(n; \mathbf{R})$)、$\alpha_j = 1$ $(j = 1, \cdots, p+q)$ と出来る。

P として直交変換に限っても対角化は可能で (もちろん p, q も変わらない)、そのときは D の対角成分 $\alpha_1, \cdots, \alpha_p, -\alpha_{p+1}, \cdots, -\alpha_{p+q}, 0, \cdots, 0$ は A の固有値である。

三角行列の積の形に分解する話との関係。

- もしも $\delta_k := \det A_k \neq 0$ $(k = 1, \cdots, n)$ ならば、$L := P^{-T}$ $(:= (P^{-1})^T)$ は単位下三角行列に取れて、
$$A = LDL^T$$
となる (A の LDL^T 分解)。L は例えば Gauss の消去法などのアルゴリズムで計算出来る。

- (さらに) もしも $\delta_k > 0$ $(k = 1, \cdots, n)$ ならば、$\sqrt{D}$ が意味を持つので、$L\sqrt{D}$ を新たに L と置き直すことで
$$A = LL^T$$
となる (A の Cholesky 分解)。

$\det A_k \neq 0$ $(k = 1, \cdots, n)$ でない場合にどうするかについて述べてみよう。その後で何をしたいかによってやることが異なる、と理解すべきである。

連立 1 次方程式 $Ax = b$ を解くために A を何らかの意味で分解するのが目的ならば、適当な置換行列 P (上の議論と文字がかぶるけど大目に見て下さい) と、単位下三角行列 L が取れて

$$PA = LDL^T$$

となる、というのが一つの目標となるだろう。

行列の符号の判別をしたい場合は、

(i) A が正値であるためには、ピボッティングなしの Gauss の消去法で対角線から下を消去できて、対角成分がすべて正数になることが必要十分。

(ii) A が負値であるためには、ピボッティングなしの Gauss の消去法で対角線から下を消去できて、対角成分がすべて負数になることが必要十分。

(iii) A が正値でも負値でもなく、さらに行列式が 0 でないならば、A は不定符号であるが、これは逆は成り立たない。

(iv) 一般の場合に A の符号数 (p, q) を求めるには、少し工夫が必要である。伊理 [2] または [3] を見よ。

11.1.1 行列についての色々な同値関係

1. $A, B \in M(n; \mathbf{C})$ に対して、

$$A \sim_s B \quad \overset{\text{def.}}{\Leftrightarrow} \quad \exists U \in U(n; \mathbf{C}) \quad \text{s.t.} \quad U^*AU = B \quad (U^{-1}AU = B).$$

代表元としては Schur 標準形というのか？上三角に出来る (もとが対称であれば対角行列に出来る)。
実数バージョンもある。

2. (今回は関係ない) $A, B \in M(n; \mathbf{C})$ に対して、

$$A \sim_{\text{特}} B \overset{\text{def.}}{\Leftrightarrow} \exists U_1, U_2 \in U(n; \mathbf{C}) \quad \text{s.t.} \quad U_2^*AU_1 = B.$$

特異値分解に対応。

3. $A, B \in M(n; \mathbf{C})$ に対して、

$$A \sim_s B \quad (A \text{ と } B \text{ は相似}) \overset{\text{def.}}{\Leftrightarrow} \exists P \in GL(n; \mathbf{C}) \quad \text{s.t.} \quad P^{-1}AP = B.$$

代表元としては Jordan 標準形が有名。実数の範囲で考えると、有理標準形とかいうのになる？
「基底の取り替えでなるべく簡単な行列にする」

4. A と B が n 次の実対称行列とするとき、

$$A \sim_c B \quad (A \text{ と } B \text{ は合同}) \overset{\text{def.}}{\Leftrightarrow} \exists P \in GL(n;k) \quad \text{s.t.} \quad P^T A P = B.$$

代表元として $\mathrm{diag}(\overbrace{1,\cdots,1}^{p},\overbrace{-1,\cdots,-1}^{q},0,\cdots,0)$ が取れる。伊理先生は “Sylvester 形” と呼んでいた。これは符号数 (p,q) で指定できることに注意。
「2 次形式を正則線形変換で平方完成 (対角化) する」
Hermite 行列版がある。

5. $k = \mathbf{R}$ or $\mathbf{C}$ とする。$A, B \in M(n;k)$ に対して、$A \sim_r B \overset{\text{def.}}{\Leftrightarrow} \exists P, Q \in GL(n;k)$ s.t. $QAP = B$.

代表元として $\mathrm{diag}(\overbrace{1,\cdots,1}^{r},0,\cdots,0)$ が取れる (1 の個数 r は実は $\mathrm{rank}\, A$ に等しい, 伊理先生はこの代表元を「階数標準形 (rank normal form) と呼んでおく」)。

実対称行列や Hermite 行列に対して、直交変換や unitary 変換で相似変換する話は、1, 3, 4 にまたがっている話になる。

2 はある意味で 1 を緩くしたものである。

3, 4 もある意味で 1 を緩くしたものである。$U \in U(n)$ を $G \in GL(n)$ に変えるのだが、そのときに $U^*(= U^{-1})$ を P^T にするか、P^{-1} にするか。

5 はとても緩い。例えば実対称行列について、$A \sim_c B \Rightarrow A \sim_r B$.

そうか、実対称行列について、直交変換に限った 1 (あるいは 3) をして、一般の正則線形変換で 4 をして、それから 5 をして、ということで、標準形が、

$$\mathrm{diag}(\lambda_1,\cdots,\lambda_n) \quad \begin{cases} \lambda_j > 0 & (1 \le j \le p) \\ \lambda_j < 0 & (p+1 \le j \le p+q) \\ \lambda_j = 0 & (p+q < j \le n) \end{cases}$$

から

$$\mathrm{diag}(\overbrace{1,\cdots,1}^{p},\overbrace{-1,\cdots,-1}^{q},0,\cdots,0)$$

を通じて

$$\mathrm{diag}(\overbrace{1,\cdots,1}^{\mathrm{rank}\, A},0,\cdots,0)$$

となるわけだ。もう $p+q = \mathrm{rank}\, A$ は当たり前に見えて来る。

書くとすごくゴチャゴチャしている感じがするけれど、頭の中でやると訳が分からなくなるのは仕方ないのが分かる。

絵的に書くと

$$Q^T A Q = B \quad \Rightarrow \quad P^T A P = B \quad \Rightarrow \quad QAP = B.$$

$$A \sim_s \mathrm{diag}(\lambda_1,\cdots,\lambda_n) \sim_c \mathrm{diag}(1,\cdots,1,-1,\cdots,-1,0,\cdots,0) \sim_r \mathrm{diag}(1,\cdots,1,0,\cdots,0).$$

11.2　見掛けが少し異なる 3 つの Sylvester の慣性律

以下に現れる行列は (時々「実」を書き漏らすけれど) すべて実行列とする。

実対称行列 A に対して、A の固有値のうちで正であるものの個数、負であるものの個数、0 であるものの個数をそれぞれ $\pi(A), \nu(A), \zeta(A)$ と書くことにする。

