第 8 章

固有値問題

行列の固有値・固有ベクトルの定義は既に述べたが，手計算で求めるには手数がかかることは容易にわかる。すべての固有値を求めるには，固有多項式を算出して解く必要があり，更に固有ベクトルは定義式からランク落ちの連立一次方程式を解かねばならない。また，５次以上の正方行列になると固有多項式も５次となり，解の公式が存在しないため近似法を用いて固有値に近い近似値を求めるしか方法がなくなる。現在では，少なくとも対角化可能な行列については高速に固有値・固有ベクトルを解く方法が確立しているため，Scilab でも簡単に求めることができる。本章ではその手法を通じて固有値と固有ベクトルの性質を Scilab を動かしながら確認していく。

8.1　固有値・固有ベクトルの性質

前述したように，n 次正方行列 $A \in \mathbb{C}^{n\times n}$ の固有値 $\lambda_i \in \mathbb{C}(i = 1, 2, ..., n)$ および対応する固有ベクトル $\mathbf{v}_i \in \mathbb{C}^n$, $\mathbf{v}_i \neq 0$, $(i = 1, 2, ..., n)$ は

$$A\mathbf{v}_i = \lambda_i \mathbf{v}_i \ (i = 1, 2, ..., n) \tag{8.1}$$

という関係式を満足するものである。特に固有値はゼロ以外と定義されているので，$(A - \lambda_i)\mathbf{v}_i = 0$ を満足することから，固有ベクトル $\mathbf{v}_i$ を未知数とする連立一次方程式

$$(A - \lambda_i I)\mathbf{v}_i = 0$$

における係数行列 $A - \lambda_i I$ は正則行列にはならない。正則であれば両辺に $(A - \lambda_i I)^{-1}$ を左から乗じることで，$\mathbf{v}_i = 0$ という解しか得られないからである。これは固有ベクトルの定義に反する。

従って，$A - \lambda_i I$ が非正則行列であることから，必ず $\mathrm{rank}(A - \lambda_i I) < n$ となる必要がある。それ故に

$$|A - \lambda I| = \lambda^n + c_{n-1}\lambda^{n-1} + \cdots + c_1\lambda + c_0 = 0 \tag{8.2}$$

という固有多項式 (特性多項式, eigen polynomial) をゼロとする高々 n 個の $\lambda_1, \lambda_2, ..., \lambda_n \in \mathbb{C}$ が固有値ということになる。

「高々 n 個」とは，重複する場合も重複分を含めて n 個とカウントする，という意味である。ここが重要なポイントとなるので，以下，$n = 2$ の場合に限定して $A \in \mathbb{R}^{2\times 2}$ の固有値が

1. $\lambda_1 \neq \lambda_2$ の場合

2. 重解 $\lambda_1 = \lambda_2$ になる場合

に分けて考えることにしよう。

8.1.1　$A \in \mathbb{R}^{2\times 2}$ の例

今，

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \in \mathbb{R}^{2\times 2}$$

の固有多項式 $|A - \lambda I|$ が

$$|A - \lambda I| = \begin{vmatrix} a_{11} - \lambda & a_{12} \\ a_{21} & a_{22} - \lambda \end{vmatrix} = 0$$

を満足するものとしよう。これを

$$\lambda^2 + c_1\lambda + c_0 = 0 \tag{8.3}$$

と書くと，c_1, c_0 はそれぞれ A の要素から計算される定数となる。A が実数行列であることから，c_1, c_0 も必ず実数になる。

さすれば判別式

$$d = c_1^2 - 4c_0 = 0$$

を計算することで，$d = 0$ の時は重解，$d \neq 0$ の時，即ち $d > 0$(実数解) もしくは $d < 0$(複素数解) の時は異なる解を持つことが分かる。

■$d \neq 0$ の場合　例えば

$$A = \begin{bmatrix} 3 & 0 \\ 1 & 2 \end{bmatrix}$$

の時，$d = 1 > 0$ となり，実数解 $3, 2$ を持つ。$\lambda_1 = 3, \lambda_2 = 2$ とすると，それぞれの固有値に対応する固有ベクトル $\mathbf{v}_1 = [v_1^{(1)}\ v_2^{(1)}]^T$, $\mathbf{v}_2 = [v_1^{(2)}\ v_2^{(2)}]^T$ は

$$(A - \lambda_1 I)\mathbf{v}_1 = 0 \Rightarrow \begin{bmatrix} 0 & 0 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} v_1^{(1)} \\ v_2^{(1)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \mathbf{v}_1 = \alpha\begin{bmatrix} 1 \\ 1 \end{bmatrix},\ \alpha\text{は非ゼロの定数}$$

$$(A - \lambda_1 I)\mathbf{v}_1 = 0 \Rightarrow \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} v_1^{(1)} \\ v_2^{(1)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \mathbf{v}_2 = \beta\begin{bmatrix} 0 \\ 1 \end{bmatrix},\ \beta\text{は非ゼロの定数}$$

となる。ここで $\mathbf{v}_1$ の集合 $\mathcal{V}_1$ が

$$\mathcal{V}_1 = \{\, \mathbf{v}_1 \mid (A - \lambda_1 I)\mathbf{v}_1 = 0 \,\} \subset \mathbb{R}^2$$

作れる。この $\mathcal{V}_1$ を固有値 $\lambda_1 = 3$ に対応する固有空間 (eigen space) と呼ぶ。

同様に，固有値 $\lambda_2 = 2$ に対応する固有空間 $\mathcal{V}_2$ は

$$\mathcal{V}_2 = \{\, \mathbf{v}_2 \mid (A - \lambda_2 I)\mathbf{v}_2 = 0 \,\} \subset \mathbb{R}^2$$

