計算数学1
MATLAB 入門

MATLAB は MATrix LABoratory の略で、数値・数式処理やグラフィックスが手軽にできるソフトウェアである。同様な機能でフリーなものにScilab, Octave などがある。理科大システムでは、MATLAB, Scilab が使える。

1 MATLAB の使用例:

n 次正方行列 A と n 次元ベクトル x の積

C言語だと・・・

(配列 A[n][n], x[n], y[n] の宣言, i , j の宣言と y の初期化の後)

```
for(i=0; i<n; i++){
  for(j=0; j<n; j++){
    y[i]+=A[i][j]*x[j];
  }
}
```

MATLAB では・・・

```
y = A*x;
```


1.1 数値, 変数, 配列

数値

- ■整数と実数を区別しない
- ■型の指定が不要である
- ■データはすべて配列として扱われる

変数

- ■変数の名前は英字で始まる
- ■大文字と小文字は区別される
- ■MATLAB関数の名前とは異なる必要がある

例:

```
>> a=1; b=2; c=a+b ↵
c =
   3
```

↵ :リターンキー

変数 a に 1, 変数 b に 2 を代入した後、c=a+b を計算している。
式の終わりがセミコロンの場合、結果は出力されない。

```
>> a=1, b=2, c=a+b ↵
a =
   1
b =
   2
c =
   3
```

```
>> a=1; b=2; c=a+b; ↵
>>
```


継続行: 行末に3つのピリオドを書くと継続できる:

```
>> a1=1; a2=2; a3=3; a4=4; ... ↵
a5=5; a6=6; a7=7; a8=8; ... ↵
a1+a2+a3+a4+a5+a6+a7+a8 ↵
ans =
   36
```

代入を行っていない場合、結果は変数 ans に保持される。
代入をせず、結果だけを表示させたい場合は disp を使う:

```
>> a1=1; a2=2; a3=3; a4=4; ... ↵
a5=5; a6=6; a7=7; a8=8; ... ↵
disp(a1+a2+a3+a4+a5+a6+a7+a8) ↵
   36
```

特別な変数

変数名	意味
pi	円周率 π
Inf, inf	無限大 ∞
i, j	虚数単位 $\sqrt{-1}$
eps	マシンイプシロン（1+ ε >1 となる最小の値）
	>> eps ans = 2.220446049250313e-016 ← 例
eye	単位行列
ones	全要素が 1 の行列
zeros	全要素が 0 の行列
NaN, nan	数値でない（Not a Number）
realmin	表現できる正の最小浮動小数点数
realmax	表現できる正の最大浮動小数点数

注意： 別の値を代入すると値が変わってしまう!

clear 命令で初期状態に戻せる：

```
>> pi
ans =
   3.14159265358979
>> pi=6
pi =
     6
>> 2*pi
ans =
    12
>> clear pi
>> 2*pi
ans =
   6.28318530717959
```


数値の出力の変更

```
>> x=1/3
x =
    0.3333
>> format short e; x
x =
  3.3333e-001
>> format long; x
x =
   0.33333333333333
>> format long e; x
x =
   3.333333333333333e-001
>> format hex; x
x =
   3fd5555555555555
```

short (標準)	5桁の固定小数点表示
short e	5桁の浮動小数点表示
long	15桁の固定小数点表示
long e	15桁の浮動小数点表示
hex	16進表示
short g や long g もある。	
（固定小数点か浮動小数点か自動表示）	

表示形式は、変更するまで有効である。
標準に戻すには format と入力

配列

ベクトル・・・1次元配列
行列・・・・・・2次元配列

配列の添え字には 1 から始まる

```
>> x=[1 2 3]
x =
     1     2     3
>> x'
ans =
     1
     2
     3
>> y=[1;2;3]
y =
     1
     2
     3
```

行ベクトル

行ベクトルの右肩に ' （転置）で
列ベクトルを表す

列ベクトル
セミコロン ; は行の区切りを表す

ベクトルの生成方法

■ 初期値，増分，最終値をコロンで区切って指定

```
>> x=[1:2:7]
x =
     1     3     5     7
>> x=[1:4]
x =
     1     2     3     4
```

