

**Trường Đại Học Bách Khoa TP HCM  
Khoa Công Nghệ Thông Tin**



**Giới thiệu sơ lược về ngôn ngữ  
Matlab  
(Matrix Laboratory)**

# Giới thiệu sơ lược về ngôn ngữ Matlab

Matlab là một ngôn ngữ thông dịch, cho phép thực hiện nhanh chóng các giải thuật, hiển thị dữ liệu (dưới dạng đồ thị 2D, 3D, hình ảnh và thậm chí chuỗi các hình ảnh) và thực hiện các giao tiếp đồ họa dễ dàng.

Tài liệu này giúp làm quen nhanh chóng với Matlab, khiến người đọc cảm thấy thích thú trong việc tìm hiểu thêm.

## 1 Bắt đầu làm quen

### 1.1 Chuẩn bị

Matlab sẽ thông dịch các lệnh được lưu trong tập tin có phần mở rộng *.m* (ví dụ *toto.m*)

Người dùng nên tạo ra một thư mục làm việc (*C:\Temp\AnhVu* chẳng hạn) để lưu các chương trình của mình, gọi *matlab* và yêu cầu nó thực hiện các lệnh có trong tập tin chương trình *toto.m*

### 1.2 Chạy Matlab

Để khởi động Matlab, nhấp chuột vào biểu tượng Matlab nếu bạn dùng HĐH Windows hoặc gõ *matlab* nếu HĐH là Unix.

Khung cửa sổ làm việc của Matlab hiện ra với dấu nhắc *>>*, cho phép người dùng gõ vào các lệnh mà nó sẽ được thực hiện sau khi người dùng gõ enter.

Phía trên cửa sổ là các thanh menu, cho phép người dùng mở tập tin, định nghĩa một số biến làm việc và nhất là truy xuất các tập tin giúp đỡ.

Trước khi làm việc, nên chỉ ra thư mục làm việc (nơi lưu trữ các chương trình của mình). Có 2 cách để thực hiện điều này:

1. Chọn File/Set Path/Browse. Để thoát ra khỏi cửa sổ này, chọn File/Exit Path Brother.
2. Từ dấu nhắc của Matlab, gõ các lệnh: *pwd*, *cd*, *dir*. Các lệnh này cho phép người dùng di chuyển đến thư mục làm việc.

### 1.3 Chạy chương trình

Nếu Matlab đang tích cực tại thư mục làm việc mong muốn và trong thư mục đó có chứa chương trình dưới dạng tập tin *.m*, người dùng chỉ cần gõ tên tập tin (không cần phần mở rộng) từ dấu nhắc Matlab để thực hiện các lệnh lưu trong tập tin đó

Ví dụ gõ  
*>> toto*

#### Chú ý:

- Tên tập tin không được có các ký tự lai (ví dụ các ký tự dấu, khoảng trắng, ...). Matlab sẽ không nhận biết được chính xác các tên tập tin có chứa các ký tự này.
- Không nên dùng các tên quá đơn giản. Ví dụ nếu tập tin có tên *max.m*, khi gõ *max* tại dấu nhắc, Matlab sẽ không biết người dùng muốn thực hiện hàm *max* (lấy số lớn nhất của một ma trận) hoặc các lệnh lưu trữ trong *max.m*
- Cách đơn giản nhất là dùng ký tự đầu tiên đặc biệt cho tên tập tin của riêng mình (ví dụ *k\_toto.m*) hoặc dùng tên tập tin bằng tiếng Việt ;-)

## 2 Cơ bản về ngôn ngữ Matlab

### 2.1 Các phần tử đầu tiên

Lệnh cơ bản đầu tiên cần thực hiện là *clear*. Nó cho phép xóa tất cả các biến trong bộ nhớ của Matlab.

Việc gõ  
*>> Var = 3;*  
sẽ gán cho biến *var* ma trận kích thước 1x1 giá trị 3

Nếu không gõ dấu chấm phẩy (;) ở cuối lệnh, giá trị của *var* sẽ được hiển thị sau phép gán.

Phần còn lại của dòng lệnh sau dấu % sẽ được coi như chú thích.

