I君の固有値問題

桂田 祐史

1992年11月20日 (火)

古い記録なのだが、懐かしいのと、現在でも参考になるところが多いので…

1 発端

それはK先生のところの学生I君からの相談で始まった。
数学概論応用編、C[1]第6章 p440 にある固有値問題

$$\begin{aligned} \frac{\partial^4 u}{\partial x^4} &= \lambda u \quad \text{in } [0,1], \\ u(0) &= u'(0) = 0, \\ u''(1) &= u'''(1) = 0 \end{aligned}$$

の数値計算がK先生から彼に与えられた課題。本に書いてある計算法は次の通り。数列 $\{a_{ij}\}$, $\{b_{ij}\}$ を

$$a_{ij} = \frac{i(i+1)j(j+1)}{i+j-1}, \quad b_{ij} = \frac{1}{i+j+3},$$

で定めて、各 $N = 1, 2, \cdots$ に対して N 次正方行列 $A = A^{(N)}$, $B = B^{(N)}$ を

$$A = (a_{ij}), \quad B = (b_{ij})$$

で定めた時の一般化固有値問題

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

の解を小さい順に並べたものを $\lambda_1^{(N)}, \lambda_2^{(N)}, \cdots, \lambda_N^{(N)}$ とする。これから

$$\lambda_i = \lim_{N \to +\infty} \lambda_i^{(N)} \quad (i \in \mathbf{N})$$

によって得られる $\{\lambda_i; i \in \mathbf{N}\}$ が最初の固有値問題の解になる。
行列の次数を N として、N が小さい時の $\lambda_i^{(N)}$ の計算値は載っている。まず $N = 1$ の時、

$$\lambda_1^{(1)} = 20,$$

$N = 2$ の時

$$\lambda_1^{(2)} = 12.480, \quad \lambda_2^{(2)} = 1211.520$$

[1]加藤敏夫先生の変分法だ !!

となるそうだ。

一般化固有値問題というと、どういう問題であるか聞いたことはあるものの、具体的にどうやって解くのがカシコイというような知識はほとんどなかった。典型的な “教えながら勉強する” パターンになりそうだ、と思った。

2　困ったもんだ、見つからない

まずは自室にある邦書で解説を探したが、ほとんど記述なし。どこかで読んだ覚えがあったのだけど（後で分かったが一松先生の本だった）。

Numerical Recipes には少し載っていたが、慌てていたのであまり読まなかった（本当はちゃんと読むべきだった）。とりあえず I 君には Jacobi 法のプログラムの載った本[2]を渡して、勉強しておきなさい、次は出張から帰ってきたらね、と逃げておいた。

B^{-1} をかけたらどうかなぁ、とか思いつつ。いわきへと旅立つ。

3　いわきにて

質問出来る人を探す。偉い人[3]を紹介されもしたが、ちょっとおそれ多いので、適当なところで K さんに SOS。K さんによれば、逆行列をかけてでもいいでしょ、でも直接やるのでも色々ある、文献としては Cleve Moler が 1970 年代後半に SIAM Review に書いたもの、さらに Van Loan and Gene H. Golub, Matrix Computations, John Hopkins University Press (1988) でしょう、これは良く売れている本ですよ、とのこと。ははあ。聞いて良かった。さらにすぐ使えるサブルーチンが、EISPACK や IMSL の中に見つかるでしょう、とおっしゃいました（急に敬語になる）。

4　再び明治にて —作戦を立てる—

4.1　数学科資料室

早速 K さん推薦の文献を探す、、、ない！書籍の方も有隣社のリストにあがっていない。困ったものだ。書籍はそのうち何とか買うとして、文献（雑誌）の方は暇を見つけて東大数学科にコピーしに行くしかないな。

