高機能数値計算・可視化機能ソフト MATLABの基本的な使い方

秋田県立大学　陳 国躍
東北大学サイバーサイエンスセンター

MATLAB

- MATLAB はMATrix LABoratory の略
- 高機能な数値計算
  - 特に行列演算
- 多彩な可視化機能
  - 計算結果を簡単にグラフ表示

高機能な数値計算機能

- 変数宣言不要
- 対話型の実行
- 豊富な関数ライブラリ
- 配列が基本的データ要素

多彩な可視化機能

- 豊富なグラフィックスツール

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

MATLAB の Toolbox（1）

- MATLAB
- Simulink
- Curve Fitting Toolbox：数式近似ツール
- Communications System Toolbox：通信システムの設計、解析
- MATLAB Compiler：MATLAB アプリケーションを実行ファイル、共有ライブラリとして生成
- Control System Toolbox：線形制御システムの設計、解析
- DSP System Toolbox：信号処理システムの設計とシミュレーション
- Fuzzy Logic Toolbox：ファジー理論に基づくシステムの設計、解析
- System Identification Toolbox：測定された入出力データから動的システムの数学モデルを構築
- Image Processing Toolbox：画像処理、解析、可視化およびアルゴリズム開発のためのグラフィカル ツール

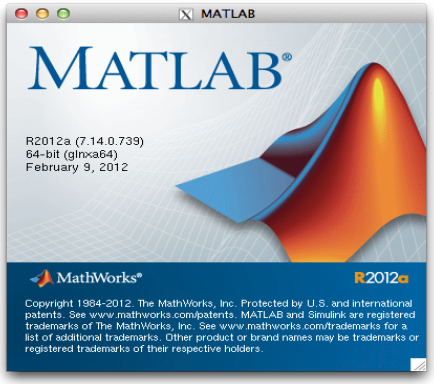
MATLAB の Toolbox（2）

- MATLAB Coder：MATLABコードからスタンドアロンの C コードと C++ コードを生成
- Model Predictive Control Toolbox：予測制御システムの設計、解析
- Neural Network Toolbox：ニューラルネットワーク理論を用いた複雑非線形事象のモデリング
- Optimization Toolbox：標準的および大規模な最適化に広く使用されるアルゴリズムを提供
- Partial Differential Equation Toolbox：偏微分方程式の解法ツール
- Fixed-Point Toolbox：固定小数点機能が利用可能
- Robust Control Toolbox：ロバスト性の評価、解析
- Simulink Coder：Simulinkダイアグラム、および MATLAB関数から C コードと C++ コードを生成

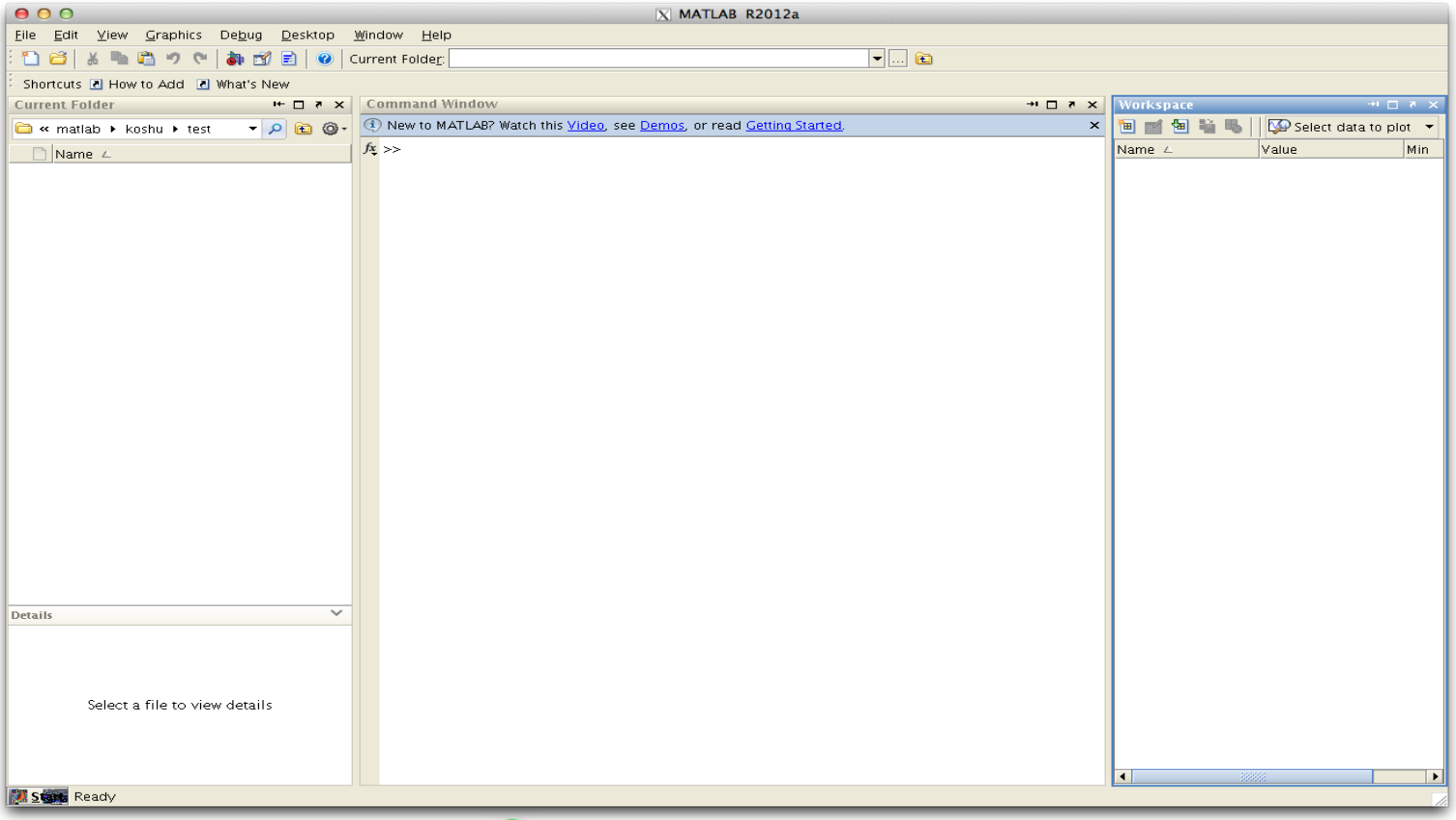
MATLAB の Toolbox（3）

- Simulink Control Design：Simulinkでモデル化された制御システムの設計と解析
- Signal Processing Toolbox：アナログおよびデジタル信号処理 (DSP) のアルゴリズムを提供
- Symbolic Math Toolbox：数式処理と可変精度演算を行うためのツールを提供
- Simulink Design Optimization：Simulinkでモデル化された制御システムの数値最適化
- Statistics Toolbox：データの整理、解析、およびモデリングのためのアルゴリズムとツールを提供
- Simulink Verification and Validation：Simulinkでモデル化された制御システムの確認、検証およびテスト
- Wavelet Toolbox：ウェーブレット変換を用いた画像処理、解析、可視化ツール

MATLAB の起動



- 起動直後のウィンドウ



MATLAB の終了

- >> は　MATLAB の入力促進記号
  （プロンプト）

- コマンド　quit　あるいは　exit　を入力

  >> quit　<R>

  >> exit　<R>

<R>　はリターン（改行）の意味

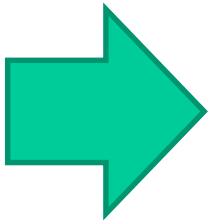
数値計算の例

そのまま入力しても簡単に演算できる

(1) $\dfrac{5(3+4)}{2}$

(2) $\dfrac{(x^2+y^5)}{\sin(z)}$

(初期値 $x=3$, $y=2$, $z=1.5$)



```
>> 5*(3+4)/2          <R>
ans =
     17.5000
>> x = 3,     y = 2,     z = 1.5     <R>
X =
     3
y =
     2
Z =
     1.5000
>> (x^2+y^5)/sin(z)    <R>
ans =
     41.1030
```

ベクトル・行列の入力

$$\begin{pmatrix} 1 & 3 & 2 \\ 5 & 1.2 & 3.4 \\ 1.5 & 4 & 5.1 \end{pmatrix} \begin{pmatrix} x1 \\ x2 \\ x3 \end{pmatrix} = \begin{pmatrix} 6.1 \\ 4 \\ 5.8 \end{pmatrix}$$

$$A\,X = B$$

```
>> A = [1 3 2; 5 1.2 3.4; 1.5 4 5.1]      <R>
A =
    1.0000    3.0000    2.0000
    5.0000    1.2000    3.4000
    1.5000    4.0000    5.1000

>> A2 = [1 3 2     <R>
5 1.2 3.4           <R>
1.5 4 5.1]          <R>
A2 =
    1.0000    3.0000    2.0000
    5.0000    1.2000    3.4000
    1.5000    4.0000    5.1000
```

```
>> C=[6.1   4   5.8]      <R>
C =
    6.1000        4.0000        5.8000
>> B = C'          <R>
B =
    6.1000
    4.0000
    5.8000
>> B2 = [ 6.1      <R>
4                   <R>
5.8]                <R>
B2 =
    6.1000
    4.0000
    5.8000
```

ベクトル・行列演算の例

数学的な記述に近い形で演算できる MATLAB の大きな特徴の一つ

$$X = inv(A)\,B$$

```
>> X = inv(A)*B  <R>
X =
      0.9166
      2.4089
     -1.0217
>> X = A \ B      <R>
X =
      0.9166
      2.4089
      -1.0217
```

```
>> max(X)        <R>
ans =
      2.4089
>> mean(X)       <R>
ans =
      0.7679
>> n=length(X)   <R>
n =
      3
>> x2 = X(2)     <R>
x2 =
      2.4089
```

```
>> eig( A )      <R>
ans =
     9.0284
    -2.9432
     1.2148
```

固有値

最大値

平均値

配列のサイズ

配列の第2要素

複素数

虚数表現として、i と j のどちらも利用可能

$$Z1 = 3 + 4i$$

$$Z2 = 1 - 7i$$

```
>> z1 = 3+4*i
z1 =
    3.0000 + 4.0000i
>> z2 = 1-7*j
z2 =
    1.0000 - 7.0000i
>> z1+z2
ans =
    4.0000 - 3.0000i
>> z1*z2
ans =
    31.0000 - 17.0000i
>> z1 / z2
ans =
    -0.5000 + 0.5000i
```

