第1章　Octave入門

GNU Octave は Matlab に似たフリー数値計算ソフトウェアである．主として，John W. Eaton によって 1992 年頃から本格的に開発が進められ，現在，安定バージョンとして 2.0.16 が，開発中バージョンとして 2.1.35 が提供されている．

Octave は，プログラミングで必要とされる基本言語（ほぼ Matlab 互換）を持ち，対話形式とバッチ形式の両方で使える．ベクトル，行列，複素数の取り扱いが簡単で，行列式や固有値などの行列に関する計算，線形および非線形方程式の求解，多項式演算，微分方程式の求解，gnuplot を用いたグラフ表示等の機能を持つ．Matlab と同じ名前・機能の関数を多く持つが，Octave 固有の関数もあり，特徴を出している．また，制御工学，信号処理などの分野における関数も用意されている．

本テキストは，先に作成したテキスト「MATLAB による動的シミュレーション入門」[1] を Octave 用に書き換えたものである．概ね，[1] に沿っているが，特に，2.7 節の古典制御理論の部分を Octave 関数を用いて加筆し，Matlab の数学的関数に関する関数の部分は Octave が相当する関数を持たないため割愛した．本テキスト作成にあたり使用した Octave のバージョンは 2.1.35 であり，2.5 節までのプログラムはバージョン 2.0.16 でも実行できることを確認している．

では，Octave によるプログラミングを学習しよう．

1.1　Octaveの起動と終了

コマンドウィンドウで

```
octave
```

と入力すれば，Octave を起動できる．Octave を終了するには

```
octave:1> quit
```

または

```
octave:1> exit
```

と入力する．

何らかのトラブルで Octave が暴走した場合，`Ctrl-c`（`Ctrl` キーを押しながら `c` を押す）で中断できる．

Octave Window では，コマンドを入力して実行する．バッチ処理のプログラムはエディタで作り，このファイル（以後 M ファイルという）を拡張子 m を付けて「ファイル名.m」として作業ディレクトリに保存する．それから，Octave Window で「ファイル名」（拡張子は付けない）を入力することにより実行する．

1.2　数値の入出力

変数に数値を代入するには次のようする．

```
octave:1> a = 1
a = 1
octave:2> b = 1.23
b = 1.2300
octave:3> c = 3;
```

変数名として，英数字と ‘_’（アンダースコア）が使える．英字の大文字と小文字は区別され，先頭文字に数字は使えない．また，長さは無制限である．文の最後に ‘;’ を付ければ，出力を抑制できる．

出力書式の指定もできる（%より右はコメントなので打たなくてよい）．

```
octave:4> x = pi            % 円周率
x = 3.1416                  % デフォルト表示
octave:5> format long       % long 表示
octave:6> x
x = 3.14159265358979
octave:7> format short      % short 表示
octave:8> x
x = 3.14
octave:9> format            % デフォルト表示
octave:10> x
x = 3.1416
octave:11> format long e    % 浮動小数点形式の long 表示
octave:12> x
x = 3.14159265358979e+00
octave:13> format short e   % 浮動小数点形式の short 表示
octave:14> x
x = 3.14e+00
octave:15> format
```

複素数も扱え，虚数単位 $\sqrt{-1}$ として

```
i, j, I, J
```

が使える．これらの記号がすでに変数として使われている場合，変数の定義が優先される．

```
octave:16> y = 1+2i
y = 1 + 2i
octave:17> z = i
z = 0 + 1i
octave:18> z*z
ans = -1
```

書式付で出力するにはC言語のスタイルが使える．

```
octave:19> printf("hello, Octave\n");
hello, Octave
octave:20> x = pi;
octave:21> printf("variable x = %f\n",x);
variable x = 3.141593
octave:22> y = 1 + 2i;
octave:23> printf("complex y = %f + %fi\n",real(y),imag(y));
complex y = 1.000000 + 2.000000i
```

1.3 行列の入出力

$$\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

を入力して Octave Window に出力しよう．

```
octave:1> A = [1 2 3;4 5 6;7 8 9]
A =
  1  2  3
  4  5  6
  7  8  9
```

行列要素の入力方法に注意されたい．行ごとに入力し，‘;’で次の行となる．Octaveでは配列を定義する必要がなく，必要な数だけ自動的に確保される．

次に，これをMファイルで実行してみよう．まず，適当なエディタを開き，プログラムを入力する．プログラムは以下のようである．

ファイル名　file1.m

```
A = input('Enter matrix A ')
```

このファイルを例えば file1.m として Octave の作業ディレクトリ（pwd で表示されるディレクトリ）に保存しよう．そして，Octave Window に戻りつぎのように入力する

```
octave:1> file1
Enter matrix A [1 2 3;4 5 6;7 8 9]
A =
  1  2  3
  4  5  6
  7  8  9
```

A がスカラーのとき，'[ ]' を省略できる．行列をプログラムの中に書く場合

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

とすればよい．

演習

1. 次の行列を入力してみよう．

$$\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix} \quad \boldsymbol{C} = \begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 0 & 0 \end{bmatrix}$$

1.4　行列の和・差・積・転置・べき乗

$\boldsymbol{A}$, $\boldsymbol{B}$ 行列の和・差は，単に

```
X = A + B
Y = A - B
```

また，積は

```
X = A*B
```

と書く．もちろん，$\boldsymbol{A}$, $\boldsymbol{B}$ の次元は，各演算が定義できるように整合性を持つとするが，一方がスカラーの場合は例外で行列 A，スカラー t に対して

```
X = A*t
```

という計算ができる．この場合，A のすべての要素に t が掛けられる．

$\boldsymbol{A}$ の共役転置行列は

```
X = A'
```

で求まる．また，$\boldsymbol{A}$ の k 乗（k は任意の実数でよい）は

```
X = A^k
```

で計算される．

例題 1.1 $\boldsymbol{A}, \boldsymbol{B}$ を入力して，$\boldsymbol{A}+\boldsymbol{B}, \boldsymbol{A}-\boldsymbol{B}$ を計算するプログラムを作成せよ．また，つぎの行列について，具体的に計算してみよ．

$$\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 1 & -1 & 1 \\ 2 & 2 & -1 \\ -1 & 2 & 1 \end{bmatrix}$$

