

Chương 1

NHẬP MÔN MATLAB

Nội dung

- Giới thiệu chung về MATLAB
- Làm việc với MATLAB
- Lập trình với MATLAB
- Các phép tính ma trận nâng cao
- Đồ thị nâng cao
- Vào ra dữ liệu

Giới thiệu chung về MATLAB

- MATLAB (Matrix Laboratory) là phần mềm của hãng MathWorks Inc.
- Đối tượng là các ma trận.
- MATLAB tích hợp các phương pháp tính toán, hiển thị và ngôn ngữ lập trình mạnh để cung cấp cho người sử dụng một môi trường làm việc thuận tiện để giải các vấn đề tính toán khoa học.
- Cấu trúc mở của MATLAB cho phép sử dụng MATLAB và các thành phần của nó để khảo sát dữ liệu, nghiên cứu các thuật toán và tạo các công cụ tiện ích của người sử dụng.

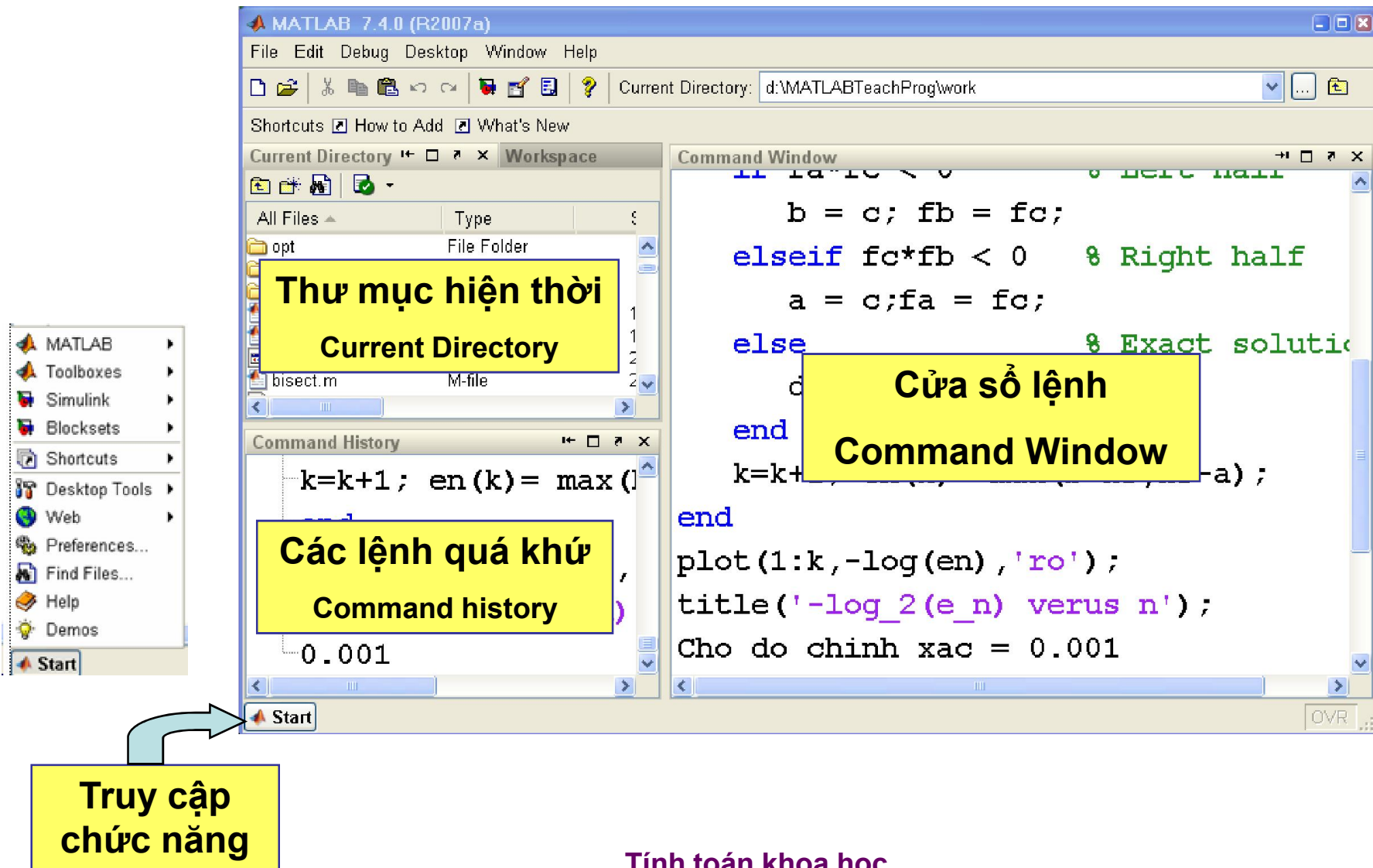
Giới thiệu chung về MATLAB (tiếp)

- Ngoài ra Matlab cũng đã tạo sẵn rất nhiều công cụ tiện ích như:
 - *Khai phá dữ liệu (Data acquisition)*
 - *Phân tích và khảo sát dữ liệu (Data analysis and exploration)*
 - *Hiển thị và xử lý ảnh (Visualization and image processing)*
 - *Dựng mẫu và Phát triển thuật toán (Algorithm prototyping and development)*
 - *Mô hình hóa và mô phỏng (Modeling and simulation)*
 - ...
- MATLAB là công cụ được các nhà khoa học, kỹ sư sử dụng để phát triển các phần mềm giải các bài toán tính toán trong khoa học kỹ thuật.
- Bản thân MATLAB cũng cung cấp công cụ để giải nhiều bài toán của khoa học kỹ thuật.
- MATLAB được dùng trong nhiều trường đại học để hỗ trợ việc giảng dạy các giáo trình toán, đặc biệt là các giáo trình liên quan đến tính toán số như đại số tuyến tính ứng dụng, giải tích số, tính toán khoa học, ...

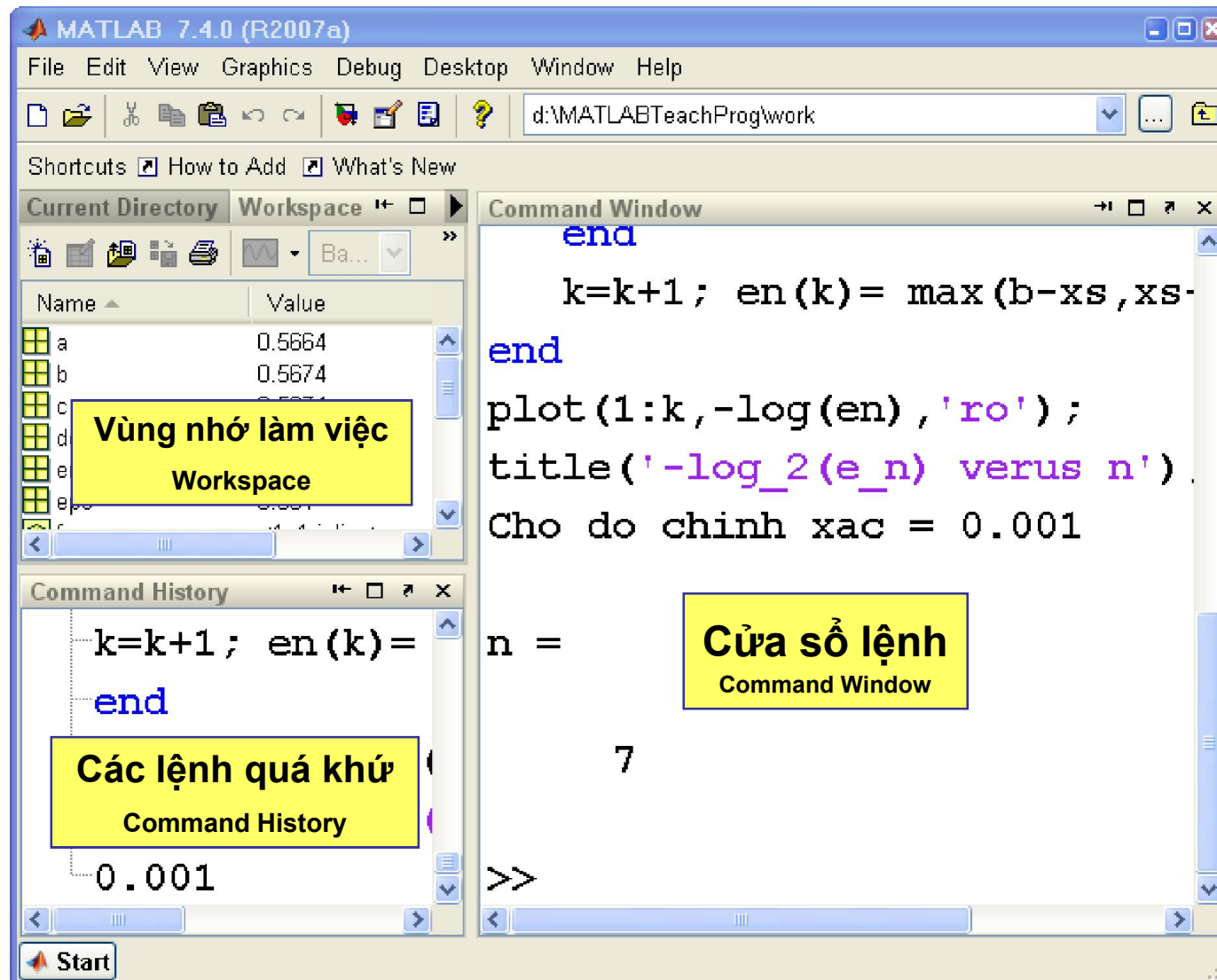
Làm việc với MATLAB

Tính toán khoa học

Màn hình làm việc của Matlab



Màn hình làm việc của Matlab (tiếp)



Tính toán khoa học

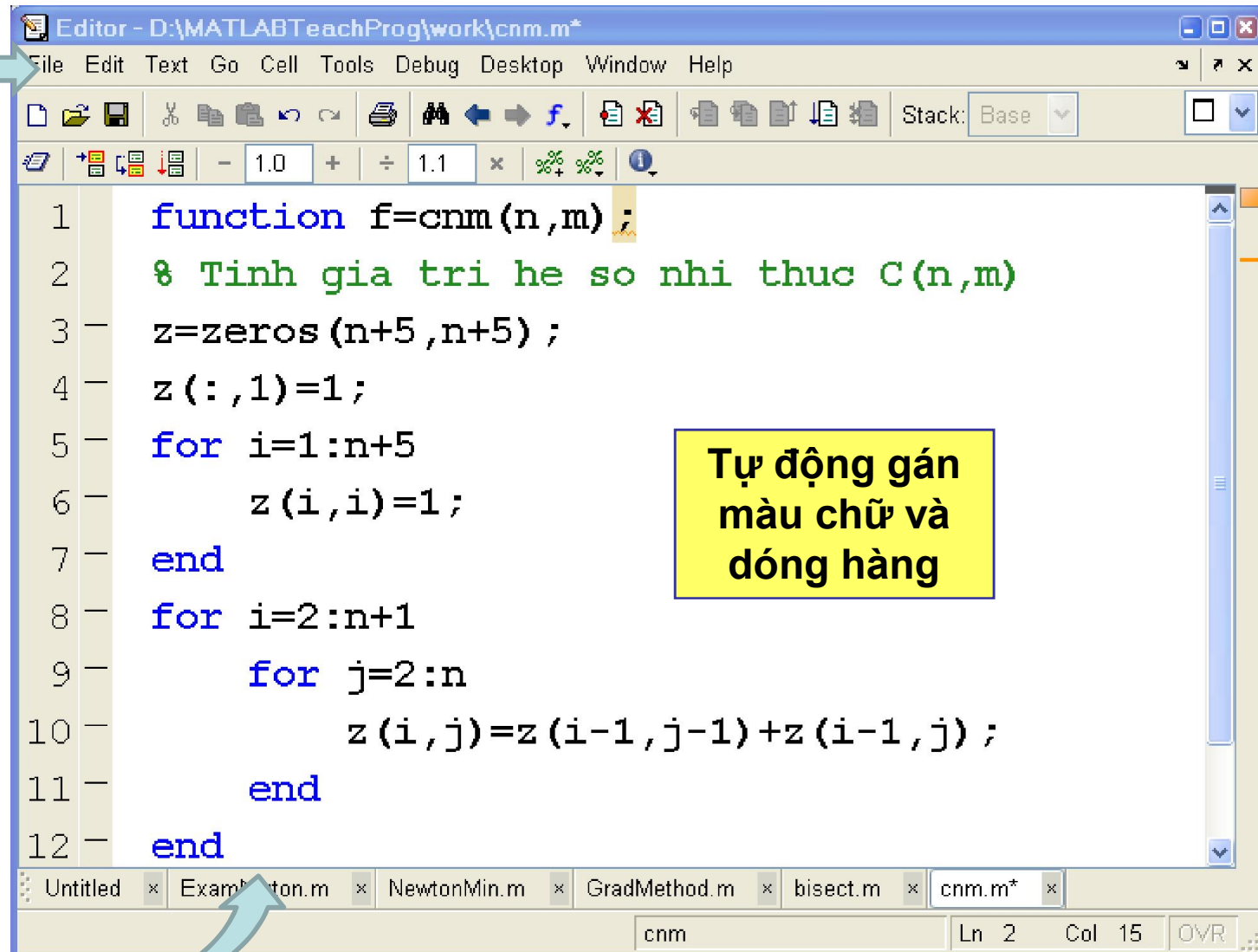
Chương trình trên Matlab

- Matlab có thể làm việc như là một siêu máy tính cầm tay nếu chúng ta chỉ cần Matlab thực hiện một số lệnh bằng cách đánh trực tiếp trên cửa sổ lệnh...
- Chương trình được thực hiện bằng cách nào?
- Chương trình trong Matlab có thể là:
 - **Kịch bản (Scripts)**, hoặc
 - **Các hàm (Functions)**
- **Scripts**: Dãy lệnh Matlab ghi trong một file được đưa vào cửa sổ lệnh và được thực hiện tức thì
- **Functions**: Các môđun chương trình tiếp nhận dữ liệu vào và trả lại kết quả (ví dụ hàm sin nhận đầu vào x và trả lại giá trị $\sin(x)$)
- Chương trình có thể được soạn thảo bằng bất cứ bộ soạn thảo văn bản nào (tuy nhiên Matlab cũng cung cấp bộ soạn thảo chương trình của riêng mình)