4.2　一松先生の本

本棚にある本を端から順番に邦書をチェックしていって、ようやく見覚えのある記述に遭遇した。一松信著、数値解析、朝倉書店の第 3 章 (P61) に曰く。

[2]戸川隼人, 数値計算, 岩波書店。今回は直接役には立たなかったが、いい本である。
[3]森正武、名取亮、両先生の名前があがった。

> §11. 数学では普通にはあまりうるさく区別しないが、実は行列の固有値問題にはいろいろの意味がある。定数係数連立線型常微分方程式系 $y' = Ay$ の係数行列 A は混合テンソルであって、座標系の変換の標準化による Jordan の標準系が自然な意味をもつ。これに対して、2 次形式の係数行列 A の固有値は、標準的な正値対称行列 B を定めて初めて意味がある。普通には B を単位行列 I としているのだが、その意味では、“一般化された固有値問題” とよばれる $\det(A - \lambda B) = 0$ の方がかえって本質的である。§16 の $m \times n$ 次の長方形行列 A の特異値も、m 次, n 次の標準正値対称行列 B, C を定めて初めて意味がある。これも普通には B, C をそれぞれの次元の単位行列に定めて扱う。以上は数値解析自体の問題ではないかも知れないが、実用上には重要な背景である。なおこの点を明確にご注意下さった東京大学伊理正夫教授に感謝する。

とか。やはり伊理先生は偉いんだ、と再認識したが、この問題に関しては直接得るところはなし。

4.3 Numerical Recipes

出張前に Numerical Recipes をきちんと読まなかったのは失敗だった。Cleve Moler の文献まではなかったが、Van Loan & Gene H. Golub の本 (正確には K さんが勧めてくれた本の前の版に相当するもの) は References に載っていたし、何より固有値問題に対する戦略について色々指示があって、それに従っていれば K 大明神の御神託を聞かなくても実用的な観点からは何とかなっていた。

我々に関係ある部分の要旨を大雑把に述べると、(1) 固有値問題は難しく、既成のパッケージの利用を避けることを勧めない数少ない例である (英国人て回りくどくて、やあね。つまり既成のパッケージを使いましょうということだな。)。この本では、その種のパッケージの中で何をやっているかの理解の助けになるようなプログラムを載せる。(2) 一般化固有値問題には、B の平方根を求めて普通の固有値問題に帰着させるという解法もあるが、既存のパッケージを利用するのが簡単で効率的だ。大抵のパッケージには一般化固有値問題のルーチンがある。

具体的なパッケージの名前の中に IMSL があった。この時点で私の頭の中は IMSL の利用で固まっていった。

4.4 I君来る

ここで突如 I 君登場 (作ったような話だが本当)。彼は K 先生 (K さんにあらず…同じイニシャルだけど) から渡されたコピーを持ってきた。そこには Jacobi 法で一般化固有値問題を解く方法の説明が載っていた。これは本質的には Numerical Recipes に説明のあった平方根を用いる方法であった。なるほど、解き方が載っている本があったか、でも、一からプログラムを書き起こすのは面倒だな、どうしようか、と少し迷った。なお、この解説に書いてあったのだが、A, B が対称であっても、$B^{-1}A$ は対称とは限らないので、$B^{-1}A$ の固有値問題に帰着させるのはうまくない、とのこと。なるほど、と感じていたところ、I 君から質問があった。$B^{-1}A$ が非対称になっても、この間渡された本に載っていた非対称行列の固有値を求めるサブルーチンを使えば解けるのではないか、と。おお、鋭い (人の心中が読めるのだろうか)。自

力で解く方法を見出したのは偉い、その線でやらせようか、と一瞬思ったが、やはり非対称行列用の固有値ルーチンは (使用を避ける道がある限り) 使うべきでないという正論を説明。先日の研究集会での某先生の言葉が甦る。曰く、非対称行列の固有値問題を解くのはアートである[4]、と。で、二人で IMSL を調べましょうということになった。

4.5 IMSL

マニュアルを引いたら、大した苦労もなしに GVLSP なるルーチンが見つかった。能書きに曰く、Purpose: Compute all of the eigenvalues of the generalized real symmetric eigenvalue problem A*z = w*B*z, with B symetric positive definite. これですね。サンプル・プログラムも短いので、IMSL の使い方の復習を兼ねて打ち込んで実行してみた。

コンパイルは

```
f77o sample393.f -limslib
```

で OK でした。実行して即 OK となりました。

うん、これで何とかなるでしょう。

5 戦い

5.1 初日

とりあえず I 君にやらせて、途中から真打ち (我輩のことだ) 登場。a_{ij}, b_{ij} の計算を紙の上で計算しておったのを、ちゃんと do loop で計算するようにした。

N の小さいところは快調に求まるも $N = 7$ でエラーが出た。

```
oyabun% test2
 n=
7


 *** FATAL    ERROR 2 from GVLSP.  Matrix B is not positive definite.
oyabun%
```