（解答例）

```
A = input('Enter matrix A ')
B = input('Enter matrix B ')
X = A + B
Y = A - B
```

（実行結果）

```
Enter matrix A [1 2 3; 4 5 6; 7 8 9]
A =
  1  2  3
  4  5  6
  7  8  9
Enter matrix B [1 -1 1; 2 2 -1; -1 2 1]
B =
   1  -1   1
   2   2  -1
  -1   2   1

X =
   2   1   4
   6   7   5
   6  10  10

Y =
  0  3  2
  2  3  7
  8  6  8
```

演習

1. $\boldsymbol{A}$, $\boldsymbol{B}$ を入力して，$\boldsymbol{AB}$ を計算するプログラムを作成せよ．そして，つぎの場合を計算してみよ．

(1) $\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$, $\boldsymbol{B} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$　(答) $\boldsymbol{AB} = 10$

(2) $\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}$, $\boldsymbol{B} = \begin{bmatrix} 1 & 1 \\ 2 & -2 \\ 3 & -3 \\ 4 & 4 \end{bmatrix}$　(答) $\boldsymbol{AB} = \begin{bmatrix} 30 & 4 \\ 70 & 4 \end{bmatrix}$

(3) $\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$, $\boldsymbol{B} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$　(答) $\boldsymbol{AB} = \begin{bmatrix} 6 \\ 5 \\ 3 \end{bmatrix}$

2. 次の $\boldsymbol{A}$, k に対する $\boldsymbol{A}^k$ を計算してみよ．

(1) $\boldsymbol{A} = \begin{bmatrix} 2 & 1 \\ -1 & 2 \end{bmatrix}$, $k = 5$　(答) $\begin{bmatrix} -38 & 41 \\ -41 & -38 \end{bmatrix}$

(2) $\boldsymbol{A} = \begin{bmatrix} 2 & 1 \\ -1 & 2 \end{bmatrix}$, $k = -2$　(答) $\begin{bmatrix} 0.12 & -0.16 \\ 0.16 & 0.12 \end{bmatrix}$

1.5　行列関数

正則行列 $\boldsymbol{A}$ の逆行列は，単に

```
X = inv(A)
```

で求まる．Octave には，このような便利な関数が多数用意されている．よく使用される関数を表 1.1 に示す．関数の使用法の詳細については，必要に応じて，help コマンドで，例えば

```
octave:1> help inv
```

と打って調べるか，オンラインマニュアル（http://www.octave.org/doc/octave_toc.html）を参照する．ちなみに、線形方程式

$$\boldsymbol{Ax} = \boldsymbol{b}$$

の解は

表 1.1: 代表的な行列関数

関数	意味
norm	行列またはベクトルノルム
rank	行列のランク（階数）
det	行列式
inv	逆行列
eig	固有値と固有ベクトル
expm	行列指数関数

```
x = A\b
```

で計算できる.

例題 1.2 固有値問題

$$\boldsymbol{A}\boldsymbol{v} = \lambda \boldsymbol{v}, \quad \boldsymbol{A} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & -1 \\ 4 & 2 & 0 \end{bmatrix}$$

を解け.
（解答）

```
octave:1> A = [1 2 1;2 1 -1;4 2 0]
A =
  1   2   1
  2   1  -1
  4   2   0
octave:2> [V,lambda] = eig(A)
V =
  -0.57735   0.54944   0.40825   % V = [v1 v2 ... vn]
   0.57735   0.13736  -0.40825
   0.57735   0.82416   0.81650
lambda =
  -2.00000   0.00000   0.00000   % lambda の対角要素は固有値
   0.00000   3.00000   0.00000
   0.00000   0.00000   1.00000
octave:3> lambda = eig(A)        % 固有値のみ計算
lambda =
  -2.00000
   3.00000
   1.00000
```

演習

1. 次の行列について，逆行列と行列式を計算せよ（逆行列 $\boldsymbol{X}$ の検算には $\boldsymbol{XA} = \boldsymbol{I}$ が利用できる）.

(1) $\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 1 & 1 \\ 0 & 1 & 3 & 2 \\ -1 & 1 & 2 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}$

（答）$\boldsymbol{X} = \begin{bmatrix} 0.2 & 0.4 & -0.8 & 0.2 \\ 0.5 & -0.5 & 0.5 & 0 \\ -0.1 & 0.3 & -0.1 & 0.4 \\ -0.1 & 0.3 & -0.1 & -0.6 \end{bmatrix}$, $|\boldsymbol{A}| = 10$

$\boldsymbol{A} = \begin{bmatrix} -1 & 0 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & 0 \end{bmatrix}$ （答）$\boldsymbol{X} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & -1 & 0 \\ 0 & -1 & -1 \end{bmatrix}$, $|\boldsymbol{A}| = -1$

2. 次の行列のランクを求めよ.

(1) $\boldsymbol{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ （答）2　(2) $\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$ （答）2

(3) $\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 2 \\ 1 & 0 & 2 \end{bmatrix}$ （答）2

(4) $\boldsymbol{A} = \begin{bmatrix} 16 & 2 & 3 & 13 \\ 5 & 11 & 10 & 8 \\ 9 & 7 & 6 & 12 \\ 4 & 14 & 15 & 1 \end{bmatrix}$ （答）3

(5) $\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 3 & 6 & 10 \\ 1 & 4 & 10 & 20 \end{bmatrix}$ （答）4

3. 次の $\boldsymbol{A}$ について，$\boldsymbol{e}^A$ を計算せよ.

