数値解析C（塩田） 2007年1月17日

— 固有値・固有ベクトル（実対称行列の Jacobi 法、一般の行列の直接法）—

1. 予備知識

(1) 正方行列 A に対して

$$Av = \lambda v$$

を満たすスカラー λ、ベクトル $\boldsymbol{v} \neq \boldsymbol{0}$ をそれぞれ A の固有値・固有ベクトルと言う。

(2) n 次行列 A が n 個の一次独立な固有ベクトル $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n$ をもつとき、それらに対応する固有値を $\lambda_1, \cdots, \lambda_n$, $P := (\boldsymbol{v}_1 \ \cdots \ \boldsymbol{v}_n)$ と置けば

$$P^{-1}AP = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}$$

となる。(「A は P によって対角化される」と言う。)

(3) 実対称行列（実数成分の対称行列）は直交行列によって対角化できる。(直交行列とは $P^{-1} = {}^tP$ となるような正方行列。)

(4) 直交行列は次の形の行列いくつかの積として書ける：

- 回転の行列 $R(i,j,\theta) := \begin{pmatrix} & \vdots & & \vdots & \\ \cdots & \cos\theta & \cdots & -\sin\theta & \cdots \\ & \vdots & & \vdots & \\ \cdots & \sin\theta & \cdots & \cos\theta & \cdots \\ & \vdots & & \vdots & \end{pmatrix}$ $\left(\begin{array}{c} i, j \text{ 行}, i, j \text{ 列以外} \\ \text{は単位行列に同じ} \end{array}\right)$

- 対称移動の行列 $\begin{pmatrix} \pm 1 & & \\ & \ddots & \\ & & \pm 1 \end{pmatrix}$

2. Jacobi 法

Jacobi 法は実対称行列に対して用いる。

方針

$A = A_0 = (a_{ij})$ を n 次実対称行列とするとき、回転の行列 $R_0, R_1, \cdots$ をうまく取って

$$A_{k+1} := {}^tR_k A_k R_k \longrightarrow \text{対角行列} \qquad (k \longrightarrow \infty)$$

となるようにする。

Jacobi 法のアルゴリズム

$A_0 := A$; $P_0 := E$; $k := 0$;

repeat

$A_k = (a_{ij})$ の非対角成分 a_{pq} を上手に選ぶ (後述) ;

$$\theta := \begin{cases} \dfrac{1}{2}\arctan\left(\dfrac{2a_{pq}}{a_{pp} - a_{qq}}\right) & a_{pp} \neq a_{qq}\text{のとき} \\ \pi/4 & a_{pp} = a_{qq}\text{のとき} \end{cases} \quad ;$$

$R := R(p, q, \theta)$;

$P_{k+1} := P_k R$;

$A_{k+1} := {}^t R A_k R$ ($= R^{-1} A_k R = P_{k+1}^{-1} A P_{k+1}$) ;

$k := k + 1$;

until (終了条件) ;

A_k の対角成分と P_k の列ベクトルを出力 ;

終了条件 は

$$\begin{cases} \text{非対角成分が全て十分小さい　または} \\ \text{反復回数が十分大きい} \end{cases}$$

a_{pq} の選び方

古典的ヤコビ法：
絶対値が最大である非対角成分 a_{pq} を選ぶ。

シリアル・ヤコビ法：
単純に

for $p := 1$ **to** n **do**

　for $q := p + 1$ **to** n **do**

の順に選んでゆく。

わいヤコビ法：
しきい値 ε を決めておき、上の順に p, q を取って $|a_{pq}| > \varepsilon$ のときのみ処理をする。ε の取り方は例えば、

- 1 周目：$\varepsilon := A$ の非対角成分の絶対値の平均
- 2 周目以降：新$\varepsilon := \frac{1}{10}\varepsilon$

単精度なら 6 周ぐらいで充分。

メモリー・計算量の節約

(1) A_k, P_k は直前のものだけを記憶しておけばよい。

(2) A_k は第 p 行、第 q 行、第 p 列、第 q 列だけが、P_k は第 p 列、第 q 列だけが変化するので、そこだけを計算する。

(3) A_k は対称行列なので、上三角部分だけを記憶すると更にメモリを節約できる。(ただし計算式は若干複雑になる。)