これはどういうことか？正定値であることは間違いないはずなのに。ピンと来るものがあって、試しに倍精度にしてみたら、大丈夫になった。さては、ということで GVLSP を call する前に B の固有値を計算させてみたら

```
oyabun% test2d
 n=
7
    1.3158636382684D-11     2.8279747505052D-09     2.8588867231650D-07
    1.8236197276094D-05     8.3397167029818D-04     2.9314481226287D-02
```

[4]本当は英語だった。

```
   0.71712364005519
    12.362363368353     485.51884143459     3808.7559124218     14669.662848705
    49899.723376200     126886.20894806     2765041.2146235
```

となった。なるほど、B は対称で、固有値も計算してみたら確かにみな正なのだけれど、絶対値がとても小さく、それで正定値でないと判断されてしまったわけだ。倍精度にした場合は限界まで少し余裕が出たというわけだね。とはいえ、かなり条件の厳しい問題らしく、倍精度にしても $N = 11$ 程度が限界で、$N = 12$ ではやはり "Matrix B is not positive definite." と言われてしまった。

それでも、絶対値の小さい方の固有値の収束ぶりはまあまあのものだ。試しに $N = 10$ と $N = 11$ の場合の結果を並べて見よう。

```
oyabun% test2d
 n=
10
    12.362363368326     485.51881851566     3806.5462784675     14617.793474394
    39954.444527442     91907.607463908     184992.73985194     598922.97555703
    1177146.5931475     30675173.939123
oyabun% test2d
 n=
11
    12.362363368326     485.51881851341     3806.5462775007     14617.275831296
    39954.439347729     89277.089283761     184984.26168742     352114.83160918
    1177043.8311788     2683602.5548179     72724488.719649
oyabun%
```

ちなみにプログラムの中身は

```
        integer i,j,n, maxn
        parameter (maxn = 100)
        real*8 a(maxn,maxn), b(maxn,maxn), eval(maxn)
        external dgvlsp
c
        write(*,*) ' n='
        read(*,*) n
c
        do 100 i=1,n
           do 110 j=1,n
              a(i,j)=i*(i+1)*j*(j+1.0d0)/(i+j-1.0d0)
              b(i,j)=1.0d0/(i+j+3.0d0)
 110       continue
 100    continue
c
c        call dwrrrn('b=', n, n, b, maxn, 0)
```

```
c
c         call devlsf(n, b, maxn, eval)
c         write(*,*) (eval(i),i=1,n)
c
         call dgvlsp (n, a, maxn, b, maxn, eval)
c
         write(*,*) (eval(i),i=1,n)
         end
```

という感じである。

ここでI君の指摘で加藤先生の本の記述に帰る (P.440 下〜 p.441)。何でも λ_i は方程式

$$\cos\theta\cosh\theta + 1 = 0$$

の解の 4 乗であって、例えば $\lambda_1 = 12.36236$, $\lambda_2 = 485.5217$ であって、別の方法[5]で得られた $\lambda_2^{(2)} = 515.86$ という値は 6% 程度の誤差がある、と。

私は $\lambda_2 = 485.5217$ という値のシッポはちょっと信じられない、と思った。$N = 11$ までの計算を見ると、収束している雰囲気があって、$\lambda_2^{(2)} = 485.5188185$ 程度はかなりの精度であると思われた。何といっても古い本だし、非線形方程式の解を疑ってみよう、と考えた。そこで Mathematica の登場。最初は Solve を用いたが、駄目だった。無限個の解があるから無理か。でもそれならば範囲を指定すれば解けるのでは？と考えて探していたら FindRoot というのを見つけた。

```
oyabun% math
Mathematica 2.0 for SPARC
Copyright 1988-91 Wolfram Research, Inc.
 -- X11 windows graphics initialized --

In[1]:= FindRoot[Cos[x]Cosh[x]+1==0,{x,4}]

Out[1]= {x -> 4.69409}

In[2]:= FindRoot[Cos[x]Cosh[x]+1==0,{x,4.69},WorkingPrecision->100]

Out[2]= {x -> 4.69409113297417457643639177801981204938989673754576682897280321\

>       7784907793680112561763976916480788340 1}

In[3]:= 4.69409113297417457643639177804

Out[3]= 485.51881851337103781169138 38
```

