# MỤC LỤC

# DANH MỤC CÁC BẢNG

***Trang***

# DANH MỤC CÁC HÌNH

*Trang*

# MỞ ĐẦU

Giải tích số hiện đang phát triển mạnh và càng mạnh hơn với sự phát triển cực kỳ nhanh của tin học. Công nghệ tin học và giải tích số có vai trò rất quan trọng trong các lĩnh vực, ngay cả trong thị trường chứng khoán đang mới nổi ở Việt Nam và diễn biến rất gay gắt hiện nay trong nền kinh tế thị trường, cũng phải nghiên cứu nhiều bài toán phương trình vi tích phân ngẫu nhiên mà không thể không sử dụng kiến thức về giải tích số và tin học ứng dụng. Ở đây chúng tôi không đi vào nghiên cứu bài toán về thị trường chứng khoán mà chỉ muốn nói rõ vai trò của giải tích số và tin học.

Thực tế hiện nay: Giải tích số, các phương pháp tính trình bày đơn thuần bằng kiến thức toán thì tài liệu và sách hiện nay có rất nhiều và phổ biến. Tài liệu và sách về giải số các bài toán bằng các chương trình máy tính hiện cũng đã có nhưng bằng MATLAB thì không nhiều và chương trình MATLAB được đưa vào còn rời rạc và chưa hướng dẫn cho người đọc cách thực hiện một chương trình cụ thể như thế nào.

Trong khi đó, MATLAB – là phần mềm nổi tiếng của công ty MathWorks, là một ngôn ngữ hiệu năng cao cho tính toán kỹ thuật. Nó tích hợp tính toán, hiện thị và lập trình trong một môi trường dễ sử dụng. Các ứng dụng tiêu biểu của MATLAB bao gồm: hỗ trợ toán học và tính toán; phát triển thuật toán; mô hình, mô phỏng; phân tích, khảo sát và hiển thị số liệu; đồ họa khoa học và kỹ thuật; phát triển ứng dụng với giao diện đồ họa.

Ngoài MATLAB cơ bản với các khả năng rất phong phú, phần mềm MATLAB còn được trang bị thêm các ToolBox – các gói chương trình (thư viện) cho các lĩnh vực ứng dụng rất đa dạng như xử lý tín hiệu, nhận dạng hệ thống, xử lý ảnh, mạng nơ ron, logic mờ, tài chính, tối ưu hóa, phương trình đạo hàm riêng,

sinh tin học,... Đây là các tập hợp mã nguồn viết bằng chính MATLAB dựa theo các thuật toán mới, hữu hiệu mà người dùng có thể chỉnh sửa hoặc bổ sung thêm các hàm mới. MATLAB được thiết kế để giải các bài toán bằng số chứ không nhằm mục đích chính là tính toán ký hiệu như MATHEMATICA và MAPLE. Tuy nhiên, trong MATLAB cũng có thể tính toán ký hiệu được nhờ các hàm trong Symbolic Math ToolBox.

Trên thế giới, MATLAB được cộng đồng hàn lâm trên thế giới chấp nhận rộng rãi như một công cụ phục vụ cho giảng dạy, nghiên cứu toán học và phát triển các ứng dụng kỹ thuật. Hơn 3500 trường đại học nhất là các trường đại học kỹ thuật đã đưa MATLAB vào giảng dạy và nghiên cứu. Hiện nay đã có trên 700 đầu sách về MATLAB dành cho giáo viên, sinh viên và các nhà chuyên môn. Ở Việt nam, theo chúng tôi được biết, MATLAB đã được đưa vào giảng dạy cho sinh viên, học viên cao học hoặc giới thiệu tại một số khoa, trường đại học và cũng đã xuất bản một số đầu sách về MATLAB dành cho sinh viên các khối khoa học và kỹ thuật. Tuy nhiên mức độ phổ biến của MATLAB chưa phải là cao. Nhất là ở đồng bằng sông Cửu Long.

Với ưu thế về tính toán số trị MATLAB rất thích hợp cho việc giảng dạy môn học giải tích số, các phương pháp số - môn học không thể thiếu được đối với sinh viên toán, lý, công nghệ thông tin và các ngành kỹ thuật. Việc sử dụng MATLAB để lập trình các thuật toán của môn học này có cái lợi là đơn giản, dễ dàng vẽ các đồ thị để hiện thị kết quả và kiểm tra kết quả các chương trình tự viết so với kết quả của các hàm đã cài đặt sẵn vì MATLAB cơ bản chứa đựng rất nhiều các hàm tính toán toán học. Từ đó, góp phần nâng cao năng lực dạy và học toán trong các trường Đại học, đặc biệt là môn học giải tích số và phương pháp tính.

Chính vì thế, PGS.TS ĐẶNG ĐỨC TRỌNG đã khuyên tôi nên chọn đề tài

"GIẢI SỐ BẰNG MATLAB".

Trong luận văn này, chúng tôi sẽ trình bày sơ lược (không đi sâu) về lý thuyết giải số mà chú trọng vào các chương trình MATLAB để giải quyết bài toán số.

Để thực hiện luận văn này, chúng tôi đã tham khảo tài liệu có liên quan trong và ngoài nước, phân tích, chọn lọc và hệ thống hoá thành cơ sở lý luận phục vụ cho đề tài. Ngoài ra, chúng tôi còn sử dụng công nghệ thông tin, internet.

Khi nhận được lời khuyên từ PGS.TS ĐẶNG ĐỨC TRỌNG về đề tài này, tôi bắt đầu tìm hiểu sơ lược về đề tài, chọn đề tài. Sau đó, nghiên cứu những tài liệu có liên quan rồi phân tích, tổng hợp, chọn lọc những nội dung cơ bản phục vụ cho đề tài. Lập đề cương dự kiến, tìm hiểu và chạy thử một số chương trình MATLAB. Tiến hành sắp xếp lại các vấn đề, điều chỉnh, bổ sung các đoạn chương trình và viết hoàn chỉnh thành nội dung của luận văn.

Nội dung của luận văn được chia thành 8 chương. Chương 1, giới thiệu sơ lược về MATLAB, một số đặc tính và một số lệnh được sử dụng trong các chương trình MATLAB trong luận văn này. Chương 2, trình bày đa thức Taylor, cách thiết lập, cách xác định giá trị và sai số của đa thức Taylor, và cách ước lượng đa thức. Chương 3, trình bày một số phương pháp tìm nghiệm thực của phương trình phi tuyến, phương pháp chia đôi, phương pháp Newton và phương pháp cát tuyến. Chương 4, phép nội suy và phép tính xấp xỉ, trong phần này chúng tôi trình bày phép nội suy đa thức, đa thức Chebyshev, phép nội suy dùng hàm ghép trơn và bài toán xấp xỉ hàm thực nghiệm. Chương 5, tích phân số và vi phân, chúng tôi trình bày một số công thức tính tích phân, công thức hình thang, công thức Simpson và công thức tích phân Gauss, và sơ lược về vi phân số. Chương 6, một số phương pháp giải hệ phương trình tuyến tính như là phương pháp Gauss, phương pháp

nhân tử LU và phương pháp lặp. Chương 7, một số phương pháp giải số phương trình vi phân thường như là phương pháp Euler, phương pháp Runge – Kutta và phương pháp đa bước đối với phương trình vi phân tuyến tính cấp một, và giải bài toán biên tuyến tính cấp hai. Chương 8, giải số phương trình đạo hàm riêng dựa trên cơ sở phân rã các bài toán, bài toán Laplace và bài toán Parabolic một chiều và hai chiều cùng với việc đánh giá bậc hội tụ và điều kiện Neumann của nó.

Nhìn chung nội dung luận văn đã trình bày tương đối đầy đủ các nội dung chương trình của môn học giải tích số cũng như môn học phương pháp tính của sinh viên chuyên ngành toán, sinh viên bách khoa và sinh viên ngành kỹ thuật. Cùng với các chương trình MATLAB ở từng mục sẽ tạo nên một phong cách tươi mới hơn, tích cực hơn, hiệu quả và hiện đại hơn trong việc dạy và học môn giải tích số cũng như môn phương pháp tính. Góp phần nâng cao năng lực dạy và học toán trong các trường đại học. Đây cũng là mục tiêu hướng tới của chúng tôi khi thực hiện luận văn này.

# NỘI DUNG

# CHƯƠNG 1
# GIỚI THIỆU SƠ LƯỢC VỀ MATLAB

## 1.1. TỔNG QUAN

Như đã biết, MATLAB được thiết kế để giải các bài toán bằng số chứ không nhằm mục đích chính là tính toán ký hiệu như MATHEMATICA và MAPLE. Tuy nhiên, trong MATLAB cũng có thể tính toán ký hiệu được nhờ các hàm trong Symbolic Math ToolBox (chúng tôi không trình bày cụ thể lý thuyết phần này mà chỉ chú trọng phần lập trình).

Tên của phần mềm MATLAB là viết tắt của thuật ngữ "MATrix LABoratory", được Cleve Moler phát minh vào cuối thập niên 1970, và sau đó là chủ nhiệm khoa máy tính tại Đại học New Mexico. Đầu tiên nó được viết bằng FORTRAN để cung cấp truy nhập dễ dàng tới phần mềm ma trận được phát triển bởi các dự án LINPACK và EISPACK. Sau đó nó được viết bằng ngôn ngữ C trên cơ sở các thư viện và phát triển thêm nhiều lĩnh vực của tính toán khoa học và các ứng dụng kỹ thuật.

Steve Bangert là người đã viết trình thông dịch cho MATLAB. Công việc này kéo dài gần 1½ năm. Sau này, Jack Little kết hợp với Moler và Steve Bangert quyết định đưa MATLAB thành dự án thương mại - công ty The MathWorks ra đời thời gian này – năm 1984.

Năm 2004 MATLAB 7 phát hành, có khả năng chính xác đơn và kiểu nguyên, hỗ trợ hàm lồng nhau, công cụ vẽ điểm, và có môi trường phân tích số liệu tương tác. Đến tháng 12 năm 2008, phiên bản 7.7 được phát hành với SP3 cải thiện Simulink cùng với hơn 75 sản phẩm khác.

**Một số đặc trưng chính của MATLAB:**

– MATLAB là ngôn ngữ thông dịch. Vì thế nó có thể làm việc ở hai chế độ: tương tác và lập trình. Trong chế độ tương tác MATLAB thực hiện từng lệnh được gõ trong cửa sổ lệnh sau dấu nhắc lệnh và kết quả tính toán được hiện ngay trong cửa sổ này, còn đồ thị được hiện trong một cửa sổ khác. Lệnh tương tác có thể là đơn giản, thí dụ tính sin(1.5) hoặc vẽ fplot('sin(1 ./ x)', [0.01 0.1]), có thể là cấu trúc điều kiện, thí dụ if x<=0; y=0; else; y=1; end hoặc các cấu trúc lặp xác định và không xác định. Trong chế độ lập trình một tập lệnh được soạn thảo và ghi thành một tệp đuôi .m (m-file). Các hàm cũng được tổ chức thành các m-file. Một chương trình có thể gồm nhiều m-file. Để chạy chương trình chỉ cần gõ tên m-file chính trong cửa sổ lệnh rồi Enter.

– Các hàm trong MATLAB cơ bản (không kể các thư viện chuyên dụng được gọi là các ToolBox) được chia làm 2 loại: hàm trong và hàm ngoài. Các hàm trong là các hàm được cài đặt sẵn (built-ins) tức là tồn tại dưới dạng mã nhị phân nên ta không thể xem được mã nguồn của chúng, thí dụ các hàm sin, sqrt, log, clear, clc,.... Đây là các hàm hay được sử dụng hoặc các hàm đòi hỏi nhiều thời gian xử lý. Các hàm ngoài là các hàm tồn tại dưới dạng mã nguồn mà người dùng có thể tham khảo hoặc chỉnh sửa, bổ sung khi cần thiết, thí dụ log10, ode23, fzero,...

– Phần tử dữ liệu chính của MATLAB là các ma trận (mảng) mà kích thước của chúng không cần khai báo trước như trong các ngôn ngữ lập trình khác. Tuy nhiên, để tăng tốc độ xử lý cần báo trước cho MATLAB biết kích thước tối đa của mảng để phân bổ bộ nhớ bằng một lệnh gán, chẳng hạn A(20,30)=0.

**Các khả năng chính của MATLAB cơ bản:**

– Thực hiện các tính toán toán học bao gồm: ma trận và đại số tuyến tính, đa thức và nội suy, phân tích số liệu và thống kê, tìm cực trị của hàm một biến hoặc nhiều biến, tìm nghiệm của phương trình, tính gần đúng tích phân, giải phương trình vi phân…

– Đồ họa 2 chiều và 3 chiều: MATLAB cung cấp rất nhiều các hàm đồ họa, nhờ đó ta có thể nhanh chóng vẽ được đồ thị của hàm bất kỳ 1 biến hoặc 2 biến, vẽ được các kiểu mặt, các contour, trường vận tốc,...Ngoài ra MATLAB còn vẽ rất tốt các đối tượng 3 chiều phức tạp như hình trụ, hình cầu, hình xuyến,..và cung cấp khả năng xử lý ảnh và hoạt hình.

– Xây dựng giao diện người dùng: với MATLAB 7 người dùng có thể dễ dàng xây dựng giao diện gồm các thực đơn, nút lệnh, hộp thoại, hộp chọn,...mà không cần phải viết mã như các phiên bản trước đây.

### 1.1.1 Chương trình

- Một chương trình MATLAB thường được soạn trong các **M-file** (các file có đuôi **.m**).
- Để chạy các dòng lệnh trong file **xyz.m** nào đó, ta vào cửa sổ làm việc và gõ **xyz** rồi Enter. *Lưu ý:* lúc này đường dẫn tới thư mục chứa file **xyz.m** (và các file liên quan) phải được khai báo trong **Current Directory** của MATLAB. Khi mới khởi động, thư mục này mặc định là Work trong chỗ cài đặt Matlab (thường là **C:\MATLAB7\work**).

### 1.1.2 Dòng lệnh

- Các dòng lệnh trong MATLAB được thực hiện tiếp nối nhau. Mỗi dòng lệnh thông thường có thể có dấu " **;** " ở cuối hoặc không. Nếu dòng lệnh không có dấu " **;** " ở cuối thì kết quả sẽ được xuất ra. Trong trường hợp không muốn nhìn các kết quả trung gian mà chỉ muốn xem kết quả cuối cùng, ta sử dụng dấu " **;** " cho các dòng lệnh mà ta không muốn xem kết quả.

Ví dụ: dòng lệnh **x=1+2** sẽ xuất ra **x=3.** Tuy nhiên dòng lệnh **x=1+2;** sẽ không xuất ra gì hết (mặc dù giá trị của biến **x** bây giờ là 3).

- Nếu muốn loại bỏ một dòng lệnh khi chạy chương trình, ta có thể để dấu **%** ở đầu dòng lệnh. Thông thường dấu **%** được sử dụng để ghi các chú thích (chỉ dùng cho người đọc, máy không thực thi).

### 1.1.3 Hàm số

- Ta có quyền viết **sin(2)** vì hàm **sin** là một hàm đã có sẵn trong thư viện MATLAB. Ngoài ra ta có quyền định nghĩa thêm các hàm mới, và sau khi định nghĩa thì ta có quyền sử dụng các hàm mới này y hệt như các hàm cơ bản như hàm **sin.**
- Hàm số **xyz** được viết trên file **xyz.m,** có cú pháp kiểu như:

**function a=xyz(b,c)** % day la ham xyz

Trong đó **b, c** là các dữ liệu nhập vào, **a** là giá trị trả về (trong chương trình sẽ có ít nhất một lệnh gán, chẳng hạn **a = b + c;**), dòng chữ **day la ham xyz** là chú thích về công dụng của hàm **xyz.**

Ví dụ: ta thành lập một hàm đổi độ sang radian

**function rad=change(do)** % day la ham doi do sang radian.
**rad=do*pi/180;** % doi do sang radian.

Đặt tên file là change.m. Nếu ta đổi 45 độ sang radian, chỉ cần gõ:

**>> rad=change(45)**

Rồi nhấn Enter sẽ cho ta kết quả

**rad =**
**0.7854**

Bây giờ ta thử tạo hàm giải phương trình bậc hai $ax^2+bx+c=0$, với tên file là bachai.m. Trong đó a, b, c được đưa vào khi gọi chương trình.

```
function [x1,x2]=bachai(a,b,c)
delta=b^2-4*a*c;
x1=(-b-sqrt(delta))/(2*a); x2=(-b+sqrt(delta))/(2*a);
```

Khi chạy chương trình từ Command Windo, ta gõ

**>>[x1,x2]=bachai(4,6,-7)**

MATLAB sẽ chạy chương trình giải phương trình bậc hai với a=4, b=6, c=-7, và cho kết quả là

**x1=**
**-2.2707**
**x2=**
**0.7707**

Ta cũng có thể lập hàm để giải phương trình bậc hai với các hệ số thay đổi được nhập từ bàn phím theo yêu cầu của từng phương trình.

- Để xem công dụng của một hàm số **xyz** (là hàm có sẵn trong thư viện hoặc do ta tự định nghĩa), ta vào cửa sổ làm việc và gõ **help xyz** rồi Enter. Ta sẽ được xem các chú thích trong file **xyz.m**.

### 1.1.4 Biến số

- Các biến được ký hiệu bằng một ký tự hoặc 1 chuỗi ký tự. MATLAB phân biệt chữ thường và chữ hoa.
- Các biến thông thường được định nghĩa trong 1 file được gọi là biến địa phương (**local variable**). MATLAB cũng cho phép sử dụng một số biến toàn cục (**global variable**). Biến toàn cục **xx** phải được khai báo là **global xx**, trong tất cả các file mà **xx** xuất hiện, có một file định nghĩa **xx,** chẳng hạn gán **xx=3**.
- Biến **i** và **j** được mặc định là số ảo đơn vị (**i^2=-1**). Tuy nhiên, nếu ta dùng lệnh gán **i=3** thì biến **i** sẽ mang giá trị 3.

## 1.2. MỘT SỐ LỆNH CƠ BẢN

Thư viện có sẵn của MATLAB rất phong phú và có thể tự học các lệnh trong MATLAB một cách dễ dàng bằng cách vào phần Help (hoặc bấm F1) của

chương trình. Trong các chương trình ở phần sau ta chỉ cần một số rất ít các lệnh dưới đây.

### 1.2.1 Lệnh gán

Có dạng **x=y.** Chú ý rằng để thực hiện lệnh gán thì **x** không cần phải khai báo trước, mà cũng không cần có cùng kiểu dữ liệu với **y**. Lệnh này đơn giản làm cho **x** trở thành một copy của **y**.

### 1.2.2 Các lệnh trên ma trận và vector

- Lệnh **A(m,n)** trả về phần tử dòng **m**, cột **n** của matrận **A.**
- Lệnh **A'** sẽ chuyển **A** thành $\mathbf{A}^t$.
- Lệnh **A=zeros(m,n)** sẽ cho **A** là ma trận **0** có **m** dòng và **n** cột. Tương tự cho lệnh **ones(m,n).**
- Lệnh **size(A)** sẽ cho số dòng và số cột của **A**.
- Lệnh **B=f(A)** sẽ cho ma trận **B** cùng cỡ với ma trận **A** và **B(m,n)=f(A(m,n))**.
- Lệnh **A+B**, **A*B** cho phép cộng và nhân ma trận.
- Lệnh **A^(-1)** cho nghịch đảo ma trận **A.**

Vì vector cũng là ma trận nên ta có thể dùng lại các lệnh của ma trận, và

- Lệnh **X=[a1,a2,a3]** sẽ cho vector **X** có **X(n)=an**, chỉ số **n** đánh từ 1.
- Lệnh **X=[a:b]** sẽ cho vector X=[a,a+1,…,b]; **a, b** là các số nguyên.
- Lệnh **A\b** tương đương với **A^(-1)*b**.
- Lệnh **norm(X,p)** cho chuẩn của **X** theo $\mathbf{l}^p$; **norm(X)** tương đương **norm(X,2)** là chuẩn Euclide, và **norm(X,inf)** là chuẩn **sup**.

### 1.2.3 Các lệnh cấu trúc

- Lệnh **if** có dạng

```
if (biểu thức logic)
        (Các dòng lệnh)
else
        (Các dòng lệnh)
end
```

Lưu ý là **else** có thể bỏ đi để được dạng thu gọn **if … end**. Trong các biểu thức logic, ta có thể dùng các toán tử so sánh như == (equal), ~= (not equal), >=, <=, >,<, và các toán tử logic như **&** (and), | (or)…

- Lệnh **for** có dạng

```
for i=a:b
        (Các dòng lệnh)
end
```

Ở đây ban đầu **i=a**, sau mỗi bước lặp **i** sẽ được tăng lên **1,** và **i=b** tại vòng lặp cuối cùng.

- Lệnh **while** có dạng

```
while (biểu thức logic)
        (Các dòng lệnh)
end
```

Vòng lặp sẽ dừng khi biểu thức logic trả về giá trị 0 (sai).

- Các lệnh **break**, **return** và **error:**

Lệnh **break**: kết thúc sự thực thi vòng lặp for hoặc while.

Lệnh **return**: thường được sử dụng trong các hàm của MATLAB, nó cho phép quay trở lại thực thi những lệnh nằm trong tác dụng của lệnh này.

Lệnh **error('dòng nhắn')**: kết thúc việc thực thi lệnh và hiển thị dòng nhắn trên màn hình.

Xét ví dụ, chọn một số nguyên dương bất kỳ, nếu số chẵn thì chia cho 2, số lẻ thì nhân nó với 3 rồi cộng 1. Lặp lại quá trình này cho đến khi được kết quả là 1. Chương trình có trong file dk.m. Khi chạy chương trình sẽ thấy tác dụng của lệnh break (dừng chương trình khi nhập số âm hoặc số 0).

**Chương trình:**

```
while 1
   n=input('Nhap vao mot so:');
   if n<=0
      break
   end
   while n>1
      if rem(n,2)==0  % phan du cua n chia 2.
         n=n/2
      else
         n=n*3+1
      end
   end
end
```

### 1.2.4 Vẽ hình

MATLAB cung cấp rất nhiều công cụ vẽ hình, tuy nhiên ta có thể dùng một số lệnh đơn giản cho phép vẽ hình từ dữ liệu rời rạc. Trước tiên ta có một số lệnh vẽ hình trong 2 chiều:

- Lệnh **plot(X,Y)** trong đó X,Y là hai vector có cùng cỡ, sẽ vẽ bằng cách nối các điểm có tọa độ (X(n),Y(n)).
- Một số option (có hoặc không, sau lệnh **plot**): Ta có thể dùng **xlabel('x'), ylabel('f(x)')** để ghi chú cho trục hoành và trục tung. Ta có thể dùng **title('Tua de cua hinh ve')** để thêm tựa đề cho một hình**.** Ta cũng có thể định trước miền hiển thị, chẳng hạn lệnh **axis([0 1 2 3])** sẽ chỉ hiển thị phần hình vẽ trên hình vuông **[0,1]x[2,3]**. Nếu muốn đồ thị có màu đỏ chẳng hạn, ta dùng **plot(X,Y, 'r').**
- Trong trường hợp muốn vẽ nhiều hình, ta phải gọi các lệnh

```
figure(1)
plot(X1,Y1,'r')
figure(2)
plot(X2,Y2,'b')
```

Khi đó ta sẽ có hai hình vẽ phân biệt. Nếu ta không có các lệnh **figure**, MATLAB sẽ mặc định ta đang vẽ trên **figure(1)** và do đó, nếu ta vẽ nhiều hình thì cũng chỉ có hình cuối cùng được lưu lại.

- Các lệnh **hold on** và **hold off** cho MATLAB biết ta muốn giữ nguyên hiện trạng đang có của một hình và thực hiện các lệnh đè lên, chẳng hạn

```
figure(2)
plot(X1,Y1,'r')
hold on
plot(X2,Y2,'b')
hold off
```

Sẽ vẽ đồ thị của **(X1,Y1)** bằng màu đỏ, và đồ thị ứng với **(X2,Y2)** bằng màu xanh trên cùng một hình vẽ. Nếu không có các lệnh hold on, hold off, thì chỉ có đồ thị thứ hai được vẽ lại.

- Ta cũng có thể dùng lệnh **plot(X1,Y1,'r',X2,Y2,'b')** để vẽ đồ thị ứng với **(X1,Y1)** bằng màu đỏ, và đồ thị ứng với **(X2,Y2)** bằng màu xanh trên cùng một hình vẽ. Cách vẽ này thường dùng để so sánh hai (hoặc nhiều) hàm số với nhau.

Để vẽ hình trong 3 chiều, ta có thể các lệnh cho trường hợp 2 chiều và

- Lệnh **surf(X,Y,Z)** sẽ vẽ bằng cách nội suy tuyến tính các điểm **(X(m),Y(n),Z(n,m))**, trong đó **X,Y** là hai vector và **Z** là ma trận với cỡ tương thích. Lệnh **surfc(X,Y,Z)** có chức năng tương tự, nhưng thêm "cái bóng" xuống đáy đồ thị.

### 1.2.5 Một số lệnh khác

- Lệnh **clear all** dùng để xóa tất cả các lưu trữ tạm của MATLAB. Nó thường được dùng để bắt đầu file **.m** chứa chương trình chính của chúng ta (để mỗi lần chạy thì không bị ảnh hưởng bởi các lần chạy trước đó).
- Lệnh **disp('thong bao')** dùng để xuất các thông báo.

### 1.2.6 Các dạng thức (format) biểu diễn số

| | |
|---|---|
| Format short: | Số dấu phẩy cố định, với 5 chữ số có nghĩa sau dấu phẩy |
| Format long: | Số dấu phẩy cố định, với 15 chữ số có nghĩa sau dấu phẩy |
| Format short e: | Số dấu phẩy động, với 5 chữ số có nghĩa sau dấu phẩy |
| Format long e: | Số dấu phẩy động, với 15 chữ số có nghĩa sau dấu phẩy |
| Format short g: | Lựa chọn tốt nhất phẩy cố định hay động với 5 chữ số có nghĩa sau dấu phẩy. |
| Format long g: | Lựa chọn tốt nhất phẩy cố định hay động với 15 chữ số có nghĩa sau dấu phẩy. |
| Format rat: | Biểu thị số thực về số hữu tỷ gần nhất. |
| disp: | Hiển thị nội dung của biến. |
| Fprintf: | Hàm cho phép hiển thị theo các khuôn dạng chỉ định. |
| Sprintf: | Hàm trả về xâu ký tự in theo các khuôn dạng chỉ định. |

## 1.3. CÁC BÀI TOÁN

MATLAB là một ngôn ngữ lập trình rất dễ sử dụng, đặc biệt nếu ta đã biết dùng C hoặc Pascal. Nó cho phép làm được khá nhiều việc trong toán dựa trên một số rất ít các lệnh.

**Bài 1.3.1** Vẽ đồ thị hàm số **f(x)=x^2+sin(x)** trên đoạn [0,1] bằng cách lấy 11 điểm $x_i = (i-1)/10,\ i = \overline{1,10}$. Yêu cầu:

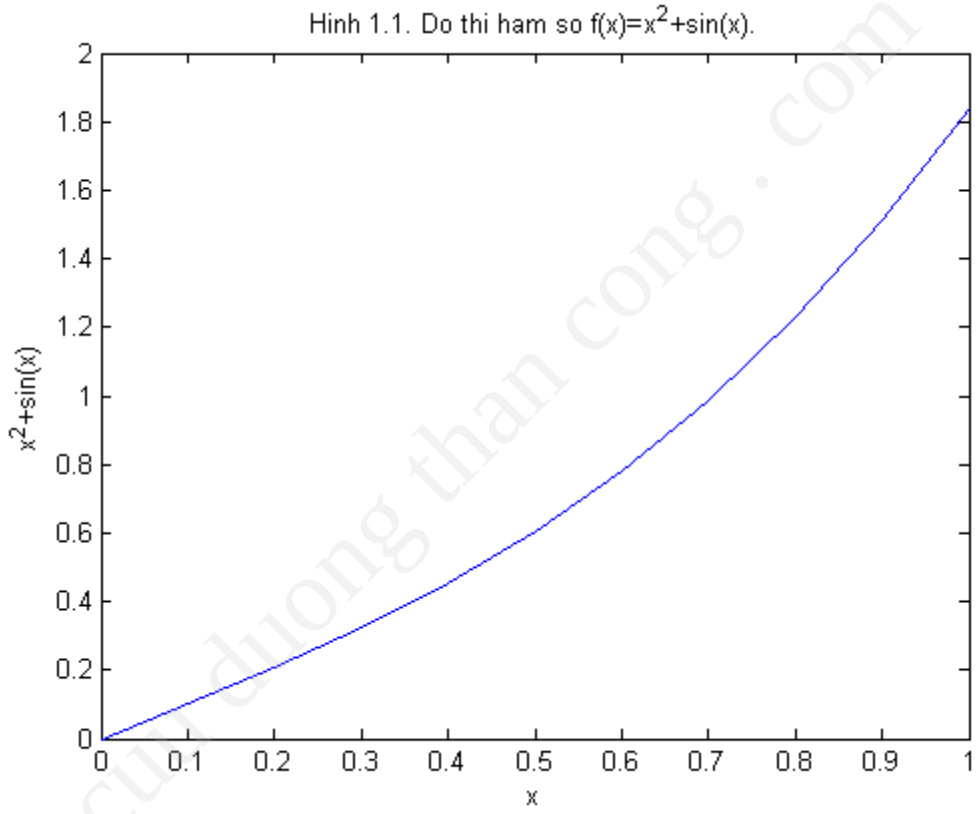
- Định nghĩa hàm số **f(x)=x^2+sin(x)** riêng trong file **f.m.** Sau đó vẽ đồ thị hàm số trong file **bai1.m.**
- Trên đồ thị ghi chú trục hoành là "**x**", trục tung là "**x^2+sin(x)",** đặt tựa đề là "**Đồ thị hàm số f(x)=x^2+sin(x)**".

**Chương trình:**

```
% file f.m
function a=f(x);
a=x^2+sin(x);

% file bai1.m
clear all
N=10;
X=[0:N]*1/N;
```

```
Y=zeros(1,N+1);
for i=1:(N+1)
        Y(i)=f(X(i));
end
plot(X,Y);
xlabel('x');
ylabel('x^2+sin(x)');
title('Hinh 1.1. Do thi ham so f(x)=x^2+sin(x).');
```

Chạy file **bai1.m**, các file này đã có sẵn trong thư mục **bai1.3.1**. Kết quả:




Để chạy chương trình, ta làm từng bước sau:

- Mở của sổ làm việc của MATLAB.
- Vào ô có chữ Current Directory, ghi đường dẫn chỉ tới thư mục, chẳng hạn D\CAOHOC\ddtrong\luanvan-ct\ch1\cacbaitoan**bai1.3.1\bai1**. Ta cũng có thể kích vào dấu **...** để MATLAB hiện cây thư mục rồi chọn thư mục D\CAOHOC\ddtrong\luanvan-ct\ch1\cacbaitoan**bai1.3.1\bai1**; tới dòng >> trên cửa sổ làm việc, gõ **bai1**, rồi Enter. Ta sẽ thu được hình vẽ như mong muốn.

- Để lưu hình vẽ này vào máy tính, ta mở hình vẽ đó, vào File\Save As, rồi đặt tên và chọn định dạng thích hợp (chẳng hạn **.bmp** hoặc **.ps**). Hình vẽ này sẽ được lưu vào thư mục hiện hành là D\CAOHOC\ddtrong\luanvan-ct\ch1\cacbaitoan**bai1.3.1**, với tên là hinh1.1.bmp.

**Bài 1.3.2** Ta biết rằng $x=\sum_{n=1}^{\infty}\frac{2(-1)^n}{n\pi}\sin(n\pi x), 0<x<1,$ theo nghĩa trong $L^2(0,1)$.

Yêu cầu:

- Viết file **s.m** để định nghĩa hàm s(x) như sau

$s(x)=\sum_{n=1}^{M}\frac{2(-1)^n}{n\pi}\sin(n\pi x)$, trong đó **M** là biến toàn cục (**global**).

- Viết file bai2.m, trong đó định nghĩa M=20. Trên đoạn [0,1] lấy N+1 điểm $x_i=(i-1)/10, i=\overline{1,N}$ với N=100. Vẽ các đồ thị s(x) và y=x trên cùng một hình vẽ.

- Tính xấp xỉ các sai số $\|s(x)-x\|_{L^2(0,1)}$ và $\|s(x)-x\|_{L^\infty(0,1)}$ dựa trên các dữ liệu rời rạc tại các điểm $x_i$. Nêu nhận xét. Để tính sai số, ta dùng các xấp xỉ

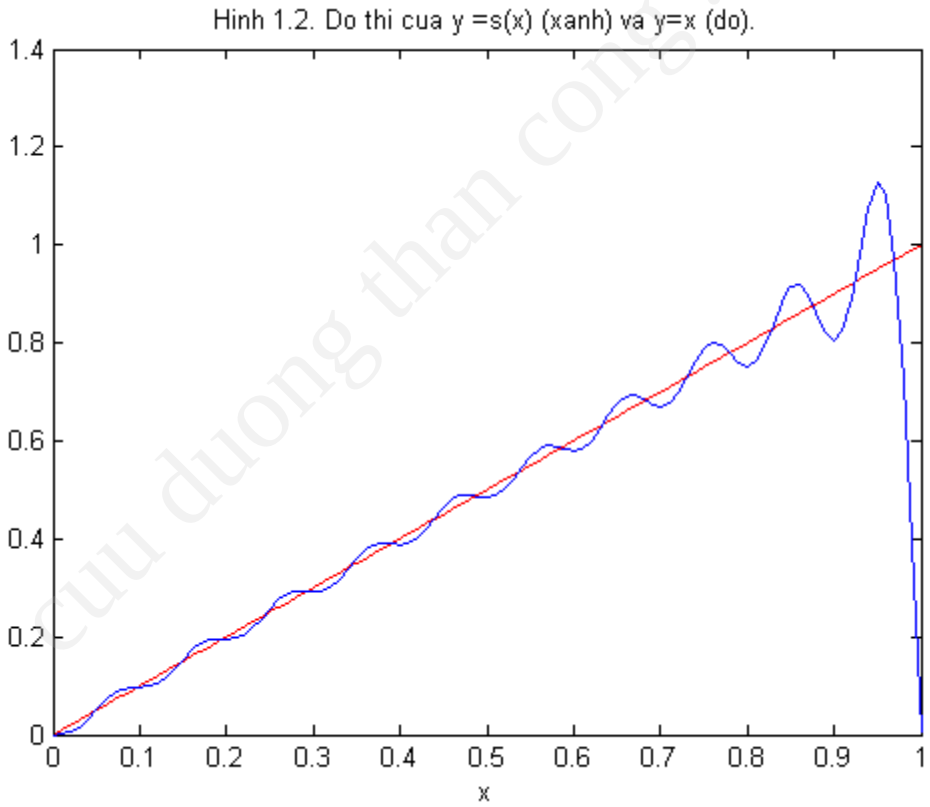
$$\|s(x)-x\|_{L^2}=\sqrt{\int_0^1|s(x)-x|^2\,dx}\approx\sqrt{\frac{1}{N+1}\sum_{i=1}^{N+1}|s(x_i)-x_i|^2},$$
$$\|s(x)-x\|_{L^\infty}=\sup_{x\in[0,1]}|s(x)-x|\approx\sup_{1\le i\le N+1}|s(x_i)-x_i|.$$

Chương trình**:** ***(các file có trong thư mục*** **bai1.3.2*)***

```
% file s.m
function a=s(x);
global M
a=0;
for n=1:M
   a=a+2*(-1)^(n+1)/n/pi*sin(n*pi*x);
end

% file bai2.m
clear all
```

```
global M
M=20; N=100;
X=[0:N]*1/N; Y=zeros(1,N+1);
for i=1:(N+1)
   Y(i)=s(X(i));
end
plot(X,X,'r',X,Y,'b');
xlabel('x');
title('Hinh 1.2. Do thi cua y =s(x) (xanh) va y=x (do). ');
disp('error in L^2');
err2=norm(X-Y)/sqrt(N+1)
disp('error in L^infinity');
errsup=norm(X-Y,inf)
```

Kết quả: (chạy file **bai2.m**; tất cả các file có trong thư mục **bai1.3.2**)




Sai số: err2 = 0.1242, errsup =1.

Nhận xét: sự xấp xỉ trong $L^2$ khá tốt nhưng xấp xỉ trong $L^\infty$ thì không tốt, nguyên nhân là tại điểm x=1 thì s(1)=0.

Ta cũng có thể vẽ đồ thị bằng cấu trúc lệnh đơn giản hơn như chương trình vẽ đồ thị của bài 1.3.3. Cũng từ bài 1.3.3 này chúng tôi sẽ làm rõ một vài vấn đề về MATLAB.

**Bài 1.3.3** Vẽ đồ thị hàm số

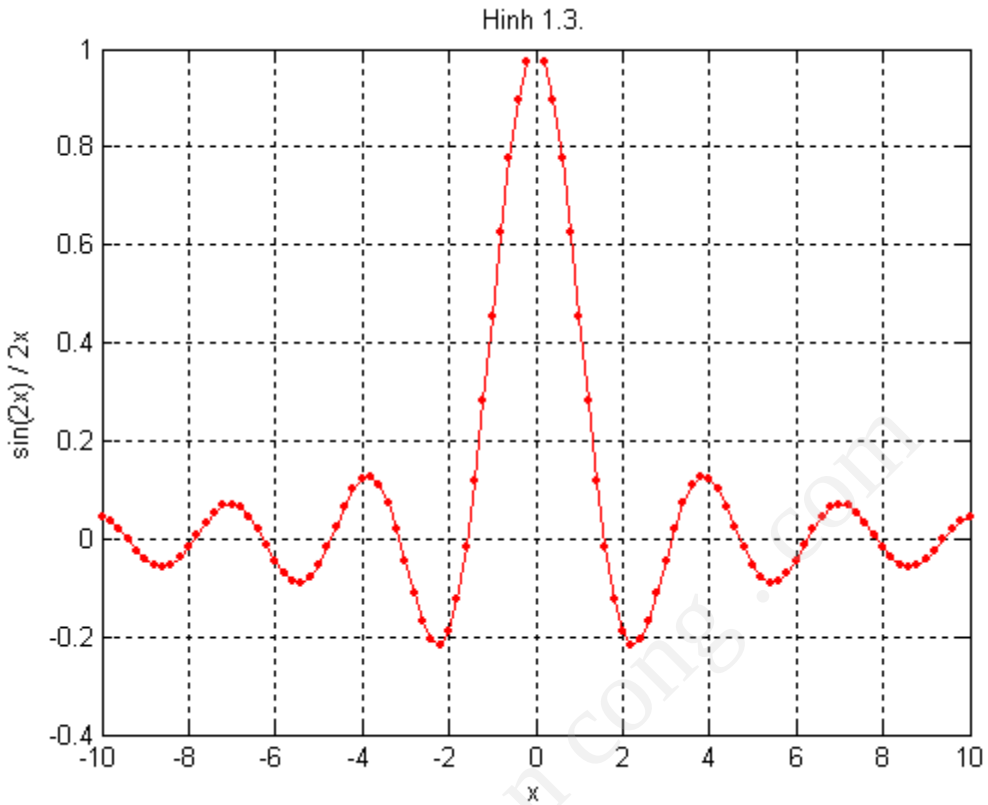
$$y(x)=\frac{\sin(2x)}{2x}, \text{ với } -10\le x\le 10.$$

**Chương trình:** ***(các file có trong thư mục* bai1.3.3*).***

```
% file bai3.m
x=-10:0.2:10;          % Chia [-10,10]thanh 101 diem, buoc nhay 0.2.
y=1/2*sin(2*x)./x;
%  y la vecto co cung chieu dai cua vecto x, mang gia tri cua ham.
plot(x,y,'r.-')        % Ve do thi voi mau do, duong .-.
grid                   % Ve cac duong dong tren do thi.
title('Hinh 1.3.')     % Dua tieu de len tren do thi.
xlabel('x')            % Dua nhan tren truc x.
ylabel('sin(2x)/2x')   % Dua nhan tren truc y.
```

Chạy file bai3.m ta được Hình 1.3.

Khi vẽ đồ thị, hàm số đã cho được lượng giá thành công ứng với mỗi phần tử trong ma trận x, ma trận $(1\times 101)$; ngoại trừ $x(51)=0$, nơi mà có phép chia cho số 0 xảy ra. Điều này có ý nghĩa gì không? Bằng cách viết một mở rộng chuỗi Taylor (sẽ được trình bày chi tiết trong chương 2) cho $\sin(2x)$ và chia toàn bộ cho $2x$, ta dễ dàng chứng tỏ rằng $y(0)=\sin(0)/0=1$. MATLAB không dừng lại ở điểm này, hàm plot() xử lý phần tử $y(51)$ dưới dạng dữ liệu bị thiếu (xem Hình 1.3)

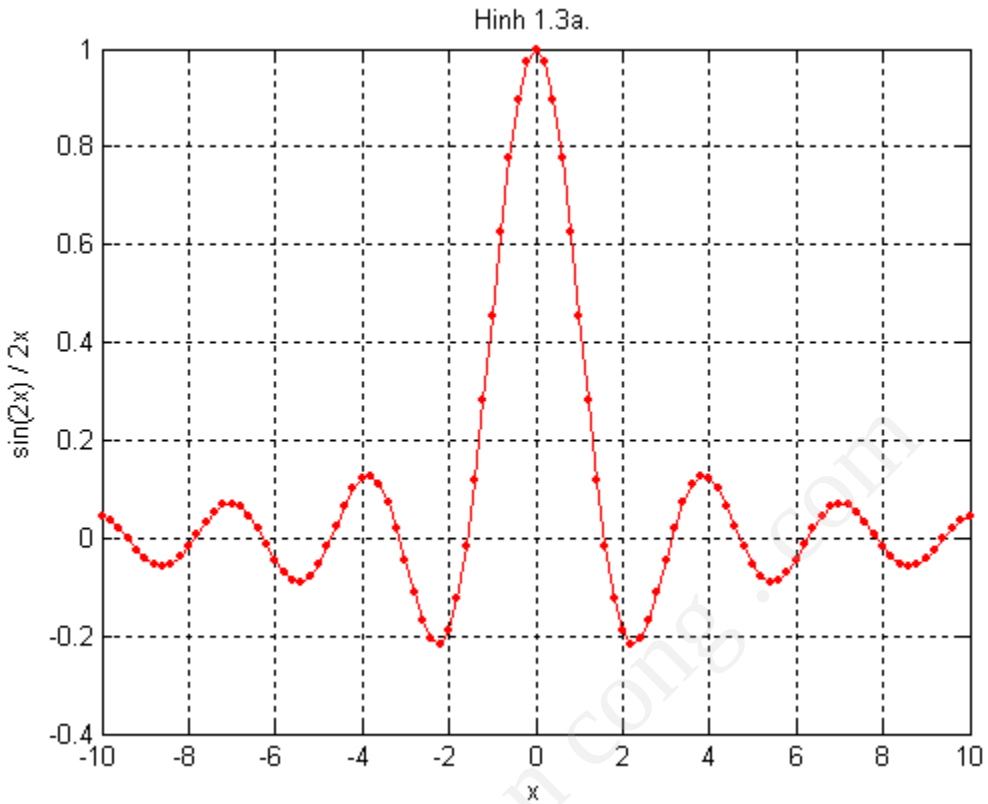



Một cách để khắc phục vấn đề tại $x(51)=0$, đó là đặt các phần tử zero trong ma trận x với eps, có nghĩa là,

```
x=-10:0.2:10;
x=x+(x==0)*eps;
```

Trong lệnh MATLAB thứ hai, một thành phần của ma trận x sẽ được tăng bởi eps lúc biểu thức logic (x==0) lượng giá sang true. Ngược lại thì các thành phần của x vẫn giữ không thay đổi. Bây giờ ma trận y được tạo ra không có lỗi và $\sin(0)/0$ lượng giá sang 1.

Ta có file bai3a.m, chỉ cần thêm dòng lệnh **x=x+(x==0)*eps;** từ file bai3.m. Chạy file bai3a.m, ta được Hình 1.3a, và chúng ta đã giải quyết được vấn đề tại $x(51)=0$.




Bây giờ ta nói thêm về các trục của đồ thị. Trong Hình 1.3 và Hình 1.3a, MATLAB đã ấn định tự động dãy các giá trị x và y để cho đồ thị được vẽ lắp đầy khoảng trống có sẵn (tức là xmin=min(x), xmax=max(x)). Để thực hiện điều đó, hạng mục dữ liệu thiếu tại y=0 phải được che giấu bởi các đường biên của hình Hình 1.3 và Hình 1.3a. Có lẽ điều này sẽ hiển nhiên hơn khi dãy các giá trị của y trong Hình 1.3 và Hình 1.3a được định lại tỷ lệ để chiếm trọn đoạn [-0.4, 1.2].

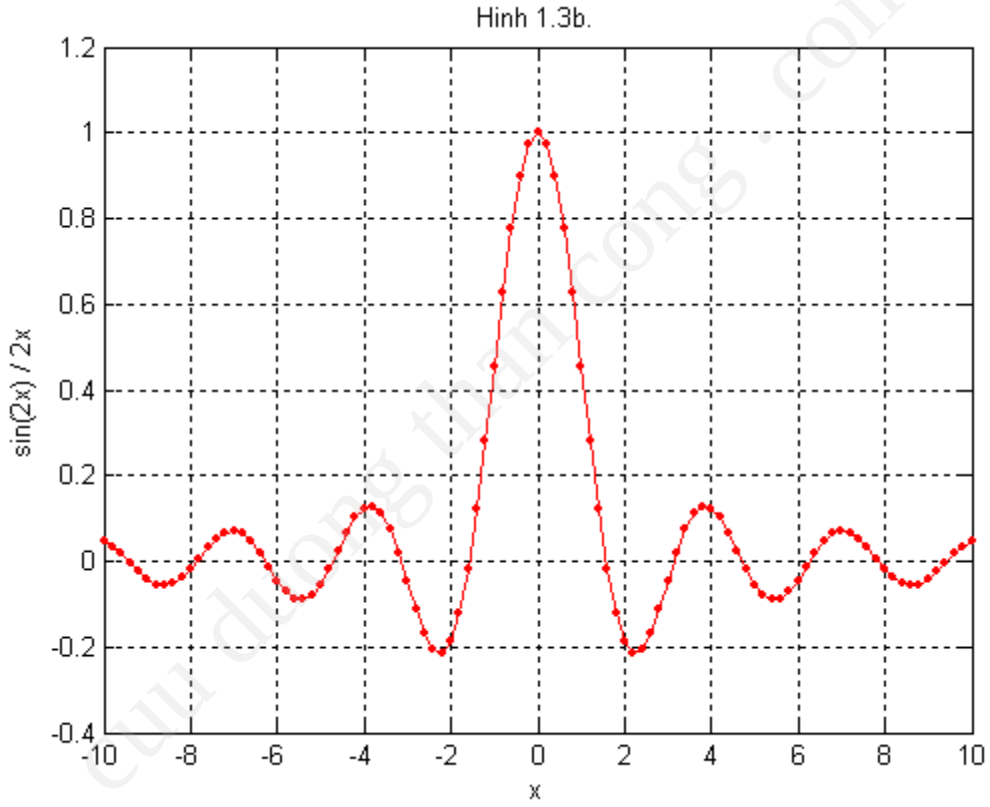
Như phần trên đã nêu, MATLAB cung cấp hàm axis để kiểm soát tỷ lệ và diện mạo của các trục trong đồ thị. Trong phần này chúng ta dùng phép gọi hàm axis([xmin,xmax,ymin,ymax]) để cài đặt dãy các giá trị x và y trong biểu đồ hiện tại sang [xmin, xmax] và [ymin,ymax] tương ứng.

Ngoài ra, lệnh axis ('equal') thay đổi kích thước của hộp ô trục hiện tại để cho các gia số của dấu kiểm trên các trục x và y đều bằng nhau. Tương tự như vậy,

lệnh axis('off') sẽ tắt tất cả các nhãn trên trục và các dấu kiểm. Các nhãn trên trục và các dấu kiểm được đưa trở về một lần nữa với axis('on').

Với file bai3b.m, chỉ cần thêm dòng lệnh **axis([-10 10 -0.4 1.2])** từ file bai3a.m, sẽ tạo ra Hình 1.3b đồng nhất với Hình 1.3 và Hình 1.3a, trong trường hợp này với các dãy giá trị của x và y bao gồm các đoạn [-10,10] và [-0.4,1.2] tương ứng và việc lượng giá đúng $\sin(2x)/2x$ tại $x=0$.

Chạy file bai3b.m ta được Hình 1.3b.




**Ghi chú:** Trên đây là phần giới thiệu rất sơ lược về MATLAB, các thông tin chuyên sâu hơn có thể tìm thấy ở **http://www.mathworks.com**; **http://www.math.mtu.edu**; **http://www.math.ufl.edu/help/matlab-tutorial**.

# CHƯƠNG 2
# ĐA THỨC TAYLOR

## 2.1. ĐA THỨC TAYLOR

Hầu hết các hàm số $f(x)$ trong toán học không thể được đánh giá một cách chính xác. Ví dụ, hãy xem xét giá trị của hàm số $f(x)=\cos x$, $e^x$, hoặc $\sqrt{x}$ mà không cần sử dụng máy tính hoặc máy vi tính. Để xác định giá trị các biểu thức như thế, chúng ta sử dụng hàm $f(x)$ gần bằng với $f(x)$ và dễ xác định giá trị hơn. Loại phổ biến nhất của hàm xấp xỉ $f(x)$ là các đa thức. Chúng dễ thực hiện và chúng là phương thức hữu hiệu để tính gần đúng $f(x)$. Trong số các đa thức, đa thức Taylor được sử dụng rộng rãi nhất. Đa thức Taylor tương đối dễ xây dựng, và nó thường là bước đầu tiên để đạt được các phép tính xấp xỉ hiệu quả hơn. Đa thức Taylor cũng quan trọng trong nhiều lĩnh vực khác của toán học.

Trong chương này, chúng tôi sẽ trình bày tóm tắt lý thuyết về chuỗi Taylor và trình bày cách sử dụng MATLAB như là một công cụ thực nghiệm nhằm tạo cho sinh viên một trực giác về sự xấp xỉ địa phương của các hàm số.

Với $f(x)$ là một hàm số cho trước, đa thức Taylor được xây dựng để phỏng theo sự biến thiên của $f(x)$ tại một điểm x = a nào đó . Kết quả, nó sẽ gần bằng $f(x)$ tại các điểm x gần với a. Cụ thể hơn, ta tìm một đa thức bậc nhất $p_1(x)$ để

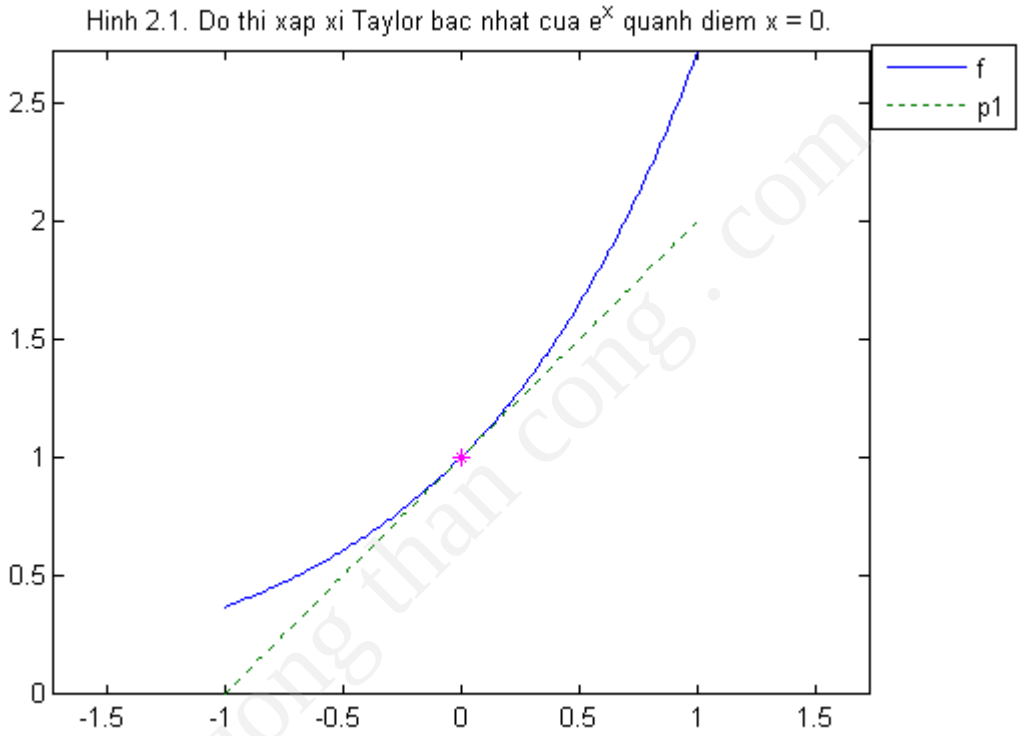
$$\begin{aligned} p_1(a) &= f(a), \\ p'_1(a) &= f'(a). \end{aligned} \tag{2.1}$$

Ta dễ dàng xác định duy nhất đa thức được cho bởi

$$p_1(x) = f(a) + (x-a).f'(a). \qquad (2.2)$$

Và như vậy, $y = p_1(x)$ là tiếp tuyến của đồ thị $y = f(x)$ tại $x = a$.

**Ví dụ 2.1.1** Cho $f(x) = e^x$ và $a = 0$, thì $p_1(x) = 1 + x$. Đồ thị của $f$ và $p_1$ được thể hiện trong Hình 2.1. Chú ý rằng $p_1(x)$ thì xấp xỉ với $e^x$ khi $x$ gần 0.




Tiếp tục cách làm trên, tìm đa thức bậc hai $p_2(x)$ là hàm xấp xỉ $f$(x) gần $x = a$. Vì có 3 hệ số trong công thức của đa thức bậc 2, nên ta có

$$p_2(x) = b_0 + b_1 x + b_2 x^2.$$

Để mô phỏng tốt hơn sự biến thiên của $f(x)$ tại x = a, chúng ta cần

$$\begin{aligned} p_2(a) &= f(a), \\ p'_2(a) &= f'(a), \\ p''_2(a) &= f''(a). \end{aligned} \qquad (2.3)$$

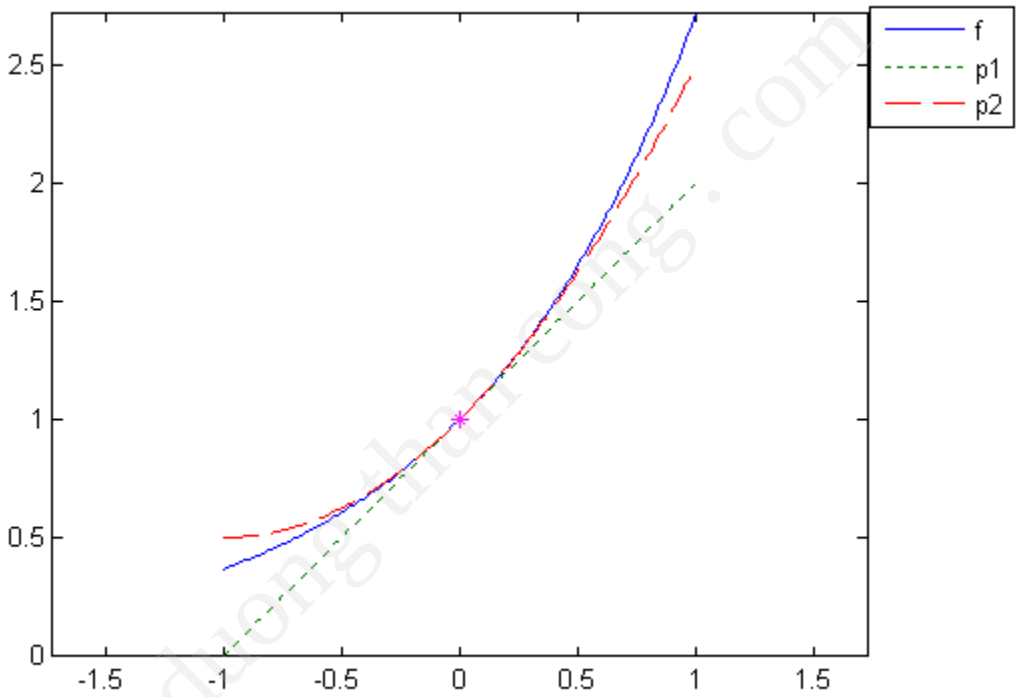
Ta được, $$p_2(x) = f(a) + (x-a)f'(a) + \frac{1}{2}(x-a)^2 f''(a). \qquad (2.4)$$

**Ví dụ 2.1.2** Tiếp tục ví dụ trước $f(x) = e^x$ và a = 0, chúng ta có

$$p_2(x) = 1 + x + \frac{1}{2}x^2.$$

Xem Hình 2.2 để so sánh $f(x) = e^x$ và $p_1(x) = 1 + x$, $p_2(x) = 1 + x + \frac{1}{2}x^2$.

