ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - TIN HỌC

LÊ NHỰT NAM

XẤP XỈ ĐẠO HÀM BẰNG ĐẠO HÀM CỦA ĐA THỨC NỘI SUY

TIỂU LUẬN GIẢI TÍCH SỐ

TP. Hồ Chí Minh - Năm 2024

ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - TIN HỌC

LÊ NHỰT NAM

XẤP XỈ ĐẠO HÀM BẰNG ĐẠO HÀM CỦA ĐA THỨC NỘI SUY

TIỂU LUẬN GIẢI TÍCH SỐ

Giáo viên hướng dẫn: **TS. TRỊNH ANH NGỌC**

TP. Hồ Chí Minh - Năm 2024

LỜI CẢM ƠN

Em xin phép gửi lời cảm ơn chân thành đến thầy *TS. Trịnh Anh Ngọc* đã trực tiếp giảng dạy và tận tình hướng dẫn cho em cùng với các anh chị học viên trong lớp Giải tích số Cao học Khóa 33. Trong quá trình tham gia học tập, em đã tiếp thu được những kiến thức bổ ích và nhận được sự giúp đỡ quý giá từ thầy và các anh chị bạn trong lớp. Đây thật sự là một trải nghiệm tuyệt vời và quý giá với em, một người xuất thân từ lĩnh vực Khoa học Máy tính, và không có chuyên môn nhiều về Toán học.

Xin cảm ơn thầy vì tất cả. Chúng em xin chúc thầy tiếp tục nhiệt huyết, và thành công hơn trong công tác giảng dạy và nghiên cứu khoa học.

Tp. Hồ Chí Minh, tháng 04 năm 2024

Lê Nhựt Nam

We never know how high we are
Till we are called to rise;
And then, if we are true to plan,
Our statures touch the skies
The Heroism we recite
Would be a daily thing,
Did not ourselves the Cubits warp
For fear to be a King

EMILY DICKINSON (1830 – 1886)

MỤC LỤC

DANH MỤC CÁC THUẬT NGỮ

Bounds on the error	Chặn sai số
Backward differences	Sai phân lùi
Determinant	Định thức
Derivative	Đạo hàm
Divided differences	Gia số chia được
Equally spaced interpolation points	Các điểm nội suy cách đều
Forward differences	Sai phân tiến
Finite difference approximation	Xấp xỉ sai phân hữu hạn
Interpolation polynominal	Đa thức nội suy
Iterative linear interpolation	Nội suy tuyến tính lặp
Least squares	Bình phương tối tiểu
Non-singular	Không suy biến
Remainder Terms	Phần dư
Singular	Suy biến
Pointwise Error	Độ lỗi từng điểm

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ

CHƯƠNG 1

MỞ ĐẦU

1.1. Động lực nghiên cứu

Trong nhiều vấn đề thực hành, chúng ta thường gặp các hàm mà các giá trị của nó được biết tại một số điểm nào thông qua các thực nghiệm. Để phục vụ cho một số bài toán nhất định, nhu cầu tính toán tích phân, đạo hàm, hay xấp xỉ giá trị của hàm tại điểm mà ta chưa biết giá trị tại đó là cần thiết. Do đó, nghiên cứu phương pháp xấp xỉ hàm bằng một hàm đã biết mà giá trị tại các điểm đã cho trùng với dữ liệu thực nghiệm là điều cần thiết.

1.2. Mục tiêu và đối tượng nghiên cứu của tiểu luận

Dựa trên bài toán ban đầu - xấp xỉ hàm $f(x)$ bằng một hàm $F(x)$, trùng với $f(x)$ tại các điểm nào đó. Hàm $F(x)$ được xem là nội suy (interpolate) $f(x)$ tại các điểm này. Quá trình xây dựng hàm $F(x)$ nói trên được gọi là phép nội suy (interpolation). Tùy vào bản chất của dữ liệu, loại hàm xấp xỉ được lựa chọn sao cho phù hợp nhất có thể, nhưng đơn giản nhất là *đa thức* bởi vì *mọi hàm liên tục trên một khoảng hữu hạn đều có thể xấp xỉ tốt bằng một đa thức*. Một điều thú vị, do các đa thức và tỉ số của chúng là hàm duy nhất, có thể tính toán được thông qua máy tính. Do đó, đa thức được dùng không chỉ trong nội suy mà còn làm cơ sở cho hầu hết phương pháp trong Giải tích số.

Trong nhiều ứng dụng, một vấn đề quan trọng là xấp xỉ đạo hàm của một hàm khi biết trước chỉ một số giá trị của hàm. Một phương pháp tiếp cận rõ ràng cho vấn đề này là thiết lập đạo hàm của một đa thức xấp xỉ như một xấp xỉ kỳ vọng đến đạo hàm của hàm. Điều này hoàn toàn có thể thực hiện được

cho các đạo hàm cấp cao, nhưng nhìn một cách thật tổng quát, sự xấp xỉ phải tệ đi bởi vì bậc của đạo hàm tăng.

1.3. Ý nghĩa nghiên cứu và thực tiễn ứng dụng

1.4. Cấu trúc của tiểu luận

Trong tiểu luận này, chúng tôi tập trung làm rõ vấn đề đạo hàm số mà trong tiếp cận xấp xỉ bằng một đa thức nội suy là đối tượng được quan tâm chính. Cấu trúc của tiểu luận được trình bày như sau:

- Chương 1: Trình bày động lực nghiên cứu, mục tiêu, đối tượng nghiên cứu chính, ý nghĩa nghiên cứu và thực tiễn ứng dụng của đề tài.
- Chương 2: Trình bày kiến thức nền tảng của đề tài.
- Chương 3: Trình bày chi tiết về vấn đề quan tâm - đạo hàm số và phương pháp.
- Chương 4: Trình bày phương pháp lập trình bài toán đạo hàm số bằng Matlab.
- Chương 5: Trình bày các kết luận về đề tài và định hướng với các câu hỏi mở cho nghiên cứu tương lai.

CHƯƠNG 2

KIẾN THỨC NỀN TẢNG

2.1. Đa thức nội suy

Đa thức nội suy (interpolation polynomial) là một đa thức xấp xỉ mà bằng với hàm mà nó xấp xỉ tại một số điểm cụ thể. Một cách cụ thể, cho trước $n+1$ điểm phân biệt $x_i, i = 0, \ldots, n$, và các giá trị hàm $f(x_i)$ tương ứng, đa thức nội suy với bậc tối đa n cực tiểu chuẩn:

$$|f(x) - P_n(x)|_{(sn)^\dagger} \equiv \sum_{i=0}^{n} |f(x_i) - P_n(x_i)| \tag{2.1}$$

Người ta chỉ ra rằng một đa thức tồn tại và duy nhất; thật vậy, giá trị nhỏ nhất của chuẩn đề cập phía trên là 0. Có hai cách chỉ ra nhận định này là đúng:

- Giải hệ phương trình tuyến tính,
- Sử dụng dạng Lagrange của đa thức.

Xem xét một đa thức có dạng:

$$Q_n(x) = \sum_{k=0}^{n} a_k x^k, Q_n(x_i) = f(x_i) \tag{2.2}$$

Bằng cách xem xét các hệ số a_k là ẩn số, ta có một hệ $n+1$ phương trình tuyến tính

$$A = Q_n(x_i) = a_0 + a_1 x_1 + \cdots + a_n x_i^n = f(x_i), i = 0, 1, \ldots, n. \tag{2.3}$$

Nếu hệ số của ma trận là không suy biến, thì hệ có nghiệm duy nhất. Xem xét

định thức Vandermonde của ma trận này

$$\mathrm{Det}(A) = \prod_{i>j}(x_i - x_j) \equiv \prod_{j=0}^{n-1}\left[\prod_{i=j+1}^{n}(x_i - x_j\right] \tag{2.4}$$

Do $\{x_i\}$ là các điểm phân biệt, nên định thức trên không suy biến, và do đó hệ phương trình tuyến tính có nghiệm duy nhất để xác định đa thức nội suy.

Bên cạnh cách vừa đề cập, ta có thể sử dụng dạng Lagrange của đa thức để nhận được trực tiếp đa thức nội suy. Bằng cách đặt

$$P_n(x) = \sum_{j=0}^{n} f(x_i)\phi_{n,j}(x) \tag{2.5}$$

trong đó $n+1$ hàm $\phi_{n,j}(x)$ là các đa thức bậc thứ n.

Những đa thức như thế được xây dựng một cách dễ dàng, bởi vì $\{x_i\}$ là các điểm phân biệt, tức là,

$$\phi_{n,j}(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_{j-1})(x-x_{j+1})\cdots(x-x_n)}{(x_j-x_0)(x_j-x_1)\cdots(x_j-x_{j-1})(x_j-x_{j+1})\cdots(x_j-x_n)}, j = 0, 1, \ldots, n. \tag{2.6}$$

Các đa thức này được gọi là các hệ số nội suy Lagrange.

Ta đặt

$$\omega_n(x) \equiv (x-x_0)(x-x_1)\cdots(x-x_{j-1})(x-x_{j+1})\cdots(x-x_n) \tag{2.7}$$

Ta có:

$$\omega_n'(x) = \left(\frac{\omega_n(x)}{dx}\right)_{x=x_j} \tag{2.8}$$

Suy ra, các hệ số nội suy Lagrange có thể được viết gọn như sau:

$$\phi_{n,j}(x) = \frac{\omega_n(x)}{(x-x_j)\omega_n'(x)} \tag{2.9}$$

Bằng cách sử dụng tích trong, dạng Lagrange của đa thức nội suy có thể được

viết:

$$P_n(x) = \sum_{j=0}^{n} f(x_i) \prod_{k=0, k\neq j}^{n} \frac{x - x_k}{x_j - x_k} \tag{2.10}$$

Đa thức nội suy Lagrange xác định đa thức được định nghĩa bởi (2.2) và (2.4) là hệ quả của định lý sau:

Định lý 2.1.1. *Giả sử $P_n(x)$ và $Q_n(x)$ là hai đa thức bất kỳ, có bậc tối đa n, mà*

$$P_n(x_i) = Q_n(x_i), i = 0, 1, \ldots, n, \tag{2.11}$$

trong đó $n+1$ điểm $\{x_i\}$ là các điểm phân biệt. Thì

$$P_n(x) \equiv Q_n(x) \tag{2.12}$$

Điều này chỉ ra rằng, có một và chỉ một đa thức bậc tối đa n mà (2.1) suy biến và được cho bởi (2.10) và (2.6)

2.2. Độ lỗi từng điểm trong nội suy đa thức

Độ lỗi từng điểm (pointwise error) giữa một hàm, $f(x)$, và một số đa thức xấp xỉ đến nó, $P_n(x)$, được định nghĩa:

$$R_n(x) \equiv f(x) - P_n(x) \tag{2.13}$$

Với các đa thức nội suy, một biểu diễn hữu ích của $R_n(x)$ dễ dàng có được. Định lý dưới đây phát biểu cho điều này.