となる。この場合，明らかに

$$\mathcal{V}_1 \cap \mathcal{V}_2 = \{\, 0 \,\}$$

であるので，それぞれの固有値に対応する固有ベクトルが一致することはない。つまり，$\mathbf{v}_1 \in \mathcal{V}_1$ と $\mathbf{v}_2 \in \mathcal{V}_2$ は一次独立となり，この二つの固有ベクトルを並べて作った 2 次の正方行列

$$V = [\mathbf{v}_1\ \mathbf{v}_2] = \begin{bmatrix} \alpha & 0 \\ \alpha & \beta \end{bmatrix}$$

は正則行列となる。一例として，$\alpha = \beta = 1$ とおくと

$$V = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

とすると，

$$V^{-1} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$$

となる。この時，相似変換 (similar transformation) によって

$$V^{-1}AV = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{8.4}$$

となることが分かる。このように，固有ベクトルを並べて作った行列を使って相似変換することによって，固有値が対角要素となる対角行列に変換できる行列を，対角化可能 (diagonalizable) な行列，と呼ぶ。つまり，異なる固有値を持つ行列の場合は，必ず対角化できる，ということになる。

■$d = 0$ かつ対角化可能な場合　重解を持つときは対角化可能なケースと対角化不可能なケースに分かれる。まず，対角化可能なケースを見ていくことにしよう。

行列 $A \in \mathbb{R}^{2\times 2}$ を

$$A = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$$

とする。この時，$\lambda_1 = \lambda_2 = 3$ となる。この時 $A - \lambda_1 I = A - \lambda_2 I = O$ であるから，固有ベクトル $\mathbf{v}_1, \mathbf{v}_2$ は任意の非ゼロベクトルであれば何でも良いということになる。つまり

$$\mathcal{V}_1 = \mathcal{V}_2 = \mathbb{R}^2$$

であるから，一時独立なベクトルを二つ $\mathbb{R}^2$ から選ぶことができ，例えば

$$V = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

とすれば，相似変換によって対角化可能であることが分かる。

■$d = 0$ かつ対角化不可能な場合　行列 $A \in \mathbb{R}^{2\times 2}$ を

$$A = \begin{bmatrix} 3 & 1 \\ 0 & 3 \end{bmatrix}$$

とする。この時，$\lambda_1 = \lambda_2 = \lambda = 3$ となるが

$$(A - \lambda I)\mathbf{v} = 0 \Rightarrow \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 0 \Rightarrow v_2 = 0$$

となることから，固有空間 $\mathcal{V}$ は

$$\mathcal{V} = \{\, \mathbf{v} \mid v_2 = 0 \,\}$$

となり，二つの一次独立な固有ベクトルを取ることは出来ない。

しかし

$$(A - \lambda I)^2 = O$$

となることから，一般化固有空間

$$\mathcal{U}_2 = \{\, \mathbf{u}_2 \mid (A - \lambda I)^2 \mathbf{u}_2 = 0 \,\}$$

からもう一つ，$\mathbf{v}$ と一次独立なベクトル $\mathbf{u}_2$ を持ってくることが可能となる。例えば $\mathbf{v} = [1\ 0]^T$ とすれば，$\mathbf{u}_2 = [0\ 1]^T$ として

$$V = [\mathbf{v}\ \mathbf{u}_2] = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$$

とすれば，V は正則行列となり，少なくとも

$$V^{-1} A V = \begin{bmatrix} 3 & 1 \\ 0 & 3 \end{bmatrix}$$

という形式にはなる。これを 2 次の Jordan ブロック (Jordan 標準形) と呼ぶ。

■まとめ　以上をまとめると，$A \in \mathbb{R}^{2\times 2}$ の時は次の (1)〜(3) のケースに分類できる。

(1) **二つの異なる固有値を持つ場合**　・・・対応する固有ベクトルを並べてできる行列 V を用いた相似変換によって対角化可能。

同じ固有値 $\lambda = \lambda_1 = \lambda_2$ を持つ場合

(2) **対角化可能な場合**　・・・固有空間 $\mathcal{V} = \mathbb{R}^2$ の時，即ち，二つの一次独立な固有ベクトルが取れる時，対角化可能。

(3) **対角化不可能な場合**　・・・固有空間 $\mathcal{V} \subset \mathbb{R}^2$ の時，一般化固有空間 $\mathcal{U}_2$ からもう一つの一次独立なベクトル $\mathbf{u}_2$ を取り，$\mathbf{v} \in \mathcal{V}$ と合わせて $V = [\mathbf{v}\ \mathbf{u}_2]$ を作り，相似変換によって二次の Jordan ブロック

$$V^{-1} A V = \begin{bmatrix} \lambda & 1 \\ 0 & \lambda \end{bmatrix}$$

を作ることができる。

問題 8.1

1. (8.3) の係数 c_1, c_0 を A の要素を使って表わせ。
2. 上の関係式を用いて，固有多項式において判別式 d が $d = 0, d > 0, d < 0$ となる行列 A を一つ以上，それぞれ作れ。
3. $A \in \mathbb{R}^{2\times 2}$ がそれぞれ

$$A = \begin{bmatrix} 3 & -2 \\ 2 & 3 \end{bmatrix}, \begin{bmatrix} 4 & 1 \\ -1 & 2 \end{bmatrix}$$

である時，対角化可能か不可能化を答えよ。また不可能な場合は Jordan 標準形を求めよ。

8.1.2 対角化可能な行列と Jordan 標準形

一般の n 次正方行列，$A \in \mathbb{C}^{n\times n}$ に対しては大きく二つのケースに分類される。

(A) **対角化可能な場合** ・・・すべての固有値に対応する固有空間の次元数 (=自由度) を合計すると次元数 n と一致する場合。例えば下記のケースが該当する。