複素数ベクトル・行列

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + i\begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$$

$$B = \begin{pmatrix} 3 & -2 \\ -4 & 8 \end{pmatrix} + i\begin{pmatrix} 7 & -3 \\ 1 & -2 \end{pmatrix}$$

```
>> A = [ 1 2 ; 3 4 ] + i*[ 5 6 ; 7 8]
A =
    1.0000 + 5.0000i    2.0000 + 6.0000i
    3.0000 + 7.0000i    4.0000 + 8.0000i
>> B = [ 3+7i   -2-3i  ;   -4+1j   8-2j ]
B =
    3.0000 + 7.0000i   -2.0000 - 3.0000i
   -4.0000 + 1.0000i    8.0000 - 2.0000i
>> eig ( A*B )
ans =
   -0.7101 - 4.5909i
   17.7101 +37.5909i
```

複素数演算の例

```
>> eig(A*B)
ans =
    -0.7101 – 4.5909i
    17.7101 +37.5909i
>>real(eig(A*B))
ans =
    -0.7101
    17.7101
>> imag(eig(A*B))
ans =
    -4.5909
    37.5909
>> abs(eig(A*B))
ans =
    4.6455
   41.5539
```

⟸ 二つの固有値

⟸ 複素数の実数部

⟸ 複素数の虚数部

⟸ 複素数の絶対値

ファイルとデータの入出力

1. sam1.datというファイルの内容
   (次のデータが一行に書かれていることに注意):

   0.0 0.3 0.95 -0.4 0.4 0.2 -0.3 0.0 0.3 0.0 -0.2 0.1 0.0 0.0 0.0 0.0

2. sam2.datというファイルの内容
   (次のデータが7行に書かれていることに注意):

   0
   3
   9
   5
   -4
   2
   -3

3. sam3.dat というファイルの内容(5行3列のデータ):

   1.1 1.2 1.3
   2.1 2.2 2.3
   3.1 3.2 3.3
   4.1 4.2 4.2
   5.1 5.2 5.3

ファイルとデータの入出力

```
>> load sam1.dat
>> sam1
sam1 =
   Columns 1 through 7
         0    0.3000    0.9500   -0.4000    0.4000    0.2000   -0.3000
   Columns 8 through 14
         0    0.3000         0   -0.2000    0.1000         0         0
   Columns 15 through 16
         0         0
>>load sam2.dat
>>sam2
sam2 =
     0
     3
     9
     5
    -4
     2
    -3
     0
>> load sam3.dat
>> sam3
sam3 =
    1.1000    1.2000    1.3000
    2.1000    2.2000    2.3000
    3.1000    3.2000    3.3000
    4.1000    4.2000    4.3000
    5.1000    5.2000    5.3000
```

スクリプトファイル（　.m ファイル）

Sample.m というファイルの内容

```
load sam1.dat
sam1
load sam2.dat
sam2
load sam3.dat
sam3
```

MATLAB上の実行は “sample<R>”

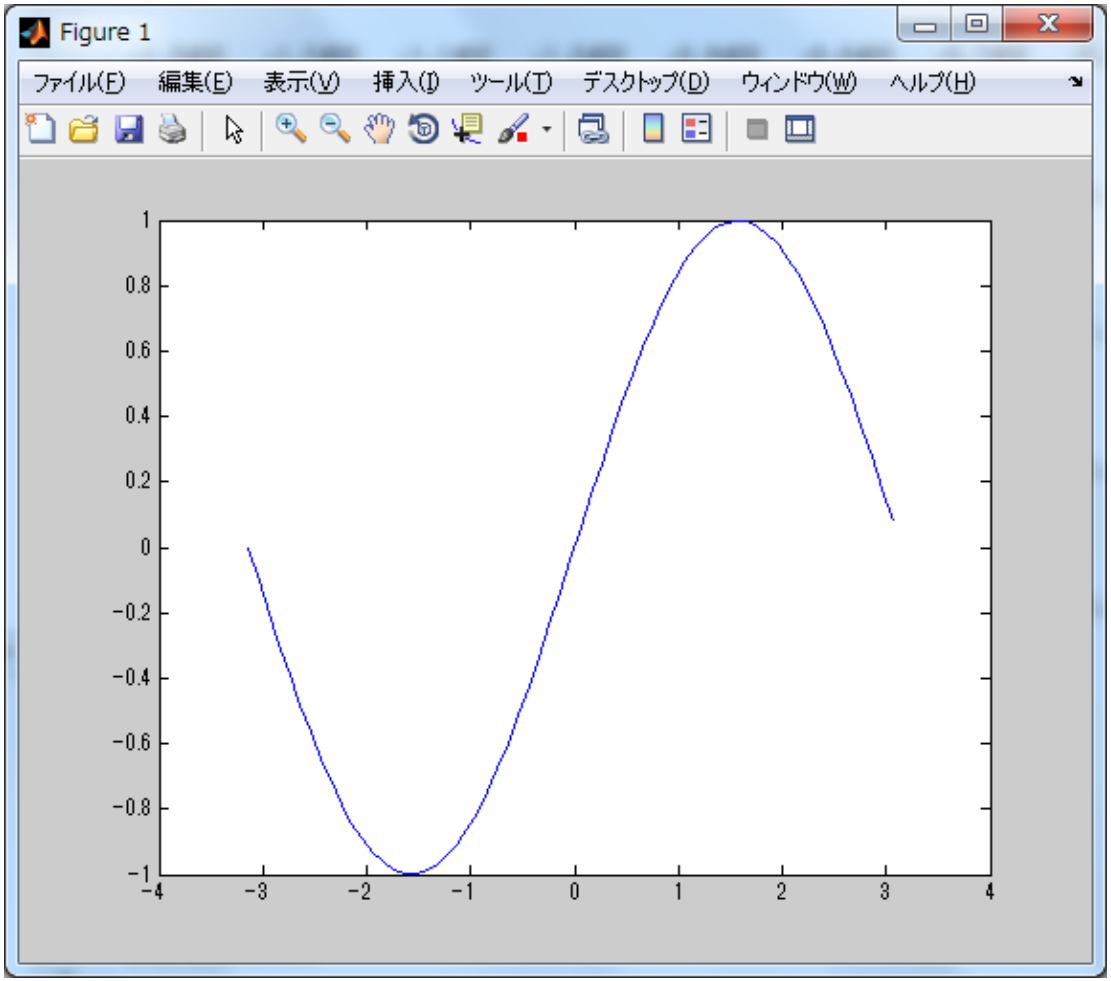
```
>>  sample<R>
```

2次元グラフィックス（plot 関数）

表示	説明
plot(y)	y の配列番号を横軸として y のグラフを描く
plot(x,y)	x を横軸として y のグラフを描く
plot(x1,y1,x2,y2,・・・)	(xn,yn) の組み合わせで複数のグラフを描く
plot(x,y,LineSpec)	LineSpec で指定された線種、マーカ、色でグラフを描く
plot(x,y,'LineSpec',lw)	'LineWidth', lwによって、lw で指定された線幅を描く

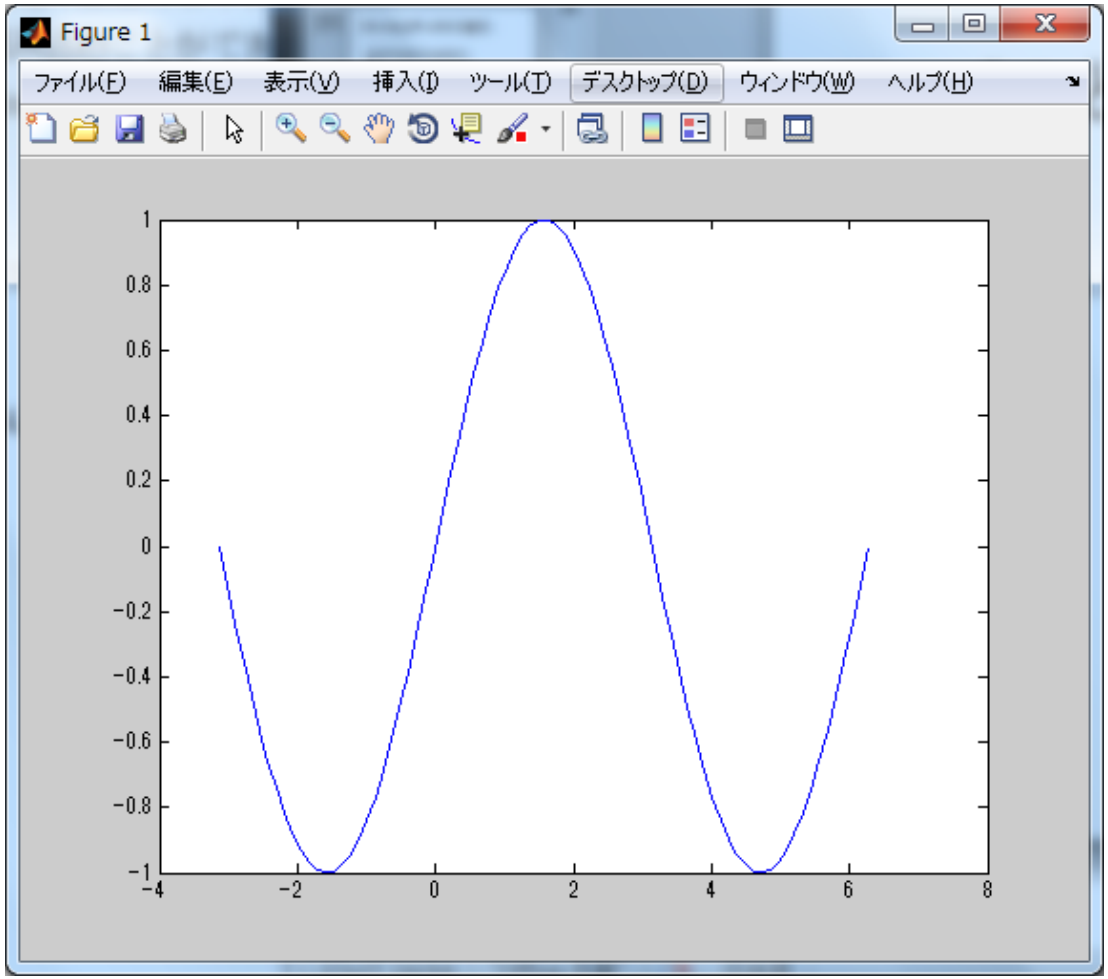
plot 関数の例(1)

```
x =[-3.14 : 0.1 : 3.14]
plot(x,sin(x))
```



plot 関数の例(2)