Định lý 2.2.1. *Giả sử $f(x)$ có đạo hàm đến cấp $(n+1)$, $f^{(n+1)}$, trong một khoảng $[a, b]$. Và $P_n(x)$ là đa thức nội suy cho $f(x)$ tương ứng với $n+1$ điểm phân biệt $x_i, i = 0, 1, \ldots, n$ trong đoạn $[a, b]$, tức là $P_n(x_i) = f(x_i)$ và $x_i \in [a, b]$. Thì với mọi*

$x \in [a, b$, *tồn tại một điểm* $\xi = \xi(x)$ *trong khoảng mở*

$$\min(x_0, x_1, \ldots, x_n, x) < \xi < \max(x_0, x_1, \ldots, x_n, x) \tag{2.14}$$

mà

$$R_n(x) \equiv f(x) - P_n(x) = \frac{(x - x_0)(x - x_1) \cdots (x - x_n)}{(n+1)!} f^{(n+1)}(\xi) \equiv \frac{\omega_n(x)}{(n+1)!} f^{(n+1)}(\xi) \tag{2.15}$$

Nếu giá trị lớn nhất và giá trị nhỏ nhất của $f^{(n+1)}(x)$ trong $[a, b]$ được xác định, (2.15) cho ta các chặn sai số. Nên lưu ý rằng sai số (2.15) cho các đa thức nội suy giống với phần dư trong khai triển Taylor. Thật vậy, ta có thể giả định một cách đơn giản rằng nếu $|x - x_i| < |x - x_0|, i = 1, 2, \ldots, n$ thì độ lỗi của đa thức nội suy nhỏ hơn độ lỗi trong khai triển Taylor xung quanh điểm x_0. Giả định này không phải bao giờ cũng đúng bởi vì $f^{(n+1)}(\xi)$ trong khai triển Taylor và trong (2.15) không được đánh giá tại cùng một điểm ξ với x cho trước.

2.3. Đa thức nội suy Newton

Giả sử $Q_k(x)$ là đa thức nội suy cho $f(x)$, bậc tối đa k, tương ứng với $k + 1$ điểm phân biệt $x_0, x_1, \ldots, x_k$. Ta cần tìm các đa thức nội suy, $\{Q_k(x)\}$ với bậc tối đa k mà trong dạng

$$Q_0(x) \equiv f(x_0) \tag{2.16}$$

và

$$Q_k(x) = Q_{k-1}(x) + q_k(x), k = 1, 2, \ldots, n, \tag{2.17}$$

trong đó $q_k(x)$ có bậc tối đa k.

Do ta cần

$$Q_k(x_j) = f(x_j) = Q_{k-1}(x_j), j = 0, 1, \ldots, k - 1 \tag{2.18}$$

nên $q_k(x_j) = 0$ tại k điểm này. Dẫn đến, ta thể hiện đa thức tổng quát nhất với

bậc tối đa k mà suy biến tại k điểm như nhau:

$$q_k(x) = a_k \prod_{j=0}^{k-1}(x - x_j) \tag{2.19}$$

Trong phương trình trên, hằng số a_k cần phải được xác định. Để mà $Q_k(x_k) = f(x_k)$, hằng số này phải

$$a_k = \frac{f(x_k) - Q_{k-1}(x_k)}{\prod_{j=0}^{k-1}(x - x_j)}, k = 1, 2, \ldots, n, \tag{2.20}$$

Rất tự nhiên, đa thức nội suy bậc không cho điểm khởi tạo x_0 là $Q_0(x) \equiv f(x_0)$. Do đó, với $a_0 = f(x_0)$, bằng kỹ thuật đệ quy, ta có đa thức nội suy duy nhất bậc n có dạng như sau:

$$Q_n(x) = a_0 + (x - a_0)a_1 + \cdots + (x - x - 0)\cdots(x - x_{n-1})a_n \tag{2.21}$$

Hệ số thứ k được gọi là *sai phân chia được bậc* k, ký hiệu:

$$\begin{aligned} a_0 &= f[x_0] \\ a_k &= f[x_0, x_1, \ldots, x_k], k = 1, 2, \ldots \end{aligned} \tag{2.22}$$

Các giá trị của $f(x)$ mà được nhập vào để xác định a_k là những tham số của $f[x_0, x_1, \ldots, x_k]$. Biểu diễn này tương mình hơn dạng đệ quy mà được cho bởi (2.17). Do tính duy nhất của (2.21), bằng cách sử dụng dạng Lagrange, ta có thể viết:

$$Q_n(x) = \sum_{j=0}^{n} f(x_j) \prod_{k=0, k\neq j}^{n} \frac{x - x_k}{x_j - x_k} \tag{2.23}$$

và hệ số của x^n là

$$a_n = f[x_0, x_1, \ldots, x_n] = \sum_{j=0}^{n} \frac{f(x_j)}{\prod_{k=0,k\neq j}^{n}(x_j - x_k)} \tag{2.24}$$

Dựa trên dạng (2.24), các *sai phân chia được* là các hàm đối xứng theo đối số của chúng. Thật vậy, nếu ta sử dụng ký hiệu truyền thống

$$f_{i,j,k,\ldots} \equiv f[x_i, x_j, x_k, \ldots] \tag{2.25}$$

thì tính đối xứng được khai triển như sau

$$f_{0,1,\ldots,n} = f_{j_0,j_1,\ldots,j_n} \tag{2.26}$$

trong đó $(j_0, j_1, \ldots, j_n)$ là bất kỳ hoán vị nào của các số nguyên $(0, 1, \ldots, n)$.

Ta có thể thu được một dạng tiện lợi hơn (2.24) bằng cách sử dụng tính duy nhất của đa thức nội suy. Ta có thể xây dựng đa thức $Q_n(x)$ bằng cách khớp các giá trị của $f(x_j)$ trong một thứ tự nghịch đảo $j = n, n-1, \ldots, 1, 0$.

$$Q_n(x) \equiv b_0 + (x - x_n)b_1 + \cdots + (x - x_n)(x - x_{n-1}) \cdots (x - x_1)b_n \tag{2.27}$$

trong đó $b_k = f[x_n, x_{n-1}, \ldots, x_{n-k}$, và $b_0 = f[x_n] = f(x_n)$.

Để ý rằng, $a_n = b_n$, nên từ (2.21) và (2.27), ta có:

$$0 \equiv [(x - x_0) - (x - x_n)](x - x_1) \cdots (x - x_{n-1})a_n + (a_{n-1} - b_{n-1}x^{n-1} + p_{n-2}(x) \tag{2.28}$$

trong đó $p_{n-2}(x)$ là một đa thức bậc cao nhất $n - 1$.

Và dựa trên tính đối xứng của các sai phân chia được, dẫn đến việc

$$b_{n-1} = f[x_n, x_{n-1}, \ldots, x_1] = f[x_1, x_2, \ldots, x_n] \tag{2.29}$$

từ $a_n = (a_{n-1} - b_{n-1})/(x_0 - x_n)$, ta có:

$$f[x_0, x_1, \ldots, x_n] = \frac{f[x_0, x_1, \ldots, x_{n-1}] - f[x_1, x_2, \ldots, x_n]}{x_0 - x_n}, n = 1, 2, \ldots \tag{2.30}$$

Và bằng cách định nghĩa cho tính hoàn thiện

$$f[x_0] = f(x_0) \tag{2.31}$$

Đa thức nội suy (2.21) có thể được viết lại như sau:

$$Q_n(x) = f[x_0] + (x - x_0)f[x_0, x_1] + \cdots + (x - x_0)\cdots(x - x_{n-1})f[x_0, x_1, \ldots, x_n] \tag{2.32}$$

Dạng này được gọi là dạng nội suy sai phân chia được Newton.

Một biểu diễn khác của độ lỗi trong đa thức nội suy có thể thu được bằng cách đặt $k = n + 1$ trong biểu thức (2.19) và (2.20) và (2.22) như sau:

$$f(x) - Q_n(x) = \left[\prod_{j=0}^{n}(x - x_j)\right] f[x_0, x_1, \ldots, x_n, x] \tag{2.33}$$

Một biểu biểu diễn khác của các sai phân chia được cho thấy nhiều lợi ích hơn trong việc ước lượng độ lớn của chúng được thể hiện trong Định lý sau

Định lý 2.3.1. *Giả sử $x, x_0, x_1, \ldots, x_{k-1}$ là $k + 1$ điểm phân biệt và gọi $f(y)$ có đạo hàm liên tục cấp k trong khoảng*

$$\min(x, x_0, x_1, \ldots, x_{k-1}) < y < \max(x, x_0, x_1, \ldots, x_{k-1}) \tag{2.34}$$

Thì với một số điểm $\xi = \xi(x)$ trong khoảng này

$$f[x_0, x_1, \ldots, x_{k-1}, x] = \frac{f^{(k)}(\xi)}{(k)!} \tag{2.35}$$

Ta có một hệ quả trực tiếp của Định lý (2.3.1) thể hiện một số thông tin trên các sai phân chia được của các đa thức. Kết quả này dược phát biểu như sau:

Hệ quả 1. *Giả sử*

$$P_n(x) = \alpha_0 + \alpha_1 x + \cdots + \alpha_n x^n, \alpha_n \neq 0 \tag{2.36}$$

là bất kỳ đa thức bậc n nào và gọi $x_0, x_1, \ldots, x_k$ là $k+1$ điểm phân biệt. Thì

$$P_n[x_0, x_1, \ldots, x_k] = \begin{cases} \alpha_n, & \text{nếu } k = 0 \\ 0, & \text{nếu } k > n \end{cases} \tag{2.37}$$

Trong một số phân tích đánh giá về độ lỗi trong đạo hàm số và tích số phân số, ta có thể cần một số tính chất liên tục (continuity) và khả vi (differentiability) của các sai phân chia được. Những kết quả đó được suy ra từ một biểu diễn khác của sai phân chia được mà được phát biểu thông qua Định lý dưới đây.

Định lý 2.3.2. *Giả sử $f(x)$ có đạo hàm liên tục đến cấp n trong khoảng $\min(x_0, x_1, \ldots, x_k) < y < \max(x_0, x_1, \ldots, x_k)$. Thì nếu những điểm $x_0, x_1, \ldots, x_k$ phân biệt,*

$$f[x_0, x_1, \ldots, x_k] = \int_0^1 dt_1 \int_0^{t_1} dt_2 \cdots \int_0^{t_{n-1}} dt_n \times f^{(n)}(t_n[x_n - x_{n-1}] + \cdots + t_1[x_1 - x_0] + x_0) \tag{2.38}$$

trong đó, $n \geq 1, t_0 = 1$

Các tích phân vế phải của biểu thức (2.38) là một hàm liên tục có $n+1$ biến $x_0, x_1, \ldots, x_n$ và do đó vế phải này là một hàm liên tục với các biến này. Điều này dẫn đến (2.38) định nghĩa duy nhất một mở rộng liên tục của $f[x_0, x_1, \ldots, x_n]$ khi mà các tham số nằm trong bất kỳ khoảng liên tục nào của đạo hàm cấp n của $f(x)$. Các sai phân chia được không chỉ có thể được định nghĩa cho các tập tham số phân biệt mà còn có thể được định nghĩa trong trường hợp không phân biệt. Một cách tự nhiên, ta có thể xây dựng tương tự nếu có sự liên tục. Nếu $f^{(n)}(x)$ liên tục, thì theo Định lý (2.3.2) chỉ ra rằng làm thế mà sự liên tục cho thể có cho tất cả sai phân của $f(x)$ theo thứ tự $0, 1, \ldots, n$. Nhận xét này có thể được tóm tắt trong Hệ quả sau.