Bộ soạn thảo của Matlab (Matlab Editor)

Các chức
năng

Dãy các
file được
mở để
soạn thảo



```
1 function f=cnm(n,m) ;
2 % Tính giá trị hệ số nhị thức C(n,m)
3 z=zeros(n+5,n+5) ;
4 z(:,1)=1;
5 for i=1:n+5
6     z(i,i)=1;
7 end
8 for i=2:n+1
9     for j=2:n
10         z(i,j)=z(i-1,j-1)+z(i-1,j) ;
11     end
12 end
```

Tự động gán
màu chữ và
dóng hàng

Untitled x Example.m x NewtonMin.m x GradMethod.m x bisect.m x cnm.m* x

cnm Ln 2 Col 15 OVR

Tính toán khoa học

Cơ cấu làm việc của Matlab

- Matlab là ngôn ngữ thông dịch (interpreted language)
 - Các câu lệnh được đánh trực tiếp trong cửa sổ lệnh và được thực hiện tức thì
 - Các biến được phân bổ bộ nhớ ngay lần đầu tiên chúng được khởi tạo
 - Muốn thực hiện lại một lệnh chỉ việc gõ lại lệnh đó
- Tất cả các biến được sử dụng trong cửa sổ lệnh được cất giữ vào Vùng nhớ làm việc Base Workspace
 - Có thể gán giá trị mới cho các biến nếu cần thiết
 - Có thể chọn để xóa bỏ một số biến khỏi vùng nhớ làm việc
 - Vùng nhớ làm việc có thể cất giữ vào một file dữ liệu
 - Phần mở rộng của file dữ liệu là `.mat` (ví dụ: `mydata.mat`)
 - File là file nhị phân
 - Các file dữ liệu (đuôi `.mat`) có thể nạp trở lại vào Vùng nhớ làm việc

Câu lệnh, Chỉ thị & Biến

- Tại dấu nhắc của cửa sổ lệnh, người sử dụng có thể gõ:
 - Lệnh (Command):
 - `save mydata` (cất giữ vùng nhớ làm việc vào `mydata.mat`)
 - `whos` (hiển thị danh mục các biến trong vùng nhớ làm việc)
 - Chỉ thị gán (Assignment Statement):
 - `A = width * length;`
 - `B = 267;`
 - Câu lệnh gán chỉ có một tên biến ở vế trái của toán tử gán (=)
 - Vế phải sẽ được tính dựa vào giá trị hiện thời của các biến và kết quả tính được sẽ gán cho biến ở vế trái.
 - Giá trị có thể có dạng số hoặc dạng ký tự
 - Kiểu của biến sẽ được cập nhật mỗi khi nó được gán giá trị (chú ý: rất thoải mái nhưng rất dễ mắc sai lầm...)
 - Biến
 - Phân biệt 31 ký tự đầu tiên (những ký tự tiếp theo bị bỏ qua); Phân biệt chữ hoa hay thường

Làm việc trong chế độ hội thoại

- Khi sử dụng chế độ hội thoại, người sử dụng đánh trực tiếp câu lệnh vào sau dấu nhắc của MATLAB. Khi ấn nút “Enter”, dòng lệnh sẽ được thực hiện.
- Ví dụ,
- `>> x = 1;`
- `>> 4*atan(x)`
- Kết quả sẽ được đưa ra màn hình dưới dạng
 - **ans =**
 - **3.1416**
- Dấu chấm phẩy ";" ở cuối dòng lệnh được sử dụng để ngăn MATLAB không đưa kết quả của phép thao tác.

Chương trình trên MATLAB

- Khi sử dụng chế độ hội thoại, người sử dụng đánh trực tiếp câu lệnh vào sau dấu nhắc của MATLAB. Khi ấn nút “Enter”, dòng lệnh sẽ được thực hiện.
- Ví dụ,
- `>> x = 1;`
- `>> 4*atan(x)`
- Kết quả sẽ được đưa ra màn hình dưới dạng
 - **ans =**
 - **3.1416**
- Dấu chấm phẩy ";" ở cuối dòng lệnh được sử dụng để ngăn MATLAB khựng đưa kết quả của phép thao tác.

Các từ khoá...

- Matlab sử dụng một loạt các từ khoá (reserved words) mà để tránh xung đột, không nên sử dụng để đặt tên biến...

```
for  
end  
if  
while  
function  
return  
elseif  
case  
otherwise
```

```
switch  
continue  
else  
try  
catch  
global  
persistent  
break
```

Các tên biến và tên hàm chuẩn của MATLAB

Tên biến hằng	Mô tả
<code>ans</code>	Biến ngầm định chứa kết quả
<code>beep</code>	Phát tiếng kêu
<code>pi</code>	Hằng số pi
<code>eps</code>	Số 0 của Matlab
<code>inf</code>	infinity
<code>NaN</code>	not a number
<code>i</code> (hoặc) <code>j</code>	Đơn vị phức
<code>realmin, realmax</code>	Số thực dương nhỏ nhất và lớn nhất
<code>bitmax</code>	Số nguyên lớn nhất
<code>nargin, nargsout</code>	Số lượng biến vào/ra của lệnh gọi hàm
<code>varargin</code>	Số lượng biến trong lệnh gọi hàm
<code>varaout</code>	Số lượng biến đầu ra trong lệnh gọi hàm

Các tên biến và tên hàm chuẩn của MATLAB

Tên hàm	Ý nghĩa
abs (x)	Giá trị tuyệt đối của số thực x (hoặc biên độ của số phức x).
acos (x)	hàm ngược của cosin.
asin (x)	hàm ngược của sin.
atan (x)	hàm ngược của tang.
conj (x)	số phức liên hợp
cos (x)	hàm cosin.
exp (x)	hàm mũ của e.

Các tên biến và tên hàm chuẩn của MATLAB

Tên hàm	Ý nghĩa
imag (x)	phần ảo của số phức
log (x)	Logarithm cơ số e
log10 (x)	Logarithm cơ số 10
real (x)	phần thực của x
sign (x)	Hàm dấu, trả lại dấu của đối số
sin (x)	hàm sin.
sqrt (x)	Căn bậc hai
tan (x)	hàm tang

Khuôn dạng dữ liệu

- Mặc dù tất cả các tính toán số trong Matlab đều được thực hiện với độ chính xác kép (double precision), nhưng khuôn dạng của dữ liệu đưa ra có thể định dạng lại nhờ các lệnh định dạng của Matlab.
- Các biến ngầm định cũng như các biến của người sử dụng định nghĩa đều có thể đưa ra theo nhiều khuôn dạng khác nhau.
- Khuôn dạng được chọn nhờ sử dụng lệnh format:
 - **FORMAT SHORT** số dấu phẩy động có 4 chữ số sau dấu.
 - **FORMAT LONG** số dấu phẩy động có 14 chữ số.
 - **FORMAT SHORTE** số dấu phẩy động có 4 chữ số với số mũ.
 - **FORMAT LONGE** số dấu phẩy động có 15 chữ số với số mũ
 - **FORMAT RAT** biểu diễn đúng hoặc gần đúng dưới dạng phân số

Khuôn dạng dữ liệu: Ví dụ

```
>> pi
```

```
ans =
```

```
3.1416
```

```
>> format long, pi
```

```
ans =
```

```
3.14159265358979
```

```
>> format long e, pi
```

```
ans =
```

```
3.141592653589793e+000
```

```
>> format short e, pi
```

```
ans =
```

```
3.1416e+000
```

```
>> format rat, pi
```

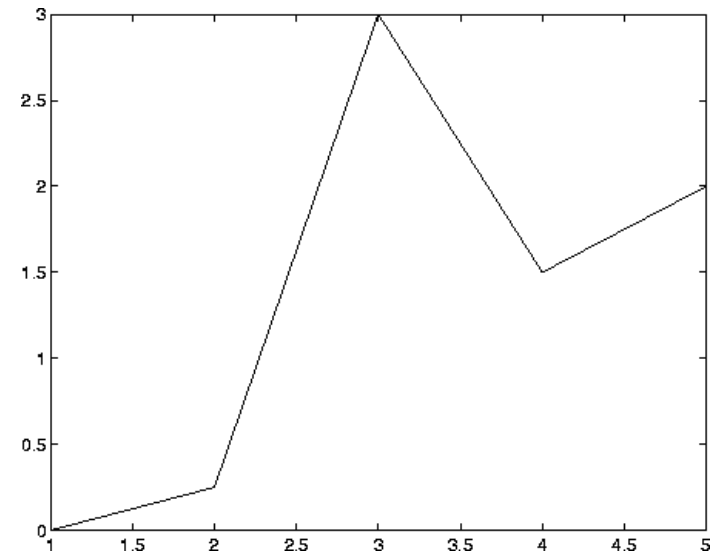
```
ans =
```

```
355/113
```

Vẽ đồ thị đơn giản

- Đơn giản nhất, đồ thị có thể được vẽ nhờ nối các điểm được đánh dấu trên mặt phẳng tọa độ đề các.
- Ví dụ:

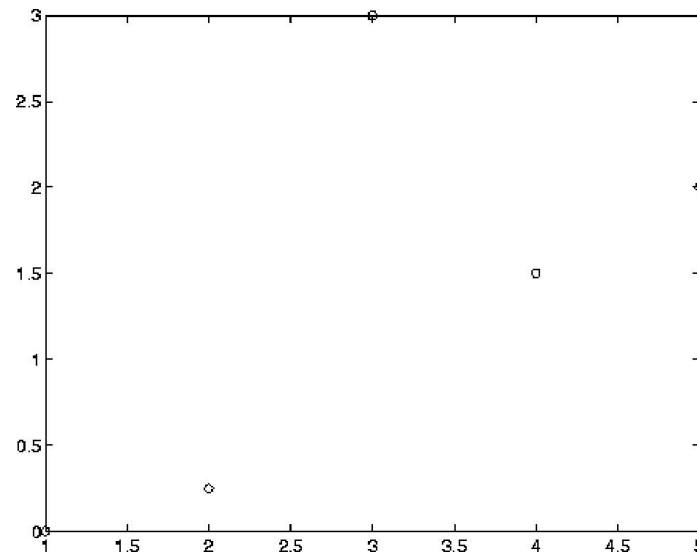
```
>> x = [1;2;3;4;5];  
>> y = [0;.25;3;1.5;2]  
>> plot(x,y)
```



Vẽ đồ thị đơn giản

- Theo ngầm định, Matlab sẽ nối các điểm đánh dấu bởi các đoạn thẳng. Một cách vẽ khác được thực hiện như sau

```
>> plot(x,y,'o')
```



Khởi tạo biến vector và ma trận

- Một trong những điểm mạnh của MATLAB là nó cho phép làm việc với các ma trận và vector. Để sử dụng một biến ta cần khởi tạo nó. Có thể khởi tạo biến vector và ma trận theo nhiều cách.
- Đối với vector (hay ma trận chỉ có một dòng) ta có thể sử dụng các cách khởi tạo sau

```
>> a = [1 2 3 4 5 6 7 8 9 10];
```

```
>> b = [1:10];
```

```
>> c = [1:0.5:5.5];
```

```
>> d = sin(a);
```

```
>> e = [5 d 6];
```

Khởi tạo biến vector và ma trận

- Một trong những cách khởi tạo vector thường dùng là sử dụng toán tử
- **first : increment : last**
- **Câu lệnh**
- **a= first : increment : last**
- **khởi tạo vector dòng a bắt đầu từ phần tử *first* và kết thúc tại phần tử *last* với độ dài bước là *increment*. Nếu không chỉ ra *increment*, thì giá trị ngầm định nó là bằng 1.**

Khởi tạo biến vector và ma trận

- Ví dụ:
- 1) Để khởi tạo vector $a = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$ có thể thực hiện
- `>> a=1:10`
- $a = \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{matrix}$
- 2) Độ dài bước có thể là số âm
- `>> a=[100:-10:20]`
- $a = \begin{matrix} 100 & 90 & 80 & 70 & 60 & 50 & 40 & 30 & 20 \end{matrix}$
- 3) Các giá trị của thông số có thể xác định bởi biểu thức toán học
- `>> c=0:pi/6:2*pi`
- $c = \text{Columns 1 through 6}$
- $\begin{matrix} 0 & 0.5236 & 1.0472 & 1.5708 & 2.0944 & 2.6180 \end{matrix}$
- $\text{Columns 7 through 12}$
- $\begin{matrix} 3.1416 & 3.6652 & 4.1888 & 4.7124 & 5.2360 & 5.7596 \end{matrix}$
- $\text{Column 13} \quad 6.2832$

Khởi tạo biến vectơ và ma trận

- Để khởi tạo ma trận ta có thể làm như sau
- `>> A = [1 2 3; 4 5 6; 7 8 9];`
- `>> B = [x; y; z];`
- `>> C = [A; 10 11 12];`
- Chú ý là các dòng của ma trận được phân tách nhau bởi dấu ";", trong khi đó các phần tử trên một dòng được phân tách bởi dấu cách (hoặc dấu phẩy).

Địa chỉ của mảng (Array addressing)

- Các phần tử của một mảng tổng quát (là vector hay ma trận) có thể địa chỉ hóa theo nhiều cách.
- Cách đơn giản nhất là chỉ ra phần tử bởi vị trí dòng và cột của nó trong mảng
- Các dòng và cột được đánh số bắt đầu từ 1.

