



# MATLAB LÝ THUYẾT VÀ ỨNG DỤNG



Giảng viên: Hoàng Xuân Dương



## TÀI LIỆU THAM KHẢO

1. Bài giảng tin học chuyên ngành  
*Giảng viên: Hoàng Xuân Dương*
2. Matlab & Simulink dành cho kỹ sư điều khiển tự động  
*Nguyễn Phùng Quang*
3. An Introduction to Matlab  
*University of DUNDEE*
4. Electronics and circuit analysis using Matlab  
*John O.Attia*
5. Matrix analysis of circuits using Matlab  
*James G.Gottling*
6. Matlab tools for Control system analysis and design  
*Duane C.Hanelman, Benjamin C.Kuo*

Giảng viên: Hoàng Xuân Dương



## 1. MATLAB LÀ GÌ ?

- Matlab (Matrix Laboratory) là **một công cụ phần mềm** của The Mathworks Ins, ban đầu phục vụ chủ yếu việc mô tả kỹ thuật bằng **toán học** với các phần tử cơ bản là **ma trận**
- **Các dữ liệu rời rạc** (discret) (trong các lĩnh vực điện, điện tử, vật lý hạt nhân, điều khiển tự động..., ngành toán như thống kê, kế toán,..., gien sinh học, khí hậu, thời tiết...) có thể lưu dưới dạng ma trận
- **Dữ liệu liên tục** như âm thanh, hình ảnh, dòng điện, điện áp, tần số, áp suất,... **chuyển đổi thành các tín hiệu số** → được xử lý bằng các hàm toán học của Matlab

Giảng viên: Hoàng Xuân Dương

## 2. ƯU ĐIỂM CỦA MATLAB

- Matlab cung cấp một công cụ tính toán và lập trình bậc cao dễ sử dụng, hiệu quả và thân thiện. Simulink giúp người dùng dễ dàng thực hiện các bài toán mô hình hóa, mô phỏng trên máy tính
- Matlab có tính mở, các **hàm** và các **toolbox** không ngừng được bổ sung theo sự phát triển của khoa học bởi chính The Mathworks Ins và cả người sử dụng trên toàn thế giới
- Có công cụ trợ giúp phong phú trực tuyến, trên mạng hay các tài liệu dạng pdf

Giảng viên: Hoàng Xuân Dương

### 3. SỨC MẠNH CỦA MATLAB ?

- ✓ Môi trường phát triển: gồm các công cụ và tiện nghi giúp viết chương trình, sử dụng các hàm **Matlab** và các file
- ✓ Thư viện các hàm toán học của **Matlab**: Các hàm sơ cấp: tổng, sin, tính số phức... các hàm phức tạp: Bessel, nghịch đảo ma trận, tính trị riêng, biến đổi Fourier nhanh, wavelet...
- ✓ Ngôn ngữ **Matlab**: Các lệnh cao cấp xử lý ma trận, lệnh rẽ nhánh, vòng lặp, xuất nhập, cấu trúc dữ liệu, lập trình hướng đối tượng...
- ✓ Xử lý đồ họa: Hiển thị dữ liệu dạng đồ họa 2D, 3D, hoạt hình, xử lý ảnh và cả GUI

Giảng viên: Hoàng Xuân Dương

### 3. SỨC MẠNH CỦA MATLAB (tt)

- ✓ Thư viện **API** của Matlab: Cho phép liên kết các chương trình C và Fortran... Các ngôn ngữ khác có thể gọi các hàm dll được tạo bởi Matlab.
- ✓ Các hộp công cụ (**Toolbox**): Tập hợp các hàm Matlab được viết sẵn để giải quyết các vấn đề thuộc các chuyên ngành khác nhau. Các **toolbox** khiến cho Matlab có thể ứng dụng vào nhiều lĩnh vực khác nhau: Điện tử, Điều khiển tự động, Kỹ thuật điện, Viễn thông, Cơ khí, Động lực...

Giảng viên: Hoàng Xuân Dương

#### 4. AI CÓ THỂ HỌC VÀ SỬ DỤNG MATLAB ?

- Các nhà chuyên môn, cán bộ nghiên cứu giảng dạy
- Các sinh viên theo học các trường Đại học và trung học chuyên nghiệp...
- Các kỹ sư, cán bộ kỹ thuật

.....

**CHƯƠNG 1: CÁC KHÁI NIỆM CƠ BẢN**

**CHƯƠNG 2: MA TRẬN VÀ CÁC PHÉP TOÁN MA TRẬN**

**CHƯƠNG 3: LẬP TRÌNH TRONG MATLAB**

**CHƯƠNG 4: XỬ LÝ CÁC HÀM TOÁN HỌC**

**CHƯƠNG 5: ĐỒ HỌA MATLAB**

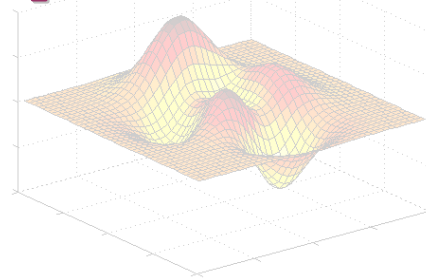
**CHƯƠNG 6: SIMULINK VÀ ỨNG DỤNG**

**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**



## CHƯƠNG 1:

# CÁC KHÁI NIỆM CƠ BẢN



Hoàng Xuân Dương



## CHƯƠNG 1: CÁC KHÁI NIỆM CƠ BẢN

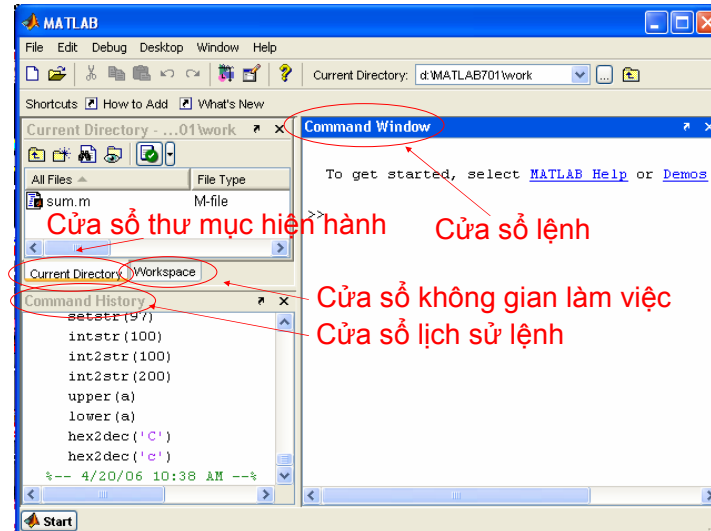
10

- I. HOẠT ĐỘNG CỦA MATLAB
- II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN
- III. SƠ LƯỢC VỀ ĐỒ HỌA TRONG MATLAB
- IV. ÂM THANH TRONG MATLAB



## I. HOẠT ĐỘNG CỦA MATLAB:

Khi chạy Matlab, một màn hình nền xuất hiện:



## I. HOẠT ĐỘNG CỦA MATLAB:

- **Cửa sổ lệnh (Command window):**  
Với dấu nhắc `>>` dùng để chạy các lệnh, viết chương trình, chạy chương trình.
- **Cửa sổ Lịch sử lệnh (Command history)**  
Liệt kê tất cả các lệnh đã sử dụng trước đó kèm theo thời gian làm việc
- **Cửa sổ thư mục hiện tại (Current Directory)**  
Cho biết thư mục hiện tại đang làm việc. Mặc định khi cài đặt là MATLAB701\work (version 7.01)
- **Cửa sổ không gian làm việc (workspace)**  
Cho biết các biến được sử dụng trong chương trình

**CHƯƠNG 1: CÁC KHÁI NIỆM CƠ BẢN**

**I. HOẠT ĐỘNG CỦA MATLAB:**

**1. Các phép toán đơn giản:**

13

The screenshot displays the MATLAB environment. The **Workspace** window lists three variables: `ans`, `gia`, and `tap`, each with a size of `1x1` and class `double array`. The **Command Window** shows the following commands and their outputs:

```
>> 4+6+2
ans =
    12
>> 4*2+6*3+2*5
ans =
    36
>> tap=4
tap =
     4
>> gia=tap*5;
>> gia
gia =
    20
```

The **Command History** window shows the same sequence of commands: `4+6+2`, `4*2+6*3+2*5`, `tap=4`, `gia=tap*5;`, and `gia`.

**CHƯƠNG 1: CÁC KHÁI NIỆM CƠ BẢN**

**I. HOẠT ĐỘNG CỦA MATLAB:**

**2. Một số lệnh hệ thống:**

14

| Lệnh                     | Ý nghĩa                                |
|--------------------------|--|
| <code>clc</code>         | xóa cửa sổ lệnh                        |
| <code>clf</code>         | xóa cửa sổ đồ họa                      |
| <code>help</code>        | xem phần trợ giúp một số lệnh          |
| <code>quit, exit</code>  | thoát Matlab                           |
| <code>Ctrl+c</code>      | dừng chương trình                      |
| <code>pause</code>       | ngừng tạm thời chương trình            |
| <code>edit</code>        | gọi chương trình soạn thảo             |
| <code>type</code>        | đọc nội dung file .m                   |
| <code>input</code>       | nhập dữ liệu từ bàn phím               |
| <code>demo</code>        | Gọi chương trình demo                  |
| <code>echo on/off</code> | Tắt mở hiển thị các lệnh trong M-files |

## II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN

### 1. Biến trong Matlab:

- Tên biến có thể dài 31 ký tự, bắt đầu là chữ
- Matlab phân biệt chữ thường và chữ hoa
- Sử dụng dấu = để định nghĩa biến
- Tên biến có thể trùng với tên hàm có sẵn, khi đó hàm không còn sử dụng được cho đến khi biến được xóa

Ví dụ:

```
>> x=1
```

```
x=1
```

```
>> ten_truong='Dai hoc DL Cong Nghe Sai Gon'
```

```
ten_truong = Dai hoc DL Cong Nghe Sai Gon
```

## II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN

### 1. Biến trong Matlab (tt)

Một số hàm liên quan đến biến:

| Lệnh                   | Ý nghĩa  |
|------------------------|--|
| clear                  | xóa tất cả các biến  |
| who                    | hiển thị danh sách các biến trong workspace                    |
| whos                   | hiển thị các biến cùng kích thước của chúng, có phải số phức ? |
| clear name1, name2,... | xóa biến có tên được khai báo                                  |
| exist ('item')         | Kiểm tra sự tồn tại của đối tượng 'item'                       |
| save                   | Lưu các biến trong workspace ra file                           |
| load                   | Tải các biến vào trong workspace từ file                       |



## II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN

### 2. Độ lớn của biến:

Xác định độ lớn hay chiều dài của biến vector cũng như ma trận thông qua một số hàm:

| Hàm           | Ý nghĩa  |
|---------------|--|
| size(A)       | Trả về 1 vector chứa kích thước A, gồm số hàng và số cột của A |
| [m n]=size(A) | giá trị trả về chứa trong m và n                               |
| size(A,p)     | p=1 → trả về số hàng<br>p=2 → trả về số cột                    |
| length(A)     | Trả về chiều dài của A, giá trị lớn nhất của hàng và cột       |

## II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN

### 2. Độ lớn của biến (tt)

Ví dụ:

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

```
A= 1 2 3
```

```
    4 5 6
```

```
>> [m n]=size(A)
```

```
m = 2
```

```
n = 3
```

```
>> length(A)
```

```
ans = 3
```

```
>> size(A,1)
```

```
ans = 2
```

## II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN

### 3. Một số biến được định nghĩa trước:

Một số biến được Matlab sử dụng để chỉ các hằng số hay ký hiệu, nên tránh dùng chúng:

```
>> 1/0
```

Warning: Divide by zero.

(Type "warning off MATLAB:divideByZero" to suppress this warning.)

```
ans = Inf
```

```
>> 0/0
```

Warning: Divide by zero.

(Type "warning off MATLAB:divideByZero" to suppress this warning.)

```
ans = NaN
```

```
>> eps
```

```
ans = 2.2204e-016
```

## II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN

### 3. Một số biến được định nghĩa trước (tt)

| Ký hiệu   | Ý nghĩa   |
|-----------|---|
| =         | Gán giá trị cho biến                                    |
| + - * / ^ | Các phép tính   |
| ;         | Nhập giá trị, dấu cách khi nhập nhiều trị trên một dòng |
| ,         | Dấu cách khi xuất nhiều giá trị trên một dòng           |
| ans       | Đáp số mới nhất   |
| eps       | Cấp chính xác tương đối khi dùng dấu phẩy động          |
| pi        | số $\pi = 3,14159265...$                                |
| i j       | Toán tử ảo  |
| inf       | vô cùng   |
| NaN       | không phải số (0/0 hay inf/inf)                         |

## II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN

### 4. Số phức:

- Các hàm đặc biệt của số phức:

|                           |                                |
|---------------------------|--------------------------------|
| <code>real(x)</code>      | phần thực của x                |
| <code>imag(x)</code>      | phần ảo của x                  |
| <code>conj(x)</code>      | liên hợp phức của x            |
| <code>abs(x)</code>       | độ lớn, trị tuyệt đối của x    |
| <code>angle(x)</code>     | góc pha của số phức            |
| <code>complex(a,b)</code> | tạo số phức từ phần thực và ảo |

Ví dụ:

```
>> a=1+3i
a = 1.0000 + 3.0000i
>> b=2-4i
b = 2.0000 - 4.0000i
>> a+b
ans = 3.0000 - 1.0000i
>> abs(a)
ans = 3.1623
>> real(b)
ans = 2
>> imag(b)
ans = -4
>> complex(2,2)
ans = 2.0000 + 2.0000i
```

## II. BIẾN VÀ CÁC THAO TÁC CỦA CÁC BIẾN

### 5. Một số hàm toán:

| Hàm                   | Ý nghĩa  |
|-----------------------|--|
| <code>sqrt(x)</code>  | Căn bậc 2  |
| <code>exp(x)</code>   | Hàm mũ cơ số e   |
| <code>abs(x)</code>   | Giá trị tuyệt đối  |
| <code>sign(x)</code>  | Hàm dấu (=1 nếu $x > 0$ ; = -1 nếu $x < 0$ ; = 0 nếu $x = 0$ ) |
| <code>rem(x,y)</code> | Số dư phép chia x/y  |
| <code>sum(v)</code>   | Tổng các phần tử vector  |
| <code>prod(v)</code>  | Tích các phần tử vector  |
| <code>min(v)</code>   | Phần tử vector bé nhất   |
| <code>max(v)</code>   | Phần tử vector lớn nhất  |
| <code>mean(v)</code>  | Giá trị trung bình cộng  |

Ví dụ 1:

```
>> x=4;
>> sqrt(x)
ans = 2
>> exp(x)
ans = 54.5982
>> sign(x)
ans = 1
>> rem(x,3)
ans = 1
>> v=[1 2 3];
>> min_v=min(v)
min_v = 1
>> mean(v)
ans = 2
>> sum(v)
ans = 6
```

Ví dụ 2: Tìm nghiệm của phương trình  $x^2-3x+2=0$

Trong command window:

```
>> a=1; b=-3; c=2;
>> x1=(-b+sqrt(b^2-4*a*c))/(2*a)
x1 =
    2
>> x2=(-b-sqrt(b^2-4*a*c))/(2*a)
x2 =
    1
```

### III. SƠ LƯỢC VỀ ĐỒ HỌA TRONG MATLAB:

- Các lệnh thông dụng trong đồ họa Matlab:

|                            |  |
|----------------------------|--|
| <code>plot(x,y)</code>     | vẽ đồ thị theo tọa độ x-y              |
| <code>plot(x,y,z)</code>   | vẽ đồ thị theo tọa độ x-y-z            |
| <code>title</code>         | đưa các title vào trong hình vẽ        |
| <code>xlabel</code>        | đưa các nhãn theo chiều x của đồ thị   |
| <code>ylabel</code>        | đưa các nhãn theo chiều y của đồ thị   |
| <code>zlabel</code>        | đưa các nhãn theo chiều z của đồ thị   |
| <code>grid</code>          | vẽ lưới trên đồ thị                    |
| <code>plot(y)</code>       | vẽ đồ thị theo y, bỏ qua chỉ số theo x |
| <code>plot(x,y,'S')</code> | S dùng để qui định màu, nét vẽ...      |



### III. SƠ LƯỢC VỀ ĐỒ HỌA TRONG MATLAB:

| Các loại màu vẽ |         | Các loại Marker (điểm) |        |
|-----------------|---------|------------------------|--------|
| y               | yellow  | .                      | điểm   |
| m               | magenta | o                      | chữ o  |
| c               | cyan    | x                      | dấu x  |
| r               | red     | +                      | dấu +  |
| g               | green   | #                      | dấu #  |
| b               | blue    | Các loại nét vẽ        |        |
| w               | white   | -                      | dấu -  |
| k               | black   | :                      | dấu :  |
|                 |         | -.                     | dấu -. |
|                 |         | --                     | dấu -- |



### III. SƠ LƯỢC VỀ ĐỒ HỌA TRONG MATLAB:

Ví dụ 1: Vẽ hàm  $\sin 2x$ ,  $\sin x^2$ ,  $(\sin x)^2$

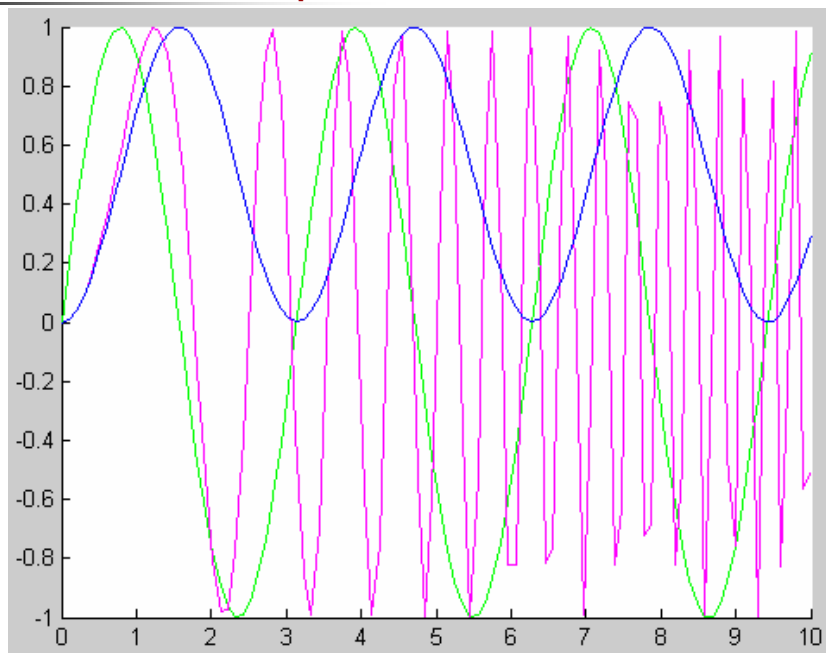
```
>> hold on
>> x=linspace(0,10);
>> y1=sin(2*x);
>> y2=sin(x.^2);
>> y3=(sin(x)).^2;
>> plot(x,y1,'g');           % green
>> plot(x,y2,'m');           % magenta
>> plot(x,y3,'b');           % blue
```





## CHƯƠNG 1: CÁC KHÁI NIỆM CƠ BẢN

29



Giảng viên: Hoàng Xuân Dương



## CHƯƠNG 1: CÁC KHÁI NIỆM CƠ BẢN

30

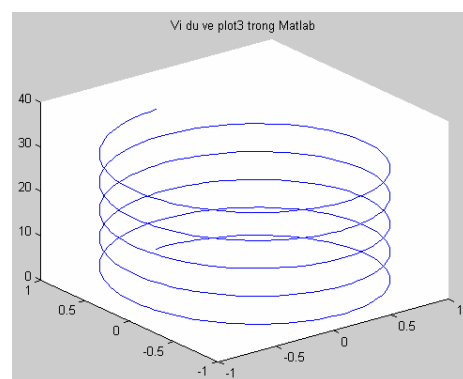
### III. SƠ LƯỢC VỀ ĐỒ HỌA TRONG MATLAB:

Ví dụ 2: Vẽ đường Helix trong không gian 3D

```
>> t = 0:pi/50:10*pi;
```

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

```
>> title('Ví dụ vẽ plot3 trong Matlab');
```



## IV. ÂM THANH TRONG MATLAB

`sound(y)`      gọi vector `y` ra loa, vector được sắp xếp với biên độ lớn nhất

`sound(y,f)`      `f` dải tần (Hz)

Ví dụ: âm thanh với lệnh load

```
>> load train;      % giá trị âm thanh tàu hỏa  
>> sound(y);      % được đưa vào tham số y  
>> load chirp;      % tiếng chim kêu  
>> sound(y);
```

## CHƯƠNG 2:

# MA TRẬN VÀ CÁC PHÉP TOÁN MA TRẬN

$$\begin{bmatrix} 16 & 2 & 3 & 13 \\ 5 & 11 & 10 & 8 \\ 9 & 7 & 6 & 12 \\ 4 & 14 & 16 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 3 & 6 & 10 \\ 1 & 4 & 10 & 20 \end{bmatrix}$$

Giảng viên: Hoàng Xuân Dương





- I. MA TRẬN
- II. CÁC MA TRẬN ĐẶC BIỆT
- III. CÁC PHÉP TOÁN TRÊN MẢNG
- IV. CÁC PHÉP TOÁN MA TRẬN
- V. GIẢI HỆ PHƯƠNG TRÌNH ĐỘC LẬP TUYẾN TÍNH
- VI. BÀI TẬP



I. **MA TRẬN:**

1. **Vector-Đại lượng vô hướng-Ma trận:**

- **Ma trận** là đối tượng chủ yếu của Matlab
- Các phần tử của ma trận được xếp theo hàng và cột
- **Đại lượng vô hướng** (giá trị đơn) là ma trận có 1 hàng và một cột
- Ma trận chỉ có 1 hàng hoặc một cột được gọi là **vector**
- Để truy cập một phần tử của ma trận, sử dụng chỉ số hàng và cột



**I. MA TRẬN:****2. Một số qui ước về ma trận:**

- Tên ma trận phải bắt đầu bằng chữ cái
- Bên phải dấu bằng là các giá trị ma trận được viết theo thứ tự hàng trong dấu ngoặc vuông
- Dấu chấm phẩy (;) phân cách hàng. Các giá trị trong hàng được phân cách bằng dấu phẩy (,) hoặc khoảng trắng. Dấu thập phân là dấu chấm (.). Kết thúc ma trận là dấu (;)

**I. MA TRẬN:**

Ví dụ:


```
>> a=[1 2 3; 4 5 6; 7 8 9]           % ma trận 3 hàng 3 cột
a =  1   2   3
     4   5   6
     7   8   9

>> b=[1 2 3 4]                       % vector hàng
b =  1   2   3   4

>> c=[1;2]                           % vector cột
c =  1
     2

>> d=[1]                             % giá trị đơn
d =  1

>> a(2,3)                            % phần tử ở hàng 2 cột 3
ans = 6
```


**CHƯƠNG 2: MA TRẬN VÀ CÁC PHÉP TOÁN MA TRẬN**


37


**I. MA TRẬN:**

**3. Khai báo vector và ma trận:**

| Khai báo                    | Ý nghĩa                              |
|-----------------------------|--------------------------------------|
| [x1 x2...; x3 x4...]        | Nhập giá trị cho vector và ma trận   |
| start:increment:destination | Toán tử (:)                          |
| linspace(start,dest,number) | Khai báo tuyến tính cho vector       |
| logspace(start,dest,number) | Khai báo logarithm cho vector        |
| rand(line,column)           | Ma trận nhận giá trị ngẫu nhiên 0->1 |
| randn(line,column)          | Ma trận nhận giá trị ngẫu nhiên      |

---




**CHƯƠNG 2: MA TRẬN VÀ CÁC PHÉP TOÁN MA TRẬN**

38

**I. MA TRẬN:**


**3. Khai báo vector và ma trận (tt)**

**➤ Kiểu liệt kê trực tiếp:**

Các phần tử được liệt kê trong dấu ngoặc vuông:

```
>> A=[3,5];
>> B=[1.7,3.2];
>> C=[-1 0 0 ; -1 1 0 ; 1 -1 0; 0 0 2]; Hoặc:
>> C = [-1 0 0
        -1 1 0
         1 -1 0
         0 0 2];
```

---



## I. MA TRẬN:

## 3. Khai báo vector và ma trận (tt)

## ➤ Kiểu liệt kê trực tiếp (tt)

```
>> F = [1, 52, 45, 84, 94, 5, 65, 42, 85,...
        23, 52, 65, 21, 74];
```

Định nghĩa ma trận từ ma trận khác:

```
>> B=[1 2 4];
>> S=[3 B];      % S=[3 1 2 4]
```

Mở rộng ma trận:

```
>> S(5)=9;
>> S(8)=3;      % S(6), S(7) nhận giá trị 0
```

## I. MA TRẬN:

## 3. Khai báo vector và ma trận (tt)

## ➤ Kiểu liệt kê trực tiếp (tt)

Khai báo tuyến tính:

```
>> x=linspace(2,20,10)
x = 2    4    6    8   10   12   14   16   18   20
>> logspace(1,2,5)
ans = 10.0000  17.7828  31.6228  56.2341 100.0000
>> y=linspace(1,10);      % y=vector 100 phần tử
```

**I. MA TRẬN:****3. Khai báo vector và ma trận (tt)****➤ Từ một file dữ liệu:**

Một file văn bản **matran.dat** có nội dung:

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

```
>> load c:\matran.dat
```

```
>> matran
```

```
matran=
```

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

**I. MA TRẬN:****3. Khai báo vector và ma trận (tt)****➤ Sử dụng toán tử (:)**

Tại vị trí dấu (:) trong ma trận, nó đại diện cho tất cả các hàng hoặc tất cả các cột

```
>> x=A(:,1);      % đưa dữ liệu ở cột 1 vào vector x
```

```
>> y=A(:,2);      % đưa dữ liệu ở cột 2 vào vector y
```

Dấu (:) sử dụng làm ký hiệu tổng quát cho ma trận mới

```
>> H=1:5
```

```
H = 1 2 3 4 5
```

```
>> TIME=0.0:0.5:2.5;
```

```
TIME = 0.0 0.5 1.0 1.5 2.0 2.5
```

## I. MA TRẬN:

## 3. Khai báo vector và ma trận (tt)

## ➤ Sử dụng toán tử (:)

Dấu (:) dùng chọn các ma trận con từ ma trận khác

Ví dụ:

```
>> C=[-1 0 0
```

```
      1 -1 0
```

```
      1 -1 2
```

```
      0 2 -1];
```

```
>> C_1=C(:,2:3);
```

```
>> C_2=C(3:4,1:2);
```

```
C_1= [ 0 0
      -1 0
      -1 2
      2 -1];
```

```
C_2= [ 1 -1
      0 2];
```

## I. MA TRẬN:

## 3. Khai báo vector và ma trận (tt)

## ➤ Sử dụng toán tử (:)

- C(:) tương đương vector có một cột, phần tử của cột chính là các cột của ma trận C

```
>> C = [ 1 4 7
```

```
        2 5 8
```

```
        3 6 9];
```

```
>> C(:)
```

```
ans =
     1
     2
     3
     4
     5
     6
     7
     8
     9
```

**I. MA TRẬN:****3. Khai báo vector và ma trận:****➤ Trực tiếp từ bàn phím:**

```
>> z=input('Nhập giá trị cho z:');
```

Nếu không có dữ liệu, z là ma trận rỗng

**I. MA TRẬN:****4. Hiển thị ma trận:**

Kết quả tính toán có thể được định dạng bằng lệnh **format**.

**long**                      số chấm cố định là 15 con số

**long e**                    số dấu chấm động 15 con số

**short**                    số chấm cố định là 5 con số (**mặc định**)

**short e**                  số dấu chấm động 5 con số

Ví dụ:

```
>> pi
```

```
ans = 3.1416
```

```
>> format long
```

```
>> pi
```

```
ans = 3.14159265358979
```

**I. MA TRẬN:****4. Hiển thị ma trận (tt)**

`disp` → Xuất chuỗi ký tự ra màn hình

`fprintf` → cho phép xuất ra theo định dạng. Với cú pháp:

>> `fprintf(định dạng, ma trận);`

| Kiểu loại | Dạng in ra                       | Ký tự   | Ý nghĩa         |
|-----------|----------------------------------|---------|-----------------|
| %c        | Kiểu ký tự                       | \n      | Xuống dòng      |
| %s        | Kiểu chuỗi                       | \t      | tab             |
| %d        | Kiểu số nguyên thập phân         | \b      | Backspace       |
| %f        | Kiểu số dấu chấm tĩnh            | \r      | Carriage return |
| %e        | Kiểu số dấu chấm động            | \f      | From feed       |
| %x        | Kiểu số Hex                      | %%      | %               |
| %bx       | Kiểu chấm tĩnh trong Hex 64 bits | " or \" | '               |

**I. MA TRẬN:****4. Hiển thị ma trận (tt)**

Ví dụ 1:

>> `disp('Hello')`

Hello

>> `disp(pi)`

3.1416

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

>> `disp(x)`

1 2 3 4

>> `temp=78;`

>> `fprintf('Nhiệt độ là: \n %6.1f độ F',temp);`

Nhiệt độ là:

78.0 độ F



## I. MA TRẬN:

### 4. Hiển thị ma trận (tt)

Ví dụ 2:

```
>> temp=78;
>> st='do F';
>> fprintf('Nhiet do la %4.1f %s\n',temp,st)
Nhiet do la 78.0 do F
>> fprintf('Nhiet do la %4.1f\b %s\n',temp,st)
Nhiet do la 78. do F
>> fprintf('Nhiet do la %4.1f\t %s\n',temp,st)
Nhiet do la 78.0    do F
>> fprintf('It's Friday.\n')
It's Friday.
```

## II. CÁC MA TRẬN ĐẶC BIỆT:

Matlab có một số hàm để tạo ma trận đặc biệt

### 1. Ma trận ma phương (magic(n))

- Ma phương bậc n là ma trận vuông cấp n
- Bao gồm các số nguyên từ 1 đến  $n^2$
- Các phần tử sắp xếp sao cho tổng các phần tử trên một hàng, một cột, đường chéo là bằng nhau

Ví dụ:

```
>> magic(4)
ans=
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

## II. CÁC MA TRẬN ĐẶC BIỆT:

### 2. Ma trận zero:

- Hàm `zeros(m,n)` là ma trận có kích thước  $m \times n$  chứa toàn số 0
- Nếu tham số chỉ có một  $\rightarrow$  ma trận vuông

Ví dụ:

```
>> zeros(3,4)
```

```
ans=
```

```
0 0 0 0
0 0 0 0
0 0 0 0
```

## II. CÁC MA TRẬN ĐẶC BIỆT:

### 3. Ma trận ones:

- Hàm `ones(m, n)` là ma trận có kích thước  $m \times n$  chứa toàn số 1
- Nếu tham số chỉ có một  $\rightarrow$  ma trận vuông

Ví dụ:

```
>> ones(3,4)
```

```
ans=
```

```
1 1 1 1
1 1 1 1
1 1 1 1
```

**II. CÁC MA TRẬN ĐẶC BIỆT:****4. Ma trận đường chéo đặc biệt (Identity Matrix):**

- Ma trận có các phần tử trên đường chéo bằng 1
- Các phần tử còn lại bằng 0

Ví dụ:

```
>> eye(4)
ans=
 1 0 0 0
 0 1 0 0
 0 0 1 0
 0 0 0 1
```

**II. CÁC MA TRẬN ĐẶC BIỆT:****5. Ma trận đường chéo mở rộng eye(m,n):**

- Ma trận kích thước  $m \times n$  có các phần tử chỉ số hàng = chỉ số cột thì bằng 1
- Các phần tử còn lại bằng 0

Ví dụ:

```
>> eye(4,5)
ans=
 1 0 0 0 0
 0 1 0 0 0
 0 0 1 0 0
 0 0 0 1 0
```

## II. CÁC MA TRẬN ĐẶC BIỆT:

### 6. Ma trận Pascal (`pascal(n)`):

- Ma trận chứa các giá trị của tam giác pascal

Ví dụ:

```
>> pascal(4)
```

```
ans=
```

```
1 1 1 1
1 2 3 4
1 3 6 10
1 4 10 20
```

## II. CÁC MA TRẬN ĐẶC BIỆT:

### 7. Các ma trận đặc biệt khác:

```
compan
gallery
hadamard
hankel
hilb
invhilb
kron
rosser
toeplitz
vander
wilkinson
```

### III. CÁC PHÉP TOÁN TRÊN MẢNG:

#### 1. Tính toán với mảng:

| Ký hiệu  | Ý nghĩa                   | Biểu thức                          |
|----------|---------------------------|------------------------------------|
| $a + b$  | Cộng từng phần tử mảng    | $[a_1+b_1 \ a_2+b_2 \ ...a_n+b_n]$ |
| $a - b$  | Trừ từng phần tử mảng     | $[a_1-b_1 \ a_2-b_2 \ ...a_n-b_n]$ |
| $a .* b$ | Nhân từng phần tử mảng    | $[a_1*b_1 \ a_2*b_2 \ ...a_n*b_n]$ |
| $a ./ b$ | Chia từng phần tử a cho b | $[a_1/b_1 \ a_2/b_2 \ ...a_n/b_n]$ |
| $a ./ b$ | Chia từng phần tử b cho a | $[b_1/a_1 \ b_2/a_2 \ ...b_n/a_n]$ |
| $a .^ b$ | Lũy thừa từng phần tử     | $[a_1^b_1 \ a_2^b_2 \ ...a_n^b_n]$ |

Lưu ý: số phần tử 2 mảng a và b phải bằng nhau

### III. CÁC PHÉP TOÁN TRÊN MẢNG:

Ví dụ:

```
>> A=[4 8 15]; B=[2 2 3];
>> A + B
ans = 6 10 18
>> A - B
ans = 2 6 12
>> A .* B
ans = 8 16 45
>> A ./ B
ans = 2 4 5
>> A ./ B
ans = 0.5000 0.2500 0.2000
>> A .^ B
ans = 16 64 3375
```



### III. CÁC PHÉP TOÁN TRÊN MẢNG:

#### 2. Thứ tự ưu tiên của các toán tử:

| Ưu tiên | Toán tử                      |
|---------|------------------------------|
| 1       | Ngoặc đơn                    |
| 2       | Lũy thừa                     |
| 3       | Nhân & chia từ trái qua phải |
| 4       | Cộng & trừ từ trái qua phải  |



### IV. CÁC PHÉP TOÁN MA TRẬN:

Một số hàm xử lý ma trận cơ bản:

| Hàm                       | Ý nghĩa                                 |
|---------------------------|---|
| <code>matrix.'</code>     | Chuyển vị ma trận                       |
| <code>matrix'</code>      | Chuyển vị ma trận có phần phức liên hợp |
| <code>inv(matrix)</code>  | Đảo ma trận                             |
| <code>det(matrix)</code>  | Tính định thức ma trận                  |
| <code>eig(matrix)</code>  | Tính các giá trị riêng của ma trận      |
| <code>rank(matrix)</code> | Xác định hạng của ma trận               |



## IV. CÁC PHÉP TOÁN MA TRẬN:

## 1. Ma trận chuyển vị:

- Ma trận chuyển vị của A ký hiệu là  $A^T$
- Các phần tử hàng của A trở thành phần tử cột của  $A^T$

Ví dụ:

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

```
A =
```

```
1 2 3
```

```
4 5 6
```

```
>> A'
```

```
ans =
```

```
1 4
```

```
2 5
```

```
3 6
```

## IV. CÁC PHÉP TOÁN MA TRẬN:

## 2. Nhân ma trận:

- $C=A.*B$  nhân vô hướng
- $C=A*B$  nhân ma trận với:  $C_{ij} = \sum A_{ik}B_{kj}$   
Số cột của ma trận A phải bằng số hàng của ma trận B

Ví dụ:

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

```
>> C=A.*B
```

```
C= 3 8 15
```

```
24 35 48
```

```
>> B = B';
```

```
>> C = A*B
```

```
C= 26 44
```

```
62 107
```

## IV. CÁC PHÉP TOÁN MA TRẬN:

## 3. Phép quay:

- Cú pháp: `rot90(matrix)` hay `rot90(matrix,num)`;
- Các phần tử của A được quay  $90^\circ$  theo ngược chiều kim đồng hồ
- Dùng tham số `num` để xác định số lần quay

Ví dụ:

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

```
>> B = rot90(A)
```

```
B= 3 6 9
```

```
    2 5 8
```

```
    1 4 7
```

```
>> C = rot90(A,2)
```

```
C= ...
```

## IV. CÁC PHÉP TOÁN MA TRẬN:

## 4. Phép đảo ma trận:

- `fliplr(A)` → Đảo các phần tử A từ trái sang phải
- `flipud(A)` → Đảo các phần tử A từ trên xuống dưới

Ví dụ:

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

```
>> B = fliplr(A)
```

```
B= 3 2 1
```

```
    6 5 4
```

```
    9 8 7
```

```
>> C = flipud(B)
```

```
C= 9 8 7
```

```
    6 5 4
```

```
    3 2 1
```



## IV. CÁC PHÉP TOÁN MA TRẬN:

## 5. Reshape:

- Cho phép định dạng lại ma trận với số hàng và số cột khác với ma trận gốc
- Số phần tử của ma trận gốc và ma trận mới phải bằng nhau
- Hàm có 3 tham số là ma trận gốc, số hàng và số cột

Ví dụ:

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

```
>> B=reshape(A,1,9)
```

```
B=
```

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

## IV. CÁC PHÉP TOÁN MA TRẬN:

## 6. Trích các phần tử từ ma trận:

| Hàm       | Ý nghĩa   |
|-----------|---|
| diag(A)   | Lấy đường chéo chính lưu vào một vector cột   |
| diag(A,k) | Chọn đường chéo dựa vào k<br>k=0 đường chéo chính<br>k>0 đường chéo thứ k trên đường chéo chính<br>k<0 đường chéo thứ k dưới đường chéo chính |
| A=diag(V) | Nếu V là vector thì A là ma trận vuông có V là đường chéo chính. Các phần tử khác bằng 0  |

## IV. CÁC PHÉP TOÁN MA TRẬN:

## 6. Trích các phần tử từ ma trận (tt)

Ví dụ:

&gt;&gt; A=[1 2 3 4; 5 6 7 8; 9 10 11 12]; V=[1:3];

&gt;&gt; diag(A)

ans = 1

6

11

&gt;&gt; diag(A,-1)

ans = 5

10

&gt;&gt; A=diag(V)

A = 1 0 0

0 2 0

0 0 3

## IV. CÁC PHÉP TOÁN MA TRẬN:

## 6. Trích các phần tử từ ma trận (tt)

| Hàm       | Ý nghĩa   |
|-----------|---|
| B=triu(A) | Sinh ra ma trận B cùng cỡ, chứa các phần tử A nằm ở đường chéo chính và trên đường chéo chính. Vị trí khác bằng 0 |
| triu(A,k) | Phần tử A nằm trên và phía trên đường chéo thứ k  |
| tril(A)   | Sinh ra ma trận cùng cỡ, chứa các phần tử A nằm ở đường chéo chính và dưới đường chéo chính. Vị trí khác bằng 0   |
| tril(A,k) | Phần tử A nằm ngay trên và phía dưới đường chéo thứ k. Các vị trí khác bằng 0                                     |

## IV. CÁC PHÉP TOÁN MA TRẬN:

## 6. Trích các phần tử từ ma trận (tt)

Ví dụ:

&gt;&gt; A = [1 2 3 4; 5 6 7 8; 9 10 11 12];

&gt;&gt; B = triu(A)

```
B=  1  2  3  4
    0  6  7  8
    0  0 11 12
```

&gt;&gt; C = triu(A,-1)

```
C=  1  2  3  4
    5  6  7  8
    0 10 11 12
```

## IV. CÁC PHÉP TOÁN MA TRẬN:

## 6. Trích các phần tử từ ma trận (tt)

Ví dụ:

&gt;&gt; B = tril(A)

```
B=  1  0  0  0
    5  6  0  0
    9 10 11  0
```

&gt;&gt; C = tril(A,-1)

```
C=  0  0  0  0
    5  0  0  0
    9 10  0  0
```

**V. GIẢI HỆ PHƯƠNG TRÌNH ĐỘC LẬP TUYẾN TÍNH:**

Xét hệ:

$$x_1 - 2x_2 + x_3 = 2$$

$$2x_1 + x_2 - 4x_3 = -1$$

$$3x_1 - 4x_2 - x_3 = 0$$

Giải:

$$D = \begin{vmatrix} 1 & -2 & 1 \\ 2 & 1 & -4 \\ 3 & -4 & -1 \end{vmatrix} = -8 \quad ; \quad D1 = \begin{vmatrix} 2 & -2 & 1 \\ -1 & 1 & -4 \\ 0 & -4 & -1 \end{vmatrix} = -28$$

$$D2 = \begin{vmatrix} 1 & 2 & 1 \\ 2 & -1 & -4 \\ 3 & 0 & 1 \end{vmatrix} = -16 \quad ; \quad D3 = \begin{vmatrix} 1 & -2 & 2 \\ 2 & 1 & -1 \\ 3 & -4 & 0 \end{vmatrix} = -20$$

**V. GIẢI HỆ PHƯƠNG TRÌNH ĐỘC LẬP TUYẾN TÍNH:**

Nghiệm của hệ là

$$x_1 = D1/D = 3.5$$

$$x_2 = D2/D = 2$$

$$x_3 = D3/D = 2.5$$

Trong Matlab:

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

```
>> b=[2;-1;0];
```

```
>> x=inv(A)*b
```

```
x =
```

```
3.5000
```

```
2.0000
```

```
2.5000
```

**V. GIẢI HỆ PHƯƠNG TRÌNH ĐỘC LẬP TUYẾN TÍNH:**

Bài tập:

$$x_1 + x_2 + x_3 + x_4 = 0$$

$$x_2 + x_3 + x_4 + x_5 = 0$$

$$x_1 + 2x_2 + 3x_3 = 2$$

$$x_2 + 2x_3 + 3x_4 = -2$$

$$x_3 + 2x_4 + 3x_5 = 2$$

**VI. BÀI TẬP:**

1) Hãy cho biết kết quả của từng dòng lệnh sau:

```
>> A = [1:3;4:6]
```

```
>> B = [A A;A A]
```

```
>> C = B(1:2,3:4)
```

```
>> D = C+2
```

```
>> E = C.*D
```

```
>> F = C*2 - 1
```

2) Hãy cho biết kết quả của từng dòng lệnh sau:

```
>> A = pascal(4)
```

```
>> diag(A)
```

```
>> diag(A,-1)
```

```
>> C=diag(diag(A,1))
```

```
>> D=diag(diag(A))
```

## VI. BÀI TẬP:

3) Hãy cho biết kết quả của từng dòng lệnh sau:

```
>> A = pascal(3)
>> B = rot90(A,3)
>> C = fliplr(flipud(B))
>> D = flipud(fliplr(C))
>> C + D
>> (A(:))'
```



## CHƯƠNG 3:

```
function numcheck(input_num)
% if input_num is in {-1,0,1}
switch input_num
case -1
    disp('negative one');
case 0
    disp('zero');
case 1
    disp('positive one');
otherwise
    disp('other value');
end
```

# LẬP TRÌNH TRONG MATLAB



- I. PHẦN TỬ CƠ BẢN
- II. HÀM TOÁN HỌC
- III. CÁC DẠNG FILE
- IV. BIỂU THỨC QUAN HỆ VÀ LOGIC
- V. CẤU TRÚC ĐIỀU KHIỂN
- VI. BÀI TẬP



## I. PHẦN TỬ CƠ BẢN

### 1. Giới hạn của các giá trị tính toán trong Matlab

- Đối với phần lớn máy tính, khoảng giá trị cho phép từ  $10^{-308}$  đến  $10^{308}$ .
- Nếu có giá trị tràn số mũ trên, nó được biểu diễn bởi inf (số vô hạn)
- Nếu tràn mũ dưới, nó được biểu diễn là 0
- Chia cho 0 là toán tử không hợp lệ, kết quả là inf. Matlab sẽ cảnh báo và sử dụng giá trị inf để tính tiếp.



## I. PHẦN TỬ CƠ BẢN

### 2. Biến string:

- Chuỗi ký tự được đặt giữa 2 dấu nháy đơn
- Chuỗi ký tự là một mảng nhiều ký tự. Ký tự được lưu dưới dạng mã ASCII.

```
>> name= 'Trường Đại học DL Công Nghệ Sài Gòn'
```

- Có thể truy xuất đến từng phần tử chuỗi

```
>> fprintf('Trường tôi là %s\n', name(8:35));
```

- Kết hợp các string tạo string mới

```
>> text1='Tôi học tại'; text=[text1 ' ' name];
```

- Nhập string từ bàn phím:

```
>> str= input('Nhập vào một chuỗi','s');
```

Giảng viên: Hoàng Xuân Dương

## I. PHẦN TỬ CƠ BẢN

### 2. Biến string:

Các lệnh với biến string:

| Hàm     | Ý nghĩa                   |
|---------|---------------------------|
| char    | Tạo mảng ký tự từ mảng số |
| double  | Đổi chuỗi sang mã ASCII   |
| num2str | Đổi số sang chuỗi         |
| str2num | Đổi chuỗi sang số         |
| int2str | Đổi số nguyên sang chuỗi  |
| str2mat | Đổi chuỗi sang ma trận    |
| mat2str | Đổi ma trận sang chuỗi    |

Giảng viên: Hoàng Xuân Dương





## II. HÀM TOÁN HỌC

### 1. Hàm toán học cơ bản

| Hàm         | Ý nghĩa                        |
|-------------|--------------------------------|
| round       | Làm tròn về số nguyên gần nhất |
| fix         | Làm tròn về 0                  |
| floor       | Làm tròn nhỏ hơn               |
| ceil        | Làm tròn lớn hơn               |
| log(x)      | $\ln(x)$                       |
| log10(x)    | log thập phân                  |
| pow2(x)     | Lũy thừa cơ số 2               |
| nextpow2(N) | Tìm $p$ : $2^p = N$            |

Giảng viên: Hoàng Xuân Dương



## II. HÀM TOÁN HỌC

### 1. Hàm toán học cơ bản

Ví dụ:

```
>> a=[-1.9 -0.2 3.4 5.6 7 2.4 +3.6i];  
>> fix(a)  
-1.0000 0 3.0000 5.0000 7.0000 2.0000 0+3.0000i  
>> ceil(a)  
-1.0000 0 4.0000 6.0000 7.0000 3.0000 0+4.0000i  
>> floor(a)  
-2.0000 -1.0000 3.0000 5.0000 7.0000 2.0000 0+3.0000i  
>> round(a)  
-2.0000 0 3.0000 6.0000 7.0000 2.0000 0+4.0000i
```

Giảng viên: Hoàng Xuân Dương





## II. HÀM TOÁN HỌC

### 2. Hàm lượng giác cơ bản:

| Hàm              | Ý nghĩa                                   |
|------------------|---|
| $\sin(x)$        | sin của x khi x có đơn vị radian          |
| $\cos(x)$        | cos của x khi x có đơn vị radian          |
| $\tan(x)$        | tan của x khi x có đơn vị radian          |
| $\text{asin}(x)$ | $\in [-\pi/2, \pi/2]$ khi $x \in [-1, 1]$ |
| $\text{acos}(x)$ | $\in [0, \pi]$ khi $x \in [-1, 1]$        |
| $\text{atan}(x)$ | khi $x \in [-\pi/2, \pi/2]$               |

Đổi radian sang độ và ngược lại:

$\text{angle\_degrees} = \text{angle\_radians} * (180/\pi)$

$\text{angle\_radians} = \text{angle\_degrees} * (\pi/180)$

Giảng viên: Hoàng Xuân Dương



## III. CÁC DẠNG FILE

### 1. Script file (m file):

- Các chương trình, thủ tục bao gồm các dòng lệnh theo một thứ tự nào đó do người sử dụng viết ra được lưu trong các file \*.m. Được gọi là script file
- Dùng trình soạn thảo **edit** của Matlab để viết hàm
- Lưu dưới dạng ASCII
- Có thể chạy giống các lệnh, thủ tục của Matlab

Giảng viên: Hoàng Xuân Dương



Ví dụ: tập tin **canhhoa.m** có nội dung như sau:

```
% M-file script tạo ra 4 hình cánh hoa
theta=-pi:0.01:pi;
rho(1,:)=2*sin(5*theta).^2;
rho(2,:)=cos(10*theta).^3;
rho(3,:)=sin(theta).^2;
rho(4,:)=5*cos(3.5*theta).^3;
for i=1:4
    polar(theta,rho(i,:))
    pause
end
```

Trong command window:

```
>> help canhhoa
```

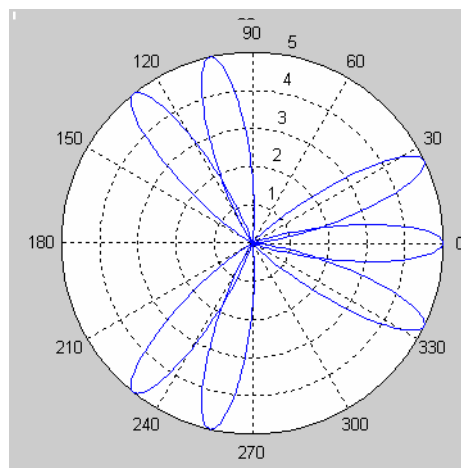
M-file script tạo ra 4 hình cánh hoa

Giảng viên: Hoàng Xuân Dương

```
>> echo on
```

```
>> canhhoa
```

```
theta=-pi:0.01:pi
rho(1,:)=2*sin(5*theta).^2;
rho(2,:)=cos(10*theta).^3;
rho(3,:)=sin(theta).^2;
rho(4,:)=5*cos(3.5*theta).^3;
for i=1:4
    polar(theta,rho(i,:))
    pause
    polar(theta,rho(i,:))
    pause
    polar(theta,rho(i,:))
    pause
    polar(theta,rho(i,:))
    pause
```



Giảng viên: Hoàng Xuân Dương

### III. CÁC DẠNG FILE

#### 2. Hàm và tạo hàm trong Matlab:

- Giống như script file. Cấu trúc tổng quát của hàm:

```
function [y1,y2,...]=function_name (a,b,c...)  
% help text in the usage of the function  
%.....  
:  
end
```

- Có thể chỉ là một nhóm dòng lệnh hay nhận vào các đối số và trả về kết quả
- Có thể gọi hàm từ các hàm, script khác
- Các biến trong hàm là các biến cục bộ

Giảng viên: Hoàng Xuân Dương

#### Quy tắc viết hàm M-files:

- 1) Bắt đầu bằng từ function, sau đó lần lượt các tham số đầu ra, dấu bằng, tên hàm và các tham số đầu vào
- 2) Một số dòng sau tên hàm bắt đầu bằng dấu % là các dòng chú thích về cách dùng hàm, nó được bỏ qua khi chạy. Được hiển thị khi lệnh help yêu cầu hàm
- 3) Matlab có thể chấp nhận nhiều tham số ngõ vào và tham số ngõ ra
- 4) Nếu hàm trả về nhiều hơn một giá trị, các giá trị được trả về như một vector
- 5) Nếu hàm nhận nhiều tham số ngõ vào, các tham số sẽ được liệt kê trong dấu ngoặc đơn
- 6) Kết thúc hàm là phát biểu 'end'

Giảng viên: Hoàng Xuân Dương

## III. CÁC DẠNG FILE

## 2. Hàm và tạo hàm trong Matlab (tt)

Ví dụ 1:

Thực hiện hàm `luythua.m` như sau:

```
function y=luythua(a,b)
% Ham tinh a^b
y=a^b;
```

Trong command window:

```
>> luythua(2,3)
ans = 8
>> c=luythua(4,2)
c = 16
```

Giảng viên: Hoàng Xuân Dương

## III. CÁC DẠNG FILE

## 2. Hàm và tạo hàm trong Matlab (tt)

Ví dụ 2:

Để giải phương trình bậc 2:  $ax^2+bx+c=0$ . Thực hiện hàm tính nghiệm như sau, lưu với tên `quadroot.m`

```
function [x1,x2]=quadroot(a,b,c)
% Hàm tính nghiệm của phương trình bậc 2
radical=sqrt(b^2-4*a*c);
x1=(-b+radical)/(2*a);
x2=(-b-radical)/(2*a);
```

Giảng viên: Hoàng Xuân Dương

## III. CÁC DẠNG FILE

## 2. Hàm và tạo hàm trong Matlab (tt)

Chương trình có tên `ptbac2.m` có nội dung như sau:

```
disp('Chương trình giải phương trình bậc 2: ax^2+bx+c=0');  
a=input('Nhập a: ');  
b=input('Nhập b: ');  
c=input('Nhập c: ');  
[x1,x2]=quadroot(a,b,c);    % gọi hàm quadroot  
disp('Nghiem cua phuong trinh: ');  
fprintf('x1=%f\n',x1);  
fprintf('x2=%f\n',x2);
```

---

Giảng viên: Hoàng Xuân Dương

## III. CÁC DẠNG FILE

## 2. Hàm và tạo hàm trong Matlab (tt)

Trong Command window:

```
>> [a,b]=quadroot(1,-3,2)  
a = 2  
b = 1  
>> ptbac2  
Chương trình giải phương trình bậc 2: ax^2+bx+c=0  
Nhập a: 1  
Nhập b: -3  
Nhập c: 2  
Nghiem cua phuong trinh:  
x1=2.000000  
x2=1.000000
```

---

Giảng viên: Hoàng Xuân Dương

### III. CÁC DẠNG FILE

#### 3. File dữ liệu:

Matlab phân biệt 2 loại dữ liệu khác nhau:

- Mat-files: thích hợp cho dữ liệu chương trình Matlab. Phần mở rộng là .mat

```
>> save <tên file> <tên ma trận>;
```

```
>> load <tên file>;
```

- ASCII files: cho dữ liệu được chia sẻ với các chương trình khác. Phần mở rộng là .dat

```
>> save <tên file>.dat <tên ma trận> /ascii;
```

```
>> load <tên file>.dat;
```

Giảng viên: Hoàng Xuân Dương

### IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

#### 1. Các phép toán quan hệ:

| Toán tử | Ý nghĩa           |
|---------|-------------------|
| <       | Nhỏ hơn           |
| <=      | Nhỏ hơn hoặc bằng |
| >       | Lớn hơn           |
| >=      | Lớn hơn hoặc bằng |
| ==      | Bằng              |
| ~=      | Không bằng        |

Phép so sánh 2 ma trận là so sánh từng phần tử. Kết quả sinh ra ma trận  $\{0,1\}$  cùng cỡ. Nếu phép so sánh đúng, các phần tử  $=1$ , ngược lại thì các phần tử bằng 0

Giảng viên: Hoàng Xuân Dương

## IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

## 1. Các phép toán quan hệ (tt)

Ví dụ:

```
>> a=[3 4 3; 4 5 6];
>> b=[1 2 3; 7 8 6];
>> a==b
ans =
    0    0    1
    0    0    1

>> a>b
ans =
    1    1    0
    0    0    0

>> a>=b
ans =
    1    1    1
    0    0    1
```

Giảng viên: Hoàng Xuân Dương

## IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

## 2. Các phép toán logic:

| Toán tử | Ký hiệu |
|---------|---------|
| not     | ~       |
| and     | &       |
| or      |         |

- Thứ tự các toán tử trong biểu thức logic từ cao đến thấp là **not**, **and**, **or**. Tuy nhiên có thể dùng ngoặc đơn để thay đổi
- Trong Matlab, tất cả các giá trị khác không đều coi như đúng (**true**), còn giá trị 0 được coi như sai (**false**)

Giảng viên: Hoàng Xuân Dương



## IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

## 2. Các phép toán logic (tt)

Ví dụ:

&gt;&gt; b=[1 1 0; 1 0 1]

&gt;&gt; a=[0 1 0; 0 0 1]

&gt;&gt; a&amp;b

```
ans =      0      1      0
          0      0      1
```

&gt;&gt; a|b

```
ans =      1      1      0
          1      0      1
```

&gt;&gt; ~a

```
ans =      1      0      1
          1      1      0
```

Giảng viên: Hoàng Xuân Dương

## IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

## 3. Các hàm quan hệ và logic:

| Hàm        | Ý nghĩa   |
|------------|---|
| any(x)     | Trả về vector hàng có các phần tử =1 nếu tồn tại phần tử cột của x khác 0, ngược lại =0         |
| all(x)     | Trả về vector hàng có các phần tử =1 nếu tất cả phần tử cột của x khác 0, ngược lại =0          |
| find(x)    | Trả về vector chứa chỉ số các phần tử của x khác 0  |
| exist('a') | = 1 nếu a là biến, = 2 nếu là file, = 0 nếu a không tồn tại...                                  |
| isnan(x)   | Trả về ma trận cùng cỡ có các phần tử = 1 nếu các phần tử tương ứng của x là nan, ngược lại = 0 |

Giảng viên: Hoàng Xuân Dương

## IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

## 3. Các hàm quan hệ và logic (tt)

| Hàm                        | Ý nghĩa  |
|----------------------------|--|
| <code>finite(x)</code>     | Trả về ma trận cùng cỡ có các phần tử = 1 nếu các phần tử tương ứng của x hữu hạn, = 0 nếu vô hạn hoặc nan |
| <code>isempty(x)</code>    | = 1 nếu x rỗng, ngược lại = 0  |
| <code>isstr(x)</code>      | = 1 nếu x là một chuỗi, ngược lại = 0  |
| <code>strcmp(y1,y2)</code> | So sánh 2 chuỗi, =1 nếu 2 chuỗi giống hệt nhau, ngược lại =0. Phân biệt hoa-thường, dấu cách, đầu dòng     |

Giảng viên: Hoàng Xuân Dương

## IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

## 3. Các hàm quan hệ và logic (tt)

Ví dụ:

```
>> a=[0 1 2; 0 0 3];
>> any(a)
ans = 0    1    1
>> all(a)
ans = 0    0    1
>> find(a)
ans =     3
      5
      6
```

Giảng viên: Hoàng Xuân Dương

## IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

## 3. Các hàm quan hệ và logic (tt)

Ví dụ:

```
>> a=[nan 12 4 0; inf 3 8 nan]
```

```
a = NaN    12    4    0  
      Inf    3    8 NaN
```

```
>> isnan(a)
```

```
ans = 1    0    0    0  
      0    0    0    1
```

```
>> finite(a)
```

```
ans = 0    1    1    1  
      0    1    1    0
```

```
>> isempty(a)
```

```
ans = 0
```

Giảng viên: Hoàng Xuân Dương

## IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

## 3. Các hàm quan hệ và logic (tt)

Ví dụ:

```
>> text1='Lop HCDH';
```

```
>> text2='Lop';
```

```
>> text3='HCDH';
```

```
>> isstr(text1)
```

```
ans = 1
```

```
>> strcmp(text1,text2)
```

```
ans = 0
```

```
>> strcmp(text1,[text2 ' ' text3])
```

```
ans = 1
```

Giảng viên: Hoàng Xuân Dương



## V. CẤU TRÚC ĐIỀU KHIỂN

### 1. Lệnh if else elseif:

Có các dạng sử dụng

```
if biểu thức logic  
    các phát biểu  
end
```

hoặc

```
if biểu thức logic  
    các phát biểu 1  
else  
    các phát biểu 2  
end
```



## V. CẤU TRÚC ĐIỀU KHIỂN

### 1. Lệnh if else elseif (tt)

hoặc

```
if biểu thức logic 1  
    các phát biểu 1  
elseif biểu thức logic 2  
    các phát biểu 2  
end
```



## V. CẤU TRÚC ĐIỀU KHIỂN

### 1. Lệnh if else elseif (tt)

Ví dụ 1:

```
if rem(a,2)==0
    disp('la mot so chan')
    b=a/2;
end
```

```
if n>0
    disp('la so duong')
elseif n==0
    disp('la so 0')
else
    disp('la so am')
end
```

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### 1. Lệnh if else elseif (tt)

Ví dụ 2: Hàm ngay\_trong\_thang.m

```
function y = ngay_trong_thang(th,nam)
if (th==4)|(th==6)|(th==9)|(th==11)
    y=30
elseif (th==2)
    if (rem(nam,4)~=0)
        y=28
    else
        y=29
    end
else
    y=31
end
```

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### 2. Lệnh switch case:

Chọn nhiều trường hợp

switch biểu thức (vô hướng hay chuỗi)

case trj\_1

Các phát biểu 1

case trj\_2

Các phát biểu 2

.....

otherwise

Các phát biểu khác

end

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### 2. Lệnh switch case (tt)

Ví dụ 1:

```
switch input_num
case -1
    disp('negative one');
case 0
    disp('zero');
case 1
    disp('positive one');
otherwise
    disp('other value');
end
```

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### 2. Lệnh switch case (tt)

Ví dụ 2:

```
switch var
    case 1
        disp('1');
    case {2,3,4}
        disp('2 or 3 or 4');
    case 5
        disp('5');
    otherwise
        disp('something else');
end
```

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### c. Lệnh while:

```
while biểu thức logic
    các phát biểu
end
```

Ví dụ 1:

```
n=1;
while prod(1:n) < 1e100    % prod tính tích các phần
    n=n+1;                 % tử cột của vectơ hay
end                        % ma trận
```

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### 4. Lệnh for:

```
for index=start:increment:end  
    các biểu thức  
end
```

Ví dụ 1:

```
x(1)=1;  
for i=2:6  
    x(i)=2*x(i-1);  
end
```

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### 4. Lệnh for (tt)

Ví dụ 2: Chương trình khởi tạo giá trị cho ma trận A(mxn)

```
for i=1:m  
    for j=1:n  
        A(i,j)=i+j;  
    end  
end
```

Giảng viên: Hoàng Xuân Dương



## V. CẤU TRÚC ĐIỀU KHIỂN

### 5. Gián đoạn bằng `continue`, `break` và `return`

- Trong vòng lặp `for` hay `while`, gọi `continue` thì ngay lập tức chu trình chuyển sang bước lặp kế tiếp, mọi lệnh chưa thực hiện của vòng lặp hiện tại sẽ bị bỏ qua
- Lệnh `break` mạnh hơn, ngừng vòng lặp đang tính
- Nếu `break` sử dụng ngoài vòng lặp `for` và `while`, nhưng nằm trong `script file` hoặc `function` thì sẽ dừng tại vị trí của `break`
- Lệnh `return` sử dụng để kết thúc sớm hàm trước khi gặp lệnh `end`

---

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### 5. Gián đoạn bằng `continue`, `break` và `return` (tt)

```
for m=3:1:7
    for n=2:1:m-1
        if mod(m, n) ~= 0
            continue;
        end
        fprintf('%2d không là một số nguyên tố !\n',m)
        break;
    end
    if n==m-1
        fprintf('%2d là một số nguyên tố !\n',m)
    end
end
```

---

Giảng viên: Hoàng Xuân Dương

## V. CẤU TRÚC ĐIỀU KHIỂN

### 5. Gián đoạn bằng continue, break và return (tt)

Kết quả:

```
!! 3 là một số nguyên tố !  
4 không là một số nguyên tố !  
!! 5 là một số nguyên tố !  
6 không là một số nguyên tố !  
!! 7 là một số nguyên tố !
```

Giảng viên: Hoàng Xuân Dương

## VI. BÀI TẬP:

1. Hãy cho biết kết quả khi chạy đoạn chương trình sau:

```
a = [1 2 3; 4 5 6; 7 8 9];  
[m n]=size(a);  
for i = (1-m):(n-1)  
    disp(triu(tril(a,i),i))  
end
```

2. Hãy cho biết kết quả khi chạy đoạn chương trình sau:

```
a = [1 2 3 4; 4 5 6 7; 7 8 9 10];  
m=size(a,2);  
for i = 1:m  
    disp(a(:,i));  
end
```

Giảng viên: Hoàng Xuân Dương

## VI. BÀI TẬP:

3. Hãy cho biết kết quả khi chạy đoạn chương trình sau:

```
n=4; giaithua=1
for i=1:n
    giaithua=giaithua*i;
    fprintf('%d! = %d\n',i,giaithua);
end
```

4. Hãy cho biết kết quả khi chạy đoạn chương trình sau:

```
a = pascal(3);
row = size(a,1); col = size(a,2);
for i = (1-row):(col-1)
    disp(tril(triu(a,i),i))
end
```

Giảng viên: Hoàng Xuân Dương

## VI. BÀI TẬP:

5. Hãy cho biết kết quả khi chạy đoạn chương trình sau:

```
a = [1 2 3 4; 4 5 6 7; 7 8 9 10];
[m n]=size(a);
for i = 1:m
    for j=1:n
        fprintf('%d ', a(i,j))
    end
end
```

6. Viết chương trình cho hiển thị trên màn hình dãy số :

1 2 3 4 5 6 7 8 ... n

Với n được nhập từ bàn phím

Giảng viên: Hoàng Xuân Dương

## VI. BÀI TẬP:

7. Viết đoạn chương trình tính tổng của n số tự nhiên, với n được nhập từ bàn phím
8. Viết một hàm **minmax.m** với tham số ngõ vào là một ma trận a, Kết quả trả về của hàm là giá trị phần tử lớn nhất và phần tử nhỏ nhất trong ma trận
9. Viết một hàm **findmax.m** với tham số ngõ vào là một ma trận a; Kết quả trả về của hàm là vị trí của phần tử lớn nhất (hàng, cột) trong ma trận
10. Viết một hàm **luythuabac3.m** với tham số vào là giá trị n; Trả về giá trị tổng lũy thừa bậc 3 của n phần tử

$$1^3 + 2^3 + 3^3 + \dots + n^3$$

Giảng viên: Hoàng Xuân Dương

## VI. BÀI TẬP:

11. Viết một hàm **tinhtong.m** có:  
 Nhận vào giá trị n  
 Trả về giá trị tổng các tích 2 số liên tiếp từ 1 đến n  

$$1*2 + 2*3 + 3*4 + \dots + (n-1)*n$$
12. Tìm giá trị lớn nhất của n sao cho tổng:  

$$1^2 + 2^2 + \dots + n^2$$
 nhận giá trị nhỏ hơn 100.
13. Mô phỏng một phép tính đơn giản cộng, trừ, nhân và chia 2 số.
14. Hàm tính **n!**. Sử dụng hàm để tính  $x=7!/(3!*4!)$

Giảng viên: Hoàng Xuân Dương



CHƯƠNG 4:

Vẽ đồ thị



Giảng viên: Hoàng Xuân Dương



CHƯƠNG 4: XỬ LÝ CÁC HÀM TOÁN HỌC

- I. ĐA THỨC
- II. PHÉP NỘI SUY
- III. HÀM CỦA HÀM
- IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC
- V. BÀI TẬP

Giảng viên: Hoàng Xuân Dương



**I. ĐA THỨC:**

- Đa thức được sắp xếp theo lũy thừa giảm
- Biểu diễn dưới dạng vector hàng, các phần tử là các hệ số của đa thức

Ví dụ:

Đa thức  $2x^3 - 8x + 7$  được biểu diễn bằng vector  $p$

$p = [2 \ 0 \ -8 \ 7]$

Giảng viên: Hoàng Xuân Dương

**I. ĐA THỨC:**

Một số hàm xử lý đa thức:

| Hàm      | Chức năng                                    |
|----------|--|
| conv     | Nhân đa thức                                 |
| poly     | Lập đa thức từ nghiệm                        |
| polyfit  | Xấp xỉ bằng đa thức                          |
| polyvalm | Tính ma trận đa thức                         |
| roots    | Tìm nghiệm đa thức                           |
| deconv   | Chia đa thức                                 |
| polyder  | Đạo hàm đa thức                              |
| polyval  | Tính giá trị đa thức                         |
| residue  | Tính thặng dư, khai triển riêng phần phân số |

Giảng viên: Hoàng Xuân Dương



**I. ĐA THỨC:****1. Nghiệm của đa thức:****➤ Nghiệm của đa thức bậc 2**

Ví dụ: Giải phương trình bậc 2:  $5x^2+6x+7=0$

```
>> p = [5 6 7]
>> r = roots(p)
r = -0.6000 + 1.0198i
    -0.6000 - 1.0198i
>> t = real(r)
t = -0.6000
    -0.6000
>> a = imag(r)
a = 1.0198
    -1.0198
```

---

Giảng viên: Hoàng Xuân Dương**I. ĐA THỨC:****1. Nghiệm của đa thức:****➤ Đa thức bậc n**

Ví dụ: Giải phương trình bậc 4:  $x^4 - 12x^3 + 25x + 116 = 0$

```
>> p = [1 -12 0 25 116]
>> r = roots(p)
r = 11.7473
    2.7028
    -1.2251 + 1.4672i
    -1.2251 - 1.4672i
>> t = real(r)
>> a = imag(r)
>> pp = poly(r)
pp = 1.0000 -12.0000 -0.0000 25.0000 116.0000
```

---

Giảng viên: Hoàng Xuân Dương

**I. ĐA THỨC:****2. Nhân 2 đa thức:**

Ví dụ: Cho 2 đa thức:  $y = x^3 + 2x^2 + 3x + 4$   
và  $z = x^3 + 4x^2 + 9x + 16$

```
>> p1 = [1 2 3 4]
p1 = 1 2 3 4
>> p2 = [1 4 9 16]
p2 = 1 4 9 16
>> p = conv(p1,p2)
p = 1 6 20 50 75 84 64
```

*Nếu nhân nhiều đa thức thì lập lại nhiều lần lệnh conv*

---

Giảng viên: Hoàng Xuân Dương

**I. ĐA THỨC:****3. Cộng đa thức:****➤ Hai đa thức cùng bậc:**

$$p = p1 + p2;$$

tương tự cho trừ đa thức

$$p = p1 - p2;$$

**➤ Hai đa thức khác bậc:**

Thêm các hệ số 0 vào đa thức có bậc thấp hơn để 2 đa thức có cùng bậc

---

Giảng viên: Hoàng Xuân Dương





**I. ĐA THỨC:****4. Chia đa thức:**

Ví dụ: Cho 2 đa thức:  $y = x^3 + 6x^2 + 12x + 8$   
 $z = x^2 + 1$

```
>> y = [1 6 12 8];
>> z = [1 0 1];
>> p = deconv(y,z)
p = 1 6
>> [p,r] = deconv(y,z)
p = 1 6
r = 0 0 11 2          % y=(x^2 + 1)(x + 6) + (11x + 2)
```

Giảng viên: Hoàng Xuân Dương

**I. ĐA THỨC:****5. Đạo hàm:**

Ví dụ: Cho đa thức  $y = x^3 + 6x^2 + 12x + 8$

```
>> y = [1 6 12 8]
y = 1 6 12 8
>> z = polyder(y);
z = 3 12 12          % z = 3x^2 + 12x + 12
```

Giảng viên: Hoàng Xuân Dương



## I. ĐA THỨC:

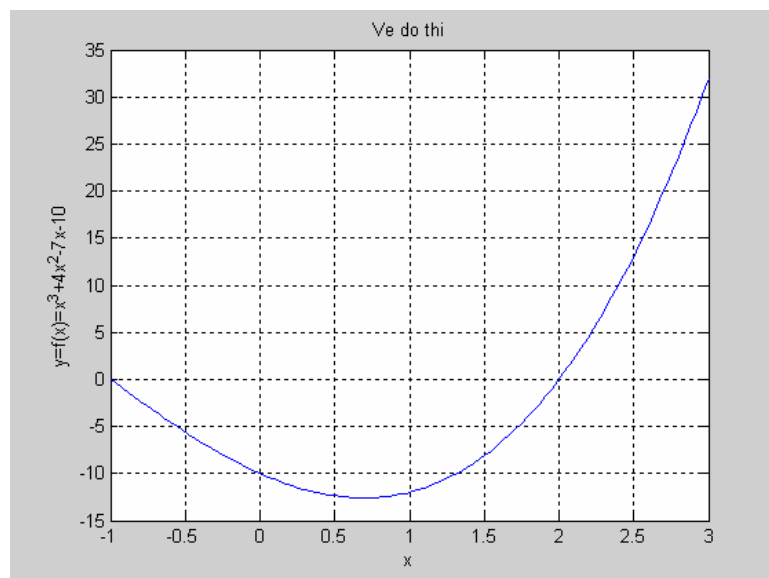
### 6. Vẽ đồ thị:

Ví dụ: đa thức  $y(x) = x^3 + 4x^2 - 7x - 10$

Cho các giá trị của  $x$ , tính các giá trị của  $y$  tương ứng

```
>> x = linspace(-1,3);
>> p = [1 4 -7 -10];
>> y = polyval(p,x);      % xác định y ứng với các giá trị x
>> plot(x,y)
>> xlabel('x')
>> ylabel('y = f(x) = x^3 + 4x^2 - 7x - 10');
>> title('Vẽ đồ thị');
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

**I. ĐA THỨC:****7. Đa thức hữu tỉ:**

Ví dụ:

Cho phân thức:

$$\frac{P(x)}{Q(x)} = \frac{2(4x+7)}{(x+1)(x+3)(x+4)}$$

Phân chia phân thức ra từng hệ số:

$$\frac{P(x)}{Q(x)} = \frac{A}{x+1} + \frac{B}{x+3} + \frac{C}{x+4} + k$$

Nếu chiều dài hay bậc của Q(x) lớn hơn P(x) thì k=0

Giảng viên: Hoàng Xuân Dương

**I. ĐA THỨC:****7. Đa thức hữu tỉ (tt)**

Giải:

```
>> num=2*[4 7];
>> den=poly([-1 ; -3 ; -4]);
>> [res,poles,k]=residue(num,den)
```

res = -6.0000

5.0000

1.0000

poles= -4.0000

-3.0000

-1.0000

k = [ ]

%

$$\frac{P(x)}{Q(x)} = \frac{1}{x+1} + \frac{5}{x+3} - \frac{6}{x+4}$$

Giảng viên: Hoàng Xuân Dương



**I. ĐA THỨC:****7. Đa thức hữu tỉ (tt)**

Ngược lại từ res, poles, k có thể tìm lại đa thức P(x), Q(x)

```
>> [P,Q]=residue(res,poles,k)
```

```
P = 0 8 14
```

```
Q = 1 8 19 12
```

**Bài tập:** Tìm các hệ số của các hàm sau

1.  $H(s)=10(s+2)/s(s+4)(s+5)$
2.  $H(s)=4/(s+1)(s+2)$
3.  $H(s)=10s/(s+1)(s+4)$
4.  $H(s)=(s+1)/s(s+2)(s+3)$
5.  $H(s)=10s^2/(s+1)(s+5)$

Giảng viên: Hoàng Xuân Dương

**II. PHÉP NỘI SUY:****1. Nội suy một chiều:**

Hàm nội suy (interpolation) một chiều thông dụng nhất:

```
Yi=interp1(X,Y,Xi)
```

```
Yi=interp1(Y,Xi)
```

```
Yi=interp1(X,Y,Xi,'method')
```

```
Yi=interp1(X,Y,Xi,'method','extrap')
```

```
Yi=interp1(X,Y,Xi,'method',extrapval)
```

*Y là tập dữ liệu ứng với giá trị cho bởi tập X*

*Yi là giá trị dữ liệu được nội suy ở giá trị Xi*

Giảng viên: Hoàng Xuân Dương

## II. PHÉP NỘI SUY:

### 1. Nội suy một chiều (tt)

**method** là phương pháp sử dụng khi nội suy:

- **nearest**: nội suy cận gần nhất
- **linear**: nội suy tuyến tính (mặc định)
- **spline, pchip, cubic, v5cubic**: nội suy bậc 3

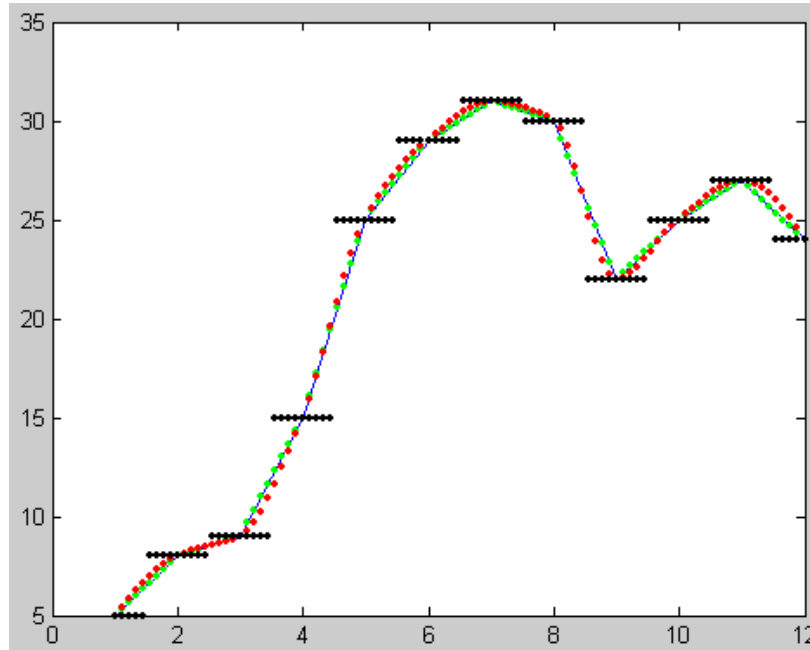
**extrap**: dùng khi ngoại suy, các giá trị ngoài tầm x, giá trị trả về là **extrapval**

## II. PHÉP NỘI SUY:

### 1. Nội suy một chiều (tt)

Ví dụ:

```
>> hour=1:12;  
>> temps=[5 8 9 15 25 29 31 30 22 25 27 24];  
>> plot(hour,temps,hour,temps,'.')  
>> h=linspace(1,12);  
>> t =interp1(hour,temps,h,'linear');  
>> t1=interp1(hour,temps,h,'cubic');  
>> t2=interp1(hour,temps,h,'nearest');  
>> hold on  
>> plot(h,t,'g.')  
>> plot(h,t1,'r.')  
>> plot(h,t2,'k.')
```



Giảng viên: Hoàng Xuân Dương

## II. PHÉP NỘI SUY:

### 2. Nội suy hai chiều:

- Nội suy 2 chiều dùng cho hàm 2 biến  $z=f(x,y)$
- Hàm nội suy hai chiều thông dụng nhất:

$Z_i = \text{interp2}(X, Y, Z, X_i, Y_i)$

$Z_i = \text{interp2}(Z, X_i, Y_i)$

$Z_i = \text{interp2}(Z, \text{ntimes})$

$Z_i = \text{interp2}(X, Y, Z, X_i, Y_i, \text{'method'})$

$Z$  là tập dữ liệu ứng với giá trị cho bởi tập  $X, Y$

$Z_i$  là giá trị dữ liệu được nội suy ở giá trị  $X_i, Y_i$

Giảng viên: Hoàng Xuân Dương

## II. PHÉP NỘI SUY:

## 2. Nội suy hai chiều (tt)

Ví dụ: Cho một tập dữ liệu lương nhân viên:

```
>> years=1950:10:1990
>> service=10:10:30
>> wage=[ 150.697      199.592      187.625
          179.323      195.072      250.287
          203.212      179.092      322.767
          226.505      153.706      426.730
          249.633      120.281      598.243]
```

Nội suy xem một nhân viên có 15 năm phục vụ lãnh lương bao nhiêu vào năm 1975

```
>> w=interp2(service,years,wage,15,1975)
w= 190.6287
```

Giảng viên: Hoàng Xuân Dương

## II. PHÉP NỘI SUY:

## 3. Nội suy nhiều chiều:

$V_i = \text{interp3}(X, Y, Z, V, X_i, Y_i, Z_i)$

$V_i = \text{interp}(X_1, X_2, X_3, \dots, V, Y_1, Y_2, Y_3, \dots)$

Giảng viên: Hoàng Xuân Dương

### III. HÀM CỦA HÀM:

Matlab biểu diễn các hàm toán học theo 2 cách: định nghĩa bằng hàm M và định nghĩa bằng **inline**

Ví dụ: 
$$y = \frac{10(s+3)}{s(s+5)(s+10)}$$

có thể tạo file **hamtruyen.m**

```
function y=hamtruyen(s)
y=10*(s+3)/(s*(s+5)*(s+10))
```

hay định nghĩa từ dòng lệnh:

```
>> f=inline('10*(s+3)/(s*(s+5)*(s+10))');
```

có thể tạo hàm nhiều biến với **inline**

```
>> f=inline('y*sin(x)+x*sin(y)','x','y')
```

Giảng viên: Hoàng Xuân Dương

### III. HÀM CỦA HÀM:

Hàm **feval** dùng để tính giá trị của một hàm theo biến:

Ví dụ:

```
>> f=inline('sin(x)+sin(y)');
```

```
>> feval(f,90,45)
```

```
ans=1.7449
```

Ví dụ: **hamtruyen.m**

```
function y=hamtruyen(x)
y=2*x^2-3*x+1;
```

```
>> feval(@hamtruyen,3)
```

```
ans=10
```

Giảng viên: Hoàng Xuân Dương



### III. HÀM CỦA HÀM:

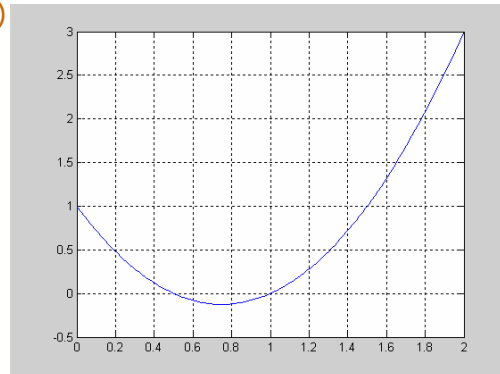
Hàm `fplot` dùng để vẽ hàm theo biến:

Ví dụ: `hamtruyen.m`

```
function y=hamtruyen(x)
y=2*x^2-3*x+1;
```

```
>> fplot(@hamtruyen,[0,2])
```

```
>> grid on
```



Giảng viên: Hoàng Xuân Dương

### IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

Matlab không chỉ tính toán trên các số cụ thể mà còn có thể thực hiện tính toán trên ký hiệu → Có thể sử dụng một chuỗi biểu thức để biểu diễn hàm

Ví dụ:

$$\frac{1}{2x^n} \Rightarrow '1/(2 * x^n)'$$

$$\frac{1}{\sqrt{2x}} \Rightarrow '1/\text{sqrt}(2 * x)'$$

$$\cos(x^2) - \sin(2x) \Rightarrow '\cos(x^2) - \sin(2 * x)'$$

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \Rightarrow \text{sym}(['a, b; c, d'])$$

Giảng viên: Hoàng Xuân Dương



#### IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

| Hàm      | Ý nghĩa              | Hàm      | Ý nghĩa               |
|----------|----------------------|----------|-----------------------|
| syms     | Khai báo biến        | symop    | Tạo hàm mới           |
| sym      | Định nghĩa hàm       | symsum   | Tổng hàm              |
| diff     | Đạo hàm              | numden   | Tử+mẫu số hàm         |
| int      | Tích phân            | compose  | Hàm của hàm           |
| linsolve | Giải hệ phương trình | finverse | Tìm hàm ngược         |
| symadd   | Cộng hàm             | poly2sym | Tìm hệ số của hàm     |
| symsub   | Trừ hàm              | sym2poly | Tạo hàm từ hệ số      |
| symmul   | Nhân hàm             | eval     | Tính trị hàm          |
| symdiv   | Chia hàm             | numeric  | Tính trị hàm          |
| sympow   | Lũy thừa hàm         | subs     | Thay đổi giá trị biến |

Giảng viên: Hoàng Xuân Dương



#### IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

| Hàm      | Ý nghĩa             | Hàm      | Ý nghĩa                   |
|----------|---------------------|----------|---------------------------|
| ezplot   | Vẽ hàm              | dsolve   | Giải phương trình vi phân |
| factor   | Phân tích tp bậc 1  | laplace  | Biến đổi laplace          |
| simplify | Đơn giản hàm        | ilaplace | Biến đổi laplace ngược    |
| simple   | Tối giản hàm        | fourier  | Biến đổi fourier          |
| pretty   | Biểu diễn trực quan | ifourier | Biến đổi fourier ngược    |
| taylor   | Khai triển taylor   | ztrans   | Biến đổi z                |
| collect  | Khai triển hàm      | iztrans  | Biến đổi z ngược          |
| horner   |                     | bode     | Vẽ biểu đồ bode           |
| expand   | Khai triển hàm      | freqs    | Vẽ đáp ứng tần số         |
| solve    | Giải phương trình   | nyquist  | Vẽ biểu đồ Nyquist        |

Giảng viên: Hoàng Xuân Dương



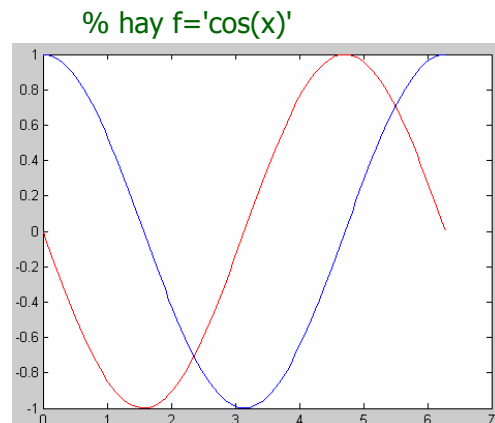
## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 1. Đạo hàm và vi phân gần đúng:

## ➤ Đạo hàm của hàm lượng giác:

Ví dụ:

```
>> f=sym('cos(x)')
f = cos(x)
>> y=diff(f)
y = -sin(x)
>> x=linspace(0,2*pi);
>> plot(x,eval(f),x,eval(y),'r')
```



Giảng viên: Hoàng Xuân Dương

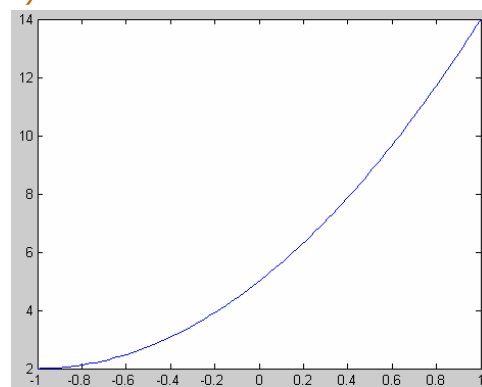
## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 1. Đạo hàm và vi phân gần đúng:

## ➤ Đạo hàm một đa thức:

Ví dụ 1:

```
>> f = diff('x^3+3*x^2+5*x+2')
f = 3*x^2+6*x+5
>> x=linspace(-1,1);
>> y=polyval([3 6 5],x);
>> plot(x,y)
```



Giảng viên: Hoàng Xuân Dương

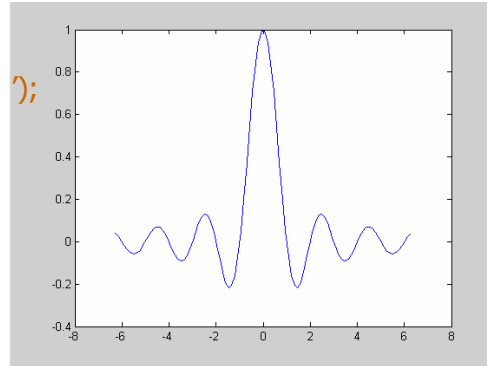
## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 1. Đạo hàm và vi phân gần đúng:

## ➤ Đạo hàm một đa thức:

Ví dụ 2:

```
>> D=['sin(x) ','cos(x) ','sinc(x)'];
>> x=linspace(-2*pi,2*pi);
>> clf
>> n=input('Chọn hình để vẽ: ');
>> plot(x,eval(D(n,:)))
```



Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 1. Đạo hàm và vi phân gần đúng:

## ➤ Đạo hàm hàm mũ:

Ví dụ:

```
>> diff('x^n','x')           % đạo hàm theo x, diff('x^n')
ans = x^n*n/x
>> diff('x^n','n')
ans = x^n*log(x)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 1. Đạo hàm và vi phân gần đúng:

## ➤ Đạo hàm đa thức hữu tỉ:

Ví dụ:

```
>> diff('x/(1-x^2)')
ans = 1/(1-x^2)+2*x^2/(1-x^2)^2
```

Rút gọn biểu thức:

```
>> simplify(sym('1/(1-x^2)+2*x^2/(1-x^2)^2'))
ans = (1+x^2)/(-1+x^2)^2
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 1. Đạo hàm và vi phân gần đúng:

## ➤ Đạo hàm mảng:

Ví dụ:

```
>> syms x % định nghĩa biến
>> A=[cos(x),sin(x);-sin(x),cos(x)]
A = [ cos(x), sin(x)]
     [-sin(x), cos(x)]
>> diff(A)
ans =
     [-sin(x), cos(x)]
     [-cos(x), -sin(x)]
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 1. Đạo hàm và vi phân gần đúng:

## ➤ Đạo hàm và vi phân cấp cao:

Ví dụ:

Đạo hàm cấp 2

```
>> syms x
```

```
>> diff(sin(x),2)
```

```
ans = -sin(x)
```

Đạo hàm cấp 3

```
>> diff(x^3,3)
```

```
ans = 6
```

---

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 1. Đạo hàm và vi phân gần đúng:

## ➤ Đạo hàm và vi phân cấp cao (tt)

Ví dụ:

Đạo hàm cấp 2 theo a

```
>> syms x a;
```

```
>> f='a^2*x^3+x^2'
```

```
f = a^2*x^3+x^2
```

```
>> diff(f,a,2)
```

```
ans = 2*x^3
```

---

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 2. Tích phân:

## ➤ Tích phân bất định:

Ví dụ:

```
>> syms x x1 alpha u t;
>> int(1/(1+x^2))           % tích phân mặc định theo x
ans = atan(x)
>> int(1/(1+x^2),t)         % tích phân theo t
ans = 1/(1+x^2)*t
>> int(sin(alpha*u),alpha)
ans = -1/u*cos(alpha*u)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 2. Tích phân:

## ➤ Tích phân bất định (tt)

```
>> int(sin(alpha*u),u)
ans = -1/alpha*cos(alpha*u)
```

Nếu khi tính tích phân hay nguyên hàm của một lượng quá lớn hay phức tạp, đòi hỏi chiếm bộ nhớ lớn thì nó không thực hiện và trả về kết quả

```
>> int('log(x)/exp(x^2)')
ans = int(log(x)/exp(x^2),x)
>> int('sin(x)/x')
ans = sinint(x)           % sinint(x) = int(sin(t)/t,t,0,x)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 2. Tích phân:

## ➤ Tích phân mảng

Ví dụ:

```
>> syms x t;
>> A=[cos(x*t),sin(x*t);-sin(x*t),cos(x*t)];
>> int(A,t)
ans =      [ 1/x*sin(x*t), -cos(x*t)/x]
          [  cos(x*t)/x, 1/x*sin(x*t)]
>> int([exp(t),exp(alpha*t)])
ans =      [exp(t), exp(alpha*t)/alpha]
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 2. Tích phân:

## ➤ Tích phân xác định

Ví dụ:

```
>> syms x x1 alpha t;
>> int(x1*log(1+x1),0,1)      % tích phân từ 0 → 1
ans = 1/4
>> int('sin(s+2*x)', 's', pi/2, pi) % s chưa khai báo
ans = 2*cos(x)^2-1-2*sin(x)*cos(x)
>> int(sin(x),0,t)             % cận không được trùng
ans = -cos(t)+1                % với đối số của hàm
```

Giảng viên: Hoàng Xuân Dương



## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

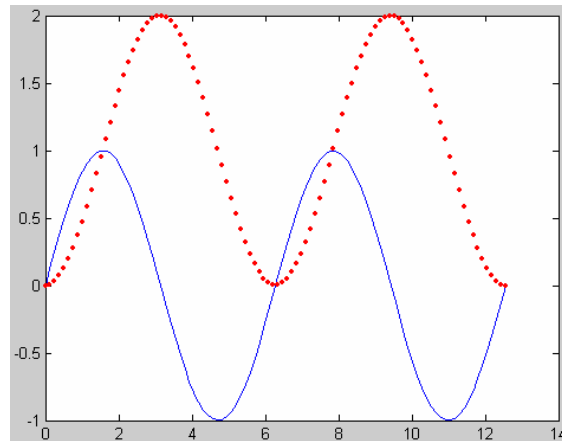
## 2. Tích phân:

## ➤ Tích phân xác định (tt)

Ví dụ:

Biểu diễn bằng đồ thị

```
>> t=linspace(0,4*pi);
>> y1=sin(t);
>> y2=1-cos(t);
>> plot(t,y1,t,y2,'r')
```



Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 2. Tích phân:

## ➤ Tích phân xác định (tt)

Tổng quát:

```
>> int('sin(s+2*x)','m','n')
ans = -1/2*cos(s+2*n)+1/2*cos(s+2*m)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 3. Ma trận:

## ➤ Định thức:

Ví dụ:

```
>> M=sym('[a,b;c,d]')
```

```
M = [ a, b]
```

```
     [ c, d]
```

```
>> determ(M)
```

*% định thức*

```
ans = a*d-b*c
```

Hay:

```
>> M='[a,b;c,d]'
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 3. Ma trận:

## ➤ Định thức (tt)

Giải hệ phương trình

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

```
>> b=[2;-1;0]
```

```
>> x=linsolve(A,b)
```

```
x = [ 7/2]
```

```
     [ 2]
```

```
     [ 5/2]
```

*% x = linsolve(A,b) tương đương với x = sym(A)\sym(b)*

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 3. Ma trận:

## ➤ Tính toán với ma trận:

Ví dụ:

```
>> G=sym('[cos(t),sin(t);-sin(t),cos(t)]')
```

```
G = [ cos(t), sin(t)]
      [-sin(t), cos(t)]
```

Cộng mỗi phần tử G cho t

```
>> symadd(G,'t')
```

```
ans =
      [ cos(t)+t, sin(t)+t]
      [-sin(t)+t, cos(t)+t]
```

Giảng viên: Hoàng Xuân Dương

Trừ mỗi phần tử G cho t

```
>> symsub(G,'t')
```

Nhân mỗi phần tử G cho t

```
>> symmul(G,'t')
```

Chia mỗi phần tử G cho t

```
>> symdiv(G,'t')
```

Nhân 2 ma trận

```
>> symmul(G,G)
```

```
ans =
      [ cos(t)^2-sin(t)^2, 2*cos(t)*sin(t)]
      [-2*cos(t)*sin(t), cos(t)^2-sin(t)^2]
```

Giảng viên: Hoàng Xuân Dương



Rút gọn biểu thức

```
>> simplify(ans)
```

```
ans =
```

```
    [ 2*cos(t)^2-1, 2*cos(t)*sin(t)]  
    [-2*cos(t)*sin(t), 2*cos(t)^2-1]
```

Hay:

```
>> simple(ans)
```

*% sau một số bước rút gọn*

```
ans =    [ cos(2*t), sin(2*t)]  
        [-sin(2*t), cos(2*t)]
```



#### IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

##### 4. Đa thức:

###### ➤ Tính toán với đa thức:

Ví dụ:

```
>> f='2*x^2+3*x-5';g='x^2+x+7';
```

```
>> symadd(f,g)
```

*% Tính biểu thức f+g*

```
ans = 3*x^2+4*x+2
```

```
>> symsub(f,g)
```

*% Tính biểu thức f-g*

```
ans = x^2+2*x-12
```

```
>> symmul(f,g)
```

*% Tính biểu thức f\*g*

```
ans = (2*x^2+3*x-5)*(x^2+x+7)
```



## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 4. Đa thức:

## ➤ Tính toán với đa thức (tt)

```
>> symdiv(f,g)           % Tính biểu thức f/g
ans = (2*x^2+3*x-5)/(x^2+x+7)
>> sympow(f,'3')         % Tính biểu thức f^3
ans = (2*x^2+3*x-5)^3
>> sympow(f,'1/2')       % Tính biểu thức f^1/2
ans = (2*x^2+3*x-5)^(1/2)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 4. Đa thức:

## ➤ Xây dựng đa thức từ các ký hiệu:

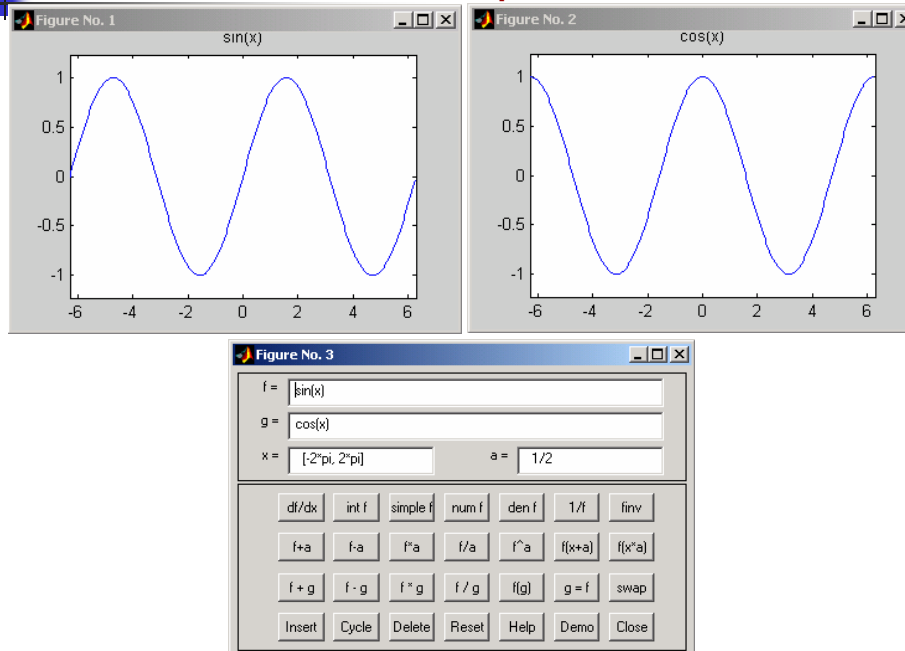
Ví dụ:

```
>> f='cos(x)';g='sin(2*x)';
>> symop(f,'/',g,'+',3)
ans = cos(x)/sin(2*x)+3
```

## ➤ Kiểm tra lại các phép toán đa thức:

```
>> funtool
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

#### IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

##### 5. Hàm hữu tỉ:

➤ **Xác định tử số và mẫu số của biểu thức**

Ví dụ:

```
>> f=sym('a*x^2/(b-x)');
>> [n,d]=numden(f)
n = -a*x^2
d = -b+x
>> f=sym('3/2*x^2+2/3*x-3/5');
>> [n,d]=numden(f)
n = 45*x^2+20*x-18
d = 30
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 5. Hàm hữu tỉ:

## ➤ Tính tổng của 2 đa thức hữu tỉ:

Ví dụ:

```
>> f=sym('(x^2+3)/(2*x-1)');g=sym('3*x/(x-1)');
```

```
>> [n,d]=numden(f+g)
```

```
n = x^3+5*x^2-3
```

```
d = (2*x-1)*(x-1)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 5. Hàm hữu tỉ:

## ➤ Ma trận:

Ví dụ:

```
>> k=sym('[3/2,(2*x+1)/3;4/x^2,(3*x+4)]')
```

```
k = [ 3/2, (2*x+1)/3]
```

```
     [ 4/x^2, (3*x+4)]
```

```
>> [n,d]=numden(k)
```

```
n = [ 3, 2*x+1]
```

```
     [ 4, 3*x+4]
```

```
d = [ 2, 3]
```

```
     [ x^2, 1]
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 6. Tìm hàm:

Ví dụ:

```
>> f=sym('1/(1+x^2)');g=sym('sin(x)');
>> h=sym('1/(1+u^2)');k=sym('sin(v)');
>> compose(f,g)           % tính f(g(x))
ans = 1/(1+sin(x)^2)
>> compose(g,f)           % tính g(f(x))
ans = sin(1/(x^2+1))
>> compose(h,k,'u','v')   % tính h(k(v))
ans = 1/(1+sin(v)^2)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 7. Tìm hàm ngược:

Ví dụ:

```
>> finverse(sym('1/x'))
ans = 1/x
>> finverse(sym('exp(x)'))
ans = log(x)
>> finverse(sym('sin(x)'))
ans = asin(x)
>> finverse(sym('x^2'))   % lỗi
ans = x^(1/2)
```

Giảng viên: Hoàng Xuân Dương



## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 8. Chuỗi:

Cú pháp: `symsum(f,a,b)` → tính tổng hàm f từ a đến b

## ➤ Tổng hữu hạn:

```
>> symsum(sym('x^2'),0,'x-1')
```

```
ans = 1/3*x^3-1/2*x^2+1/6*x
```

$$\% \sum_0^{x-1} x^2 = \frac{1}{3}x^3 - \frac{1}{2}x^2 + \frac{1}{6}x$$

```
>> factor(ans)
```

*% đổi lại dạng kết quả*

```
ans = 1/6*x*(2*x-1)*(x-1)
```

*% có thể dùng simple(ans)*

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 8. Chuỗi (tt)

```
>> symsum(sym('(2*n-1)^2'),1,'n')
```

```
ans = 11/3*n+8/3-4*(n+1)^2+4/3*(n+1)^3
```

```
>> factor(ans)
```

```
ans = 1/3*n*(2*n-1)*(2*n+1)
```

$$\% \sum_1^n (2n-1)^2 = \frac{n(2n-1)(2n+1)}{3}$$

Không tính được các tổng hội tụ có điều kiện

```
>> symsum(sym('x^k'),0,'n')
```

```
ans = sum(x^k,x = 0 .. n)
```

$$\% \sum_{k=0}^n x^k = \frac{1}{1-x}, \quad |x| < 1$$

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 8. Chuỗi:

## ➤ Tổng vô hạn:

```
>> symsum(sym('k'),0,inf)
```

```
ans = inf
```

Khi không có điều kiện hội tụ:

```
>> symsum(sym('x^k'),0,'inf')
```

```
ans = sum(x^k,x = 0 .. inf)
```

Tính tổng  $1/(2^n-1)^2$  với n từ 1..vô cùng

```
>> symsum(sym('1/(2^n-1)^2'),1,inf)
```

```
ans = 1/8*pi^2
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 9. Thay đổi giá trị hàm và biến:

## ➤ Hàm:

```
>> phi='(1+sqrt(5))/2'
```

```
phi = (1+sqrt(5))/2
```

```
>> numeric(phi)
```

*% đổi dạng số*

```
ans = 1.6180
```

Hoặc dùng hàm tính trị

```
>> eval(phi)
```

```
ans = 1.6180
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 9. Thay đổi giá trị hàm và biến:

## ➤ Hàm (tt)

```
>> syms x
>> f=2*x^2+x^3-3*x+5;
>> n=sym2poly(f)           % tìm các hệ số đa thức
n = 1    2   -3    5
% tạo lại đa thức từ hệ số
>> p=poly2sym(n)           % mặc định là x
p = 2*x^2+x^3-3*x+5
>> p=poly2sym(n,'s')       % thay x bằng s
p = s^3+2*s^2-3*s+5
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 9. Thay đổi giá trị hàm và biến:

## ➤ Biến:

```
>> syms f g h x a b c
>> f=a*x^2+b*x+c;
Thay x bằng s
>> subs(f,'s',x)
ans = a*s^2+b*s+c
Thay a bằng alpha
>> subs(f,'alpha',a)
ans = alpha*x^2+b*x+c
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 9. Thay đổi giá trị hàm và biến:

➤ **Biến (tt)**

```
>> g=3*x^2+5*x-4;  
>> h=subs(g,'x','2')  
h = 3*(2)^2+5*(2)-4  
>> numeric(h)  
ans = 18  
>> isstr(h)  
ans = 0
```

*% h không là chuỗi*

Giảng viên: Hoàng Xuân Dương

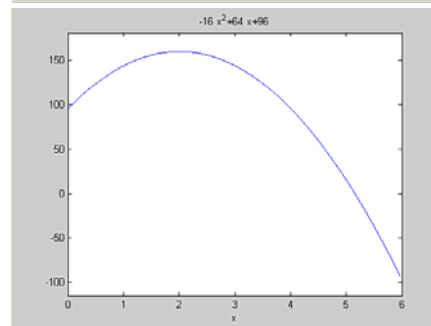
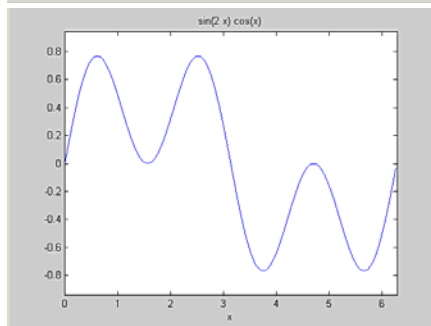
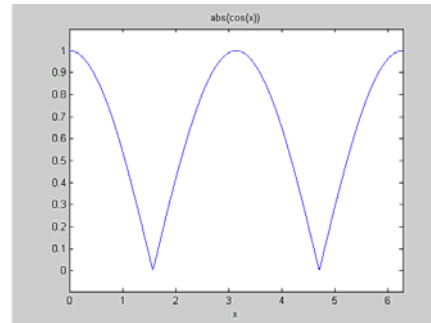
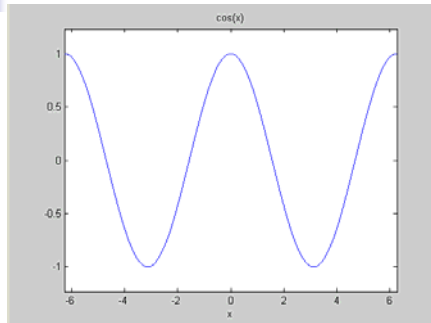
## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 10. Vẽ đồ thị theo biểu thức:

Ví dụ:

```
>> ezplot('cos(x)') % vẽ hàm cos và tự điền nhãn vào  
>> ezplot('abs(cos(x))',[0 2*pi])  
>> ezplot('sin(2*x)*cos(x)',[0 2*pi])  
>> ezplot('-16*x^2+64*x+96',[0 6])
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

#### IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

##### 11. Định dạng và đơn giản biểu thức:

###### ➤ Sắp xếp

```
>> int('sin(s+2*x)', 's', pi/2, pi)
ans = 2*cos(x)^2-1-2*sin(x)*cos(x)
>> pretty(ans)
      2
2 cos(x) - 1 - 2 sin(x) cos(x)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 11. Định dạng và đơn giản biểu thức:

## ➤ Khai triển Taylor

```
>> syms x
>> f=taylor(log(x+1)/(x-5))
f=-1/5*x+3/50*x^2-41/750*x^3+293/7500*x^4-1207/37500*x^5
>> pretty(f)
```

$$-\frac{1}{5}x + \frac{3}{50}x^2 - \frac{41}{750}x^3 + \frac{293}{7500}x^4 - \frac{1207}{37500}x^5$$

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 11. Định dạng và đơn giản biểu thức:

## ➤ Khai triển hằng đẳng thức

```
>> f=sym('x^2-1')
f = x^2-1
>> factor(f)
ans = (x-1)*(x+1)
>> collect(ans) % lấy lại biểu thức f
ans = x^2-1
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 11. Định dạng và đơn giản biểu thức:

## ➤ Tổng quát:

```
>> f=sym('(x^2-1)*(x-2)*(x-3)')
f = (x^2-1)*(x-2)*(x-3)
>> collect(f)
ans = x^4-5*x^3+5*x^2+5*x-6
>> horner(ans)
ans = -6+(5+(5+(x-5)*x)*x)*x
>> factor(ans)
ans = (x-1)*(x-2)*(x-3)*(x+1)
>> expand(f)
ans = x^4-5*x^3+5*x^2+5*x-6
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 11. Định dạng và đơn giản biểu thức:

## ➤ Đơn giản biểu thức

```
>> simplify(sym('log(2*x/y)'))
ans = log(2)+log(x/y)
>> simplify(sym('sin(x)^2+3*x+cos(x)^2-5'))
ans = -4+3*x
>> simplify(sym('(-a^2+1)/(1-a)'))
ans = a+1
>> f=sym('(1/x^3+6/x^2+12/x+8)^(1/3)');
>> simple(f)
ans = (2*x+1)/x
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 12. Giải phương trình bậc n:

## ➤ Giải phương trình bậc 2

```
>> solve('x^2+2*x-1')
ans =      [ 2^(1/2)-1]
           [ -1-2^(1/2)]

>> numeric(ans)
ans =
    0.4142
   -2.4142
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 12. Giải phương trình bậc n:

## ➤ Giải phương trình bậc 2 (tt)

```
>> [x,y]=solve('x^2+x*y+y=3','x^2-4*x+3=0')
x =
    [ 1]
    [ 3]
y =
    [ 1]
    [-3/2]
```

Giảng viên: Hoàng Xuân Dương



## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 12. Giải phương trình bậc n:

## ➤ Giải phương trình bậc 3

```
>> solve('x^3+2*x^2-4*x+1')
ans =      [      1]
           [ -3/2+1/2*13^(1/2)]
           [ -3/2 -1/2*13^(1/2)]

>> numeric(ans)
ans =      1.0000
           0.3028
          -3.3028
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 12. Giải phương trình bậc n:

## ➤ Giải phương trình lượng giác

```
>> solve('2*cos(x)+2')
ans = pi
>> solve('cos(x)=sin(x)')
ans = 1/4*pi
>> solve('exp(x)=tan(x)')
ans = 1.3063269404230792361743566584407
>> numeric(ans)
ans = 1.3063
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 13. Giải hệ phương trình bậc nhất tuyến tính:

Ví dụ:

```
>> y1=sym('2*x1+2*x2-3*x3+x4-4');
>> y2=sym('4*x1+3*x2-x3+2*x4-6');
>> y3=sym('8*x1+5*x2-3*x3+4*x4-12');
>> y4=sym('3*x1+3*x2-2*x3+2*x4-6');
>> [x1,x2,x3,x4]=solve(y1,y2,y3,y4,'x1,x2,x3,x4')
x1 = 1/3
x2 = 1/3
x3 = -1/3
x4 = 5/3
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 14. Giải phương trình vi phân:

## ➤ Phương trình vi phân cấp 1

Ví dụ: giải phương trình vi phân  $y' = dy/dx = y \tan(x) + \cos(x)$ 

```
>> dsolve('Dy=y*tan(x)+cos(x)','x')
ans = (1/4*sin(2*x)+1/2*x+C1)/cos(x)    % C1 là đk đầu
```

Ví dụ: giải phương trình  $y' = dy/dx = 1 + y^2$  với  $y(0) = 1$ 

```
>> dsolve('Dy=1+y^2','y(0)=1','x')
ans = tan(x+1/4*pi)
```

Hay:

```
>> dsolve('Dy-y^2-1=0','y(0)=1','x')
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 14. Giải phương trình vi phân:

## ➤ Phương trình vi phân cấp 2

Ví dụ: giải  $y'' = \cos(2x) - y$  với  $y(0)=1$  và  $y'(0)=0$

```
>> dsolve('D2y=cos(2*x)-y','Dy(0)=0','y(0)=1','x')
```

```
ans = 4/3*cos(x)-1/3*cos(2*x)
```

Ví dụ: giải  $y'' - 2y' - 3y = 0$  với  $y(0)=1$  và  $y(1)=1$

```
>> y= dsolve('D2y-2*Dy-3*y=0','y(0)=0','y(1)=1','x')
```

```
y = 1/(exp(3)-exp(-1))*exp(3*x)-1/(exp(3)-exp(-1))*exp(-x)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 14. Giải phương trình vi phân:

## ➤ Phương trình vi phân cấp 2 (tt)

```
>> y=simple(y);
```

```
y = (exp(-x)-exp(3*x))/(exp(-1)-exp(3))
```

```
>> pretty(y)
```

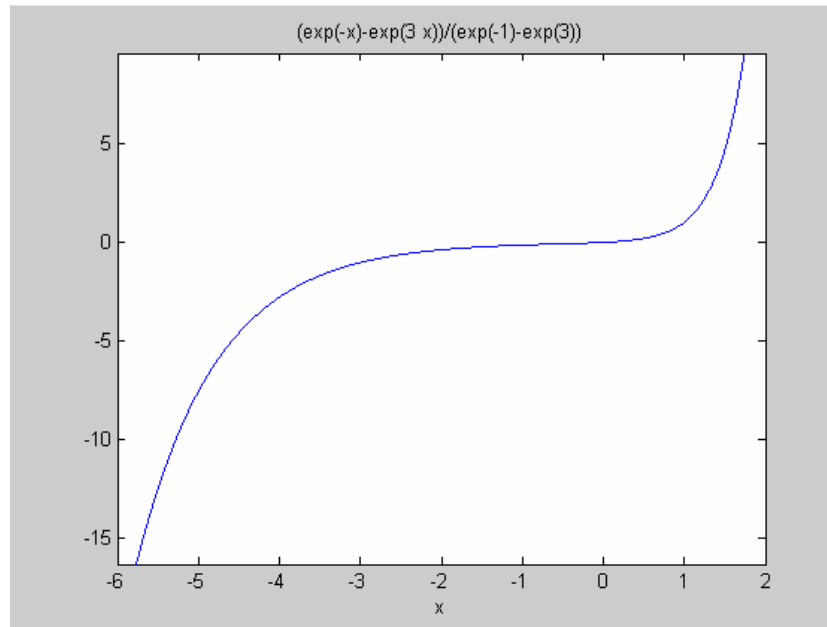
```
exp(-x) - exp(3 x)
```

```
-----
```

```
exp(-1) - exp(3)
```

```
>> ezplot(y,[-6 2])
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

#### IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

##### 14. Giải phương trình vi phân:

###### ➤ Hệ phương trình vi phân cấp 1

Ví dụ:  $\frac{df}{dx} = 3f + 4g$  với  $f(0) = 0$

$\frac{dg}{dx} = -4f + 3g$  với  $g(0) = 0$

```
>>[f,g] = dsolve('Df=3*f+4*g','Dg=-4*f+3*g','f(0)=0','g(0)=1','x')
```

```
f = exp(3*x)*sin(4*x)
```

```
g = exp(3*x)*cos(4*x)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 15. Các phép biến đổi:

➤ **Biến đổi Laplace:**

```
>> syms t s x a w
>> laplace(sin(t))
ans = 1/(s^2+1)
>> ilaplace(1/(s^2+1))           % Biến đổi ngược
ans = sin(t)
>> laplace(12*exp(-3*x))
ans = 12/(s+3)
>> laplace(sym(1))
ans = 1/s
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 15. Các phép biến đổi:

➤ **Biến đổi Laplace (tt)**

```
>> laplace(sym(diff(x^2)))
ans = 2/s^2
>> laplace(cos(x*w),w,t)
ans = t/(t^2+x^2)
>> laplace(sin(w*x),t)
ans = w/(t^2+w^2)
>> laplace(x^sym(3/2),t)
ans = 3/4/t^(5/2)*pi^(1/2)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 15. Các phép biến đổi:

➤ **Biến đổi Laplace (tt)**

Ví dụ: Tìm nghiệm một hệ thống biết hàm truyền đạt:

$$H(S) = \frac{1}{(s+1)(s^2+5s+6)} + \frac{(s+6)}{(s^2+5s+6)}$$

```
>> ilaplace(1/((s+1)*(s^2+5*s+6))+(s+6)/(s^2+5*s+6))
ans = 1/2*exp(-t)-5/2*exp(-3*t)+3*exp(-2*t)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 15. Các phép biến đổi:

➤ **Biến đổi Fourier:**

```
>> syms t x
>> fourier(exp(-x^2),t)
ans = pi^(1/2)*exp(-1/4*t^2)
>> ifourier(pi^(1/2)*exp(-1/4*t^2))           % Biến đổi ngược
ans = 3991211251234741/2251799813685248/pi^(1/2)*exp(-x^2)
>> factor(3991211251234741/2251799813685248/pi^(1/2)*exp(-x^2))
ans = exp(-x^2)
>> simplify(3991211251234741/2251799813685248/pi^(1/2)*exp(-x^2))
ans = exp(-x^2)
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 15. Các phép biến đổi:

➤ **Biến đổi z:**

```
>> syms a n z
>> ztrans(a^n)
ans = z/a/(z/a-1)
>> simplify(ans)
ans = -z/(-z+a)
>> iztrans(z/(z-a))      % Biến đổi ngược
ans = a^n
```

Giảng viên: Hoàng Xuân Dương

## IV. XỬ LÝ HÀM DƯỚI DẠNG CHUỖI BIỂU THỨC:

## 15. Các phép biến đổi:

➤ **Vẽ trong miền tần số:**

Cho hệ thống có hàm số chuyển:

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

Nhập vào tử và mẫu số:

```
>> num=[2 5 1];den=[1 2 3];
```

Vẽ giản đồ bode

```
>> bode(num,den)
```

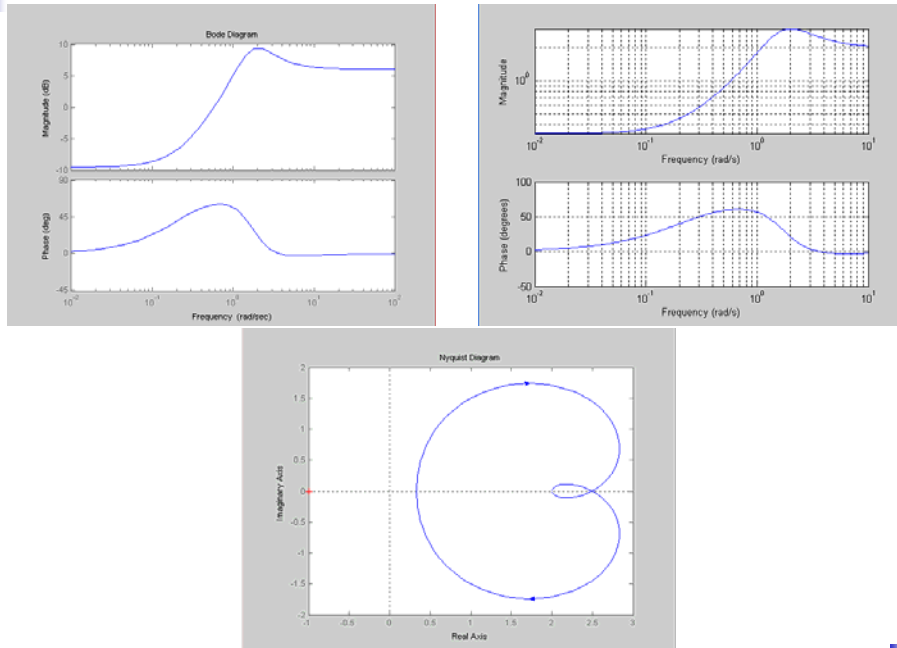
Vẽ đáp ứng tần số

```
>> freqs(num,den)
```

Vẽ giản đồ Nyquist

```
>> nyquist(num,den)
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

### V. BÀI TẬP:

1) Tính trị các đa thức sau với x=1

$$P1 = \frac{(2x^2 + 6x - 1)(x^3 + 4x - 2)}{(x + 1)(x^2 + 8x - 3)} + \frac{5x^2 - 1}{(8x^2 + 6x + 1)(7x - 6)} + (9x^3 - 6x^2 + 7x)$$

$$P2 = \frac{(8x^2 + 6x + 1)(7x - 6)}{(x + 1)(9x^3 - 6x^2 + 7x)} + \frac{5x^2 - 1}{(2x^2 + 6x - 1)(x^2 + 8x - 3)} + (x^3 + 4x - 2)$$

$$P3 = \frac{(8x^2 + 6x + 1)(7x - 6)(x + 1)(x^2 + 8x - 3)}{(9x^3 - 6x^2 + 7x)(2x^2 + 6x - 1)(5x^2 - 1)(x^3 + 4x - 2)}$$

2) Thực hiện vẽ đồ thị cho các hàm sau (khoảng vẽ tự chọn)

$$f(x) = 3x^2 + 7x + 4$$

$$g(x) = (x + 1)/(x - 2) + 6x$$

$$h(x) = 3x^3 + 6x^2 + 3x$$

Giảng viên: Hoàng Xuân Dương



3) Hãy cho biết các dòng sau sai ở vị trí nào:

```
>> hamtruyen=inline('8*(s+a)/(s*(s+4)(s+3))')
>> fval(@hamtruyen,3,7)
>> plot(@hamtruyen,[4 8])
>> grid
```

4) Giải thích các dòng sau:

```
>> syms x
>> a=[sin(x);cos(x)]
>> y=diff(a);
>> x=linspace(0,2*pi);
>> plot(x,eval(y(1,1)), 'r',x,eval(y(2,1)), 'b');
>> gtext('cos(x)'); gtext('sin(x)'); title('Bai tap chuong 4');
```

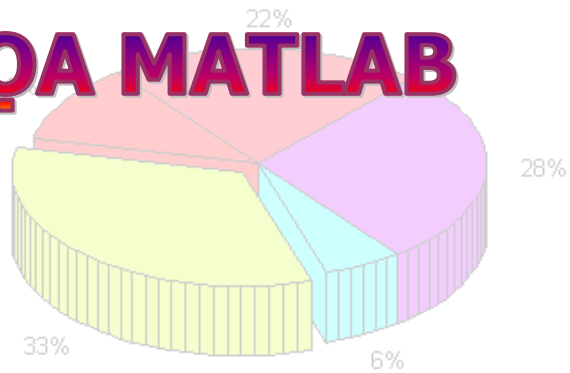
5) Hãy cho biết kết quả các dòng sau:

```
>> A=sym('[x^2-1,x+1,x-1,4]')
>> symadd(A,2)
>> symadd(A,'2')
>> symadd(A(1),A(2))
>> symmul(A(1),A(3))
>> compose(A(1),A(3))
>> compose(compose(A(1),A(2)),A(4))
>> sympow(A(4),'2')
>> symop(A(1),'+',A(2),'*',A(3),'-',A(4))
>> symmul(symmul(A(1),A(4)),symmul(A(2),A(3)))
>> sym2poly(symmul(A(2),A(1)))
>> symmul(A,A)
```



## CHƯƠNG 5:

# ĐỒ HỌA MATLAB



Giảng viên: Hoàng Xuân Dương



## CHƯƠNG 5: ĐỒ HỌA MATLAB

212

- I. ĐỒ HỌA 2D
- II. ĐỒ HỌA 3D
- III. CÁC LOẠI HÀM ĐẶC BIỆT

Giảng viên: Hoàng Xuân Dương



**I. ĐỒ HỌA 2D:**

Các bước cơ bản để sử dụng các hàm vẽ:

**1. Chuẩn bị dữ liệu**

```
x = 0:0.2:12;
y1 = bessell(1,x);
y2 = bessell(2,x);
y3 = bessell(3,x);
```

**2. Chọn cửa sổ và vị trí một vùng vẽ trong cửa sổ**

```
figure(1)
subplot(2,2,1)
```

**3. Gọi các hàm vẽ**

```
h = plot(x,y1,x,y2,x,y3);
```

**4. Chọn nét vẽ và màu sắc**

```
set(h,'LineWidth',2,'LineStyle',{'--';':';'-.'})
set(h,'Color',{'r';'g';'b'})
```

Giảng viên: Hoàng Xuân Dương

**I. ĐỒ HỌA 2D:****5. Cài đặt thông số trục và lưới**

```
axis([0 12 -0.5 1])
grid on
```

**6. Tạo các chú thích và canh lề cho hình vẽ**

```
xlabel('Time')
ylabel('Amplitude')
legend(h,'First','Second','Third')
title('Bessel Functions')
[y,ix] = min(y1);
text(x(ix),y,'First Min \rightarrow',...
     'HorizontalAlignment','right')
```

**7. Xuất hình vẽ**

```
print -depsc -tiff -r200 myplot
```

Giảng viên: Hoàng Xuân Dương

**I. ĐỒ HỌA 2D:**

Các hàm vẽ cơ bản:

| Hàm      | Ý nghĩa                                      |
|----------|--|
| plot     | Vẽ 2D với 2 trục x và y tuyến tính           |
| plot3    | Vẽ 3D với 3 trục x, y và z tuyến tính        |
| loglog   | Vẽ với 2 trục x và y là logarithmic          |
| semilogx | Vẽ với trục x là logarithmic và y tuyến tính |
| semilogy | Vẽ với trục y là logarithmic và x tuyến tính |
| plotyy   | Vẽ có 2 trục y                               |

Giảng viên: Hoàng Xuân Dương

**I. ĐỒ HỌA 2D:****1. Hàm plot:**

Cú pháp hàm plot như sau

`plot(Y)`

`plot(X1,Y1,...)`

`plot(X1,Y1,LineStyle,...)`

`plot(...,'PropertyName',PropertyValue,...)`

`plot(axes_handle,...)`

`h = plot(...)`

`hlines = plot('v6',...)`

Giảng viên: Hoàng Xuân Dương

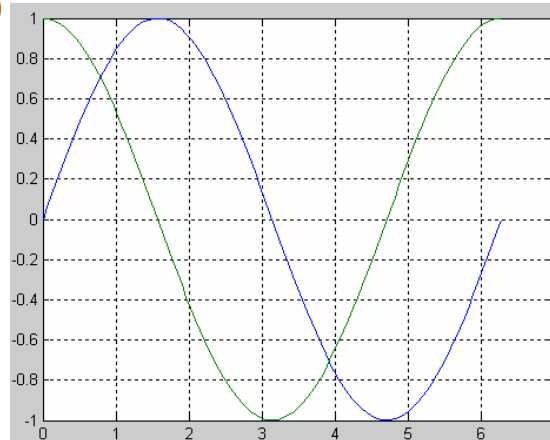


## I. ĐỒ HỌA 2D:

## 1. Hàm plot:

Ví dụ:

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



Giảng viên: Hoàng Xuân Dương

## I. ĐỒ HỌA 2D:

## 1. Hàm plot (tt)

- **Handle:** Mỗi một đối tượng trong màn hình đồ họa đều được nhận diện bằng một con số, được gọi là handle của đối tượng

Một số hàm liên quan đến các handle đặc biệt:

- 0 → handle đối tượng gốc
- gcf → trả về handle cho figure hiện hành
- gca → trả về handle cho trục vẽ hiện hành
- gco → trả về handle cho đối tượng hiện hành
- gcbf → trả về handle cho callback figure
- gcbo → trả về handle cho callback object

Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

### 1. Hàm plot (tt)

#### ➤ Handle (tt)

Ví dụ:

```
>> x=linspace(0,2*pi);  
>> h=plot(x,sin(x),x,cos(x))  
h = 103.0004  
      3.0016  
>>(gcf)  
ans = 1  
>>(gca)  
ans = 101.0009  
>> set(h(1),'color','r')
```



## I. ĐỒ HỌA 2D:

### 1. Hàm plot (tt)

#### ➤ Các loại nét vẽ đặc biệt:

`plot(x,y,'linestyle_marker_color')`

| linestyle | Kiểu đường           |
|-----------|----------------------|
| '-'       | Solid line (default) |
| '--'      | Dashed line          |
| '.'       | Dotted line          |
| '-.'      | Dash-dot line        |
| 'none'    | No line              |





| marker             | Ý nghĩa                       |
|--------------------|-------------------------------|
| '+'                | Plus sign                     |
| 'o'                | Circle                        |
| '*'                | Asterisk                      |
| '.'                | Point                         |
| 'x'                | Cross                         |
| 'square' or 's'    | Square                        |
| 'diamond' or 'd'   | Diamond                       |
| '^'                | Upward-pointing triangle      |
| 'v'                | Downward-pointing triangle    |
| '>'                | Right-pointing triangle       |
| '<'                | Left-pointing triangle        |
| 'pentagram' or 'p' | Five-pointed star (pentagram) |
| 'hexagram' or 'h'  | Six-pointed star (hexagram)   |
| 'none'             | No marker (default)           |

**Giảng viên: Hoàng Xuân Dương**

| Giá trị RGB | Color | Màu     |
|-------------|-------|---------|
| [1 1 0]     | y     | yellow  |
| [1 0 1]     | m     | magenta |
| [0 1 1]     | c     | cyan    |
| [1 0 0]     | r     | red     |
| [0 1 0]     | g     | green   |
| [0 0 1]     | b     | blue    |
| [1 1 1]     | w     | white   |
| [0 0 0]     | k     | black   |

**Giảng viên: Hoàng Xuân Dương**

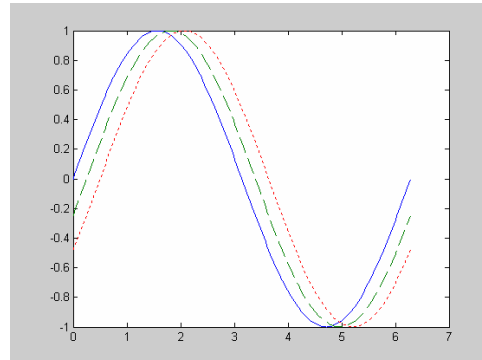
## I. ĐỒ HỌA 2D:

## 1. Hàm plot (tt)

## ➤ Các loại nét vẽ đặc biệt (tt)

Ví dụ

```
>> t=0:pi/100:2*pi;
>> y1=sin(t);y2=sin(t-0.25);y3=sin(t-0.5);
>> plot(t,y1,'-',t,y2,'--',t,y3,':')
```



Giảng viên: Hoàng Xuân Dương

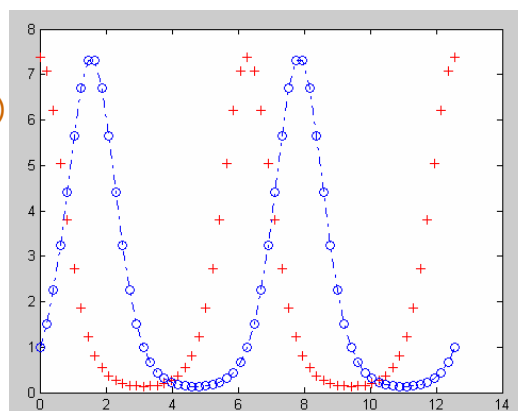
## I. ĐỒ HỌA 2D:

## 1. Hàm plot (tt)

## ➤ Vẽ điểm - đường - các ký hiệu:

Ví dụ 1:

```
>> x = 0:pi/15:4*pi;
>> y1 = exp(2*cos(x));
>> y2 = exp(2*sin(x));
>> plot(x,y1,'+r',x,y2,'-ob')
```



Giảng viên: Hoàng Xuân Dương



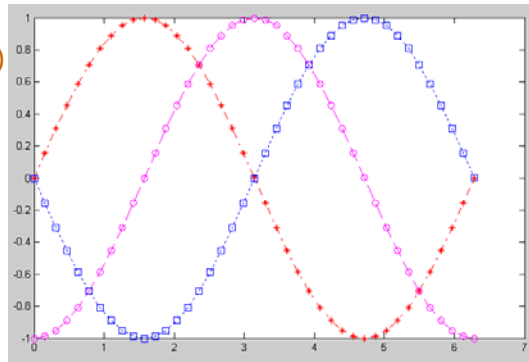
## I. ĐỒ HỌA 2D:

## 1. Hàm plot (tt)

## ➤ Vẽ điểm - đường - các ký hiệu:

Ví dụ 2:

```
>> t = 0:pi/20:2*pi;
>> plot(t,sin(t),'-r*')
>> hold on
>> plot(t,sin(t-pi/2),'--mo')
>> plot(t,sin(t-pi),' :bs')
>> hold off
```



Giảng viên: Hoàng Xuân Dương

## I. ĐỒ HỌA 2D:

## 1. Hàm plot (tt)

## ➤ Các màu và kích thước nét vẽ:

```
plot(x,y,'-mo',...
      'LineWidth',0.5,...
      'MarkerEdgeColor','k',...
      'MarkerFaceColor',[.49 1 .63],...
      'MarkerSize',6)
```

Trong đó:

- **lineWidth** là độ rộng nét vẽ (mặc định là 0.5)
- **markerEdgrcolor** là màu cạnh marker
- **markerfacecolor** là màu tô
- **Markersize** là kích thước nét vẽ (mặc định là 6)

Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

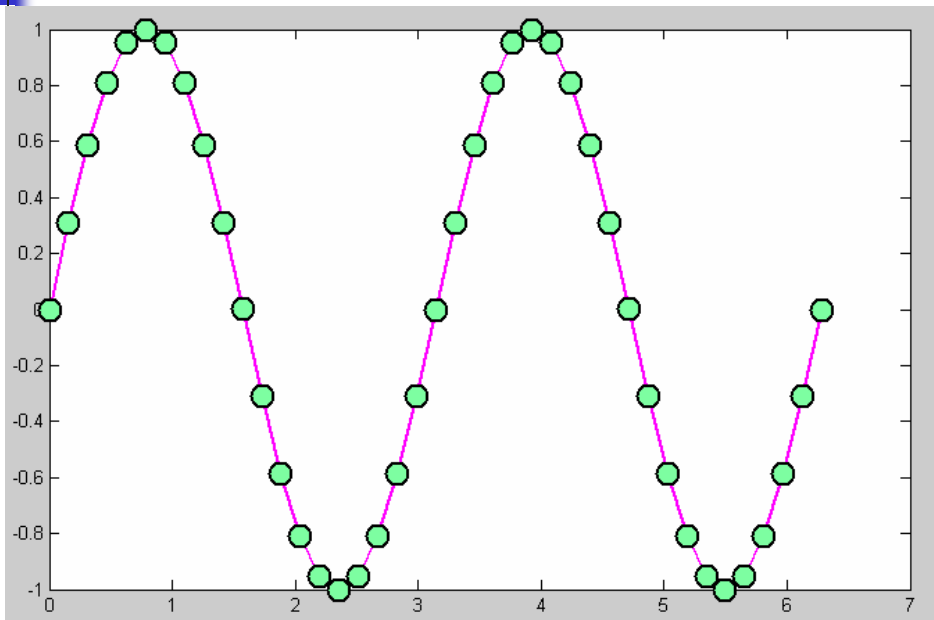
### 1. Hàm plot:

#### ➤ Các màu và kích thước nét vẽ (tt)

Ví dụ 1:

```
>> t = 0:pi/20:2*pi;  
>> plot(t,sin(2*t),'-mo',...    % nét vẽ hình tròn  
'LineWidth',2,...  
'MarkerEdgeColor','k',...      % màu cạnh hình tròn  
'MarkerFaceColor',[.49 1 .63],... % màu tô các hình tròn  
'MarkerSize',12)              % kích thước các hình tròn
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương





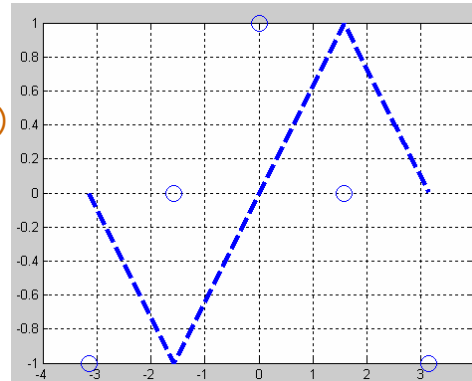
## I. ĐỒ HỌA 2D:

### 1. Hàm plot (tt)

#### ➤ Các màu và kích thước nét vẽ (tt)

Ví dụ 2:

```
>> x=(-pi:pi/2:pi);  
>> y1=sin(x); y2=cos(x);  
>> plot(x,y1,'--','linewidth',4)  
>> hold on  
>> plot(x,y2,'o','markersize',12)  
>> grid on
```



Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

### 2. Các hàm gán nhãn

| Hàm    | Ý nghĩa                        |
|--------|--------------------------------|
| text   | Hiển thị 1 chuỗi               |
| title  | Tựa đề                         |
| xlabel | Đặt nhãn cho trục x            |
| ylabel | Đặt nhãn cho trục y            |
| zlabel | Đặt nhãn cho trục z            |
| legend | Chú thích trên hình            |
| gtext  | Hiển thị 1 chuỗi sử dụng chuột |

Giảng viên: Hoàng Xuân Dương





## I. ĐỒ HỌA 2D:

### 2. Các hàm gán nhãn (tt)

#### a. text

Cú pháp:

```
text(x,y,'string')  
text(x,y,z,'string')  
text(...'PropertyName',PropertyValue...)  
h = text(...)
```

Trong đó:

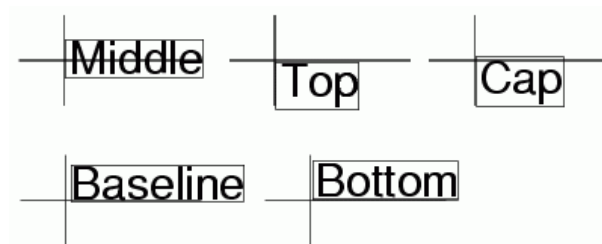
(x,y) → tọa độ vị trí string  
PropertyName → tên thuộc tính  
PropertyValue → giá trị thuộc tính



## ❖ Các thuộc tính của text:

### ➤ verticalalignment {top | cap | middle | baseline | bottom}

Thuộc tính canh lề theo chiều đứng cho text theo vị trí đặt text



Ví dụ:

```
>> text(x,y,'string','Verticalalignment','top',...
```





❖ **Các thuộc tính của text:**

➤ **horizontalalignment {left | center | right}**

Thuộc tính canh lề theo hàng ngang cho text theo vị trí đặt text



Ví dụ:

```
>> text(x,y,'string','HorizontalAlignment','right',...
```



❖ **Các thuộc tính của text:**

➤ **rotation scalar(degrees)**

Xoay text đi một góc, mặc định là 0

➤ **Fontname**

Kiểu font chữ của text (mặc định Helvetica). Có thể kết hợp với các option để định dạng:

\bf - bold font

\it - italics font

\sl - oblique font (rarely available)

\rm - normal font

Ví dụ:

```
>> text(11,380,'\itConcentration','Rotation',-55,...  
'FontName','Tahoma')
```





❖ **Các thuộc tính của text:**

➤ **FontSize**

Kích thước font chữ (mặc định là 10)

Ví dụ:

```
>> text(11,380,'Concentration','Rotation',-55, 'fontsize',12)
```

➤ **string**

Chuỗi văn bản cần được hiển thị. Có thể sử dụng các ký hiệu trong bảng sau để tạo các ký tự đặc biệt:

**Giảng viên: Hoàng Xuân Dương**



| Character Sequence | Symbol      | Character Sequence | Symbol     | Character Sequence | Symbol            |
|--------------------|-------------|--------------------|------------|--------------------|-------------------|
| \alpha             | $\alpha$    | \upsilon           | $\upsilon$ | \sim               | $\sim$            |
| \beta              | $\beta$     | \phi               | $\phi$     | \leq               | $\leq$            |
| \gamma             | $\gamma$    | \chi               | $\chi$     | \infty             | $\infty$          |
| \delta             | $\delta$    | \psi               | $\psi$     | \clubsuit          | $\clubsuit$       |
| \epsilon           | $\epsilon$  | \omega             | $\omega$   | \diamondsuit       | $\diamondsuit$    |
| \zeta              | $\zeta$     | \Gamma             | $\Gamma$   | \heartsuit         | $\heartsuit$      |
| \eta               | $\eta$      | \Delta             | $\Delta$   | \spadesuit         | $\spadesuit$      |
| \theta             | $\theta$    | \Theta             | $\Theta$   | \leftrightarrow    | $\leftrightarrow$ |
| \vartheta          | $\vartheta$ | \Lambda            | $\Lambda$  | \leftarrow         | $\leftarrow$      |
| \iota              | $\iota$     | \Xi                | $\Xi$      | \uparrow           | $\uparrow$        |
| \kappa             | $\kappa$    | \Pi                | $\Pi$      | \rightarrow        | $\rightarrow$     |
| \lambda            | $\lambda$   | \Sigma             | $\Sigma$   | \downarrow         | $\downarrow$      |
| \mu                | $\mu$       | \Upsilon           | $\Upsilon$ | \circ              | $\circ$           |
| \nu                | $\nu$       | \Phi               | $\Phi$     | \pm                | $\pm$             |
| \xi                | $\xi$       | \Psi               | $\Psi$     | \geq               | $\geq$            |
| \pi                | $\pi$       | \Omega             | $\Omega$   | \propto            | $\propto$         |

**Giảng viên: Hoàng Xuân Dương**





|                        |             |                        |             |                         |              |
|------------------------|-------------|------------------------|-------------|-------------------------|--------------|
| <code>\rho</code>      | $\rho$      | <code>\forall</code>   | $\forall$   | <code>\partial</code>   | $\partial$   |
| <code>\sigma</code>    | $\sigma$    | <code>\exists</code>   | $\exists$   | <code>*</code>          | $*$          |
| <code>\varsigma</code> | $\varsigma$ | <code>\ni</code>       | $\ni$       | <code>\div</code>       | $\div$       |
| <code>\tau</code>      | $\tau$      | <code>\cong</code>     | $\cong$     | <code>\neq</code>       | $\neq$       |
| <code>\equiv</code>    | $\equiv$    | <code>\approx</code>   | $\approx$   | <code>\aleph</code>     | $\aleph$     |
| <code>\Im</code>       | $\Im$       | <code>\Re</code>       | $\Re$       | <code>\wp</code>        | $\wp$        |
| <code>\otimes</code>   | $\otimes$   | <code>\oplus</code>    | $\oplus$    | <code>\oslash</code>    | $\oslash$    |
| <code>\cap</code>      | $\cap$      | <code>\cup</code>      | $\cup$      | <code>\supseteq</code>  | $\supseteq$  |
| <code>\supset</code>   | $\supset$   | <code>\subseteq</code> | $\subseteq$ | <code>\subset</code>    | $\subset$    |
| <code>\int</code>      | $\int$      | <code>\in</code>       | $\in$       | <code>\circ</code>      | $\circ$      |
| <code>\rfloor</code>   | $\rfloor$   | <code>\lceil</code>    | $\lceil$    | <code>\nabla</code>     | $\nabla$     |
| <code>\lfloor</code>   | $\lfloor$   | <code>\cdot</code>     | $\cdot$     | <code>\ldots</code>     | $\ldots$     |
| <code>\perp</code>     | $\perp$     | <code>\neg</code>      | $\neg$      | <code>\prime</code>     | $\prime$     |
| <code>\wedge</code>    | $\wedge$    | <code>\times</code>    | $\times$    | <code>\emptyset</code>  | $\emptyset$  |
| <code>\rceil</code>    | $\rceil$    | <code>\surd</code>     | $\surd$     | <code>\mid</code>       | $\mid$       |
| <code>\vee</code>      | $\vee$      | <code>\varpi</code>    | $\varpi$    | <code>\copyright</code> | $\copyright$ |
| <code>\angle</code>    | $\angle$    | <code>\angle</code>    | $\angle$    |                         |              |

**Giảng viên: Hoàng Xuân Dương**

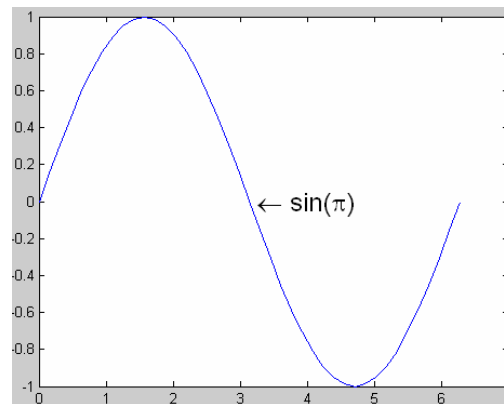


❖ **Các thuộc tính của text:**

➤ **string**

Ví dụ:

```
>> plot(0:pi/20:2*pi,sin(0:pi/20:2*pi))
>> text(pi,0,' \leftarrow sin(\pi)',FontSize,18)
```



**Giảng viên: Hoàng Xuân Dương**





## I. ĐỒ HỌA 2D:

### 2. Các hàm gán nhãn (tt)

#### b. **title:**

Cú pháp:

```
title('string')
```

```
title(fname)
```

```
title(...,'PropertyName',PropertyValue,...)
```

```
h = title(...)
```

Ví dụ:

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

```
>> plot(x,sin(x))
```

```
>> title('Do thi sin(x)', 'FontName',...
```

```
'SVNHelvetica','FontSize',15,'color','r')
```

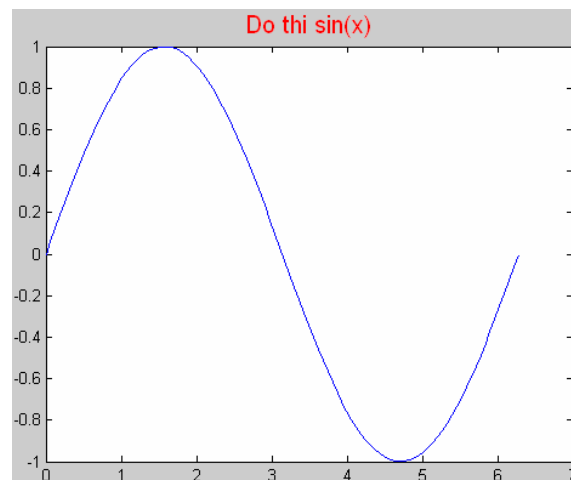
Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

### 2. Các hàm gán nhãn (tt)

#### b. **title (tt)**



Giảng viên: Hoàng Xuân Dương







**I. ĐỒ HỌA 2D:**

**2. Các hàm gán nhãn (tt)**

**c. xlabel-ylabel-zlabel**

Cú pháp:

```
xlabel('string')
xlabel(fname)
xlabel(...,'PropertyName',PropertyValue,...)
h = xlabel(...)

ylabel(...)
h = ylabel(...)

zlabel(...)
h = zlabel(...)
```



**I. ĐỒ HỌA 2D:**

**2. Các hàm gán nhãn (tt)**

**c. xlabel-ylabel-zlabel (tt)**

Ví dụ:

```
>> xlabel('Trục x','Fontname','SVNhelvetica',...
'FontSize',15,'color','g')
>> ylabel('Đồ thị y và z','Fontname','SVNhelvetica',...
'FontSize',15,'color','b')
>> title('Đồ họa 2D','Fontname','SVNhelvetica',...
'FontSize',15,'color','r')
```





## I. ĐỒ HỌA 2D:

### 2. Các hàm gán nhãn (tt)

**d. legend:** Đặt chú thích cho hình vẽ

Cú pháp:

```
legend('string1','string2',...)  
legend(h,'string1','string2',...)  
legend(string_matrix)  
legend(h,string_matrix)  
legend(axes_handle,...)  
legend('off')  
legend(h,...)  
legend(...,pos)  
h = legend(...)
```

Giảng viên: Hoàng Xuân Dương



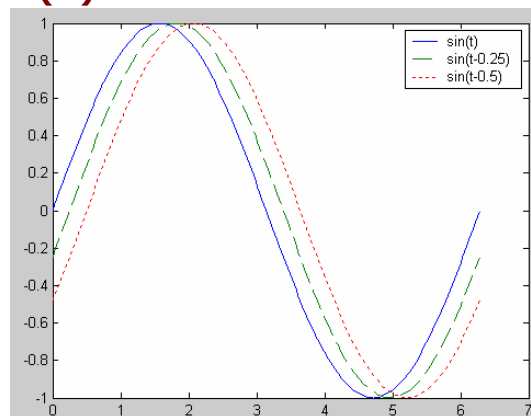
## I. ĐỒ HỌA 2D:

### 2. Các hàm gán nhãn (tt)

**d. legend (tt)**

Ví dụ:

```
>> t=0:pi/100:2*pi;  
>> y1=sin(t);y2=sin(t-0.25);y3=sin(t-0.5);  
>> plot(t,y1,'-',t,y2,'--',t,y3,':')  
>> legend('sin(t)','sin(t-0.25)','sin(t-0.5)',1)
```



Giảng viên: Hoàng Xuân Dương



**I. ĐỒ HỌA 2D:****2. Các hàm gán nhãn (tt)****e. gtext**

Đặt text theo vị trí click chuột trên màn hình đồ họa, trong không gian 2 chiều

Cú pháp:

```
gtext('string')
```

```
h = gtext('string')
```

---

Giảng viên: Hoàng Xuân Dương

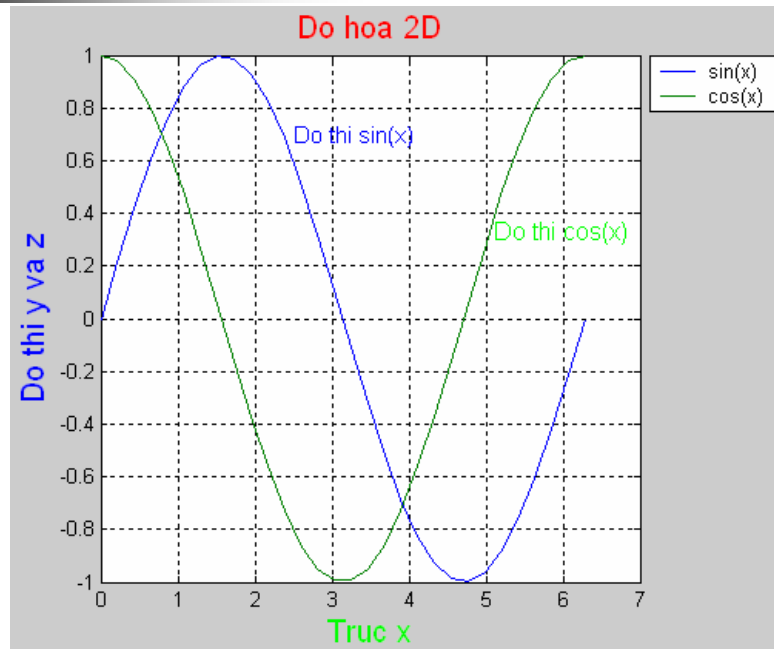
**I. ĐỒ HỌA 2D:****2. Các hàm gán nhãn (tt)**

Ví dụ:

```
>> x=linspace(0,2*pi,30);  
>> y=sin(x);z=cos(x);  
>> plot(x,y,x,z)  
>> grid  
>> xlabel('Trục x','Fontname','SVNhelvetica','FontSize',15,'color','g')  
>> ylabel('Do thi y va z','Fontname','SVNhelvetica','FontSize',15,'color','b')  
>> title('Do hoa 2D','Fontname','SVNhelvetica','FontSize',15,'color','r')  
>> text(2.5,0.7,'Do thi sin(x)',...  
'FontName','SVNhelvetica','FontSize',11,'color','b')  
>> gtext('Do thi cos(x)','FontName','SVNhelvetica','FontSize',12,'color','g')  
>> legend('sin(x)','cos(x)',-1) % ghi chú về hình vẽ
```

---

Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

### 3. Các hàm cài đặt:

#### ➤ Hàm subplot(m,n,p):

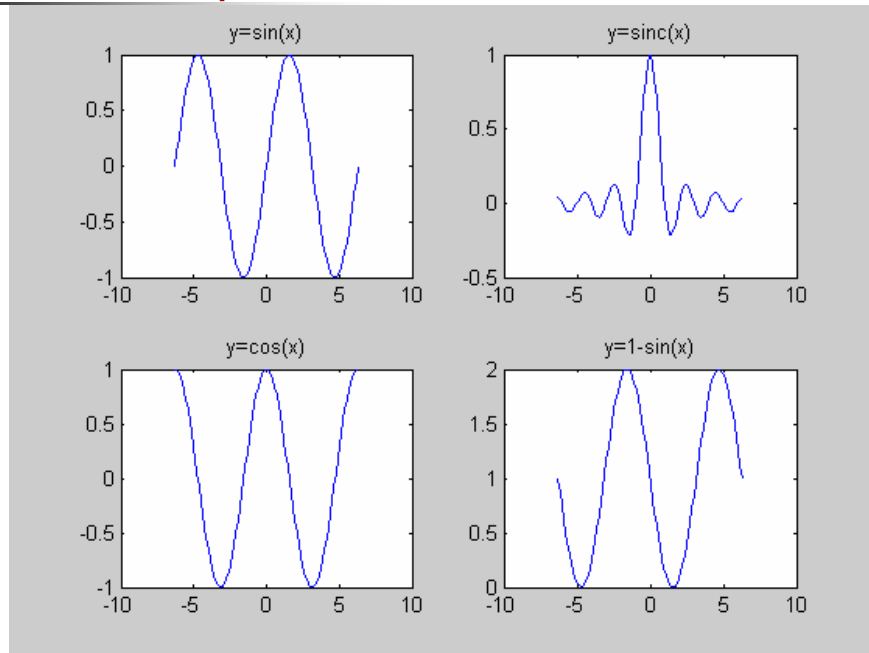
Chia màn hình ra làm m hàng và n cột để vẽ hình, với:

- m: là hàng
- n: là cột
- p: là vị trí cần vẽ

```
>> x=linspace(-2*pi,2*pi);
>> subplot(2,2,1); plot(x,sin(x)); title('y=sin(x)')
>> subplot(2,2,2); plot(x,sinc(x)); title('y=sinc(x)')
>> subplot(2,2,3); plot(x,cos(x)); title('y=cos(x)')
>> subplot(2,2,4); plot(x,1-sin(x)); title('y=1-sin(x)')
>> subplot          % trả về mặc định
```

Giảng viên: Hoàng Xuân Dương





Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

### 3. Các hàm cài đặt (tt)

➤ **`set(h,'LineWidth',2,{'LineStyle'},{'--';':';'-.'})`**

Thay đổi nét vẽ, với:

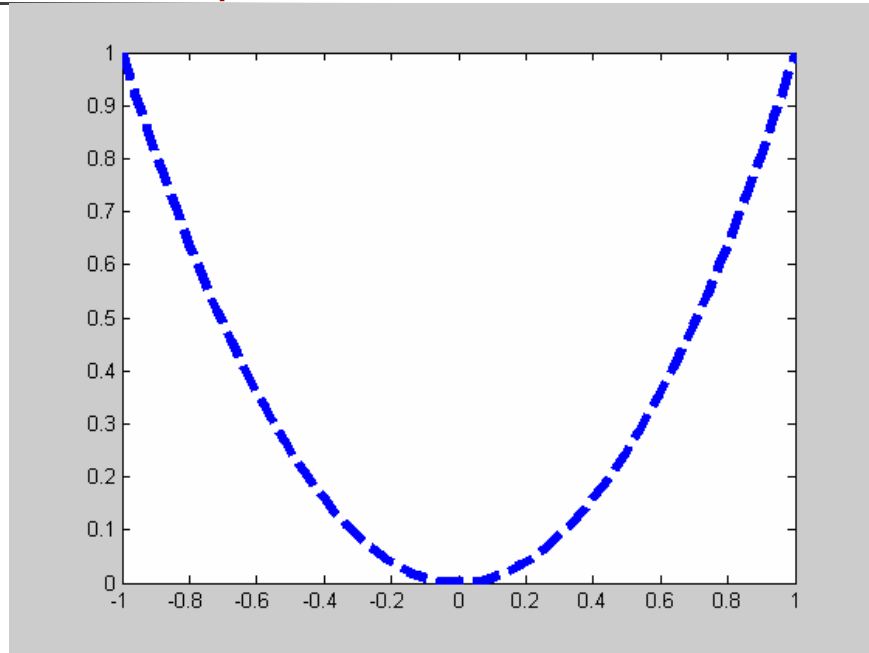
- **LineWidth**: Độ lớn nét vẽ
- **LineStyle**: Loại viết vẽ liên tục hay gián đoạn

Ví dụ:

```
>> x=linspace(-1,1);  
>> h=plot(x,x.*x)  
>> set(h,'LineWidth',5,'LineStyle','--')
```

Giảng viên: Hoàng Xuân Dương





Giảng viên: Hoàng Xuân Dương

## I. ĐỒ HỌA 2D:

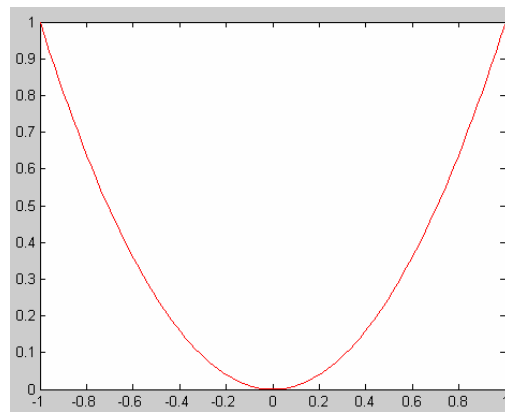
### 3. Các hàm cài đặt (tt)

#### ➤ Hàm `set(h,{'Color'},{'r';'g';'b'})`

Thay đổi màu vẽ (với các ký tự màu đã được đề cập trong chương 1)

Ví dụ:

```
>> x=linspace(-1,1);  
>> h=plot(x,x.*x)  
>> set(h,'Color','r')
```



Giảng viên: Hoàng Xuân Dương

## I. ĐỒ HỌA 2D:

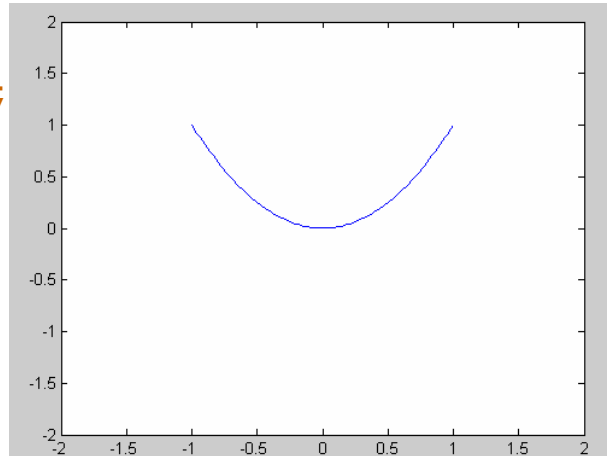
## 3. Các hàm cài đặt (tt)

➤ Hàm `axis([xmin xmax ymin ymax])`

Đặt lại trục vẽ

Ví dụ:

```
>> x=linspace(-1,1);
>> h=plot(x,x.*x)
>> axis([-2 2 -2 2])
```



Giảng viên: Hoàng Xuân Dương

## I. ĐỒ HỌA 2D:

## 3. Các hàm cài đặt (tt)

Ví dụ:

```
>> x=0:0.2:12;
>> y1=bessel(1,x);
>> y2=bessel(2,x);
>> y3=bessel(3,x);
>> figure(1)
>> h=plot(x,y1,x,y2,x,y3);
>> set(h,'LineWidth',2,'LineStyle',{'--';':';'-.'})
>> set(h,'Color',{'r';'g';'b'})
>> axis([0 12 -0.5 1])
>> grid on
```

Giảng viên: Hoàng Xuân Dương

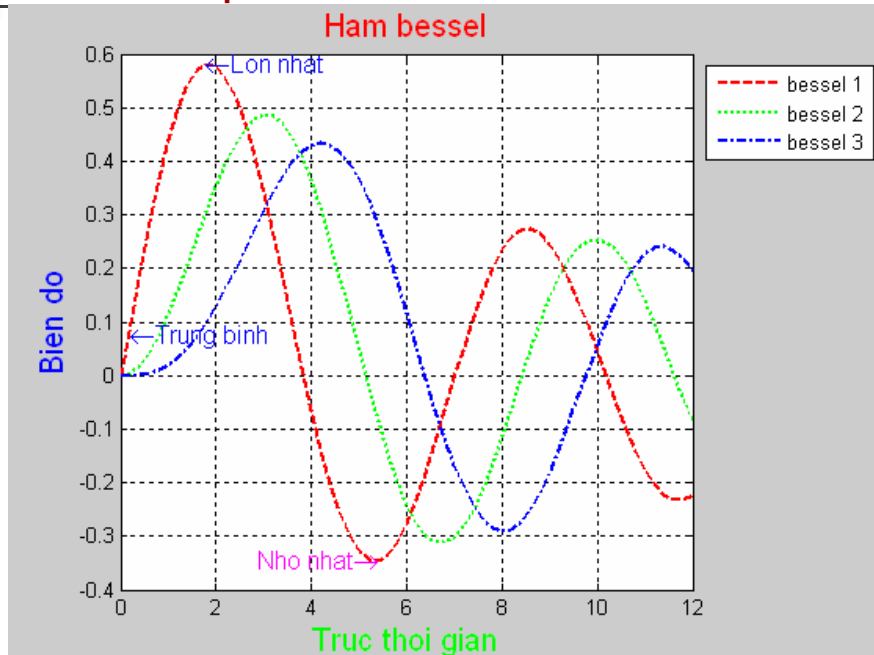
*% Đặt nhãn cho trục:*

```
>> xlabel('Trục thời gian','FontName','SVNhelvetica','FontSize',15,'color','g')
>> ylabel('Biên độ','FontName','SVNhelvetica','FontSize',15,'color','b')
>> title('Hàm Bessel','FontName','SVNhelvetica','FontSize',15,'color','r')
>> legend(h,'bessel 1','bessel 2','bessel 3',-1)
```

*% Chú thích hình:*

```
>> [y,ix]=min(y1);           % tìm min trong mảng (trả về giá trị, chỉ số)
>> text(x(ix),y,'Nhỏ nhất\rightarrow','HorizontalAlignment','right',...
'FontName','SVNhelvetica','FontSize',12,'color','m')
>> [yy,ixx]=max(y1);         % tìm max trong mảng (trả về giá trị, chỉ số)
>> text(x(ixx),yy,'\leftarrow Lon nhất','HorizontalAlignment','left',...
'FontName','SVNhelvetica','FontSize',12,'color','b')
>> ymean=mean(y1);           % tìm trung bình trong mảng (trả về giá trị)
>> text(.2,ymean,'\leftarrow Trung bình','HorizontalAlignment','left',...
'FontName','SVNhelvetica','FontSize',12,'color','b')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương





## I. ĐỒ HỌA 2D:

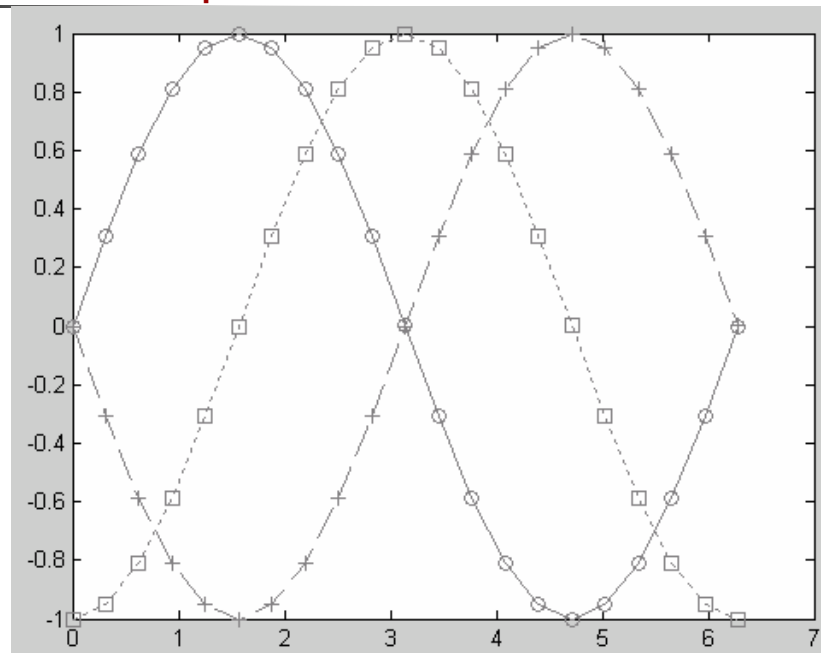
### 3. Các hàm cài đặt (tt)