Hệ quả 2. *Giả sử $f^{(n)}(x)$ liên tục trên đoạn $[a, b]$. Với bất kỳ tập điểm $x_0, x_1, \ldots, x_k$ trong $[a, b]$ với $k \leq n$, giả sử $f[x_0, x_1, \ldots, x_k]$ được cho trước bởi $(10)_k$. Sai phân chia được được định nghĩa là một hàm liên tục của $k+1$ đối số của nó trong*

$[a,b]$ và trùng với các định nghĩa ở (2.24), hay (2.32) khi các đối số phân biệt nhau.

Thật vậy, như trong chứng minh của First Mean Value Theorem cho tích phân, $(10)_n$

$$m\int_0^1 dt_1\int_0^{t_1} dt_2\cdots\int_0^{t_{n-1}} dt_n \le f[x_0,\ldots,x_n] \le M\int_0^1 dt_1\int_0^{t_1} dt_2\cdots\int_0^{t_{n-1}} dt_n \tag{2.39}$$

trong đó $m \equiv \min f^{(n)}(x)$ và $M \equiv \max f^{(n)}(x)$ với x trong

$$\min(x_0,\ldots,x_n) \le x \le \max(x_0,\ldots,x_n) \tag{2.40}$$

Thì bởi tính liên tục của $f^{(n)}$, có một điểm μ_n trong khoảng này sao cho

$$f[x_0,\ldots,x_n] = \frac{f^{(n)}(\mu_n)}{n!} \tag{2.41}$$

Bởi vì các điểm x_i không nhất thiết phải phân biệt, nên ta có một phiên bản tổng quát cho Định lý (2.3.1) trong Hệ quả sau:

Hệ quả 3. *Nếu $f^{(n)}(x)$ liên tục trên đoạn $[a,b]$ và $x_0,x_1,\ldots,x_n$ trong $[a,b]$ thì*

$$f[x_0,x_1,\ldots,x_n] = \frac{f^{(n)}(\xi)}{n!}, \tag{2.42}$$

trong đó

$$\min(x_0,x_1,\ldots,x_n) \le \xi \le \max(x_0,x_1,\ldots,x_n). \tag{2.43}$$

Một trường hợp cụ thể là

Hệ quả 4. *Nếu $f^{(n)}(x)$ liên tục trong một lân cận của x, thì*

$$f[\underbrace{x,x,\ldots,x}_{n+1}] = \frac{f^{(n)}(\xi)}{n!}. \tag{2.44}$$

Và có thể suy ra một biểu diễn khác của sai phân chia được khi nhiều nhân

tử xuất hiện

Hệ quả 5. *Nếu $f^{(n)}(x)$ liên tục trên đoạn $[a,b]$, $y_0, y_1, \ldots, y_n$ trong đoạn $[a,b]$, thì*

$$f[x, y_0, y_1, \ldots, y_n] = \frac{f[x, y_1, \ldots, y_n] - f[y_0, y_1, \ldots, y_n]}{x - y_0} \tag{2.45}$$

cho một mở rộng liên tục duy nhất của định nghĩa về sai phân chia được.

Hệ quả 6. *Nếu $x_i \neq y_i$ với $0 \leq i \leq p, 0 \leq j \leq q; f^{(m)}(x)$ liên tục trong đoạn $[a,b]; \{x_i\}, \{y_i\}$ trong $[a,b]; 0 \leq p, q \leq m$ thì*

$$f[x_0, \ldots, x_p, y_0, \ldots, y_q] = g[x_0, \ldots, x_p] = h[y_0, \ldots, y_q] \tag{2.46}$$

trong đó: $g(x) \equiv f[x, y_0, \ldots, y_q$, $h(x) \equiv f[x_0, \ldots, x_p, y$,

cho một mở rộng liên tục duy nhất của định nghĩa của sai phân chia được.

Hệ quả 7. *Nếu $f(x)$ có đạo hàm liên tục cấp m trong $[a,b]; x_0, \ldots, x_p, y_0, \ldots, y_q, z_0, \ldots, z_r$ trong $[a,b]; x_i \neq y_i, x_i \neq z_k, y_j \neq z_k$ với mọi $i, j, k; 0 \leq p, q, r \leq m;$ thì*

$$f[x_0, \ldots, x_p, y_0, \ldots, y_q, z_0, \ldots, z_r] = \frac{1}{p!q!r!} \frac{\partial^p}{\partial x^p} \frac{\partial^q}{\partial y^q} \frac{\partial^r}{\partial z^r} f[x, y, z] \Big|_{\xi,\eta,\zeta} \tag{2.47}$$

trong đó:

$$\begin{aligned} &\min(x_0, \ldots, x_p) \leq \xi \leq \max(x_0, \ldots, x_p), \\ &\min(y_0, \ldots, y_q) \leq \eta \leq \max(y_0, \ldots, y_q), \\ &\min(z_0, \ldots, z_r) \leq \zeta \leq \max(z_0, \ldots, z_r). \end{aligned} \tag{2.48}$$

Một trường hợp đặc biệt được thể hiện trong Hệ quả sau:

Hệ quả 8. *ếu $f^{(m)}(x)$ liên tục trên đoạn $[a,b]; x, y, z$ là các điểm phân biệt trong đoạn $[a,b]; 0 \leq p, q, r \leq m;$ thì*

$$f[\underbrace{x, x, \ldots, x}_{p+1} \underbrace{y, y, \ldots, y}_{q+1} \underbrace{z, z, \ldots, z}_{r+1}] = \frac{1}{p!q!r!} \frac{\partial^p}{\partial x^p} \frac{\partial^q}{\partial y^q} \frac{\partial^r}{\partial z^r} f[x, y, z] \tag{2.49}$$

x_0	$f(x_0)$				
x_1	$f(x_1)$	$P_{0,1}(x)$			
x_2	$f(x_2)$	$P_{1,2}(x)$	$P_{0,1,2}(x)$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
x_k	$f(x_k)$	$P_{k-1,l}(x)$	$P_{k-2,k-1,k}(x)$	$\dots$	$P_{0,1,2,\dots,k}(x)$

Bảng 2.1: *Lược đồ nội suy lặp Neville (Neville's iterated interpolation)*

2.4. Nội suy tuyến tính tuần tự

Dạng Newton của đa thức nội suy cho phép dễ dàng tăng độ chính xác cho phương pháp xấp xỉ đa thức. Trong thực hành, các thủ tục lặp được sử dụng và rất hiệu quả khi kết hợp với tính toán máy tính. Bổ đề dưới đây đóng vai trò nền tảng cho các lược đồ *nội suy tuyến tính tuần tự*.

Bổ đề 1. *Giả sử $x_{i_1}, x_{i_2}, \dots, x_{i_n}$ là n điểm phân biệt và $P_{i_1,i_2,\dots,i_n}(x)$ là đa thức nội suy bậc $n-1$ mà thỏa*

$$P_{i_1,i_2,\dots,i_n}(x_{i_v}) = f(x_{i_v}), v = 1, 2, \dots, n. \tag{2.50}$$

Thì nếu x_j, x_k và $x_{i_v}, v = 1, 2, \dots, n$ là bất kỳ $m+2$ điểm phân biệt nào,

$$P_{i_1,i_2,\dots,i_m,j,k}(x) \equiv \frac{(x-x_k)P_{i_1,i_2,\dots,i_m,j}(x) - (x-x_j)P_{i_1,i_2,\dots,i_m,k}(x)}{x_j - x_k}, m = 0, 1, 2, \dots \tag{2.51}$$

Nhiều lược đồ sử dụng Bổ đề trên để xác định các đa thức nội suy bậc cao (higher order interpolation polynomials) theo thứ tự của các cặp giá trị $(x_j, f(x_j)$. Một ví dụ điển hình là *nội suy lặp Neville* như trong Bảng 2.1. Phương pháp nội suy này có nhiều ứng dụng trong thực hành, điển hình nhất là trong phương pháp lặp nội suy ngược (iterative inverse iterpolation).

CHƯƠNG 3

XẤP XỈ ĐẠO HÀM SỐ BẰNG ĐẠO HÀM CỦA CÁC ĐA THỨC NỘI SUY

Trong chương này, chúng tôi trình bày một vấn đề quan trọng trong nhiều ứng dụng - xấp xỉ đạo hàm của một hàm mà chỉ biết trước một số giá trị của của hàm.

3.1. Một số thiết lập ban đầu

Ta gọi $P_n(x)$ là đa thức nội suy bậc n cho $f(x)$ tương ứng với $n+1$ điểm phân biệt $x_0, x_1, \ldots, x_n$. Nếu ta sử dụng $P_n^{(k)}(x)$, thì một xấp xỉ đến đạo hàm cấp k của $f(x)$ được viết như sau:

$$\frac{d^k f(x)}{dx^k} \equiv f^{(k)}(x), k < n \tag{3.1}$$

Tuy nhiên, để đánh giá được xấp xỉ này ta cần một số biểu diễn hiệu quả cho độ lỗi

$$R_n^{(k)}(x) \equiv f^{(k)}(x) - P_n^{(k)}(x) \tag{3.2}$$

Nếu $f^{(n+1)}(x)$ là liên tục trong khoảng I_x mà bao gồm cả x_j và x, theo Định lý (2.2.1) ta có:

$$R_n(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_n)}{(n+1)!} f^{(n+1)}(\xi) = \prod_{j=0}^{n}(x-x_j)\frac{f^{(n+1)}(\xi)}{(n+1)!} \tag{3.3}$$

trong đó $\xi = \xi(x)$ là một điểm chưa biết trong I_x với mỗi x.

Ta nhận thấy tiềm năng trong việc đạo hàm biểu thức cho $R_n(x)$ để thu được

$R_n^{(k)}(x)$ nhưng điều này **không hoàn toàn đúng đắn**. Có hai lý do cho điều này:

- Lý do 1: $\xi(x)$ có thể không có giá trị đơn, nhưng có đạo hàm cấp k,
- Lý do 2: $f(x)$ có thể không có đạo hàm cấp $n+1+k$

Nếu $f(x)$ có tính chất khả vi, thì một biểu diễn thay thế cho độ lỗi được biểu diễn như sau:

$$R_n(x) = \prod_{j=0}^{n}(x - x_j)f[x_0, \ldots, x_n, x] \tag{3.4}$$

Và bây giờ bằng cách ứng dụng Định lý (2.3.1) thì biểu diễn này có đạo hàm cấp k. Tuy nhiên, biểu thức kết quả khá phức tạp và chỉ có thể sử dụng trong trường hợp đạo hàm cấp 1, tức $k = 1$ mà trong đó $x = x_i$ là một điểm nội suy. Độ lỗi trở thành trong trường hợp này như sau:

$$\begin{aligned} f'(x_i) - P_n'(x_i) = R_n'(x_i) &= \prod_{j=0, j\neq i}^{n}(x_i - x_j)f[x_0, x_1, \ldots, x_n, x_i] \\ &= \prod_{j=0, j\neq i}^{n}(x_i - x_j)\frac{f^{(n+1)}(\eta)}{(n+1)!} \end{aligned} \tag{3.5}$$

Biến đổi cuối cùng được suy ra từ Định lý (2.3.1) và (2.2.1).

