MATLAB Overview

http://www.mathworks.com/products/matlab/

MATLAB® is a high-level language and interactive environment that enables you to perform computationally intensive tasks faster than with traditional programming languages such as C, C++, and Fortran.

- All data are treated as array
- Any declaration for variable or array is not needed as in C(C++) or Fortran
- Many functions are available
- Beautiful graphics are obtained without writing complicated code
- Many useful toolbox

Example : multiplication of a matrix A and a vector x

C language ・・・

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

MATLAB ・・・

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


Numerical Value, Variable, Array

Numerical Value

- MATLAB does not distinguish an integer from a real number with decimal point.
- You do not need to specify the type of a variable.
- Real number is displayed with a decided digit number.

Variable

- The first character of a name of variable should be written by an alphabet and its name should be different from the name of MATLAB functions.
- A capital letter is distinguished from a small letter in the name of variable.

Example:

```
>> a=1; b=2; c=a+b
c =
   3
```

↵ : Enter Key

This is the command which substitutes 1 for a, substitutes 2 for b and calculates c=a+b.
If a formula ends with semicolon then the result is not displayed, but if it ends with comma then the result is displayed.

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

```
>> a=1; b=2; c=a+b;
>>
```

In case that all formulas end with semicolon then nothing is displayed.

By adding three periods in the end of a line, the line could be continued to the next line.

```
>> 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
```

The expression "ans =" is displayed in case that there was no subsitution. You can use a command "disp" if you want to obtain only a result (without "ans=").

```
>> 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
```


Special variable

In order to express some special numerical values or special matrices, some names are fixed as follows:

Name	Meaning
pi	π
Inf, inf	infinity ∞
i, j	imaginary unit $\sqrt{-1}$
eps	machine epsilon (minimum ε satisfying 1+ε>1)
eye	unit matrix
ones	matrix whose all elements are 1
zeros	matrix whose all elements are 0
NaN, nan	Not a Number
realmin	the smallest positive floating point number which is expressible in a computer
realmax	the greatest positive floating point number which is expressible in a computer

※If you substitute a different value for these special variables, then it is maintained until you use “clear” command. Especially i, j are often used as a variable for loop, so you must take care in case you use imaginary unit.

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

7

complex variable

```
>> z1=1+2*i
z1 =
        1 +  2i
>> z2=3+4*j
z2 =
        3 +  4i
>> z1+z2
ans =
        4 +  6i
```

Once a value is substituted for i , then you cannot use i for imaginary unit until you use “clear” command in order to restore it.

```
>> i=2; z2=5+6*i
z2 =
   17
>> clear i
>> z2=5+6*i
z2 =
        5 +  6i
```

8

Changing of accuracy for display 【Note that it is different from the accuracy in computation】

```
>> 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 (default)	fixed point number with 5-digit number
short e	floating point number with 5-digit number
long	fixed point number with 15-digit number
long e	floating point number with 15-digit number
hex	hexadecimal number
short g, long g	(fixed point number or floating point number is automatically determined)

If you changed the accuracy by “format” command, then it is maintained unless you change it again.

9

Array

vector･･･one dimensional array
matrix･･･two dimensional array

An index of an array must be a positive integer starting from 1.

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

row vector

column vector is expressed adding ’ .

Since semicolon expresses the end of a row, a column vector can be also generated like this.

10

There are other methods to generate vectors.

■ Specify the initial value, increment and last value with colon

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

From 1 to 7 incrementing by 2

If the increment is 1 then it can be omitted.

■ Specify the initial value, last value and the number of element using the function “linspace”.

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

Generate 5 elements from 1 to 3 with equal space.

11

```
>> 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
```

matrix with 2 rows and 3 columns

A’ stands for the transposed matrix of A.

transformation of two dimensional array to one dimensional array.
The row is prior to the column in such transformation.

12

Specification of an element

x(k): k-th element of one dimensional array x.
A(i,j): (i,j)-element of two dimensional array A.