(1) $\boldsymbol{A} = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix}$ （答）$\boldsymbol{e}^A = \begin{bmatrix} 0.65970 & 0.53351 \\ -0.53351 & 0.12619 \end{bmatrix}$

(2) $\boldsymbol{A} = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix}$ （答）$\boldsymbol{e}^A = \begin{bmatrix} 0.36788 & 0.36788 & 0.18394 \\ 0 & 0.36788 & 0.36788 \\ 0 & 0 & 0.36788 \end{bmatrix}$

1.6 単位行列・零行列・対角行列

n 次単位行列 $\boldsymbol{X} = \boldsymbol{I}(n \times n)$ を作る場合

```
X = eye(n,n)
```

また，零行列 $\mathbf{0}(n \times m)$ の場合

```
zeros(n,m)
```

とする．同様に，すべての要素が 1 の $n \times m$ 行列は

```
ones(n,m)
```

である．

対角行列

$$\boldsymbol{D} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

の場合

```
v=[1 9 4 4 2];
D=diag(v,0)
```

と書く．

1.7 条件判定・繰り返し

1.7.1 if, else, elseif

if の使い方を例題によって説明しよう．

例題 1.3 a, b を入力して，$b \neq 0$ ならば

$$y = a/b \tag{1.1}$$

を出力するプログラムを作成せよ．

(解答例)

```
a = input('Enter a ');
b = input('Enter b ');
if b ~= 0
   y = a/b
endif
```

例題 1.4 x を入力して

$$y = \begin{cases} 1 & \text{if} \quad x \geq 0 \\ -1 & \text{if} \quad x < 0 \end{cases} \tag{1.2}$$

を出力するプログラムを作成せよ.

(解答例)

```
x=input('Enter x ');
if x >= 0
   y = 1;
else
   y = -1;
endif
y
```

例題 1.5 x を入力して

$$y = \begin{cases} 1 & \text{if} \quad x > 0 \\ 0 & \text{if} \quad x = 0 \\ -1 & \text{if} \quad x < 0 \end{cases} \tag{1.3}$$

を出力するプログラムを作成せよ.

(解答例)

```
x=input('Enter x ');
if x > 0
   y=1;
elseif x<0
   y=-1;
else
   y=0;
endif
y
```

`if` ループは何重にもできる. また, `if` の中で `elseif` を何回でも使える. `if` や `while` で使われる関係演算子を表 1.2 に示す.

表 1.2: 関係演算子

演算子	意味
<	$<$
<=	$\leq$
>	$>$
>=	$\geq$
==	$=$
~=	$\neq$

1.7.2 switch

例題 1.6 n を入力し

$n=-1$	ならば	`minus one`
$n=0$	ならば	`zero`
$n=1$	ならば	`one`
その他の場合		`other value`

を出力するプログラムを作成せよ.

(解答例)

```
n=input('Enter n ');
switch n
case -1
   disp('minus one');
case 0
   disp('zero');
case 1
   disp('one');
otherwise
   disp('other value');
endswitch
```

`switch` は次のような使い方もできる.

```
n=input('Enter n ');
switch n
case 1
   disp('1');
case {2,3,4}
```

```
   disp('2 or 3 or 4');
case 5
   disp('5');
otherwise
   disp('other value');
endswitch
```

1.7.3　while

while は次の構文で使う．

```
while　条件
   statements
endwhile
```

while は条件が成立する限り，*statements* を実行する．while ループは break でいつでも中断できる．

例題 1.7 与えられた正数 a を $\epsilon = 0.001$ よりも小さくなるまで 2 で割り続けるプログラムを作成せよ．

（解答例）

```
eps=0.001;
a=input('Enter a ');
while a>=eps
   a=a*0.5
endwhile
```

1.7.4　for

for は一般に次の構文で使われる．

```
for index = start : increment : end
    statements
endfor
```

index を増分 *increment* で変えながら，*statements* を *end* まで実行する．増分は負でもよい．増分を省略すると $increment = 1$ となる．

例題 1.8 n は正整数とする．1 から n までの和を求めるプログラムを作成せよ．

（解答例）

```
n=input('Enter n ');
s = 0;
for i = 1:n
   s = s + i;
endfor
s
```

for も break によっていつでも中断できる．

C 言語の goto に相当する文は Octave にはないが，本節で述べた構文を組み合わせることによって，任意のプログラムを記述できる．

MATLAB 互換のプログラムを書きたい場合，endif，endswitch，endwhile，endfor の代わりに end を使う．

演習

1. 実数 x と正整数 n が与えられたとき

   $$y = 1 + x + x^2 + \cdots + x^n$$

   を計算するプログラムを作成せよ．
   （ヒント）次のプログラムが使える．

   ```
   a = 1;
   y = 1;
   for i = 1:n
     a = a*x;
     y = y + a;
   endfor
   ```

2. 実数 $x(|x| < 1)$ と正数 ϵ が与えられたとき

   $$y = 1 + x + x^2 + \cdots$$

   を計算するプログラムを作成せよ．ただし，終了基準は

   $$|x^k| < \epsilon$$

   とする．

3. 正整数 n が与えられたとき，n の階乗

   $$n! = 1 \cdot 2 \cdot 3 \cdots n$$

   を計算するプログラムを作成せよ．
   （ヒント）次のプログラムが使える．

```
y = 1;
for i = 1:n
  y = y*i;
endfor
```

または

```
x = 1:n;
y = prod(x)        % x の要素の積を計算する関数
```

でもよい.

4. 正整数 n, $r(n \geq r)$ が与えられたとき，組み合わせの数

$$ {}_nC_r = \frac{n(n-1)(n-2)\cdots(n-r+1)}{1\cdot 2\cdot 3\cdots r} $$

を計算するプログラムを作成せよ.

5. 実数 x に対して，次の級数を計算するプログラムを作成せよ．ただし，ϵ を与えた正数とするとき，新たに加える項の絶対値が ϵ より小さくなるまで計算するものとする.