A は n 次正方行列として、

$$\pi(A), \nu(A), \zeta(A) \geq 0, \quad \pi(A) + \nu(A) + \zeta(A) = n$$

である。

まず実対称行列の実直交行列による対角化に関する常識的事項 (線形代数で習うはず) を復習する (これがなくても Sylvester の慣性律の証明は出来るわけだが、使った方が私には見通しが良いので…)。

A が n 次実対称行列ならば、実直交行列 U $(U^{-1} = U^T)$ と実数 $\lambda_1, \cdots, \lambda_n$ が存在して、

$$U^T A U = \mathrm{diag}\,(\lambda_1, \cdots, \lambda_n)\,.$$

ゆえに $\forall x \in \mathbf{R}^n$ に対して $y := U^{-1}x$ とおくと、$x = Uy$ で、

$$(Ax, x) = (AUy, Uy) = (U^T AUy, y) = \sum_{j=1}^{n} \lambda_j y_j^2.$$

こうして 2 次形式 (Ax, x) を平方和 — 定数 $\times$ (何か)^2 の和 — に変形できる。平方完成と呼んでも良いだろうが、これを 2 次形式の対角化という。標語的に

平方完成は対角化である

と言っても良いであろう (筋の通った主張にするには色々言葉を足さなければいけないが、迷子にならないために手短に言い切っておく)。

ところで、上の対角化の議論の要点は

(i) $U^T AU$ が対角行列 $\mathrm{diag}\,(\lambda_1, \cdots, \lambda_n)$ である

(ii) U は正則

の二つである、と言えるだろう。

(Ax, x) が平方和の形に書けるためには、U が実直交行列であることは必要ない (だから以下では U でなく、P と書いたりする)。

U が実直交行列でなければ、λ_j は A の固有値とは限らないが、λ_j $(j = 1, \cdots, n)$ のうちの正であるものの個数、負であるものの個数は、A で定まる。それを定式化したものが、次の Sylvester の慣性律と呼ばれる定理である。

定理 11.1 (Sylvester の慣性律, (Sylvester's law of inertia)) 任意の実対称行列 A に対して、正則行列 P があって、$P^TAP = \mathrm{diag}\,(\lambda_1, \cdots, \lambda_n)$ となるとき、

$$n_p := \sharp\{j \in \{1, \cdots, n\}; \lambda_j > 0\},$$
$$n_n := \sharp\{j \in \{1, \cdots, n\}; \lambda_j < 0\},$$
$$n_z := \sharp\{j \in \{1, \cdots, n\}; \lambda_j = 0\}$$
($\sharp$ は集合の要素数を表す)

は P に依らず、A だけで定まる。(実対称行列が正則行列で対角化されるとき、対角成分のうち正であるものの個数、負であるものの個数、0 であるものの個数は、正則行列によらず、A だけで定まる。)

この定理は、少し後で紹介する「良くテキストに載っている証明」からすると自然な主張であるが、やや正体が分かりづらいのでは？と思う。次の定理とセットにして覚えることを勧める。

定理 11.2 (老婆心版 Sylvester の慣性律) 任意の実対称行列 A に対して、正則行列 P があって、$P^TAP = \mathrm{diag}\,(\lambda_1, \cdots, \lambda_n)$ となるとき、

$$n_p := \sharp\{j \in \{1, \cdots, n\}; \lambda_j > 0\},$$
$$n_n := \sharp\{j \in \{1, \cdots, n\}; \lambda_j < 0\},$$
$$n_z := \sharp\{j \in \{1, \cdots, n\}; \lambda_j = 0\}$$

とおくと、

$$n_p = \pi(A), \quad n_n = \nu(A), \quad n_z = \zeta(A).$$

証明　定理 11.1 を認めた上での証明を与える。P として $U^TAU = \mathrm{diag}[\lambda_1, \cdots, \lambda_n]$ となるような実直交行列 U を取れば、$P^TAP = U^TAU = U^{-1}AU$ は A の相似変換であり、$\lambda_1, \cdots, \lambda_n$ は A の固有値である。ゆえに n_p, n_n, n_z はそれぞれ $\pi(A), \nu(A), \zeta(A)$ に等しい。 ■

杉原・室田 [8] には、次の形の Sylvester の慣性律が載っている。

定理 11.3 (杉原・室田版 Sylvester の慣性律) A を実対称行列とするとき，任意の正則行列 P に対して、$B := P^TAP$ とおくと、

$$\pi(A) = \pi(B), \quad \nu(A) = \nu(B), \quad \zeta(A) = \zeta(B)$$

が成り立つ。

結論部分が「〜に依らず、〜で定まる」でなくて、具体的な等式である点は、定理 11.2 と同じである。対角化でなくて、変換した行列 B の固有値の話にしてしまうのも、少なくとも彼らの目的 (固有値の数値計算アルゴリズムの議論をする) にとっては使いやすいようである。

ここでは、定理 11.2 と定理 11.3 が同等であること (一方を認めれば他方がすぐに導かれる

こと) を見てみよう。

まず、定理 11.3 を認めよう。n 次実対称行列 A, 正則行列 P に対して、$\exists\lambda_1, \cdots, \lambda_n$ s.t. $P^TAP = \mathrm{diag}[\lambda_1, \cdots, \lambda_n]$ となったとすると、$B := P^TAP$ の固有値は (対角成分である) λ_1, $\cdots$, λ_n であるから、

$$n_p = \pi(B) = \pi(A), \quad n_n = \nu(B) = \nu(A), \quad n_z = \zeta(B) = \zeta(A).$$

逆に定理 11.2 を認めよう。実対称行列 A, 正則行列 P に対して、$B := P^TAP$ とおく。B は実対称行列であるから、適当な実直交行列 U で対角化できる: $\exists\lambda_1, \cdots, \lambda_n \in \mathbf{R}$ s.t. $U^TBU = \mathrm{diag}[\lambda_1, \cdots, \lambda_n]$. このとき

$$\begin{aligned}&\mathrm{diag}\,(\lambda_1, \cdots, \lambda_n) = U^TBU = U^TP^TAPU = (PU)^TA(PU) = P'^TAP',\\ &P' := PU.\end{aligned}$$

P と U が正則であるから、P' は正則である。定理 11.2 から、(B の固有値である) λ_1, $\cdots$, λ_n のうちで正であるものの個数、負であるものの個数、0 であるものの個数は、それぞれ $\pi(A)$, $\nu(A)$, $\zeta(A)$ に等しい。ゆえに $\pi(B) = \pi(A)$, $\nu(B) = \nu(A)$, $\zeta(B) = \zeta(A)$. ■

11.3 Sylvester の慣性律の証明

それでは Sylvester の慣性律を証明しよう。前項で 3 つのバージョンの同値性は分かっているので、どれか一つだけ証明すれば十分であるが、独立に証明することを目指す。

