固有ベクトル計算法

志村氏論文の理解の為に

中野 嘉弘 （札幌市、後 3 年で 91 歳）

FAX 専 011-588-3354

E-mail: yoshihiro@river.ocn.ne.jp

は し が き

Yahoo 知恵袋・数学カテゴリーは、年寄りのホビーとして最適である。

固有値問題、特に、固有ベクトルの質問が多い。 世の中に、良い参考書の

少ない証拠であろう。 私は J 言語の御蔭で、若者のお役にたつ機会が多い。

しかし、J 言語そのものの「普及は今一つ」の恨みがある。

この 8 月の JAPLA の恒例・夏季合宿の成果として、有能なる世話人・志村正人

氏の報告（文献 1）を拝見した。 大変、有益であった。

関係して、気が付いた事を述べて置きたい。

1． 固有ベクトルの計算は難物 ？

●質問（ikmh212211 さん・文献 2）:

2x2 行列 1 3 4 2 の固有値は 5 と _2 と判りました。 後者の

固有ベクトルは （_1， 1） とも判りましたが、固有値 5 の方は判りません。

答が （0 0） になって困ります。 お願いします！

◎ 私の回答（文献 2）: 東南アジアで好評なカナダ産の J 言語なら簡単！

与行列 ikm2=.>1 3; 4 2 と定義する。 ルヴェリエ・ファデーエフ法

の関数 LF0 で、 LF0 ikm2 から、解の固有値は 5, _2 です。

固有ベクトルの計算は、関数（中野） eigvec を用い、

5 _2 eigvec ikm2 から（_4 4）、即ち 4 で割って、（_1 1） と

(3 4) である。

[検算] は、 ip を 「行列とベクトル」の 内積関数として、例えば、

ikm2 ip (3 4) -> (15 20)、 更に固有値 5 で割って、

(15 20) % 5 -> (3 4)、確かに 固有ベクトルに戻った。

●質問(ppink0917 さん・文献 3):

3x3 対称行列 ppi3 即ち 2 0 1 0 2 1 1 1 3 の固有値 1 2 4 は判りました。 しかし、固有値 2 の時の固有ベクトルが判りません。途中で Z = 0 となってしまい、そのあとの続け方が判りません。 宜しく!

◎ 回答は、後に例題として述べる。

★ 志村報告(文献 1)中にも、しきりと、同じ悩みを云々して居る。

例えば、p.28、先頭行に「ランク落ちして逆行列もとれない厄介者が多い」と告白して居る。

2. 志村報告流の処方

有効な LF 法による諸関数の Scripts を挙げて置く。

p.23: tr=: (<0 1)&|: NB. diag

志村 char_lf (TR_SUM 等が stack error を起こすらしいので、中野はその代わりに、自前の関数 nchar_lf を用いた。)

p.29: 志村 char_lf_evec_sub

p.33: 志村 char_evec (stack error を起こすらしい上記の関数 char_lf を含む。)

私は多少のエラーらしきものを避けるため、下記の如き自前の関数を使っている。しかし、大筋に於いて中野流は、志村流と大差は無い。

```
ip=: idot=: +/ . *   NB. inner product
NB.  Leverrier-Faddeev method by Nakano
LF=: >@{:@p.@charn  (途中経過の印刷あり)
LF0=: >@{:@p.@charn0  (結果のみ印刷)
```

```
  charn0 =: 3 : 0
 In=. =@i.n=. # y
X=.In
i=.0
p=.1
 for_k. >: i.n do.
 X =. y + / . * X
 trX =. +/(<0 1)|:X
 pk=.-k%~trX
p=.p,pk
  X=.X+ pk * In
i=.i + 1
end.
  |.p
)

 NB. by Nakano 2010 Sep.19

 nchar_lf =: 3 : 0
1; (LF0 y);charn0 y
)

nchar_evec=: 3 : 0  NB. by Nakano
NB. char_evec=: 3 : 0  NB. Shimura
NB.  EIGEN=: {@> ; 1{ char_lf y  NB. Shimura
EIGEN=: {@> ; 1{ nchar_lf y  NB. by Nakano
 EIGEN2=: {@> L:0 EIGEN ^/L:0 |.i.# EIGEN
ADJMAT=: char_lf_evec_sub y
ANS=.<''
for_LF. i. # y do.
TMP=. +/> (> LF{ EIGEN2) * L:0 ADJMAT
ANS=. ANS,<TMP
end.
  EIGEN,:}.ANS
)
```