(1) $y = 1 + x + x^2/2! + x^3/3! + \cdots$　（$x = 2$ のとき $y \to 7.38906$）

(2) $y = 1 - x^2/2! + x^4/4! - + \cdots$　（$x = 2$ のとき $y \to -0.41615$）

(3) $y = x - x^3/3! + x^5/5! - + \cdots$　（$x = 2$ のとき $y \to 0.90930$）

6. 次の級数を計算するプログラムを作成せよ．ただし，ϵ を与えた正数とするとき，新たに加える項の絶対値が ϵ より小さくなるまで計算するものとする.

(1) $y = \dfrac{1}{1\cdot 2} + \dfrac{1}{2\cdot 3} + \dfrac{1}{3\cdot 4} + \cdots \quad (= 1)$

(2) $y = \dfrac{1\cdot 2}{1\cdot 3} + \dfrac{1\cdot 2\cdot 3}{1\cdot 3\cdot 5} + \dfrac{1\cdot 2\cdot 3\cdot 4}{1\cdot 3\cdot 5\cdot 7} + \cdots \quad \left(= \dfrac{\pi}{2}\right)$

1.8　種々の行列の作成法

1.8.1　行列要素の指定

行列の要素は

```
x(i)      A(i,j)
```

などと指定する．よって，n を正整数とすると

$$ \boldsymbol{x} = \begin{bmatrix} 1 & 2 & \cdots & n \end{bmatrix} $$

を生成するプログラムは

```
x = [];
for i=1:n
  x(i) = i;
endfor
x = x'
```

と書ける．また

```
x = 1:n
```

と書くこともできる．後者の方が Octave らしい書き方で，前者に比べ計算速度も速い．

ある区間を分割した点をベクトルとして与える以下の関数がある．これらは，関数や微分方程式の解を計算する座標点を与える際によく利用される．

```
octave:1> x = linspace(0,1,5)  % [0, 1] を (5-1) 等分する．
x =
  0.00000  0.25000  0.50000  0.75000  1.00000
octave:2> y = logspace(-1,1,5)
                % [1e-1, 1e+1] を対数的に (5-1) 等分する．
y =
  0.10000  0.31623  1.00000  3.16228  10.00000
```

今度は n 次 Jordan 細胞 $\boldsymbol{J}_n(\lambda)$ を作ってみよう．Jordan 細胞は例えば

$$\boldsymbol{J}_3(\lambda) = \begin{bmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{bmatrix} \qquad \boldsymbol{J}_4(\lambda) = \begin{bmatrix} \lambda & 1 & 0 & 0 \\ 0 & \lambda & 1 & 0 \\ 0 & 0 & \lambda & 1 \\ 0 & 0 & 0 & \lambda \end{bmatrix}$$

という規則的な形をしている．

```
n=input('Enter n ');
lambda=input('Enter lambda ');
J=eye(n,n)*lambda;
for i=1:n-1
  J(i,i+1) = 1;
endfor
J
```

1.8.2 行列のサイズ

いま，プログラム中で $\boldsymbol{A}$ 行列が与えられていて，このサイズ（行と列の数）を知りたいとき

```
d=size(A)
```

とする．すると，$\boldsymbol{A}$ が 3×4 行列の場合，答えが

```
d =
  3  4
```

となる．すなわち，d(1)=3 d(2)=4 という意味である．
$\boldsymbol{A}$ と同じサイズの零行列 $\boldsymbol{Z}$ を作る場合

```
Z=zeros(size(A))
```

とする．同様に，$\boldsymbol{A}$ とサイズが同じで，要素がすべて 1 の行列 $\boldsymbol{S}$ は

```
S=ones(size(A))
```

で作れる．

1.8.3　行列の結合と削除

$\boldsymbol{A}$ に $\boldsymbol{B}$ を追加して新しい行列 $\boldsymbol{C}$ を作る場合以下のようにする．

$$\boldsymbol{C} = \left[\begin{array}{cc} \boldsymbol{A} & \boldsymbol{B} \end{array} \right]$$

の場合

```
C = [A B]
```

また

$$\boldsymbol{C} = \left[\begin{array}{c} \boldsymbol{A} \\ \boldsymbol{B} \end{array} \right]$$

の場合

```
C = [A;B]
```

と書く．
$\boldsymbol{A}$ の第 i 列を削除した行列 $\boldsymbol{X}$ を作る場合

```
X = A;
X(:,i)=[]
```

同様に，$\boldsymbol{A}$ の第 i 行を削除した行列 $\boldsymbol{X}$ を作る場合

```
X = A;
X(i,:)=[]
```

とする．また，$\boldsymbol{A}$ の第 i 列から第 j 列までを削除した行列 $\boldsymbol{X}$ を作る場合

```
X = A;
X(:,i:j)=[]
```

とする．

例題 1.9 動的システム

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \tag{1.4}$$

ただし

$$\boldsymbol{A}(n \times n), \quad \boldsymbol{B}(n \times r)$$

の可制御性行列は次式で定義される．

$$\boldsymbol{U}_c := [\ \boldsymbol{B} \quad \boldsymbol{A}\boldsymbol{B} \quad \cdots \quad \boldsymbol{A}^{n-1}\boldsymbol{B}\ ] \tag{1.5}$$

この行列を作るプログラムを作成せよ．そして，次の $(\boldsymbol{A}, \boldsymbol{B})$ に対する可制御性行列を計算してみよ．

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

（解答例）

```
n=input('Enter n ');
A=input('Enter A ');
B=input('Enter B ');
Uc=B;
X=B;
for i=1:n-1
   X=A*X;
   Uc=[Uc X];
endfor
Uc
```

（実行例）

```
Enter n 3
Enter A [0 1 0; 0 0 1; 1 0 0]
Enter B [1;0;1]
Uc =
  1  0  1
  0  1  1
  1  1  0
```