Địa chỉ của mảng (Array addressing)

- Ví dụ

```
>> x = [10 5 3 7 -2 8];
```

```
>> x(5)
```

```
ans =
```

```
-2
```

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

```
A =
```

```
3 4 9
```

```
2 5 1
```

```
7 4 1
```

```
>> A(1,3)
```

```
ans =
```

```
9
```

Địa chỉ của mảng (Array addressing)

- Trong trường hợp ma trận, có thể xác định địa chỉ dòng hay cột bởi lệnh :.
- Ví dụ: Nếu cần lấy các phần tử của cột thứ hai và các phần tử của dòng thứ 3 trong ma trận A có thể dùng các lệnh

>> A (: , 2)

>> A (3 , :)

Địa chỉ của mảng (Array addressing)

☐ Thay vectơ chỉ số vào chỗ các chỉ số ta có thể sinh ra hoặc lấy được các mảng con phức tạp hơn.

☐ Ví dụ

```
>> Asub = A(i:j,k:l) ;
```

xác định Asub là ma trận gồm các phần tử nằm trên giao của các dòng từ i đến j, các cột từ k đến l.

☐ Nếu chỉ số được chỉ ra bởi ký tự : thì điều đó có nghĩa là tất cả các giá trị có thể của chỉ số của chiều đó của mảng đều được chọn.

☐ Ví dụ:

```
>> B = A(:, [3 1 2]) ;
```

Địa chỉ của mảng (Array addressing)

□ Ta cũng có thể nối nhiều ma trận để được ma trận mới.

□ Ví dụ:

□ Nếu A và B là các ma trận có cùng số dòng thì

>> C = [A B] ;

cho ma trận C có cùng số dòng như A và B, nhưng có số cột là tổng số cột của A và B.

- Nếu A và B là các ma trận có cùng số cột thì

>> C = [A; B]

cho ma trận C có số cột bằng số cột của A hoặc B nhưng số dòng của C là bằng tổng số dòng của A và B.

Địa chỉ của mảng (Array addressing)

- Để xoá một dòng hay cột của ma trận ta sử dụng cặp ngoặc vuông rỗng []:
- **Ví dụ:** Giả sử A là ma trận trong ví dụ trước. Khi đó, lệnh

$$>> A(:,2) = [];$$
cho ta ma trận thu được từ A bởi việc xoá đi cột thứ 2.
- Không thể xoá bỏ một phần tử của ma trận, bởi vì khi đó kết quả không còn là ma trận nữa.
- Tuy nhiên, nhờ mô tả chỉ số bởi toán tử : có thể thực hiện việc xoá bỏ một phần tử hoặc một dãy phần tử và các phần tử còn lại được sắp xếp lại trong một vectơ dòng

Các phép toán đối với vectơ và ma trận

- Trước hết xét một số hàm của MATLAB cho thông tin về đặc trưng của mảng.
- Hàm **length** cho phép xác định số phần tử của vectơ. Nếu gọi hàm đối với ma trận thì nó sẽ trả lại số lớn hơn trong số dòng và số cột của ma trận.
- Hàm **size**, nó trả lại số dòng và số cột của ma trận hay vectơ. Kết quả của hàm size là một vectơ 1×2 chứa số dòng và số cột của ma trận. Cách sử dụng chuẩn của hàm này là

```
>> [rows cols] = size(A);
```


Các phép toán đối với vectơ và ma trận

- Cộng (và trừ) các ma trận cùng kích thước được thực hiện từng thành phần.

- Ví dụ

```
>> A=[5 -1 2; 3 4 7] ; B=[2 2 1; 5 0 3] ;
```

```
>> A+B
```

```
ans = 7 1 3
```

```
8 4 10
```

- Chú ý các ma trận phải có cùng kích thước:

```
>> C=[3 1; 6 4] ;
```

```
>> A+C
```

```
??? Error using ==> + Matrix dimensions  
must agree.
```

Các phép toán đối với vectơ và ma trận

- Nhân với một số (chia cho một số khác không) cũng được thực hiện theo từng thành phần. Ví dụ:

```
>> A=[5 -1 2; 3 4 7];
```

```
>> 2*A
```

```
ans =
```

```
10 -2 4
```

```
6 8 14
```

- Phép toán * trong tích trên là bắt buộc phải có:

```
>> 2A
```

```
??? 2 | Missing operator, comma, or semi-  
colon.
```

Các phép toán đối với vectơ và ma trận

- Cộng vectơ và nhân vectơ với một số được thực hiện tương tự.

```
>> v=[3; 5] ; w=[-2; 7] ;
```

```
>> 10*v-5*w
```

```
ans =
```

```
40
```

```
15
```

Các phép toán đối với vectơ và ma trận

- Phép nhân hai ma trận cũng có thể thực hiện được trên Matlab.
- Để nhân hai ma trận hay nhân ma trận với vector chỉ việc dùng toán tử $*$ giống như phép nhân của các đại lượng vô hướng.
- Matlab nhận dạng kích thước của đầu vào và thực hiện phép nhân.
- Điều mà người sử dụng quan tâm là kích thước của các ma trận và vector phải phù hợp để có thể thực hiện được phép toán.

Các phép toán đối với vectơ và ma trận

- Ví dụ:

```
>> x = [1 2 3];
```

```
>> A = [4 5 6; 5 4 3];
```

```
>> b = A*x
```

```
??? Error using ==> *
```

```
Inner matrix dimensions must agree.
```

```
>> y = [1; 2; 3];
```

```
>> b = A*y
```

```
b =
```

```
32
```

```
22
```

Các phép toán đối với vectơ và ma trận

- Để thực hiện chuyển vị vectơ hoặc ma trận có thể dùng lệnh chuyển vị ' viết ngay sau tên biến.

```
>> A = [ 4 5; 5 4; 6 3]
```

```
>> A'
```

```
ans =
```

4	5	6
5	4	3

Các phép toán đối với vectơ và ma trận

□ Nếu A là ma trận vuông còn m là số nguyên dương thì A^m sẽ cho ta tích của m nhân tử là A .

- Ví dụ:

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

```
ans =
```

```
      7      10
     15      22
```

```
>> pi^2
```

```
ans =
```

```
10975/1112
```

```
>> mu2= 2^100;
```

```
>> format long, mu2
```

```
mu2 =
```

```
1.267650600228229e+030
```

Các phép toán đối với vectơ và ma trận

- Phép toán lũy thừa áp dụng với ma trận kích thước 1×1 trở thành phép toán lũy thừa của đại lượng vô hướng.

- **Ví dụ:**

```
>> c=3.1;  
>> c^3  
ans =  
29.7910
```

- Số mũ có thể là số thực tùy ý

- **Ví dụ:**

```
>> 3^(-1/3)  
ans =  
0.6934
```


Các phép toán đối với vectơ và ma trận

- **Tích theo từng thành phần** của hai ma trận cùng kích thước A và B là ma trận $A.*B$ với các phần tử là tích của các phần tử tương ứng của A và B.

- Ví dụ:

```
>> A=[ 1 2 3; 4 5 6] ; B=[3 2 1;-1 2 2] ;
```

```
>> A.*B
```

```
ans = 3    4    3
```

```
    -4   10   12
```

Các phép toán đối với vectơ và ma trận

- **Phép chia và lũy thừa theo từng thành phần:**
 $A./B$ và $A.^B$ được định nghĩa tương tự. Lưu ý là các phép tính với các thành phần phải là có nghĩa, nếu không MATLAB sẽ báo lỗi.
- Ví dụ:

```
>> A./B  
ans = 1/3    1    3  
      -4  5/2    3
```

```
>> A.^B  
ans = 1      4    3  
      1/4  25  36
```

Các phép toán đối với vectơ và ma trận

- **Phép cộng ma trận với vô hướng:** Giá trị của vô hướng được cộng vào từng thành phần của ma trận.
- **Ví dụ:**

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

```
>> A+5
```

```
ans =
```

6	7	8
7	8	9

Các hàm và phép toán vectơ hóa

- Các hàm của Matlab đều là ***hàm được vectơ hóa***, nghĩa là nếu áp dụng các hàm của Matlab đối với mảng, nó sẽ tạo ra mảng mới có cùng kích thước với các thành phần là giá trị hàm tại các thành phần tương ứng của mảng ban đầu.
- **Ví dụ:** Vẽ đồ thị hàm $y=\sin(x)$ trên đoạn $[0, 2*\pi]$
 - >> `x = (0:.1:2*pi) ;`
 - >> `y = sin(x) ;`
 - >> `plot(x,y)`

Các hàm và phép toán vectơ hóa

- Matlab cũng cung cấp các phép toán số học được vectơ hóa, được ký hiệu giống như phép toán số học thông thường nhưng có thêm dấu chấm "." ở trước.
- **Ví dụ:** Để vẽ đồ thị hàm số: $y = x/(1+x^2)$ trên đoạn $[-5, 5]$ ta có thể dùng các lệnh

```
>> x = (-5:.1:5) ;
>> y = x. / (1+x.^2) ;
>> plot(x,y)
```

Các hàm tạo ma trận đặc biệt

- **zeros(m,n)** tạo ma trận $m \times n$ gồm toàn số 0;
- **ones(m,n)** tạo ma trận $m \times n$ gồm toàn số 1;
- **eye(n)** tạo ma trận đơn vị $n \times n$;
- **diag(v)** (giả thiết v là vectơ n chiều) tạo ma trận đường chéo kích thước $n \times n$ với v là đường chéo.

Biến chuỗi trong MATLAB

- MATLAB cho phép sử dụng biến chuỗi ký tự: Sử dụng lệnh gán

S = 'Any Characters'

cho phép tạo mảng ký tự (chuỗi ký tự).

- Ví dụ:

```
>> msg = 'You''re right!'
```

```
msg =
```

```
    You're right!
```

- Lưu ý: Để tạo dấu ' trong chuỗi ký tự cần đánh hai dấu '

Biến chuỗi trong MATLAB

- Lệnh `S = [S1 S2 ...]` ghép các chuỗi `S1, S2, ...` thành một chuỗi mới `S`.

- Ví dụ:

```
>> name = ['Michel' ' Paul ' ' Heath ']  
name =  
      Michel Paul  Heath  
  
>> name(1)  
ans =  
      M  
  
>> name(1)+name(9)  
ans =  
     174
```

- Lưu ý đến cách truy cập đến các thành phần trong chuỗi.

Biến chuỗi trong MATLAB

- $S = \text{char}(X)$ tạo S là ký tự có mã ASCII là X .
- $X = \text{double}(S)$ chuyển ký tự thành số
- Ví dụ:

```
>> char([65 66 67])
```

```
ans =
```

```
ABC
```

```
>> X = double('ABC')
```

```
X =
```

```
65      66      67
```

Biến chuỗi trong MATLAB

- Để khởi tạo biến mảng mà mỗi phần tử là một chuỗi, sử dụng:

```
>> S = { 'Hello' 'Yes' 'No' 'Goodbye' }  
S =  
    'Hello'    'Yes'    'No'    'Goodbye'  
>> S(4)  
ans =  
    'Goodbye'
```

LẬP TRÌNH TRÊN MATLAB

Tính toán khoa học

Câu lệnh trong Matlab

- Câu lệnh hay gặp nhất trong Matlab có dạng

variable = expression

Câu lệnh này sẽ gán giá trị của biểu thức ***expression*** cho biến ***variable***.

Ví dụ:

```
>> x=2^10*pi
```

```
x =
```

```
3.216990877275948e+003
```

Câu lệnh trong Matlab

- Câu lệnh của Matlab cũng còn có thể có dạng đơn giản như sau:

expression

trong trường hợp này giá trị của biểu thức sẽ được gán cho biến ngầm định có tên là ***ans***.

Ví dụ:

```
>> 2*pi*exp(-5)
```

```
ans =
```

```
0.04233576958521
```

Câu lệnh trong Matlab

- Cuối cùng, một dạng nữa của câu lệnh trong Matlab là

variable

- Nếu như biến đã được gán giá trị trước đó thì nội dung của biến được đưa ra màn hình,
 - Nếu trái lại sẽ có thông báo rằng biến chưa được xác định.
- Người sử dụng có thể tận dụng điều này để kiểm tra xem một tên biến đã được dùng hay chưa.

Câu lệnh trong Matlab

- Câu lệnh của Matlab sẽ được thực hiện ngay sau khi nhấn phím Enter. Nếu câu lệnh Matlab kết thúc bởi Enter, theo ngầm định, Matlab sẽ đưa kết quả thực hiện lên thiết bị ra chuẩn (ngầm định là màn hình).
- Muốn tránh việc đưa kết quả ra ngay trực tiếp sau câu lệnh, cần kết thúc câu lệnh bởi dấu ; sau đó mới đến Enter.
- Khi câu lệnh của Matlab quá dài có thể ngắt thành hai hoặc nhiều dòng sử dụng dấu nối dòng ... ở cuối mỗi dòng chứa câu lệnh.

Câu lệnh trong Matlab

- Ví dụ:

```
>> avariablewithlongname = 100 + (32-17.33)*5 ...  
+ 2^3 - log(10)/log(2);
```

- Để xóa tất cả biến trong Matlab, ta dùng lệnh

```
>> clear
```

- Để xóa một số biến cụ thể, ta dùng lệnh

```
>> clear var1 var 2 ...
```

trong đó *var1*, *var2*, .. Là các tên của các biến cần xóa.