[5]改良された基底関数を取った場合。

```
In[4]:= Quit
oyabun%
```

やはり、と思った (こんな細かい計算でツッパルのは大人げないな)。

5.2 二日目

これを書いているのは実は三日目なのだが、大変な問題に関わってしまったと思い始めている。

初日のので大体話は済んだと思っていたのだが、I 君は別の計算を始めていた。そもそもこの計算は Galerkin 法に基づいているのだが、これまで使ってきた基底関数は境界条件を満足していないので、あまりうまくないのだという。境界条件を満たす基底関数を使うと収束が良くなる、と本に書いてあった。ところがそこには基底関数が 2 つしか書かれていなかった:

$$w_1(x) = x^2(6 - 4x + x^2), \quad w_2(x) = x^3(10 - 10x + 3x^2)$$

そこで基底関数の一般形を求めることが問題。はっきりとは書いていないが、加藤先生の頭にあったのは

$$w_n(x) = x^n \times (x \text{ の } 2 \text{ 次式で } 1 \text{ における } 2 \text{ 階、} 3 \text{ 階の微分係数が } 0 \text{ になるもの})$$

ということらしい。こういうものを具体的に求めるのは高校数学の問題だが、面倒だね。I 君に紙の上で計算させたが、時間がかかりそうだったので、Mathematica を持ち出して計算を始めてみた。実際にどういう計算をしたのかは、ログに残っているが、エッセンスをプログラムにまとめると、

```
u=x^(n+1)(x^2+a x+b)
DDu=D[u,{x,2}]
u2=Simplify[DDu /. x->1]
DDDu=D[u,{x,3}]
u3=Simplify[DDDu /. x->1]
sol=Simplify[Solve[{u2==0,u3==0},{a,b}]]
```

ということである。その結果は

```
oyabun% math
Mathematica 2.0 for SPARC
Copyright 1988-91 Wolfram Research, Inc.
 -- X11 windows graphics initialized --

In[1]:= << ishibashi1.m

                                          2
               -2 (3 + n)       6 + 5 n + n
Out[1]= {{a -> ----------, b -> ------------}}
```

```
                1 + n                    2
                                    n + n

In[2]:=
```

となる。そこで

$$w_n(x) = x^{n+1}[n(n+1)x^2 - 2n(n+3)x + (n+2)(n+3)]$$

という基底関数の一般形が求まる。これは $n = 1, 2$ の時に加藤先生のあげたものと一致する ($n = 1$ の時は加藤先生の関数を 2 倍したもの)。

僕がこの計算を終える頃にはI君も大体計算を終えていて、同じ結果を得た (まあ間違っていない限り同じになるのは当たり前だけど)。

これで話が済んだわけではない。この基底関数を用いて行列 A, B を計算しないといけない。実は

$$a_{ij} = (w_i'', w_j''), \quad b_{ij} = (w_i, w_j)$$

ということだそうだ。ここで $(,)$ は区間 $(0, 1)$ 上の内積を表す。

あまり手でやりたくない計算だ、ということで Mathematica との格闘が始まった。これも一部始終がログに残っているが、エッセンスをまとめると、

```
%
u=x^(i+1)((i+1)i x^2-2i(i+3)x+(i+2)(i+3))
v=x^(j+1)((j+1)j x^2-2j(j+3)x+(j+2)(j+3))
intuv=Integrate[u v, {x,0,1}]
uv=Simplify[intuv]
Table[uv, {i,5}, {j,5}]
FortranForm[uv]
Print[Table[uv, {i,15}, {j,15}]]
Du=Simplify[D[u,{x,2}]]
Print["Du=",Du]
Dv=Simplify[D[v,{x,2}]]
Print["Dv=",Dv]
DuDv = Simplify[Du Dv]
Print["DuDv=",DuDv]
tmp=Integrate[DuDv /. i+j-2->m, {x,0,1}]
tmp=Simplify[tmp]
u2v2 = tmp/. m->i+j-2
Print["u2v2=", u2v2]
FortranForm[u2v2]
Print[Table[u2v2, {i,15}, {j,15}]]
```