Để kết nối dòng lệnh hiện tại với dòng lệnh sau đó, gõ dấu ... Ví dụ

```
A = [ 1 2 3 ...  
      4 5 6 ]
```

tương đương với

```
A = [ 1 2 3 4 5 6 ]
```

## 2.2 Xử lý ma trận

### 2.2.1 Tổng quát

```
>> A = [1, 2, 3; 4, 5, 6; 7, 8, 9] % dấu phẩy (hoặc khoảng cách) ngăn cách các cột  
      % dấu chấm phẩy (;) ngăn cách các hàng
```

```
cho A = 1 2 3  
        4 5 6  
        7 8 9
```

```
>> t = 0:0.2:2.8 % tăng các thành phần của vector t từ 0 đến 2.8  
           % mỗi bước 0.2
```

```
cho t = 0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8
```

```
>> signal = sin(t) % tính hàm sin cho các thành phần của t  
signal = 0 0.19 0.38 0.56 0.71 0.84 0.93 0.98 0.99 0.97 0.90 0.80 0.67 0.51  
0.33
```

```
>> ZZ = [1 2 5] + i*[8 6 4] % giá trị ma trận có dạng phức  
ZZ = 1.0 + 8.0i 2.0 + 6.0i 5.0 + 4.0i
```

Dùng lệnh *size* nếu muốn biết kích thước một ma trận

```
>> size(ZZ)  
sẽ cho
```

```
ans = 1 3 % một dòng và 3 cột
```

### 2.2.2 Lấy các giá trị của một ma trận

$A(i, j)$  biểu diễn phần tử dòng  $i$  cột  $j$  của ma trận  $A$

```
>> B = A(2, 3)  
sẽ cho
```

```
B = 6
```

$A(:, j)$  biểu diễn cột thứ  $j$

```
>> C = A(:, 2)  
sẽ cho
```

```
C = 2  
    5  
    8
```

$A(i:k, :)$  biểu diễn các dòng từ  $i$  đến  $k$

```
>> D = A(1:2, :)  
cho
```

```
D = 1 2 3  
    4 5 6
```

$A(i:k, j:l)$  biểu diễn ma trận con

```
>> E = A(2:3, 2:3)  
cho
```

```
E = 5 6  
    8 9
```

### 2.2.3 Xây dựng ma trận có kích thước tăng

Dấu phẩy phân cách các cột và dấu chấm phẩy phân cách các hàng.

```
>> F = [A C]  
sẽ cho
```

```
F = 1 2 3 2
```

```
4 5 6 5
7 8 9 8
>> G = [A; A(2, :)]
cho
G = 1 2 3
    4 5 6
    7 8 9
    4 5 6
>> Z = [] % ma trận rỗng
Có thể bỏ một dòng của ma trận bằng cách sau
>> A(:,2) = []
sẽ cho
A = 1 3
    4 6
    7 9
```

## 2.3 Nhập/Xuất

### 2.3.1 Nhập/Xuất màn hình

```
>> x = input('Nhập giá trị ban đầu: '); % in ra chuỗi "Nhập giá trị ban đầu: " trên
                                        % màn hình, giá trị nhập vào sẽ gán cho x
```

### 2.3.2 Nhập/Xuất tập tin

*Nhập/xuất riêng của Matlab*

- Dạng nhị phân

```
>> save file1 A B C % lưu A, B, C trong tập tin file1.mat
>> load file1 % nạp A, B, C trong bộ nhớ bởi các giá trị
               % lưu trong tập tin file1.mat
```

- Dạng văn bản

Chỉ lưu trữ một ma trận trong một tập tin.

```
>> save file2.dat A -ascii % lưu các giá trị của A trong tập tin
                           % file2.dat dưới dạng văn bản
>> load file2.dat % lấy các giá trị lưu trữ trong file2.dat và
                  % gán nó cho biến file2
```

*Nhập/xuất chuẩn*

Để đọc các giá trị lưu trữ trong tập tin nhị phân, cần phải dùng các lệnh:

```
>> fidin = fopen('file3.dat', 'r'); % mở tập tin file3.dat trong thư mục hiện hành để
                                   % đọc và gán handle trả về cho fidin
>> data = fread(fidin, 2000, 'uchar'); % đọc 2000 giá trị được lưu như unsigned char
                                       % và gán nó cho data
>> fclose(fidin); % đóng tập tin được mở bởi fidin (file3.dat)
```