まず定理 11.1 を証明しよう。

定理 11.1 の証明　線型形式の独立性 (行列のランクというべき？) の議論をしている。

$x = Sy$, $x = Rz$ という正則変数変換で

$$\begin{aligned}(Ax, x) &= \alpha_1y_1^2 + \cdots + \alpha_ry_r^2 - \alpha_{r+1}y_{r+1}^2 - \cdots - \alpha_py_p^2 \quad (\alpha_j > 0)\\ &= \beta_1z_1^2 + \cdots + \beta_sz_s^2 - \beta_{s+1}z_{s+1}^2 - \cdots - \beta_qz_q^2 \quad (\beta_j > 0)\end{aligned}$$

と対角化できたとする。一般性を失うことなく $p \geq q$ として良い。目標は $p = q$, $r = s$ を証明することである。

各 y_j, z_k は変数 x についての線形形式と見なせる。y_j 同士、z_k 同士は 1 次独立である。

もしも $r < s$ ならば、y_1, $\cdots$, y_p は y_1, $\cdots$, y_r, z_{s+1}, $\cdots$, z_q の 1 次結合では表せない。(もし出来たとすると、独立な p 個の 1 次形式 y_1, $\cdots$, y_p が、それより少ない $r + (q - s)$ 個の 1 次形式で表せることになって矛盾する。$r + (q - s) = (r - s) + q < q \leq p$ に注意)。例えば y_k, $r + 1 \leq k \leq p$ が y_1, $\cdots$, y_r, z_{s+1}, $\cdots$, z_q で表せなかったとすると、

$$y_1 = \cdots = y_r = z_{s+1} = \cdots = z_q = 0, \quad y_k = 1$$

は x_1, $\cdots$, x_n に関する連立 1 次方程式として解を持つ。しかし、

$$\begin{aligned}&(Ax, x) = \alpha_1 \cdot 0^2 + \cdots + \alpha_r \cdot 0^2 - \alpha_{r+1}y_{r+1}^2 - \cdots - \alpha_k \cdot 1^2 - \cdots \alpha_py_p^2 \leq -\alpha_k < 0,\\ &(Ax, x) = \beta_1z_1^2 + \cdots + \beta_sz_s^2 - \beta_{s+1} \cdot 0^2 - \cdots - \beta_q \cdot 0^2 \geq 0\end{aligned}$$

となって矛盾する。

同様に $r > s$ ならば、$y_1, \cdots, y_r$ が $y_{r+1}, \cdots, y_p, z_1, \cdots, z_s$ で表されない。例えば y_k $(1 \leq k \leq r)$ が $y_{r+1}, \cdots, y_p, z_1, \cdots, z_s$ で表されないとすると、

$$y_{r+1} = \cdots = y_p = z_1 = \cdots = z_s = 0, \quad y_k = 1$$

は $x_1, \cdots, x_n$ に関する連立 1 次方程式として解を持つ。しかし、

$$\begin{aligned}(Ax, x) &= \alpha_1 \cdot y_1^2 + \cdots + \alpha_k \cdot 1^2 + \cdots \alpha_r \cdot y_r^2 - \alpha_{r+1} \cdot 0^2 - \cdots - \alpha_p \cdot 0^2 \geq \alpha_k > 0, \\ (Ax, x) &= \beta_1 \cdot 0^2 + \cdots + \beta_s \cdot 0^2 - \beta_{s+1} \cdot y_{s+1}^2 - \cdots - \beta_q \cdot \beta_q^2 \leq 0\end{aligned}$$

となって矛盾する。

ゆえに $r = s$. 必要ならば $(-Ax, x)$ を考えることによって、負の係数を持つ項の個数が等しいことも示せる。■

上で書いたように、実は $p = q = \operatorname{rank} A$ であるから、それを利用するともう少し見通しの良い証明が得られる。

証明　(準備中) ■

この証明はやや分り辛い (少なくとも私にとって)。そこでこれとは独立に定理 11.2 の証明を与える (これは自前でででっち上げたものなので、後でもう一度チェックすること)。

定理 **11.2** の証明　A を n 次実対称行列とする。

$$\begin{aligned}\mathcal{P} &:= \{V; V \text{ は } \mathbf{R}^n \text{ の部分空間}, \forall x \in V \setminus \{0\} \quad (Ax, x) > 0\}, \\ \mathcal{N} &:= \{V; V \text{ は } \mathbf{R}^n \text{ の部分空間}, \forall x \in V \setminus \{0\} \quad (Ax, x) < 0\}, \\ \mathcal{Z} &:= \{V; V \text{ は } \mathbf{R}^n \text{ の部分空間}, \forall x \in V \setminus \{0\} \quad (Ax, x) = 0\}\end{aligned}$$

とおく。$\{0\} \in \mathcal{P}, \mathcal{N}, \mathcal{Z}$ であるから、$\mathcal{P}, \mathcal{N}, \mathcal{Z} \neq \emptyset$.

$$N_p := \max\{\dim V; V \in \mathcal{P}\}, \quad N_n := \max\{\dim V; V \in \mathcal{N}\}, \quad N_z := \max\{\dim V; V \in \mathcal{Z}\}$$

が定まる (念のため: $\dim\{0\} = 0$ である)。

主張: $N_p = \pi(A)$, $N_n = \nu(A)$, $N_z = \zeta(A)$.

A は実対称行列であるから、固有ベクトルからなる正規直交基底 $v_1, \cdots, v_n$ が存在する。$Av_j = \lambda_j v_j$ として、

$$V_p := \operatorname{Span}\{v_j; \lambda_j > 0\}, \quad V_n := \operatorname{Span}\{v_j; \lambda_j < 0\}, \quad V_z := \operatorname{Span}\{v_j; \lambda_j = 0\}$$

とおくと、$V_p \in \mathcal{P}$, $V_n \in \mathcal{N}$, $V_z \in \mathcal{Z}$. $\dim V_p = \pi(A)$, $\dim V_n = \nu(A)$, $\dim V_z = \zeta(A)$ であるから、N_p, N_n, N_z の最大性によって

$$(\sharp) \qquad N_p \geq \pi(A), \quad N_n \geq \nu(A), \quad N_z \geq \zeta(A).$$

さて、一般に

$$W_p \in \mathcal{P}, \quad W_n \in \mathcal{N}, \quad W_z \in \mathcal{Z} \quad \Rightarrow \quad W_p \cap W_n = W_N \cap W_z = W_z \cap W_p = \{0\}$$

が成り立つことは容易に証明できる。W_p, W_n, W_z として、特に

$$\dim W_p = N_p, \quad \dim W_n = N_n, \quad \dim W_z = N_z$$

を満たすものを取って、

$$W_p \oplus W_n \oplus W_z \subset \mathbf{R}^n$$

において、次元を調べる。(♯) を用いると

$$n = \pi(A) + \nu(A) + \zeta(A) \leq N_p + N_n + N_z \leq n.$$

左辺と右辺が一致するので、不等式はすべて等式で

$$N_p = \pi(A), \quad N_n = \nu(A), \quad N_z = \zeta(A) \quad \text{(主張の証明終わり)}.$$

さて、正則行列 P による変換 $x = Py$ で、

$$(Ax, x) = \sum_{j=1}^{n} \mu_j y_j^2$$

になったとする。簡単のために順番を修正して、最初の p 個は正，次の q 個は負、残りは 0, と出来る。つまり

$$(Ax, x) = \alpha_1 y_1^2 + \cdots + \alpha_p y_p^2 - \alpha_{p+1} y_{p+1}^2 - \cdots - \alpha_{p+q} y_{p+q}^2, \quad \alpha_j > 0 \quad (1 \leq j \leq p+q)$$

として良い。

(a) p 次元空間 $\{Py; y_{p+1} = \cdots = y_n = 0\}$ では $(Ax, x) > 0$ になるので、$p \leq \pi(A)$.

