CHƯƠNG 3:

LẬP TRÌNH TRONG MATLAB

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^{-323} 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('Nhap vao mot chuoi','s');
```


I. PHẦN TỬ CƠ BẢN

2. Biến string:

Các lệnh với biến string:

Hàm	Ý nghĩa
char	Tạo mảng ký tự
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

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

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
```

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
asin(x)	$\in[-\pi/2,\pi/2]$ khi $x \in [-1,1]$
acos(x)	$\in[0,\pi]$ khi $x \in [-1,1]$
atan(x)	khi $x \in [-\pi/2,\pi/2]$

Đổi radian sang độ và ngược lại:

```
angle_degrees=angle_radians*(180/pi)
angle_radians=angle_degrees*(pi/180)
```


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

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

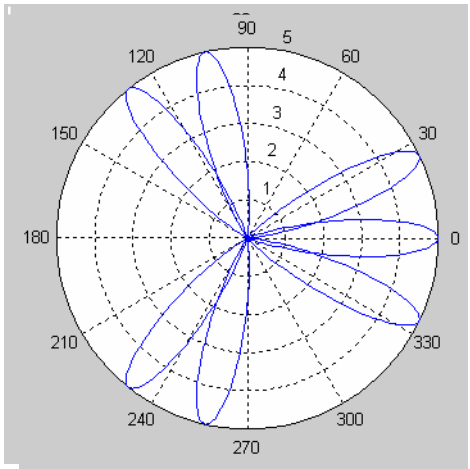
```
% M-file script tao ra 4 hinh canh hoa
theta=-pi:0.01:pi;
rho(1,:)=2*sin(5*theta).^2;
rho(2,:)=cos(10*theta).^3;
rho(3,:)=sin(theta).^2;
rho(4,:)=5*cos(3.5*theta).^3;
for i=1:4
    polar(theta,rho(i,:))
    pause
end
```

Trong command window:

```
>> help canhhoa
M-file script tao ra 4 hinh canh hoa
```



```
>> 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
```



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

Qui 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'

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
```


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);
```

III. CÁC DẠNG FILE

2. Hàm và tạo hàm trong Matlab (tt)

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

```
disp('Chuong trinh giai phuong trinh bac 2: ax^2+bx+c=0');
a=input('Nhap a: ');
b=input('Nhap b: ');
c=input('Nhap 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);
```


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
Chuong trinh giai phuong trinh bac 2: ax^2+bx+c=0
Nhap a: 1
Nhap b: -3
Nhap c: 2
Nghiem cua phuong trinh:
x1=2.000000
x2=1.000000
```

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;

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

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
```


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)

IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

2. Các phép toán logic (tt)

Ví dụ:

```
>> b=[1 1 0; 1 0 1]
>> a=[0 1 0; 0 0 1]
>> a&b
ans =     0    1    0
          0    0    1
>> a|b
ans =     1    1    0
          1    0    1
>> ~a
ans =     1    0    1
          1    1    0
>> 1 | 0 & ~ 1
```


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 bất cu 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

IV. BIỂU THỨC QUAN HỆ VÀ LOGIC

3. Các hàm quan hệ và logic (tt)

Hàm	Ý nghĩa
finite(x)	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
isempty(x)	= 1 nếu x rỗng, ngược lại = 0
isstr(x)	= 1 nếu x là một chuỗi, ngược lại = 0
strcmp(y1,y2)	So sánh 2 chuỗi, =1 nếu 2 chuỗi giống hệt nhau, ngược lại =0. Phân biệt hoa-thường, dấu cách, đầu dòng

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
```

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
```


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
```

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
```


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
```

V. CẤU TRÚC ĐIỀU KHIỂN

2. Lệnh switch case:

Chọn nhiều trường hợp

```
switch biểu thức (vô hướng hay chuỗi)
        case trị_1
                Các phát biểu 1
        case trị_2
                Các phát biểu 2
        ……
        otherwise
                Các phát biểu khác
end
```


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
```

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
```


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
```

V. CẤU TRÚC ĐIỀU KHIỂN

4. Lệnh for:

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

Ví dụ 1:

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


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
```

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

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
```

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 !
```


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
```

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
```


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

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

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

VI. BÀI TẬP:

11. Viết một hàm tinhtong.m có:

    Nhận vào giá trị n

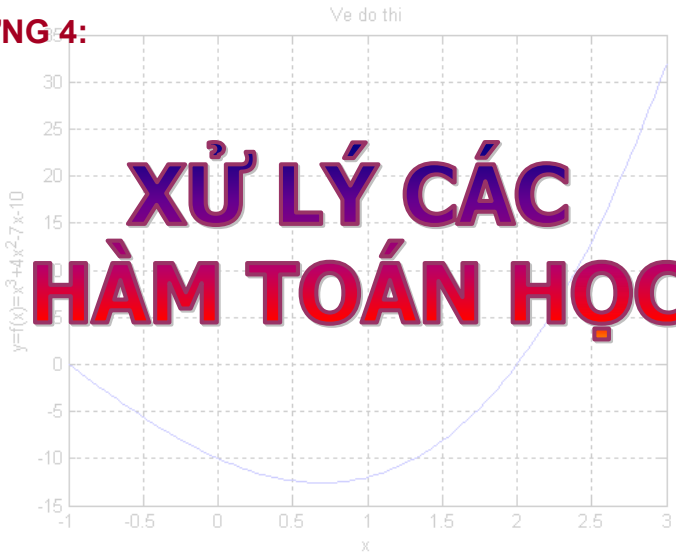
    Trả về giá trị tổng các tích 2 số liên tiếp từ 1 đến n

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

    $1^2 + 2^2 + \ldots + n^2$

    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!)

CHƯƠNG 4:



XỬ LÝ CÁC HÀM TOÁN HỌC

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]

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

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
```