Để ghi các giá trị trong một tập tin nhị phân có khả năng đọc bởi các công cụ phần mềm khác, ta cần dùng các lệnh:

```
>> fidout = fopen('file4.dat', 'w'); % mở tập tin file4.dat để ghi (dạng nhị phân)
>> fwrite(fidout, data, 'uchar'); % ghi các giá trị data dưới dạng unsigned char
>> fclose(fidout); % đóng tập tin
```

## 2.4 Hiện thị đồ họa

```
>> plot(signal) % vẽ dạng sóng signal
>> mesh(A) % hiển thị đồ họa 3D các giá trị ma trận
>> title('Hình 1') % hiển thị chuỗi trên hình đồ họa
>> subplot(d1, d2, d) % phân chia màn hình thành ma trận d1xd2
                    % và vẽ đồ thị trong vùng thứ d
```

## 2.5 Gỡ lỗi (debug)

Dùng lệnh *whos* để biết danh sách và kích thước các biến trong bộ nhớ hiện tại.

Để ngừng tạm thời trong một danh sách các lệnh, dùng lệnh *pause*. Chương trình sẽ được thực hiện tiếp khi có một phím bất kỳ được gõ.

Để có thể tích cực trong cửa sổ môi trường Matlab (tức người dùng có thể gõ lệnh) trong khi chương trình đang được thực thi, dùng lệnh *keyboard* trong chương trình. Quá trình thực hiện lệnh trong chương trình bị ngắt tạm thời, cho phép người dùng hiển thị giá trị các biến. Khi đó dấu nhắc sẽ trở thành *K>>*. Để chương trình tiếp tục được thực thi, gõ *enter* trong cửa sổ lệnh.

Để dừng chương trình, nhấn *Ctrl-C*.

### 3 Ví dụ chương trình *zap.m*

```
clear % Xóa tất cả dữ liệu trong bộ nhớ
%----- Tạo các tín hiệu -----%
FeSparc=8192; % Tần số lấy mẫu dùng trên các trạm làm việc Sun (Sparc)
TeSparc=1/FeSparc;
FreqSig=input('Tan so tin hieu ?'); % Đặt câu hỏi và gán câu trả lời cho FreqSig
% (thứ 4096 = FeSparc/2)
NbEch=4096 % số mẫu được hiển thị trong cửa sổ làm việc Matlab (không có ;)
t=0:TeSparc:(NbEch-1)*TeSparc; % tạo một vector
Signal=sin(2*pi*FreqSig*t); % tạo ra vector Signal
Coef=0.1;
Bruit=Coef*(2*rand(1,NbEch)-1); % rand: tạo ma trận mà các thành phần có giá trị ngẫu nhiên
SignalBruit=Signal+Bruit;
%----- Xử lý chuỗi các ký tự -----%
FreqString=num2str(FreqSig); % chuyển một số thành chuỗi các ký tự
CoefString=num2str(Coef);
chaine2=['Nhiều trang tai ',CoefString,'%'] % Nối chuỗi
%Minh họa việc ngắt lệnh bằng ...
chaine1=['Tin hieu: hinh sin voi tan so ',FreqString,...
' Hertz']
%----- Hiển thị đồ họa -----%
subplot(2,2,1); % Phân chia cửa sổ đồ họa thành ma trận 2x2, và chọn vùng 1
plot(Signal); % Phác họa vector Signal
title('Signal'); % Tựa đề của đồ họa hiện hành
sound(Signal,FeSparc); % Phát âm thanh của vector Signal, được lấy mẫu tại tần số FeSparc
subplot(2,2,2);
plot(Bruit);
title('Nhiều');
disp('Go phim bat ky de tiep tuc ...');
pause
sound(Bruit,FeSparc);
subplot(2,2,3);
plot(SignalBruit);
title('Tin hieu + nhieu');
disp('Go phim bat ky de tiep tuc ...');
pause
sound(SignalBruit,FeSparc);
subplot(2,2,4);
text('units','normalized','Position',... % Hiển thị chuỗi "chaine2"
[0,0.75],'String',chaine2,'Color','r');
text('units','normalized','Position',[0,0.25],'String',chaine1,'Color','g');
axis off % Xóa trục tọa độ trên hình vẽ hiện tại
clear
desiderata=input('Ban muon nghe mot tap tin am thanh ?','s');
delete(gcf) % Đóng cửa sổ đồ họa hiện tại
if (desiderata=='yes')
FichierIn='_rvmaitr.wav';
[Data,freq]=wavread(FichierIn); % Nạp tần số và tín hiệu trong tập tin "Gong.mat"
whos % Hiển thị dữ liệu mới trong bộ nhớ
```