```
>> 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
>> A(5,5)
??? The index exceeds the dimension of array
```

An error code is displayed if you accessed to a non-existing element.

Useful command and control key

■ help command

You can check how to use the *command* as follows:
>> help *command*

```
>> 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。
  ...
```

how to use "format"

The statement which starts with % in your original M-file can be also displayed by the help command.
For example, if the file flow.m has the following content,

```
% Program to plot the solution
function [f] = flow(K);
...
```

then help command gives us the following output.

```
>> help flow
 Program to plot the solution
```

You had better add some appropriate comment in the beginning of your M-file in order to remember of which kind of program it is.

■ lookfor command

You can search a command with some keyword.
For example, if you want to know something about maximum then you may input "lookfor max" in the command line.

```
>> 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
...
```

Some functions or commands are displayed with simple explanations.

■ arrow key and control key

In MATLAB the history of command is preserved in the memory, so you can use some key as follows in order to call it.

↑	call the command which was used immediately before
↓	call the command which was used immediately after
→	Move to the right for one letter
←	Move to the left for one letter
Delete	Delete the left letter
Ctrl-A	Move to the front of the line
Ctrl-E	Move to the end of the line
Ctrl-U	Delete the current line
Ctrl-K	Delete some letters from current position to the end of the line
Ctrl-D	Delete the letter with cursor on

Operation, Mathematical function

operation

formula	expression in MATLAB
$a+b$	a + b
$a-b$	a - b
ab	a * b
a/b	a / b
a^b	a^b

comparison operation ⟶ true: 1 false: 0

formula	expression in MATLAB
$a < b$	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	|

19

Example:

```
>> a=1; b=[2;3]; A=[1 2;3 4]; B=[5 2; 3 5];
>> a>0
ans =
   1
>> b==2
ans =
   1
   0
>> A==B
ans =
   0   1
   1   0
>> A<B
ans =
   1   0
   0   1
>> A~=B
ans =
   1   0
   0   1
```

20

element-wise operation

For example, for two matrices $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, B = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$

```
>> A*B
```

gives $AB = \begin{pmatrix} 5 & 4 \\ 11 & 10 \end{pmatrix}$

If you want a matrix such as $\left(a_{ij}b_{ij}\right)$ then replace

'*' by '.*' as follows.

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

21

Example of vectors

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

formula	expression in MATLAB
$(a_1 + b_1, \cdots, a_n + b_n)$	a + b
$(a_1 b_1, \cdots, a_n b_n)$	a .* b
$(a_1 / b_1, \cdots, a_n / b_n)$	a ./ b
$(a_1^{b_1}, \cdots, a_n^{b_n})$	a .^ b

22

Mathematical function

Various mathematical functions are available in MATLAB.
There are also many mathematical functions in each toolbox.
Concerning how to use each functions, use "help" command.

Basic mathematical functions

abs	absolute value
sqrt	square root
log	natural logarithm
exp	exponential function
real	real part of a complex variable
imag	imaginary part of a complex variable
angle	argument of a complex variable
conj	conjugate complex variable
•••	

23

Basic statistical functions

sum	sum of elements
mean	mean value
cov	covariance
corrcoef	correlation coefficient
sort	sort in ascending or descending order
prod	product of elements
cumsum	cumlative sum
cumprod	cumlative product
•••	

trigonometric functions

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	$\cosh(x)$
•••	

matrix functions ••• etc

inv	inverse matrix
det	determinant
lu	LU decomposition
chol	Cholesky decomposition
norm	(various) norms
cond	condition number
eig	eigenvalue
•••	

24

Script and function

There are “script” and “function” among M−files named ○○○○.m.

script file･･･consisting of some commands

You can execute it by typing its file-name.

function file･･･････consisting of definition for function

The name of the file is the name of the function.
You can use the function without any declaration if its file exists in the same directory.

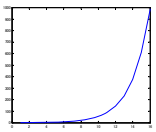
25

Example of a script

```
% An 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
```



26

Definition of function

In the beginning of M−file, describe the following:

function [list of output] = function-name(list of input)

Example of function

```
% 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

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

27

Global variables

The common variables between script and function can be declared as global variables.