実行例

```
A :
   6.00000   0.00000   1.00000   6.00000   1.00000
   0.00000   2.00000   4.00000   4.00000   3.00000
   1.00000   4.00000   7.00000   8.00000   5.00000
   6.00000   4.00000   8.00000   3.00000   5.00000
   1.00000   3.00000   5.00000   5.00000   8.00000

1st step :
A_1 :
   5.67841  -1.96415   1.96608  -1.33491  -1.48453
  -1.96415   0.32550  -0.65919  -0.10960   0.36852
   1.96608  -0.65919  21.21323   0.50413   0.05047
  -1.33491  -0.10960   0.50413  -5.07016  -0.00000
  -1.48453   0.36852   0.05047  -0.00000   3.85303

P_1 :
   0.80592  -0.21865   0.16557  -0.48174  -0.20782
   0.00000   0.90044   0.34299  -0.23864  -0.12093
  -0.49809  -0.36363   0.61612  -0.28431  -0.39905
   0.31717  -0.00262   0.48062   0.78915  -0.21367
   0.04250  -0.09575   0.49432  -0.08596   0.85865

2nd step :
A_2 :
   6.92678   0.09577   0.03596   0.08614  -0.00295
   0.09577  -0.26514  -0.03691  -0.00229  -0.00027
   0.03596  -0.03691  21.50199   0.00138  -0.00003
   0.08614  -0.00229   0.00138  -5.27911   0.00000
  -0.00295  -0.00027  -0.00003  -0.00000   3.11547

P_2 :
   0.85778   0.08630   0.27060  -0.38010   0.19764
  -0.21141   0.87552   0.30121  -0.19235  -0.24708
  -0.18776  -0.47211   0.56181  -0.39945  -0.51639
   0.22242   0.00991   0.52812   0.80742  -0.13994
  -0.36713  -0.05507   0.49143  -0.08386   0.78336

3rd step :
A_3 :
   6.92858   0.00007   0.00001   0.00000  -0.00000
   0.00007  -0.26647  -0.00000   0.00000  -0.00000
   0.00001  -0.00000  21.50214  -0.00000  -0.00000
   0.00000   0.00000  -0.00000  -5.27972   0.00000
  -0.00000  -0.00000   0.00000  -0.00000   3.11547
```

```
P_3 :
   0.85534   0.07562   0.27254  -0.38611   0.19830
  -0.20163   0.87888   0.29920  -0.19034  -0.24730
  -0.19780  -0.46837   0.56212  -0.39842  -0.51651
   0.22705   0.00729   0.52870   0.80582  -0.13976
  -0.37022  -0.04923   0.49061  -0.08132   0.78307

4-th step :
A_4 :
   6.92858   0.00000  -0.00000  -0.00000  -0.00000
   0.00000  -0.26647  -0.00000  -0.00000  -0.00000
   0.00000   0.00000  21.50214   0.00000  -0.00000
   0.00000   0.00000  -0.00000  -5.27972   0.00000
   0.00000  -0.00000   0.00000  -0.00000   3.11547

P_4 :
   0.85534   0.07562   0.27254  -0.38611   0.19830
  -0.20162   0.87888   0.29920  -0.19034  -0.24730
  -0.19781  -0.46837   0.56212  -0.39842  -0.51651
   0.22705   0.00729   0.52870   0.80582  -0.13976
  -0.37022  -0.04922   0.49061  -0.08132   0.78307

Check

P^{-1} A P :
   6.92858   0.00000  -0.00000  -0.00000   0.00000
   0.00000  -0.26647  -0.00000  -0.00000   0.00000
   0.00000  -0.00000  21.50214  -0.00000   0.00000
  -0.00000  -0.00000   0.00000  -5.27972   0.00000
   0.00000   0.00000   0.00000   0.00000   3.11547
```

3. 直接法

直接法では A は対称行列に限らなくて良い。

直接法のアルゴリズム

(1) A の固有多項式

$$\varphi(t) = \det(tE - A) = t^n + c_1 t^{n-1} + \cdots + c_n$$

を求める。

(2) ニュートン法などを用いて方程式 $\varphi(t) = 0$ を解く。解 $\lambda_1, \cdots, \lambda_n$ が固有値になる。

(3) 各 j について連立一次方程式 $(\lambda_j E - A)\boldsymbol{x} = \boldsymbol{0}$ の $\boldsymbol{0}$ でない解 $\boldsymbol{x} = \boldsymbol{v}_j$ を求める。(例えば x_n に乱数を入れて $x_1, x_2, \cdots, x_{n-1}$ の方程式を作れば、たいていガウスの消去法が使える。)

固有多項式は、数式処理が使えなくても次の Frame 法で計算できる。

Frame 法のアルゴリズム

$X := E$;

for $k := 1$ **to** n **do begin**

$X := AX$;

$c_k := -(X$ **の対角成分和** $) / k$;

$X := X + c_k E$;

end;

実行例

```
A :
   1.0000000   1.0000000  -1.0000000
   3.0000000   4.0000000   3.0000000
   1.0000000   0.0000000  -3.0000000

X :
   1.0000000   1.0000000  -1.0000000
   3.0000000   4.0000000   3.0000000
   1.0000000   0.0000000  -3.0000000

c[1] = -2.000000

X :
   1.0000000   3.0000000   7.0000000
  12.0000000  11.0000000  -6.0000000
  -4.0000000   1.0000000  14.0000000

c[2] = -13.000000

X :
   4.0000000   0.0000000   0.0000000
   0.0000000   4.0000000   0.0000000
   0.0000000   0.0000000   4.0000000

c[3] = -4.000000
```

本講義の概観

- 目的：計算機を用いた、正確かつ高速な数値計算
- 計算機には誤差を生み出す危険がいっぱい

  オーバーフロー、アンダーフロー、丸め誤差、積み残し、桁落ち ...

  → それらに気をつけて
- 扱った計算：
  - 非線形方程式
  - 補間法
  - 解析編：数値微分、数値積分、微分方程式
  - 線形代数編：連立一次方程式、固有値・固有ベクトル
- 手法は大きく分けて 2 通り：
  - 反復法
  - 直接法
- 精度の証明道具：
  - 解析編：テイラー展開
  - 線形代数編：ノルム
- 解析と線形代数の絡み合いも色々：
  - 解析編のガウスの積分公式」に、線形代数の内積の概念が使われていたり、
  - 偏微分方程式が、差分を用いて連立一次方程式に書き直せたり、
  - 固有値計算に微分を使ったニュートン法が使えたり。