## Giới thiệu sơ lược về ngôn ngữ Matlab

---

```
plot(Data);
Data=-0.5+Data/max(Data);
sound(Data,freq);
end
% Đọc tập tin sys1.mat được lưu trữ dưới dạng văn bản
fid=fopen('sys1.mat','r');
[h,count]=fscanf(fid,'%f');
status =fclose(fid);
plot(h);
% Xử lý hình ảnh
clear
Data=imread('im.bmp','bmp'); % Lưu hình trong ma trận 3D
coucou=imfinfo('im.bmp','bmp') % Lấy thông tin của hình ảnh
image(Data) % Xem ảnh trắng đen (B&W)
DataYY= 0.299*double(Data(:,:,1))+ ...
0.587*double(Data(:,:,2))+ ...
0.114*double(Data(:,:,3));
% Chỉ lấy các giá trị nguyên
% Các điểm ảnh chạy từ 0 đến 255
DataYY=floor(DataYY);
% Tạo một palette xám có trị từ 0 đến 1
GrayMap=(0:255)/255;
GrayMap=[GrayMap',GrayMap',GrayMap'];
disp('Go nhẹ một phim');
pause
% Khởi tạo palette mặc định
colormap(GrayMap)
% Chỉ số ma trận đi từ 0 đến 255 được đi kèm những chỉ số của palette từ 0 đến 1 (255)
image(DataYY)
% Lệnh sau bắt buộc (xem help imwrite)
DataYY=uint8(DataYY);
% Đề lưu hình trên đĩa cứng:
% Chú ý: độ rộng của hình phải là bội số của 4 (pb windows)
imwrite(DataYY,GrayMap,'new_ima.bmp','bmp')
```

### 4 Danh sách các lệnh

Sau đây là danh sách các lệnh thường dùng. Dừng ngay tón 5 phút để xem qua, nó sẽ giúp bạn tiết kiệm rất nhiều thời gian sau này: nó sẽ giúp bạn tránh việc viết lại các đoạn chương trình vô ích.

#### General Purpose Commands

##### Managing Commands and Functions

**help** Online help for MATLAB functions and M-files  
**helpdesk** Display Help Desk page in Web browser, giving access to extensive help  
**help** for all commands

##### Managing Variables and the Workspace

**clear** Remove items from memory  
**disp** Display text or array  
**length** Length of vector  
**load** Retrieve variables from disk  
**pack** Consolidate workspace memory  
**save** Save workspace variables on disk  
**saveas** Save figure or model using specified format  
**size** Array dimensions  
**who, whos** List directory of variables in memory  
**workspace** Display the Workspace Browser, a GUI for managing the workspace.

#### Controlling the Command Window

**clc** Clear command window  
**echo** Echo M-files during execution  
**format** Control the output display format

#### Working with Files and the Operating Environment

**cd** Change working directory  
**copyfile** Copy file  
**delete** Delete files and graphics objects  
**dir** Directory listing  
**ls** List directory on UNIX  
**mkdir** Make directory  
**pwd** Display current directory  
**!** Execute operating system command