(b) q 次元空間 $\{Py; y_1 = \cdots = y_p = y_{p+q+1} = \cdots = y_n = 0\}$ では $(Ax, x) < 0$ になるので、$q \leq \nu(A)$.

(c) $n-(p+q)$ 次元空間 $\{Sy; y_1 = \cdots = y_{p+q} = 0\}$ では $(Ax, x) = 0$ になるので、$n-(p+q) \leq \zeta(A)$.

(c) から $p+q+\zeta(A) \geq n$ であるが、(a) から $\pi(A) \geq p$, (b) から $\nu(A) \geq q$ であるから、

$$n = \pi(A) + \nu(A) + \zeta(A) \geq p + q + \zeta(A) \geq n.$$

両端が等しいので、途中はすべて等式で $p = \pi(A)$, $q = \nu(A)$. ■

定理 11.3 の直接的な証明が見たければ、杉原・室田 [8] を見よ (もちろん前項の議論から、間接的には証明は済んでいる)。

11.4 2 次形式の対角化のアルゴリズム (後でどこかで使うもの)

2 次形式の対角化 (平方完成)、すなわち与えられた実対称行列 A に対して、$R^T AR = D$, $D =$ 対角行列 となる正則行列 R を求めるアルゴリズムについて考える。これが解決すれば、2 次形式の符号数 (n_+, n_-, n_0) を求める 1 つのアルゴリズムが得られるわけである。

古典的な 2 次形式論では、Lagrange の方法と言うのだそうである。でも折角だから行列の変形操作の話として書き表したい (線形代数、あるいは数値線形代数として説明したいので)。

伊理 [2], [3] に書いてあるはずのこと。

11.4.1 例に基づく説明

例 **11.4** (係数行列の前進消去だけで済む場合 ($\det A_k \neq 0$ のとき)) まずピボットに 0 が現れない単純な例から。

$$A = \begin{pmatrix} 1 & -2 & 1 \\ -2 & 6 & -6 \\ 1 & -6 & 12 \end{pmatrix}$$

に対して Gauss の消去法の前進消去過程で、

$$A = \begin{pmatrix} 1 & -2 & 1 \\ -2 & 6 & -6 \\ 1 & -6 & 12 \end{pmatrix} \to \begin{pmatrix} 1 & -2 & 1 \\ 0 & 2 & -4 \\ 0 & -4 & 11 \end{pmatrix} \to \begin{pmatrix} 1 & -2 & 1 \\ 0 & 2 & -4 \\ 0 & 0 & 3 \end{pmatrix}.$$

これはつまり

$$L := \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 1 & -2 & 1 \end{pmatrix}, \quad U := \begin{pmatrix} 1 & -2 & 1 \\ 0 & 2 & -4 \\ 0 & 0 & 3 \end{pmatrix}$$

と置くと

$$L^{-1}A = U, \quad \text{すなわち} \quad A = LU$$

である、ということを意味している (A の LU 分解)。

手で検算することを考えて

$$L^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix}$$

を残しておこう (私は軟弱なので MATLAB で計算している)。

閑話休題。連立 1 次方程式を解くならば、これから後退代入の過程に入るわけだが、今の目標は 2 次形式の対角化である。行に関して行った操作を列に関しても行うことで、対角線の上側も消去されて

$$L^{-1}A\left(L^{-1}\right)^T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

となる ($(L^{-1})^T$ は単位上三角行列なので、上三角行列 $U = (L^{-1}A)$ に右から掛けると対角線の上側しか変化しない。一方で、結果が対称行列になることが分かるので、実は対角行列である。結局、対角線の上側をゼロ・クリアするだけである。)。

$$D := \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

とおくと、

$$L^{-1}A(L^{-1})^T = D \quad \text{あるいは} \quad A = LDL^T$$

を意味している。

理解できていれば、A を前進消去して上三角行列 U に変換した段階で、L と D が求まる (求まったことが分かる)。念のため LDL^T を計算して A になることを確認すれば完璧だろう。$R := (L^{-1})^T$ とおき、$x = Ry$ と変数変換すると、

$$A[x] = (Ax, x) = (ARy, Ry) = (R^T ARy, y) = (Dy, y) = D[y]$$

となるはず。

ここで遠い記憶が蘇る。中国からの留学生 C さんのチューターをしていたとき、C さんが僕の知らない計算をしていたのである。

$$(A \mid I)$$

と A と I を並べて、掃き出し (前進消去) をしていく。

- A だけでなく I も並べて同時に変形していくこと
- 左側のブロックについては、行について行った変形を列についてもしていく
  (上の例題では行・列の交換が起こらないのでさぼっても良いのだけど)

の二点が要点。

$$\left(L^{-1}A \mid L^{-1}\right)$$

となるわけで、$U(= L^{-1}A)$ だけでなく、L^{-1} の具体形も得られることに注意しよう。行列 A (あるいは 2 次形式 $A[x] = (Ax, x)$) の符号数が知りたいだけならば必要ないが、対角化するための変数変換 $x = Ry$, $R = (L^{-1})^T$ が具体的に知りたい場合は、便利であろう (L^{-1} があれば、その転置行列はもう分っているに等しい)。つまり中国では、こういう筆算も教えているのでしょう。似たようなのを古屋先生の本か何かで見たような気がする。いつの間にか、日本の線形代数のテキストでは落ちてしまったわけだね。

上の例でこれを実行してみると

$$(A \mid I) = \begin{pmatrix} 1 & -2 & 1 & 1 & 0 & 0 \\ -2 & 6 & -6 & 0 & 1 & 0 \\ 1 & -6 & 12 & 0 & 0 & 1 \end{pmatrix} \to \begin{pmatrix} 1 & -2 & 1 & 1 & 0 & 0 \\ 0 & 2 & -4 & 2 & 1 & 0 \\ 0 & -4 & 11 & -1 & 0 & 1 \end{pmatrix} \to \begin{pmatrix} 1 & -2 & 1 & 1 & 0 & 0 \\ 0 & 2 & -4 & 2 & 1 & 0 \\ 0 & 0 & 3 & 3 & 2 & 1 \end{pmatrix}.$$

これから

$$D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}, \quad L^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix}.$$

確かに

$$L^{-1}A(L^{-1})^T = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & -2 & 1 \\ -2 & 6 & -6 \\ 1 & -6 & 12 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix} = D.$$