例題: 上記の行列 ppi3 にて、固有値 4 2 1。

```
   ppi3
2 0 1
0 2 1
1 1 3

  nchar_evec ppi3
┌─────────────────────┐
│4    │2      │1       │
├─────────────────────┤
│1 1 2│_1  1 0│ 1  1 _1│
│1 1 2│ 1 _1 0│ 1  1 _1│
│2 2 4│ 0  0 0│_1 _1  1│
└─────────────────────┘
```

［検算］

```
   ppi3 ip (1 1 2)
4 4 8
4 4 8 % 4
1 1 2

  ppi3 ip (_1 1 0)
_2 2 0
_2 2 0 %2
_1 1 0

  ppi3 ip (1 1 _1)
1 1 _1
```

確かに、予想通りに戻った。

3.　随伴マトリックス　の話題

志村報告(文献 1) の pp.28-29 に、丁寧な説明があるが、余因子行列の個数

が多い時 には、数値計算が大変である(志村例では 3 次)。

幸い、4 次の場合の丁寧な解説を掲載する本に遭遇した(文献 4 pp.100-103)。

このやり方を、J 言語でプログラム化して見た。 関数名 adjmat。

(wr は作業中の便宜で入れたものが多い。 適当に省略可。)

```
  adjmat=: 3 : 0

 k=.1
 wr A=.y
wr  uA=.un n=.#A
 k=.1
wr ' k = ', ":k
 c1=. +/ diag A
wr 'c1 = '
wr  c1

 wr A1=.(|: A)- c1*uA
 wr 'ans'
  A1
  :
  wr A=. y
  wr A1=. x
  k=.1
label_k2.

 k=.k+1

 wr ' k = ', ": k
 ck=. (+/ +/"1 A*A1)%k
 wr 'ck = ', ": ck
```

```
 R=.0
 i=.0
 while. i < n =. # y do.

 wr 'i=  ', ":i
 wr R
  j=.0

 while. j < (# A) do.
 wr 'j  =', ":j
 wr ja=.j{A
 R=.R ,  (+/ (j{A) * i{A1)
   j=.j+1
  end.
   i=.i+1
  end.
wr Rn=.  (n,n) $ }. R

 wr 'ans = '
wr  Rnn=. Rn-ck*uA=.un #A
label_kn.
wr 'next ck '
wr  dag=.(+/"1 A*Rnn)
wr  ck=. (+/dag)%(k+1)
 A1=.Rnn

wr ' k = ', ":(k+1)
 if. k< (#A) do. goto_k2.  end.

  ' '
   )
```

未だ作業中のものであるが、名人達の改良を待ちたいので、敢えて

書いて置く。

古屋先生の原著の p.100 の例の 4 次行列 A を FA4 と命名して

トライした初段の計算結果を、以下に示す。

```
  adjmat FA4
 1  3 _2  5
 2  1  4  0
_1 _6  3  7
 5 _4 _1 _3
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
 k = 1
c1 =
2
_1  2 _1  5
```

```
 3 _1 _6 _4
_2  4  1 _1
 5  0  7 _5
ans
_1  2 _1  5
 3 _1 _6 _4
_2  4  1 _1
 5  0  7 _5
```