3.2. Ước lượng độ lỗi cho đạo hàm số

Trong trường hợp tổng quát, để có ước lượng độ lỗi thực tế cho đạo hàm số, ta trở về với Định lý Rolle mà là cơ sở cho ước lượng độ lỗi nội suy của Định lý (2.3.1). Kết quả này được phát biểu như sau:

Định lý 3.2.1. *Gọi các điểm nội suy có thứ tự $x_0 < x_1 < \cdots < x_n$. Gọi $f^{(n+1}(x)$ liên tục. Thì với mọi $k \leq n$,*

$$R_n^{(k)}(x) = \prod_{j=0}^{n-k}(x - \xi_j)\frac{f^{(n+1)}(\eta)}{(n+1-k)!} \tag{3.6}$$

$R_n(x)$	$R_n^{(1)}$	$R_n^{(2)}$	$\dots$	$R_n^{(1)}$
x_0				
x_1	(x_0, x_1)			
x_2	(x_2, x_3)	(x_1, x_3)		
$\vdots$	$\vdots$	$\vdots$		
x_k	(x_{k-1}, x_k)	(x_{k-2}, x_k)	$\dots$	(x_0, x_k)
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_n	(x_{n-1}, x_n)	(x_{n-2}, x_n)	$\dots$	(x_{n-k}, x_n)

Bảng 3.1: *Không điểm của các đạo hàm cấp cao của* $R_n(x)$

trong đó với $n+1-k$ *điểm phân biệt,* ξ_j*, độc lập với* x *và nằm trong khoảng*

$$x_j < \xi_j < x_{j+k}, j = 0, 1, \dots, n-k \tag{3.7}$$

và $\eta = \eta(x)$ *là một số điểm trong khoảng chứa* x *và* ξ_j*.*

Chứng minh. Bởi vì $R_n(x) = f(x) - P_n(x)$ có đạo hàm liên tục cấp $n+1$ và tiêu biến tại $x = x_j, j = 0, 1, \dots, n$, ta có thể áp dụng Định lý Rolle $k \leq n$ lần. Bằng cách lập bảng, ta có thể theo dõi được vị trí của không điểm của các đạo hàm cấp cao của $R_n(x)$ như Bảng (3.1).

Cột thứ k trong bảng liệt kê các khoảng mở, (x_j, x_{j+k}) trong đó có ích nhất một ξ_j phân biệt của $R_n^{(k)}$ phải nằm trong. Do đó điểm ξ_j của khoảng $x_j < \xi_j < x_{j+k}, j = 0, 1, \dots, n-k$ được định nghĩa, và ta chú ý rằng chúng chỉ phụ thuộc vào hàm $f(x)$ và các điểm nội suy x_j chứ không phụ thuộc vào x.

Định nghĩa một hàm

$$F(z) = R_n^{(k)}(z) - \alpha \prod_{j=0}^{n-k} (z - \xi_j) \tag{3.8}$$

Ta nhận thấy rằng

$$F(\xi_j) = 0, j = 0, 1, 2, \dots, n-k \tag{3.9}$$

Với bất kỳ x cố định phân biệt từ ξ, ta chọn được $\alpha = \alpha(x)$ sao cho $F(x) = 0$ thì $F(z)$ có $n - k + 2$ không điểm và ta có thể áp dụng được Định lý Rolle một lần nữa. Ta suy ra rằng $F^{(n-k+1)}(z)$ có nghiệm, tại η trong khoảng chứa x và ξ_j

$$0 = F^{(n-k+1)}(z) = R_n^{(n+1}(\eta) - \alpha(n-k+1)! = f^{(n+1)}(\eta) - \alpha(n-k+1)! \tag{3.10}$$

hay

$$\alpha = \frac{f^{(n+1)}(\eta)}{(n-k+1)!} \tag{3.11}$$

Bằng cách sử dụng giá trị α trong phương trình $F(z) = 0$, kết quả (3.6) đúng cho mọi x. Cụ thể là, (3.6) thỏa mãn $x = \xi_j$ với η bất kỳ vì $F(\xi_j) = 0$ với bất kỳ α. Chứng minh định lý hoàn tất. □

Dạng Larange của đa thức nội suy là tiện lợi để thiết kế công thức vi phân số. Để xấp xỉ đạo hàm của $f(x)$ tại điểm z, cho trước các giá trị x_j, ta thiết lập nội suy, đạo hàm của nó, và đánh giá nó tại z:

$$f^k(z) \approx P_n^{(k)}(z) = \sum_{j=0}^{N} f(x_j) \left[\prod_{k=0, k\neq j}^{N} \frac{x - x_k}{x_j - x_k} \right]^{(k)} \tag{3.12}$$

Bởi vì các hệ số trong biểu thức này chỉ phụ thuộc vào các nút, ta có ở đây một công thức mà có thể dùng cho bất kỳ hàm $f(x)$ nào. Một số ví dụ sẽ được trình bày trong phần **Một số ví dụ**.

3.3. Chặn sai số cho ước lượng độ lỗi cho đạo hàm số

Biểu diễn độ lỗi trong đạo hàm số của Biểu thức (3.6) đúng với mọi x và cũng có khả tính tổng quát hơn biểu diễn trong Biểu thức (3.5). Bằng cách sử dụng các khoảng đã biết (3.7), nó hoàn toàn khả thi để thu được những chặn sai số. Cụ thể, nếu x và x_j đều nằm trong $[a, b]$ và trong khoảng này $\left|f^{(n+1)}(x)\right| \leq M$

thì rõ ràng thấy được

$$\left|R_n^{(k)}(x)\right| \leq M\frac{|b-a|^{n-k+1}}{(n-k+1)!} \tag{3.13}$$

Một ước lượng đẹp đẽ có thể thu được bằng cách sử dụng hiệu quả các bất đẳng thức trong Biểu thức (3.7) trong việc chặn hệ số độ lõi, $\prod_{j=0}^{n-k}(x-\xi_j)$

3.4. Phương pháp hệ số bất định

Có nhiều cách khác để xác định biểu thức đạo hàm số và độ lỗi của chúng. Giả định rằng giá trị $f^{(k)}(a)$ được xấp xỉ bằng cách sử dụng giá trị $f(x_i), i = 1, 2, \ldots, m$. Với một $f(x)$ có đạo hàm liên tục cấp $n+1$ trong đó $n+1 \geq m$, ta định nghĩa $h_i \equiv x_i - a$ và sử dụng khai triển Taylor

$$\begin{aligned} f(x_i) &= f(a+h_i) \\ &= f(a) + h_i f^{(1)}(a) + \frac{h_i^2}{2!}f^{(2)}(a) + \cdots + \frac{h_i^n}{n!}f^{(n)}(a) + \frac{h_i^{n+1}}{(n+1)!}f^{(n+1)}(a+\theta_i h_i) \end{aligned} \tag{3.14}$$

trong đó $i = 1, 2, \ldots, m$, $0 < \theta_i < 1$.

Ta hình thành được một tổ hợp tuyến tính của các đẳng thức với trong số α_i đã được xác định.

$$\begin{aligned} \sum_{i=1}^{m}\alpha_i f(x_i) = &\left(\sum_{i=1}^{m}\alpha_i\right) f(a) + \left(\sum_{i=1}^{m}\alpha_i h_i\right) f^{(1)}(a) + \ldots \\ &+ \left(\sum_{i=1}^{m}\alpha_i h_i^n\right)\frac{f^{(n)}(a)}{n!} + \frac{1}{(n+1)!}\left(\sum_{i=1}^{m}\alpha_i h_i^{n+1} f^{(n+1)}(a+\theta_i h_i)\right) \end{aligned} \tag{3.15}$$

Ta chọn α_i để mà tổ hợp tuyến tính của các giá trị $f(x_i)$ xấp xỉ chính xác đến $f^{(k)}(a)$. Do đó, ta áp đặt m điều kiện trên m ẩn chưa biết α_i

$$\sum_{i=1}^{m}\alpha_i h_i^v = v!\delta_{vk}, v = 0, 1, \ldots, m-1 \tag{3.16}$$

Rõ ràng thấy được hệ này có một nghiệm duy nhất do các hệ số của định thức là một định thức Vandermonde. Do đó, một điều kiện cần và đủ cho hệ này để có một nghiệm không tầm thường là không đồng nhất, tức là $m > k$. Thế nên, **để xấp xỉ một đạo hàm cấp k, ta cần nhiều hơn k điểm**. Với nghiệm của hệ trên, ta có:

$$f^{(k)}(a) = \sum_{i=1}^{m} \alpha_i f(x_i) - \frac{1}{m!}\left(\sum_{i=1}^{m} \alpha_i h_i^m\right) f^{(m)}(a) - \dots \\ - \frac{1}{n!}\left(\sum_{i=1}^{m} \alpha_i h_i^n\right) f^{(n)}(a) - \frac{1}{(n+1)!}\left(\sum_{i=1}^{m} \alpha_i h_i^{n+1} f^{(n+1)}(a + \theta_i h_i)\right), m > k \tag{3.17}$$

Thủ tục này tương đương với *phương pháp hệ số bất định*: nếu ta đã biết m giá trị hàm, $f(x_i)$, ta tìm kiếm tổ hợp tuyến tính của các giá trị tại những điểm này mà cho giá trị chính xác của đạo hàm $f^{(k)}(a)$ cho tất cả đa thức tại điểm cố định a. Một cách cụ thể, với đạo hàm đầu tiên, bởi vì

$$\left.\frac{dx^v}{dx}\right|_{x=a} = va^{v-1} \tag{3.18}$$

ta tìm α_i sao cho

$$\sum_{i=1}^{m} \alpha_i x_i^v = va^{v-1}, v = 0, 1, \dots, m-1 \tag{3.19}$$

Hệ phương trình này cũng có một nghiệm duy nhất và thật nó nó trùng với nghiệm của hệ phương trình

$$\sum_{i=1}^{m} \alpha_i h_i^v = v!\delta_{vk}, v = 0, 1, \dots, m-1 \tag{3.20}$$

với $k = 1$.