ということになる。

```
oyabun% math
```

```
Mathematica 2.0 for SPARC
Copyright 1988-91 Wolfram Research, Inc.
 -- X11 windows graphics initialized --

In[1]:= << foo.m
   1 + i                                                 2
u=x      ((2 + i) (3 + i) - 2 i (3 + i) x + i (1 + i) x )
   1 + j                                                 2
v=x      ((2 + j) (3 + j) - 2 j (3 + j) x + j (1 + j) x )
                              2        3      4
Bij=(8 (3780 + 3564 i + 1209 i  + 174 i  + 9 i  + 3564 j + 2505 i j +

             2         3           2          2       2  2        3
>        565 i  j + 40 i  j + 1209 j  + 565 i j  + 65 i  j  + 174 j  +

              3      4
>        40 i j  + 9 j )) /

>    ((3 + i + j) (4 + i + j) (5 + i + j) (6 + i + j) (7 + i + j))
TeXForm={{8\,\left( 3780 + 3564\,i + 1209\,{i^2} + 174\,{i^3} + 9\,{i^4} +

>         3564\,j + 2505\,i\,j + 565\,{i^2}\,j + 40\,{i^3}\,j + 1209\,{j^2} +

>         565\,i\,{j^2} + 65\,{i^2}\,{j^2} + 174\,{j^3} + 40\,i\,{j^3} +

>         9\,{j^4} \right) }\over

>     {\left( 3 + i + j \right) \,\left( 4 + i + j \right) \,

>       \left( 5 + i + j \right) \,\left( 6 + i + j \right) \,

>       \left( 7 + i + j \right) }}
FortranForm=8*(3780 + 3564*i + 1209*i**2 + 174*i**3 + 9*i**4 + 3564*j +

>       2505*i*j + 565*i**2*j + 40*i**3*j + 1209*j**2 + 565*i*j**2 +

>       65*i**2*j**2 + 174*j**3 + 40*i*j**3 + 9*j**4)/

>    ((3 + i + j)*(4 + i + j)*(5 + i + j)*(6 + i + j)*(7 + i + j))
                    2    3          2  -1 + i
Du=i (6 + 11 i + 6 i  + i ) (-1 + x)  x
                    2    3          2  -1 + j
```

```
Dv=j (6 + 11 j + 6 j  + j ) (-1 + x)  x
                     2    3                   2    3          4  -2 + i + j
DuDv=i (6 + 11 i + 6 i  + i ) j (6 + 11 j + 6 j  + j ) (-1 + x)  x
                        2    3                  2    3
Aij=(24 i (6 + 11 i + 6 i  + i ) j (6 + 11 j + 6 j  + j )) /

                                                 2                   3
>    (120 + 274 (-2 + i + j) + 225 (-2 + i + j)  + 85 (-2 + i + j)  +

                       4               5
>      15 (-2 + i + j)  + (-2 + i + j) )
TeXForm={{24\,i\,\left( 6 + 11\,i + 6\,{i^2} + {i^3} \right) \,j\,

>       \left( 6 + 11\,j + 6\,{j^2} + {j^3} \right) }\over

>     {120 + 274\,\left( -2 + i + j \right)  +

>       225\,{{\left( -2 + i + j \right) }^2} +

>       85\,{{\left( -2 + i + j \right) }^3} +

>       15\,{{\left( -2 + i + j \right) }^4} +

>       {{\left( -2 + i + j \right) }^5}}}
FortranForm=24*i*(6 + 11*i + 6*i**2 + i**3)*j*(6 + 11*j + 6*j**2 + j**3)/

>    (120 + 274*(-2 + i + j) + 225*(-2 + i + j)**2 + 85*(-2 + i + j)**3 +

>      15*(-2 + i + j)**4 + (-2 + i + j)**5)
  416  2608  8636  1123  13384    2608  5216  227  904  17540
{{---, ----, ----, ----, -----}, {----, ----, ---, ---, -----},
  45   315   1155  165   2145     315   693   33   143  3003

     8636  227  908  5890  5480    1123  904  5890  2356  401
>   {----, ---, ---, ----, ----}, {----, ---, ----, ----, ---},
     1155  33   143  1001  1001    165   143  1001  429   78

     13384  17540  5480  401  3208
>   {-----, -----, ----, ---, ----}}
     2145   3003   1001  78   663
  576      576                960  1080
{{---, 96, ---, 72, 64}, {96, ---, ----, 160, 160},
```

```
   5        7                   7    7

     576  1080  1440        2880                    3360  3920
>   {---, ----, ----, 240, ----}, {72, 160, 240, ----, ----},
      7    7     7           11                      11    11

               2880  3920  62720
>   {64, 160, ----, ----, -----}}
               11    11    143
```

```
In[2]:= uv
```