演習

1. 次の行列を任意の正整数 n に対して作るプログラムを書いてみよう．

$$\boldsymbol{R}_1 = 1 \quad \boldsymbol{R}_2 = \begin{bmatrix} 1 & 2 \\ 0 & 3 \end{bmatrix} \quad \boldsymbol{R}_3 = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 4 & 5 \\ 0 & 0 & 6 \end{bmatrix} \quad \cdots$$

2. $\boldsymbol{v} = [\ 1 \quad 1 \quad 1 \quad 1\ ]$ とする．

```
diag(v,1), diag(v,-1), diag(v,2), diag(v,-2)
```

がどのような行列になるか調べてみよう．

3. 任意の正整数 n について，対角行列

$$\boldsymbol{D} = \begin{bmatrix} 1 & & & \\ & 2 & & \boldsymbol{0} \\ & & \ddots & \\ \boldsymbol{0} & & & n \end{bmatrix}$$

を作るプログラムを作成せよ．
（ヒント）

```
n = input('Enter n ');
v = 1:n;
D = diag(v,0)
```

4. 動的システム

$$\left.\begin{array}{rcl} \dot{\boldsymbol{x}}(t) & = & \boldsymbol{A}\boldsymbol{x}(t) \\ \boldsymbol{y}(t) & = & \boldsymbol{C}\boldsymbol{x}(t) \end{array}\right\} \tag{1.6}$$

ただし

$$\boldsymbol{A}(n \times n), \quad \boldsymbol{C}(m \times n)$$

の可観測性行列

$$\boldsymbol{U}_o := \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{C}\boldsymbol{A} \\ \vdots \\ \boldsymbol{C}\boldsymbol{A}^{n-1} \end{bmatrix} \tag{1.7}$$

を作るプログラムを作成せよ．そして，次の $(\boldsymbol{A}, \boldsymbol{C})$ に対する可観測性行列を計算してみよ．

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \quad \text{（答）} \boldsymbol{U}_o = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

5. 動的システム

$$\left.\begin{array}{rcl} \dot{\boldsymbol{x}}(t) & = & \boldsymbol{A}\boldsymbol{x}(t)+\boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) & = & \boldsymbol{C}\boldsymbol{x}(t) \end{array}\right\} \tag{1.8}$$

の次数を n とするとき，可制御性条件および可観測性条件は

$$\operatorname{rank}\boldsymbol{U}_c = n, \quad \operatorname{rank}\boldsymbol{U}_o = n \tag{1.9}$$

である．可制御性・可観測性を判定するプログラムを作成せよ．そして，次のシステムの可制御性・可観測性を判定してみよ．

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix}$$

（答）$\operatorname{rank}\boldsymbol{U}_c = 3, \quad \operatorname{rank}\boldsymbol{U}_o = 4$ したがって，不可制御・可観測である．

6. $\boldsymbol{A}(n\times n)$, $\boldsymbol{x}(n\times 1)$ とする．差分方程式

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, \quad k = 0, 1, \ldots$$

の解を $k = 0, 1, \ldots, t_f$ について求めるプログラムを作成せよ．そして，次の $\boldsymbol{A}$, $\boldsymbol{x}_0$, t_f について実行してみよ．

$$\boldsymbol{A} = \begin{bmatrix} 0.8 & 1 \\ 0 & 0.8 \end{bmatrix}, \quad \boldsymbol{x}_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad t_f = 20$$

（ヒント）

```
x = input('Enter x ');
tf = input('Enter tf ');
A = [0.8 1; 0 0.8]
y = [];
for i = 1:tf+1
   y(:,i) = x;
   x = A*x;
endfor
y
```

7. 差分方程式

$$y(k+2) + a_1 y(k+1) + a_0 y(k) = 0, \quad y(0) = y_0, \quad y(1) = y_1, \quad k = 0, 1, \ldots$$

の解 $y(k)$, $k = 0, 1, \ldots, t_f$ を求めるプログラムを作成せよ．そして，次のデータについて実行してみよ．

$$a_0 = 0.25, \quad a_1 = -1, \quad y(0) = 1, \quad y(1) = 1, \quad t_f = 20$$

8. n を正整数とする．フィボナッチ数列

   1, 1, 2, 3, 5, 8, ...

   （第 1 項 = 第 2 項 = 1 とし，第 i 項を第 $i-2, i-1$ 項の和により求める）

   を第 n まで作るプログラムを作成せよ．

9. 実数 x と正整数 n が与えられたとき，ベクトル

   $$\boldsymbol{y} = \left[\begin{array}{ccccc} 1 & x & x^2 & \ldots & x^n \end{array} \right]$$

   を作るプログラムを作成せよ．

10. [0, 1] の範囲で n 個の乱数（実数）

    $x_1, x_2, x_3, \ldots, x_n$

    を発生させ，この内の最大値と最小値を求めよ．そして，小さい順に並べ替えよ．
    （ヒント）

```
n=input('Enter n ');
x = rand(1,n)          % 1 × n の一様分布乱数行列を発生させる関数
xmax = max(x)          % 最大値を求める関数
xmin = min(x)          % 最小値を求める関数
y = sort(x)            % 小さい順に並べ替える関数
```

11. [0, 1] の範囲で n 個の乱数（実数）

    $x_1, x_2, x_3, \ldots, x_n$

    を発生させ，大きい順に並べ替えよ．
    （ヒント）大きい順に並べ替える関数は用意されていないので，やはり `sort` を利用する．

1.9　数学関数

sin, cos などの数学関数が一通り用意されている．また，このような関数はベクトル計算もできる．例えば，$\sin(t)$ を種々の t に対して計算する場合

```
y = [];
i = 0;
for t = 0:0.1:10
    i = i + 1;
    y(i) = sin(t);
endfor
y = y'
```

