

MATLAB 手習い

桂田祐史

2003年5月8日

1 起源

MATLAB (MATrix LABoratory) は、著名な線形演算ライブラリ LINPACK, EISPACK の開発でも中心的な役割を果たした Cleve Moler が、1980 年頃に製作したものが発展した数値実験環境 (「実験室」) である (当時の開発言語は FORTRAN)。彼は 1985 年に C 言語で MATLAB を書き直し、MathWorks 社を設立して販売を開始した (現在 Moler は会長兼技師長であるとか)。

2 覚え書き

(以下は実際には MATLAB で試していない。ほとんどは Octave で実行した。)

- 行単位の編集機能、ヒストリー、タブによる補間
- 名前の大文字、小文字は区別する。
- `clear` 変数名 あるいは `clear` で変数消去。
- `who` とすると使用されている変数名を表示する。
`whos` とすると使用されている変数の情報を表示する。
- 変数名 とすると、“変数名= 変数の値” と表示される。[変数名] とすると、“ans = 変数の値” と表示される。(ans=変数名 と同値ということ?)
- 関数呼び出し or 変数 = 関数呼び出し
前者の場合 ans という変数に結果が入る。
- 行末に ; をつけると結果は表示されない。
- Octave, Scilab などの互換システムがある。
- 横ベクトル (1,2,3) は [1 2 3] で、縦ベクトル $\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$ は [1; 2; 3] あるいは [1 2 3]’ で表す。

- 行列 $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ は

[1 2 3; 4 5 6; 7 8 9]

と表わす。

- ones(m, n) で m 行 n 列の、すべての成分が 1 である行列
- zeros(m, n) で m 行 n 列の、すべての成分が 0 である行列
- eye(n, n) で n 次の単位行列¹
- hilb(n), invhilb(n) はそれぞれ n 次の Hilbert 行列, n 次の Hilbert 行列の逆行列を表わす。
- hadamard(n) は n 次の Hadamard 行列 (Octave にはない)。
- rand(m, n) で一様乱数行列、randn(m, n) で正規乱数行列を表わす。
- sylvester_matrix(n) でシルベスター行列を表わす。
- toeplitz(v, n)
- 2:5 は [2 3 4 5], 0:0.2:1 は [0.0 0.2 0.4 0.6 0.8 1.0]
- ベクトルの成分は、ベクトルの変数名 (インデックス) で表わせる (Fortran 風ですね)。
- +, -, *, / は普通の意味の四則を表わす (数、ベクトル、式に対して使える。)
- A' は行列またはベクトル A のエルミート共役を表す。単なる転置は A.' で表す。
- 縦ベクトル x, y の内積は o

$x' * y$

となる。

- 九九の表を作る (1:9) .* (1:9)
- det(正方行列) は行列式
- trace() はトレース
- rank(a, tol) はランク
- kron(行列, 行列) は Kronecker 積

¹Octave では eye(n) で良いが、Scilab では eye(n, n) とする必要がある。

- 特性多項式は `poly(行列)` で計算できる (結果は多項式の係数の作るベクトル)。
- 多項式の積は `conv(係数ベクトル, 係数ベクトル)` で計算できる。結果も係数の作るベクトルである。
- `inv(行列)` は逆行列 (でも減多なことでは使わないこと！)
- 行列については B / A は BA^{-1} , $A \setminus B$ は $A^{-1}B$ を表す。連立 1 次方程式 $Ax = b$ の解 $x = A^{-1}b$ は $A \setminus b$ で計算できる。
- LU 分解するには `lu()` を用いる。ピボットつき LU 分解は

```
[L,U,P]=lu(A);
```

でできる ($PA = LU$ となる² P, L, U が求められる)。LU 分解に引き続き $Ax = b$ を解くには

```
[L,U,P]=lu(A);
y=L\u(P*b);
x=U\u
```

- Cholesky 分解をするには `chol()`

```
u=chol(A)
l=u'
```

とすると $A = l u$ が成り立つ。

- `hess()`, `shur()`, `svd()`
- `eig(正方行列)` は固有値, `[v,lambda]=eig(正方行列)` は固有ベクトル、固有値

```
A=[1 2 3;4 5 6;8 5 2];
[v,lambda]=eig(A);
inv(v) * A * v
```

- 行列変数 (行始まり:行終り, 列始まり:列終り) で部分行列
- 行列の名前 (:,j) とすると第 j 列ベクトル、行列の名前 (i,:) とすると第 i 行ベクトルを表わす。
- 行列を 1 次元ベクトル化するには `v=A(:)` のようにする。
- 1 次元ベクトルを 2 次元化するには