Hinh 2.2. Do thi xap xi Taylor bac nhat va bac hai cua $e^x$ quanh diem x = 0.




Tiếp tục cách thức trên, cho $p_n(x)$ là đa thức bậc n, và nó cần thoả

$$p_n^{(j)}(a) = f^{(j)}(a), \text{ với } j = \overline{1,n}. \qquad (2.5)$$

Với $f^{(j)}(x)$ là đạo hàm bậc j của $f(x)$ và quy ước $f^{(0)}(a) = f(a)$. Thì

$$\begin{aligned} p_n(x) &= f(a) + (x-a)f'(a) + \frac{(x-a)^2}{2!}f''(a) + \ldots + \frac{(x-a)^n}{n!}f^{(n)}(a) \\ &= \sum_{j=0}^{n} \frac{(x-a)^j}{j!} f^{(j)}(a). \end{aligned} \qquad (2.6)$$

Nếu trong (2.6) chúng ta cần ghi chú một cách rõ ràng sự phụ thuộc của khai triển vào điểm a, chúng ta viết $p_n(x;a)$. Đa thức $p_n(x)$ trong (2.6) được gọi là đa thức Taylor bậc n đối với hàm số $f(x)$ và điểm xấp xỉ a. [ Tuy nhiên, chú ý rằng đa thức $p_n(x)$ có bậc thật sự nhỏ hơn n nếu $f^{(n)}(a)=0$ ]

**Ví dụ 2.1.3** Lại cho $f(x)=e^x$ và a = 0 , thì $f^{(j)}(x)=e^x$, $f^{(j)}(0)=1, \forall j \geq 0$

$$p_n(x)=1+x+\frac{1}{2!}x^2+...+\frac{1}{n!}x^n=\sum_{j=0}^{n}\frac{x^j}{j!}. \qquad (2.7)$$

Bảng 2.1 chứa các giá trị của $p_1(x)$, $p_2(x)$, $p_3(x)$ và $e^x$ tại những giá trị khác nhau của x trong đoạn [-1, 1]. Với a cố định x, độ chính xác tăng khi bậc n tăng. Và với bậc của đa thức cố định, độ chính xác giảm khi x di chuyển ra xa a = 0.

**Bảng 2.1.** Xấp xỉ Taylor của $e^x$ quanh điểm x = 0.

| $x$ | $p_1(x)$ | $p_2(x)$ | $p_3(x)$ | $e^x$ |
|---|---|---|---|---|
| -1.0 | 0 | 0.500 | 0.33333 | 0.36788 |
| -0.5 | 0.5 | 0.625 | 0.60417 | 0.60653 |
| -0.1 | 0.9 | 0.905 | 0.90483 | 0.90484 |
| 0 | 1.0 | 1.000 | 1.00000 | 1.00000 |
| 0.1 | 1.1 | 1.105 | 1.10517 | 1.10517 |
| 0.5 | 1.5 | 1.625 | 1.64583 | 1.64872 |
| 1.0 | 2.0 | 2.500 | 2.66667 | 2.71828 |

**Ví dụ 2.1.4** Cho $f(x)=e^x$ và điểm a bất kỳ, không nhất thiết là 0, thì $f^{(j)}(x)=e^x$, $f^{(j)}(0)=1, \forall j \geq 0$. Chúng ta có công thức

$$p_n(x;a)=e^a[1+(x-a)+\frac{1}{2!}(x-a)^2+...+\frac{1}{n!}(x-a)^n=e^a\sum_{j=0}^{n}\frac{(x-a)^j}{j!}.$$

Và như vậy,

$$p_n(x;1) = e^1 \sum_{j=0}^{n} \frac{(x-1)^j}{j!}. \tag{2.8}$$

Đa thức $p_n(x;1)$ xấp xỉ tốt nhất khi x gần bằng 1, và đa thức $p_n(x;0)$ [được cho trong (2.7)] thì xấp xỉ tốt nhất khi x gần bằng 0.

Ở đây, khi thực hiện tính toán số, chúng tôi thay thế cách xác định giá trị một hàm số, chẳng hạn như $e^x$ với cách xác định giá trị một đa thức.

**Ví dụ 2.1.5** Cho $f(x) = \log(x)$, tức là $f(x) = \ln(x)$, và a =1, thì $f(1) = 0$. Bằng phương pháp quy nạp, với $j \geq 1$, ta có

$$f^{(j)}(x) = (-1)^{j-1}(j-1)!\frac{1}{x^j}. \text{ Suy ra, } f^{(j)}(1) = (-1)^{j-1}(j-1)!.$$

Nếu điều này được áp dụng vào (2.6), thì đa thức Taylor được cho bởi

$$\begin{aligned} p_n(x) &= (x-1) - \frac{1}{2}(x-1)^2 + \frac{1}{3}(x-1)^3 - \ldots + (-1)^{n-1}\frac{1}{n}(x-1)^n \\ &= \sum_{j=0}^{n} \frac{(-1)^{j-1}}{j}(x-1)^j. \end{aligned} \tag{2.9}$$

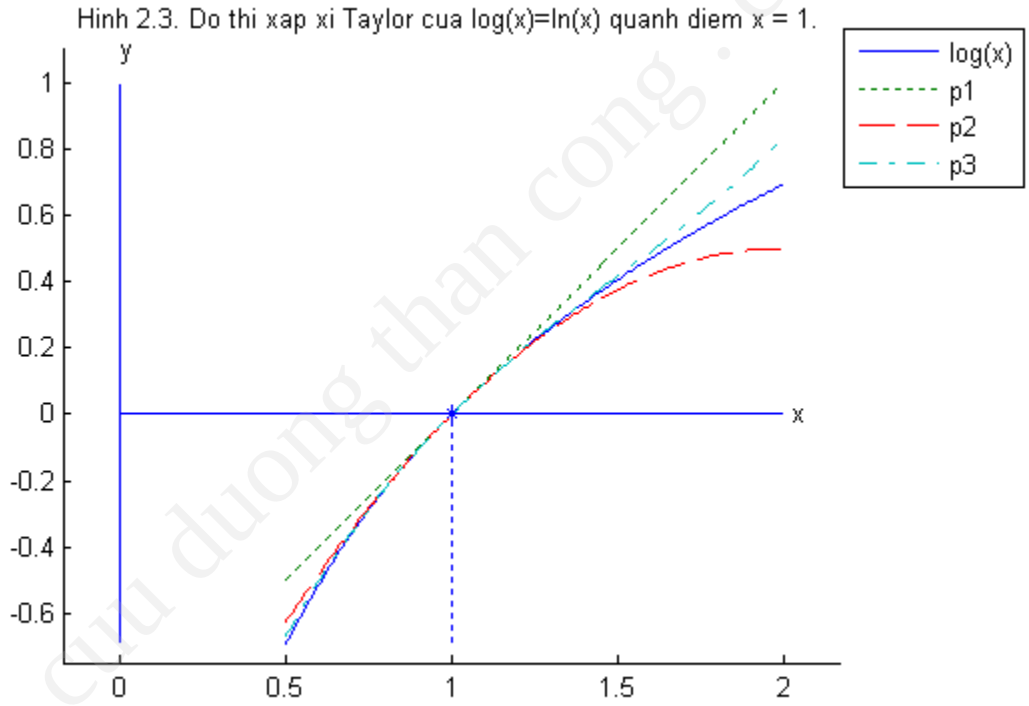
Đồ thị của log(x) và các đa thức Taylor $p_1(x)$, $p_2(x)$ và $p_3(x)$ được thể hiện trên Hình 2.3.

Trong bài viết này, chúng tôi dùng 2 ký hiệu có nghĩa là "gần bằng". Ký hiệu "$\approx$" thì thường được dùng chỉ sự gần đúng của các biểu thức đại số, ví dụ $x \approx 5$, nghĩa là x gần bằng 5; và $e^x \approx 1+x,\ x \approx 0$, nghĩa là $e^x$ gần bằng $1+x$ khi x gần bằng 0. Ký hiệu "$\doteq$" thường được dùng với số, như $2\pi \doteq 6.2832;\ \sqrt{168} \doteq 12.961$. Ký hiệu "$\doteq$" thường được dùng với độ sai số tính toán thực sự.

Trong MATLAB, log(x) là logarit tự nhiên ln(x) và log10(x) là logarit thập phân lg(x).

Xuyên suốt bài viết này, chúng tôi sẽ phát biểu một vài nhận xét chung hoặc các qui tắc để sử dụng khi xem xét giải số của các bài toán.

NHẬN XÉT CHUNG: Khi xem xét cách giải một bài toán, nếu không biết phương pháp giải trực tiếp bài toán đó thì hãy thay thế bằng một "bài toán xấp xỉ" mà có thể giải bằng máy tính. (2.10)




**CHƯƠNG TRÌNH MATLAB:** Sau đây là chương trình MATLAB, chương trình này sẽ tính toán một vài xấp xỉ đa thức Taylor của $e^x$ nằm trong đoạn [-b, b], với giá trị của b được đưa vào khi chạy chương trình. Chương trình sẽ xác định giá trị các đa thức Taylor bậc 1, 2, 3 và 4 tại các điểm x được chọn trong đoạn [-b, b],

in giá trị x, giá trị đúng của $e^x$ và các sai số của chúng theo dạng biểu bảng. Kết quả sẽ được xuất ra trên cả màn hình đang sử dụng khi chạy file exp_taylor_simple.m trong cùng thư mục với file polyeval.m.

**Chương trình 2.1:** Xác định giá trị đa thức Taylor.

```
%TIEU DE: Uoc luong da thuc Taylor cua exp(x) xung quanh x = 0.
% Vai uoc luong cua da thuc Taylor va sai so cua no voi bac tang dan. Xap xi % cua ham exp(x) tren doan [-b,b].
% Initialize: Khoi chay
b = input('Nhap so b vao de tao doan [-b,b], b= ');
h = b/10;                 % Khoang cach giua hai so
x = -b:h:b;               % x di tu -b den b voi buoc nhay la h
max_deg = 4;              % bac cao nhat bang 4

% Tao ra cac he so cua ham exp(x) khi mo rong quanh diem a = 0.
% Nhung he so nay duoc luu tru trong mang c, voi do dai la max_deg+1.
c=ones(max_deg+1,1);      % c la ma tran don vi, max_deg+1 dong, mot cot
fact = 1;                 % Gan cho bien fact gia tri bang 1
for i = 1:max_deg  %Thuc hien vong lap for voi gia tri i di tu 1 den max_deg
  fact = i*fact;    % Lan luot thay the gia tri cua bien fact bang i nhan voi fact;
  c(i+1) = 1/fact;          % Gan gia tri cho cac phan tu c(i+1) bang 1/fact;
end                       % Ket thuc vong lap
% Tinh gia tri cua da thuc Taylor bac1, bac2, bac3, bac4 bang cach goi ham % polyeval la mot file trong cung thu muc.
p1 = polyeval(x,0,c,1); p2 = polyeval(x,0,c,2);
p3 = polyeval(x,0,c,3); p4 = polyeval(x,0,c,4);
% Tinh gia tri dung cua e^x va cac sai so trong khai trien da thuc Taylor
%  tuong ung tung bac o tren
true = exp(x); err1 = true-p1; err2 = true-p2; err3 = true-p3; err4 = true-p4;

% Ghi ket qua ra mang hinh. Ket qua ghi thanh mot ma tran, voi cac cot  %tuong ung voi gia tri x, gia tri chinh xac, sai so bac1, sai so bac2, sai so % bac3, sai so bac 4. Ung voi moi gia tri x chay tren doan [-b,b] thi cho ra 1  % bang ket qua tuong ung.
diary exp_taylor          % Ghi nho cac ket qua theo dinh dang ben duoi.
disp('  x      exp(x)    err1      err2      err3      err4')
[x',true',err1',err2',err3',err4']
diary off                 % Ket thuc viec ghi nho.
```

Kết quả chạy file exp_taylor_simple.m, với **b = 1**.

Nhap so b vao de tao doan [-b,b], **b = 1**

| x | exp(x) | err1 | err2 | err3 | err4 |
|---|---|---|---|---|---|
| ans = | | | | | |
| -1.0000 | 0.3679 | 0.3679 | -0.1321 | 0.0345 | -0.0071 |
| -0.9000 | 0.4066 | 0.3066 | -0.0984 | 0.0231 | -0.0043 |
| -0.8000 | 0.4493 | 0.2493 | -0.0707 | 0.0147 | -0.0024 |
| -0.7000 | 0.4966 | 0.1966 | -0.0484 | 0.0088 | -0.0013 |
| -0.6000 | 0.5488 | 0.1488 | -0.0312 | 0.0048 | -0.0006 |
| -0.5000 | 0.6065 | 0.1065 | -0.0185 | 0.0024 | -0.0002 |
| -0.4000 | 0.6703 | 0.0703 | -0.0097 | 0.0010 | -0.0001 |
| -0.3000 | 0.7408 | 0.0408 | -0.0042 | 0.0003 | -0.0000 |
| -0.2000 | 0.8187 | 0.0187 | -0.0013 | 0.0001 | -0.0000 |
| -0.1000 | 0.9048 | 0.0048 | -0.0002 | 0.0000 | -0.0000 |
| 0 | 1.0000 | 0 | 0 | 0 | 0 |
| 0.1000 | 1.1052 | 0.0052 | 0.0002 | 0.0000 | 0.0000 |
| 0.2000 | 1.2214 | 0.0214 | 0.0014 | 0.0001 | 0.0000 |
| 0.3000 | 1.3499 | 0.0499 | 0.0049 | 0.0004 | 0.0000 |
| 0.4000 | 1.4918 | 0.0918 | 0.0118 | 0.0012 | 0.0001 |
| 0.5000 | 1.6487 | 0.1487 | 0.0237 | 0.0029 | 0.0003 |
| 0.6000 | 1.8221 | 0.2221 | 0.0421 | 0.0061 | 0.0007 |
| 0.7000 | 2.0138 | 0.3138 | 0.0688 | 0.0116 | 0.0016 |
| 0.8000 | 2.2255 | 0.4255 | 0.1055 | 0.0202 | 0.0031 |
| 0.9000 | 2.4596 | 0.5596 | 0.1546 | 0.0331 | 0.0058 |
| 1.0000 | 2.7183 | 0.7183 | 0.2183 | 0.0516 | 0.0099 |

Ta có thể định dạng lại số xuất ra màn hình và vị trí tương ứng của các số bằng cách thay đoạn chương trình cuối bởi

```
diary exp_taylor
disp('   x      exp(x)     err1        err2        err3        err4')
for i=1:length(x)
   fprintf('%7.3f%10.3f%14.3e%14.3e%14.3e%14.3e\n',...
      x(i),true(i),err1(i),err2(i),err3(i),err4(i))
end
diary off
```

Rồi lưu lại với tên exp_taylor.m.

Thì ta được bảng kết quả sau: (chạy file exp_taylor.m )

Nhap so b vao de tao doan [-b,b], **b= 1.**

| x | exp(x) | err1 | err2 | err3 | err4 |
|---|---|---|---|---|---|
| -1.000 | 0.368 | 3.679e-001 | -1.321e-001 | 3.455e-002 | -7.121e-003 |
| -0.900 | 0.407 | 3.066e-001 | -9.843e-002 | 2.307e-002 | -4.268e-003 |
| -0.800 | 0.449 | 2.493e-001 | -7.067e-002 | 1.466e-002 | -2.404e-003 |
| -0.700 | 0.497 | 1.966e-001 | -4.841e-002 | 8.752e-003 | -1.252e-003 |
| -0.600 | 0.549 | 1.488e-001 | -3.119e-002 | 4.812e-003 | -5.884e-004 |
| -0.500 | 0.607 | 1.065e-001 | -1.847e-002 | 2.364e-003 | -2.402e-004 |
| -0.400 | 0.670 | 7.032e-002 | -9.680e-003 | 9.867e-004 | -7.995e-005 |
| -0.300 | 0.741 | 4.082e-002 | -4.182e-003 | 3.182e-004 | -1.928e-005 |
| -0.200 | 0.819 | 1.873e-002 | -1.269e-003 | 6.409e-005 | -2.580e-006 |
| -0.100 | 0.905 | 4.837e-003 | -1.626e-004 | 4.085e-006 | -8.196e-008 |
| 0.000 | 1.000 | 0.000e+000 | 0.000e+000 | 0.000e+000 | 0.000e+000 |
| 0.100 | 1.105 | 5.171e-003 | 1.709e-004 | 4.251e-006 | 8.474e-008 |
| 0.200 | 1.221 | 2.140e-002 | 1.403e-003 | 6.942e-005 | 2.758e-006 |
| 0.300 | 1.350 | 4.986e-002 | 4.859e-003 | 3.588e-004 | 2.131e-005 |
| 0.400 | 1.492 | 9.182e-002 | 1.182e-002 | 1.158e-003 | 9.136e-005 |
| 0.500 | 1.649 | 1.487e-001 | 2.372e-002 | 2.888e-003 | 2.838e-004 |
| 0.600 | 1.822 | 2.221e-001 | 4.212e-002 | 6.119e-003 | 7.188e-004 |
| 0.700 | 2.014 | 3.138e-001 | 6.875e-002 | 1.159e-002 | 1.582e-003 |
| 0.800 | 2.226 | 4.255e-001 | 1.055e-001 | 2.021e-002 | 3.141e-003 |
| 0.900 | 2.460 | 5.596e-001 | 1.546e-001 | 3.310e-002 | 5.766e-003 |
| 1.000 | 2.718 | 7.183e-001 | 2.183e-001 | 5.162e-002 | 9.948e-003 |

Để chạy được chương trình trên ta cần tạo hàm polyeval rồi lưu vào cùng thư mục với file chương trình trên, được đặt tên là polyeval.m, để xác định giá trị các đa thức. Và nó cũng được dùng chung cho các chương trình MATLAB của chương này.

**Chương trình 2.2:**

```
% Chuong trinh tao file polyeval.m
function value=polyeval(x,alpha,coeff,n);
% Ham tinh gia tri da thuc gom 4 bien duoc dua vao. Tinh toan cac gia tri da % thuc Taylor tai cac diem duoc cho boi x, cach tinh theo ly thuyet trinh bay %trong muc 2.3, voi alpha la diem tham chieu mo rong cua da thuc Taylor %(alpha tung ung a trong ly thuyet), va n la bac toi da cua da thuc. Cac he %so duoc dua vao thong qua bien coeff; va no bao gom n+1 phan tu coeff voi %coeff(1) la mot hang so (gioi han) trong da thuc.
value=coeff(n+1)*ones(size(x));
z=x-alpha;
for i=n:-1:1
```

```
 value = coeff(i) + z.*value;
end
```

## 2.2. SAI SỐ TRONG ĐA THỨC TAYLOR

Thực hành sử dụng phép tính xấp xỉ đa thức Taylor đối với $f(x)$, chúng ta cần biết tính chính xác của nó. Định lý sau đây cho chúng ta công cụ chính để đánh giá tính chính xác này. Chúng tôi trình bày chúng mà không chứng minh vì nó được ghi trong hầu hết các giáo trình toán.

**Định lý 2.2.1** (Định lý số dư Taylor) *Thừa nhận rằng* $f(x)$ *có đạo hàm liên tục trong đoạn* $\alpha \le x \le \beta$, *và cho điểm a thuộc đoạn đó. Đối với đa thức Taylor* $p_n(x)$ *(2.6), cho* $R_n(x) \equiv f(x) - p_n(x)$ *biểu thị số dư trong phép tính xấp xỉ* $f(x)$ *bởi* $p_n(x)$. *Thì*

$$R_n(x) = \frac{(x-a)^{n+1}}{(n+1)!} f^{(n+1)}(c_x), \quad \alpha \le x \le \beta. \tag{2.11}$$

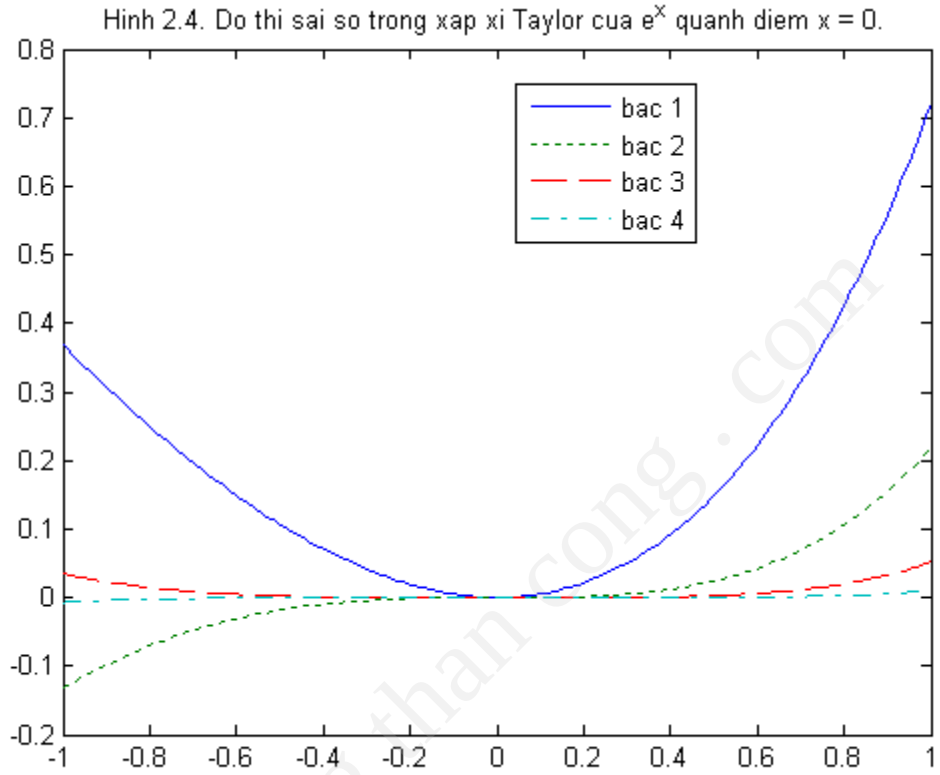
*Với* $c_x$ *là một số nằm giữa a và x.*

**Ví dụ 2.2.2** Cho $f(x) = e^x$ và a = 0, đa thức Taylor được cho trong (2.7). Từ định lý trên, sai số xấp xỉ được cho bởi

$$e^x - p_n(x) = \frac{x^{n+1}}{(n+1)!}.e^c, \quad n \ge 0. \tag{2.12}$$

Với c nằm giữa 0 và x. Từ công thức này, chúng ta có thể chứng minh rằng đối với mỗi điểm x cố định, sai số tiến về 0 khi $n \to \infty$; càng rõ ràng hơn khi $|x| \le 1$. Cũng từ công thức, cho thấy rằng đối với mỗi giá trị cố định của n, sai số trở nên lớn hơn khi x di chuyển ra xa 0. Minh họa điều này bằng đồ thị, và ta vẽ đồ thị sai số $e^x - p_n(x)$ sẽ tốt hơn vẽ đồ thị đơn giản của hàm số $e^x$ và đa thức $p_n(x)$

(phần 2.1), để quan sát sự xấp xỉ của chúng. Điều này được minh họa trong Hình 2.4 cho các sai số bậc n=1, 2, 3, 4 nằm trong đoạn [-1,1].




Rõ ràng, bậc càng cao thì sai số càng nhỏ.

**Ví dụ 2.2.3** Trường hợp đặc biệt của (2.12), cho x=1. Thì từ (2.7)

$$e \approx p_n(1) = 1+1+\frac{1}{2!}+\frac{1}{3!}+\ldots+\frac{1}{n!}.$$

Và từ (2.12)

$$e - p_n(1) = R_n(1) = \frac{e^c}{(n+1)!}.$$

Vì e<3, chúng ta có thể giới hạn $R_n(1)$ như sau

$$\frac{1}{(n+1)!} \le R_n(1) \le \frac{e}{(n+1)!} < \frac{3}{(n+1)!}.$$

Điều này sử dụng bất đẳng thức $e^0 \le e^c \le e^1$. Như một ví dụ số thực, giả định rằng chúng ta muốn tính xấp xỉ $e$ bởi $p_n(1)$ với $R_n(1) \le 10^{-9}$. Do đó, để có thể đạt được sai số mong, giới hạn trên cần thỏa

$$\frac{3}{(n+1)!} \le 10^{-9}.$$

Điều này đúng khi $n \ge 12$; vì vậy, $p_{12}(1)$ là một phép tính xấp xỉ có độ chính xác cao đối với $e$.

Công thức (2.6) và (2.11) có thể được dùng để hình thành phép tính xấp xỉ và công thức số dư cho hầu hết các hàm số chuẩn mà hay gặp trong các bài toán của sinh viên đại học.

Để tham khảo sau này, chúng tôi nêu một vài công thức thường gặp

$$e^x = 1 + x + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + \frac{x^{n+1}}{(n+1)!}e^c. \qquad (2.13)$$

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \ldots + (-1)^{n-1}\frac{x^{2n-1}}{(2n-1)!} + (-1)^n \frac{x^{2n+1}}{(2n+1)!}\cos(c). \qquad (2.14)$$

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots + (-1)^n \frac{x^{2n}}{(2n)!} + (-1)^{n+1}\frac{x^{2n+2}}{(2n+2)!}\cos(c). \qquad (2.15)$$

$$\frac{1}{1-x} = 1 + x + x^2 + \ldots + x^n + \frac{x^{n+1}}{1-x} \quad , \qquad x \ne 1. \qquad (2.16)$$

$$(1+x)^\alpha = 1 + C_\alpha^1 x + C_\alpha^2 x^2 + \ldots + C_\alpha^n x^n + C_\alpha^{n+1} x^{n+1}(1+c)^{\alpha-n-1}. \qquad (2.17)$$

Trong tất cả các công thức, ngoại trừ (2.16), điểm c nằm giữa 0 và x.

**Ví dụ 2.2.4** Tính xấp xỉ cos(x) với $|x| \le \pi/4$, với sai số không lớn hơn $10^{-5}$. Vì điểm c nằm trong số dư của (2.15) là không xác định, và sai số mong muốn là

$$\left|R_{2n+1}(x)\right| \le \frac{x^{2n+2}}{(2n+2)!} \le 10^{-5}, \qquad \text{với } |x| \le \pi/4.$$

Để bất đẳng thức này đúng, chúng ta cần có $\frac{(\pi/4)^{2n+2}}{(2n+2)!} \le 10^{-5}$, nó thỏa mãn khi $n \ge 3$. Phép tính xấp xỉ mong muốn là

$$\cos(x) \approx 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!}.$$

Nhiều đa thức Taylor và số hạng dư không được tạo ra một cách trực tiếp từ (2.6) và (2.11). Thay vào đó, các công thức Taylor chuẩn trên được khéo léo thực hiện để đạt được xấp xỉ Taylor cho các hàm số khác. Ví dụ, xem xét việc xây dựng một phép tính xấp xỉ đa thức Taylor đối với $f(x) = e^{-x^2}$ quanh điểm x = 0. Hãy bắt đầu bằng việc thay thế x bằng t trong (2.13), ta có

$$e^t = 1 + t + \frac{t^2}{2!} + \ldots + \frac{t^n}{n!} + \frac{t^{n+1}}{(n+1)!} e^c.$$

Với c là một số nằm giữa 0 và t. Điều này đúng với mọi số thực t. Bây giờ thay t bằng $-x^2$, cho ra

$$e^{-x^2} = 1 - x^2 + \frac{x^4}{2!} - \frac{x^6}{3!} \ldots + \frac{(-1)^n x^{2n}}{n!} + \frac{(-1)^{n+1} x^{2n+2}}{(n+1)!} e^c. \qquad (2.18)$$

Với c là một số thỏa $-x^2 \le c \le 0$. Điều này cho phép tính xấp xỉ đa thức Taylor đối với $e^{-x^2}$ tới bậc 2n (và bậc 2n+1). Nếu chúng ta cố gắng xây dựng phép tính xấp xỉ Taylor trực tiếp, như dạng của phần 2.1, thì đạo hàm của $e^{-x^2}$ nhanh chóng trở nên không thể kiểm soát được. Như một ví dụ hơi phức tạp hơn một chút của việc xây dựng trực tiếp phép tính xấp xỉ đa thức Taylor, chúng ta xuất phát từ phép tính xấp xỉ đối với log(1-t).

Bắt đầu bằng việc lấy tích phân (2.16) từ 0 đến t

$$\int_0^t \frac{1}{1-x}dx = \int_0^t (1+x+x^2+...+x^n)dx + \int_0^t \frac{x^{n+1}}{1-x}dx.$$

$$-\log(1-t) = t + \frac{1}{2}t^2 + \frac{1}{3}t^3 + ... + \frac{1}{n+1}t^{n+1} + \int_0^t \frac{x^{n+1}}{1-x}dx.$$

$$\log(1-t) = -(t + \frac{1}{2}t^2 + \frac{1}{3}t^3 + ... + \frac{1}{n+1}t^{n+1}) - \int_0^t \frac{x^{n+1}}{1-x}dx. \qquad (2.19)$$

Điều này đúng khi $0 \le t \le 1$. Số hạng dư có thể được làm đơn giản bằng việc áp dụng định lý giá trị trung bình tích phân, để đạt được

$$\int_0^t \frac{x^{n+1}}{1-x}dx = \frac{1}{1-c}\int_0^t x^{n+1}dx = \left(\frac{1}{1-c}\right)\frac{t^{n+2}}{n+2},$$

với c nằm giữa 0 và t, và cũng có giá trị đúng khi $-1 \le t < 0$.

Tóm tắt trường hợp quan trọng này, chúng ta có

$$\log(1-t) = -(t + \frac{1}{2}t^2 + \frac{1}{3}t^3 + ... + \frac{1}{n+1}t^{n+1}) - \left(\frac{1}{1-c_t}\right)\frac{t^{n+2}}{n+2}, \qquad (2.20)$$

với $c_t$ nằm giữa 0 và t; và điều này đúng khi -1≤ t<1.

**Định lý 2.2.5** (Định lý giá trị trung bình tích phân)

*Cho w(x) là hàm khả tích không âm trên đoạn [a, b], và f(x) liên tục trên đoạn [a,b]. Thì có ít nhất một điểm c trong [a, b] sao cho*

$$\int_a^b f(x).w(x)dx = f(c)\int_a^b w(x)dx.$$

*Đặc biệt, nếu w(x)=1 thì* $\int_a^b f(x)dx = f(c)(b-a).$

**Ghi chú 2.2.6** Khi chúng ta nói về giới hạn sai số trong một lượng nào đó, thì hãy nói giới hạn sai số $f(x)-p_n(x)$, nghĩa là chúng ta tìm một số $M_1$ sao cho

$$\left|f(x)-p_n(x)\right|\le M_1.$$

Khi chúng ta nói đến giới hạn sai số trong đoạn $\alpha\le x\le\beta$, nghĩa là chúng ta muốn tìm một số $M_2$ sao cho

$$\max_{\alpha\le x\le\beta}\left|f(x)-p_n(x)\right|\le M_2.$$

Hầu hết các ví dụ thì không "đẹp" như ví dụ 2.2.3 , và nói chung chúng ta phải sử dụng các giá trị tuyệt đối giải các bài toán sai số.

### 2.2.1 Chuỗi số vô hạn

Bằng việc sắp xếp lại các số hạng trong (2.16), chúng ta có được tổng một cấp số nhân

$$1+x+x^2+\ldots+x^n=\frac{1-x^{n+1}}{1-x}\quad,\qquad x\ne 1. \tag{2.21}$$

Cho $n\to\infty$ trong (2.16) khi $|x|<1$, chúng ta có được một cấp số nhân lùi vô hạn

$$\frac{1}{1-x}=1+x+x^2+\ldots=\sum_{j=0}^{\infty}x^j\quad,\qquad |x|<1. \tag{2.22}$$

Đây cũng chính là một chuỗi số vô hạn.

Một cách tổng quát, chúng ta nhắc lại rằng một chuỗi vô hạn $\sum_{j=0}^{\infty}c_j$, thì hội tụ nếu tổng từng phần $S_n=\sum_{j=0}^{n}c_j$, $n\ge 0$, là một dãy hội tụ. Điều này có nghĩa là $S=\lim_{n\to\infty}S_n$, tồn tại và khi đó ta có thể viết

$$S=\sum_{j=0}^{\infty}c_j. \tag{2.23}$$

Đối với cấp số vô hạn trong (2.22) với $x \neq 1$, tổng từng phần được cho bởi (2.21) là

$$S_n = \frac{1-x^{n+1}}{1-x}.$$

Khi $|x|<1$, rõ ràng dãy $\{S_n\}$ có giá trị giới hạn $1/(1-x)$ và vì thế chúng ta có thể nói chuỗi số vô hạn hội tụ với giá trị này. Khi $|x|>1$, dãy $\{S_n\}$ rõ ràng không hội tụ với bất cứ giá trị giới hạn nào; chúng ta nói chuỗi số vô hạn phân kỳ trong trường hợp này.

Giả định $f(x)$ có đạo hàm bậc bất kỳ tại x = a. Khi đó,

Chuỗi số vô hạn $\sum_{j=0}^{\infty} \frac{(x-a)^j}{j!} f^{(j)}(a)$, được gọi là sự khai triển chuỗi Taylor của hàm số $f(x)$ quanh điểm x = a.

Và tổng từng phần $\sum_{j=0}^{n} \frac{(x-a)^j}{j!} f^{(j)}(a)$, là đa thức Taylor $p_n(x)$.

Dãy $\{p_n(x)\}$ có giới hạn $f(x)$ nếu sai số dần về 0 khi $n \to \infty$. Tức là,

$$\lim_{n\to\infty}[f(x)-p_n(x)]=0.$$

Trong trường hợp này, chúng ta có thể viết

$$f(x)=\sum_{j=0}^{\infty} \frac{(x-a)^j}{j!} f^{(j)}(a). \qquad (2.24)$$

Như các ví dụ, chúng ta có thể chỉ ra rằng các số hạng sai số trong (2.13) đến (2.17) và (2.20) dần về 0 khi $n \to \infty$ khi giá trị x phù hợp. Điều này dẫn đến sự khai triển Taylor sau

$$e^x = \sum_{j=0}^{\infty} \frac{x^j}{j!}, \qquad -\infty < x < \infty. \qquad (2.25)$$

$$\sin x = \sum_{j=0}^{\infty} \frac{(-1)^j x^{2j+1}}{(2j+1)!}, \qquad -\infty < x < \infty. \tag{2.26}$$

$$\cos x = \sum_{j=0}^{\infty} \frac{(-1)^j x^{2j}}{(2j)!}, \qquad -\infty < x < \infty. \tag{2.27}$$

$$(1+x)^{\alpha} = \sum_{j=0}^{\infty} C_{\alpha}^{j} x^{j}, \qquad -1 < x < 1. \tag{2.28}$$

$$\log(1-t) = -\sum_{j=0}^{\infty} \frac{t^j}{j}, \qquad -1 \le x < 1. \tag{2.29}$$

Chuỗi số vô hạn có dạng $\sum_{j=0}^{\infty} a_j (x-a)^j$ (2.30)

được gọi là chuỗi số lũy thừa. Công thức Taylor là một cách để đạt được những chuỗi số như thế, nhưng chúng cũng có thể được tạo ra bằng cách khác. Sự hội tụ của chúng có thể được xem xét một cách trực tiếp mà không cần nhờ vào công thức sai số của Taylor. Chúng tôi nhắc lại 2 định lý quan trọng được dùng để xem xét sự hội tụ.

**Định lý 2.2.7** *Giả định rằng chuỗi số (2.30) hội tụ với giá trị $x_0$ nào đó. Thì chuỗi số (2.30) cũng hội tụ với tất cả các giá trị x thỏa $|x-a| < |x-x_0|$.*

**Định lý 2.2.8** *Cho chuỗi số (2.30), giả định rằng giới hạn $R = \lim_{n\to\infty} \left| \frac{a_{n+1}}{a_n} \right|$, tồn tại. Thì khi x thỏa $|x-a| < \frac{1}{R}$, chuỗi số (2.30) hội tụ về một giới hạn S(x). Khi R = 0, chuỗi số (2.30) hội tụ với bất kỳ số thực x nào.*

Ví dụ, chúng ta hãy xem xét sự hội tụ của chuỗi số lũy thừa trong công thức (2.27). Cho $t = x^2$, chúng ta có được chuỗi số

$$\sum_{j=0}^{\infty} \frac{(-1)^j t^j}{(2j)!}. \tag{2.31}$$

Áp dụng định lý 2.2.8 với $a_j = \frac{(-1)^j}{(2j)!}$, chúng ta thấy R = 0.

Vì thế chuỗi số (2.31) hội tụ với bất kỳ giá trị nào của t, và khi đó chuỗi số trong công thức (2.27) hội tụ với bất cứ giá trị nào của x.

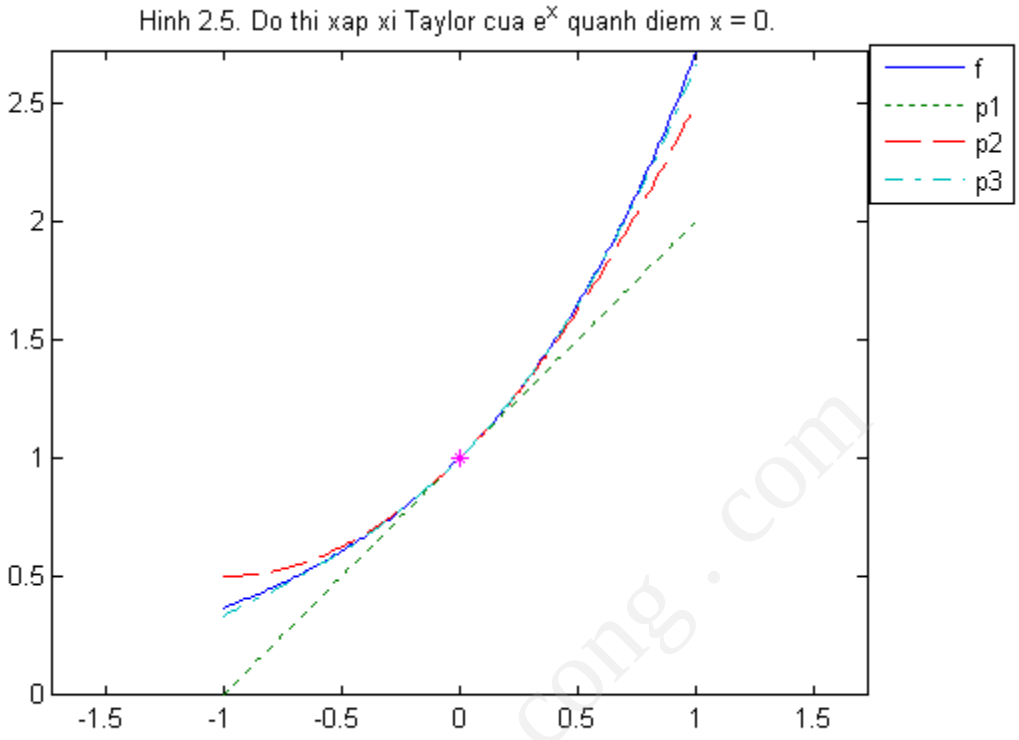
**CHƯƠNG TRÌNH MATLAB:** Chương trình MATLAB sau, vẽ lần lượt đồ thị các đa thức Taylor bậc 1, 2, 3 và các sai số bậc 1, 2, 3, 4 của e^x quanh điểm x = 0 trên đoạn [-b, b], với b là giá trị được đưa vào khi chạy chương trình. Trường hợp b = 1 sẽ vẽ ra các Hình 2.1, Hình 2.2, Hình 2.4, Hình 2.5; và được lưu với file plot_exp.m.

**Chương trình 2.3:**

```
% TIEU DE: Do thi cac da thuc Taylor cua exp(x) quanh diem x = 0.
% Day la do thi cua vai da thuc Taylor va sai so cua chung voi bac tang dan % cua ham exp(x) tren doan [-b,b].
% Truoc het, neu nhap b = 1 ta duoc cac hinh da trinh bay trong ly thuyet.
disp('Hay nhap b = 1 de duoc cac Hinh trong ly thuyet; Nhap xong nhan Enter de tiep tuc.')
b = input('Nhap gia tri cua b de tao [-b,b], b = ');
h = b/100;
x = -b:h:b;
max_deg = 4;
% Tao ra cac he so cua ham exp(x) khi mo rong quanh diem a = 0.
% Nhung he so nay duoc luu tru trong mang c, voi do dai la max_deg+1.
c = ones(max_deg+1,1);
fact = 1;
for i = 1:max_deg
  fact = i*fact;
  c(i+1) = 1/fact;
end
% Tinh gia tri khai trien da thuc Taylor bac1, bac2, bac3, bac4 bang cach goi % ham polyeval la mot file trong cung thu muc.
p1 = polyeval(x,0,c,1); p2 = polyeval(x,0,c,2);
p3 = polyeval(x,0,c,3); p4 = polyeval(x,0,c,4);
% Tinh gia tri cua exp(x) va gia tri cac sai so cua cac da thuc Taylor.
```

```
true = exp(x); err1 = true-p1; err2 = true-p2; err3 = true-p3; err4 = true-p4;
% Ve lan luot do thi cac da thuc Taylor.
% Bac 1(Hinh 2.1)
plot(x,true,x,p1,':',0,1,'m*');
title('Hinh 2.1. Do thi xap xi Taylor bac nhat cua e^x quanh diem x = 0.')
axis('equal')
legend('f','p1',-1)
disp('Ta da duoc Hinh 2.1. Vui long nhan Enter de ve tiep Hinh 2.2.')
pause
% Bac 1, 2 (Hinh 2.2)
hold off
plot(x,true,x,p1,':',x,p2,'--',0,1,'m*')
title('Hinh 2.2. Do thi xap xi Taylor bac nhat va bac 2 cua e^x quanh diem x = 0.')
axis('equal')
legend('f','p1','p2',-1)
disp('Vui long nhan Enter de ve tiep Hinh 2.5.')
pause
% Bac 1, 2, 3 (Hinh 2.5)
hold off
plot(x,true,x,p1,':',x,p2,'--',x,p3,'-.',0,1,'m*');
titel('Hinh 2.5. Do thi xap xi Taylor cua e^x quanh diem x = 0.')
axis('equal')
legend('f','p1','p2','p3',-1)
disp('Vui long nhan Enter de ve tiep Hinh 2.4.')
pause
% Ve do thi cac sai so (Hinh 2.4)
plot(x,err1,x,err2,':',x,err3,'--',x,err4,'-.');
title('Hinh 2.4. Do thi sai so trong xap xi Taylor cua e^x quanh diem x = 0.')
legend('bac 1','bac 2','bac 3','bac 4',0)
disp('Nhan Enter de ket thuc.')
hold off
pause
disp('Hay chay chuong trinh voi nhung gia tri b khac nhau de thay ro van de
hon nhe. Chuc cac ban thanh cong. Bye!')
```

Kết quả chạy file plot_exp.m với b = 1, ta được các hình vẽ Hình 2.1, Hình 2.2, Hình 2.4 và Hình 2.5.




**Chương trình 2.4:** Chương trình MATLAB sau, vẽ đồ thị các đa thức Taylor bậc 1, 2, 3 và các sai số bậc 1, 2, 3, 4 của log(x) quanh điểm x = 1 trên [a,b], 0<a<1 và b>1. Và được lưu với file plot_log.m.

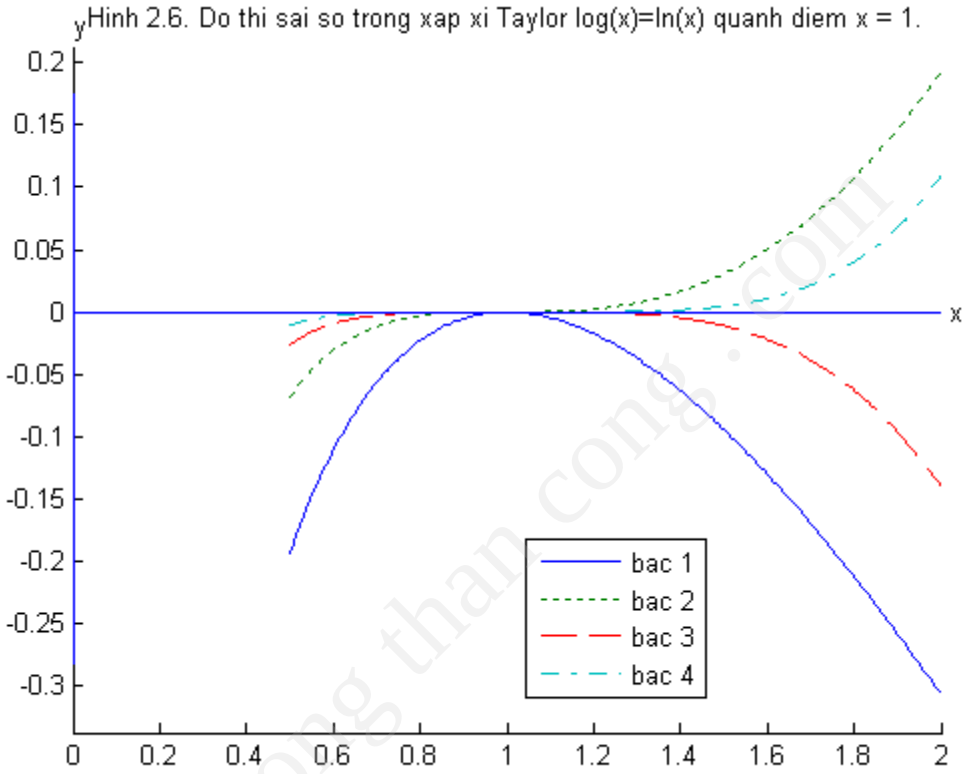
% TIEU DE: Ve do thi cua nhung da thuc Taylor cua log(x) quanh x = 1.
% Day la nhung do thi cua nhung da thuc Taylor rieng biet va nhung sai so % cua no doi voi bac tang dan. Do la ham dac biet xap xi voi log(x) va duoc %khai trien quanh diem x = 1. Khoang xap xi la [a, b] voi 0 < a < 1 va b > 1. % Truoc het, chung ta ve do thi cua nhung da thuc Taylor va tam dung va % roi chung ta ve do thi cua nhung sai so cua chung khi nhan phim Enter. % Neu chon a = 1/2, b = 2 thi ta duoc Hinh 2.3, Hinh 2.6.
% Khoi chay.

```
disp('Hay nhap a=1/2 va b=2 de duoc Hinh 2.3 va Hinh 2.6 trong ly thuyet.')
a = input('Khoi tao khoang [a,b], Nhap gia tri a voi 0 < a < 1, a =  ');
b = input('Nhap gia tri b > 1, b =  ');
h = (b-a)/200; x = a:h:b; max_deg = 4;
c = zeros(max_deg+1,1);
sign = 1;
for i = 1:max_deg
  c(i+1) = sign/i; sign = -sign;end
```

```
% Tinh gia tri khai trien da thuc Taylor bac1, bac2, bac3, bac4 bang cach goi % ham polyeval la mot file trong cung thu muc.
p1 = polyeval(x,1,c,1); p2 = polyeval(x,1,c,2);
p3 = polyeval(x,1,c,3); p4 = polyeval(x,1,c,4);
% Tinh gia tri dung cua ln(x) va cac sai so trong khai trien da thuc Taylor %tuong ung tung bac o tren.
true = log(x); err1 = true-p1; err2 = true-p2; err3 = true-p3; err4 = true-p4;
% Khoi tao de ve cac do thi cua cac da thuc Taylor.
% Xoa  bo he truc toa do da co, xay dung  he truc toa do moi va giua lai de ve % do thi tren he truc toa do moi nay.
hold off
clf
m = 1.1*min([min(p1),min(p2),min(p3),min(p4)]);
M = 1.1*max([max(p1),max(p2),max(p3),max(p4)]);
axis([a,b,m,M])
hold on
% Ve do thi cua nhung da thuc Taylor.
plot(x,true,x,p1,':',x,p2,'--',x,p3,'-.');%,'linewidth',1.5,'markersize',3);
plot([0 b],[0 0])
plot([0 0],[m/1.1 M/1.1])
plot([1 1],[m/1.1 0],':')
hold on
plot(1,0,'*')
title('Hinh 2.3. Do thi xap xi Taylor cua log(x)=ln(x) quanh diem x = 1.')
text(b+3*h,0,'x')
text(0,M,'y')
axis('equal')
legend('log(x)','p1','p2','p3',-1)     % Dua ghi chu len goc phai ngoai hinh ve.
disp('Nhan Enter de tiep tuc.')
pause
% Khoi tao de ve do thi cac sai so. Xoa  bo hinh da co, xay dung  he truc toa %do moi va giua lai de ve do thi  %cac sai so tren he truc toa do moi nay.
hold off
clf
m = 1.1*min([min(err1),min(err2),min(err3),min(err4)]);
M = 1.1*max([max(err1),max(err2),max(err3),max(err4)]);
axis([0,b,m,M])
hold on
% Ve do thi cac sai so
plot(x,err1,x,err2,':',x,err3,'--',x,err4,'-.');
plot([0 b],[0 0])
plot([0 0],[m/1.2 M/1.2])
```

```
title('Hinh 2.6. Do thi sai so trong xap xi Taylor cua log(x) quanh diem x= 1.')
text(b+3*h,0,'x')
text(0,M+.05*max(M,abs(m)),'y')
legend('bac 1','bac 2','bac 3','bac 4',0)          % Dua ghi chu vao hinh ve.
```

Kết quả chạy file plot_log.m ta được Hình 2.3, Hình 2.6 với a = 1/2, b = 2.




## 2.3. TÍNH GIÁ TRỊ SỐ CỦA ĐA THỨC

Tính giá trị số của một đa thức có vẻ như một công việc dễ làm và để minh họa cụ thể, chúng ta xem xét sự ước lượng của

$$p(x) = 3 - 4x - 5x^2 - 6x^3 + 7x^4 - 8x^5 .$$

Đối với một người lập trình, phương pháp đơn giản nhất để tính toán giá trị số là ước tính từng số hạng độc lập của các số hạng còn lại. Chính xác hơn, số hạng $cx^k$ được tính trong một chương trình bởi c*x^k hoặc c*x**k.

Phụ thuộc vào ngôn ngữ máy tính đang được dùng. Điều này đòi hỏi phép nhân k với hầu hết các trình biên dịch, mặc dù các trình biên dịch thông minh hơn có thể tạo ra một mã hữu hiệu hơn. Với phép tính gần đúng này, sẽ có 1+2+3+4+5=15 phép nhân trong tính toán giá trị của $p(x)$.

Phương pháp thứ 2 của tính giá trị số, hữu hiệu hơn. Chúng ta ước tính mỗi lũy thừa của x bằng cách nhân x với lũy thừa trước của x,

$$x^3 = x(x^2),\ x^4 = x(x^3),\ x^5 = x(x^4). \qquad (2.32)$$

Vì thế, mỗi số hạng $cx^k$ thực hiện 2 phép nhân khi k>1. Kết quả tính giá trị của p(x) dùng 1+2+2+2+2=9 phép nhân; tiết kiệm đáng kể hơn phương pháp thứ nhất, đặc biệt với các đa thức bậc cao hơn.

Phương pháp thứ 3 là phép nhân xếp lồng vào nhau (phương pháp tổ chim). Với phương pháp này, chúng ta có thể viết và tính giá trị số của p(x) theo dạng $p(x) = 3 + x(-4 + x(5 + x(-6 + x(7 - 8x)))).$

Số của các phép nhân chỉ là 5, tiết kiệm hơn phương pháp 2. Phương pháp phép nhân xếp lồng vào nhau là tiến trình tính giá trị số được ưa thích hơn, và sự tiện lợi của nó tăng lên khi bậc của đa thức lớn hơn.

Hãy xem xét đa thức tổng quát bậc n sau

$$p(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n, \quad a_n \neq 0. \qquad (2.33)$$

Nếu chúng ta sử dụng phương pháp thứ 2, với lũy thừa của x được tính như trong (2.32), thì số phép nhân trong tính giá trị số của $p(x)$ bằng 2n-1. Đối với phương pháp phép nhân xếp lồng vào nhau, viết và ước lượng $p(x)$ theo dạng

$$p(x) = a_0 + x(a_1 + x(a_2 + ... + x(a_{n-1} + a_n x)...). \qquad (2.34)$$

Điều này chỉ sử dụng n phép nhân, tiết kiệm khoảng 50% so với phương pháp 2. Do đó, chúng tôi sử dụng phương pháp phép nhân xếp lồng vào nhau để thực thi trong chương trình MATLAB được cho ở cuối phần 2.1.

**Ví dụ 2.3.1** Tính số đa thức Taylor $p_5(x)$, đối với log(x) gần điểm a=1. Công thức tổng quát được cho trong (2.19), với t được thay bằng $-(x-1)$. Từ đó,

$$p_5(x) = (x-1) - \frac{1}{2}(x-1)^2 + \frac{1}{3}(x-1)^3 - \frac{1}{4}(x-1)^4 + \frac{1}{5}(x-1)^5.$$

Cho $w = x-1$, và viết $p_5(x) = w(1 + w(-\frac{1}{2} + w(\frac{1}{3} + w(-\frac{1}{4} + \frac{1}{5}w))))$.

Trong chương trình máy tính, bạn nên lưu trữ các hệ số dưới dạng số thập phân, cho phù hợp với dạng số trong máy.

Chúng tôi cung cấp thêm một số thuật toán chính thức cho (2.34) bởi vì nó liên quan tới các chủ đề khác. Giả sử chúng tôi muốn ước tính $p(x)$ tại một số z nào đó. Xác định dãy hệ số $b_i$ như sau

$$\begin{aligned} b_n &= a_n, \\ b_{n-1} &= a_{n-1} + zb_n, \\ b_{n-2} &= a_{n-2} + zb_{n-1}, \\ &\cdots \\ b_0 &= a_0 + zb_1. \end{aligned} \tag{2.35}$$

Thì $$p(z) = b_0. \tag{2.36}$$

Các hệ số $b_j$ là các phép tính kéo dài trong một cặp thích hợp trong dấu ngoặc ở (2.34). Trong đó $b_{n-1}$ là phép tính tận cùng, và $b_0$ là phép tính cuối cùng. Khi nhìn theo cách này, phương pháp phép nhân xếp lồng vào nhau được gọi là phương pháp Horner; nó quan hệ gần với phép chia tổng hợp, phép chia này được trình bày trong các bài viết về đại số sơ cấp.

Dùng các hệ số trong (2.35), xác định đa thức

$$p(x) = b_1 + b_2 x + b_3 x^2 + \ldots + b_n x^{n-1}. \qquad (2.37)$$

thì $$p(x) = b_0 + (x - z)q(x). \qquad (2.38)$$

Trong đó, $q(x)$ là một thương số từ phép chia $p(x)$ bởi $x - z$ và $b_0$ là số dư. Kết quả này được dùng để liên kết với các phương pháp khai căn đa thức để giảm bậc của một đa thức khi căn z được tìm thấy, từ đó $b_0 = 0$ và $p(x) = (x - z).q(x)$.