#### Operators and Special Characters

**+** Plus  
**-** Minus  
**\*** Matrix multiplication  
**.\*** Array multiplication  
**^** Matrix power

.^ Array power  
**kron** Kronecker tensor product. 1-4  
\ Backslash or left division  
/ Slash or right division  
./ and .\ Array division, right and left  
: Colon  
( ) Parentheses  
[] Brackets  
{ } Curly braces  
. Decimal point  
... Continuation  
, Comma  
; Semicolon  
% Comment  
! Exclamation point  
' Transpose and quote  
' Nonconjugated transpose  
= Assignment  
== Equality  
<> Relational operators  
& Logical AND  
| Logical OR  
~ Logical NOT  
**xor** Logical EXCLUSIVE OR

#### Logical Functions

**all** Test to determine if all elements are nonzero  
**any** Test for any nonzeros  
**exist** Check if a variable or file exists  
**find** Find indices and values of nonzero elements  
**is\*** Detect state  
**isa** Detect an object of a given class  
**logical** Convert numeric values to logical

#### Language Constructs and Debugging

**MATLAB as a Programming Language**  
**eval** Interpret strings containing MATLAB expressions  
**evalc** Evaluate MATLAB expression with capture.  
**evalin** Evaluate expression in workspace  
**feval** Function evaluation  
**function** Function M-files  
**global** Define global variables  
**nargchk** Check number of input arguments

#### Control Flow

**break** Terminate execution of for loop or while loop  
**case** Case switch  
**catch** Begin catch block  
**else** Conditionally execute statements  
**elseif** Conditionally execute statements  
**end** Terminate for, while, switch, try, and if statements or indicate last index  
**for** Repeat statements a specific number of times  
**if** Conditionally execute statements  
**otherwise** Default part of switch statement

**return** Return to the invoking function  
**switch** Switch among several cases based on expression  
**try** Begin try block  
**warning** Display warning message  
**while** Repeat statements an indefinite number of times

#### Interactive Input

**input** Request user input  
**keyboard** Invoke the keyboard in an M-file  
**menu** Generate a menu of choices for user input  
**pause** Halt execution temporarily

#### Object-Oriented Programming

**double** Convert to double precision  
**int8, int16, int32** Convert to signed integer  
**uint8, uint16, uint32** Convert to unsigned integer

#### Elementary Matrices and Matrix Manipulation

##### Elementary Matrices and Arrays

**eye** Identity matrix  
**ones** Create an array of all ones  
**rand** Uniformly distributed random numbers and arrays  
**randn** Normally distributed random numbers and arrays  
**zeros** Create an array of all zeros  
: (colon) Regularly spaced vector

##### Special Variables and Constants

**ans** The most recent answer  
**eps** Floating-point relative accuracy  
**flops** Count floating-point operations  
**i** Imaginary unit.  
**Inf** Infinity  
**j** Imaginary unit  
**NaN** Not-a-Number  
**nargin, nargout** Number of function arguments  
**pi** Ratio of a circle's circumference to its diameter,  $\pi$   
**varargin, varargout** Pass or return variable numbers of arguments

##### Time and Dates

**calendar** Calendar  
**clock** Current time as a date vector  
**cputime** Elapsed CPU time  
**date** Current date string  
**etime** Elapsed time  
**now** Current date and time  
**tic, toc** Stopwatch timer

##### Matrix Manipulation

**cat** Concatenate arrays  
**diag** Diagonal matrices and diagonals of a matrix  
**fliplr** Flip matrices left-right  
**flipud** Flip matrices up-down  
**repmat** Replicate and tile an array

*reshape* Reshape array  
*rot90* Rotate matrix 90 degrees  
*tril* Lower triangular part of a matrix  
*triu* Upper triangular part of a matrix  
: (colon) Index into array, rearrange array.