#### ➤ Đặt loại nét vẽ mặc định :

Ví dụ:

```
>> x = 0:pi/10:2*pi;
>> y1 = sin(x); y2 = sin(x-pi/2); y3 = sin(x-pi);
% Đặt 3 loại đường vẽ
>> set(0,'DefaultAxesLineStyleOrder',{'-o','s','--+'})
% Đặt màu vẽ là xám
>> set(0,'DefaultAxesColorOrder',[0.5,0.5,0.5])
% Vẽ 3 hình
>> plot(x,y1,x,y2,x,y3)
% Trả về mặc định
>> set(0,'DefaultAxesLineStyleOrder','remove')
>> set(0,'DefaultAxesColorOrder','remove')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương





## I. ĐỒ HỌA 2D:

### 3. Các hàm cài đặt (tt)

#### ➤ Đặt trục vẽ:

Cú pháp:

`semilogx(Y)`

`semilogx(X1,Y1,...)`

`semilogx(X1,Y1,LineSpec,...)`

`semilogx(...,'PropertyName',PropertyValue,...)`

`h = semilogx(...)`

`hlines = semilogx('v6',...)`

`semilogy(...)`

`h = semilogy(...)`

`hlines = semilogy('v6',...)`

Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

### 3. Các hàm cài đặt (tt)

#### ➤ Đặt trục vẽ:

Ví dụ:

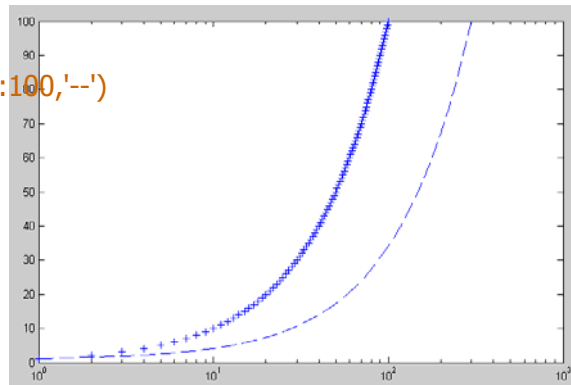
`>> x=0:.1:10;`

`>> semilogx(1:100,'+')`

`>> hold on`

`>> semilogx(1:3:300,1:100,'--')`

`>> hold off`



Giảng viên: Hoàng Xuân Dương



**I. ĐỒ HỌA 2D:****4. Hàm plotyy:**

Cú pháp hàm plotyy như sau:

```
plotyy(X1,Y1,X2,Y2)
```

```
plotyy(X1,Y1,X2,Y2,'function')
```

```
plotyy(X1,Y1,X2,Y2,'function1','function2')
```

[AX,H1,H2] = plotyy(...) sẽ trả về:

AX = handle của trục

H1 = handle của hình 1

H2 = handle của hình 2

'function' có thể là `plot`, `semilogx`, `semilogy`, `loglog`, `stem`, hay bất kỳ hàm Matlab theo cú pháp:

```
h = function(x,y)
```

---

Giảng viên: Hoàng Xuân Dương

**I. ĐỒ HỌA 2D:****4. Hàm plotyy (tt)**

Ví dụ 1:

```
>> t = 0:pi/20:2*pi;
```

```
>> y1=sin(t);
```

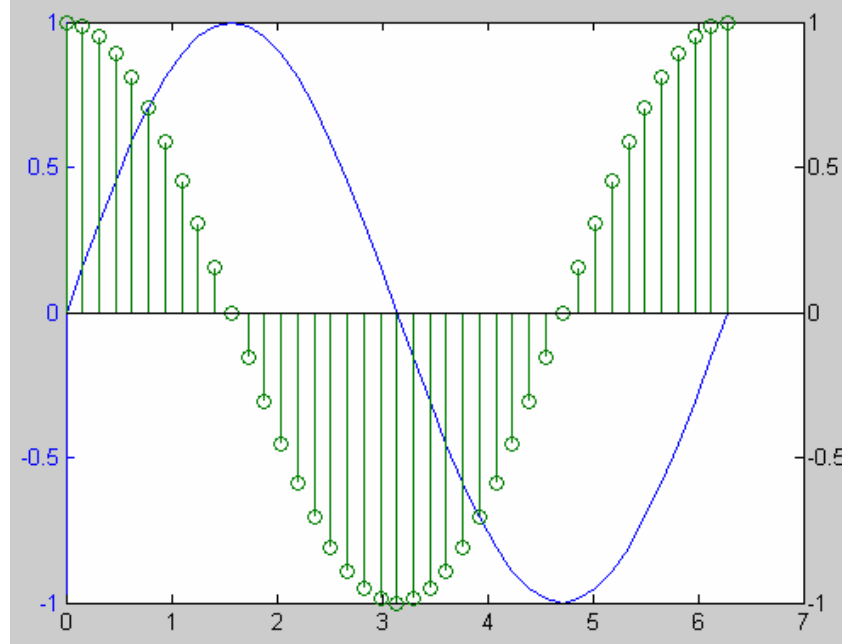
```
>> y2=cos(t);
```

```
>> plotyy(t,y1,t,y2,'plot','stem')
```

*% stem là hàm vẽ dữ liệu rời rạc.*

---

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

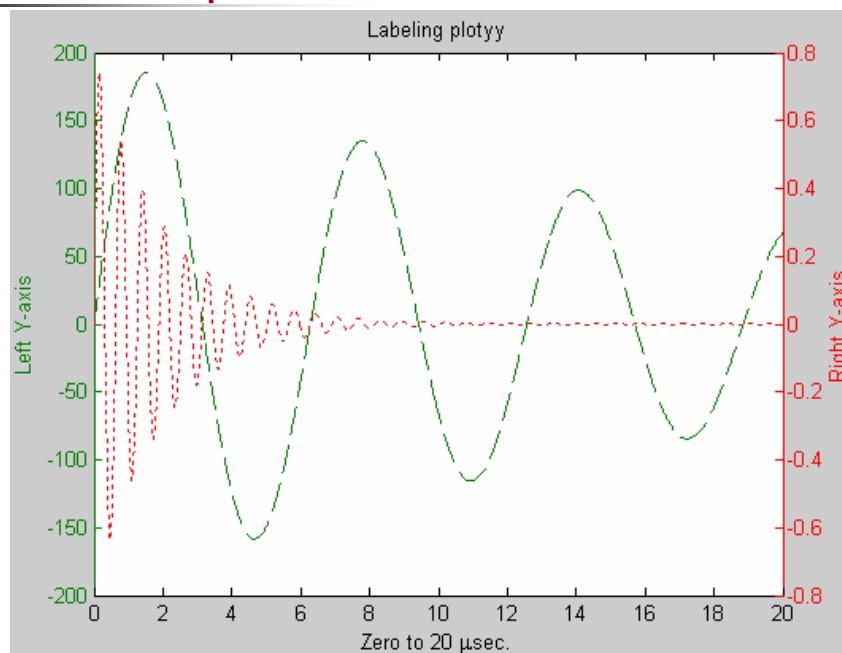
### 4. Hàm plotyy (tt)

Ví dụ 2:

```
>> x = 0:0.01:20;
>> y1 = 200*exp(-0.05*x).*sin(x);
>> y2 = 0.8*exp(-0.5*x).*sin(10*x);
>> [AX,H1,H2] = plotyy(x,y1,x,y2,'plot');
>> set(get(AX(1),'Ylabel'),'String','Left Y-axis')
>> set(get(AX(2),'Ylabel'),'String','Right Y-axis')
>> title('Labeling plotyy')
>> set(H1,'LineStyle','--')
>> xlabel('Zero to 20 \musec.')
>> set(H2,'LineStyle',':')
```

Giảng viên: Hoàng Xuân Dương





Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

### 5. Hàm plot3:

Cú pháp:

`plot3(X1,Y1,Z1,...)`

`plot3(X1,Y1,Z1,LineSpec,...)`

`plot3(...,'PropertyName',PropertyValue,...)`

`h = plot3(...)`

Giảng viên: Hoàng Xuân Dương

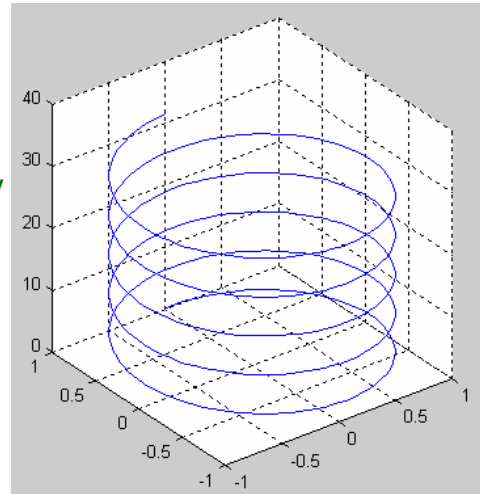


## I. ĐỒ HỌA 2D:

## 5. Hàm plot3 (tt)

Ví dụ 1:

```
>> t = 0:pi/50:10*pi;
>> plot3(sin(t),cos(t),t)
>> grid on
>> axis square
% chọn 3 trục x,y,z bằng nhau
```



Giảng viên: Hoàng Xuân Dương

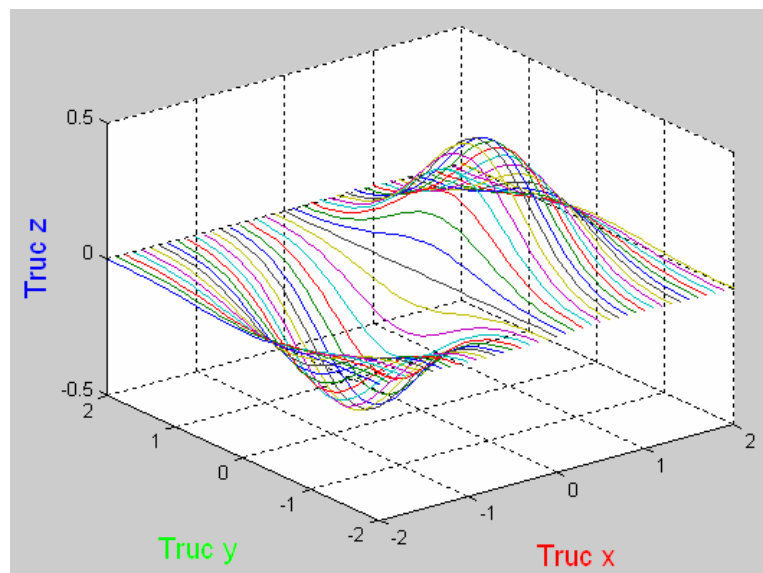
## I. ĐỒ HỌA 2D:

## 5. Hàm plot3 (tt)

Ví dụ 2: Dữ liệu là ma trận

```
>> [X,Y]=meshgrid([-2:0.1:2]); % Tạo tập ma trận dữ liệu
>> Z=X.*exp(-X.^2-Y.^2);
>> plot3(X,Y,Z)
>> grid on
>> xlabel('Trục x','FontName','SVNhelvetica','FontSize',15,'color','r')
>> ylabel('Trục y','FontName','SVNhelvetica','FontSize',15,'color','g')
>> zlabel('Trục z','FontName','SVNhelvetica','FontSize',15,'color','b')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

### 6. Đặt thông số cho trục:

➤ **axis:** Đặt giá trị trục

`axis([xmin xmax ymin ymax])`

`axis([xmin xmax ymin ymax zmin zmax])`

`v = axis`

`axis auto`

`axis manual`

`axis tight`

`axis fill`

....

Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

## 6. Đặt thông số cho trục (tt)

....

axis equal      *% tỉ lệ các trục bằng nhau*  
axis square    *% độ dài các trục bằng nhau*  
axis normal  
axis off  
axis on

## I. ĐỒ HỌA 2D:

## 6. Đặt thông số cho trục (tt)

Ví dụ 1:

```
>> x = -pi:.1:pi;  
>> y = sin(x);  
>> plot(x,y,'r')  
>> set(gca,'XTick',-pi:pi/2:pi)      %định các điểm trên trục x  
>> set(gca,'XTickLabel',{'-pi','-pi/2','0','pi/2','pi'})  
>> xlabel('-\pi \leq \Theta \leq \pi','FontSize',15,'color','b')  
>> ylabel('sin(\Theta)','FontSize',15,'color','g')
```





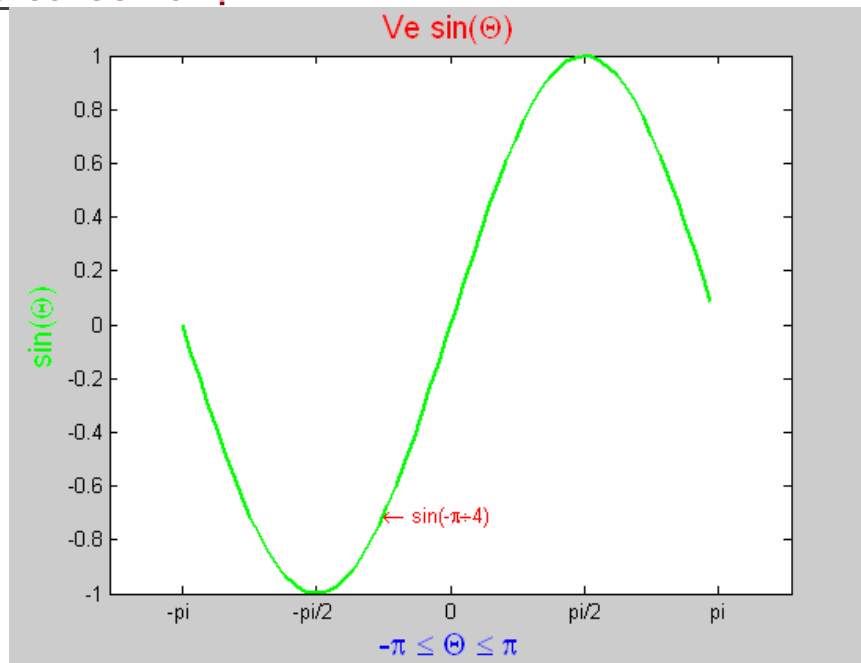
## I. ĐỒ HỌA 2D:

### 6. Đặt thông số cho trục (tt)

Ví dụ 1 (tt)

```
>> title('Ve sin(\Theta)','Fontname','SVNhelvetica',...  
        'FontSize',15,'color','r')  
>> text(-pi/4,sin(-pi/4),'\leftarrow sin(-\pi\div4)',...  
        'HorizontalAlignment','left','color','r')  
>> hold on  
>> set(findobj(gca,'Type','line','Color',[1 0 0]),...  
        'Color',[0,0,1],'LineWidth',2)  
>> hold off  
>> set(findobj(gca,'Type','line','Color',[0 0 1]),...  
        'Color',[0 1 0],'LineWidth',2)
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương





## I. ĐỒ HỌA 2D:

### 6. Đặt thông số cho trục (tt)

Ví dụ 2: Cài đặt tỉ lệ

```
>> t = 0:pi/20:2*pi;  
>> plot(sin(t),2*cos(t))  
>> grid on  
>> axis square
```

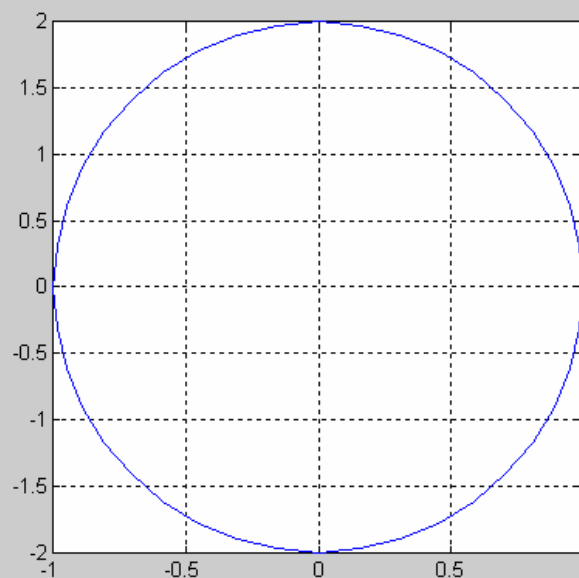
hoặc:

```
>> axis equal
```

hoặc:

```
>> axis equal tight
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương



## I. ĐỒ HỌA 2D:

## 7. Vẽ nhiều hình:

Ví dụ:

```
>> h1=figure % Tạo khung hình 1
>> x=linspace(0,2*pi);
>> plot(x,sin(x));
>> axis([0 2*pi -1 1]);title('sin(x)');
>> h2=figure % Tạo khung hình 2
>> plot(x,cos(x));
>> axis([0 2*pi -1 1]);title('cos(x)');
>> h3=figure
>> plot(x,2.*sin(x).*cos(x));
>> axis([0 2*pi -1 1]);title('2*sin(x)*cos(x)');
```

Giảng viên: Hoàng Xuân Dương

## I. ĐỒ HỌA 2D:

## 7. Vẽ nhiều hình (tt)

Ví dụ (tt)

```
>> h4=figure
>> plot(x,sin(x)./cos(x));
>> axis([0 2*pi -1 1]);title('sin(x)/cos(x)');
>> close % Xóa hình bất kỳ
>> close(h1) % Xóa hình 1
>> clf % Xóa hình không xóa khung
>> close all % Xóa tất cả
>> clf reset
```

Giảng viên: Hoàng Xuân Dương



## II. ĐỒ HỌA 3D:

### 1. Cách sử dụng các hàm vẽ:

1. Chuẩn bị dữ liệu  
`z=peaks(20)`
2. Chọn vị trí trong cửa sổ để vẽ  
`figure(1)`  
`subplot(2,1,2)`
3. Gọi hàm vẽ 3D  
`h = surf(z);`
4. Chọn màu và tô bóng  
`colormap hot`  
`shading interp`  
`set(h,'EdgeColor','k')`

Giảng viên: Hoàng Xuân Dương



5. Thêm lighting  
`light('Position',[-2,2,20])`  
`lighting phong`  
`material([0.4,0.6,0.5,30])`  
`set(h,'FaceColor',[0.7 0.7 0],...'BackFaceLighting','lit')`
6. Chọn view  
`view([30,25])`  
`set(gca,'CameraViewAngleMode','Manual')`
7. Chọn trục  
`axis([5 15 5 15 -8 8])`  
`set(gca,'ZTickLabel','Negative||Positive')`

Giảng viên: Hoàng Xuân Dương





8. Chọn tỉ lệ

```
set(gca,'PlotBoxAspectRatio',[2.5 2.5 1])
```

9. Tạo các nhãn

```
xlabel('X Axis')
```

```
ylabel('Y Axis')
```

```
zlabel('Function Value')
```

```
title('Peaks')
```

10. In

```
set(gcf,'PaperPositionMode','auto')
```

```
print -dps2
```



II. ĐỒ HỌA 3D:

2. Vẽ dữ liệu 3D:

Nếu x, y và z là 3 vector có cùng kích thước:

```
plot3(x,y,z)
```

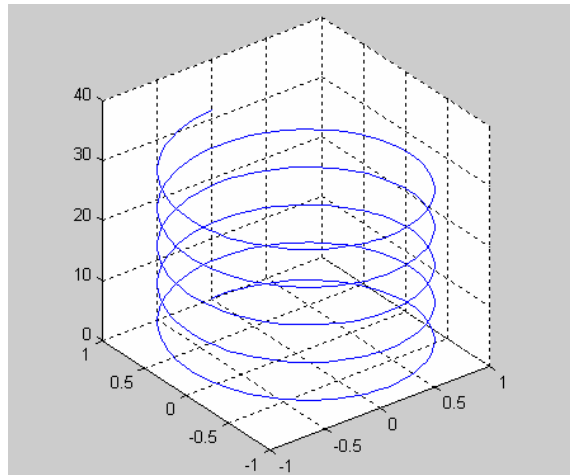
Ví dụ 1:

```
>> t = 0:pi/50:10*pi;
```

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

```
>> axis square;
```

```
>> grid on
```

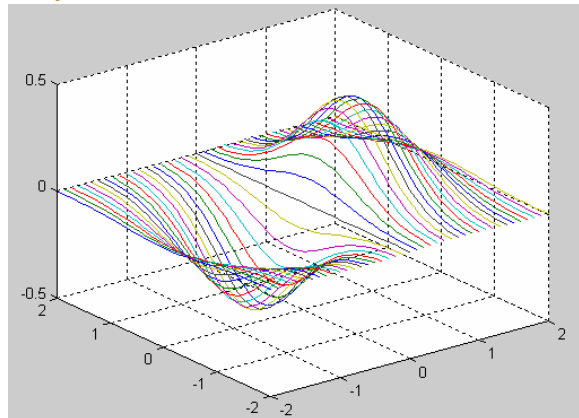


## II. ĐỒ HỌA 3D:

## 2. Vẽ dữ liệu 3D (tt)

Ví dụ 2:

```
>> [X,Y] = meshgrid([-2:0.1:2]);  
>> Z = X.*exp(-X.^2-Y.^2);  
>> plot3(X,Y,Z)  
>> grid on
```



Giảng viên: Hoàng Xuân Dương

## II. ĐỒ HỌA 3D:

## 3. Các loại hàm vẽ bề mặt :

## a. Hàm mesh:

```
mesh(X,Y,Z)  
mesh(Z)  
mesh(...,C)  
mesh(...,'PropertyName',PropertyValue,...)  
mesh(axes_handles,...)  
meshc(...)  
meshz(...)
```

Giảng viên: Hoàng Xuân Dương

## II. ĐỒ HỌA 3D:

## 3. Các loại hàm vẽ bề mặt (tt)

## a. Hàm mesh (tt)

Ví dụ:

```
>> [X,Y] = meshgrid(-3:.125:3);
```

```
>> Z = peaks(X,Y);
```

```
>> meshc(X,Y,Z);
```

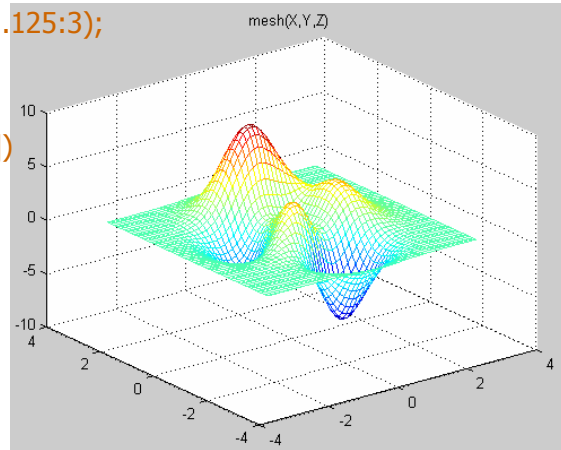
```
>> axis([-3 3 -3 3 -10 5])
```

hoặc

```
>> meshz(X,Y,Z);
```

hoặc

```
>> mesh(X,Y,Z);
```



Giảng viên: Hoàng Xuân Dương

## II. ĐỒ HỌA 3D:

## 3. Các loại hàm vẽ bề mặt (tt)

## b. Hàm surf:

```
surf(Z)
```

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

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

```
surf(...,'PropertyName',PropertyValue)
```

```
surf(axes_handle,...)
```

```
surfc(...)
```

Giảng viên: Hoàng Xuân Dương

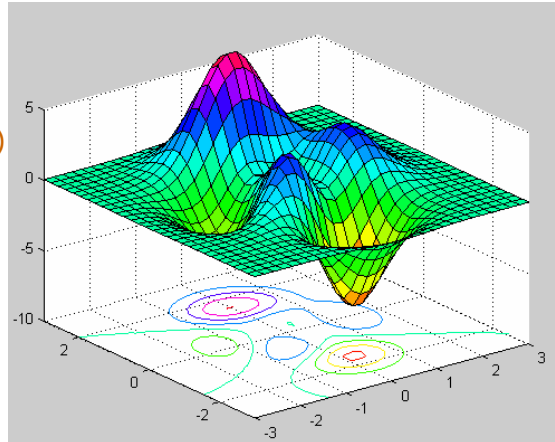
## II. ĐỒ HỌA 3D:

## 3. Các loại hàm vẽ bề mặt (tt)

## b. Hàm surf (tt)

Ví dụ:

```
>> [X,Y,Z] = peaks(30);
>> surf(X,Y,Z)
>> colormap hsv
>> axis([-3 3 -3 3 -10 5])
```



Giảng viên: Hoàng Xuân Dương

## II. ĐỒ HỌA 3D:

## 4. Sử dụng màu:

Cú pháp:

```
colormap(map)
colormap('default')
cmap = colormap
```

Một colormap là một ma trận m hàng, 3 cột (ứng với 3 màu R G B). Các giá trị từ 0.0 đến 1.0

```
map(k,:) = [r(k) g(k) b(k)]
```

Ví dụ:

```
>> cm = colormap;
>> cm(57,:)
ans = 1 0 0
```

Giảng viên: Hoàng Xuân Dương





## II. ĐỒ HỌA 3D:

### 4. Sử dụng màu (tt)

`colormap(func(n))` tạo ma trận n hàng theo hàm `func`

`func` có thể là: `hsv`, `hot`, `cool`, `summer`, `gray`, `jet`, `bone`, `winter`...

Ví dụ:

```
>> cm=colormap(hot(20))
```

```
>> colormap(gray)
```

```
>> colormap jet
```

Giảng viên: Hoàng Xuân Dương



| Red  | Green | Blue | Color      |
|------|-------|------|------------|
| 0    | 0     | 0    | black      |
| 1    | 1     | 1    | white      |
| 1    | 0     | 0    | red        |
| 0    | 1     | 0    | green      |
| 0    | 0     | 1    | blue       |
| 1    | 1     | 0    | yellow     |
| 1    | 0     | 1    | magenta    |
| 0    | 1     | 1    | cyan       |
| 0.5  | 0.5   | 0.5  | gray       |
| 0.5  | 0     | 0    | Dark red   |
| 1    | 0.62  | 0.40 | copper     |
| 0.49 | 1     | 0.83 | aquamarine |

Giảng viên: Hoàng Xuân Dương



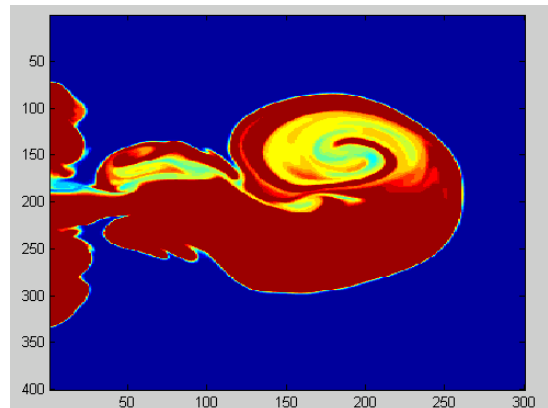


## II. ĐỒ HỌA 3D:

### 4. Sử dụng màu (tt)

Ví dụ 1:

```
>> load flujet  
>> image(X)  
>> colormap(jet)
```



Giảng viên: Hoàng Xuân Dương

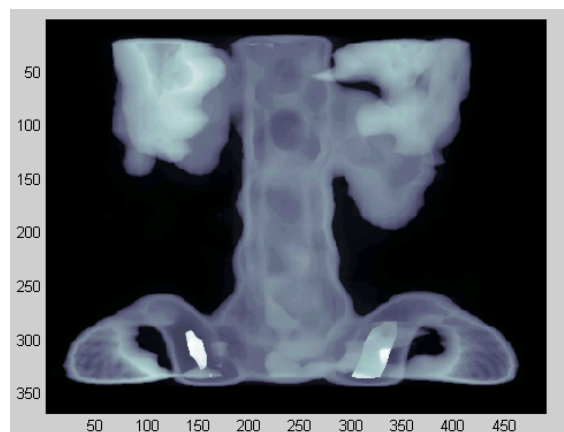


## II. ĐỒ HỌA 3D:

### 4. Sử dụng màu (tt)

Ví dụ 2:

```
>> load spine  
>> image(X)  
>> colormap bone
```



Giảng viên: Hoàng Xuân Dương



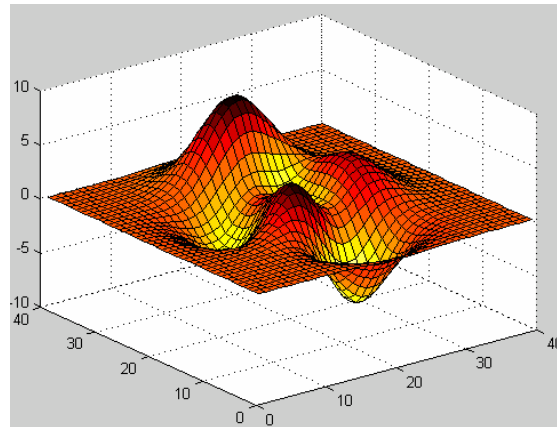


## II. ĐỒ HỌA 3D:

### 4. Sử dụng màu (tt)

Ví dụ 3:

```
>> P = peaks(40);  
>> C = del2(P);  
>> surf(P,C)  
>> colormap hot
```



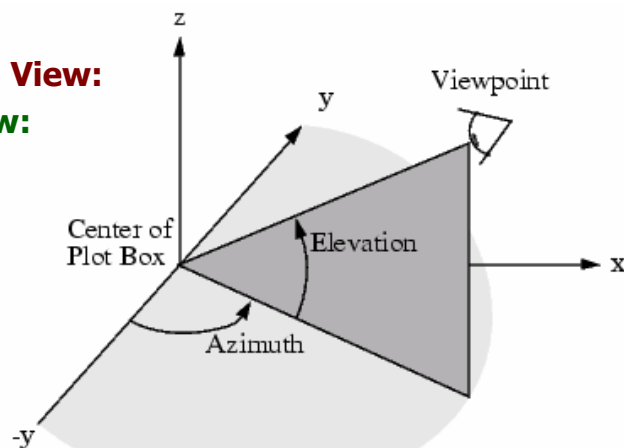
Giảng viên: Hoàng Xuân Dương



## II. ĐỒ HỌA 3D:

### 5. Định nghĩa View:

#### a. Đặt view:



Trong trường hợp mặc định, Matlab tự động chọn view  
+ 2D thì có azimuth (phương vị) =  $0^\circ$  và elevation (độ cao)  $90^\circ$   
+ 3D thì có azimuth =  $-37.5^\circ$  và elevation  $30^\circ$

Giảng viên: Hoàng Xuân Dương



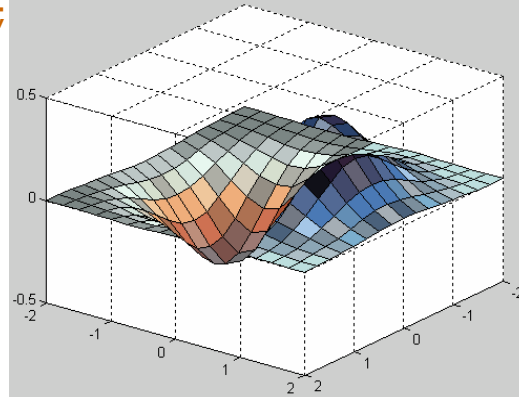
## II. ĐỒ HỌA 3D:

## 5. Định nghĩa View (tt)

## a. Đặt view (tt)

Ví dụ:

```
>> [X,Y]=meshgrid([-2:.25:2]);
>> Z=X.*exp(-X.^2 -Y.^2);
>> surf(X,Y,Z)
>> view([180 0])
>> view([-37.5 -30])
```



Giảng viên: Hoàng Xuân Dương

## II. ĐỒ HỌA 3D:

## 5. Định nghĩa View:

## b. Đặt trục:

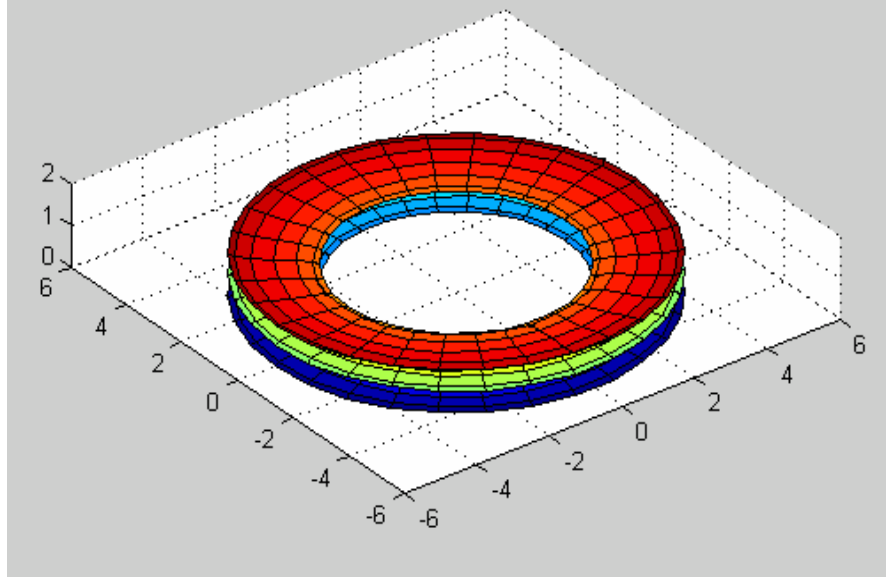
```
axis([xmin xmax ymin ymax zmin zmax])
```

Mặc định có tọa độ axis([-5 5 -5 5 0 1])

Ví dụ:

```
>> t = 0:pi/6:4*pi;
>> [x,y,z] = cylinder(4+cos(t),30);
>> surf(x,y,z)
>> axis square
>> axis equal
>> axis([-6 6 -6 6 0 2])
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương



## II. ĐỒ HOA 3D:

### 6. Di chuyển hình:

Ví dụ 1:

```
>> axis equal  
>> for j=1:30  
    plot(fft(eye(j+16)))  
    M(j)=getframe;  
end
```

Hay:

```
>> movie(M)
```

Giảng viên: Hoàng Xuân Dương



**II. ĐỒ HỌA 3D:****6. Di chuyển hình (tt)**

Ví dụ 1(tt)

Hay:

```
>> axis equal  
>> set(gca,'Nextplot','replacechildren')  
>> for j=1:30  
    plot(fft(eye(j+16)))  
    M(j)=getframe;  
end
```

Hay:

```
>> movie(M)  
>> movie(M,30)
```

Giảng viên: Hoàng Xuân Dương

**II. ĐỒ HỌA 3D:****6. Di chuyển hình (tt)**

Ví dụ 2:

```
>> [xx,yy,zz]=peaks(30);  
>> surf(xx,yy,zz);  
>> axis off  
>> for j=1:50  
    view(-37.5+24*(j-1),30)  
    n(:,1)=getframe;  
end  
>> movie(n)
```

Giảng viên: Hoàng Xuân Dương

### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 1. Hàm bar:

Dùng để diễn tả các dữ liệu rời rạc theo dạng biểu đồ cột

##### a. Hàm bar và barh:

```
bar(Y) % Đồ thị thanh đứng
bar(x,Y)
bar(...,width)
bar(...,'style')
bar(...,LineStyle)
[xb,yb] = bar(...)
h = bar(...)
barh(...) % Đồ thị thanh ngang
[xb,yb] = barh(...)
h = barh(...)
```

Giảng viên: Hoàng Xuân Dương

### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 1. Hàm bar (tt)

##### a. Hàm bar và barh (tt)

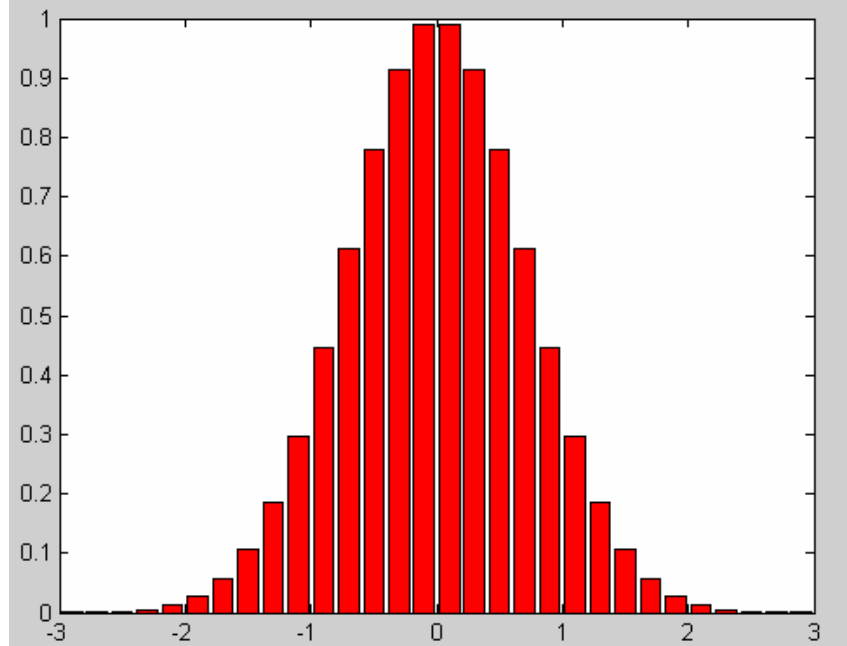
Trong đó:

- **width**: là độ rộng có mặc định là 0.8
- **LineStyle**: là màu vẽ [R G B]
- **style**: Kiểu của bar
  - group
  - stack

Ví dụ 1:

```
>> x = -2.9:0.2:2.9;
>> bar(x,exp(-x.*x))
>> colormap hsv
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 1. Hàm bar (tt)

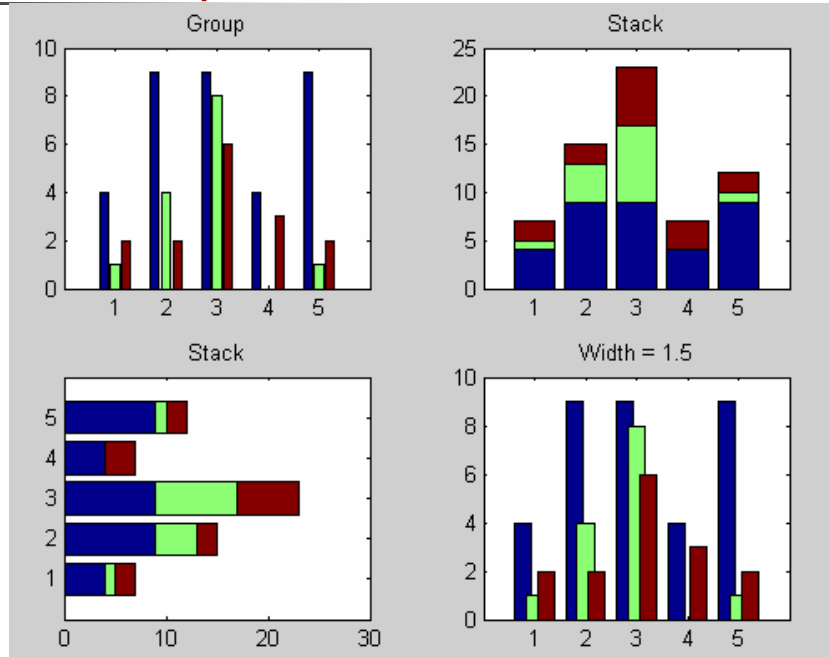
##### a. Hàm bar và barh (tt)

Ví dụ 2:

```
>> Y = round(rand(5,3)*10);  
>> subplot(2,2,1); bar(Y,'group'); title 'Group'  
>> subplot(2,2,2); bar(Y,'stack'); title 'Stack'  
>> subplot(2,2,3); barh(Y,'stack'); title 'Stack'  
>> subplot(2,2,4); bar(Y,1.5); title 'Width = 1.5'
```

Giảng viên: Hoàng Xuân Dương





Giảng viên: Hoàng Xuân Dương



### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 1. Hàm bar (tt)

##### a. Hàm bar và barh (tt)

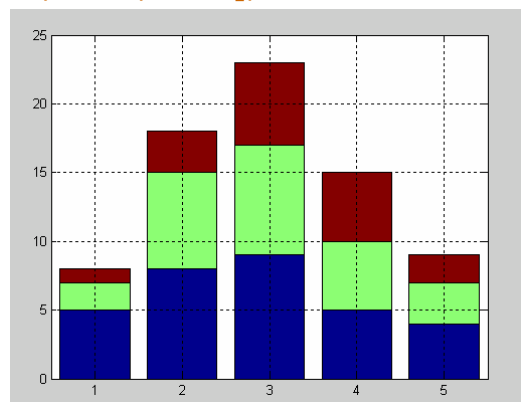
Ví dụ 3:

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

```
>> bar(Y,'stack')
```

```
>> grid on
```

```
>> set(gca,'layer','top')
```



Giảng viên: Hoàng Xuân Dương





### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 1. Hàm bar (tt)

##### b. Hàm bar3 và bar3h:

```
bar3(Y)
bar3(x,Y)
bar3(...,width)
bar3(...,'style')
bar3(...,LineStyle)
h = bar3(...)
bar3h(...)
h = bar3h(...)
```



### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 1. Hàm bar (tt)

##### b. Hàm bar3 và bar3h (tt)

Trong đó:

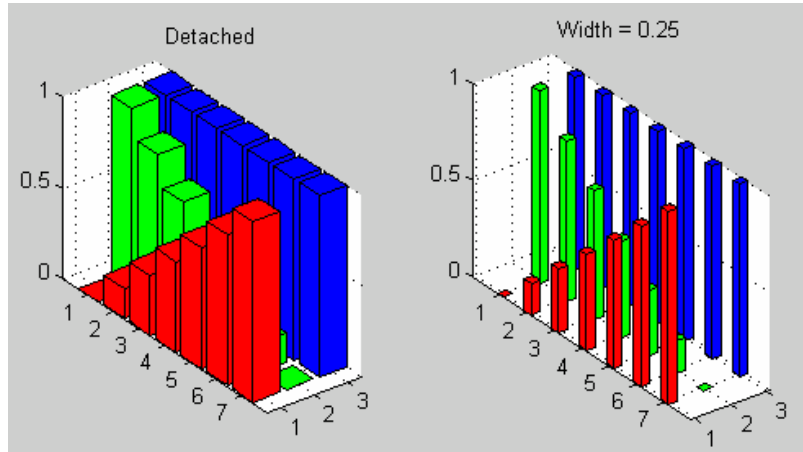
- **width**: là độ rộng có mặc định là 0.8
- **LineStyle**: là màu vẽ [R G B]
- **style**: Kiểu của bar
  - grouped
  - stacked
  - detached





Ví dụ:

```
>> Y = cool(7); colormap([1 0 0;0 1 0;0 0 1]);  
>> subplot(1,2,1); bar3(Y,'detached'); title('Detached')  
>> subplot(1,2,2); bar3(Y,0.25,'detached'); title('Width = 0.25')
```

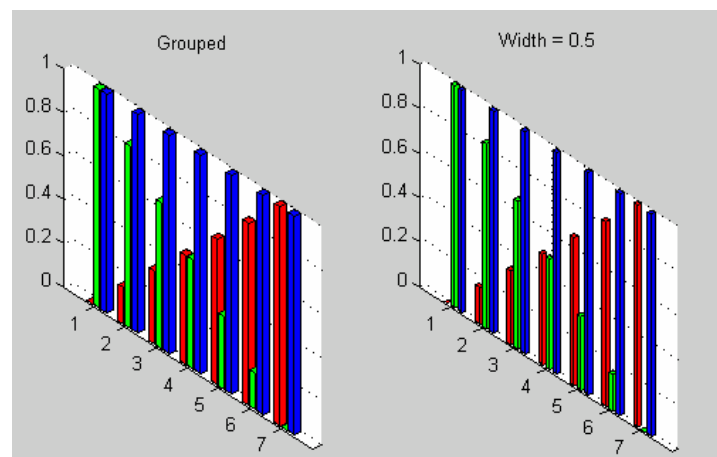


Giảng viên: Hoàng Xuân Dương



Ví dụ (tt)

```
>> subplot(1,2,1); bar3(Y,'grouped'); title('Grouped')  
>> subplot(1,2,2); bar3(Y,0.5,'grouped'); title('Width = 0.5')
```



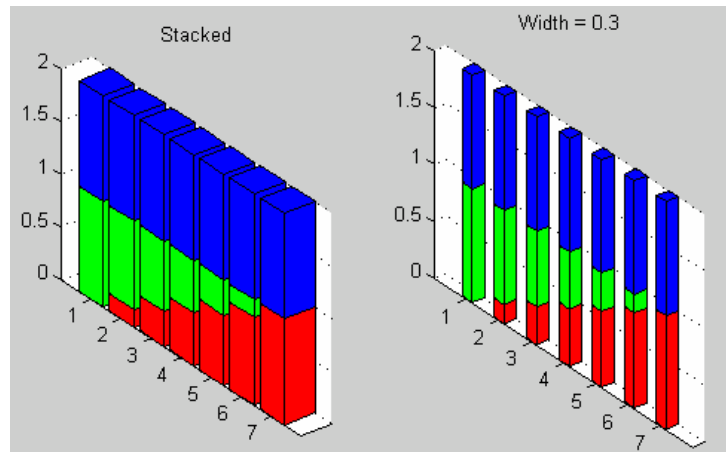
Giảng viên: Hoàng Xuân Dương





Ví dụ (tt)

```
>> subplot(1,2,1); bar3(Y,'stacked'); title('Stacked')  
>> subplot(1,2,2); bar3(Y,0.3,'stacked'); title('Width = 0.3')
```



Giảng viên: Hoàng Xuân Dương



### III. CÁC LOẠI HÀM ĐẶC BIỆT:

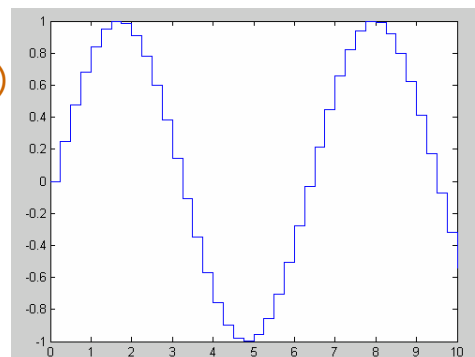
#### 1. Hàm bar (tt)

##### c. Hàm stairs:

```
stairs(Y)  
stairs(X,Y)  
stairs(...,LineStyle)  
[xb,yb] = stairs(Y)  
[xb,yb] = stairs(X,Y)
```

Ví dụ:

```
>> x = 0:.25:10;  
>> stairs(x,sin(x))
```



Giảng viên: Hoàng Xuân Dương





### III. CÁC LOẠI HÀM ĐẶC BIỆT:

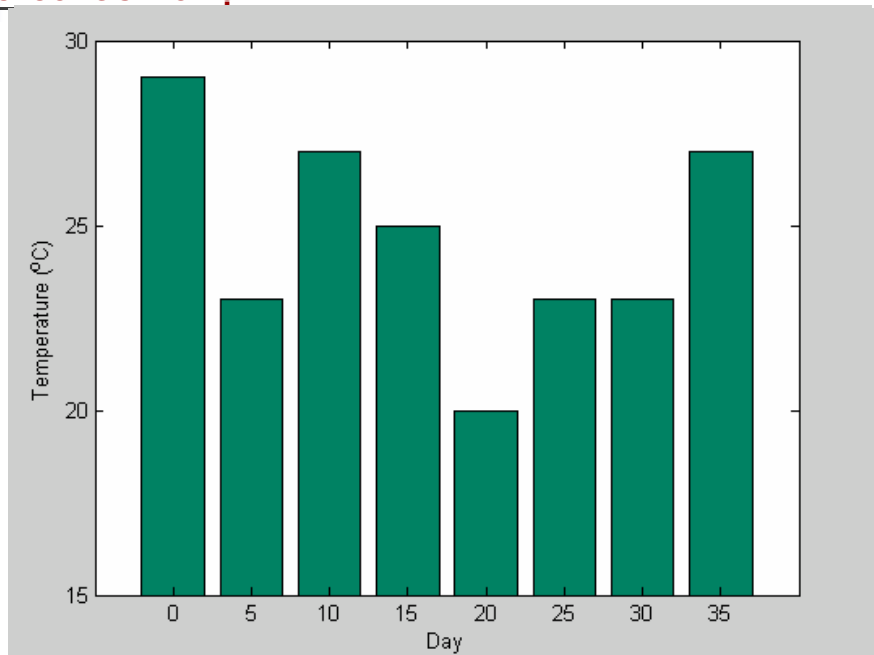
#### 1. Hàm bar (tt)

##### d. Giới hạn trục:

Ví dụ:

```
>> temp = [29 23 27 25 20 23 23 27];  
>> days = 0:5:35;  
>> bar(days,temp)  
>> xlabel('Day')  
>> ylabel('Temperature (^{o}C)')  
>> set(gca,'YLim',[15 30],'Layer','top')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương





### III. CÁC LOẠI HÀM ĐẶC BIỆT:

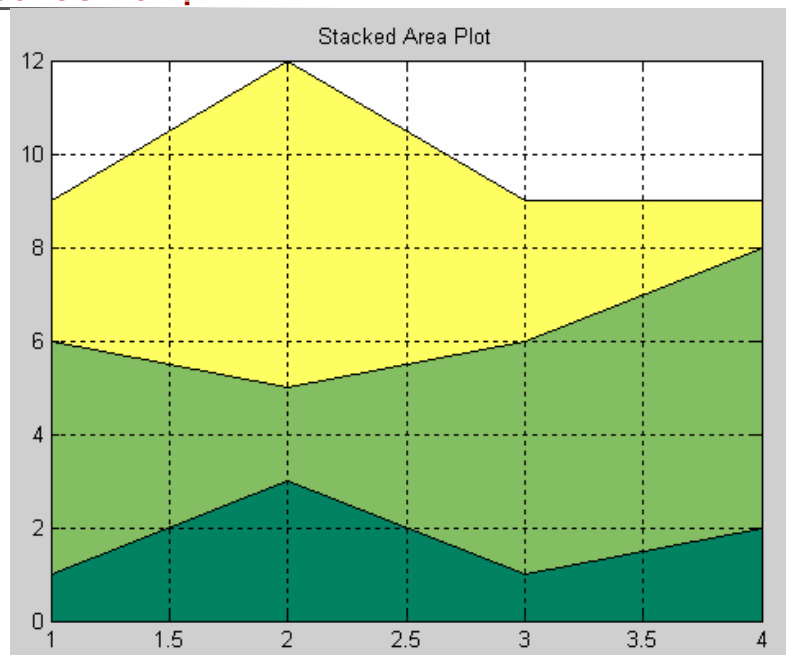
#### 2. Hàm area:

area(Y)  
area(X,Y)  
area(...,ymin)  
area(...,'PropertyName',PropertyValue,...)  
h = area(...)

Ví dụ:

```
>> Y = [1, 5, 3; 3, 2, 7; 1, 5, 3; 2, 6, 1];  
>> area(Y); grid on  
>> colormap summer  
>> set(gca,'Layer','top')  
>> title 'Stacked Area Plot'
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương





### III. CÁC LOẠI HÀM ĐẶC BIỆT:

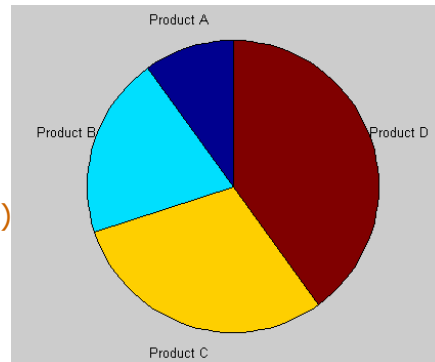
#### 3. Hàm pie: Hàm vẽ dạng rẽ quạt

##### a. pie:

```
pie(X)  
pie(X,explode)  
pie(...,labels)  
pie(axes_handle,...)  
h = pie(...)
```

Ví dụ 1:

```
>> pie(1:4,{'Product A',...  
'Product B','Product C','Product D'})
```



Giảng viên: Hoàng Xuân Dương



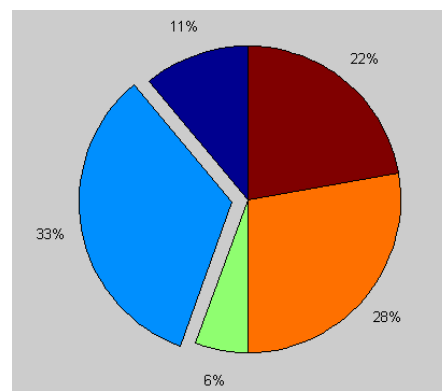
### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 3. Hàm pie (tt)

##### a. pie (tt)

Ví dụ 2:

```
>> x = [1 3 0.5 2.5 2];  
>> explode = [0 1 0 0 0];  
>> pie(x,explode)  
>> colormap jet
```



Giảng viên: Hoàng Xuân Dương



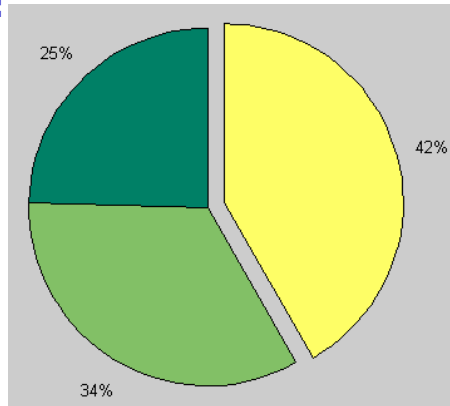
## III. CÁC LOẠI HÀM ĐẶC BIỆT:

## 3. Hàm pie (tt)

## a. pie (tt)

Ví dụ 3:

```
>> X = [ 19.3 22.1 51.6;
        34.2 70.3 82.4;
        61.4 82.9 90.8;
        50.5 54.9 59.1;
        29.4 36.3 47.0];
>> x = sum(X); explode = zeros(size(x));
>> [c,offset] = max(x);
>> explode(offset) = 1;
>> h = pie(x,explode); colormap summer
```



Giảng viên: Hoàng Xuân Dương

## III. CÁC LOẠI HÀM ĐẶC BIỆT:

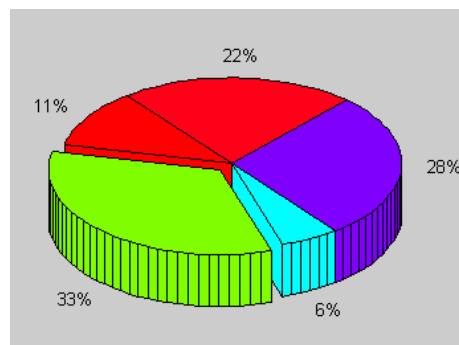
## 3. Hàm pie (tt)

## b. pie3:

```
pie3(X)
pie3(X,explode)
pie3(...,labels)
pie3(axes_handle,...)
h = pie3(...)
```

Ví dụ 1:

```
>> x = [1 3 0.5 2.5 2];
>> explode = [0 1 0 0 0];
>> pie3(x,explode)
>> colormap hsv
```



Giảng viên: Hoàng Xuân Dương





### III. CÁC LOẠI HÀM ĐẶC BIỆT:

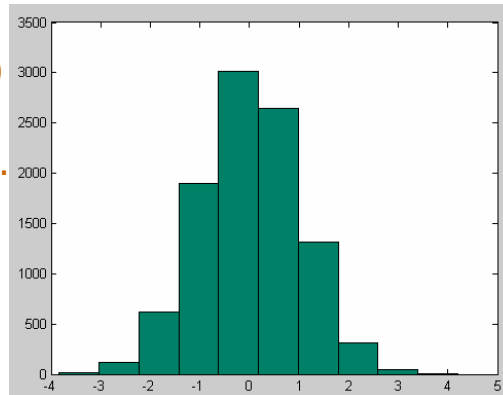
#### 4. Hàm Histograms:

##### a. Histograms trong tọa độ phẳng:

```
n = hist(Y)
n = hist(Y,x)
n = hist(Y,nbins)
[n,xout] = hist(...)
hist(...)
hist(axes_handle,...
```