Example

```
global a b c
a=1; b=2; c=3;
x=1;
y=func(x);
disp(y)
```

→ cal2.m

```
function y=func(x)
global a b c
y=a*x^2 + b*x +c;
```

→ func.m

```
>> cal2
    6
```

28

Branch, Loop

if statement

```
if formula
  command
else if formula
  command
else
  command
end
```

or

```
if formula, command
else if formula, command
else command
end
```

※”else if” and “else” can be omitted.

29

Example

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

If x does not have any value then the following error message appears:

??? Undefined function or variable 'x'.

30

Loop

A group of repetitive commands are called as loop, and the variable which controls such repetition is called as loop variable.

There are two types concerning a loop.

■ To specify the number of iteration

→ for

■ To specify a condition which the loop should be stopped

→ while

31

for

```
for  loop variable = range of the loop variable
       command
end
```

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

→ i: loop variable, iteration number is 10

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

→ If you want to execute the iteration conversely then these expressions are available.

32

The range of a loop variable can be set with some intervals:

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

→ Loop variable i is set as even number. (i changes from 2 to 10 at two intervals)

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

→ Converse version

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

→ Loop variable can be set as a vector.

33

The continue command is used when you want to move to the next loop, and the break command is used when you want to quit the loop.

```
>> 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
```

→ If k is negative then go to next loop.

→ If s became greater than 5 then quit the loop.

34

while

```
while  condition
         command
end
```

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

→ The commands within the loop are executed while a is greater than or equal to 1.

How to use “break” and “continue” is the same as the case of “for”.

35

Measurement of elapsed time

The elapsed time between “tic” and “toc” can be measured.

```
>> tic;
A=rand(1000,1000);
toc;
elapsed_time = 0.078000
```

```
>> tic;
A=rand(1000,1000);
time=toc;
disp(time);
   0.07800000000000
```

→ The elapsed time can be reserved in a variable.

36

Vectorization, High-speed

Vectorization

Management as vector instead of as each point

High speed is expected

```
>> clear;
tic;
for j=1:1000
   for k=1:1000
      A(j,k) = j+k;
   end
end
toc;
elapsed_time = 17.500000
```

→ We consider making these processes to be more fast.

Method 1

```
>> clear;
tic;
A=zeros(1000,1000);
for j=1:1000
   for k=1:1000
      A(j,k) = j+k;
   end
end
toc;
elapsed_time is 1.938000
```

→ Add this line

In MATLAB any array does not need to be declared in advance.
However, if the size of it is much larger then much time is needed in a loop because the memory for the array has to be reserved in each process of the loop.
In above example, zeros(1000,1000) reserves all needed memory at once (before the loop).

Method 2 (vectorization)

```
>> clear;
tic;
A=zeros(1000,1000);
k=[1:1000];
for j=1:1000
   A(j,k) = j+k;
end
toc;
elapsed_time is 0.125000
```

→ Treat the row as vector

Graphics in 2 dimensions

■ ezplot ･･･displays the graph of function

ezplot('*f(x)*') → Plot f(x) in $-2\pi \le x \le 2\pi$

ezplot('*f(x)*', [a,b]) → Plot f(x) in $a \le x \le b$

ezplot('*f(x,y)*') → Plot f(x,y)=0 in $-2\pi \le x, y \le 2\pi$

ezplot('*f(x,y)*', [a,b]) → Plot f(x,y)=0 in $a \le x, y \le b$

ezplot('*f(x,y)*', [a,b,c,d]) → Plot f(x,y)=0 in $a \le x \le b, c \le y \le d$

ezplot('*x(t), y(t)*') → Plot x=x(t), y=y(t) in $0 \le t \le 2\pi$

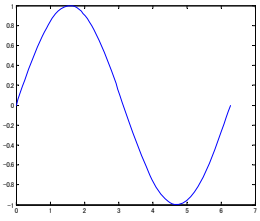
ezplot('*x(t), y(t)*', [a,b]) → Plot x=x(t), y=y(t) in $a \le t \le b$