めでたし、めでたし。 ■

例 **11.5** (楽屋裏の紹介と例の追加) 上の例は逆算して作った。L と D を適当に決めて、LDL^T を計算して A を決めたのである。この例は正値対称行列だったけれど、不定符号が欲しければ、

```
L=[1,0,0;2,1,0;3,2,1]
d=diag([1,-2,3])
a=L*d*L'
```

として

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 2 & 2 \\ 3 & 2 & 4 \end{pmatrix}$$

とか。

$$(A \mid I) = \begin{pmatrix} 1 & 2 & 3 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 1 & 0 \\ 3 & 2 & 4 & 0 & 0 & 1 \end{pmatrix} \to \begin{pmatrix} 1 & 2 & 3 & 1 & 0 & 0 \\ 0 & -2 & -4 & -2 & 1 & 0 \\ 0 & -4 & -5 & -3 & 0 & 1 \end{pmatrix} \to \begin{pmatrix} 1 & 2 & 3 & 1 & 0 & 0 \\ 0 & -2 & -4 & -2 & 1 & 0 \\ 0 & 0 & 3 & 1 & -2 & 1 \end{pmatrix}$$

であるから

$$D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 3 \end{pmatrix}, \quad L^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 1 & -2 & 1 \end{pmatrix}.$$

実際

$$L^{-1}A\left(L^{-1}\right)^T = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 1 & -2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 \\ 2 & 2 & 2 \\ 3 & 2 & 4 \end{pmatrix} \begin{pmatrix} 1 & -2 & 1 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 3 \end{pmatrix} = D.\blacksquare$$

例 **11.6** (ピボットの交換が必要な場合 ($\forall k \ \det A_k \neq 0$ が成り立たないとき))

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$$

を考えよう。$(1,1)$ 成分が 0 なので、このままでは掃き出せない。これも連立 1 次方程式を解くだけならば、第 1 行と第 2 行を交換して、つまり

$$P := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

として、

$$PA = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$$

を LU 分解すれば良い。我々の目標は対角化なので (少しくどいかな？)、

$$\widetilde{A} := PAP^T = PAP = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}$$

という対称行列を考える。この LU 分解は

$$L := \begin{pmatrix} 1 & 0 \\ 1/2 & 1 \end{pmatrix}, \quad L^{-1}\widetilde{A} = \begin{pmatrix} 2 & 1 \\ 0 & -1/2 \end{pmatrix}.$$

ゆえに $\widetilde{A}$ の対角化は (対角線の上側をゼロ・クリアした)

$$L^{-1}\widetilde{A}\left(L^{-1}\right)^T = \begin{pmatrix} 2 & 0 \\ 0 & -1/2 \end{pmatrix}.$$

検算してみよう。

$$D := \begin{pmatrix} 2 & 0 \\ 0 & -1/2 \end{pmatrix} \quad \text{と置くと} \quad LDL^T = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} = \widetilde{A}.$$

$\widetilde{A} = PAP^T$, $L^{-1}\widetilde{A}\,(L^{-1})^T = D$ であるから、

$$L^{-1}PAP^T\left(L^{-1}\right)^T = D.$$

ゆえに

$$R := P^T\left(L^{-1}\right)^T = \cdots(\text{ちびっと計算})\cdots = \begin{pmatrix} 0 & 1 \\ 1 & -1/2 \end{pmatrix}$$

とおくと、

$$R^TAR = D.$$

ゆえに

$$x = Ry \quad \text{すなわち} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_2 \\ y_1 - y_2/2 \end{pmatrix}$$

と変数変換すると対角化されるはずである。確認しよう。

$$\begin{aligned} A[x] &= 2x_1x_2 + 2x_2^2 \\ &= 2y_2(y_1 - y_2/2) + 2(y_1 - y_2/2)^2 = 2y_1y_2 - y_2^2 + 2(y_1^2 - y_1y_2 + y_2^2/4) \\ &= 2y_1^2 - y_2^2/2 \\ &= D[y]. \end{aligned}$$

とりあえず、めでたし、めでたし。

中国流の筆算を考えてみよう。左のブロック (もともと A の部分) には、行に関する基本変形と列に関する基本変形の両方を行うが (列に関する変形は要するに対称になるような操作をするわけで、0 に掃き出したところと対称な場所にあるところは 0 にする、行交換をしたら列交換もする、だけであり、計算不要とも言える)、右のブロック (もともと I の部分) には、行に関する基本変形のみを行う。つまり

$$\begin{aligned} (A \mid I) &\to \left(R_1AR_1^T \mid R_1I\right) \to \left(R_2R_1AR_1^TR_2^T \mid R_2R_1I\right) \to \cdots \\ &\to \left(R_k\cdots R_2R_1AR_1^TR_2^T\cdots R_k^T \mid R_k\cdots R_2R_1I\right) \\ &= (D \mid R), \quad D = R_k\cdots R_2R_1AR_1^TR_2^T\cdots R_k^T, \quad R := R_k\cdots R_2R_1. \end{aligned}$$

これは

$$R^T AR = D$$

を意味している。

上の A に対して実行すると、

これだけで良い！

$$(A \mid I) = \left(\begin{array}{cc|cc} 0 & 1 & 1 & 0 \\ 1 & 2 & 0 & 1 \end{array}\right) \to \left(\begin{array}{cc|cc} 1 & 2 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{array}\right) \to \left(\begin{array}{cc|cc} 2 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{array}\right)$$
$$\to \left(\begin{array}{cc|cc} 2 & 0 & 0 & 1 \\ 0 & -1/2 & 1 & -1/2 \end{array}\right).$$

これから、

$$D = \begin{pmatrix} 2 & 0 \\ 0 & -1/2 \end{pmatrix}, \quad R^T = \begin{pmatrix} 0 & 1 \\ 1 & -1/2 \end{pmatrix}, \quad R = \begin{pmatrix} 0 & 1 \\ 1 & -1/2 \end{pmatrix}.$$

こうして上で「(ちびっと計算)」と書いたところが、枠内の計算で済んでしまった。 ■

例 **11.7** (対角成分がすべて 0 — 自明でない最も簡単な場合) $A = \begin{pmatrix} 0 & a \\ a & 0 \end{pmatrix}$ とする。A の対角成分はすべて 0 なので、これまでと同じようには掃き出せない。連立 1 次方程式を解く場合は、単に第 1 行と第 2 行を交換すれば良いが、対角化をする場合はそれは役に立たない。($P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ とするとき、$PA = \begin{pmatrix} a & 0 \\ 0 & a \end{pmatrix}$ であるが、右から P^T をかける必要があり、そうすると $PAP^T = \begin{pmatrix} 0 & a \\ a & 0 \end{pmatrix} = A$ と元に戻ってしまう。)

