Bộ Giáo Dục và Đào Tạo
Trường Đại Học Nông Lâm, TP. Hồ Chí Minh

ỨNG DỤNG COMPUTER TRONG CÔNG NGHỆ HÓA HỌC VÀ THỰC PHẨM

(Dùng cho kỹ sư các ngành CN Hoá học, Thực phẩm, Sinh học)

Biên soạn: PGS.TS. Trương Vĩnh

TP. Hồ Chí Minh – 2006

PHẦN A: MATLAB

MỞ ĐẦU

Trong kỹ thuật việc tính toán bình thường nhiều lúc gặp khó khăn ví dụ giải phương trình của đa thức bậc cao, hàm phi tuyến, phương trình và hệ phương trình vi phân. Lúc này áp dụng phương pháp số là thích hợp. Đặc biệt khi mô tả quá trình theo thời gian hay không gian thì hệ phương trình vi phân sẽ được giải theo biến thời gian hoặc không gian đó, bài toán trở nên phức tạp mà chỉ có phương pháp số mới phù hợp.

Giáo trình bao gồm hai phần là Matlab và Excel.

Matlab là một trong những ngôn ngữ lập trình giúp giải các bài toán trên bằng phương pháp số. Dùng Matlab có nhiều thuận lợi hơn do đồ họa đã được tích hợp sẵn, các thủ tục giải toán cũng được viết sẳn giúp rút ngắn thời gian lập trình tính toán. Do vậy, giáo trình này chọn Matlab để giới thiệu với sinh viên. Việc giới thiệu mang tính chất cơ bản, chủ yếu tập trung giải phương trình và hệ phương trình vi phân mà trong kỹ thuật sẽ ứng dụng nhiều.

Với Excel, giáo trình chủ yếu giới thiệu một số chức năng quan trọng như giải hệ phương trình, tìm cực trị có điều kiện dùng hàm Solver giúp sinh viên làm quen với xử lý số liệu. Riêng phần Data Analysis sẽ thực hành trong khuôn khổ môn thống kê nên không trình bày trong môn học này.

Tác giả

Chương 1 GIỚI THIỆU MATLAB

1.1 Phần mềm Matlab

Matlab là một phần mềm dùng để lập trình giống như các phần mềm Pascal, Fortran, Visual Basic hay C. Đây là phần mềm sử dụng ngôn ngữ Matlab để giải các bài tóan theo phương pháp số. Matlab là một ngôn ngữ lập trình bậc cao với hàng trăm hàm số xây dựng sẵn cho việc tính tóan kỹ thuật, vẽ đồ thị và phim họat hình. Matlab có nghĩa là Matrix Laboratory.

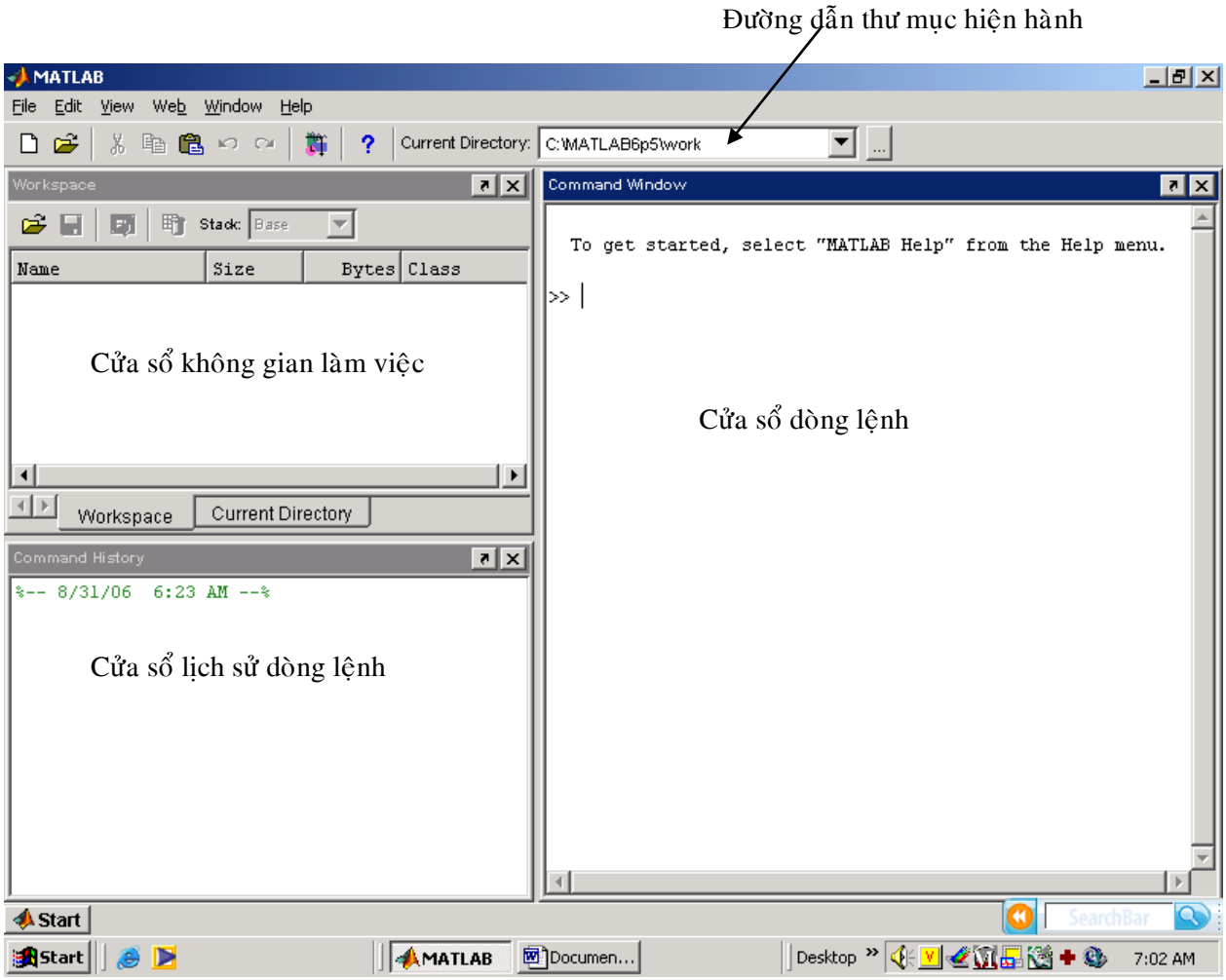
Matlab xây dựng chủ yếu trên nền tảng của ma trận và tóan tử của ma trận. Nó bao gồm các hàm số sẵn có và do người dùng tự viết. Matlab có thể giao diện với các phần mềm C và Fortran. Đối với phần mềm chuyên nghiệp, Matlab cung cấp một hộp công cụ (Toolbox) trong đó có thể xử lý thống kê, kiểm sóat hệ thống, tối ưu hóa,..., đặc biệt là phần tóan học biểu tượng có thể tính tóan trên ký tự. Ví dụ ta viết $(x+1)^2$ thì Matlab tính ra (x^2+2x+1). Matlab là sản phẩm của tổng công ty MathWorks và có thể xem thông tin trên website: http://www.mathworks.com.

1.2 Sử dụng Matlab

1.2.1 Màn hình Matlab

Khi khởi động, màn hình Matlab gồm có các cửa sổ chính sau đây (Hình 1.1):

- *Cửa sổ dòng lệnh* (command window): đây là cửa sổ chính có dấu nhắc >>. Tất cả các lệnh đều có thể đánh ở đây và **Enter** để thực hiện lệnh. Ngòai ra, khi thực hiện một file đã viết, tất cả các dòng lệnh sẽ được thực hiện và cho kết qủa ở đây. Điều này có thể thấy trên cửa sổ này sau khi chạy một chương trình. Ví dụ từ dấu nhắc, ta đánh dòng lệnh A = [1 2 3; 4 5 6] và **Enter**, trên cửa sổ dòng lệnh xuất hiện kết quả ma trận A (Hình 1.2).
- *Cửa sổ không gian làm việc* (workspace): ở đây sẽ tóm tắt đặc tính của các mảng (array) của chương trình vừa thực hiện. Ở ví dụ trên, cửa sổ không gian làm việc cho biết ma trận A có kích thước 2x3 chiếm 48 byte (Hình 1.2).
- *Cửa sổ lịch sử dòng lệnh* (command history): ghi lại các lệnh hoặc file đã thể hiện trên cửa sổ dòng lệnh. Ở ví dụ trên, cửa sổ này ghi lại dòng lệnh A = [1 2 3; 4 5 6] thực hiện ngày 31/8/06.

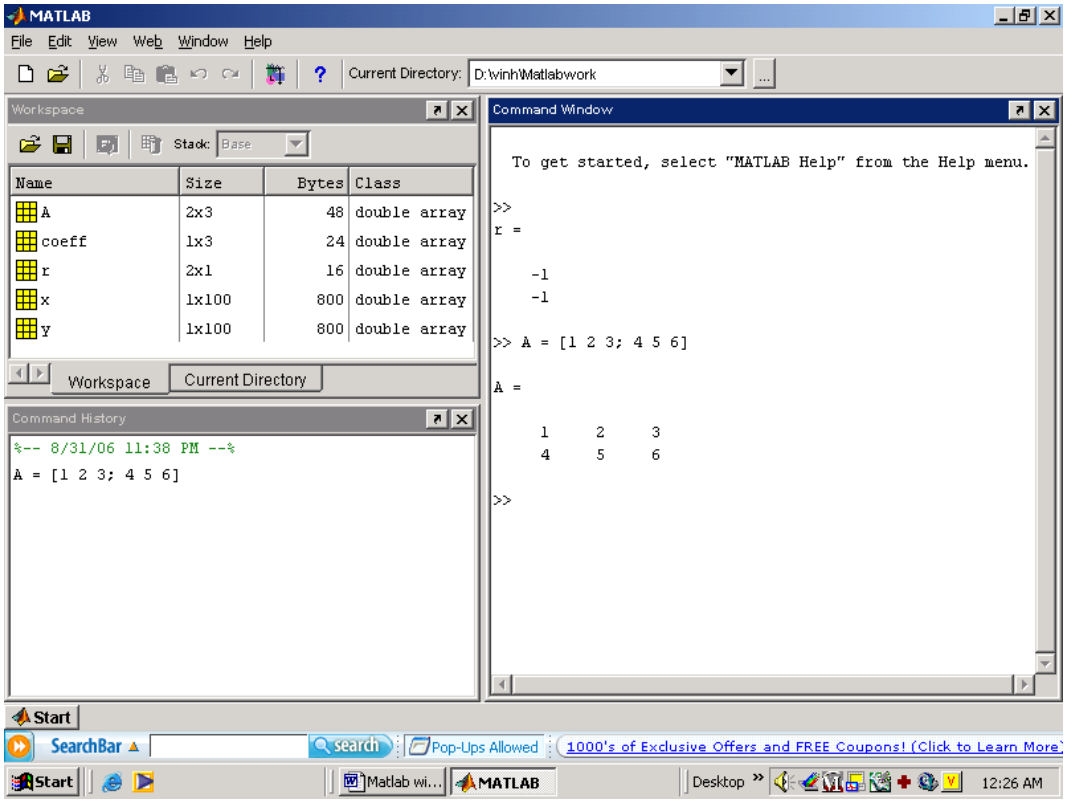


Hình 1.1: Các cửa sổ của màn hình Matlab khi khởi động.