Ví dụ 1:

```
>> yn = randn(10000,1);
>> hist(yn)
```



Giảng viên: Hoàng Xuân Dương



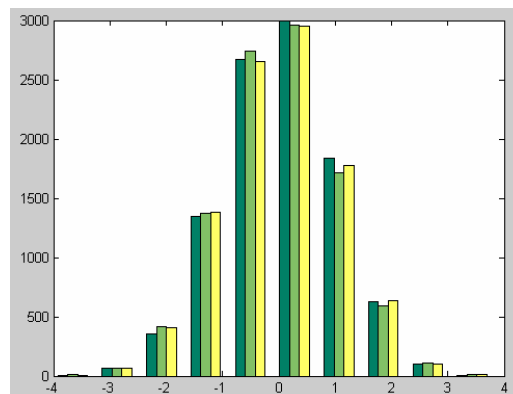
### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 4. Hàm Histograms (tt)

##### a. Histograms trong tọa độ phẳng (tt)

Ví dụ 2:

```
>> Y = randn(10000,3);
>> hist(Y)
```



Giảng viên: Hoàng Xuân Dương





### III. CÁC LOẠI HÀM ĐẶC BIỆT:

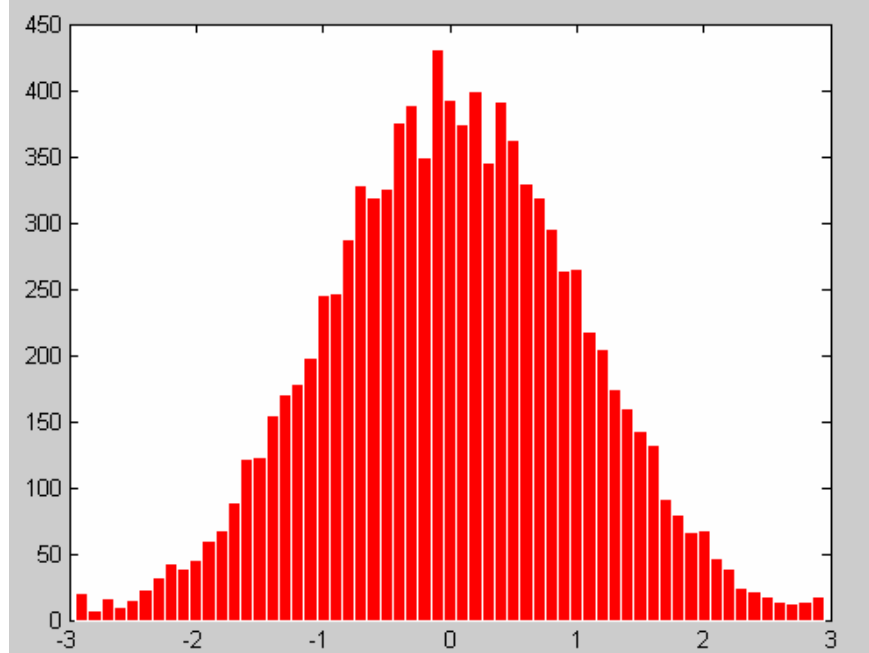
#### 4. Hàm Histograms (tt)

##### a. Histograms trong tọa độ phẳng (tt)

Ví dụ 3:

```
>> x = -2.9:0.1:2.9;  
>> y = randn(10000,1);  
>> hist(y,x)  
% thay đổi màu  
>> h = findobj(gca,'Type','patch');  
>> set(h,'FaceColor','r','EdgeColor','w')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương



## III. CÁC LOẠI HÀM ĐẶC BIỆT:

## 4. Hàm Histograms (tt)

## b. Histograms trong tọa độ cực:

```
rose(theta)
rose(theta,x)
rose(theta,nbins)
rose(axes_handles,...)
h = rose(...)
[tout,rout] = rose(...)
```

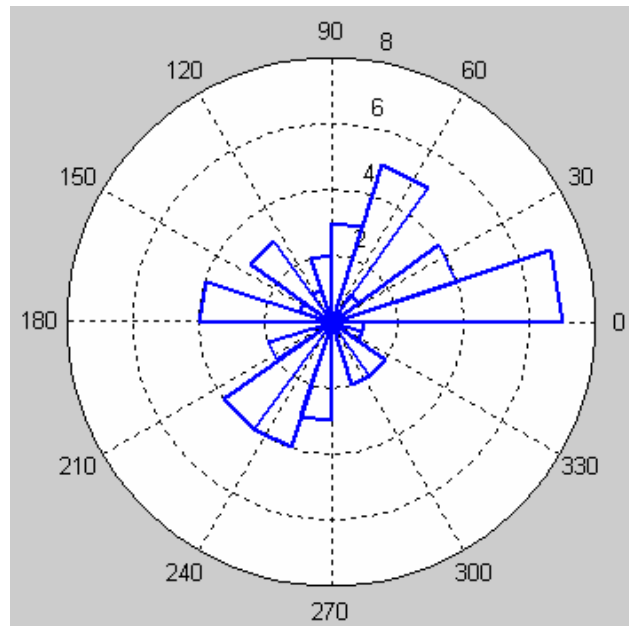
## III. CÁC LOẠI HÀM ĐẶC BIỆT:

## 4. Hàm Histograms (tt)

## b. Histograms trong tọa độ cực (tt)

Ví dụ:

```
>> theta = 2*pi*rand(1,50);
>> rose(theta)
>> hline = findobj(gca,'Type','line');
>> set(hline,'LineWidth',1.5)
```



### **III. CÁC LOẠI HÀM ĐẶC BIỆT:**

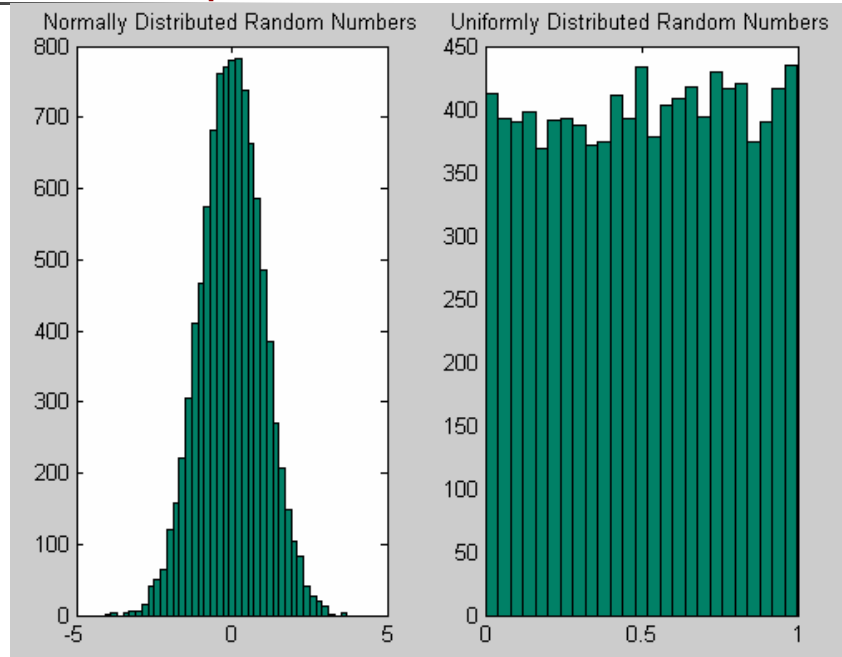
#### **4. Hàm Histograms (tt)**

##### **c. Chỉ định số Bin trong Histograms:**

Ví dụ:

```
>> yn = randn(10000,1);
>> yu = rand(10000,1);
>> x = min(yn):.2:max(yn);
>> subplot(1,2,1); hist(yn,x)
>> title('Normally Distributed Random Numbers','FontSize',10)
>> subplot(1,2,2); hist(yu,25)
>> title('Uniformly Distributed Random Numbers','FontSize',10)
```





Giảng viên: Hoàng Xuân Dương



### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 5. Dữ liệu rời rạc:

a. **Hàm stem:** Vẽ 1 chuỗi dữ liệu

`stem(Y)`

`stem(X,Y)`

`stem(...,'fill')` *% tô màu vòng tròn cuối stem*

`stem(...,LineStyle)`

`stem(axes_handle,...)`

`h = stem(...)`

`hlines = stem('v6',...)`

Giảng viên: Hoàng Xuân Dương





### III. CÁC LOẠI HÀM ĐẶC BIỆT:

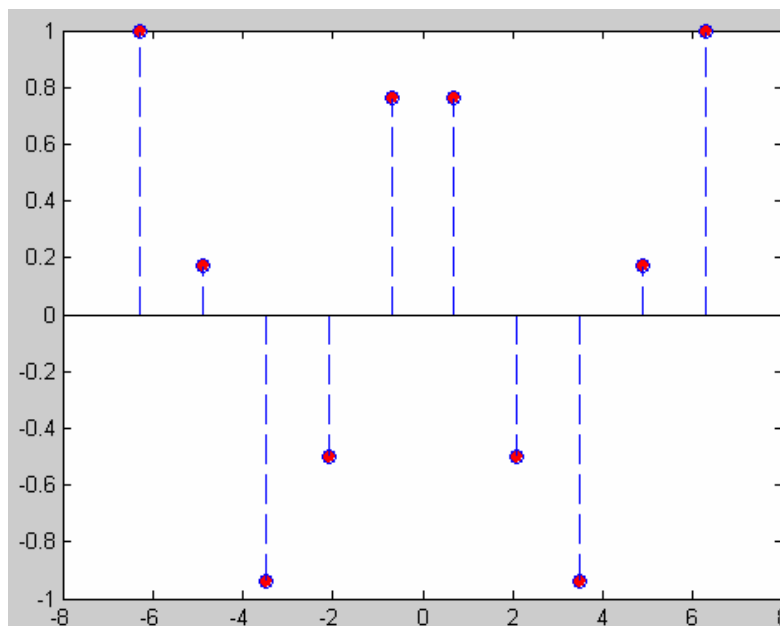
#### 5. Dữ liệu rời rạc (tt)

##### a. Hàm stem (tt)

Ví dụ:

```
>> t = linspace(-2*pi,2*pi,10);  
>> h = stem(t,cos(t),'fill','--');  
>> set(h,'MarkerFaceColor','red')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương





### III. CÁC LOẠI HÀM ĐẶC BIỆT:

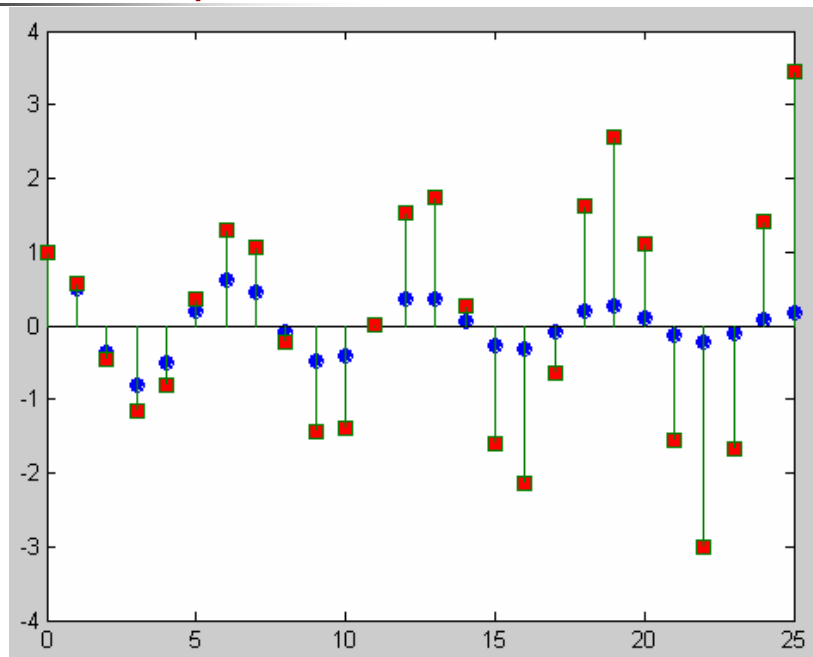
#### 5. Dữ liệu rời rạc (tt)

##### b. Vẽ nhiều hình:

Ví dụ 1: Vẽ 2 chuỗi dữ liệu trên 1 hình

```
>> x = 0:25;  
>> y = [exp(-.07*x).*cos(x);exp(.05*x).*cos(x)]';  
>> h = stem(x,y);  
>> set(h(1),'MarkerFaceColor','blue')  
>> set(h(2),'MarkerFaceColor','red','Marker','square')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương



## III. CÁC LOẠI HÀM ĐẶC BIỆT:

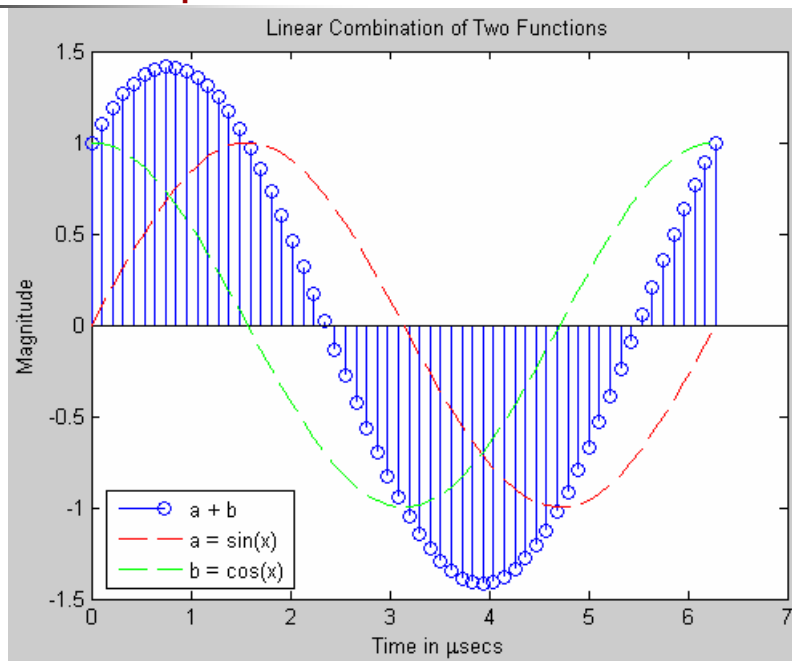
## 5. Dữ liệu rời rạc (tt)

## b. Vẽ nhiều hình (tt)

Ví dụ 2:

```
>> x = linspace(0,2*pi,60);
>> a = sin(x); b = cos(x);
>> stem_handles = stem(x,a+b);
>> hold on
>> plot_handles = plot(x,a,'-r',x,b,'--g');
>> hold off
>> legend_handles = [stem_handles(1);plot_handles];
>> legend(legend_handles,'a + b','a = sin(x)','b = cos(x)',3)
>> xlabel('Time in \musecs'); ylabel('Magnitude')
>> title('Linear Combination of Two Functions')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương



## III. CÁC LOẠI HÀM ĐẶC BIỆT:

## 5. Dữ liệu rời rạc (tt)

c. Hàm **stem3**: Vẽ 1 chuỗi dữ liệu 3 chiều

```
stem3(Z)
stem3(X,Y,Z)
stem3(...,'fill')
stem3(...,LineStyle)
h = stem3(...)
hlines = stem3('v6',...)
```

Giảng viên: Hoàng Xuân Dương

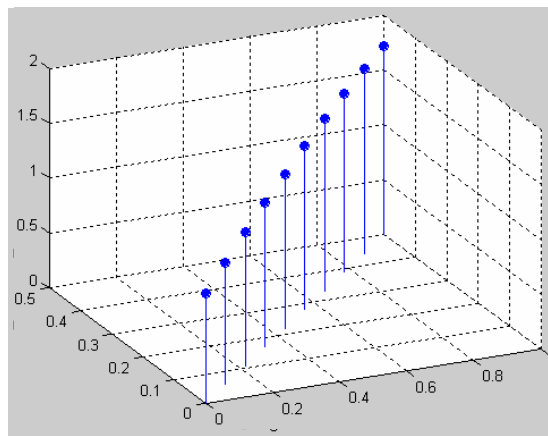
## III. CÁC LOẠI HÀM ĐẶC BIỆT:

## 5. Dữ liệu rời rạc (tt)

c. Hàm **stem3** (tt)

Ví dụ 1:

```
>> X = linspace(0,1,10);
>> Y = X./2;
>> Z = sin(X) + cos(Y);
>> stem3(X,Y,Z,'fill')
>> view(-25,30)
```



Giảng viên: Hoàng Xuân Dương

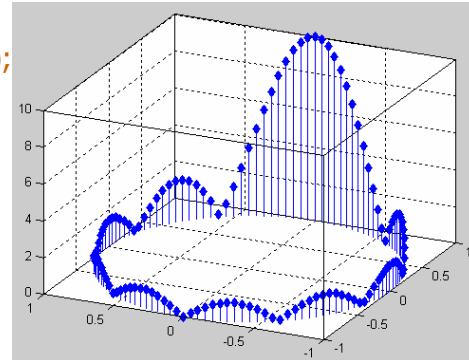
### III. CÁC LOẠI HÀM ĐẶC BIỆT:

#### 5. Dữ liệu rời rạc (tt)

##### c. Hàm stem3 (tt)

Ví dụ 2:

```
>> th = (0:127)/128*2*pi;
>> x = cos(th);
>> y = sin(th);
>> f = abs(fft(ones(10,1),128));
>> stem3(x,y,f,'d','fill')
>> view([-65 30])
```

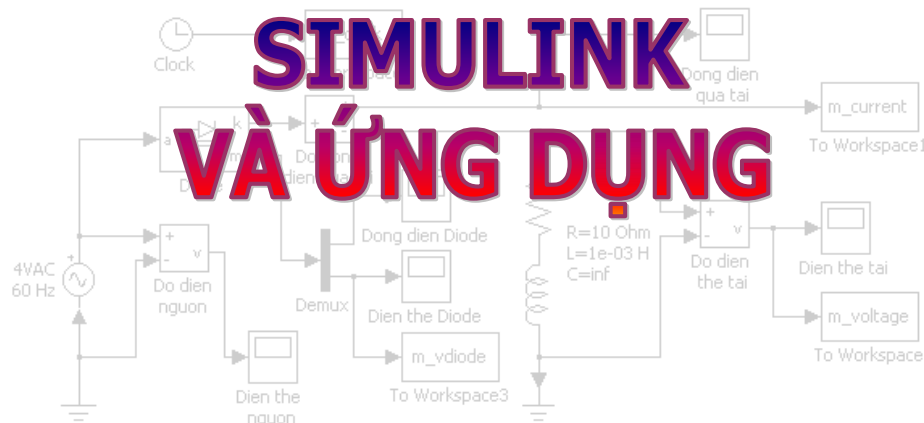


Giảng viên: Hoàng Xuân Dương



### CHƯƠNG 6:

## SIMULINK VÀ ỨNG DỤNG



Giảng viên: Hoàng Xuân Dương



- I. SIMULINK
- II. MỘT SỐ HỆ THỐNG
- III. MẠCH ĐIỆN
- IV. KHỐI SUBSYSTEM
- V. BÀI TẬP



## I. SIMULINK

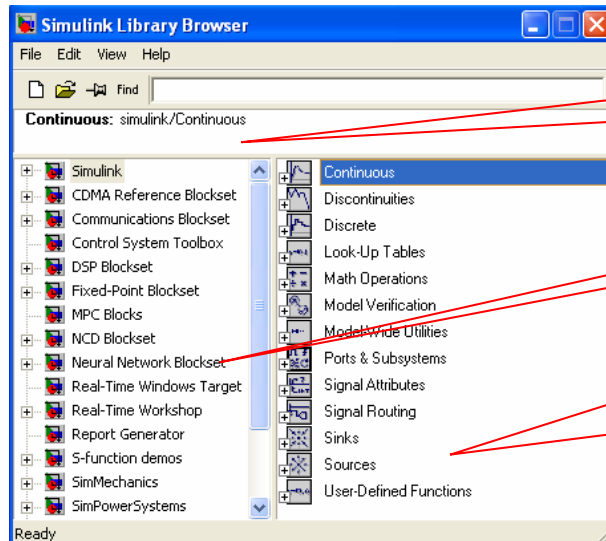
### 1. Khái niệm:

- Simulink là công cụ dùng để mô phỏng và phân tích các hệ thống liên tục, rời rạc, tuyến tính và phi tuyến thông qua giao diện dạng sơ đồ khối
- Trên cửa sổ lệnh gõ **simulink** hoặc chọn biểu tượng **simulink** trên thanh công cụ của Matlab
- Cửa sổ **Simulink Library Browser** xuất hiện



## I. SIMULINK

## 1. Khái niệm (tt)



*Mô tả thư viện  
được chọn*

*Các thư viện gốc*

*Các thư viện con  
của thư viện  
được chọn*

Giảng viên: Hoàng Xuân Dương

## I. SIMULINK

## 1. Khái niệm (tt)

- Simulink tổ chức các khối theo thư viện, mỗi thư viện gồm nhiều thư viện con. Các thư viện phục vụ cho các chuyên ngành khác nhau
- Các thư viện con là tập hợp các khối
- Khối biểu diễn một hệ thống động sơ cấp, gồm có đầu vào, đầu ra và các trạng thái bên trong
- Mỗi khối liên kết một hàm hệ thống cho biết quan hệ giữa đầu vào với đầu ra

Giảng viên: Hoàng Xuân Dương

## I. SIMULINK

## 1. Khái niệm (tt)

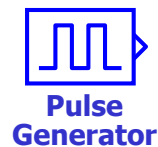
Ví dụ:



→ Khối nhân vô hướng hoặc ma trận



→ Khối tính min (hoặc max) vector ngõ vào



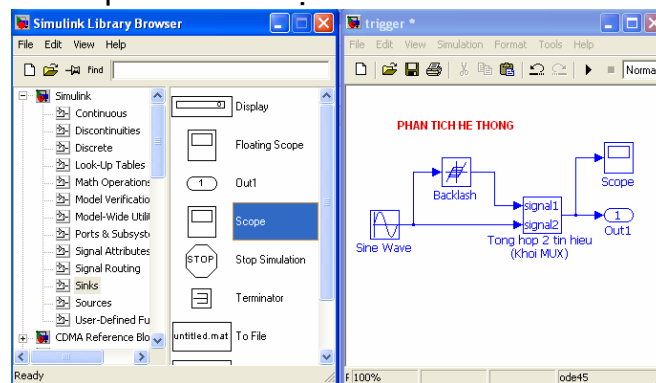
→ Khối phát xung

Giảng viên: Hoàng Xuân Dương

## I. SIMULINK

## 2. Thực hiện mô hình:

- Muốn tạo một mô hình, chọn **File-New-Model**, một cửa sổ soạn thảo mở ra.
- Chọn các khối trong cửa sổ **Simulink Library Browser** rồi kéo qua cửa sổ soạn thảo



Giảng viên: Hoàng Xuân Dương

**I. SIMULINK****2. Thực hiện mô hình (tt)**

| Các thao tác khi tạo sơ đồ khối |                                    |
|---------------------------------|------------------------------------|
| Thao tác                        | Phím + chuột                       |
| Chọn khối hay đường             | LMB (left mouse button)            |
| Chọn nhiều khối hay nhiều đường | Shift+LMB                          |
| Chọn khối kế                    | Tab                                |
| Chọn khối trước                 | Shift+Tab                          |
| Chép khối từ cửa sổ khác        | Bấm chuột và kéo thả (kéo khối)    |
| Di chuyển khối                  | Kéo khối                           |
| Tạo khối giống nhau             | RMB và kéo thả hay LMB+Ctrl và kéo |
| Nối các khối                    | LMB                                |
| Tháo khối                       | Shift+kéo khối                     |

Giảng viên: Hoàng Xuân Dương

**I. SIMULINK****2. Thực hiện mô hình (tt)**

| Các thao tác khi tạo sơ đồ khối |                                 |
|---------------------------------|---------------------------------|
| Thao tác                        | Phím + chuột                    |
| Mở hệ con đã chọn               | Enter                           |
| Chuyển đến cha của hệ con       | Esc                             |
| Xóa khối                        | Chọn + bấm del                  |
| Tạo chú giải                    | Double click trong giản đồ      |
| Chép chú giải                   | Ctrl+kéo                        |
| Di chuyển chú giải              | Kéo                             |
| Biên tập chú giải               | LMB vào text                    |
| Bỏ chú giải                     | Shift+chọn chú giải rồi bấm del |
| Quay khối 90° theo chiều kim    | Ctrl+R                          |
| Lật khối (đảo đầu vào ra)       | Ctrl+I                          |

Giảng viên: Hoàng Xuân Dương

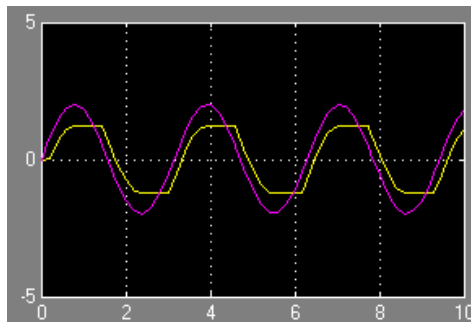




## I. SIMULINK

### 2. Thực hiện mô hình (tt)

- Muốn thêm chú thích vào hình, chọn vào chỗ trống trong sơ đồ khối, nhập các chú thích.
- Sau khi vẽ xong sơ đồ khối, chuyển sang giai đoạn mô phỏng. Chọn **Simulation/start**
- Chọn **File/save** để lưu sơ đồ thành tập tin với đuôi **.mdl**



Giảng viên: Hoàng Xuân Dương



### 3. Thư viện simulink:

#### Continuous-Khối hàm liên tục

|                          |                                  |
|--------------------------|----------------------------------|
| Derivative               | Đạo hàm tín hiệu vào             |
| Integrator               | Tích phân tín hiệu vào           |
| Memory                   | Khối nhớ                         |
| State-Space              | Phương trình trạng thái liên tục |
| Transfer Fcn             | Hàm truyền liên tục              |
| Transport Delay          | Delay                            |
| Variable Transport Delay | Delay thay đổi                   |
| Zero-Pole                | Hàm truyền theo cực và zero      |

#### Discontinuous-Khối hàm phi tuyến

|                            |                   |
|----------------------------|-------------------|
| Backlash                   | Khe hở            |
| Coulomb & Viscous Friction | Ma sát khô và ướt |
| Dead Zone                  | Vùng chết         |
| Manual Switch              | Chọn bằng tay     |

Giảng viên: Hoàng Xuân Dương





|                                  |                                  |
|----------------------------------|----------------------------------|
| Multiport Switch                 | Chọn các khối vào                |
| Quantizer                        | Lượng tử                         |
| Rate Limiter                     | Giới hạn đạo hàm tín hiệu        |
| Relay                            | Khâu rơle                        |
| Saturation                       | Khâu bão hòa                     |
| Switch                           | Chuyển mạch giữa hai ngõ vào     |
| <b>Discrete-Các khối rời rạc</b> |                                  |
| Discrete Filter                  | Lọc IIR và FIR                   |
| Discrete State-Space             | Phương trình trạng thái rời rạc  |
| Discrete-Time Integrator         | Tích phân rời rạc                |
| Discrete Transfer Fcn            | Hàm truyền rời rạc               |
| Discrete Zero-Pole               | Hàm truyền rời rạc theo cực zero |
| First-Order Hold                 | Bộ lấy mẫu và giữ bậc một        |
| Unit Delay                       | Bộ trễ một chu kỳ lấy mẫu        |

**Giảng viên: Hoàng Xuân Dương**



|  |  |
|--|--|
| Zero-Order Hold                          | Bộ lấy mẫu và giữ bậc zero             |
| <b>Look up Tables-Khối tra bảng</b>      |  |
| Direct Look-Up Table (n-D)               | Bảng tra hai chiều                     |
| Fcn                                      | Tạo hàm                                |
| Look-Up Table                            | Tra bảng                               |
| Look-Up Table (2-D)                      | Tra bảng hai chiều                     |
| Look-Up Table (n-D)                      | Tra bảng n chiều                       |
| MATLAB Fcn                               | Hàm Matlab                             |
| S-Function                               | Hàm S                                  |
| <b>Math Operations-Thư viện toán học</b> |  |
| Abs                                      | Lấy trị tuyệt đối                      |
| Algebraic Constraint                     | Giới hạn đại số                        |
| Bitwise Logical Operator                 | Toán tử logic (dịch bit, mask, invert) |
| Combinatorial Logic                      | Mạch tổ hợp                            |

**Giảng viên: Hoàng Xuân Dương**







|                            |  |
|----------------------------|--|
| Complex to Magnitude-Angle | Tính biên độ và pha tín hiệu số phức   |
| Complex to Real-Imag       | Tính phần thực và phần ảo của số phức  |
| Derivative                 | Tính đạo hàm                           |
| Dot Product                | Tính tích chấm                         |
| Gain                       | Khối tỉ lệ                             |
| Logical Operator           | Toán tử logic                          |
| Magnitude-Angle to Complex | Tạo tín hiệu số phức từ biên độ và pha |
| Math Function              | Hàm toán học                           |
| Matrix Gain                | Nhân tín hiệu vào với ma trận          |
| MinMax                     | Lấy cực đại hay cực tiểu               |
| Product                    | Tính tích hay thương đầu vào           |
| Real-Imag to Complex       | Đổi phần thực và ảo ra tín hiệu phức   |
| Relational Operator        | Toán tử quan hệ                        |
| Rounding Function          | Làm tròn                               |

**Giảng viên: Hoàng Xuân Dương**



|  |                                       |
|--|---------------------------------------|
| Sign                                     | Lấy dấu                               |
| Slider Gain                              | Con trượt thay đổi độ lợi             |
| Sum                                      | Tính tổng                             |
| Trigonometric Function                   | Tính hàm lượng giác                   |
| <b>Ports &amp; Subsystems-Tạo hệ con</b> |                                       |
| In1                                      | Tạo ngõ vào cho subsystem             |
| Out1                                     | Tạo ngõ ra cho subsystem              |
| Subsystem                                | Khối subsystem chứa inport và outport |
| <b>Signal Routing-Tạo bus tín hiệu</b>   |                                       |
| Bus Selector                             | Chọn tín hiệu vào                     |
| Data Store Memory                        | Ấn định khối nhớ dữ liệu              |
| Data Store Read                          | Đọc khối nhớ dữ liệu                  |
| Data Store Write                         | Ghi khối nhớ dữ liệu                  |
| Demux                                    | Tách tín hiệu                         |

**Giảng viên: Hoàng Xuân Dương**





|  |  |
|--|--|
| From   | Nhận tín hiệu từ khối goto             |
| Goto   | Chuyển tín hiệu đến khối From          |
| Goto Tag Visibility                                | Ẩn định tag của khối goto              |
| Manual Switch                                      | Khóa hai chiều đk bằng tay             |
| Merge  | Hợp nhiều đường thành đường vô hướng   |
| Multiport Switch                                   | Khóa nhiều chiều                       |
| Mux  | Kết hợp nhiều đường thành đường vector |
| Selector   | Chọn các phần tử của vector ngõ vào    |
| Switch   | Khóa hai chiều                         |
| <b>Sinks-Các khối hiển thị hay lưu tín hiệu ra</b> |  |
| Display  | Hiển thị giá trị đầu vào               |
| Scope  | Hiển thị giá trị ra khi mô phỏng       |
| Stop Simulation                                    | Ngừng mô phỏng                         |
| To File  | Ghi dữ liệu vào file                   |

**Giảng viên: Hoàng Xuân Dương**



|                                      |                                      |
|--------------------------------------|--------------------------------------|
| To Workspace                         | Ghi dữ liệu vào biến trong Workspace |
| XY Graph                             | Vẽ đồ thị X-Y                        |
| <b>Sources-Các khối tạo tín hiệu</b> |                                      |
| Band-Limited White Noise             | Tạo nhiễu trắng                      |
| Chirp Signal                         | Tạo sóng sin tần số tăng dần         |
| Clock                                | Thời gian mô phỏng                   |
| Constant                             | Tạo giá trị hằng                     |
| Digital Clock                        | Tạo tg mô phỏng ở các khoảng lấy mẫu |
| Digital Pulse Generator              | Tạo xung số                          |
| From File                            | Đọc dữ liệu từ file                  |
| From Workspace                       | Đọc dữ liệu từ biến trong Workspace  |
| Pulse Generator                      | Tạo xung                             |
| Ramp                                 | Tạo hàm dốc                          |
| Random Number                        | Tạo số ngẫu nhiên phân bố chuẩn      |

**Giảng viên: Hoàng Xuân Dương**





|                       |                               |
|-----------------------|-------------------------------|
| Repeating Sequence    | Tạo tín hiệu lặp lại          |
| Signal Generator      | Tạo dạng sóng                 |
| Sine Wave             | Tạo sóng sin                  |
| Step                  | Tạo hàm nấc                   |
| Uniform Random Number | Tạo số ngẫu nhiên phân bố đều |

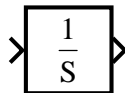


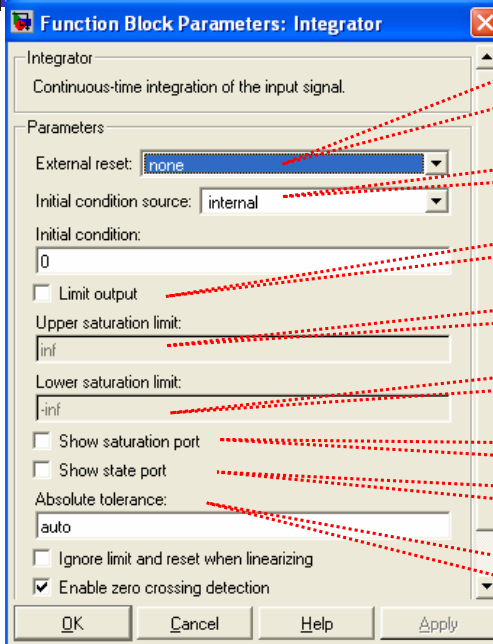
## I. SIMULINK

### 4. Function block parameters:

- Để định các thông số cho các hàm hệ thống liên kết với khối, double click vào khối để mở cửa sổ **Function block parameters**.

Ví dụ 1: Khối tích phân **integrator**





Reset về điều kiện đầu khi tín hiệu reset tác động (none, rising, falling,...)

Nguồn điều kiện đầu (int, ext)

Chọn giới hạn ra hay không ?

Giới hạn trên

Giới hạn dưới

Thêm ngõ ra bão hoà

Thêm ngõ ra trạng thái

Dùng sai tuyệt đối trạng thái khối

Giảng viên: Hoàng Xuân Dương

## I. SIMULINK

### 4. Function block parameters (tt)

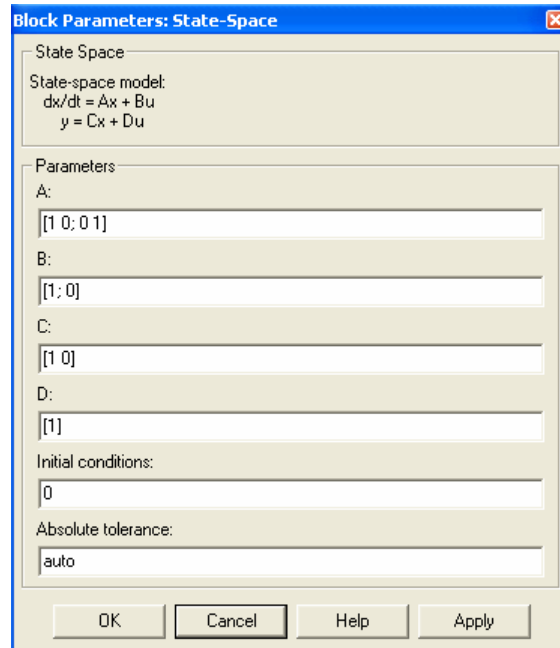
Ví dụ 2: Khối phương trình trạng thái State-Space

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

Điều chỉnh các thông số để phương trình trạng thái của hệ thống có dạng:

$$\begin{aligned} \dot{x} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix} x + u \end{aligned}$$

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

## I. SIMULINK

### 5. Mô phỏng:

- Có thể chạy mô phỏng bằng **Simulation/Start** hoặc trong command window:

```
>> [t,x,y]=sim(model,timespan,options,ut)
```

```
>> [t,x,y1,y2,...,yn]=sim(model,timespan,options,ut)
```

#### Trong đó:

- ★ **t**: vector thời gian mô phỏng
- ★ **x**: ma trận các trạng thái liên tục và rời rạc
- ★ **y**: Ma trận các ngõ ra
- ★ **model**: tên mô hình
- ★ **timespan**: thời gian bắt đầu và kết thúc mô phỏng
- ★ **options**: Các thông số tùy chọn theo lệnh simset
- ★ **ut**: giá trị tín hiệu vào

Giảng viên: Hoàng Xuân Dương

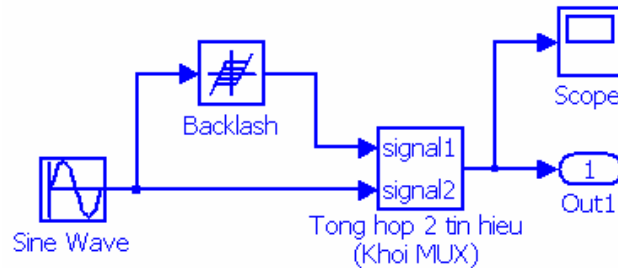
## II. MỘT SỐ HỆ THỐNG

### 1. Hệ thống Backlash:

#### a. Khối MUX có 2 ngõ vào:

Thiết kế hệ thống như hình vẽ, lưu thành **trigger.mdl**

#### PHÂN TÍCH HỆ THỐNG



Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

### 1. Hệ thống Backlash:

#### a. Khối MUX có 2 ngõ vào (tt)

- Khối Sin Wave: **Simulink/ Sources**
- Khối Backlash: **Simulink/ Discontinuities**
- Khối Mux: **Simulink/ Signal routing**
- Khối Outport: **Simulink/ Ports & Subsystems**
- Khối Scope: **Simulink/ Sinks**

Giảng viên: Hoàng Xuân Dương

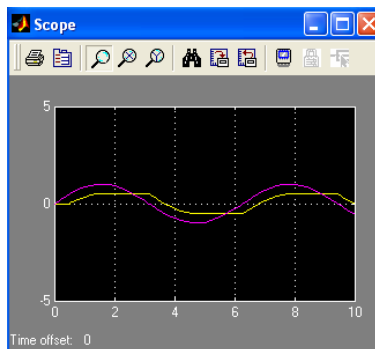
## II. MỘT SỐ HỆ THỐNG

### 1. Hệ thống Backlash:

#### a. Khối MUX có 2 ngõ vào (tt)

Chạy thử:

1. Chọn **simulation/start**
2. Double click khối **scope** để xem dạng sóng



Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

### 1. Hệ thống Backlash:

#### a. Khối MUX có 2 ngõ vào (tt)

Trong command window:

```
>> [t,x,y]=sim('trigger',20)    % tập thời gian 0-20
```

y là ma trận kết quả có 2 cột, cột 1 chứa kết quả sau khi qua khối backlash, cột 2 là sóng sin nguyên thủy

```
>> plot(t,y)
```

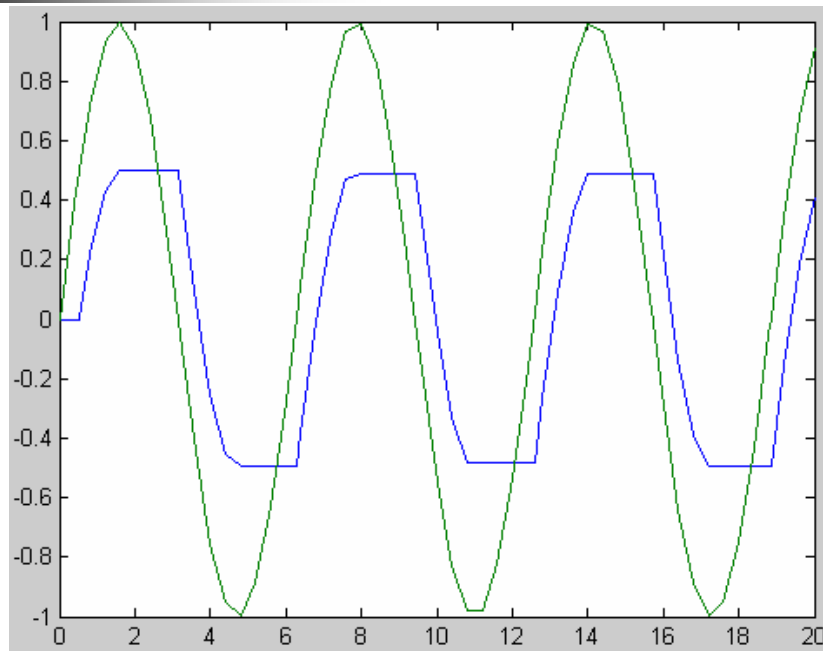
hay

```
>> plot(t,y(:,1))                % Vẽ cột 1
```

hay

```
>> plot(t,y(:,2))                % Vẽ cột 2
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

### 1. Hệ thống Backlash:

#### a. Khối MUX có 2 ngõ vào (tt)

- Có thể thay đổi thông số cho nguồn sin bằng cách double click khối Sin Wave, thay đổi giá trị như biên độ, tần số,...
- Có thể gọi lại mô hình bằng cách trong Command window:

```
>> trigger
```

Giảng viên: Hoàng Xuân Dương

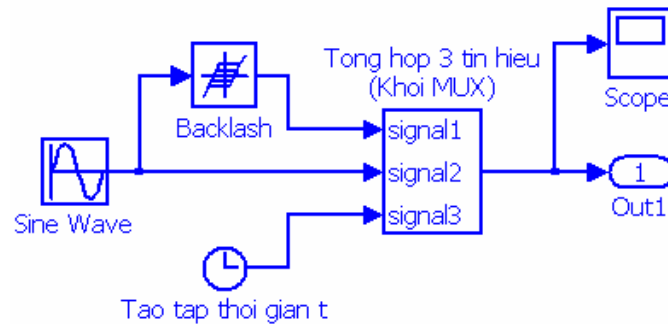


## II. MỘT SỐ HỆ THỐNG

### 1. Hệ thống Backlash (tt)

#### b. Khối MUX có 3 ngõ vào

##### KHOI MUX CO 3 NGO VAO



Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

### 1. Hệ thống Backlash (tt)

#### b. Khối MUX có 3 ngõ vào (tt)

```
>> [t,x,y]=sim('trigger3',5)
```

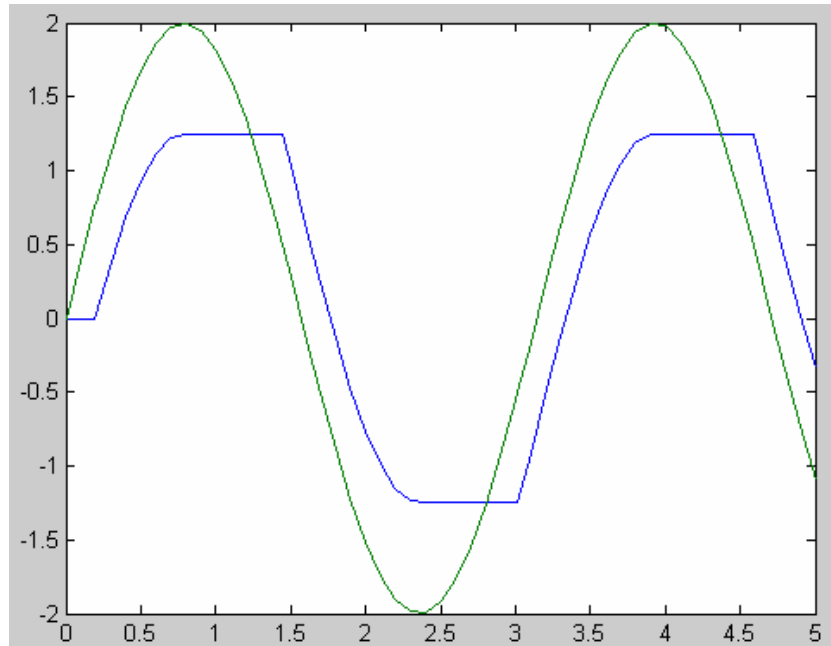
y là ma trận kết quả có 3 cột, với cột 3 là ma trận t

```
>> plot(y(:,3),y(:,1:2))
```

Hay

```
>> plot(y(:,3),y(:,1), y(:,3),y(:,2))
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương



## II. MỘT SỐ HỆ THỐNG

### 2. Hệ thống phương trình Van Der Pol

Phương trình Van der Pol có dạng:

$$x'' + (x^2 + 1)x' + x = 0$$

Với:

$$x' = x_1(1 - x_2^2) - x_2$$

$$x_2' = x_1$$

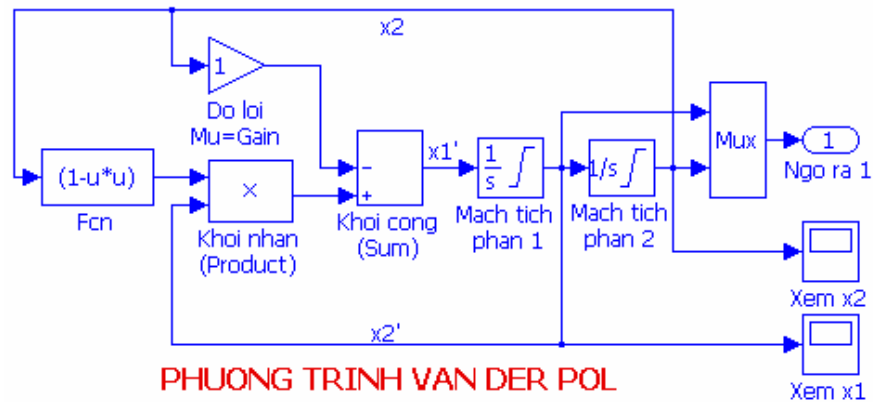
trong đó hằng số  $\mu = 1$

Giảng viên: Hoàng Xuân Dương



## II. MỘT SỐ HỆ THỐNG

## 2. Hệ thống phương trình Van Der Pol



Giảng viên: Hoàng Xuân Dương

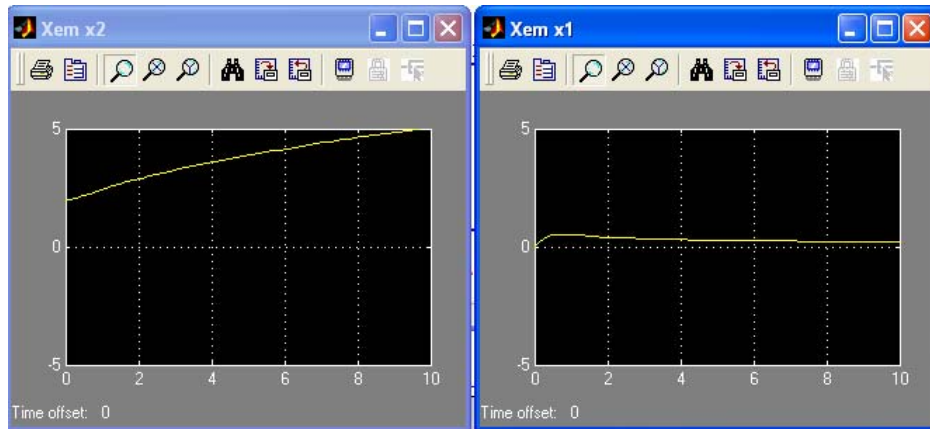
## II. MỘT SỐ HỆ THỐNG

## 2. Hệ thống phương trình Van Der Pol

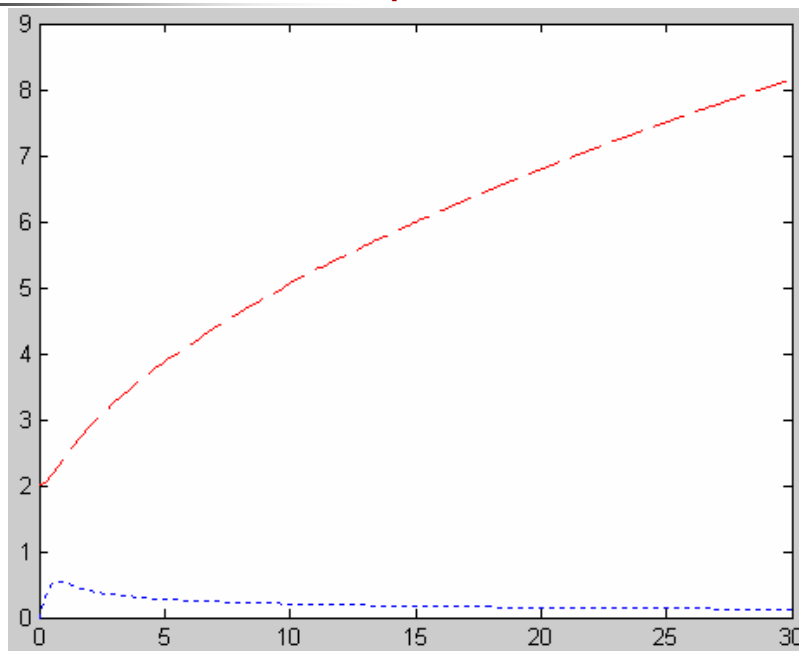
Thực hiện với:

- Khối Fcn: **Simulink/ User-defines Function**
  - Khối Product, Gain, Sum: **Simulink/ Math operations**
  - Khối Integrator: **Simulink/ Continuous**
- Ở khối tích phân thứ 2, vào properties chọn điều kiện đầu là bằng 1
- Lưu mô hình với tên **ptvdp.mdl**
- ```
>> [t,x,y]=sim('ptvdp',30);
>> plot(t,y(:,1),'b',t,y(:,2),'--r')
```

Giảng viên: Hoàng Xuân Dương



**Giảng viên: Hoàng Xuân Dương**



**Giảng viên: Hoàng Xuân Dương**

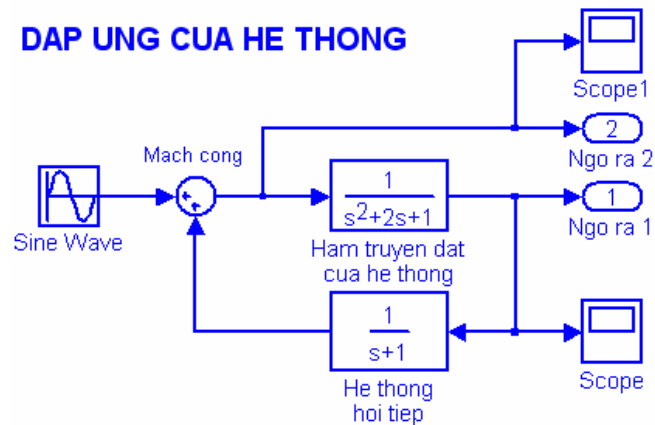


## II. MỘT SỐ HỆ THỐNG

## 3. Đáp ứng hệ thống

## a. Sử dụng sóng sin vào cố định

## DAP UNG CUA HE THONG



Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

## 3. Đáp ứng hệ thống

## a. Sử dụng sóng sin vào cố định

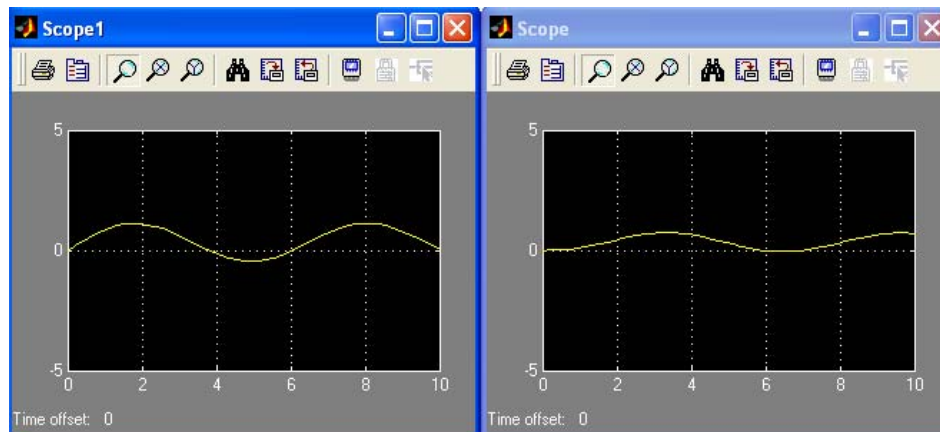
- Lưu mô hình với tên `transfer.mdl`

```
>> [t,x,y]=sim('transfer',10);
```

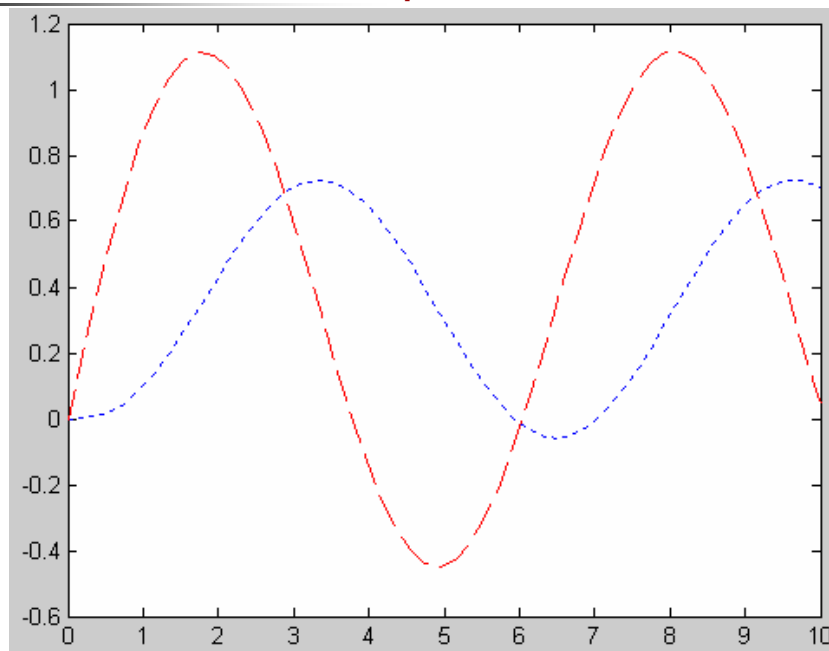
```
>> plot(t,y(:,1),'b',t,y(:,2),'-r')
```

- Hệ thống có hồi tiếp âm nên sóng sin vào có tần số càng cao thì sóng ra tại 'Ngo ra 1' có biên độ càng nhỏ. Đây là dạng mạch lọc thông thấp

Giảng viên: Hoàng Xuân Dương



**Giảng viên: Hoàng Xuân Dương**



**Giảng viên: Hoàng Xuân Dương**

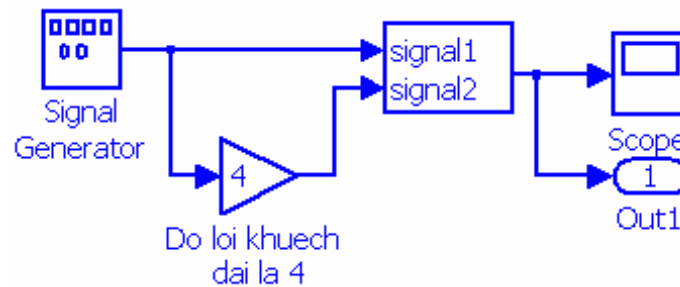


## II. MỘT SỐ HỆ THỐNG

## 3. Đáp ứng hệ thống (tt)

## b. Khối Signal Generator

## KHUẾCH ĐẠI TIN HIỆU



Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

## 3. Đáp ứng hệ thống

## b. Khối Signal Generator (tt)

Thực hiện với:

- Khối Signal Generator: **Simulink/ Source**

Vào properties chọn đơn vị của tần số là rad/s (hình vẽ), dạng sóng có thể là sin, vuông, tam giác hoặc ngẫu nhiên.