(A-1) **すべて異なる固有値 λ_i を持つ場合** ・・・対応する固有空間 $\mathcal{V}_i$ から固有ベクトル $\mathbf{v}_i \in \mathcal{V}_i$ を持ってくると一次独立となり，この固有ベクトルを並べてできる正則行列 $V = [\mathbf{v}_1\ \mathbf{v}_2\ ...\ \mathbf{v}_n] \in \mathbb{C}^{n\times n}$ を用いて

$$V^{-1}AV = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} = \Lambda \tag{8.5}$$

と対角化できる。

(A-2) **重複固有値の重複度と固有空間の次元数 (=自由度) が一致する場合** ・・・例えば n 個の固有値がすべて重複していても ($\lambda = \lambda_1 = \lambda_2 = \cdots = \lambda_n$)，$\mathcal{V} = \mathbb{C}^n$ であれば，n 個の一次独立な固有ベクトル $\mathbf{v}_i \in \mathbb{C}^n$ が取れ，(8.5) 同様の対角化が可能である。

(B) **対角化不可能な場合** ・・・重複度が 2 以上の固有値に対応する固有空間が一つでも重複度未満の次元数であれば，その部分が Jordan ブロックとなり，対角化は不可能になる。この場合は例えば $\lambda_1 = \lambda_2 = \lambda$ に該当する固有空間の次元数が 1 だとすると，$\mathbf{u}_2 \in \mathcal{U}_2 = \{\, \mathbf{u}_2 \mid (A - \lambda I)^2\mathbf{u}_2 = 0 \,\}$ から一般化固有ベクトル $\mathbf{u}_2 \in \mathcal{U}_2$ を取り，固有ベクトル $\mathbf{v}_1 \in \mathcal{V}_1$ と合わせて $V = [\mathbf{v}_1\ \mathbf{u}_2\ \mathbf{v}_3\ \cdots \mathbf{v}_n]$ を作り

$$V^{-1}AV = \left[\begin{array}{cc|ccc} \lambda & 1 & & & \\ 0 & \lambda & & & \\ \hline & & \lambda_3 & & \\ & & & \ddots & \\ & & & & \lambda_n \end{array}\right] = J \tag{8.6}$$

となる。

一般に，重複度と固有空間の次元数 (=自由度) が一致しない固有値を持つ場合は対角化不可能となる。この場合は一般化固有空間の次元数と同じ次数の Jordan ブロックを持つ。

問題 8.2

$A \in \mathbb{C}^{3\times3}$ の時，A の固有値 $\lambda_1, \lambda_2, \lambda_3$ と固有空間の次元数によってどのような Jordan 標準形 $J = V^{-1}AV$ になるか，整理せよ。

固有値の状態	固有空間の次元数	Jordan 標準形 $J = V^{-1}AV$	対角化可能？
$\lambda_1 = \lambda_2 = \lambda_3$	3	$\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_1 \end{bmatrix}$	○（可能）
	2	$\left[\begin{array}{cc\|c} \lambda_1 & 1 & 0 \\ 0 & \lambda_1 & 0 \\ \hline 0 & 0 & \lambda_1 \end{array}\right]$	×（不可能)
	1		×
$\lambda_1 = \lambda_2 \neq \lambda_3$	2 ($\mathcal{V}_1$ の次元数)	$\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$	○
	1 ($\mathcal{V}_1$ の次元数)		×
$\lambda_1 \neq \lambda_2 \neq \lambda_3$	1 ($\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3$ の次元数)	$\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$	○

8.2　固有値・固有ベクトルが未知の場合

固有値と，それに対応する固有ベクトルを求めるには，spec 関数を使えばよい。

8.2.1　固有値のみを求める場合: eig.sce

spec 関数を用いて固有値のみを求める場合は

```
eig = spec(行列)
```

とすればよい。さすれば，eig に固有値がベクトルの形式で格納される。

```
1: // 次元数: n
2: disp("次元数 n を入力してください：");
3: n = scanf("%d");
4: mprintf("次元数 (n) = %d", n);
5:
6:
7: // 係数行列 A(実対称行列)
8: a = [];
```

```
 9: for i = 1:n
10:     for j = 1:n
11:         a(i, j) = n - max(i, j) + 1;
12:     end;
13: end;
14:
15: // 計算時間計測開始
16: tic();
17:
18: // 固有値計算
19: eig = spec(a);
20:
21: // 計算時間計測終了
22: time_spec = toc();
23: disp( "固有値計算（秒） = ")
24: disp(time_spec);
25:
26: // 固有値出力
27: disp("eig(A) = "); disp(eig);
```

このスクリプトを実行すると次のような結果を得る。

```
 次元数 n を入力してください：
-->5
次元数（n） = 5
 固有値計算（秒） =

    0.012

 eig(A) =

    0.2715541
    0.3532533
    0.5829645
    1.4486906
    12.343538
```

8.2.2　固有値と対応する固有ベクトルを求める場合

前述の spec 関数を用いる際,

```
[V, eig] = spec(A);
```

と呼び出すと，V に固有値に対応した固有ベクトルが格納される。例えば 18 行目から 27 行目を下記のように書き直して実行すると

```
27: // 固有値計算
28: [V, eig] = spec(a);
29:
30: // 計算時間計測終了
31: time_spec = toc();
32: disp( "固有値・固有ベクトル計算（秒）= ")
33: disp(time_spec);
34:
35: // 固有値出力（対角行列形式）
36: disp("lambda(A) = ");
37: disp(lambda);
38:
39: // 固有ベクトル出力（行列形式）
40: disp("V = ");
41: disp(V);
```