3.5. Đạo hàm sử dụng các điểm cách đều

Một cách tự nhiên, công thức đạo hàm số đơn giản hơn khi các điểm dữ liệu cách đều nhau. Cụ thể, toán tử dơn vị (operator identity) sinh ra các xấp xỉ có dạng như sau:

$$f'(x) = \frac{1}{h}[\Delta f(x) - \frac{1}{2}\Delta^2 f(x) + \cdots + (-1)^{n+1}\frac{1}{n}\Delta^n f(x)] + R'_n(x) \tag{3.21}$$

Trong trường hợp này, dữ liệu cần có dạng $f(x), f(x+h), \ldots, f(x+nh)$, và nên biểu thức này chỉ xấp xỉ đạo hàm tại một điểm và sử dụng dữ liệu trên một phía của điểm này. Công thức này thu được bằng cách đạo hàm số gia tiến Newton theo biểu thức

$$Q_n(x_0 + th) = f(x_0) + \frac{\pi_0(t)}{1!}\Delta f(x_0) + \frac{\pi_1(t)}{2!}\Delta^2 f(x_0) + \cdots + \frac{\pi_{n-1}(t)}{n!}\Delta^n f(x_0) \tag{3.22}$$

và đánh giá tại $t = 0$. Do đó, độ lỗi trong biểu thức (3.5) trở thành

$$R'_n(x) = \frac{(-1)^n h^n}{n+1} f^{(n+1)}(\eta), x < \eta < x + nh \tag{3.23}$$

Một ví dụ khác có thể được minh họa bằng cách đạo hàm dạng Gaussian của đa thức nội suy với $n = 2m$, và thiết lập tại $t = 0$,

$$f'(x_0) = \frac{1}{h}\left[\Delta f(x_0) - \frac{1}{2!}\Delta^2 f(x_{-1}) - \frac{1}{3!}\Delta^3 f(x_{-1}) + \cdots + (-1)^m \frac{m!(m-1)!}{(2m)!}\Delta^{2m} f(x_{-m})\right] + R'_{2m}(x_0) \tag{3.24}$$

Những điểm liên quan được thay thế một cách đối xứng tại x_0 và công thức độ lỗi

$$R'_{2m}(x_0) = -(1)^m \frac{(m!)^2}{(n+1)!} h^n f^{(n+1)}(\eta), x_0 - mh < \eta < x_0 + mh, \tag{3.25}$$

Với n và m tiến gần về 1, xấp xỉ Stirling cho $n!$ chỉ ra rằng khi $n = 2m$

$$\frac{(m!)^2}{(n+1)!} \approx \frac{n^{1/2}\sqrt{2\pi}}{2^{n+1}} \tag{3.26}$$

Khi so sánh hai độ lỗi (3.23) và (3.25) chỉ ra rằng với đạo hàm, ưu tiên hơn những điểm dữ liệu trung tâm so với điểm xấp xỉ.

Một trường hợp đặc biệt quan trọng của phương trình

$$f'(x_0) = \frac{1}{h}\left[\Delta f(x_0) - \frac{1}{2!}\Delta^2 f(x_{-1}) - \frac{1}{3!}\Delta^3 f(x_{-1}) + \cdots + (-1)^m \frac{m!(m-1)!}{(2m)!}\Delta^{2m} f(x_{-m})\right] + R'_{2m}(x_0) \tag{3.27}$$

là khi $n = 2$, nó có thể được viết lại

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - \frac{h^2}{6} f^3(\eta), x - h < \eta < x + h, \tag{3.28}$$

Đây là biểu thức xấp xỉ gia số trung tâm (centered difference approximation) đến đạo hàm cấp một (first derivative).

Đối với đạo hàm cấp hai, hay thậm chí đạo hàm cấp cao, có thể được xấp xỉ bởi một biểu thức trung tâm bằng cách đạo hàm những đa thức nội suy Gaussian với $n = 2m + 1$. Lấy ví dụ, với $n = 3$, xấp xỉ của $f''(x_0)$ trở thành trên thiết lập $x_0 = x$:

$$\begin{aligned} f''(x) &= \frac{\Delta^2 f(x-h)}{h^2} + R_3^{(2)}(x) \\ &= \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} + R_3^{(2)}(x) \end{aligned} \tag{3.29}$$

Bằng cách áp dụng Định lý (3.2.1), ta có thể ước lượng được độ lỗi như sau:

$$\left|R_3^{(2)}(x)\right| \leq h^2 \left|f^{(4)}(\eta)\right|, x - h < \eta < x + 2h \tag{3.30}$$

Nhưng ta có thể tìm thấy một chặn sai số tốt hơn bằng cách sử dụng khai triển Tayolor. Nếu ta đặt $k = 2, m = 4, a = x,$ và $h_i = (i-2)h, i = 1, 2, 3, 4$ thì, ta giải hệ

$$\sum_{i=1}^{m} \alpha_i h_i^v = v!\delta_{vk}, v = 0, 1, \ldots, m-1 \tag{3.31}$$

và từ

$$\begin{aligned} f^{(k)}(a) = &\sum_{i=1}^{m} \alpha_i f(x_i) - \frac{1}{m!}\left(\sum_{i=1}^{m} \alpha_i h_i^m\right) f^{(m)}(a) - \ldots \\ &- \frac{1}{n!}\left(\sum_{i=1}^{m} \alpha_i h_i^n\right) f^{(n)}(a) - \frac{1}{(n+1)!}\left(\sum_{i=1}^{m} \alpha_i h_i^{n+1} f^{(n+1)}(a+\theta_i h_i)\right), m > k \end{aligned} \tag{3.32}$$

suy ra

$$\alpha_1 = \alpha_3 = \frac{1}{h^2}, \alpha_2 = -\frac{2}{h^2}, \alpha_4 = 0 \tag{3.33}$$

Phần độ lỗi trở thành

$$R_3^{(2)}(x) = -\frac{h^2}{4!}\left[f^{(4)}(\xi_1) - f^{(4)}(\xi_2)\right], x - h < \xi_1 < x < \xi_2 < x + h \tag{3.34}$$

Nhưng mà $f^{(4)}(x)$ được giả định là liên tục trong đạo hàm này nên với một số ξ trong khoảng $\xi_1 < \xi < \xi_2$, phải có

$$\frac{1}{2}\left[f^{(4)}(\xi_1) - f^{(4)}(\xi_2)\right] = f^{(4)}(\xi) \tag{3.35}$$

Vậy độ lỗi có công thức là

$$R_3^{(2)}(x) = -\frac{h^2}{4!} f^{(4)}(\xi), x - h < \xi < x + h \tag{3.36}$$

3.6. Một số ví dụ

Xem xét bài toán xấp xỉ đạo hàm cấp 1 bằng nội suy Larange. Ta đã biết trước các nút nội suy như sau:

$$\begin{aligned} &(x_0, f(x_0)) \\ &(x_1, f(x_1)) \\ &(x_2, f(x_2)) \\ &\dots \\ &(x_N, f(x_N)) \end{aligned} \tag{3.37}$$

Dựa trên Định lý nội suy Lagrange 2.1.1, ta có:

$$P_n(x) = \sum_{j=0}^{N} f(x_j) \prod_{k=0, k\neq j}^{N} \frac{x - x_k}{x_j - x_k} + \frac{1}{(N+1)!} \prod_{j=0}^{N} (x - x_j) f^{N+1}(\xi(x)) \tag{3.38}$$

Lấy đạo hàm cấp 1 cho biểu thức trên

$$\begin{aligned} P_n'(x) = &\sum_{j=0}^{N} f(x_i) \left(\frac{d}{dx} \prod_{k=0, k\neq j}^{N} \frac{x - x_k}{x_j - x_k} \right) \\ &+ \frac{1}{(N+1)!} \prod_{j=0}^{N} (x - x_j) \left(\frac{d\left(f^{N+1}(\xi(x))\right)}{dx} \right) \\ &+ \frac{1}{(N+1)!} \left(\frac{d}{dx} \prod_{j=0}^{N} (x - x_j) \right) f^{N+1}(\xi(x)) \end{aligned} \tag{3.39}$$

Đặt $x = x_t$ với x_t là hoành độ của từng nút nội suy, $t = 0, 1, \dots, N$. Ta có:

$$P_n'(x) = \sum_{j=0}^{N} f(x_j) \left(\frac{d}{dx} \prod_{k=0}^{N} \frac{x - x_k}{x_t - x_k} \right) + \frac{f^{(N+1)}(\xi(x))}{(N+1)!} \prod_{k=0, k\neq j}^{N} (x_t - x_k) \tag{3.40}$$

Biểu thức độ lỗi

$$R_n(x_t)' = \frac{f^{(N+1)}(\xi(x))}{(N+1)!} \prod_{k=0,k\neq j}^{N} (x_t - x_k) \tag{3.41}$$

Ví dụ: Tìm công thức ba điểm với độ lỗi để xấp xỉ đạo hàm cấp một $f'(x_j)$

Gọi các nút nội suy là $(x_0, f(x_0)), (x_1, f(x_1)), (x_2, f(x_2))$

$$\begin{aligned} f'(x_j) = & f(x_0)\left[\frac{2x_j - x_1 - x_2}{(x_0 - x_1)(x_0 - x_2)}\right] \\ & + f(x_1)\left[\frac{2x_j - x_0 - x_2}{(x_1 - x_0)(x_1 - x_2)}\right] \\ & + f(x_2)\left[\frac{2x_j - x_0 - x_1}{(x_2 - x_0)(x_2 - x_1)}\right] \\ & + \frac{f^{(3)}(\xi(x))}{6} \prod_{k=0,k\neq j}^{2} (x_j - x_k) \end{aligned} \tag{3.42}$$

CHƯƠNG 4

PHƯƠNG PHÁP LẬP TRÌNH

Trong chương này, vấn đề phương pháp lập trình MATLAB cho bài toán xấp xỉ đạo hàm bằng đạo hàm của đa thức nội suy được tập trung trình bày.

4.1. Phương pháp

Một phương pháp nội suy đa thức phổ biến là nội suy đa thức Lagrange. Ta có thể sử dụng đạo hàm của đa thức Lagrange để tính toán đạo hàm số. Các bước thực hiện như sau:

1. Chọn các điểm: ta chọn $n+1$ điểm $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ gần điểm mà ta mong muốn xấp xỉ đạo hàm x. Các giá trị y_i là các giá trị của hàm tại điểm x_i.

2. Xây dựng đa thức nội suy Lagrange: đa thức $P(x)$ qua các điểm nội suy được cho bởi công thức:

$$P(x) = \sum_{i=0}^{n} y_i L_i(x) \tag{4.1}$$

   trong đó $L_i(x)$ là các đa thức Lagrange cơ sở

$$L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \tag{4.2}$$

3. Đạo hàm đa thức: đạo hàm đa thức $P(x)$ tương ứng với biến x để tìm $P'(x)$, tức là đạo hàm của đa thức nội suy. Đạo hàm của mỗi hàm Lagrange cơ sở

được cho bởi công thức:

$$L_i'(x) = \sum_{j=0, j\neq i}^{n} \prod_{k=0, k\neq i, k\neq j} \frac{x - x_k}{x_i - x_k} \tag{4.3}$$

Đạo hàm của đa thức được tính

$$P'(x) = \sum_{i=0}^{n} y_i L_i'(x) \tag{4.4}$$

4. Đánh giá đạo hàm tại điểm mục tiêu. Để xấp xỉ đạo hàm của hàm gốc tại x, ta thế ngược x vào $P'(x)$.

4.2. Thực nghiệm

Xem xét thí nghiệm với hàm Runge:

$$f(x) = \frac{1}{1 + 25x^2} \tag{4.5}$$

Ta sử dụng nội suy đa thức tại $N = 2m+1$ điểm cách đều $x_j = -5+5(j-1)/m$ cho xấp xỉ xấu hàm Runge trên đoạn $[-5, 5]$. Sau đó, sử dụng đa thức nội suy để xấp xỉ đạo hàm cho hàm Runge tại điểm $z = 0.5$.