### 2.3.1 Một chương trình mẫu

Chúng tôi kết thúc phần này bằng việc trình một chương trình MATLAB để tính xấp xỉ đa thức Taylor đối với hàm số

$$Sint\, x = \frac{1}{x}\int_0^x \frac{\sin(t)}{t}dt, \qquad x \neq 0. \qquad (2.39)$$

Với Sint(0)=1. Hàm số này được gọi là "sine tích phân". Chúng ta bắt đầu bằng cách lấy xấp xỉ đa thức Taylor trong đoạn [-1, 1] đối với Sint*x*, đòi hỏi rằng sai số lớn nhất trong đoạn [-1, 1] được giới hạn bởi $5.10^{-9}$. Bắt đầu bằng việc sử dụng (2.14) đối với sin(t), với x được thay thế bằng t. Đầu tiên xem xét trường hợp x>0. Chia cho t và tích hợp trên [0, x], chúng ta có

$$\begin{aligned} Sint\, x &= \frac{1}{x}\int_0^x [1 - \frac{t^2}{3!} + \frac{t^4}{5!} - \ldots + (-1)^{n-1}\frac{t^{2n-2}}{(2n-1)!}]dt + R_{2n-2}(x) \\ &= 1 - \frac{x^2}{3!3} + \frac{x^4}{5!5} - \ldots + (-1)^{n-1}\frac{x^{2n-2}}{(2n-1)!(2n-1)} + R_{2n-2}(x), \qquad (2.40) \\ R_{2n-2}(x) &= \frac{1}{x}\int_0^x (-1)^n \frac{t^{2n}}{(2n+1)!}\cos(c_t)dt. \end{aligned}$$

Điểm $c_t$ nằm giữa 0 và t. Vì $|\cos(c_t)| \leq 1$, ta có

$$\left|R_{2n-2}(x)\right| \le \frac{1}{x}\int_0^x (-1)^n \frac{t^{2n}}{(2n+1)!}dt = \frac{x^{2n}}{(2n+1)!(2n+1)}. \qquad (2.41)$$

Thật dễ dàng thấy rằng giới hạn này cũng đúng khi $x<0$. Ta cần, chọn bậc n để

$$\left|R_{2n-2}(x)\right| \le 5.10^{-9}. \qquad (2.42)$$

Từ (2.41), ta có $\max_{|x|\le 1}\left|R_{2n-2}(x)\right| \le \frac{1}{(2n+1)!(2n+1)}.$

Chúng ta chọn n để chặn trên này nhỏ hơn hoặc bằng $5.10^{-9}$. Điều này đúng nếu $2n+1 \ge 11$. Ví dụ, $n \ge 5$ thì (2.42) được thỏa. Đa thức chúng ta dùng là

$$p(x) = 1 - \frac{x^2}{3!3} + \frac{x^4}{5!5} - \frac{x^6}{7!7} + \frac{x^8}{9!9}, \qquad -1 \le x \le 1. \qquad (2.43)$$

Và khi xấp xỉ đối với $Sin\,t\,x$ trong [-1, 1], thì sai số của nó thỏa (2.42).

Đa thức $p(x)$ trên là một hàm số chẵn, nghĩa là $p(-x) = p(x)$, $\forall x$, ta gọi là đa thức chẵn; và các đa thức chẵn chỉ chứa lũy thừa chẵn của x. Vì thế chúng ta có thể ước lượng $p(x)$ hữu hiệu hơn bằng cách ước lượng đa thức bậc 4, $g(u)$ bằng cách thay $u = x^2$ trong $p(x)$

$$g(u) = 1 - \frac{u}{18} + \frac{u^2}{600} - \frac{u^3}{35280} + \frac{u^4}{3265920}. \qquad (2.44)$$

**CHƯƠNG TRÌNH MATLAB:** Các đa thức Taylor đối với Sin*t*x.

Các đa thức Taylor $p_n(x)$ gần 0 của Sin*t*x là các đa thức chẵn. Tiến hành tương tự với việc rút gọn (2.43) đến (2.44), chúng ta có thể rút gọn bằng xấp xỉ 1/2 số phép nhân cần thiết để ước lượng nó. Ở đây chúng tôi cho một chương trình MATLAB plot_sint để ước lượng các xấp xỉ Taylor của Sin*t*x. Các đa thức Taylor có 4 bậc n khác nhau (ở đây cho bậc 2, 4, 6 và 8) và tính ước lượng trong đoạn sử

dụng cụ thể [0,b](vì tính đối xứng của hàm chẵn). Với b là giá trị nguyên được nhập vào khi chạy chương trình.

**Chương trình 2.5:**

```
% Tao file plot_sint.m
function ans = plot_sint_total
% TIEU DE:Khai trien da thuc Taylor cho ham "Sine tich phan" quanh x = 0
% Day la cac do thi cua vai khai trien da thuc Taylor va sai so cho cac bac
% tang dan. Da thuc Taylor trong truong hop nay chi gom cac khai trien co
% bac chan, duoc goi la cac "even polynomials: Cac da thuc chan", nen do thi %
duoc ve doi xung khi x gan 0 va duoc trinh bay khi xap xi x=0. Vi vay,ta
% chi ve cac ham cu the xap xi cua Sint(x) tren doan [0,b], voi x = 0 cung cac
% diem mo rong cho viec tao cac da thuc Taylor. Chung ta ve do thi chi vai
% bac cua khai trien Taylor, co the duoc thay doi boi nguoi dung.
% De thay ro van de: Chay chuong trinh voi nhieu gia tri cua "b". Cung co %the
thay doi so bac cuc dai trong bien "max_degree", nham thu nghiem ket %qua voi
nhieu bac khai trien xap xi Taylor khac nhau.
% Bat dau chuong trinh.
b = input('Nhap 1 gia tri nguyen cho b de tao doan [0,b], nhap xong nhan enter;b=
');
h = b/200; x = 0:h:b; max_degree = 20;
% Xay dung cac he so Taylor cho "sine tich phan" cua ham "Sint.".
c = sint_tay(max_degree);
% Xac dinh 4 bac gia tri can khao sat.  Tat ca phai la bac chan, va phai nho % hon
hoac bang bien max_degree. Duoi day la cac bac 2, 4, 6 va 8.
degree=[2,4,6,8];
if max(degree) > max_degree
   fprintf('Vai gia tri bac khai trien lon hon bien max_degree = %2.0f\n',...
      max_degree)
   return
end
% Khoi tao mot day chua dung cac gia tri da thuc.  Hang #i thi mang gia tri
% cua bac da thuc degree=degree(i).
p = zeros(4,length(x));
% Tinh gia tri cac da thuc Taylor.
for i=1:4
   p(i,:) = poly_even(x,c,degree(i)); end
% Thiet lap cho viec ve do thi Taylor.
hold off
clf
axis([0,b,0,1])
```

```
hold on
% Ve do thi cac da thuc Taylor.
plot(x,p(1,:),x,p(2,:),':',x,p(3,:),'--',x,p(4,:),'-.')
plot([0,b],[0,0])
plot([0 0],[0 1])
title('Hinh 2.7. Cac xap xi Taylor cua Sint(x).')
text(1.025*b,0,'x')
text(0,1.03,'y')
legend(strcat('degree = ',int2str(degree(1))),...
      strcat('degree = ',int2str(degree(2))),...
      strcat('degree = ',int2str(degree(3))),...
      strcat('degree = ',int2str(degree(4))),-1)
```

% De chay chuong trinh nay ta can lap mot chuong trinh tao file sint_tay.m % de tinh cac he so cua (2.40) cua Sintx. Tao file sint_tay.m

```
function coeff=sint_tay(n)
```

% Ham coeff = sint_tay(n). Tinh toan cac he so cua khai trien xap xi Taylor %den bac n cho ham "sine tich phan ". Bien n dua vao phai la 1 so nguyen %chan. Dau ra la vector cac he so se co do dai m+1 trong do m=n/2. %Nguyen nhan la vi chi co cac he so bac chan khac 0, va dieu nay phai duoc %xem xet, danh gia trong da thuc Taylor.

```
m = double(int32(n/2));
if n ~= 2*m
  disp('Sai so trong poly_even(x,coeff,n):')
  disp('Tham so n phai la mot so nguyen chan.')
end
%
coeff = ones(m+1,1);
sign = 1;fact = 1;
for i=2:m+1
  sign = -sign;
  d = 2*i-1;
  fact = fact*(d-1)*d;
  coeff(i) = sign/(fact*d);end
```

% Chuong trinh plot_sint con su dung file poly_even.m de uoc luong bac % cua da thuc don gian hon trong (2.44)

% Tao file poly_even.m

```
function value = poly_even(x,coeff,n);
```

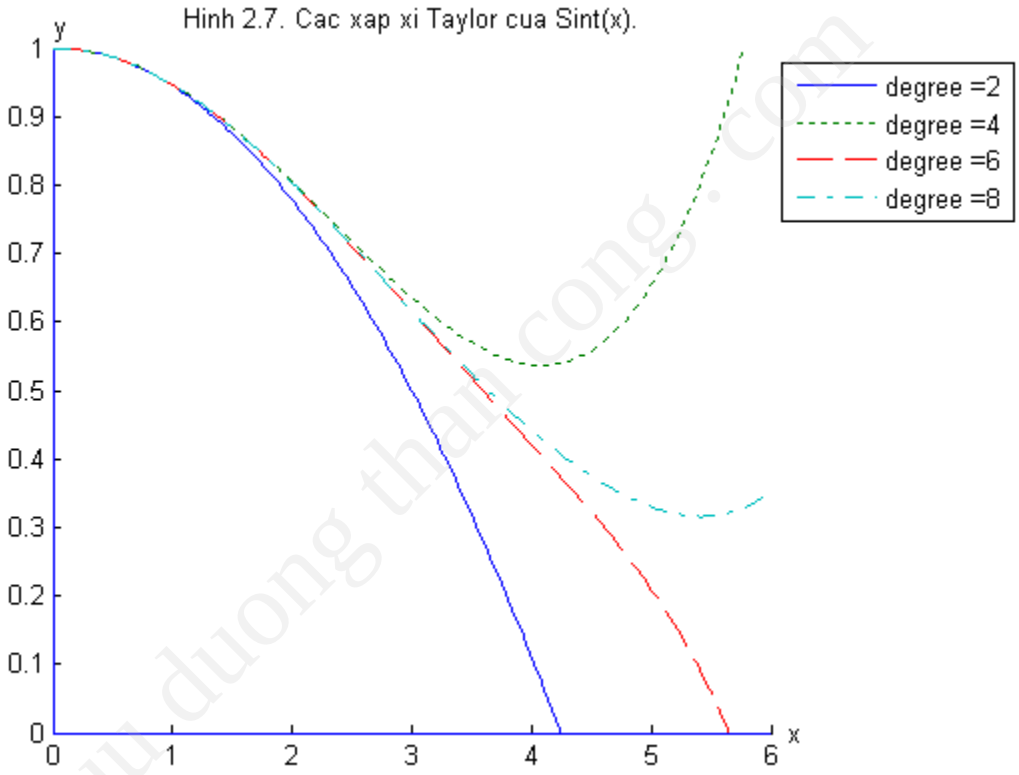
% Ham value = poly_even(x,coeff,n)

% Tinh toan gia tri so mot khai trien Taylor chan tai cac diem cho trong x, %voi n la bac cua da thuc. Cac he so duoc chua trong coeff. Cac he so nay %phai la so nguyen chan, va tham so n cung phai la so nguyen chan.

```
m = double(int32(n/2));
if n ~= 2*m
```

```
  disp('Sai so trong poly_even(x,coeff,n):')
  disp('Tham so n cung phai la so nguyen chan.')
end
xsq = x.*x;
value = coeff(m+1)*ones(size(x));
for i = m:-1:1
  value = coeff(i) + xsq.*value;end
```

Kết quả chạy file plot_sint.m, với giá trị **b = 6**.




Trong các chương sau, chúng tôi chỉ trình bày sơ lược các kiến thức về giải tích số. Với mỗi nội dung cốt lõi của giải tích số, chúng tôi sẽ trình bày một chương trình MATLAB chuẩn để giải số nó. Cũng có một số kết quả của giải tích số chúng tôi dùng mà không nêu sự chứng minh cụ thể vì ý tưởng chính của chúng tôi là xây dựng những chương trình MATLAB để giải số.

# CHƯƠNG 3

# TÌM NGHIỆM

Bài toán phổ biến trong toán ứng dụng là tìm nghiệm thực của phương trình

$$f(x)=0. \tag{3.1}$$

Trong chương này, ta giả định rằng $f(x)$ là hàm thực biến x. Như đã biết

- Số thực $\alpha$ thỏa $f(\alpha)=0$ được gọi là nghiệm thực của phương trình (3.1).
- Hoành độ giao điểm $\alpha$ của đồ thị hàm số $y=f(x)$ với trục hoành (nếu có) là nghiệm của phương trình (3.1).
- Hoành độ giao điểm $\alpha$ của hai đồ thị của hai hàm số $y=f(x)$ và $y=g(x)$ (nếu có) là nghiệm của phương trình $f(x)=g(x)$.

Ngoài ra, trước khi giải số phương trình (3.1) ta cũng cần phải biết phương trình đã cho có nghiệm thực hay không? Một trong những tiêu chuẩn để nhận biết là các định lý sau

**Định lý 3.1.** *Giả sử hàm số* $f:[a,b]\to R$ *liên tục và* $f(a).f(b)<0$. *Khi đó sẽ tồn tại* $\alpha\in(a,b)$ *sao cho* $f(\alpha)=0$.

**Định lý 3.2.** *Giả sử hàm số* $f:[a,b]\to R$ *liên tục, đơn điệu và* $f(a).f(b)<0$. *Khi đó sẽ tồn tại duy nhất* $\alpha\in(a,b)$ *sao cho* $f(\alpha)=0$.

**Định lý 3.3.** *Giả sử hàm số* $f:[a,b]\to R$ *liên tục, có đạo hàm không đổi dấu trên khoảng* $(a,b)$ *và* $f(a).f(b)<0$. *Khi đó sẽ tồn tại duy nhất* $\alpha\in(a,b)$ *sao cho* $f(\alpha)=0$.

Chúng tôi sẽ trình bày một vài phương pháp giải số đơn giản để giải phương trình (3.1).

## 3.1. PHƯƠNG PHÁP CHIA ĐÔI (BISECTION)

### 3.1.1 Mô tả phương pháp

Giả sử $f(x)$ liên tục trong đoạn $a \le x \le b$ và $f(a).f(b) < 0$. Khi đó, theo định lý 3.1 thì phương trình (3.1) có ít nhất một nghiệm $\alpha$ trên khoảng $(a, b)$ và với một sai số có thể chấp nhận được $\varepsilon > 0$. Thì phương pháp chia đôi gồm các bước sau:

B1. Xác định $c = (a+b)/2$.

B2. Nếu $b - c \le \varepsilon$, thì chấp nhận c là nghiệm và dừng lại.

B3. Nếu $sign[f(a)].sign[f(b)] \le 0$, thì đặt $a = c$ (sign: Hàm dấu hoặc dấu).

Ngược lại, đặt $b = c$. Trở lại B1.

Vòng lặp sẽ dừng lại khi $|\alpha - c| \le \varepsilon$ được thỏa.

**Ví dụ 3.1.1** Tìm nghiệm lớn nhất của $f(x) \equiv x^6 - x - 1 = 0$, chính xác trong phạm vi $\varepsilon = 0.0001$.

- Với một đồ thị, ta dễ dàng kiểm tra $1 < \alpha < 2$. Chúng ta chọn $a = 1, b = 2$; thì $f(a) = -1$, $f(b) = 61$, trái dấu.

**Bảng 3.1.** Phương pháp chia đôi đối với ví dụ 3.1.1.

| $n$ | $a$ | $b$ | $c$ | $b-c$ | $f(c)$ |
|---|---|---|---|---|---|
| 1 | 1.0000 | 2.0000 | 1.5000 | 0.5000 | 8.8906 |
| 2 | 1.0000 | 1.5000 | 1.2500 | 0.2500 | 1.5647 |
| 3 | 1.0000 | 1.2500 | 1.1250 | 0.125 | -0.0977 |

| | | | | | |
|---|---|---|---|---|---|
| 4 | 1.1250 | 1.2500 | 1.1875 | 0.0625 | 0.6167 |
| 5 | 1.1250 | 1.1875 | 1.1562 | 0.0312 | 0.2333 |
| 6 | 1.1250 | 1.1562 | 1.1406 | 0.0156 | 0.0616 |
| 7 | 1.1250 | 1.1406 | 1.1328 | 0.0078 | -0.0196 |
| 8 | 1.1328 | 1.1406 | 1.1367 | 0.0039 | 0.0206 |
| 9 | 1.1328 | 1.1367 | 1.1348 | 0.0020 | 0.0004 |
| 10 | 1.1328 | 1.1348 | 1.1338 | 0.00098 | -0.00096 |

Các kết quả của thuật toán B1 đến B3 được thể hiện trong Bảng 3.1. Mục $n$ chỉ hàng liên đới tương ứng với số lần lặp đi lặp lại $n$ của các bước B1 đến B3.

### 3.1.2 Đánh giá sai số

Cho $a_n, b_n$ và $c_n$ biểu thị các giá trị được tính lần thứ $n$ của $a, b$ và $c$ tương ứng, $b-a$ biểu thị độ dài của khoảng gốc mà chúng ta đã bắt đầu với nó. Ta được

$$|\alpha - c_n| \le \frac{1}{2^n}(b-a). \quad (3.2)$$

Để biết được bao nhiêu lần lặp là cần thiết, giả sử rằng chúng ta muốn có

$$|\alpha - c_n| \le \varepsilon.$$

Điều này sẽ được thỏa nếu

$$\frac{1}{2^n}(b-a) \le \varepsilon.$$

Suy ra,

$$n \ge \frac{\log\left(\frac{b-a}{\varepsilon}\right)}{\log 2}. \quad (3.3)$$

Đối với ví dụ 3.1.1, ta cần

$$n \ge \frac{\log\left(\frac{1}{0.001}\right)}{\log 2} \approx 9.97.$$

Vậy, chúng ta cần có $n = 10$ lần lặp để có kết quả mong đợi.

**CHƯƠNG TRÌNH MATLAB:** Thực thi phương pháp chia đôi bởi hàm **bisect**. Hàm số $f$ được cho là một hàm số nội bộ, nghĩa là nó không thể nhận biết được đối với các chương trình MATLAB được lưu trữ trong các file khác biệt với chương trình này. Chương trình liên quan tới việc in bên trong của các bước trung gian trong phương pháp chia đôi, bước này có thể bỏ qua. Kết thúc chương trình, in giá trị của nghiệm, giới hạn sai số và số lần lặp để có được kết quả.

**Chương trình 3.1:**

```
function root=bisect(a0,b0,ep,max_iterate,index_f)
% Ham chia doi bisect(a0,b0,ep,max_iterate,index_f). Day la phuong phap lap
%chia doi nham tim nghiem cua phuong trinh f(x)=0.
% Ham f duoc dinh nghia ben duoi boi nguoi dung. Ham f phai thoa dieu kien lien
%tuc tren doan [a0,b0], va gia tri ham phai trai dau tai a0 va b0.  Gia tri "ep" la gia
%tri sai so cho phep. (1) doan gioi han ban dau la phai dung du kien va (2) "ep"
%khong duoc qua nho de dam bao rang so lan lap khong qua lon. Hau het cac dieu
%kien nay khong duoc kiem tra trong chuong trinh!  Tham so max_iterate la mot
%gioi han tren cho so lan lap can tinh toan (viec them tham so nay de tranh
chuong %trinh lap vo han do "ep" qua be). Cach thuc hien goi ham f(x), theo cau
truc cua vi %du sau day
%               root = bisect(1,1.5,1.0E-6,10,1)
% Tham so index_f xac dinh cu the mot ham nao duoc dinh nghia ben duoi duoc
% su dung de tinh nghiem (case 1, case 2 vv..).
% Va sau khi thuc hien xong vong lap se cho ra cac ket qua sau (6 cot):
%               count, a, b, c, f(c), (b-a)/2
% c vong lap hien tai va (b-a)/2 la gioi han sai so cua c.
% bien count cho biet so vong lap hien tai.
% --------------- MOT KET QUA TINH TOAN SAU N VONG LAP BAO GOM: ----------
--%
%  root=nghiem giai so.
%  error_bound = pham vi sai so.
%  it_count = so lan lap de co duoc nghiem tren.
%  root = gia tri nghiem lay xap xi.
% Hay chay thu voi root = bisect(1,1.5,1.0E-6,10,1)
if a0 >= b0
   disp('a0 < b0 gia tri chua thoa.  Stop!')
   return
end
%
```

```
format short e
a = a0; b = b0;
fa = f(a,index_f); fb = f(b,index_f);
if sign(fa)*sign(fb) > 0
   disp('f(a0) va f(b0) cung dau.  Stop!')
   return
end
%
c = (a+b)/2;
it_count = 0;
while b-c > ep & it_count < max_iterate
   it_count = it_count + 1;
   fc = f(c,index_f);
% Thuc thi phuong phap chia doi. Nhan Enter roi nhan lien tuc phim bat ky tren
%may de di den ket qua can tinh.
 iteration = [it_count a b c fc b-c]
 if sign(fb)*sign(fc) <= 0
       a = c;
       fa = fc;
   else
       b = c;
       fb = fc;
   end
   c = (a+b)/2;
   pause
end

format long
root = c
format short e
error_bound = b-c
format short
it_count
%% DINH NGHIA CAC HAM CAN TINH TOAN THEO CAU TRUC CUA MATLAB %%
function value = f(x,index)
% Dinh nghia mot ham f can tim nghiem boi nguoi dung. Co the mo rong them
case %5, case 6, case 7 vv... Tham so index_f khi goi ham co the la 1;2;3;4;5... tuy
viec %muon chon ham nao de tinh.
switch index
case 1
        value = x.^6 - x - 1;
        case 2
```

```
            value = x - exp(-x);
            case 3
                  value = sin(x) - exp(-x);
                  case 4
                        value = x.^4-x-1;
end
```

Dùng chương trình 3.1 giải ví dụ 3.1.1, hàm được gọi theo cú pháp

root = bisect(1,2,0.0001,10,1).

Kết quả sau 10 bước lặp theo sẽ là

- Nghiệm: root =1.13427734375000.
- Sai số: error_bound = 4.8828e-004.
- Số lần lặp thực tế: it_count =10.
- Nghiệm xấp xỉ: root =1.1343.

Chương trình trên dùng để giải ví dụ 3.1.1 và một lớp các phương trình sau với một sai số có thể chấp nhận được $\varepsilon$=0.0001. Ta có thể thêm vào hoặc thay đổi các 'case i' để giải các phương trình khác.

a. Nghiệm của $x = e^{-x}$ (Chạy chương trình với root=bisect(0,1,0.0001,15,2)).

b. Nghiệm dương nhỏ nhất của $e^{-x} = \sin(x)$. (root=bisect(0,1,0.0001,15,3))

c. Tất cả nghiệm thực của $x^4 - x - 1 = 0$. (root=bisect(1,2,0.0001,15,4) và root=bisect(-1,0,0.0001,15,4))

**Chú thích:** nếu thay $c = (a+b)/2$ bởi $c = \dfrac{af(b) - bf(a)}{f(b) - f(a)}$ thì ta thu được phương pháp cát tuyến (hay còn gọi là phương pháp dây cung) .

## 3.2. PHƯƠNG PHÁP NEWTON

### 3.2.1 Mô tả phương pháp

Trên Hình 3.1, ta có $x_0$ là ước đoán ban đầu của $\alpha$, nghiệm của (3.1); $x_1$ là nghiệm của $p_1(x)$ với

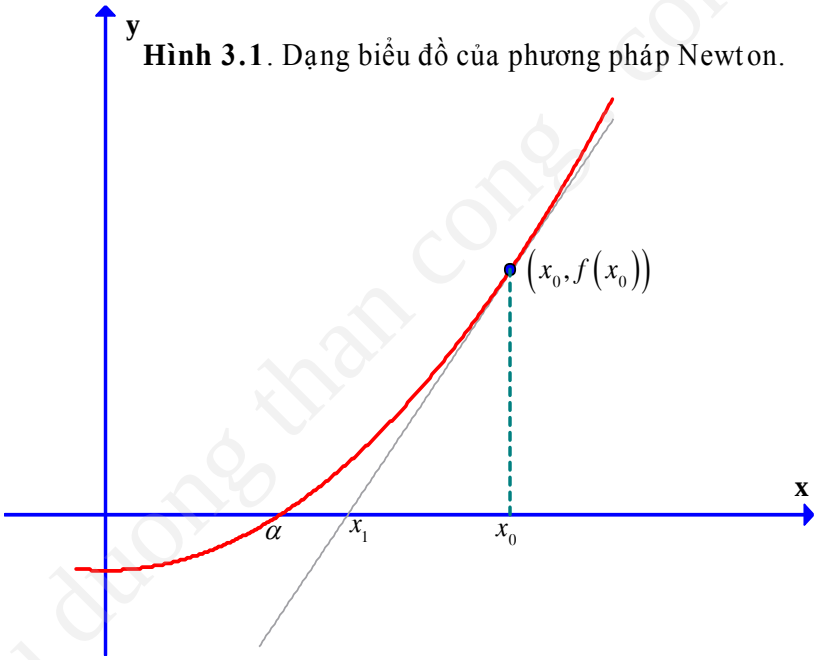
$$p_1(x) = f(x_0) + f'(x_0)(x - x_0).$$

Dẫn đến,

$$f(x_0) + f'(x_0)(x_1 - x_0) = 0.$$

Kết quả, $$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

**Hình 3.1**. Dạng biểu đồ của phương pháp Newton.




Ta được $x_1$ xấp xỉ với $\alpha$ tốt hơn là $x_0$. Lặp lại tiến trình với $x_1$ như ước đoán ban đầu. Dẫn đến

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}.$$

Tổng quát ta có: $$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \qquad n = 0,1,2,...$$

(3.4)

Đây là Phương pháp Newton dùng để giải (3.1). Nó cũng được gọi là phương pháp Newton-Raphson.

**Ví dụ 3.2.1** Dùng phương pháp Newton, giải phương trình trong ví dụ 3.1.1.

$$f(x) = x^6 - x - 1, \qquad f'(x) = 6x^5 - 1.$$

Và vòng lặp được cho bởi

$$x_{n+1} = x_n - \frac{x_n^{\,6} - x_n - 1}{6x_n^{\,5} - 1}, \qquad n \geq 0.$$

Chúng ta dùng ước đoán ban đầu của $x_0 = 1.5$. Những kết quả được thể hiện trong Bảng 3.2. Cột " $x_n - x_{n-1}$ " là một ước lượng của sai số $\alpha - x_{n-1}$. Nghiệm đúng là $\alpha$ = 1.134724138, và $x_6$ bằng $\alpha$ với 9 số có nghĩa, tức là $\varepsilon = 1.0\text{E-}9$. So sánh điều này với những kết quả trước đó được thể hiện trong bảng 3.1 đối với phương pháp chia đôi. Quan sát thấy phương pháp Newton có lẽ hội tụ chậm lúc đầu. Tuy nhiên, khi các vòng lặp đến gần nghiệm hơn, thì tốc độ hội tụ tăng lên, như được thể hiện trong bảng.

**Bảng 3.2.** Phương pháp Newton giải $x^6 - x - 1 = 0$.

| $n$ | $x_n$ | $f(x_n)$ | $x_n - x_{n-1}$ | $\alpha - x_{n-1}$ |
|---|---|---|---|---|
| 0 | 1.50000000 | 8.89R+1 | -2.00E-1 | -3.65E-1 |
| 1 | 1.30049088 | 2.54E+1 | -1.19E-1 | -1.66E-1 |
| 2 | 1.18148042 | 5.38E-1 | -4.20E-2 | -4.68E-2 |
| 3 | 1.13945559 | 4.92E-2 | -4.68E-3 | -4.73E-3 |
| 4 | 1.13477763 | 5.50E-4 | -5.35E-5 | -5.35E-5 |
| 5 | 1.13472415 | 7.11E-8 | -6.91E-9 | -6.91E-9 |
| 6 | 1.13472414 | 1.11E-15 | 0 | 0 |

### 3.2.2 Đánh giá sai số

Giả thiết $f(x)$ có đạo hàm đến cấp hai liên tục với mọi x trong khoảng lân cận của $\alpha$ và $f'(\alpha) \neq 0$. Dùng định lý Taylor để viết

$$f(\alpha) = f(x_n) + (\alpha - x_n) f'(x_n) + \frac{1}{2}(\alpha - x_n)^2 f''(c_n), \text{ với } c_n \text{ nằm giữa } \alpha \text{ và } x_n .$$

Ta có $f(\alpha) = 0$, chia hai vế cho $f'(x_n)$ để có được:

$$0 = \frac{f(x_n)}{f'(x_n)} + \alpha - x_n + (\alpha - x_n)^2 \frac{f''(c_n)}{2f'(x_n)} .$$

Kết hợp với (3.4), suy ra $0 = x_n - x_{n+1} + \alpha - x_n + (\alpha - x_n)^2 \frac{f''(c_n)}{2f'(x_n)}$.

Hay, $$\alpha - x_{n+1} = (\alpha - x_n)^2 \left[ \frac{-f''(c_n)}{2f'(x_n)} \right].$$

(3.5)

Công thức này nói lên rằng sai số trong $x_{n+1}$ gần với số hạng của tỷ lệ thức với bình phương của sai số trong $x_n$. Khi sai số ban đầu là đủ nhỏ, điều này cho thấy sai số trong những vòng lặp kế tiếp sẽ giảm rất nhanh chóng. Công thức (3.5) còn được dùng để chứng minh sự hội tụ của phương pháp Newton, nhưng chúng tôi bỏ qua nó.

Để ước lượng $\alpha - x_n$. Ta có, $f(\alpha) = 0$ và áp dụng định lý giá trị trung bình

$$f(x_n) = f(x_n) - f(\alpha) = f'(\xi_n)(x_n - \alpha), \text{ với } \xi_n \text{ nằm giữa } x_n \text{ và } \alpha .$$

Suy ra, $\alpha - x_n = \frac{-f(x_n)}{f'(\xi_n)} \approx \frac{-f(x_n)}{f'(x_n)}$, với điều kiện $f'(x_n) \doteq f'(\xi_n)$.

Kết hợp với (3.5) ta được $\alpha - x_n \approx x_{n+1} - x_n$.

(3.6)

Đây là công thức ước lượng sai số chuẩn cho phương pháp Newton và được dùng trong chương trình MATLAB bên dưới.

**CHƯƠNG TRÌNH MATLAB:** Thực thi phương pháp Newton bởi hàm **newton**, với ước lượng sai số và loại trừ vòng lặp vô hạn. Tham số nhập vào max_iterate là giới hạn trên về số vòng lặp được tính; điều này ngăn chặn sự cố về vòng lặp vô hạn. Kết thúc chương trình, in giá trị của nghiệm, giới hạn sai số và số lần lặp để có được kết quả.

**Chương trình 3.2:**

```
function root = newton(x0,error_bd,max_iterate,index_f)
% Ham newton(x0,error_bd,max_iterate,index_f). Day la phuong phap Newton % de giai phuong trinh f(x) = 0. Ham f(x) va deriv_f(x) duoc cho truoc. Tham so %error_bd duoc dung de kiem tra sai so, do chinh xac cho moi vong lap. Tham so % max_iterate la gioi han tren (can tren) cua so lan lap thuc hien tren may tinh. %Mot phong doan x0 ban dau duoc cho truoc.
%  Doi voi ham f(x) da cho, chuong trinh duoc goi theo cu phap vi du nhu:
%                root = newton(1,1.0E-12,10,1)
% Tham so index_f=1 chi ro ham duoc su dung la case 1.
% Chuong trinh in cac gia tri vong lap:  iterate_number, x, f(x), deriv_f(x), error
% Gia tri x la gia tri phong doan ban dau cua nghiem va roi la gia tri cua vong lap
% hien tai. Bien error la: error = gia tri vong lap moi tinh toan duoc tru di gia tri %vong lap ngay truoc do va do la uoc luong sai so cua vong lap; f(x), deriv_f(x) lan %luot la gia tri cua ham f va dao ham cua no . Den day se quay lai de thuc hien %tiep tuc vong lap cho den khi co duoc ket qua mong doi.

format short e      % Dau phay dong voi 5 chu so co nghia sau dau phay.
error = 1;
it_count = 0; % Vong lap ban dau.
```

```
while abs(error) > error_bd & it_count <= max_iterate      %Kiem tra dieu kien
dau.
   fx = f(x0,index_f);
   dfx = deriv_f(x0,index_f);
   if dfx == 0
      disp('Phep chia cho 0.  Stop')
      return
   end
   x1 = x0 - fx/dfx;
   error = x1 - x0;
% Thuc hien phuong phap Newton. Nhan Enter roi nhan lien tuc phim bat ky tren
% may de di den ket qua can tinh.
   iteration = [it_count x0 fx dfx error]
   pause
   x0 = x1;
   it_count = it_count + 1;% Thuc hien lien tiep cac vong lap.
end

if it_count > max_iterate
   disp('So vong lap thuc te lon hon so vong lap cho phep, khong tinh duoc
nghiem.')
   else
   format long                % Lay 15 chu so co nghia sau dau phay.
   root = x1
   format short e
   error
   format short
   it_count
end

% Ham so doi voi phuong trinh dinh nghia cua bai toan tim nghiem.
function value = f(x,index)
switch index
case 1
   value = x.^6 - x - 1;
   case 2
       value = x - exp(-x);
         case 3
             value = sin(x) - exp(-x);
                 case 4
                   value = x.^4-x-1;
end
```

```
% Dao ham cua ham so dinh nghia cua bai toan tim nghiem.
function value = deriv_f(x,index)
switch index
case 1
   value = 6*x.^5 - 1;
   case 2
    value = 1 + exp(-x);
     case 3
        value = cos(x) + exp(-x);
        case 4
           value = 4*x.^3-1;
end
```

Dùng chương trình 3.2 giải ví dụ 3.2.1, hàm được gọi theo cú pháp

root = newton(1.5,1.0E-9,10,1).

Kết quả sau 10 bước lặp theo sẽ là

- Nghiệm: root =1.13472413840152.
- Sai số: error = 0.
- Số lần lặp thực tế: it_count =7.
- Nghiệm xấp xỉ: root =1.1347.

Chương trình trên dùng để giải ví dụ 3.2.1 và một lớp các phương trình sau với một sai số có thể chấp nhận được $\varepsilon$=0.0001. Ta có thể thêm vào hoặc thay đổi các 'case i' để giải các phương trình khác.

a. Nghiệm của $x = e^{-x}$.(Chạy chương trình với root=newton(1,0.0001,15,2)).

b. Nghiệm dương nhỏ nhất của $e^{-x} = \sin(x)$.(root=newton(0.5,0.0001,15,3)).

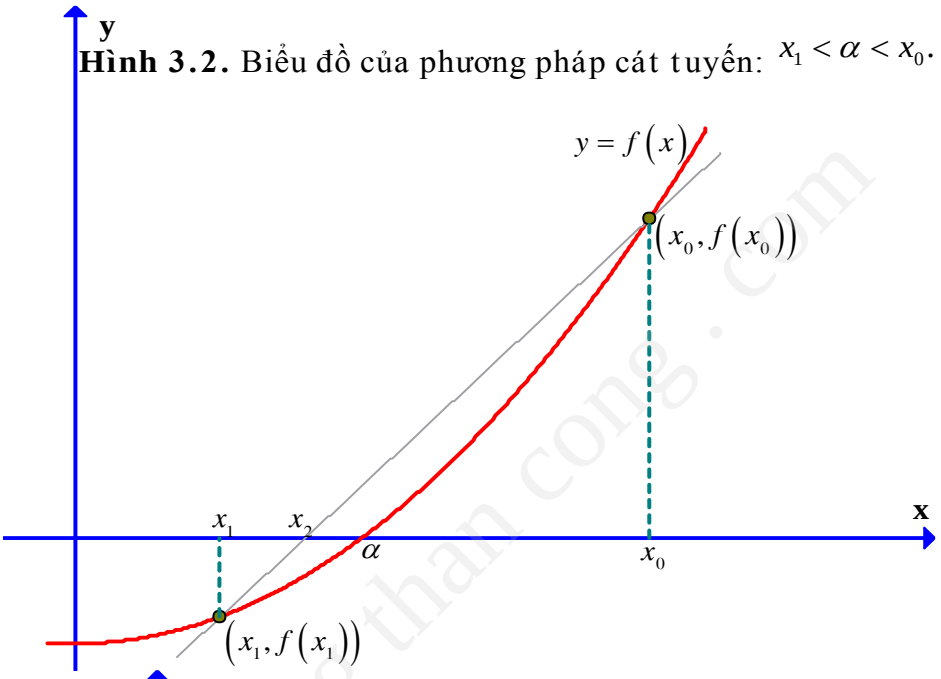
c. Tất cả nghiệm thực của $x^4 - x - 1 = 0$.(root=newton(1,0.0001,15,4) và root=newton(-0.5,0.0001,15,4)).

## 3.3. PHƯƠNG PHÁP CÁT TUYẾN (SECANT)

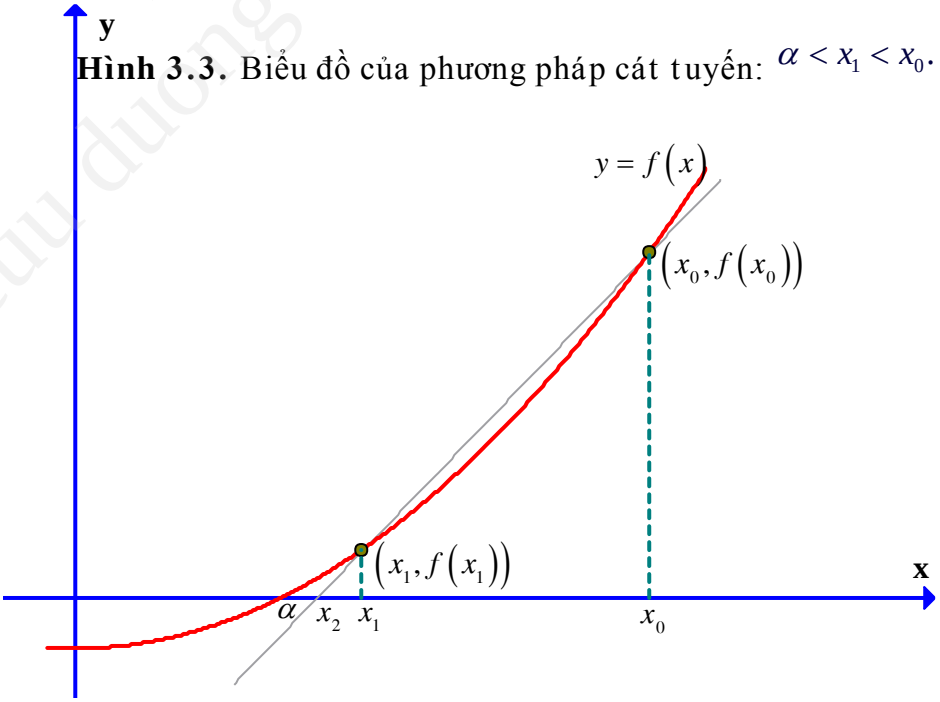
### 3.3.1 Mô tả phương pháp

Trên Hình 3.2, Hình 3.3, với $x_0$ và $x_1$ là hai ước đoán ban đầu của $\alpha$. Cát tuyến $y = p(x)$ qua hai điểm $(x_0, f(x_0))$ và $(x_1, f(x_1))$ là một xấp xỉ của đồ thị hàm số $y = f(x)$ và nghiệm $x_2$ của $y = p(x)$ là một xấp xỉ mới của $\alpha$.

**Hình 3.2.** Biểu đồ của phương pháp cát tuyến: $x_1 < \alpha < x_0$.




**Hình 3.3.** Biểu đồ của phương pháp cát tuyến: $\alpha < x_1 < x_0$.



Ta có, $$y = p(x) \equiv f(x_1) + (x - x_1).\frac{f(x_1) - f(x_0)}{x_1 - x_0}.$$

Giải $p(x_2) = 0$, chúng ta có được

$$x_2 = x_1 - f(x_1).\frac{x_1 - x_0}{f(x_1) - f(x_0)}.$$

Tìm $x_2$, chúng ta có thể bỏ $x_0$ và dùng $x_1$, $x_2$, như một tập hợp mới của các giá trị xấp xỉ cho $\alpha$. Điều này dẫn đến một giá trị được cải tiến $x_3$. Tiếp tục tiến trình này ta có được công thức vòng lặp tổng quát

$$x_{n+1} = x_n - f(x_n).\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}.$$

(3.7)

Đây là phương pháp cát tuyến (hay phương pháp dây cung).

**Ví dụ 3.3.1** Giải phương trình $f(x) \equiv x^6 - x - 1 = 0.$ Phương trình đã được sử dụng trước đây như một ví dụ cho cả phương pháp chia đôi và phương pháp Newton. Các kết quả được cho trong Bảng 3.3, bao gồm $x_n - x_{n-1}$ như một ước lượng của $\alpha - x_{n-1}$. Vòng lặp $x_8$ bằng $\alpha$ được làm tròn tới 9 số có nghĩa. Như với phương pháp Newton đối với phương trình này, các vòng lặp trước không hội tụ nhanh. Nhưng khi vòng lặp trở nên gần hơn với $\alpha$, tốc độ hội tụ tăng lên.

**Bảng 3.3.** Phương pháp cát tuyến giải $x^6 - x - 1 = 0$.

| $n$ | $x_n$ | $f(x_n)$ | $x_n - x_{n-1}$ | $\alpha - x_{n-1}$ |
|---|---|---|---|---|
| 0 | 2.0 | 61.0 | | |
| 1 | 1.0 | -1.0 | -1.0 | |
| 2 | 1.01612903 | -9.15E-1 | 1.61E-2 | 1.35E-1 |
| 3 | 1.19057777 | 6.57E-1 | 1.74E-1 | 1.19E-1 |
| 4 | 1.11765583 | -1.68E-1 | -7.29E-2 | -5.59E-2 |
| 5 | 1.13253155 | -2.24E-2 | 1.49E-2 | 1.71E-2 |

| | | | | |
|---|---|---|---|---|
| 6 | 1.13481681 | 9.54E-4 | 2.29E-3 | 2.19E-3 |
| 7 | 1.13472365 | -5.07E-6 | -9.32E-5 | -9.27E-5 |
| 8 | 1.13472414 | -1.13E-9 | 4.92E-7 | 4.92E-7 |

### 3.3.2 Đánh giá sai số

Chúng tôi xin trình bày ngắn gọn vấn đề này và bỏ qua việc biến đổi. Ta có

$$\alpha - x_{n+1} = \left(\alpha - x_n\right)\left(\alpha - x_{n-1}\right)\left[\frac{-f''\left(\xi_n\right)}{2f'\left(\zeta_n\right)}\right].$$

(3.8)

Ẩn số $\zeta_n$ nằm giữa $x_n$ và $x_{n-1}$, và ẩn số $\xi_n$ nằm giữa số lớn nhất và số nhỏ nhất của những số $\alpha$, $x_n$, và $x_{n-1}$. Công thức sai số gần giống với công thức sai số Newton (3.5). Công thức (3.8) có thể được dùng để có được kết quả sai số xa hơn nếu $x_0$ và $x_1$ được chọn đủ gần với $\alpha$, sau đó chúng ta có sự hội tụ và

$$\lim_{n\to\infty}\frac{\left|\alpha - x_{n+1}\right|}{\left|\alpha - x_n\right|^r} = \left|\frac{f''\left(\alpha\right)}{2f'\left(\alpha\right)}\right| \equiv c, \text{ tại } r = \left(\sqrt{5}+1\right)/2 \doteq 1.62$$

(3.9)

Như vậy, $$\left|\alpha - x_{n+1}\right| \approx c\left|\alpha - x_n\right|^{1.62}.$$

(3.10)

Kết quả (3.10) có thể được dùng để điều chỉnh ước lượng sai số.

$$\alpha - x_{n-1} \approx x_n - x_{n-1}, \text{ với } x_n \text{ đủ gần với nghiệm.}$$

(3.11)

Đây là công thức ước lượng sai số được dùng trong chương trình MATLAB.

**CHƯƠNG TRÌNH MATLAB:** Thực thi phương pháp cát tuyến bởi hàm **secant**, với ước lượng sai số và bảo đảm an toàn đối với vòng lặp vô hạn. Tham số nhập vào max_iterate là chặn trên số vòng lặp được tính, điều này ngăn cản việc xảy ra của một vòng lặp vô hạn. Kết thúc chương trình, in giá trị của nghiệm, giới hạn sai số và số lần lặp để có được kết quả.

**Chương trình 3.3:**

```
function root = secant(x0,x1,error_bd,max_iterate,index_f)
% Ham so secant(x0,x1,error_bd,max_iterate,index_f)
% Day la phuong phap cat tuyen de giai phuong trinh f(x) = 0.
% Tham so error_bd dung de kiem tra su sai so cho su chinh xac cua moi vong
lap. % Tham so max_iterate la mot gioi han tren cua so vong lap can tinh toan. Hai
uoc %doan ban dau x0 va x1 duoc cho truoc.
% Doi voi ham so cho truoc f(x). Cach thuc hien chuong trinh, theo cau truc cua vi
% du sau day:                root = secant(x0,x1,1.0E-12,10,1)
% Tham so index_f xac dinh cu the mot ham nao duoc dinh nghia ben duoi duoc
su % dung de tinh nghiem (case 1, case 2 vv..). Va sau khi thuc hien xong mot
vong lap % se cho ra cac ket qua sau: iterate_number, x, f(x), error
% Gia tri x la gia tri phong doan ban dau cua nghiem va roi la gia tri cua
% vong lap hien tai. iterate_number: so vong lap hien tai. Bien error la:
% error = gia tri vong lap moi tinh toan duoc tru di gia tri vong lap ngay truoc do
% va do la uoc luong sai so cua moi vong lap. Den day se quay lai de thuc hien
tiep % tuc vong lap cho den khi co duoc ket qua mong doi.
format short e
error = 1;
fx0 = f(x0,index_f);
it_count = 0;
iteration = [it_count x0 fx0]
while abs(error) > error_bd & it_count <= max_iterate
   it_count = it_count + 1;
   fx1 = f(x1,index_f);
   if fx1 - fx0 == 0
      disp('f(x1) = f(x0); Phep chia boi 0;  Stop')
      return
```

```
   end
   x2 = x1 - fx1*(x1-x0)/(fx1-fx0);
   error = x2 - x1;
%  Thuc hien phuong phap cat tuyen. Nhan Enter roi nhan lien tuc phim bat ky
%tren may de di den ket qua can tinh.
   iteration = [it_count x1 fx1 error]
   pause
   x0 = x1;
   x1 = x2;
   fx0 = fx1;
end
if it_count > max_iterate
   disp('The number of iterates calculated exceeded')
   disp('max_iterate.  An accurate root was not')
   disp('calculated.')
else
   format long
   root = x2
   format short e
   error
   format short
   it_count
end
% Ham so doi voi phuong trinh dinh nghia cua bai toan tim nghiem.
function value = f(x,index)
switch index
case 1
   value = x.^6 - x - 1;
      case 2
       value = x - exp(-x);
            case 3
             value = sin(x) - exp(-x);
               case 4
                value = x.^4-x-1;
end
```

Dùng chương trình 3.3 giải ví dụ 3.3.1, hàm được gọi theo cú pháp

root = secant(1,2,1.0E-9,10,1).

Kết quả sau 10 bước lặp theo sẽ là

- Nghiệm: root =1.13472413840152.
- Sai số: error = 5.0195e-012.

- Số lần lặp thực tế: it_count =9.
- Nghiệm xấp xỉ: root =1.1347.

Chương trình trên dùng để giải ví dụ 3.3.1 và một lớp các phương trình sau với một sai số có thể chấp nhận được $\varepsilon$=0.0001. Ta có thể thêm vào hoặc thay đổi các 'case i' để giải các phương trình khác.

a. Nghiệm của $x = e^{-x}$.(Chạy chương trình với root=secant(0,1,0.0001,15,2)).

b. Nghiệm dương nhỏ nhất của $e^{-x} = \sin(x)$. (root=secant(0.5,1,0.0001,15,3)).

c. Tất cả nghiệm thực của $x^4 - x - 1 = 0$.(root=secant(1.1,1.5,0.0001,15,4) và root=secant(-1,0,0.0001,15,4)).

### 3.3.3 Hàm số MATLAB fzero

Matlab bao hàm chuỗi thao tác tìm nghiệm fzero để sử dụng những ý tưởng liên quan đến phương pháp chia đôi và phương pháp cát tuyến. Như với nhiều chương trình Matlab, có vài chuỗi tập hợp có thể thực hiện được.

Lệnh **root = fzero (f_name, [a, b])**, tạo ra một nghiệm trong đoạn [a, b], tại nơi nó được giả thiết rằng $f(a).f(b) \le 0$.

Lệnh **root = fzero (f_name, x0)**, tìm một nghiệm của hàm số gần x0. Sai số có thể chấp nhận mặc định là độ chính xác tối đa của máy, mặc dù điều này có thể được thay đổi bởi người sử dụng. Đây là chuỗi thao tác tìm nghiệm tuyệt vời, kết hợp với việc bảo đảm hội tụ với hiệu quả cao.

# CHƯƠNG 4
# PHÉP NỘI SUY VÀ PHÉP TÍNH XẤP XỈ

## 4.1. PHÉP NỘI SUY ĐA THỨC

### 4.1.1 Đa thức nội suy

Cho hàm số $f:[a,b]\to R$ và trên đoạn $[a,b]$ cho ta một lưới các điểm chia (các điểm nút) $x_i$, i = 0, 1, …, n: $a\le x_1<x_2<...<x_n\le b$. Và tại các điểm nút $x_i$ có các giá trị hàm số $y=f(x)$ là $y_i=f(x_i)$ được cho dưới dạng bảng

| x | $x_0$ | $x_1$ | $x_2$ … $x_{n-1}$ | $x_n$ |
|---|---|---|---|---|
| y | $y_0$ | $y_1$ | $y_2$ … $y_{n-1}$ | $y_n$ |

Bài toán là tìm đa thức bậc n

$$P_n(x)=\sum_{i=0}^{n}a_ix^i \text{, sao cho } P_n(x_i)=f(x_i), \text{ i=0,…, n.} \qquad (4.1)$$

Ta gọi $P_n(x)$ là đa thức nội suy của hàm $f(x)$ tại các điểm nút $x_i$.

### 4.1.2 Sự tồn tại và duy nhất của đa thức nội suy

**Định lý 4.1.1** *Trong tất cả các đa thức có bậc $\le n$, có duy nhất một đa thức $P_n(x)$ thỏa $P_n(x_i)=y_i$, $i=0,1,...,n$.*

Chứng minh

Đặt $P_n(x)=\sum_{i=0}^{n}a_ix^i$. Theo giả thiết ta có

$$\sum_{j=0}^{n} a_j x_i^j = P_n(x_i) = a_0 + a_1 x_i + a_2 x_i^2 + ... + a_n x_i^n = y_i, \forall i = \overline{0,n}.$$

Đây là hệ n+1 phương trình tuyến tính $A\vec{a} = \vec{y}$, với n+1 ẩn là các hệ số của $P_n(x)$ được sắp xếp thành vectơ cột, n+1 chiều $\vec{a} = (a_0, a_1, ..., a_n)^T \in R^{n+1}$, và ma trận hệ số của hệ này là

$$A = \begin{pmatrix} 1 & x_0 & x_0^2 & ... x_0^n \\ 1 & x_1 & x_1^2 & ... x_1^n \\ ... & ... & ... & ... \\ 1 & x_n & x_n^2 & ... x_n^n \end{pmatrix}, \text{ và } \vec{y} = (y_0, y_1, ..., y_n)^T \in R^{n+1}.$$

Mặt khác, $\det(A) = \prod_{\substack{i,j=0 \\ i \neq j}}^{n} (x_i - x_j) \neq 0$ (Định thức Vandermon).

Vậy, tồn tại duy nhất $\vec{a} = A^{-1}\vec{y}$ là nghiệm của hệ $A\vec{a} = \vec{y}$. Định lý đã được chứng minh.

### 4.1.3 Sai số nội suy và chọn nút nội suy

**Định lý 4.1.2** *Giả sử $f(x)$ liên tục trên [a, b] và có đạo hàm liên tục đến cấp n+1 trên [a, b]. Khi đó tồn tại c thuộc [a, b] sao cho*

$$f(x) - P_n(x) = \frac{\omega_{n+1}(x)}{(n+1)!} f^{(n+1)}(c). \qquad (4.2)$$

Trong đó $\omega_{n+1}(x) = (x - x_0)(x - x_1)...(x - x_n)$.

**Chú ý**

i) Sai số nội suy $f(x) - P_n(x)$ có thể đánh giá

$$\left|f(x)-P_n(x)\right| \le \frac{M}{(n+1)!}\left|\omega_{n+1}(x)\right|, \text{ với } M=\max_{a\le x\le b}\left|f^{(n+1)}(x)\right|.$$

ii) Sai số nội suy $f(x)-P_n(x)$ phụ thuộc vào $\omega_{n+1}(x)$, tức là phụ thuộc vào sự phân bố các nút nội suy $x_i$. Do đó, cần chọn $x_i$ sao cho $\max_{a\le x\le b}\left|\omega_{n+1}(x)\right|$ là nhỏ nhất. Và chọn $x_i$ là nghiệm của đa thức Chebychev là cách chọn tối ưu nhất (điều này sẽ được làm rõ trong mục 4.2).

### 4.1.4 Đa thức nội suy Lagrange

Xét đa thức $L_i(x)$ là đa thức bậc $n$ được cho trước bởi

$$L_i(x)=\frac{(x-x_0)...(x-x_{i-1})(x-x_{i+1})...(x-x_n)}{(x_i-x_0)...(x_i-x_{i-1})(x_i-x_{i+1})...(x_i-x_n)}. \quad (4.3)$$

Với $L_i(x_j)=\delta_{ij}=\begin{cases}1 & khi \quad i=j\\ 0 & khi \quad i\ne j\end{cases}$. $L_i(x)$ được gọi là đa thức Lagrange cơ sở.

Khi đó, đa thức nội suy Lagrange là

$$P_n(x)=y_0L_0(x)+y_1L_1(x)+...+y_nL_n(x). \quad (4.4)$$

### 4.1.5 Các tỷ sai phân

Cho hàm số $y=f(x)$ xác định trên đoạn $[a,b]$ và tại các điểm nút $x_i$ có các giá trị hàm số $y=f(x)$ là $y_i=f(x_i)$ được cho dưới dạng bảng

| x | $x_0$ | $x_1$ | $x_2$ ... $x_{n-1}$ | $x_n$ |
|---|---|---|---|---|
| y | $y_0$ | $y_1$ | $y_2$ ... $y_{n-1}$ | $y_n$ |

- Tỷ sai phân cấp 1: $f[x_0,x_1]=\dfrac{f(x_1)-f(x_0)}{x_1-x_0}$.

- Tỷ sai phân cấp 2: $f[x_0, x_1, x_2] = \dfrac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}$.
- Tỷ sai phân cấp 3: $f[x_0, x_1, x_2, x_3] = \dfrac{f[x_1, x_2, x_3] - f[x_0, x_1, x_2]}{x_3 - x_0}$.
- Tổng quát, $f[x_0,..., x_n] = \dfrac{f[x_1,..., x_n] - f[x_0,..., x_{n-1}]}{x_n - x_0}$. (4.5)

Đây là tỷ sai phân cấp $n$, đôi khi còn được gọi là tỷ sai phân Newton.