以下のような結果を得る。

```
次元数（n）=

   5.

 固有値・固有ベクトル計算（秒）=

   0.

 lambda(A) =

   0.2715541    0.           0.           0.           0.
   0.           0.3532533    0.           0.           0.
```

```
    0.           0.           0.5829645    0.           0.
    0.           0.           0.           1.4486906    0.
    0.           0.           0.           0.           12.343538

 V =

  - 0.1698911    0.3260187    0.4557341  - 0.5485287    0.5968848
    0.4557341  - 0.5968848  - 0.3260187  - 0.1698911    0.5485287
  - 0.5968848    0.1698911  - 0.5485287    0.3260187    0.4557341
    0.5485287    0.4557341    0.1698911    0.5968848    0.3260187
  - 0.3260187  - 0.5485287    0.5968848    0.4557341    0.1698911

 || A - lambda * ev ||_2 / ||ev||_2 =

    4.460D-16

 max(||A - lambda(i, i) * ev(:,i)||_2 / ||ev(:,i)||_2) =

    4.166D-15
```

8.3 固有値・固有ベクトルが既知の問題の作り方

実験的に，固有値と固有ベクトル，あるいは Jordan 標準形が既知の問題を使って，本当にプログラムが正確な固有値・固有ベクトルを求めることができているか，確認したいことがある。そのような場合は (8.5) や (8.6) の関係式を使って，あらかじめ対角行列 Λ や Jordan 標準形 J を与えておき，適当な正則行列 V を使って

$$A = V\Lambda V^{-1} \text{ または } VJV^{-1} \tag{8.7}$$

として行列 $A \in \mathbb{C}^{n\times n}$ を生成すればよい。

8.4 べき乗法と逆べき乗法

べき乗法は最も単純な，絶対値最大固有値 $\lambda_1 = \lambda_1(A)$ とそれに属する固有ベクトル $\mathbf{v}_1$ を同時に求める方法である。もし全ての固有値が相異なる ($i < j$ の時，$\lambda_i \neq \lambda_j$, かつ，$|\lambda_i| > |\lambda_j|$) ならば，各固有値 $\lambda_i = \lambda_i(A)$ に属する固有ベクトル $\mathbf{v}_i$ は n 次元線型空間の基底となるため，任意のベクトル $\mathbf{x}_0$ は

$$\mathbf{x}_0 = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_n\mathbf{v}_n$$

と表現できる。従って $\mathbf{x}_k := A^k\mathbf{x}_0$ とすれば

$$\mathbf{x}_k = (\lambda_1)^k \left\{ c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^k \mathbf{v}_2 + \cdots + c_n\left(\frac{\lambda_n}{\lambda_1}\right)^k \mathbf{v}_n \right\}$$

であるから，

$$\mathbf{x}_k = (\lambda_1)^k c_1\mathbf{v}_1 + O\left(\left(\frac{|\lambda_2|}{|\lambda_1|}\right)^k\right)$$

となり，固有ベクトル $\mathbf{v}_1$ へ収束する。さすれば固有値 λ_1 は Reiley 商

$$\lambda_1 \approx \frac{(A\mathbf{x}_{k+1}, \mathbf{x}_k)}{(\mathbf{x}_k, \mathbf{x}_k)}$$

を計算することで得られる。実際には overflow を防ぐため，反復一回ごとに $\|\mathbf{x}_k\| = 1$ となるように正規化する。

1. 初期ベクトル $\mathbf{x}_0$(ここで $\|\mathbf{x}_0\| = 1$) を決める。
2. for $k = 0, 1, 2, \ldots$
   （a）$\mathbf{y}_{k+1} := A\mathbf{x}_k$
   （b）$\gamma_{k+1} := (\mathbf{y}_{k+1}, \mathbf{x}_k)/(\mathbf{x}_k, \mathbf{x}_k)$
   （c）収束判定
   （d）$\mathbf{x}_{k+1} := \mathbf{y}_{k+1}/\|\mathbf{y}_{k+1}\|$

このアルゴリズムに従うと，γ_k が $\lambda_1(A)$ へ，$\mathbf{x}_k$ はそれに属する固有ベクトルへと収束する。収束判定は固有値の近似値 γ_k，あるいは固有ベクトルの近似値 $\mathbf{x}_k$ を見て判断する

■べき乗法の計算例　実対称行列 A を

$$A = \begin{bmatrix} 5 & 4 & 3 & 2 & 1 \\ 4 & 4 & 3 & 2 & 1 \\ 3 & 3 & 3 & 2 & 1 \\ 2 & 2 & 2 & 2 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

とする。この時，IEEE754 倍精度で計算すると，べき乗法の場合以下のような結果を得る。

$\lambda_1(A)$ の近似値が

```
Maximum Eigenvalue:   1.23435375196795842e+01
```

になっている時の固有ベクトルの近似値 $\mathbf{x}$，及び $A\mathbf{x}$ の各要素の $\mathbf{x}$ との比をそれぞれ出力すると

```
i       eigenvector[i]        A * eivenvector[i] / eigenvector[i]
0   2.23606797749978981e+00   1.23435375196795842e+01
1   2.05491504837138317e+00   1.23435375196779056e+01
2   1.70728512307438196e+00   1.23435375196750883e+01
3   1.22134111072129303e+00   1.23435375196720223e+01
4   6.36451305172487269e-01   1.23435375196696810e+01
```

となる。

逆べき乗法は A の代わりに A^{-1} を用いることで，A の絶対値最小固有値とそれに対応する固有ベクトルを求める方法である。正則行列 A の固有値と対応する固有ベクトルが $\lambda_i, \mathbf{v}_i$ である時，

$$A\mathbf{v}_i = \lambda_i \mathbf{v}_i \Rightarrow A^{-1}\mathbf{v}_i = \frac{1}{\lambda_i}\mathbf{v}_i$$

であることから，A^{-1} の固有値は A の固有値の逆数 $1/\lambda_i$ であることが分かる。また，固有ベクトルは変化しない。

実際に計算する際には A^{-1} を直接求めるのではなく，$A = LU$ と LU 分解しておき，$\mathbf{z}_{k+1}$ を未知数とする連立一次方程式

$$(LU)\mathbf{z}_{k+1} = \mathbf{z}_k$$

を前進代入・後退代入で求める。

8.5 LR(LU) 分解法，QR 分解法による固有値の近似解法

Scilab の基盤となっている LAPACK における固有値問題は，通常，QR 分解法に基づいたアルゴリズムが使用されている。ここではその詳細は述べないが，基本となる考え方は QR 分解を繰り返すだけなので至って簡単である。