#### Elementary Math Functions

*abs* Absolute value and complex magnitude  
*acos*, *acosh* Inverse cosine and inverse hyperbolic cosine  
*acot*, *acoth* Inverse cotangent and inverse hyperbolic cotangent  
*acsc*, *acsch* Inverse cosecant and inverse hyperbolic cosecant  
*angle* Phase angle  
*asec*, *asech* Inverse secant and inverse hyperbolic secant  
*asin*, *asinh* Inverse sine and inverse hyperbolic sine  
*atan*, *atanh* Inverse tangent and inverse hyperbolic tangent  
*atan2* Four-quadrant inverse tangent  
*ceil* Round toward infinity  
*complex* Construct complex data from real and imaginary components  
*conj* Complex conjugate  
*cos*, *cosh* Cosine and hyperbolic cosine  
*cot*, *coth* Cotangent and hyperbolic cotangent  
*csc*, *csch* Cosecant and hyperbolic cosecant  
*exp* Exponential  
*fix* Round towards zero  
*floor* Round towards minus infinity  
*gcd* Greatest common divisor  
*imag* Imaginary part of a complex number  
*lcm* Least common multiple  
*log* Natural logarithm  
*log2* Base 2 logarithm and dissect floating-point numbers into exponent and mantissa  
*log10* Common (base 10) logarithm  
*mod* Modulus (signed remainder after division)  
*nchoosek* Binomial coefficient or all combinations.  
*real* Real part of complex number  
*rem* Remainder after division  
*round* Round to nearest integer  
*sec*, *sech* Secant and hyperbolic secant  
*sign* Signum function  
*sin*, *sinh* Sine and hyperbolic sine  
*sqrt* Square root  
*tan*, *tanh* Tangent and hyperbolic tangent

#### Eigenvalues and Singular Values

*eig* Eigenvalues and eigenvectors  
*gsvd* Generalized singular value decomposition  
*svd* Singular value decomposition

## Data Analysis and Fourier Transform Functions

### Basic Operations

*max* Maximum elements of an array  
*mean* Average or mean value of arrays  
*median* Median value of arrays  
*min* Minimum elements of an array  
*perms* All possible permutations  
*prod* Product of array elements  
*sort* Sort elements in ascending order  
*sortrows* Sort rows in ascending order  
*std* Standard deviation  
*sum* Sum of array elements  
*var* Variance  
*voronoi* Voronoi diagram

### Finite Differences

*del2* Discrete Laplacian  
*diff* Differences and approximate derivatives.  
*gradient* Numerical gradient

### Correlation

*corrcoef* Correlation coefficients  
*cov* Covariance matrix

### Filtering and Convolution

*conv* Convolution and polynomial multiplication  
*conv2* Two-dimensional convolution  
*deconv* Deconvolution and polynomial division  
*filter* Filter data with an infinite impulse response (IIR) or finite impulse response (FIR) filter  
*filter2* Two-dimensional digital filtering

### Fourier Transforms

*abs* Absolute value and complex magnitude  
*angle* Phase angle  
*fft* One-dimensional fast Fourier transform  
*fft2* Two-dimensional fast Fourier transform  
*ifft* Inverse one-dimensional fast Fourier transform  
*ifft2* Inverse two-dimensional fast Fourier transform  
*unwrap* Correct phase angles

## Polynomial and Interpolation Functions

### Polynomials

*conv* Convolution and polynomial multiplication  
*deconv* Deconvolution and polynomial division

### Sound Processing Functions

#### General Sound Functions

*sound* Convert vector into sound

#### SPARCstation-Specific Sound Functions

*auread* Read NeXT/SUN (.au) sound file  
*auwrite* Write NeXT/SUN (.au) sound file

### **.WAV Sound Functions**

*wavread* Read Microsoft WAVE (.wav) sound file  
*wavwrite* Write Microsoft WAVE (.wav) sound file.

### **Character String Functions**

#### **General**

*abs* Absolute value and complex magnitude  
*eval* Interpret strings containing MATLAB expressions  
*real* Real part of complex number  
*strings* MATLAB string handling

#### **String to Number Conversion**

*char* Create character array (string)  
*int2str* Integer to string conversion  
*mat2str* Convert a matrix into a string  
*num2str* Number to string conversion  
*sprintf* Write formatted data to a string  
*sscanf* Read string under format control  
*str2double* Convert string to double-precision value  
*str2num* String to number conversion

### **Low-Level File I/O Functions**

#### **File Opening and Closing**

*fclose* Close one or more open files  
*fopen* Open a file or obtain information about open files

#### **Unformatted I/O**

*fread* Read binary data from file  
*fwrite* Write binary data to a file

#### **Formatted I/O**

*fgetl* Return the next line of a file as a string without line terminator(s)  
*fgets* Return the next line of a file as a string with line terminator(s)  
*fprintf* Write formatted data to file  
*fscanf* Read formatted data from file