1 から 7 まで 2 刻みで生成

増分が 1 の時は省略可

■ 関数 linspace を使って，初期値，最終値，要素の数を指定

```
>> y=linspace(1,3,5)
y =
     1     1.5     2     2.5     3
```

1 から 3 まで等間隔で
5要素生成


```
>> A=[1 2 3; 4 5 6]
A =
     1     2     3
     4     5     6
>> A'
ans =
     1     4
     2     5
     3     6
>> a=A(:)
a =
     1
     4
     2
     5
     3
     6
```

2行3列の行列

右肩に ' をつけると転置行列

2次元配列を行優先で
1次元配列に変換

配列要素の指定

1次元配列 x の第 k 要素は x(k)
2次元配列 A の i 行 j 列要素 は A(i,j)

```
>> x=[1 2 3]
x =
    1    2    3
>> x(2)
ans =
    2
>> A=[1 2 3;4 5 6]
A =
    1    2    3
    4    5    6
>> A(2,2)
ans =
    5
```


1.2 便利なコマンドとコントロールキー

■ help コマンド

>> help *command*
とすると, *command* の使い方を調べることができる。

```
>> help format
  FORMAT   出力書式の設定
  FORMAT は、MATLAB での計算方法に影響を与えません。たとえば、浮動小数点
  (単精度および倍精度)変数の計算は、変数の表示方法に依らず、適切な浮動
  小数点精度(単精度または倍精度)で行われます。
  FORMATは、つぎに示すような、異なる表示書式間の切り替えを行います。

  MATLABは、すべての計算を倍精度で実行します。
  FORMATは、つぎに示すような、異なる表示書式間の切り替えを行います。
    FORMAT       デフォルト。SHORTと同じです。
    FORMAT SHORT    5桁のスケーリングされた固定小数点。
    FORMAT LONG     15桁のスケーリングされた固定小数点。
    FORMAT SHORT E  5桁の浮動小数点。
    FORMAT LONG E   15桁の浮動小数点。
    ...
```


■ lookfor コマンド

コマンド名が分からないときに, キーワードをもとに探すことができる。
例えば, 最大値に関して調べたいときは lookfor max と入力する

```
>> lookfor max
svl.m: %function [sv,msv,E] = svl(A,sigma,n,kmax);
svlsym.m: %function [sv,msv,E] = svlsym(A,sigma,n,kmax);
svs.m: %function [sv,msv,E] = svs(A,n,kmax);
svssym.m: %function [sv,msv,E] = svssym(A,n,kmax);
RANDOM       Random numbers in [min,max],  default [-1,+1]
RANDOMC      Complex random numbers in M+i*M with M=[min,max], default [-1,+1]
LONGPRECISION  Sets/gets maximum precision for long number arithmetic
 NAMELENGTHMAX  MATLABの関数または変数名の最大長
BITMAX 最大の浮動小数点整数
 NAMELENGTHMAX   MATLAB名の最大の長さ
INTMAX Largest positive integer value.
 REALMAX 正の最大浮動小数点数
 MAX   最大要素
 NZMAX   行列内の非ゼロ要素に対して割り当てられるストレージの総量
...
```


■ 矢印キーやコントロールキー

MATLAB では入力したコマンドの履歴がメモリ中に残っている

↑	直前の行を呼び出す
↓	直後の行を呼び出す
→	一文字右に移動
←	一文字左に移動
Delete	左の一文字を削除
Ctrl-A	行の先頭に移動
Ctrl-E	行の終わりに移動
Ctrl-U	現在の行を削除
Ctrl-K	行の終わりまでを削除
Ctrl-D	カーソルが示す文字を削除

1.3 演算, 数学関数

演算

式	MATLABでの表記
$a+b$	a + b
$a-b$	a - b
ab	a * b
a/b	a / b
a^b	a^b

比較演算 → 真のときは 1, 偽のときは 0 を返す

式	MATLABでの表記
$a<b$	a < b
$a\le b$	a <= b
$a>b$	a > b
$a\ge b$	a >= b
$a=b$	a == b
$a\ne b$	a ~= b
and	&
or	|