8.5.1 LR 分解法

今，行列 $A \in \mathbb{C}^{n\times n}$ があるとすると，これを $A_0 := A$ として，A_{k+1} を以下のように求める。

1. $A_k = L_k R_k$ となるよう，A_k を LU(LR) 分解する。
2. $A_{k+1} := R_k L_k$

実際に Scilab で計算した結果を以下に示す。

```
-->A = [1, 1/2, 1/3; 1/2, 1/3, 1/4; 1/3, 1/4, 1/5];

-->[L, R] = lu(A)
 R  =

    1.    0.5          0.3333333
    0.    0.0833333    0.0888889
    0.    0.         - 0.0055556
 L  =

    1.           0.    0.
    0.5          1.    1.
```

```
    0.3333333    1.    0.

-->A = R * L
 A  =

    1.3611111    0.8333333    0.5
    0.0712963    0.1722222    0.0833333
  - 0.0018519  - 0.0055556    0.

-->[L, R] = lu(A)
 R  =

    1.3611111    0.8333333    0.5
    0.           0.1285714    0.0571429
    0.           0.           0.0026455
 L  =

    1.           0.           0.
    0.0523810    1.           0.
  - 0.0013605  - 0.0343915    1.

-->A = R * L
 A  =

    1.4040816    0.8161376    0.5
    0.0066569    0.1266062    0.0571429
  - 0.0000036  - 0.0000910    0.0026455

-->[L, R] = lu(A)
 R  =

    1.4040816    0.8161376    0.5
    0.           0.1227368    0.0547723
    0.           0.           0.0026865
 L  =

    1.           0.           0.
    0.0047411    1.           0.
```

```
  - 0.0000026  - 0.0007242    1.

-->A = R * L
 A  =

    1.4079498    0.8157754    0.5
    0.0005818    0.1226971    0.0547723
  - 6.887D-09  - 0.0000019    0.0026865

-->[L, R] = lu(A)
 R  =

    1.4079498    0.8157754    0.5
    0.           0.1223600    0.0545657
    0.           0.           0.0026873
 L  =

    1.           0.           0.
    0.0004132    1.           0.
  - 4.891D-09  - 0.0000159    1.

-->A = R * L
 A  =

    1.4082869    0.8157675    0.5
    0.0000506    0.1223592    0.0545657
  - 1.314D-11  - 4.264D-08    0.0026873
```

8.5.2 QR 分解法

今，行列 $A \in \mathbb{C}^{n\times n}$ があるとすると，これを $A_0 := A$ として，A_{k+1} を以下のように求める。

1. $A_k = Q_k R_k$ となるよう，A_k を QR 分解する。
2. $A_{k+1} := R_k Q_k$

実際に Scilab で計算した結果を以下に示す。

```
-->A = [1, 1/2, 1/3; 1/2, 1/3, 1/4; 1/3, 1/4, 1/5];
```

```
-->[Q,R] = qr(A);


-->Q
 Q  =

  - 0.8571429    0.5016049    0.1170411
  - 0.4285714  - 0.5684856  - 0.7022469
  - 0.2857143  - 0.6520864    0.7022469

-->R
 R  =

  - 1.1666667  - 0.6428571  - 0.45
    0.         - 0.1017143  - 0.1053370
    0.           0.           0.0039014

-->A = R * R
 A  =

    1.3611111    0.8153878    0.5909610
    0.           0.0103458    0.0103033
    0.           0.           0.0000152

-->A = R * Q
 A  =

    1.4040816    0.0736882  - 0.0011147
    0.0736882    0.1265120  - 0.0025440
  - 0.0011147  - 0.0025440    0.0027397

-->[Q,R] = qr(A);

-->A = R * Q
 A  =

    1.4082869    0.0064222    0.0000021
```

```
    0.0064222    0.1223591    0.0000544
    0.0000021    0.0000544    0.0026874

-->[Q,R] = qr(A);

-->A = R * Q
 A  =

    1.4083187    0.0005578  - 4.066D-09
    0.0005578    0.1223273  - 0.0000012
  - 4.066D-09  - 0.0000012    0.0026873

-->[Q,R] = qr(A);

-->A = R * Q
 A  =

    1.4083189    0.0000485    7.759D-12
    0.0000485    0.1223271    2.626D-08
    7.759D-12    2.626D-08    0.0026873
```

8.6 代数方程式を固有値問題として解く方法

本章の最初に述べたように，行列の固有値を求める問題は，本質的に代数方程式 (固有方程式)

$$|A - \lambda I| = 0$$

の解を求める問題と同じである。では逆に，最高次数の係数が 1 である n 次多項式 $p(\lambda)$ が

$$p(\lambda) = \lambda^n + c_{n-1}\lambda^{n-1} + c_{n-2}\lambda^{n-2} + \cdots + c_1\lambda + c_0 \ (c_i \in \mathbb{C})$$

と与えられる時の代数方程式

$$p(\lambda) = 0 \tag{8.8}$$

が与えられたとき，これを固有方程式と見立てることの行列を作ることができれば，既に述べた行列の固有値を求める Scilab スクリプトを用いてこの代数方程式の解を求めることもできることになる。結論か

ら言うと，このような固有多項式を持つ行列は下記のようになる。

$$C = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & \cdots & \cdots & 0 & 1 \\ -c_0 & -c_1 & \cdots & \cdots & -c_{n-2} & -c_{n-1} \end{bmatrix} \tag{8.9}$$