Mối quan hệ của các tỷ sai phân bậc cao hơn đối với đạo hàm của bậc tương ứng được cho bởi định lý sau đây.

**Định lý 4.1.3** *Cho $n \geq 1$, và giả định $f(x)$ có thể khả vi $n$ lần liên tục trên đoạn $\alpha \leq x \leq \beta$. Cho $x_0, x_1,..., x_n$ là $n+1$ số phân biệt trong $[\alpha, \beta]$. Thì*

$$f[x_0,..., x_n] = \frac{1}{n!} f^{(n)}(c), \tag{4.6}$$

*với $c$ nằm giữa số nhỏ nhất và số lớn nhất của các số $x_0, x_1,..., x_n$.*

**CHƯƠNG TRÌNH MATLAB:** Ước lượng các tỷ sai phân.

Cho một tập hợp các giá trị $f(x_0),..., f(x_n)$, chúng ta cần tính một tập hợp các tỷ sai phân

$$f[x_0, x_1], f[x_0, x_1, x_2],..., f[x_0, x_1,..., x_n].$$

Hàm MATLAB được cho dưới đây có thể được dùng để làm điều đó, bằng cách dùng hàm số được gọi là

divdif_y = divdif(x_nodes,y_values).

Tuy nhiên chú ý rằng MATLAB không thực hiện phép chia cho 0, và vì thế vector nhập vào các biến x_nodes và y_values sẽ cần được xác định như là các vector chứa n+1 yếu tố. Một cách chính xác hơn, khi được cho trước các nút $\{x_0, x_1,..., x_n\}$ và các giá trị hàm số liên kết, thì chúng ta xác định được các vector nhập vào x_nodes và y_values như sau

$$x\_nodes = [x_0, x_1,..., x_n]$$
$$x\_nodes(i) = x_{i-1}, \quad i = 1,..., n+1$$
$$y\_values = [f(x_0), f(x_1),..., f(x_n)]$$
$$y\_nodes(i) = f(x_{i-1}), \quad i = 1,..., n+1$$

**Chương trình 4.1:**

```
function divdif_y = divdif(x_nodes,y_values)
% Day la ham
%       divdif_y = divdif(x_nodes,y_values)
% No se tinh toan cac ty sai phan cua ham. Gia tri cho truoc trong vector
%y_values, la gia tri cua ham f(x) tai nut da cho x_nodes.  Voi,
%       divdif_y(i) = f[x_1,...,x_i],  i=1,...,m
% m la do dai cua x_nodes.  Nhap vao gia tri cua x_nodes va y_values phu %hop
voi chuong trinh.

divdif_y = y_values;
m = length(x_nodes);
for i=2:m
   for j=m:-1:i
      divdif_y(j) = (divdif_y(j)-divdif_y(j-1)) ...
                    /(x_nodes(j)-x_nodes(j-i+1));
   end
end
```

Chạy chương trình với: x_nodes=[0.0 0.2 0.4 0.6 0.8 1.0 1.2];

y_values=[1.0 0.980067 0.921061 0.825336 0.696707 0.540302 0.362358];

Để được $D_i$ trong bảng 4.1.

**Bảng 4.1.** Các giá trị và các tỷ sai phân của cos(x).

| $i$ | $x_i$ | $\cos(x_i)$ | $D_i$ |
|---|---|---|---|
| 0 | 0.0 | 1.000000 | 0.1000000E+1 |
| 1 | 0.2 | 0.980067 | -0.9966711E-1 |
| 2 | 0.4 | 0.921061 | -0.4884020E+0 |
| 3 | 0.6 | 0.825336 | 0.4900763E-1 |
| 4 | 0.8 | 0.696707 | 0.3812246E-1 |
| 5 | 1.0 | 0.540302 | -0.3962047E-2 |
| 6 | 1.2 | 0.362358 | -0.1134890E-2 |

Với $D_i = f[x_0,..., x_i], \quad i = 0, 1, 2,...$

### 4.1.6 Công thức nội suy tỷ sai phân Newton

Cho $P_n(x)$ biểu thị đa thức nội suy $f(x_i)$ tại $x_i$ với $i$=0, 1,…, $n$. Thì, $\deg(P_n) \le n$ và $P_n(x_i) = f(x_i), \quad i = 0,1,..., n$. Vì vậy, các đa thức nội suy có thể được viết như sau

$$P_1(x) = f(x_0) + (x - x_0) f[x_0, x_1],$$

$$P_2(x) = f(x_0) + (x - x_0) f[x_0, x_1] + (x - x_0)(x - x_1) f[x_0, x_1, x_2],$$

…

$$P_n(x) = f(x_0) + (x - x_0) f[x_0, x_1] + ... + (x - x_0)(x - x_1)...(x - x_{n-1}) f[x_0, x_1,..., x_n]. \quad (4.7)$$

Đây được gọi là công thức đa thức nội suy tỷ sai phân Newton .

Chú ý rằng với $k \ge 0$,

$$P_{k+1}(x) = P_k(x) + (x - x_0)...(x - x_k) f[x_0, x_1,..., x_k, x_{k+1}]. \quad (4.8)$$

Thì chúng ta có thể đi từ bậc *k* đến bậc *k*+1 với số lượng tối thiểu các phép tính, khi các hệ số tỷ sai phân được tính toán.

Với $f(x)=\cos(x)$, Bảng 4.1 chứa 1 tập hợp các nút $x_i$, các giá trị hàm $f(x_i)$, và các tỷ sai phân $D_i = f[x_0,...,x_i]$, $i \geq 0$. Và nó được tính bằng việc dùng chương trình MATLAB divdif được cho ở trên. Bảng 4.2 chứa các giá trị của $P_n(x)$ với $x = 0.1, 0.3, 0.5$ đối với các giá trị khác nhau của $n$. Giá trị đúng của $f(x)$ được cho trong hàng cuối cùng của bảng. Các phép tính được làm bằng cách dùng công thức tỷ sai phân Newton và được thực hiện trong chương trình MATLAB interp được cho dưới đây.

**Bảng 4.2.** Nội suy cos(x).

| | $P_n(0.1)$ | $P_n(0.3)$ | $P_n(0.5)$ |
|---|---|---|---|
| 1 | 0.9900333 | 0.9700999 | 0.9501664 |
| 2 | 0.9949173 | 0.9554478 | 0.8769061 |
| 3 | 0.9950643 | 0.9553008 | 0.8776413 |
| 4 | 0.9950071 | 0.9553351 | 0.8775841 |
| 5 | 0.9950030 | 0.9553369 | 0.8775823 |
| 6 | 0.9950041 | 0.9553365 | 0.8775825 |
| True | 0.9950042 | 0.9553365 | 0.8775826 |

**CHƯƠNG TRÌNH MATLAB:** Ước lượng tỷ sai phân Newton của đa thức nội suy**.** Để giúp tạo nên các ví dụ giống như trong Bảng 4.1 và Bảng 4.2, chúng tôi trình bày chương trình MATLAB nội suy sau đây. Chương trình nên được thực hiện trước bởi hàm số divdif mà được cho trước đó trong phần này. Như phần divdif trước đó, chú ý rằng MATLAB không thực hiện phép chia cho 0.

**Chương trình 4.2:**

```
function p_eval = interp(x_nodes,divdif_y,x_eval)
% Day la ham  p_eval = interp(x_nodes,divdif_y,x_eval)
% No dung de tinh lop ty sai phan Newton cua da thuc noi suy bac m-1, tai  %cac nut duoc cho boi x_nodes, m la do dai cua x_nodes, va cac ty sai phan %duoc cho
```

```
trong divdif_y. Nhung diem ma phep noi suy se duoc thuc hien %va duoc cho
trong x_eval; va o dau ra, p_eval chua dung nhung gia tri  %tuong ung cua phep
noi suy da thuc .
%
m = length(x_nodes);
p_eval = divdif_y(m)*ones(size(x_eval));
for i=m-1:-1:1
   p_eval = divdif_y(i) + (x_eval - x_nodes(i)).*p_eval;
end
```

## 4.2. ĐA THỨC CHEBYSHEV

Xét hàm số $T_n(x)=\cos\left(n\cos^{-1}(x)\right)$, với $|x|\le 1$ và $n\in N$. (4.9)

Đặt $\theta=\cos^{-1}(x)$ hay $x=\cos(\theta)$ với $0\le\theta\le\pi$, thì: $T_n(x)=\cos(n\theta)$. (4.10)

Ta có,

$$T_0(x)=\cos(0.\theta)=1;\ T_1(x)=\cos(\theta)=x;\ T_2(x)=\cos(2\theta)=2x^2-1. \quad (4.11)$$

Tổng quát với $n\ge 1$, ta có

$$T_{n+1}(x)=\cos\left[(n+1)\theta\right]=c\cos(n\theta+\theta)=c\cos(n\theta)\cos\theta-\sin(n\theta)\sin\theta,$$

$$T_{n-1}(x)=\cos\left[(n-1)\theta\right]=c\cos(n\theta-\theta)=c\cos(n\theta)\cos\theta+\sin(n\theta)\sin\theta.$$

$$T_{n+1}(x)+T_{n-1}(x)=2c\cos(n.\theta)\cos(\theta)=2xT_n(x),$$

$$T_{n+1}(x)=2xT_n(x)-T_{n-1}(x). \quad (4.12)$$

Từ (4.12) ta tính được

$$T_3(x)=2xT_2(x)-T_1(x)=4x^3-3x;$$

$$T_4(x)=2xT_3(x)-T_2(x)=8x^4-8x^2+1...$$

Như vậy, $T_n(x)$ là đa thức đại số, bậc n và hệ số cao nhất là $2^{n-1}$.

Đa thức $T_n(x)$ được gọi là đa thức Chebyshev.

**Nhận xét**

i) $\max_{-1\le x\le 1}\left|T_n(x)\right|=1$ khi $x=x_i=\cos\frac{i\pi}{n}$, i=0, 1, …, n. Và $T_n(x_i)=1$ với i chẵn và $T_n(x_i)=-1$ với i lẻ.

ii) Gọi E là tập các đa thức hệ số thực bậc n (n>0). Xét phiếm hàm $\varphi: E\to R^+$, $\varphi(P)=\max_{-1\le x\le 1}\left|P(x)\right|$, $\forall P(x)\in E$. Khi đó $\frac{1}{2^{n-1}}T_n(x)$ là cực tiểu phiếm hàm $\varphi$.

Thật vậy, ta có $\varphi\left(\frac{1}{2^{n-1}}T_n(x)\right)=\frac{1}{2^{n-1}}$.

Giả sử có đa thức $P(x)\in E$ sao cho: $\varphi(P(x))<\frac{1}{2^{n-1}}$.

Khi đó xét $Q(x)=\left[\frac{1}{2^{n-1}}T_n(x)-P(x)\right]$ thì $Q(x)$ là đa thức bậc ≤ (n-1). Mặt khác với $x_i=\cos\frac{i\pi}{n}$, i=0, 1,…, n thì $Q(x_i)=\left[\frac{\cos i\pi}{2^{n-1}}-P(x_i)\right]$ có dấu dương, âm luân phiên nhau. Vậy $Q(x)$ có ít nhất n nghiệm (vô lý)■.

Do đó nếu chọn mốc nội suy là nghiệm của đa thức Chebyshev $T_{n+1}(x)$ thì $\omega_{n+1}(x)=\frac{1}{2^n}T_{n+1}(x)$ là cực tiểu phiếm hàm $\varphi$ và phép nội suy sẽ đạt ước lượng tốt nhất.

iii) Trên đoạn [-1, 1], $T_{n+1}(x)$ có nghiệm $x_i=\cos\left(\frac{2i+1}{2(n+1)}\right)\pi,\ i=\overline{0,n}$. Khi đó phép nội suy đạt ước lượng tốt nhất là

$$\left|f(x)-P_n(x)\right|\le\frac{M}{(n+1)!2^n},\ \text{với}\ M=\max_{a\le x\le b}\left|f^{(n+1)}(x)\right|. \qquad (4.13)$$

Trường hợp này, trong chương trình MATLAB dùng $c_n(x)$ thay cho $P_n(x)$.

Nếu xét trên đoạn [a, b], dùng phép biến đổi $t = \frac{2x-a-b}{b-a}$ sẽ đưa đoạn $a \le x \le b$ về đoạn $-1 \le t \le 1$. Do đó chúng tôi trình bày chương trình MATLAB sau đây trên đoạn [-1, 1].

**CHƯƠNG TRÌNH MATLAB:** Chương trình tạo hàm nội suy bậc n, hàm được dùng trong chương trình là $f(x) = e^x$ trên [-1, 1] (người sử dụng có thể thay thế hàm khác bằng lệnh gán cho fval hàm mong muốn), các mốc nội suy được chọn là nghiệm của đa thức Chebyshev và sai số lớn nhất là $\max_{-1 \le x \le 1} \left| e^x - c_n(x) \right|$. Đối số của chương trình là n, chỉ số điểm cần tham chiếu để tạo hàm nội suy. Khi chạy chương trình, nhấn liên tiếp Enter để xem đồ thị của đa thức nội suy và đồ thị thể hiện dáng điệu của sai số trong phép nội suy (Hình 4.1 và Hình 4.2 là kết quả của trường hợp n=2); trong cửa sổ Command của MATLAB sẽ in ra các tham số trả về của chương trình là sai số cực đại: maximum error, các nút: nodes, các giá trị nội suy tương ứng: fcn_values và các tỷ sai phân của hàm nội suy: div_diff_fcn.

**Chương trình 4.3:**

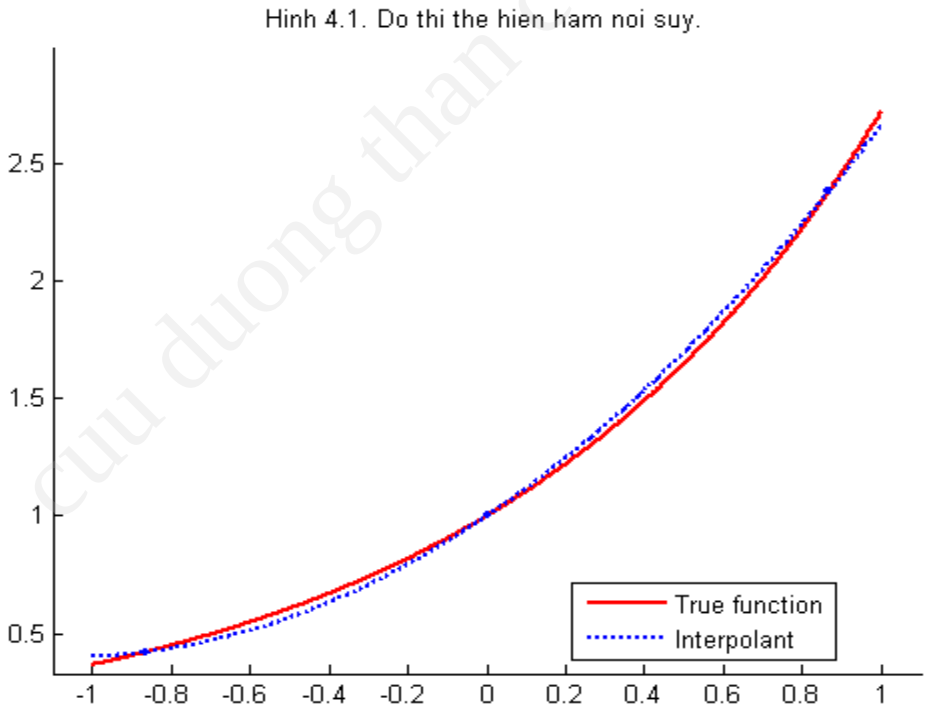
```
function [nodes, fcn_values, div_diff_fcn] = chebyshev_interp(n)
% Day la chuong trinh tao ham noi suy bac n, fcn(x) tren doan [-1,1], duoc %cho
ben duoi nhu la mot chuong trinh con co diem nodes la nghiem cua da %thuc
Chebyshev bac n+1 tren doan [-1,1] va chuong trinh ve hai do thi, %mot la do thi
ham chinh xac va da thuc noi suy cua no tren mot hình, va %hai la do thi the hien
sai so cua da thuc noi suy.
% Dau tien la viec tao cac nodes (cac hoanh do diem) va vi phan tuong ung %voi
tung diem do. n la tham so thay doi duy nhat cua chuong trinh, chi so % diem can
tham chieu de tao ham noi suy
% h la khoang cach giua hai hoanh do diem ke nhau. pi=3.1416
h = pi/(2*(n+1));
% Bien nodes la vecto chua cac hoanh do diem tham chieu gia tri nam tren
% [-1,1] thong qua tap gia tri ham cosin, n+1 diem nodes duoc tao
nodes = cos(h*[1:2:2*n+1]);
% fcn_values la vec to chua cac gia tri tung do diem tuong ung
fcn_values = fcn(nodes);
```

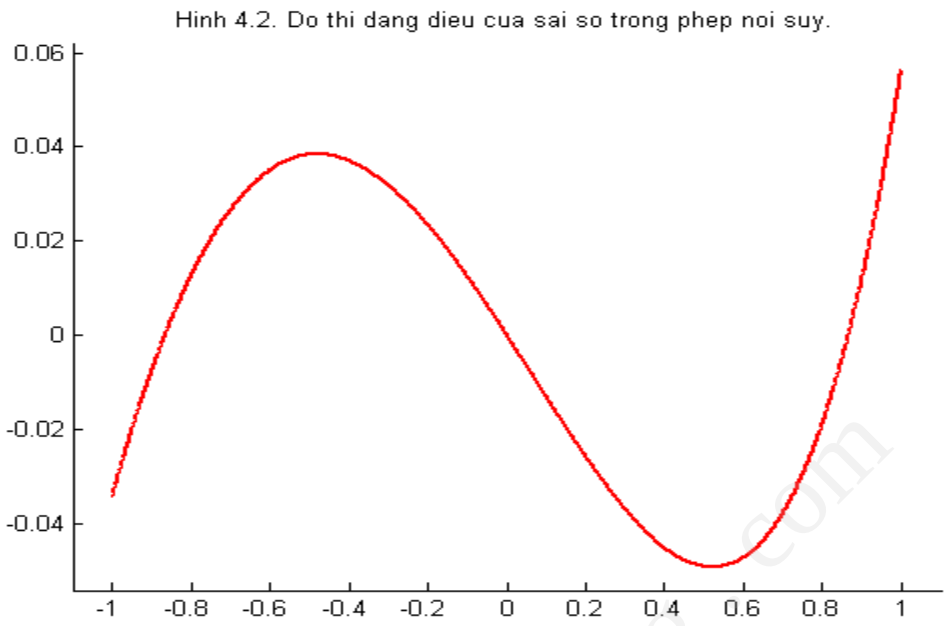
```
% divdif() la ham co san trong matlab dung de tinh ty sai phan cua da thuc % noi suy.
div_diff_fcn = divdif(nodes,fcn_values);
% Tao cac diem de ve do thi
x_eval = -1:.002:1;
true_fcn = fcn(x_eval);
y_eval = interp(nodes,div_diff_fcn,x_eval);
% Tao he truc de ve do thi cua ham va cac noi suy cua no. Gioi han do thi co % hoanh do thuoc [-1.1,1.1], gia tri tung do duoc khao sat tren doan [m,M].
m = min([min(true_fcn),min(y_eval)]);
if m > 0
   m = .9*m;
else
   m = 1.1*m; end
M = max([max(true_fcn),max(y_eval)]);
if M < 0
   M = .9*M;
else
   M = 1.1*M; end
axis([-1.1,1.1,m,M])
hold on
plot(x_eval,true_fcn,'r','LineWidth',2)
plot(x_eval,y_eval,':','LineWidth',2)
legend('True function','Interpolant',0)
title('Hinh 4.1. Do thi the hien ham noi suy.')
hold on
plot(nodes,fcn_values,'.','LineWidth',2,'MarkerSize',6)
hold off
disp('Vui long nhan Enter de tiep tuc.')
pause
clf
% Tao cua so cho viec ve do thi sai so error.
error = true_fcn - y_eval;
M = max(error);
if M < 0
   M = .9*M;
else
   M = 1.1*M; end
m = min(error);
if m > 0
   m = .9*m;
else
```

```
   m = 1.1*m; end
axis([-1.1,1.1,m,M])
hold on
% Tao bieu do the hien sai so trong phep noi suy.
plot(x_eval,error,'r','LineWidth',2)
title('Hinh 4.2. Do thi dang dieu cua sai so trong phep noi suy.')
hold off
disp('Vui long nhan Enter de tiep tuc.')
pause
% In ra man hinh sai so lon nhat.
disp(['maximum error = ',num2str(max(abs(error)))])
% Tao ham can tinh.
function fval = fcn(x)
fval = exp(x);
```

Kết quả khi chạy chương trình theo cú pháp

[nodes, fcn_values, div_diff_fcn] = chebyshev_interp(2)






```
maximum error = 0.056468
nodes =
   0.8660    0.0000   -0.8660
fcn_values =
   2.3774    1.0000    0.4206
div_diff_fcn =
   2.3774    1.5905    0.5320
```

Vài giá trị sai số của chương trình được thể hiện trong bảng sau

**Bảng 4.3.** Giá trị $\max_{-1\le x\le 1}\left|e^x - c_n(x)\right|$.

| *n* | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| max(erros) | 3.72E-1 | 5.65E-2 | 6.66E-3 | 6.40E-4 | 5.18E-5 | 3.62E-6 |

## 4.3. PHÉP NỘI SUY DÙNG HÀM GHÉP TRƠN (HÀM SPLINE)

Một hạn chế căn bản của các đa thức đã xét là nếu tăng số mốc nội suy lên thì bậc của đa thức nội suy cũng tăng theo. Điều này rất không thuận tiện trong việc tính toán. Hơn thế nữa người ta cần làm cho các đồ thị của hàm nội suy trơn mượt trên từng đoạn con của đoạn [a, b]. Dùng hàm spline trong phép nội suy sẽ giải quyết được các vấn đề trên, tức là ghép các đa thức bậc thấp lại với nhau

nhưng vẫn đảm bảo được tính khả vi liên tục tại các điểm nút. Ở đây chúng tôi chỉ trình bày hàm spline bậc ba tự nhiên vì nó thường đáp ứng yêu cầu của nhiều bài toán thực tế.

### 4.3.1 Spline bậc ba

Cho n điểm dữ kiện $(x_i, y_i)$, i=1, ..., n và $a=x_1<x_2<....<x_n=b$. Tìm hàm số s(x) xác định trên [a, b] nội suy dữ kiện: $s(x_i) = y_i$ thỏa các điều sau:

S1. s(x) là một đa thức có bậc $\le 3$ trên mỗi đoạn con $[x_{j-1}, x_j]$, $j = 2,3,..., n$.

S2. s(x), s'(x) và s"(x) liên tục với $a \le x \le b$.

S3. $s''(x_1) = s''(x_n) = 0$.

s(x) được gọi là hàm spline bậc 3 tự nhiên nội suy dữ kiện $\{(x_i, y_i)\}$. Nếu s(x) chỉ thỏa S1 và S2, gọi là hàm spline bậc 3 nội suy dữ kiện $\{(x_i, y_i)\}$.

### 4.3.2 Xây dựng hàm spline bậc 3 tự nhiên nội suy

Đưa vào các biến số $M_1, \ldots, M_n$ với $M_i \equiv s''(x_i)$. (4.14)

Chúng ta sẽ diễn đạt s(x) trong các số hạng của các giá trị không thể tính $M_i$, sau đó chúng ta sẽ tạo ra một hệ thống các phương trình tuyến tính để từ đó các giá trị $M_i$ có thể tính được. Vì s(x) là bậc 3 trên mỗi đoạn $[x_{j-1}, x_j]$, nên hàm số s"(x) là tuyến tính trên đoạn đó. Hàm số tuyến tính được xác định bởi các giá trị của nó tại 2 điểm và chúng ta dùng

$$s''(x_{j-1}) = M_{j-1}, \quad s''(x_j) = M_j. \tag{4.15}$$

Thì

$$s''(x) = \frac{(x_j - x)M_{j-1} + (x - x_{j-1})M_j}{x_j - x_{j-1}}, \qquad x_{j-1} \le x \le x_j. \tag{4.16}$$

Bây giờ chúng ta sẽ hình thành nguyên hàm thứ 2 của s"(x) trên $\left[x_{j-1}, x_j\right]$ và áp dụng các điều kiện nội suy

$$s\left(x_{j-1}\right)=y_{j-1}, \qquad s\left(x_j\right)=y_j. \tag{4.17}$$

Sau một vài thao tác, điều này đưa đến kết quả trong đa thức bậc 3

$$\begin{aligned} s(x)=&\frac{\left(x_j-x\right)^3 M_{j-1}+\left(x-x_{j-1}\right)^3 M_j}{6\left(x_j-x_{j-1}\right)}+\frac{\left(x_j-x\right)y_{j-1}+\left(x-x_{j-1}\right)y_j}{x_j-x_{j-1}} \\ &-\frac{1}{6}\left(x_j-x_{j-1}\right)\left[\left(x_j-x\right)M_{j-1}+\left(x-x_{j-1}\right)M_j\right]. \end{aligned} \tag{4.18}$$

với $x_{j-1}\le x\le x_j$. Sau nhiều phép đơn giản, điều này dẫn đến hệ các phương trình tuyến tính sau

$$\begin{aligned} &\frac{x_j-x_{j-1}}{6}M_{j-1}+\frac{x_{j+1}-x_{j-1}}{3}M_j+\frac{x_{j+1}-x_j}{6}M_{j+1} \\ &=\frac{y_{j+1}-y_j}{x_{j+1}-x_j}-\frac{y_j-y_{j-1}}{x_j-x_{j-1}}, \qquad j=2,3,\ldots,n-1. \end{aligned} \tag{4.19}$$

Các n-2 phương trình này cùng với giả định trước (S3)

$$M_1=M_n=0. \tag{4.20}$$

Cho phép tính các giá trị $M_1, \ldots, M_n$ và sau đó tính hàm số nội suy s(x). Hệ tuyến tính này được gọi là hệ thống 3 đường chéo (tridiagonal) và có các phương pháp đặc biệt để giải nó.

**Ví dụ 4.3.1** Tìm hàm spline bậc 3 tự nhiên nội suy dữ kiện

$$\left\{(1,1),(2,1/2),(3,1/3),(4,1/4)\right\}.$$

Số lượng các điểm là n=4; và tất cả $x_j-x_{j-1}=1$. Ta có

$$\begin{aligned} &M_1/6+2M_2/3+M_3/6=1/3, \\ &M_2/6+2M_3/3+M_4/6=1/12. \end{aligned}$$

Theo (4.20) suy ra $M_2=1/2, \quad M_3=0.$

Thay thế vào (4.18), chúng ta có được

$$s(x)=\begin{cases}\frac{1}{12}x^3-\frac{1}{4}x^2-\frac{1}{3}x+\frac{3}{2}, & 1\le x\le 2,\\ -\frac{1}{12}x^3+\frac{3}{4}x^2-\frac{7}{3}x+\frac{17}{6}, & 2\le x\le 3,\\ -\frac{1}{12}x+\frac{7}{12}, & 3\le x\le 4.\end{cases} \qquad (4.21)$$

### 4.3.3 Chương trình MATLAB spline

Để xây dựng một chương trình hoàn chỉnh tìm các hệ số của spline bậc ba tự nhiên chúng ta cần xây dựng và sử dụng chương trình giải hệ phương trình với ma trận ba đường chéo. Kiến thức này chúng tôi trình bày ở chương 6. Ở đây chúng tôi chỉ xin phép được giới thiệu gói chương trình MATLAB tiêu chuẩn chứa hàm ghép trơn (function spline). Nó có một vài chuỗi có thể thực hiện được; chuỗi tiêu chuẩn là

y=spline(x_nodes,y_nodes,x)

Điều này tạo ra hàm spline bậc 3 s(x) mà đồ thị của nó đi qua các điểm $\{(\xi_i,\eta_i)\mid i=1,...,n\}$, với $(\xi_i,\eta_i)=(x\_node(i),y\_node(i))$.

Và n là chiều dài của x_nodes (và y_nodes). Và với giá trị x đã cho, sẽ tính được giá trị y tương ứng của hàm spline. Nếu dùng các phát biểu

pp=spline(x_nodes,y_nodes)

[breaks, coefs, l, k, d]=unmkpp(pp)

thì ta có thể đạt được các hệ số của sự biểu diễn đa thức bậc 3 của s(x) trên mỗi đoạn con được xác định bởi tọa độ kề nhau trong x_nodes. Các thông tin chi tiết hơn về các lệnh này có thể đạt được từ lệnh help của MATLAB.

## 4.4. BÀI TOÁN XẤP XỈ HÀM THỰC NGHIỆM

Cho tập hợp điểm $\{M_k(x_k,y_k)\}_{k=1}^n$, trong đó có ít nhất hai điểm nút $x_i$ và $x_j$ khác nhau với $i \neq j$ và n là rất lớn. Khi đó việc xây dựng một đường cong đi qua tất cả các điểm đã cho sẽ không có ý nghĩa thực tế. Thay vào đó chúng ta sẽ tìm một hàm f(x) đơn giản sau cho nó thể hiện tốt nhất dáng điệu của tập hợp điểm đã cho và không nhất thiết phải đi qua các điểm đó. Có nhiều phương pháp để giải quyết vấn đề này, và một trong những phương pháp đó, là phương pháp bình phương bé nhất. Nội dung của phương pháp này là tìm cực tiểu của phiếm hàm

$$g(f)=\sum_{k=1}^{n}\left(f(x_k)-y_k\right)^2 \to \min. \qquad (4.22)$$

(Tổng bình phương các sai số là bé nhất)

Dạng của hàm cần xác định f(x) phụ thuộc vào nhiều yếu tố. Các dạng đơn giản nhất thường gặp trong thực tế là $f(x)=Ax+B$, $f(x)=Ax^2+Bx+C$, $f(x)=Ae^{Bx}$, $f(x)=Ax^B$, $f(x)=A\cos x+B\sin x$, … Để minh hoạ cho phương pháp, chúng tôi trình bày hai trường hợp đơn giản đầu và chương trình MATLAB để xác định hàm đa thức xấp xỉ thực nghiệm.

### 4.4.1 Trường hợp $f(x)=Ax+B$

Khi đó (4.22) có dạng $g(A,B)=\sum_{k=1}^{n}\left(Ax+B-y_k\right)^2 \to \min.$

Bài toán quy về việc tìm cực tiểu của hàm hai biến $g(A,B)$. Tọa độ điểm dừng của hàm được xác định từ hệ $\frac{\partial g}{\partial A}=\frac{\partial g}{\partial B}=0$ tương đương với

$$\begin{cases} A\sum_{k=1}^{n}x_k+nB=\sum_{k=1}^{n}y_k \\ A\sum_{k=1}^{n}x_k^2+B\sum_{k=1}^{n}x_k=\sum_{k=1}^{n}x_k y_k \end{cases} \qquad (4.23)$$

Theo giả thiết bài toán ta có $\left(\sum_{k=1}^{n} x_k\right)^2 - n\sum_{k=1}^{n} x_k^2 \neq 0$. Do đó hệ (4.23) có duy nhất nghiệm $A$, $B$. Đây là hệ số cần tìm.

**Ví dụ 4.4.1** Tìm hàm $f(x) = Ax + B$ xấp xỉ tốt nhất dữ kiện

| X | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|---|---|
| Y | 1 | 2 | 2 | 3 | 4 | 4 | 5 | 5 | 6 | 7 |

Ta có, n=10, $\sum_{k=1}^{n} x_k = 29$, $\sum_{k=1}^{n} y_k = 39$, $\sum_{k=1}^{n} x_k^2 = 109$ và $\sum_{k=1}^{n} x_k.y_k = 140$. Thế vào (4.22) ta được

$\begin{cases} 29A + 10B = 39 \\ 109A + 29B = 140 \end{cases}$. Suy ra $A \doteq 1.0803;\ B \doteq 0.7671$.

Vậy, $f(x) \approx 1.0803x + 0.7671$.

### 4.4.2 Trường hợp $f(x) = Ax^2 + Bx + C$

Khi đó (4.22) có dạng $g(A,B,C) = \sum_{k=1}^{n} \left(Ax^2 + Bx + C - y_k\right)^2 \to \min$.

Hệ xác định $A$, $B$ và $C$ là

$$\begin{cases} A\sum_{k=1}^{n} x_k^4 + B\sum_{k=1}^{n} x_k^3 + C\sum_{k=1}^{n} x_k^2 = \sum_{k=1}^{n} x_k^2 y_k \\ A\sum_{k=1}^{n} x_k^3 + B\sum_{k=1}^{n} x_k^2 + C\sum_{k=1}^{n} x_k = \sum_{k=1}^{n} x_k y_k \\ A\sum_{k=1}^{n} x_k^2 + B\sum_{k=1}^{n} x_k + Cn = \sum_{k=1}^{n} y_k \end{cases} \quad (4.24)$$

**Ví dụ 4.4.2** Tìm hàm $f(x) = Ax^2 + Bx + C$ xấp xỉ tốt nhất dữ kiện

| x | 1 | 1 | 2 | 3 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|---|
| y | 4.12 | 4.18 | 6.23 | 8.34 | 8.38 | 12.13 | 18.32 |

Từ bảng dữ kiện và hệ (4.24), ta có

$$\begin{cases} 1061A + 253B + 65C = 835.78 \\ 253A + 65B + 19C = 211.04 \\ 65A + 19B + 7C = 61.70 \end{cases} \text{. Suy ra, } \begin{cases} A \doteq 0.69 \\ B \doteq -0.71 \\ C \doteq 4.30 \end{cases}$$

Vậy, $f(x) \approx 0.69x^2 - 0.71x + 4.30$.

Trường hợp f(x) là đa thức bậc m (m>2) ta cũng thực hiện tương tự và trong MATLAB có hàm polyfit cho phép ta xây dựng những đa thức này.

Cú pháp: f=polyfit(x, y, n). Trong đó x, y là các vector chứa giá trị của dữ kiện, n là bậc của đa thức f được trả về bởi hàm.

**CHƯƠNG TRÌNH MATLAB:** Xác định đa thức xấp xỉ hàm thực nghiệm.

**Chương trình 4.4:**

```
function testpolyfit
% x=[1 1 2 2 2 3 3 4 5 6];y=[1 2 2 3 4 4 5 5 6 7];n=1;          % Vi du 4.4.1.
x=[1 1 2 3 3 4 5];y=[4.12 4.18 6.23 8.34 8.38 12.13 18.32]; n=2;     % Vi du 4.4.2
%x=[1 1 2 3 4 4 5];y=[3 4 6 9 12 13 20];n=3;% Da thuc bac ba.
f=polyfit(x,y,n)
x2 = 1:.1:5;
y2 = polyval(f,x2);
plot(x,y,'ro',x2,y2,'LineWidth',2)
title('Hinh 4.3. Do thi cua da thuc f (mau xanh) va cac diem du kien (mau do)')
grid on
```

Kết quả chạy chương trình 4.4, giải ví dụ 4.4.2 với

x=[1 1 2 3 3 4 5];y=[4.12 4.18 6.23 8.34 8.38 12.13 18.32]; n=2;

ta được

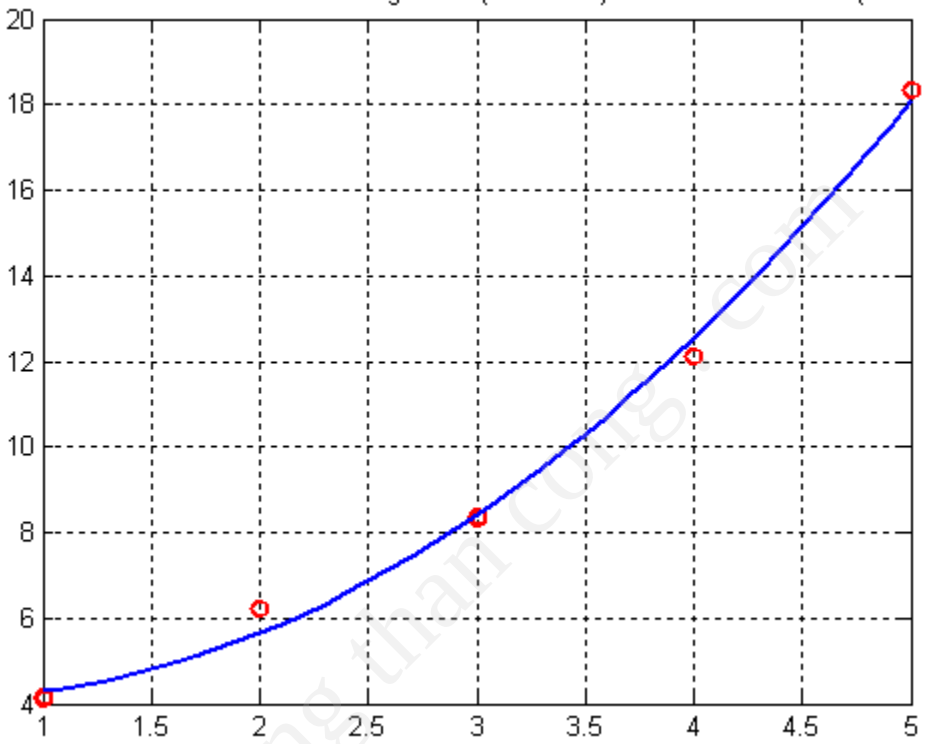
```
f =

    0.6929   -0.7064    4.2979
```

Tức là đa thức xấp xỉ hàm thực nghiệm

$$f(x) \approx 0.69x^2 - 0.71x + 4.30 .$$

Và đồ thị minh hoạ là Hình 4.3, vẽ đồ thị của đa thức xấp xỉ hàm thực nghiệm f(x) và các điểm dữ kiện đã cho.

Hinh 4.3. Do thi cua da thuc thuc nghiem f (mau xanh) va cac diem du kien (mau do).



# CHƯƠNG 5
# TÍCH PHÂN SỐ VÀ VI PHÂN

Cho tích phân xác định

$$I(f)=\int_a^b f(x)dx. \tag{5.1}$$

Như đã biết, với $F(x)$ là một nguyên hàm của $f(x)$ liên tục trên $[a,b]$, ta có công thức Newton – Leibniz $I(f)=F(b)-F(a)$. Ý tưởng chính của tích phân số là thay thế $f(x)$ bằng hàm xấp xỉ mà ta có thể tính được tích phân của nó.

## 5.1. CÔNG THỨC HÌNH THANG

### 5.1.1 Thiết lập công thức

Xấp xỉ $f(x)$ bằng đa thức nội suy tuyến tính $P_1(x)$ của nó tại a và b. Tích phân của $P_1(x)$ trên $[a,b]$ là

$$T_1(f)=(b-a)\left[\frac{f(a)+f(b)}{2}\right]. \tag{5.2}$$

Công thức này có thể tính được gần đúng tích phân $I(f)$ nếu $f(x)$ gần như là đường thẳng trên $[a,b]$. Và nếu $f(x)\geq 0$ trên $[a,b]$ thì (5.2) là diện tích hình thang có hai đáy là $f(a)$, $f(b)$ và chiều cao là $(b-a)$.

**Ví dụ 5.1.1** Tính tích phân $I=\int_0^1 \frac{dx}{1+x}$. (5.3)

Giá trị thực là $I=\log(2)\doteq 0.693147$. Dùng công thức (5.2), ta có

$$T_1 = \frac{1}{2}\left[1 + \frac{1}{2}\right] = \frac{3}{4}. \tag{5.4}$$

Công thức này gặp phải sai số là

$$I - T_1 \doteq -0.0569. \tag{5.5}$$

Để cải thiện phép tính xấp xỉ $T_1(f)$ trong (5.4), chia đoạn [a, b] thành những đoạn con có cùng độ dài và dùng (5.2) cho mỗi đoạn nhỏ đó.

**Ví dụ 5.1.2** Hãy tính giá trị I của ví dụ trước bằng cách dùng $T_1(f)$ trên 2 đoạn con có cùng độ dài. Với 2 đoạn con,

$$I = \int_0^{1/2} \frac{dx}{1+x} + \int_{1/2}^{1} \frac{dx}{1+x},$$

$$I \doteq \frac{1}{2}\left[\frac{1+2/3}{2}\right] + \frac{1}{2}\left[\frac{2/3+1/2}{2}\right].$$

$$T_2 = \frac{17}{24} \doteq 0.70833. \tag{5.6}$$

$$I - T_2 \doteq -0.0152. \tag{5.7}$$

Sai số trong $T_2$ chỉ bằng ¼ so với $T_1$ trong (5.5)

Ta có công thức tổng quát, bằng cách chia đoạn $[a, b]$ thành n đoạn con có cùng độ dài $h = (b-a)/n$ bởi các điểm chia $x_j = a + jh,\ j = 0,1,...,n$; và trên mỗi đoạn con $[x_{j-1}, x_j]$ ta thay $f(x)$ bởi đa thức nội suy tuyến tính. Thì

$$I(f) \approx h\left[\frac{f(x_0)+f(x_1)}{2}\right] + h\left[\frac{f(x_1)+f(x_2)}{2}\right] + ... + h\left[\frac{f(x_{n-1})+f(x_n)}{2}\right]. \tag{5.8}$$

Những số hạn ở bên phải được kết hợp để có công thức đơn giản hơn

$$T_n(f)=h\left[\frac{1}{2}f(x_0)+f(x_1)+f(x_2)+...+f(x_{n-1})+\frac{1}{2}f(x_n)\right]. \quad (5.9)$$

Đây gọi là công thức hình thang để tính xấp xỉ (5.1).

### 5.1.2 Đánh giá sai số

$$\left|I(f)-T_n(f)\right|\le\frac{M}{12}(b-a)h^2, \qquad M=\max_{a\le x\le b}\left|f''(x)\right|. \quad (5.10)$$

**Ví dụ 5.1.3** Tích phân $I=\int_0^1\frac{dx}{1+x}$

Ta có $f(x)=(1+x)^{-1}$, $f'(x)=-(1+x)^{-2}$, $f''(x)=2(1+x)^{-3}$.

Suy ra, $M=\max_{0\le x\le 1}\left|f''(x)\right|=2$, dẫn đến $\left|I(f)-T_n(f)\right|\le\frac{h^2}{12}(2)=\frac{h^2}{6}$.

### 5.1.3 Nhận xét chung

✓ Bài toán tích phân số có thể cho trước số đoạn chia n hoặc cho trước sai số $\varepsilon$. Khi cho trước sai số ta tính n như sau

$$\left|I(f)-T_n(f)\right|\le\frac{M}{12}(b-a)h^2=\frac{M}{12}\frac{(b-a)^3}{n^2}<\varepsilon.$$

✓ Trong (5.10), khi $n\to\infty$ thì $I(f)-T_n(f)\to 0$. Do đó, nếu $n$ được nhân đôi liên tục, thì giá trị của hàm số được dùng trong mỗi $T_{2n}(f)$ sẽ bao gồm tất cả giá trị của hàm trước đó được sử dụng trong $T_n(f)$, càng thể hiện chính xác hơn $I(f)$. Vì thế, trong chương trình MATLAB chúng tôi trình bày theo ý tưởng

này. Để minh hoạ giá trị của hàm số dùng như thế nào khi gấp đôi $n$, chúng ta hãy xét $T_2(f)$ và $T_4(f)$.

$$T_2(f) = h\left[\frac{f(x_0)}{2} + f(x_1) + \frac{f(x_2)}{2}\right]. \tag{5.11}$$

Với

$$h = \frac{b-a}{2},\ x_0 = a,\ x_1 = \frac{a+b}{2},\ x_2 = b.$$

$$T_4(f) = h\left[\frac{f(x_0)}{2} + f(x_1) + f(x_2) + f(x_3) + \frac{f(x_4)}{2}\right]. \tag{5.12}$$

Với

$$h = \frac{b-a}{4},\ x_0 = a,\ x_1 = \frac{3a+b}{4},\ x_2 = \frac{a+b}{2},\ x_3 = \frac{a+3b}{4},\ x_4 = b.$$

Trong (5.12), chỉ $f(x_1)$ và $f(x_2)$ cần tính giá trị vì các giá trị của hàm số khác được tìm ra từ (5.11).

**Ví dụ 5.1.4** Tính $T_n(f)$ của 3 tích phân

$$I^{(1)} = \int_0^1 e^{-x^2} dx \doteq 0.746824132812427. \tag{5.13}$$

$$I^{(2)} = \int_0^4 \frac{dx}{1+x^2} = \tan^{-1}(4) \doteq 1.32581766366803. \tag{5.14}$$

$$I^{(3)} = \int_0^{2\pi} \frac{dx}{2+\cos(x)} = \frac{2\pi}{\sqrt{3}} \doteq 3.62759872846846844. \tag{5.15}$$

Các kết quả được cho trong Bảng 5.1, sai số trong cách tính $I^{(1)}$ và $I^{(2)}$ giảm theo hệ số 4 khi $n$ gấp đôi. Ví dụ thứ 3, $I^{(3)}$, hội tụ nhanh hơn rất nhiều. Các đáp số cho $n$ = 32, 64 và 128 là đúng dựa theo giới hạn bởi sai số làm trong trên máy tính (16 số thập phân), nó kí hiệu bởi dấu * trong Bảng.

**Bảng 5.1.** Các ví dụ về quy tắc hình thang.

| $n$ | $I^{(1)}$ | | $I^{(2)}$ | | $I^{(3)}$ | |
|---|---|---|---|---|---|---|
| | Sai số | Tỷ lệ | Sai số | Tỷ lệ | Sai số | Tỷ lệ |
| 2 | 1.55E-2 | | -1.33E-1 | | -5.61E-1 | |
| 4 | 3.84E-3 | 4.08 | -3.59E-3 | 5.09 | -3.76E-2 | |
| 8 | 9.59E-4 | 4.03 | 5.64E-4 | 31.12 | -1.93E-4 | |
| 16 | 2.40E-4 | 4.01 | 1.44E-4 | -9.90 | -5.19E-9 | |
| 32 | 5.99E-5 | 4.00 | 3.60E-5 | 3.89 | * | 0.04 |
| 64 | 1.50E-5 | 4.00 | 9.01E-6 | 4.00 | * | -2.90 |
| 128 | 3.74E-6 | 4.00 | 2.25E-6 | 4.00 | * | |

**CHƯƠNG TRÌNH MATLAB:** Chúng tôi cung cấp 1 chương trình cho quy tắc hình thang. Chúng tôi tính được $T_n(f)$ khi $n = n_0, 2n_0, 4n_0, ..., 256n_0$, với $n_0$ được cho bởi người dùng. Khi $n$ được gấp đôi thành $2n$, tất cả các giá trị của hàm số xuất hiện trong $T_n(f)$ cũng có thể dùng để tính cho $T_{2n}(f)$. Chúng tôi cũng chấp nhận đối với một số tích phân của hàm $f(x)$, khi người sử dụng định rõ tích phân thông qua chương trình con $f$. Dòng chú giải của chương trình nói rõ cách tổ chức của chương trình.

**Chương trình 5.1:**

```
% Tao file trapezoidal.m
function [integral,difference,ratio]=trapezoidal(a,b,n0,index_f)
%  Day la ham [integral,difference,ratio]=trapezoidal(a,b,n0,index_f)
% Chuong trinh su dung phuong phap hinh thang voi n diem chia nham uoc % tinh
ham f tren doan [a,b].  Nhung gia tri cua n duoc su dung la
%       n = n0,2*n0,4*n0,...,256*n0
% Toan bo cac so tuong ung duoc tra ve trong vecto integral. Nhung gia tri  % sai
phan tuong ung voi cac so trong vecto integral duoc tra lai cho vecto %difference
%                 difference(i) = integral(i)-integral(i-1),  i=2,...,9
% Cac ghi nhan ve ti le dat ra cac muc giam duoc cho trong nhung gia tri ve % phan
nay. De su dung chuong trinh nay, dinh nghia tich phan su dung %trong ham duoc
cho ben duoi. Tham so index_f cho phep nguoi dung tinh % toan voi nhieu ham can
tinh tich phan khac nhau.

%  Khoi tao cac vecto dau ra.
integral = zeros(9,1);
```

```
difference = zeros(9,1);
ratio = zeros(9,1);

% Khoi tao cong thuc tinh tich phan theo quy tac hinh thang, n=2.
sumend = (f(a,index_f) +f(b,index_f))/2;
sum = 0;
%  Khoi tao cho truong hop n0 > 2.
if(n0 > 2)
   h = (b-a)/n0;
   for i=2:2:n0-2
      sum = sum + f(a+i*h,index_f);
   end
end

% Tinh toan cac tich phan, viec thuc hien tien hanh cho tung truong hop %duoc bo
sung thich hop.
for i=1:9
   n = n0*2^(i-1);
   h = (b-a)/n;
   for k=1:2:n-1
      sum = sum + f(a+k*h,index_f);
   end
   integral(i) = h*(sumend + sum);
end
% Tinh toan vi sai phan cho quy tac hinh thang tuong ung voi so nguyen lan %
luong hinh thang duoc chia nho. Va ti le giam cua tung vi sai phan nay.
difference(2:9)=integral(2:9)-integral(1:8);
ratio(3:9)=difference(2:8)./difference(3:9);

function f_value = f(x,index)
% Day la dinh nghia cac ham tinh theo quy tac hinh thang viec lua chon ham % ap
dung tinh phu thuoc vao tham so index_f.
switch index
case 1
   f_value = exp(-x.^2);
case 2
   f_value = 1 ./(1+x.^2);
case 3
   f_value = 1 ./(2+cos(x));
case 4
   f_value = 1 ./(2+sin(x))
case 5
```

```
   f_value = exp(x).*cos(x);
case 6
   f_value = (x.^pi).*sin(sqrt(x));
end
```

## 5.2. CÔNG THỨC SIMPSON

### 5.2.1 Thiết lập công thức

Ta xấp xỉ $f(x)$ bằng đa thức nội suy bậc hai $P_2(x)$ của nó tại $a$, $c=(a+b)/2$ và $b$. Khi đó

$$I(f)\approx\int_a^b P_2(x)dx=\frac{b-a}{6}\left[f(a)+4f\left(\frac{b+a}{2}\right)+f(b)\right] \qquad (5.16)$$

Tổng quát, ta chia đoạn $[a,b]$ thành 2n đoạn con có cùng độ dài $h=(b-a)/2n$ bởi các điểm chia $x_j=a+jh,\ j=0,1,...,2n$; và trên mỗi đoạn con $[x_{j-1},\ x_j]$ ta thay $f(x)$ bởi đa thức nội suy bậc hai $P_2(x)$. Ta được

$$I(f)\approx\frac{h}{3}\left[f(a)+4\sum_{i=0}^{n-1}f(x_{2i+1})+2\sum_{i=0}^{n-1}f(x_{2i})+f(b)\right]\equiv S_n(f). \qquad (5.17)$$

Đây gọi là công thức Simpson để tính xấp xỉ (5.1).

### 5.2.2 Đánh giá sai số

$$|I(f)-S_n(f)|\le\frac{M}{180}(b-a)h^4,\qquad M=\max_{a\le x\le b}|f^{(4)}(x)|. \qquad (5.18)$$

### 5.2.3 Nhận xét chung

✓ Bài toán tích phân số giải bằng công thức Simpson có thể cho trước số đoạn chia 2n hoặc cho trước sai số $\varepsilon$. Khi cho trước sai số ta tính n như sau:

$$\left|I(f)-S_n(f)\right| \le \frac{M}{180}(b-a)h^4 < \varepsilon .$$

✓ Trong (5.18), khi $n \to \infty$ thì $I(f)-T_n(f) \to 0$. Do đó, nếu $n$ được nhân đôi liên tục, thì giá trị của hàm số được dùng trong mỗi $S_{2n}(f)$ sẽ bao gồm tất cả giá trị của hàm trước đó được sử dụng trong $S_n(f)$, càng thể hiện chính xác hơn $I(f)$. Vì thế, trong chương trình MATLAB chúng tôi trình bày theo ý tưởng này.

Với $I^{(1)}$, $I^{(2)}$ và $I^{(3)}$ được cho trong ví dụ 5.1.3. Đối với các tích phân $I^{(1)}$ và $I^{(2)}$, tỷ lệ sai số đã giảm xuống theo hệ số 16. Đối với tích phân $I^{(3)}$, sai số hội tụ tới 0 nhanh hơn. Hãy so sánh Bảng 5.1 và bảng 5.2 để thấy rõ vấn đề.

**Bảng 5.2** . Các ví dụ về quy tắc Simpson.

| $n$ | $I^{(1)}$ | | $I^{(2)}$ | | $I^{(3)}$ | |
|---|---|---|---|---|---|---|
| | Sai số | Tỷ lệ | Sai số | Tỷ lệ | Sai số | Tỷ lệ |
| 2 | -3.56E-4 | | 8.66E-2 | | -1.26 | |
| 4 | -3.12E-5 | -895.93 | 3.95E-2 | 137.62 | 1.37E-1 | |
| 8 | -1.99E-6 | 11.11 | 1.95E-3 | 344.93 | 1.23E-2 | -2.76 |
| 16 | -1.25E-7 | 15.70 | 4.02E-6 | 160.49 | 6.43E-5 | 0.02 |
| 32 | -7.79E-9 | 15.95 | 2.33E-8 | 63.90 | 1.71E-9 | 0.02 |
| 64 | -4.87E-10 | 15.99 | 1.46E-9 | 63.99 | * | 0.02 |
| 128 | -3.04E-11 | 16.00 | 9.15E-11 | 64.10 | * | 0.02 |

**CHƯƠNG TRÌNH MATLAB.** Chúng tôi cung cấp một chương trình MATLAB cho quy tắc Simpson. Chương trình tính được $S_n(f)$ khi $n = n_0, 2n_0, 4n_0, ..., 256n_0$, với $n_0$ được cho bởi người dùng. Khi $n$ được gấp đôi thành $2n$, tất cả các giá trị của hàm số xuất hiện trong $S_n(f)$ cũng có thể dùng để

tính cho $S_{2n}(f)$. Chúng tôi cũng chấp nhận đối với một số tích phân của hàm $f(x)$, khi người sử dụng định rõ tích phân thông qua chương trình con $f$. Dòng chú giải của chương trình nói rõ cách tổ chức của chương trình.

**Chương trình 5.2:**

```
% Tao file simpson.m
function [integral,difference,ratio]=simpson(a,b,n0,index_f)
%  Day la ham [integral,difference,ratio]=simpson(a,b,n0,index_f)
% Ham nay ap dung nguyen tac Simpson voi n tieu doan tinh toan ham f  %tren doan [a,b].  Cac gia tri n duoc su dung
%                        n = n0,2*n0,4*n0,...,256*n0
%  Cac gia tri n phai khac khong #0. Gia tri tich phan tuong ung duoc tra lai % trong vecto integral. Cac vi sai phan thuc hien giua cac phep tich phan %duoc luu lai trong difference
%                 difference(i) = integral(i)-integral(i-1),  i=2,...,9
% Trong viec su dung chuong trinh, dinh nghia tich phan cac ham cho ben %duoi tham so index_f cho phep nguoi dung chon lua viec tinh tich phan  %giua cac ham khac nhau. Ratio la ty le cua cac vi sai phan.

% Dinh nghia vecto ket qua dau ra.
integral = zeros(9,1);
difference = zeros(9,1);
ratio = zeros(9,1);

% Dinh nghia cong thuc tich phan simpson.
sumend = f(a,index_f) + f(b,index_f);
sumodd = 0;
sumeven = 0;

% Khoi tao cho truong hop n0 > 2.
if(n0 > 2)
   h = (b-a)/n0;
   for i=2:2:n0-2
      sumeven = sumeven + f(a+i*h,index_f);
end
end
```

```
% Tinh toan so cac tich phan, cho moi truong hop bang viec bo sung thich  %hop
vao truong hop tinh truoc do.
for i=1:9
   n = (n0)*(2^(i-1));
   h = (b-a)/n;
   sumeven = sumeven + sumodd;
   sumodd = 0;
   for k=1:2:n-1
      sumodd = sumodd + f(a+k*h,index_f);
   end
   integral(i) = h*(sumend + 4*sumodd + 2*sumeven)/3;
end

% Tinh toan cac sai phan cua qua trinh tinh toan tich phan simpson va ti le % giam
cua cac vi sai phan nay.
difference(2:9)=integral(2:9)-integral(1:8);
ratio(3:9)=difference(2:8)./difference(3:9);

function f_value = f(x,index)
% Day la dinh nghia cac ham tich phan. Viec lua chon ham ap dung tinh
% phu thuoc vao tham so index_f.
switch index
case 1
   f_value = exp(-x.^2);
case 2
   f_value = 1 ./(1+x.^2);
case 3
   f_value = 1 ./(2+cos(x));
case 4
   f_value = 1 ./(2+sin(x));
case 5
   f_value = exp(x).*cos(x);
end
```

## 5.3. CÔNG THỨC TÍCH PHÂN GAUSS

### 5.3.1 Thiết lập công thức

Các kỹ thuật tích phân số như công thức hình thang, công thức Simpson sẽ không hiệu quả trong tính toán phần tử hữu hạn vì sự phức tạp khi dùng phép ánh xạ. Ở đây, chúng tôi trình bày công thức tích phân Gauss (Gauss – Legendre) sẽ thích hợp hơn và đó là một công cụ chuẩn để phát triển các phần tử ánh xạ. Trong khuôn khổ bài viết này, chúng tôi trình bày đối với trường hợp một chiều để giải số tích phân (5.1).