後はI君に任せよう。Mathematica 万歳。

5.3 二日目の晩

以前からの懸案だった EISPACK のインストールにやっと成功して、IMSL の代わりに EISPACK でも計算できた。結果は IMSL 内のルーチンを利用した場合と少し食い違う。少なくとも単精度計算の結果は IMSL のそれの方が良いみたい (倍精度計算の結果と照合しての判断)。ふうむ、さすが商品、か。中で反復法でもやっているのか。それとも、何か本質的な改良してあるのか。もっとも、どの N まで計算できるかという限界は、両者に差がなかった。

ところで、僕は次の問について考える必要があると気がついた。

すべての固有値が必要なのか、それとも小さい方の数個があれば十分なのか？

これは加藤先生の本を真面目に読む必要があるかなぁ？、ということになった。

5.4 三日目 – 差分法で挑戦 –

昨日ゼミ卒研があって、久しぶりに菊地＆山本の本を読んでいて、常微分方程式の固有値問題の章があるのを思い出した。そうだ、これをやってみようということで、実行。

まずちょっと走らせてみて、あまり精度が低いのでコーディング・ミスを疑ったが、普通のルーチンと、帯行列用のルーチンで同じ答が出る。単純なコーディング・ミスではなさそう。それではアルゴリズムのミスか？。ところが、次数を上げていくと段々精度が上がって来てそれらしい結果を出すようになる。正しいのかな？この時点で、次数を大きくするためにも帯行列用のルーチンでないと駄目だと分かった。さらに固有値を全部計算するのはしんどいので、小さい方から数個求めるようにしないと話にならないことも分かった。というわけで、次のようなプログラムを走らせることになった。

```
      program fdm
      integer i,n,m,maxn,N1,N2,NSTEP,MAXW
      parameter (maxn = 10000, ncoda=2, lda=ncoda+1)
      parameter (MAXW = 2*maxn*(ncoda+4)+maxn)
```

```
      logical small
      parameter (small = .true.)
      real*8 a(lda,maxn), eval(maxn), c, h, h4inv
      external devasb
      COMMON /WORKSP/  RWKSP
      REAL RWKSP(MAXW)
*
      CALL IWKIN(MAXW)
c
      write(*,*) ' NSTART,NEND,NSTEP='
      read(*,*) N1,N2,NSTEP
      if (N1 .lt. 6) then
        N1 = 6
      endif
c
      do n = N1,N2,NSTEP
        m = n - 3
        h = 1.0d0 / n
        h4inv = 1.0d0 / h**4
c
        a(1,1)=0.0d0
        a(1,2)=0.0d0
        c = 1.0d0 * h4inv
        do i=3,m
          a(1,i) = c
        end do
c
        a(2,1) = 0.0d0
        c = -4.0d0 * h4inv
        do i=2,m-1
          a(2,i)= c
        end do
        a(2,m)=-2.0 * h4inv
c
        c = 6.0d0 * h4inv
        do i=1,m-2
          a(3,i)= c
        end do
        a(3,m-1)=5.0d0 * h4inv
        a(3,m)=1.0d0 * h4inv
c
        call devasb(m, 5, a, lda, ncoda, small, eval)
```

```
c
        write(*,*) ' N=', N
        write(*,*) (eval(i),i=1,5)
      end do
      end
```

さて、結果はどうなるかというと、、、

```
oyabun%  test-FDM2
 NSTART,NEND,NSTEP=
1000 10000 1000
 N=  1000
    12.462191808892    489.42847340945    3837.1448241929    14734.664365594
    40264.212562460
 N=  2000
    12.395044092660    487.34666340961    3821.7483886199    14675.816251591
    40103.824563559
 N=  3000
    13.134802409501    486.94196351073    3816.7670363691    14656.253512514
    40050.098696306
 N=  4000
    11.946737822504    486.33007299197    3813.4893156216    14645.588853085
    40023.221212918
 N=  5000
    19.102980867991    494.31233271986    3818.8659709859    14646.849156585
    40013.678099232
 N=  6000
    28.244970754340    503.45967693861    3832.3625652045    14659.027784007
    40018.703056702
 N=  7000
   -3.6834530007044    492.44054690281    3820.1818452542    14648.801176336
    39988.834453875
 N=  8000
   -63.982736504047    408.39409769491    3740.3961247383    14541.477295313
    39945.408837434
 N=  9000
    84.591125935711    508.35975169377    3837.5120895015    14655.358069608
    39998.978912310
 N=  10000
   -52.363946332173    418.17647792973    3803.9786566901    14652.305272351
    40010.437030957
oyabun%
```