表 1.3: 代表的数学関数

関数	意味
`sin(x)`	$\sin x$
`cos(x)`	$\cos x$
`tan(x)`	$\tan x$
`asin(x)`	$\arcsin x$
`acos(x)`	$\arccos x$
`atan(x)`	$\arctan x$
`atan2(y,x)`	4 象限逆正接
`sinh(x)`	$\sinh x$
`cosh(x)`	$\cosh x$
`tanh(x)`	$\tanh x$
`exp(x)`	e^x
`log(x)`	$\ln x$
`log10(x)`	$\log_{10} x$
`sqrt(x)`	$\sqrt{x}$
`abs(x)`	$\|x\|$
`real(x)`	複素数の実部
`imag(x)`	複素数の虚数部
`sign(x)`	符号関数

の代わりに

```
t = 0:0.1:10;
y = sin(t);
```

とできる．後者の方が Octave に適した書き方で，計算速度も前者より速い．ちなみに，後者のプログラムで得た `t`，`y` をグラフ表示するには

```
grid "on"
plot(t,y)
```

とする（図 1.1）．代表的な数学関数を表 1.3 に示す.

例題 1.10 関数

$$y(t) = e^{-0.5t} \cos 5t \tag{1.10}$$

のグラフを描け．ただし，t の範囲を $[0, 10]$，プロット点の刻み幅を 0.01 とする．

（解答例）

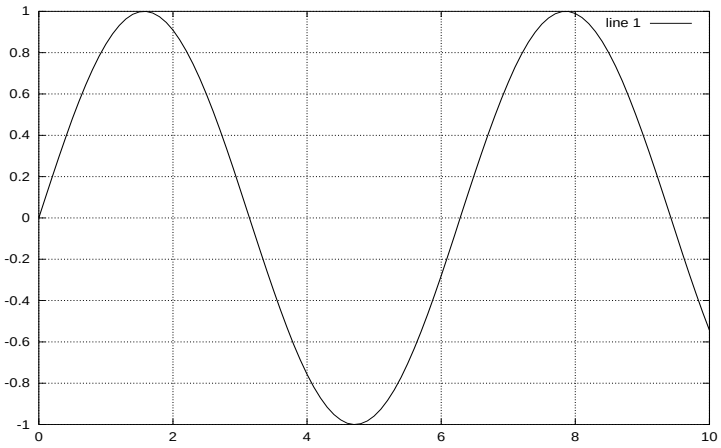

図 1.1: $y = \sin t$ のグラフ

```
t = 0:0.01:10;
y = exp(-0.5*t).*cos(5*t);
grid "on"
plot(t,y)
```

‘.*’ は要素毎の掛け算を表す．グラフを図 1.2 に示す．

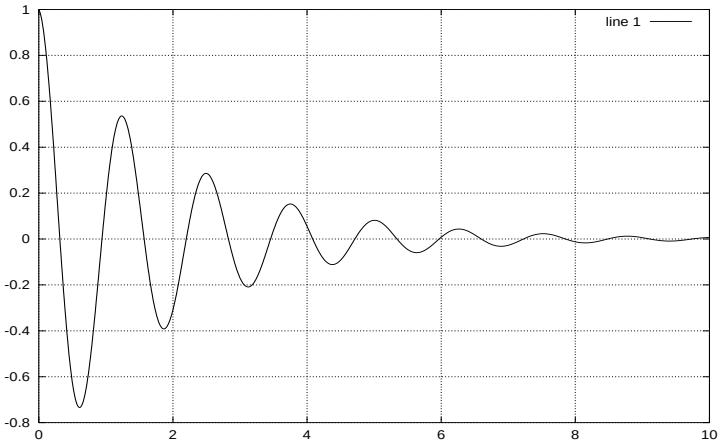

図 1.2: $y = e^{-0.5t} \cos 5t$ のグラフ

例題 1.11 図 1.3 に示す関数は

$$f(x) = \frac{4k}{\pi} \left(\sin x + \frac{1}{3} \sin 3x + \frac{1}{5} \sin 5x + \cdots \right) \tag{1.11}$$

とフーリエ級数展開される．この級数を第 n 項まで計算した結果をグラフ表示せよ．ただし，グラフ表示の範囲を $[-10, 10]$，プロット点の刻み幅を 0.02 とする．

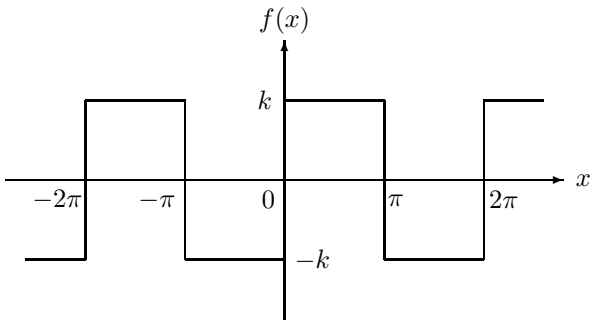

図 1.3: 矩形波関数

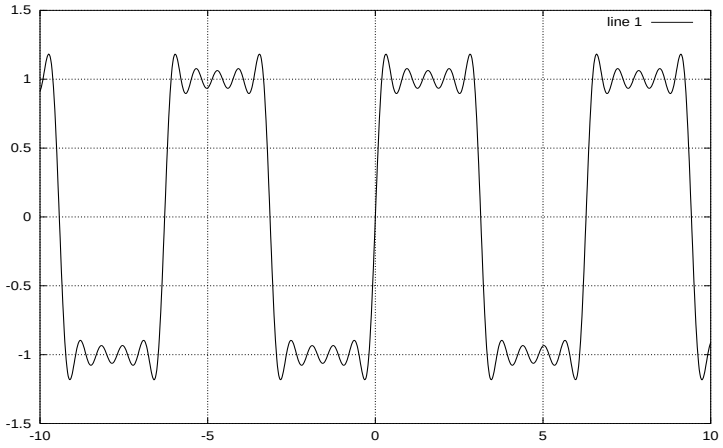

図 1.4: 例題 1.11 $k = 1$, $n = 5$ の場合

（解答例）

```
k = input('Enter k ');
n = input('Enter n ');
x = -10:0.02:10;
y = zeros(size(x));
a = 1;
for i=1:n
   y = y + sin(a*x)/a;
   a = a + 2;
endfor
```

```
y = y*4*k/pi;
grid "on"
plot(x,y)
```