■ plot ･･･display the 2D graph

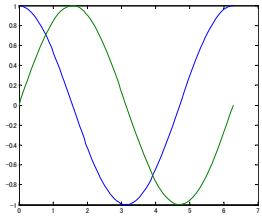
(giving the data of x and y)

Example:

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



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



When the command of ezplot or plot are executed,
The graph is displayed in the window named "Figure 1".

→ The graph would be erased in the next plotting.

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

→ In this way, two graphs are displayed in Figure 1 and Figure 2 respectively.

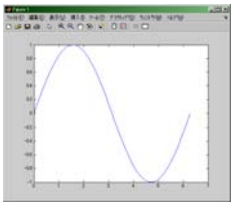


Figure 1

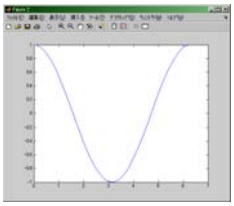


Figure 2

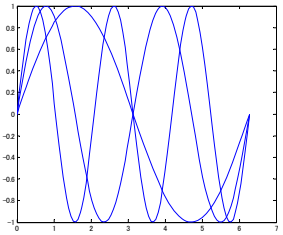
The command "hold on" is useful when you want to display some graphs in the same Figure.
(The command "hold off" is used to quit it.)

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

The graph of

$\sin(x), \sin(2x), \sin(3x)$

are displayed in a Figure.



43

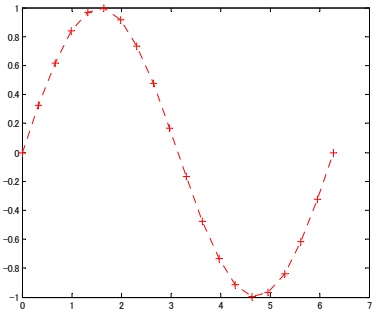
Some options in graphics

plot(x, y, 's') ··· 's' is specified as follows:

b	blue	.	dot	-	solid line
g	green	o	circle shape	:	dotted line
r	red	x	x shape	-.	chained line
c	cyan	+	plus shape	--	broken line
m	magenta	*	star shape	(none)	no line
y	yellow	s	square		
w	white	d	diamond shape		
k	black	v	triangle		
		^	inverted triangle		
		<	triangle (left direction)		
		>	triangle (right direction)		
		p	pentagon		
		h	hexagon		

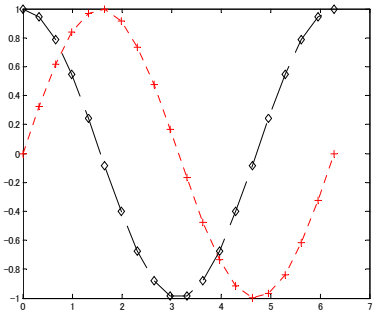
44

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



45

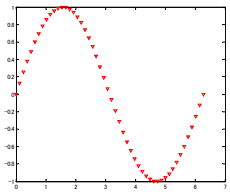
```
>> x=linspace(0, 2*pi, 20);
>> plot(x, sin(x), 'r+:', x, cos(x), 'kd--')
```



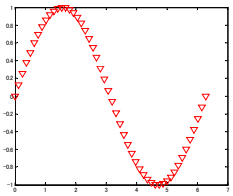
46

The size of marker can be specified by "MarkerSize":

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



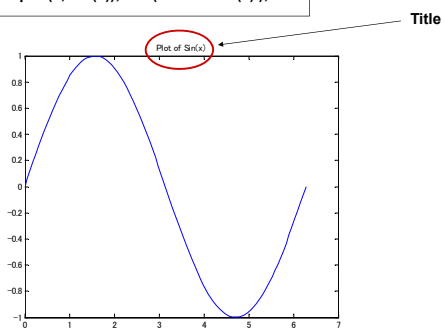
```
>> x=linspace(0, 2*pi, 50);
>> plot(x, sin(x), 'rv', 'MarkerSize', 10)
```