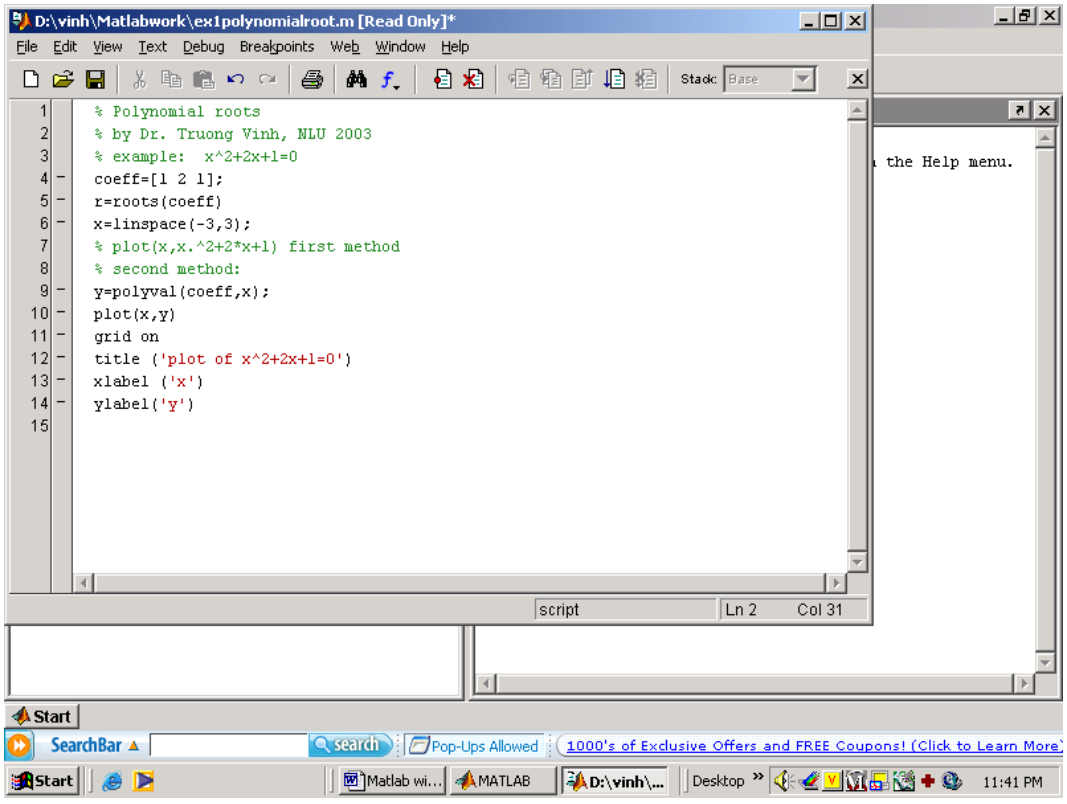
- *Cửa sổ lập trình và hiệu chỉnh* (edit window): đây là cửa sổ dùng để lập trình hoặc hiệu chỉnh chương trình (Hình 1.3). Khi bắt đầu một chương trình mới thì ta vào **File** và **New**. Khi hiệu chỉnh một chương trình đã viết thì vào **File** và **Open**. File chương trình do chúng ta tự viết khi lưu trữ sẽ có dạng *.m, còn gọi là "**m – file**". Ví dụ tên file là *hephuongtrinh* thì dạng file là "hephuongtrinh.m".
- *Cửa sổ đồ thị*: cửa sổ này xuất hiện khi trong chương trình ta có viết lệnh xuất đồ thị (ví dụ lệnh **plot(x,y)**, Hình 1.4). Người sử dụng có thể xuất ra bao nhiêu đồ thị cũng được, miễn là bộ nhớ computer cho phép.

1.2.2 Làm việc với Matlab

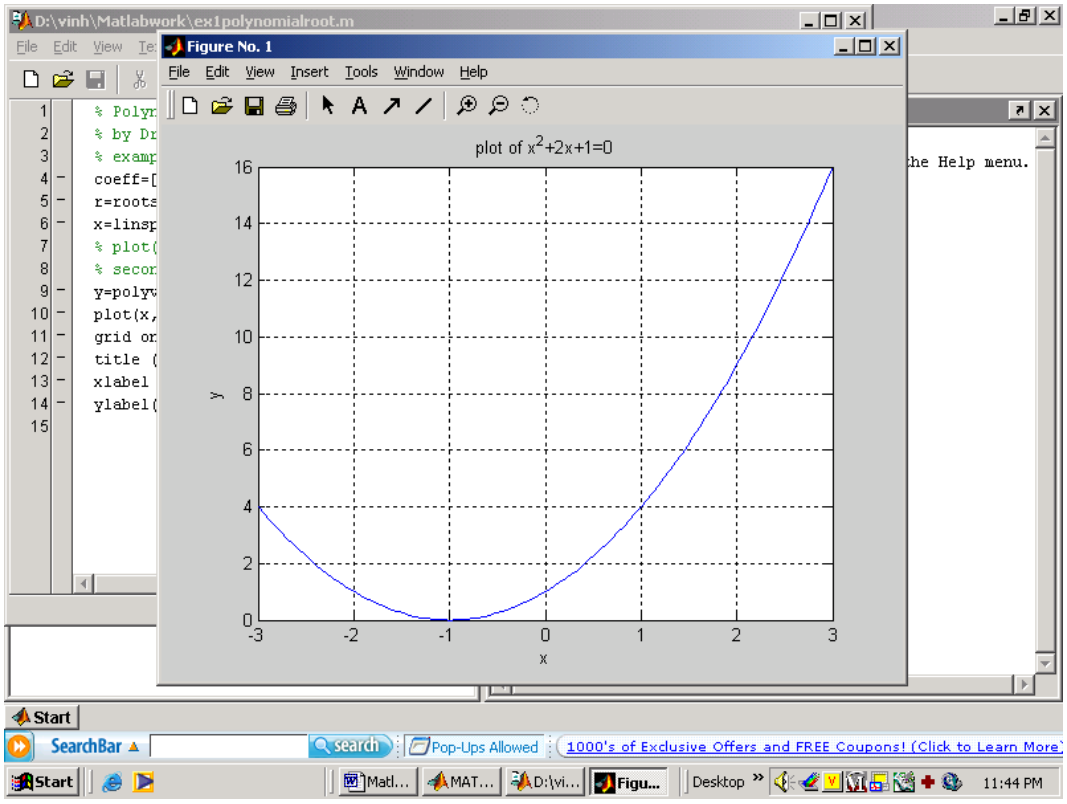
- *Cài đặt đường dẫn*: khi mới khởi động Matlab có đường dẫn mặt định (Hình 1.1)



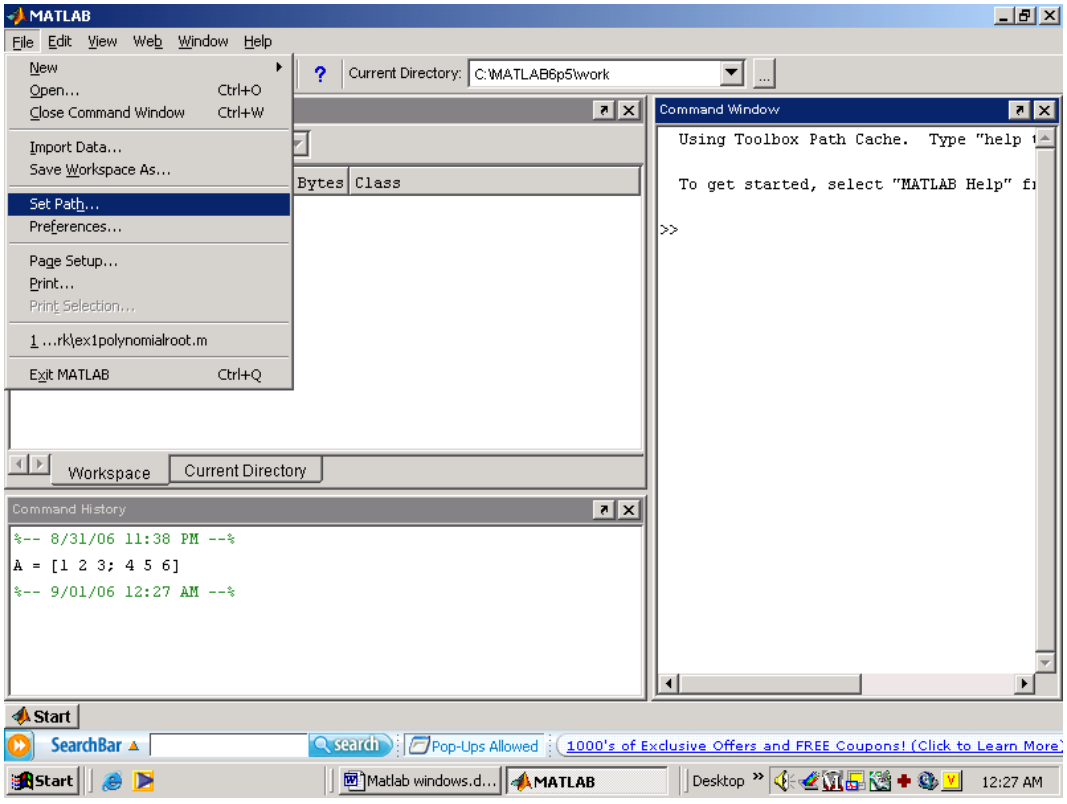
Hình 1.2: Các cửa sổ sau khi thực hiện lệnh



Hình 1.3: Cửa sổ lập trình và hiệu chỉnh.