要素ごとの演算

例えば, $A=\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}, B=\begin{pmatrix}1 & 2\\ 2 & 1\end{pmatrix}$ のとき,

```
>> A*B
```

とすると, $AB=\begin{pmatrix}5 & 4\\ 11 & 10\end{pmatrix}$ が計算される

$\left(a_{ij}b_{ij}\right)$ のように, 行列 A の各要素に行列 B の各要素をかけた行列

が欲しい場合は, 「*」 の代わりに「.*」とする

```
>> A .* B
ans =
   1   4
   6   4
```


ベクトルの場合

$\mathbf{a}=(a_1,\cdots,a_n), \mathbf{b}=(b_1,\cdots,b_n)$ のとき

式	MATLABでの表記
$(a_1+b_1,\cdots,a_n+b_n)$	a + b
$(a_1b_1,\cdots,a_nb_n)$	a .* b
$(a_1/b_1,\cdots,a_n/b_n)$	a ./ b
$(a_1^{b_1},\cdots,a_n^{b_n})$	a .^ b

数学関数

MATLAB では数学関数の活用が重要になる。
関数の使い方は help コマンドで確認すること。

基本数学関数

abs	絶対値
sqrt	平方根
log	自然対数
exp	指数関数
real	複素変数の実部
imag	複素変数の虚部
angle	複素変数の偏角
conj	複素変数の複素共役
・・・	

基本統計関数

sum	要素の和
mean	平均値
cov	共分散
corrcoef	相関係数
sort	昇順または降順にソート
prod	要素の積
cumsum	累積和
cumprod	累積積
・・・	

その他, 特殊関数や数式処理など多数

三角関数

sin	$\sin(x)$
cos	$\cos(x)$
tan	$\tan(x)$
asin	$\sin^{-1}(x)$
acos	$\cos^{-1}(x)$
atan	$\tan^{-1}(x)$
sinh	$\sinh(x)$
cosh	$\cosh(x)$
・・・	

行列関数

inv	逆行列
det	行列式
lu	LU分解
chol	コレスキー分解
norm	(各種)ノルム
cond	条件数
eig	固有値
・・・	

1.4 スクリプトと関数

file-name.m (Mーファイルと呼ぶ)には,
・スクリプトファイル
・関数ファイル
の2種類がある

スクリプトファイル・・・一連のコマンドを並べたもの

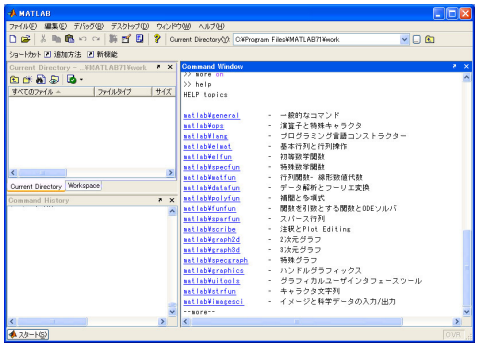
ファイル名を打ち込んで実行

関数ファイル・・・・・・自分で定義した関数を記述したもの

ファイル名が関数の名前となる。
カレントディレクトリ* にファイルがあれば
何もしなくても関数として利用できる。

* 理科大の環境では z:¥ ドライブを使うこと

Current Directory を z:¥ に設定すること

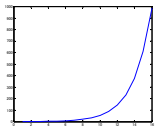


スクリプトの例

```
% M-file to calculate Fibonacci numbers
f=[1 1]; i=1;
while f(i)+f(i+1)<1000
   f(i+2)=f(i)+f(i+1);
   i=i+1;
end
disp(f)
plot(f)
```

→ fibo.m

```
>> fibo
 Columns 1 through 11
    1    1    2    3    5    8   13   21   34   55   89
 Columns 12 through 16
  144  233  377  610  987
```



関数の定義

Mーファイルの先頭に次のように記述する:

function [出力変数のリスト] = 関数名(入力変数のリスト)

関数の例

```
% Function to work out the sum and remainder
% of two real numbers
function [wa,sa]=cal(a,b)
wa=a+b;
sa=a-b;
```