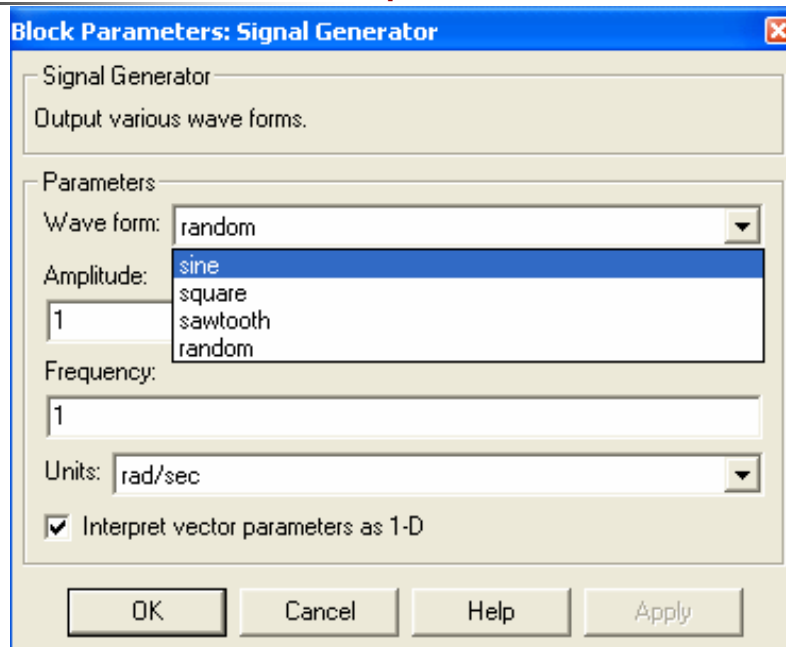
- Lưu mô hình với tên **k dai.mdl**

```
>> [t,x,y]=sim('k dai',10);
```

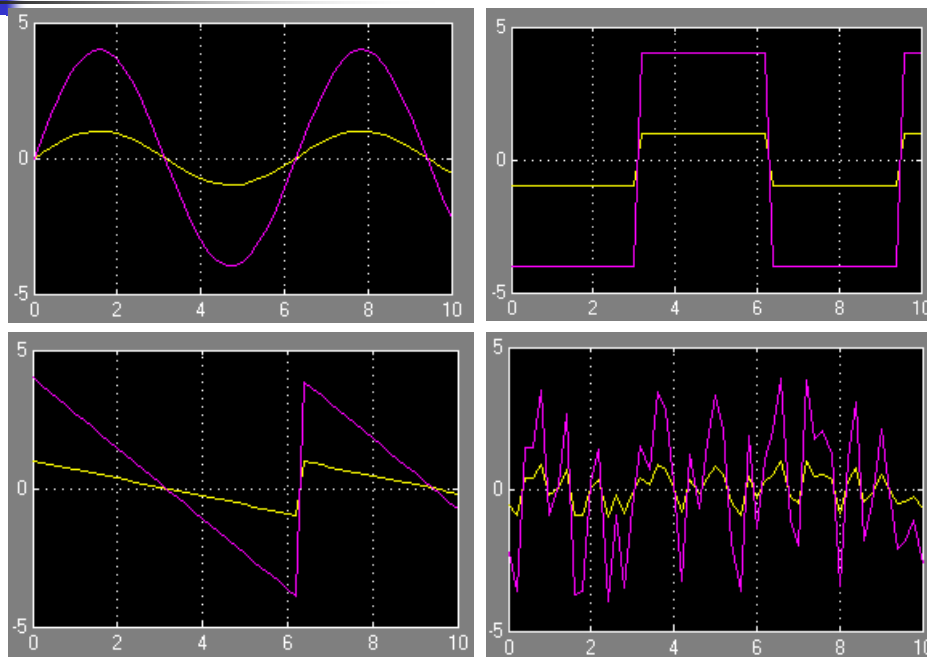
```
>> plot(t,y)
```

*Khối Signal Generator có thể thay bằng khối Inport để nhập tín hiệu cần khuếch đại từ ngoài vào*

Giảng viên: Hoàng Xuân Dương



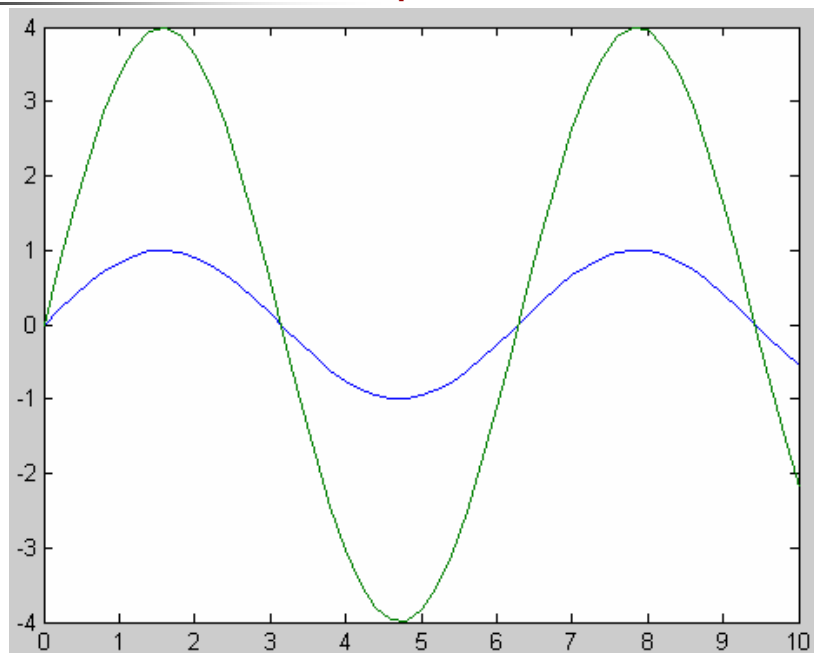
**Giảng viên: Hoàng Xuân Dương**



**Giảng viên: Hoàng Xuân Dương**







Giảng viên: Hoàng Xuân Dương

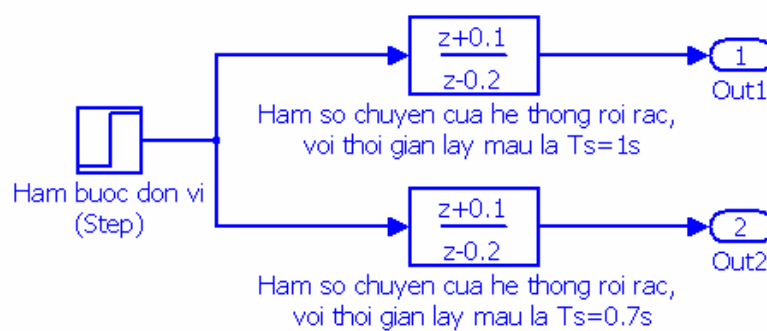


## II. MỘT SỐ HỆ THỐNG

### 3. Đáp ứng hệ thống

#### c. Hệ thống rời rạc

#### HE THONG ROI RAC



Giảng viên: Hoàng Xuân Dương



## II. MỘT SỐ HỆ THỐNG

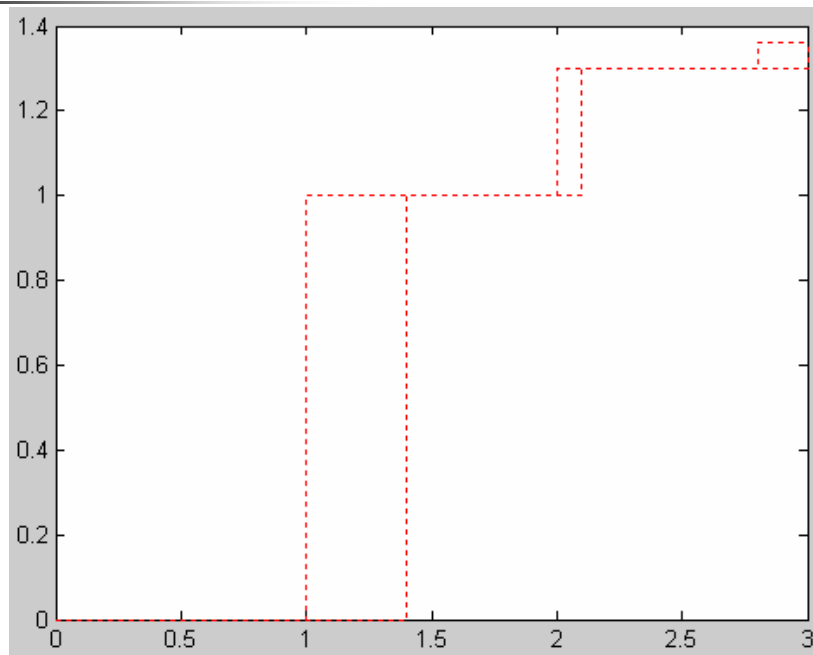
### 3. Đáp ứng hệ thống

#### c. Hệ thống rời rạc

Thực hiện với:

- Khối Discrete Transfer Fcn: **Simulink/ discrete**  
Vào properties định các tham số cho hàm truyền
  - Khối Step: **Simulink/ Source**  
Ở khối Fcn thứ 2, vào properties chọn thời gian lấy mẫu bằng 0.7
  - Lưu mô hình với tên **htrrac.mdl**
- ```
>> [t,x,y]=sim('htrrac',30);
>> stairs(t,y,'r')
```

Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

### 4. Khối to Workspace

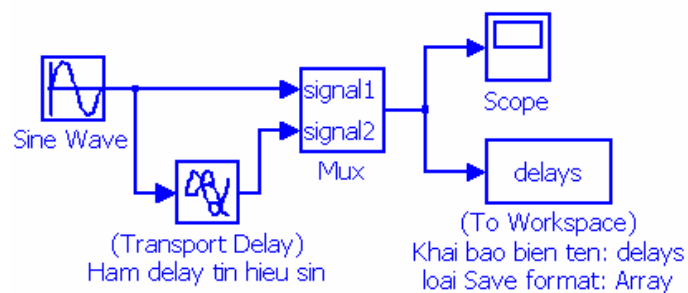
- Khối Workspace sẽ tự động trả về giá trị nằm trong biến được khai báo trong property mà không cần sử dụng hàm `sim()`
- Khi dùng khối này phải khai báo biến và chọn loại giá trị trả về

Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

### 4. Khối to Workspace (tt)

Ví dụ: Thiết kế hệ thống như hình vẽ



## SU DỤNG KHỐI TO WORKSPACE

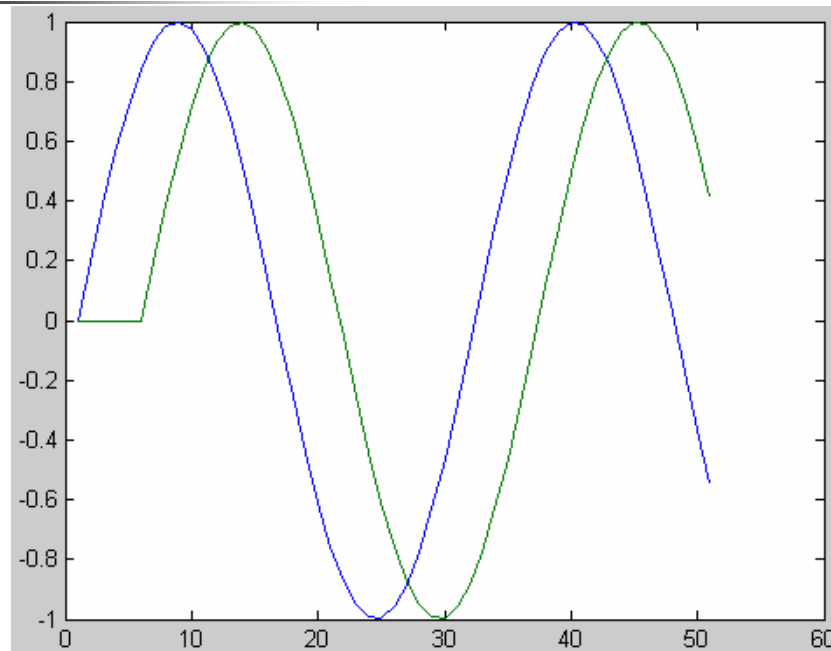
Giảng viên: Hoàng Xuân Dương

## II. MỘT SỐ HỆ THỐNG

### 4. Khởi tạo Workspace (tt)

- Khởi tạo Workspace: **Simulink/ Sinks**  
Khai báo biến delays và trị trả về là array
  - Khởi tạo Transport delay: **Simulink/ Continuous**
  - Lưu mô hình với tên **delay.mdl**
- ```
>> delays      % quan sát biến trả về
>> plot(delays)
```

Giảng viên: Hoàng Xuân Dương



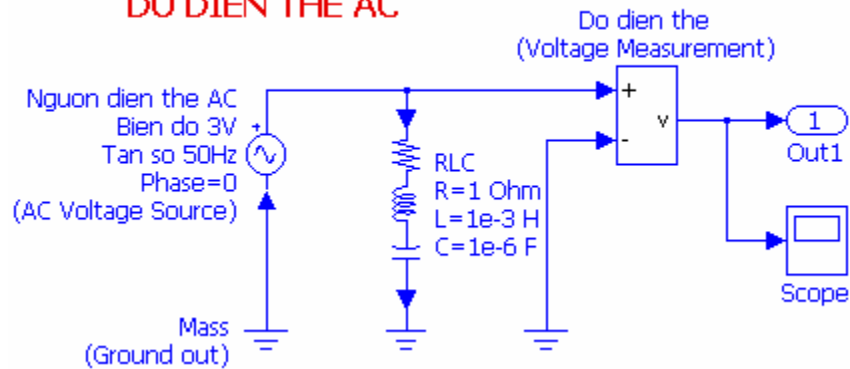
Giảng viên: Hoàng Xuân Dương

### III. MẠCH ĐIỆN

#### 1. Đo điện thế:

##### a. Khối Voltage Measurement:

#### DO DIEN THE AC



Giảng viên: Hoàng Xuân Dương

### III. MẠCH ĐIỆN

#### 1. Đo điện thế:

##### a. Khối Voltage Measurement (tt)

Thực hiện với:

- AC Voltage Source: **Simpowersystems/ Electrical Source**  
Khai báo 3 VAC, tần số 50Hz và pha=0
- Khối mass: **Simpowersystems/ Connectors**
- Voltage Measurement: **Simpowersystems/ Measurement**
- Khối Series RLC Branch: **Simpowersystems/ Elements**
- Chọn thời gian Stop time là 10s
- Chọn time range là 0.10s
- Lưu mô hình với tên **voltRLC.mdl**

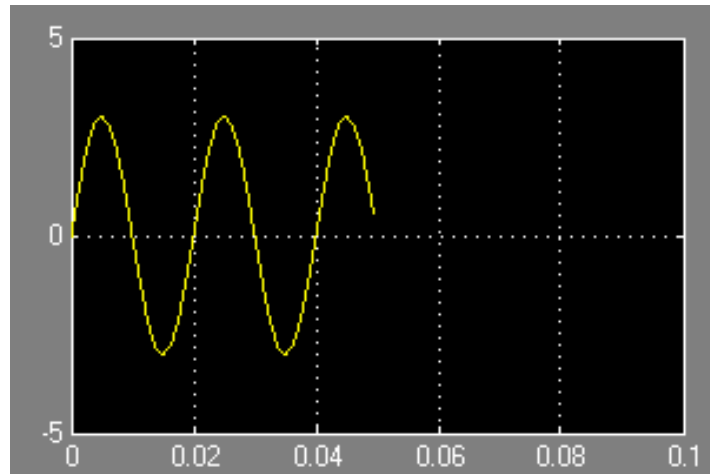
Giảng viên: Hoàng Xuân Dương



## CHƯƠNG 6: SIMULINK VÀ ỨNG DỤNG

395

Chọn **simulink** / **start** để chạy mô phỏng



Giảng viên: Hoàng Xuân Dương

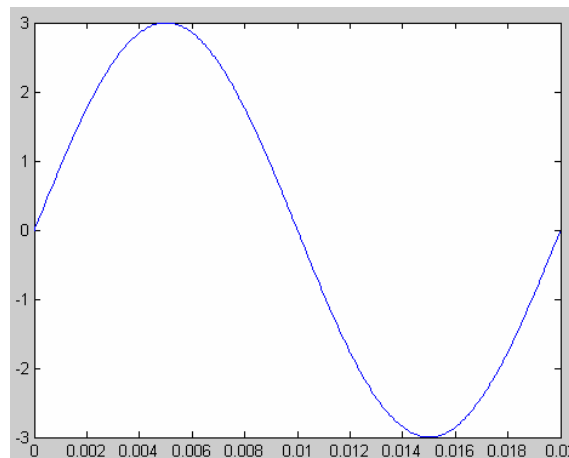


## CHƯƠNG 6: SIMULINK VÀ ỨNG DỤNG

396

Trong command window:

```
>> [t,x,y]=sim('voltRLC',0.02); plot(t,y)
```

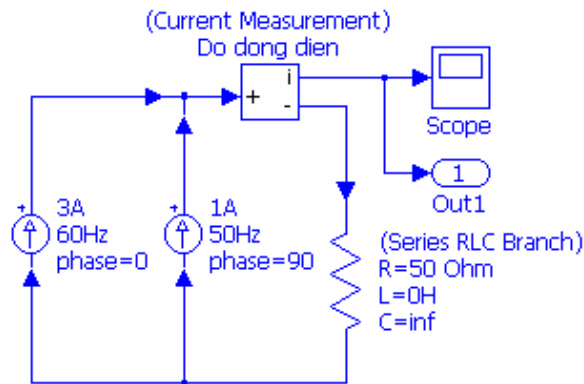


Giảng viên: Hoàng Xuân Dương



## III. MẠCH ĐIỆN

## 2. Đo dòng điện:



DO DÒNG ĐIỆN

Giảng viên: Hoàng Xuân Dương

## III. MẠCH ĐIỆN

## 2. Đo dòng điện:

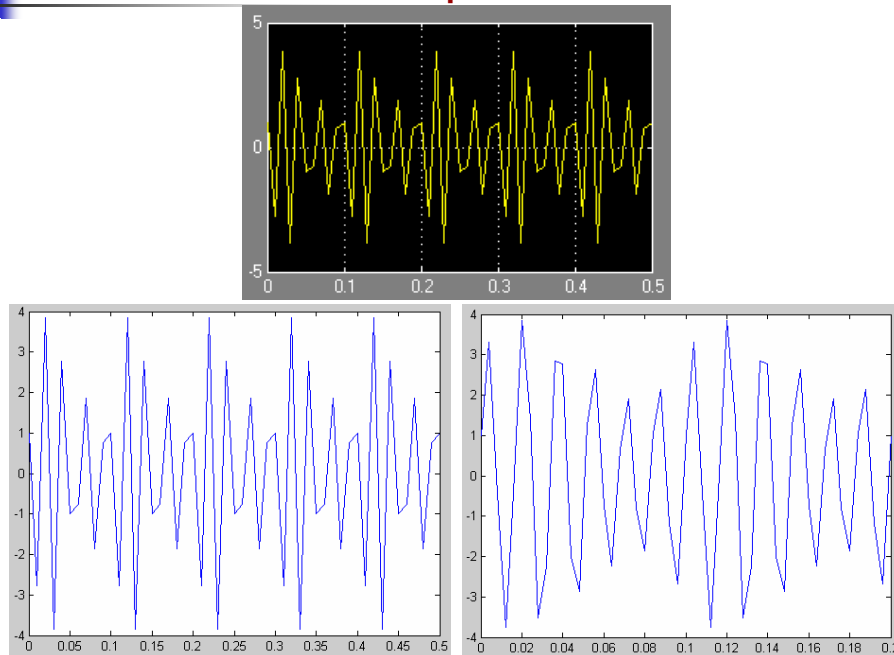
Thực hiện với:

- AC Current Source: [Simpowersystems/ Electrical Source](#)  
Khai báo dòng, tần số và pha.
- Khối T Connect: [Simpowersystems/ Connectors](#)
- Current Mesurement: [Simpowersystems/ Mesurement](#)
- Chọn thời gian Time range là 0.5s
- Lưu mô hình với tên [currnet.mdl](#)

```
>> [t,x,y]=sim('current'); plot(t,y)
```

```
>> [t,x,y]=sim('current',0.2); plot(t,y)
```

Giảng viên: Hoàng Xuân Dương



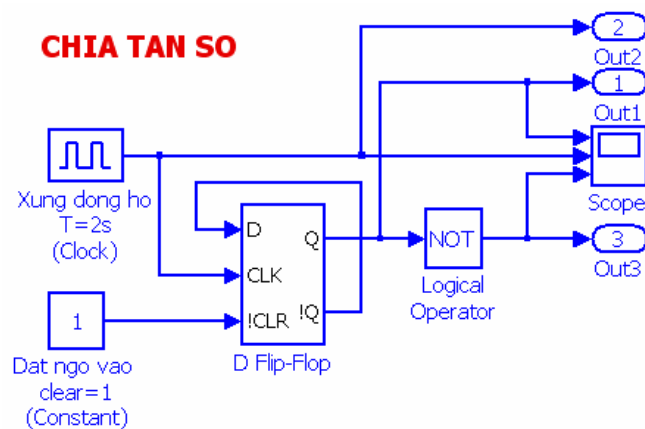
Giảng viên: Hoàng Xuân Dương



### III. MẠCH ĐIỆN

#### 3. Phần mạch số

##### CHIA TẦN SỐ



Giảng viên: Hoàng Xuân Dương







### III. MẠCH ĐIỆN

#### 3. Phân mạch số (tt)

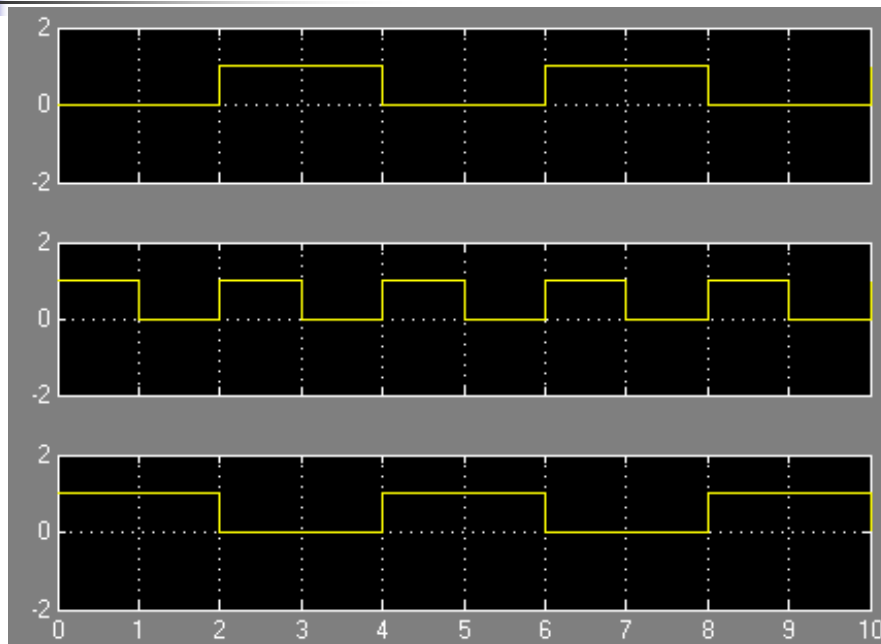
Thực hiện với:

- Khối Constant: Simulink/ Sources
- Khối Clock: Simulink Extras/ Flip Flops
- Khối D-FF: Simulink Extras/ Flip Flops
- Khối NOT: Simulink/ Math Operations/ Logic operation

Vào parameters chọn NOT

- Lưu mô hình với tên chiaf.mdl

Giảng viên: Hoàng Xuân Dương



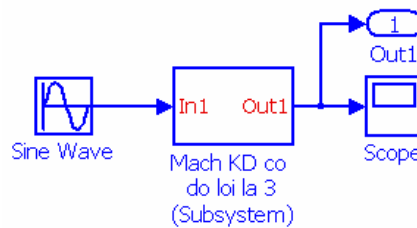
Giảng viên: Hoàng Xuân Dương



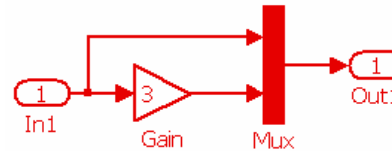
## IV. KHỐI SUBSYSTEM:

- Trong thiết kế hệ thống, để đơn giản người ta thường chia hệ thống ra từng phần nhỏ được gọi là subsystem

KHOI SUBSYSTEM



MACH KD CO DO LOI LA 3



Giảng viên: Hoàng Xuân Dương

## IV. KHỐI SUBSYSTEM:

- Có thể thực hiện theo 2 cách:

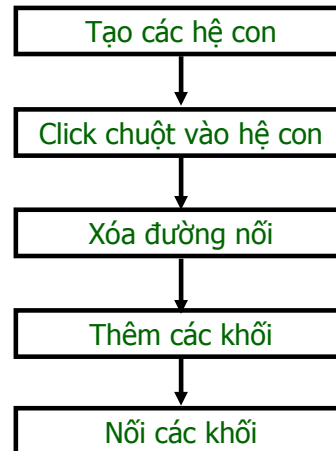
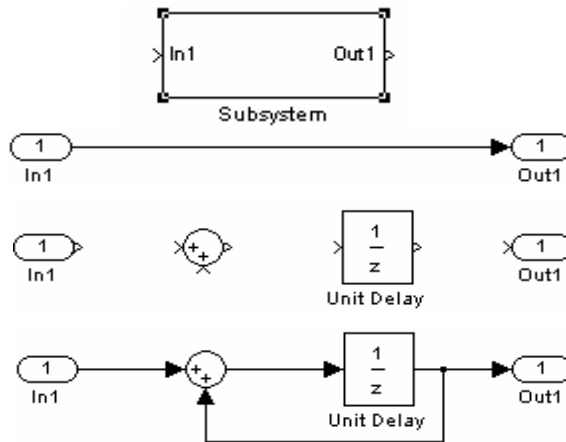
**1. Tạo hệ con trước:**

- Chọn **Ports & Subsystems**, kéo khối vào cửa sổ soạn thảo
- Tạo các khối liên kết bên trong
- Dùng khối **inport/outport** để biểu diễn tín hiệu vào ra của hệ con

Giảng viên: Hoàng Xuân Dương

## IV. KHỐI SUBSYSTEM:

## 1. Tạo hệ con trước (tt)

Ví dụ: Tạo khối  $1/(z-1)$ 

Giảng viên: Hoàng Xuân Dương

## IV. KHỐI SUBSYSTEM:

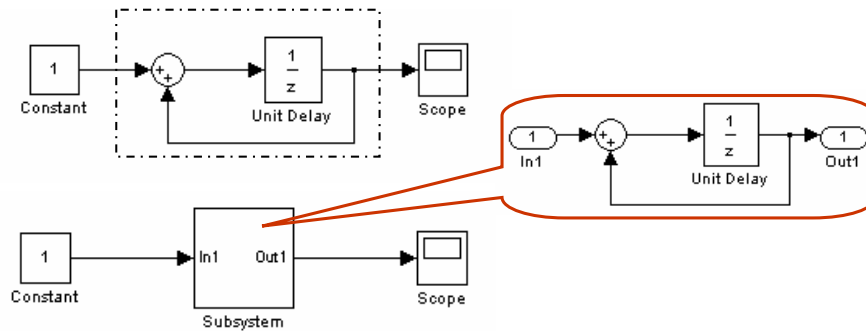
## 2. Tạo hệ con từ khối có sẵn:

- Dùng chuột đóng khung các khối và đường nối
- Chọn **Edit/Create Subsystem**
- Simulink** thay các khối đã chọn bằng một khối chung và mang một tên chung

Giảng viên: Hoàng Xuân Dương

## IV. KHỐI SUBSYSTEM:

## 2. Tạo hệ con từ khối có sẵn:



Giảng viên: Hoàng Xuân Dương

## IV. KHỐI SUBSYSTEM:

## 3. Tạo mặt nạ hệ con:

- Hệ con gồm nhiều khối có các thông số khác nhau, có thể dùng một mặt nạ chung cho các khối này, đại diện cho hệ con và các thông số cho khối

Ví dụ: Tạo một hệ con thực hiện hàm  $y=mx+b$

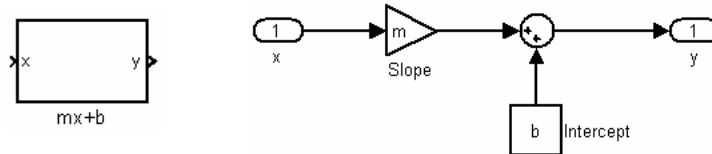
- $m$ ,  $b$  là các thông số phải đưa vào
- $x$  là tín hiệu vào
- $y$  là tín hiệu ra

Giảng viên: Hoàng Xuân Dương

#### IV. KHỐI SUBSYSTEM:

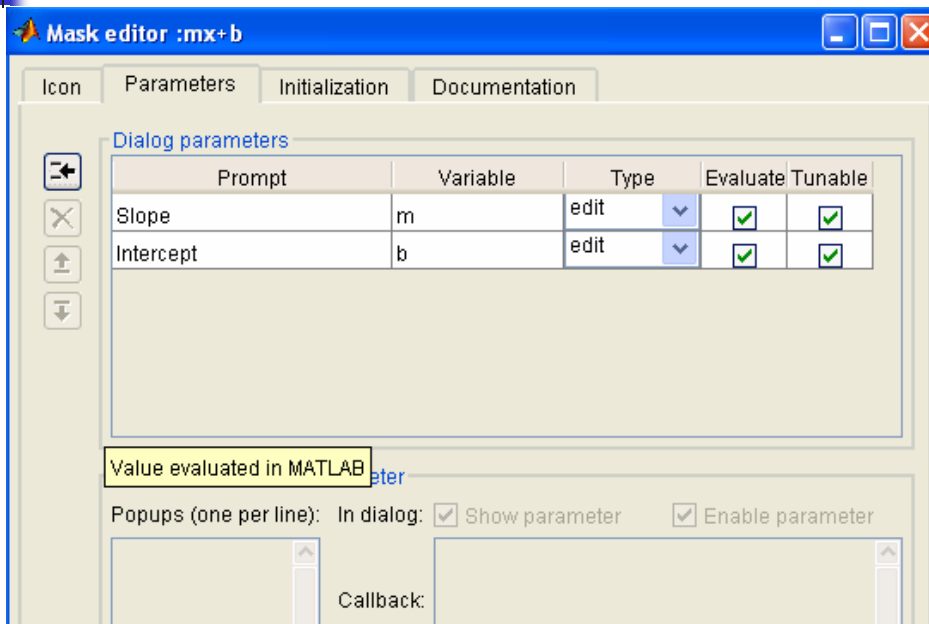
##### 3. Tạo mặt nạ hệ con:

- Tạo hệ con:



- Vào menu **Edit/Mask Subsystem**
  - Trong **Documentation**: Đặt tên mặt nạ ở **Mask type**, các chú thích trong **Mask Description**,...
  - Thêm các thông số, biến, kiểu loại... trong tab **Parameters**

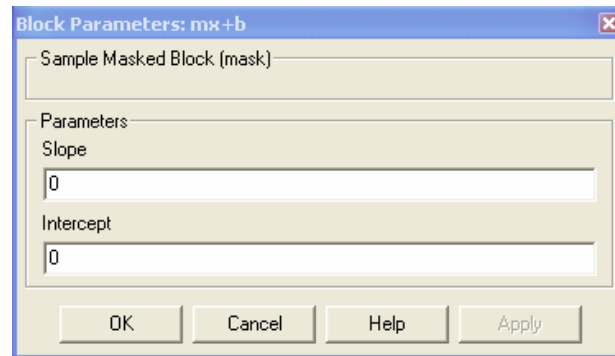
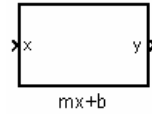
Giảng viên: Hoàng Xuân Dương



Giảng viên: Hoàng Xuân Dương

#### IV. KHỐI SUBSYSTEM:

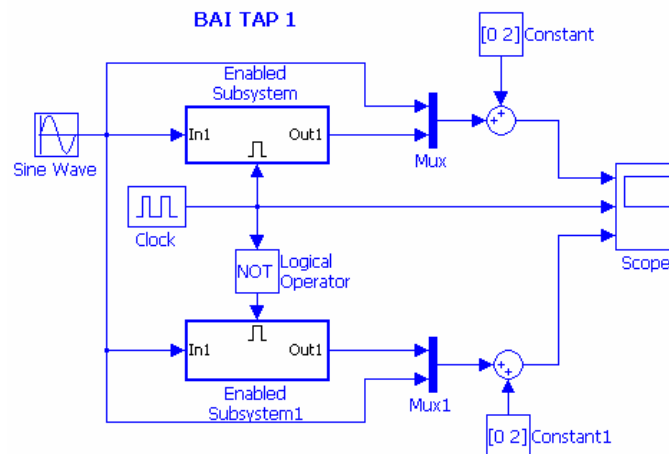
##### 3. Tạo mặt nạ hệ con:




Giảng viên: Hoàng Xuân Dương

#### V. BÀI TẬP:

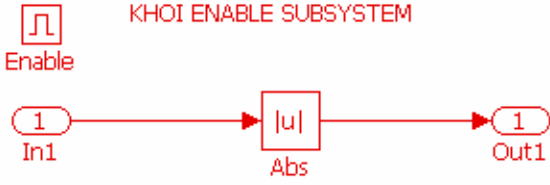
##### Bài tập 1:



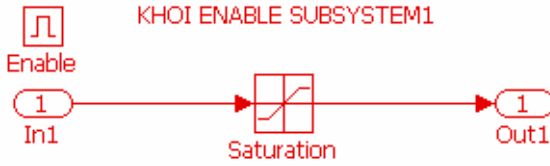
Giảng viên: Hoàng Xuân Dương


**CHƯƠNG 6: SIMULINK VÀ ỨNG DỤNG**


413




KHOI ENABLE SUBSYSTEM



KHOI ENABLE SUBSYSTEM1


**Giảng viên: Hoàng Xuân Dương**

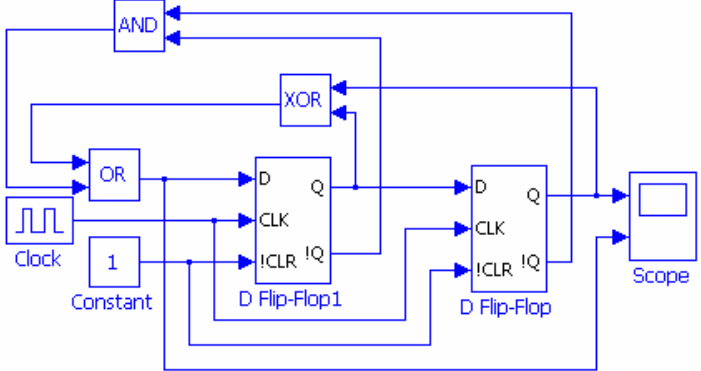

**CHƯƠNG 6: SIMULINK VÀ ỨNG DỤNG**


414

**V. BÀI TẬP:**

Bài tập 2:

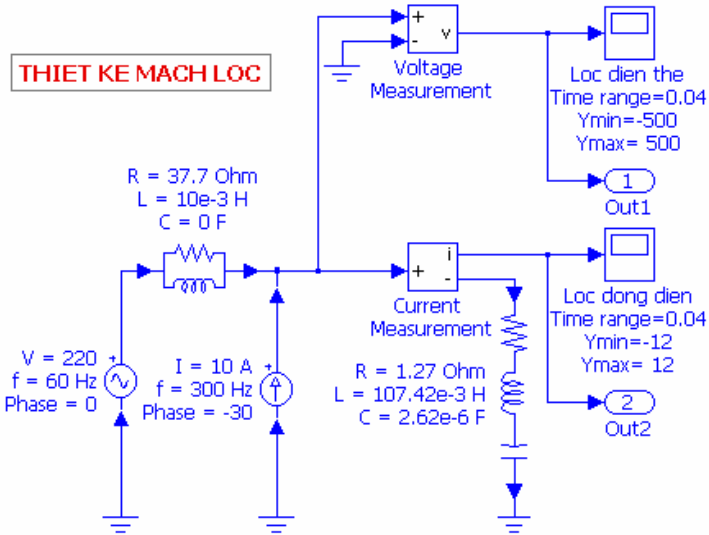
**THIẾT KẾ MẠCH LOGIC**




**Giảng viên: Hoàng Xuân Dương**

## V. BÀI TẬP:

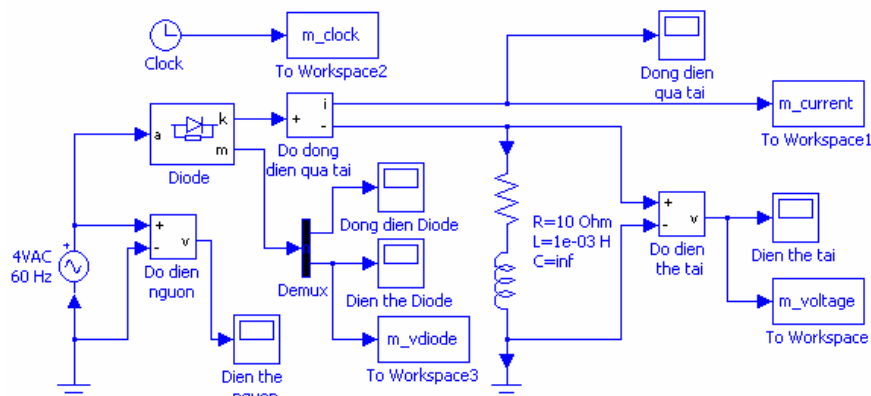
### Bài tập 3:



Giảng viên: Hoàng Xuân Dương

## V. BÀI TẬP:

### Bài tập 4:



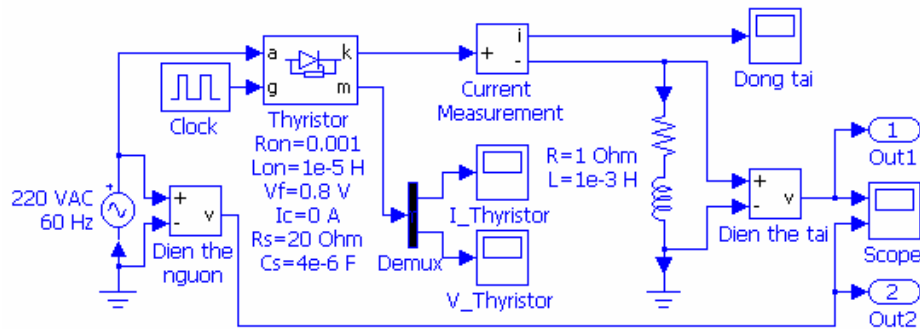
**MẠCH NẠM ĐIỆN 1 BẠN KỲ**  
(Stop time=0.06s)  
(time range=0.06s)

Giảng viên: Hoàng Xuân Dương



## V. BÀI TẬP:

### Bài tập 5:

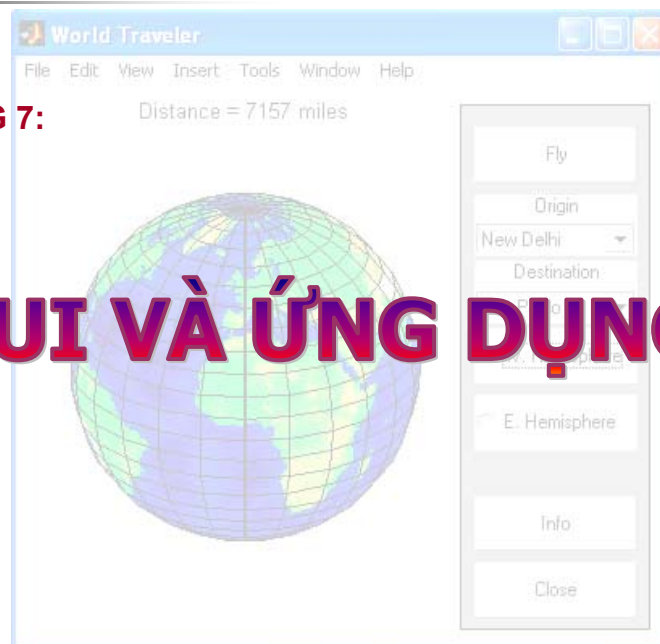


**KICH THYRISTOR**  
(time range=0.08s)  
(Ymin=-250; Ymax=250)

Giảng viên: Hoàng Xuân Dương

## CHƯƠNG 7:

# GUI VÀ ỨNG DỤNG



Giảng viên: Hoàng Xuân Dương



- I. GRAPHICAL USER INTERFACE
- II. TẠO MENU BẰNG GUI
- III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN
- IV. ĐO TẦN SỐ
- V. ĐỒ HOA 2D
- VI. CÁC HÀM VẼ 3D
- VII. BIẾN ĐIỀU ANALOG
- VIII. BIẾN ĐIỀU DIGITAL
- IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

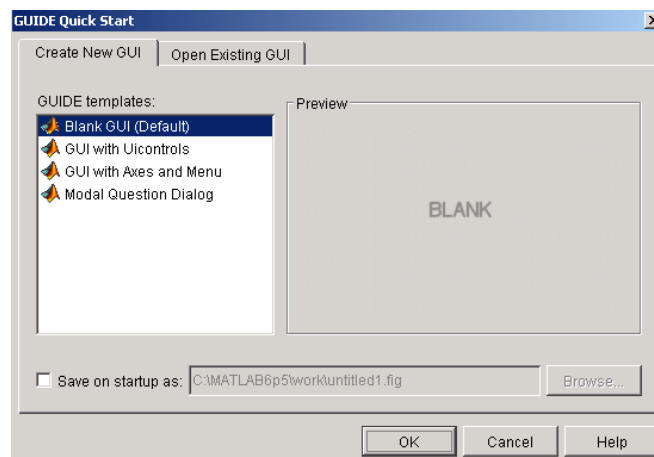
Giảng viên: Hoàng Xuân Dương



## I. GRAPHICAL USER INTERFACE

### 1. Giao diện GUI:

- Chọn biểu tượng **guide** trên thanh toolbar, hoặc thực hiện **guide** trên dòng lệnh → cửa sổ **GUIDE Quick Start**



Giảng viên: Hoàng Xuân Dương

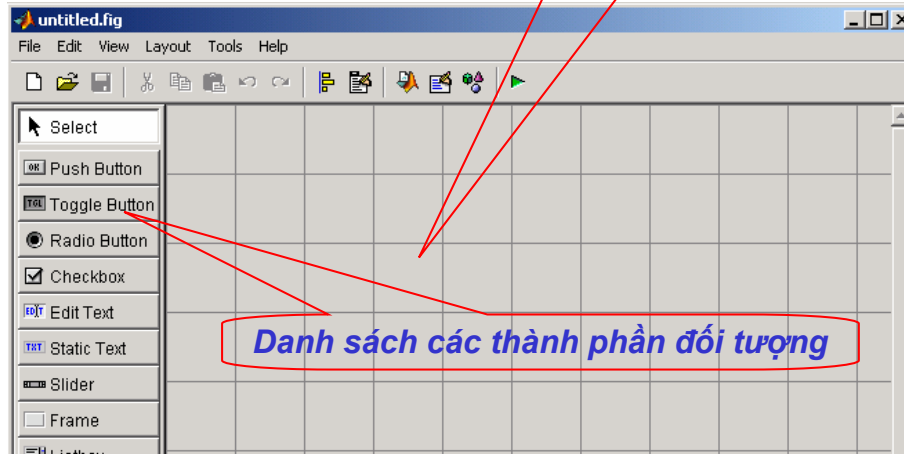


## I. GRAPHICAL USER INTERFACE

Vùng layout, nơi đặt các thành phần đối tượng

## 1. Giao diện GUI (tt)

- Có thể chọn các mẫu giao diện thiết kế sẵn hay bấm chọn GUI trống theo mặc định:



Giảng viên: Hoàng Xuân Dương

## I. GRAPHICAL USER INTERFACE

## 1. Giao diện GUI (tt)

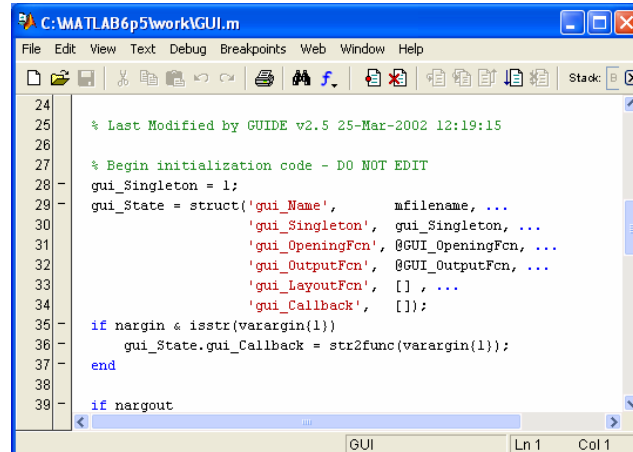
| Component     | Ý nghĩa                                                          |
|---------------|------------------------------------------------------------------|
| Axes          | Vẽ hệ trục                                                       |
| Check box     | Hộp kiểm tra, đưa vào các chọn lựa bằng chuột                    |
| Edit text     | Hộp đưa vào văn bản                                              |
| Frame         | Khung bao một vùng cửa sổ hình                                   |
| List box      | Bảng các mục để chọn lựa                                         |
| Pop-up menu   | Menu sổ xuống, chọn lựa bằng chuột                               |
| Push button   | Nút nhấn, kích hoạt một hành động                                |
| Radio button  | Giống check box nhưng chỉ chọn một                               |
| Slider        | Con trượt, đưa giá trị vào trong một tầm giới hạn                |
| Static text   | Dòng văn bản để đặt tiêu đề, nhãn, hướng dẫn                     |
| Toggle button | Như push button nhưng hiển thị trạng thái khác nhau mỗi khi nhấn |

Giảng viên: Hoàng Xuân Dương

## I. GRAPHICAL USER INTERFACE

## 1. Giao diện GUI (tt)

- Song song với việc tạo ra một giao diện **.fig** là một file **.m**. Nó chứa các nội dung có liên quan trực tiếp đến giao diện



```

24
25
26
27 % Last Modified by GUIDE v2.5 25-Mar-2002 12:19:15
28
29 % Begin initialization code - DO NOT EDIT
30
31 gui_Singleton = 1;
32 gui_State = struct('gui_Name',       mfilename, ...
33                   'gui_Singleton',   gui_Singleton, ...
34                   'gui_OpeningFcn', @GUI_OpeningFcn, ...
35                   'gui_OutputFcn',  @GUI_OutputFcn, ...
36                   'gui_LayoutFcn',  [], ...
37                   'gui_Callback',    []);
38
39 if nargin & isstr(varargin{1})
40     gui_State.gui_Callback = str2func(varargin{1});
41 end
42
43 if nargout
44     [h1, h2, ..., hN] = gui_State.gui_Callback(hObject, varargin{1:nargout});
45 end
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Giảng viên: Hoàng Xuân Dương

## I. GRAPHICAL USER INTERFACE

## 1. Giao diện GUI (tt)

- Trong đó có một số biến thông dụng:
  - **varargout**: chỉ chung các đối số trả về
  - **varargin**: Chỉ chung các đối số đưa vào hàm
  - **nargin**: Số lượng các đối số đưa vào
  - **nargout**: Số lượng các đối số trả về
  - **handles**: Cấu trúc handle của mọi component trong figure

Ví dụ:

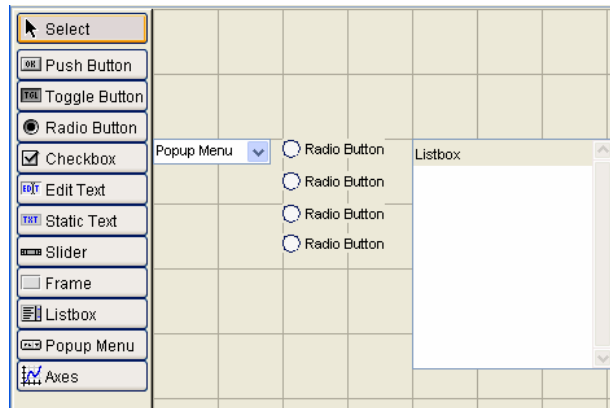
- nút nhấn có tag **pushbutton1** có handle là **handles.pushbutton1**
- figure có tag **figure1** có handle là **handles.figure1**

Giảng viên: Hoàng Xuân Dương

## I. GRAPHICAL USER INTERFACE

## 1. Giao diện GUI (tt)

- Bấm chuột trái vào các **component** muốn tạo, kéo chuột ra vùng **layout**, nơi muốn đặt **component**. Có thể dùng chuột để thay đổi kích thước **component**

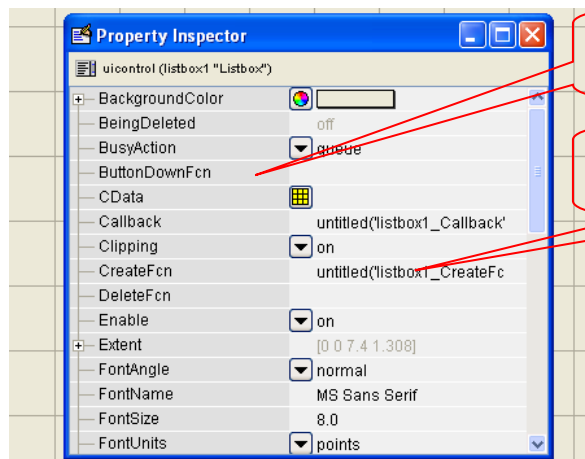


Giảng viên: Hoàng Xuân Dương

## I. GRAPHICAL USER INTERFACE

## 2. Soạn thảo các thuộc tính

- Double click tại **component** để mở **Inspect Properties** → định các thông số cho các **component**



Vùng các thuộc tính  
cho component

Vùng giá trị của các  
thuộc tính

Giảng viên: Hoàng Xuân Dương



## I. GRAPHICAL USER INTERFACE

## 2. Soạn thảo các thuộc tính (tt)

| Thuộc tính              | Ý nghĩa                                  |
|-------------------------|------------------------------------------|
| Tag                     | Tên gán cho component                    |
| BackgroundColor         | Màu nền của component                    |
| ForegroundColor         | Màu chữ trên component                   |
| FontName, FontAngle,... | Các đặc tính của font chữ trên component |
| String                  | Văn bản hiển thị trên component          |
| Enable                  | Cho phép component hoạt động hay không   |
| Visible                 | Hiển thị component hay không             |
| <b>Nếu là layout</b>    |                                          |
| Color                   | Màu nền cửa sổ                           |
| Name                    | Tên cửa sổ                               |
| Resize                  | Điều chỉnh kích thước cửa sổ             |

Giảng viên: Hoàng Xuân Dương



## I. GRAPHICAL USER INTERFACE

## 3. Các Callback:

- Quan trọng nhất đối với các **component** là **callback**, là các hàm con (**function**) mà file **.m** sẽ gọi khi tác động vào **component**
- Mỗi khi thêm vào một **component**, Matlab đều thêm vào file **.m** một hàm **callback** tương ứng (trừ **frame**, **static text**, **axes**)
- Hầu hết nội dung các **callback** được người sử dụng viết

Ví dụ: Xem nội dung file **.m** của một **figure** với một nút nhấn (**pushbutton**) như sau

Giảng viên: Hoàng Xuân Dương





## I. GRAPHICAL USER INTERFACE