ここでは、直交行列による対角化を思い出そう (必ず出来るはずだ)。固有値は $\pm a$ で、固有ベクトルはそれぞれ $\begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \end{pmatrix}$ である。そこで $R = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$ とすると、R は原点の回りの角 $\pi/4$ の回転であり、$R^T = R^{-1}$ は角 $-\pi/4$ の回転である。

$$R^T AR = \begin{pmatrix} a & 0 \\ 0 & -a \end{pmatrix}.$$

あるいは $R = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ とすると、$R^T = R^{-1} = R$. これは鏡映変換で、こちらを使っても

$$R^T AR = \begin{pmatrix} a & 0 \\ 0 & -a \end{pmatrix}.$$

対角化が目的の場合、必ずしも固有値を求める必要はないので、$\frac{1}{\sqrt{2}}$ はなくても良い。$R =$

$\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$ とすれば、

$$R^T AR = \begin{pmatrix} 2a & 0 \\ 0 & -2a \end{pmatrix}.$$

あるいは $R = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ として

$$R^T AR = \begin{pmatrix} 2a & 0 \\ 0 & -2a \end{pmatrix}.$$

後者は対称行列なので、扱いやすいかもしれない。

中国流筆算では、

$$(A \mid I) = \begin{pmatrix} 0 & a & 1 & 0 \\ a & 0 & 0 & 1 \end{pmatrix} \to (R^T AR \mid R^T) = \begin{pmatrix} 2a & 0 & 1 & 1 \\ 0 & -2a & 1 & -1 \end{pmatrix}$$

とするところだろう。変数変換は $x = Ry$, すなわち

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} y_1 + y_2 \\ y_1 - y_2 \end{pmatrix}$$

で、実際に

$$\begin{aligned} A[x] &= 2ax_1x_2 \\ &= 2a(y_1 + y_2)(y_1 - y_2) = 2a(y_1^2 - y_2^2) \\ &= 2ay_1^2 - 2ay_2^2 \end{aligned}$$

と対角化が出来る。 ■

なお、

$$R = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$$

の逆行列は

$$R^{-1} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} = \frac{1}{2} R.$$

上の例は簡単であるが、一般の場合に基本操作として使うことが出来る。それを説明しよう。

$$A = \left(\begin{array}{c|c|c} D & O & O \\ \hline O & A' & B^T \\ \hline O & B & C \end{array}\right), \quad D = \text{対角行列}, \quad A' = \begin{pmatrix} 0 & a \\ a & 0 \end{pmatrix}, \quad C = \text{対称行列}$$

とする (つまり掃き出しの過程で対角成分が 0 のブロックが出て来た場合を考える)。

$$S := \left(\begin{array}{c|c|c} I & O & O \\ \hline O & R & O \\ \hline O & O & I \end{array}\right), \quad R := \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

とすると、$R^T = R$ であるから、$S^T = S$ で、

$$S^T AS = \left(\begin{array}{c|c|c} D & O & O \\ \hline O & R^T A' R & R^T B^T \\ \hline O & BR & C \end{array}\right), \quad R^T A' R = \begin{pmatrix} 2a & 0 \\ 0 & -2a \end{pmatrix}.$$

ピボットが見つからないという障害物を突破できた。
　すなわち

$$(\star) \qquad S^T AS = \left(\begin{array}{c|cc|c} D & O & & O \\ \hline O & 2a & 0 & (BR)^T \\ & 0 & -2a & \\ \hline O & BR & & C \end{array}\right).$$

後で使いそうなので

$$B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ \vdots & \vdots \\ b_{r1} & b_{r2} \end{pmatrix} \quad \text{とするとき} \quad BR = \begin{pmatrix} b_{11}+b_{12} & b_{11}-b_{12} \\ b_{21}+b_{22} & b_{21}-b_{22} \\ \vdots & \vdots \\ b_{r1}+b_{r2} & b_{r1}-b_{r2} \end{pmatrix}$$

であることを注意しておく。
　$(\star)$ の形になれば、2 段階の掃き出しが出来て

$$S^T AS \to \left(\begin{array}{c|cc|c} D & O & & O \\ \hline O & 2a & 0 & O \\ & 0 & -2a & \\ \hline O & O & & C' \end{array}\right).$$

少し脇道にそれるが、後のために注意しておくと

$$S^{-1} = \left(\begin{array}{c|c|c} I & O & O \\ \hline O & \frac{1}{2}R & O \\ \hline O & O & I \end{array}\right)$$

である。

例 **11.8**

$$A = \left(\begin{array}{c|cc|cc} 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 2 & 1 & 2 \\ 0 & 2 & 0 & 3 & 1 \\ \hline 0 & 1 & 3 & 1 & 3 \\ 0 & 2 & 1 & 3 & 2 \end{array}\right).$$

$$(A \mid I) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 3 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 3 & 1 & 3 & 0 & 0 & 0 & 1 & 0 \\ 0 & 2 & 1 & 3 & 2 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \to \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 4 & 3 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -4 & -2 & 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 4 & -2 & 1 & 3 & 0 & 0 & 0 & 1 & 0 \\ 0 & 3 & 1 & 3 & 2 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\to \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 4 & 3 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -4 & -2 & 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1/2 & 0 & -3/2 & -1/2 & 1 & 0 \\ 0 & 0 & 1 & 0 & -1/4 & 0 & -3/4 & -3/4 & 0 & 1 \end{pmatrix}$$

$$\to \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -4 & -2 & 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1/2 & 0 & -3/2 & -1/2 & 1 & 0 \\ 0 & 0 & 1 & 0 & -1/4 & 0 & -3/4 & -3/4 & 0 & 1 \end{pmatrix}$$

$$\to \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -4 & -2 & 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1/2 & 0 & -3/2 & -1/2 & 1 & 0 \\ 0 & 0 & 0 & -1/2 & 0 & 0 & -3/4 & -3/4 & 0 & 1 \end{pmatrix}$$

$$\to \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -4 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1/2 & 0 & -3/2 & -1/2 & 1 & 0 \\ 0 & 0 & 0 & -1/2 & 0 & 0 & -3/4 & -3/4 & 0 & 1 \end{pmatrix}$$

$$\to \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -4 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1/2 & 0 & -3/2 & -1/2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1/8 & 0 & -1/8 & -7/8 & -1/4 & 1 \end{pmatrix}$$

$$\to \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -4 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 & 0 & 0 & -3/2 & -1/2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1/8 & 0 & -1/8 & -7/8 & -1/4 & 1 \end{pmatrix}.$$

これから

$$D=\begin{pmatrix}1&0&0&0&0\\0&4&0&0&0\\0&0&-4&0&0\\0&0&0&-2&0\\0&0&0&0&1/8\end{pmatrix},\quad S^T=\begin{pmatrix}1&0&0&0&0\\0&1&1&0&0\\0&1&-1&0&0\\0&-3/2&-1/2&1&0\\0&-1/8&-7/8&-1/4&1\end{pmatrix}.$$

実際

$$S^TAS=D$$

が成り立つ。■

より一般の場合、すなわち

$$A=\left(\begin{array}{c|c}D&O\\\hline O&A'\end{array}\right),\quad D=k-1\text{次対角行列},\quad A'=\text{対称行列}$$

で、A' の対角成分はすべて 0 であるが、非対角成分に 0 でないものがある場合を考える。まずは $a_{ij}\neq 0$ となる $(i,j)\in\{k,\cdots,n\}\times\{k,\cdots,n\}$ を探す。

```
found = 0;
for (i=k; i<=n; i++) {
  for (j=k; j<=n; j++) { // j=i+1 からで良いのかな？
    if (a[i][j] != 0) {
      found = 1;
      break;
    }
  }
}
```