→ cal.m

実行例

```
>> x=1; y=2;
>> [c,d]=cal(x,y)
c =
     3
d =
    -1
```


関数のコメント

M-ファイルの先頭にある % 文も, help コマンドで表示される。
例えば, flow.m というファイルが

```
% 近似解をプロットするプログラム
function [f] = flow(K);
...
```

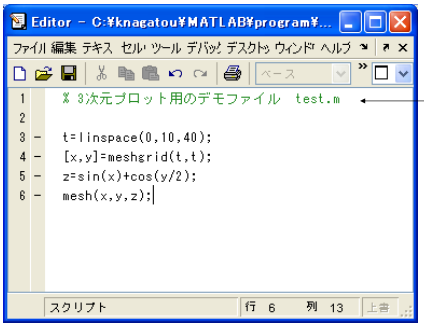
のような場合, 次のように表示される:

```
>> help flow
 近似解をプロットするプログラム
```

M-ファイルの先頭には中身が分かるようなコメントを書いておくこと!

Mーファイルの作成: 理科大の環境では z:¥ ドライブに保存すること

[ファイル] → [新規作成] → [M-ファイル]



→ %以下はコメント

重要:
プログラミングの際は
こまめにコメントを書く

1.5 条件, 繰り返し

if 文

```
if 式, 命令
else if 式, 命令
else 命令
end
```

例

```
>> x=-1;
>> if x>0, y=1;
   elseif x==0, y=0;
   else y=-1;
   end
>> y
y =
    -1
```

※else if や else は省略可能。
, を省略し、次の行に書いてもよい

ループ

ある処理を何度も繰り返すことが必要な場合があり、
そのとき繰り返されるコマンド群を**ループ**,
制御する変数のことを**制御変数**あるいは**ループ変数**という

ループには二種類の方法がある:

■ 繰り返す回数を指定

→ for 文

■ 繰り返しを終了する条件(終了条件)を指定

→ while 文

for 文

```
for  ループ変数 = ループ変数の範囲
       命令
end
```

例:

```
>> s=0;
>> for i=1:10
     s=s+i;
   end;
>> disp(s)
   55
```

→ ループ変数は i, 繰り返しは 1 から 10 まで（10回の繰り返し）

```
>> s=0;
>> for i=10:-1:1
     s=s+i;
   end;
>> disp(s)
   55
```

→ ループ変数の大きい値から順に処理したい場合は，このように記述する（情報落ちを防ぎたい場合など）

ループ変数の範囲をとびとびに設定することもできる:

例:

```
>> s=0;
>> for i=2:2:10
     s=s+i;
   end;
>> disp(s)
   30
```

→ ループ変数は i が偶数の場合のみ（i = 2 から 2刻みで10まで）

```
>> s=0;
>> for i=10:-2:2
     s=s+i;
   end;
>> disp(s)
   30
```

→ ループ変数の大きいほうからの場合

```
>> v=[1 10 100];
>> for k=v
     disp(k)
   end
    1
   10
  100
```

→ ループ変数 k の範囲をベクトルで指定
k はベクトル v の要素の値を順にとる

ループの途中で次のループに移るときは continue を，
ループを抜けるときは break を用いる

例:

```
>> s=0;
>> for k=[1 -1 2 -2 3 -3 4 -4 5 -5]
     if k<0, continue;
     end
     s=s+k;
     if s>5, break;
     end
     disp(s);
   end
    1
    3
```

→ k が負の場合は次のループ変数へ

→ s が 5 より大きくなったらループを終了

while 文

```
while  条件
       命令
end
```

例:

```
>> a=10;
>> while a>=1
     a=a/2;
     disp(a)
   end
    5
   2.50000000000000
   1.25000000000000
   0.62500000000000
```