Các phép toán quan hệ và logic trong Matlab

Phép toán quan hệ	Ý nghĩa
<	Nhỏ hơn
<=	Nhỏ hơn hoặc bằng
>	Lớn hơn
>=	Lớn hơn hoặc bằng
==	Bằng
~=	Không bằng
Phép toán logic	Ý nghĩa
&	AND (hội)
	OR (tuyển)
~	NOT (phủ định)
0	FALSE
Số khác không	TRUE

Câu lệnh if

- Dạng tổng quát của câu lệnh if là

```
if expr1
  statements1
elseif expr2
  statements2
...
else
  statements
end
```

elseif và else là tùy chọn

- Nhóm lệnh đi ngay sau biểu thức (*expr*) đầu tiên có giá trị khác 0 sẽ được thực hiện.
- Nếu không có *expr* nào khác 0 thì nhóm lệnh sau else được thực hiện

Câu lệnh if

□ Ví dụ:

```
>> t = rand(1) ;  
>> if t > 0.75  
    s = 0;  
elseif t < 0.25  
    s = 1;  
else  
    s = 1-2*(t-0.25) ;  
end  
>> s  
s =  
    0  
  
>> t  
t =  
    0.7622
```

Câu lệnh if

- Các phép toán quan hệ trong Matlab là <, >, <=, >=, == (so sánh bằng logic), và ~= (không bằng).
- Các phép toán hai ngôi này sẽ trả lại giá trị 0 hoặc 1 (đối với các biến vô hướng):

```
>> 5>3
```

```
ans =
```

```
1
```

```
>> 5<3
```

```
ans =
```

```
0
```

```
>> 5==3
```

```
ans =
```

```
0
```

Câu lệnh vòng lặp for

- Vòng lặp for lặp lại các câu lệnh trong thân của nó đối với các giá trị của biến chạy được lấy từ một vector dòng cho trước.
- Ví dụ

```
>> for i=[1,2,3,4]
```

```
    disp(i^2)
```

```
end
```

```
1
```

```
4
```

```
9
```

```
16
```

Câu lệnh vòng lặp for

- Chú ý đến việc sử dụng hàm nội trú **disp**: hàm này đưa ra màn hình nội dung của biến.
- Vòng lặp, cũng giống như câu lệnh if, phải kết thúc bởi **end**. Vòng lặp ở trên thường được viết dưới dạng như sau.

```
>> for i=1:4  
    disp(i^2)  
end  
1  
4  
9  
16
```

- Nhớ lại cách khởi tạo 1:4 cũng chính là [1, 2, 3, 4].

Câu lệnh vòng lặp for

- Đoạn lệnh

```
n=4; x = []; for i=1:n, x = [x, i^2], end
```

hay

```
x=[];
```

```
for i= 1:n
```

```
    x = [x, i^2];
```

```
end
```

sẽ tạo ra vector $x = [1, 4, 9, 16]$.

- Đoạn lệnh

```
n=4;x=[]; for i=n:-1:1, x = [x, i^2],end
```

sẽ tạo ra vector $x = [16, 9, 4, 1]$.

Câu lệnh while

- Câu lệnh có dạng sau:

while *expr*

 Các câu lệnh

end

- Các câu lệnh trong thân của vòng lặp **while** sẽ được lặp lại chừng nào biểu thức *expr* còn là true (có giá trị khác 0).

Câu lệnh while

- Ví dụ:

```
>> x=1;  
>> while 1+x > 1  
    x = x/2;  
end  
>> x  
x =  
    1.1102e-16
```

- Chú ý: Về mặt chính xác toán học thì $1 + x > 1$ đối với mọi $x > 0$.

Câu lệnh switch

- Câu lệnh **switch** cho phép thực hiện rẽ nhánh dựa trên giá trị biểu thức.
- Dạng tổng quát của câu lệnh **switch** là:

```
switch biethuc
    case giatri
        cac cau lenh
    case {giatri1, giatri2, giatri3,...}
        cac cau lenh
    ...
    otherwise
        cac cau lenh
end
```

Ví dụ:

```
>> date= 'Sunday' ;  
>> switch lower(date)  
    case { 'sunday' , 'saturday' }  
        disp('the weekend. ' )  
    otherwise  
        disp('the workday. ' )  
    end
```

Kết quả: the weekend.

Câu lệnh break

- Lệnh break dùng để ngắt việc thực hiện vòng lặp WHILE hay là FOR. Trong vòng lặp lồng nhau, BREAK sẽ chỉ thoát khỏi vòng lặp trong.
- Nếu sử dụng lệnh break ngoài vòng lặp FOR hay WHILE trong kịch bản hoặc hàm của MATLAB, lệnh này sẽ chấm dứt việc thực hiện kịch bản hay hàm tại chính điểm đó. Lệnh break sử dụng trong câu lệnh IF, SWITCH-CASE, sẽ ngắt câu lệnh tại điểm đó.

Kịch bản (Script)

- Kịch bản là dãy các câu lệnh của Matlab được ghi lại trong một m – file, file đuôi .m.
- Khi đánh tên của file (không cần .m), dãy các lệnh này sẽ được thực hiện.
- Lưu ý: m – file phải được đặt tại một trong các thư mục mà Matlab sẽ tự động tìm kiếm m – file trong đó; danh mục các thư mục như thế có thể xem nhờ lệnh **path**.
- Một trong những thư mục mà Matlab luôn khảo sát chính là thư mục hiện thời. Thư mục này được hiển thị trong cửa sổ Current Directory.
- Người sử dụng có thể thay đổi thư mục hiện thời nhờ dùng các tiện ích trong cửa sổ này.
- **Câu hỏi:** thay đổi thư mục hiện thời như thế nào?

Kịch bản: Ví dụ

□ Ví dụ: Giả sử file `plotsin.m` chứa các dòng

```
x = 0:2*pi/N:2*pi;
```

```
y = sin(w*x);
```

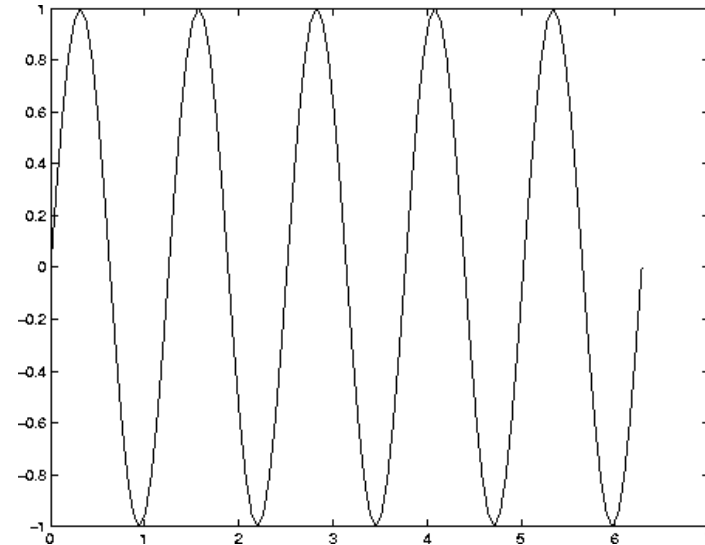
```
plot(x,y)
```

□ Khi đó dãy lệnh

```
>> N=100;w=5;
```

```
>> plotsin
```

sẽ đưa ra đồ thị



Kịch bản (Script)

- Ví dụ này cho thấy các lệnh trong một kịch bản có thể truy nhập các biến được xác định trong Matlab được coi là biến tổng thể (lưu ý truy nhập biến N và w trong `plotsin.m`).
- Các câu lệnh trong kịch bản được thực hiện giống như là chúng được đánh trực tiếp từ bàn phím.

Hàm (function)

- Việc sử dụng hàm có những ưu điểm hơn hẳn kịch bản:
 - Dùng hàm, người sử dụng có thể tạo những lệnh mới trong Matlab.
- Một hàm được định nghĩa trong m-file bắt đầu bởi dòng có cấu trúc sau:
function [out1,out2,...] = funcname(inp1, inp2,...)
- Phần còn lại của m-file sẽ chứa các câu lệnh của Matlab tính giá trị của các đầu ra (output_i) và hoàn thành các công việc đòi hỏi khác.

Hàm (function)

- Điểm quan trọng cần lưu ý là: khi hàm được gọi để thực hiện Matlab sẽ tạo một không gian nhớ địa phương.
- Các lệnh trong hàm không được truy nhập các biến từ không gian nhớ toàn cục (trong phần hội thoại), ngoại trừ các biến được liệt kê trong danh sách đầu vào.
- Cũng chính vì lẽ đó, các biến được tạo ra khi thực hiện hàm sẽ bị xóa bỏ khi việc thực hiện hàm kết thúc, ngoại trừ các biến được liệt kê trong danh sách đầu ra

Ví dụ

- Lập Hàm giải phương trình bậc 2

CÁC PHÉP TÍNH MA TRẬN NÂNG CAO

Trị riêng và vectơ riêng

- Một trong những phép tính rất hay phải thực hiện với ma trận đó là tìm giá trị riêng và vectơ riêng với lệnh **eig**.
Nếu A là ma trận vuông, thì

$$\mathbf{ev} = \mathbf{eig}(A)$$

sẽ cho ta giá trị riêng của A trong một vectơ, còn

$$[V, D] = \mathbf{eig}(A)$$

cho ta phân tích phổ của A :

- V là ma trận mà các cột của nó là các vectơ riêng của A ,
 - D là ma trận đường chéo với các phần tử trên đường chéo là các trị riêng của A .
 - Đẳng thức $AV = VD$ được thực hiện.
- Nếu A là chéo hoá, thì V là chéo hoá, còn nếu A là đối xứng thì V là trực giao (nghĩa là $V^T V = I$).

Ví dụ

```
A = [1 3 2;4 5 6;7 8 9];
>> eig(A)
ans =
    15.9743
   -0.4871 + 0.5711i
   -0.4871 - 0.5711i
>> [V,D] = eig(A)
V =
   -0.2155          0.0683 + 0.7215i    0.0683 - 0.7215i
   -0.5277   -0.3613 - 0.0027i   -0.3613 + 0.0027i
   -0.8216    0.2851 - 0.5129i    0.2851 + 0.5129i
D =
    15.9743          0          0
         0   -0.4871 + 0.5711i          0
         0         0   -0.4871 - 0.5711i
>> A*V-V*D
ans = 1.0e-14 *
   -0.0888          0.0777 - 0.1998i    0.0777 + 0.1998i
         0   -0.0583 + 0.0666i   -0.0583 - 0.0666i
         0   -0.0555 + 0.2387i   -0.0555 - 0.2387i
```

Ví dụ: A là ma trận đường chéo

```
>> A=[1 2 3; 0 3 4; 0 0 5]
```

```
A =
```

1	2	3
0	3	4
0	0	5

```
>> [V, D]= eig(A)
```

```
V =
```

1.0000	0.7071	0.6163
0	0.7071	0.7044
0	0	0.3522

```
D =
```

1	0	0
0	3	0
0	0	5

Ví dụ: A là ma trận đối xứng

```
>> A=[5 6 7; 6 7 8; 7 8 9]
```

```
A =
```

5	6	7
6	7	8
7	8	9

```
>> [V, D]= eig(A) ; V
```

```
V =
```

0.7685	0.4082	0.4927
0.0660	-0.8165	0.5736
-0.6365	0.4082	0.6544

```
>> V*V'
```

```
ans =
```

1.0000	-0.0000	0.0000
-0.0000	1.0000	0.0000
0.0000	0.0000	1.0000

Các phép toán nâng cao

- Matlab còn cung cấp rất nhiều hàm liên quan đến xử lý ma trận. Phần lớn các hàm như vậy liên quan đến phân tích ma trận dưới dạng tích. Một số phép phân tích hay dùng trong số các phân tích như vậy là:
 - **lu** tính phân tích LU của ma trận;
 - **chol** tính phân tích Cholesky của ma trận đối xứng xác định dương;
 - **qr** tính phân tích QR của ma trận;
 - **svd** tính trị chính qui (singular values) hay trị chính qui phân rã (singular value decomposition) của ma trận;
 - **cond, condest, rcond** tính hoặc đánh giá các số điều kiện của ma trận (condition numbers);
 - **norm** tính các loại chuẩn (norms) của ma trận hay vectơ;

ĐỒ HOẠ NÂNG CAO

Matlab có thể đưa ra nhiều dạng đồ thị khác nhau:

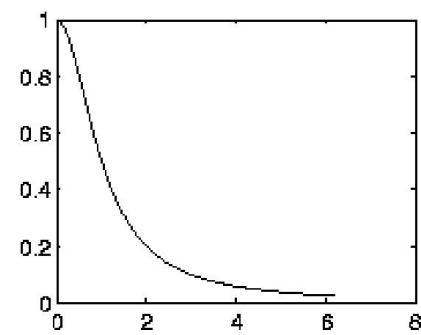
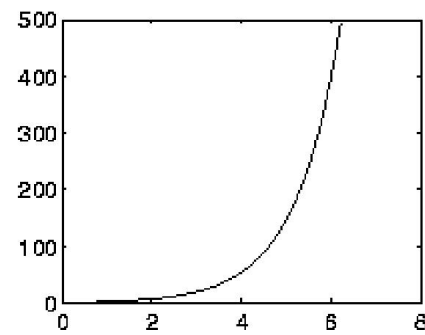
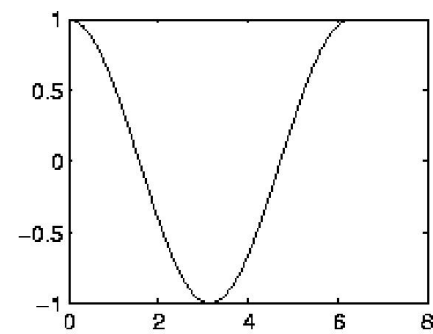
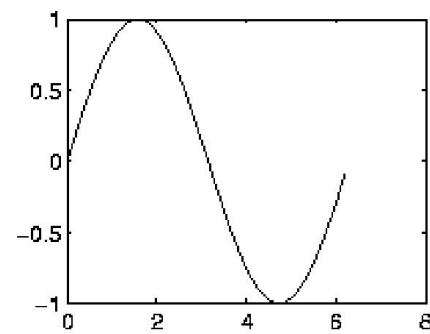
- ☐ đường cong hai chiều (2D curves),
 - ☐ mặt 3 chiều (3D surfaces),
 - ☐ đường khung của mặt 3 chiều (contour plots of 3D surfaces),
 - ☐ đường cong tham số 2D và 3D (parametric curves in 2D and 3D).
-

Đưa nhiều đồ thị lên cùng một cửa sổ

- Lệnh subplot tạo nhiều bản vẽ trên cùng một cửa sổ. Chính xác hơn, subplot(m,n,i) tạo mn bản vẽ, được xếp trong mảng với m dòng và n cột. Lệnh này cũng bắt buộc lệnh plot tiếp theo phải chuyển đến hệ tọa độ thứ i (đếm theo số dòng). Ví dụ

```
>> t = (0:.1:2*pi)';  
>> subplot(2,2,1)  
>> plot(t,sin(t))  
>> subplot(2,2,2)  
>> plot(t,cos(t))  
>> subplot(2,2,3)  
>> plot(t,exp(t))  
>> subplot(2,2,4)  
>> plot(t,1./(1+t.^2))
```

Đưa ra nhiều đồ thị



Tính toán khoa học

Vẽ đường

- Hàm **plot** được sử dụng để vẽ dữ liệu trên mặt phẳng. Cho vectơ x gồm các hoành độ x_1, \dots, x_n và vectơ y gồm các tung độ y_1, \dots, y_n , lệnh **plot(x,y)** sẽ vẽ các điểm từ (x_1, y_1) đến (x_n, y_n) . Theo ngầm định các điểm này sẽ được nối theo thứ tự bởi các đoạn thẳng.
- Tổng quát câu lệnh có dạng sau **plot(x1, y1, linestyle1, ..., xn, yn, linestylen)** trong đó **linestyle i** ($i=1, 2, \dots, n$) là xâu hoặc biến xâu chứa kiểu đường vẽ.