演習

1. 図 1.5 に示す関数は

$$f(x) = \frac{\pi}{2} - \frac{4}{\pi}\left(\cos x + \frac{1}{3^2}\cos 3x + \frac{1}{5^2}\cos 5x + \cdots\right) \tag{1.12}$$

とフーリエ級数展開される．この級数を第 n 項まで計算した結果をグラフ表示せよ．ただし，グラフ表示の範囲を $[-10, 10]$，プロット点の刻み幅を 0.02 とする．

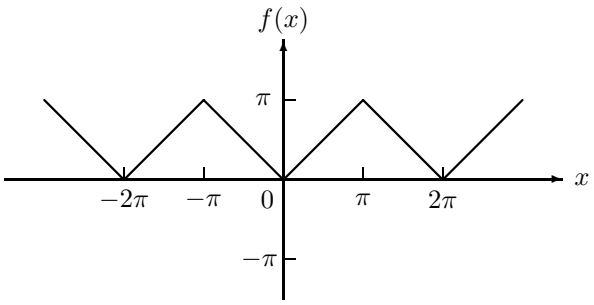

図 1.5: 三角波関数

1.10　多項式に関する関数

多項式に関する代表的な関数を表 1.4 に示す.

1.10.1　多項式の表現，零点，特性方程式

Octave では，多項式を降べき順に係数を並べたベクトルとして表現する．すなわち，多項式

$$p(x) = -x^4 + 20x^2 - 20x + 5 \tag{1.13}$$

は

```
octave:1> p = [-1 0 20 -20 5]
```

で表される．

この多項式の零点（多項式 $= 0$ の根）は `roots` を用いて次のように求められる．

表 1.4: 多項式に関する関数

関数	説明
roots	根を求める
poly	指定根を与える多項式
poly	正方行列に対する特性多項式
polyval	多項式の値
polyvalm	行列多項式の値
polyfit	多項式によるデータの最小二乗内挿
conv	多項式の掛け算
deconv	多項式の割り算
residue	有理関数の部分分数展開

```
octave:2> r = roots(p)
r =
  -4.92606
   3.89858
   0.57356
   0.45393
```

逆に，零点から多項式の係数を求めるには

```
octave:3> p1 = poly(r)
p1 =
    1.0000e+00   -2.2760e-15  -2.0000e+01   2.0000e+01   -5.0000e+00
```

とする．poly は，正方行列 $\boldsymbol{A}$ に対する特性多項式を求めるときにも使用できる．例えば

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

の場合

```
octave:4> A = [0 1 0; -1 0 0; 0 -1 2];
octave:5> p2 = poly(A)
p2 =
   1  -2   1  -2
```

と計算できる．すなわち，特性多項式は

$$p_2(x) = |x\boldsymbol{I} - \boldsymbol{A}| = x^3 - 2x^2 + x - 2$$

である．

$p(x)$ の $x = 1$ における値は，polyval を用いて

```
octave:6> y = polyval(p,1)
y = 4
```

と計算できる．

例題 1.12 polyval 関数を使って $p(x) = -x^4 + 20x^2 - 20x + 5$ のグラフを描いてみよう．ただし，x の範囲を $[-5, 5]$，プロット点の刻み幅を 0.02 とせよ．

（解答例）

```
p =[ -1 0 20 -20 5];
x = -5:0.02:5;
y = polyval(p,x);
grid "on"
plot(x,y)
```

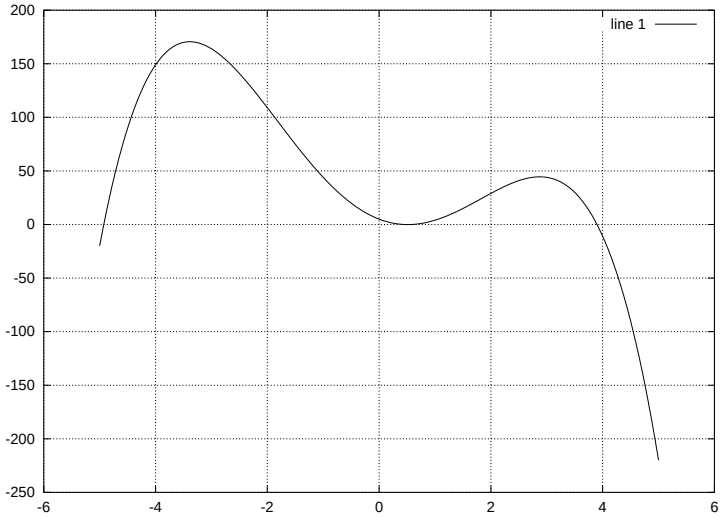

図 1.6: $p(x) = -x^4 + 20x^2 - 20x + 5$ のグラフ

1.10.2　多項式曲線のあてはめ

polyfit は，平面上の与えられたデータ点の集合を n 次多項式曲線で最小二乗近似する関数である．次のデータを考えよう．

$$\boldsymbol{x} = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{bmatrix}$$

$$\boldsymbol{y} = \begin{bmatrix} 0 & 3 & 4 & 10 & 18 & 27 & 41 & 59 & 79 & 93 & 101 \end{bmatrix}$$

このデータを n 次多項式曲線で近似するプログラムは次のように書ける.

```
n = input('Enter n ');
x = 0:10;
y = [0 3 4 10 18 27 41 59 79 93 101];
p = polyfit(x,y,n)
x1 = 0:0.1:10;
y1 = polyval(p,x1);
grid "on"
plot(x,y,'+',x1,y1)     % (x,y) データを+で表示（他に，o,*,x など）
                        % (x1,y1) データを線で結ぶ
```

$n = 3$ とした場合の実行例を図 1.7 に示す.