この結果 ans を、左データとして(右データは不変)、同様の計算 adjmat

を続ければ、最後まで進行して、停止する。

その各段階の出力 ck から、固有多項式の各次の係数が判る。

```
  ]A10=.> _1  2 _1  5; 3 _1 _6 _4;_2  4  1 _1; 5  0  7 _5
_1  2 _1  5
 3 _1 _6 _4
_2  4  1 _1
 5  0  7 _5

  A10 adjmat FA4
 1  3 _2  5
 2  1  4  0
_1 _6  3  7
 5 _4 _1 _3
_1  2 _1  5
 3 _1 _6 _4
_2  4  1 _1
 5  0  7 _5
 k = 2
ck = 10
i=  0
0
j  =0
1 3 _2 5
j  =1
2 1 4 0
j  =2
_1 _6 3 7
j  =3
5 _4 _1 _3
i=  1
0 32 _4 21 _27
j  =0
1 3 _2 5
j  =1
2 1 4 0
j  =2
_1 _6 3 7
j  =3
5 _4 _1 _3
i=  2
0 32 _4 21 _27 _8 _19 _43 37
j  =0
1 3 _2 5
j  =1
2 1 4 0
```

```
j  =2
_1 _6 3 7
j  =3
5 _4 _1 _3
i=  3
0 32 _4 21 _27 _8 _19 _43 37 3 4 _26 _24
j  =0
1 3 _2 5
j  =1
2 1 4 0
j  =2
_1 _6 3 7
j  =3
5 _4 _1 _3
 32  _4  21 _27
 _8 _19 _43  37
  3   4 _26 _24
_34  38 _19  33
ans =
 22  _4  21 _27
 _8 _29 _43  37
  3   4 _36 _24
_34  38 _19  23
next ck
_167 _217 _303 _372
_353
 k = 2
 k = 3
ck = _353
i=  0
0
j  =0
1 3 _2 5
j  =1
2 1 4 0
j  =2
_1 _6 3 7
j  =3
5 _4 _1 _3
i=  1
0 _167 124 _124 186
j  =0
1 3 _2 5
j  =1
2 1 4 0
j  =2
_1 _6 3 7
j  =3
5 _4 _1 _3
i=  2
0 _167 124 _124 186 176 _217 312 8
j  =0
1 3 _2 5
j  =1
2 1 4 0
j  =2
_1 _6 3 7
j  =3
5 _4 _1 _3
i=  3
0 _167 124 _124 186 176 _217 312 8 _33 _134 _303 107
```

```
j  =0
1 3 _2 5
j  =1
2 1 4 0
j  =2
_1 _6 3 7
j  =3
5 _4 _1 _3
_167  124 _124  186
 176 _217  312    8
 _33 _134 _303  107
 233 _106  _90 _372
ans =
186  124 _124 186
176  136  312   8
_33 _134   50 107
233 _106  _90 _19
next ck
1736 1736 1736 1736
1736
 k = 4
```

結局、係数 c1=2、c2=10、 c3= -353、 c4=1736 で、 固有多項式

f(λ)=λ^4 - c1･λ^3 - c2･λ^2 - c3･λ - c4　と成る。

文献 4 の古屋先生の原著(p.102 第 1 行)と比較されたい。

（勿論、今では ルベリエ･ファデーエフ法の関数 LF0 を用いて、一挙に

算出出来るものではあるが･････）

4. 古屋行列 FA4 の 固有ベクトルの解

与行列 FA4

```
 1  3 _2  5
 2  1  4  0
_1 _6  3  7
 5 _4 _1 _3
```

演算 nchar_evec FA4 （結果は膨大で、分割せねば印刷出来ない。）

固有値 4 ケ（内訳は 実数 2 ケ、複素共役 1 組）

```
 _8.04131
  4.73208
2.65462j6.21085
2.65462j_6.21085
```

固有ベクトル (縦列に読め):

固有値 1 (実数) に対し、

```
 _575.544  434.319 _186.449  829.718
  285.491 _215.438  92.4855  _411.57
 _357.53     269.8 _115.823  515.424
 726.429 _548.179  235.329 _1047.24
```

固有値２　(実数)に対し、

```
 373.677  205.321 _63.5889  184.072
 149.857  82.3406 _25.5013  73.8191
_47.0189 _25.8351  8.00125 _23.1614
 170.197  93.5166 _28.9626  83.8385
```

固有値３(複素数)　に対し、

```
 _12.5662j_4.61412   60.1801j49.2377  38.0191j_47.3171 _14.8951j_46.2947
  50.3262j41.1062 _197.952j_321.367  _249.492j156.743  _5.12462j236.012
  _36.7254j97.453  387.017j_464.915 _365.589j_298.894  _361.131j112.818
_43.3128j_2.81878   232.331j97.9021  74.8169j_182.035  _88.801j_130.303
```

固有値４(複素数)　に対し、

```
 _12.5662j4.61412 60.1801j_49.2377   38.0191j47.3171  _14.8951j46.2947
50.3262j_41.1062 _197.952j321.367 _249.492j_156.743 _5.12462j_236.012
_36.7254j_97.453  387.017j464.915  _365.589j298.894 _361.131j_112.818
_43.3128j2.81878 232.331j_97.9021   74.8169j182.035   _88.801j130.303
```

(いやはや、印刷を省略したいなー！)

［検算］

(1)　固有値１　(実数)　lm1=. _8.04131 に対して:

その　固有ベクトル　の例として（第１縦列)v11　を選ぶ。

```
  v11=._575.544  285.491 _357.53 726.429
```

計算

```
    FA4 ip v11
4628.13 _2295.72 2875.01 _5841.44
4628.13 _2295.72 2875.01 _5841.44 % lm1
```

(固有値で割る)

```
_575.544 285.491 _357.53 726.429
```

・・・・確かに v11　に戻った。

(4)　固有値４　(複素数)　lm4=. 2.65462j_6.21085 に対して:

その　固有ベクトル　の例として（第１縦列)v41　を選ぶ。

```
 _12.5662j4.61412 50.3262j_41.1062 _36.7254j_97.453 _43.3128j2.81878
```

計算

```
    FA4 ip v41
 _4.7008j90.2954 _121.708j_421.69 _702.757j_30.6045 _97.472j276.492
```

固有値で割る

```
_4.7008j90.2954 _121.708j_421.69 _702.757j_30.6045 _97.472j276.492 % lm4

_12.5662j4.61412 50.3262j_41.1063 _36.7254j_97.4529 _43.3128j2.81878
```

確かに v41　に戻った。

と云う具合で、固有ベクトルの計算は成功している。

5.　固有値　に　重複がある場合

円滑な結果にならない。　下記の如く、重複しない固有値では良いが、重複ある

時は、ゼロ・ベクトルになる等々、その結果は何かおかしい。 演算の実例を示す。

```
   1) nka3
 3 _2 1
 2 _1 1
_2  2 0

   nchar_evec nka3
┌─────┬─────┬────────┐
│1    │1    │0       │
├─────┼─────┼────────┤
│0 0 0│0 0 0│_2  2 _1│
│0 0 0│0 0 0│_2  2 _1│
│0 0 0│0 0 0│ 2 _2  1│
└─────┴─────┴────────┘
  nka3 ip _2 _2 2
0 0 0
  nka3 ip 2 2 _2
0 0 0
  nka3 ip _1 _1 1
0 0 0

  nka3 ip 0 0 0
0 0 0

   2) nkb3
0 1 1
1 0 1
1 1 0

  nchar_evec nkb3
┌─────┬─────┬─────┐
│2    │_1   │_1   │
├─────┼─────┼─────┤
│3 3 3│0 0 0│0 0 0│
│3 3 3│0 0 0│0 0 0│
│3 3 3│0 0 0│0 0 0│
└─────┴─────┴─────┘
  nkb3 ip 3 3 3
6 6 6
6 6 6 %2 （固有値で割る）
3 3 3 ・・固有ベクトルの戻った。

  nkb3 ip 0 0 0
0 0 0

  3) nkd3
1  1  1
1  1 _1
1 _1  1

  nchar_evec nkd3
┌─────┬─────┬────────┐
│2    │2    │_1      │
├─────┼─────┼────────┤
│0 0 0│0 0 0│ 3 _3 _3│
│0 0 0│0 0 0│_3  3  3│
```

```
|0 0 0|0 0 0|_3  3  3|
└──────────┴──────────┴──

  nkd3 ip 0 0 0
0 0 0

  nkd3 ip 3 _3 _3
_3 3 3
_3 3 3 % _1 (固有値で割る)
 3 _3 _3 ・・固有ベクトルの戻った。
```

今後の研究課題であろう。

6. むすび

誰かが申された。 ちょっと以前までは、逆行列の計算だけに、数日を要し、

それだけで一仕事した気分になったとか？

今は、あっと云う間に、事は済んで仕舞う。 Ｊ言語の威力も大いに預って居る

と思われる。 我々は、幸せと感謝すべきだ。 その一里塚の勉強の例を報告

した次第です。 その、きっかけを提供された志村さまに感謝したい！

しかし、大きな難問題が残る。 それは、志村さまも言及すみだが

「固有値に重複値のある場合」の処理である！

文献

1) 志村正人:「マトリックスの数学と数値計算 (1)
行列式と行列の固有値」pp. 47、JAPLA 夏の合宿 2010/8/3

2) http://detail.chiebukuro.yahoo.co.jp/qa/question_detail.php?qid=
1144612321 時に 2010/08/02、yahoo 知恵袋、数学カテゴリー。

3)http://detail.chiebukuro.yahoo.co.jp/qa/question_detail/q1346371703
yahoo 知恵袋、数学カテゴリー。

4) 古屋 茂著:「行列と行列式」 培風館、1959.5.20 増補版発行 p.100～