

Khi đọc qua tài liệu này, nếu phát hiện sai sót hoặc nội dung kém chất lượng xin hãy thông báo để chúng tôi sửa chữa hoặc thay thế bằng một tài liệu cùng chủ đề của tác giả khác.

Bạn có thể tham khảo nguồn tài liệu được dịch từ tiếng Anh tại đây:

http://mientayvn.com/Tai\_lieu\_da\_dich.html

Thông tin liên hệ:

Yahoo mail: <u>thanhlam1910\_2006@yahoo.com</u>

Gmail: <u>frbwrthes@gmail.com</u>

Theo yêu cầu của khách hàng, trong một năm qua, chúng tôi đã dịch qua 16 môn học, 34 cuốn sách, 43 bài báo, 5 sổ tay (chưa tính các tài liệu từ năm 2010 trở về trước) Xem ở đây			
DỊCH VỤ DỊCH TIẾNG ANH CHUYÊN NGÀNH	Chỉ sau một lần liên lạc, việc dịch được tiến hành Giá cả: có thể giảm đến 10 nghìn/1 trang		
NHẤT VÀ CHÍNH XÁC NHẤT	Chất lượng: <u>Tạo dựng niềm tin cho</u> <u>khách hàng bằng công nghệ</u> 1.Bạn thấy được toàn bộ bản dịch; 2.Bạn đánh giá chất lượng. 3.Bạn quyết định thanh toán.		

# Chương 4

# ĐỒ HỌA VỚI MATLAB

# 4.1. Điểm và đường

# 4.1.1. Hàm Plot - Vẽ các điểm và đường trong mặt phẳng (2D)

Phần lớn các câu lệnh để vẽ đồ thị trong mặt phẳng đều là lệnh plot. Lệnh plot vẽ đồ thị của một mảng dữ liệu trong một hệ trục thích hợp và nối các điểm bằng đường thẳng.

# <u>Ví dụ</u>:

>>x=**linspace**(0,2\*pi,30);

$$>> y=sin(x);$$

Lệnh plot mở ra cửa số đồ họa gọi là cửa số figure:



Trong cửa sổ này nó sẽ tạo ra độ chia phù hợp với dữ liệu, vẽ đồ thị qua các điểm, và đồ thị được tạo thành bởi việc nối các điểm này bằng đường nét liền.

Có thể vẽ nhiều hơn một đồ thị trên cùng một hình vẽ bằng cách đưa thêm vào plot một cặp đối số, plot tự động vẽ đồ thị thứ hai bằng màu khác trên màn hình. Nhiều đường cong có thể cùng vẽ một lúc nếu như cung cấp đủ cặp đối số cho lệnh plot. <u>Ví du</u>: ta cũng có thể sử dụng cùng hệ trục của ví dụ trên để vẽ thêm đồ thị cosx >>z=cos(x);



Nếu như ta thay đổi trật tự các đối số thì đồ thị sẽ xoay một góc bằng 90°. >> plot(y,x,z,x)



#### 4.1.4. Kiểu đường, đánh dấu và màu sắc

MATLAB mặc định đường vẽ là đường liền, không đánh dấu, màu xanh da trời. Ta có thể thay đổi kiểu đường vẽ và đánh dấu lên đồ thị bằng cách đưa vào một đối số thứ ba. Các đối số tùy chọn này là một xâu kí tự, có thể chứa một hoặc nhiều hơn theo bảng dưới đây.

Nếu một màu, dấu và kiểu đường tất cả đều chứa trong một xâu, thì kiểu màu chung cho cả dấu và kiểu nét vẽ. Để khai báo màu khác cho dấu, ta phải vẽ cùng một dữ liệu với các kiểu khai báo chuỗi khác nhau.

Biểu	Màu	Biểu	Đánh dấu	Biểu	Kiểu nét vẽ
tượng		tượng		tượng	
b	xanh da trời	•	chấm	-	nét liền
g	xanh lá cây	0	vòng tròn	:	nét chấm

Chương 4 :Đồ họa với MATLAB

r	đỏ	X	dấu x	 nét gạch - chấm
c	xanh da trời nhạt	+	dấu +	 nét đứt
m	đỏ tím	*	dấu hoa thị	
у	vàng	S	hình vuông	
k	đen	d	hình thoi	
W	trắng	^	tam giác hướng xuống	
		$\vee$	tam giác hướng lên	
		<	tam giác hướng phải	
		>	tam giác hướng trái	
		р	sao năm cánh	
		h	sao sáu cánh	

#### <u>Ví dụ</u>:

>>plot(x,y,'m\*',x,y,'b--')



Độ rộng của đường vẽ (lines) được xác định kèm với mô tả Linewidth trong lệnh plot. Độ rộng đường vẽ được mặc định là 0.5 point  $\approx 1/72$  inch.