Vẽ đường

- Câu lệnh này cho phép vẽ các bộ dữ liệu: (x_1, y_1) với kiểu đường vẽ `linestyle1`, (x_2, y_2) với kiểu đường vẽ `linestyle2`, ..., (x_n, y_n) với kiểu đường vẽ `linestylen` trên cùng một hệ trục tọa độ.
- Kiểu đường vẽ xác định màu vẽ, ký tự dùng để hiển thị điểm dữ liệu và dạng của đường nối. Kiểu đường vẽ là xâu được tạo bằng cách ghép các ký tự từ các cột sau, mỗi cột lấy không quá một ký tự:

Vẽ đường

Màu của đường vẽ	Ký tự hiển thị	Nét vẽ
y yellow m magenta c cyan r red g green b blue w white k black	. dấu chấm (point) o vòng tròn (circle) x dấu x (x-mark) + dấu cộng (plus) * dấu sao (star) s h×nh vuông (square) d kim cương (diamond) v tam giác chỉ xuống ^ tam giác chỉ lên < tam giác chỉ sang trái > tam giác chỉ sang phải p ông sao 5 cánh (pentagram) h ông sao 6 cánh (hexagram)	- solid : dotted -. dashdot -- dashed

Vẽ đường

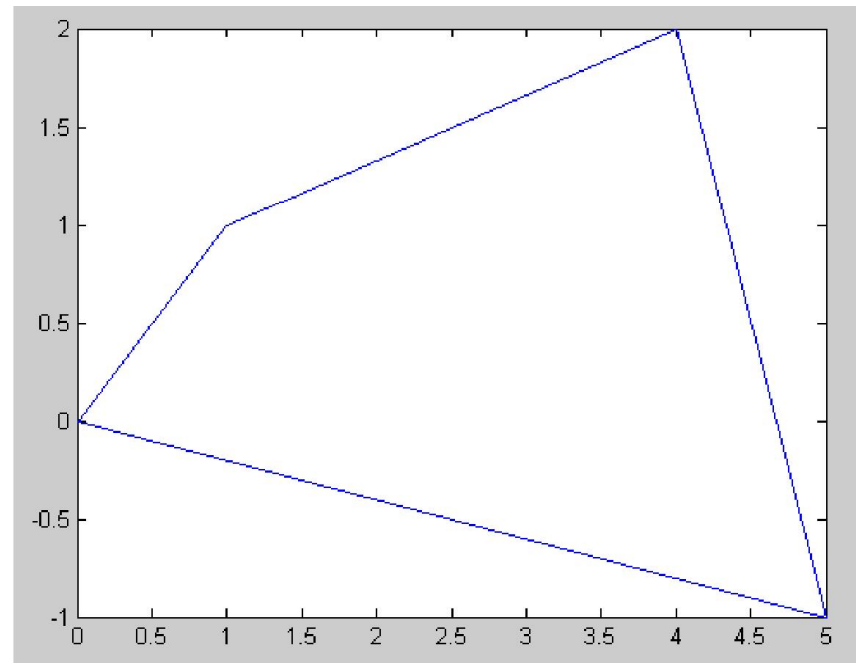
□ Ví dụ.

- Lệnh `plot(x,y,'ro:')` vẽ đường nối bởi các chấm màu đỏ với điểm dữ liệu là vòng tròn nhỏ.
- Lệnh `plot(x,y,'gd')` vẽ các điểm dữ liệu hình dạng kim cương màu xanh lá cây mà không nối các điểm này.
- Lệnh `plot(x,y,'y-',x,y,'go')` vẽ dữ liệu hai lần với đường nối liền nét màu vàng và các điểm dữ liệu là các vòng tròn xanh lá cây.

Vẽ đường

□ Để vẽ tứ giác với các đỉnh $(0,0)$, $(1,1)$, $(4,2)$ và $(5,-1)$ ta có thể sử dụng các lệnh sau

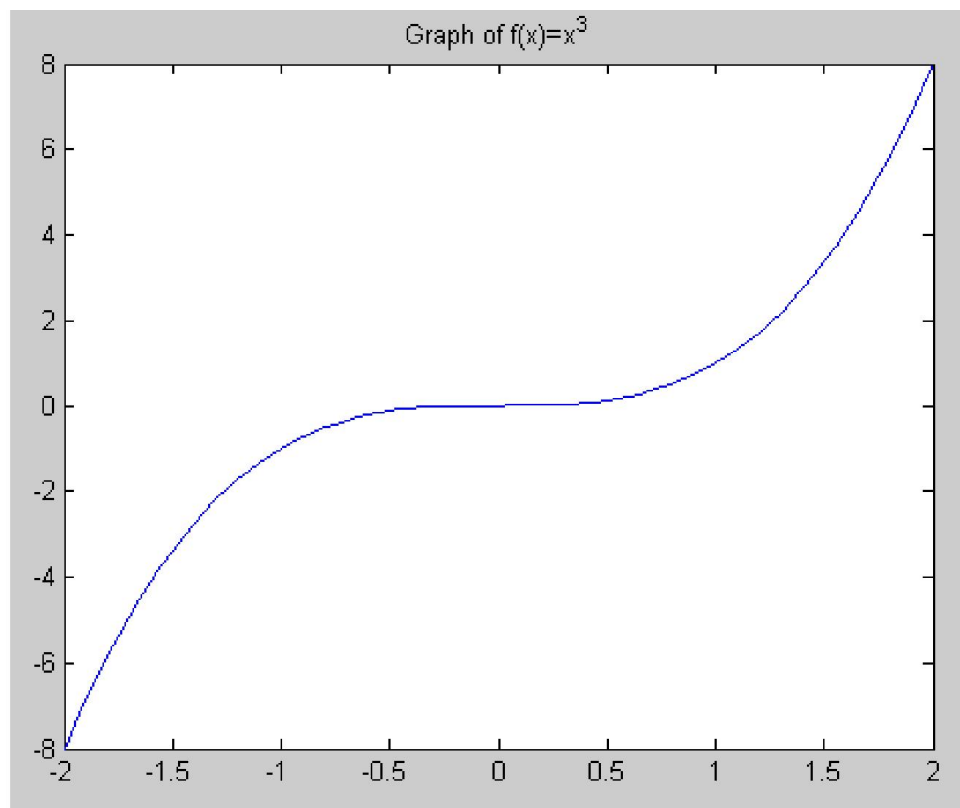
```
>> x=[0 1 4 5 0];  
>> y=[0 1 2 -1 0];  
>> plot(x,y)
```



Vẽ đường

- Ta có thể vẽ gần đúng đồ thị của hàm số nhờ vẽ nhiều điểm trên đồ thị của nó.
- Ví dụ: Vẽ đồ thị của hàm $y=x^3$ trên đoạn $[-2,2]$,
 - Trước hết xác định vectơ lưới chia đoạn $[-2, 2]$ với độ dài bước chia là 0.05
>> $x=-2:.05:2;$
 - Tiếp đến tính vectơ y có kích thước giống x với các thành phần là lũy thừa bậc ba của các thành phần của vectơ x
>> $y=x.^3;$
 - Sử dụng lệnh `plot(x, y)` để vẽ đồ thị của hàm số:
>> `plot(x,y)`
 - Đặt tên cho hình vẽ nhờ đánh lệnh
>> `title('Graph of $f(x)=x^3$ ')`

Đồ thị hàm số $y = x^3$



Tính toán khoa học

Vẽ đường cong tham số

- Việc vẽ các đường cong tham số có thể thực hiện một cách tương tự. Ví dụ: Để vẽ đường $\mathbf{r}(t) = (2t\cos t/(t+1), 2t\sin t/(t+1))$ với $t \in [0, 4\pi]$, trước hết ta xác định vectơ lưới chia t :

```
>> t=0:.1:4*pi;
```

Tiếp đến tính các tọa độ x, y và vẽ đường cong:

```
>> x=2*t.*cos(t)./(t+1);
```

```
>> y=2*t.*sin(t)./(t+1);
```

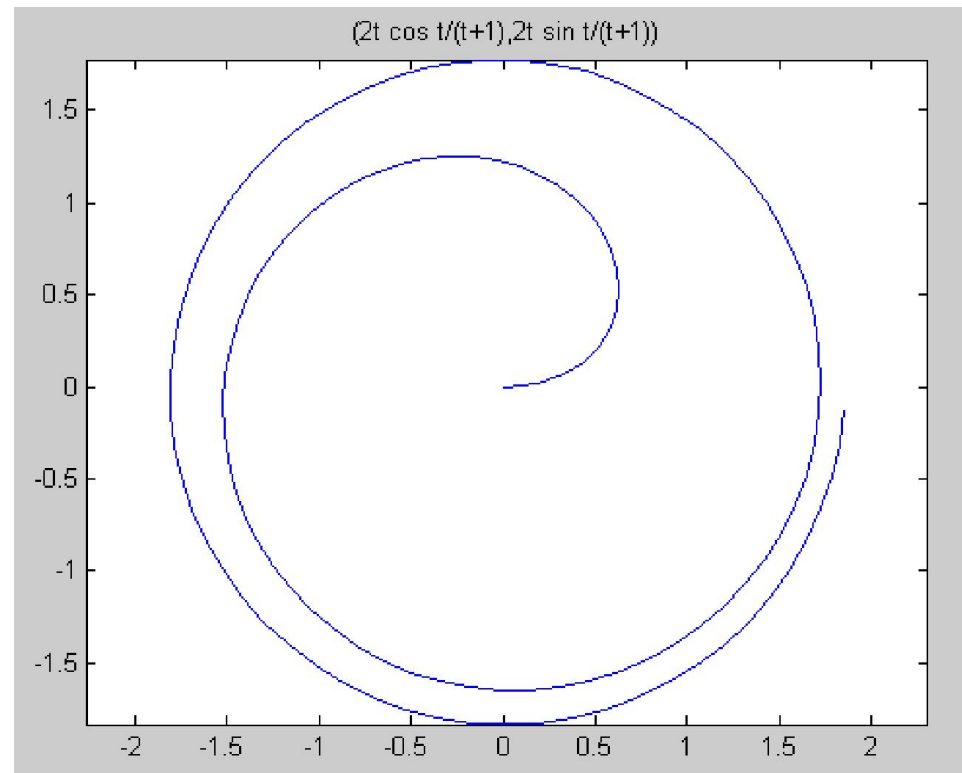
```
>> plot(x,y);
```

```
>> title('(2tcos t/(t+1), 2tsin t/(t+1))')
```

Vẽ đường cong tham số

- MATLAB tự động phóng các trục để đồ thị lấp đầy màn hình. Để có được tỷ lệ chính xác hãy đánh lệnh `axis equal`.

>> axis equal



Vẽ đường

□ Để vẽ nhiều đường trên cùng một hình vẽ, hãy sử dụng lệnh **hold on**.

□ **Ví dụ:** Vẽ hai vòng tròn

$$x^2 + y^2 = 4 \text{ và } (x-1)^2 + (y-1)^2 = 1.$$

□ Phương trình tham số của chúng là

$$\mathbf{r}_1(t) = (2\cos t, 2\sin t)$$

và

$$\mathbf{r}_2(t) = (1 + \cos t, 1 + \sin t) \text{ với } t \in [0, 2\pi].$$

Vẽ đường

```
>> t=0:pi/20:2*pi;  
>> plot(2*cos(t),2*sin(t))  
>> hold on  
>> plot(1+cos(t),1+sin(t))  
>> axis equal  
>> title('The circles  $x^2+y^2=4$  and  $(x-1)^2+(y-1)^2=1$ ')
```

Vẽ đường cong 3D

□ Trong không gian 3 chiều lệnh tương tự như **plot** là **plot3**.