これをコンパニオン行列 (companion matrix) と呼ぶ。この時,

$$|C - \lambda I| = |(-I)(\lambda I - C)| = (-1)^n |\lambda I - C| = (-1)^n p(\lambda)$$

となる。よって，$p(\lambda) = 0$ が与えられれば，対応するコンパニオン行列を自動的に得ることができる。

例えば

$$x^3 - 6x^2 + 11x - 6 = 0$$

という代数方程式に対しては

$$C = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 6 & -11 & 6 \end{bmatrix}$$

というコンパニオン行列が対応する。もし最高次数の係数 c_n が 1 でなければ，全ての係数を c_n で割ってからコンパニオン行列を作ればよい。

これを利用すると，代数方程式の係数が与えられれば，対応するコンパニオン行列を作り，その固有値を spec 関数から求めて代数方程式の解とすればよい。

これを eig_algebraic_eq.sce という Scilab スクリプトにしてみよう。要は，eig.sce の行列生成部をコンパニオン行列に修正するだけなので，前述の eig.sce の 1 行目〜19 行目を次のように置き換えればよい。

```
// 次元数: n
disp("代数方程式の次数 n を入力してください：");
n = scanf("%d");
mprintf("次数 (n) = %d", n);

disp("代数方程式の係数 (c(0), c(1), ..., c(n-1)) を入力してください：");
c = []; // 添え字 1 から開始
for i = 1 : n
    mprintf("c(%d) = ", i - 1); c(i) = scanf("%f");
end

// コンパニオン行列 C
C = zeros(n, n);
for i = 1 : n
```

```
    C(n, i) = -c(i);
end;
for i = 1 : n - 1
    C(i, i + 1) = 1;
end

// 計算時間計測開始
tic();

// 固有値計算
eig = spec(C);
```

先の例を入力してみると，確かに代数方程式の解が得られていることが分かる。

```
 代数方程式の次数 n を入力してください：
-->3
次数 (n) = 3
 代数方程式の係数 (c(0), c(1), ..., c(n-1)) を入力してください：
c(0) =
-->-6
c(1) =
-->11
c(2) =
-->-6

 固有値計算（秒） =

    0.001

 eig(C) =

    1.
    2.
    3.
```

問題 8.3

次の代数方程式の解を eig_algebraic_eq.sce を用いて求めよ。

1. $x^3 + x^2 + x + 1 = 0$
2. $x^{10} - 1 = 0$

3. $x^5 + 4x^4 + 3x^3 + 2x^2 + 1 = 0$

8.7　コンパニオン行列の応用：線型漸化式によって定義される数列の一般項

数列 $\{x_n\}_{n=0}^{\infty}$ が線型漸化式

$$x_{n+m} := -c_m x_{n+m-1} - c_{m-1} x_{n+m-2} - \cdots - c_1 x_n \ (c_i \in \mathbb{R} \text{ は定数}) \tag{8.10}$$

を満足している時，この漸化式によって定義される数列 $x = \{x_n\}_{n=0}^{\infty}$ を要素とする集合 X_m は m 次線形空間となる。実際，任意の $x, y = \{y_n\}_{n=0}^{\infty} \in X_m$ に対して，加算とスカラー倍を

$$x + y = \{x_n + y_n\}_{n=0}^{\infty},\ \alpha x = \{\alpha x_n\}_{n=0}^{\infty}$$

と定義すると，線型空間 (定義 3.2) の性質をすべて満足することが分かる。

では，任意の線型漸化式 (8.10) によって生成される数列 $x = \{x_n\}_{n=0}^{\infty}$ の一般項はどのように表現できるか？　これを固有値・固有ベクトルを使って求める方法を考える。

8.7.1　$m = 1, 2$ の場合

もっとも簡単な X_1, 即ち $m = 1$ の場合を考えよう。この場合は例えば

$$x_{n+1} := -3x_n$$

という漸化式になるので，初期値 x_0 が決まれば任意の x_n (一般項) は

$$x_n = (-3)^n x_0$$

となることはすぐにわかる。

では $m = 2$ の時，例えば

$$x_{n+2} := -3x_{n+1} - 2x_n \tag{8.11}$$

の時はどうなるだろう？　もちろん，初期値としては x_0, x_1 がセットで決まらなければ任意の x_n を決定することはできないことになる。

この場合は次のように考える。まず $(x_0, x_1) = (1, 0)$ のケースの場合は $x_2 = (-2)x_0 = -2$, $x_3 = (-3)x_2 = 6$ となり，以後の数列が決定される。この数列を $e_1 = \{1, 0, -2, 6, ...\}$ と置く。同様に，$(x_0, x_1) = (0, 1)$ のケースに作られる数列を $e_2 = \{1, 0, -3, 7, ...\}$ と置く。

写像 $\varphi : X_2 \to \mathbb{R}^2$ を

$$\varphi(x) = \varphi(\{x_0, x_1, ..., x_n, ...\}) = [x_0\ x_1]^T$$

とすると

$$\varphi(e_1) = [1\ 0]^T,\ \varphi(e_2) = [0\ 1]^T$$

であるから，任意の数列 $x = \{x_0, x_1, ..., x_n, ...\}$ に対しては

$$\varphi(x) = x_0\varphi(e_1) + x_1\varphi(e_2) = x_0\mathbf{e}_1 + x_1\mathbf{e}_2 \in \mathbb{R}^2$$

と見ることができる。

今，写像 T を，数列 $x = \{x_0, x_1, ..., x_n, ...\}$ から漸化式 (8.11) に基づいて一つ後ろにずらした数列 $\{x_1, x_2, ..., x_{n+1}, ...\}$ に対応させるものとする。さすれば，これに対応する $\mathbb{R}^2$ 上における線型変換は

$$\begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

になることは自明である。つまり

$$\begin{array}{ccccc} x = \{x_0, x_1, ..., x_n, ...\} & \longrightarrow T \longrightarrow & & & T(x) = \{x_1, x_2, ..., x_{n+1}, ...\} \\ \downarrow & & & & \uparrow \\ \varphi & & & & \varphi^{-1} \\ \downarrow & & & & \uparrow \\ \varphi(x) = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} & \longrightarrow C_2 \longrightarrow & C_2\varphi(x) = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} & = & \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \end{array} \tag{8.12}$$