→ a が 1 以上の間は処理を実行

break や continue の使い方は for 文と同様

2 グラフィックス

■ ezplot ･･･関数のグラフを表示

ezplot('*f(x)*') → $-2\pi \le x \le 2\pi$ の範囲で f(x) をプロット

ezplot('*f(x)*', [a,b]) → $a \le x \le b$ の範囲で f(x) をプロット

ezplot('*f(x,y)*') → $-2\pi \le x, y \le 2\pi$ の範囲で f(x,y)=0 をプロット

ezplot('*f(x,y)*', [a,b]) → $a \le x, y \le b$ の範囲で f(x,y)=0 をプロット

ezplot('*f(x,y)*', [a,b,c,d]) → $a \le x \le b, c \le y \le d$ の範囲で f(x,y)=0 をプロット

ezplot('*x(t), y(t)*') → $0 \le t \le 2\pi$ の範囲で x=x(t), y=y(t) をプロット

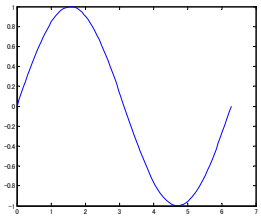
ezplot('*x(t), y(t)*', [a,b]) → $a \le t \le b$ の範囲で x=x(t), y=y(t) をプロット

■ plot ･･･二次元グラフを表示

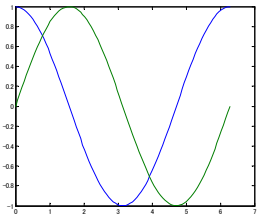
（x, y のデータをベクトルとして与える）

例:

```
>> x=linspace(0, 2*pi, 100);
>> plot(x, sin(x));
```



```
>> x=linspace(0, 2*pi, 100);
>> plot(x, sin(x), x, cos(x));
```



グラフはFigure 1 というウィンドウに表示される

→ 次回の描画では消去される（上書き）

```
>> x=linspace(0, 2*pi, 100);
>> figure(1); plot(x, sin(x));
>> figure(2); plot(x, cos(x));
```

こうすると，Figure 1 と Figure 2 という別々のウィンドウに描画される。

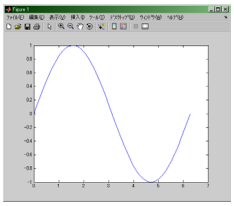


Figure 1

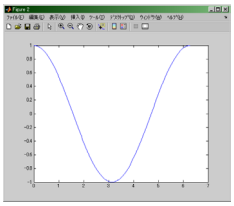
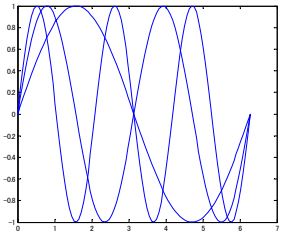


Figure 2

一つのウィンドウに重ねてグラフを描きたい場合は
hold on を利用する（hold off で重ね描き解除）

```
>> x=linspace(0, 2*pi, 100);
>> plot(x,sin(x));
>> hold on;
>> plot(x,sin(2*x));
>> plot(x,sin(3*x));
```

$\sin(x), \sin(2x), \sin(3x)$
のグラフが重ね描きされる

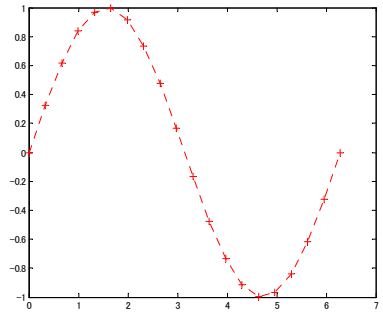


グラフの装飾

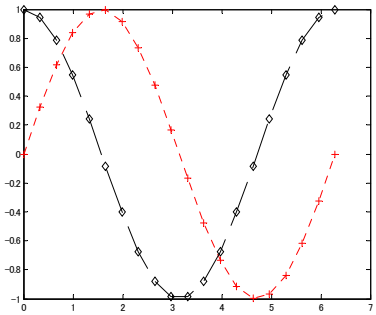
plot(x, y, ‘s’) の形式で s の部分に以下の記号を用いる

b	青	.	点	-	実線
g	緑	o	円	:	点線
r	赤	x	x印	-.	鎖線
c	シアン	+	プラス記号	--	破線
m	マゼンタ	*	星印	(none)	線なし
y	黄	s	正方形		
w	白	d	ダイアモンド		
k	黒	v	三角形(上向き)		
		^	三角形(下向き)		
		<	三角形(左向き)		
		>	三角形(右向き)		
		p	五角形		
		h	六角形		


```
>> x=linspace(0, 2*pi, 20);
>> plot(x, sin(x), 'r+:')
```