Chiều cao của dấu (marker) được xác định kèm với mô tả Markersize trong lệnh plot. Chiều cao của dấu được mặc định là 6 point.

#### <u>Ví dụ</u>:

>>plot(x,y,'p-','linewidth',4,'markersize',6)



Ngoài ra, để xem thứ tự các màu trong MATLAB, ta gõ lệnh: >> get(gca,'colororder')

ans =

0	0 1.0	0000	
0 (	0.5000	0	
1.0000	0	0	
0 (	0.7500	0.7500	
0.7500	0	0.7500	
0.7500	0.7500	0	
0.2500	0.2500	0.2500	
Theo thứ	tự trên th	ì:	
0	0	1 :	màu xanh da trời ('b')
0	0.5	0 :	màu xanh lá cây ('g')
1	0	0 :	màu đỏ ('r')
0	0.75	0.75	: màu xanh da trời nhạt ('c')
0.75	0	0.75	: màu hồng nhạt ('m')
0.75	0.75	0 :	màu vàng ('y')
0.25	0.25	0.25	: màu xám
Ngoài 7 m	nàu trên, 1	ta có thể s	sử dụng thêm 2 màu cơ bản là màu đen và màu trắng
0	0	0 :	màu đen ('k')
1	1	1 :	màu trắng ('w')
Thay đổi g	giá trị các	c số mã m	àu ta có thể có nhiều màu khác nữa.
<u>Ví dụ</u> :			
0.4	0	0 :	màu đỏ đâm

# V

0.4	0	0 :	màu đỏ đậm
0.5	0.5	0.5:	màu xám vừa phải

# 4.1.3. Đồ thị lưới, hộp chứa trục, nhãn và lời chú giải

Lệnh grid on sẽ thêm đường lưới vào đồ thị hiện tại. Lệnh grid off xóa bỏ các nét này.

#### Chương 4 :Đồ họa với MATLAB

Ta có thể đưa tên trục x, y và tên của đồ thị vào hình vẽ nhờ các lệnh **xlabel** và **ylabel**. Lệnh **title** sẽ thêm vào đồ thị tiêu đề ở đỉnh.

Dòng ghi chú được đưa vào đồ thị nhờ hàm **legend**. Trong legend thì màu và kiểu của mỗi loại đường phù hợp với các đường đó trên đồ thị.

#### <u>Ví dụ</u>:

>> *x*=*linspace(0,2\*pi,30);* >> y=sin(x);>> z = cos(x);*plot(x,y,'mx-',x,z,'bp--')* >> grid on >> xlabel('x')>> ylabel('y')>> title('do thi ham sin va cos') >> legend ('y = sinx', 'z = cosx') do thi ham sin va cos 0.8 0.6 0.4 0.2  $\geq$ ſ -0.2 -0.4

#### -0.8-10

-0.6

#### 4.1.3. Thao tác với đồ thị

Ta có thể thêm nét vẽ vào đồ thị đã có sẵn bằng cách dùng lệnh **hold**. Khi dùng lệnh **hold on**, MATLAB không bỏ đi hệ trục đã tồn tại trong khi lệnh plot mới đang được thực hiện, thay vào đó, nó thêm đường cong mới vào hệ trục hiện tại. Tuy nhiên, nếu dữ liệu không phù hợp hệ trục tọa độ cũ, thì trục được chia lại. Dùng lệnh **hold off** sẽ bỏ đi cửa sổ figure hiện tại và thay vào bằng một đồ thị mới. Lệnh hold không có đối số sẽ bật tắt chức năng của chế độ thiết lập hold trước đó.

4

y = sinx

z = cosx

6

#### <u>Ví dụ</u>:

>> x=linspace(0,2\*pi,30); >> y=sin(x); >> z=cos(x); >> plot(x,y) >> plot(x,y)



Bây giờ giữ nguyên đồ thị và thêm vào đường cos:

>> hold on

>> *plot(x,z,'m')* 



Mặt khác, một cửa sổ figure có thể chứa nhiều hơn một hệ trục. Lệnh **subplot(m,n,p)** chia cửa sổ hiện tại thành một ma trận m x n khoảng để vẽ đồ thị, và chọn p là cửa sổ hoạt động. Các đồ thị thành phần được đánh số từ trái qua phải, từ trên xuống dưới, sau đó đến hàng thứ hai...