という流れになる。もし C_2 の固有値 λ_1, λ_2 が異なれば対角化可能，即ち，それぞれの固有値に対する固有ベクトル $\mathbf{v}_1, \mathbf{v}_2$ が一次独立となるので，行列 $P = [\mathbf{v}_1\ \mathbf{v}_2]$ は正則行列となるので，連立一次方程式

$$P \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \varphi(x)$$

の解 $[\alpha_1\ \alpha_2]^T$ は一意に定まることになる。よって，任意の数列は

$$\varphi(x) = \alpha_1\mathbf{v}_1 + \alpha_2\mathbf{v}_2$$

と表現できる。即ち

$$C_2\varphi(x) = \lambda_1(\alpha_1\mathbf{v}_1) + \lambda_2(\alpha_2\mathbf{v}_2)$$

となるので，任意の数列 x は

$$\begin{aligned} T(x) = \varphi^{-1}(C_2\varphi(x)) &= \varphi^{-1}(\lambda_1\alpha_1\mathbf{v}_1) + \varphi^{-1}(\lambda_2\alpha_2\mathbf{v}_2) \\ &= \lambda_1\alpha_1\varphi^{-1}(\mathbf{v}_1) + \lambda_2\alpha_2\varphi^{-1}(\mathbf{v}_2) \\ &= \lambda_1\alpha_1 v_1 + \lambda_2\alpha_2 v_2 \end{aligned}$$

と表現できる。ここで v_1, v_2 は C_2 の固有ベクトルと対応付けされる数列で，$v_1 = \{v_n^{(1)}\}_{n=0}^{\infty} = \varphi^{-1}(\mathbf{v}_1)$, $v_2 = \{v_n^{(2)}\}_{n=0}^{\infty} = \varphi^{-1}(\mathbf{v}_2) \in X_2$ である。

さすれば

$$T(x) = \{x_1, x_2, ..., x_{n+1}, ...\} = \{\lambda_1\alpha_1 v_0^{(1)} + \lambda_2\alpha_2 v_0^{(2)}, \lambda_1\alpha_1 v_1^{(1)} + \lambda_2\alpha_2 v_1^{(2)}, ..., \lambda_1\alpha_1 v_n^{(1)} + \lambda_2\alpha_2 v_n^{(2)}, ...\}$$

と表現できる。これを繰り返すことで

$$\begin{aligned} T^n(x) = T(T(\cdots T(x)\cdots)) &= \varphi^{-1}(C_2^n\varphi(x)) \\ \Rightarrow \{x_n, x_{n+1}, ...\} &= \{\lambda_1^n\alpha_1 v_0^{(1)} + \lambda_2^n\alpha_2 v_0^{(2)}, \lambda_1^n\alpha_1 v_1^{(1)} + \lambda_2^n\alpha_2 v_1^{(2)}, ...\} \end{aligned} \tag{8.13}$$

となるので，一般項 x_n は

$$x_n = \lambda_1^n\alpha_1 v_0^{(1)} + \lambda_2^n\alpha_2 v_0^{(2)} \tag{8.14}$$

と表現できる。実際 (8.14) の場合,

$$\lambda_1 = -1, \mathbf{v}_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix},\ \lambda_2 = -2, \mathbf{v}_2 = \begin{bmatrix} -1 \\ 2 \end{bmatrix} \Rightarrow P = [\mathbf{v}_1\ \mathbf{v}_2] = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix} \Rightarrow P^{-1} = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

より,

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = P^{-1} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 2x_0 + x_1 \\ x_0 + x_1 \end{bmatrix}$$

となり，かつ $v_1 = \varphi^{-1}(\mathbf{v}_1) = \{1, -1, ...\}, v_2 = \varphi^{-1}(\mathbf{v}_2) = \{-1, 2, ...\}$ となるから，一般項 (8.14) は

$$x_n = (-1)^n(2x_0 + x_1) - (-2)^n(x_0 + x_1) \tag{8.15}$$

と表わすことができる。

■linear_diff.sce スクリプト　2 項線型漸化式の係数 c_0, c_1，初期値 x_0, x_1 が与えられた時，一般項を計算する Scilab スクリプト (`linear_diff.sce`) を下記に示す。

```
 1: clear;
 2:
 3: // 漸化式に基づく数列の第 n 項計算
 4: // x_{n+2} := c1 * x_{n+1} + c0 * x_n
 5: function xn = seq_n(n, c, x)
 6:     temp = [];
 7:     temp(1) = x(1);
 8:     temp(2) = x(2);
 9:
10:     if n == 0 then
11:         xn = temp(1);
12:     elseif n == 1 then
13:         xn = temp(2);
14:     else
15:         for i = 2:n
16:             xn = c(2) * temp(2) + c(1) * temp(1);
17:             temp(1) = temp(2);
18:             temp(2) = xn;
19:         end;
20:     end;
21: endfunction;
22:
23: // 線型漸化式
24: // x_{n+2} := c1 * x_{n+1} + c0 * x_n
25: // c = [c0; c1]
```

```
26: c0 = -2;
27: c1 = -3;
28:
29: // 初期値
30: // x = [x0; x1]
31: x0 = 1; x1 = 0;
32: //x0 = 0; x1 = 1;
33:
34: // 数列
35: c = [c0; c1];
36: x = [x0; x1];
37: printf("漸化式: x_{n+2} := (%g) * x_{n+1} + (%g) * x_n\n", c(2), c(1));
38: printf("初期値: x0 = %g, x1 = %g\n", x(1), x(2));
39:
40: // Companion Matrix
41: C = [0, 1; c(1), c(2)];
42: disp("C = "); disp(C);
43:
44: [P, lambda] = spec(C);
45: disp("P = "); disp(P);
46: disp("lambda = "); disp(lambda);
47:
48: // alpha
49: alpha = inv(P) * x;
50:
51: // 一般項
52: printf("\n一般項：\n");
53: printf("xn = ((%g) + (%g) * i)^n * ((%g) + (%g) * i) * ((%g) + (%g) * i)
    + ((%g) + (%g) * i)^n * ((%g) + (%g) * i) * ((%g) + (%g) * i)\n",
    real(lambda(1, 1)), imag(lambda(1, 1)) , real(alpha(1)), imag(alpha(1)),
    real(P(1, 1)), imag(P(1, 1)), real(lambda(2, 2)), imag(lambda(2, 2)),
    real(alpha(2)), imag(alpha(2)), real(P(1, 2)), imag(P(1, 2)));
54: printf("\n");
55:
56: // 検算
57: printf(" n        漸化式計算        一般項計算\n");
58: for n = 0:10
59:     printf("%2d %15.7g %15.7g\n", n, seq_n(n, c, x), lambda(1, 1)^n * alpha(1)
```

```
    * P(1, 1) + lambda(2,2)^n * alpha(2) * P(1, 2));
60: end;
```