Hình 1.4: Cửa sổ đồ thị, là kết qủa của việc thực hiện chương trình.



Hình 1.5: Cài đặt đường dẫn mới.

là C\Matlab6p5\work. Ta có thể đổi lại đường dẫn nơi lưu các file của chương trình bằng cách vào **File** và **Set Path** (Hình 1.5) rồi **Add Folder**.

- *Lọai dữ liệu*: lọai dữ liệu cơ bản trong Matlab là mảng (array) bao gồm nhiều đối tượng số liệu khác nhau: số nguyên, số thực, matrận, chuổi kí tự, cấu trúc, tế bào. Hầu như mọi trường hợp Matlab tự động đặt lọai số liệu cho biến.
- *Kích thước ma trận hoặc mảng*: được tự động nhận biết mà không cần khai báo. Nếu muốn biết kích thước của một ma trận nào đó thì dùng lệnh **size**.
- *Chữ hoa và chữ thường*: với Matlab chữ hoa khác chữ thường, ví dụ **a** khác **A**. Hầu hết các lệnh và phương trình có sẵn của Matlab được viết chữ thường. Ta có thể đóng hoặc mở sự khác biệt này bằng lệnh **casesen**.
- *Màn hình xuất kết qủa*: ngòai đồ thị được xuất trên màn hình riêng, các kết qủa bằng số sẽ xuất hiện trên cửa sổ dòng lệnh nếu sau dòng lệnh có dấu ";". Sau đây là các tính chất dùng kiểm sóat màn hình xuất kết qủa:
    - Xuất kết qủa từng trang: Matlab thường xuất kết qủa ra cùng một lúc nên nhiều khi chúng ta không đọc hết được. Muốn xuất từng trang cho dễ đọc dùng lệnh **more on** ở sau dấu nhắc trước khi thực hiện chương trình.
    - Dạng chữ số: chữ số kết qủa có thể định dạng bằng nhiều kiểu nhờ lệnh **format**. Ví dụ: tỉ số 170/7 có nhiều kiểu định dạng như sau:

Format short	24.2857
Format short e	2.4286e+001
Format long	24.28571428571430
Format long e	2.428571428571430e+0.1

- *Lọai file*: Matlab có 3 lọai file lưu trữ thông tin:
    - *M-file*: là tập tin ASCII dạng kí tự (text) gồm 2 lọai là *file chữ (script file)* và *file hàm (function file)*. Đây là các file mà đuôi là ".m", trong đó file chữ là chương trình ta viết gồm tập hợp các lệnh cần thiết, còn file hàm là hàm số do ta lập ra. Hàm số sẳn có của Matlab cũng có dạng M-file. Phần này sẽ đươc giới thiệu kỹ trong chương "Lập trình trong Matlab".
    - *Mat-file*: là tập tin dữ liệu nhị phân (binary data-file) sẽ được lưu trữ khi dùng lệnh **save**. Mat-file có đuôi là ".mat" sẽ được tải vào Matlab bằng lệnh **load**.
    - *Mex-file*: là file có đuôi ".mex" dùng tương thích với ngôn ngữ Fortran và C.

1.2.3 Các lệnh chung cần nhớ

- *Lệnh trợ giúp*:
    - **Help**: liệt kê các chủ đề trợ giúp
    - **lookfor** *string*: liệt kê các chủ đề trợ giúp chứa đựng *string*.
    - **demo**: thực hiện chương trình demo.
- *Thông tin về không gian làm việc*:
    - **whos**: liệt kê các biến hiện hành trong không gian làm việc cùng với kích thước của chúng.
    - **what:** liệt kê các file ".m", ".mat", và ".mex" trong thư mục.
    - **clear:** xóa không gian làm việc, tất cả các biến bị xóa.
    - **clear** x y z**:** chỉ xóa x, y và z.
    - **clear all :** xóa các biến và các phương trình.
    - **clc :** xóa cửa sổ dòng lệnh và lịch sử dòng lệnh.
- *Thông tin chung* :
    - **clock** : cho biết ngày và giờ đồng hồ dưới dạng một vector

ví dụ : đánh lệnh clock và Enter ta được :

ans =

1.0e+003 *

2.0060 0.0090 0.0060 0.0060 0.0220 0.0143

tức ngày 06/09/06 vào lúc 6giờ 22phút 14,3giây.

- **date** : cho biết ngày dưới dạng chuỗi

ví dụ : >> date

ans =

06-Sep-2006

- Dừng chương trình :
    - **^ C (Ctrl + C)** : dừng lệnh hiện hành
    - **quit** : thoát khỏi Matlab
    - **exit** : giống như **quit**

Chương 2 LÀM QUEN VỚI CHƯƠNG TRÌNH MATLAB

Phần này gồm các bài tập giúp chúng ta làm quen với môi trường của Matlab

2.1 Bài 1: Vào, ra Matlab, thực hiện các phép tính cơ bản trên cửa sổ dòng lệnh

Các toán tử số học của Matlab : + - * / ∧ (Mũ)

Các hàm cơ bản : sin, cos, log

Ví du 2.1: Đánh vào cửa sổ dòng lệnh các cú pháp sau và Enter

```
>> 2+2
>> x = 2+2
>> y = 2 ^2 + log (pi ) * sin (x)
>> t = acos (-1)
>> format short
>> t
>> quit
```

Bài tập tại lớp: Viết dưới dạng cú pháp Matlab các bài tóan sau đây

1. Dùng toán tử số học

$$\begin{cases} \dfrac{2^5}{2^5-1} \qquad 3\dfrac{\sqrt{5}-1}{(\sqrt{5}+1)^2}-1 \\ Area = \pi r^2 \text{ với } r = \pi^{1/3} - 1 \end{cases}$$

2. Dùng hàm mũ và logarit

Tính : e^3, ln (e^3), $\log_{10}$ (e^3), $\log_{10}$ (10^5), $e^{\pi\sqrt{163}}$

Giải: $3^x = 17$

3. Hàm lượng giác

$\sin\dfrac{\pi}{6}$, $\cos\pi$, $\tan\dfrac{\pi}{2}$

$\sin^2\dfrac{\pi}{6}+\cos^2\dfrac{\pi}{6}$

Bảng 2.1 : Tóm tắt cách viết cú pháp

Hàm số	**Cú pháp**
e^x	exp (x)
ln x	log (x)
$\log_{10}$ x	log 10(x)
(arc)sin	(a)sin
cos	cos
tan	tan
cotan	cot

4. số phức

$i = \sqrt{-1}$

tính : $\frac{1+3i}{1-3i}$ kết quả -0,8000 + 0,6000i

$\exp(i * pi/4)$ kết quả 0,7071 + 0,7071i

2.2 Bài 2: tạo mảng (array) và vector, tính toán trên mảng và vector

(.*)nhân , (./) chia , (.∧) mũ

Ví dụ:

```
>>x = [1 2 3]            %vector hàng
>>y = [2; 1; 5]          %vector cột
>> z =[2 1 0]; a=x+z
>> b=x.*z
>>c=2*a
>>t=linspace (0,10,5)   %tạo 5 phần tử giữa 0 và 10 để có vector t
```

Bài tập tại lớp:

1. Tính giá trị hàm số y = 0,5 x –2 với x = [0, 1. 5, 3, 4, 5, 7, 9, 10]

```
>>x = [ 0, 1. 5, 3, 4, 5, 7, 9, 10]; y = 0,5*x –2
```

2. Nhân, chia, mũ: cho vector t

```
>>t = 1:10;
>>x = t.* sin(t)
>>y = (t-1)./(t+1)
>>z = sin(t.^2)./(t.^3)
```

3. Dãy số

Tính tổng $S = 1+r+r^2 + ...+r^n$; so sánh 3 trường hợp n = 10, 50 và 100

```
>>r = 0.5;
>>n = 0:10; x = r.^n;S1 = sum (x)  %kết qủa 1.9990
>>n = 0:50; x = r.^n; S2 = sum (x)  %kết qủa 2.0000
>>n = 0:100; x = r.^n; S3 = sum (x) %kết qủa 2
```

Về lý thuyết ta có $S = (1 - r^{n+1})/(1-r)$. Khi $r < 1$ và $n \to \infty$ thì $S \to 1/(1-r)$ và $S = 2$ khi $r = 0.5$. Ví dụ trên cho thấy $S = 2$ khi $n = 100$.

2.3 Bài 3: Tạo hình vẽ đơn giản

Các lệnh axis, xlabel, ylabel, title.

1. Vẽ vòng tròn có nhãn toạ độ $x = r\cos\theta$, $y = r \sin\theta$, $0 \le \theta \le 2\pi$,

dùng 100 điểm để vẽ:

```
>> r = 2
>> theta = linspace (0, 2*pi, 100);
>> x = r.*cos(theta);
>> y= r.*sin(theta);
>> plot(x,y)
>> axis ('equal'); % hai trục có tỉ lệ xích giống (nếu không thì sẽ có ellipse)
>> xlabel ('x')
>> ylabel ('y')
>> title ('vong tron bk 2')
```

2. Bài tập tại lớp:

1. Vẽ $z = \sin x$, $0 \le x \le 2\pi$ dùng 100 điểm

Chú ý: plot (x,y,'+') = vẽ đường cong bằng dấu chấm '+'

2. $y = e^{-0,4\,x}$. sinx
3. Vẽ không gian plot3(x, y, z)
   Hàm x(t) = sin (t), y(t) = cos(t), z(t) =t
4. On- line help: help phot, help graphid
5. Vẽ nhiều đường cong trên 1 đồ thị

Cách 1: plot (x, y, x,z; '_,_')

Cách 2: dùng lệnh hold on.