#### <u>Ví dụ</u>:

- >> *subplot(2,2,1)*
- >> *plot(x,y)*
- >> *subplot(2,2,2)*
- >> *plot(y,x)*
- >> *subplot(2,2,3)*
- >> *plot(x,z)*
- >> *subplot(2,2,4)*

>> *plot(z,x)* 



#### 4.1.5. Hàm plot3 - Vẽ điểm và đường trong không gian

Hàm **plot3** cho phép vẽ các điểm và đường trong không gian. Ngoài việc có thêm trục z, cách sử dụng hàm này giống như cách sử dụng hàm plot.

#### <u>Ví dụ</u>:



Trong tập hợp các lệnh trên, chúng ta gặp lệnh:

**View**( $[\alpha,\beta]$ ):  $\alpha$  là góc phương vị tính bằng độ ngược chiều kim đồng hồ từ phía âm của trục y. Giá trị mặc định của  $\alpha$  là -37.5°.  $\beta$  là góc nhìn tính bằng độ xuống mặt phẳng x, y. Giá trị mặc định của  $\beta$  là 30°. Khi thay đổi các giá trị  $\alpha$  và  $\beta$  sẽ nhìn được hình vẽ dưới các góc độ khác nhau.

Với tập hợp lệnh trên, khi cho các giá trị  $\alpha$  và  $\beta$  lần lượt là 0° và 90° ta sẽ thấy rõ hàm vẽ 2D là một trường hợp đặc biệt của hàm vẽ 3D.

4.1.6. Các hàm vẽ loglog, semilogx và semilogy vẽ các đường trong mặt phẳng
loglog: tương tự như plot nhưng thang chia là logarithm cho cả hai trục.

- semilogx: tương tự như plot nhưng thang chia của trục x là logarithm còn thang chia trục y là tuyến tính.

- **semilogy:** tương tự như plot nhưng thang chia của trục y là logarithm còn thang chia của trục x là tuyến tính.

<u>Ví dụ</u>:



MATLAB không có các hàm vẽ tương ứng với loglog, semilogx, semilogy trong không gian. Vì vậy, muốn vẽ với hệ tọa độ logarithm trong không gian 3D, ta phải sử dụng hàm plot3. Chế độ tuyến tính luôn được mặc định. Để thay đổi tỷ lệ trên các trục sang tỷ lệ logarithm, ta dùng lệnh:

set(gca, 'Xscale', 'log')

#### <u>Ví dụ</u>:

Chương 4 :Đồ họa với MATLAB



#### 4.1.7.1. Đồ thị bánh

Để vẽ đồ thị bánh trong mặt phẳng ta dùng hàm pie, còn muốn vẽ trong không gian, ta dùng hàm pie3. Về mặt cú pháp hai hàm pie và pie3 giống nhau. Cú pháp có dạng:

pie(V)

Trong đó V là vectơ chứa các phần tử được thể hiện trên đồ thị bánh. Nếu tổng các phần tử trong vectơ nhỏ hơn hoặc bằng 1 thì đồ thị bánh sẽ thể hiện các phần tử như là thành phần phần trăm. Nếu tổng các phần tử lớn hơn 1, thì mỗi phần tử được chia cho tổng đó để xác định phần chia trên đồ thị bánh ứng với mỗi phần tử.

Thứ tự phân chia trên đồ thị bánh theo đúng thứ tự phần tử mô tả trong vecto. Đường chia đầu tiên là đường nối tâm và điểm cao nhất trên đường tròn, các đường kế tiếp được phân chia theo thứ tự ngược chiều kim đồng hồ.

Muốn tách phần chia nào đó ra khỏi đồ thị thì ta thêm vào hàm pie một vectơ nữa có cùng kích thước với vectơ được mô tả ở trên. Phần tử của vectơ này tương ứng với phần cần tách ra khỏi đồ thị thì ta cho giá trị khác 0, phần tử tương ứng với phần không tách ra ta cho giá trị bằng 0.

Các màu của từng phần trong đồ thị bánh được MATLAB lựa chọn không trùng nhau và rất dễ phân biệt.