#### **File Positioning**

*feof* Test for end-of-file  
*ferror* Query MATLAB about errors in file input or output  
*frewind* Rewind an open file  
*fseek* Set file position indicator  
*ftell* Get file position indicator

#### **String Conversion**

*sprintf* Write formatted data to a string  
*sscanf* Read string under format control

#### **Specialized File I/O**

*imfinfo* Return information about a graphics file  
*imread* Read image from graphics file.  
*imwrite* Write an image to a graphics file

*textread* Read formatted data from text file

### **Multidimensional Array Functions**

*reshape* Reshape array

### **Plotting and Data Visualization**

#### **Basic Plots and Graphs**

*bar* Vertical bar chart  
*barh* Horizontal bar chart  
*hist* Plot histograms  
*hold* Hold current graph  
*loglog* Plot using log-log scales  
*plot* Plot vectors or matrices.  
*semilogx* Semi-log scale plot  
*semilogy* Semi-log scale plot  
*subplot* Create axes in tiled positions

#### **Three-Dimensional Plotting**

*plot3* Plot lines and points in 3-D space

#### **Plot Annotation and Grids**

*grid* Grid lines for 2-D and 3-D plots  
*gtext* Place text on a 2-D graph using a mouse  
*legend* Graph legend for lines and patches  
*plotyy* Plot graphs with Y tick labels on the left and right  
*title* Titles for 2-D and 3-D plots  
*xlabel* X-axis labels for 2-D and 3-D plots  
*ylabel* Y-axis labels for 2-D and 3-D plots  
*zlabel* Z-axis labels for 3-D plots

#### **Surface, Mesh, and Contour Plots**

*contour* Contour (level curves) plot  
*meshc* Combination mesh/contourplot  
*mesh* 3-D mesh with reference plane  
*peaks* A sample function of two variables  
*surf* 3-D shaded surface graph  
*surface* Create surface low-level objects  
*surf* Combination surf/contourplot  
*surf* 3-D shaded surface with lighting

#### **Domain Generation**

*griddata* Data gridding and surface fitting  
*meshgrid* Generation of X and Y arrays for 3-D plots

#### **Color Operations**

*colormap* Set the color look-up table  
*hsv2rgb* Hue-saturation-value to RGB conversion  
*rgb2hsv* RGB to HSV conversion  
*rgbplot* Plot color map

#### **Colormaps**

*bone* Gray-scale with a tinge of blue color map  
*contrast* Gray color map to enhance image contrast  
*cool* Shades of cyan and magenta color map

*copper* Linear copper-tone color map  
*flag* Alternating red, white, blue, and black color map  
*gray* Linear gray-scale color map  
*hot* Black-red-yellow-white color map  
*hsv* Hue-saturation-value (HSV) color map  
*spring* Shades of magenta and yellow color map  
*summer* Shades of green and yellow colormap  
*winter* Shades of blue and green color map

**Printing**

*print* Print graph or save graph to file  
*printopt* Configure local printer defaults  
*saveas* Save figure to graphic file

**Handle Graphics, Object Creation**

*axes* Create Axes object  
*figure* Create Figure (graph) windows  
*image* Create Image (2-D matrix)  
*line* Create Line object (3-D polylines)  
*text* Create Text object (character strings)

**Handle Graphics, Figure Windows**

*capture* Screen capture of the current figure  
*clc* Clear figure window  
*clf* Clear figure  
*clg* Clear figure (graph window)  
*close* Close specified window  
*gcf* Get current figure handle  
*newplot* Graphics M-file preamble for NextPlot property  
*refresh* Refresh figure  
*saveas* Save figure or model to desired output format

**Handle Graphics, Axes**

*axis* Plot axis scaling and appearance  
*cla* Clear Axes  
*gca* Get current Axes handle

**Interactive User Input**

*ginput* Graphical input from a mouse or cursor  
*zoom* Zoom in and out on a 2-D plot

**Region of Interest**

*drawnow* Complete any pending drawing