$a_{ij}=a_{ji}=a\neq 0$ として、i 行と k 行、i 列と k 列を交換、j 行と $k+1$ 行、j 列と $k+1$ 列を交換、すると、

$$A\to P_{j,k+1}P_{ik}AP_{ik}P_{j,k+1}=\left(\begin{array}{c|cc|c}D&O&&O\\\hline O&0&a&(B')^T\\&a&0&\\\hline O&B'&&C'\end{array}\right)$$

の形になる。ここで P_{rs} は左からかけると、r 行と s 行を交換することになる置換行列とする。

11.4.2 プログラム例

```
/*
 * testsign-v1.c
 */

#include <stdio.h>
#include <math.h>

#define MAXN (100)

void printmatrix(double a[MAXN+1][MAXN+1], int n)
{
  int i, j;
  for (i=1; i<=n; i++) {
    for (j=1;j<=n; j++) {
      printf("%g ", a[i][j]);
    }
    printf("\n");
  }
}

void find_sign(double a[MAXN+1][MAXN+1], int n,
       int *np, int *nn, int *nz)
{
  int i,j,k,p,ll,found,verbose=1;
  double c,alpha,beta,t;

 for (k=1; k<n; k++) {
    // 対角成分で 0 でないものを探す
    found = 0;
    for (p=k; p<=n; p++)
      if (a[p][p] != 0) {
        found = 1;
        break;
      }
    if (found) {
      if (p!=k) {
        if (verbose) {
          printf("ピボットは 0 だけど、対角成分で 0 でないものを見つけた。\n");
          printmatrix(a,n);
        }
        printf("%d 行,%d 行を交換\n", k, p);
        // p 行と k 行を交換
        for (ll=k;ll<=n;ll++) {
          t=a[p][ll]; a[p][ll]=a[k][ll]; a[k][ll]=t;
        }
        printf("%d 列,%d 列を交換\n", k, p);
        // p 列と k 列を交換
        for (ll=k; ll<=n; ll++) {
          t=a[ll][p]; a[ll][p]=a[ll][k]; a[ll][k]=t;
        }
        if (verbose) {
          printmatrix(a,n);
        }
      }
```

```
      // p==k のときは、もともと a[k][k]!=0 であるわけで交換の必要はない
      // a[k][k] をピボットとして掃き出す
      for (i=k+1; i<=n; i++) {
        c=a[i][k]/a[k][k];
        a[i][k]=0; // さぼっても良い。c を書いておいて後で使う手も
        for (j=k+1;j<=n;j++)
          a[i][j] -= c*a[k][j];
      }
    }
    else { // 非対角成分で 0 でないものを探す
      found = 0;
      for (i=k;i<=n;i++) {
        for (j=k;j<=n;j++) { // 対称なんだし右上だけ: j=i+1 からで良いな
          if (a[i][j] != 0) {
            found = 1;
            break;
          }
        }
if (found) break;
      }
      if (found) {
if (verbose) {
  printf("非対角成分で 0 でないものを見つけた。\n");
  printmatrix(a, n);
}
        if (!(i==k && j==k+1)) {
  // i 行と k 行, i 列と k 列を交換
  printf("%d 行 (列),%d 行 (列) を交換\n", k, i);
  for (ll=k;ll<=n;ll++) {
    t=a[i][ll]; a[i][ll]=a[k][ll]; a[k][ll]=t;
  }
  for (ll=k; ll<=n; ll++) {
    t=a[ll][i]; a[ll][i]=a[ll][k]; a[ll][k]=t;
  }
  // j 行と k+1 行, j 列と k+1 列を交換
  printf("%d 行 (列),%d 行 (列) を交換\n", k+1, j);
  for (ll=k;ll<=n;ll++) {
    t=a[j][ll]; a[j][ll]=a[k+1][ll]; a[k+1][ll]=t;
  }
  for (ll=k; ll<=n; ll++) {
    t=a[ll][j]; a[ll][j]=a[ll][k+1]; a[ll][k+1]=t;
  }
  printmatrix(a, n);
}
        // a[k][k+1] == a[k+1][k] != 0 となっている
        a[k][k] = a[k][k+1];
        a[k+1][k+1] = - a[k][k+1];
        a[k][k+1] = a[k+1][k] = 0;
        for (j=k+2; j<=n; j++) {
          alpha = a[k][j]; beta = a[k+1][j];
          a[k  ][j] = a[j][k  ] = alpha + beta;
          a[k+1][j] = a[j][k+1] = alpha - beta;
        }
        // a[k][k] をピボットとして掃き出す
```

```
        for (i=k+1; i<=n; i++) {
          c=a[i][k]/a[k][k];
          a[i][k]=0;
          for (j=i+1;j<=n;j++)
            a[i][j] -= c*a[k][j];
        }
        // a[k+1][k+1] をピボットとして掃き出す
        for (i=k+2; i<=n; i++) {
          c=a[i][k+1]/a[k+1][k+1];
          a[i][k+1]=0;
          for (j=i+1;j<=n;j++)
            a[i][j] -= c*a[k+1][j];
        }
      }
      else {
        // もう右下は 0 である
        break;
      }
    }
  }
  // 対角成分のうち正、負、零のものの個数を数える
  *np=0;
  *nn=0;
  *nz=0;
  for (k=1;k<=n;k++)
    if (a[k][k] > 0)
      (*np)++;
    else if (a[k][k] < 0)
      (*nn)++;
    else
      (*nz)++;
}

int main(void)
{
  int n, np, nn, nz;
  double a[MAXN+1][MAXN+1];

  // 例題 1
  n=3;
  a[1][1] =  1; a[1][2] = -2; a[1][3] =  1;
  a[2][1] = -2; a[2][2] =  6; a[2][3] = -6;
  a[3][1] =  1; a[3][2] = -6; a[3][3] = 12;
  printf("例題 1\n");
  printmatrix(a,n);
  find_sign(a, n, &np, &nn, &nz);
  printmatrix(a,n);
  printf("np=%d, nn=%d, nz=%d\n", np, nn, nz);

  // 例題 2
  n=3;
  a[1][1] =  1; a[1][2] = 2; a[1][3] = 3;
  a[2][1] =  2; a[2][2] = 2; a[2][3] = 2;
  a[3][1] =  3; a[3][2] = 2; a[3][3] = 4;
```

```
  printf("\n 例題 2\n");
  printmatrix(a,n);
  find_sign(a, n, &np, &nn, &nz);
  printmatrix(a,n);
  printf("np=%d, nn=%d, nz=%d\n", np, nn, nz);

  // 例題 3
  n=2;
  a[1][1] =  0; a[1][2] = 1;
  a[2][1] =  1; a[2][2] = 2;
  printf("\n 例題 3\n");
  printmatrix(a,n);
  find_sign(a, n, &np, &nn, &nz);
  printmatrix(a,n);
  printf("np=%d, nn=%d, nz=%d\n", np, nn, nz);

  // 例題 4
  n=2;
  a[1][1] = 0; a[1][2] = 2;
  a[2][1] = 2; a[2][2] = 0;
  printf("\n 例題 4\n");
  printmatrix(a,n);
  find_sign(a, n, &np, &nn, &nz);
  printmatrix(a,n);
  printf("np=%d, nn=%d, nz=%d\n", np, nn, nz);

  // 例題 5
  n=3;
  a[1][1] = 0; a[1][2] = 0; a[1][3] = 0;
  a[2][1] = 0; a[2][2] = 0; a[2][3] = 2;
  a[3][1] = 0; a[3][2] = 2; a[3][3] = 1;
  printf("\n 例題 5\n");
  printmatrix(a,n);
  find_sign(a, n, &np, &nn, &nz);
  printmatrix(a,n);
  printf("np=%d, nn=%d, nz=%d\n", np, nn, nz);

  // 例題 6
  n=3;
  a[1][1] = 0; a[1][2] = 0; a[1][3] = 0;
  a[2][1] = 0; a[2][2] = 0; a[2][3] = 2;
  a[3][1] = 0; a[3][2] = 2; a[3][3] = 0;
  printf("\n 例題 6\n");
  printmatrix(a,n);
  find_sign(a, n, &np, &nn, &nz);
  printmatrix(a,n);
  printf("np=%d, nn=%d, nz=%d\n", np, nn, nz);

  return 0;
}
```

```
例題 1
```

```
1 -2 1
-2 6 -6
1 -6 12
1 -2 1
0 2 -4
0 0 3
np=3, nn=0, nz=0

例題2
1 2 3
2 2 2
3 2 4
1 2 3
0 -2 -4
0 0 3
np=2, nn=1, nz=0

例題3
0 1
1 2
ピボットは0だけど、対角成分で0でないものを見つけた。
0 1
1 2
1行,2行を交換
1列,2列を交換
2 1
1 0
2 1
0 -0.5
np=1, nn=1, nz=0

例題4
0 2
2 0
非対角成分で0でないものを見つけた。
0 2
2 0
2 0
0 -2
np=1, nn=1, nz=0

例題5
0 0 0
0 0 2
0 2 1
ピボットは0だけど、対角成分で0でないものを見つけた。
0 0 0
0 0 2
0 2 1
1行,3行を交換
1列,3列を交換
1 2 0
2 0 0
0 0 0
1 2 0
```

```
0 -4 0
0 0 0
np=1, nn=1, nz=1

例題 6
0 0 0
0 0 2
0 2 0
非対角成分で 0 でないものを見つけた。
0 0 0
0 0 2
0 2 0
1 行 (列),2 行 (列) を交換
2 行 (列),3 行 (列) を交換
0 2 0
2 0 0
0 0 0
2 0 0
0 -2 0
0 0 0
np=1, nn=1, nz=1
```