² $P^2 = I$ なので $A = PLU$ と言っても同じこと。

```
A=zeros(m,n);
A(:)=v;
```

のようにする。Octave には reshape() という関数があり、

```
A=reshape(v,m,n);
```

のように使える。

- diag(ベクトル) は対角行列 (対角成分の作るベクトルを指定), 例えば diag([1 2 3]) は $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$ を表す。diag([1 2], 1) は $\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{pmatrix}$ を表わす。
- triu(行列) は上三角部分
- tril(行列) は下三角部分
- [A B] とか [A, B] は並べて作る (それぞれ横、縦)。
- cond() は条件数, rcond() は LINPACK の条件数 (正確には条件数の粗い評価). (Octave には rcond() はない。)
- norm(v,p), norm(v,inf)
- norm(x) は norm(x,2) に等しい。
- norm(x,'fro') は Frobenius ノルム。
- lookfor は Octave にはない。
- roots(係数を表すベクトル) で多項式の根
- sum(ベクトル) で成分の和、prod(ベクトル) で成分の積、mean(ベクトル) で成分の平均値、median(ベクトル) で中央値、max(ベクトル) で成分の最大値、min(ベクトル) で成分の最小値、var(ベクトル) で分散 (ただし要素数 ÷ (n-1) で計算するやつ)、cov(ベクトル) で分散 (ただし要素数 ÷ (n-1) で計算するやつ)、std(ベクトル) で標準偏差 (分散の正の平方根)、length(ベクトル) でベクトルとしての次元 (列としての長さ)、
- いわゆる特殊関数も揃っている。
- plot(ベクトル), closeplot
- 制御構文は

```
while 条件式
  文 1
  文 2
  ...
  文 n
end
```

```
if 条件式
  文 1
  ..
  文 n
else
  文'1
  ...
  文'n
end
```

```
if 条件式 1
  文
elseif 条件式 2
  文
end
```

```
for i=1:n
  文 1
  ...
  文 n
end
```

- for 変数=初期値:終了値 または for 変数=初期値:増分:終了値
- break はループを抜け出す
- 「かつ」は &, 「または」は |, 「等しくない」は ~= または != で表わす。つまり否定は ~ または ! という事だろうな。
- キーボード入力

変数名 = input('何か入力して下さい')

Octave for FreeBSD は日本語が入力できない。

- `pause(秒数)` で時間待ちをする。 `pause` でキーボードに触れるまで待つ。
- 演算子の前に `.` をつけると成分ごとという意味? `c=[1 2 3]` に対して `c*c` は型がおかしいから計算できない。 `c .* c` とすると `[1 4 9]`
- 変数名=値
ベクトルの場合は `[変数名 1, 変数名 2]=値` や `[変数名 1 変数名 2]=値` のようなことができる。
- 1 変数関数のグラフを描くには、ベクトルを 2 本用意する。

```
% y=x^2 のグラフ
x=0:0.1:10;
y=x.^2;
plot(x,y)
```

```
% y=sin(x) のグラフ
x=0:0.1:10;
y=sin(x);
plot(x,y)
```

- Octave では `gnuplot` を使ってグラフ描画しているので、
`gset term postscript; gset output "foo.ps"; replot`
のようなことができる。
- `contour(z, 等高線のレベル数, x, y)`

```
n=10; x=(-1:1/n:1)'; y=(-1:1/n:1); z=x*y; contour(z,10,x,y)
```

- `mesh(x,y,z)` は 3 次元グラフ
- `i, I, j, J` は虚数単位、`pi` は円周率、`e` は自然対数の底 (Napier の数)、`inf` は無限大、`eps` は計算機イプシロン
- `3 + 5i` は `3+5i` で表せる。5 と `i` の間に空白を置いてはいけない。
- `real()`, `imag()`, `conj()`, `arg()`, `angle()` などが使える。
- `eval(式)` は式の評価

```
t='1/(i+j-1)';
for i=1:n
for j=1:n
a(i,j)=eval(t);
end
end
```

- 画面出力の精度は

```
format long
format short
format long e   長い桁数、指数形式
format short e  短い桁数、指数形式
format free     自由形式
format none     自由形式
```

- `quad('関数名', 初期値, 終端値)` `quad8()` は Octave にはない。
- `save -ascii ファイル名 変数名1 変数名2`, `save -binary ファイル名 変数名1 変数名2`, `save -mat-binary ファイル名 変数名1 変数名2`

3 Octave

- `octave --traditional` あるいは `octave --braindead` で MATLAB に近くなる。
- `octave>> help -i 項目`
- Octave には `sparse` 関係の命令がない (困る)。
- Octave のグラフ表示機能は `gnuplot` を用いて実現されているが、MATLAB, Scilab と比べて劣っている。

4 Scilab

疎行列にする命令 `sparse()` がある。Asp が疎行列として、`Aspx=b` を解くには、

```
[h, rk]=lufact(Asp); x=lusolve(h,b); ludel(h);
```