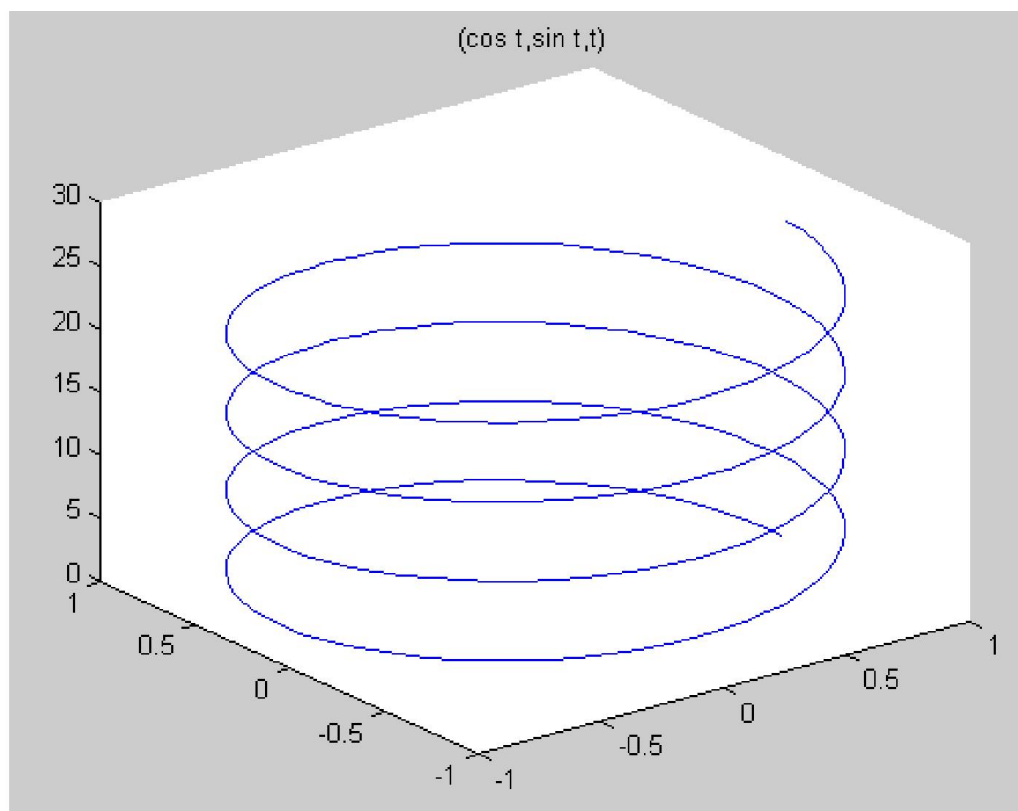
□ **Ví dụ.** Để vẽ đường tham số $\mathbf{r}(t)=(\cos(t),\sin(t),t)$ với $t \in [0,8\pi]$ ta thực hiện các lệnh sau.

```
>> t=0:.1:8*pi;
```

```
>> plot3(cos(t),sin(t),t)
```

```
>> title('(cos t,sin t,t)')
```

Vẽ đường cong 3D



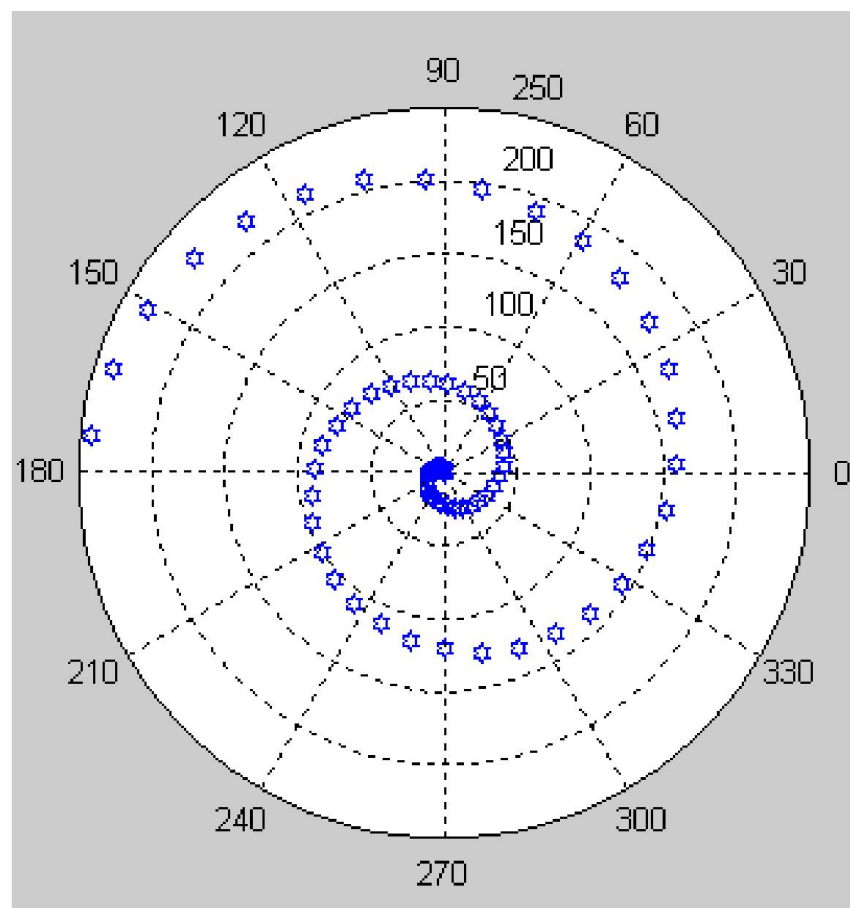
Tính toán khoa học

Vẽ đường cong trong tọa độ cực

□ Ví dụ. Để vẽ đường trong tọa độ cực $\rho = \theta^2$, với $0 \leq \theta \leq 5\pi$ ta có thể đánh các lệnh sau

```
>>theta = 0:0.2:5*pi;  
>>rho=theta.^2;  
>>polar(theta,rho,'hb')
```

Vẽ đường cong trong tọa độ cực



Tính toán khoa học

Vẽ mặt

□ Để vẽ đồ thị của hàm số $f(x,y)$ trên miền hình chữ nhật

$$R = [a,b] \times [c,d] = \{(x,y) \mid a \leq x \leq b; c \leq y \leq d\},$$

trước hết ta cần tạo lưới điểm trong miền khảo sát nhờ hàm **meshgrid**:

$$[X,Y] = \text{meshgrid}(x,y)$$

- tạo lưới chia hình chữ nhật được mô tả bởi vector x, y trong hai mảng X, Y . Các dòng trong X là các bản sao của vector x , các cột trong Y là các bản sao của vector y

Vẽ mặt

- Ví dụ, để chia lưới miền chữ nhật $[0,4] \times [0,3]$ thành các lưới chữ nhật con kích thước rộng 1 cao 0.5, trước hết ta tạo vectơ x và y xác định phép chia lưới.

```
>> x=0 : 4 ;
```

```
>> y=0 : .5 : 3 ;
```

- Tiếp đến dùng `meshgrid` để xác định các điểm trên lưới.

```
>> [X,Y]=meshgrid(x,y)
```

Vẽ mặt

- Giả sử ta cần vẽ đồ thị của hàm số

$$f(x,y)=3x - 2y.$$

- Khi đó ta cần tính ma trận Z gồm các tọa độ theo trục z .

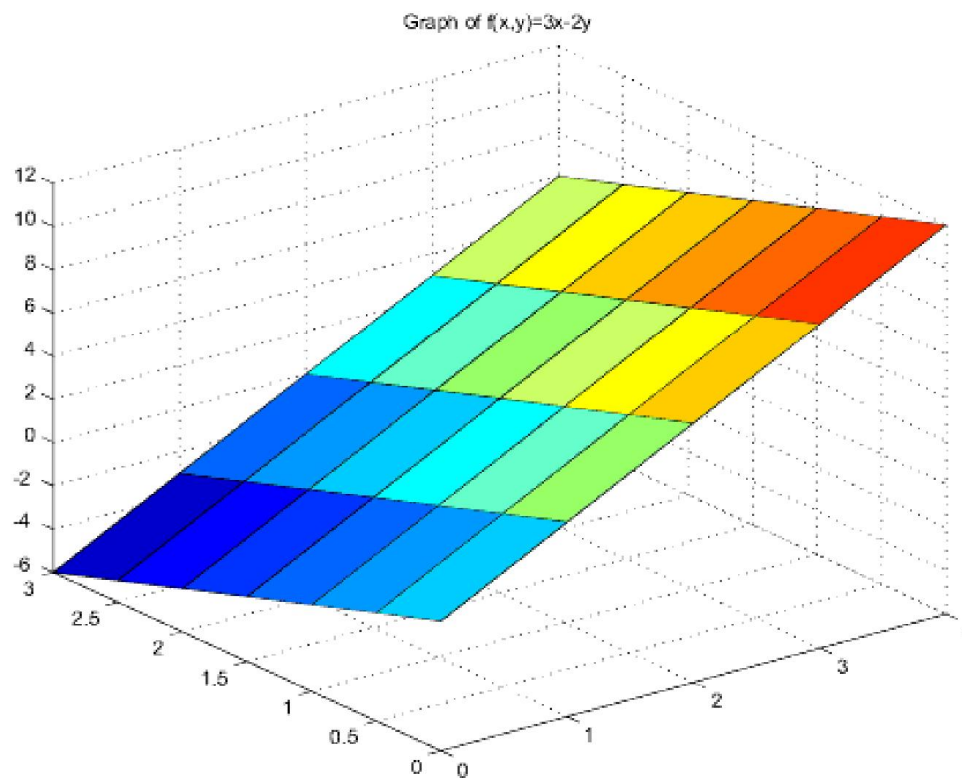
```
>> Z=3*X-2*Y
```

- Cuối cùng sử dụng lệnh **surf** để vẽ mặt.

```
>> surf(X,Y,Z)
```

```
>> title('Graph of  $f(x,y)=3x-2y$ ')
```

Vẽ mặt



Tính toán khoa học

Vẽ mặt

□ Tiếp theo ta xét việc vẽ đồ thị của hàm:

$f(x,y)=x^2y - 2y$ trên miền $[-2,2] \times [-1,1]$.

□ Ta sử dụng lưới vuông với độ dài bước chia 0.1:

```
>> [X,Y]=meshgrid(-2:.1:2,-1:.1:1);
```

□ Sử dụng phép toán vectơ hoá để xác định Z.

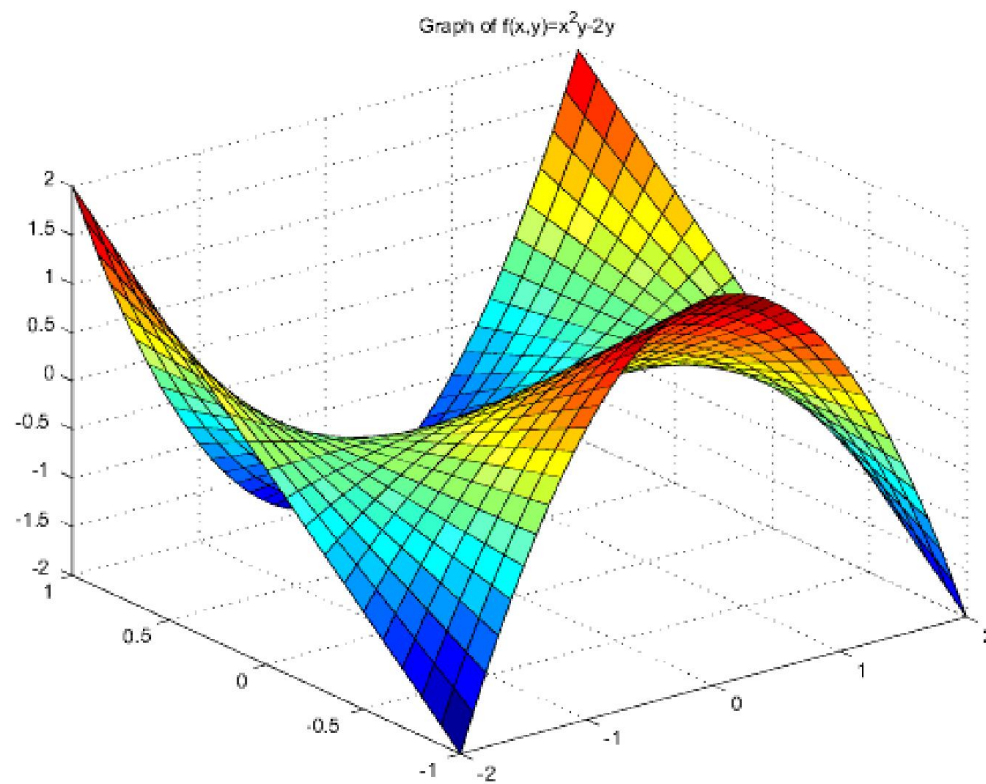
```
>> Z=(X.^2).*Y-2*Y;
```

□ Cuối cùng ta vẽ mặt nhờ lệnh.

```
>> surf(X,Y,Z)
```

```
>> title('Graph of  $f(x,y)=x^2y-2y$ ')
```

Vẽ mặt



Tính toán khoa học

Vẽ mặt

- Một trong những khó khăn khi vẽ mặt là phải đối mặt với phép chia cho 0. Chẳng hạn, ta muốn vẽ đồ thị của hàm số

$$f(x, y) = \frac{xy}{\sqrt{x^2 + y^2}}$$

- trên lưới vuông $[-1, 1] \times [-1, 1]$.

Vẽ mặt

- **Sử dụng các lệnh**

```
>> [X,Y]=meshgrid(-1:.1:1) ;
```

```
>> Z=X.*Y./sqrt(X.^2+Y.^2) ;
```

```
Warning: Divide by zero.
```

□ Lỗi do $(0,0)$ là điểm thuộc lưới chia.