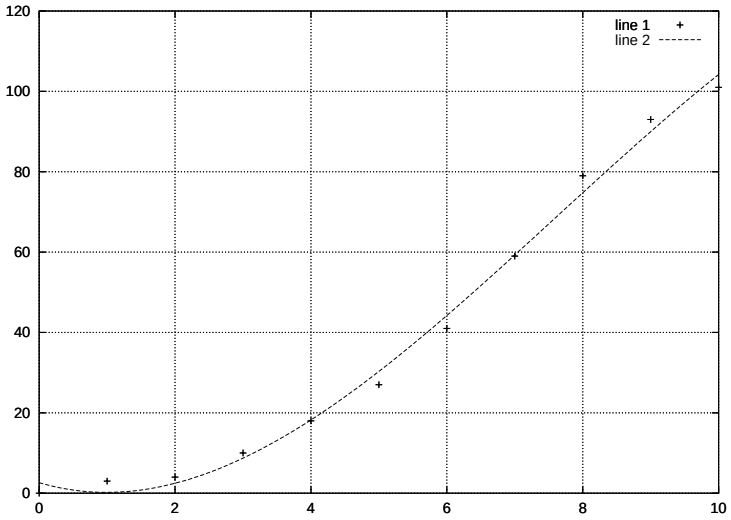

図 1.7: データ点に対する多項式曲線のあてはめ

1.11 ユーザ定義関数の書き方

ユーザ定義関数の書き方を例題で見てみよう.

例題 1.13 n 個のデータ $x_1, x_2, \cdots, x_n$ の平均 $\bar{x}$ と分散 σ^2 は次式で求められる.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{1.14}$$

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 \tag{1.15}$$

ベクトル $\boldsymbol{x}$ を

$$\boldsymbol{x} := \left[\begin{array}{cccc} x_1 & x_2 & \cdots & x_n \end{array}\right]$$

と定義する．$\boldsymbol{x}$ を入力して，$\bar{x}, \sigma^2$ を出力する関数を作成せよ．

（解答例）

ファイル名：heikin.m

```
function [a, b] = heikin(x)
% 平均と分散を求める関数　a:平均　b:分散
n = length(x);  % x の要素数
a = sum(x)/n;   % sum(x):要素の総和
b = sum((x-a).^2)/n;
endfunction
```

この関数を呼ぶプログラムの例を次に示す．

```
octave:1> x = rand(1,5)
x =
  0.70536  0.31711  0.52546  0.80152  0.14361
octave:2> [mean,sigma2] = heikin(x)
mean = 0.49861
sigma2 = 0.058838
```

引数（複数の場合コンマで区切る）を関数に渡し，戻り値（複数の場合コンマで区切る）を関数から得る．引数と戻り値以外の関数中の変数はメインプログラムや他の関数プログラムに対して独立している．もちろん，引数や戻り値は行列でもよい．

関数名も変数名と同様，英字で始め，他に数字やアンダーバーの記号が使える．ファイル名は

関数名.m

とする．

function 後のコメントは help コマンドで参照される．

関数をファイルに保存して，M ファイルとして使用する場合，MATLAB との互換性を考慮して，endfunction を省略できる．

1.12 ファイルに対する入出力

数値などのファイルへの出力には fprintf，ファイルからの入力には fscanf 関数を使う．引数の書き方は C 言語に従う．ベクトルを扱えるところが C 言語と大きく違う点である．以下に例を示す．

```
octave:1> x = 0:0.1:1;
octave:2> y = [x; exp(x)];
octave:3> fid = fopen("exp.dat','w');
octave:4> fprintf(fid,'%6.2f %12.8f \n',y);
octave:5> fclose(fid);
```

以上を実行すると，カレントディレクトリに次の内容のファイル exp.dat が出力される．

```
 0.00   1.00
 0.10   1.11
 0.20   1.22
 0.30   1.35
 0.40   1.49
 0.50   1.65
 0.60   1.82
 0.70   2.01
 0.80   2.23
 0.90   2.46
 1.00   2.72
```

exp.dat を変数名 a で読み込む場合，次のようにする．

```
octave:6> fid = fopen("exp.dat");
octave:7> a = fscanf(fid,'%f %f',[2 inf]);   % It has two rows now.
octave:8> a = a';
octave:9> fclose(fid);
```

読み込み専用ファイルでオープンするには fopen で"r"を指定してもよいが，これはデフォルトなので省略することもできる．[2 inf] は,「2 次元配列でファイルの最後まで読み込め」というオプションである．

関連図書

[1] 吉田和信：MATLAB による動的システムシミュレーション入門,
http://www.ecs.shimane-u.ac.jp/ kyoshida/matlab.htm, 2001.

[2] John W. Eaton: GNU Octave A high-level interactive language for numerical computations Edition 3 for Octave version 2.0.5,
http://www.octave.org/doc/octave_toc.html, 1997.

[3] 佐久間元敬他：線形代数教科書，共立出版，1978.

[4] 志賀浩二：数学 30 講シリーズ１ 微分積分 30 講，朝倉書店，1988.

[5] 伊理正夫他：別冊・数学セミナー 現代応用数学の基礎１，日本評論社，1987.

[6] 吉田和信他:重心移動による振子系の振動制御，システム制御情報学会論文誌，pp.470-477, 2000.

[7] 加藤寛一郎：最適制御入門 レギュレータとカルマンフィルタ，東京大学出版会，1987.

[8] 日高照晃他：機械力学–振動の基礎から制御まで–，朝倉書店，2000.

[9] 奥山佳史他：制御工学–古典から現代まで–，朝倉書店，2001.

[10] 浪速智英：Octave/Matlab で見るシステム制御，科学技術出版，2000.

Octave による動的システムシミュレーション入門

2002 年 2 月 1 日　初版