Ta thực hiện từng bước như sau:

1. Bước 1: Định nghĩa hàm Runge và đạo hàm của nó

```
% Define the Runge function and its derivative
syms x;
f = 1 / (1 + 25*x^2);

% Define range
a = -5;
b = 5;
```

```
m = 13;
N = 2*m+1;

% Differentiate the function symbolically
df = diff(f, x);

% Function handles for numerical evaluations
f_func = matlabFunction(f);
df_func = matlabFunction(df);

% Target point for evaluating the derivative
x_target = 0.5;
actual_derivative = df_func(x_target);
```

2. Bước 2: Phát sinh các điểm và thực hiện nội suy Larange

```
% Points for interpolation
xs = linspace(a, b, N); % 15 points from -2 to 2
ys = f_func(xs);

% Lagrange Interpolating Polynomial
L = 0;
n = length(xs);
for i = 1:n
    Li = 1;
    for j = 1:n
        if i ~= j
            Li = Li * (x - xs(j))/(xs(i) - xs(j));
        end
    end
```

```
    L = L + Li * ys(i);
end

% Differentiate the interpolating polynomial
L_prime = diff(L, x);

% Evaluate the derivative of the Lagrange
   polynomial at the target point
derivative_at_target = double(subs(L_prime, x,
   x_target));
```

3. Bước 3: Tính toán độ lỗi giữa đạo hàm thật sự và đạo hàm xấp xỉ được từ đa thức Larange.

```
% Calculate error
error = abs(derivative_at_target -
   actual_derivative);

fprintf('The actual derivative of the Runge
   function at x = %.2f is: %.4f\n', x_target,
   actual_derivative);
fprintf('The approximate derivative from Lagrange
   interpolation at x = %.2f is: %.4f\n', x_target,
   derivative_at_target);
fprintf('The error in the derivative approximation
   is: %.4f\n', error);
```

4.2.1. Thí nghiệm với xấp xỉ xấu

Kết quả thực nghiệm với các giá trị m lần lượt là $m = 7, m = 10, m = 13, m = 15, m = 17$ như sau

m	N	Đạo hàm đúng	Đạo hàm xấp xỉ	Sai số trên đoạn $[-5, 5]$
7	15	-0.4756	-1.7975	1.3219
10	21	-0.4756	-1.9744	1.4988
13	27	-0.4756	-1.1288	0.6532
15	31	-0.4756	-0.3658	0.1099
17	35	-0.4756	0.2162	0.6918

Bảng 4.1: *Bảng đánh giá thực nghiệm xấp xỉ đạo hàm cho hàm Runge tại* $z = 0.5$ *bằng nội suy đa thức Larange trên đoạn* $[-5, 5$.

Nhận xét: Giá trị xấp xỉ đạo hàm không quá tốt. Mặc dù, ta đã cố gắng tăng các điểm nội suy lên nhưng kết quả xấp xỉ vẫn không có nhiều cải thiện.

Ta lập lại thí nghiệm nhưng lần này chỉ tính trên đoạn $[-1, 1]$. Kết quả thực nghiệm với các giá trị m lần lượt là $m = 7, m = 10, m = 13, m = 15, m = 17$ như sau:

m	N	Đạo hàm đúng	Đạo hàm xấp xỉ	Sai số trên đoạn $[-5, 5]$
7	15	-0.4756	-0.4775	0.0019
10	21	-0.4756	-1.1795	0.7039
13	27	-0.4756	-0.3734	0.1023
15	31	-0.4756	-0.5383	0.0627
17	35	-0.4756	-0.4381	0.0376

Bảng 4.2: *Bảng đánh giá thực nghiệm xấp xỉ đạo hàm cho hàm Runge tại* $z = 0.5$ *bằng nội suy đa thức Larange trên đoạn* $[-1, 1$.

Nhận xét: Khi thu nhỏ khoảng nội suy và cố gắng tăng số lượng các điểm nội suy, chúng ta thu được kết quả xấp xỉ đạo hàm tốt hơn thí nghiệm trước đó. Tuy nhiên, không có sự ổn định khi tăng từ 15 lên 21 và lên 27 điểm nội suy.

4.2.2. Thí nghiệm với xấp xỉ tốt

Ta thực hiện thí nghiệm tương tự nhưng lần này ta dùng nội suy đa thức tại các điểm Chebyshev cho xấp xỉ tốt hàm Runge trên đoạn $[-5, 5]$. Kết quả thực nghiệm với các giá trị m lần lượt là $m = 7, m = 10, m = 13, m = 15, m = 17$ như sau:

m	N	Đạo hàm đúng	Đạo hàm xấp xỉ	Sai số trên đoạn $[-1, 1]$
7	15	-0.4756	-0.5265	0.0508
10	21	-0.4756	-0.4602	0.0154
13	27	-0.4756	-0.4803	0.0047
15	31	-0.4756	-0.5198	0.0441
17	35	-0.4756	-0.4521	0.0235

Bảng 4.3: *Bảng đánh giá thực nghiệm xấp xỉ tốt đạo hàm cho hàm Runge tại* $z = 0.5$ *bằng nội suy đa thức Larange trên đoạn* $[-1, 1$.

Nhận xét: Khi ta sử dụng khoảng nhỏ $[-1, 1]$ và sử dụng các điểm nội suy Chebyshev, các kết quả xấp xỉ đạo hàm tốt với độ lỗi nhỏ.

m	N	Đạo hàm đúng	Đạo hàm xấp xỉ	Sai số trên đoạn $[-5, 5]$
7	15	-0.4756	73945.3885	73945.8641
10	21	-0.4756	-63.3438	62.8681
13	27	-0.4756	-6.2810	5.8054
15	31	-0.4756	-1.8440	1.3684
17	35	-0.4756	-1.5730	1.0974

Bảng 4.4: *Bảng đánh giá thực nghiệm xấp xỉ tốt đạo hàm cho hàm Runge tại* $z = 0.5$ *bằng nội suy đa thức Larange trên đoạn* $[-5, 5$.

Nhận xét: Khi ta sử dụng khoảng lớn $[-5, 5]$ và sử dụng các điểm nội suy Chebyshev, các kết quả xấp xỉ đạo hàm chưa thật sự tốt và độ lỗi còn cao.

CHƯƠNG 5

KẾT LUẬN VÀ HƯỚNG NGHIÊN CỨU TƯƠNG LAI

Trong tiểu luận này, phương pháp xấp xỉ cho việc tính toán đạo hàm số dựa trên các đa thức nội suy được trình bày. Trong đó, phương pháp lập trình với nội suy Larange được khảo sát. Thông qua các kết quả đã đạt được, ta nhận thấy được một số vấn đề trong việc lựa chọn số điểm nội suy và từ đó thu được đa thức nội suy để nhằm đánh giá đạo hàm. Một số lưu ý trong thực hành, ta nhận thấy không phải lúc nào dùng nhiều điểm nội suy cũng tốt và sử dụng các điểm xấp xỉ tốt như các điểm Chebyshev là một trong những lựa chọn ưu tiên.

Trong kế hoạch nghiên cứu tương lai, phương pháp đạo hàm số vẫn còn có nhiều đề tài thú vị và cần được nghiên cứu như áp dụng và cài đặt các phương pháp khớp đa thức trực tiếp (direct fit polynominal), phương pháp Newton, phương pháp sai phân chia được. Bên cạnh đó, có thể phân tích thêm về độ lỗi hoặc sử dụng các phương pháp đa thức nội suy khác như nội suy Hermite.

TÀI LIỆU THAM KHẢO

Tiếng Anh:

[1] A Björck and Germand Dahlquist (2008), "Numerical methods in scientific computing volume II", *Society for Industrial and Applied Mathematics, Philadelphia, PA*.

[2] Steven C Chapra (2018), *Applied numerical methods with MATLAB for engineers and scientists*, Mcgraw-hill.

[3] Germund Dahlquist and Åke Björck (2008), *Numerical methods in scientific computing, volume I*, SIAM.

[4] Eugene Isaacson and Herbert Bishop Keller (1994), *Analysis of numerical methods*, Courier Corporation.

[5] Jaan Kiusalaas (2009), *Numerical Methods in Engineering with MATLAB®*, 2nd ed., Cambridge University Press.

PHỤ LỤC 1: NỘI SUY HERMITE VÀ THIẾT LẬP CÁC CHẶN SAI SỐ

.1. Nội suy Hermite

Giả định rằng tại một điểm cho trước x_j, ta biết một giá trị hàm y_i, và có thể có nhiều giá trị đạo hàm $y'_j, y''_j, \ldots, y_j^{m_j}$. Ta có thể sử dụng ma trận Vandermonde để chứng tỏ rằng tồn tại một đa thức duy nhất P với bậc không nhỏ hơn d mà thỏa mãn

$$P^{(k)}(x_j) = y_j^{(k)}, k = 0, 1, \ldots, m_j, j = 0, 1, \ldots, n \tag{1}$$

trong đó $d = n + m_0 + m_1 + \cdots + m_m$. Đa thức osculating (kissing) là một đa thức nội suy rất tổng quát.

Thật vậy, ta có thể đặt tên cho loại đa thức xấp xỉ được gắn với mỗi loại dữ liệu và cho bậc của chúng.

1. $n > 0$, $m_j = 0$, $j = 0, 1, \ldots, n$

2. $n > 0$, $m_j = 1$, $j = 0, 1, \ldots, n$

3. $n = 0$, $n_0 = N$

Khối Vandermonde liên kết với thành phần j bây giờ có $1 + m_j$ dòng, và

$$\begin{bmatrix} 1 & x & x^2 & x^3 & \ldots & x^d \\ 0 & 1 & 2x & 3x^2 & \ldots & dx^{d-1} \\ 0 & 0 & 2 & 6x & \ldots & d(d-1)x^{d-2} \\ & & & & \vdots & \end{bmatrix} \tag{2}$$

được đánh giá tại $x = x_j$. Ma trận Vandermonde $(d + 1) \times (d + 1)$ này là không suy biến nếu và chỉ nếu x_j phân biệt.

Trong một số ứng dụng, y_i là những giá trị của một hàm đã biết f. Nếu $f \in C^{d+1}([a,b])$, và $[a,b]$ chứa tất cả các điểm, thì $\forall x \in [a,b], \exists \xi \in [a,b]$ sao cho

$$f(x) = P(x) + \frac{f^{(d+1)}(\xi)}{(d+1)!} \prod_{j=0}^{n} (x - x_j)^{m_j+1} \tag{3}$$

Việc sử dụng một đa thức để nội suy một tập hợp lớn các điểm (hoặc một số lượng lớn các điều kiện trên một tập hợp điểm) đòi hỏi bộ nội suy phải có bậc lớn. Điều này thường tạo ra các dao động lớn và không mong muốn trong bộ nội suy và khiến việc đánh giá $P(x)$ tốn kém hơn. Nếu ta có quyền tự do lựa chọn các nút, chúng có thể được chọn để giảm thiểu các dao động dữ dội; đây được gọi là lựa chọn nút Chebyshev.

PHỤ LỤC 2: LÝ THUYẾT CƠ BẢN VỀ XẤP XỈ ĐA THỨC

Xem xét một không gian tuyến tính (linear space) không nhất thiết phải là không gian hữu hạn chiều (finte dimensional space) mà các phần tử của nó là các hàm $\{f(x)\}$. Chuẩn (norm) được định nghĩa là một phép gán một số thực vào mỗi phần tử của không gian tuyến tính trên, ký hiệu $\text{Norm}(f) \equiv N(f) \equiv \|f\|$ thỏa mãn:

- $\|f\| \geq 0$,
- $\|f\| = 0$ nếu và chỉ nếu $f(x) \equiv 0$,
- $\|cf\| = |c| \cdot \|f\|$, với mọi hằng số $c \in \mathbb{R}$,
- $\|f + g\| \leq \|f\| + \|g\|$

Một độ đo của độ lệch chuẩn (measure of the deviation) hay độ lỗi (error) trong một xấp xỉ của $f(x)$ bởi $P_n(x)$ được định nghĩa một cách tổng quát bởi khái niệm bán chuẩn (semi-norm):

$$|f(x) - P_n(x)|_{sn} \tag{4}$$

Ta giả định rằng các đa thức và hàm, $f(x)$, cần được xấp xỉ trong một không gian tuyến $C[a, b]$ của các hàm đươc định nghĩa trên một khoảng bị chặn đóng, $[a, b]$.