```

75
76 % --- Executes on button press in pushbutton1.
77 function pushbutton1_Callback(hObject, eventdata, handles)
78 % hObject    handle to pushbutton1 (see GCBO)
79 % eventdata  reserved - to be defined in a future version of MATLAB
80 % handles    structure with handles and user data (see GUIDATA)
81

```

Trong đó:

- **pushbutton1\_Callback**: hàm được gọi khi nhấn vào **pushbutton1**
- **hObject**: handle của riêng nút nhấn
- **handles**: Chứa tất cả các handle có trong file **.m**
- **eventdata**: Tham số gọi hàm

Giảng viên: Hoàng Xuân Dương



## I. GRAPHICAL USER INTERFACE

### 3. Các Callback (tt)

- Các hàm Callback chung cho mọi component

| Callback              | Ý nghĩa                                               |
|-----------------------|-------------------------------------------------------|
| ButtonDownFcn         | Callback được gọi khi click chuột trên đối tượng      |
| CreateFcn             | Callback được tạo khi thiết lập đối tượng             |
| DeleteFcn             | Callback được gọi trước khi hủy bỏ đối tượng          |
| <b>Nếu là Figure</b>  |                                                       |
| CloseRequestFcn       | Được gọi khi đóng bởi lệnh close hay quit Matlab      |
| KeyPressFcn           | Nhấn phím khi con chạy trong cửa sổ                   |
| ResizeFcn             | Khi resize của sổ figure                              |
| WindowButtonDownFcn   | Click chuột trên cửa sổ figure (không trên đối tượng) |
| WindowButtonMotionFcn | Khi di chuyển chuột trong cửa sổ figure               |
| WindowButtonUpFcn     | Nhả chuột sau khi đã bấm                              |

Giảng viên: Hoàng Xuân Dương





## **II. TẠO MENU BẰNG GUI**

### **Nội dung:**

- Cách tạo menu bằng GUI
- Tạo phím nóng cho menu
- Cách tạo trục vẽ
- Cách vẽ hình trong GUI
- Cách đổi View
- Sử dụng biến toàn cục global
- Hàm zoom fill
- Hàm zoom out
- Hàm CreateFcn
- Hàm CloseRequestFcn của đối tượng
- Tạo Contextmenu

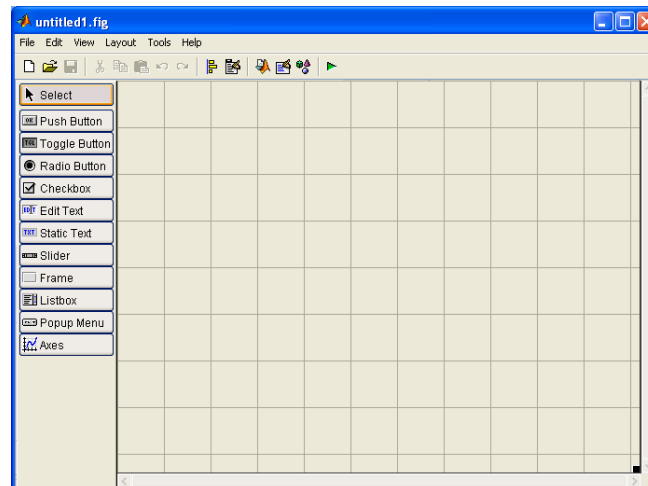
**Giảng viên: Hoàng Xuân Dương**



## **II. TẠO MENU BẰNG GUI**

### **1. Tạo GUI:**

- Tạo một blank GUI



**Giảng viên: Hoàng Xuân Dương**





**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**

**II. TẠO MENU BẰNG GUI**

**1. Tạo GUI (tt)**

- Trong **Inspect Properties**:
  - **Name**: Tao menu bang GUI
  - **Filename**: GUI\_1
  - **Position**:
    - x=1;
    - y=1;
    - width=130;
    - height=30;
  - **Resize**: off
  - **Handle Visibility**: on

**Property Inspector**

**Giảng viên: Hoàng Xuân Dương**

**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**

**II. TẠO MENU BẰNG GUI**

**2. Tạo menu**

- Chọn **Tools-Menu Editor**
- Chọn **New Menu**
  - Thay **Untitled\_1** bằng **Ve\_hinh**

**Tools Help**

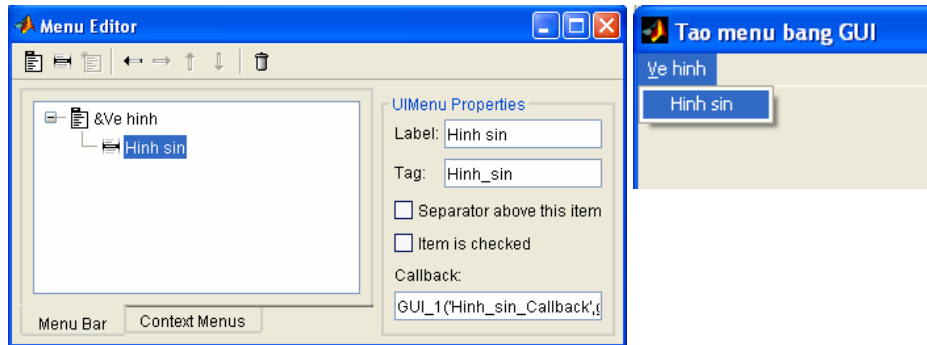
**Menu Editor**

**Giảng viên: Hoàng Xuân Dương**

## II. TẠO MENU BẰNG GUI

### 2. Tạo menu (tt)

- Chọn **New Menu Item** để tạo menu con
- Thay **Untitled\_2** bằng **Hinh\_sin**

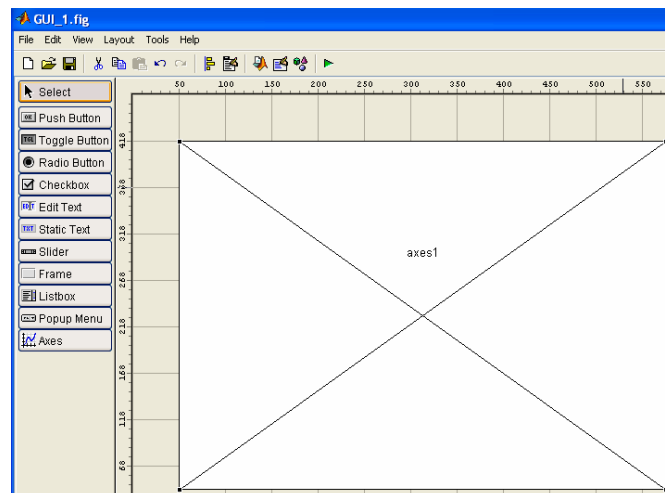


Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

### 3. Tạo trục để vẽ hình

- Chọn biểu tượng **Axes** để tạo trục vẽ

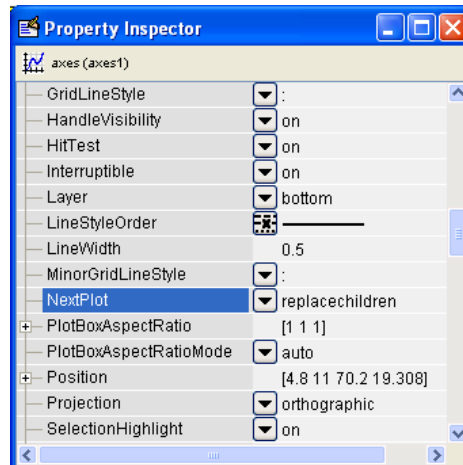


Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

## 3. Tạo trục để vẽ hình (tt)

- Trong **Inspect Properties**
  - Nextplot: replacechildren
  - Màu nền: tùy ý
  - Xcolor: tùy ý
  - Xgrid: on
  - Ycolor: tùy ý
  - Ygrid: on
  - Zcolor: tùy ý
  - Zgrid: on

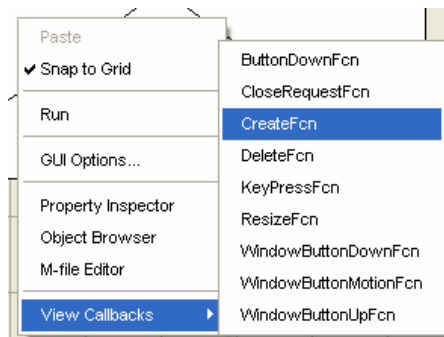


Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

## 4. Tạo hàm CreateFcn và CloseRequestFcn:

- Right click trên giao diện chính, chọn lần lượt
  - View Callbacks-CreateFcn
  - View Callbacks-CloseRequestFcn



- Trong tập tin **GUI\_1.m** có thêm 2 hàm mới

Giảng viên: Hoàng Xuân Dương

```

200
201 % --- Executes during object creation, after setting all properties.
202 function figure1_CreateFcn(hObject, eventdata, handles)
203 % hObject    handle to figure1 (see GCBO)
204 % eventdata  reserved - to be defined in a future version of MATLAB
205 % handles    empty - handles not created until after all CreateFcns call
206 |
207 % --- Executes when user attempts to close figure1.
208 function figure1_CloseRequestFcn(hObject, eventdata, handles)
209 % hObject    handle to figure1 (see GCBO)
210 % eventdata  reserved - to be defined in a future version of MATLAB
211 % handles    structure with handles and user data (see GUIDATA)
212
213 % Hint: delete(hObject) closes the figure
214 delete(hObject);
215
216 % -----

```

Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

### 4. Tạo hàm CreateFcn và CloseRequestFcn (tt)

- Hàm `figure1_CreateFcn`:
  - Sẽ thi hành đầu tiên khi chạy ứng dụng `GUI_1.m`
  - Thường để khai báo biến toàn cục `global` hay vẽ `demo` ban đầu
- Hàm `figure1_CloseRequestFcn`:
  - Để phòng người sử dụng thoát ngang ứng dụng

```

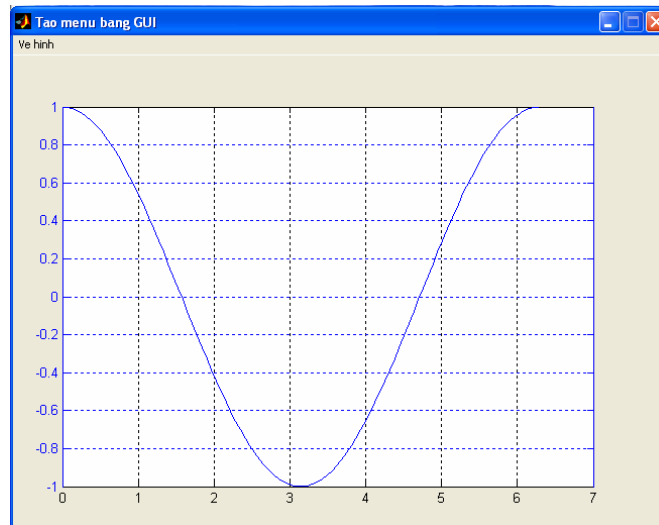
% --- Executes during object creation, after setting all properties.
function figure1_CreateFcn(hObject, eventdata, handles)
% .....
global hsin
x=linspace(0,2*pi);
y=cos(x);
hsin=plot(x,y);
% --- Executes when user attempts to close figure1.
function figure1_CloseRequestFcn(hObject, eventdata, handles)
% .....
delete(hObject);

```

Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

## 4. Tạo hàm CreateFcn và CloseRequestFcn (tt)



Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

## 5. Tạo hàm kích hoạt:

- Hàm tự thêm vào tập tin **GUI\_1.m** để kích hoạt menu **Hình\_sin**

```
% -----
function Hình_sin_Callback(hObject, eventdata, handles)
global hsin
x=linspace(0,2*pi);
y=sin(x);
hsin=plot(x,y);
title('Ham sin(x)')
% -----
```

} Đoạn thêm vào

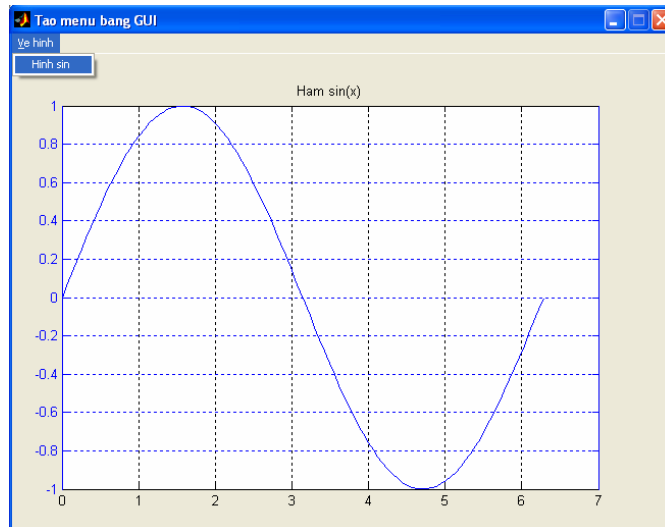
- Chạy tập tin **GUI\_1.m**, vào menu **Hình\_sin** để xem kết quả

Giảng viên: Hoàng Xuân Dương



## II. TẠO MENU BẰNG GUI

### 5. Tạo hàm kích hoạt (tt)

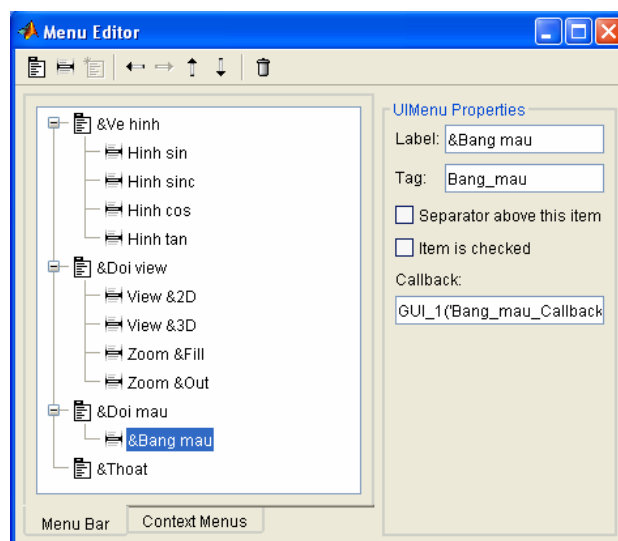


Giảng viên: Hoàng Xuân Dương



## II. TẠO MENU BẰNG GUI

### 6. Tạo các menu khác :



Giảng viên: Hoàng Xuân Dương



## II. TẠO MENU BẰNG GUI

## 6. Tạo các menu khác (tt)

- Kích hoạt các menu mới thêm vào:

```
% -----  
function Hinh_sin_Callback(hObject, eventdata, handles)  
global hsin  
x=linspace(0,2*pi);  
y=sin(x);  
hsin=plot(x,y);  
title('Ham sin(x)')  
% -----  
function Hinh_sinc_Callback(hObject, eventdata, handles)  
global hsin  
x=linspace(-2*pi,2*pi);  
y=sinc(x);  
hsin=plot(x,y);  
title('Ham sinc(x)')
```

Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

```
% -----  
function Hinh_cos_Callback(hObject, eventdata, handles)  
global hsin  
x=linspace(0,2*pi);  
y=cos(x);  
hsin=plot(x,y);  
title('Ham cos(x)')  
% -----  
function Hinh_tan_Callback(hObject, eventdata, handles)  
global hsin  
x=linspace(-2*pi,2*pi);  
y=tan(x);  
hsin=plot(x,y);  
title('Ham tan(x)')  
% -----  
function View_2D_Callback(hObject, eventdata, handles)  
View(2)
```

Giảng viên: Hoàng Xuân Dương

**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**  
**II. TẠO MENU BẰNG GUI**

447

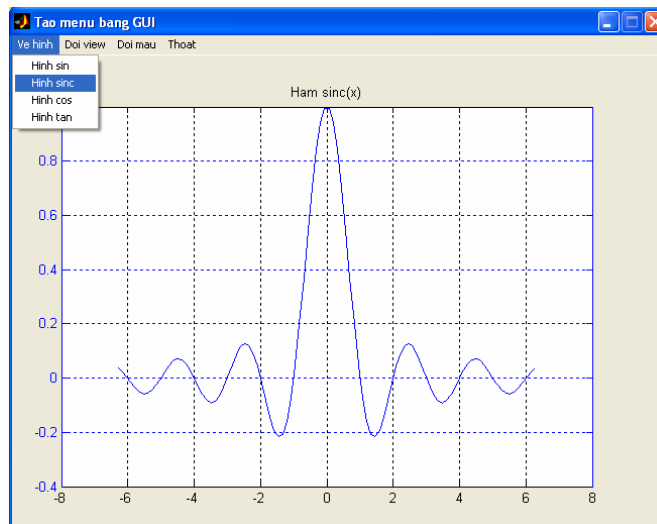
```
% -----
function View_3D_Callback(hObject, eventdata, handles)
View(3)
% -----
function Zoom_fill_Callback(hObject, eventdata, handles)
zoom fill
% -----
function Zoom_out_Callback(hObject, eventdata, handles)
ax=get(handles.figure1,'CurrentAxes');
set(ax,'CameraViewAngleMode','auto');
% -----
function Bang_mau_Callback(hObject, eventdata, handles)
global hsin
uisetcolor(hsin,'Bang mau Windows')
% -----
function Thoat_Callback(hObject, eventdata, handles)
closereq
```

**Giảng viên: Hoàng Xuân Dương**

**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**  
**II. TẠO MENU BẰNG GUI**

448

**6. Tạo các menu khác (tt)**



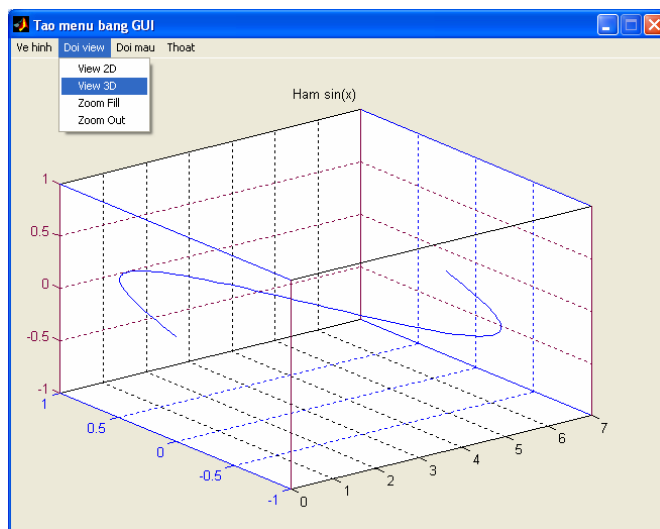
**Giảng viên: Hoàng Xuân Dương**





## II. TẠO MENU BẰNG GUI

### 6. Tạo các menu khác (tt)

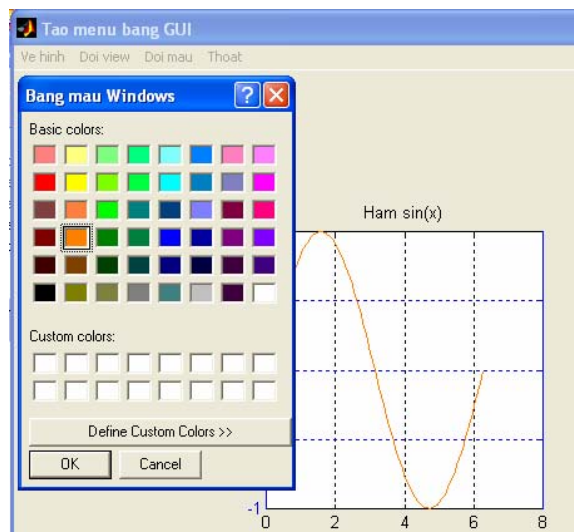


Giảng viên: Hoàng Xuân Dương



## II. TẠO MENU BẰNG GUI

### 6. Tạo các menu khác (tt)



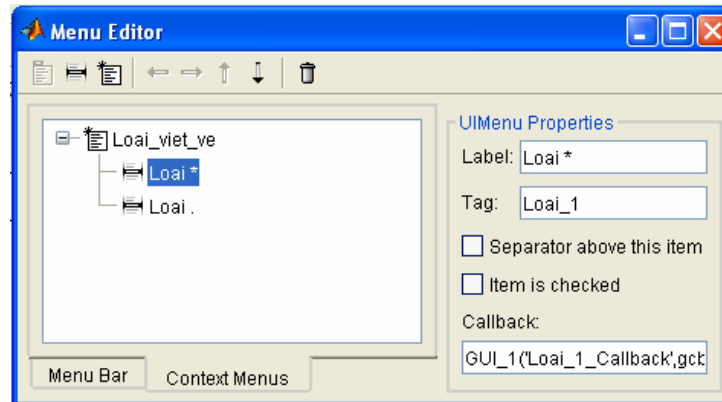
Giảng viên: Hoàng Xuân Dương



## II. TẠO MENU BẰNG GUI

## 7. Tạo Contextmenu:

- Menu hiển thị khi **right click** trên giao diện chính
- Mở **Menu Editor-Context Menus**
- Tạo các **contextmenu** như hình vẽ:



Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

## 7. Tạo Contextmenu (tt)

- Viết các hàm kích hoạt trong **GUI\_1.m**

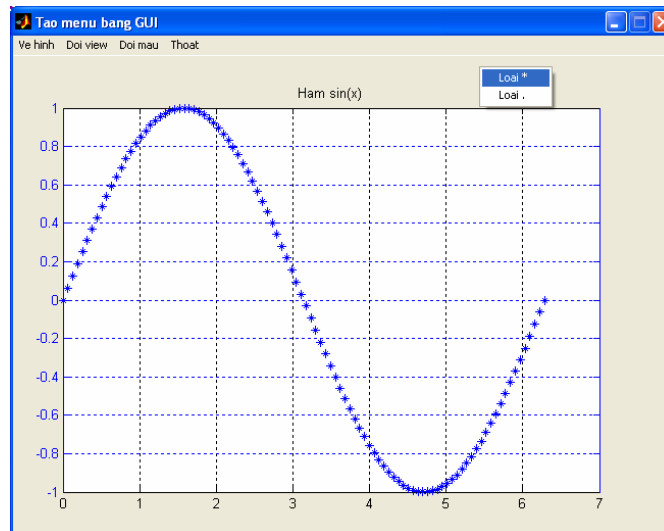
```
% -----
function Loai_1_Callback(hObject, eventdata, handles)
global hsin
set(hsin,'LineStyle','*')
% -----
function Loai_2_Callback(hObject, eventdata, handles)
global hsin
set(hsin,'LineStyle','.')
% -----
```

- Vào **Property Inspector-UIContextMenu** chọn **Loai\_viet\_ve**

Giảng viên: Hoàng Xuân Dương

## II. TẠO MENU BẰNG GUI

### 7. Tạo Contextmenu (tt)



Giảng viên: Hoàng Xuân Dương

## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

### Nội dung:

- Tạo giao diện để giải bài toán đổi nhiệt độ
- Tạo và cách kích hoạt Edit
- Tạo và cách kích hoạt Slider
- Tạo và cách kích hoạt RadioButton
- Tạo và vẽ nhiều hình trên cùng một trục Axes

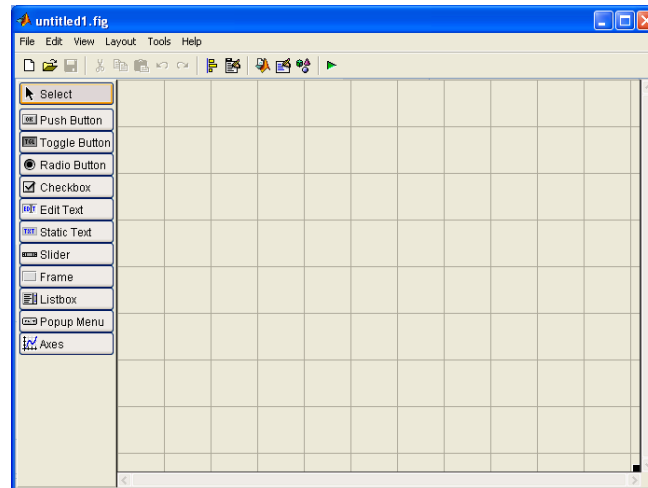
Giảng viên: Hoàng Xuân Dương



### III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

#### 1. Tạo GUI:

- Tạo một blank GUI



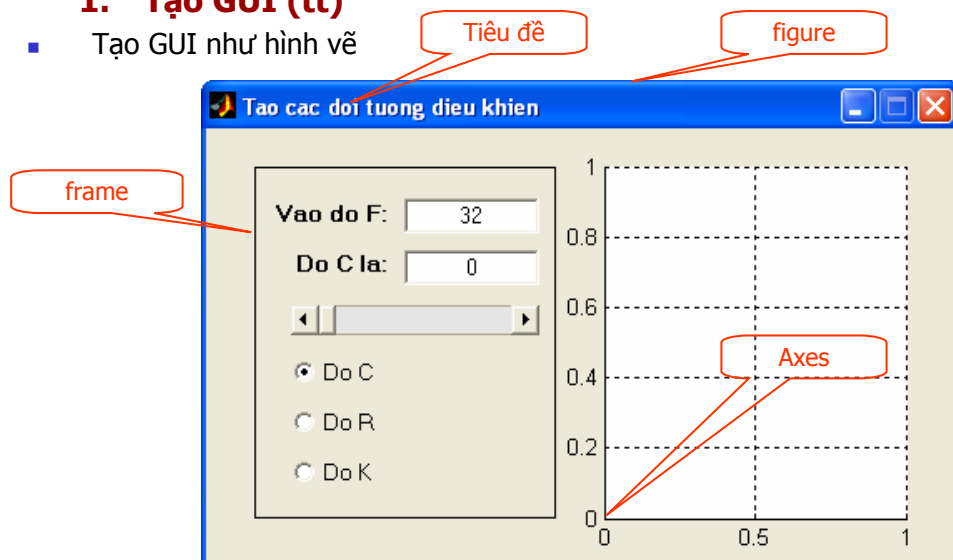
Giảng viên: Hoàng Xuân Dương



### III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

#### 1. Tạo GUI (tt)

- Tạo GUI như hình vẽ



Giảng viên: Hoàng Xuân Dương





## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

## 1. Tạo GUI (tt)

- Menu-Inspect Properties → Định thuộc tính các đối tượng

| Figure            |                   |                              |
|-------------------|-------------------|------------------------------|
| Màu nền giao diện | Color             | Tùy ý                        |
| Tên tập tin .m    | Filename          | GUI_2                        |
| Tên Tiêu đề       | Name              | Tạo các đối tượng điều khiển |
| Độ lớn giao diện  | Position          | [10 7 90 20]                 |
| Chọn trục vẽ      | Handle Visibility | on                           |

| Frame         |                 |        |
|---------------|-----------------|--------|
| Màu nền       | BackgroundColor | Tùy ý  |
| Tên của frame | Tag             | frame1 |

Giảng viên: Hoàng Xuân Dương



## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

## 1. Tạo GUI (tt)

| Static Text (số lượng 2) |          |           |          |
|--------------------------|----------|-----------|----------|
| STT                      | Fontsize | String    | Tag      |
| 1                        | 12       | Vào do F: | text_DoF |
| 2                        | 12       | Do C là:  | text_DoC |

| Edit (số lượng 2) |          |        |          |
|-------------------|----------|--------|----------|
| STT               | Fontsize | String | Tag      |
| 1                 | 12       | 32     | edit_DoF |
| 2                 | 12       | 0      | edit_DoC |

| Slider (số lượng: 1) |     |     |            |       |
|----------------------|-----|-----|------------|-------|
| STT                  | Max | Min | SliderStep | Value |
| 1                    | 100 | 0   | [0.01 0.1] | 32    |

Giảng viên: Hoàng Xuân Dương





## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

## 1. Tạo GUI (tt)

| Axes (số lượng: 1) |                 |       |       |       |
|--------------------|-----------------|-------|-------|-------|
| STT                | NextPlot        | XGrid | YGrid | ZGrid |
| 1                  | replacechildren | on    | on    | on    |

| RadioButton (số lượng: 3) |        |       |                 |
|---------------------------|--------|-------|-----------------|
| STT                       | String | Value | Tag             |
| 1                         | Do C   | [1.0] | radiobutton_DoC |
| 2                         | Do R   | [0.0] | radiobutton_DoR |
| 3                         | Do K   | [0.0] | radiobutton_DoK |

Giảng viên: Hoàng Xuân Dương



## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

## 2. Viết hàm kích hoạt các đối tượng:

- Thêm vào nội dung GUI\_2.m

```
% Chương trình kích hoạt edit_DoF để lần lượt đổi nhiệt độ
function edit_DoF_Callback(hObject, eventdata, handles)
F=get(handles.edit_DoF,'string');
F=eval(F);
doC=get(handles.radiobutton_DoC,'value');
doR=get(handles.radiobutton_DoR,'value');
doK=get(handles.radiobutton_DoK,'value');
if (doC)
    kq=(F-32)*(5/9);
elseif (doK)
    C=(F-32)*(5/9);
    kq=C+273.15;
elseif (doR)
    kq=F+459.7;
end
set(handles.edit_DoC,'string',num2str(kq))
```

Giảng viên: Hoàng Xuân Dương



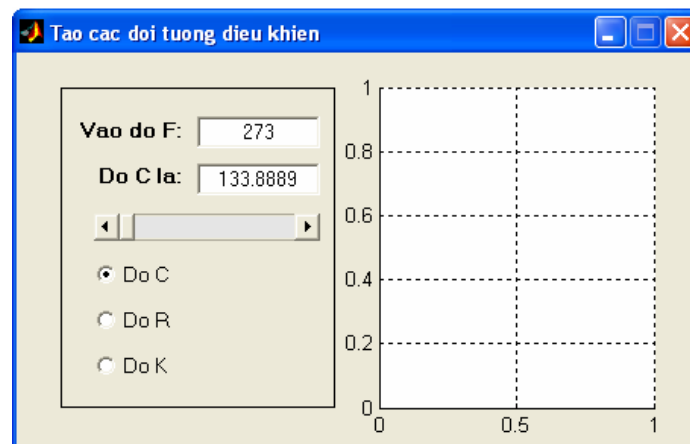
```
% Chương trình cho phép chọn một RadioButton duy nhất
% --- Executes on button press in radiobutton_DoC.
function radiobutton_DoC_Callback(hObject, eventdata, handles)
set(handles.radiobutton_DoC,'value',1);
set(handles.radiobutton_DoR,'value',0);
set(handles.radiobutton_DoK,'value',0);
% Gọi đến hàm kích hoạt edit_DoF
edit_DoF_Callback(hObject, eventdata, handles)
% --- Executes on button press in radiobutton_DoR.
function radiobutton_DoR_Callback(hObject, eventdata, handles)
set(handles.radiobutton_DoC,'value',0);
set(handles.radiobutton_DoR,'value',1);
set(handles.radiobutton_DoK,'value',0);
edit_DoF_Callback(hObject, eventdata, handles)
% --- Executes on button press in radiobutton_DoK.
function radiobutton_DoK_Callback(hObject, eventdata, handles)
set(handles.radiobutton_DoC,'value',0);
set(handles.radiobutton_DoR,'value',0);
set(handles.radiobutton_DoK,'value',1);
edit_DoF_Callback(hObject, eventdata, handles)
```

Giảng viên: Hoàng Xuân Dương

### III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

#### 2. Viết hàm kích hoạt các đối tượng (tt)

- Chạy tập tin GUI\_2.m
- Nhập giá trị vào → lần lượt chọn các RadioButton



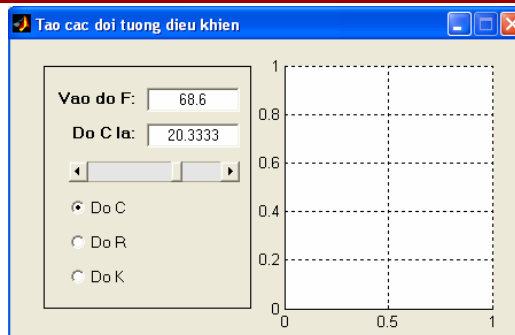
Giảng viên: Hoàng Xuân Dương

## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

## 2. Viết hàm kích hoạt các đối tượng (tt)

- Viết hàm kích hoạt cho slider

```
% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
value=get(handles.slider1,'value');
set(handles.edit_DoF,'string',num2str(value));
edit_DoF_Callback(hObject, eventdata, handles);
```



Giảng viên: Hoàng Xuân Dương

## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

## 3. Vẽ nhiều hình trên cùng một trục:

- Viết hàm thêm vào để vẽ hình:

```
function vehinh_Callback(hObject, eventdata, handles)
fahr=get(handles.edit_DoF,'string');
fahr=str2num(fahr);
f=[fahr-50,fahr,fahr+50];
t=f+459.7;
t(2,1:3)=(f-32)*5/9;
t(3,1:3)=t(1,1:3)+273.15;
names(1,:)= 'Do rankine';
names(2,:)= 'Do C';
names(3,:)= 'Do Kenvil';
hh=plot(f,t(1,1:3),'-',f(2),t(1,2),'*',f,t(2,1:3),'-',f(2),t(2,2),'+',...
f,t(3,1:3),'-',f(2),t(3,2),'>');
text(f(1),t(1,3),names(1,:), 'fontname','SVnHelvetica','fontsize',10,'color','r');
text(f(1),t(2,3),names(2,:), 'fontname','SVnHelvetica','fontsize',10,'color','b');
text(f(1),t(3,3),names(3,:), 'fontname','SVnHelvetica','fontsize',10,'color','g');
```

Giảng viên: Hoàng Xuân Dương



## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

## 3. Vẽ nhiều hình trên cùng một trục (tt)

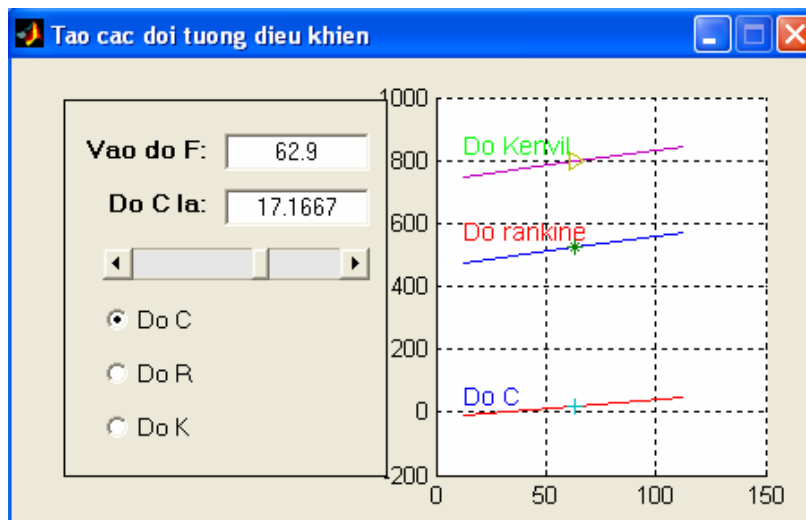
- Viết thêm vào cuối hàm `edit_DoF_Callback` câu lệnh sau:

```
function edit_DoF_Callback(hObject, eventdata, handles)
% hObject handle to edit_DoF (see GCBO)
...
vehinh_Callback(hObject, eventdata, handles);
```

Giảng viên: Hoàng Xuân Dương

## III. TẠO CÁC ĐỐI TƯỢNG ĐIỀU KHIỂN

## 3. Vẽ nhiều hình trên cùng một trục (tt)



Giảng viên: Hoàng Xuân Dương



## IV. ĐO TẦN SỐ

### Nội dung:

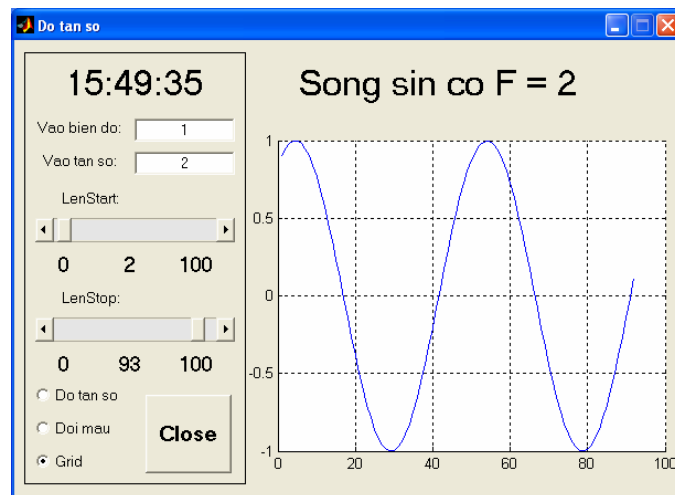
- Tạo và Callback Button
- Tạo TooltipString cho các đối tượng
- Tạo timer
- Đo tần số



## IV. ĐO TẦN SỐ

### 1. Tạo GUI:

- Tạo một GUI như hình vẽ:





## IV. ĐO TẦN SỐ

## 1. Tạo GUI (tt)

- Menu-Inspect Properties → Định thuộc tính các đối tượng

| Figure            |                   |              |
|-------------------|-------------------|--------------|
| Màu nền giao diện | Color             | Tùy ý        |
| Tên tập tin .m    | Filename          | GUI_3        |
| Tên Tiêu đề       | Name              | Do tan so    |
| Độ lớn giao diện  | Position          | [1 1 128 36] |
| Chọn trực vẽ      | Handle Visibility | on           |

| Frame         |                 |       |
|---------------|-----------------|-------|
| Màu nền       | BackgroundColor | Tùy ý |
| Tên của frame | Tag             | frame |

Giảng viên: Hoàng Xuân Dương



## IV. ĐO TẦN SỐ

| Static Text (số lượng 12) |          |                 |               |
|---------------------------|----------|-----------------|---------------|
| STT                       | Fontsize | String          | Tag           |
| 1                         | 30       | Song sin co F = | text_title    |
| 2                         | 30       | 7:38:18         | text_clock    |
| 3                         | 11       | Vao bien do:    | text_BDo      |
| 4                         | 11       | Vao tan so:     | text_TSo      |
| 5                         | 11       | LenStart        | text_LenStart |
| 6                         | 15       | 0               | text_Start0   |
| 7                         | 15       | 1               | text_Start1   |
| 8                         | 15       | 100             | text_Start100 |
| 9                         | 11       | LenStop         | text_LenStop  |
| 10                        | 15       | 0               | text_Stop0    |
| 11                        | 15       | 100             | text_Stop100  |
| 12                        | 15       | 100             | text_Stop1000 |

Giảng viên: Hoàng Xuân Dương





## IV.ĐO TẦN SỐ

| Edit (số lượng: 2) |        |          |               |
|--------------------|--------|----------|---------------|
| STT                | String | Tag      | TooltipString |
| 1                  | 1      | edit_BDo | Vao bien do   |
| 2                  | 2      | edit_TSo | Vao tan so    |

| Slider (số lượng: 2) |     |     |            |       |              |               |
|----------------------|-----|-----|------------|-------|--------------|---------------|
| STT                  | Max | Min | SliderStep | Value | Tag          | TooltipString |
| 1                    | 100 | 0   | [0.01 0.1] | 100   | slider_Start | Gioi han dau  |
| 2                    | 100 | 0   | [0.01 0.1] | 100   | slider_Stop  | Gioi han cuoi |

| RadioButton (số lượng: 3) |           |       |                   |                        |
|---------------------------|-----------|-------|-------------------|------------------------|
| STT                       | String    | Value | Tag               | TooltipString          |
| 1                         | Do tan so | [0.0] | radiobutton_TSo   | Check vao de do tan so |
| 2                         | Doi mau   | [0.0] | radiobutton_Color | Chack vao de doi mau   |
| 3                         | Grid      | [1.0] | radiobutton_Grid  | Check vao de chon luoi |

Giảng viên: Hoàng Xuân Dương



## IV.ĐO TẦN SỐ

| PushButton (số lượng:1) |        |          |                  |                     |
|-------------------------|--------|----------|------------------|---------------------|
| STT                     | String | Fontsize | Tag              | TooltipString       |
| 1                       | Close  | 20       | pushbutton_Close | Thoat khoi ung dung |

| Axes (số lượng:1) |                 |       |       |       |
|-------------------|-----------------|-------|-------|-------|
| STT               | NextPlot        | XGrid | YGrid | ZGrid |
| 1                 | replacechildren | on    | on    | on    |

Giảng viên: Hoàng Xuân Dương

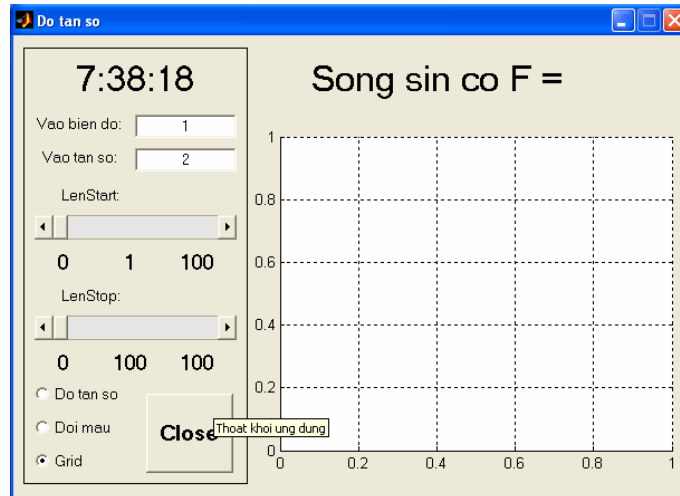


**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**  
**IV. ĐO TẦN SỐ**

473

**1. Tạo GUI (tt)**

- Khi chạy ta có giao diện như sau:



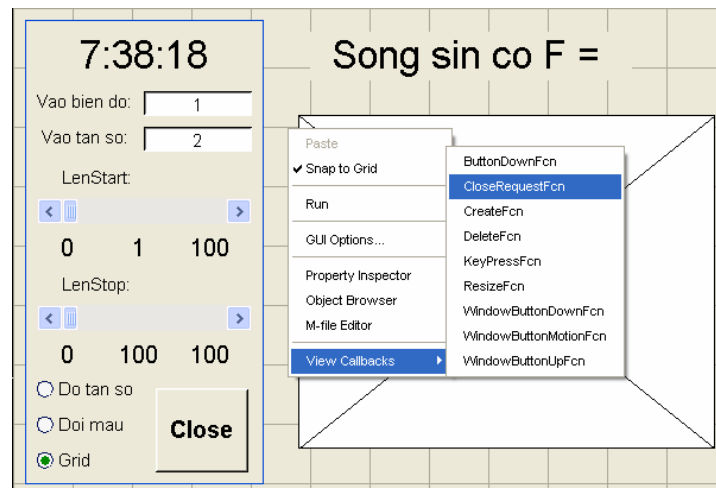
**Giảng viên: Hoàng Xuân Dương**

**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**  
**IV. ĐO TẦN SỐ**

474

**2. Viết hàm kích hoạt các đối tượng:**

- Tạo các hàm **CreateFcn** và **CloseRequestFcn**



**Giảng viên: Hoàng Xuân Dương**



## IV. ĐO TẦN SỐ

## 2. Viết hàm kích hoạt các đối tượng (tt)

- Thêm vào nội dung GUI\_3.m

```
%-----
function varargout = GUI_3_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
Timer(handles); % Gọi hàm timer
%-----
function Timer(handles)
% Hàm hiển thị đồng hồ hệ thống
while find(get(0,'children')==handles.figure1)
    now=fix(clock);
    timestr=[num2str(now(4)) ':' sprintf('%02d',now(5))];
    timestr=[timestr ':' sprintf('%02d',now(6))];
    set(handles.text_clock,'string',timestr);
    pause(1);
end
%-----
```

Giảng viên: Hoàng Xuân Dương



## IV. ĐO TẦN SỐ

## 2. Viết hàm kích hoạt các đối tượng (tt)

```
function Vesin(handles)
global Hsin y %Khai bao bien toan cuc
Biendo=get(handles.edit_BDo,'string'); %Lay day du thong so de ve
Biendo=eval(Biend);
Tanso=get(handles.edit_TSo,'string');
Tanso=eval(Tanso);
Lenstart=get(handles.slider_Start,'value');
Lenstop=get(handles.slider_Stop,'value');
x=linspace(1,2*Tanso*pi+1);
y=Biendo*sin(x); %Noi suy truoc khi ve
Hsin=plot(y(Lenstart:Lenstop)); %Ve vao truc duoc tao
str=[sprintf('Song sin co F = %.2g',Tanso)]; %Thay doi tua de
set(handles.text_title,'string',str);
str=[sprintf('%g',Lenstart)];
set(handles.text_Start1,'string',str);
str=[sprintf('%g',Lenstop)];
set(handles.text_Stop100,'string',str);
```

Giảng viên: Hoàng Xuân Dương





## IV. ĐO TẦN SỐ

## 2. Viết hàm kích hoạt các đối tượng (tt)

- Callback cho các đối tượng:

```
%-----
function edit_BDo_Callback(hObject, eventdata, handles)
Vesin(handles);
%-----
function edit_TSo_Callback(hObject, eventdata, handles)
Vesin(handles);
% --- Executes on slider movement.
function slider_Start_Callback(hObject, eventdata, handles)
Vesin(handles);
% --- Executes on slider movement.
function slider_Stop_Callback(hObject, eventdata, handles)
Vesin(handles);
% --- Executes on button press in pushbutton_Close.
function pushbutton_Close_Callback(hObject, eventdata, handles)
closereq;
```

Giảng viên: Hoàng Xuân Dương



## IV. ĐO TẦN SỐ

## 2. Viết hàm kích hoạt các đối tượng (tt)

```
% --- Executes on button press in radiobutton_TSo.
function radiobutton_TSo_Callback(hObject, eventdata, handles)
global y
[x1,y1]=ginput(1);
text(x1,y1,'-->', 'color', 'r')
[x2,y2]=ginput(1);
text(x2-4,y2,'<--|', 'color', 'r')
Len=length(y);
F=Len/(x2-x1);
string=[sprintf('%0.2g',F)];
text((x2+x1-4)/2,y1,string, 'color', 'r');
set(gcbo,'value',0);
string=[sprintf('Sóng sin có F = %0.2g',F)];
set(handles.text_title,'string',string);
```

Giảng viên: Hoàng Xuân Dương





## IV. ĐO TẦN SỐ

### 2. Viết hàm kích hoạt các đối tượng (tt)

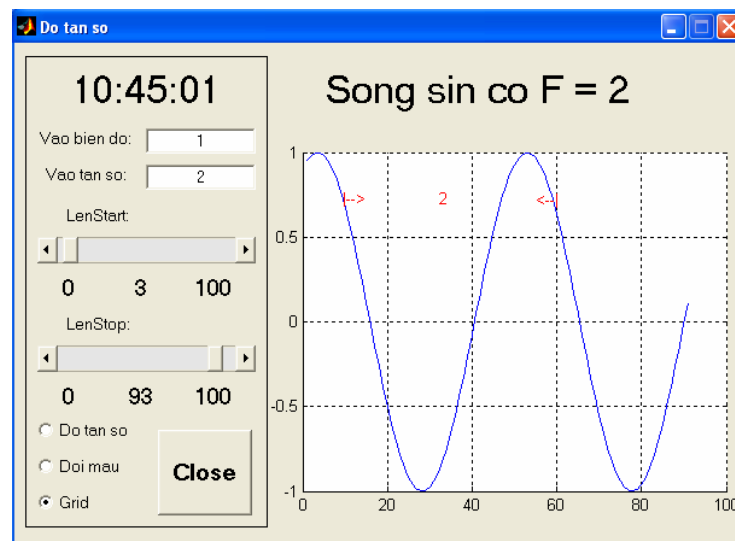
```
% --- Executes on button press in radiobutton_Color.
function radiobutton_Color_Callback(hObject, eventdata, handles)
global Hsin
uisetcolor(Hsin,'Bang mau windows');
set(handles.radiobutton_Color,'value',0);
% --- Executes on button press in radiobutton_Grid.
function radiobutton_Grid_Callback(hObject, eventdata, handles)
check=get(gcbo,'value');
if (check==1)
    grid on
else
    grid off
end
```

Giảng viên: Hoàng Xuân Dương



## IV. ĐO TẦN SỐ

### 3. Thực thi ứng dụng:



Giảng viên: Hoàng Xuân Dương





CHƯƠNG 7: GUI VÀ ỨNG DỤNG

481

V. ĐỒ HỌA 2D

Nội dung:

- Tạo và callback checkbox
- Tạo TooltipString cho các đối tượng
- Format và đặt nhãn cho trục vẽ
- Vẽ hình bằng cách đặt lại
- Tạo popupmenu

Giảng viên: Hoàng Xuân Dương

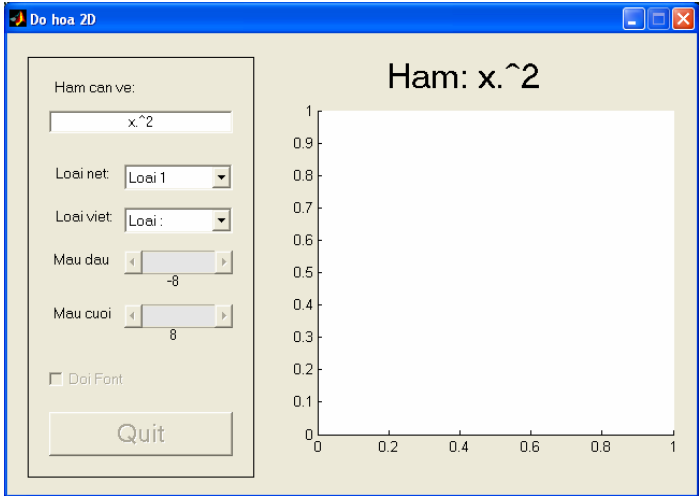
CHƯƠNG 7: GUI VÀ ỨNG DỤNG

482

V. ĐỒ HỌA 2D

1. Tạo GUI:

- Tạo một GUI như hình vẽ:



Giảng viên: Hoàng Xuân Dương



## V. ĐỒ HỌA 2D

## 1. Tạo GUI (tt)

- Định các thuộc tính cho các đối tượng

| Figure            |                   |              |
|-------------------|-------------------|--------------|
| Màu nền giao diện | Color             | Tùy ý        |
| Tên tập tin .m    | Filename          | GUI_4        |
| Tên Tiêu đề       | Name              | Do hoa 2D    |
| Độ lớn giao diện  | Position          | [1 1 128 36] |
| Chọn trục vẽ      | Handle Visibility | on           |

| Frame         |                 |       |
|---------------|-----------------|-------|
| Màu nền       | BackgroundColor | Tùy ý |
| Tên của frame | Tag             | frame |

| Axes (số lượng:1) |                 |       |       |       |
|-------------------|-----------------|-------|-------|-------|
| STT               | NextPlot        | XGrid | YGrid | ZGrid |
| 1                 | replacechildren | on    | on    | on    |

Giảng viên: Hoàng Xuân Dương



## V. ĐỒ HỌA 2D

## 1. Tạo GUI (tt)

| Static Text (số lượng 8) |          |                 |               |
|--------------------------|----------|-----------------|---------------|
| STT                      | Fontsize | String          | Tag           |
| 1                        | 30       | Ham: $x.^2$     | text_hamve    |
| 2                        | 12       | Vao ham can ve: | text_ham      |
| 3                        | 12       | Loai net:       | text_loainet  |
| 4                        | 12       | Loai viet       | text_loaiviet |
| 5                        | 12       | Mau dau         | text_Start    |
| 6                        | 12       | Mau cuoi        | text_Stop     |
| 7                        | 12       | -8              | text_piStart  |
| 8                        | 12       | 8               | text_piStop   |

| Edit (số lượng: 1) |        |          |                |
|--------------------|--------|----------|----------------|
| STT                | String | Tag      | TooltipString  |
| 1                  | $x.^2$ | edit_ham | Vao ham can ve |

Giảng viên: Hoàng Xuân Dương





## V. ĐỒ HỌA 2D

## 1. Tạo GUI (tt)

| popupmenu (số lượng: 2) |        |                                  |       |                |               |
|-------------------------|--------|----------------------------------|-------|----------------|---------------|
| STT                     | String | UserData                         | Value | Tag            | TooltipString |
| 1                       | Loại 1 | str2mat('1','2','3','4','5')     | [1.0] | PopupMenu_net  | Chon net pen  |
|                         | Loại 2 |                                  |       |                |               |
|                         | Loại 3 |                                  |       |                |               |
|                         | Loại 4 |                                  |       |                |               |
|                         | Loại 5 |                                  |       |                |               |
| 2                       | Loại : | str2mat(':', '-', '+', '*', '>') | [1.0] | PopupMenu_loai | Chon loai pen |
|                         | Loại - |                                  |       |                |               |
|                         | Loại + |                                  |       |                |               |
|                         | Loại * |                                  |       |                |               |
|                         | Loại > |                                  |       |                |               |

Giảng viên: Hoàng Xuân Dương



## V. ĐỒ HỌA 2D

## 1. Tạo GUI (tt)

| Slider (số lượng: 2) |     |     |                |       |              |               |
|----------------------|-----|-----|----------------|-------|--------------|---------------|
| STT                  | Max | Min | SliderStep     | Value | Tag          | TooltipString |
| 1                    | 0   | -8  | [0.0625 0.125] | -8    | slider_Start | Gioi han thap |
| 2                    | 100 | 0   | [0.0625 0.125] | 8     | slider_Stop  | Gio han cao   |

| PushButton (số lượng:1) |        |          |                 |                  |
|-------------------------|--------|----------|-----------------|------------------|
| STT                     | String | FontSize | Tag             | TooltipString    |
| 1                       | Quit   | 20       | pushbutton_Quit | Thoat ve Windows |

| Check box (số lượng:1) |          |          |               |               |
|------------------------|----------|----------|---------------|---------------|
| STT                    | String   | FontSize | Tag           | TooltipString |
| 1                      | Doi Font | 12       | Checkbox_Font | Chon Font     |

Giảng viên: Hoàng Xuân Dương



## V. ĐỒ HỌA 2D

## 1. Các hàm kích hoạt:

- Thêm vào nội dung GUI\_4.m:

```
% --- Executes during object creation, after setting all properties.
function frame_CreateFcn(hObject, eventdata, handles)
global Start Stop hình
Start=-8;
Stop=8;
% Dặt tọa độ cho nhãn và trục
title('Hình vẽ','FontSize',15,'Color','r');
xlabel('Trục x','FontSize',15,'Color','r');
ylabel('Trục y','FontSize',15,'Color','r');
x=linspace(-8,8);
func=x.^2;
hình=plot(x,func);
% --- Executes when user attempts to close figure1.
function figure1_CloseRequestFcn(hObject, eventdata, handles)
%delete(hObject);
closereq
```

Giảng viên: Hoàng Xuân Dương

## V. ĐỒ HỌA 2D

## 1. Các hàm kích hoạt (tt)

```
function edit_ham_Callback(hObject, eventdata, handles)
global Start Stop hình
func=get(hObject,'string'); str=['Ham : ' func];
x=linspace(Start,Stop); func=eval(func);
hình=plot(x,func);
set(handles.text_hamve,'string',str);
% --- Executes on selection change in popupmenu_net.
function popupmenu_net_Callback(hObject, eventdata, handles)
global hình
giatri=get(hObject,'value');
loai=get(hObject,'UserData');
set(hình,'LineWidth',str2num(loai(giatri,:)))
% --- Executes on selection change in popupmenu_loaiviet.
function popupmenu_loaiviet_Callback(hObject, eventdata, handles)
global hình
giatri=get(hObject,'value');
loai=get(hObject,'UserData');
set(hình,'LineStyle',loai(giatri,:))
```

Giảng viên: Hoàng Xuân Dương

## V. ĐỒ HỌA 2D

```
% --- Executes on button press in checkbox_Font.
function checkbox_Font_Callback(hObject, eventdata, handles)
check=get(hObject,'value')
if (check==1)
    uisetfont(handles.text_hamve,'Bang chon Font')
    set(hObject,'value',0)
end
% --- Executes on slider movement.
function slider_Start_Callback(hObject, eventdata, handles)
global Start
Start=get(hObject,'value')
set(handles.text_piStart,'string',num2str(Start))
Truc;
% --- Executes on slider movement.
function slider_Stop_Callback(hObject, eventdata, handles)
global Stop
Stop=get(hObject,'value')
set(handles.text_piStop,'string',num2str(Stop))
Truc;
```

Giảng viên: Hoàng Xuân Dương

## V. ĐỒ HỌA 2D

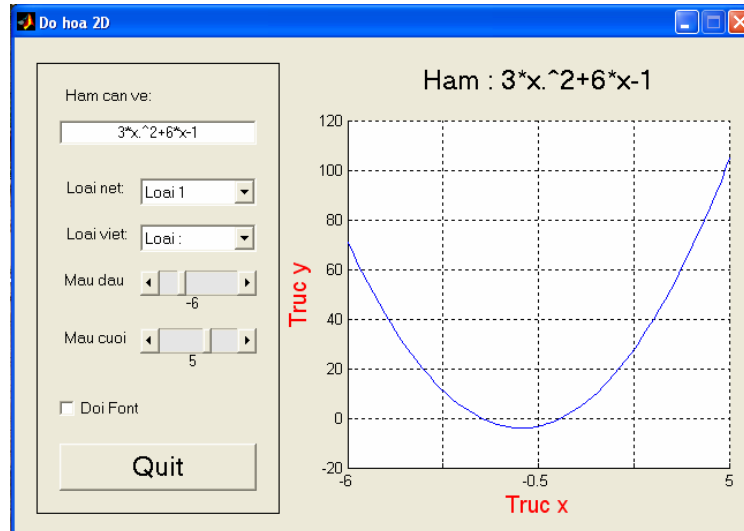
## 1. Các hàm kích hoạt (tt)