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
```

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

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

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)
```


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
```

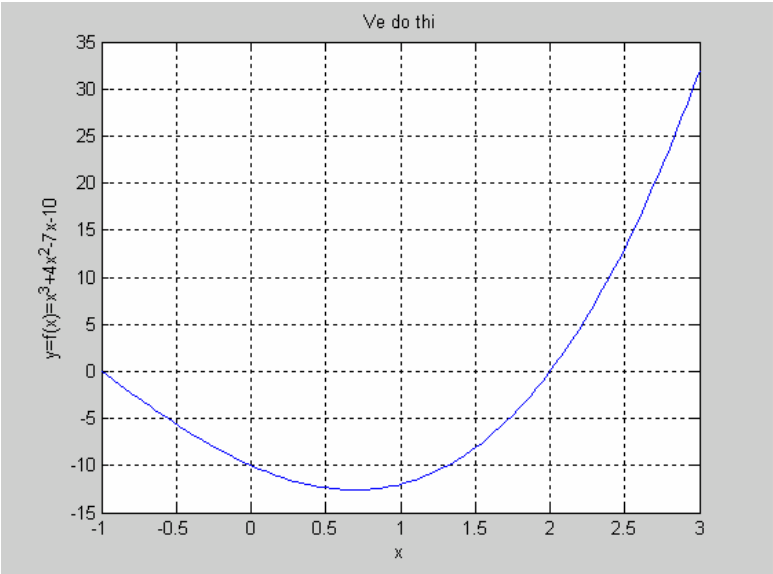
I. ĐA THỨC:

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');
```

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

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 = [ ]
```

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

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

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

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

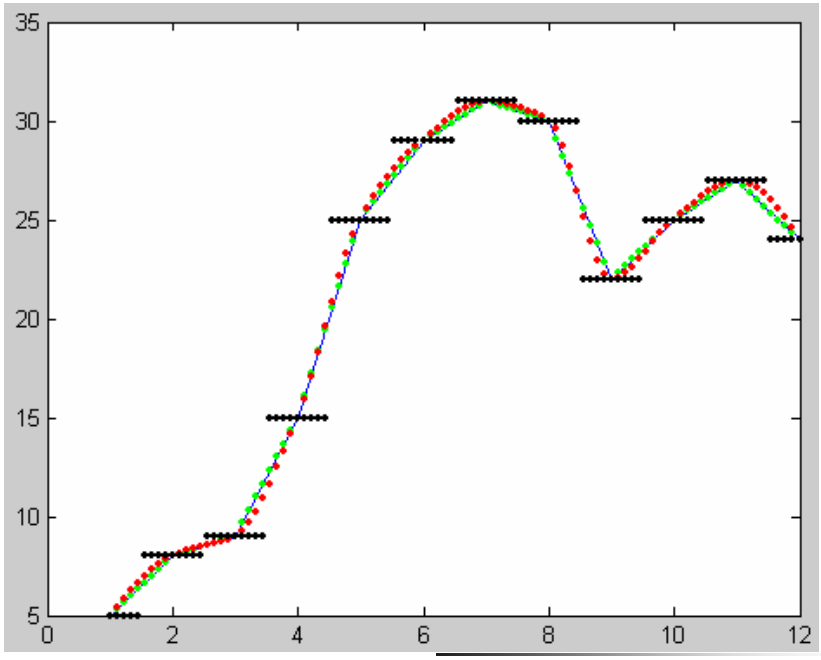
extrap: dùng khi ngoại suy, các giá trị ngoài tầm x, giá trị trả về là extrapval

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.')
```



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:

  Zi=interp2(X,Y,Z,Xi,Yi)

  Zi=interp2(Z,Xi,Yi)

  Zi=interp2(Z,ntimes)

  Zi=interp2(X,Y,Z,Xi,Yi,'method')

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

Zi là giá trị dữ liệu được nội suy ở giá trị Xi,Yi

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
```


II. PHÉP NỘI SUY:

3. Nội suy nhiều chiều:

Vi=interp3(X,Y,Z,V,Xi,Yi,Zi)

Vi=interpn(X1,X2,X3,…,V, Y1, Y2, Y3,…)

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')
```


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
```

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
```