47

The title can be added by title('*text*') in upper part of the graph:

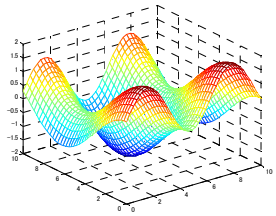
```
>> x=linspace(0, 2*pi, 100);
>> plot(x, sin(x)); title('Plot of Sin(x)');
```



48

Graphics in 3 dimensions

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



Additionally ezplot3 etc.
(Check via "help ezplot3".)

Making use of pause command

When pause command appears in MATLAB script, the execution is interrupted there, and Return key is used to go on. Moreover, e.g. pause(2) can interrupt the execution for two seconds.

Example:

```
%A demo file for graphics

x=linspace(0,2*pi,100);
for a=10:-1:1
  plot(x, a*sin(x));
  grid on;
  hold on;
  pause(2);
end
```

to display the grid lines in the graph

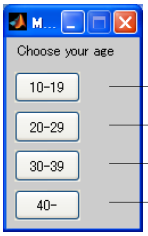
Interface with the screen

■ Choice of meny using "menu" function

```
>> item=menu('Choose your age', '10-19', '20-29', '30-39', '40-')
```

header

Items of menu



click

item = 1

item = 2

item = 3

item = 4

If none of menus was chosen then item = 0.

For example, if you want to choose some formulae to calculate definite integrals for function "f", then you can make the following script after preparing for each functions such as "midpoint".

```
method=menu('Choose integration formula', …
'midpoint formula', 'trapezoid formula', 'Simpson formula')

if method==1      S=midpoint(f)   % Calculation by midpoint formula
elseif method==2  S=trape(f)      % Calculation by trapezoid formula
elseif method==3  S=simpson(f)    % calculation by Simpson formula
end
```

click

method = 1

method = 2

method = 3

■ input function

Using "input" function you can input some value when you execute a MATLAB script.

Example:

```
>> N=input('The number of data?')
```

If you execute above then you have the following question

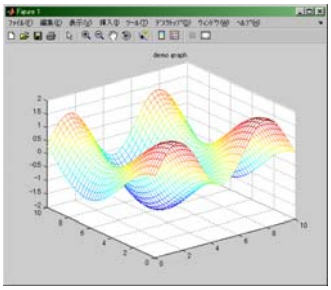
```
The number of data?
```

So if you enter 100 via keyboard then

```
N =
   100
```

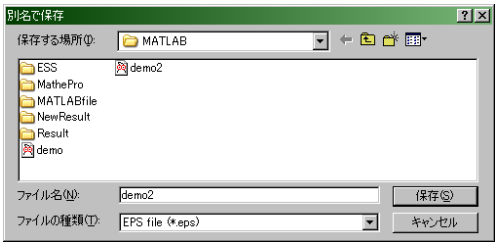
appears and N is set as 100. Here N could be a vector or matrix.

Saving figures as PostScript files



We consider to save this graph as PostScript file.

You only have to choose「EPS file (*.eps)」as a type of saving file.

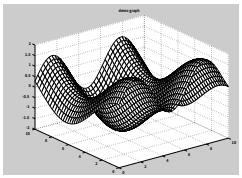


The EPS file can be inserted in TeX documents.

The print command is also available for making PostScript file:

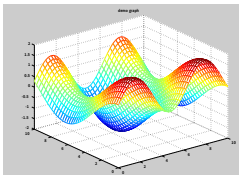
(after displaying graph)

```
>> print test1.ps
```



test1.ps

```
>> print test2 -depsc
```



test2.eps

Input and Output

If you want to save some numerical value or characters into an ASCII file, commands “fopen, fprintf, fclose” are useful.