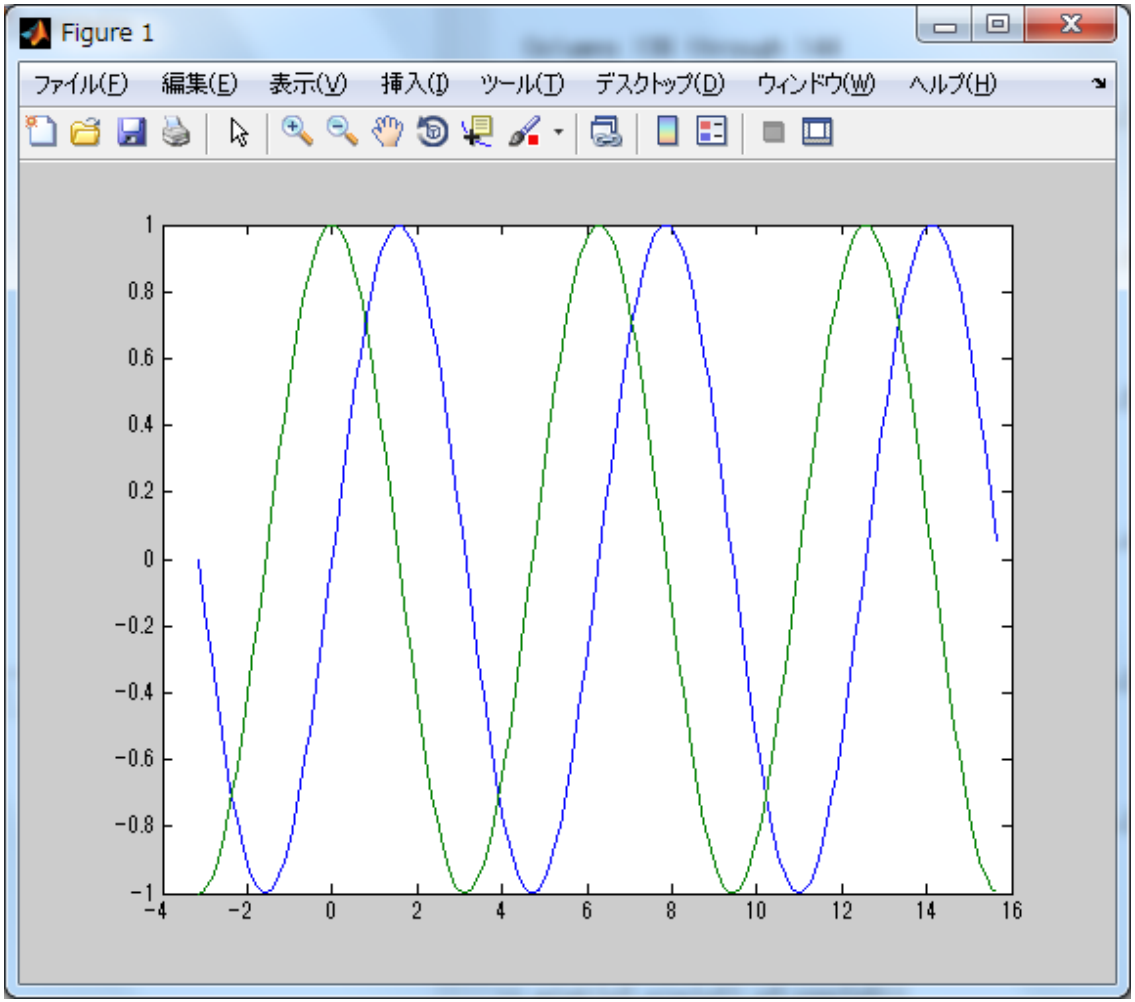
```
x1=linspace(-3.14,6.28,80)
plot(x1,sin(x1))
```



plot 関数の例(3)

```
x2=[-pi:0.1:5*pi]

plot(x2,sin(x2),x2,con(x2))
```



グラフの装飾(1)

plot(x, y, 'linetype')

'linetype' を省略した場合、自動的に青色の実線

ラインのタイプと色

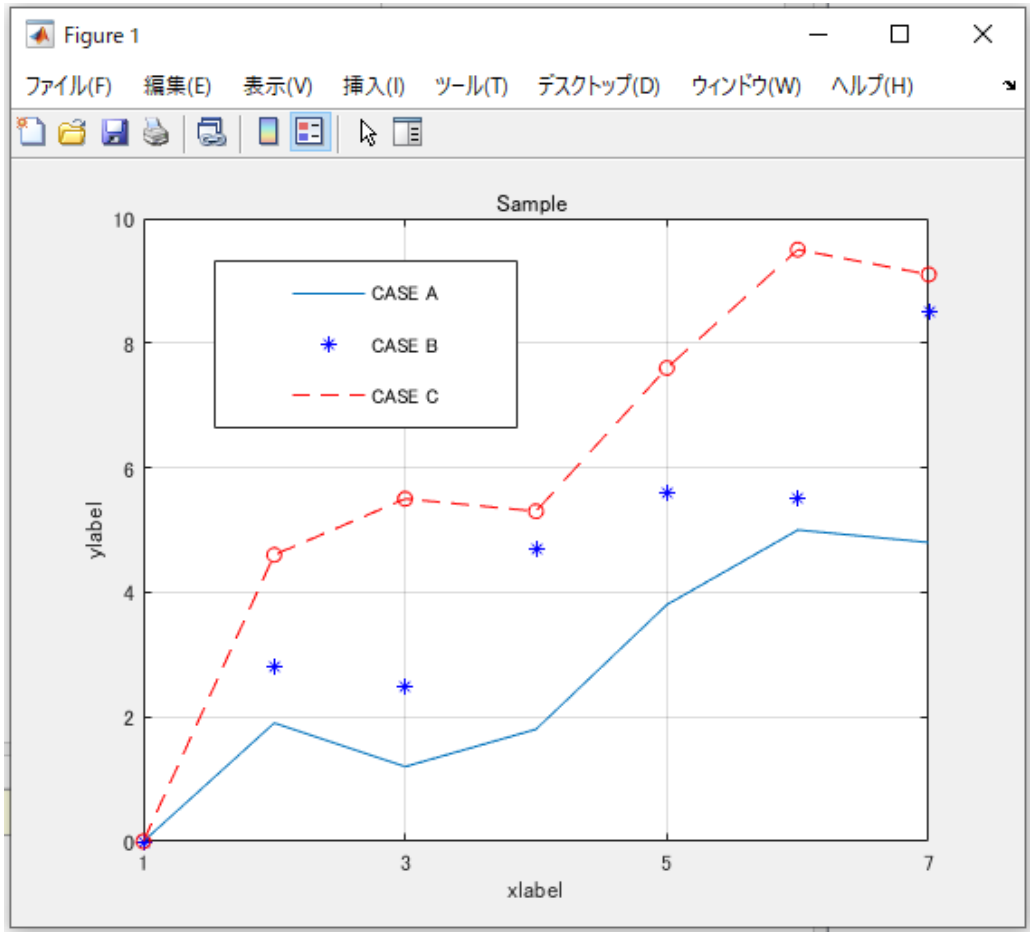
シンボル	ラインタイプ	シンボル	ラインタイプ	シンボル	色
-	実線	.	点	b	青
:	点線	o	円	g	緑
-.	鎖線	x	x 印	r	赤
- -	破線	+	プラス記号	c	シアン
(none)	線なし	*	星印	m	マゼンタ
		s	正方形	y	黄
		d	ダイアモンド	w	白
		v	三角形（上向き）	k	黒
		^	三角形（下向き）		
		<	三角形（左向き）		
		>	三角形（右向き）		
		p	五角形		
		h	六角形		

グラフの装飾(2)

title	タイトル	text	文字列表示
xlabel	x軸ラベル	gtext	マウス指定による文字列表示
ylabel	y軸ラベル	legend	凡例
zlabel	z軸ラベル	grid	グリッドライン

plot 関数の例(4)

```
C1 = [ 0.0 1.9 1.2 1.8 3.8 5.0 4.8 ];
C2 = [ 0.0 2.8 2.5 4.7 5.6 5.5 8.5 ];
C3 = [ 0.0 4.6 5.5 5.3 7.6 9.5 9.1 ];
plot(C1)
hold on
plot(C2, 'b*')
plot(C3, 'r--')
plot(C3, 'ro')
hold off
axis([1  7  0 10])
set(gca,'XTick',[1:2:7])
set(gca,'YTick',[0:2:10])
xlabel('xlabel');
ylabel('ylabel');
title('Sample ');
grid
h=legend('CASE A','CASE B','CASE C');
set(h,'Position',[0.2,0.65,0.3,0.22]);
%print  -depsc  fig4
```



３次元グラフィックス関数

mesh(Z)	3次元プロットの基本
mesh(X,Y,Z)	
meshc(Z)	meshプロットの底面にコンター・プロット
meshc(X,Y,Z)	
surf(Z)	表面プロットの基本
surf(X,Y,Z)	
surfc(Z)	surfプロットの底面にコンター・プロット
surfc(X,Y,Z)	