Các lệnh, làm 2D: photo page 155 –159

Vẽ 3D. Surf; bề mặt cong

1. Lệnh mesh : vẽ bề mặt 3 chiều
   Lệnh mesh : vẽ bề mặt contour
2. Lệnh comet 3 vẽ đường animater
3. Lệnh contour 3 vẽ contour → tìm cực trị

Bài 4: Lập trình đơn giản (script file)

bài tập xem lại các phần bài 3 và vẽ hình

vẽ vòng tròn có bán kính khác nhau, vẽ nhiều vòng tròn

Bài 5: Lập một hàm số

Ví dụ: hàm vẽ vòng tròn (ellipse)

```
function [x,y]= circle fn(r)
        hoặc = vongtron ( r )
theta = linspace(0,2 *pi , 100);
x = r * cos (theta);
y = r * sin (theta);
plot(x,y)
```

Save file tên : vongtron.m

Cách dùng: trên "command window" đánh "vongtron' và Enter hoặc thực hiện lệnh 'vongtron' trong một '.m' file bất kỳ.

Chương 3 Tính toán trên Matlab

3.1 Mảng (array)

Mảng là một tập hợp số liệu mà ta muốn tính toán. Ví dụ: X = [0, 1, 3, 5] là một mảng có 4 số liệu, khác với Y = [5, 3, 1, 0] vì thứ tự khác nhau.

3.1.1 Các cách khai báo mảng

- Đánh tất cả các phần tử trong ngoặc vuông: X = [0, 1, 3, 5]
- Đánh số đầu và cuối nếu khỏang cách giữa các số như nhau: Y = [1:0.5:5] là tập hợp các số: 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, và 5.

3.1.2 Tính toán dùng số liệu của mảng

Khi tính toán dùng số liệu mảng, các toán tử nhân, chia và mũ phải thêm dấu chấm:

.* ./ và .^

Ví dụ: với X ở trên, tính $Z = \sin(X^2)/(1+t^3)$ có cú pháp như sau: Z = sin(X.^2)./(1+t.^3)

3.2 Nghiệm đa thức

Ví dụ giải phương trình $x^3-7x^2+40x - 34 = 0$, cú pháp như sau:

```
a=[1 -7 +40 -34];          % khai báo các hệ số của phương trình
r=roots(a)                 % giải nghiệm của phương trình
ans =
3.000 + 5.000 i
3.000 – 5.000 i
1.000
```

Có 3 nghiệm là 1 và 3 ± 5i. Có thể viết gọn roots ([1 -7 +40 -34]). Ngược lại, để tìm hàm số khi biết nghiệm ta dùng hàm poly(r) với r là nghiệm:

```
>> r = [1,3+5i,3-5i];
>>poly(r)
ans =
1       -7       -40      34
```

3.3 Tóan tử quan hệ quyết định chương trình

Đây là các tóan tử giúp chương trình quyết định dựa vào kết quả tính tóan. Có 6 tóan tử quan hệ giữa các mảng:

Tóan tử	Ý nghĩa
<	nhỏ hơn
<=	nhỏ hơn hoặc bằng
>	lớn hơn
==	bằng
~=	không bằng

Ví dụ: x =[6,3,9] và y =[14,2,9] có các trường hợp sau đây:

```
>> z = (x<y)
z =
1     0     0     (đúng cho 6<14, sai cho 3<2 và 9<9)
>> z = (x~=y)
z =
1     1     0     (đúng vì 6<14, đúng vì 3>2 và sai vì 9=9)
```

3.4 Giải hệ phương trình

$$2x + 4y - 3z = 8$$

$$-2x + 3y + 2z = -6$$

$$-2x + 4y + z = -4$$

Giải : việc giải hệ phương trình trên tương đương với phương trình viết dưới dạng đại số tuyến tính như sau : $Au = b$

Trong đó u là vector nghiệm cần tìm, u = (x,y,z) ; A là ma trận các hệ số bên trái còn b là vector cột các hệ số bên phải :

$$A = \begin{vmatrix} 2 & 4 & -3 \\ -2 & 3 & 2 \\ -2 & 4 & 1 \end{vmatrix}$$

và

$$b = \begin{vmatrix} 8 \\ -6 \\ -4 \end{vmatrix}$$

Vì vậy, nghiệm của hệ phương trình là: u = A\ b

Việc giải trong Matlab rất đơn giản gồm 3 dòng như sau :

```
b=[8;-6;-4]                 % vector cột
A=[2,4,-3;-2,3,2;-2,4,1]    % ma trận 3x3
u =A\b                      % nghiệm của HPT
```

Chú ý:

- dấu \ là dấu chia trái dùng để giải hệ phương trình khi hệ chỉ có một nghiệm. Trong trường hợp hệ vô nghiệm hoặc có vô số nghiệm phải giải bằng cách khác.

3.5 Ma trận và vector

3.5.1 Nhập số liệu

A = [1 2 5; 3 9 0] là ma trận 2 hàng, 3 cột

X = [1 2 5] là vector hàng

Y = [1; 2; 5] là vector cột

Trường hợp quá dài có thể qua hàng dùng '...', ví dụ

A = [1 2 5 3 9 0; 2 3 5 4 6 7; 1 6 7 9 4 5; 2 5 6 7 4 3; 6 7 8 9 1 2; 2 3 5 7 8 9; 2 1 4 5 7... 8; 2 4 6 8 9 1]

3.5.2 Ma trận con và phần tử

Ta có thể lấy một phần tử hoặc ma trận con của một ma trận đã có, ví dụ khi viết A(i,j) là số liệu của phần tử hàng i và cột j. Chẳng hạn với ma trận A ở trên,

```
>> A(2,3)
ans =
     5
```

```
>> B = A(2 :3,1 :3)          % hàng 2 đến 3, cột 1 đến 3
B =
2     3     5
1     6     7
```

3.6 Ma trận

3.6.1 Nhân

Có 2 ma trận A và B, phép nhân 2 ma trận là C = A*B. Ví dụ :

```
>> A = [6, -2 ;10, 3 ;4, 7]
>> B = [9, 8 ;-5, 12]
>> C = A*B
C =
64    24
75    116
1     116
```

Chú ý : muốn nhân 2 ma trận thì số cột của ma trận A phải bằng số hàng của ma trận B

3.6.2 Hóan vị

Ma trận hóan vị của A là A'. Với số thực, B = A' tạo ra ma trận A^T với các phần tử của B là $b_{ij} = a_{ji}$.

3.6.3 Xóa hàng hay cột

Muốn xóa hàng hay cột ta cho cột đó bằng mảng trống [ ]. Ví dụ xóa hàng 2 của ma trận A,

```
>> A(2, :) = [ ]
```

3.6.4 Các ma trận đặc biệt

Zeros(m,n)	là ma trận m x n bằng 0
Ones(m,n)	là ma trận m x n bằng 1
Eye(m,n)	là ma trận m x n với số 1 trên đường chéo chính

Ví dụ :

```
>> eye (3)
ans =
1     0     0
0     1     0
0     0     1
>> B = [ones(3)  zeros(3,2) ; zeros(2,3)  4*eye(2)]
B =
1     1     1     0     0
1     1     1     0     0
1     1     1     0     0
0     0     0     4     0
0     0     0     0     4
```

3.6.5 Các hàm ma trận

expm(A) % tìm hàm mũ của ma trận A, tức là e^A

logm(A) % tìm log (A) để A = $e^{\log(A)}$

sqrtm(A) % tìm căn A

3.6.6 Biến kí tự

Các kí tự dùng trong matlab được để dưới hai dấu ‘

Names = [‘Hoa’; ‘Buoi’; ‘Quyt’]

Chương 4 Lập trình với matlab

4.1 Vòng lặp For

Dùng để lập lại một câu hay một nhóm câu

Ví dụ 4.1:

```
for m = 1: 100
        num = 1/(m+1)        % thực hiện tính tóan giá trị num khi n = 1 đến 100
end
```

Ví dụ 4.2:

```
for n = 100: - 2:0
        k = 1/exp(n)         % thực hiện tính tóan giá trị k khi n = 100 đến 0 bước -2
end
```

Ví dụ 4.3: vòng lặp for có dùng hàm eval, int2str, và lệnh save

```
for k = 1:10
        n = k^2 + 5
        outputfile = ['A', int2str(k)];  % tạo các file Ak có các kết quả x,y, z
        % viết các dòng lệnh tính tóan để thực hiện hàm eval, lệnh save:
        x = 1/exp(n)
        y = k*x^2 + x/2
        z = log(x^2 + 2) + sin(y)
        % Hãy save các biến x, y, z vào Mat-file
        %eval(['save ',outputfile,' x y z'])% các lệnh save, và các biến x, y, z đều ở trong dấu ' để trở thành các chuỗi, còn biến outputfile không cần vì nó đã là dạng chuỗi.
        save (outputfile, 'x', 'y', 'z') % tạo ra các file Ak.mat có 3 biến x, y, z có thể gọi lên.

end
```

(chú ý: sau khi chạy, đã lưu vào các file dạng Ak.mat chứa các giá trị x, y, z. Nếu gọi x sẽ có giá trị ứng với A10.mat là x = 2.5066e-046. Nếu gọi A1.mat bằng lệnh load A1.mat và gọi x sẽ có x = 0.0025 lưu trong A1.mat)

4.2 Vòng lập "while":

Vòng lập while tiến hành 1 câu hay nhóm câu trong một số lần không xác định cho đến lúc điều kiện bởi while không còn thỏa mãn

Ví dụ 4.4: % hãy tìm tất cả các số mũ của 2 dưới 1000

```
v=1; i=1; num = 1;
while num <2000
        num =3^i
        v = [ v; num] % tạo ma trận cột gồm số v và các số num
        i = i+1
end
```

Ví dụ 4.5:

```
x =1;
while x~ = 5
         disp(x)
         x = x+1
end
```

4.3 Câu điều kiện: If –else if – else

Cấu trúc điều kiện giúp phân nhánh lúc tính toán

Ví dụ 4.6:

```
i=1;
while i<15
              if i>5
                 k=i,
                 else if (i>1)
                    k= 5*i ,
                 else
                    k=1,
                 end     % kết thúc else if
              end        % kết thúc if
        i=i+1
end                      % kết thúc while
```

4.4 Vòng lặp xếp lồng

Có thể tạo nhiều vòng lặp xếp lồng vào nhau. Giả sử ta muốn tạo một ma trận có giá trị 1 ở hàng và cột thứ nhất, các số tiếp theo bằng tổng của số hạng ở trên và bên trái nếu tổng bé hơn 20, ngược lại các số bằng số lớn nhất trong hai số hạng đó. Gọi chỉ số hàng và cột tương ứng là r và c, ta viết như sau:

```
%function A = matrandacbiet (n)
n=5;
A = ones(n)
for r = 1:n
        for c = 1:n
                if (r>1) & (c>1)
                        s = A(r-1,c) + A(r,c-1);
                        if s<20
                                A(r,c) = s
                        else
                                A(r,c) = max(A(r-1,c), A(r,c-1))
                        end
                end
        end
end
```

Chạy chương trình bằng cách cho giá trị của n.