Example:

```
>> x=[0:pi/8:2*pi];
>> y=[x;sin(x)];
>> fid=fopen('sine.dat', 'w');
>> fprintf(fid, '%s¥n', 'Value of Sine function');
>> fprintf(fid, '%s¥n', 'x  Sine');
>> fprintf(fid, '%3.1f  %6.3f ¥n', y);
>> fclose(fid)
ans =
    0
```

```
fid=fopen('sine.dat', 'w');
```

'w': open the file with writing mode


```
>> fprintf(fid, '%s¥n', 'Value of Sine function');
```

'%s¥n': format for characters; 'Value of Sine function': actual characters

value which is output of fopen

```
>> fprintf(fid, '%3.1f  %6.3f ¥n', y);
```

%m.nf (m, n: natural number)
・・・ m-digit-number with n digits under decimal point

※ ¥n means starting a new line


```
>> type sine.dat

Value of Sine function
x  Sine
0.0  0.000
0.4  0.383
0.8  0.707
1.2  0.924
1.6  1.000
2.0  0.924
2.4  0.707
2.7  0.383
3.1  0.000
3.5 -0.383
3.9 -0.707
4.3 -0.924
4.7 -1.000
5.1 -0.924
5.5 -0.707
5.9 -0.383
6.3 -0.000
```

You can check the content of the file by “type” command.

When the data is writen in the file, the array on MATLAB is transposed.

If you want to read some numerical value from a file, commands “fopen, fscanf, fclose” are useful.

Example:

```
0.0  0.000
0.4  0.383
0.8  0.707
1.2  0.924
1.6  1.000
2.0  0.924
```

sine2.dat

```
>> fid=fopen('sine2.dat', 'r');
>> y=fscanf(fid, '%f %f', [2 inf])';
y =
                0                0
   0.40000000000000   0.38300000000000
   0.80000000000000   0.70700000000000
   1.20000000000000   0.92400000000000
   1.60000000000000   1.00000000000000
   2.00000000000000   0.92400000000000
>> fclose(fid)
ans =
    0
```

transpose !

```
>> fid=fopen('sine.dat', 'r');
```

open the file with reading mode

```
>> y=fscanf(fid, '%f %f', [2 inf])'
```

specify the number of elements which you want to read

N	read N elements as row vector
inf	read until the end of the file
[M,N]	read M*N elements by row order (M lines and N rows) N can be set as inf, but M cannot be.

Supplementations for input and output

Input and output for matrix data

```
A=[1.0 2.0 3.0;4.0 5.0 6.0];
fid=fopen('matrix.dat','w');
for i=1:2
  fprintf(fid,'%f  ',A(i,1:3));
  fprintf(fid,'¥n');
end
fclose(fid)
```

$$B = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

```
fid=fopen('matrix.dat','r');
B=fscanf(fid,'%f',[3 inf])';
fclose(fid)
```

```
1.000000 2.000000 3.000000
4.000000 5.000000 6.000000
```

matrix.dat

Preservation of variables

If you want to save some information about variables on command window, the following command

```
>> save filename
```

makes a file named "filename.mat" and all information about variables are preserved in this file. The following command

```
>> load filename
```

can restore such information later.

※If you omit the file name after "save" command then the file named "matlab.mat" are made automatically.

```
>> save

Saving to: matlab.mat
```


e.g.

```
>> whos
  Name      Size              Bytes  Class

  A       100x100             80000  double array
  B       200x200            320000  double array
  fid       1x1                   8  double array
  i         1x1                   8  double array
  j         1x1                   8  double array
  z         1x1                  16  double array (complex)

Grand total is 50004 elements using 400040 bytes

>> save variable
>> clear
>> whos
>> load variable
>> whos
Name      Size              Bytes  Class

  A       100x100             80000  double array
  B       200x200            320000  double array
  fid       1x1                   8  double array
  i         1x1                   8  double array
  j         1x1                   8  double array
  z         1x1                  16  double array (complex)

Grand total is 50004 elements using 400040 bytes
```