Định lý .1.1. *Gọi một độ đo của độ lệch chuẩn $|.|_{sn}$ được định nghĩa trong $C[a, b]$, và tồn tại các số dương m_n và M_n thỏa mãn:*

$$0 < m_n \leq \left| \sum_{j=0}^{n} b_j x^j \right|_{sn} \leq M_n, n = 0, 1, \ldots \tag{5}$$

với mọi $\{b_j\}$ *mà thỏa mãn*

$$\sum_{j=0}^{n} b_j = 1 \tag{6}$$

Thì với bất kỳ số nguyên n *và* $f(x)$ *trong* $C[a,b]$*, tồn tại một đa thức bậc lớn nhất* n *mà*

$$d_n = |f(x) - P_n(x)|_{sn} \tag{7}$$

đạt được giá trị nhỏ nhất trên tất cả các đa thức.

Một nhận xét: hàm $f(x)$ không nhất thiết phải liên tục. Hơn nữa, định lý trên không cho một ước lượng về độ lớn của d_n.

Về các kết quả về tính duy nhất, ta cần độ đo của độ lệch chuẩn nên nghiêm ngặt (strict). Bằng cách định nghĩa một chuẩn $|.|_{sn}$ là nghiêm ngặt nếu

$$|f+g|_{sn} = |f|_{sn} + |g|_{sn} \tag{8}$$

dẫn đến tồn tại các hằng α, β mà $|\alpha| + |\beta| \neq 0$ và

$$\alpha f(x) + \beta g(x) \equiv 0 \tag{9}$$

Ta có định lý sau.

Định lý .1.2. *Giả thiết của định lý .1.1 thêm vào yêu cầu* $|.|_{sn}$ *là nghiêm ngặt. Thì đa thức nhỏ nhất, gọi là* $P_n(x)$ *là duy nhất.*

.2. Định lý xấp xỉ Weierstrass và Đa thức Bernstein

Định lý xấp xỉ Weierstrass được phát biểu như sau:

Định lý .2.1. *Gọi* $f(x)$ *là bất kỳ hàm liên tục nào trong khoảng (đóng)* $[a,b]$*. Thì với bất kỳ* $\epsilon > 0$*, tồn tại một số nguyên* $n = n(\epsilon)$ *và một đa thức* $P_n(x)$ *với bậc cao nhất* n *thỏa*

$$|f(x) - P_n(x)| < \epsilon \tag{10}$$

với mọi $x \in [a, b]$

Định lý .2.1 đảm bảo có thể xấp xỉ đa thức gần thông qua một khoảng giới hạn đóng chỉ với điều kiện là hàm được xấp xỉ là liên tục. Phát biểu của định lý là về sự tồn tại và không cho một gợi ý nào về cách xây dựng những xấp xỉ. Tuy nhiên, một chứng minh đơn giản và tao nhã của kết quả này do Bernstein trình bày trong định lý

Đa thức Bernstein bậc n cho hàm $f(x)$ trên $[0, 1]$ được định nghĩa:

$$B_n(f; x) \equiv \sum_{j=0}^{n} f(x_j)\beta_{n,j}(x) \tag{11}$$

với

$$\beta_{n,j}(x) = \binom{n}{j} x^j (1-x)^{n-j} \tag{12}$$

Định lý .2.2. *Gọi* $f(x)$ *là bất kỳ hàm liên tục nào được định nghĩa trên* $[0, 1]$. *Thì với mọi* $x \in [0, 1]$, *và bất kỳ số nguyên dương* n *nào,*

$$|f(x) - B_n(f; x)| \leq \frac{9}{4}\omega(f; n^{-1/2}) \tag{13}$$

trong đó modulus của tính liên tục của $f(x)$ *trong* $[0, 1]$ *được định nghĩa*

$$\omega(f; \delta) = \underset{x,x' \in [0,1], |x-x'| \leq \delta}{\textit{Least-upper-bound}} \left| f(x) - f'(x) \right| \tag{14}$$

Định lý xấp xỉ Weierstrass được suy ra nhờ chọn n đủ lớn sao cho $\omega(f; n^{-1/2}) < \frac{4\epsilon}{9}$.

Nếu $f(x)$ thỏa mãn điều kiện Lipschitz, ta dễ dàng tìm được

Hệ quả 9. *Gọi* $f(x)$ *thỏa mãn điều kiện Lipschitz*

$$|f(x) - f(y)| \leq \lambda |x - y| \tag{15}$$

với mọi $x, y \in [0,1]$. *Thì với mọi* $x \in [0,1]$

$$|f(x) - B_n(f;x)| \leq \frac{9}{4}\lambda n^{-1/2}. \tag{16}$$

PHỤ LỤC 3: SAI PHÂN HỮU HẠN

.2.1. Thiết lập sai phân tiến

Khi các điểm nội suy là cách đều, nhiều kết quả được trình bày trong các phần trước được lược giản và thu được nhiều hệ quả bổ sung quan trọng. Ta lấy một điểm x_0 là một điểm cố định bất kỳ và gọi $h > 0$ là khoảng cách giữa các điểm kề nhau. Thì các điểm cần được xem xét là

$$x_j = x_0 + jh, j = 0, \pm 1, \pm 2, \ldots \tag{17}$$

Liên hệ với các điểm cách đều là sai phân tiến (forward difference) được định nghĩa

$$\Delta f(x) = f(x + h) - f(x) \tag{18}$$

Các sai phân bậc cao hơn được định nghĩa như sau

$$\Delta^{n+1} f(x) = \Delta^n [\Delta f(x)] = \Delta^n f(x + h) - \Delta^n f(x) \tag{19}$$

Một quan hệ giữa các sai phân chia được với tham số cách đều và số gia tiến dễ dàng có được bằng cách lấy x là một điểm bất kỳ của các điểm x_j trong (17), ta có:

$$\Delta f(x_0) = f(x_1) - f(x_0) = (x_1 - x_0) \frac{f(x_1) - f(x_0)}{x_1 - x_0} = hf[x_0, x_1] \tag{20}$$

Dựa trên nguyên lý quy nạp, ta giả định rằng

$$\Delta^n f(x_0) = n! h^n f[x_0, x_1, \ldots, x_n] \tag{21}$$

và thu được

$$\begin{aligned}\Delta^{n+1}f(x_0) &= \Delta^n f(x_1) - \Delta^n f(x_0)\\ &= n!h^n f[x_1, x_2, \ldots, x_{n+1}] - n!h^n f[x_0, x_1, \ldots, x_n]\\ &= n!h^n(x_{n+1} - x_0)\frac{f[x_1, x_2, \ldots, x_{n+1}] - f[x_0, x_1, \ldots, x_n]}{(x_{n+1} - x_0)}\\ &= (n+1)!h^{n+1}f[x_0, x_1, \ldots, x_{n+1}]\end{aligned} \tag{22}$$

Vậy, $\Delta^n f(x_0) = n!h^n f[x_0, x_1, \ldots, x_n]$ đúng với tất cả $n \geq 1$.

Một biểu diễn khác của sai phân tiến có thể thu được bằng cách cụ thể (2.24) trong trường hợp các điểm cách đều. Ta có

$$\begin{aligned}\prod_{j=0, j\neq i}^{n}(x_i - x_j) &= \prod_{j=0, j\neq i}^{n}(i-j)h\\ &= h^n \prod_{j=0}^{i} -1(i-j) \prod_{l=i+1}^{n}(i-l)\\ &= (-1)^{n-i}h^n(i)!(n-i)!\end{aligned} \tag{23}$$

Sử dụng kết quả từ (2.24), ta có:

$$f[x_0, x_1, \ldots, x_n] = \frac{1}{n!h^n}\sum_{i=0}^{n}(-1)^{n-i}\binom{n}{i}f(x_i) = \frac{1}{n!h^n}\sum_{i=0}^{n}(-1)^{n-i}\frac{n!}{i!(n-i)!}f(x_i) \tag{24}$$

Kết hợp các kết quả vừa có được:

$$\Delta^n f(x_0) = \sum_{i=0}^{n}(-1)^{n-i}\binom{n}{i}f(x_i) = \sum_{i=0}^{n}(-1)^{n-i}\frac{n!}{i!(n-i)!}f(x_i) \tag{25}$$

Một biểu diễn cuối cùng cho số gia tiến có thể thu được bằng cách sử dụng phương trình

$$\Delta^n f(x_0) = n!h^n f[x_0, x_1, \ldots, x_n] \tag{26}$$

trong Định lý (2.3.1) khi $x = x_k$, ta thu được kết quả:

$$\Delta^n f(x_0) = h^n f^{(n)}(\xi) \tag{27}$$

.2.2. Phân tích chặn sai số cho số gia tiến

Biểu diễn

$$\Delta^n f(x_0) = h^n f^{(n)}(\xi) \tag{28}$$

đúng với giả định rằng $f(x)$ có đạo hàm cấp n trong một khoảng chỉ định.

Rõ ràng mà nói, số gia tiếp cấp n của một đa thức bậc n là một hằng số và các sai phân bậc cao hơn tiêu biến. Tổng quát hơn, nếu $f(x)$ có các đạo hàm bị chặn, tức là $\left|f^{(n)}\right| \leq M, \forall n$ thì biểu diễn trên chỉ ra rằng

$$|\Delta^n f| \leq M n^n \tag{29}$$

Do đó, nếu $h < 1$, độ lớn của sai phân cấp n của $f(x)$ giảm khi n tăng. Ngược lại, nếu đạo hàm cấp n của $f(x)$ tăng theo n, thì sai phân cấp n cũng sẽ giảm nếu và chỉ nếu h "đủ nhỏ".

.2.3. Đa thức nội suy và phần dư cho các điểm cách đều

Các công thức đa thức nội suy Lagrange và Newton trở nên đơn giản khi các điêm nội suy cách đều nhau. Ta thiết lập

$$x = x_0 + th \tag{30}$$

trong đó, t đo $x - x_0$ trong đơn vị của h và là một số nguyên chỉ ở những điểm x_j.