うーん。Galerkin 法に惨敗している。$N = 2000$ 程度で頭打ちみたいで、その際の結果が悪い。

5.5 再び Mathematica

この時点の状況を見ると Galerkin 法の優秀性が際だっている。差分法で大きな計算をしても Galerkin 法の小さな計算にかなわないのならば、Galerkin 法で、$N = 10, 20$ 程度の計算をして満足するべきなのではないか？(最初のうちは暗に、Galerkin 法で $N = 100, 200$ というような計算をする気でいたわけだ。だから Jacobi 法の採用や、非対称行列の固有値問題への帰着を蹴って来たわけだね。)

そういうわけで、全部 Mathematica で計算する方法を考えることになった。Mathematica に一般化固有値問題を解く組み込み手続きはないようだが、問題が小さければ非対称の問題になっても構わないのでないか？そこで一度は捨てた方法、すなわち $B^{-1}A$ の固有値の計算をしてみることにした。プログラムは

```
% ishibashi3.m
u=x^(i+1)((i+1)i x^2-2i(i+3)x+(i+2)(i+3))
v=x^(j+1)((j+1)j x^2-2j(j+3)x+(j+2)(j+3))
Print["u=",u]
Print["v=",v]
Bij=Simplify[Integrate[u v, {x,0,1}]]
Print["Bij=",Bij]
DDu=Simplify[D[u,{x,2}]]
DDv=Simplify[D[v,{x,2}]]
DDuDDv = Simplify[DDu DDv]
Print["DDuDDv=",DDuDDv]
Aij=Simplify[Integrate[DDuDDv /. i+j-2->m, {x,0,1}] /. m->i+j-2]
Print["Aij=", Aij]
B5=Table[Bij, {i,5}, {j,5}]
A5=Table[Aij, {i,5}, {j,5}]
X5=N[Inverse[B5] . A5, 40]
Print["(N=5) Eigenvalues of B^(-1) * A =", Sort[Eigenvalues[X5]]]
B20=Table[Bij, {i,20}, {j,20}]
A20=Table[Aij, {i,20}, {j,20}]
X20=N[Inverse[B20] . A20, 40]
Print["(N=20) Eigenvalues of B^(-1) * A =", Sort[Eigenvalues[X20]]]
Remove[u,v,DDu,DDv,DDuDDv,A5,B5,X5,A20,B20,X20]
```

というものである。固有値は

```
(N=5) Eigenvalues of inverse B^(-1) * A =

>   {12.362363368409529402317875850671541600 33,

>    485.518974408300611662555897472057989328 4,

>    3811.768990034377066087752823718141246277,
```

```
>    14784.491700638099163189711399552268624,

>    57102.465067439349017222022123348860402 6}

(N=20) Eigenvalues of B^(-1) * A =

>   {12.362363368326190218719261661188690136 03,

>    485.518818513371037811691383834744388645,

>    3806.546266391451058088482228842380210229,

>    14617.273305118780663739784339335879078 48,

>    39943.831778509466747602338184590856826 5,

>    89135.405071423787031412816497573328912,

>    173881.315656156365530902969367235780677,

>    308208.452457543497629731432566557019254 3,

>    508481.549656710723291718338337788982288 7,

>    793407.140343260324224285231205468035843,

                                                    6
>    1.184039968145607930407722269718423310352 10 ,

                                                    6
>    1.706190976968393934292624562047574379220 10 ,

                                                    6
>    2.386493083493206213818601932268233908251 10 ,

                                                    6
>    3.392589029623894784136767942540982255932 10 ,

                                                    6
>    4.637299649039793360085698921066680760250 10 ,
```

```
                                                        6
>     7.85416773748417801457297775449574357188 10 ,

                                                         7
>     1.10415900916850507875212812433083284958 3 10 ,

                                                         7
>     2.76143159806433249764830585800135181750 8 10 ,

                                                        7
>     4.34651133206433406410291335571975185642 10 ,

                                                         8
>     2.08083134525913379831312160493166668415 8 10 }

In[2]:= Quit
```

となります。まあ、ここらへんが満足のしどころかな。

6 戦い済んで

やはり不勉強を痛感する。とりあえず、いくつか身につけることの出来たものがある。

- Mathematica の使い方。使えるという自信。
- IMSL の使用経験。
- 使える状態の EISPACK。EISPACK の使用経験。
- 常微分方程式の固有値問題の数値解法についての知識。
- 素朴な Galerkin 法の経験。
- 固有値問題への知見。必要な文献等。