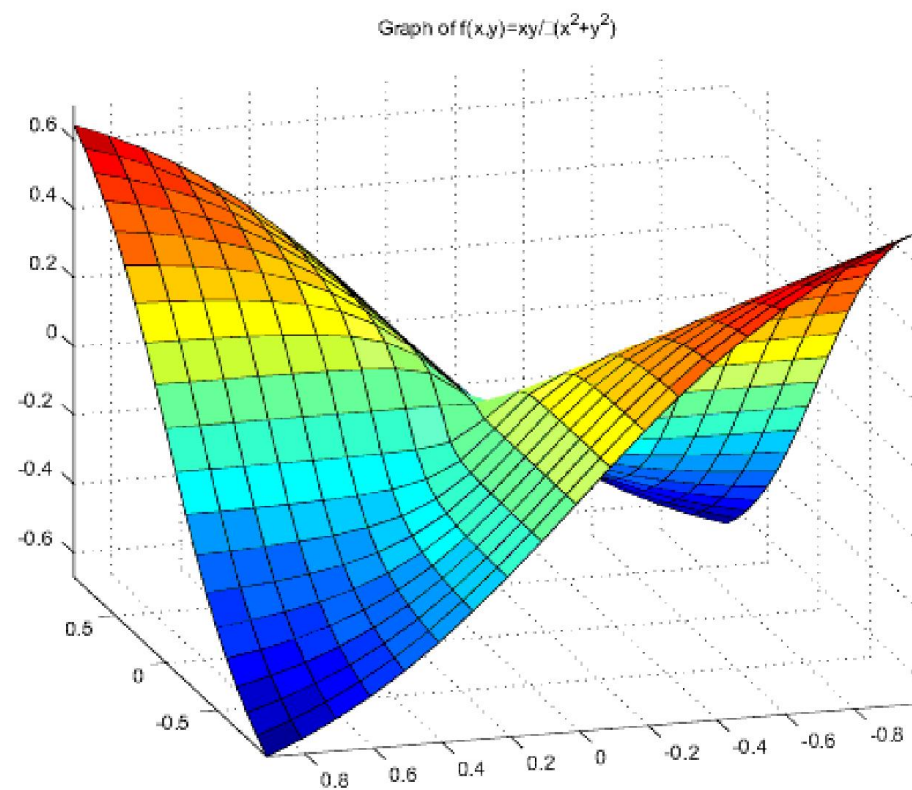
□ Nếu ta cứ vẽ mặt thì điểm của lưới tại $(0,0)$ sẽ khuyết

Vẽ mặt

- Để thoát khỏi tình huống này một cách làm đơn giản là hãy xác định lưới sao cho điểm (0,0) không thuộc lưới điểm chia.
- Chẳng hạn, ta có thể tiến hành như sau

```
>> [X,Y]=meshgrid(-.99:.1:1) ;  
>> Z=X.*Y./sqrt(X.^2+Y.^2) ;  
>> surf(X,Y,Z)  
>> title('Graph of  
f(x,y)=xy/\surd(x^2+y^2) ' )  
>> axis equal
```

Vẽ mặt



Tính toán khoa học

Vẽ mặt

□ Ví dụ. Vẽ

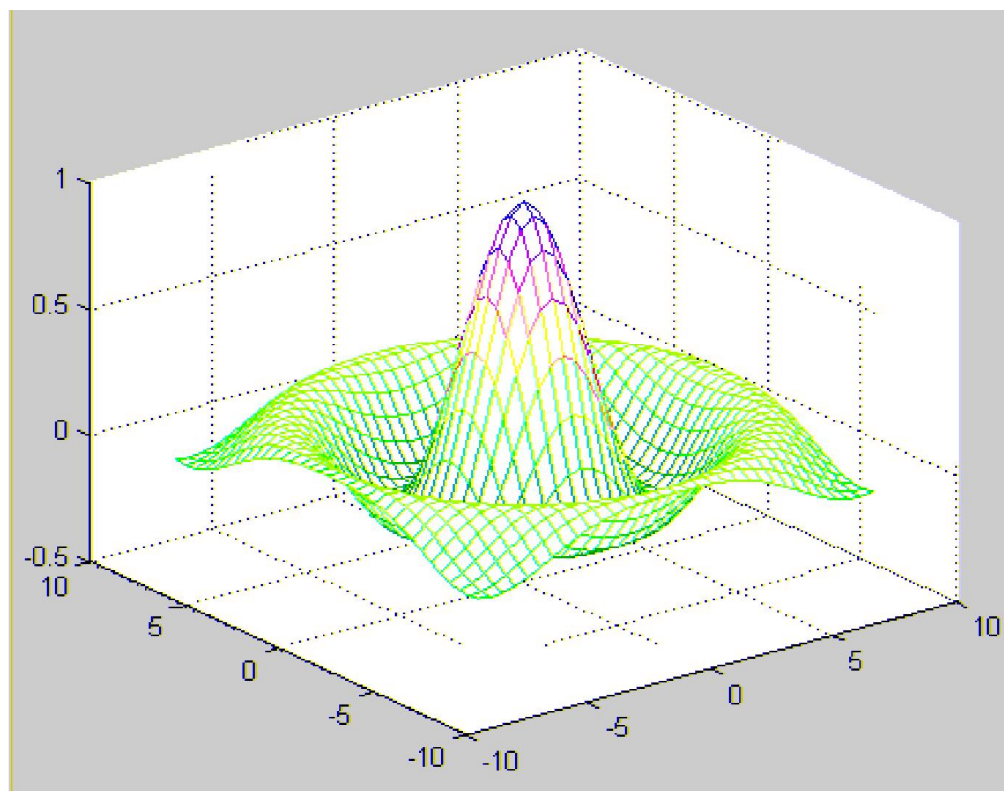
$$z = \sin(r)/r$$

□ với

$$r = \sqrt{x^2 + y^2}, -8 \leq x \leq 8; -8 \leq y \leq 8.$$

```
x = -8:0.5:8; y = -8:0.5:8;  
[X,Y]=meshgrid(x,y) ;  
R=sqrt(X.^2+Y.^2)+eps;  
Z=sin(R) ./R;  
mesh(X,Y,Z)
```

Vẽ mặt



Tính toán khoa học

Vẽ mặt

□ Điểm khác nhau giữa hai lệnh vẽ surf và mesh là: surf tô màu bề mặt, còn mesh thì không.

□ Việc vẽ mặt tham số được tiến hành một cách tương tự. Cho phương trình tham số

$$\mathbf{r}(u,v)=(x(u,v),y(u,v),z(u,v))$$

của mặt, trong đó miền biến thiên của (u,v) là hình chữ nhật, trước hết ta tạo lưới theo các tham số u và v , sau đó xác định các tọa độ x , y và z theo lưới này sử dụng phương trình tham số.

Vẽ mặt

- Ví dụ. Xét mặt cầu bán kính ρ tâm tại gốc tọa độ của không gian R^3 có phương trình tham số

$$r(\phi, \theta) = (\rho \sin \phi \cos \theta, \rho \sin \phi \sin \theta, \rho \cos \phi);$$

$$0 \leq \phi \leq \pi; 0 \leq \theta \leq 2\pi$$

- Giả sử ta cần vẽ mặt cầu đơn vị.
- Trước hết ta tạo lưới theo các tham số ϕ và θ .

```
>> phi=0:pi/20:pi;
```

```
>> theta=0:pi/10:2*pi;
```

```
>> [Phi, Theta]=meshgrid(phi, theta);
```


Vẽ mặt

- Tiếp theo ta sử dụng phương trình tham số với $r = 1$ để tính các tọa độ x , y và z

```
>> X=sin(Phi) .* cos(Theta) ;
```

```
>> Y=sin(Phi) .* sin(Theta) ;
```

```
>> Z=cos(Phi) ;
```

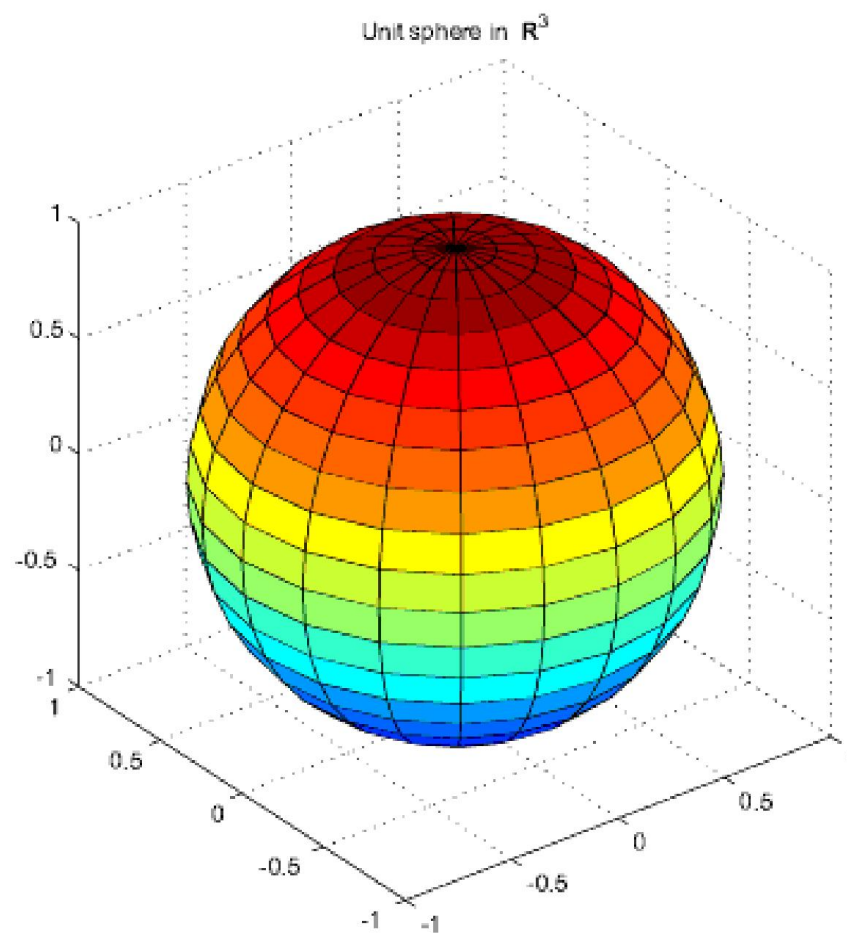
- Cuối cùng ta vẽ mặt và phóng trục sao cho thu được hình ảnh trông giống mặt cầu hơn.

```
>> surf(X,Y,Z)
```

```
>> axis equal
```

```
>> title('Unit sphere in {\bf R}^3')
```

Vẽ mặt



Tính toán khoa học

Đa thức trong Matlab

- Đa thức bậc n

$$P = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$$

- Biểu diễn bằng vector hàng $n+1$ phần tử
 - Sắp xếp hệ số từ bậc cao nhất đến bậc 0
 - Ví dụ $p = 3x^4 + x^2 - 3x + 5$
 - `>>p=[3 0 1 -3 5]`
 - `p= 3 0 1 -3 5`

Các phép tính với đa thức

HÀM	Ý NGHĨA
<code>conv(p1,p2)</code>	Nhân hai đa thức
<code>[k,d]=deconv (p1,p2)</code>	Chia hai đa thức (k= kết quả; d =phần dư)
<code>k=polyder(p)</code>	Tìm đạo hàm của đa thức p
<code>k=polyder(p,q)</code>	Tìm đạo hàm của đa thức tích ($p \cdot q$)
<code>[n,d]=polyder(num,den)</code>	Tìm đạo hàm (dạng n/d) của phân thức (num/den)
<code>roots(p)</code>	Tìm nghiệm đa thức p
<code>p=poly(r)</code>	Lập đa thức p từ vector r chứa các nghiệm.
<code>polyval(p,x)</code>	Tính giá trị của đa thức tại x (x có thể là mảng)
<code>[r,p,k]= residue(num,den)</code>	Tìm các thành phần tối giản của phân thức
<code>[num,den]=residue(r,p,k)</code>	Chuyển các thành phần tối giản thành 1 phân thức
<code>printsys(num,den,'s')</code>	in phân thức có dạng tỉ số 2 đa thức theo s
<code>[z,p,k]=tf2zp(num,den)</code>	Tìm các zero z, cực p, độ lợi k của phân thức

Các phép tính với đa thức

- Ví dụ 1:

```
>> r=[1 3];
```

```
>> p=poly(r)
```

```
p = 1 - 4 3
```

```
>> polyval(p,[1 3 5])
```

```
ans =
```

```
0 0 8
```

```
>> printsys(r,p,'x')
```

```
num/den =
```

```
x + 3
```

```
-----
```

```
x^2 - 4 x + 3
```

- Ví dụ 2:

$$\frac{x^4 + 3x - 14}{x^2 - 4} = \frac{(x^4 - 16 + 3x + 2)}{x^2 - 4}$$

$$= x^2 + 4 + \frac{1}{x+2} + \frac{2}{x-2}$$

$$k(x) = x^2 + 4$$

$$r = [1 \ 2]$$

$$p = [-2 \ 2]$$

Các phép tính với đa thức

Ví dụ 3:

$$G(s) = \frac{4s^2 + 16s + 12}{s^4 + 12s^3 + 44s^2 + 48s}$$

```
>> num=[4 16 12]
>> den=[1 12 44 48 0]
>> [z,p,k]=tf2zp(num,den)
z =
    -3
    -1
p =
     0
 -6.0000
 -4.0000
 -2.0000
k =
     4
```

Do đó :

$$G(s) = \frac{K(s - z_1)(s - z_2)}{(s - p_1)(s - p_2)(s - p_3)} = \frac{4(s + 3)(s + 1)}{(s + 6)(s + 4)(s + 2)}$$

Vào-Ra dữ liệu

- **Vào dữ liệu từ bàn phím**
 - Các lệnh liên quan đến nhập dữ liệu từ bàn phím: `input`, `keyboard`, `menu` và `pause`.
- **Lệnh `input`:** Lệnh này đưa ra thông báo nhắc người sử dụng nhập dữ liệu và nhận dữ liệu đánh từ bàn phím. Lệnh có dạng

`R = input(string)`

trong đó `R` là tên biến, *string* là chuỗi ký tự chứa thông báo nhắc người sử dụng biết về dữ liệu cần nhập. Dữ liệu có thể là biểu thức bất kỳ. Nếu người sử dụng ấn Enter ngay thì `R` sẽ là ma trận rỗng.

Vào dữ liệu từ bàn phím

- Ví dụ:

```
>> m=input('Nhap so dong cua ma tran: ')
```

```
Nhap so dong cua ma tran: 3
```

```
m =
```

```
3
```

```
>> n = input('Hay nhap so cot cua ma tran  
n =')
```

```
Hay nhap so cot cua ma tran n =5
```

```
n =
```

```
5
```


Vào dữ liệu từ bàn phím

- Ví dụ

```
>> a =input('Hay nhap cac phan tu cua ma tran a  
theo khuan dang:\n [Cacphan tu dong 1 ;\n ...  
\n Cac phan tu dong m] : \n\n')
```

Hay nhap cac phan tu cua ma tran a theo khuan dang:

[Cacphan tu dong 1 ;

...