4.5 Vòng phân nhánh switch-case-otherwise

Ngòai if else end, đây cũng là một phân nhánh của chương trình.

Ví dụ 4.7:

```
switch color
case ’red’
        c = [1 0 0];
case ’green’
        c = [0 1 0];
case ’blue’
```

```
        c = [0 0 1];
otherwise
        error (’invalid choice of color’) % lệnh error trình bày một dòng thông báo lỗi và chuyển sự điều khiển về keyboard
end
```

Ví dụ 4.8: ví dụ lập trình điểm chuyển động xuất phát từ tọa độ (1,0). Chú ý lệnh **line(t,y)** là dạng bậc thấp của lệnh **plot(t,y)**. Bản chất của đường cong được qui định bởi lệnh **set**(handle, ’PropertyName’, ’PropertyValue’) trong đó handle là một đối tượng.

```
clf
theta = linspace(0,2*pi,100);
x = cos(theta); y = sin(theta);  %vòng tròn bán kính 1
hbead = line(x(1),y(1),’marker’,’o’,’markersize’,8,’erase’,’xor’); % điểm ’o’ tại x(1) = cos(0), y(1) = sin(0) tức tại điểm (1,0). hbead là một handle
htrail = line(x(1),y(1),’marker’,’.’,’color’,’r’,’erase’,’none’); % điểm ’.’ tại cos(0), sin(0) tức tại điểm (1,0)
axis([-1 1 -1 1]);
axis(’square’);
for k=2:length(theta)
        set(hbead,’xdata’,x(k),’ydata’,y(k)); % thiết lập hbead ở tọa độ mới: vẽ ’o’ và xóa
        set(htrail,’xdata’,x(k),’ydata’,y(k)); % thiết lập htrail ở tọa độ mới: vẽ ’.’ và không xóa
        pause(0.1)  % dừng 0.1 giây rồi tiếp tục vẽ
        drawnow
end
```

Chương 5 Tích phân, phương trình vi phân

5.1 Tính tích phân

5.2 Công thức hình thang

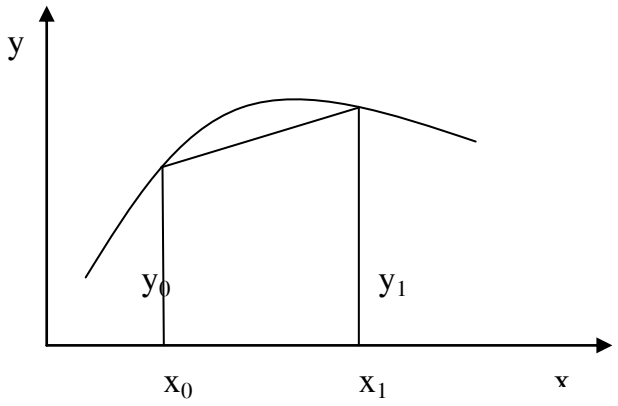
Ví dụ cần tính tích phân trong khoảng $x_0 - x_1$:

$$I = \int_{x_0}^{x_1} y dx$$

$$I \approx \frac{1}{2} h(y_0 + y_1)$$

với khoảng $x_0 \rightarrow x_n$, tương tự ta có

$$I \approx \left[\frac{h}{2}(y_0 + y_1)\right] + \left[\frac{h}{2}(y_1 + y_2)\right] + ... + \left[\frac{h}{2}(y_{n-1} + y_n)\right]$$



Hình 5.1: Tính tích phân theo phương pháp hình thang

hay $$I \approx \frac{1}{2} h(y_0 + 2y_1 + 2y_2 + ... + 2y_{n-1} + y_n) \qquad (5.1)$$

với $n = \frac{x_n - x_0}{h}$ là số các khoảng chia và h là độ dài khoảng chia (hay là bước chia) sai số phép tính là: $E = -\frac{1}{12}(x_n - x_0)h^2 y''(\varepsilon)$ với $x_0 < \varepsilon < x_n$

Cú pháp Matlab: dùng lệnh trapz (x, y)

Ví dụ 5.1: Tính $I = \int_0^{\pi} \sin x dx = -\cos x \Big|_0^{\pi} = 2$

Dùng Matlab:

```
>> x = linspace (0,pi, 10);
>> y = sin(x);
>> trapz (x,y)
```

ans

1.9797

Sai số $E = \frac{100(2-1{,}9797)}{2} = 1\%$

Nếu lấy 100 điểm $x = linspace(0, pi, 100)$ thì kết quả là 1,9998 và sai số $E = \frac{100(2-1.998)}{2} = 0.01\%$

5.3 Quy tắc Simpson:

Tính tích phân từ $x_0 \rightarrow x_2$ dùng công thức Simpson như sau :

$$I = \frac{h}{3}(y_0 + 4y_1 + y_2)$$

Sai số $E = -\frac{h^5}{90} y^{(4)}(\varepsilon)$

Tổng quát tích phân từ $x_1 \rightarrow x_n$:

$$I = \frac{h}{3}(y_0 + 4y_1 + 2y_2 + 4y_3 + ... + 2y_{n-2} + 4y_{n-1} + y_n) \qquad (5\text{-}2)$$

Cú pháp trong Matlab: quad ('function', x_0, x_n, 'tol')

Trong đó hàm 'function' có 2 cận tích phân là x_0 và x_n. Thông số tol là độ chính xác lựa chọn.

Ví dụ 5.2: Tính $I = \int_0^{\pi} \sin x dx$

>> I= quad('sin', 0, pi) %độ chính xác mặc định.

hoặc

>> I = quad ('sin', 0, pi, 'reltol', 1e-6) % dùng độ chính xác tương đối reltol

Ngoài ra, Matlab còn cung cấp lệnh quadl dùng phương pháp Lobatto khi biết y(x). khi cần tăng độ chính xác, dùng quad 8 (dựa vào công thức Newton Cotes)

Ví dụ: >> I = quad 8 ('sin', 0, pi, 'reltol', 1e-6)

5.4 Giải phương trình vi phân bằng phương pháp số

Khi không có thể giải bằng phương pháp phân tích, ta dùng phương pháp số.

5.4.1 Phương pháp Euler

Đây là phương pháp đơn giản nhất để giải phương trình vi phân. Nó có độ chính xác thấp, nhưng giúp hiểu rõ phương pháp tính. Xem phương trình:

$$\frac{dy}{dt} = r(t)y \qquad (5\text{-}3)$$

Với r(t) là một phương trình biết trước. Ta có

$$\frac{dy}{dt} = \lim_{\Delta t \to 0} \frac{y(t+\Delta t) - y(t)}{\Delta t}$$

Khi Δt nhỏ :

$$\frac{dy}{dt} \approx \frac{y(t+\Delta t) - y(t)}{\Delta t} \qquad (5\text{-}4)$$

Hay $$r(t).y(t) = \frac{y(t+\Delta t) - y(t)}{\Delta t}$$

Suy ra $$y(t+\Delta t) = y(t) + r(t).y(t).\Delta t \qquad (5\text{-}5)$$

Hay có thể viết dưới dạng sau gọi là phương pháp Euler :

$$y(t_{k+1}) = y(t_k) + r(t_k).y(t_k).\Delta t \qquad (5\text{-}6)$$

Trong đó Δt là bước, có thể ký hiệu là h

Phương trình vi phân bậc 1 tổng quát : y' = f(t,y)

Có lời giải theo phương pháp Euler là :

$$\begin{cases} t_{k+1} = t_k + \Delta t \\ y(t_{k+1}) = y(t_k) + \Delta t.f[t_k, y(t_k)] \end{cases} \qquad (5\text{-}7)$$

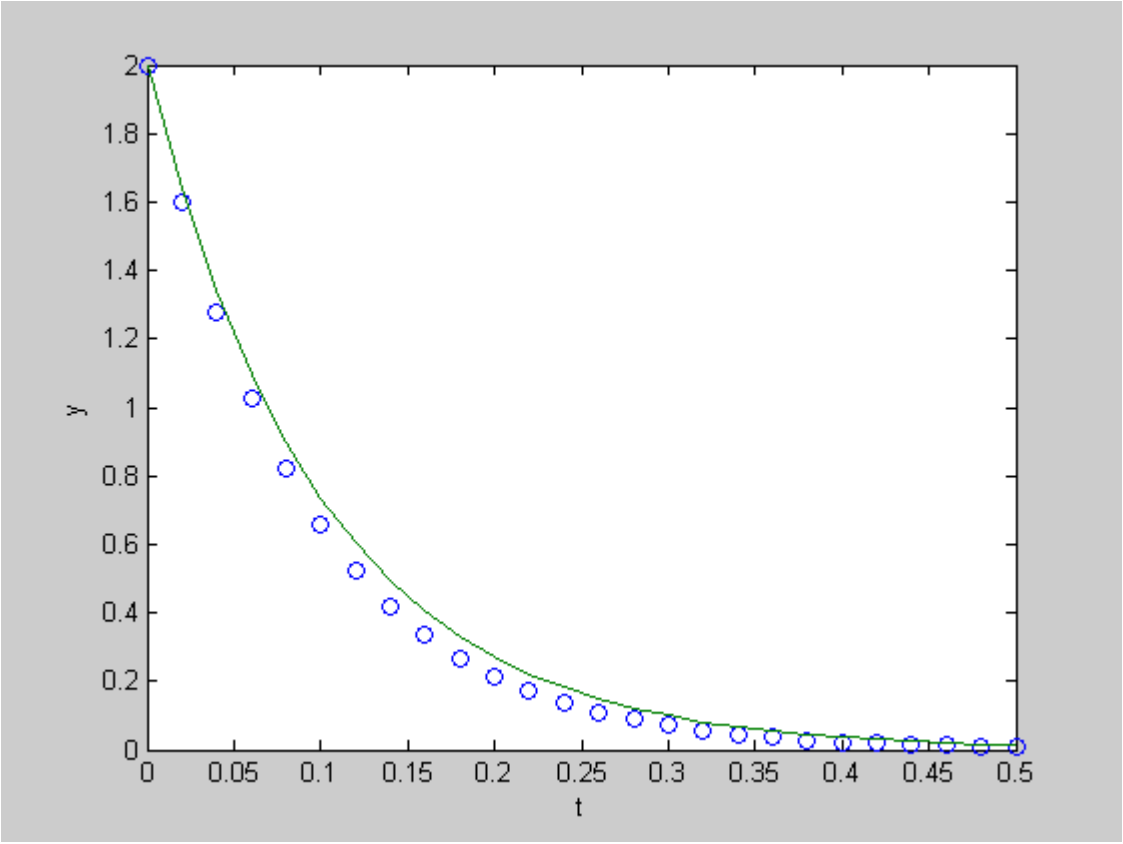
Bằng cách áp dụng phương trình (5-7) trong vòng lặp 'for' trong Matlab ta giải được

<u>Ví dụ 5.3</u> : giải phương trình y'= r.y dùng phương pháp Euler và vẽ đồ thị trong khoảng 0≤ t ≤ 0,5 khi r = -10 với điều kiện ban đầu y_0 = 2.