IMSL は使い続けることにしよう (バージョンアップを続けるようにお願いしよう)。そして EISPACK や NUMPAC 等も整備しておくことにする (NUMPAC 入手については動き出さないといけないな)。加藤先生の変分法は一度きちんと読む (必要—価値) があるだろう。数値計

算に関する基本的な文献は早めに揃えておくことにしよう。また、学生用の参考書も複数準備したい。

最後に自室でなく、計算機室備えつけの書籍には、一般化固有値問題の解説もちゃんと載っていることが明らかになった。(ああ、恥ずかし。不明を恥じる。) 特に渡辺&名取&小国の本はプログラム付きだった。これもちゃんとインストールしておく必要があるな。

A 差分方程式

Δ を中心差分演算子とする。$v_j := \Delta^2 u_j$ とする。すなわち

$$v_j = u_{j-1} - 2u_j + u_{j+1}.$$

$$\begin{aligned}\Delta^4 u_j = \Delta^2 v_j &= v_{j-1} - 2v_j + v_{j+1} \\ &= u_{j-2} - 2u_{j-1} + u_j - 2(u_{j-1} - 2u_j + u_{j+1}) + (u_j - 2u_{j+1} + u_{j+2}) \\ &= u_{j-2} - 4u_{j-1} + 6u_j - 4u_{j+1} + u_{j+2} \quad (2 \le j \le N-2).\end{aligned}$$

ゆえに

$$u_{j-2} - 4u_{j-1} + 6u_j - 4u_{j+1} + u_{j+2} = \lambda u_j \quad (2 \le j \le N-2). \tag{1}$$

一方で $u(0) = u'(0) = 0$ より $u_0 = 0,\ u_1 - u_0 = 0$ として、

$$u_0 = u_1 = 0. \tag{2}$$

また $u''(0) = u'''(0) = 0$ より

$$u_{N-2} - 2u_{N-1} + u_N = 0, \quad (u_{N-3} - 2u_{N-2} + u_{N-1}) - (u_{N-2} - 2u_{N-1} + u_N) = 0.$$

これから

$$u_{N-2} - 2u_{N-1} + u_N = 0, \quad u_{N-3} - 2u_{N-2} + u_{N-1} = 0.$$

整理して

$$u_{N-1} = 2u_{N-2} - u_{N-3}, \quad u_N = 2u_{N-1} - u_{N-2} = 2(2u_{N-2} - u_{N-3}) - u_{N-2} = 3u_{N-2} - 2u_{N-3}. \tag{3}$$

よって最後から 2 行目は

$$\begin{aligned}u_{N-5} - 4u_{N-4} + 6u_{N-3} - 4u_{N-2} + u_{N-1} &= u_{N-5} - 4u_{N-4} + 6u_{N-3} - 4u_{N-2} + (2u_{N-2} - u_{N-3}) \\ &= u_{N-5} - 4u_{N-4} + 5u_{N-3} - 2u_{N-2}.\end{aligned}$$

最後の行は

$$\begin{aligned}u_{N-4} - 4u_{N-3} + 6u_{N-2} - 4u_{N-1} + u_N &= u_{N-4} - 4u_{N-3} + 6u_{N-2} - 4(2u_{N-2} - u_{N-3}) + (3u_{N-2} - 2u_{N-3}) \\ &= u_{N-4} - 2u_{N-3} + u_{N-2}.\end{aligned}$$

よって

$$\frac{1}{h^4}\begin{pmatrix} 6u_2 & -4u_3 & +u_4 & & & & & \\ -4u_2 & +6u_3 & -4u_4 & +u_5 & & & & \\ u_2 & -4u_3 & +6u_4 & -4u_5 & +u_6 & & & \\ & u_3 & -4u_4 & +6u_5 & -4u_6 & +u_7 & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & u_{N-6} & -4u_{N-5} & +6u_{N-4} & -4u_{N-3} & +u_{N-2} \\ & & & & u_{N-5} & -4u_{N-4} & +5u_{N-3} & -2u_{N-2} \\ & & & & & u_{N-4} & -2u_{N-3} & +u_{N-2} \end{pmatrix} = \lambda \begin{pmatrix} u_2 \\ u_3 \\ u_4 \\ u_5 \\ \vdots \\ u_{N-4} \\ u_{N-3} \\ u_{N-2} \end{pmatrix}.$$

これは $N-3$ 次実対称行列の固有値問題である。