peaks 多峰型関数

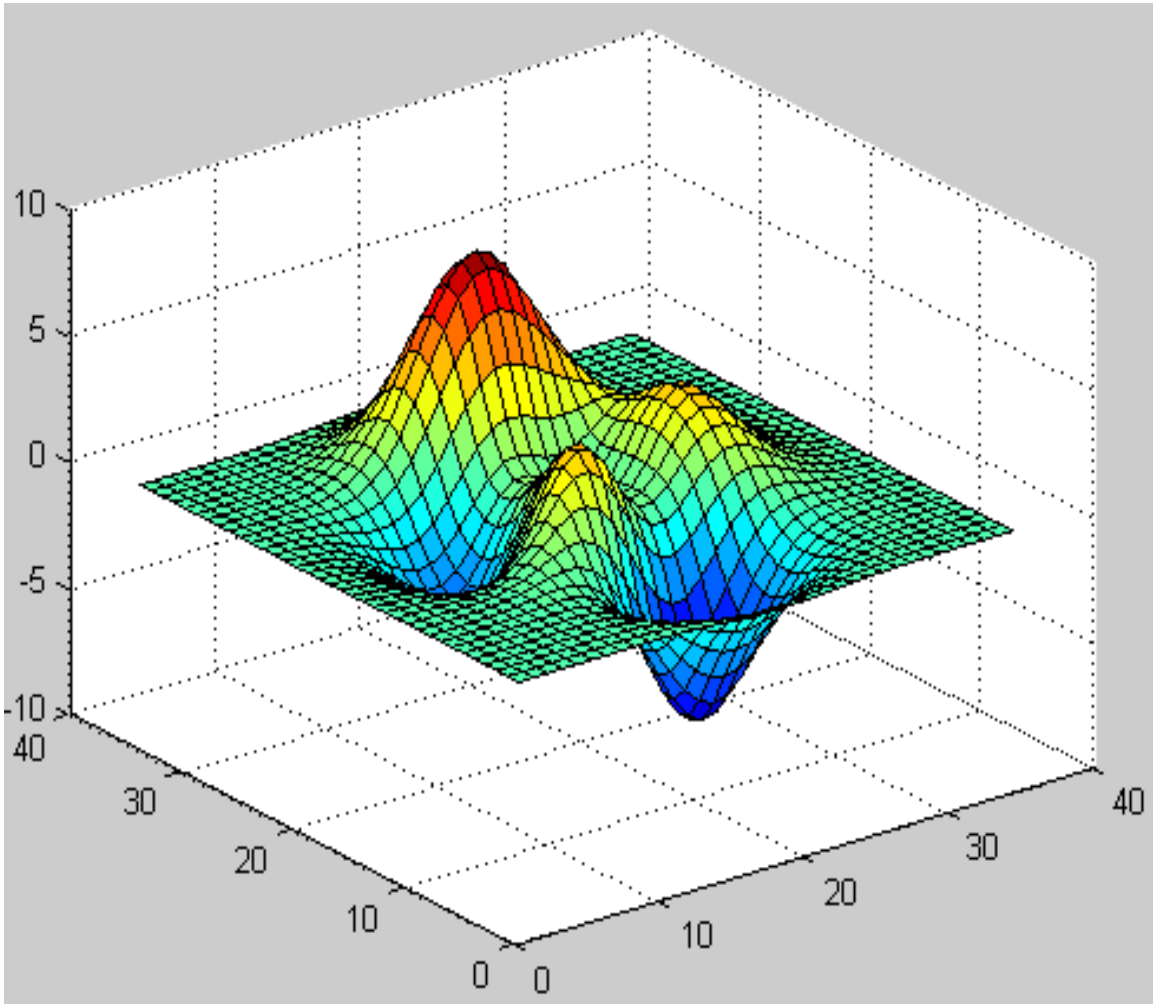
$$f(x,y) = 3(1-x)^2 e^{-x^2-(y+1)^2} - 10\left(\frac{x}{5} - x^3 - y^5\right)e^{-x^2-y^2} - \frac{1}{3}e^{-(x+1)^2-y^2}$$

- peaks(n), 引数のnを省くと、標準値49
- mesh, surf , pcolor 等のデモに役立

```
>> z = peaks(7)
z =
    0.0001    0.0034   -0.0299   -0.2450   -0.1100   -0.0043    0.0000
    0.0007    0.0468   -0.5921   -4.7596   -2.1024   -0.0616    0.0004
   -0.0088   -0.1301    1.8559   -0.7239   -0.2729    0.4996    0.0130
   -0.0365   -1.3327   -1.6523    0.9810    2.9369    1.4122    0.0331
   -0.0137   -0.4808    0.2289    3.6886    2.4338    0.5805    0.0125
    0.0000    0.0797    2.0967    5.8591    2.2099    0.1328    0.0013
    0.0000    0.0053    0.1099    0.2999    0.1107    0.0057    0.0000
```

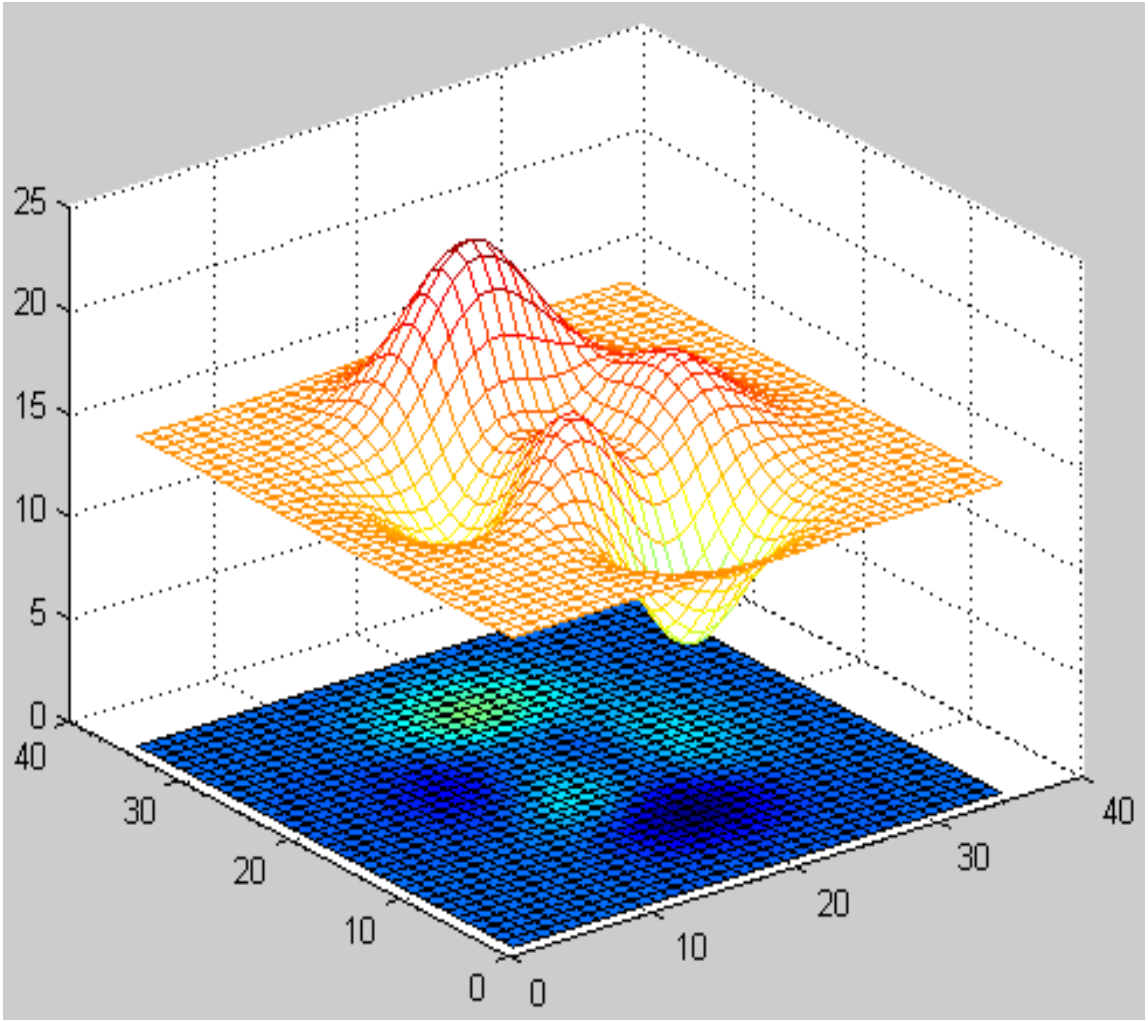
3次元グラフィックス関数

```
surf(peaks(35))
print -depsc fig5
```



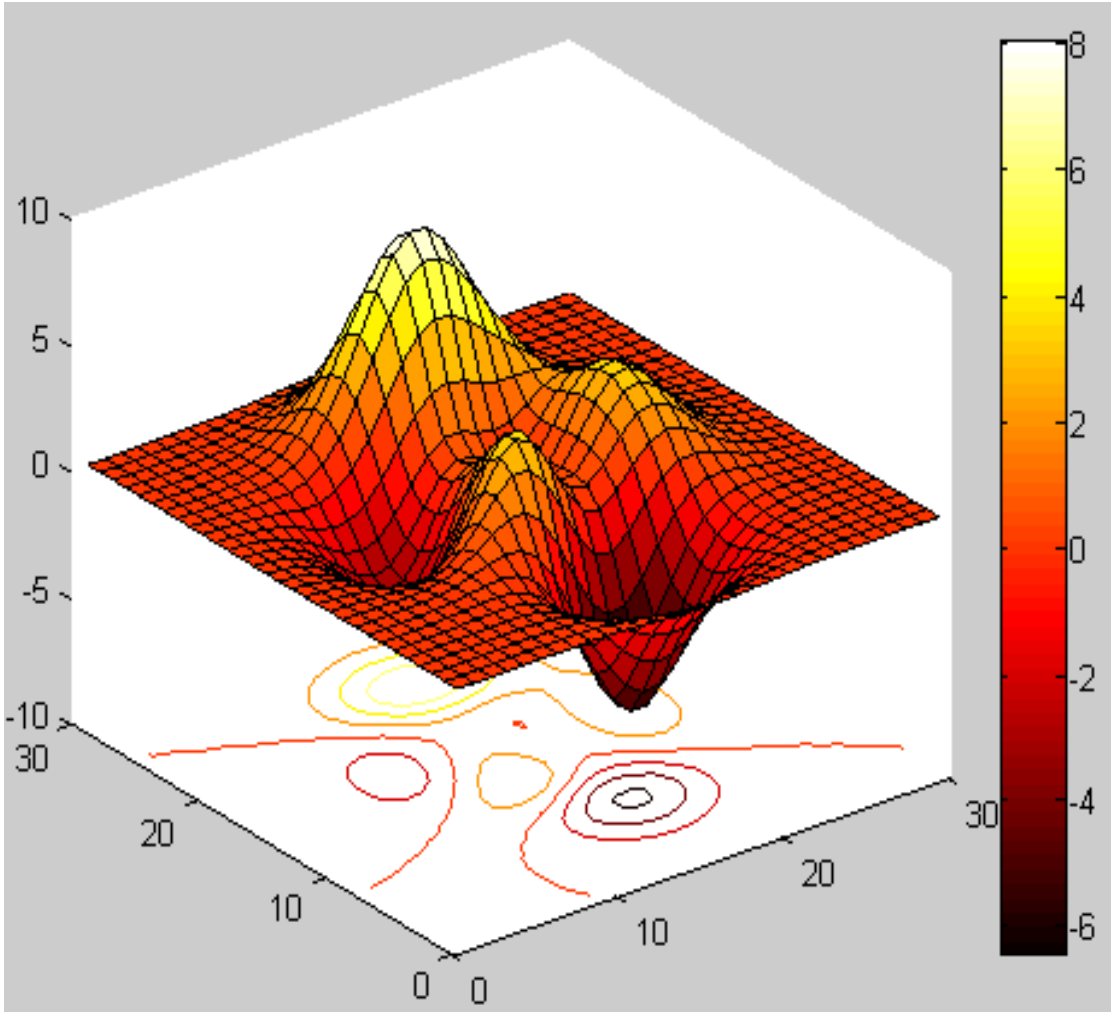
3次元グラフィックス例(mesh)

```
mesh(peaks(35)+15)
hold
pcolor(peaks(35))
hold off
print -depsc fig6
```