Lời giải chính xác là $y(t) = 2\, e^{-10\, t}$. Ta sẽ so sánh lời giải chính xác với phương pháp Euller khi Δt = 0,02

```
>> r = -10 ; h = 0.02 ; y(1) = 2;
   k=0 ;
   for time =[h: h: 0.5]
      k = k+1 ;
      y (k+1) =y(k) + r*y(k)*h;
   end
   t = [0 : h : 0.5] ;
   ycx =2* exp(-10*t) ; % là lời giải chính xác.
   plot (t, y, 'o', t, ycx), xlabel ('t'), ylabel ('y')
```

Kết quả cho ra trên hình 5.2, trong đó đường nét liền là kết qủa chính xác còn điểm 'o' là phương pháp Euler. Độ chính xác sẽ tăng lên khi giảm h xuống. Tuy nhiên giảm bước h sẽ tăng thời gian chạy máy và có thể gây nên sai số tích lũy do quá trình làm tròn số.



Hình 5.2: Giải phương trình vi phân y' = -10 y với $y_o = 2$ theo phương pháp Euler

Trong Matlab có sẵn hàm Euler, cú pháp viết như sau :

>> [t,y] = eul ('f', tspan, y_0, h)

Trong đó tspan là khoảng của biến t: tspan = [t_0, t_f] với $t_0 \le t \le t_f$, y_0 là giá trị ban đầu. Còn hàm f là hàm số của phương trình vi phân bậc 1 tổng quát y'= f (t,y) mà ta phải lập file hàm số và lưu vào tên file là f.m, cú pháp như sau :

```
>> function y'= f (t,y)
                y' = r *y
```

Trở lại ví dụ trên, dùng hàm Euler ta viết :

```
>> r = -10 ;
```

>> [t, y] = eul('f', [0 0.5], 2, 0.02] ; ycx = 2*exp (-10*t) ;

>> plot (t, y, 'o', t, ycx)

Nếu ta không khai báo h thì mặc định là $h = \frac{t_f - t_0}{100}$

Bài tập tại lớp : Giải phương trình vi phân: y'= y+1, y_o = 1.

5.4.2 Phương pháp Runge – Kutta:

Đây là phương pháp dựa vào khai triển chuỗi Taylor,

$$y(t+h) = y(t) + h.y'(t) + \frac{1}{2}.h^2.y''(t) + ... \qquad (5\text{-}8)$$

Càng tăng số hạng của chuỗi thì giá trị càng chính xác. Tuy nhiên, trong thực tế việc tính đạo hàm cấp cao rất khó. Vì vậy, phương pháp Runge – Kutta được ứng dụng để tránh sự khó khăn này. Đây là phương pháp dùng nhiều đánh giá hàm f(t, y) để xấp xỉ chuỗi Taylor.

Phương pháp Runge – Kutta bậc 2 như sau:

$y_{k+1} = y_k + \omega_1 g_1 + \omega_2 g_2$ (5-9)

$g_1 = h f(t_k, y_k) = h f_k$ (5-10)

$g_2 = h f(t_k + \alpha h, y_k + \beta h f_k) = h f(t_k + \alpha h, y_k + \beta g_1)$ (5-11)

Và các hệ số α, β, ω_1 , ω_2 phải chọn để thỏa mãn để xấp xỉ chuỗi Taylor:

$$\begin{cases} \omega_1 + \omega_2 = 1 \\ \omega_1 \alpha = \frac{1}{2} \\ \omega_2 \beta = \frac{1}{2} \end{cases}$$

Phương pháp Runge – Kulta bậc 4 như sau:

$$y_{k+1} = y_k + \omega_1 g_1 + \omega_2 g_2 + \omega_3 g_3 + \omega_4 g_4 \qquad (5\text{-}12)$$

$g_1 = h f(t_k, y_k)$

$g_2 = h f(t_k + \alpha_1 h, y_k + \alpha_1 g_1)$

$g_3 = h f[t_k - \alpha_2 h, y_k + \beta_2 g_2 + (\alpha_2 - \beta_2) g_1]$

$g_4 = h f[t_k + \alpha_3 h, y_k + \beta_3 g_3 + \gamma_3 g_3 + (\alpha_3 - \beta_3 - \gamma_3) g_1]$ (5-13)

Có thể chọn nhiều giá trị của các hằng số để xấp xỉ chuỗi Taylor, phương pháp cổ điển dùng qui tắc Simpson cho:

$$\begin{cases} \omega_1 = \omega_4 = 1/6 \\ \omega_2 = \omega_3 = 1/3 \\ \alpha_1 = \alpha_2 = 1/2 \\ \gamma_3 = \alpha_3 = 1 \end{cases}$$

Phương pháp Runge – Kutta cho độ chính xác rất cao so với phương pháp Euller.

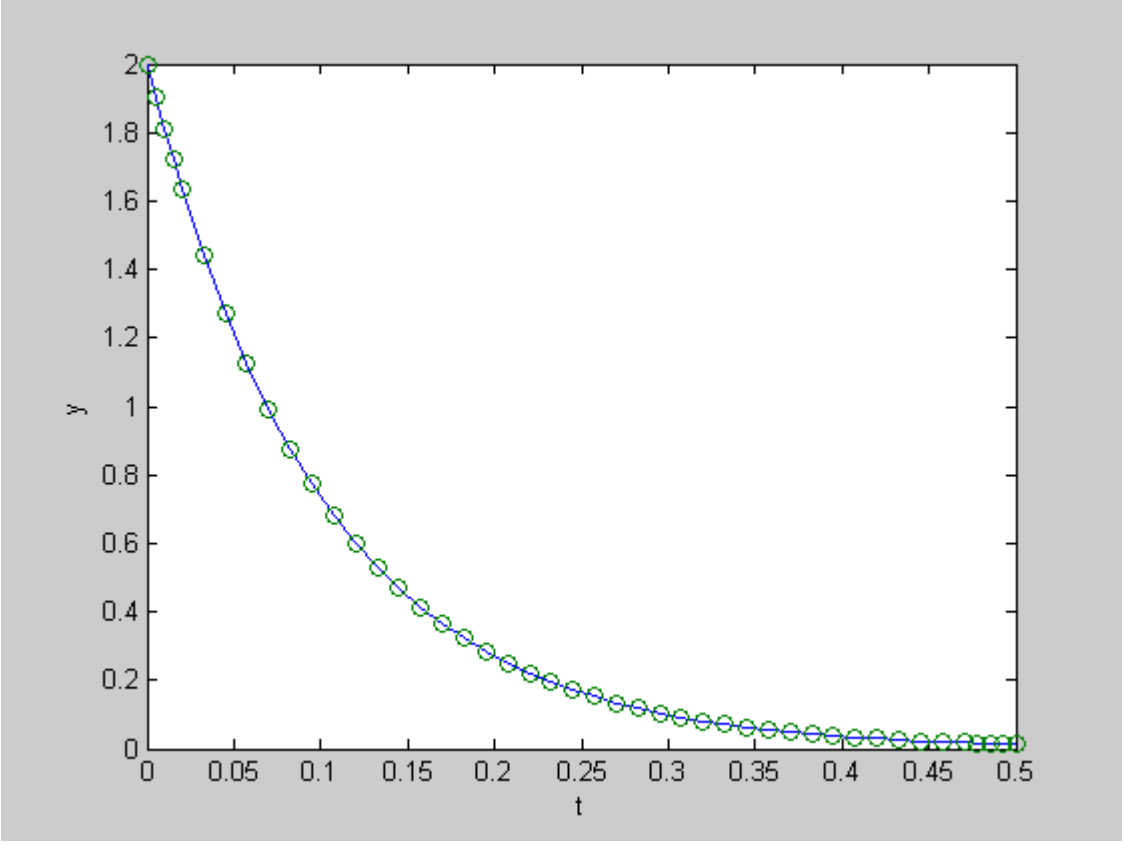
Trong Matlab, phương pháp Runge – Kutta được lập sẵn bằng các hàm ode23, ode45 và ode 113. Hàm ode23 kết hợp Runge – Kutta bậc 2 và 3. Hàm ode45 kết hợp phương pháp Runge – Kulta bậc 4 và 5. Tổng quát, hàm ode45chính xác và nhanh hơn ode23, nhưng nó dùng bước h khá lớn nên hình vẽ không được láng như ode23.

Ví dụ 5.4: Giải bài ví dụ 5.3 dùng phương pháp Runge – Kutta, kết quả cho trên hình 5.3

```
>>      function y'= f54(t,y) ; global r      %khai báo biến toàn cục
        y' = r*y ; % lập file hàm, lưu tên file là f54. m
```

Tiếp tục giải

```
>> global r     r = -10 ; [t, y] = ode45 ('f54', [0 0.5], 2) ; ycx = 2*exp (-10*t) ;
plot (t, y, 'o', t, ycx), xlabel ('t '), ylabel (' y')
```



Hình 5.3: Giải phương trình vi phân y' = -10 y với $y_o = 2$ theo phương pháp Runge-Kutta

Ví dụ 5.5 : so sánh ode45 và ode23 để giải

Phương trình vi phân : y' = sint, y(0) = 0

Lời giải chính xác là y(t) = 1-cost

>> function y'= hamsin (t,y)

y'= sin(t) ; % lập file hàm có tên 'hamsin'.