```
% --- Ham doi truc ve
function Truc
global Start Stop
handlesAxis=get(gcf,'CurrentAxes');
set(handlesAxis,...
    'XLim',[Start Stop],...
    'XTick',linspace(Start,Stop,5),...
    'XTickLabel',{num2str(Start),...
        "...
        num2str((Start+Stop)/2),...
        "...
        num2str(Stop)}});
% --- Executes on button press in pushbutton_quit.
function pushbutton_quit_Callback(hObject, eventdata, handles)
quit
```

Giảng viên: Hoàng Xuân Dương

## V. ĐỒ HỌA 2D

### 3. Chạy ứng dụng:



Giảng viên: Hoàng Xuân Dương

## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

### Nội dung:

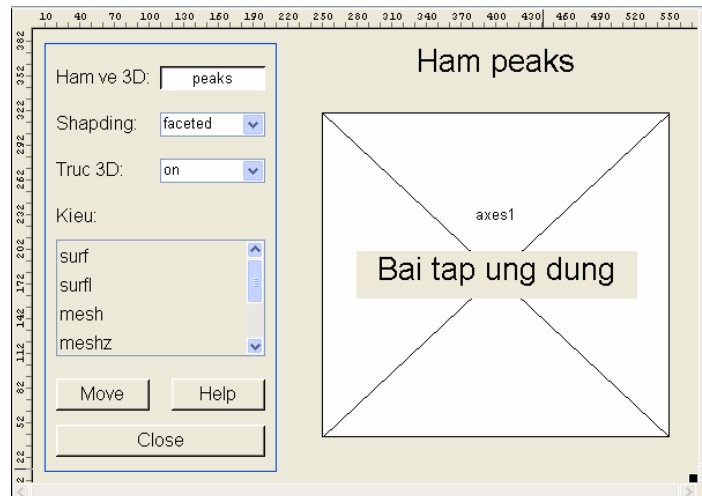
- Tạo và callback listbox
- Đồ họa 3D

Giảng viên: Hoàng Xuân Dương

## VI.ỨNG DỤNG CÁC HÀM VẼ 3D

## 1. Tạo GUI:

- Tạo một GUI như hình vẽ:



**Giảng viên: Hoàng Xuân Dương**

## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

## 1. Tạo GUI (tt)

- Định thuộc tính các đối tượng

| Figure            |                   |              |
|-------------------|-------------------|--------------|
| Màu nền giao diện | Color             | Tùy ý        |
| Tên tập tin .m    | Filename          | GUI_5        |
| Tên Tiêu đề       | Name              | Do hoa 3D    |
| Độ lớn giao diện  | Position          | [1 1 128 36] |
| Chọn trục vẽ      | Handle Visibility | on           |

| Frame         |                 |        |
|---------------|-----------------|--------|
| Màu nền       | BackgroundColor | Tùy ý  |
| Tên của frame | Tag             | frame1 |

**Giảng viên: Hoàng Xuân Dương**



## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

| Static Text (số lượng 6) |          |                  |         |               |
|--------------------------|----------|------------------|---------|---------------|
| STT                      | Fontsize | String           | Visible | Tag           |
| 1                        | 30       | Ham peaks        | off     | text_title    |
| 2                        | 12       | Ham vẽ 3D:       | off     | text_hamve3D  |
| 3                        | 12       | Shapding:        | off     | text_shapding |
| 4                        | 12       | Trục 3D:         | off     | text_truc     |
| 5                        | 12       | Kieu:            | off     | text_kieu     |
| 6                        | 30       | Bài tập ứng dụng | on      | text_baitap   |

| Edit (số lượng 1) |          |        |         |              |
|-------------------|----------|--------|---------|--------------|
| STT               | Fontsize | String | Visible | Tag          |
| 1                 | 12       | peaks  | off     | Edit_hamve3D |

Giảng viên: Hoàng Xuân Dương



## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

| Pushbutton (số lượng 3) |          |        |         |                  |
|-------------------------|----------|--------|---------|------------------|
| STT                     | Fontsize | String | Visible | Tag              |
| 1                       | 12       | Move   | off     | pushbutton_move  |
| 2                       | 12       | Help   | off     | pushbutton_help  |
| 3                       | 12       | Close  | off     | pushbutton_close |

| PopupMenu (số lượng 2) |                           |       |                               |         |                    |
|------------------------|---------------------------|-------|-------------------------------|---------|--------------------|
| STT                    | String                    | value | UserData                      | Visible | Tag                |
| 1                      | faceted<br>flat<br>interp | [1.0] | {'faceted'; 'flat'; 'interp'} | off     | PopupMenu_shapding |
| 2                      | on<br>off<br>ij<br>xy     | [1.0] | {'on'; 'off'; 'ij'; 'xy'}     | off     | PopupMenu_truc     |

Giảng viên: Hoàng Xuân Dương







## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

| listbox (số lượng 1) |                                                       |       |                                                                    |         |              |
|----------------------|-------------------------------------------------------|-------|--------------------------------------------------------------------|---------|--------------|
| STT                  | String                                                | value | UserData                                                           | Visible | Tag          |
| 1                    | surf<br>surfl<br>mesh<br>meshz<br>waterfall<br>pcolor | [1.0] | str2mat('surf','surfl','mesh',...<br>'meshz','waterfall','pcolor') | off     | listbox_kieu |

| Axes (số lượng:1) |                 |         |       |       |       |
|-------------------|-----------------|---------|-------|-------|-------|
| STT               | NextPlot        | Visible | XGrid | YGrid | ZGrid |
| 1                 | replacechildren | on      | on    | on    | on    |

Giảng viên: Hoàng Xuân Dương



## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

## 2. Viết các hàm kích hoạt:

- Thêm vào nội dung GUI\_5.m

```
%-----
function edit_hamve3D_Callback(hObject, eventdata, handles)
    Ve3D(handles);
% --- Executes on selection change in popupmenu_shapding.
function popupmenu_shapding_Callback(hObject, eventdata, handles)
    value=get(handles.popupmenu_shapding,'value');
    color=get(handles.popupmenu_shapding,'UserData');
    Shading(color{value})
% --- Executes on selection change in popupmenu_truc.
function popupmenu_truc_Callback(hObject, eventdata, handles)
    value=get(handles.popupmenu_truc,'value');
    truc=get(handles.popupmenu_truc,'UserData');
    axis(truc{value})
% --- Executes on selection change in listbox_kieu.
function listbox_kieu_Callback(hObject, eventdata, handles)
    Ve3D(handles);
```

Giảng viên: Hoàng Xuân Dương



## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

## 2. Viết các hàm kích hoạt (tt)

```
% --- Executes on button press in pushbutton_move.
function pushbutton_move_Callback(hObject, eventdata, handles)
for j=1:10
    view(-37.5+24*(j-1),30);
    n(:,1)=getframe;
end
movie(n,5)
% --- Executes on button press in pushbutton_help.
function pushbutton_help_Callback(hObject, eventdata, handles)
graf3d('info')
% --- Executes on button press in pushbutton_close.
function pushbutton_close_Callback(hObject, eventdata, handles)
closereq
% --- Executes when user attempts to close figure1.
function figure1_CloseRequestFcn(hObject, eventdata, handles)
delete(hObject);
closereq
```

Giảng viên: Hoàng Xuân Dương

## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

## 2. Viết các hàm kích hoạt (tt)

```
% --- If Enable == 'on', executes on mouse press in 5 pixel border.
function varargout = text_baitap_ButtonDownFcn(hObject, eventdata, handles)
handles=guihandles(gcbo); %Lay tat ca cac handle
promptstr={'Cho vao password'};
inistr={''};
dlgTitle='Nhap Password';
lineNo=1;
result=inputdlg(promptstr,dlgTitle,lineNo,inistr);
if strcmp(result,'GUI_5')
    set(handles.text_baitap,'visible','off');
    set(handles.frame1,'visible','on');
    set(handles.text_hamve3D,'visible','on');
    set(handles.edit_hamve3D,'visible','on');
    set(handles.text_shapding,'visible','on');
    set(handles.popupmenu_shapding,'visible','on');
    .....
end
```

Giảng viên: Hoàng Xuân Dương

## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

## 2. Viết các hàm kích hoạt (tt)

```

.....
set(handles.text_truc,'visible','on');
set(handles.popupmenu_truc,'visible','on');
set(handles.text_kieu,'visible','on');
set(handles.listbox_kieu,'visible','on');
set(handles.text_title,'visible','on');
set(handles.axes1,'visible','on');
set(handles.pushbutton_close,'visible','on');
set(handles.pushbutton_help,'visible','on');
set(handles.pushbutton_move,'visible','on');
Ve3D(handles);
else
    errordlg('Password ?');
end

```

Giảng viên: Hoàng Xuân Dương

## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

## 2. Viết các hàm kích hoạt (tt)

```

function Ve3D(handles)
global Hinh
func=get(handles.edit_hamve3D,'string');
[x,y]=meshgrid(-2*pi:.5:2*pi,-2*pi:.5:2*pi);
z=eval(func);
giatri=get(handles.listbox_kieu,'value');
loai=get(handles.listbox_kieu,'UserData');
str=['Loai ':' ' loai(giatri,:)];
set(handles.text_title,'string',str)
if (giatri==1) Hinh=surf(z);
elseif (giatri==2) Hinh=surfl(z);
elseif (giatri==3) Hinh=mesh(z);
elseif (giatri==4) Hinh=meshz(z);
elseif (giatri==5) Hinh=waterfall(z);
elseif (giatri==6) Hinh=pcolor(z);
end

```

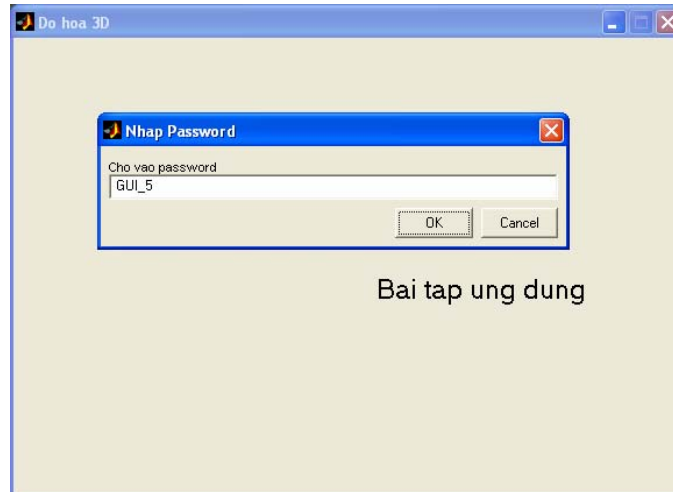
Giảng viên: Hoàng Xuân Dương



## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

### 3. Thực thi ứng dụng:

- Nhập mật khẩu

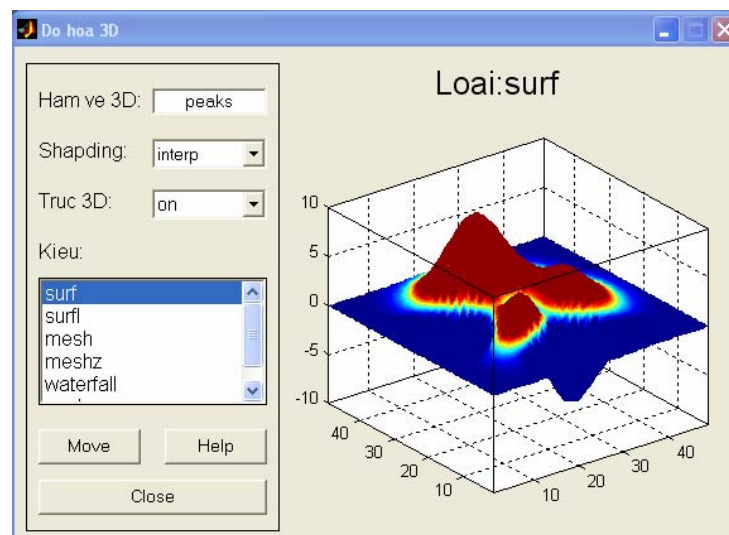


Giảng viên: Hoàng Xuân Dương



## VI. ỨNG DỤNG CÁC HÀM VẼ 3D

### 3. Thực thi ứng dụng (tt)



Giảng viên: Hoàng Xuân Dương





## VII.BIẾN ĐIỆU ANALOG

### Nội dung:

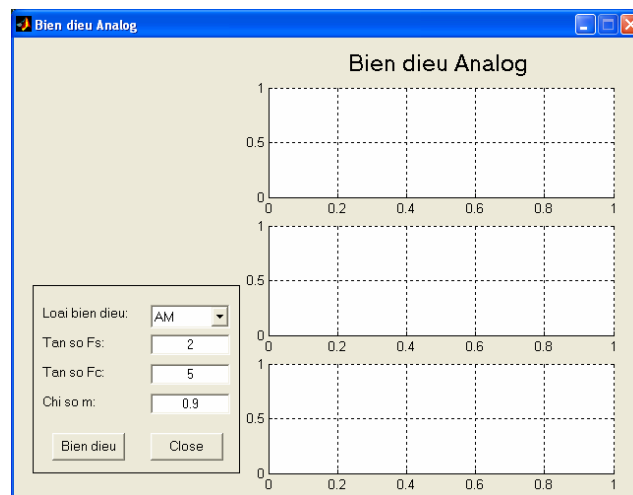
- Vẽ với nhiều trục
- Biến điệu AM, FM, PM



## VII.BIẾN ĐIỆU ANALOG

### 1. Tạo GUI:

- Tạo một GUI như hình vẽ:





## VII.BIẾN ĐIỆU ANALOG

## 1. Tạo GUI (tt)

- Định các thuộc tính:

| Đối tượng   | STT | String           | Fontsize | Tag           |
|-------------|-----|------------------|----------|---------------|
| Static Text | 1   | Bien dieu Analog | 30       | text_title    |
|             | 2   | Loai bien dieu:  | 12       | text_hamve3D  |
|             | 3   | Tan so FS        | 12       | text_shapding |
|             | 4   | Tan so Fc        | 12       | text_truc     |
|             | 5   | Chi so m         | 12       | text_kieu     |
| Popup Menu  | 1   | AM<br>FM<br>PM   | 10       | Popupmenu_BD  |
| Edit Text   | 1   | 2                | 10       | edit_Fs       |
|             | 2   | 5                | 10       | edit_Fc       |
|             | 3   | 0.9              | 10       | edit_m        |

Giảng viên: Hoàng Xuân Dương



## VII.BIẾN ĐIỆU ANALOG

## 1. Tạo GUI (tt)

- Định các thuộc tính:

| Đối tượng  | STT | String    | Fontsize | Tag              |
|------------|-----|-----------|----------|------------------|
| PushButton | 1   | Bien dieu |          | pushbutton_BD    |
|            | 2   | Close     |          | pushbutton_close |

| Đối tượng | STT | NextPlot        | XGrid | YGrid | ZGrid | Tag   |
|-----------|-----|-----------------|-------|-------|-------|-------|
| Axes      | 1   | replacechildren | on    | on    | on    | axes1 |
|           | 2   |                 |       |       |       | axes2 |
|           | 3   |                 |       |       |       | axes3 |

Giảng viên: Hoàng Xuân Dương



## VII. BIẾN ĐIỀU ANALOG

## 2. Tạo các hàm kích hoạt:

- Bổ sung vào tập tin GUI\_6.m

```
% --- Executes on button press in pushbutton_BD.
function pushbutton_BD_Callback(hObject, eventdata, handles)
Biendieu(handles)
% --- Executes on button press in pushbutton_close.
function pushbutton_close_Callback(hObject, eventdata, handles)
closereq;
% --- Executes during object creation, after setting all properties.
function frame_CreateFcn(hObject, eventdata, handles)
handles=guihandles(gcbo);
Biendieu(handles)
```

Giảng viên: Hoàng Xuân Dương

## VII. BIẾN ĐIỀU ANALOG

## 2. Tạo các hàm kích hoạt (tt)

```
% Ham tu viet de ve hinh
function Biendieu(handles)
loai=get(handles.popupmenu_BD,'value');
Fs=get(handles.edit_Fs,'string'); Fs=eval(Fs);
Fc=get(handles.edit_Fc,'string'); Fc=eval(Fc);
m=get(handles.edit_m,'string'); m=eval(m);
switch (loai)
case 1
    AM(handles,Fs,Fc,m)
    set(handles.text_title,'string','Bien dieu AM');
case 2
    FM(handles,Fs,Fc,m)
    set(handles.text_title,'string','Bien dieu FM');
case 3
    PM(handles,Fs,Fc,m)
    set(handles.text_title,'string','Bien dieu PM');
end
```

Giảng viên: Hoàng Xuân Dương

## VII. BIẾN ĐIỆU ANALOG

## 2. Tạo các hàm kích hoạt (tt)

```
% Ham tu viet de bien dieu AM
function AM(handles,Fs,Fc,m)
t=(0:1000)/1000; x=cos(2*pi*Fs*t); c=cos(2*pi*Fc*t);
y=0.5*(1+m*x).*c;
plot(t,m*x,'Parent',handles.axes1);
plot(t,y,'Parent',handles.axes2);
[f,Pyy]=Pho(handles,y);
plot(f,Pyy(1:257),'r','Parent',handles.axes3);
% Ham tu viet de bien dieu FM
function FM(handles,Fs,Fc,m)
t=(0:1000)/1000; x=cos(2*pi*Fs*t); c=cos(2*pi*Fc*t);
y=cos((2*pi*Fc*t)+(m/(2*pi*Fs))*200*sin(2*pi*Fs*t));
plot(t,m*x,'Parent',handles.axes1);
plot(t,y,'Parent',handles.axes2);
[f,Pyy]=Pho(handles,y);
plot(f,Pyy(1:257),'r','Parent',handles.axes3);
```

Giảng viên: Hoàng Xuân Dương

## VII. BIẾN ĐIỆU ANALOG

## 2. Tạo các hàm kích hoạt (tt)

```
% Ham tu viet de bien dieu PM
function PM(handles,Fs,Fc,m)
t=(0:1000)/1000; x=cos(2*pi*Fs*t); c=cos(2*pi*Fc*t);
y=cos((2*pi*Fc*t)+(m/(2*pi*Fs))*200*cos(2*pi*Fs*t));
plot(t,m*x,'Parent',handles.axes1);
plot(t,y,'Parent',handles.axes2);
[f,Pyy]=Pho(handles,y);
plot(f,Pyy(1:257),'r','Parent',handles.axes3);
% Ham tu viet de ve pho
function [f,Pyy]=Pho(handles,y)
yy=fft(y,512);
Pyy=yy.*conj(yy)/512;
f=1000*(0:256)/512
```

Giảng viên: Hoàng Xuân Dương

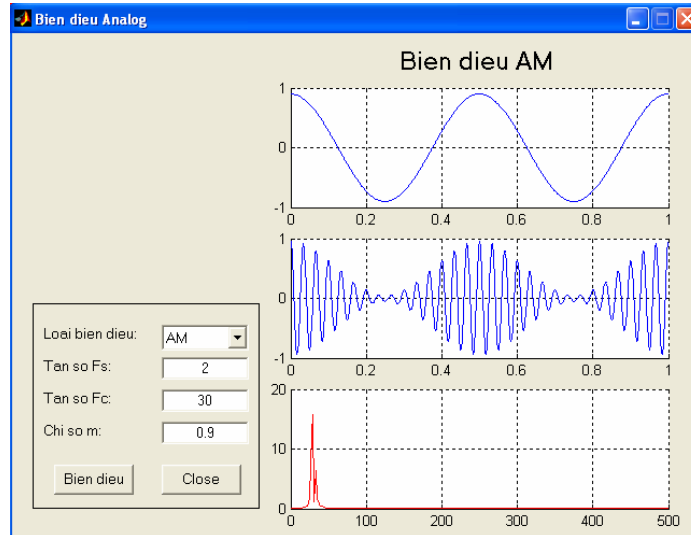


CHƯƠNG 7: GUI VÀ ỨNG DỤNG

VII.BIẾN ĐIỆU ANALOG

513

3. Chạy ứng dụng:



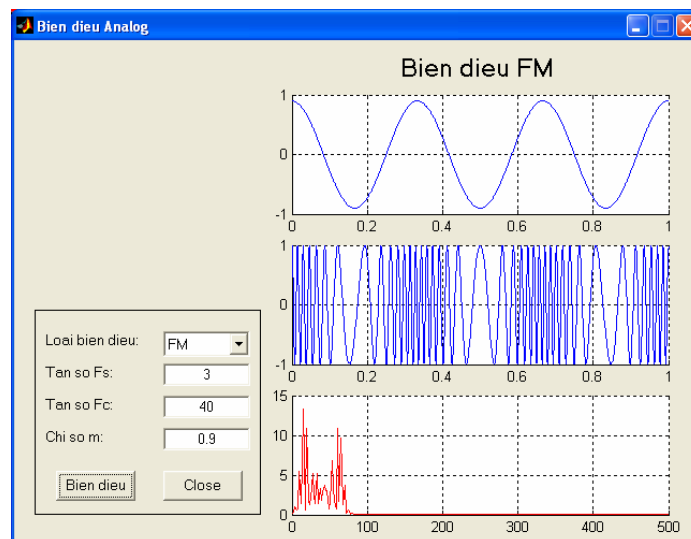
Giảng viên: Hoàng Xuân Dương

CHƯƠNG 7: GUI VÀ ỨNG DỤNG

VII.BIẾN ĐIỆU ANALOG

514

3. Chạy ứng dụng:



Giảng viên: Hoàng Xuân Dương



## VIII.BIẾN ĐIỆU DIGITAL

### Nội dung:

- Vẽ với nhiều trục
- Biến điệu ASK, FSK, PSK



## VIII.BIẾN ĐIỆU DIGITAL

### 1. Tạo GUI:

- Tạo một GUI như hình vẽ:

The screenshot shows a software interface for digital modulation simulation. On the left, a control panel allows users to configure parameters: modulation type (ASK), input binary sequence (101001), bit rate (300), sampling rate (120), carrier frequency (1500), mark frequency (1200), and space frequency (2000). A 'Bien dieu' button is provided to execute the simulation. On the right, three separate plots, each titled 'Bien dieu', show the resulting modulated signal waveforms over time (0 to 1). Each plot has a vertical axis ranging from 0 to 1.



**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**

**VIII.BIẾN ĐIỆU DIGITAL**

**1. Tạo GUI (tt)**

- Các thuộc tính của các component

517

| Đối tượng   | STT | String            | Fontsize | Tag          |
|-------------|-----|-------------------|----------|--------------|
| Static Text | 1   | Bien dieu         | 30       | text_title   |
|             | 2   | Loai bien dieu:   | 12       | text_BD      |
|             | 3   | Nhap chuoiso:     | 12       | text_chuoiso |
|             | 4   | Toc do bit:       | 12       | text_tocdo   |
|             | 5   | Hang so lay mau:  | 12       | text_laymau  |
|             | 6   | Song mang Fc:     | 12       | text_Fc      |
|             | 7   | Nhap Fmark        | 12       | text_Fmark   |
|             | 8   | Nhap Fspace:      | 12       | text_Fspace  |
| Popup Menu  | 1   | ASK<br>FSK<br>PSK | 10       | Popupmenu_BD |

**Giảng viên: Hoàng Xuân Dương**

**CHƯƠNG 7: GUI VÀ ỨNG DỤNG**

**VIII.BIẾN ĐIỆU DIGITAL**

**1. Tạo GUI (tt)**

518

| Đối tượng  | STT | String      | Fontsize | Tag           |
|------------|-----|-------------|----------|---------------|
| Edit Text  | 1   | 1 0 1 0 0 1 | 10       | edit_chuoiso  |
|            | 2   | 300         | 10       | edit_tocdo    |
|            | 3   | 120         | 10       | edit_laymau   |
|            | 4   | 1500        | 10       | edit_Fc       |
|            | 5   | 1200        | 10       | edit_Fmark    |
|            | 6   | 2000        | 10       | edit_Fspace   |
| PushButton | 1   | Bien dieu   | 20       | pushbutton_BD |

| Đối tượng | STT | NextPlot        | XGrid | YGrid | ZGrid | Tag   |
|-----------|-----|-----------------|-------|-------|-------|-------|
| Axes      | 1   | replacechildren | on    | on    | on    | axes1 |
|           | 2   |                 |       |       |       | axes2 |
|           | 3   |                 |       |       |       | axes3 |

**Giảng viên: Hoàng Xuân Dương**

## VIII. BIẾN ĐIỀU DIGITAL

## 2. Tạo các hàm callback:

```
function popupmenu_BD_Callback(hObject, eventdata, handles)
value=get(gcbo,'value');
if value==1    %ASK
    set(handles.edit_Fc,'enable','on')
    set(handles.edit_Fmark,'enable','off')
    set(handles.edit_laymau,'enable','off')
elseif value==2    %FSK
    set(handles.edit_Fc,'enable','off')
    set(handles.edit_Fmark,'enable','on')
    set(handles.edit_laymau,'enable','on')
elseif value==3    %PSK
    set(handles.edit_Fc,'enable','on')
    set(handles.edit_Fmark,'enable','off')
    set(handles.edit_laymau,'enable','off')
end
% --- Executes on button press in pushbutton_BD.
function pushbutton_BD_Callback(hObject, eventdata, handles)
Biendieu(handles);
```

Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỀU DIGITAL

```
% --- Executes during object creation, after setting all properties.
function frame1_CreateFcn(hObject, eventdata, handles)
handles=guihandles(gcbo);
Biendieu(handles);
% --- Ham bien dieu
function Biendieu(handles)
% Nhan chuoì so
binary_seq=get(handles.edit_chuoiso,'string');
binary_seq=str2num(binary_seq);
%Nhan gia tri toc do bit
R=get(handles.edit_tocdo,'string');
R=eval(R);
%Nhan gia tri Fc
Fc=get(handles.edit_Fc,'string');
Fc=eval(Fc);
%Nhan gia tri hang lay mau
Sampling=get(handles.edit_laymau,'string');
Sampling=eval(Sampling);
%Nhan gia tri Fmark(Fcmin) la tan so nho nhat cua song mang
.....
```

Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỆU DIGITAL

```

.....
Fmark=get(handles.edit_Fmark,'string');
Fmark=eval(Fmark);
%Nhan gia tri Fspace(Fcmin) la tan so cao nhât của sóng mang
Fspace=get(handles.edit_Fspace,'string');
Fspace=eval(Fspace);
Fs=R*Sampling;
%Xet muc duoc chon trong popupmenu
LoaiBD=get(handles.popupmenu_BD,'value');
if LoaiBD==1 %ASK
    if (Fc>=(R*Sampling)/2)
        ErrorDlg('Fspace<Sampling*(R/2)'); return;
    end
    ve_digital(handles,binary_seq,Fs)
    out=ASK(handles,binary_seq,Fc,Fs,R);
    Pho_digital(handles,out);
    set(handles.text_title,'string','Bien dieu ASK')
elseif LoaiBD==2 %FSK
    if (Fc>=(R*Sampling)/2)
.....

```

Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỆU DIGITAL

```

.....
    ErrorDlg('Fspace<Sampling*(R/2)'); return;
end
if (Fmark<R)
    ErrorDlg('Fmark>=R'); return;
end
ve_digital(handles,binary_seq,Fs)
out=FSK(handles,binary_seq,[Fmark Fspace],Fs,R);
Pho_digital(handles,out);
set(handles.text_title,'string','Bien dieu FSK')
elseif LoaiBD==3 %PSK
    if (Fc>=(R*Sampling)/2)
        ErrorDlg('Fspace<Sampling*(R/2)'); return;
    end
    ve_digital(handles,binary_seq,Fs)
    out=PSK(handles,binary_seq,Fc,Fs,R);
    Pho_digital(handles,out);
    set(handles.text_title,'string','Bien dieu PSK')
end

```

Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỆU DIGITAL

```

% Ham ve tin hieu so
function ve_digital(handles,binary_seq,Fs)
SAMPLING_FREQ=Fs;
Ts=1/SAMPLING_FREQ;
binary_seq=binary_seq(:); %Doi thanh cot
no_sample=length(binary_seq); %Xet chieu dai chuoai
amplitude=max(abs(binary_seq));
time=[1:(no_sample)]*Ts;
ax=[min(time)/100 max(time) -2*amplitude 2*amplitude];
axes(handles.axes1)
stair(time,binary_seq); %Ham ve tin hieu so
axis(ax);
set(gca,'XTickLabel',{' ',' ',' ',' ',' ',' ',' ',' ',' '});
xlabel('Tin hieu nen','fontname','SVNhelvetica','fontsize',12,'color','r');
% --- Ham stair ---
function [xo,yo]=stair(x,y)
n=length(x);
.....

```

Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỆU DIGITAL

```

.....
if nargin==1
    y=x; x=1:n;
end
delta=(max(x)-min(x))/(n-1);
nn=2*n;
yy=zeros(nn+2,1);
xx=yy;
t=x(:)';
xx(1:2:nn)=t;
xx(2:2:nn)=t;
xx(nn+1:nn+2)=t(n)+[delta;delta];
yy(2:2:nn)=y;
yy(3:2:nn+1)=y;
if nargout==0
    plot(xx,yy)
else
    xo=xx; yo=yy;
end

```

Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỆU DIGITAL

```
% Ham bien dieu ASK
function out=ASK(handles,binary_seq,Fc,Fs,R)
%Tao data cua dang song khong tro ve zero nrz
x=wave_gen(handles,binary_seq,'unipolar_nrz',Fs,R);
out=mixer(x,osc(Fc,Fs));
Ts=1/Fc;
out=out(:);
no_sample=length(out);
amplitude=max(abs(out));
t=[1:(no_sample)]*Ts;
ax=[min(t) max(t) -2*amplitude 2*amplitude]
axes(handles.axes2)
y=plot(t,out);
axis(ax);
set(gca,'XTickLabel',{' ',' ',' ',' ',' '});
xlabel('Tin hieu sau khi bien dieu','fontname',...
'SVNhelvetica','fontsize',12,'color','r');
```

Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỆU DIGITAL

```
% Ham tao dang song nrz
function out=wave_gen(handles,binary_sequence,linecode,Fs,Rb)
binary_sequence=binary_sequence(:);
if (any(abs(binary_sequence)-sign(binary_sequence)))
error('Khong phai tin hieu digital')
end
if strcmp(linecode,'unipolar_nrz')
pulse='rect_nrz(Rb,Fs)';
b_seq=binary_sequence;
elseif strcmp(linecode,'polar_nrz')
pulse='rect_nrz(Rb,Fs)';
b_seq=bin2pol(handles,binary_sequence);
else
error('Ma khong phu hop')
end
x=(b_seq*eval(pulse));
out=x(:);
```

Giảng viên: Hoàng Xuân Dương

## VIII. BIÊN ĐIỀU DIGITAL

```
%Tao tin hieu khong tro ve zero
function out=rect_nrz(Rb,Fs)
out=ones(1,Fs/Rb);
%Ham tao song mang
function [carrier]=osc(fc,Fs)
t=[1:50000]/Fs;
carrier=sin(2*pi*t*fc);
% Ham tron 2 tin hieu
function [out]=mixer(in,fc)
%Z=MIXER(X,Y) Tao chuoai Z:  $Z(n)=X(n)*Y(n)$ 
n=length(in);
carrier=fc(1:n);
x=in;
x=x(:)';
out=carrier.*x;
```

## VIII. BIẾN ĐIỆU DIGITAL

```
% Ham ve pho
function Pho_digital(handles,x)
x=x(:);
xx=fft(x,512);
Pxx=xx.*conj(xx)/512;
f=1000*(0:255)/512;
no_sample=length(Pxx);
amplitude=max(abs(Pxx));
ax=[min(f) max(f) -2*amplitude 2*amplitude]
axes(handles.axes3)
plot(f,Pxx(1:256),'r');
axis(ax);
set(gca,'XTickLabel',{' ',' ',' ',' ',' '});
xlabel('Pho cua song bien dieu','fontname',...
'SVNhelvetica','fontsize',12,'color','r');
```



## VIII. BIẾN ĐIỆU DIGITAL

```
%Ham bien dieu FSK
function out=FSK(handles,binary_seq,Fc,Fs,Rb)
CARRIER_FREQUENCY=[min(Fc) max(Fc)];
x=wave_gen(handles,binary_seq,'polar_nrz',Fs,Rb);
f_r=(CARRIER_FREQUENCY(2)+CARRIER_FREQUENCY(1))/2
kf=(CARRIER_FREQUENCY(2)-CARRIER_FREQUENCY(1))/2
out=vco_digital(x,f_r,kf,Fs);
Ts=1/f_r; out=out(:); no_sample=length(out);
amplitude=max(abs(out)); t=[1:(no_sample)]*Ts;
ax=[min(t) max(t) -2*amplitude 2*amplitude]
axes(handles.axes2)
y=plot(t,out);
axis(ax);
set(gca,'XTickLabel',{' ',' ',' ',' ',' '});
xlabel('Tin hieu sau khi bien dieu','fontname',...
'SVNhelvetica','fontsize',12,'color','r');
% Ham luong cuc hoa chuoai nhi phan vao
function [polar_sequence]=bin2pol(handles,binary_sequence)
polar_sequence=2*binary_sequence-ones(size(binary_sequence));
```

Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỆU DIGITAL

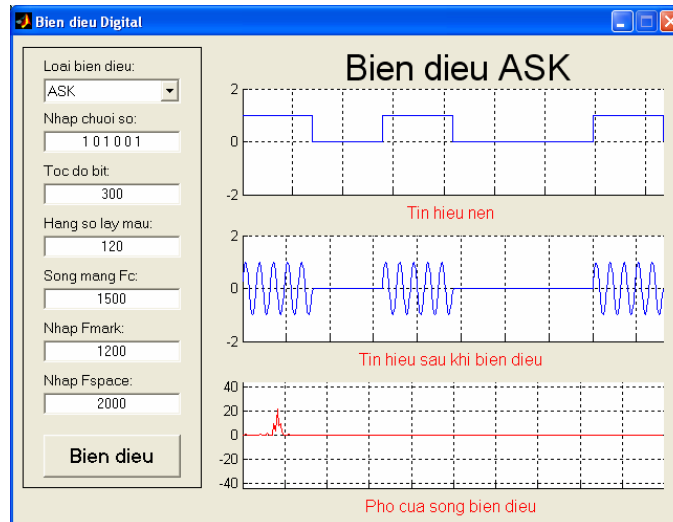
```
% Dao dong duoc dieu khien bang dien the
function [out]=vco_digital(in,arg2,arg3,Fs)
Ts=1/Fs; fc=arg2; kf=arg3;
lenfc=ones(length(in),1)*fc;
phase=cumsum((lenfc+in*kf)*Ts*2*pi);
out=sin(phase);
% Ham bien dieu PSK
function out=PSK(handles,binary_seq,Fc,Fs,Rb)
x=wave_gen(handles,binary_seq,'polar_nrz',Fs,Rb);
out=mixer(x,osc(Fc,Fs));
Ts=1/Fc; out=out(:); no_sample=length(out);
amplitude=max(abs(out));
t=[1:(no_sample)]*Ts;
ax=[min(t) max(t) -2*amplitude 2*amplitude]
axes(handles.axes2)
plot(t,out);
axis(ax);
set(gca,'XTickLabel',{' ',' ',' ',' ',' '});
xlabel('Tin hieu sau khi bien dieu','fontname',...
'SVNhelvetica','fontsize',12,'color','r');
```

Giảng viên: Hoàng Xuân Dương

CHƯƠNG 7: GUI VÀ ỨNG DỤNG  
VIII. BIẾN ĐIỆU DIGITAL

531

3. Chạy ứng dụng:

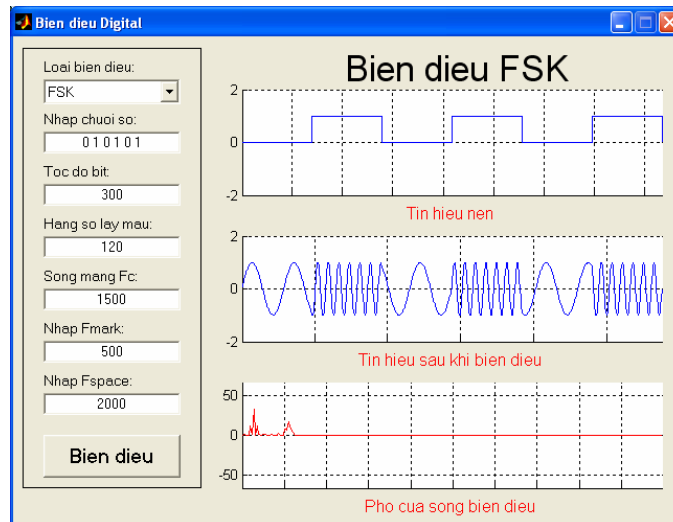


Giảng viên: Hoàng Xuân Dương

CHƯƠNG 7: GUI VÀ ỨNG DỤNG  
VIII. BIẾN ĐIỆU DIGITAL

532

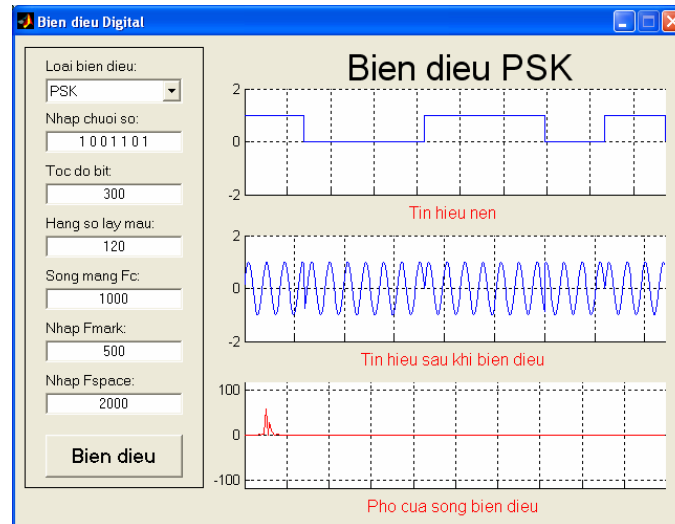
3. Chạy ứng dụng (tt)



Giảng viên: Hoàng Xuân Dương

## VIII. BIẾN ĐIỆU DIGITAL

### 3. Chạy ứng dụng (tt)



**Giảng viên: Hoàng Xuân Dương**

## IX.CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

### ❖ Các hộp thoại dialog:

| Dialog boxes | Ý nghĩa                           |
|--------------|-----------------------------------|
| errordlg     | Tạo dialog box báo lỗi            |
| helpdlg      | Hiển thị một dialog box giúp đỡ   |
| inputdlg     | Tạo một dialog box nhập liệu      |
| listdlg      | Tạo một dialog danh sách chọn lựa |
| msgbox       | Tạo một dialog box thông tin      |
| pagedlg      | Tạo một dialog box page layout    |
| printdlg     | Hiển thị một dialog in            |

**Giảng viên: Hoàng Xuân Dương**

## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

## ❖ Các hộ thoại dialog (tt)

| Dialog boxes | Ý nghĩa                                       |
|--------------|-----------------------------------------------|
| questdlg     | Tạo một dialog hỏi                            |
| uigetfile    | Hiển thị dialog box nhận tên của file cần đọc |
| uiputfile    | Hiển thị dialog box nhận tên của file để ghi  |
| uicolor      | Chọn màu bằng bảng màu của windows            |
| uisetfont    | Chọn font                                     |
| warndlg      | Tạo một dialog cảnh báo                       |

Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

1. **errordlg**

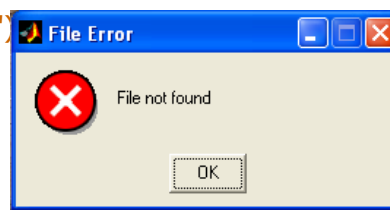
Cú pháp:

`errordlg``errordlg('errorstring')``errordlg('errorstring','dlgname')``errordlg('errorstring','dlgname','on')`

%on' → cho phép hay không thay thế dialog có cùng tên

h = `errordlg(...)`

Ví dụ:

`errordlg('File not found','File Error')`

Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

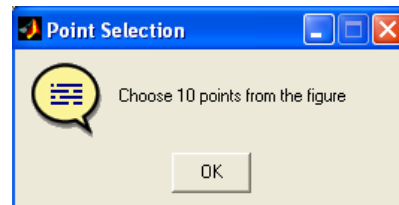
2. **helpdlg**

Cú pháp:

```
helpdlg
helpdlg('helpstring')
helpdlg('helpstring','dlgname')
h = helpdlg(...)
```

Ví dụ:

```
helpdlg('Choose 10 points from the figure','Point Selection');
```



Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

3. **inputdlg**

Cú pháp:

```
answer = inputdlg(prompt)
answer = inputdlg(prompt,title)
answer = inputdlg(prompt,title,lineNo)
answer = inputdlg(prompt,title,lineNo,defAns)
answer = inputdlg(prompt,title,lineNo,defAns,Resize)
```

Trong đó:

**prompt** → Các tring xuất hiện trên các hộp nhập liệu  
**title** → Tiêu đề của hộp thoại  
**lineNo** → Số dòng trong ô nhập liệu  
**defAns** → Kết quả nhập liệu mặc định (xuất hiện ban đầu)  
**resize** → Cho phép hay không thay đổi kích thước dialog box  
 'on' hay 'off'

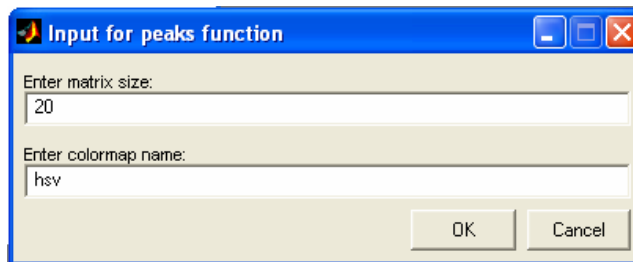
Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

3. **inputdlg (tt)**

Ví dụ:

```
prompt = {'Enter matrix size:','Enter colormap name:'};
title = 'Input for peaks function';
lines= 1;
def = {'20','hsv'};
answer = inputdlg(prompt,title,lines,def);
```



Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

4. **listdlg**

Cú pháp:

```
[Selection,ok] = listdlg('ListString',S,...)
```

% Cho phép chọn một hay nhiều item trong danh sách

Trong đó:

- Selection → vector chứa các string được chọn
- ok = 1 → khi chọn nút OK  
0 → khi chọn cancel hoặc đóng hộp thoại
- S là các thông số trong bảng sau:

Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

## 4. listdlg (tt)

| Parameter       | Ý nghĩa                                                                                  |
|-----------------|------------------------------------------------------------------------------------------|
| 'ListString'    | Dãy các chuỗi để chọn nằm trong list box                                                 |
| 'SelectionMode' | 'single' chỉ cho chọn 1<br>'multiple' (the default) cho phép chọn nhiều                  |
| 'ListSize'      | Kích thước list box, tính bằng pixel,<br>là vector [width height]. Mặc định là [160 300] |
| 'InitialValue'  | Item được chọn ban đầu. Mặc định là 1 (item đầu)                                         |
| 'Name'          | Tiêu đề của dialog box. Mặc định là ''                                                   |
| 'PromptString'  | Các string xuất hiện phía trên listbox. Mặc định {}                                      |
| 'OKString'      | String cho nút nhấn OK. Mặc định là 'OK'                                                 |
| 'CancelString'  | String cho nút nhấn Cancel. Mặc định là 'Cancel'.                                        |

Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

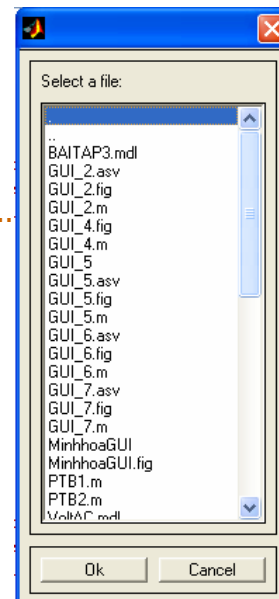
## 4. listdlg (tt)

Ví dụ:

```

d=dir;
str={d.name};
[s,v]=listdlg('PromptString','Select a file:',...
             'SelectionMode','single',...
             'ListString',str)

```



Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP THOẠI DIALOG CỦA WINDOWS

5. **msgbox**

Cú pháp:

```
msgbox(message)
msgbox(message,title)
msgbox(message,title,'icon')
msgbox(message,title,'custom',iconData,iconCmap)
msgbox(...,'createMode')
h = msgbox(...)
```

Trong đó:

'icon' → là {'none','error','help','warn','custom'}  
 iconData → chứa dữ liệu ảnh tạo nên icon  
 iconCmap → Màu dùng cho ảnh  
 'createMode' → {'modal','non-modal','replace'}

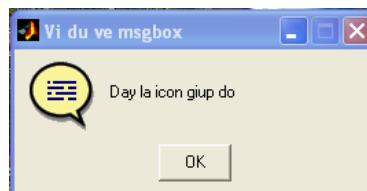
Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP THOẠI DIALOG CỦA WINDOWS

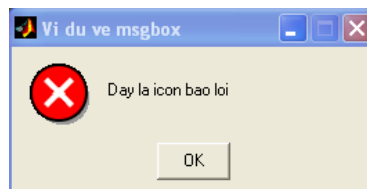
5. **msgbox (tt)**

Ví dụ:

```
msgbox('Day la icon giup do','Vi du ve msgbox','help')
```



```
msgbox('Day la icon bao loi','Vi du ve msgbox','error')
```



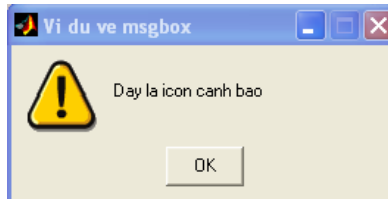
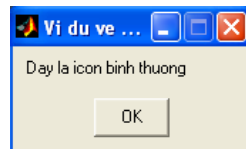
Giảng viên: Hoàng Xuân Dương



## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

5. **msgbox (tt)**

Ví dụ:

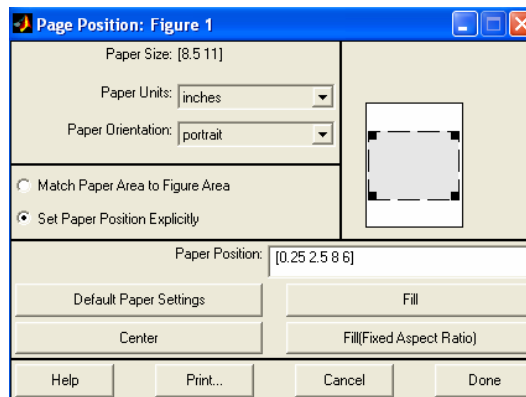
`msgbox('Day la icon canh bao','Vi du ve msgbox','warn')``msgbox('Day la icon binh thuong','Vi du ve msgbox')`

Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

6. **pagedlg**

Cú pháp:

`pagedlg``pagedlg(fig)`

Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

### 7. printdlg

Cú pháp:

```
printdlg
printdlg(fig)
printdlg('-crossplatform',fig)
printdlg('-setup',fig)
```

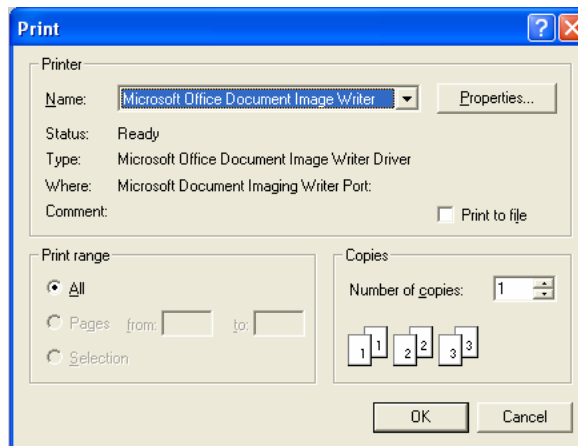
Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

### 7. printdlg (tt)

Ví dụ:

`printdlg(fig)` → in cửa sổ đồ họa được chỉ định

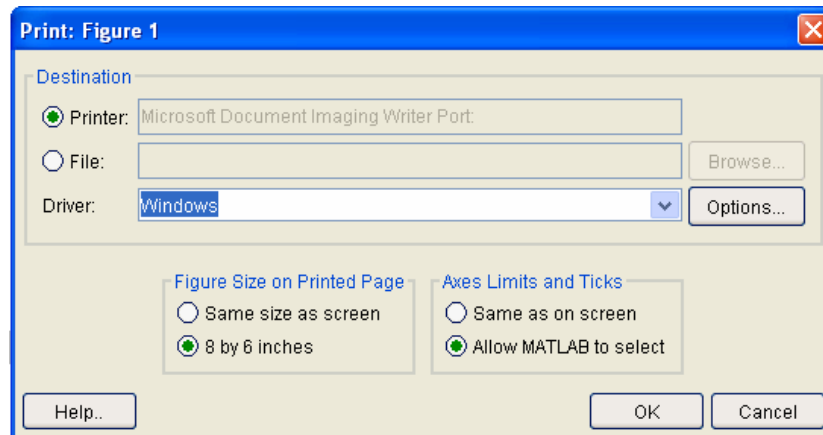


Giảng viên: Hoàng Xuân Dương

## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

7. **printdlg (tt)**

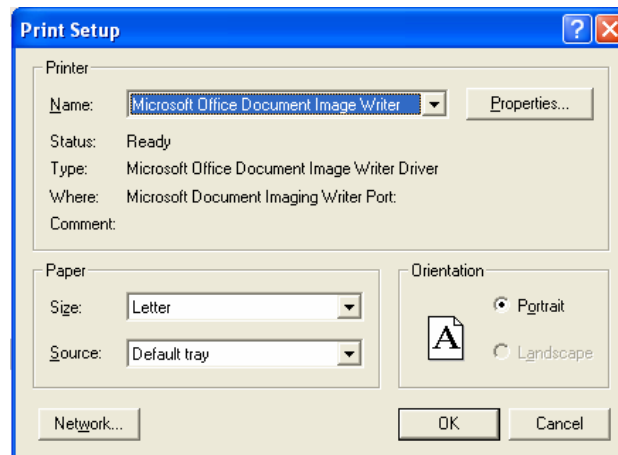
Ví dụ:

`printdlg('-crossplatform',fig)` → sử dụng chuẩn của Matlab

## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

7. **printdlg (tt)**

Ví dụ:

`printdlg('-setup',fig)` → cho phép cài đặt thông số in

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

8. **questdlg**

Cú pháp:

```
button = questdlg('qstring')
button = questdlg('qstring','title')
button = questdlg('qstring','title','default')
button = questdlg('qstring','title','str1','str2','default')
button = questdlg('qstring','title','str1','str2','str3','default')
% hộp thoại có 3 nút 'Yes', 'No', 'Cancel'
% button nhận giá trị trả về
```

Trong đó:

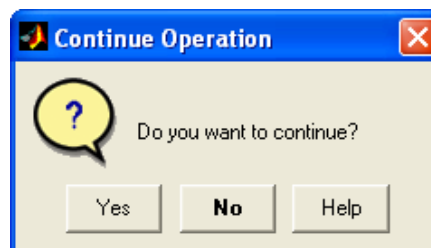
'default' → Nút chọn mặc định {'Yes', 'No', 'Cancel'}  
 'str1','str2','str3' → Tạo các nút nhấn có tên...

## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

8. **questdlg (tt)**

Ví dụ:

```
button = questdlg('Do you want to continue?',...
'Continue Operation','Yes','No','Help','No');
if strcmp(button,'Yes') disp('Creating file')
elseif strcmp(button,'No') disp('Canceled file operation')
elseif strcmp(button,'Help') disp('Sorry, no help available')
end
```



## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

9. **uigetfile**

Cú pháp:

```

uigetfile
uigetfile('FilterSpec')
uigetfile('FilterSpec','DialogTitle')
uigetfile('FilterSpec','DialogTitle',x,y)
[fname,pname] = uigetfile(...)

```

Trong đó:

'FilterSpec' → Lọc chọn các tập tin. Mặc định là \*.m  
 [x,y] → Vị trí xuất hiện hộp thoại  
 [fname,pname] → Trả về tên tập tin và đường dẫn

## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

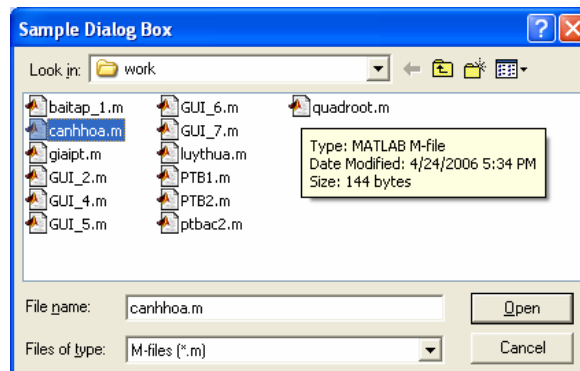
9. **uigetfile (tt)**

Ví dụ:

```

>> [fname,pname] = uigetfile('*.m','Sample Dialog Box')
fname = canhhoa.m
pname = D:\work\

```



## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

## 10. uiputfile

Cú pháp:

```

uiputfile
uiputfile('InitFile')
uiputfile('InitFile','DialogTitle')
uiputfile('InitFile','DialogTitle',x,y)
[fname,pname] = uiputfile(...)

```

Trong đó:

'InitFile' → Hộp thoại hiển thị các file trong thư mục hiện hành xác định bởi 'InitFile'

[fname,pname] → Trả về tên file và đường dẫn ghi file

## IX. CÁC HỘ THOẠI DIALOG CỦA WINDOWS

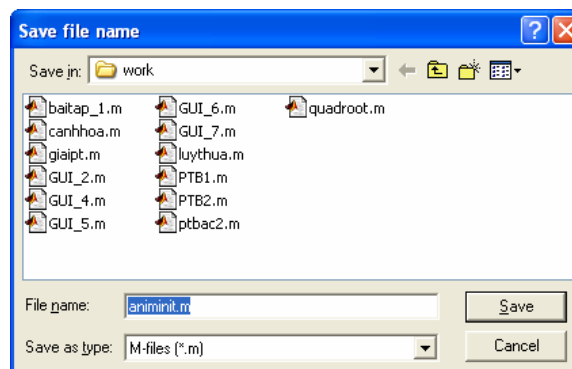
## 10. uiputfile (tt)

Ví dụ:

```

>> [newfile,newpath] = uiputfile('animinit.m','Save file name');
newfile = animinit.m
newpath = D:\work\

```



## IX. CÁC HỘP HỘI THOẠI DIALOG CỦA WINDOWS

### 11. **warndlg**

Cú pháp:

```
h = warndlg('warningstring','dlgname')
```

Ví dụ:

```
warndlg('Pressing OK will clear memory','!! Warning !!')
```