A TODO

1. LR 法
2. Gerschgorin の定理の後半
3. Courant-Fischer の mini-max

B ChageLog

思い立ってこれからは加筆修正内容を記録することにした。

- 2005 年 8 月 24 日　数日来余暇に遊んできた Lanczos 法の説明を書く。
- 2005 年 8 月 25 日　木村 [5] を読んでいて、Householder 変換による QR 分解の話が載っていたので、この際書くことにした。
- 2005 年 8 月 26 日　Householder 変換の原理を確認するための MATLAB プログラムを書いて実験する。
- 2005 年 8 月 26 日　試しに既に書いてあった Jacobi 法のプログラムを実験してみたら、バグで動かなかったので、デバッグする。プログラム以外にも誤植があったので訂正する。やはり実験までしないとミスは減らない…
- 2005 年 8 月 27 日　昨日のデバッグで分かったこと: Octave では `function [q r]=myqr(a)` のような書き方ができるが、MATLAB では `function [q,r]=myqr(a)` のようにカンマが必要。他のプログラムも修正。

- 2005 年 8 月 27 日　本来は「本に書いてあること」を理解すべきだとは思うが、QR 分解をする各種手法の比較実験をしてみた。中途半端な実験しかできていないせいだろうとは思うがやや意外な結果 (修正 Gram-Schmidt の直交化法を用いる理由は見えなかった)。やってみて分かったのは、何を尺度にして評価するのか、実は分かっていなかったということ。

- 2005 年 9 月 6 日　QR 分解の各方法の比較の話、ちょっと気になって Trefethen and Bau III を見たら、ありゃりゃ。やはりこの本の内容は日本でも常識になっていないといけないのだろう。翻訳出さないかなあ。忙しいので、自分で試してここに書き足すのはいつの日か。

- 2013 年 8 月 26 日　Sylvester の慣性律について少し書く (Cholesky 分解ノートからこちらにコピーした)。やっと少し分かった。

- 2013 年 9 月 4 日　いよいよ一般の場合の 2 次形式の対角化、符号数の求め方、の話に手をつける。伊理先生の本を読んでプログラムを書こうかと思ったが、まずは実例攻撃でいくことにした。そのうちに古い記憶が蘇った。

参考文献

[1] F. Chatelin. *Valeurs propres de matrices*. Masson, Paris, 1988. (邦訳) F. シャトラン著, 伊理 正夫, 伊理由美 訳, 行列の固有値問題, シュプリンガー・フェアラーク東京 (1993).

[2] 伊理正夫. 一般線形代数. 岩波書店, 2003. 伊理正夫, 線形代数 I, II, 岩波講座応用数学, 岩波書店 (1993, 1994) の単行本化.

[3] 伊理正夫. 線形代数汎論. 朝倉書店, 2009. 「一般線形代数」のリニューアル.

[4] 桂田祐史. 多変数の微分積分学 1 講義ノート (2013 年度版). http://nalab.mind.meiji.ac.jp/~mk/lecture/tahensuu1-2013/tahensuu1-new-text.pdf, 2013.

[5] 木村英紀. 線形代数　数理科学の基礎. 東京大学出版会, 2003.

[6] 森正武, 杉原正顯, 室田一雄. 線形計算. 岩波講座 応用数学. 岩波書店, 1994.

[7] 山本哲朗. 数値解析入門 [新訂版]. サイエンス社, 2003. 1976 年初版発行の定番本の待望の改訂版.

[8] 杉原正顯, 室田一雄. 線形計算の数理. 岩波書店, 2009.

[9] 齋藤正彦. 線型代数入門. 東京大学出版会, 1966.

[10] 佐武一郎. 線型代数学. 裳華房, 1958, 1974. もともと『行列と行列式』という書名であったのを、テンソル代数の章を加筆した機会に改題した。

[11] 戸川隼人(はやと). マトリクスの数値計算. オーム社, 1971.

[12] Lloyd Nicholas Trefethen and David Bau III. *Numerical Linear Algebra*. SIAM, 1997.

[13] 桂田祐史. 一般化固有値問題. http://nalab.mind.meiji.ac.jp/~mk/labo/text/generalized-eigen/generalized-eigenvalue-problem.pdf, 2003.