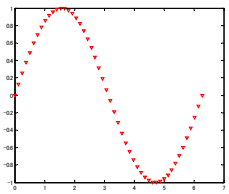



```
>> x=linspace(0, 2*pi, 20);
>> plot(x, sin(x), 'r+:', x, cos(x), 'kd--')
```

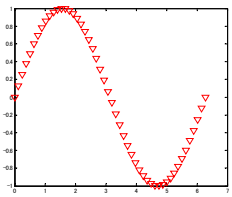


マーカーのサイズは MarkerSize で指定：

```
>> x=linspace(0, 2*pi, 50);
>> plot(x, sin(x), 'rv', 'MarkerSize', 5)
```



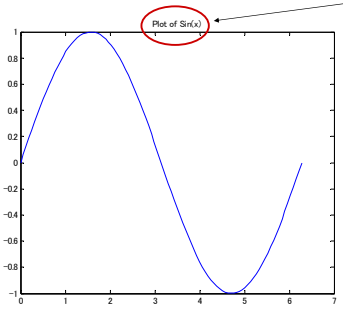
```
>> x=linspace(0, 2*pi, 50);
>> plot(x, sin(x), 'rv', 'MarkerSize', 10)
```



グラフの上部にタイトルを表示するには title('*text*') ：

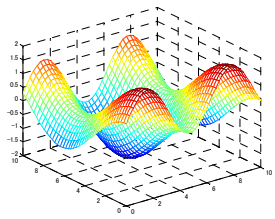
```
>> x=linspace(0, 2*pi, 100);
>> plot(x, sin(x)); title('Plot of Sin(x)');
```

タイトル



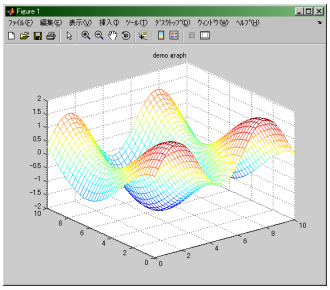
■ 三次元グラフィックス

```
>> t=linspace(0,10,40);
>> [x,y]=meshgrid(t,t);
>> z=sin(x)+cos(y/2);
>> mesh(x,y,z);
```



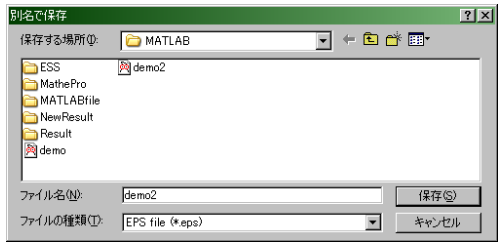
他にezplot3 など

2.1 実行結果のグラフを保存



→ 結果をPostScriptファイルとして保存したい場合

メニューバーから, [ファイル] → [別名で保存] を選択,
ファイルの種類を「EPS file (*.eps)」とすればよい



EPSファイルは TeX文書 に貼り付けることができる

3 変数の保存について

定義した変数や配列情報などを保存したい場合,

```
>> save filename
```

とすると, filename.mat というファイルに保存される。
ファイル名を省略すると matlab.mat に保存される。

保存された情報は,

```
>> load filename
```

として、後から読み込んで使うことができる。

どんな変数が定義されているかを調べるには whos を入力する。
すべての変数を消去するには clear を、
特定の変数 aha を削除するには clear aha を入力する。

資料など

■長藤かおり: MATLAB / INTLAB によるプログラミングの基礎（数値解析チュートリアル 2008 —九州大学21世紀COEプログラム—）

をもとに作成しました。多謝！

参考文献

■櫻井 鉄也： MATLAB/Scilab で理解する数値計算 （東京大学出版会）

■大石 進一： MATLABによる数値計算 （培風館）

■ http://www.slis.tsukuba.ac.jp/~hasegawa/MATLAB/numerics.html

■ http://www.mathworks.com/

■ http://www.cybenet.co.jp/matlab/