Giải bằng M-file :

>>tcx = [0: 0.01: 4*pi]; subplot (2,1,1) % *hình phụ 1*

[t, y] = ode45 ('hamsin', [0 4*pi],0) ; ycx = 1- cos (tcx) ;

plot (t, y, 'o', tcx, ycx), xlabel ('t'), ylabel ('y') …

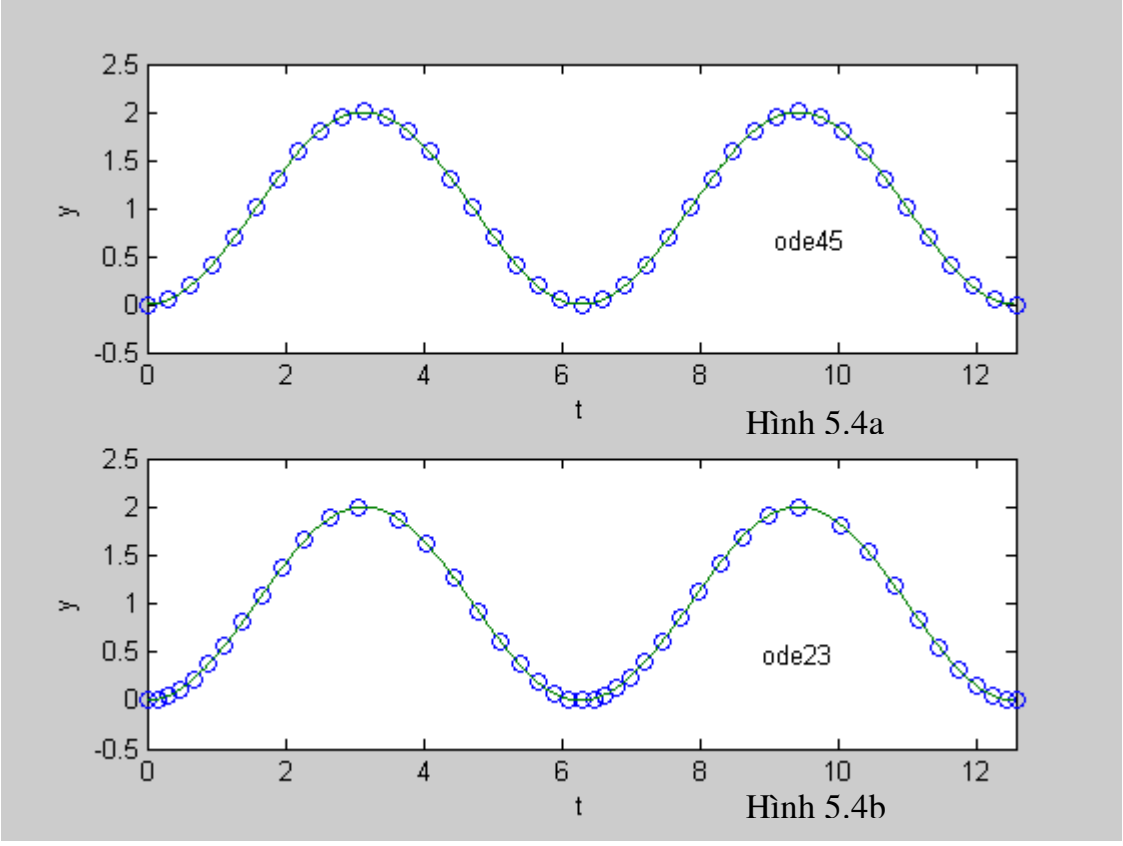
axis ([0 4*pi -0.5 2.5]), gtext ('ode 45') % *chương trình sẽ dừng lại tại lệnh gtext, ta dùng chuột bấm vào vị trí cần ghi sẽ thấy xuất hiện ode 45 trên hình*

subplot (2, 1, 2) % *hình phụ 2*

[t,y] = ode23 ('hamsin', [0 4*pi], 0); plot (t, y, 'o', tcx, ycx), xlabel ('t'), ylabel ('y'), …

axis ([0 4*pi -0.5 2.5]), gtext ('ode23') % *chương trình sẽ dừng lại tại lệnh gtext, ta dùng chuột bấm vào vị trí cần ghi sẽ thấy xuất hiện ode 23 trên hình*

Kết quả cho trên 2 hình 5.4a và 5.4b có sự khác biệt về bước h.



Hình 5.4a

Hình 5.4b

5.5 Hệ phương trình vi phân bậc 1

Xét hệ n phương trình vi phân bậc 1

$$\begin{cases} x_1' = f_1(x_1, x_2, ..., x_n, t) \\ x_2' = f_2(x_1, x_2, ..., x_n, t) \\ ... \\ x_n' = f_n(x_1, x_2, ..., x_n, t) \end{cases} \qquad (5.14)$$

Dưới dạng vector ta viết: x' = F(t, x)

trong đó x là vector cột, $x = [x_1, x_2, ..., x_n]^T$

để giải, ta có 2 bước :

Bước 1 : Viết phương trình để tính $f_1, f_2, \ldots, f_n$ khi nhập x, t

>> function x' = F(t,x) ;

% *Cú pháp gọi* : x' = F(t,x)

% *Nhập liệu* : t = thời gian

x = [$x_1, x_2, \ldots, x_n$]

% *Xuất kết quả* : x' = [$f_1, f_2, \ldots, f_n$]

$x_1' = f_1(x_1, \ldots, x_n, t)$; $x_2' = f_2(x_1, \ldots, x_n, t)$; $x_n' = f_n(x_1, \ldots, x_n, t)$

và save hàm số dưới tên F.m

Bước 2 : Viết script file dùng ode23 hay ode45 để giải

>> tspan = [t_i t_f] ; x_0 = [$x_{10}, x_{20}, \ldots, x_{n0}$] % *giá trị chọn cho tspan và x_0 (ban đầu)*

[t, x] = ode23 ('F', tspan, x_0); % chạy ode23

x_1 = x (:, 1) ; % giá trị của x_1, tức cột thứ nhất của x

x_n = x(:, n) ; % giá trị của x_n, tức cột thứ n của x

% các giá trị $x_1, x_2, \ldots, x_n$ là các giá trị ta cần tìm tại thời điểm t (ví dụ : nhiệt độ = x_1 ; áp suất = x_2 ; ẩm độ = x_3,…)

Ví dụ 5.6:

Giải hệ phương trình vi phân sau:

$x_1' = x_2 - x_1^2$

$x_2' = -x_1 - 2x_1x_2$

trong khoảng t = (0, 10), biết rằng điều kiện ban đầu $x_1(0) = 0$ và $x_2(0) = 1$.

Để giải chúng ta thực hiện 2 bước:

Bước 1: lập file phương trình

```
function  dxdt = F2pt(t,x)
dxdt = zeros(2,1); % tạo vector cột [0;0] cho 2 đạo hàm x1' và x2'
dxdt(1) = x(2) - x(1)^2; % phần tử thứ nhất của vector dxdt
dxdt(2) = -x(1) - 2*x(1)*x(2);% phần tử thứ hai của vector dxdt
```