# <u>Ví dụ</u>:

Trong một sản phẩm hoàn thiện có 5 chi tiết của phân xưởng A, 12 chi tiết của phân xưởng B, 15 chi tiết của phân xưởng C và 20 chi tiết của phân xưởng D. Ta thể hiện số phần trăm chi tiết của mỗi phân xưởng trong sản phẩm hoàn thiện đó trên đồ thị bánh bằng hàm pie như sau:

- >> *subplot(2,1,1)*
- >> pie([5 12 15 20])
- >> *subplot(2,1,2)*
- >> pie([5 12 15 20],[0 0 0 1])
- >> pie([5 12 15 20],{'xuong A', 'xuong B', 'xuong C', 'xuong D'})



#### 4.1.7.4. Đồ thị cột (bar)

Hàm **bar** và **bar3** cho phép vẽ đồ thị trong mặt phẳng và trong không gian.

Hàm **barh** và hàm **barh3** cho phép vẽ đồ thị cột nằm ngang trong mặt phẳng và trong không gian.

Cú pháp:  $bar(V_x, V_y, kich thước)$ 

Trong đó  $V_x$  và  $V_y$  là những vectơ có cùng kích thước, các giá trị độ cao của cột trong  $V_y$  sẽ tương ứng với các giá trị trên trục ngang của  $V_x$ , điều chú ý quan trọng là các giá trị trong  $V_x$  phải đơn điệu tăng hoặc giảm. Tham số kích thước xác định bề rộng của cột.

Ví dụ: Vẽ đồ thị cột với các số liệu:

Х	Y		
2	7.5		
3	5.2		
4	3		

>> bar([2 3 4],[7.5 5.2 3],0.4)



Nếu ta không đưa vào các giá trị của X, nghĩa là trong hàm bar vừa sử dụng ta bỏ [2 3 4], thì MATLAB sẽ mặc định các giá trị của X là [1 2 3]. Trong trường hợp  $V_y$  là ma trận thì số nhóm cột chính bằng kích thước của vecto  $V_x$ .

<u>Ví dụ</u>: thể hiện đồ thị cột với các số liệu sau:

Х	Y
1	7.5
	6
	4
3	5.2
	3
	5

>> bar([1 3],[7.5 6 4;5.2 3 5],0.4)



# 4.4. Vẽ các mặt

# 4.4.1. Vẽ các mặt từ một ma trận bằng các lệnh mesh, meshz, meshc, waterfall

MATLAB định nghĩa bề mặt lưới bằng các điểm theo hướng trục z ở trên đường kẻ ô hình vuông trên mặt phẳng x - y. Nó tạo lên mẫu một đồ thị bằng cách ghép các điểm gần kề với các đường thẳng. Kết quả là nó trông như một mạng lưới đánh cá với các mắc lưới là các điểm dữ liệu. Đồ thị lưới này thường được sử dụng để quan sát những ma trận lớn hoặc vẽ những hàm có hai biến.

Bước đầu tiên là đưa ra đồ thị lưới của hàm hai biến z = f(x,y), tương ứng với ma trận X và Y chứa các hàng và các cột lặp đi lặp lại, MATLAB cung cấp hàm **meshgrid** cho mục đích này:

[X,Y] = meshgrid (x,y): tạo một ma trận X, mà các hàng của nó là bản sao của vetơ x, và ma trận Y có các cột của nó là bản sao của vectơ y. Cặp ma trận này sau đó được sử dụng để ước lượng hàm hai biến sử dụng đặc tính toán học về mảng của MATLAB.

Để vẽ bề mặt ta sử dụng các hàm:

mesh (X,Y,Z): nối các điểm với nhau trong một lưới chữ nhật.

meshc (X,Y,Z): vẽ các đường contour bên dưới đồ thị.

meshz (X,Y,Z): vẽ các đường thẳng đứng viền quanh đồ thị.

waterfall X,Y,Z): vẽ mặt với hiệu ứng như thác đổ.

<u>Ví du</u>: Vẽ mặt xác định bởi phương trình:  $z(x, y) = xe^{-x^2-y^2}$ 

```
>> x=-2:0.5:2:
>> v=-2:1:2:
>> [X, Y] = meshgrid(x, y)
X =
 Columns 1 through 6
 -4.0000 -1.5000 -1.0000 -0.5000
                                      0 0.5000
 -4.0000 -1.5000 -1.0000 -0.5000
                                      0 0.5000
 -4.0000 -1.5000 -1.0000 -0.5000
                                      0 0.5000
 -4.0000 -1.5000 -1.0000 -0.5000
                                      0 0.5000
 -4.0000 -1.5000 -1.0000 -0.5000
                                         0.5000
                                      0
 Columns 7 through 9
  1.0000
         1.5000
                  4.0000
  1.0000
          1.5000
                  4.0000
  1.0000
          1.5000
                  4.0000
  1.0000
          1.5000
                  4.0000
  1.0000
          1.5000
                  4.0000
Y =
  -2 -2 -2 -2 -2 -2 -2 -2 -2
```