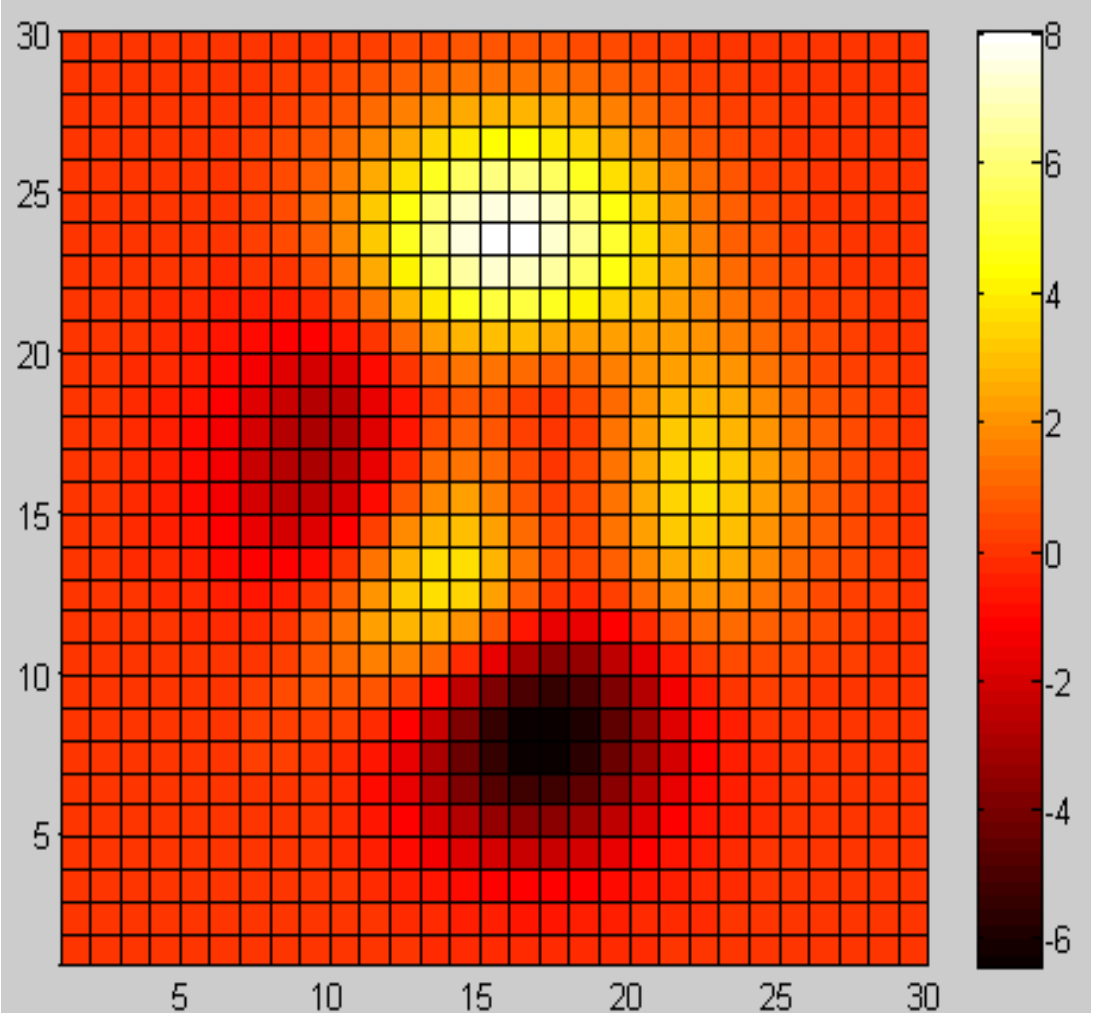
3次元グラフィックス例(surfc)

```
surfc(peaks(30))
colormap(hot)
colorbar('vert')
grid
print -depsc fig7
```



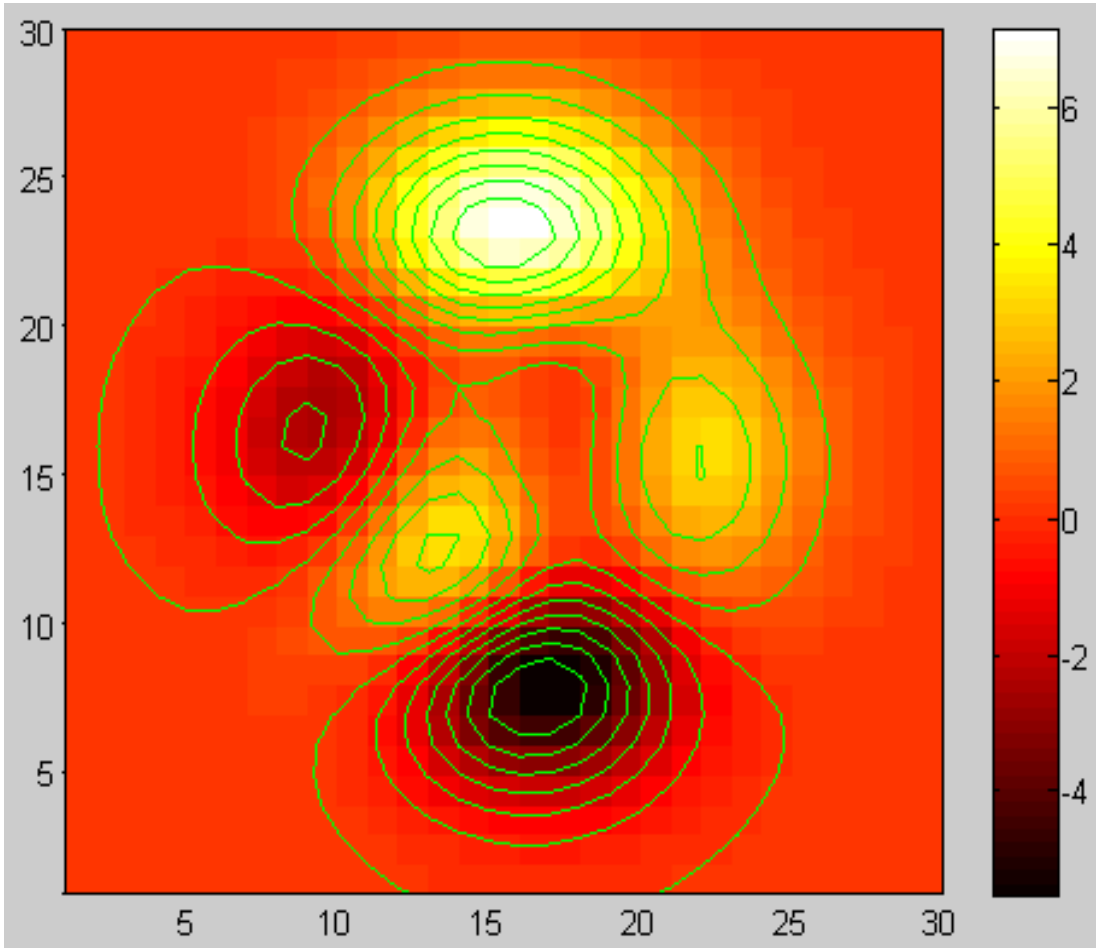
2次元グラフィックス例(pcolor)

```
pcolor(peaks(30))
colormap(hot)
colorbar('vert')
print -depsc fig8
```



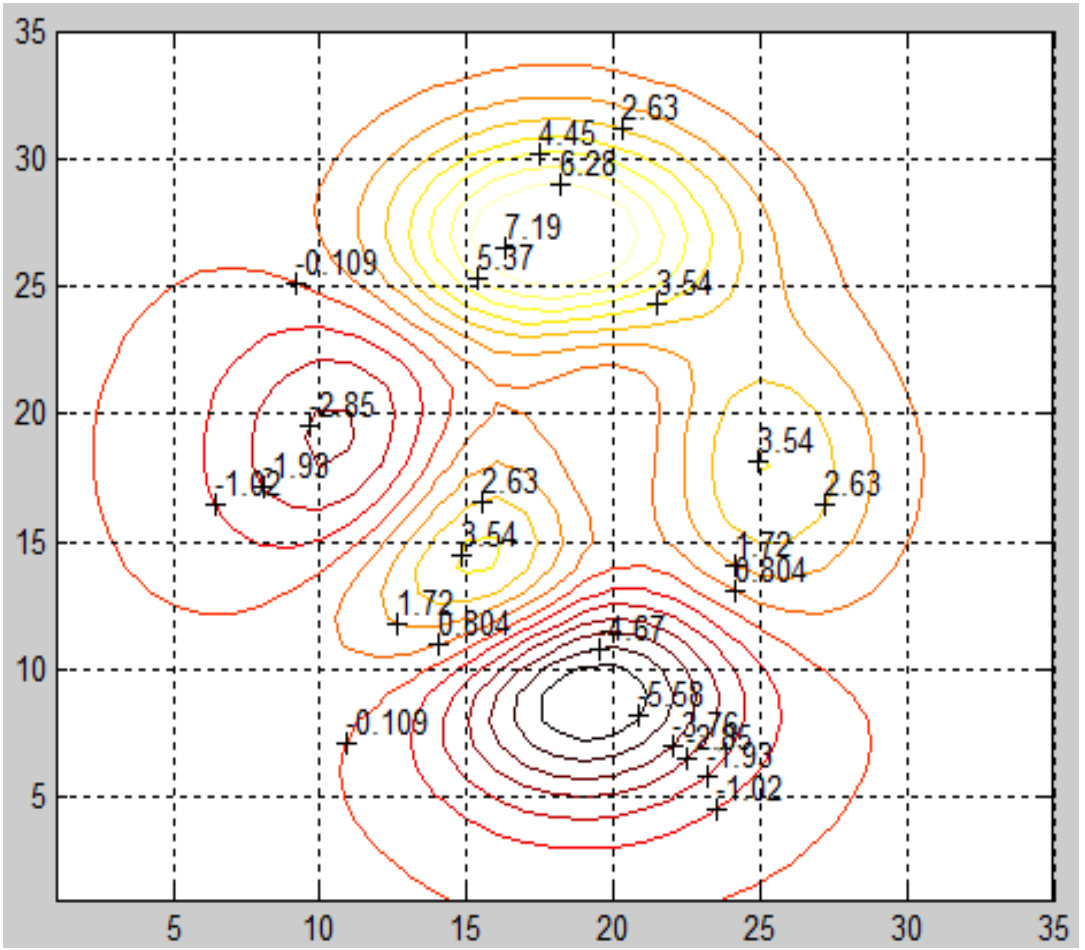
2次元グラフィックス例（pcolor, contour）

```
pcolor(peaks(30))
colormap(hot)
shading flat
hold on
contour(peaks(30),15,'g')
hold off
colorbar('vert')
print -depsc fig9
```