とする。

5 ある晩の遊戯 — 正方形領域におけるラプラシアン固有値問題

まずは Octave で解いてみよう。

```
eigen_square.m
## 正方形領域のラプラシアンの固有値
function retval = eigen_square(n)
    h = 1/n;
    B=diag(ones(n-2,1),1)+diag(ones(n-2,1),-1);
    I=eye(n-1,n-1);
    A = - n * n * (- 4 * kron(I,I) + kron(B,I) + kron(I,B));
    retval = eig(A);
endfunction
```

eigen_square2.m

```
## 正方形領域のラプラシアン固有値、固有関数
function [v,lambda] = eigen_square2(n)
    h = 1/n;
    B=diag(ones(n-2,1),1)+diag(ones(n-2,1),-1);
    I=eye(n-1,n-1);
    A = - n * n * (- 4 * kron(I,I) + kron(B,I) + kron(I,B));
    [v,lambda] = eig(A);
endfunction
```

octave:1> eigen_square(10)

Scilab では、retval=eig(A) の代わりに retval=spec(A)、[v,lambda]=eig(A) の代わりに [v,lambda]=bdiag(A) くらいで行くか? sparse() という命令があるそうだ...期待に胸が膨らむ。

eigen_square3.m

```
# 正方形領域のラプラシアン固有値 (Scilab version) バグあり
function retval = eigen_square3(n)
    h = 1/n;
    B=sparse(diag(ones(n-2,1),1)+diag(ones(n-2,1),-1));
    I=speye(n-1,n-1);
    A = - n * n * (- 4 * kron(I,I) + kron(B,I) + kron(I,B));
    retval = spec(A);
endfunction
```

おや、kron(I,I) でエラーになる。

```
-->A = - n * n * (- 4 * kron(I,I) + kron(B,I) + kron(I,B));
      p                !--error 246
impossible to overload this function for given argument type(s)
      undefined function %sp_kronm
```

つまり、残念ながら疎行列用の kron() はないわけだ。また疎行列用の spec() もないらしい。結局、Octave と本質的には変わらずに

改訂版 eigen_square3.m

```
# 正方形領域のラプラシアン固有値 (Scilab version)
function retval = eigen_square3(n)
    B=diag(ones(n-2,1),1)+diag(ones(n-2,1),-1);
    I=eye(n-1,n-1);
    A = n * n * (4 * kron(I,I) - kron(B,I) - kron(I,B));
    retval = spec(A);
endfunction
```

とするしかない。timer();a=eigen_square3(10);timer() として計算時間を計測した。

n	計算時間 (秒)
10	0.05
20	2.17
25	8.45
30	28.69
40	191.82

確かに計算時間は n^6 に比例している。ちなみに事前に何もしないと $n = 30$ で異常終了する。スタックがあふれたらしい。stacksize() で調べると、double を一単位として 1M となっていた。つまり 1000 次程度の正方行列が覚えられるリミットである。そこで stacksize(50*1000*1000) としたところ、大きな n に対しても計算できるようになった。これなら 50M 要素 = 400MB だから、主記憶 512MB の oyabun ではそれほど破滅的なことにはならない。

6 リンク

- 『MATLAB ホームページ』 <http://www.cybernet.co.jp/matlab/>
日本での MATLAB 取り扱い業者のページ。FAQ など日本語で色々な情報が得られる。
- 『Octave Home Page』 <http://www.octave.org/>
- 『Scilab Home Page』 <http://www-rocq.inria.fr/scilab/>
- 『MATLAB Clones』 <http://www.dspguru.com/sw/opensp/mathclo2.htm>
- 『Rlab』 <http://rlab.sourceforge.net/>
- 『Euler』 <http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/euler/index.html>
ドイツ製。区間演算サポート。
- 『ATLAS』

<ftp://ftp.u-aizu.ac.jp/pub/SciEng/numanal/netlib/atlas/>
- 『数値演算言語 Octave』

<http://adlib.rsch.tuis.ac.jp/~akira/unix/octave/index-j.html>
- 『ATLAS による Octave の高速化』

http://mackako.im.uec.ac.jp/chiba-f/octave/octave_speed_up_by_atlas_j.html
- 『Scilab を中心とした MATLAB クローン即席入門講座』

<http://www.bekkoame.ne.jp/~ponpoko/Math/Scilab.html>

参考文献

- [1] 有木進, 工学のための線形代数, 日本評論社 (2000).
- [2] 大石進一, Linux 数値計算ツール, コロナ社 (2000).
- [3] 大石進一, MATLAB による数値計算, 培風館 (2001).
- [4] 小国^{つとむ} 力, MATLAB と利用の実際, サイエンス社 (1995).
- [5] 小国力/Dongarra, Jack J., MATLAB による線形計算ソフトウェア, 丸善 (1998).