Trong công thức tích phân Gauss, người ta luôn giả sử rằng tích phân của hàm $f(x)$ được lấy trên đoạn $-1 \le x \le 1$. Thật vậy, bằng cách đặt

$$x = \frac{b+a}{2} + \frac{b-a}{2}t,$$

thì
$$\int_a^b f(x)dx = \int_{-1}^{1} f\left(\frac{b+a}{2} + \frac{b-a}{2}t\right)\frac{b-a}{2}dt.$$

Vấn đề đặt ra là tìm n số $x_1, x_2, ..., x_n \in [-1, 1]$ và n số $w_1, w_2, ..., w_n$, để

$$I(f) = \int_{-1}^{1} f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i) = G_n(f). \qquad (5.19)$$

Và là công thức đúng với mọi đa thức có bậc $\le 2n-1$.

Trong đó, n điểm $x_i$ gọi là các điểm Gauss và chọn theo thứ tự là nghiệm của đa thức Legendre, các hệ số tương ứng $w_i$ gọi là các trọng số và là nghiệm của hệ

$$\sum_{i=1}^{n} w_i x_i^j = \frac{1-(-1)^{j+1}}{j+1}, \quad j = \overline{1,(n-1)}.$$

Công thức (5.19) là công thức tích phân Gauss.

Vài kết quả ban đầu

- n=1 $I(f)=\int_{-1}^{1} f(x)dx \approx 2f(0)=G_1(f).$
- n=2 $I(f)=\int_{-1}^{1} f(x)dx \approx f\left(-\frac{1}{\sqrt{3}}\right)+f\left(\frac{1}{\sqrt{3}}\right)=G_2(f).$
- n=3 $I(f)=\int_{-1}^{1} f(x)dx \approx \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right)+\frac{8}{9}f(0)+\frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right)=G_3(f).$

### 5.3.2 Công thức sai số

$$\left|I(f)-G_n(f)\right| \le \frac{2^{2n+1}(n!)^4}{\left[(2n)!\right]^3(2n+1)}M, \qquad M=\max_{a\le x\le b}\left|f^{(2n)}(x)\right|. \quad (5.20)$$

## CHƯƠNG TRÌNH MATLAB: Tính tích phân bằng công thức Gauss.

**Chương trình 5.3:**

```
% Tao file gaussint.m
function [G,xi,wi]=gaussint(a,b,n,index_f)
% Ham gaussint(a,b,n,index_f), tinh tich phan cua mot ham tu diem a den %diem
b su dung n diem chia, quy tac Gauss duoc tien hanh cho ham bac %2*n-1.
% a,b   -  Gioi han tinh tich phan.
% n     -  Bac cua cong thuc.
% G     -  Gia tri tinh tich phan theo cong thuc Gauss
% xi,wi -  Cac diem Gauss va cac trong so, phep tinh xac dinh tren doan[-1,1]
% index_f - Tham so lua chon ham nao de tinh tinh phan.

% Kiem tra doi so nhap vao khi goi chuong trinh, phai co it nhat 3 doi so.
if nargin < 4, n=4; end;
if nargin<3, error('Ham phai co it nhat 4 doi so.'); end;
u=(1:n-1)./sqrt((2*(1:n-1)).^2-1);          % Tao ma tran cot u.
[vc,xi]=eig(diag(u,-1)+diag(u,1));          % vc la tri rieng, xi la vecto rieng cua %
ma tran tong cua diag(u,-1), diag(u,1); voi diag(u,k) la lay phan tu tren %duong
cheo thu k cua u.
```

```
[xi,k]=sort(diag(xi));          % Lay xi la cac phan tu tren duong
%cheo chinh, tra ve ma tran cot va sap tu nho den lon; xi chinh la nghiem %cua da
thuc Legendre.
wi=2*vc(1,k)'.^2;               % Tinh wi; A' la chuyen vi ma tran A.
x=(a+b)/2+((b-a)/2)*xi;         % Doi bien de tinh tich phan Gauss.
f=fcn(x,index_f)*(b-a)/2;       % Tinh f theo cong thuc Gauss.
G=wf(:)'*f(:);                  % Tinh tich phan theo cong thuc Gaus

function f_value = fcn(x,index)
% Day la dinh nghia cac ham tinh tich phan.Viec lua chon ham ap dung tinh
% phu thuoc vao tham so index_f.
switch index
case 1
   f_value = exp(-x.^2);          %%[0,1]
case 2
   f_value = 1 ./(1+x.^2);        %%[0,4]
case 3
   f_value = 1 ./(2+cos(x));      %%[0,2*pi]
case 4
   f_value = 1 ./(2+sin(x));      %%[0,2*pi]
case 5
   f_value = exp(x).*cos(x);      %%[0,pi]
case 6
   f_value = (x.^pi).*sin(sqrt(x));  %%[0,pi]
end
```

## Chú thích

- Cả ba chương trình 5.1, 5.2 và 5.3 đều có thể chọn hàm để tính tích phân
  - index_f=1, 2, 3 là dùng case 1, 2, 3 để tính $I^{(1)}$, $I^{(2)}$, $I^{(3)}$.
  - index_f=4 là dùng case 4 để tính $I^{(4)} = \int_0^{2\pi} \frac{dx}{2+\sin(x)} = I^{(3)}$.
  - index_f=5, 6 là dùng case 5, 6 để tính $\int_0^{\pi} e^x.\cos(x)dx$, $\int_0^{\pi} x^{\pi}.\sin\left(\sqrt{x}\right)dx$.

- Chương trình 5.3 dựa trên ý tưởng chính sau

Ý tưởng chính của việc giải bài toán tìm $x_i \in [-1, 1]$ và $w_i$ thỏa (5.19) là chọn $x_i$ là nghiệm của đa thức Legendre. Đa thức Legendre $P_n(x)$ xác định theo công thức truy hồi sau

$$\begin{cases} P_{n+1}(x) = \dfrac{2n+1}{n+1} x P_n(x) - \dfrac{n}{n+1} P_{n-1}(x), \quad n \geq 1, \\ P_0(x) = 1, \ P_1(x) = x. \end{cases} ; \ \deg(P_n) = n, \ P_n(1) = 1.$$

Cũng có thể xác định các đa thức Legendre bằng công thức

$$P_0(x) = 1, \ P_n(x) = \frac{1}{n! 2^n} . \frac{d^n}{x^n} \left( \left( x^2 - 1 \right)^n \right), \ n = 1, 2, ....$$

Vài đa thức Legendre đầu tiên

$$P_0(x) = 1; P_1(x) = x;$$

$$P_2(x) = \frac{3}{2} x^2 - \frac{1}{2};$$

$$P_3(x) = \frac{5}{2} x^3 - \frac{3}{2} x;$$

$$P_4(x) = \frac{35}{8} x^4 - \frac{15}{4} x^2 + \frac{3}{8}, \ldots$$

Tính chất quan trọng của đa thức Legendre là nó có n nghiệm thực phân biệt nằm trong [-1, 1] và đối xứng qua gốc tọa độ. Khi đó các $w_i$ là nghiệm của hệ

$$\begin{cases} \sum_{i=1}^{n} w_i x_i^j = \dfrac{1 - (-1)^{j+1}}{j+1}, \\ j = 0, 1, ..., (n-1). \end{cases}$$

Định thức của hệ này là định thức Vandermond $D = \prod_{i>j} (x_i - x_j) \neq 0$.

Vì vậy, $w_i$ xác định duy nhất.

Chúng tôi cung cấp một chương trình tìm các hệ số của đa thức Legendre và tính nghiệm của các đa thức đó. Tham số đưa vào là n, bậc của đa thức cần tìm. Hàm trả về là đa thức Legendre bậc n và các nghiệm của nó.

**Chương trình 5.3a:**

```
% Tao file legendre.m
function [p,r]=legendre(n)
if n<0, error('Bac da thuc am.');end;
p1=[1]; p2=[1 0]; pp=p2;
if n==0, p=p1, r=[ ]; return; end;
if n==1, p=p2; r=[0]; return; end;
for i=2:n
   m=length(p1)+2;
   for j=m:-1:3, p1(j)=p1(j-2); end;
   p1(1)=0; p1(2)=0;
   p=(2*i-1)/i*conv(pp,p2)-(i-1)/i*p1;
   p1=p2; p2=p;
end;
r=roots(p);
```

## 5.4. VI PHÂN SỐ

Ta có $$f'(x)=\lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$$

Do đó, $$f'(x)\approx\frac{f(x+h)-f(x)}{h}=D_h f(x) \qquad (5.21)$$

Công thức (5.21) là công thức xấp xỉ đạo hàm của $f(x)$.

Dùng định lý Taylor để khai triển $f(x+h)$ tại $x$, ta có

$$f(x+h)=f(x)+hf'(x)+\frac{h^2}{2}f''(c)\rightarrow f'(x)-D_h(x)=-\frac{h}{2}f''(c). \quad (5.22)$$

Với $c$ nằm giữa $x$ và $x+h$.

Với $h>0$ thì (5.21) gọi là công thức sai phân tiến và sai số là $\frac{h}{2}f''(c)$.

Thay $h$ bởi $-h$ vào (5.21) ta được $f'(x)\approx\frac{f(x)-f(x-h)}{h}$ và gọi là công thức sai phân lùi và sai số cũng là $\frac{h}{2}f''(c)$ với $c$ nằm giữa $x$ và $x-h$.

### 5.4.1. Vi phân số dùng phép nội suy

Lấy $P_n(x)$ là đa thức nội suy bậc n của $f(x)$ tại n+1 điểm $x_0$, …, $x_n$. Để tính $f'(x)$ tại x=t, ta dùng $f'(t)\approx P_n'(t)$. (5.23)

Chẳng hạn với n=2, t=$x_1$, $x_0$=$x_1$ – h, $x_2$=$x_1$+h, thì

$$P_2(x)=\frac{(x-x_1)(x-x_2)}{2h^2}f(x_0)+\frac{(x-x_0)(x-x_2)}{-h^2}f(x_1)+\frac{(x-x_0)(x-x_1)}{2h^2}f(x_2),$$

$$P_2'(x)=\frac{x-x_0}{2h^2}\left[f(x_2)-2f(x_1)\right]+\frac{x-x_1}{2h^2}\left[f(x_2)-2f(x_0)\right]+\frac{x-x_2}{2h^2}\left[f(x_0)-2f(x_1)\right], \quad (5.24)$$

$$P_2''(x)=\frac{f(x_2)-2f(x_1)+f(x_0)}{h^2}. \quad (5.25)$$

Từ (5.24), ta được

$$\begin{cases} f'(x_0)\approx P_2'(x_0)=\dfrac{-3f(x_0)+4f(x_0+h)-f(x_0+2h)}{2h} & (5.26)\\ f'(x_1)\approx P_2'(x_1)=\dfrac{f(x_0+2h)-f(x_0)}{2h} & (5.27)\\ f'(x_2)\approx P_2'(x_2)=\dfrac{f(x_0)-4f(x_0+h)+3f(x_0+2h)}{2h} & (5.28) \end{cases}$$

Công thức (5.26) gọi là công thức sai phân tiến; công thức (5.27) gọi là công thức sai phân hướng tâm và thường viết dưới dạng

$$f'(x) \approx \frac{f(x+h)-f(x-h)}{2h}.$$

(5.29)

Công thức (5.28) gọi là công thức sai phân lùi và thường được viết

$$f'(x) \approx \frac{f(x-2h)-4f(x-h)+3f(x)}{2h}.$$

(5.30)

Đối với đạo hàm cấp hai, dùng (5.25) ta viết

$$f''(x) \approx \frac{f(x+h)-2f(x)+f(x-h)}{2h}.$$

(5.31)

Đây gọi là công thức sai phân hướng tâm đối với đạo hàm cấp hai của $f(x)$ tại x.

Làm tương tự như đối với (5.21), sai số trong (5.26), (5.27) và (5.28) lần lượt là: $\frac{h^2}{3}f'''(c_0)$, $\frac{h^2}{6}f'''(c_1)$ và $\frac{h^2}{3}f'''(c_2)$ với $c_0, c_1, c_2 \in [x_0, x_2]$.

Các công thức phần vi phân số sẽ được dùng ở phần sau, ở đây chúng tôi sẽ giới thiệu vấn đề liên quan đến Symbolic MATLAB, nó rất dễ áp dụng và có thể áp dụng vào chương trình toán Trung học phổ thông.

**CHƯƠNG TRÌNH MATLAB:** Vì đã có chương trình MATLAB để tính các tỷ sai phân và mối liên hệ giữa các tỷ sai phân và đạo hàm, nên chúng tôi chỉ trình bày thêm một vài chương trình sử dụng biến Symbolic và hướng đến ứng dụng nó vào chương trình toán Trung học phổ thông, đồng thời cũng góp phần làm nổi bật thêm các công dụng của MATLAB. Symbolic MATLAB là thư viện các phép toán kiểu ký tự được đưa vào môi trường số học của MATLAB; nó làm phong phú và tiện ích thêm với nhiều kiểu tính toán khác cho phần tính số và đồ hoạ đã có trước đây trong thư viện MATLAB (Thông tin chi tiết xin xem Tài liệu tham khảo tiếng Việt [3]). Chúng ta hãy cụ thể hoá vấn đề với bài toán cơ bản sau và các chương trình MATLAB dưới đây là để giải bài toán này.

Cho hàm $f(x)=\dfrac{1}{5+4\cos x}$, với $x\in[-2\pi;\,2\pi]$, đây cũng là miền xác định mặc nhiên của Symbolic MATLAB. Tính $f'$, $f''$ và $f'''$, vẽ kết hợp các đồ thị của $f$, $f'$, $f''$ và $f'''$. Ngoài ra

a. Tìm x để $f'''=0$ và vẽ đồ thị minh hoạ.

b. Xác định điểm cực tiểu của $f$.

c. Tính hai lần nguyên hàm của $f''$, so sánh với $f$ và nêu nhận xét.

**Chương trình 5.4:** (Các file chương trình có trong thư mục viphanso.)

```
% Chuong trinh tao file diff.m.
% Tieu de: Tinh dao ham cap 1, cap 2, cap 3 cua ham f va ve do thi ket hop
syms x                  % Khai bao x la bien Symbolic.
f=1/(5+4*cos(x));
f1=diff(f);             % Dao ham cap 1 cua f.
f2=diff(f,2);           % Dao ham cap 2 cua f.
pretty(f1);             % Viet f1 o dang quen thuoc cua toan hoc.
pretty(f2);             % Viet f2 o dang quen thuoc cua toan hoc.
f3=diff(f,3);           % Dao ham cap 3 cua f.
f3=simple(f3); pretty(f3);  % Viet dang thu gon cua f3.
% Ve ket hop, chia mien ve do thi thanh hai dong, hai cot.
subplot(2,2,1)
```

```
ezplot(f)                % Ve f voi mien xac dinh mac nhien.
title('Do thi cua ham f.')
subplot(2,2,2)
ezplot(f1)
title('Do thi cua ham f1, dao ham cap 1 cua f.')
subplot(2,2,3)
ezplot(f2)
title('Do thi cua ham f2, dao ham cap 2 cua f.')
subplot(2,2,4)
ezplot(f3)
title('Do thi cua ham f3, dao ham cap 3 cua f.')
```

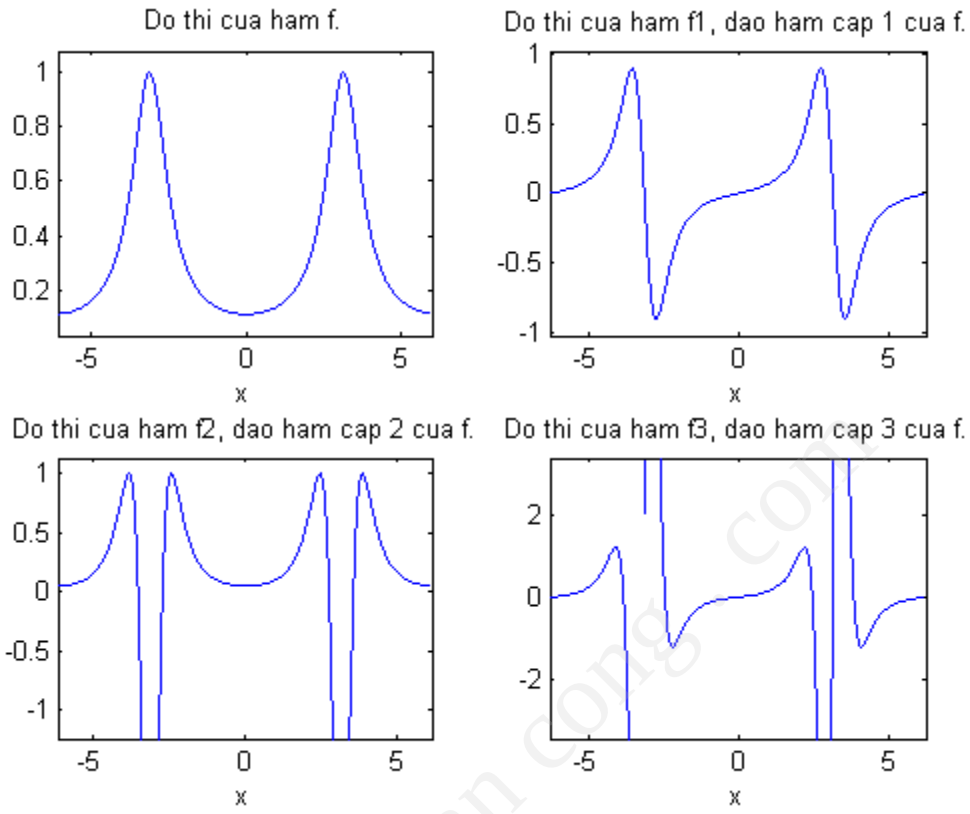
- Kết quả chạy file diff.m ta được lần lượt $f'$, $f''$ và $f'''$ và Hình 5.1.

$$4\,\frac{\sin(x)}{(5+4\cos(x))^2}$$

$$32\,\frac{\sin(x)^2}{(5+4\cos(x))^3}+4\,\frac{\cos(x)}{(5+4\cos(x))^2}$$

$$4\,\frac{\sin(x)\,(96\sin(x)^2+80\cos(x)+80\cos(x)^2-25)}{(5+4\cos(x))^4}$$

**Hình 5.1. Đồ thị của $f$, $f'$, $f''$ và $f'''$.**



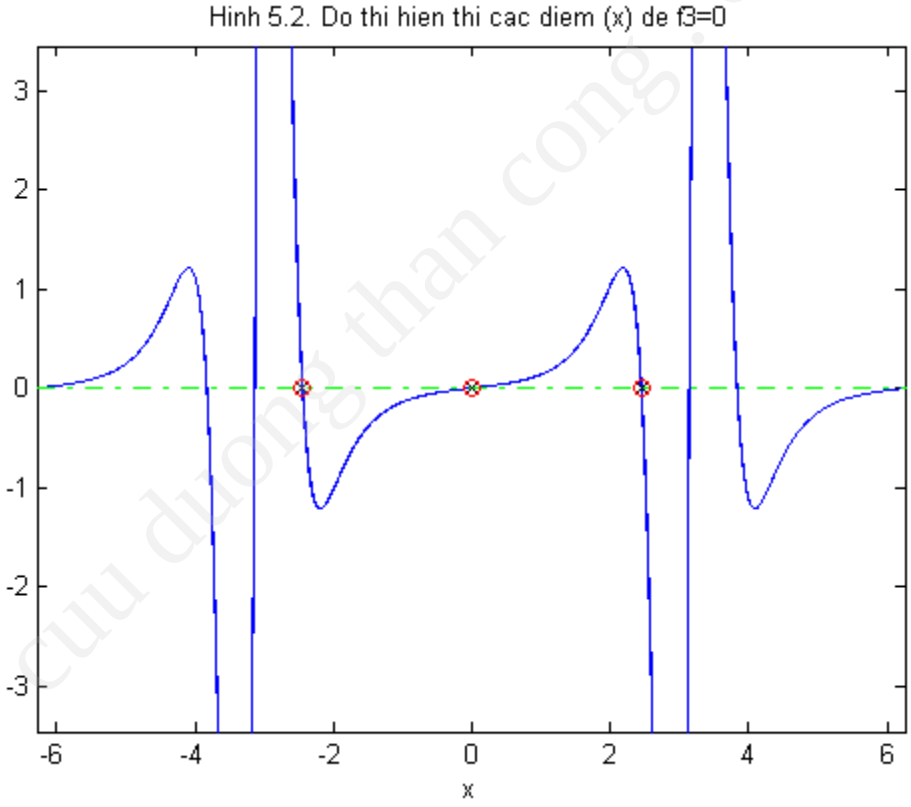

**Chương trình 5.4a:** (Các file chương trình có trong thư mục viphanso.)

```
% Chuong trinh tao file diffa.m.
% Tieu de: Tim cac diem x lam cho dao ham cap 3 cua f bang 0 va ve do thi.
syms x
f=1/(5+4*cos(x));
f3=diff(f,3);                % Dao ham cap 3 cua f.
f3=simple(f3);               % Viet dang thu gon cua f3.
pretty(f3);
z=solve(f3);                 % Giai phuong trinh f3=0.
format;                      % Lay 5 chu so le.
x0=double(z)                 % Chuyen ma tran, bieu thuc Symbolic ve dang so.
ezplot(f3)      % Ve do thi f3 kieu bien Symbolic voi mien xac dinh mac nhien.
hold on;                     % Giu he truc toa do vua ve cho cac hinh sau.
plot([-2*pi,2*pi],[0,0],'g-.')          % Ve duong "dut khuc" mau xanh la cay voi
f3=0.
title('Hinh 5.2. Do thi hien thi tat ca cac diem (x) de f3=0')
%%% x0=[x0-2*pi x0 x0+2*pi];          % Lay tat ca cac diem lam cho f3=0.
plot(x0,0*x0,'kx')                   % Ve diem dau "x" mau den de f3=0.
plot(x0, 0*x0,'ro')                  % Ve diem dau "o" mau do de f3=0.
```

Kết quả chạy file diffa.m ta được lần lượt f3 và các giá trị x=x0 để f3=0 và Hình 5.2 như sau

$$4\,\frac{\sin(x)\,(96\sin(x)^2 + 80\cos(x) + 80\cos(x)^2 - 25)}{(5 + 4\cos(x))^4}$$

```
x0 =
         0
         0 + 2.4381i
         0 - 2.4381i
    2.4483
   -2.4483
```




**Chương trình 5.4b:** (Các file chương trình có trong thư mục viphanso.)

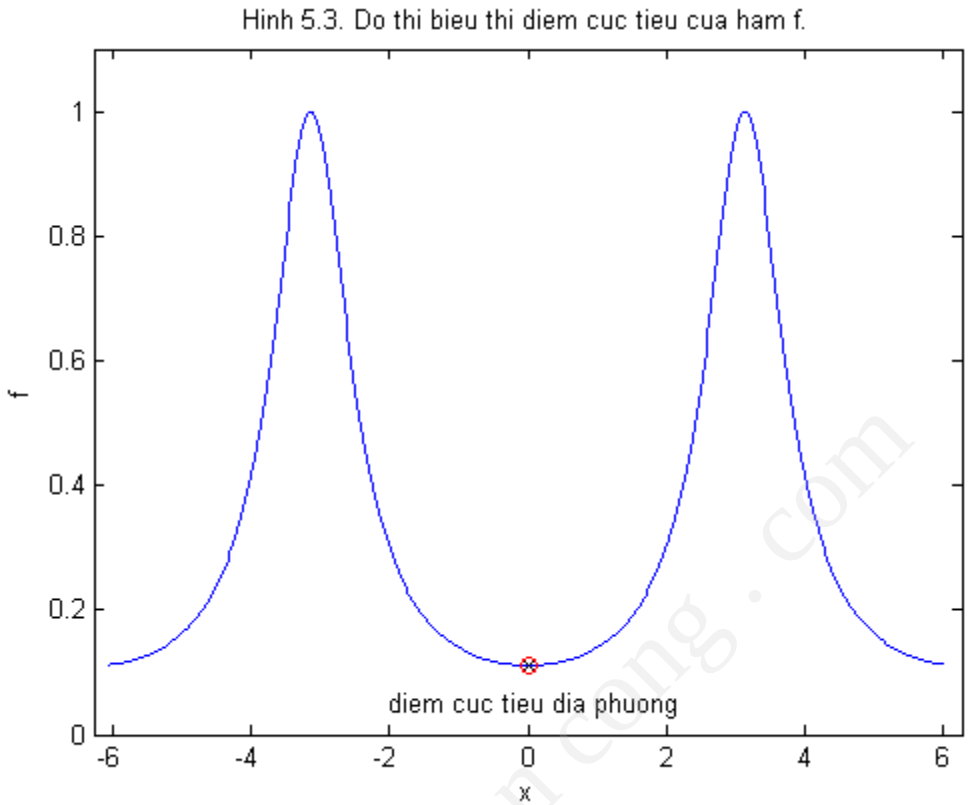
```
% Chuong trinh tao file diffb.m.
% Tieu de: Xac dinh cac diem cuc tieu cua ham f va ve do thi.
syms x
f=1/(5+4*cos(x));
f1=diff(f);                    % Dao ham cap 1 cua f.
```

```
f1=simple(f1);                  % Viet dang thu gon cua f1.
pretty(f1);
z=solve(f1);                    % Giai phuong trinh f1=0.
format;                         % Lay 5 chu so le.
xct=double(z)                % Chuyen ma tran, bieu thuc Symbolic ve dang so.
fct=subs(f,x,xct)               % Thay tat ca cac bien x trong f boi xct.
ezplot(f)
axis([-2*pi 2*pi 0 1.1])     % Xac dinh he toa do voi [xmin xmax ymin ymax].
ylabel('f')
title('Hinh 5.3. Do thi bieu thi cuc tieu cua ham f.')
hold on
%%%xct=[xct-2*pi xct xct+2*pi];        % Lay tat ca cac diem cuc tieu.
plot(xct,fct,'ro')
plot(xct,fct,'kx')              % Ve diem dau "x" mau den la diem cuc tieu.
text(-2,0.05,'diem cuc tieu dia phuong')  % Xuat diem cuc tieu dia phuong tai
                                          % toa do x=-2, y=0.05.
```

Kết quả chạy file diffb.m ta được lần lượt f1, điểm cực tiểu xct và giá trị cực tiểu fct và Hình 5.3 như sau

$$4\,\frac{\sin(x)}{(5+4\cos(x))^2}$$

```
xct =
     0

fct =
    0.1111
```




**Chương trình 5.4c:** (Các file chương trình có trong thư mục viphanso.)

```
% Chuong trinh tao file diffc.m.
% Tieu de: Tinh lien tiep hai lan nguyen ham cua f2 la k va so sanh voi f.
syms x
f=1/(5+4*cos(x));
f2=diff(f,2);          % Tinh dao ham cap hai cua f.
h=int(f2);             % Tinh nguyen ham lan thu nhat cua f2.
k=int(h);              % Tinh nguyen ham lan hai cua f2.
pretty(k)
%%% hoac k=int(int(f2)); % Tinh tich phan boi 2 cua f2.
d=f-k;
pretty(d)
simplify(d)            % Do lech giua k va f o dang toi gian.
ezplot(f,k)            % Ve do thi ham bieu thi do lech cua f va k.
title('Hinh 5.4. Do thi bieu thi do lech cua f va k')
```

Kết quả chạy file diffc.m ta được lần lượt k, d, độ lệch của f và k ở dạng tối giản và Hình 5.4.

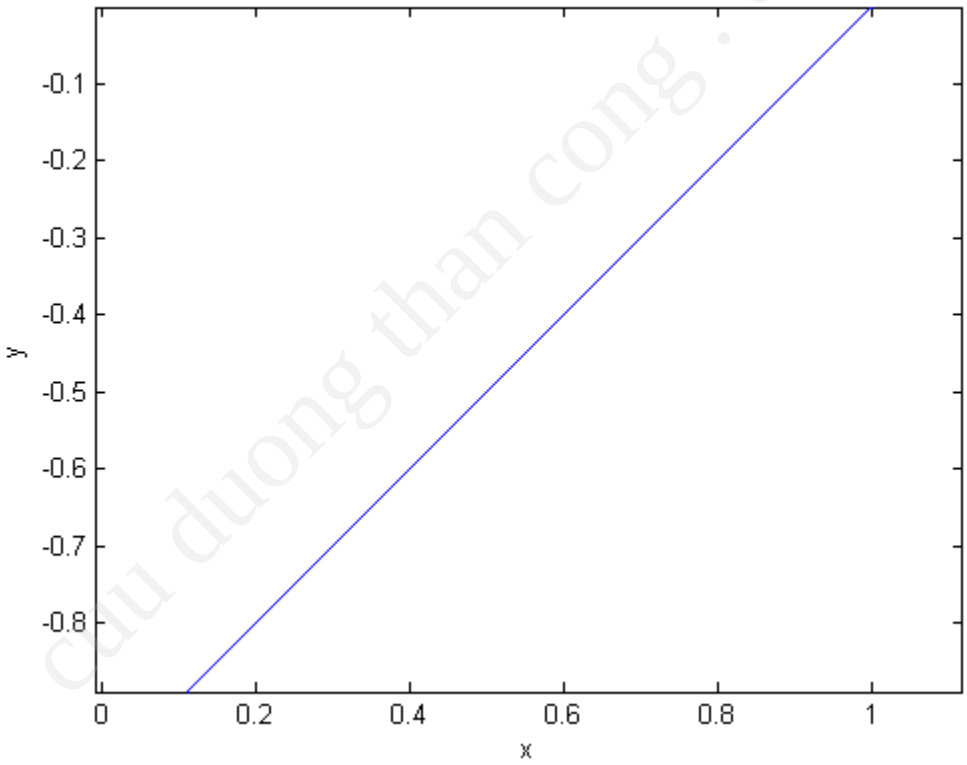
$$-\frac{}{\tan(1/2\,x)^2+9}$$

$$\frac{1}{5+4\cos(x)}+\frac{8}{\tan(1/2\,x)^2+9}$$

ans =

1

Hinh 5.4. Do thi bieu thi do lech cua f va k.






**Nhận xét**

Khi tính hai lần nguyên hàm của f'=f2 sẽ lệch nhau một hàm tuyến tính dạng $c_1x+c_2$. Do đó độ lệch ở dạng tối giản giữa f và k bằng 1 là phù hợp.

# CHƯƠNG 6
# GIẢI HỆ PHƯƠNG TRÌNH TUYẾN TÍNH

Trong chương này, chúng tôi trình bày một số phương pháp giải hệ phương trình đại số tuyến tính (hay là hệ phương trình tuyến tính)

$$Ax = b. \tag{6.1}$$

Với giả thiết rằng $\det A \neq 0$, tức là hệ (6.1) luôn có nghiệm duy nhất $x = A^{-1}b$. Tuy nhiên trong nhiều trường hợp việc tìm ma trận nghịch đảo $A^{-1}$ sẽ khó khăn hơn nhiều so với việc giải trực tiếp hệ phương trình đã cho. Chúng tôi sẽ trình bày một số phương pháp giải trực tiếp này, và ta chỉ xét hệ gồm n phương trình với n ẩn, tức là A là ma trận vuông cấp n; vector nghiệm x và vector tự do b là các vector cột n chiều.

## 6.1. PHƯƠNG PHÁP GAUSS

Đầu tiên ta xét các trường hợp đơn giản khi ma trận hệ số A của hệ (6.1) có dạng đặc biệt.

Trường hợp đơn giản nhất là A có dạng đường chéo

$$A = \begin{pmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & a_{nn} \end{pmatrix}.$$

Khi đó, nghiệm của hệ là $x_i = \dfrac{b_i}{a_{ii}},\ i = 1,2,...,n.$ (6.2)

Trường hợp tiếp theo là A có dạng tam giác trên

$$A=\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ 0 & a_{22} & \dots & a_{2n} \\ \dots & \dots & \ddots & \dots \\ 0 & 0 & \dots & a_{nn} \end{pmatrix}.$$

Khi đó, nghiệm của hệ là

$$\begin{cases} x_n = \dfrac{b_n}{a_{nn}}, \\ x_k = \dfrac{1}{a_{kk}}\left( b_{kk} - \sum\limits_{j=k+1}^{n} a_{kj}x_j \right), k = n-1,...,1. \end{cases} \tag{6.3}$$

Cuối cùng là ma trận hệ số A có dạng tam giác dưới

$$A=\begin{pmatrix} a_{11} & 0 & \dots & 0 \\ a_{21} & a_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}.$$

Khi đó, nghiệm của hệ là

$$\begin{cases} x_1 = \dfrac{b_1}{a_{11}}, \\ x_k = \dfrac{1}{a_{kk}}\left( b_k - \sum\limits_{j=1}^{k-1} a_{kj}x_j \right), k = 2,...,n. \end{cases} \tag{6.4}$$

Bây giờ chúng tôi sẽ trình bày phương pháp Gauss. Nội dung của phương pháp là sử dụng các phép biến đổi sơ cấp theo hàng để chuyển về một hệ phương trình tương đương mà ma trận hệ số có dạng tam giác. Các phép biến đổi sơ cấp thường hay sử dụng là

- Nhân một hàng cho một số khác không.
- Hoán chuyển hai hàng cho nhau.
- Cộng một hàng cho một hàng khác đã nhân với một số khác không.

Xét hệ

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases}$$

Do $\det A \neq 0$ nên một trong các số $a_{11}$, $a_{21}$, …, $a_{n1}$ phải khác không. Giả sử $a_{11} \neq 0$, lấy phương trình thứ $k$, $k = \overline{2,n}$ trừ cho phương trình một đã nhân với $m_{k1} = a_{k1} / a_{11}$, ta được hệ mới có dạng

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ \quad a_{22}^{(1)}x_2 + \ldots + a_{2n}^{(1)}x_n = b_2 \\ \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\ \quad a_{n2}^{(1)}x_2 + \ldots + a_{nn}^{(1)}x_n = b_n \end{cases}$$

Trong các số $a_{22}^{(1)}$, …, $a_{n2}^{(1)}$ phải có một số khác không vì $\det A \neq 0$. Giả sử $a_{22}^{(1)} \neq 0$, còn nếu có $a_{p2}^{(1)} \neq 0$ và $a_{22}^{(1)} = 0$ thì ta thực hiện phép hoán chuyển hai phương trình thứ p và thứ hai cho nhau; tiếp tục biến đổi như trên cho n – 2 phương trình cuối với hệ số nhân vào là $m_{k1} = a_{k2}^{(1)} / a_{22}^{(1)}$. Và tiếp tục tiến trình cho đến phương trình thứ n, ta được hệ

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ \quad a_{22}^{(1)}x_2 + \ldots + a_{2n}^{(1)}x_n = b_2 \\ \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\ \quad\quad\quad a_{nn}^{(n-1)}x_n = b_n \end{cases}$$

Đây là hệ mà ma trận hệ số có dạng tam giác trên và giải theo (6.3).

**Ví dụ 6.1.1** Giải hệ

$$\begin{cases} 4x_1 + 3x_2 + 2x_3 + x_4 = 1 \\ 3x_1 + 4x_2 + 3x_3 + 2x_4 = 1 \\ 2x_1 + 3x_2 + 4x_3 + 3x_4 = -1 \\ x_1 + 2x_2 + 3x_3 + 4x_4 = -1 \end{cases} \tag{6.5}$$

Qua một số phép biến đổi sơ cấp trên dòng theo phương pháp Gauss (các dòng biến đổi sẽ được thể hiện trong vector piv của chương trình 6.1)

$$\left(\begin{array}{cccc|c} 4 & 3 & 2 & 1 & 1 \\ 3 & 4 & 3 & 2 & 1 \\ 2 & 3 & 4 & 3 & -1 \\ 1 & 2 & 3 & 4 & -1 \end{array}\right) \xrightarrow[m_{41}=1/4]{\substack{m_{21}=3/4 \\ m_{31}=1/2}} \left(\begin{array}{cccc|c} 4 & 3 & 2 & 1 & 1 \\ 0 & 7/4 & 3/2 & 5/4 & 1/4 \\ 0 & 3/2 & 3 & 5/2 & -3/2 \\ 0 & 5/4 & 5/2 & 1/45 & -5/4 \end{array}\right) \xrightarrow[m_{42}=5/7]{m_{32}=6/7}$$

$$\left(\begin{array}{cccc|c} 4 & 3 & 2 & 1 & 1 \\ 0 & 7/4 & 3/2 & 5/4 & 1/4 \\ 0 & 0 & 12/7 & 10/7 & -12/7 \\ 0 & 0 & 10/7 & 20/7 & -5/4 \end{array}\right) \xrightarrow{m_{43}=5/6} \left(\begin{array}{cccc|c} 4 & 3 & 2 & 1 & 1 \\ 0 & 7/4 & 3/2 & 5/4 & 1/4 \\ 0 & 0 & 12/7 & 10/7 & -12/7 \\ 0 & 0 & 0 & 5/3 & 0 \end{array}\right)$$

Ta thu được nghiệm của hệ (6.5) là $x_4 = 0$, $x_3 = -1$, $x_2 = 1$, $x_1 = 0$.

Như đã trình bày ở trên, nếu $a_{p2}^{(1)} \neq 0$ và $a_{22}^{(1)} = 0$ thì ta thực hiện phép hoán chuyển hai phương trình thứ p và thứ hai cho nhau. Để tránh trường hợp này, ta có thể cải tiến phương pháp Gauss theo hướng như sau. Tại mỗi bước, khi chọn phần tử để biến đổi, ta sẽ chọn phần tử có trị tuyệt đối lớn nhất, sao cho không trùng hàng và cột với những phần tử đã chọn trước đó. Phần tử như vậy thường được gọi là phần tử chính hay phần tử trội. Sau đó ta sẽ thực hiện biến đổi cho các phần tử trên cùng cột với phần tử trội bằng không. Qua n bước như vậy ta sẽ thu được nghiệm của hệ đã cho. Đây gọi là phương pháp Gauss với phép trục từng phần (Partial Pivoting**).**

**CHƯƠNG TRÌNH MATLAB:** Chúng tôi trình bày chương trình MATLAB dùng phương pháp Gauss để giải hệ (6.1) với phép trục từng phần. Để chạy chương trình ta cần phải nhập vào ma trận hệ số A và vector b. Kết quả trả về là x là vector nghiệm; lu là một ma trận mà có chứa phần tử của ma trận tam giác trên và ma trận tam giác dưới, ma trận tam giác trên là kết quả từ phép khử Gauss với phép trục từng phần, và các số nhân được dùng trong phép khử là các phần tử của ma trận tam giác dưới; piv là vector ghi lại phép hoán vị được thực hiện trong phép trục từng phần. Chương trình 6.1 được gọi theo cú pháp

[x,lu,piv] = GEpivot(A,b).

**Chương trình 6.1:**

```
function [x,lu,piv] = GEpivot(A,b)
% Day la ham [x,lu,piv] = GEpivot(A,b), su dung phuong phap khu Gauss   % voi phep truc tung phan de giai he phuong trinh tuyen tinh Ax=b.
% Tri dau vao
%    A: he so ma tran vuong
%    b: vecto cot phai
% Giai vi du 6.1.1 voi: A=[4 3 2 1;3 4 3 2;2 3 4 3;1 2 3 4]; b=[1 1 -1 -1]
% Hay chay thu voi cac tri khac de test chuong trinh
%A=[1 1 0 3;2 1 -1 1;3 -1 -1 2;-1 2 3 -1];b=[4 1 -3 4];
%A=[1 -1 2 -1;2 -2 3 -3;1 1 1 0;1 -1 4 3];b=[-8 -20 -2 4];
%A=[2 1 -1 2;  4 5 -3 6; -2 5 -2 6; 4 11 -4 8]; b=[5 9 4 2];
% Ket qua dau ra
%    x la vecto nghiem
%   lu - la ma tran tich hai ma tran tam giac tren U, va tam giac duoi L ghi lai
%   ket qua, va cac he so nhan dung trong cac phep bien doi.
%    piv la pivoting vector ghi nhan phep hoan vi trong phep bien doi.

% Kiem tra thu tu ma tran va kich thuoc hang , cot cua ma tran A.
[m,n] = size(A);
if m ~= n
  error('Ma tran khong phai ma tran vuong.')
end
m = length(b);
if m ~= n
  error('ma tran va vec to khong co kich thuoc phu hop.')
```

```
end
% Khoi tao ban dau cho vector pivoting.
piv = (1:n)';
% Cac buoc khu he so
for k = 1:n-1
%Tim yeu to lon nhat trong cot pivot, ben duoi vi tri cua pivot cung cac chi % so toi da cua cac thanh phan do.
    [col_max  index] = max(abs(A(k:n,k)));
  index = index + k-1;
    if index ~= k
% Chuyen sang dong thu k cung chi so, trong cot k den n. Tuong tu cho cot b
    tempA = A(k,k:n);
    A(k,k:n) = A(index,k:n);
    A(index,k:n) = tempA;
    tempb = b(k);
    b(k) = b(index);
    b(index) = tempb;
    temp = piv(k);
    piv(k) = piv(index);
    piv(index) = temp;
  end
% Dinh dang sau do luu lai thanh cac phan tu trong cot pivot, ben duoi  %duong cheo chinh.
    A(k+1:n,k) = A(k+1:n,k)/A(k,k);

% Tien hanh cac buoc khu, dau tien sua doi ma tran, va sau do bo sung cho b
   for i = k+1:n
    A(i,k+1:n) = A(i,k+1:n) - A(i,k)*A(k,k+1:n);
  end
  b(k+1:n) = b(k+1:n) - A(k+1:n,k)*b(k);
end
% Thiet lap ma tran tam giac tren cua he tuyen tinh.
x = zeros(n,1);
x(n) = b(n)/A(n,n);
for i = n-1:-1:1
  x(i)=(b(i)-A(i,i+1:n)*x(i+1:n))/A(i,i);
end
% Ghi nhan lai ma tran phan tich LU voi viec bien doi dong.
lu = A;
```

## 6.2. PHƯƠNG PHÁP NHÂN TỬ LU

Nội dung của phương pháp này là phân tích ma trận $A$ thành tích của hai ma trận, ma trận tam giác dưới $L$ và ma trận tam giác trên $U$. Khi đó việc giải hệ (6.1) sẽ đưa về việc giải hai hệ

$$Lg = b \text{ và } Ux = g. \tag{6.6}$$

**Định lý 6.2.1** *Nếu $A$ là một ma trận không suy biến, thì bao giờ cũng tồn tại một ma trận $P$ không suy biến sao cho ma trận $PA$ phân tích được thành hai ma trận, ma trận tam giác dưới $L$ và ma trận tam giác trên $U$, tức là*

$$PA = LU. \tag{6.7}$$

Có nhiều phương pháp phân tích $A = LU$, tuy nhiên ta thường xét trường hợp ma trận $L$ có các phần tử trên đường chéo chính bằng 1 và gọi đây là phân rã Doolittle. Khi đó $L$ và $U$ có dạng

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots \\ l_{n1} & l_{n2} & \dots & 1 \end{pmatrix} \text{ và } U = \begin{pmatrix} u_{11} & u_{12} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & u_{nn} \end{pmatrix}. \tag{6.8}$$

Với,

$$\begin{cases} u_{1j} = a_{1j} & (1 \le j \le n), \\ l_{i1} = \dfrac{a_{i1}}{u_{11}} & (2 \le i \le n), \\ u_{ij} = a_{ij} - \sum\limits_{k=1}^{i-1} l_{ik} u_{kj} & (1 \le i \le j), \\ l_{ij} = \dfrac{1}{u_{jj}} \left( a_{ij} - \sum\limits_{k=1}^{j-1} l_{ik} u_{kj} \right) & (1 < j < i). \end{cases} \tag{6.9}$$

Nghiệm của hệ $Lg = b$ xác định theo công thức (6.4) và nghiệm của hệ $Ux = g$ xác định theo công thức (6.3).

**Ví dụ 6.2.2** Đối với hệ (6.5), ta có

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 3/4 & 1 & 0 & 0 \\ 1/2 & 6/7 & 1 & 0 \\ 1/4 & 5/7 & 5/6 & 1 \end{pmatrix} \text{ và } U = \begin{pmatrix} 4 & 3 & 2 & 1 \\ 0 & 7/4 & 3/2 & 5/4 \\ 0 & 0 & 12/7 & 10/7 \\ 0 & 0 & 0 & 5/3 \end{pmatrix}.$$

Mà $b = (1, 1, -1, -1)^T$, nên nghiệm của $Lg = b$ là $g = (1, 1/4, -12/7, 0)^T$, dẫn đến hệ $Ux = g$ có nghiệm là $x = (0, 1, -1, 0)^T$.

**CHƯƠNG TRÌNH MATLAB:** Chúng tôi trình bày chương trình 6.2 để giải hệ (6.1) bằng phương pháp nhân tử LU. Đối số của chương trình là ma trận hệ số A và vector cột b. Kết quả trả về là các ma trận l và u theo phân tích (6.8), các vector nghiệm g và x của (6.6) và x cũng chính là nghiệm cần tìm của hệ (6.1). Chương trình được gọi theo cú pháp

[l,u,g,x] = LUfactor(A,b).

**Chương trình 6.2:**

```
function [l,u,g,x] = LUfactor(A,b)
% Day la ham [l,u,g,x] = LUfactor(A,b), dung de giai he (6.1) theo pp LU.
%Giai vi du 6.2.2 theo cu phap
%       [l,u,g,x] = LUfactor([4 3 2 1;3 4 3 2;2 3 4 3;1 2 3 4],[1 1 -1 -1])
% Kiem tra cac doi so nhap vao khi goi chuong trinh
if nargin < 2,    error('Cho ma tran he so'); end;
% Phan tich lu theo cong thuc (6.9)
N=length(b);
l=zeros(N); u=zeros(N);
for i=1:N, l(i,i)=1; end;
for j=1:N
   u(1,j)=A(1,j);
```

```
end;
if u(1,1)==0
   error('Khong the phan tich duoc.');
end;
for i=2:N
   l(i,1)=A(i,1)/u(1,1);
end;
for i=2:N-1
   for j=i:N
      sum=0;
      for k=1:i-1
         sum=sum+l(i,k)*u(k,j);
      end;
      u(i,j)=A(i,j)-sum;
   end;
   if u(i,i)==0
      error('Khong the phan tich duoc.');
   end;
   for j=i+1:N
      sum=0;
      for k=1:i-1
         sum=sum+l(j,k)*u(k,i);
      end;
      l(j,i)=(A(j,i)-sum)/u(i,i);
   end;
end;
sum=0;
for k=1:N-1
   sum=sum+l(N,k)*u(k,N);
end;
u(N,N)=A(N,N)-sum;
% Giai he lg=b theo dang ma tran he so tam giac duoi, cong thuc (6.4).
g = zeros(N,1);
g(1) = b(1)/l(1,1);
for k=2:N
   sum = 0;
   for j=1:k-1
      sum=sum+l(k,j)*g(j);
   end;
   g(k)=(b(k)-sum)/l(k,k);
end;
% Giai he ux=g theo dang ma tran he so tam giac tren, cong thuc (6.3)
```

```
x = zeros(N,1);
x(N) = x(N)/u(N,N);
for k=N-1:-1:1
   sum = 0;
   for j=k+1:N
      sum=sum+u(k,j)*x(j);
   end;
   x(k)=(g(k)-sum)/u(k,k);
end;
```

- Phương pháp nhân tử LU áp dụng rất hiệu quả cho trường hợp ma trận hệ số A có dạng ba đường chéo.

Để tránh trùng lắp trong ký hiệu, ta viết lại hệ (6.1) trong trường hợp này là $Ax = d$ (chương trình 6.3 giải hệ theo cách viết này), với

$$A = \begin{pmatrix} a_{11} & a_{12} & 0 & \dots & 0 & 0 \\ a_{21} & a_{22} & a_{23} & \dots & 0 & 0 \\ 0 & a_{32} & a_{33} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & a_{n-1,n-1} & a_{n-1,n} \\ 0 & 0 & 0 & \dots & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

Khi đó phân rã Doolittle cho ta

$$L = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ l_{21} & 1 & 0 & \dots & 0 \\ 0 & l_{32} & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} \text{ và } U = \begin{pmatrix} u_{11} & u_{12} & 0 & \dots & 0 \\ 0 & u_{22} & u_{23} & \dots & 0 \\ 0 & 0 & u_{33} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & u_{n,n} \end{pmatrix}.$$

Khi đó, từ công thức (6.9) ta có

$$\begin{cases} u_{11} = a_{11};\ u_{12} = a_{12};\ l_{21} = a_{21} / u_{11}; \\ u_{i,i} = a_{i,i} - l_{i,i-1} u_{i-1,i}, \qquad i = 2,3,\ldots,n; \\ u_{i,i+1} = a_{i,i+1};\ l_{i+1,i} = a_{i+1,i} / u_{i,i}, \quad i = 2,3,\ldots,n-1. \end{cases} \tag{6.10}$$

Và theo phân tích LU thì ta cần giải hai hệ $Lg = d$ và $Ux = g$, theo công thức (6.4) và (6.3). Cụ thể là

- Hệ $Lg = d$ thì $g_1 = d_1,\ g_j = d_j - l_{j,j-1}.g_{j-1},\quad j = 2,...,n$.
- Hệ $Ux = g$ thì $x_n = g_n / u_{n,n};\ \ x_j = \dfrac{g_j - b_j x_{j+1}}{u_{jj}},\quad j = n-1,...,1.$

**Ví dụ 6..2.3** Giải hệ $Ax = d$, với

$$A = \begin{pmatrix} 2 & 1 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 1 & 2 \end{pmatrix} \text{ và } d = [5,\ 4,\ 3,\ 2,\ 1]^T.$$

Áp dụng (6.10), (6.4), (6.3) ta tính được

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1/2 & 1 & 0 & 0 & 0 \\ 0 & 2/3 & 1 & 0 & 0 \\ 0 & 0 & 3/4 & 1 & 0 \\ 0 & 0 & 0 & 4/5 & 1 \end{pmatrix} \text{ và } U = \begin{pmatrix} 2 & 1 & 0 & 0 & 0 \\ 0 & 3/2 & 1 & 0 & 0 \\ 0 & 0 & 4/3 & 1 & 0 \\ 0 & 0 & 0 & 5/4 & 1 \\ 0 & 0 & 0 & 0 & 6/5 \end{pmatrix}.$$

Nghiệm của $Lg = d$ là $g = \left(5,\ 3/2,\ 2,\ 1/2,\ 3/5\right)^T$.

Hệ $Ux = g$ có nghiệm là $x = \left(5/2,\ 0,\ 3/2,\ 1/2,0\right)^T$.

Tiếp theo, **c**húng tôi trình bày chương trình 6.2a để giải hệ (6.1) khi nó có dạng hệ thống ba đường chéo. Trong trường hợp đặc biệt này, thay vì đưa vào ma trận hệ số A, ta sẽ đưa vào ba vector, a chứa các phần tử trên đường chéo chính

của A, b là đường chéo nằm trên đường chéo chính và c là đường chéo nằm dưới đường chéo chính; vector cột tự do là d; N là số phương trình và cũng là số ẩn của hệ. Kết quả trả về là các vector, l chứa các phần tử của đường chéo dưới đường chéo chính của $L$; u và v chứa các phần tử của đường chéo chính và đường chéo trên đường chéo chính của $U$; g là nghiệm của $Lg = d$ và x là nghiệm của $Ux = g$ và cũng chính là nghiệm của hệ (6.1). Chương trình được gọi theo cú pháp

[l,u,v,g,x]=tridiag(a,b,c,d,N).

**Chương trình 6.2a:**

```
function [l,u,v,g,x]=tridiag(a,b,c,d,N)
% Ham dung de giai he (6.1) dang ba duong cheo.
%Giai vi du 6.2.3, [l,u,v,g,x]=tridiag([2 2 2 2 2],[1 1 1 1],[1 1 1 1],[5 4 3 2 1],5)
if nargin < 5, error('Ham co toi thieu 5 doi so.'); end;
                                    % Kiem tra cac doi so khi goi chuong trinh.
% Phan tich lu.
u(1)=a(1);v(1)=b(1);l(1)=c(1)/u(1);
g(1)=d(1);
for i=2:N-1
   u(i)=a(i)-l(i-1)*v(i-1);
   v(i)=b(i);
   l(i)=c(i)/u(i);
   g(i)=d(i)-l(i-1)*g(i-1); end;
% Giai he lg=d va ux=g.
u(N)=a(N)-l(N-1)*v(N-1);
g(N)=d(N)-l(N-1)*g(N-1);
x(N)=g(N)/u(N);
for i=N-1:-1:1
   x(i)=(g(i)-v(i)*x(i+1))/u(i); end;
```

## 6.3. PHƯƠNG PHÁP LẶP

Trước tiên, chúng tôi trình bày phần lý thuyết về chuẩn vector và chuẩn ma trận.

Trong không gian tuyến tính $R^n$. Chuẩn của vector $x \in R^n$ là một số thực, ký hiệu là $\|x\|$, thoả các điều kiện sau

i. $\forall x \in R^n, \|x\| \geq 0, \|x\| = 0 \Leftrightarrow x = 0$.

ii. $\forall x \in R^n, \forall \lambda \in R, \|\lambda x\| = \lambda \|x\|$.

iii. $\forall x, y \in R^n, \|x + y\| = \|x\| + \|y\|$.

Trong $R^n$ có rất nhiều chuẩn, ở đây chúng tôi chỉ nêu hai chuẩn sau

$$\|x\|_1 = |x_1| + |x_2| + \ldots + |x_n| = \sum_{i=1}^{n} |x_i|. \tag{6.11}$$

$$\|x\|_\infty = max\left(|x_1|, |x_2|, \ldots, |x_n|\right) = \max_{1 \leq i \leq n} |x_i|. \tag{6.12}$$

**Định nghĩa 6.3.1** Chuẩn ma trận tương ứng với chuẩn vector được xác định theo công thức

$$\|A\| = \max_{\|x\|=1} \|Ax\| = \max_{\|x\| \neq 0} \frac{\|Ax\|}{\|x\|}. \tag{6.13}$$

**Ví dụ 6.3.2** Xác định chuẩn của ma trận $A = \begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix}$ tương ứng với chuẩn của một vector. Với mọi $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in R^2$, sao cho: $\|x\|_1 = |x_1| + |x_2| = 1$. Ta có:

$$\|Ax\|_1 = |4x_1 + 3x_2| + |2x_1 + x_2| \leq 6|x_1| + 4|x_2| = 2|x_1| + 4 \leq 6 \rightarrow \|A\|_1 = 6.$$

Từ (6.13) ta có: $\|Ax\| \leq \|A\|.\|x\|$.