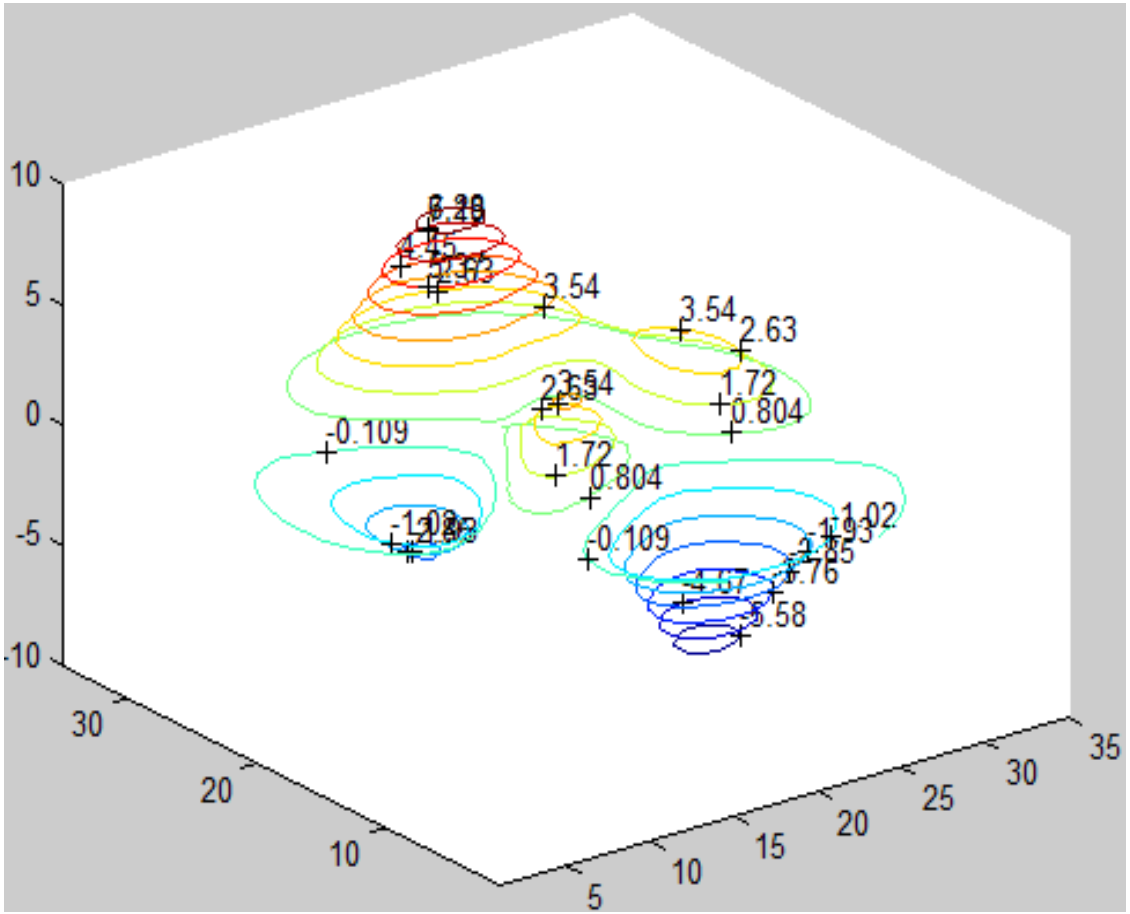
2次元グラフィックス例(surf)

```
b=contour(peaks(35),15);
clabel(b)
grid
print   -depsc   fig10
```



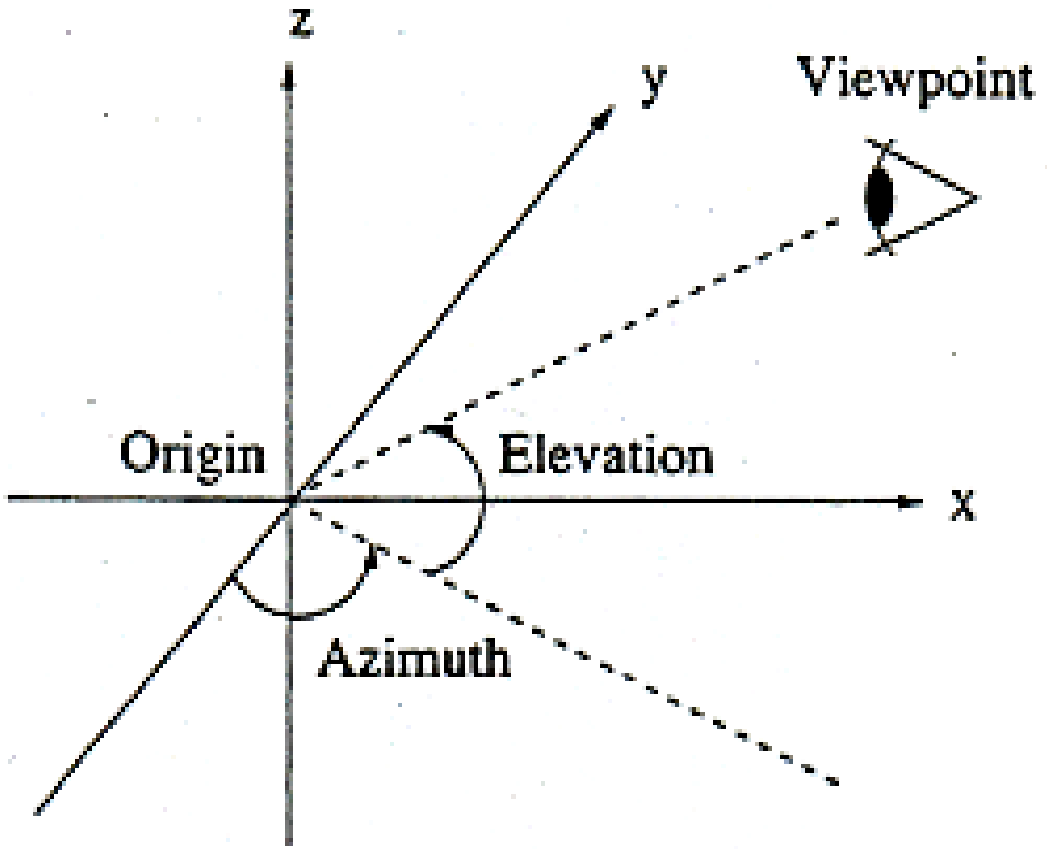
3次元グラフィックス例(contour3)

```
b=contour3(peaks(35),15);
clabel(b)
grid
print  -depsc  fig11
```



3次元プロットの視点角度の設定

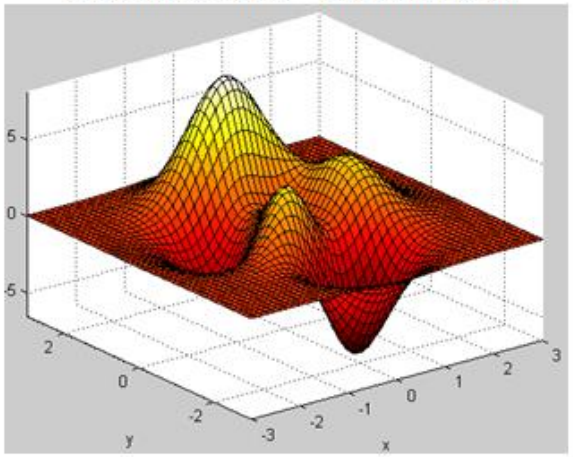
view(Azimuth, Elevation)



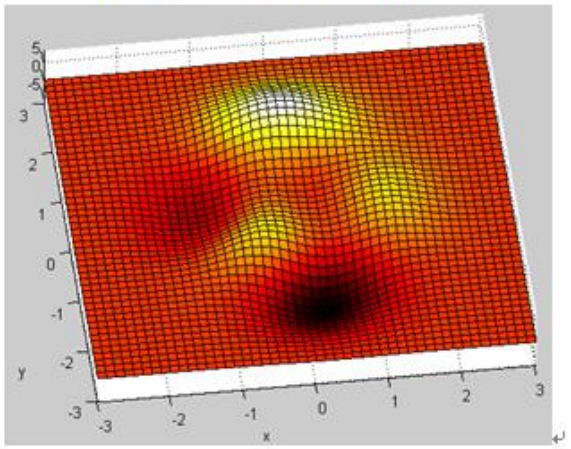
視点角度の設定例

view(-37.5,30)　view(-7,80)
view(-90,0)　view(-7,-10)