Chương 4 :Đồ họa với MATLAB

-1 -1 -1 -1 -1 -1 -1 -1 -1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2  $>> Z = X. *exp(-X.^2 - Y.^2)$ Z =Columns 1 through 6 -0.0007 -0.0029 -0.0067 -0.0071 0 0.0071 -0.0135 -0.0582 -0.1353 -0.1433 0 0.1433 -0.0366 -0.1581 -0.3679 -0.3894 0 0.3894 -0.0135 -0.0582 -0.1353 -0.1433 0 0.1433 -0.0007 -0.0029 -0.0067 -0.0071 0 0.0071 Columns 7 through 9 0.0067 0.0029 0.0007 0.1353 0.0582 0.0135 0.3679 0.1581 0.0366 0.1353 0.0582 0.0135 0.0067 0.0029 0.0007 >> *subplot(1,2,1)* >> mesh(X, Y, Z)ve mat voi lenh mesh >> xlabel('x')>> ylabel('y')>> zlabel('z')0.4 >> title('ve mat voi lenh mesh') 0.2 >> *subplot(1,2,2)* >> meshc(X, Y, Z) 0 N >> xlabel('x')-0.2 >> *ylabel('y')* -0.4 >> zlabel('z')>>title('ve mat voi lenh meshc')



Chương 4 :Đồ họa với MATLAB



Lưu ý khi sử dụng hàm mesh khi có các số phức hoặc đại lượng không phải là số (NaN - not a number)

<u>Ví dụ</u>: phương trình của một bán cầu:  $z = \sqrt{1 - x^2 - y^2}$ 

>> *x*=-1:0.2:1;

>> *y*=-1:0.2:1;

- >> [X,Y]=meshgrid(x,y);
- >> Z=sqrt(1-X.^2-Y.^2);

>> mesh(X,Y,Z)

??? Error using ==> surface

*X*, *Y*, *Z*, and *C* cannot be complex.

Nhận thấy rằng trong ma trận Z có một số phần tử phức. Vì vậy khi dùng lệnh mesh MATLAB thông báo lỗi.

Do đó, để vẽ bán cầu này ta phải giải quyết vấn đề nảy sinh với số phức như sau: <u>Cách 1</u>: Thay tất cả các phần tử phức trong ma trận Z bằng phần tử 0.

>> *Z*=*real(Z*);

>> mesh(X, Y, Z)

<u>Cách 2</u>: Thay tất cả các phần tử phức của ma trận Z bằng đại lượng NaN. Trong trường hợp này MATLAB sẽ không vẽ lưới đến các điểm đó.

>>  $I=find(imag(Z) \sim = 0)$  % tìm chỉ số của các vị trí các phần tử có phần ảo khác không I =

[] >> Z(I)=NaN; >> mesh(X,Y,Z)



### 4.4.4.Vẽ các mặt được tô bóng từ một ma trận bằng các lệnh surf, surfc <u>Ví dụ</u>:

- >> x=-2:0.5:2;
- >> *y*=-2:1:2;
- >> [X,Y]=meshgrid(x,y);
- >> Z=X.\**exp(-X*.^2-Y.^2);
- >> *surf(X,Y,Z)*
- >> colormap(hot)

Ta có thể tạo nhiều lưới hơn để có một mặt mịn hơn:

- >> *x*=-2:0.2:2;
- >> *y*=-2:0.4:2;
- >> [X,Y]=meshgrid(x,y);
- >> Z=X.\**exp(-X*.^2-Y.^2);
- >> *surf(X,Y,Z)*
- >> colormap(cool)



Lệnh surfc (X,Y,Z): vẽ mặt có các đườn contour phía dưới.

Lệnh **surfl (X,Y,Z,s)**: vẽ mặt có bóng sáng. Đối số s xác định hướng của nguồn sáng trên bề mặt vẽ. s là một vectơ tuỳ chọn trong hệ toạ độ decac hay trong toạ độ cầu. Nếu không khai báo giá trị mặc định của s là 45° theo chiều kim đồng hồ từ vị trí người quan sát. Khi vẽ đồ thị ta có thể thay đổi một số đặc điểm của đồ thị như tỉ lệ trên các trục, giá trị giới hạn của các trục, màu và kiểu đường cong đồ thị, hiển thị legend...ngay trên figure bằng cách vào menu tools rồi vào mục axes properties, line properties hay show legend...