**Định lý 6.3.3** *Chuẩn của ma trận theo công thức (6.13), tương ứng với chuẩn vector được xác định như sau:*

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{n} |a_{ij}|. \tag{6.14}$$

$$\|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij}|. \tag{6.15}$$

**Định nghĩa 6.3.4** Xét dãy các vector $\left\{x^{(m)}\right\}_{m=0}^{\infty}$ với $x^{(m)} \in R^n$. Ta nói dãy các vector này hội tụ về vector $x$ khi $m \to +\infty$ nếu và chỉ nếu $\left\|x^{(m)} - x\right\| \to 0$, khi $m \to +\infty$ (hội tụ theo chuẩn). Ta viết: $\lim\limits_{m\to\infty} x^{(m)} = x$. Ta cũng có thể nói dãy các vector $\left\{x^{(m)}\right\}$ hội tụ về vector $x$ theo chuẩn đã cho.

**Định lý 6.3.5** *Dãy các vector* $\left\{x^{(m)}\right\}$ *hội tụ theo chuẩn về vector* $x$ *khi* $m \to +\infty$ *khi và chỉ khi* $\left\{x_k^{(m)}\right\}$ *hội tụ về* $x_k$, $k = \overline{1,n}$ *(hội tụ theo toạ độ).*

Xét hệ (6.1), $Ax = b$ có nghiệm $x = A^{-1}b$. Cho $b$ một số gia $\Delta b$, khi đó nghiệm $x$ tương ứng sẽ có số gia $\Delta x$, và $A\Delta x = \Delta b \Leftrightarrow \Delta x = A^{-1}\Delta b$. Vậy,

$$\left\|\Delta x\right\| = \left\|A^{-1}\Delta b\right\| \le \left\|A^{-1}\right\|.\left\|\Delta b\right\| \text{ và } \left\|b\right\| = \left\|Ax\right\| \le \left\|A\right\|.\left\|x\right\|.$$

Suy ra,
$$\frac{\left\|\Delta x\right\|}{\left\|x\right\|} \le \left\|A\right\|\left\|A^{-1}\right\|\frac{\left\|\Delta b\right\|}{\left\|b\right\|}. \qquad (6.16)$$

$$\text{Số } k(A) = Cond(A) = \left\|A\right\|\left\|A^{-1}\right\|,\ 1 \le k(A) \le +\infty. \qquad (6.17)$$

là số điều kiện của ma trận $A$. Số điều kiện của ma trận đặc trưng cho tính ổn định của hệ (6.1). Giá trị $k(A)$ càng gần với số 1 thì hệ càng ổn định. Số điều kiện càng lớn thì hệ càng mất ổn định.

Trong các chương trình MATLAB của mục 6.4, chúng tôi dùng số điều kiện này để kiểm tra bước lặp có thể chấp nhận được để dừng việc tính toán và xuất kết quả cuối cùng.

**Ví dụ 6.3.6** Xét các hệ

$$Ax = b \text{ với } A = \begin{pmatrix} 1 & 2 \\ 1 & 2.01 \end{pmatrix}, b = \begin{pmatrix} 3 \\ 3.01 \end{pmatrix} \text{ có nghiệm là: } x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

$$Ax = b, \text{ với } b = \begin{pmatrix} 3 \\ 3.1 \end{pmatrix} \text{ có nghiệm là: } x = \begin{pmatrix} -17 \\ 10 \end{pmatrix}.$$

Ta có $k_\infty(A) = 1207.01 \geq 1$. Do đó, $b \approx b$, nhưng $x$, $x$ khác nhau rất xa.

Bây giờ chúng ta đi giải hệ (6.1) theo phương pháp lặp, trước hết ta chuyển hệ về dạng tương đương $x = Tx + c$, với $T$ là ma trận vuông cấp n và $c$ là một vector đã biết. Xuất phát từ vector ban đầu $x^{(0)}$, ta xây dựng một dãy các vector $\left\{ x^{(m)} \right\}_{m=0}^{\infty}$ theo công thức lặp

$$x^{(m)} = Tx^{(m-1)} + c, \quad m = 1, 2, 3, \ldots \tag{6.18}$$

Ta có định lý sau

**Định lý 6.3.7** *Nếu* $\|T\| < 1$ *thì dãy lặp các vector xác định theo công thức (6.18) sẽ hội tụ về nghiệm* $x^*$ *hệ với mọi vector lặp ban đầu* $x^{(0)}$. *Khi đó ta có các công thức đánh giá sai số như sau*

$$\left\| x^{(m)} - x^* \right\| \leq \frac{\|T\|^m}{1 - \|T\|} \left\| x^{(1)} - x^{(0)} \right\|, \tag{6.19}$$

$$\left\| x^{(m)} - x^* \right\| \leq \frac{\|T\|^m}{1 - \|T\|} \left\| x^{(m)} - x^{(m-1)} \right\|. \tag{6.20}$$

Bây giờ chúng ta sẽ xét một dạng ma trận hệ số của hệ (6.1) mà có thể chuyển về dạng $x = Tx + c$.

**Định nghĩa 6.3.8** Ma trận $A$ được gọi là ma trận đường chéo trội nghiêm ngặt nếu nó thoả mãn điều kiện sau đây

$$\sum_{j=1,\, j\neq i}^{n} \left|a_{ij}\right| < \left|a_{ii}\right|. \tag{6.21}$$

Khi đó $\det A \neq 0$ và $a_{ii} \neq 0,\ \forall i = \overline{1,n}$. Xét hệ phương trình $Ax = b$ với $A$ là ma trận đường chéo trội nghiêm ngặt. Khi đó ta phân tích ma trận $A$ thành

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{2n} & \dots & a_{nn} \end{pmatrix}$$

$$= \begin{pmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & a_{nn} \end{pmatrix} - \begin{pmatrix} 0 & 0 & \dots & 0 \\ -a_{21} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ -a_{n1} & -a_{n2} & \dots & 0 \end{pmatrix} - \begin{pmatrix} 0 & -a_{12} & \dots & -a_{1n} \\ 0 & 0 & \dots & -a_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 \end{pmatrix}$$

$$= D - L - U$$

Chú ý rằng do $a_{ii} \neq 0,\ \forall i = \overline{1,n}$ nên $\det D \neq 0$. Và như vậy tồn tại ma trận nghịch đảo:

$$D^{-1} = \begin{pmatrix} 1/a_{11} & 0 & \dots & 0 \\ 0 & 1/a_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1/a_{nn} \end{pmatrix}.$$

Khi đó, $$Ax = b \Leftrightarrow \left(D - L - U\right)x = b. \tag{6.22}$$

Tiếp theo chúng tôi sẽ trình bày một vài phương pháp để chuyển phương trình (6.1) về dạng $x = Tx + c$. Từ hệ (6.22) ta có

$$Dx = \left(L + U\right)x + b \Leftrightarrow x = D^{-1}\left(L + U\right)x + D^{-1}b.$$

Ký hiệu: $T_j = D^{-1}\left(L+U\right)$ và $c_j = D^{-1}b$. Khi đó theo (6.18) thì

$$x^{(m)} = T_j x^{(m-1)} + c_j, \quad m = 1, 2, 3, \ldots \tag{6.23}$$

Phương pháp lặp dựa trên công thức lặp (6.23) được gọi là ***phương pháp Jacobi*** . Dạng tường minh của công thức (6.23) là

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left( -\sum_{j=1}^{i-1} a_{ij}x^{(k)} - \sum_{j=i+1}^{n} a_{ij}x^{(k)} + b_i \right), \quad i = \overline{1, n}. \tag{6.24}$$

Do $A$ là ma trận đường chéo trội nghiêm ngặt, nên

$$\left\|T_j\right\|_\infty = \left\|D^{-1}\left(L+U\right)\right\|_\infty = \max_{1\le i\le n} \sum_{j=1, j\ne i}^{n} \left|\frac{a_{ij}}{a_{ii}}\right| = \max_{1\le i\le n} \frac{\sum_{j=1, j\ne i}^{n} \left|a_{ij}\right|}{\left|a_{ii}\right|} < 1$$

Điều này có nghĩa là phương pháp Jacobi luôn hội tụ với mọi vector lặp ban đầu $x^{(0)}$.

**Ví dụ 6.3.9** Xét hệ phương trình

$$\begin{cases} 10x_1 - x_2 + 2x_3 = 6 \\ -x_1 + 11x_2 - x_3 + 3x_4 = 25 \\ 2x_1 - x_2 + 10x_3 - x_4 = -11 \\ 3x_2 - x_3 + 8x_4 = 15 \end{cases}$$

Có nghiệm duy nhất là $x^* = \left(1, 2, -1, 1\right)^T$. Theo (6.24), để chuyển từ hệ $Ax = b$ về dạng $x = T_j x + c_j$ ta có

$$\begin{cases} x_1 = \dfrac{1}{10}\left(b_1 + x_2 - 2x_3\right) \\ x_2 = \dfrac{1}{11}\left(b_2 + x_1 + x_3 - 3x_4\right) \\ x_3 = \dfrac{1}{10}\left(b_3 - 2x_1 + x_2 + x_4\right) \\ x_4 = \dfrac{1}{8}\left(b_4 - 3x_2 + x_3\right) \end{cases}, \quad b = \begin{pmatrix} 6 \\ 25 \\ -11 \\ 15 \end{pmatrix}.$$

Hay,

$$\begin{cases} x_1 = 0.100x_2 - 0.200x_3 + 0.600 \\ x_2 = 0.091x_1 + 0.091x_3 - 0.273x_4 + 2.273 \\ x_3 = 0.200x_1 + 0.100x_2 + 0.100x_4 - 1.100 \\ x_4 = -0.357x_2 + 0.125x_3 + 1.875 \end{cases}$$

Ta có, $T_j = \begin{pmatrix} 0 & 0.100 & -0.200 & 0 \\ 0.091 & 0 & 0.091 & -0.273 \\ -0.200 & 0.100 & 0 & 0.100 \\ 0 & -0.375 & 0.125 & 0 \end{pmatrix}$ và $c_j = \begin{pmatrix} 0.600 \\ 2.273 \\ -1.100 \\ 1.875 \end{pmatrix}$.

Công thức lặp theo phương pháp Jacobi có dạng:

$$\begin{cases} x_1^{(k+1)} = 0.100x_2^{(k)} - 0.200x_3^{(k)} + 0.600 \\ x_2^{(k+1)} = 0.091x_1^{(k)} + 0.091x_3^{(k)} - 0.273x_4^{(k)} + 2.273 \\ x_3^{(k+1)} = 0.200x_1^{(k)} + 0.100x_2^{(k)} + 0.100x_4^{(k)} - 1.100 \\ x_4^{(k)} = -0.357x_2^{(k)} + 0.125x_3^{(k)} + 1.875 \end{cases}$$

Chọn chuẩn vô cùng và ta có $\|T_j\|_\infty = 1/2$, tức là phương pháp hội tụ, và chọn vector $x^{(0)} = (0,0,0,0)^T$. Bảng 6.1 cho kết quả tính toán bằng phương pháp Jacobi đối với ví dụ 6.3.9 của 5 lần lặp đầu và lần lặp thứ 10, 12.

Sai số thực sự là $\left\|x^{(10)} - x^*\right\|_\infty = 2.0 \times 10^{-4}$.

**Bảng 6.1.** Vài kết quả giải ví dụ 6.3.9 bằng phương pháp Jacobi.

| $k$ | 1 | 2 | 3 | 4 | 5 | 10 | 12 |
|---|---|---|---|---|---|---|---|
| $x_1^{(k)}$ | 0.6000 | 1.0473 | 0.9326 | 1.0152 | 0.9890 | 1.0001 | 1.0000 |
| $x_2^{(k)}$ | 2.2727 | 1.7159 | 2.0533 | 1.9537 | 2.0114 | 1.9998 | 2.0000 |
| $x_3^{(k)}$ | -1.1000 | -0.8052 | -1.0493 | -0.9681 | -1.0103 | -0.9998 | -1.0000 |

Ta có đánh giá $\left\|x^{(10)}-x^*\right\|_\infty \le \frac{1/2}{1-1/2}\left\|x^{(10)}-x^{(9)}\right\|_\infty = 8.0\times 10^{-4} < 10^{-3}$.

Trong trường hợp tổng quát và trong chương trình MATLAB chúng tôi đánh giá theo công thức (6.16), tức là nếu cho trước phạm vi sai số là $\delta$ thì quá trình lặp sẽ dừng lại khi

$$\frac{\left\|x^{(k+1)}-x^{(k)}\right\|_\infty}{\left\|x^{(k+1)}\right\|_\infty} < \delta .$$

Như đã trình bày ở chương 1, để xử lý trường hợp mẫu số bằng 0, trong MATLAB chúng tôi dùng

$$\frac{\left\|x^{(k+1)}-x^{(k)}\right\|_\infty}{\left\|x^{(k+1)}\right\|_\infty + eps} < \delta .$$

Trở lại công thức (6.24), để tính các toạ độ của vector lặp $x^{(k+1)}$, chúng ta sử dụng toạ độ của $x^{(k)}$. Nhưng với $i>1$, thì $x_i^{(k+1)}$ đã được tính và xấp xỉ nghiệm chính xác $x_i^*$ sẽ tốt hơn $x_i^{(k)}$. Do đó, để tính $x_i^{(k+1)}$ ta nên sử dụng lại các giá trị vừa tính xong $x_1^{(k+1)}$, $x_2^{(k+1)}$,..., $x_{i-1}^{(k+1)}$. Ta có

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(-\sum_{j=1}^{i-1} a_{ij}x^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x^{(k)} + b_i\right),\ i=\overline{1,n}. \qquad (6.25)$$

Phương pháp lặp dựa trên công thức lặp (6.25) được gọi là ***phương pháp Gauss – Seidel.*** Trong ví dụ 6.3.8, công thức lặp ***Gauss – Seidel*** có dạng

$$\begin{cases} x_1^{(k+1)} = & 0.100x_2^{(k)} - 0.200x_3^{(k)} + 0.600 \\ x_2^{(k+1)} = 0.091x_1^{(k)} & + 0.091x_3^{(k)} - 0.273x_4^{(k)} + 2.273 \\ x_3^{(k+1)} = 0.200x_1^{(k)} & + 0.100x_2^{(k)} + 0.100x_4^{(k)} - 1.100 \\ x_4^{(k)} = & - 0.357x_2^{(k)} + 0.125x_3^{(k)} + 1.875 \end{cases}$$

Bảng 6.2 cho kết quả tính toán bằng phương pháp Gauss – Seidel đối với ví dụ 6.3.9 của 7 lần lặp đầu.

**Bảng 6.2.**Vài kết quả giải ví dụ 6.3.9 bằng phương pháp Gauss – Seidel.

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $x_1^{(k)}$ | 0.6000 | 1.0302 | 1.0066 | 1.0009 | 1.0001 | 1.0001 | 1.0000 |
| $x_2^{(k)}$ | 2.3273 | 2.0369 | 2.0036 | 2.0003 | 2.0000 | 1.9998 | 2.0000 |
| $x_3^{(k)}$ | -0.9873 | -1.0145 | -1.0025 | -1.0003 | -1.0000 | -0.9998 | -1.0000 |
| $x_4^{(k)}$ | 0.8789 | 0.9843 | 0.9984 | 0.9998 | 1.0000 | 0.9998 | 1.0000 |

So sánh với Bảng 6.1, ta thấy đến lần lặp thứ 5, nghiệm thu được bằng phương pháp Gauss – Seidel tốt hơn nhiều so với phương pháp Jacobi. Hơn thế nữa, đến lần lặp thứ 6 thì phương pháp Gauss – Seidel đã cho nghiệm chính xác, còn phương pháp Jacobi thì đến lần lặp thứ 12.

**CHƯƠNG TRÌNH MATLAB:** Chúng tôi trình bày hai chương trình MATLAB. Chương trình 6.3, cho phương pháp Jacobi và chương trình 6.3a, cho phương pháp Gauss-Seidel để giải hệ phương trình tuyến tính (6.1). Các đối số và kết quả đầu ra của chương trình được giải thích cụ thể trong chương trình. Chương trình sẽ dừng lại khi

$$\frac{\left\|x^{(k+1)}-x^{(k)}\right\|_\infty}{\left\|x^{(k+1)}\right\|_\infty+eps}<\delta,\ \delta\text{-sai số cho phép.}$$

**Chương trình 6.3:**

**function [x, iflag, itnum] = Jacobi(A,b,x0,delta,max_it)**
% function [x, iflag, itnum] = Jacobi(A,b,x0,delta,max_it)
%Chuong trinh thuc hien phep lap Jacobi de giai he phuong trinh tuyen tinh
%Ax=b
%
% % Gia tri dau vao
% A: cac he so ma tran vuong A
% b: Vecto cot phai b
% x0: vector lap ban dau,
% delta: uoc luong sai so cho phep giua cac lan lap lien tiep.
% max_it: so lan lap toi da cho phep trong gioi han sai so tren

% Khi ta da nhap cac tri dau vao, chuong trinh co the goi theo cu phap
% [x, iflag, itnum] = Jacobi
% Giai vi du 6.3.9.
**A=[10,-1,2,0;-1,11,-1,3;2,-1,10,-1;0,3,-1,8];**
**b=[6;25;-11;15];**
**max_it=12;**
**x0=[0;0;0;0];**
**delta=0.0001;**

% Tri dau ra
% x: vecto nghiem
% iflag: 1 neu sai so thoa man sai so uoc luong cho phep khi so lan lap nho %hon max_it , -1 neu nguoc lai
% itnum: so lan lap duoc su dung de tinh ra x.

% Khoi tao cac gia tri dau tien
% n la do dai (so cac phan tu trong) ma tran cot b
**n = length(b);**

% khoi tao gia tri dau cho bien iflag, k.
**iflag = 1; k = 0;**

% tao vecto voi cac phan tu la duong cheo ma tran A

```
diagA = diag(A);

% Hieu chinh A sao cho cac phan tu duong cheo cua no bang 0
A = A-diag(diag(A));

% Qua trinh lap, tinh toan theo cong thuc (6.24)
while k < max_it
  k = k+1;
  x = (b-A*x0)./diagA;

% Kiem tra dieu kien dung chuong trinh.
  relerr = norm(x-x0,inf)/(norm(x,inf)+eps);
  x0 = x;
  if relerr < delta
    break
  end
end
%
itnum = k;
if (itnum == max_it)
  iflag = -1;
end
```

**Chương trình 6.3a:**

```
function [x,iflag,itnum] = GS(A,b,x0,delta,max_it)
% Chuong trinh su dung phuong phap lap Gauss-Seidel de giai cac he phuong % trinh tuyen tinh Ax=b

% Gia tri dua vao:
%   A: cac he so ma tran vuong A
%   b: Vecto cot phai b
%   x0: vector lap ban dau,
%   delta:  uoc luong sai so cho phep giua cac lan lap lien tiep.
%   max_it:  so lan lap toi da cho phep trong gioi han sai so tren

% Khi ta da nhap cac tri dau vao, chuong trinh co the goi theo cu phap
%          [x, iflag, itnum] = GS
% Giai vi du 6.3.9.
A=[10,-1,2,0;-1,11,-1,3;2,-1,10,-1;0,3,-1,8];
b=[6;25;-11;15];
max_it=6;
```

```
x0=[0;0;0;0];
delta=0.0001;

% Tri dau ra
%   x: vecto nghiem
%   iflag: 1 neu sai so thoa man sai so uoc luong cho phep khi so lan lap thuc % te
nho hon max_it , -1 neu nguoc lai
%   itnum: so lan lap duoc su dung de tinh ra x.

% Khoi tao cac gia tri dau tien
% n la do dai (so cac phan tu trong) ma tran cot b
n = length(b);
% khoi tao gia tri dau cho bien iflag, k, nghiem x
iflag = 1; k = 0; x = x0;
% Qua trinh lap, tinh toan theo cong thuc (6.25)
while k < max_it
  k = k+1;
  x(1) = (b(1)-A(1,2:n)*x0(2:n))/A(1,1);
  for i = 2:n
    if i < n
      x(i) = (b(i)-A(i,1:i-1)*x(1:i-1) ...
            -A(i,i+1:n)*x0(i+1:n))/A(i,i);
    else
      x(n) = (b(n)-A(n,1:n-1)*x(1:n-1))/A(n,n);
    end
  end
 % Kiem tra dieu kien dung chuong trinh.
  relerr = norm(x-x0,inf)/(norm(x,inf)+eps);
  x0 = x;
  if relerr < delta
    break
  end
end
%
itnum = k;
if (itnum == max_it)
  iflag = -1;
end
```

# CHƯƠNG 7
# GIẢI SỐ PHƯƠNG TRÌNH VI PHÂN THƯỜNG

Nhiều bài toán của khoa học kỹ thuật dẫn đến việc giải phương trình vi phân thường. Bài toán đơn giản nhất và có nhiều ứng dụng thực tế đó là bài toán Cauchy hay còn gọi là bài toán điều kiện đầu của phương trình vi phân cấp 1.

Bài toán: Tìm nghiệm $Y = Y(x)$ thoả

$$\begin{cases} Y' = f(x, Y(x)), & x_0 \le x \le b, \\ Y(x_0) = Y_0. \end{cases} \tag{7.1}$$

Chúng tôi chỉ quan tâm đến các phương pháp giải số cho bài toán (7.1) với giả thiết rằng chúng có đầy đủ các điều kiện để đảm bảo sự tồn tại và tính duy nhất nghiệm của bài toán.

✓ Đối với bài toán hệ phương trình vi phân cấp 1. Tìm $Y_1(x)$, …, $Y_m(x)$ trên đoạn $x_0 \le x \le b$, thoả

$$\begin{aligned} &Y_1' = f_1(x, Y_1, ..., Y_m), \quad Y_1(x_0) = Y_{1,0}, \\ &\vdots \\ &Y_m' = f_m(x, Y_1, ..., Y_m), \quad Y_m(x_0) = Y_{m,0}. \end{aligned}$$

Ta đặt,

$$Y(x) = \begin{bmatrix} Y_1(x) \\ \vdots \\ Y_m(x) \end{bmatrix}, \quad Y_0 = \begin{bmatrix} Y_{1,0} \\ \vdots \\ Y_{m,0} \end{bmatrix}, \quad f(x,z) = \begin{bmatrix} f_1(x, z_1, ..., z_m) \\ \vdots \\ f_m(x, z_1, ..., z_m) \end{bmatrix}.$$

Với $z = [z_1, z_2, ..., z_m]^T$, thì bài toán hệ trên sẽ đưa về bài toán (7.1).

✓ Còn đối với bài toán phương trình vi phân cấp m

$$\frac{d^m Y}{dx^m} = f\left(x, Y, \frac{dY}{dx}, ..., \frac{d^{m-1}Y}{dx^{m-1}}\right),\ Y(x_0) = y_0, \ldots,\ Y^{(m-1)}(x_0) = Y_0^{(m-1)}.$$

Ta đặt, $Y_1 = Y,\quad Y_2 = Y',\quad ...,\quad Y_m = Y^{(m-1)}.$

Thì ta được bài toán hệ phương trình vi phân cấp 1 như sau

$$\begin{aligned} Y_1' &= Y_2 & Y_1(x_0) &= Y_0, \\ &\vdots & &\vdots \\ Y_{m-1}' &= Y_m & Y_{m-1}(x_0) &= Y_0^{(m-2)}, \\ Y_m' &= f(x, Y_1, ..., Y_m) & Y_m(x_0) &= Y_0^{(m-1)}. \end{aligned}$$

Và do đó lại đưa về bài toán Cauchy (7.1).

Sau đây chúng tôi trình bày một vài phương pháp phổ biến để giải (7.1), từ mục 7.1 đến mục 7.3 và bài toán biên tuyến tính cấp hai ở mục 7.4.

## 7.1. PHƯƠNG PHÁP EULER

Để tìm nghiệm gần đúng của (7.1), chia đoạn $[x_0, b]$ thành N đoạn nhỏ bằng nhau bởi các điểm chia $x_n = x_0 + nh, n = \overline{0,N}$; $x_N = b$; $h = (x_N - x_0)/2$. Ký hiệu: $Y_n$ là giá trị của nghiệm đúng $Y(x_n)$ và $y(x_n) = y_n \approx Y_n$. Ta có

$$Y'(x) \approx \frac{1}{h}\left[Y(x+h) - Y(x)\right],\quad Y'(x_n) = f(x_n, Y(x_n)).$$

Dẫn đến, $$\frac{1}{h}\left[Y(x_{n+1}) - Y(x_n)\right] \approx f(x_n, Y(x_n)),$$

Hay, $$Y(x_{n+1}) \approx Y(x_n) + hf(x_n, Y(x_n)). \qquad (7.2)$$

Như vậy, $$Y_{n+1} \approx y_{n+1} = y_n + hf\left(x_n, y_n\right), n = \overline{0,\left(N-1\right)}. \quad (7.3)$$

Công thức (7.3) là công thức Euler để giải số phương trình (7.1). Công thức này cho ta phép tính một cách đơn giản $y_{n+1}$ khi biết $y_n$ mà không phải giải một phương trình nào. Vì vậy, phương pháp Euler sử dụng (7.3) là một phương pháp hiện.

Người ta chứng minh được rằng sai số trong (7.3) tại $x_n$ là

$$\left|Y_n - y_n\right| \le Mh, \text{ ($M$ là hằng số dương không phụ thuộc h).} \quad (7.4)$$

Đánh giá (7.4) chỉ có giá trị lý thuyết vì tính M tương đối phức tạp. Trong thực hành người ta xác định sai số bằng cách " tính hai lần" như sau: lần thứ nhất tính bằng công thức (7.3) với bước h, nhận được $y_n\left(h\right)$ là giá trị gần đúng của $Y_n$; sau đó lại tính lần thứ hai với bước h/2, nhận được $y_{2n}\left(h/2\right)$ là giá trị gần đúng của $Y_n$, và sai số được xác định bởi

$$\left|y_{2n}\left(h/2\right) - Y_n\right| \approx \left|y_{2n}\left(h/2\right) - y_n\left(h\right)\right|. \quad (7.5)$$

Như vậy, công thức (7.3) có ưu điểm là tính toán đơn giản nhưng độ chính xác thấp (sai số bậc 1). Để tăng độ chính xác của nó, ta làm như sau:

Áp dụng công thức Newton – Leibnizt $Y\left(x_{n+1}\right) - Y\left(x_n\right) = \int_{x_n}^{x_{n+1}} Y'\left(x\right)dx.$

Áp dụng công thức (5.2), thì $Y\left(x_{n+1}\right) - Y\left(x_n\right) \approx \frac{h}{2}\left[Y'\left(x_{n+1}\right) + Y'\left(x_n\right)\right].$

Kết hợp (7.1), thì $Y\left(x_{n+1}\right) \approx Y\left(x_n\right) + \frac{h}{2}\left[f\left(x_{n+1}, Y\left(x_{n+1}\right)\right) + f\left(x_n, Y\left(x_n\right)\right)\right].$

Hay, $$y_{n+1} = y_n + h\frac{f(x_n, y_n) + f(x_{n+1}, y_{n+1})}{2},\ n = \overline{0, (N-1)}. \qquad (7.6)$$

Công thức (7.6) được gọi là công thức Euler cải tiến. Người ta chứng minh được rằng trong sai số trong (7.6) được đánh giá là

$$|Y_n - y_n| \le Mh^2, \text{ (} M \text{ là hằng số dương không phụ thuộc h).} \qquad (7.7)$$

Như vậy, độ chính xác của (7.6) sẽ tốt hơn (7.3) vì nó có độ chính xác bậc hai. Tuy nhiên việc tính toán theo (7.6) phức tạp vì để giải tìm $y_{n+1}$ ta phải thông qua việc giải phương trình đại số phi tuyến (nếu f phi tuyến), nên phương pháp Euler dùng (7.6) là phương pháp ẩn. Ngoài ra, để khai thác những ưu điểm và khắc phục nhược điểm của hai công thức (7.3) và (7.6), ta thay $y_{n+1}$ trong (7.3) vào vế phải của (7.6), ta được

$$y_{n+1} = y_n + h\frac{f(x_n, y_n) + f(x_{n+1}, y_n + hf(x_n, y_n))}{2},\ n = \overline{0, (N-1)}. \qquad (7.8)$$

**CHƯƠNG TRÌNH MATLAB:** Dùng phương pháp Euler để giải số bài toán

$$\begin{cases} y' = y - x^2 + 1, & 0 \le x \le 2, \\ y(0) = 0.5. \end{cases}$$

Với n=10, bước h=0.2 và nghiệm chính xác là $Y_{ex}(x) = (x+1)^2 - 0.5e^x$.

Chúng tôi cung cấp hai chương trình MATLAB để giải số bài toán dưới đây bằng phương pháp Euler lần lượt dùng (7.3) và (7.8). Để chạy chương trình ta cần tạo các file tạo hàm fcn và Yex và lưu và cùng thư mục với các file chương trình euler_hien.m và euler_an.m (các file có trong thư mục euler) . Kết quả trả về là các vector cột N dòng gồm có giá trị các điểm chia $x$, giá trị giải số tương ứng

$y$, giá trị chính xác $Y$ và sai số $err$, được biểu thị trong Bảng 7.1 và Bảng 7.2 và vẽ đồ thị minh hoạ, Hình 7.1 và Hình 7.2. Hàm được gọi theo cú pháp

[x,y,Y,err] = euler_hien(fcn) và [x,y,Y,err] = euler_an(fcn).

**Chương trình 7.1:**

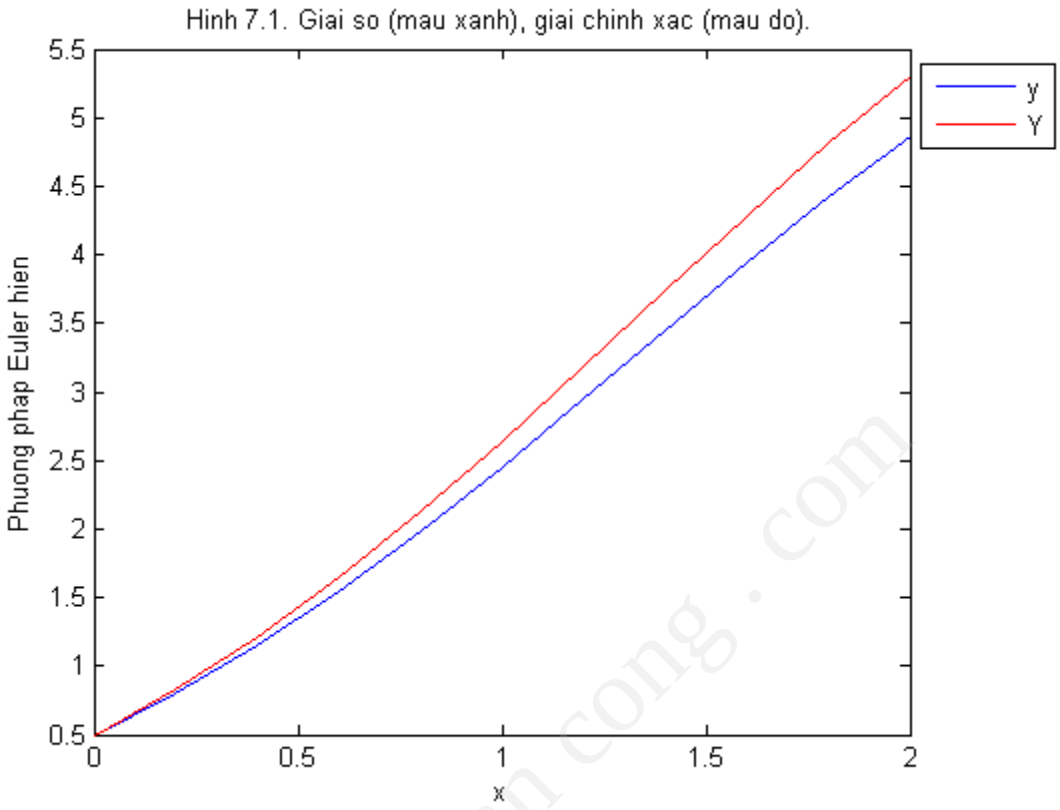
```
% file fcn.m; Tao ham fcn(x,y)
function a=fcn(x,y)
a=inline('y-x^2+1');
                % Gia tri tra ve cho ham hai bien a(x,y) de gan cho ham fcn.
% file Yex.m; Tao ham chinh xac Yex(x)
function a=Yex(x)
a=(x+1)^2-0.5*exp(x);
% Chuong trinh giai bai toan Cauchy bang phuong phap Euler hien
% Tao file euler_hien.m
function [x,y,Y,err] = euler_hien(fcn)
% Day la ham [x,y,Y,err] = euler_an(fcn)
% Ham giai so bai toan Cauchy theo phuong phap Euler hien.
% Cac doi so la: x0, y0, h, x_end=xN.
% Output:
%  Vector x duoc xac dinh thong qua cac diem:
%   x(1)=x0, x(j)=x0+(j-1)*h, j=1,2,...,N
% voi x(N) <= x_end-h,  x(N)+h > x_end-h
% Vector y duoc tinh tuong ung theo x thong qua ham fcn, bang (7.3).
% Vector Y duoc tinh tuong ung theo x thong qua ham Yex.
% Vector err duoc tinh bang cong thuc err(i)=abs(Y(i)-y(i)).

% Nhap cac doi so theo yeu cau tung bai toan.
x0=0;y0=0.5;
h=0.2; x_end=2;
% Khoi tao cac gia tri
N = fix((x_end-x0)/h)+1                 % Dem so diem chia.
x = linspace(x0,x_end,N)';              % Chia [x0, x_end] thanh N diem.

% Giai so y=y(x)
y = zeros(N,1);
y(1) = y0;                  % Danh so y(1)=y0 dan den tinh y(2),...,y(N).
for i = 2:N
  y(i)=y(i-1)+h*feval(fcn,x(i-1),y(i-1));          % Giai so theo cong thuc (7.3).
end

% Giai chinh xac Y=y(x)
```

```
X=x;Y = zeros(N,1);
for i=1:N
   Y(i)=Yex(X(i));                    % Tinh chinh xac bang ham Yex.
end
% Tinh toan cac sai so.
err = zeros(N,1);
for i=1:N
   err(i)=Y(i)-y(i);
end

% Ve do thi phep giai so va phep giai chinh xac tren cung mot hinh.
plot(x,y,'b',X,Y,'r')
xlabel('x')
ylabel('Phuong phap Euler hien')
title('Hinh 7.1. Giai so (mau xanh), giai chinh xac (mau do).')
legend('y','Y',-1)
```

Kết quả chạy chương trình 7.1 theo cú pháp [x,y,Y,err] = euler_hien(fcn).

**Bảng 7.1.** Kết quả giải số bằng phương pháp Euler hiện với n=0, 1,…,10.

| $x_n$ | $y_n$ | $Y_n$ | $err$ |
|---|---|---|---|
| 0 | 0.5000 | 0.5000 | 0 |
| 0.2000 | 0.8000 | 0.8293 | 0.0293 |
| 0.4000 | 1.1520 | 1.2141 | 0.0621 |
| 0.6000 | 1.5504 | 1.6489 | 0.0985 |
| 0.8000 | 1.9885 | 2.1272 | 0.1387 |
| 1.0000 | 2.4582 | 2.6409 | 0.1827 |
| 1.2000 | 2.9498 | 3.1799 | 0.2301 |
| 1.4000 | 3.4518 | 3.7324 | 0.2806 |
| 1.6000 | 3.9501 | 4.2835 | 0.3334 |
| 1.8000 | 4.4282 | 4.8152 | 0.3870 |
| 2.0000 | 4.8658 | 5.3055 | 0.4397 |




**Chương trình 7.1a:** Giải bài toán bằng phương pháp Euler ẩn.

Ta chỉ cần thay đoạn chương trình giải số theo công thức (7.3) thành (7.8) như sau

```
 % Giai so y=y(x)
y = zeros(N,1);
y(1) = y0;                % Danh so y(1)=y0 dan den tinh y(2),...,y(N).
for i = 2:N
  y(i)=y(i-1)+h*feval(fcn,x(i-1),y(i-1));
  y(i)=y(i-1)+h*(feval(fcn,x(i-1),y(i-1))+feval(fcn,x(i),y(i)))/2;
        % Giai so theo cong thuc (7.8).
end
```

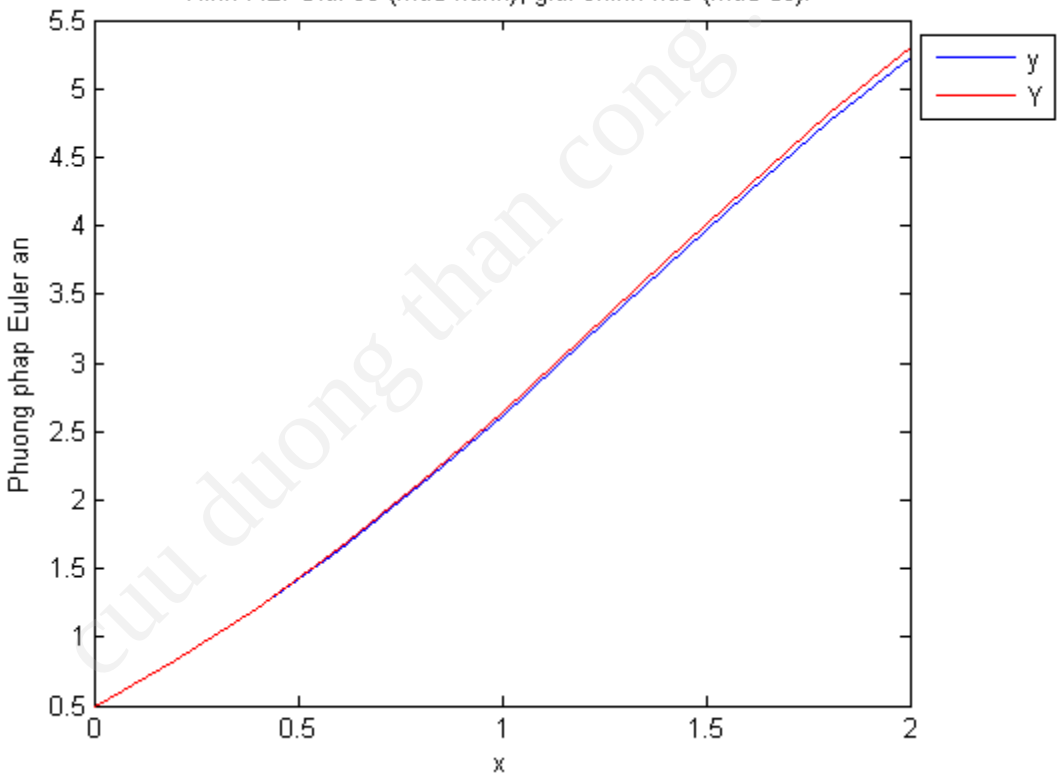
Lời giải chi tiết xin xem trong thư mục euler.

Kết quả chạy chương trình theo cú pháp [x,y,Y,err] = euler_an(fcn).

**Bảng 7.2**. Kết quả giải số bằng phương pháp Euler ẩn với n=0,1, …,10.

| $x_n$ | $y_n$ | $Y_n$ | $err$ |
|---|---|---|---|
| 0 | 0.5000 | 0.5000 | 0 |
| 0.2000 | 0.8260 | 0.8293 | 0.0033 |
| 0.4000 | 1.2069 | 1.2141 | 0.0072 |
| 0.6000 | 1.6372 | 1.6489 | 0.0117 |
| 0.8000 | 2.1102 | 2.1272 | 0.0170 |
| 1.0000 | 2.6177 | 2.6409 | 0.0232 |
| 1.2000 | 3.1496 | 3.1799 | 0.0304 |
| 1.4000 | 3.6937 | 3.7324 | 0.0387 |
| 1.6000 | 4.2351 | 4.2835 | 0.0484 |
| 1.8000 | 4.7556 | 4.8152 | 0.0596 |
| 2.0000 | 5.2331 | 5.3055 | 0.0724 |

Hinh 7.2. Giai so (mau xanh), giai chinh xac (mau do).




## 7.2. PHƯƠNG PHÁP RUNGE – KUTTA

Phương pháp Runge – Kutta là phương pháp giải bài toán (7.1) theo công thức xấp xỉ

$$\begin{cases} Y\left(x_n+h\right) \approx y_n + \sum_{j=1}^{M} A_j K_j^n, \\ K_1^n = hf\left(x_n,\ y_n\right), \\ K_2^n = hf\left(x_n+\alpha_2 h,\ y_n+\beta_{21}K_1^n\right), \\ K_2^n = hf\left(x_n+\alpha_3 h,\ y_n+\beta_{31}K_1^n+\beta_{32}K_2^n\right), \\ \ldots \\ K_M^n = hf\left(x_n+\alpha_M h,\ y_n+\beta_{M1}K_1^n+\beta_{M2}K_2^n+\ldots+\beta_{M,\,M-1}K_{M-1}^n\right). \end{cases} \quad (7.9)$$

Trong đó các hệ số $A_j$, $\alpha_j$, $\beta_{j\,i}$, được xác định bằng cách đặt

$$\varphi(h) = Y\left(x_n+h\right) - y_n - \sum_{j=1}^{M} A_j K_j^n .$$

Hàm $\varphi(h)$, phụ thuộc h có đạo hàm đến cấp m. Giá trị tuyệt đối của $\varphi(h)$ là sai số tuyệt đối của giá trị gần đúng tại $x_n+h$. Chú ý rằng $\varphi(0)=0$, do đó:

$$\varphi(h) = \varphi'(0)h + \frac{1}{2}\varphi''(0)h^2 + \ldots + \frac{1}{m!}\varphi^{(m)}(0)h^m + o\left(h^m\right).$$

Các hệ số $A_j$, $\alpha_j$, $\beta_{j\,i}$ được xác định theo điều kiện:

$$\varphi'(0) = \varphi''(0) = \ldots = \varphi^{(m)}(0) = 0. \quad (7.10)$$

Khi đó, $\varphi(h) = o\left(h^m\right)$. Chúng ta sẽ xét vài trường hợp thường được dùng trong giải số.

- Trường hợp 1. M=m=1, khi đó công thức (7.9) có dạng

$$Y\left(x_n+h\right) \approx y_n + A_1 K_1^n,\ \ K_1^n = hf\left(x_n, y_n\right).$$

Xác định hệ số $A_1$ với

$$\varphi(h) = Y\left(x_n+h\right) - y_n - A_1 K_1^n = y\left(x_n+h\right) - y_n - A_1 hf\left(x_n, y_n\right).$$

Điều kiện (7.10) trở thành

$\varphi'(0) = Y'(x_n) - A_1 f(x_n, y_n) \approx f(x_n, y_n) - A_1 f(x_n, y_n)$, nên $A_1 = 1$.

Đây là trường hợp của công thức Euler (7.3).

- Trường hợp 2. M=m=2, khi đó công thức (7.9) có dạng

$$\begin{cases} Y(x_n + h) \approx y_n + A_1 K_1^n + A_2 K_2^n, \\ K_1^n = hf(x_n,\ y_n), \\ K_2^n = hf(x_n + \alpha_2 h,\ y_n + \beta_{21} K_1^n). \end{cases}$$

Các hệ số cần tìm $A_1$, $A_1$, $\alpha_2$, $\beta_{21}$ được xác định theo điều kiện (7.10).

$$\varphi(h) = y(x_n + h) - y_n - A_1 K_1^n - A_2 K_2^n.$$

Tính $\varphi'(h)$, $\varphi''(h)$ và áp dụng $\varphi'(0) = \varphi''(0) = 0$, ta được hệ:

$$\begin{cases} 1 - A_1 - A_2 = 0, \\ 1 - 2\alpha_2 A_2 = 0, \\ 1 - 2\beta_{21} A_2 = 0. \end{cases}$$

Hệ này có vô số nghiệm. Trong đó hai bộ nghiệm tương ứng với hai công thức thường dùng là

i. $A_1 = A_2 = \frac{1}{2}$, $\alpha_2 = \beta_{21} = 1$. Ta được

$$\begin{cases} Y(x_n + h) \approx y_n + \frac{1}{2}(K_1^n + K_2^n), \\ K_1^n = hf(x_n,\ y_n), \\ K_2^n = hf(x_n + h,\ y_n + K_1^n). \end{cases}$$

Đây là công thức Euler cải tiến (7.5).

ii. $A_1 = A_2 = 1$, $\alpha_2 = \beta_{21} = \frac{1}{2}$. Ta được

$$\begin{cases} y\left(x_n+h\right) \approx y_n + K_1^n + K_2^n, \\ K_1^n = hf\left(x_n,\ y_n\right), \\ K_2^n = hf\left(x_n + h/2,\ y_n + K_1^n/2\right). \end{cases} \tag{7.11}$$

- Trường hợp 3. M=m=4, ta có

$$\begin{cases} Y\left(x_n+h\right) \approx y_n + \dfrac{1}{6}\left(K_1^n + 2K_2^n + 2K_3^n + K_4^n\right), \\ K_1^n = hf\left(x_n,\ y_n\right), \\ K_2^n = hf\left(x_n + h/2,\ y_n + K_1^n/2\right), \\ K_3^n = hf\left(x_n + h/2,\ y_n + K_2^n/2\right), \\ K_4^n = hf\left(x_n + h,\ y_n + K_3^n\right). \end{cases} \tag{7.12}$$

Công thức (7.12) gọi là công thức Runge – Kutta cấp 4.

**CHƯƠNG TRÌNH MATLAB:** Chúng tôi trình bày một chương trình MATLAB để giải số bài toán (7.1) bằng công thức Runge – Kutta cấp 4. Công thức này có độ chính xác bậc 4, tốt hơn phương pháp Euler. Chúng tôi lấy cùng bài toán cụ thể của phương pháp Euler để tiện việc so sánh. Kết quả cho độ chính xác cao, sai số không đáng kể và đồ thị gần như trùng lấp nhau thành một, kết quả thể hiện trong Bảng 7.3 và Hình 7.3 . Cấu trúc chương trình cũng giống như chương trình 7.1 và chương trình 7.1a, chỉ khác nhau ở cách tính giá trị giải số vector y. Chúng tôi trình bày chổ khác biệt này, còn chi tiết xin xem trong thư mục rk. Chương trình được gọi theo cú pháp

[x,y,Y,err] = rk4(fcn).

**Chương trình 7.2:**

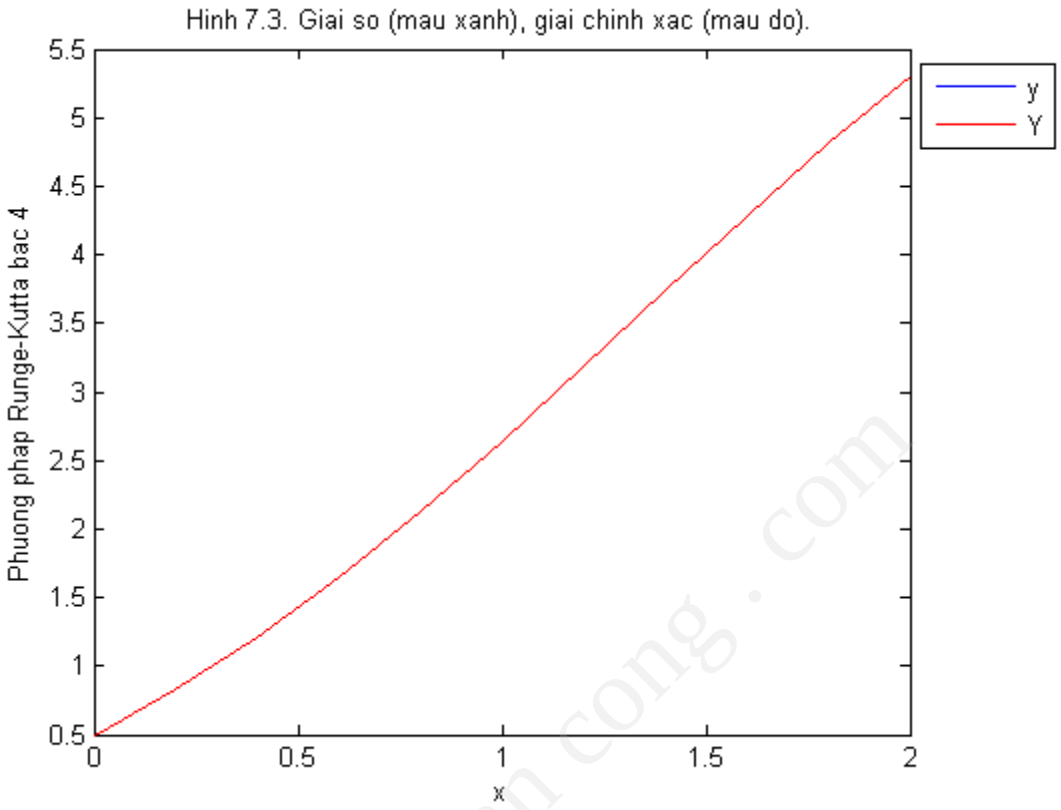
```
function [x,y,Y,err] = rk4(fcn)
% Day la ham [x,y,Y,err] = rk4(fcn)
% Ham giai so bai toan Cauchy theo phuong phap Runge-Kutta (bac 4).
% Cac doi so la: x0, y0, h, x_end=xN.
```

```
% Output:
%  Vector x duoc xac dinh thong qua cac diem:
%   x(1)=x0, x(j)=x0+(j-1)*h, j=1,2,...,N
% voi x(N) <= x_end-h,  x(N)+h > x_end-h
% Vector y duoc tinh tuong ung theo x thong qua ham fcn, bang (7.12).
% Vector Y duoc tinh tuong ung theo x thong qua ham Yex.
% Vector err duoc tinh bang cong thuc err(i)=abs(Y(i)-y(i)).

% Nhap cac doi so theo yeu cau tung bai toan.
x0=0;y0=0.5;
h=0.2; x_end=2;
% Khoi tao cac gia tri
N = fix((x_end-x0)/h)+1;          % Dem so diem chia.
x = linspace(x0,x_end,N)';        % Chia [x0, x_end] thanh N diem.
% Giai so y=y(x)
y = zeros(N,1);
y(1) = y0;              % Danh so y(1)=y0 dan den tinh y(2),...,y(N).
for i = 2:N
  K1=h*feval(fcn,x(i-1),y(i-1));
  K2=h*feval(fcn,x(i-1)+h/2,y(i-1)+K1/2);
  K3=h*feval(fcn,x(i-1)+h/2,y(i-1)+K2/2);
  K4=h*feval(fcn,x(i-1)+h,y(i-1)+K3);
  y(i)=y(i-1)+(K1+2*K2+2*K3+K4)/6;      % Giai so theo cong thuc (7.12).
end
```

Kết quả chạy chương trình 7.3 theo cú pháp [x,y,Y,err] = rk4(fcn).

**Bảng 7.3.** Kết quả giải số bằng công thức (7.12), với n=0, 1, …, 10.

| $x_n$ | $y_n$ | $Y_n$ | $err_{\times 10^{-3}}$ |
|---|---|---|---|
| 0 | 0.5000 | 0.5000 | 0 |
| 0.2000 | 0.8293 | 0.8293 | 0.0053 |
| 0.4000 | 1.2141 | 1.2141 | 0.0114 |
| 0.6000 | 1.6489 | 1.6489 | 0.0186 |
| 0.8000 | 2.1272 | 2.1272 | 0.0269 |
| 1.0000 | 2.6408 | 2.6409 | 0.0364 |
| 1.2000 | 3.1799 | 3.1799 | 0.0474 |
| 1.4000 | 3.7323 | 3.7324 | 0.0599 |
| 1.6000 | 4.2834 | 4.2835 | 0.0743 |
| 1.8000 | 4.8151 | 4.8152 | 0.0906 |
| 2.0000 | 5.3054 | 5.3055 | 0.1089 |




## 7.3. PHƯƠNG PHÁP ĐA BƯỚC (MULTISTEP METHODS)

Tiếp tục xét bài toán (7.1). Ta có

$$Y\left(x_{n+1}\right)-Y\left(x_n\right)=\int_{x_n}^{x_{n+1}} Y'\left(x\right)dx\,. \qquad (7.13)$$

Từ (7.1) đẳng thức trên suy ra $Y\left(x_{n+1}\right)=Y\left(x_n\right)+\int_{x_n}^{x_{n+1}} f\left(x,Y\left(x\right)\right)dx\,.$

Ta tính tích phân $\int_{x_n}^{x_{n+1}} g\left(x\right)dx, \qquad g\left(x\right)=f\left(x,Y\left(x\right)\right).$

Phương pháp đa bước là tên gọi chung của hai phương pháp: Adams – Bashforth (AB) và Adams – Moulton (AM). Ý tưởng chính của hai phương pháp

này là xấp xỉ g(x) bởi đa thức nội suy bậc q của nó tại các điểm $\{x_n, x_{n-1},..., x_{n-q}\}$ (AB) và $\{x_{n+1}, x_n,..., x_{n-q+1}\}$ (AM). Chúng tôi trình bày phương pháp AB.

Đa thức nội suy tuyến tính của $g(x)$ tại $\{x_n, x_{n+1}\}$ là

$$p_1(x) = \frac{1}{h}\left[(x_n - x)g(x_{n-1}) + (x - x_{n-1})g(x_n)\right]. \quad (7.14)$$

Do đó, $\int_{x_n}^{x_{n+1}} g(x)dx \approx \int_{x_n}^{x_{n+1}} p_1(x)dx = \frac{3h}{2}g(x_n) - \frac{h}{2}g(x_{n-1})$.

Suy ra, $Y(x_{n+1}) \approx Y(x_n) + \frac{h}{2}\left[3f(x_n, Y(x_n)) - f(x_{n-1}, Y(x_{n-1}))\right]$.

Vậy, $y_{n+1} = y_n + \frac{h}{2}\left[3f(x_n, y_n) - f(x_{n-1}, y_{n-1})\right]$.

Hay, $y_{n+1} = y_n + \frac{h}{2}\left[3y'_n - y'_{n-1}\right]$. (7.15)

Công thức (7.15) gọi là công thức AB bậc 2.

Hoàn toàn tương tự ta có

Phương pháp AB

| q | Bậc | Công thức giải số |
|---|---|---|
| 0 | 1 | $y_{n+1} = y_n + hy'_n$ |
| 1 | 2 | $y_{n+1} = y_n + h\left[3y'_n - y'_{n-1}\right]/2$ |
| 2 | 3 | $y_{n+1} = y_n + h\left[23y'_n - 16y'_{n-1} + 5y''_{n-2}\right]/12$ |
| 3 | 4 | $y_{n+1} = y_n + h\left[55y'_n - 59y'_{n-1} + 37y''_{n-2} - 9y'_{n-3}\right]/24$ |

Bậc ở đây là bậc của phương pháp và cũng là bậc của sai số.

Đối với phương pháp AM ta có

Phương pháp AM

| q | Bậc | Công thức giải số |
|---|---|---|
| 0 | 1 | $y_{n+1} = y_n + hy'_n$ |
| 1 | 2 | $y_{n+1} = y_n + h\left[y'_{n+1} + y'_n\right]/2$ |
| 2 | 3 | $y_{n+1} = y_n + h\left[5y'_{n+1} + 8y'_n - y''_{n-1}\right]/12$ |
| 3 | 4 | $y_{n+1} = y_n + h\left[9y'_{n+1} + 19y'_n - 5y''_{n-1} + y'_{n-2}\right]/24$ |

**CHƯƠNG TRÌNH MATLAB:** Chúng tôi trình bày một chương trình MATLAB để giải số bài toán (7.1) bằng công thức AB bậc 2. Cụ thể, chúng tôi cũng lấy cùng một bài toán của các phương pháp trước để tiện việc so sánh. Để chạy chương trình ta cần tạo các file tao hàm fcn và Yex và lưu và cùng thư mục với các file chương trình AB2.m, các file có trong thư mục AB; ở đây chúng tôi trình bày đoạn chương trình chính tạo file AB2.m. Kết quả trả về là các vector cột N dòng gồm giá trị các điểm chia $x$, giá trị giải số tương ứng $y$, giá trị chính xác $Y$ và sai số $err$, được biểu thị trong Bảng 7.4 và vẽ đồ thị minh hoạ, Hình 7.4. Hàm được gọi theo cú pháp

[x,y,Y,err] = AB2(fcn).