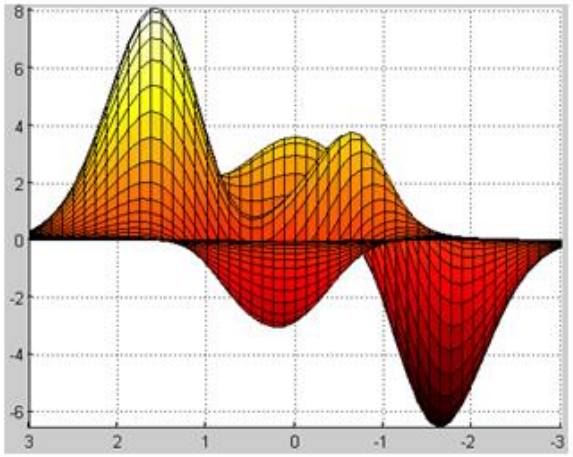
Azimuth=-37.5　Elevation=30



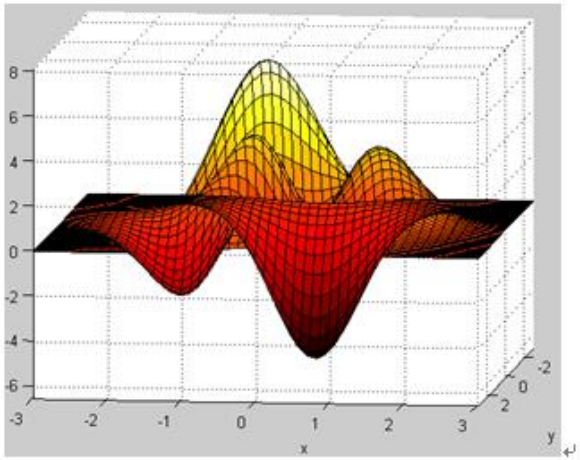
Azimuth=-7　Elevation=80



Azimuth=-90　Elevation=0



Azimuth=-7　Elevation=-10



グラフィックスの出力

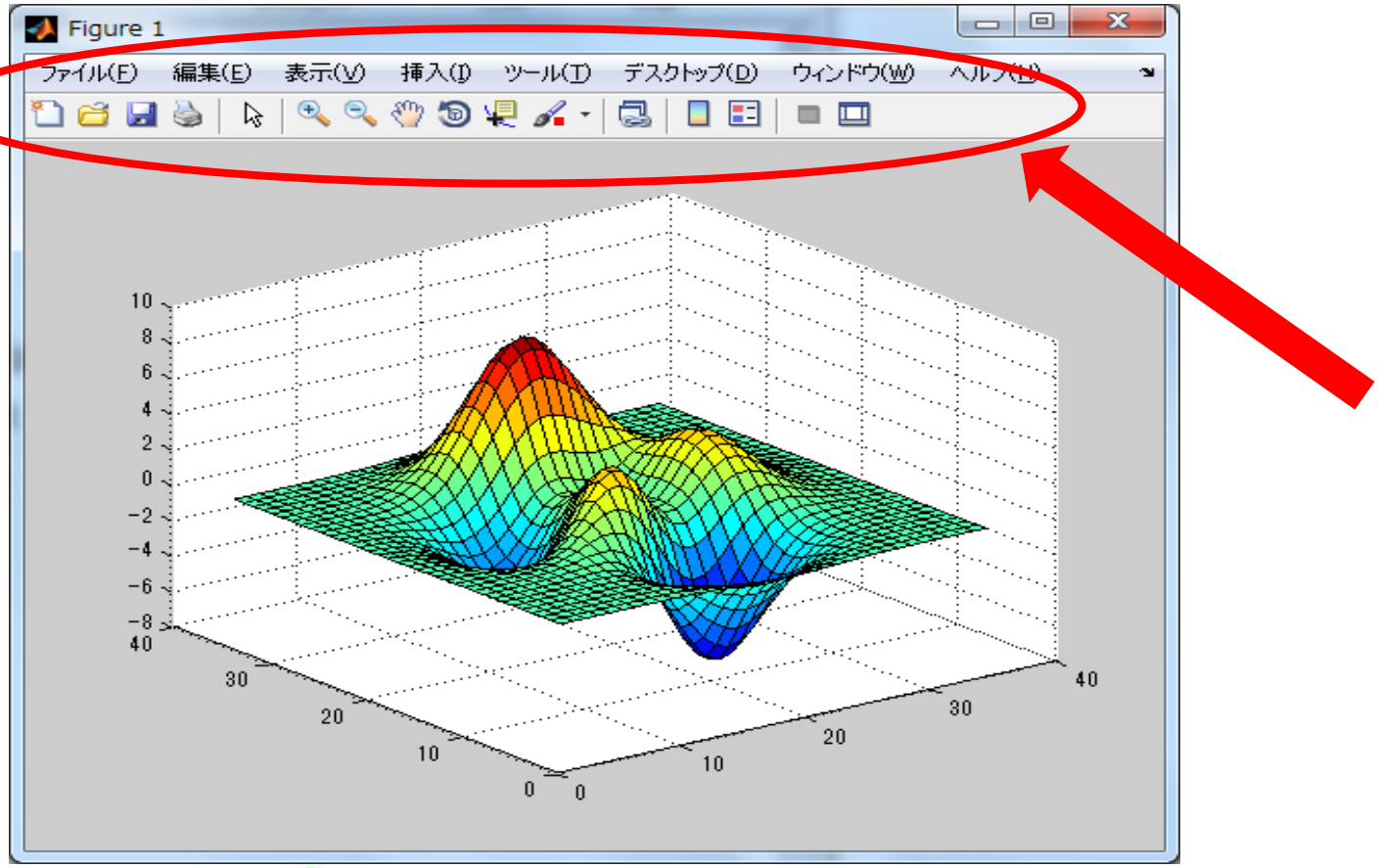
print [-option] filename

option:

dps:	白黒	ps	ファイル
dpsc:	カラー	ps	ファイル
deps:	白黒	eps	ファイル
depsc:	カラー	eps	ファイル

出力されたグラフの例

グラフを表示した後でも図の編集が可能



補足説明

一部コマンドの紹介

配列

- ベクトル・・・・・・・・ 1 次元配列
- 行列・・・・・・・・・・ 2 次元配列

- 配列の添え字は 1 から始まる

配列要素の指定

- ベクトル・ 1 次元配列 X の第 k 要素は X(k)
- 行列・・・2 次元配列 A の i 行 j 列要素は A(i , j)

ベクトルの作成方法（1）

行ベクトル作成

行ベクトルの右肩に‘（転置）で列ベクトル作成

セミコロン　；　は行の区切りで、列ベクトル作成

列ベクトルの右肩に‘（転置）で行ベクトル作成

```
>> X=[2 4 6 8]

X =

     2     4     6     8

>> X'

ans =

     2
     4
     6
     8

>> Y=[3; 6; 9]

Y =

     3
     6
     9

>> Y'

ans =

     3     6     9
```

ベクトルの作成方法 (2)

初期値，増分，最終値をコロンで区切って指定

1 から 30 まで 5 刻みで作成

50 から 1 まで -10 刻みで作成

増分が 1 の時は省略可

```
>> X1=[1:5:30]

X1 =

     1     6    11    16    21    26

>> X2=[50:-10:1]

X2 =

    50    40    30    20    10

>> X3=[1:8]

X3 =

     1     2     3     4     5     6     7     8
```

関数 linspace を使って，初期値，最終値，要素の数を指定

-1 から 1 まで等間隔で 5 要素作成

```
>> X4=linspace(-1,1,5)

X4 =

   -1.0000   -0.5000         0    0.5000    1.0000
```

行列の結合と縮小

```
>> A=[11 12 13; 21 22 23; 31 32 33]

A =

    11    12    13
    21    22    23
    31    32    33

>> B=[4 5
6 7
8 9]

B =

     4     5
     6     7
     8     9

>> C=[10 10 10]

C =

    10    10    10
```

```
>> AB=[A,B]

AB =

    11    12    13     4     5
    21    22    23     6     7
    31    32    33     8     9

>> AC=[A; C]

AC =

    11    12    13
    21    22    23
    31    32    33
    10    10    10
```

```
>> AC2=AC(2:3,2:3)

AC2 =

    22    23
    32    33

>> AC2=AC(3:4,:)

AC2 =

    31    32    33
    10    10    10

>> AC3=AC(:,1)

AC3 =

    11
    21
    31
    10
```

- 継続行：行末に3つのピリオド(…)を書くと継続できる
- ans ： 直前のコマンドの答え
- 最後にセミコロン (;) があった場合，結果を非表示にする

```
>> A5=[ 11 12 13 14 15; 21 22 23 24 25; ...
31 32 33 34 35; 41 42 43 44 45; ...
51 52 53 54 55]

A5 =

    11    12    13    14    15
    21    22    23    24    25
    31    32    33    34    35
    41    42    43    44    45
    51    52    53    54    55

>> A5'

ans =

    11    21    31    41    51
    12    22    32    42    52
    13    23    33    43    53
    14    24    34    44    54
    15    25    35    45    55
```

```
>> A6=A5+100

A6 =

   111   112   113   114   115
   121   122   123   124   125
   131   132   133   134   135
   141   142   143   144   145
   151   152   153   154   155

>> A7=A5+100;
>> A7=A5+100

A7 =

   111   112   113   114   115
   121   122   123   124   125
   131   132   133   134   135
   141   142   143   144   145
   151   152   153   154   155
```

一般行列関数

関数名など	機能	記述
:	等間隔ベクトル	x=m1:m2 x=m1:a:m2
linspace	線形に等間隔のベクトル	x=linspace(m1,m2,n) (初期値m1, 最終値m2, 個数n)
zeros	すべての要素が0の行列	X=zeros(m,n) X=zeros(m)　(mxmの正方行列)
ones	すべての要素が1の行列	同　上
eye	単位行列	同　上
rand	0と1の間の一様乱数行列	同　上
randn	平均値が0、標準偏差が1の正規乱数行列	同　上
diag	対角行列	X=diag([x1,x2,・・・,xn]) ([x1,x2,・・・,xn]は対角要素ベクトル)

一般行列関数(例)

```
>> zeros(5)

ans =

     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0

>> A3=ones(3,4)

A3 =

     1     1     1     1
     1     1     1     1
     1     1     1     1

>> A5=eye(3)

A5 =

     1     0     0
     0     1     0
     0     0     1
```

```
>> rand(2,4)

ans =

    0.2111    0.0994    0.8387    0.7514
    0.1511    0.1808    0.3381    0.9033

>> randn(5,3)

ans =

    0.0233   -0.8081   -1.0146
   -0.2674   -2.1329    0.6051
   -0.3428    0.0166    0.5789
   -0.2026    1.0177   -1.9378
    2.3891   -0.4801    0.4474

>> diag([2 5 8])

ans =

     2     0     0
     0     5     0
     0     0     8
```

各要素ごとの演算

.*	乗算
./	割算
.^	べき乗

```
>> A=[1 2; 3 4]

A =

     1     2
     3     4

>> B=[5 6; 7 8]

B =

     5     6
     7     8
```

```
>> A*B

ans =

    19    22
    43    50

>> A.*B

ans =

     5    12
    21    32
```