Cac phan tu dong m] :

[1 2 3 4 5;

6 7 8 9 10;

11 12 13 14 15]

Vào-Ra dữ liệu

- **Lệnh keyboard**

- Khi lệnh này được đặt trong m – file thì nó ngắt việc thực hiện chương trình.
- Màn hình xuất hiện dấu nhắc >>K.
- Người sử dụng có thể xem hoặc biến đổi giá trị các biến bằng các lệnh của Matlab.
- Chấm dứt chế độ dùng **Enter**.
- Tiện lợi cho việc gỡ rối (debug) chương trình.

Vào-Ra dữ liệu

- **Lệnh menu:** Cho phép tạo bảng lựa chọn. Lệnh có dạng sau.

CHOICE = menu(HEADER, ITEM1, ..., ITEMn)

- Lệnh sẽ hiển thị tiêu đề của bảng chọn chứa trong chuỗi HEADER, tiếp đến là dãy các lựa chọn trong các chuỗi: ITEM1, ..., ITEMn.
 - Chỉ số của lựa chọn sẽ được trả lại cho biến CHOICE.
- Trong màn hình đồ họa Matlab sẽ hiển thị bảng lựa chọn trong một cửa sổ MENU với các phím ấn chọn.

Vào-Ra dữ liệu

- Ví dụ:

```
>> CHOICE = menu('Hay chon cach vao du lieu: ', 'Keyboard', 'File', 'Random ')
```

- Trên màn hình đồ họa hiển thị bảng lựa chọn:



- Người sử dụng chọn bằng việc click chuột vào lựa chọn cần thiết.

Vào-Ra dữ liệu

- Cách sử dụng khác của menu

CHOICE = menu(HEADER, ITEMLIST)

trong đó ITEMLIST là mảng chuỗi.

- Ví dụ:

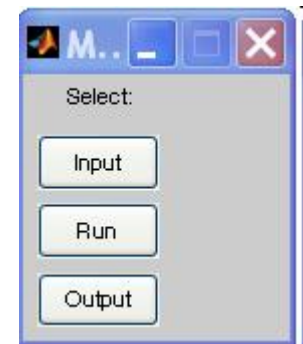
```
>> ItemList={'Input', 'Run', 'Output'};
```

```
>> HEADER='Select:';
```

```
>> CHOICE = MENU(HEADER, ItemList)
```

```
CHOICE =
```

```
1
```



Vào-Ra dữ liệu

- ☐ **Lệnh pause:** Sử dụng để chờ trả lời của người sử dụng.

pause(n)

trong đó n có thể là số không nguyên. PAUSE(n) sẽ dừng n seconds rồi mới tiếp tục.

- ☐ Lệnh pause không có tham số sẽ dừng cho đến khi người sử dụng gõ một phím.

- ☐ Lệnh

pause off

chỉ thị rằng tất cả các lệnh PAUSE or PAUSE(n) đi sau nó sẽ không có tác dụng.

- ☐ Lệnh

pause on

chỉ thị rằng các lệnh PAUSE sau nó sẽ có tác dụng ngắt.

Đưa ra màn hình

- **Đưa ra màn hình nội dung mảng**
 - Gõ tên biến không có ; cuối lệnh

- **Ví dụ:**

```
>> x = [1 2 3] ;
```

```
>> y=[2; 3; 4] ;
```

```
>> x*y
```

```
ans =
```

```
20
```

Đưa ra màn hình

- **Lệnh disp:**

- Dùng để hiển thị nội dung của biến ra màn hình và không hiển thị tên biến
- Làm việc dù có ; hay không
- Lệnh có dạng

disp(X)

với X là biến hoặc biểu thức cần hiển thị.

- **Ví dụ:**

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

```
>> disp(B^2) ;
```

```
    30    36    42
    66    81    96
   102   126   150
```

```
>> disp(B.^2) ;
```

```
     1     4     9
    16    25    36
    49    64    81
```


Đưa ra màn hình

- **Ví dụ:** Lập bảng tính giá trị hàm sin

```
clc
n = input('Give number of point n = ');
x = linspace(0,1,n);
y = sin(2*pi*x);
disp(' ')
disp('!      k      !      x(k) !      sin(x(k))      !')
disp('-----')
for k=1:n    degrees = (k-1)*360/(n-1);
disp(sprintf('!      %2.0f      !      %3.0f      !      %6.3f\n',k,degrees,y(k)));
end
disp(' ');
disp('x(k) is given in degrees.')
disp(sprintf('One Degree = %5.3e Radians',pi/180))
```

Đưa ra màn hình

- Kết quả thực hiện có thể có dạng:

Give number of point $n = 10$

!	k	!	x(k)	!	sin(x(k))	!

!	1	!	0	!	0.000	!
!	2	!	40	!	0.643	!
!	3	!	80	!	0.985	!
!	4	!	120	!	0.866	!
!	5	!	160	!	0.342	!
!	6	!	200	!	-0.342	!
!	7	!	240	!	-0.866	!
!	8	!	280	!	-0.985	!
!	9	!	320	!	-0.643	!
!	10	!	360	!	-0.000	!

x(k) is given in degrees.

One Degree = 1.745e-002 Radians

Đưa ra màn hình

- **Lệnh fprintf**
- Lệnh có dạng

`fprintf(dialog_format, danh_sach_bien)`

trong đó **dialog_format** là biến (hằng) chuỗi ký tự thông báo và các ký tự định khuôn dạng dữ liệu ra.

Đưa ra màn hình

- Ví dụ:

```
>> A = rand(3,3)
```

```
A =
```

```
    0.1389    0.6038    0.0153  
    0.2028    0.2722    0.7468  
    0.1987    0.1988    0.4451
```

```
>> fprintf('Length of matrix A: %i%i',length(A))
```

```
Length of matrix A: 3
```

```
>> fprintf('Square of A:\n  
    %i%i%i\n%i%i%i\n%i%i%i\n',A^2)
```

```
Square of A:
```

```
    1.447541e-0012.317550e-0011.563636e-001  
    2.512430e-0013.449860e-0012.625931e-001  
    4.598234e-0015.387547e-0013.496177e-001
```

Vào ra với file văn bản

- Matlab cung cấp các lệnh cho phép làm việc với các file
- Ở đây chỉ hạn chế trình bày các thao tác với file văn bản
- Các lệnh chính liên quan đến làm việc với file văn bản của Matlab là:
 - fopen, fclose
 - frewind, fread, fwrite
 - fscanf, fprintf..

Vào ra với file văn bản

- **Lệnh fopen.** Dùng để mở file

fileID = fopen(FILENAME, PERMISSION)

- Lệnh này mở file văn bản có tên cho bởi FILENAME (hằng xâu ký tự hoặc biến xâu)
- Chế độ làm việc được chỉ ra bởi PERMISSION.
- PERMISSION có thể là:
 - ‘rt’ mở file để đọc
 - ‘wt’ mở file để ghi (nếu chưa có file sẽ tạo ra file mới có tên FILENAME)
 - ‘at’ mở file nối đuôi (tạo file mới có tên FILENAME nếu chưa có)
 - ‘rt+’ mở file để đọc và ghi (không tạo file mới)
- Biến nhận dạng *fileID* nhận giá trị nguyên và ta sẽ sử dụng nó để thâm nhập vào file.

Vào ra với file văn bản

- Trong trường hợp muốn kiểm tra lỗi mở file có thể sử dụng lệnh

`[FID, MESSAGE] = fopen(FILENAME, PERMISSION)`

nếu gặp lỗi mở file biến MESSAGE sẽ chứa thông báo lỗi.

Vào ra với file văn bản

- Ví dụ:

```
>> [fileID, MESSAGE] = fopen('ETU.txt','rt');  
>> fileID  
fileID =  
    -1  
  
>> MESSAGE  
MESSAGE =  
No such file or directory
```


Vào ra với file văn bản

- **Lệnh đóng file fclose:** Đóng file làm việc.

ST = fclose(FID)

- Lệnh này sẽ đóng file ứng với biến nhận dạng FID
- fclose trả lại biến ST giá trị 0 nếu nó hoàn thành việc đóng file và -1 nếu gặp lỗi.
- Lệnh ST=fclose('all') đóng tất cả các file đang mở ngoại trừ 0, 1, 2.

- **Lệnh frewind.**

frewind(FID)

đặt con trỏ file của FID vào đầu file.

Vào ra với file văn bản

- **Lệnh fscanf:** Đọc dữ liệu vào từ file

[A, COUNT] = fscanf(FID, FORMAT, SIZE)

ě Lệnh này đọc dữ liệu từ file tương ứng với FID

ě Chuyển đổi dữ liệu về khuôn dạng được xác định bởi biến xâu FORMAT

ě Gán giá trị cho mảng A

ě COUNT là biến ra tùy chọn dùng để chứa số lượng phần tử đọc được.

ě SIZE là biến tùy chọn chỉ giới hạn số phần tử được đọc vào từ file, nếu vắng mặt thì toàn bộ file được xét. Các giá trị có thể có của biến là:

ě N : đọc không quá N phần tử từ file vào vector cột

ě Inf : đọc không quá kết thúc file

ě [M, N] : đọc không quá M*N phần tử và đưa vào ma trận kích thước không quá MxN theo từng cột, N có thể là inf nhưng M thì phải là hữu hạn.

Vào ra với file văn bản

- Nếu ma trận A là kết quả của việc chuyển khuôn dạng ký tự và biến SIZE không có dạng [M, N] thì vector dòng sẽ được trả lại.
- FORMAT là biến chứa các ký tự chuyển đổi khuôn dạng của ngôn ngữ C.
- Các ký tự định khuôn dạng bao gồm: %, các ký hiệu thay thế, độ dài trường, các ký tự chuyển đổi khuôn dạng: d, i, o, u, x, e, f, g, s, c và [...] (liệt kê tập hợp).
- Nếu %s được sử dụng thì khi đọc một phần tử có thể dẫn đến phải sử dụng một loạt các thành phần của ma trận, mỗi thành phần giữ một ký tự.
- Hãy sử dụng %c để đọc ký tự trắng (space) và khuôn dạng %s bỏ qua các ký tự trắng.

Vào ra với file văn bản

- Nếu chỉ thị định dạng gồm cả số lẫn ký tự thì ma trận kết quả sẽ là ma trận số và mỗi ký tự sẽ được chuyển thành số bằng giá trị mã ASCII của nó.
- fscanf khác với lệnh này trong C ở chỗ nó là lệnh vector hóa trả lại đối số là ma trận.
- Biến xâu định dạng sẽ được sử dụng lặp lại cho đến khi gặp kết thúc file hoặc đọc đủ số lượng phân tử chỉ ra bởi SIZE.

Vào ra với file văn bản

- Ví dụ:

- Lệnh

S = fscanf(fid,'%s')

đọc (và trả lại) một xâu.

- Lệnh

A = fscanf(fid,'%5d')

đọc các số nguyên có 5 chữ số thập phân.

Vào ra với file văn bản

- **Ví dụ:** Giả sử có file văn bản với tên “kq” chứa xâu “Day la ket qua dua ra”. Khi đó ta có thể đọc dữ liệu vào như sau:

```
>> fid=fopen('kq','rt')
fid =
      3
>> x = fscanf(fid,'%s')
x =
Daylaketquaduara
>> frewind(fid)
>> x = fscanf(fid,'%s%c')
x =
Day la ket qua dua ra
```

Vào ra với file văn bản

- **Ví dụ:** Giả sử file văn bản Matrix.txt chứa hai dòng

```
1 2 3 4 5
6 7 8 9 10
```

- Hãy theo dõi kết quả làm việc của các lệnh sau để thấy tác động của lệnh fscanf

```
>> fopen('matrix.txt','rt')
```

```
ans =
```

```
4
```

```
>> A=fscanf(4,'%i',[2,5])
```

```
A =
```

```
1      3      5      7      9
2      4      6      8     10
```

Vào ra với file văn bản

```
>> frewind(4);  
>> B = fscanf(4, '%i', [5,2])
```

B =

1	6
2	7
3	8
4	9
5	10

```
>> frewind(4);  
>> C = fscanf(4, '%i', 6)
```

C =

1
2
3
4
5
6

Vào ra với file văn bản

- **Lệnh fprintf** : ghi dữ liệu theo khuôn dạng ra file.

COUNT = FPRINTF(FID, FORMAT, A, . . .)

- Lệnh này sẽ định dạng các thành phần của ma trận A (và các biến tiếp theo trong danh sách) theo khuôn dạng được xác định bởi biến xâu format, và ghi ra file tương ứng với biến nhận dạng FID.
- Biến:
 - COUNT đếm số byte dữ liệu được ghi ra.
 - FID là số nguyên nhận dạng tên file thu được từ lệnh fopen. Có thể dùng số 1 nếu sử dụng thiết bị ra chuẩn (màn hình). Nếu lệnh này không có FID thì mặc định đưa kết quả ra màn hình.
 - FORMAT là biến xâu chứa các ký tự mô tả định dạng dữ liệu giống như trong ngôn ngữ C
 - Các ký tự \n, \r, \t, \b, \f có thể dùng để tạo linefeed, carriage return, tab, backspace, và formfeed character tương ứng.
 - Sử dụng \\ để tạo dấu \ và %% để tạo ký tự %.

Vào ra với file văn bản

- Ví dụ:

```
x = 0:.1:1; y = [x; exp(x)];  
fid = fopen('exp.txt','w');  
fprintf(fid,'%6.2f   %12.8f\r',y);  
fclose(fid);
```

Tạo ra file văn bản có tên 'exp.txt' chứa bảng giá trị của hàm mũ.

```
0.00 1.000000000  
0.10 1.10517092  
0.20 1.22140276  
0.30 1.34985881  
0.40 1.49182470  
0.50 1.64872127  
.  
.  
.
```