**Chương trình 7.3:**

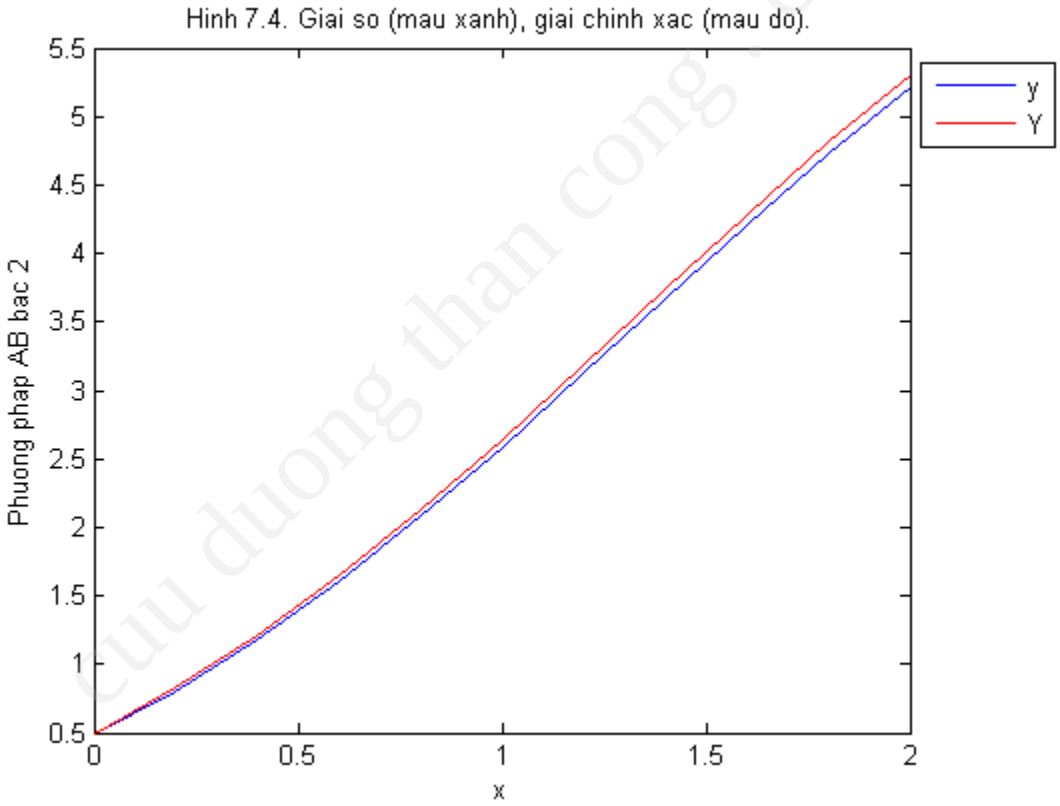
```
% Tao file AB2.m
function [x,y,Y,err] = AB2(fcn)
% Day la ham  [x,y,Y,err]=AB2(fcn).
% Dung de giai so bai toan Cauchy 7.1 bang cong thuc AB bac 2.
%  Cac doi so la: x0, y0, h, x_end=xN.
% Ham duoc goi theo cu  phap [x,y,Y,err]=AB2(fcn).
% Output:
%  Vector x duoc xac dinh thong qua cac diem:
%   x(1)=x0, x(j)=x0+(j-1)*h, j=1,2,...,N
% voi x(N) <= x_end-h,  x(N)+h > x_end-h
% Vector y duoc tinh tuong ung theo x thong qua ham fcn, bang (7.15).
% Vector Y duoc tinh tuong ung theo x thong qua ham Yex.
% Vector err duoc tinh bang cong thuc err(i)=abs(Y(i)-y(i)).
% Nhap cac doi so theo yeu cau tung bai toan.
x0=0;
```

```
y0=0.5;
h=0.2;
x_end=2;

% Khoi tao cac gia tri
N = fix((x_end-x0)/h)+1;        % Dem so diem chia.
x = linspace(x0,x_end,N)';      % Chia [x0, x_end] thanh N diem.

% Giai so y=y(x)
y = zeros(N,1);
y(1) = y0;                      % Danh so y(1)=y0 dan den tinh y(2),...,y(N).
ft1 = feval(fcn,x(1),y(1));
y(2) = y(1)+h*ft1;              % Tinh y(2)theo truong hop q=0.
for i = 3:N
  ft2 = feval(fcn,x(i-1),y(i-1));
  y(i) = y(i-1)+h*(3*ft2-ft1)/2;     % Giai so theo cong thuc AB bac hai.
  ft1 = ft2;
end

% Giai chinh xac Y=y(x)
X=x;
Y = zeros(N,1);
for i=1:N
   Y(i)=Yex(X(i));                   % Tinh chinh xac bang ham Yex.
end
% Tinh toan cac sai so.
err = zeros(N,1);
for i=1:N
   err(i)=Y(i)-y(i);
end

% Ve do thi phep giai so va phep giai chinh xac tren cung mot hinh.
plot(x,y,'b',X,Y,'r')
xlabel('x')
ylabel('Phuong phap AB bac 2')
title('Hinh 7.4. Giai so (mau xanh), giai chinh xac (mau do).')
legend('y','Y',-1)
```

Kết quả chạy chương trình 7.4 theo cú pháp [x,y,Y,err] = AB2(fcn).

**Bảng 7.4.** Kết quả giải số theo phương pháp AB2, với n=0, 1, …, 10.

| $x_n$ | $y_n$ | $Y_n$ | $err$ |
|---|---|---|---|
| 0 | 0.5000 | 0.5000 | 0 |
| 0.2000 | 0.8000 | 0.8293 | 0.0293 |
| 0.4000 | 1.1780 | 1.2141 | 0.0361 |
| 0.6000 | 1.6074 | 1.6489 | 0.0415 |
| 0.8000 | 2.0798 | 2.1272 | 0.0474 |
| 1.0000 | 2.5870 | 2.6409 | 0.0538 |
| 1.2000 | 3.1192 | 3.1799 | 0.0608 |
| 1.4000 | 3.6642 | 3.7324 | 0.0682 |
| 1.6000 | 4.2075 | 4.2835 | 0.0759 |
| 1.8000 | 4.7314 | 4.8152 | 0.0838 |
| 2.0000 | 5.2140 | 5.3055 | 0.0914 |




## 7.5. BÀI TOÁN BIÊN TUYẾN TÍNH CẤP HAI

Trong phần này chúng tôi trình bày phương pháp sai phân hữu hạn giải bài toán biên của phương trình vi phân thường tuyến tính cấp hai với điều kiện biên được cho ở hai điểm có dạng

$$\begin{cases} p(x)Y''(x)+q(x)Y'(x)+r(x)Y(x)=f(x), \quad a\le x\le b, \\ Y(a)=g_a, Y(b)=g_b. \end{cases} \tag{7.16}$$

Cho $N\in Z^+$, $h=(b-a)/N$, chia đoạn $[a,b]$ thành $N$ đoạn độ dài h bởi $N+1$ điểm $x_i=a+ih,\ 0\le i\le N$. Với $f_i=f(x_i)$, $p_i=p(x_i)$, $q_i=q(x_i)$, $r_i=r(x_i)$ và $Y_i=Y(x_i)$, $y_i\approx Y_i$. Ta áp dụng công thức sai phân hướng tâm (5.27) và (5.31), ta có

$$Y'(x_i)=\frac{Y_{i+1}-Y_{i-1}}{2h}+o(h^2), \tag{7.17}$$

$$Y'(x_i)=\frac{Y_{i+1}-2Y_i+Y_{i-1}}{h^2}+o(h^2). \tag{7.18}$$

Thay vào (7.16) với $x=x_i$, ta được

$$p_i\frac{Y_{i+1}-2Y_i+Y_{i-1}}{h^2}+q_i\frac{Y_{i+1}-Y_{i-1}}{2h}+r_iY_i+o(h^2)=f_i. \tag{7.19}$$

Dẫn đến, $p_i\dfrac{y_{i+1}-2y_i+y_{i-1}}{h^2}+q_i\dfrac{y_{i+1}-y_{i-1}}{2h}+r_iy_i=f_i,\quad 1\le i\le N-1.$

Cùng với điều kiện biên, ta thu được

$$\begin{cases} C_iy_{i-1}+A_iy_i+B_iy_{i+1}=f_i;\ A_i=r_i-\dfrac{2p_i}{h^2},\ B_i=\dfrac{p_i}{h^2}+\dfrac{q_i}{2h},\ C_i=\dfrac{p_i}{h^2}-\dfrac{q_i}{2h}, \\ y_0=g_a,\ y_N=g_b, \qquad\qquad 1\le i\le N-1. \end{cases}$$

Với i=1: $A_1y_1+B_1y_2=f_i-C_1g_a$.

Với i=N – 1: $C_{N-1}y_{N-2}+A_{N-1}y_{N-1}=f_{N-1}-B_{N-1}g_b$.

Ta giải số $y=[y_1,...,y_{N-1}]^T$ từ hệ $Ay=b$. Trong đó, A là ma trận ba đường chéo

$$A=\begin{pmatrix} A_1 & B_1 & 0 & \dots & 0 & 0 \\ C_2 & A_2 & B_2 & \dots & 0 & 0 \\ 0 & C_3 & A_3 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & A_{N-2} & B_{N-2} \\ 0 & 0 & 0 & \dots & C_{N-2} & A_{N-1} \end{pmatrix}, \quad b=\begin{pmatrix} f_1 - A_1 g_a \\ f_2 \\ \dots \\ f_{N-2} \\ f_{N-1} - B_{N-1} g_b \end{pmatrix}.$$

**CHƯƠNG TRÌNH MATLAB:** Chúng tôi trình bày chương trình MATLAB để giải số bài toán biên tuyến tính cấp hai (7.16) với bài toán cụ thể

$$\begin{cases} \left(1+x^2\right)Y''+2xY'-\left(1+x^2\right)Y=2-\left(1+x^2\right)\log\left(1+x^2\right), & 0\le x\le 1, \\ Y(0)=0,\ Y(1)=\log(2). \end{cases}$$

Lời giải chính xác là $Yex=\log\left(1+x^2\right)$.

Các đối số của chương trình 7.4 là a, b, ga, gb, n và các hàm dạng inline{} là f, p, q, r; các hàm này được tạo ra theo từng bài toán cụ thể, cùng với file Yex.m và lưu cùng thư mục với file chương trình btb2.m. Các file hàm có trong thư mục btb; ở đây chúng tôi chỉ trình bày đoạn chương trình chính tạo file btb2.m. Kết quả trả về là các vector cột N dòng gồm giá trị các điểm chia $x$, giá trị giải số tương ứng $y$, giá trị chính xác $Y$ và sai số $err$, được biểu thị trong Bảng 7.5 và đồ thị minh hoạ Hình 7.5.

Hàm được gọi theo cú pháp [x,y,Y,err]=btb2(p,q,r,f,a,b,ga,gb,n).

**Chương trình 7.4:**

```
function [x,y,Y,err]=btb2(p,q,r,f,a,b,ga,gb,n)
% Day la ham x,y,Y,err]=btb2(p,q,r,f,a,b,ga,gb,n); giải bai toan bien cap hai.
if nargin<9, error('Ham co it nhat 9 doi so.'); end;
                    % Kiem tra cac doi so nhap vao khi goi chuong trinh.
% Khoi tao ban dau:
h=(b-a)/n; N=n+1;
```

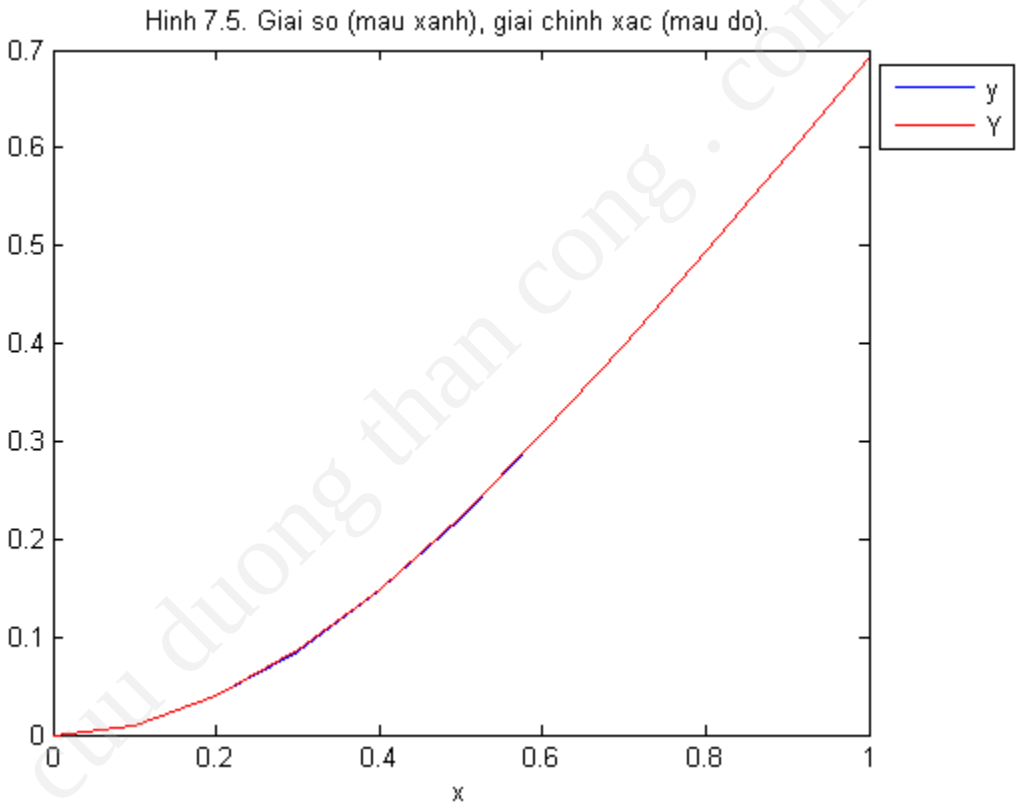
```
x=linspace(a,b,N);          % Chia doan [a, b] thanh n+1 diem.
y(1)=ga;y(N)=gb;            % Dieu kien bien.

% Tinh cac he so cua ma tran ba duong cheo.
for i=1:n-1, A(i)=r(x(i+1))-2*p(x(i+1))/h/h; end;      % Duong cheo chinh giua
for i=1:n-2, B(i)=p(x(i+1))/h/h+q(x(i+1))/2/h; end;    % Duong cheo tren
for i=1:n-2, C(i)=p(x(i+2))/h/h-q(x(i+2))/2/h; end;    % Duong cheo duoi

% Tao cac gia tri cua ma tran b:
for i=1:n-1, b(i)=f(x(i+1)); end;
b(1)=b(1)-(p(x(2))/h/h-q(x(2))/2/h)*ga;
b(n-1)=b(n-1)-(p(x(n))/h/h+q(x(n))/2/h)*gb;

% Giai so y theo he ma tran ba duong cheo:
z=tridiag(A,B,C,b,n-1);
for i=1:n-1, y(i+1)=z(i); end;
x=x'; y=y';                 %Chuyen vi x, y de ket qua xuat ra dang cot.

%Giai chinh xac Y=y(x)
X=x;
Y = zeros(N,1);
for i=1:N
   Y(i)=Yex(X(i));     % Tinh chinh xac bang ham Yex.
end
% Tinh toan cac sai so.
err = zeros(N,1);
for i=1:N
   err(i)=abs(Y(i)-y(i));
end
% Ve do thi phep giai so va phep giai chinh xac tren cung mot hinh.
plot(x,y,'b',X,Y,'r')
xlabel('x')
title('Hinh 7.5. Giai so (mau xanh), giai chinh xac (mau do).')
legend('y','Y',-1)
```

Kết quả giải bài toán cụ thể trên, chạy chương trình theo cú pháp
[x,y,Y,err]=btb2(p,q,r,f,0,1,0,log(2),10).

**Bảng 7.5.** Kết quả giải số bài toán biên tuyến tính cấp 2, n=0, 1, …, 10.

| $x_n$ | $y_n$ | $Y_n$ | $err_{\times 10^{-3}}$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |

| | | | |
|---|---|---|---|
| 0.1000 | 0.0097 | 0.0100 | 0.2044 |
| 0.2000 | 0.0389 | 0.0392 | 0.3141 |
| 0.3000 | 0.0858 | 0.0862 | 0.3459 |
| 0.4000 | 0.1481 | 0.1484 | 0.3232 |
| 0.5000 | 0.2229 | 0.2231 | 0.2692 |
| 0.6000 | 0.3073 | 0.3075 | 0.2028 |
| 0.7000 | 0.3986 | 0.3988 | 0.1372 |
| 0.8000 | 0.4946 | 0.4947 | 0.0798 |
| 0.9000 | 0.5933 | 0.5933 | 0.0339 |
| 1.0000 | 0.6931 | 0.6931 | 0 |



# CHƯƠNG 8
# GIẢI SỐ PHƯƠNG TRÌNH ĐẠO HÀM RIÊNG

Trong chương này chúng tôi dùng phương pháp phân rã để giải số một số phương trình đạo hàm riêng. Chúng tôi bàn nhiều về các bài toán một chiều, và ở các bài toán hai chiều chúng tôi trình bày sơ lược về ý tưởng chính của phương pháp tính để thiết lập chương trình giải số. Ở mỗi bài toán, chúng tôi trình bày một chương trình MATLAB để giải số một bài toán cụ thể để làm rõ ý tưởng của phương pháp và chương trình MATLAB được dùng để giải số bài toán đó. Có nhiều phương pháp giải số khác nhau, chúng tôi không trình bày chi tiết các phương pháp giải số này mà chỉ nêu ra sự thay đổi chương trình MATLAB như thế nào khi có sự thay đổi nhỏ trong cách xây dựng công thức tính. Cụ thể là các thay đổi về cách thiết lập ma trận giải số trong chương trình MATLAB. Trên cơ sở đó, chúng ta có thể thay đổi các dữ kiện phù hợp để dùng chương trình MATLAB này giải số các bài toán tương tự.

## 8.1. BÀI TOÁN LAPLACE 1 CHIỀU

### 8.1.1 Bài toán

Tìm hàm số u(x) trên [0, 1] thỏa mãn

$$\begin{cases} u''(x) = f(x), x \in [0,1], \\ u(0) = a_0, u(1) = a_1. \end{cases}$$

### 8.1.2 Phân rã bài toán

Cho $N \in Z^+, h = 1/N = \Delta x$. Ta chia đoạn [0, 1] thành N đoạn độ dài h bởi N+1 điểm

$$x_i = (i-1)h, \quad i = 1,2,,,N+1.$$

Ta sẽ tính xấp xỉ $U = [u(x_1),...,u(x_{N+1})]^T$. Có hai trường hợp

Trường hợp điểm biên. Với $i = 1, i = N+1$ ta có ngay

$$u(x_1) = a_0,\ u(x_{N+1}) = a_1.$$

Vậy $$U(1) = a_0, \quad U(N+1) = a_1.$$

Trường hợp điểm trong. Với $i = \overline{2, N}$, ta sử dụng công thức xấp xỉ

$$\frac{u(x-h) - 2u(x) + u(x+h)}{h^2} \approx u''(x).$$

và thu được $$u(x_{i-1}) - 2u(x_i) + u(x_{i+1}) \approx f(x_i)h^2.$$

Vậy $$U(i-1) - 2U(i) + U(i+1) = f(x_i)h^2.$$

Tóm lại, ta có $$AU = b,$$

trong đó

$$A = \begin{pmatrix} 1 & 0 & & \dots & & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 \\ 0 & 1 & -2 & 1 & 0 & \dots & 0 \\ \dots & & & & & & \\ 0 & \dots & & 0 & 1 & -2 & 1 \\ 0 & & \dots & & 0 & 1 & \end{pmatrix}, \quad b = \begin{pmatrix} a_0 \\ f(x_2)h^2 \\ f(x_3)h^2 \\ \dots \\ f(x_N)h^2 \\ a_1 \end{pmatrix}.$$

Hệ phương trình này có nghiệm duy nhất $U = A^{-1}b$.

## CHƯƠNG TRÌNH MATLAB:

Xét bài toán

$$\begin{cases} u''(x) = -\pi^2 \sin(\pi x),\ x \in [0,1], \\ u(0) = 0,\ u(1) = 0. \end{cases}$$

Phương trình này có nghiệm chính xác là $u_{ex}(x) = \sin(\pi x)$. Chương trình dưới đây sẽ giải xấp xỉ phương trình trên bằng cách phân rã với N=20 và so sánh với nghiệm chính xác.

**Chương trình 8.1:** ***Các file dưới đây có trong thư mục laplace1d.***

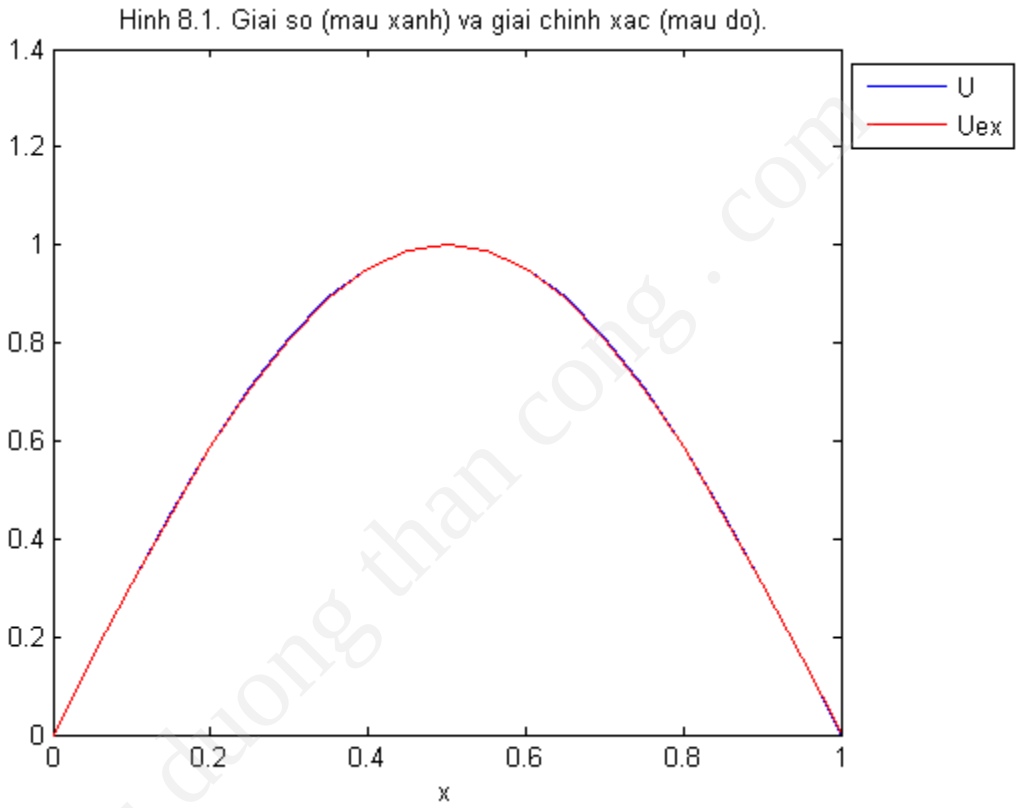
```
% file f.m
function a=f(x);
a=-pi^2*sin(pi*x);          %  Tao ham f.
```

```
% file uex.m
% Giai chinh xac U.
function a=uex(x);
a=sin(pi*x);                    % Tao ham uex.

% file laplace1d.m
% Chuong trinh giai phuong trinh Laplace mot chieu.
clear all                       %  Xoa tat ca cac gia tri cu.
N=20;                           % Chia doan [0, 1] thanh 20 doan.
h=1/N;                          % Khoang cach giua hai so (hay buoc nhay)
a0=0;
a1=0;                           % Gia tri bien.
X=[0:N]*h;                      % Tao N+1=21 diem chia.

% Tao ma tran A. Dau tien tra A ve ma tran zeros (N+1)x(N+1), sau do se % lan
luot gan cac vi tri dau, cuoi la 1 va cac gia tri con lai tu vong lap for.
A=zeros(N+1,N+1);
A(1,1)=1;
A(N+1,N+1)=1;
for i=2:N
   A(i,i-1)=1;
        A(i,i)=-2;
        A(i,i+1)=1;
end
% Tao ma tran b. Tuong tu voi tao ma tran A.
b=zeros(N+1,1);
b(1)=a0;
b(N+1)=a1;
for i=2:N
        b(i)=f(X(i))*h^2;
end
% Giai he AU=b.
U=A\b;                                          % Giai so U.
% Giai chinh xac Uex.
Uex=zeros(N+1,1);
for i=1:(N+1)
        Uex(i)=uex(X(i));
end;
% Ve do thi cua U va Uex tren cung mot hinh.
plot(X,U,'b',X,Uex,'r');
xlabel('x');
legend('U','Uex',-1);
```

**title('Hinh 8.1. Giai so (mau xanh) va giai chinh xac (mau do)'.);**

% Tinh sai so; err2: la sai so trong L^2 va errmax la sai so trong L^(vo cung)
**disp('error in norm L^2 and in norm sup')** % Hien thong bao trong (‘ ’).
**err2=norm(U-Uex)/sqrt(N+1)**
**errmax=norm(U-Uex,inf)**

Kết quả, chạy file laplace1d.m.




Sai số trong chuẩn L^2 và trong chuẩn sup là err2 = 0.0014, errmax =0.0021.

## 8.2. BÀI TOÁN PARABOLIC MỘT CHIỀU

### 8.2.1 Bài toán

Tìm hàm số u(x, t) trên $[0,1]\times[0,T]$ thỏa mãn

$$\begin{cases} \dfrac{\partial u}{\partial t} = a\dfrac{\partial u^2}{\partial x^2} + f(x,t), & \text{in } (x,t) \in (0,1)\times(0,T), \\ u(x,0) = u_0(x), & x \in [0,1], \\ u(0,t) = g_1(t), & t > 0, \\ u(1,t) = g_2(t), & t > 0. \end{cases}$$

### 8.2.2 Phân rã bài toán

Cho $M, N \in Z^+, h = 1/N = \Delta x, k = T/M = \Delta t$. Trên [0,1] lấy N+1 điểm $x_i = (i-1)h$, $i = \overline{1, N+1}$, và trên [0, T] lấy M+1 điểm $t_r = (r-1)k$, $r = \overline{1, M+1}$.

Ta sẽ tính xấp xỉ $U_r = [u(x_1, t_r), u(x_2, t_r), ..., u(x_{N+1}, t_r)]^T$.

Giả sử đã có $U_{r-1}$, ta tính $U_r$. Ký kiêu: $U_r(i) = u(x_i, t_r)$

Chú ý là $U_1$ đã được cho từ điều kiện đầu, $U_1(i) = u_0(x_i)$, $1 \le i \le N+1$.

Có hai trường hợp

Trường hợp điểm biên. Với $i = 1, i = N+1$, ta có ngay

$$u(x_1, t_r) = g_1(t_r), \quad u(x_{N+1}, t_r) = g_2(t_r).$$

Vậy $$U_r(1) = g_1(t_r), \quad U_r(N+1) = g_2(t_r), \quad 2 \le j \le M+1.$$

Trường hợp điểm trong. Với $i = \overline{2, N}$, ta sử dụng công thức xấp xỉ

$$\frac{\partial u}{\partial t}(x, t) \approx \frac{u(x, t) - u(x, t-k)}{k},$$

$$\frac{\partial^2 u}{\partial x^2}(x, t) \approx \frac{u(x+h, t) - 2u(x, t) + u(x-h, t)}{h^2}.$$

Suy ra

$$\frac{u(x, t) - u(x, t-k)}{k} \approx a\frac{u(x+h, t) - 2u(x, t) + u(x-h, t)}{h^2} + f(x, t).$$

Nhân hai vế cho $h^2$ và chuyển vế ta được

$$-au(x-h, t) + \left(2a + \frac{h^2}{k}\right)u(x, t) - au(x+h, t) \approx h^2 f(x, t) + \frac{h^2}{k}u(x, t-k).$$

$$-au(x_{i-1},t_r)+\left(2a+\frac{h^2}{k}\right)u(x_i,t_r)-au(x_{i+1},t_r)\approx h^2 f(x_i,t_r)+\frac{h^2}{k}u(x_i,t_{r-1}).$$

Hay, với $c=h^2/k+2a$,

$$-aU_r(i-1)+cU_r(i)-aU_r(i+1)\approx h^2 f(x_i,t_r)+\frac{h^2}{k}U_{r-1}(i).$$

Tóm lại ta có : $AU_r=b_r$ hay $AU=b$. Hệ có nghiệm duy nhất: $U=A^{-1}b$.

$$A=\begin{pmatrix} 1 & 0 & \dots & & & 0\\ -a & c & -a & 0 & \dots & 0\\ 0 & -a & c & -a & 0 & \dots & 0\\ \dots & & & & & & \\ 0 & \dots & 0 & -a & c & -a\\ 0 & & \dots & & 0 & 1 \end{pmatrix},\quad b_r=\begin{pmatrix} g_1(t_r)\\ f(x_2,t_r)h^2+U_{r-1}(2)\frac{h^2}{k}\\ \dots\\ f(x_N,t_r)h^2+U_{r-1}(N)\frac{h^2}{k}\\ g_2(t_r)\end{pmatrix}.$$

## CHƯƠNG TRÌNH MATLAB:

Xét bài toán cụ thể

$$\begin{cases} u_t=u_{xx} & \text{in } (x,t)\in(0,1)\times(0,T),\\ u(x,0)=\sin(\pi x), & x\in[0,1],\\ u(0,t)=u(1,t)=0, & t>0. \end{cases}$$

Phương trình này có nghiệm chính xác là $u_{ex}(x,t)=e^{-\pi^2}\sin(\pi x)$. Chương trình dưới đây giải số phương trình trên với T=0.1, bằng cách phân rã với $\Delta x=1/20,\ \Delta t=1/100$.

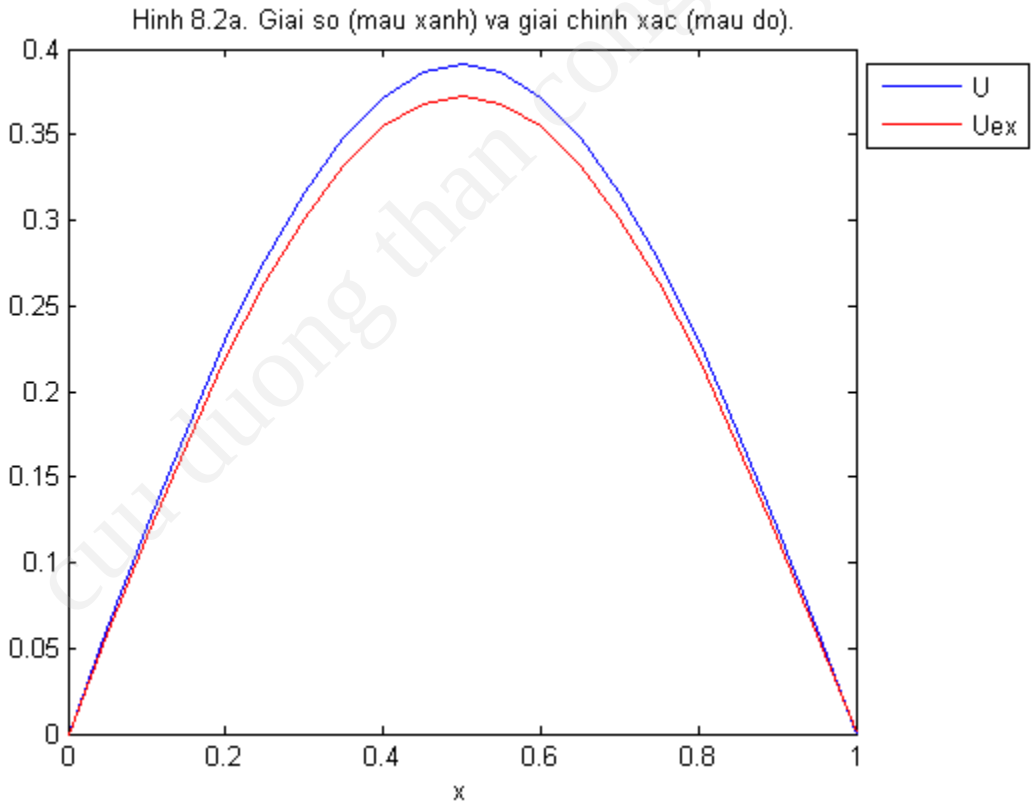
**Chương trình 8.2:** ***Các file dưới đây có trong thư mục Heat1d.***

```
% file f.m.
function a=f(x,t);
a=0;                                   %  Tao ham f(x,t).
% Dieu kien dau (file u0.m).
function a=u0(x);
a=sin(pi*x);                           %  Tao ham f(x,t).
% Dieu kien bien trai (file g1.m).
function a=g1(t);
a=0;                                   %  Tao ham g1(t).
% Dieu kien bien trai (file g2.m).
```

```
function a=g2(t);
a=0;                                        %  Tao ham g2(t).
%  Nghiem chinh xac (file uex.m).
function a=uex(x);
a=exp(-pi^2*t)*sin(pi*x);                   % Tao ham uex(x,t).

% Chuong trinh giai phuong trinh  Parabolic 1d.
%  Duoi day la file heat1d.m.
clear all;
T=0.1;
M=10;                   % Chia khoang thoi gian thanh 10 khoang.
k=T/M;                  % Khoang cach giua hai thoi diem lien tiep .
N=20;                   % Chia khoang khong gian thanh 20 khoang.
h=1/N;                  % Khoang cach giua hai diem lien tiep (buoc nhay).
X=[0:N]*h;              %  Luoi khong gian.
R=[0:M]*k;              %  Luoi thoi gian.
a=1;                    % He so trong pt dau.
c=h^2/k+2*a;
% Tao vecto dieu kien dau.
U=zeros(N+1,1);
for i=1:(N+1)
   U(i)=u0(X(i));
end
% May tinh giai so U tai t=T.
for r=2:(M+1)
        % Tao ma tran A va vector b
        A=zeros(N+1,N+1);
        b=zeros(N+1,1);
                A(1,1)=1;
                b(1)=g1(R(r));
                A(N+1,N+1)=1;
                b(N+1)=g2(R(r));
        for i=2:N
                A(i,i)=c;
                A(i,i-1)=-a;
                A(i,i+1)=-a;
     b(i)=f(X(i),R(r))*h^2+U(i)*h^2/k;          % Giai U tai t=R(r-1).
end
%solve AU=b
        U=A\b;                  % Giai U tai t=R(r).
end
% Giai chinh xac Uex.
```

```
Uex=zeros(N+1,1);
for i=1:(N+1)
        Uex(i)=uex(X(i),T);
end;
% Ve do thi giai so va giai chinh xac tren cung mot hinh.
plot(X,U,'b',X,Uex,'r');
xlabel('x');
title(' Hinh 8.2a. Giai so (mau xanh) va giai chinh xac (mau do).');
legend('U','Uex',-1);          % Dua ghi chu vao do thi.

% Tinh cac sai so: err2 trong L^2 va errmax trong L^(vo cung).
disp('Sai so trong chuan L^2 va trong chuan sup')
err2=norm(U-Uex)/sqrt(N+1)
errmax=norm(U-Uex,inf)
```

Kết quả chạy file heat1d.m trong thư mục Heat1d.



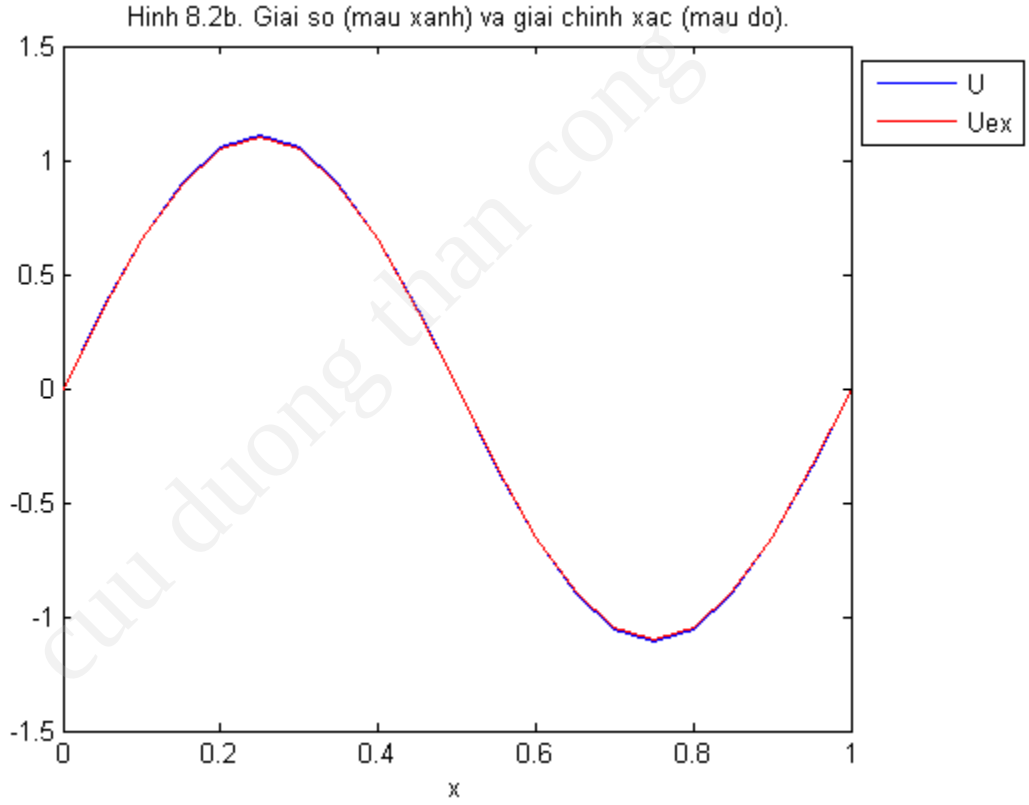

Sai số trong chuẩn L^2 và trong chuẩn sup là err2 =0.0125, errmax =0.0182.

Các bài toán tương tự ta có thể sử dụng chương trình này để giải, chỉ cần thay thế các dữ kiện cho phù hợp với đầu bài. Cụ thể, xét bài toán

$$\begin{cases} u_t = u_{xx} + (1+4\pi^2)e^t \sin(2\pi x) & \text{in } (x,t) \in (0,1)\times(0,1), \\ u(x,t) = 0 & \text{on } (x,t) \in [0,1]\times\{0,1\}, \\ u(x,0) = \sin(2\pi x) & \text{on } [0,1]. \end{cases}$$

Phương trình này có nghiệm chính xác là $u_{ex}(t,x) = e^t \sin(2\pi x)$. Hãy tính xấp xỉ $u(0.1,x)$ phương trình trên bằng cách phân rã với $\Delta x = 1/20, \Delta t = 1/100$. Dùng chương trình 8.2 để giải, ta giữ nguyên đoạn chương trình giải phương trình Parabolic 1d (tức là giữ nguyên file heat1d.m) và thay đổi công thức tính hàm cho phù hợp với bài toán.

Kết quả chạy file heat1d.m trong thư mục Heat1d-1.




Sai số trong chuẩn L^2 và trong chuẩn sup là err2=0.0060, errmax=0.0087.

Chương trình chi tiết xin xem trong thư mục Heat1d-1.

Các file tạo hàm của bài toán này được viết như sau

% file f.m.

**function a=f(x,t);**

```
a=(1+4*pi^2)*exp(t)*sin(2*pi*x);          % Tao ham f(x,t).
% Dieu kien dau (file u0.m).
function a=u0(x);
a=sin(2*pi*x);                            % Tao ham f(x,t).
% Dieu kien bien trai (file g1.m).
function a=g1(t);
a=0;                                      % Tao ham g1(t).
% Dieu kien bien trai (file g2.m).
function a=g2(t);
a=0;                                      % Tao ham g2(t).
%  Nghiem chinh xac (file uex.m).
function a=uex(x);
a=exp(t)*sin(2*pi*x);                     % Tao ham uex.
```

Giữ nguyên file heat1d.m.

## 8.3. BẬC HỘI TỤ VÀ ĐIỀU KIỆN BIÊN NEUMANN CỦA BÀI TOÁN MỘT CHIỀU

### 8.3.1 Bậc hội tụ

Bây giờ ta quan tâm đến sai số giữa lời giải số và lời giải chính xác. Thông thường sai số này được tính theo chuẩn $L^2$ hoặc $L^\infty$. Câu hỏi là sự hội tụ của sai số về 0 phụ thuộc thế nào tới các mesh-sizes $h = 1/N = \Delta x$ và $k = T/M = \Delta t$? Trong hầu hết các trường hợp, ta có:

- Bài toán trong 8.1.1 có bậc hội tụ là $O(h^2)$, trong đó $O(h^2) \le \text{const}.h^2$ khi $h \to 0$. Lý do là ta đã dùng công thức xấp xỉ

$$\frac{u(x-h)-2u(x)+u(x+h)}{h^2} \approx u''(x).$$

- Bài toán trong 8.2.1 có bậc hội tụ là $O(\max\{k, h^2\})$. Lý do là ta đã sử dụng hai xấp xỉ

$$\frac{\partial u}{\partial t}(x,\, t) \approx \frac{u(x,\ t)-u(x,\ t-k)}{k},$$

$$\frac{\partial^2 u}{\partial x^2}(x,\, t) \approx \frac{u(x+h,\, t)-2u(x,\, t)+u(x-h,\, t)}{h^2}.$$

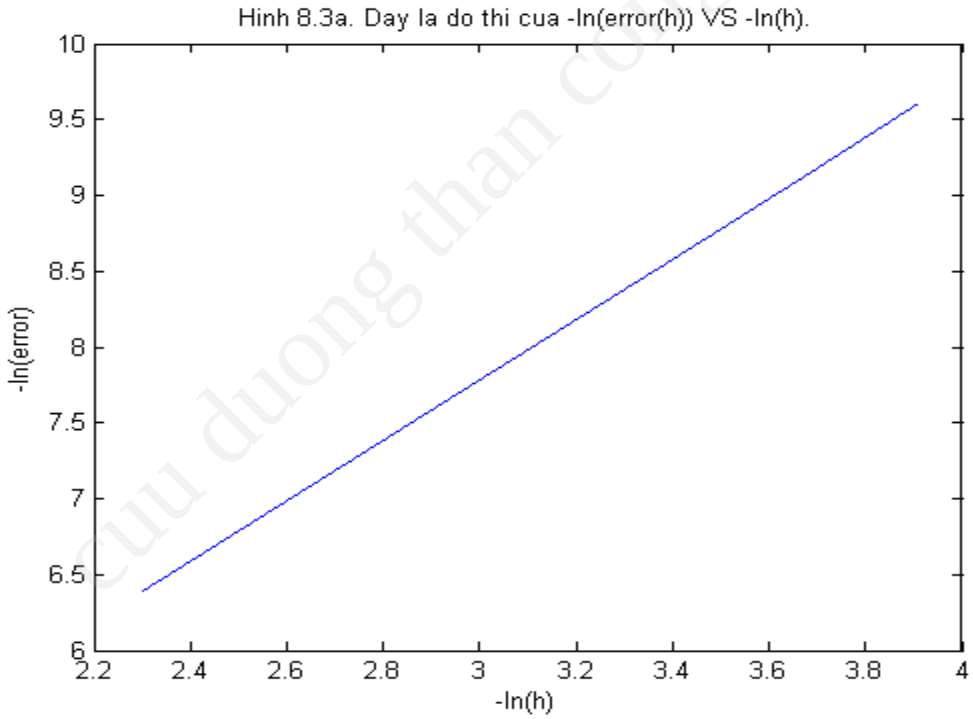
Trong đó xấp xỉ thứ nhất có bậc $O(k)$, xấp xỉ thứ hai có bậc $O(h^2)$. Như vậy, ta nên chọn $k = ch^2$, với c là hằng số dương không quá lớn, để bậc hội tụ cũng là $O(h^2)$.

- **Kiểm tra bậc hội tụ bằng MATLAB**

Hàm số s(h) thuộc $O(h^2)$ nghĩa là: $|s(h)| \le \text{const}.h^2$ khi $h \to 0$.

Điều này tương đương với: $-\ln(|s(h)|) \ge -2\ln(h) + \text{const}$ khi $h \to 0$.

Ở đây các dấu – được thêm vào để thu được các số dương. Ta có thể kiểm tra bất đẳng thức thứ hai bằng cách vẽ đồ thị của $-\ln(|s(h)|)$ theo $-\ln(h)$ khi $h \to 0$. Nếu đồ thị này có dạng đường thẳng thì hệ số góc của đường thẳng chính là hệ số hội tụ (xem hình vẽ minh họa, Hình 8.3a.)




## 8.3.2 Điều kiện biên Neumann

Ta xét bài toán trong 8.1.1 với điều kiện biên phức tạp hơn

$$\begin{cases} u''(x) = f(x), x \in [0,1], \\ u'(0) = a_0,\ u(1) = a_1. \end{cases}$$

Khi đó, ta cần có sự điều chỉnh để xây dựng ma trận A và vector b. Chỗ khác biệt là tại điểm $x_1 = 0$ ta phải sử dụng một công thức xấp xỉ. Ta có hai cách làm.

**Cách 1.** Ta có thể sử dụng xấp xỉ đơn giản (Taylor bậc 1)

$$u(x_2) - u(x_1) \approx u'(x_1) = a_0 h.$$

Khi đó, $A = \begin{pmatrix} -1 & 1 & & \dots & & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 \\ 0 & 1 & -2 & 1 & 0 \dots & 0 \\ \dots & & & & & \\ 0 & \dots & & 0 & 1 & -2 & 1 \\ 0 & & \dots & & & 0 & 1 \end{pmatrix}$, $b = \begin{pmatrix} a_0 h \\ f(x_2)h^2 \\ f(x_3)h^2 \\ \dots \\ f(x_N)h^2 \\ a_1 \end{pmatrix}$

Tuy nhiên, cách làm này cho bậc hội tụ chỉ là $O(h)$.

**Cách 2.** Dùng khai triển Taylor bậc 2 ta có

$$u(x_2) - u(x_1) = u'(x_1)h + u''(x_1)\frac{h^2}{2} + O(h^2) \approx a_0 h + f(x_1)\frac{h^2}{2}..$$

Vậy

$$A = \begin{pmatrix} -1 & 1 & & \dots & & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 \\ 0 & 1 & -2 & 1 & 0 & \dots & 0 \\ \dots & & & & & & \\ 0 & \dots & & 0 & 1 & -2 & 1 \\ 0 & & \dots & & & 0 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} a_0 h + f(x_1)\frac{h^2}{2} \\ f(x_2)h^2 \\ f(x_3)h^2 \\ \dots \\ f(x_N)h^2 \\ a_1 \end{pmatrix}.$$

Cách làm này cho bậc hội tụ là $O(h^2)$.

## CHƯƠNG TRÌNH MATLAB :

Xét phương trình sau trên (0,1)

$$\begin{cases} u - u'' = (1+\pi^2)\cos(\pi x), \\ u(0) = 1, u(1) = -1. \end{cases}$$

- Viết hàm solve.m nhận dữ liệu N, giải xấp xỉ bằng cách phân rã ứng với $\Delta x = 1/N$, trả về lời giải xấp xỉ và sai số (chuẩn sup) với lời giải chính xác (là $\cos(\pi x)$).
- Cho N=10. Vẽ đồ thị lời giải số và lời giải chính xác trên cùng 1 hình vẽ, và tính sai số (trong chuẩn sup)
- Cho N=10, 20, 30, 40, 50, vẽ đồ thị các sai số để kiểm tra bậc hội tụ.

Chúng tôi trình bày cách phân rã bài toán và phần chương trình để kiểm tra bậc hội tụ (vẽ Hình 8.3a và Hình 8.3b). Chương trình đầy đủ xin xem trong thư mục bachoitu1d.

**Phân rã bài toán.** Cho $N \in Z^+, h = 1/N = \Delta x$. Ta chia đoạn [0,1] thành N đoạn độ dài h bởi N+1 điểm $x_i = (i-1)h$, $i = 1,2,,,N+1$.

Ta sẽ tính xấp xỉ $U = [u(x_1),...,u(x_{N+1})]^T$. Kí hiệu: $U(i) = u(x_i)$.

Trường hợp điểm biên. Với $i = 1$, $i = N+1$ ta có ngay

$$u(x_1) = a_0,\ u(x_{N+1}) = a_1.$$

Vậy $$U(1) = a_0, \quad U(N+1) = a_1.$$

Trường hợp điểm trong. Với $i = \overline{2, N}$, ta sử dụng công thức xấp xỉ

$$\frac{u(x-h) - 2u(x) + u(x+h)}{h^2} \approx u''(x)$$

Suy ra, $$\frac{-u(x-h) + (2+h^2)u(x) - u(x+h)}{h^2} \approx u(x) - u''(x) = f(x)$$

và thu được $$-u(x_{i-1}) + (2+h^2)u(x_i) - u(x_{i+1}) \approx f(x_i)h^2.$$

Vậy $$-U(i-1) + (2+h^2)U(i) - U(i+1) = f(x_i)h^2.$$

hay $$U(i-1) - (2+h^2)U(i) + U(i+1) = -f(x_i)h^2$$

Tóm lại, ta có

$$AU = b,$$

$$A=\begin{pmatrix} 1 & 0 & & \dots & & 0 \\ 1 & \lambda & 1 & 0 & \dots & 0 \\ 0 & 1 & \lambda & 1 & 0 \dots & 0 \\ \dots & & & & & \\ 0 & & \dots & & 0\ 1\ \lambda & 1 \\ 0 & & & \dots & & 0\ 1 \end{pmatrix}, \quad b=\begin{pmatrix} a_0 \\ -f(x_2)h^2 \\ -f(x_3)h^2 \\ \dots \\ -f(x_N)h^2 \\ a_1 \end{pmatrix}.$$

Trong đó, a0=1, a1= -1, $\lambda=-(2+h^2)$ và $f(x)=(1+\pi^2)\cos(\pi x)$.

Hệ phương trình này có nghiệm duy nhất $U=A^{-1}b$.

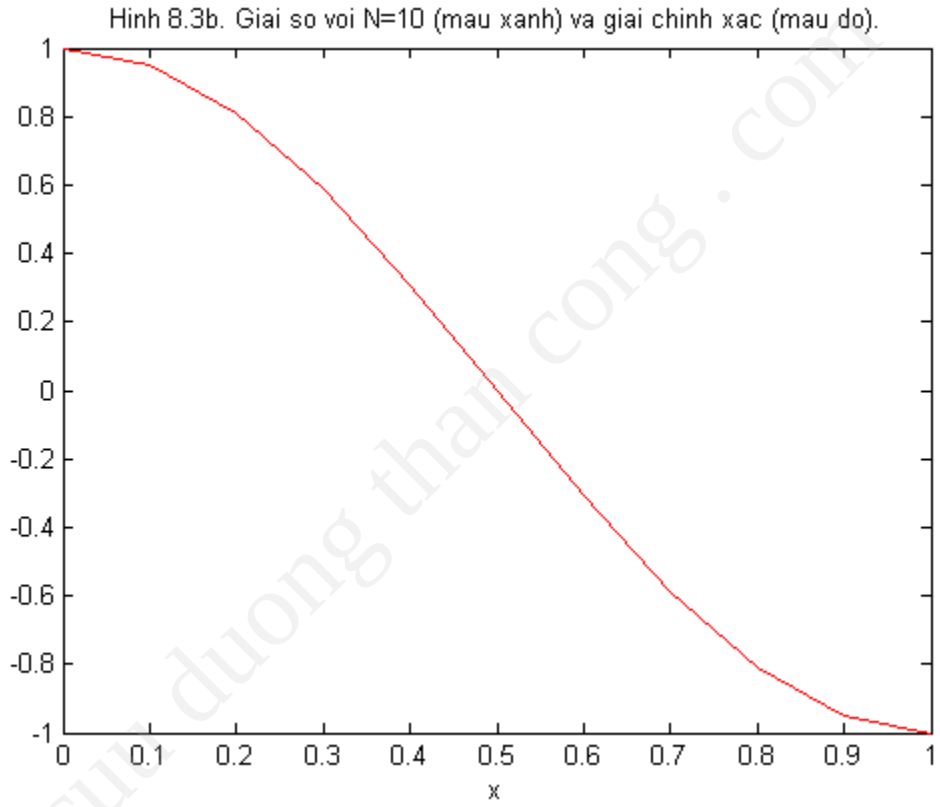
**Chương trình 8.3 :**

Chương trình đầy đủ xin xem trong thư mục bachoitu1d. Ở đây chúng tôi chỉ trình bày chương trình tạo file plotsame.m

```
% Chuong trinh kiem tra bac hoi tu.
% Tao file plotsame.m
clear all
global a0
global a1                    % a0, a1 la bien toan cuc.
a0=1; a1=-1;
% Giai so doi voi N=10
N=10;
X=[0:N]*1/N;                 % Tao luoi diem.
[U,errmax]=solve(N);         % Giai so U va sai so, nam trong file solve.m.
Uex=zeros(N+1,1);            % Giai chinh xac.
for i=1:(N+1)
   Uex(i)=uex(X(i));
end
figure(1)
plot(X,U,'b',X,Uex,'r');
xlabel('x');
title('Hinh 8.3b. Giai so voi N=10 (mau xanh) va giai chinh xac (mau do).');
disp('error in sup norm');
errmax
pause                        % Tam dung, cho den khi go phim bat ky.
% Tinh cac sai so voi N=10, 20, 30, 40, 50.
M=5;
numN=[1:M]*10;
err=zeros(1,M);
for i=1:5
   [U,err(i)]=solve(numN(i));
```

```
end
% Do thi error VS mesh-size
figure(2)
plot(log(numN),-log(err))
xlabel('-ln(h)');
ylabel('-ln(error)');
title('Hinh 8.3a. Day la do thi cua -ln(error(h)) VS -ln(h).');
```

Kết quả chạy file plotsame.m ta được Hình 8.3a. và Hình 8.3b.




Sai số errmax =0.0017.

Cách xử lý điều kiện biên Neumann cho bài toán Parabolic cũng tương tự.

Ngoài ra, trong bài toán Parabolic nếu từ công thức xấp xỉ

$$\frac{u(x,\ t)-u(x,\ t-k)}{k} \approx a\frac{u(x+h,t)-2u(x,t)+u(x-h,t)}{h^2}+f\left(x,t\right).$$

Nhân hai vế k rồi chuyển vế, ta được

$$-\frac{ka}{h^2}u\left(x-h,t\right)+\left(1+\frac{2ka}{h^2}\right)u\left(x,t\right)-\frac{ka}{h^2}u\left(x+h,t\right)\approx kf\left(x,t\right)+u\left(x,t-k\right).$$

Hay,

$$-cu(x_{i-1}, t_r)+(1+2c)u(x_i, t_r)-cu(x_{i+1}, t_r)\approx kf(x_i, t_r)+u(x_i, t_{r-1}),\quad c=ka/h^2.$$

Vậy, $$-cU_r(i-1)+(1+2c)U_r(i)-cU_r(i+1)\approx kf(x_i, t_r)+U_{r-1}(i).$$

Tóm lại ta có : $AU_r=b_r$ hay $AU=b$. Hệ có nghiệm duy nhất $U=A^{-1}b$.

trong đó $c=ka/h^2$,

$$\begin{pmatrix} 1 & 0 & 0 & \dots & & 0\\ -c & 1+2c & -c & 0 & \dots & 0\\ 0 & -c & 1+2c & -c & 0 & \dots & 0\\ \dots & & & \ddots & \dots & \vdots\\ 0 & \dots & 0 & \dots & -c & 1+2c & -c\\ 0 & & \dots & & 0 & 0 & 1 \end{pmatrix},\quad b_r=\begin{pmatrix} g_1(t_r)\\ f(x_2, t_r)k+U_{r-1}(2)\\ \dots\\ f(x_N, t_r)k+U_{r-1}(N)\\ g_2(t_r) \end{pmatrix}.$$

Khi đó trong chương trình 8.2 ta chỉ cần thay đoạn chương trình tạo ma trận A và vector b như sau

```
c=a*k/h/h;
% May tinh giai so U tai t=T.
for r=2:(M+1)
    % Tao ma tran A va vector b
    A=zeros(N+1,N+1);
    b=zeros(N+1,1);
        A(1,1)=1; b(1)=g1(R(r)); A(N+1,N+1)=1;b(N+1)=g2(R(r));
    for i=2:N
        A(i,i)=1+2*c; A(i,i-1)=-c; A(i,i+1)=-c;
  b(i)=f(X(i),R(r))*k+U(i);end                    % Giai U tai t=R(r-1).
%solve AU=b
    U=A\b;              % Giai U tai t=R(r).
end
```

Chương trình chi tiết xin xem trong thư mục Heat1d-2 và Heat1d-3.