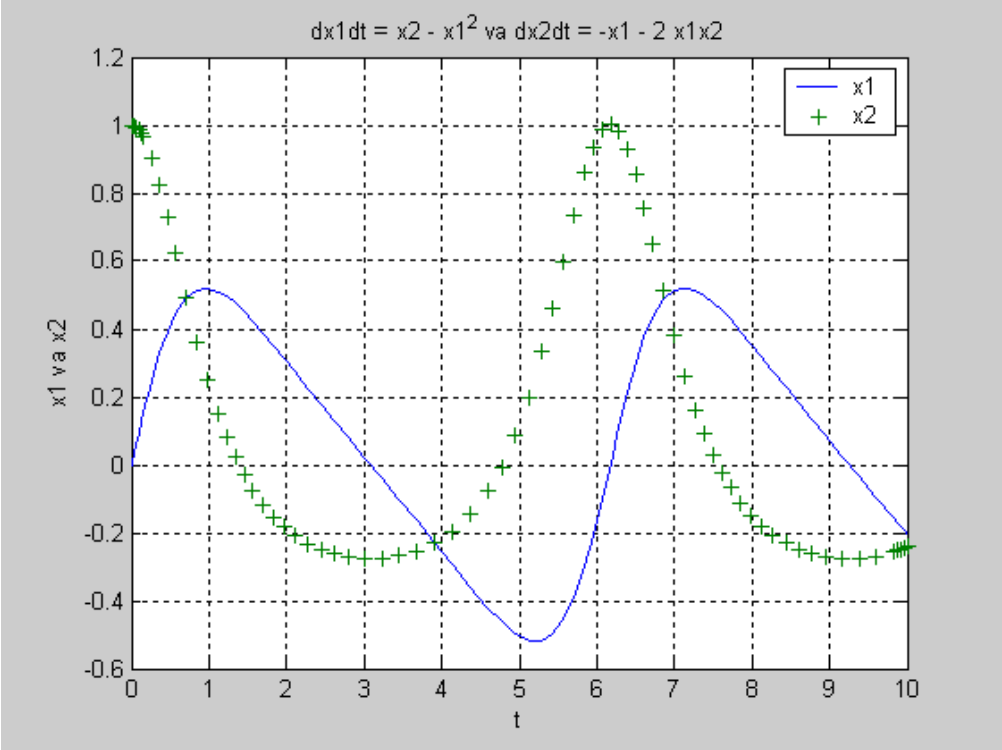
save file này dưới tên **F2pt.m**

Bước 2: lập script file để giải

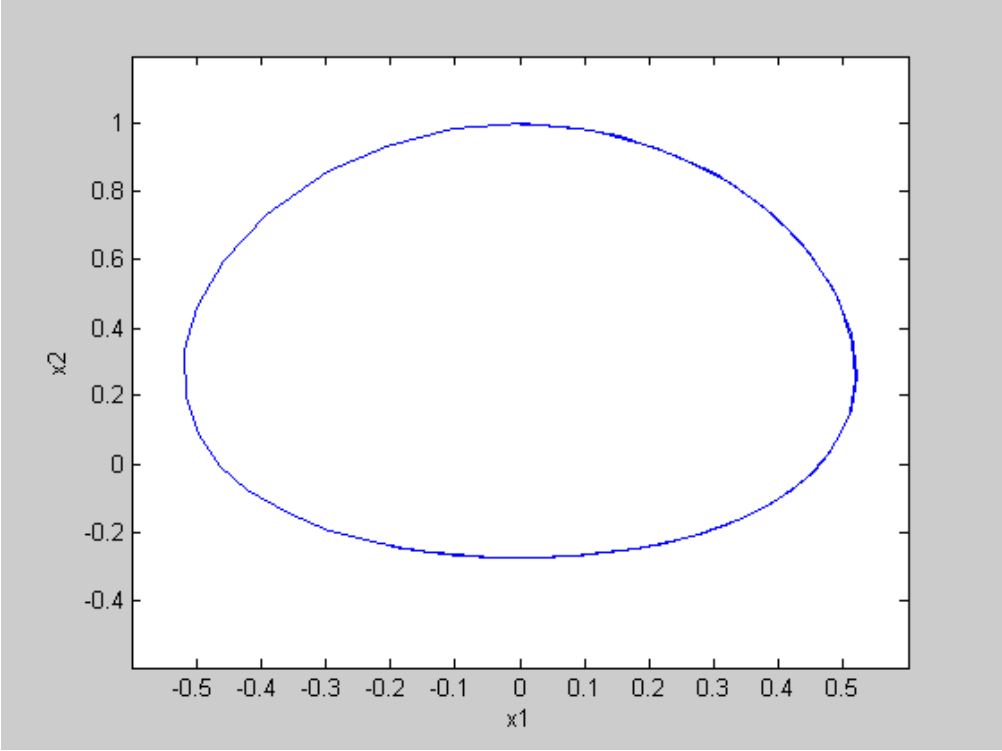
```
tspan = [0 10]; x0 = [0;1]; % khỏang thời gian t và điều kiện ban đầu của x1 và x2
[t,x] = ode45('F2pt', tspan, x0); % giai HPT
x1 = x(:,1); x2 = x(:,2);% cot thu nhat va thu hai cua x
plot (t, x1, t, x2, '+'), xlabel ('t'), ylabel ('x1 va x2')
title('dx1dt = x2 - x1^2 va dx2dt = -x1 - 2 x1x2')
legend('x1','x2')
grid
figure(2) % hinh 2
plot (x1,x2), xlabel ('x1'), ylabel ('x2')
```

Kết qủa cho trên Hình 5.5a (lời giải) x_1 và x_2 và quan hệ giữa x_1 và x_2 (Hình 5.5b). Hai hình 5.5a và 5.5b đại diện cho quan hệ giữa x_1, x_2 và t. Có thể vẽ đường cong quan hệ giữa x1, x2 và t trong không gian 3 chiều dùng lệnh plot3(t, x1, x2):

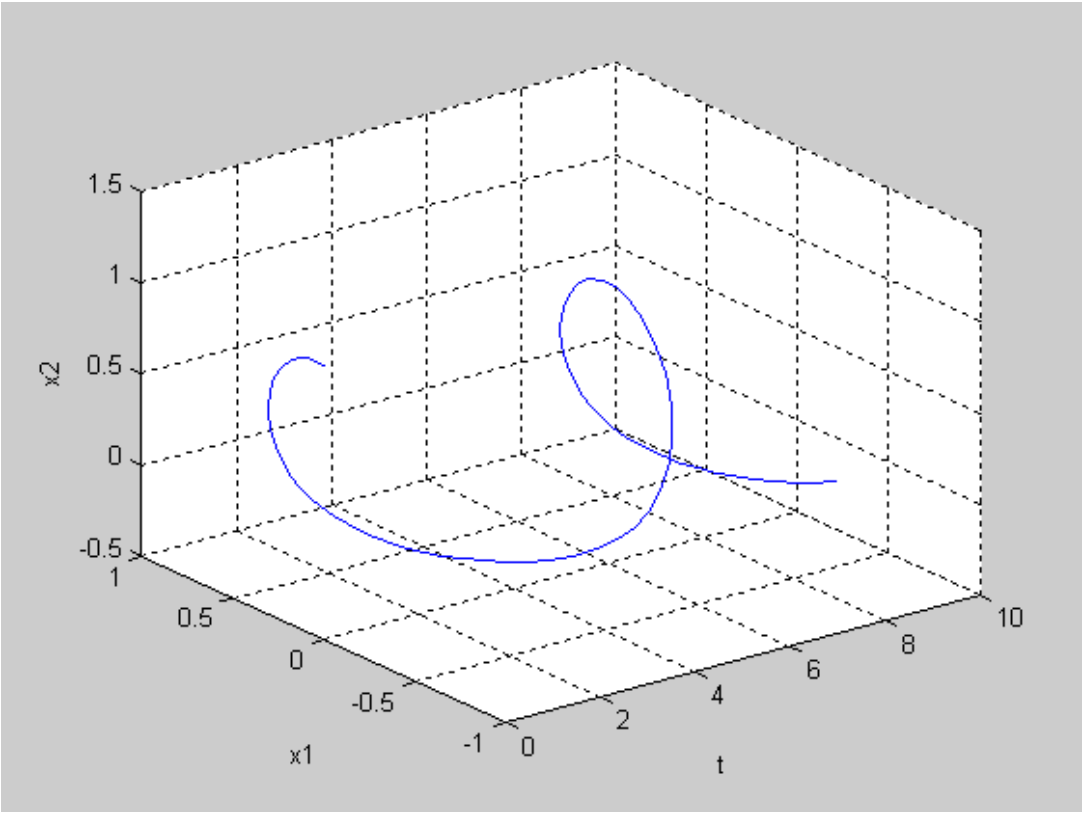
```
plot3 (t,x1,x2), xlabel ('t'), ylabel ('x1'), zlabel ('x2')
```



Hình 5.5 a: Lời giải hệ phương trình vi phân ví dụ 5.6 dùng ode23



Hình 5.5 b: Quan hệ giữa x1 và x2, còn gọi là hình vẽ mặt phẳng pha.



Hình 5.5c: Vẽ đường cong trong không gian 3 chiều quan hệ giữa x_1, x_2 và t bằng lệnh plot3(t, x1, x2).

5.6 Phương trình vi phân bậc 2

Để giải một phương trình vi phân bậc 2, cần thay thế nó bằng hệ phương trình vi phân bậc 1 tương ứng. Xem phương trình:

$$y'' = f(t, y, y') \qquad (5.15)$$

ta đặt x1 = y và x2 = y'. Như vậy x = (x1, x2) là nghiệm của hệ phương trình vi phân bậc 1 sau đây:

x1' = x2

x2' = f(t, x1, x2)

Ví dụ 5.7: Giải phương trình vi phân bậc 2 sau trong khoảng [0, 10]:

$$y'' + yy' + y = 0$$

với y(0) = 0 và y'(0) = 1.

Bằng cách đặt x1 = y và x2 = y' ta có hệ:

x1' = y' = x2

x2’ = y’’ = - yy’ – y = -x1x2 – x1

Làm tương tự như trên với hệ phương trình bậc 1:

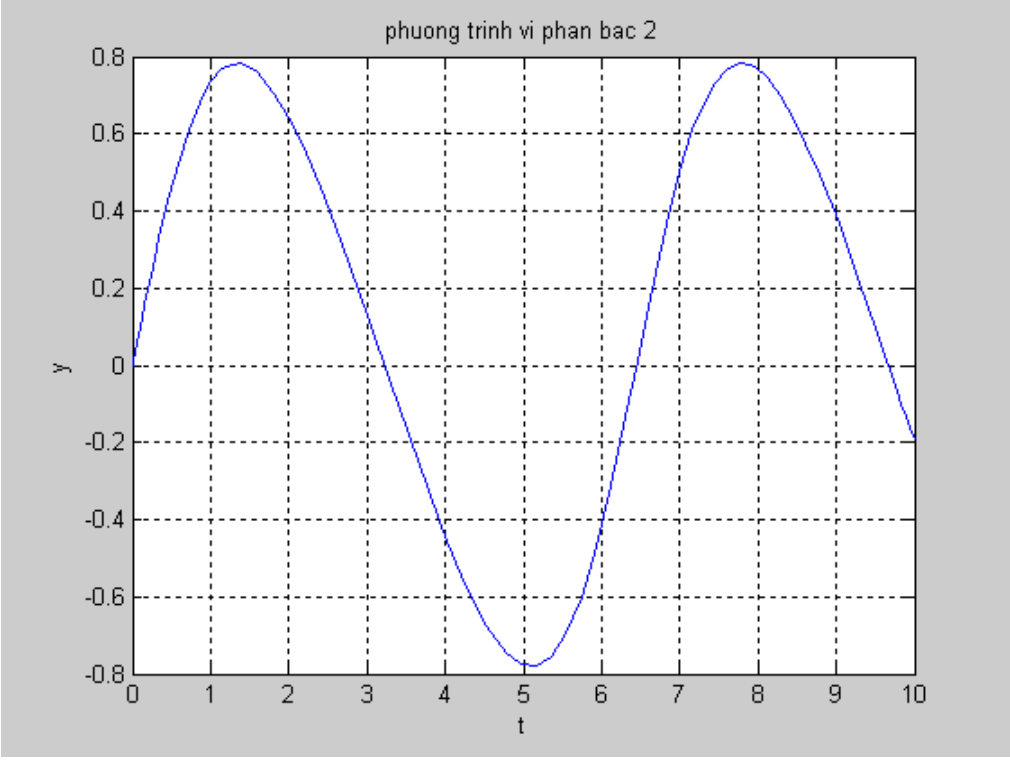
Bước 1: lập file hàm

```
function  dxdt= Fptb2(t,x)
dxdt=zeros(2,1); %tao vector cot [0;0] cho 2 dao ham x1' va x2'
dxdt(1) = x(2) ; % phan tu thu nhat cua vector dxdt
dxdt(2) = -x(1)*x(2)-x(1);% phan tu thu hai cua vector dxdt
```

save file ten **Fptb2.m**

Bước 2: lập script file

```
tspan=[0 10]; x0=[0;1]; %range of time and initial x
[t,x]=ode45('Fptb2',tspan,x0);giai HPT
x1=x(:,1);%cot thu nhat cua x
plot (t,x1), xlabel ('t'), ylabel ('y')
title('phuong trinh vi phan bac 2')
grid
```



Hình 5.6: Nghiệm phương trình vi phân bậc 2.

PHẦN B: EXCEL

Chương 6 GIỚI THIỆU Excel

6.1 Dùng Excel giải hệ phương trình

Xem bài thực hành

6.2 Dùng Excel tìm cực trị hàm có điều kiện ràng buộc

Xem bài thực hành sử dụng Solver, pp đơn hình (linear programing)

Tài liệu tham khảo

Matlab

Excel