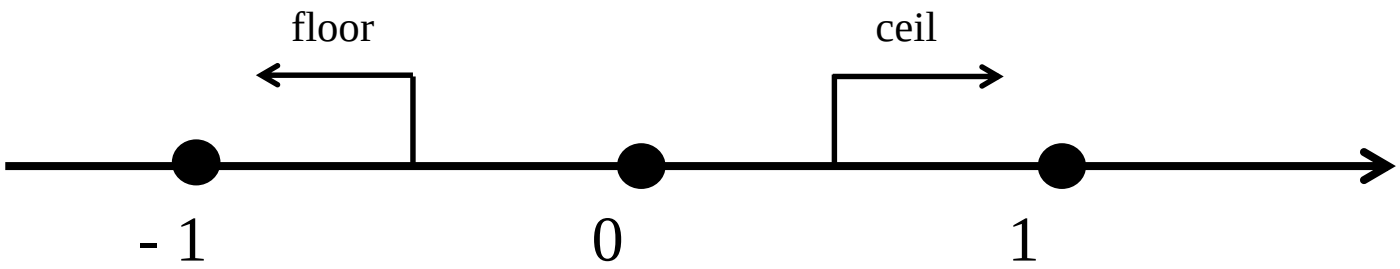
行列積

各要素ごとの積

結果が異なる

小数点、4種類の「丸め」

round	四捨五入
fix	切り捨て
ceil	＋無限大の方へ切り上げ
floor	－無限大の方へ切り上げ



バイナリ･ファイルの保存と読み出し

fopen	ファイルのオープン
fclose	ファイルのクローズ
fwrite	バイナリ･データの保存
fread	バイナリ･データの読み込み

C と Fortran のようにバイナリ･データの保存と読み出しが可能

予約定数・変数

ans	直前のコマンドの答
eps	浮動小数点の相対精度
flops	浮動小数点演算回数
i, j	虚数単位
inf	（無限大）
NaN	Not-a-number（不定値）
pi	π (3.1416)
realmax	最大の浮動少数
realmin	最小の浮動少数

```
>> pi

ans =

    3.1416

>> 5+8i

ans =

   5.0000 + 8.0000i

>> 3-6j

ans =

   3.0000 - 6.0000i

>> x=0/0
警告: ゼロ割です

x =

   NaN

>> y=pi/0
警告: ゼロ割です

y =

   Inf
```

2次元グラフィックス関数

bar	棒グラフプロット
errorbar	エラーバーのプロット
stem	離散データ列のプロット
hist	ヒストグラム
polar	極座標プロット

分割プロット

subplot(m,n,p)
Figureウインドウを　m × n　に分割し、p で指定する位置

1	2	3
4	5	6

分割プロットの解除には subplot(1,1,1) を実行する

例

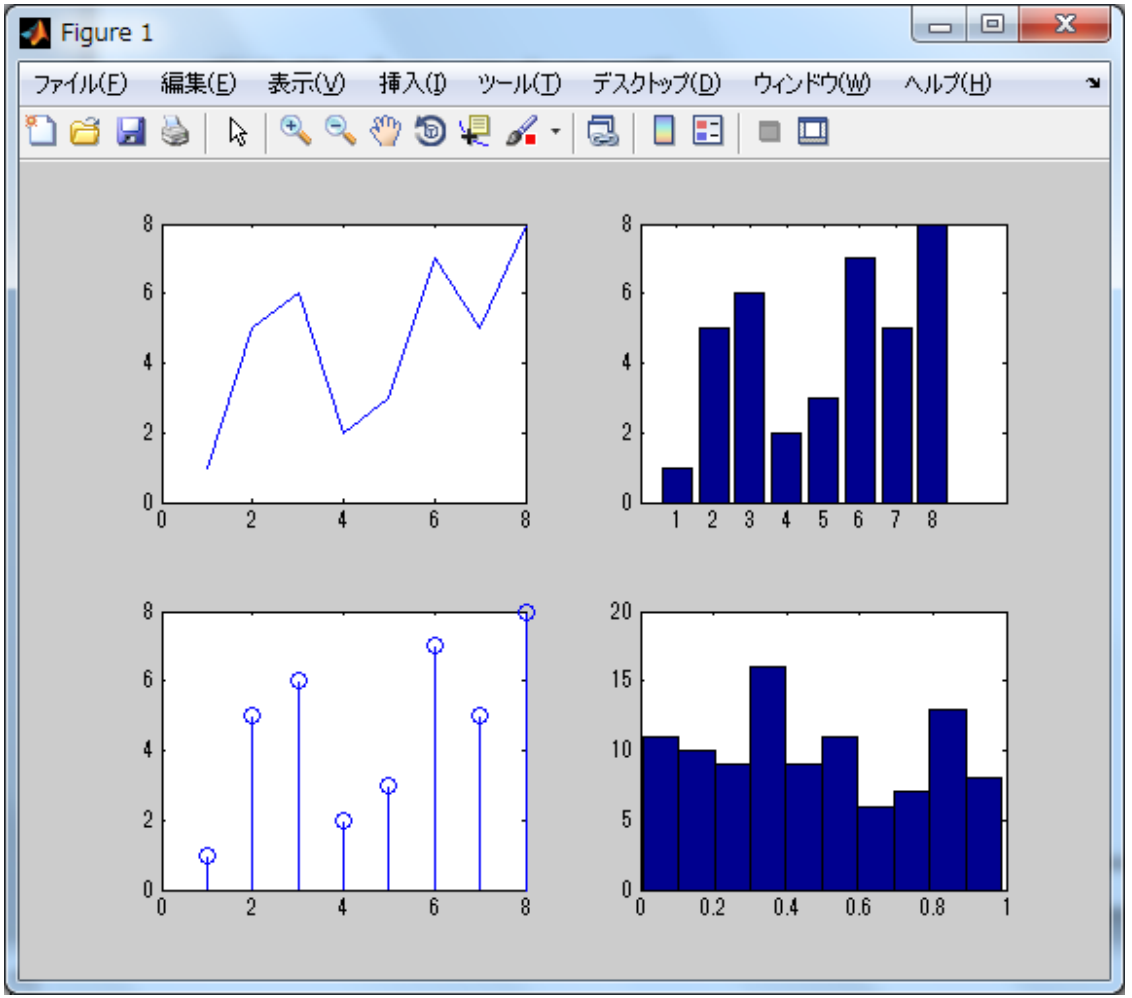
```
>> x=[1:8];
y=[1 5 6 2 3 7 5 8];

subplot(2,2,1)
plot(x,y)

subplot(2,2,2)
bar(x,y)

subplot(2,2,3)
stem(x,y)

subplot(2,2,4)
z=rand(1,100);
hist(z)
```



有用なコマンド

input	a=input(‘statement’) a=input(‘statement’ , ’s’)
clock	t=clock
now	t=now
date	day=date
datestr	day=datestr(now) day=datestr(now,s) 例：s=0→6-Jun-2014 13:35:45 s=1→6-Jun-2014 s=2→ 06/06/14

ヘルプ・サポートコマンド

demo	デモプログラムの実行
help	オンラインでのドキュメント表示
info	MATLABとThe MathWorks に関する情報の表示
lookfor	ヘルプの記載事項からのキーワード検索
path	MATLABの検索パスの制御
type	M-ファイル内容の表示
what	ディレクトリ内のM-,MAT-MEX-ファイル名の表示
which	関数やファイルが存在する位置の表示

変数およびワークスペースの管理

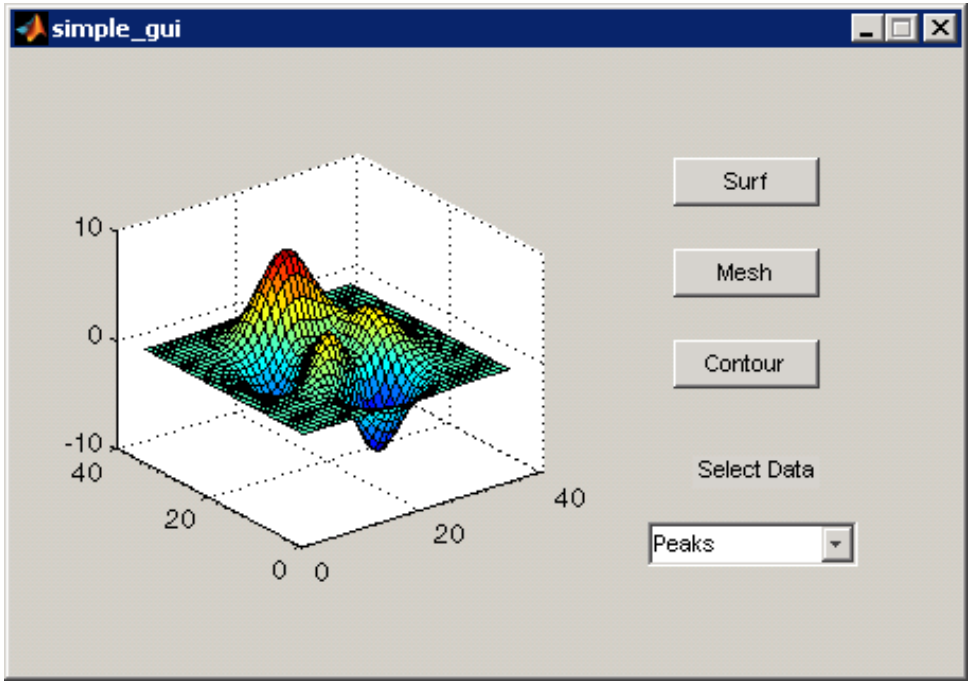
clc	コマンドウィンドウ上の表示の削去
clear	メモリから変数や関数を消去
close	表示しているウィンドウを消去
disp	行列あるいはテキストの表示
length	ベクトルの長さを求める
load	ディスクから変数を読み込む
save	ワークスペースの変数をディスクに保存
size	行列の大きさを求める
who	メモリ内にある変数名の表示
whos	メモリ内にある変数の詳細な表示

MatlabのGUI

- グラフィカル ユーザー インターフェイスの略称
- GUI を使用して、ユーザーは対話的に作業することができる
- GUI のユーザーは、タスクを実行するためにスクリプトを作成したり、コマンド ラインでコマンドを入力する必要はない
- タスクを達成するプログラムのコーディングとは異なり、GUI のユーザーは、タスクがどのように実行されるかについての詳細を理解する必要もない

GUIの例

- GUI コンポーネントの例として、メニュー、ツールバー、プッシュ ボタン、ラジオ ボタン、リスト ボックス、スライダーなどがある



GUI について（以下資料参照）

- **MATLAB Graphical User Interface 開発環境 GUIDEについて**
  SENAC　Vo1.47, No.3, 37-56, 2014

参考文献

- **陳国躍，共同利用支援係 「高機能数値計算・可視化ソフトMATLABの基本的な使い方」**
  SENAC　Vo1.46, No.3, 29-43, 2013