## 8.4. BÀI TOÁN LAPLACE 2 CHIỀU

### 8.4.1 Bài toán

Tìm hàm số u(x,y) trên $\Omega=(0,1)\times(0,1)$ thỏa mãn

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} = f(x, y) & \text{in } \Omega, \\ u = g(x, y) & \text{on } \partial\Omega. \end{cases}$$

### 8.4.2 Phân rã bài toán

Cho $N \in Z^+, h = 1/N$. Ta chia $\overline{\Omega}$ thành các ô vuông cạnh h bởi $(N+1)^2$ điểm

$$x_{ij} = ((j-1)h, (N+1-i)h),\ i, j = 1, 2, ..., N+1.$$

Ta sẽ tính xấp xỉ $U = (u(x_{ij}))_{ij}$. Có hai điểm quan trọng,

- **Đánh lại chỉ số.** Để viết lại U như một vector, ta cần một song ánh

$$\{1, 2, ..., N+1\} \times \{1, 2, ..., N+1\} \to \{1, 2, ..., (N+1)^2\}.$$

Một cách đơn giản là sử dụng

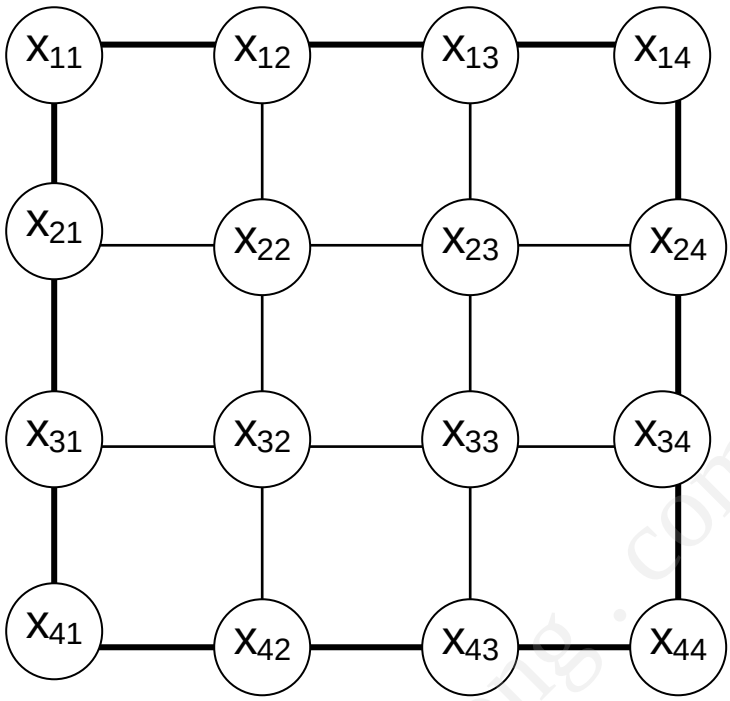
$$\text{change(N},i, j) = (N+1)(i-1) + j.$$

**Ghi chú.** Cách đánh số này tương ứng với thứ tự « từ điển », tức là

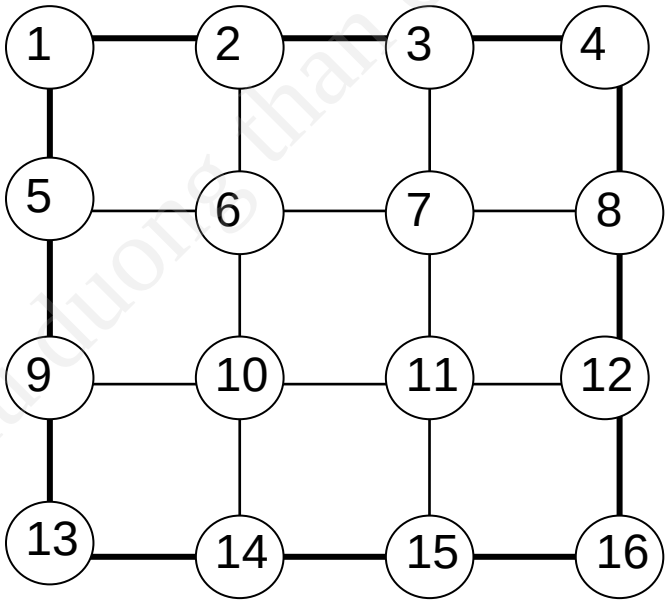
$$(1,1) < (1,2) < ... < (1, N+1) < (2,1) < (2,2) < ... < (N+1, N+1).$$

Mặt khác, ta cũng có thể sử dụng bất cứ song ánh nào khác. Ta sẽ viết lại bài toán dưới dạng : $AU = b$, trong đó A là ma trận $(N+1)^2 \times (N+1)^2$.

Minh họa vị trí các điểm $x_{ij}$ trên hình vuông $[0,1] \times [0,1]$




Minh họa cách đánh số lại




- **Áp dụng công thức Taylor.** Với $x_{ij} \in \Omega$, ta có

$$u(x_{i+1,j}) + u(x_{i-1,j}) + u(x_{i,j+1}) + u(x_{i,j-1}) - 4u(x_{ij}) \approx \Delta u(x_{ij})h^2.$$

## CHƯƠNG TRÌNH MATLAB:

Xét bài toán

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2}+\dfrac{\partial^2 u}{\partial y^2}=-2\pi^2\sin(\pi x)\sin(\pi y), \quad 0<x,y<1, \\ u(x,y)=0, \quad x=0,1 \; or \; y=0,1. \end{cases}$$

(lời giải chính xác là $u_{ex}(x,y)=\sin(\pi x)\sin(\pi x).$)

**Chương trình 8.4:**

```
% file f.m ; Tao ham f(x, y).
function a=f(x,y)
a=-2*pi^2*sin(pi*x)*sin(pi*y);
% file uex.m ; tao ham uex(x,y), giai chinh xac.
function a=uex(x,y)
a=sin(pi*x)*sin(pi*y);
% file change.m; tao ham change(N,i,j), danh lai chi so diem.
function num=change(N,i,j)
num=(N+1)*(i-1)+j;
% Chuong trinh tao file laplace2d.m.
clear all;
N=20;h=1/N;
% Tao luoi.
X=[0:N]*h;  Y=1-X;

% Tao ma tran A va vector b.
A=zeros((N+1)^2,(N+1)^2);
b=zeros((N+1)^2,1);
for i=1:(N+1)
   for j=1:(N+1)
      id=change(N,i,j);              % chi so cua diem (i,j)
      idlelf=change(N,i,j-1);        % chi so cua diem nam ben trai diem (i,j)
      idright=change(N,i,j+1);       % chi so cua diem nam ben phai (i,j)
      idup=change(i-1,j);            % chi so cua diem nam tren (i,j)
      iddown=change(N,i+1,j);        % chi so cuadiem nam duoi (i,j)
      if ((i>1)&&(i<N+1)&&(j>1)&&(j<N+1))
            A(id,id)=-4;
        A(id,idlelf)=1;
        A(id,idright)=1;
        A(id,idup)=1;
        A(id,iddown)=1;
         b(id)=f(X(j),Y(i))*h^2;
      else
         A(id,id)=1;
```

```
            b(id)=g(X(j),Y(i)); end
    end
end

% Day la phep giai so
U=A\b;                  % Giai he AU=b

% Ve do thi cua loi giai so.
% Dong nhat vector U voi matran UU thong qua cach danh lai chi so.
UU=zeros(N+1,N+1);
for i=1:(N+1)
    for j=1:(N+1)
        UU(i,j)=U(change(N,i,j)); end
end
figure(1)
surfc(X,Y,UU);
xlabel('x');
ylabel('y');
title('Hinh 8.4a. Day la phep giai so.')
pause
% Ve do thi cua loi giai chinh xac.
% Tinh toan ma tran UUex tuong ung voi phep giai chinh xac.
UUex=zeros(N+1,N+1);
for i=1:(N+1)
    for j=1:(N+1)
        UUex(i,j)=uex(X(j),Y(i)); end
end
figure(2)
surfc(X,Y,UUex);
xlabel('x');
ylabel('y');
title('Hinh 8.4b. Day la phep giai chinh xac.');
% Tinh sai so.
% Tinh toan ma tran Uex tuong ung voi phep giai chinh xac.
Uex=zeros((N+1)^2,1);
for i=1:(N+1)
    for j=1:(N+1)
        Uex(change(N,i,j))=UUex(i,j); end
end
disp('error in norm sup')
errmax=norm(U-Uex,inf)
```

**Chú thích**

- Minh họa cho các chỉ số

**idup**

**idleft** **id** **idright**

**iddown**

- Dòng A(id, .) của ma trận A tương ứng với phương trình phân rã tại điểm $id \equiv (i, j)$, tức là tại điểm $x_{ij}$ với change$(N, i, j) = id$.

Kết quả, chạy file **laplace2.m** để giải bài toán trên với N=20 là Hình 8.4a và Hình 8.4b; với sai số trong chuẩn sup là errmax =0.0021.

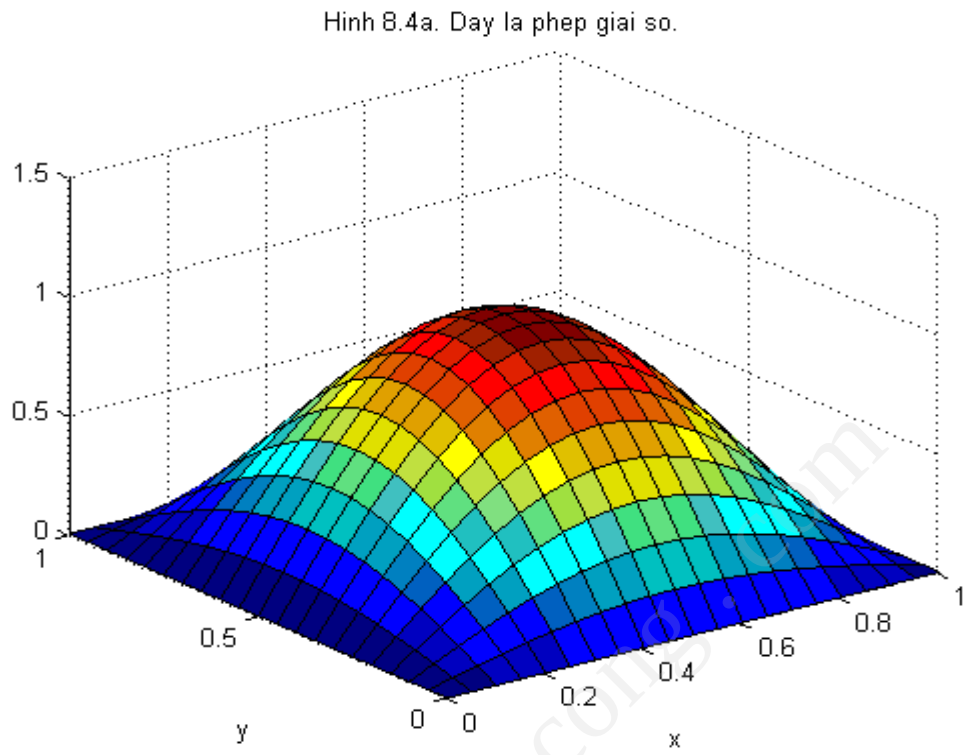
Trong chương trình 8.4 này ta chỉ cần thay đoạn chương trình tạo file f.m và uex.m thì sẽ được lời giải của bài toán mới với đồ thị là Hình 8.4c và Hình 8.4d. Lời giải chi tiết xin xem trong thư mục Laplace2d-1.

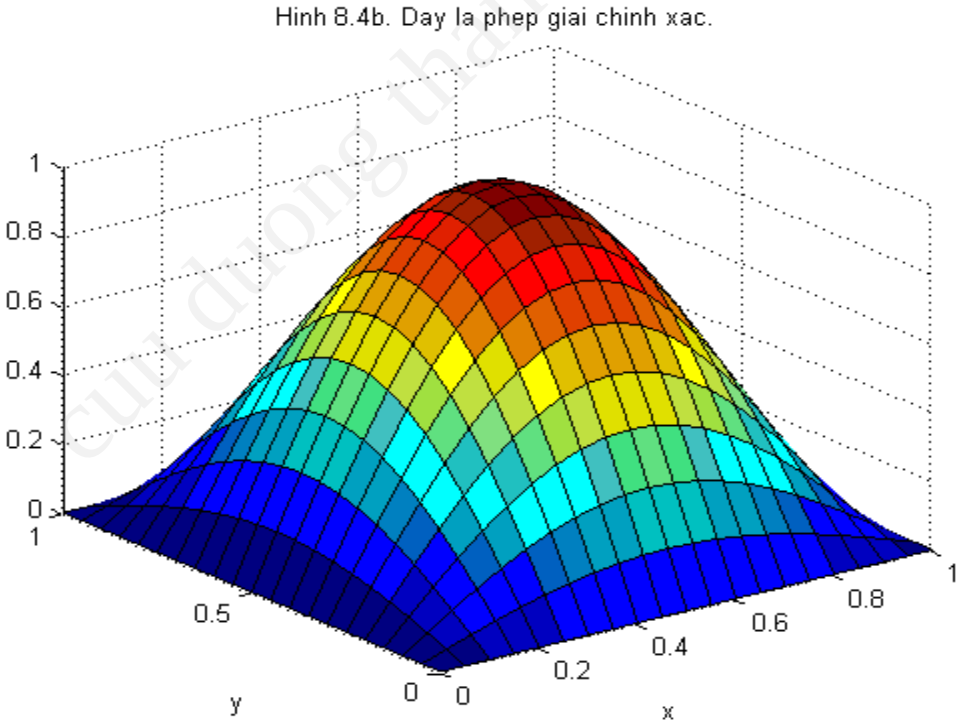
Bài toán được xét ở đây là

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} = 16\cos^2(\pi x)\pi^2 - 8\pi^2 + 4\sin(\pi x)\pi^2 + 48y - 24, \quad 0 < x, y < 1, \\ u(x, y) = 0, \quad x = 0,1 \text{ or } y = 0, 1. \end{cases}$$

(lời giải chính xác là $u_{ex}(x, y) = (2\sin(\pi x) - 1)^2 + (2y - 1)^3$

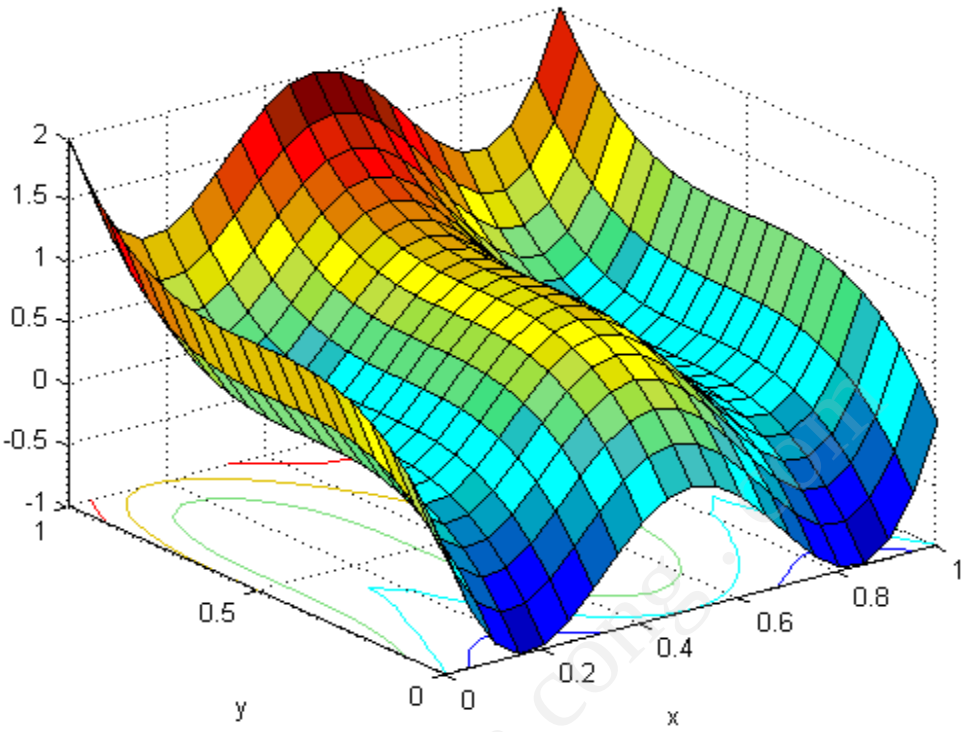
```
% file f.m ; Tao ham f(x, y).
function a=f(x,y)
a=16*cos(pi*x)^2*pi^2-8*pi^2+4*sin(pi*x)*pi^2+48*y-24;
% file uex.m ; tao ham uex(x,y), giai chinh xac.
function a=uex(x,y)
a=(2*sin(pi*x)-1)^2+(2*y-1)^3;
```



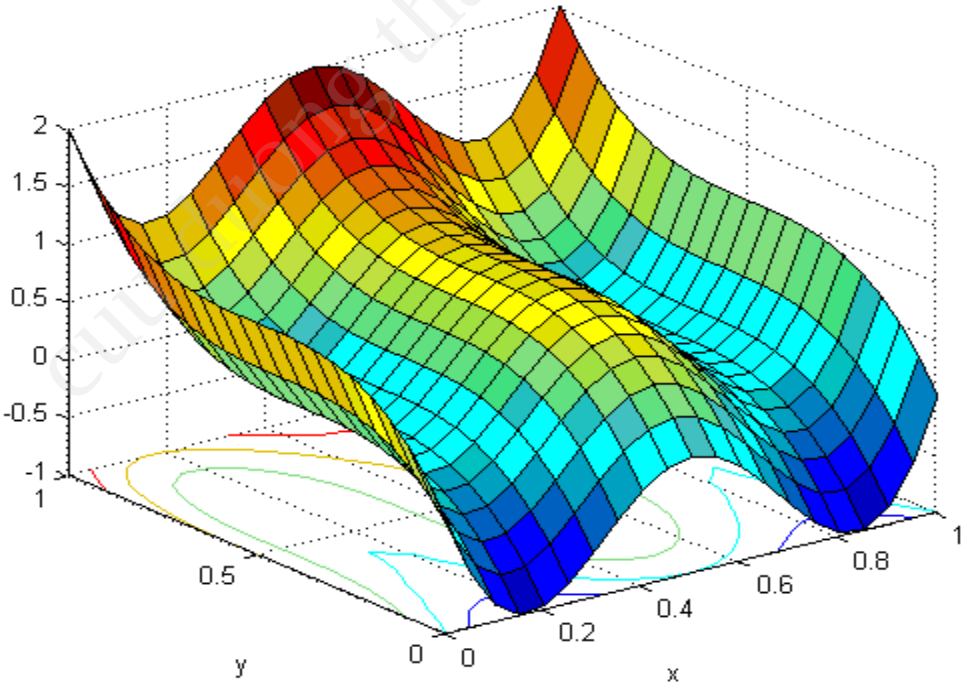




Sai số trong chuẩn sup là errmax =0.0021.

Hinh 8.4c. Day la do thi cua phep giai so.




Hinh 8.4d. Day la do thi cua phep giai chinh xac.




Sai số trong chuẩn sup là errmax =0.0168.

## 8.5. BÀI TOÁN PARABOLIC 2 CHIỀU

### 8.5.1 Bài toán

Tìm hàm số u(x,y,t) trên $\Omega\times[0,T]$, $\Omega=(0,1)\times(0,1)$, thỏa mãn

$$\begin{cases} u_t = a\Delta u + f(x,\ y,\ t), & in\ (x,\ y,\ t)\in\Omega\times[0,T], \\ u(x,\ y,\ t) = g(x,\ y,\ t), & on\ \partial\Omega\times[0,T], \\ u(x,\ y,\ 0) = u_0(x,\ y). & on\ \overline{\Omega}. \end{cases}$$

### 8.5.2 Phân rã bài toán

Không giảm tính tổng quát, ta giải bài toán trên với a=1.

Cho $M,N\in Z^+$, $h=1/N$, $k=T/M$. Trên hình vuông $\Omega$ lấy $(N+1)^2$ điểm $(x_i,\ y_j)=\{(i-1)h,(j-1)h\}$, $1\le i,\ j\le N+1$. Trên [0,T] lấy (M+1) điểm $t_r=(r-1)k,\ r=1,\ 2,...,\ M+1$. Ta sẽ sẽ tính xấp xỉ $U_r=\{u(x_i,\ y_j,\ t_r)\}_{1\le i,\ j\le N+1}$. Kí hiệu $U_{i,\ j,\ r}=u(x_i,\ y_j,\ t_r)$. Khi đó, nghiệm xấp xỉ $U_r$ được viết dưới dạng ma trận (N+1)x(N+1) là

$$U_r=\begin{pmatrix} U_{1,1,r} & \cdots & U_{1,N+1,r} \\ \vdots & \ddots & \vdots \\ U_{N+1,1,r} & \cdots & U_{N+1,N+1,r} \end{pmatrix}.$$

Điều kiện đầu $u(x,\ y,\ 0)=u0(x,\ y)$, $0\le x,\ y\le 1$, với $u0_{i,j}=u0(x_i,y_j)$,

$$U_1=\begin{pmatrix} u0_{1,1} & \cdots & u0_{1,N+1} \\ \vdots & \ddots & \vdots \\ u0_{N+1,1} & \cdots & u0_{N+1,N+1} \end{pmatrix}.$$

Ta dùng công thức xấp xỉ

$$u_t \approx \frac{u(x,\, y,\, t+k)-u(x,\, y,\, t)}{k},$$

$$u_{xx} \approx \frac{u(x+h,\, y,\, t)-2u(x,\, y,\, t)+u(x-h,\, y,\, t)}{h^2},$$

$$u_{yy} \approx \frac{u(x,\, y+h,\, t)-2u(x,\, y,\, t)+u(x,\, y-h,\, t)}{h^2}.$$

Khi đánh số lại bởi quy tắc $\text{change}\big(N,\, i,\, j\big)=(N+1)(i-1)+j$, thì

$$x_{ij}=\big(x_i,\, y_j\big),\quad 1\le i,\, j\le N+1 \text{ và } U_r=(u(x_{\text{ij}},\, t_r))_{\text{ij}}.$$

Khi đó, với $x_{ij}\in\Omega$ ta có

$$\frac{u(x_{ij},\, t_r)-u(x_{ij},\, t_{r-1})}{k}\approx u_t(x_{ij},\, t_r),$$

và

$$u(x_{i+1,\,j},\, t_r)+u(x_{i-1,\,j},\, t_r)+u(x_{i,\,j+1},\, t_r)+u(x_{i,\,j-1},\, t_r)-4u(x_{ij},\, t_r)\approx \Delta u(x_{ij},\, t_r)h^2,$$

suy ra

$$\big(h^2/k+4\big)u(x_{ij},\, t_r)-u(x_{i+1,\,j},\, t_r)-u(x_{i-1,\,j},\, t_r)-u(x_{i,\,j+1},\, t_r)-u(x_{i,\,j-1},\, t_r)$$
$$\approx f(x_{ij},\, t_r)h^2+u(x_{ij},\, t_{r-1}).h^2/k.$$

Như vậy bài toán được viết lại dưới dạng $AU=b$.

## CHƯƠNG TRÌNH MATLAB:

Giải số u(x,y,1) của bài toán:

$$\begin{cases} u_t-\Delta u=(1+2\pi^2)e^t\sin(\pi x)\sin(\pi y), & in\ (x,\, y,\, t)\in\Omega\times[0,T], \\ u(x,\, y,\, t)=0, & on\ \partial\Omega\times[0,T], \\ u(x,\, y,\, 0)=\sin(\pi x)\sin(\pi y). & on\ \overline{\Omega}. \end{cases}$$

Lời giải chính xác là $u_{\text{ex}}(x,\, y,\, t)=e^t\sin(\pi x)\sin(\pi y)$.

**Chương trình 8.5:** ***Các file có trong thư mục Heat2d***

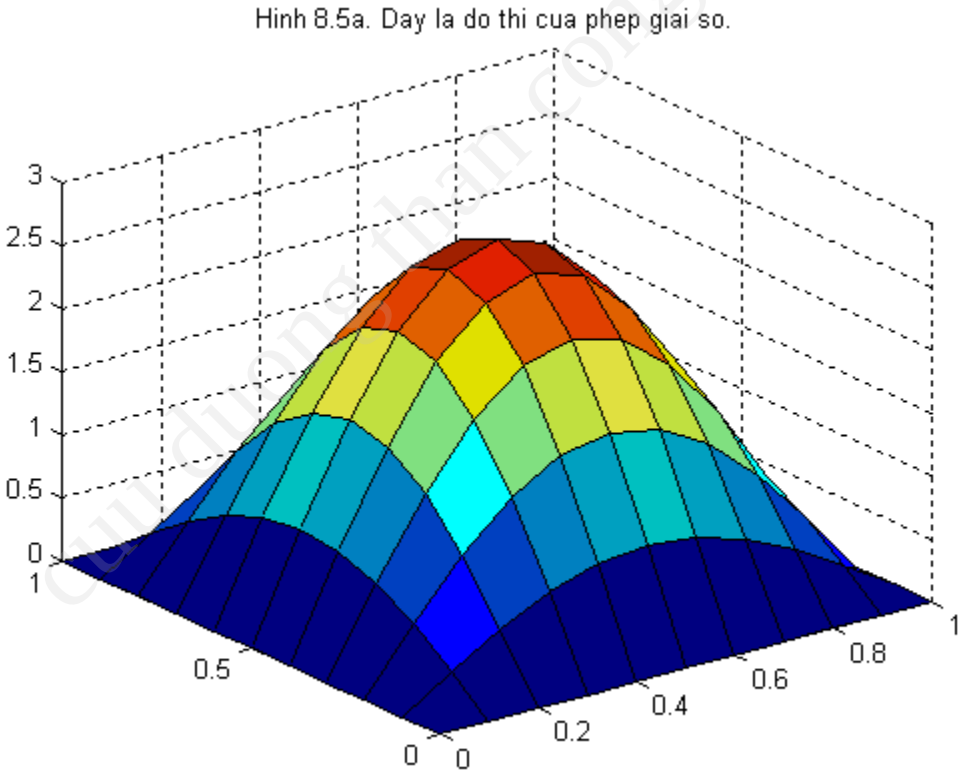
```
% file f.m ; Tao ham f(x,y,t).
function a=f(x,y,t)
a=(1+2*pi^2)*exp(t)*sin(pi*x)*sin(pi*y);
% file uex.m; Tao ham chinh xac .
function a=uex(x,y,t)
a=exp(t)*sin(pi*x)*sin(pi*y);
```

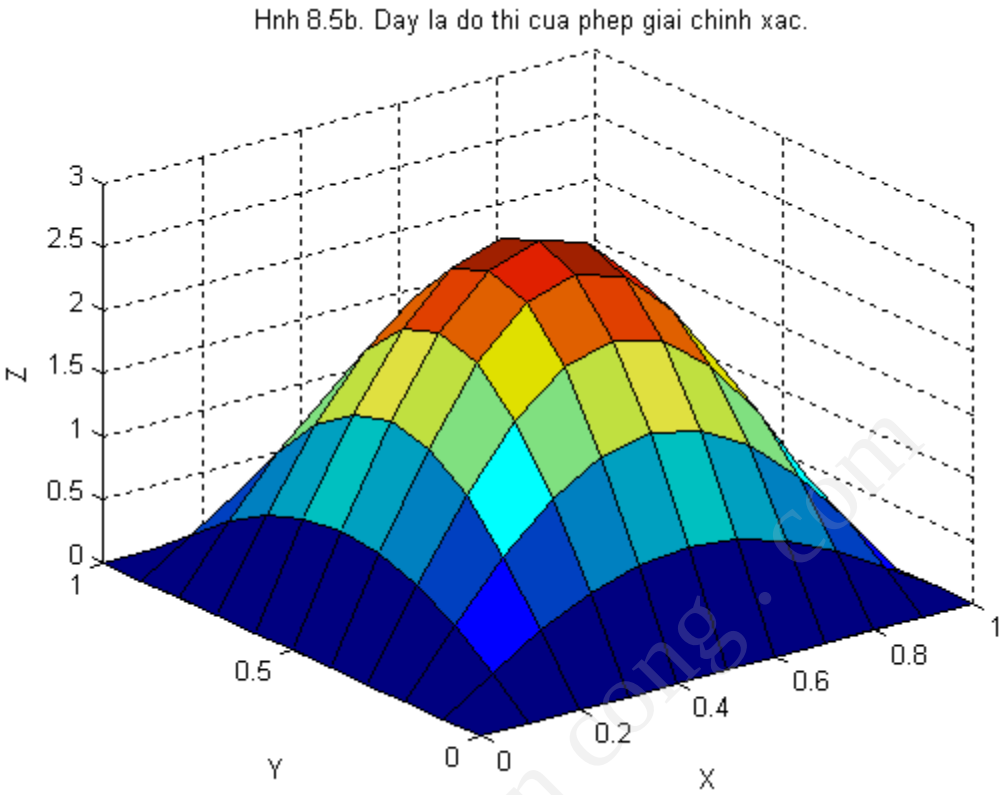
```
% file u0.m; tao ham u0. Day la dieu kien dau.
function a=u0(x,y)
a=sin(pi*x)*sin(pi*y);
% file  g.m.
function a=g(x,y,t)
a=0;
% file change.m. Viet lai chi so diem.
function num=change(N,i,j)
num=(N+1)*(i-1)+j;

% Chuong trinh chinh giai phuong trinh Parabolic hai chieu.
% file heat2d.m
clear all;
T=1; M=100; k=T/M;
N=10; h=1/N;
% Tao luoi.
X=[0:N]*h; Y=[0:N]*h; R=[0:M]*k;
% Dieu kien dau, tao U tai t=0.
U=zeros((N+1)^2,1);
for i=1:(N+1)
   for j=1:(N+1)
      id=change(N,i,j);
      U(id)=u0(X(j),Y(i));
   end
end
% Tinh U tai t=T
for r=1:M
        % Tao ma tran A va vector b.
         A=zeros((N+1)^2,(N+1)^2);
         b=zeros((N+1)^2,1);
for i=1:(N+1)
   for j=1:(N+1)
      id=change(N,i,j);              % chi so cua diem (i,j)
      idlelf=change(N,i,j-1);        % chi so cua diem nam ben trai diem (i,j)
      idright=change(N,i,j+1);       % chi so cua diem nam ben phai (i,j)
      idup=change(N,i-1,j);                % chi so cua diem nam tren (i,j)
      iddown=change(N,i+1,j);        % chi so cuadiem nam duoi (i,j)
      if ((i>1)&&(i<N+1)&&(j>1)&&(j<N+1))
         A(id,id)=h^2/k+4;
         A(id,idlelf)=-1;
         A(id,idright)=-1;
         A(id,idup)=-1;
```

```
            A(id,iddown)=-1;
            b(id)=f(X(j),Y(i),R(r))*h^2+U(id)*h^2/k;
         else
            A(id,id)=1;
            b(id)=uex(X(j),Y(i),R(r));
         end
      end
end

% Giai he AU=b.
U=A\b;                                  %Day la phep giai U tai t=R(r).
end

% Tinh toan cac sai so.
Uex=zeros((N+1)^2,1);
for i=1:(N+1)
   for j=1:(N+1)
      id=change(N,i,j);
      Uex(id)=uex(X(j),Y(i),T);
   end
end
disp('error (in norm max)')
errmax=norm(U-Uex,inf)
% Ve cac do thi.
UU=zeros(N+1,N+1);
for i=1:(N+1)
   for j=1:(N+1)
      UU(i,j)=U(change(N,i,j));
   end
end
figure(1)
surf(X,Y,UU);
axis([0 1 0 1 0 3])
hold on
title('Hinh 8.5a. Day la do thi cua phep giai so.')
hold off
disp('Vui long nhan phim bat ky de tiep tuc')
pause          % Dung thuc thi chuong trinh cho den khi an phim bat ky.

% Tao ham gia tri dung de ve do thi.
UUex=zeros(N+1,N+1);
for i=1:(N+1)
```

```
    for j=1:(N+1)
       UUex(i,j)=Uex(change(N,i,j));
    end
end
figure(2)
surf(X,Y,UUex);
axis([0 1 0 1 0 3])
xlabel('X');
ylabel('Y');
zlabel('Z');
hold on
title('Hnh 8.5b. Day la do thi cua phep giai chinh xac. ')
hold off
end
```

Kết quả chạy file **heat2d.m** với $\Delta x = 1/10, \Delta t = 1/100$ và giải $u(x, y, 1)$.






Sai số trong chuẩn sup là errmax = 0.0052.

Hãy thay đổi giá trị N, M để có những kết quả giải số khác nhau.

Bài toán tương tự sau cũng giải số $u(x,y,1)$ với $\Delta x = 1/10, \Delta t = 1/100$.

$$\begin{cases} u_t - \Delta u = (1+5\pi^2)e^t \sin(2\pi x)\sin(\pi y),, & in\ (x,\ y,\ t) \in \Omega \times [0,T], \\ u(x,\ y,\ t) = 0, & on\ \partial\Omega \times [0,T], \\ u(x,\ y,\ 0) = \sin(2\pi x)\sin(\pi y). & on\ \overline{\Omega}. \end{cases}$$

Có lời giải chính xác là $u_{ex}(x,\ y,\ t) = e^t \sin(2\pi x)\sin(\pi y)$.

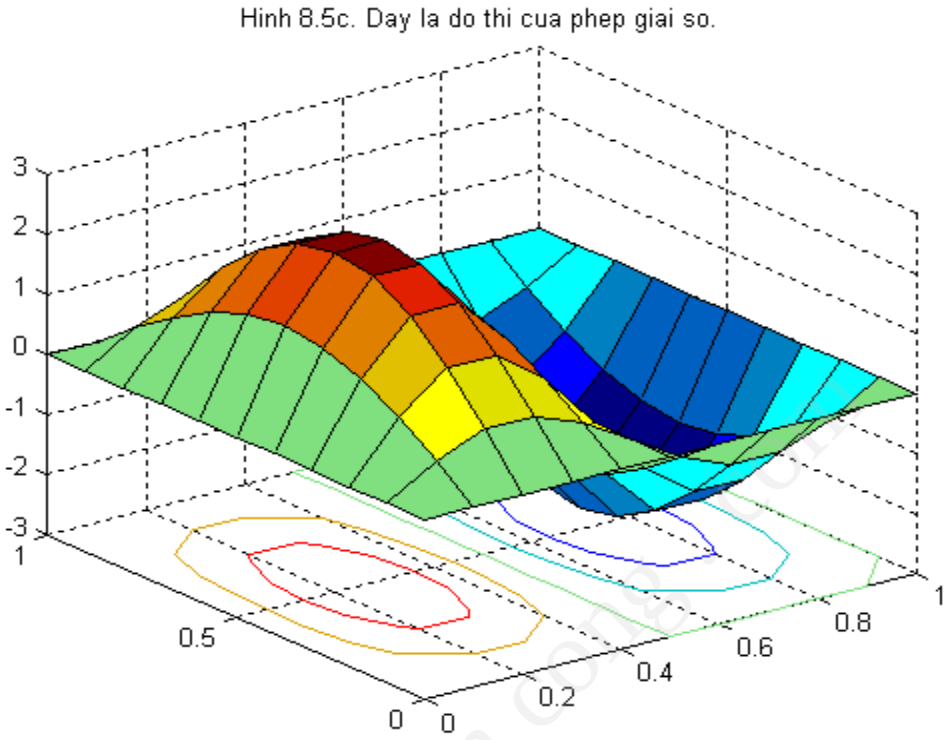
Ta chỉ cần thay đổi các hàm trong chương trình 8.5 theo yêu cầu bài toán sẽ được kết quả . Cụ thể là các hàm f, u0 và uex.

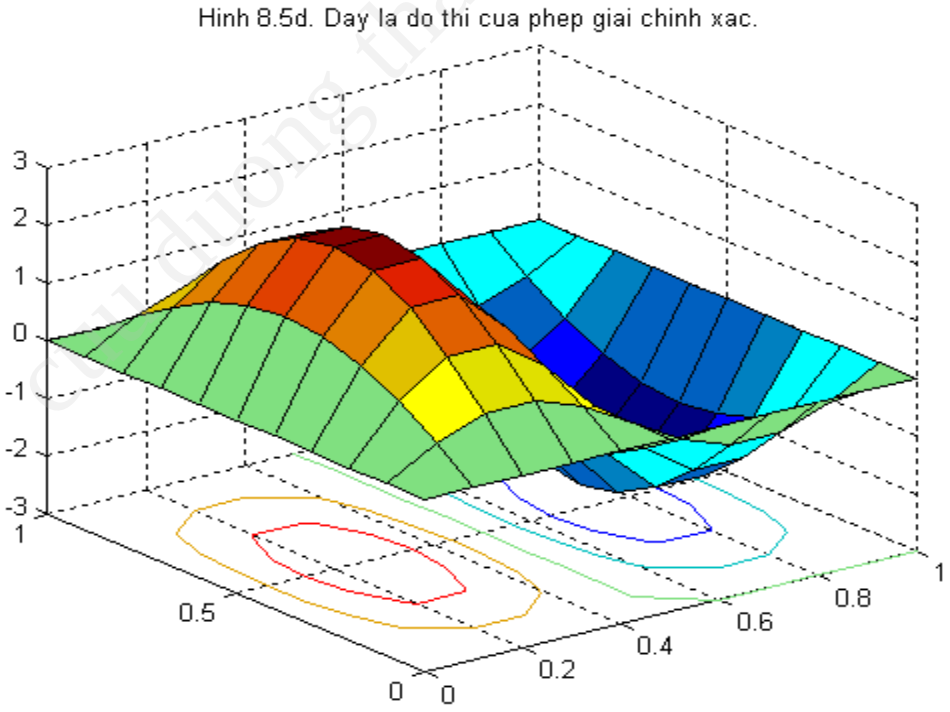
**function a=f(x,y,t); a=(1+2*pi^2)*exp(t)*sin(pi*x)*sin(pi*y);**

**function a=uex(x,y,t); a=exp(t)*sin(pi*x)*sin(pi*y);**

**function a=u0(x,y,t); a=sin(pi*x)*sin(pi*y);**

Các file có trong thư mục Heat2d-1. Kết quả




Sai số trong chuẩn sup là errmax =0.0722.

## 8.6. BẬC HỘI TỤ VÀ ĐIỀU KIỆN BIÊN NEUMANN CHO BÀI TOÁN HAI CHIỀU

- Tương tự như trường hợp 1 chiều, trong trường hợp 2 chiều bài toán trong 8.4.1 cũng có bậc hội tụ là $O(h^2)$ và bài toán trong 8.5.1 có bậc hội tụ là $O(\max\{k,h^2\})$.
- **Xét bài toán trong 8.4.1 với điều kiện biên Neumann**

$$\begin{cases} \Delta u = f \text{ in } \Omega, \\ u = g_0 \text{ on } \Gamma_1, \\ \dfrac{\partial u}{\partial \mathrm{n}} = g_1 \text{ on } \Gamma_2. \end{cases}$$

Trong đó $\Gamma_1 = [0,1]\times\{0,1\}, \Gamma_2 = \{0,1\}\times[0,1]$. Khi đó, ta cần có sự điều chỉnh để xây dựng ma trận A và vector b. Chỗ khác biệt là khi điểm $x_{ij}$ thuộc biên $\Gamma_2$. Ta xét, chẳng hạn khi $x_{ij}$ thuộc $\{0\}\times(0,1)$.

$$\begin{matrix} & x_{i,j-1} & \\ x_{ij} & & x_{i+1,j} \\ & x_{i,j+1} & \end{matrix}$$

Phương trình tương ứng với điểm $x_{ij}$ là gì? Một cách đơn giản, ta có thể dùng xấp xỉ

$$\frac{u(x_{i+1,j}) - u(x_{ij})}{h} \approx g_1(x_{ij}).$$

Tuy nhiên, cách làm này cho bậc hội tụ chỉ là $O(h)$. Để đạt được bậc hội tụ là $O(h^2)$, ta cần một tính toán tinh tế hơn. Sử dụng các xấp xỉ

$$2u(x_{i+1,j}) - 2u(x_{ij}) = 2g_1(x_{ij})h + \frac{\partial^2 u}{\partial \xi_1^2}(x_{ij})h^2 + O(h^2),$$

$$u(x_{i,j+1}) + u(x_{i,j-1}) - 2u(x_{ij}) = \frac{\partial^2 u}{\partial \xi_2^2}(x_{ij})h^2 + O(h^2),$$

ta thu được

$$2u(x_{i+1,j})+u(x_{i,j+1})+u(x_{i,j-1})-4u(x_{ij})=2g_1(x_{ij})h+\Delta u(x_{ij})h^2+O(h^2).$$

Bài toán Parabolic ta cũng làm tương tự.

## CHƯƠNG TRÌNH MATLAB :

*Chúng tôi trình bày một bài toán Laplace có xử lý điều kiện biên Neumann.*

Xét bài toán sau trên $(x,y)\in\Omega=(0,1)\times(0,1)$

$$\begin{cases} u-\Delta u=-(4+4\pi^2)\sin(\pi x)-(4+16\pi^2)\cos^2(\pi x) \\ \qquad\qquad +8y^3-12y^2-42y+8\pi^2+28, \\ u(x,0)=u(x,1)=(2\sin(\pi x)-1)^2+1, \\ u(1,y)=(2y-1)^3+1,\ u_x(0,y)=-4\pi. \end{cases}$$

Bài toán này có lời giải chính xác là $u_{ex}(x,y)=(2\sin(\pi x)-1)^2+(2y-1)^3)$.

- Với $h=\Delta x=1/30$, chúng tôi giải bài toán theo hai cách xấp xỉ điều kiện biên Neumann tại biên $\{x=0\}$.
- So sánh 2 lời giải số và lời giải chính xác: vẽ hình (vẽ 3 hình: Hinh 8.6a, Hinh 8.6b và Hinh 8.6c) và tính sai số (trong $L^\infty$ và $L^2$).

Dưới đây chúng tôi trình bày phần chính của chương trình. Lời giải chi tiết xin xem trong thư mục bachoitu2d.

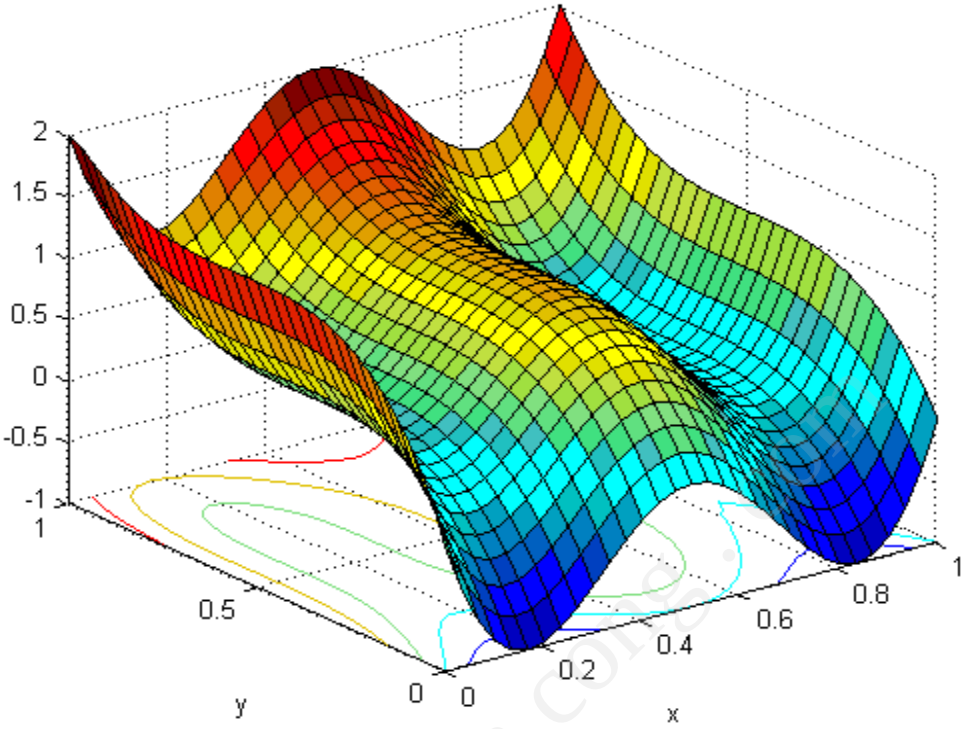
**Chương trình 8.6:**

```
% Loi giai so thu nhat.
% Tao ma tran A va vector b.
A=zeros((N+1)^2,(N+1)^2);
b=zeros((N+1)^2,1);
for i=1:(N+1)
   for j=1:(N+1)
      id=change(N,i,j);
      idlelf=change(N,i,j-1);
      idright=change(N,i,j+1);
      idup=change(N,i-1,j);
      iddown=change(N,i+1,j);
      if ((i>1)&&(i<N+1)&&(j>1)&&(j<N+1))
            A(id,id)=-4-h^2;
        A(id,idlelf)=1;
```

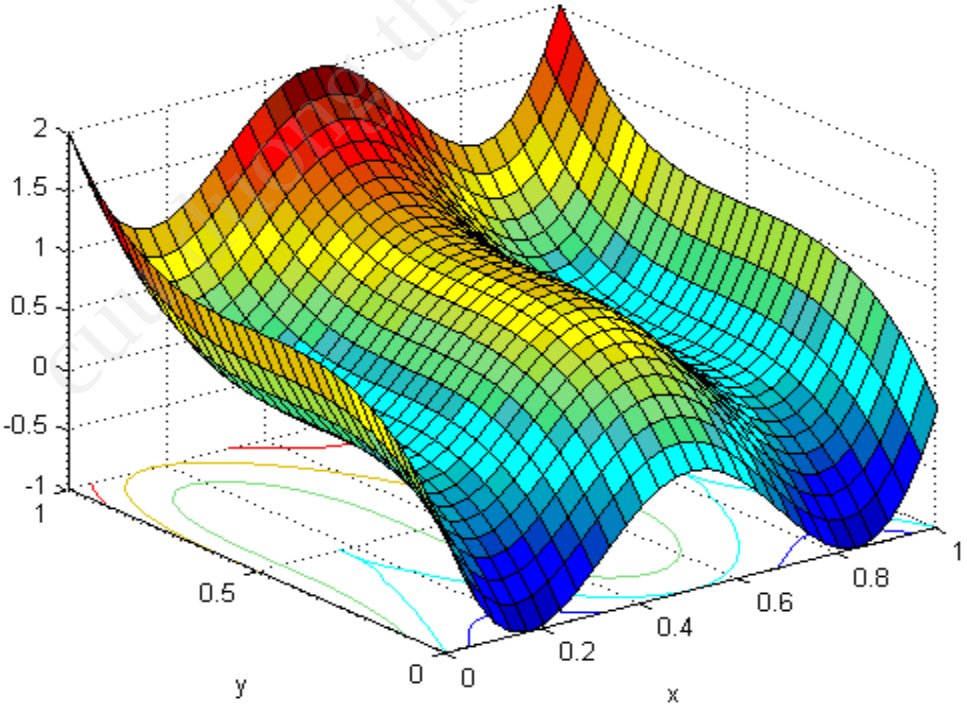
```
            A(id,idright)=1;
            A(id,idup)=1;
            A(id,iddown)=1;
             b(id)=-f(X(j),Y(i))*h^2;
         else if ((j==1)&&(i>1)&&(i<N+1))
               A(id,id)=-1;
               A(id,idright)=1;
               b(id)=g(X(j),Y(i))*h;
% Day la cach xu ly dieu kien bien Neumann theo xap xi bac 1.
            else
               A(id,id)=1;
               b(id)=uex(X(j),Y(i));
            end
         end
      end
end
U1=A\b;              % Giai he AU=b

% Loi giai so thu hai. Su khac biet la tai bien x=0.
j=1;                 % tuong duong voi x=0
for i=2:N
   id=change(N,i,j);
   idlelf=change(N,i,j-1);
   idright=change(N,i,j+1);
   idup=change(N,i-1,j);
   iddown=change(N,i+1,j);
   A(id,id)=-4-h^2;
   A(id,idright)=2;
   A(id,idup)=1;
   A(id,iddown)=1;
   b(id)=2*g(X(j),Y(i))*h-f(X(j),Y(i))*h^2;
% Day la cach xu ly dieu kien Neumann theo xap xi bac hai.
end
U2=A\b;              % Giai he AU=b
```

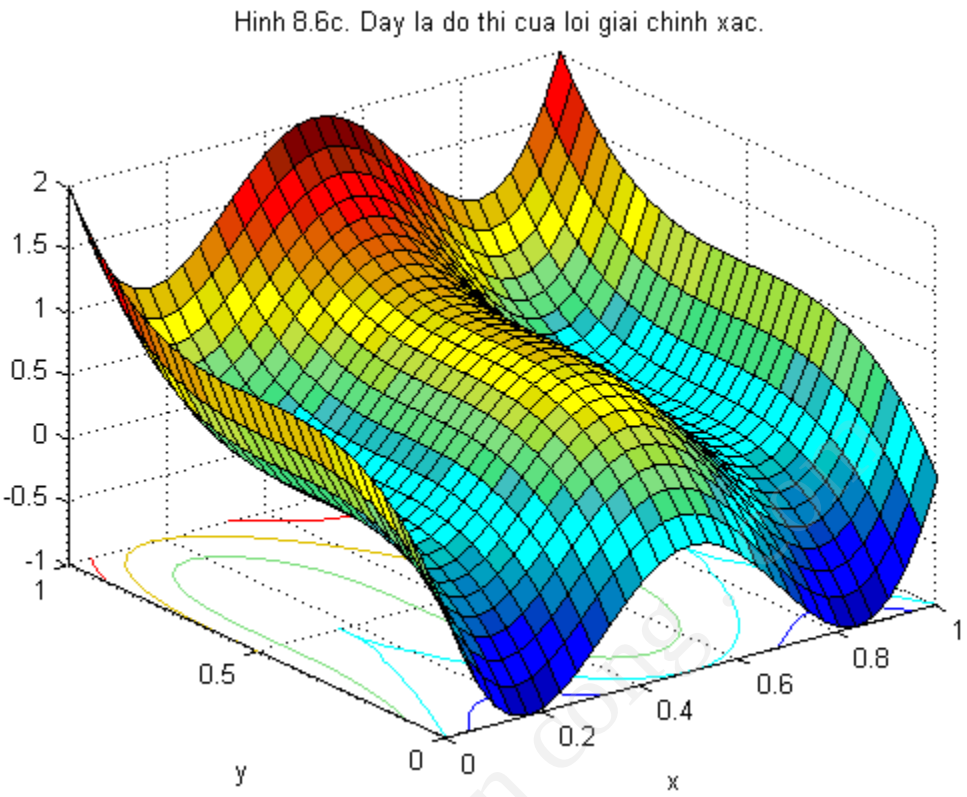
Kết quả chạy file bachoitu.m trong thư mục bachoitu2d.

Hinh 8.6a. Day la do thi cua loi giai so thu nhat.




Hinh 8.6b. Day la do thi cua loi giai so thu hai.






Sai số trong L^infinity và L^2 của lời giải số thứ nhất là

errsup =0.4843 và err2 =0.1510.

Sai số trong L^infinity và L^2 của lời giải số thứ nhất là

errsup =0.0071 và err2 =0.0029.

**Nhận xét:** Trong bài này, lời giải thứ nhất xấp xỉ không tốt tại biên $\{x=0\}$ so với lời giải xấp xỉ thứ hai.

# KẾT LUẬN

## 1. NHẬN ĐỊNH CHUNG

Trong luận văn này, chúng tôi đã trình bày các vấn đề cơ bản của Giải tích số và Phương pháp tính. Trong mỗi chương, mỗi mục chúng tôi đều trình bày chương trình MATLAB để giải số các bài toán. Các chương trình MATLAB được chúng tôi viết ở dạng mở, có nghĩa là người sử dụng có thể chọn lựa hoặc bổ sung phương trình để giải hay chọn hàm để tính cho phù hợp với yêu cầu thực tế. Như vậy với luận văn này, nếu được kết hợp để đưa vào giảng dạy môn học Giải tích số hoặc Phương pháp tính trong các trường Đại học sẽ giúp cho các môn học này trở nên sinh động và hiệu quả hơn rất nhiều so với cách giảng dạy truyền thống. Bởi lẽ tục ngữ có câu: "Trăm nghe không bằng một thấy", "Một hình ảnh có giá trị hơn ngàn lời nói"; mà ở đây, người học có thể thực hành để kiểm tra cụ thể các vấn đề mình vừa học một cách trực tiếp bằng MATLAB dựa trên cơ sở là các chương trình MATLAB của chúng tôi đã trình bày trong luận văn này. Ngoài ra, từ các chương trình MATLAB của chúng tôi, như đã nói người học có thể chọn lựa hoặc bổ sung phương trình để giải hay chọn hàm để tính cho phù hợp với yêu cầu cụ thể của từng bài toán, và cũng có thể dựa vào đó để lập các chương trình MATLAB tương tự để giải các bài toán theo ý riêng của mình. Điều này sẽ giúp nâng cao được hiệu quả giảng dạy cho môn học, phát huy tính tự học, tính ứng dụng và phát triển tư duy của học sinh, sinh viên; đáp ứng yêu cầu của Bộ Giáo Dục – Đào Tạo về việc ứng dụng công nghệ thông tin vào giảng dạy trong các trường học và nâng cao năng lực dạy và học toán trong các trường Đại học hiện nay.

Qua luận văn này, tôi thực sự bắt đầu làm quen với công việc đọc tài liệu khoa học một cách nghiêm túc và có hệ thống từ sự chỉ dẫn tận tình của Thầy ĐẶNG ĐỨC TRỌNG. Tôi đã hệ thống lại được các vấn đề cơ bản về Giải tích số, tiếp cận được các kiến thức về MATLAB và lập trình trong MATLAB, và đã tự

điều chỉnh, bổ sung để các chương trình MATLAB phù hợp với cấu trúc và mục tiêu của luận văn này; thấy được tác dụng rất tích cực của việc ứng dụng công nghệ thông tin vào học tập và giảng dạy.

## 2. MỘT SỐ HƯỚNG NGHIÊN CỨU TIẾP THEO

Ngoài những nhận định trên, khi tìm hiểu về nội dung của luận văn này, tôi đã phát hiện một số vấn đề có thể tiếp tục nghiên cứu sâu hơn. Chẳng hạn như

- Nghiên cứu chuyên sâu các vấn đề về Giải tích số.
- Lập trình để giải số phương trình đạo hàm riêng bằng cấu trúc hàm trong MATLAB.
- Giải số phương trình tích phân bằng MATLAB.
- Giải số phương trình toán tử bằng MATLAB.
- Vận dụng kết hợp ngôn ngữ lập trình C với MATLAB trong giải số.