■実行結果　上記の linear_diff.sce を実行すると下記のような結果を得る。漸化式に基づく x_2〜x_{10} の値と，一般項の定義式に基づく値が一致していることが分かる。

漸化式: x_{n+2} := (-3) * x_{n+1} + (-2) * x_n
初期値: x0 = 1, x1 = 0

```
 C =

    0.    1.
  - 2.  - 3.

 P =

    0.7071068  - 0.4472136
  - 0.7071068    0.8944272

 lambda =

  - 1.    0
    0   - 2.
```

一般項：

```
xn = ((-1) + (0) * i)^n * ((2.82843) + (0) * i) * ((0.707107) + (0) * i)
 + ((-2) + (0) * i)^n * ((2.23607) + (0) * i) * ((-0.447214) + (0) * i)
```

n	漸化式計算	一般項計算
0	1	1
1	0	0
2	-2	-2
3	6	6
4	-14	-14
5	30	30
6	-62	-62
7	126	126
8	-254	-254

```
 9             510             510
10           -1022           -1022
```

問題 8.4

1. 線型漸化式 (8.11) によって規定される数列 $x \in X_2$ に対して次の問いに答えよ。
   （a）$T(v_1) = \lambda_1 v_1, T(v_2) = \lambda_2 v_2$ であることを確認せよ。
   （b）一般項 x_n が (8.15) として表現できることを数学的帰納法を用いて答えよ。
2. $x_0 = x_1 = 1$ という初期値を取る時，線型漸化式

$$x_{n+2} = x_{n+1} + x_n$$

   で規定される数列をフィボナッチ数列と呼ぶ。この一般項 x_n を求めよ。

8.7.2　X_m の場合

以上見てきたように，線型漸化式 (8.10) の一般項 x_n は，写像 T と $\varphi : X_m \to \mathbb{R}^m$,

$$\varphi(x) = \varphi(\{x_0, x_1, ..., x_{m-1}, ...\}) = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{m-1} \end{bmatrix}$$

そしてコンパニオン行列 C_m

$$C_m = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & \cdots & \cdots & 0 & 1 \\ -c_1 & -c_2 & \cdots & \cdots & -c_{m-1} & -c_m \end{bmatrix}$$

が対角化可能である場合，(8.12) 同様，次のような写像の構造ができる。

$$\begin{array}{ccccc} x = \{x_0, x_1, ..., x_n, ...\} & \longrightarrow T \longrightarrow & T(x) = \{x_1, x_2, ..., x_{n+1}, ...\} \\ \downarrow & & \uparrow \\ \varphi & & \varphi^{-1} \\ \downarrow & & \uparrow \\ \varphi(x) = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{m-1} \end{bmatrix} & \longrightarrow C_m \longrightarrow & C_m\varphi(x) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \end{array} \tag{8.16}$$

従って，次のようにして一般項 x_n を求めることができる。

1. 初期値 $\mathbf{x}_0 = [x_0\ x_1\ ...\ x_{m-1}]^T$ を定める。
2. C_m の固有値 $\lambda_1, \lambda_2, ..., \lambda_m$ と対応する固有ベクトル $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_m$ を求め，$v_1 = \varphi(\mathbf{v}_1) = \{v_0^{(1)}, ...\}$, $v_2 = \varphi(\mathbf{v}_2) = \{v_0^{(2)}, ...\}, ..., v_m = \varphi(\mathbf{v}_m) = \{v_0^{(m)}, ...\}$ を得る。

3. $P = [\mathbf{v}_1\ \mathbf{v}_2\ ...\mathbf{v}_m]$ を作り，連立一次方程式 $P\alpha = \mathbf{x}_0$ を解いて $\alpha = [\alpha_1\ \alpha_2\ ...\ \alpha_m]^T$ を求める。
4. $x_n = \lambda_1^n \alpha_1 v_0^{(1)} + \lambda_2^n \alpha_2 v_0^{(2)} + \cdots + \lambda_m^n \alpha_m v_0^{(m)}$ を得る。

問題 8.5

1. $T(x)$ が線型変換，即ち，任意の定数 α, β および $x, y \in X_m$ に対して，

$$T(\alpha x + \beta y) = \alpha T(x) + \beta T(y)$$

   であることを示せ。
2. 線型漸化式

$$x_{n+3} = -6x_{n+2} + 11x_{n+1} - 6x_n$$

   に対して初期値 x_0, x_1, x_2 を与えた時に一般項を求める Scilab プログラムを作れ。