Các hệ số nội suy Larange có thể được viết lại như sau:

$$\begin{aligned}\phi_{n,j}(x) = \phi_{n,j}(x_0 + th) &= \prod_{k=0,k\neq j}^{n} \frac{x - x_k}{x_j - x_k} \\ &= \prod_{k=0,k\neq j}^{n} \frac{t - k}{j - k} \\ &= \frac{(-1^{n-j}}{n!}\binom{n}{j} \prod_{k=0,k\neq j}^{n} (t - k)\end{aligned} \tag{31}$$

Đặt

$$\begin{aligned}&\pi_0(t) \equiv t, \\ &\pi_n(t) \equiv t(t-1)\cdots(t-n), n = 1, 2, \ldots\end{aligned} \tag{32}$$

Các hệ số nội suy Lagrange trở thành

$$\phi_{n,j}(x_0 + th) = \frac{\pi_n(t)}{n!}\binom{n}{j}\frac{(-1)^{n-j}}{t-j} \tag{33}$$

Công thức đa thức nội suy Lagrange được lược giản thành

$$P_n(x_0 + th) = \frac{\pi_n(t)}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}\frac{f(x_j}{t-j} \tag{34}$$

Dạng Newton của đa thức nội suy cũng được lược giản thành

$$Q_n(x_0 + th) = f(x_0) + \frac{\pi_0(t)}{1!}\Delta f(x_0) + \frac{\pi_1(t)}{2!}\Delta^2 f(x_0) + \cdots + \frac{\pi_{n-1}(t)}{n!}\Delta^n f(x_0) \tag{35}$$

Độ lỗi trong các đa thức nội suy hay phần dư của đa thức nội suy được tính như sau:

$$R_n(x) = R_n(x_0 + th) = \pi_n(t)h^{n+1}f[x_0, \ldots, x_n, x] = \pi_n(t)h^{n+1}\frac{f^{(n+1)}(\xi)}{(n+1)!} \tag{36}$$

Từ định nghĩa, các đa thức $\pi_n(t)$

Bổ đề 2. *Với n lẻ,*

$$\pi_n\left(\frac{n}{2}-\tau\right)=\pi_n\left(\frac{n}{2}+\tau\right), \tag{37}$$

tức là $\pi_n(t)$ đối xứng khi $t=n/2$; Với n chẵn

$$\pi_n\left(\frac{n}{2}-\tau\right)=-\pi_n\left(\frac{n}{2}+\tau\right), \tag{38}$$

tức là $\pi_n(t)$ phản xứng khi $t=n/2$.

Một kết quả khác thể hiện sự so sánh về độ lớn của $\pi_n(t)$ tại nhiều điểm

Bổ đề 3. *Gọi $t+1$ là điểm không thể tách rời trong $0<t+1\leq n/2$. Thì*

$$|\pi_n(t+1)|<|\pi_n(t)| \tag{39}$$

Gọi t là điểm không thể tách rời trong $n/2\leq t<t$. Thì

$$|\pi_n(t)|<|\pi_n(t+1)| \tag{40}$$

.2.4. Công thức nội suy trung tâm

Xem xét độ lỗi của đa thức nội suy cho khoảng cách đều như sau:

$$R_n(x)=R_n(x_0+th)=\pi_n(t)h^{n+1}f[x_0,\ldots,x_n,x]=\pi_n(t)h^{n+1}\frac{f^{(n+1)}(\xi)}{(n+1)!} \tag{41}$$

Độ lỗi này có thể được ước lượng nếu $f^{(n+1)}(\xi)$ không "quá lớn" trong khoảng $\min(x_0,x)<\xi<\max(x_n,x)$. Nếu phương sai không quá lớn, thì một ước lượng của độ lỗi mà ta có thể sử dụng được là:

$$R_n(x)\equiv R_n(x_0+th)\cong\frac{\pi_n(t)}{(n+1)!}\Delta^{n+1}f(x_0) \tag{42}$$

Một xấp xỉ chặn sai số thu được hay không khi $\pi_n(t)$ được thay thế bởi giá trị cực đại tuyệt đối của nó trong khoảng này là một câu hỏi.

Tổng quát, ta không biết gì về $f^{(n+1)}(\xi)$, điều tốt nhất có thể làm để thu được một độ lỗi nhỏ nhất có thể là chỉ sử dụng đa thức nội suy cho khoảng của t mà trong đó $\pi_n(t)$ có giá trị tuyệt đổi nhỏ nhất có thể, tức là với t gần $n/2$ hoặc một cách tương đương x gần điểm trung tâm của $[x_0, x_n]$.

Giả định rằng trong khoảng mà nội suy tốt là $x_0 < x < x_1$, và có ngẫu nhiên các điểm $x_j, j = 0, \pm 1, \pm 2, \ldots$ về khoảng này. Thì một đa thức nội suy Newton trực giao (ordinary Newton interpolation polynominal) dựa trên các điểm $x_0, x_1, x_{-1}, \ldots$ có những đặc trưng mong muốn tương ứng với khoảng $[x_0, x_1]$. Đa thức này có dạng

$$Q_n(x) = f_0 + (x - x_0)f_{0,1} + (x - x_0)(x - x_1)f_{0,1,-1} + \ldots \tag{43}$$

Tuy nhiên, bởi vì các sai phân chi được là các hàm đối xứng của các tham số của chúng nên ta có:

$$\begin{aligned} f_{0,1,-1} &= f_{-1,0,1} \\ f_{0,1,-1,2} &= f_{-1,0,1,2} \\ &\vdots \\ f_{0,1,-1,\ldots,m,-m} &= f_{-m,\ldots,-1,0,1,\ldots,m} \end{aligned} \tag{44}$$

Bằng cách sử dụng

$$\Delta f(x_0) = f(x_1) - f(x_0) = (x_1 - x_0)\frac{f(x_1) - f(x_0)}{x_1 - x_0} = hf[x_0, x_1] \tag{45}$$

ta có:

$$\begin{aligned} f_{0,1,-1} &= \frac{1}{2!h^2}\Delta^2 f_{-1} \\ f_{0,1,-1,2} &= \frac{1}{3!h^3}\Delta^3 f_{-1} \\ f_{0,1,-1,\ldots,m} &= \frac{1}{(2m)!h^{2m}}\Delta^{2m} f_{-m} \\ f_{0,1,-1,\ldots,m,m+1} &= \frac{1}{(2m+1)!h^{2m+1}}\Delta^{2m+1} f_{-m} \end{aligned} \tag{46}$$

Đa thức nội suy Newton trực giao có thể được viết lại, với $n = 2m$

$$\begin{aligned} Q_n(x_0 + th) = f_0 + t\Delta f_0 + \frac{t(t-1)}{2!}\Delta^2 f_{-1} \\ + \frac{t(t-1)(t+1}{3!}\Delta^3 f_{-1} + \ldots \\ + \frac{t(t-1)(t+1)\cdots(t-m)}{(2m)!}\Delta^{2m} f_{-m} \end{aligned} \tag{47}$$

và với $n = 2m + 1$

$$Q_n(x_0 + th) = f_0 + t\Delta f_0 + \cdots + \frac{t(t-1)(t+1)\cdots(t-m)(t+m)}{(2m+1)!}\Delta^{2m+1} f_{-m} \tag{48}$$

Đây là dạng Gaussian tiến cho đa thức nội suy.

Một dạng đối xứng của đa thức nội suy có thể thu được khi $n = 2m + 1$. Để có được dạng đối xứng cho trường hợp các sai phân chẵn, ta thêm vào ký hiệu sai phân trung tâm

$$\begin{aligned} \delta^2 f_r &\equiv \Delta f_r - \Delta f_{r-1} = \Delta^2 f_{r-1} \\ \delta^{2k} f_r &= \delta^2(\delta^{2(k-1)} f)_r = \delta^{2k} f_{r-k} \end{aligned} \tag{49}$$

Ta thấy điểm x_r luôn luôn là điểm giữa mà các số gia của nó là trung tâm. Với ký hiệu này, các bậc sai phân lẻ cao hơn trước, có thể viết lại như hiệu của hai

sai phân trung tâm

$$\Delta^{2m+1} f_{-m} = \delta^{2m} f_1 - \delta^{2m} f_0, m = 1, 2, \ldots \tag{50}$$

Sử dụng phương trình trên trong biến đổi dạng Gaussian, ta có:

$$\begin{aligned}
Q_n(x_0 + th) &= f_0 \\
&\quad + t\Delta f_0 + \frac{t(t-1)}{2!}\Delta^2 f_{-1} + \frac{t(t-1)(t+1)}{3!}\Delta^3 f_{-1} + \ldots \\
&\quad + \frac{t(t-1)(t+1)\cdots(t-m)}{(2m)!}\Delta^{2m} f_{-m} \\
&\quad + \frac{t(t-1)(t+1)\cdots(t-m)(t+m)}{(2m+1)!}\Delta^{2m+1} f_{-m} \\
&= f_0 + t(f_1 - f_0) + \frac{t(t-1)}{2!}\delta^2 f_0 \\
&\quad + \frac{t(t-1)(t+1)}{3!}(\delta^2 f_1 - \delta^2 f_0) + \ldots \\
&\quad + \frac{t(t-1)(t+1)\cdots(t-m)}{(2m)!}\delta^{2m} f_0 \\
&\quad + \frac{t(t-1)(t+1)\cdots(t-m)(t+m)}{(2m+1)!}(\delta^{2m} f_1 - \delta^{2m} f_0) \\
&= tf_1 + \frac{t(t-1)(t+1)}{3!}\delta^2 f_1 + \ldots \\
&\quad + \frac{t(t-1)(t+1)\cdots(t-m)(t+m)}{(2m+1)!}\delta^{2m} f_1 \\
&\quad + (1-t)f_0 + \frac{t(t-1)}{3!}(2-t)\delta^2 f_0 + \ldots \\
&\quad + \frac{t(t-1)(t+1)\cdots(t-m)}{(2m+1)!}(m+1-t)\delta^{2m} f_0
\end{aligned} \tag{51}$$

Đặt $s \equiv 1 - t$, ta có thể đơn giản hệ số của $\delta^{2k} f_0$ và ta được:

$$\begin{aligned}
Q_n(x_0 + th) = sf_0 + \frac{s(s^2-1)}{3!}\delta^2 f_2 + \cdots + \frac{s(s^2-1)\cdots(s^2-m^2)}{(2m+1)!}\delta^{2m} f_0 \\
+ tf_1 + \frac{t(t^2-1)}{3!}\delta^2 f_1 + \cdots + \frac{t(t^2-1)\ldots(t^2-m^2)}{(2m+1)!}\delta^{2m} f_1
\end{aligned} \tag{52}$$

PHỤ LỤC 4: PHƯƠNG PHÁP LẬP TRÌNH SAI PHÂN HỮU HẠN CHO ĐẠO HÀM SỐ

Trong phụ lục, vấn đề phương pháp lập trình MATLAB cho bài toán xấp xỉ đạo hàm bằng đạo hàm của đa thức nội suy được tập trung trình bày. Đạo hàm số không phải một quá trình chính xác hoàn toàn do lỗi làm tròn (rounding errors) và lỗi cắt bỏ (truncation errors) trong quá trình tính toán. Và vì lý do này mà giá trị đạo hàm tính toán thông qua đạo hàm số không bao giờ chính xác như giá trị đạo hàm được đánh giá tại một điểm cụ thể.

.3. Giới thiệu chung

Phương pháp được sử dụng để tính gần đúng sẽ là xấp xỉ sai phân hữu hạn, dựa trên Chuỗi Taylor xung quanh một điểm cụ thể x. Cụ thể, ta cố gắng tính gần đúng các giá trị của $f(x+(n)h)$ và $f(x-(n)h)$ với một số n lớn hơn 0. Việc giải hệ phương trình của các chuỗi này dẫn đến xấp xỉ các đạo hàm. Điều này dẫn đến hậu quả là có hai loại lỗi không thể tránh khỏi được đề cập ở trên bao gồm:

- Độ lỗi làm tròn (do tính toán số học của máy tính)
- Độ lỗi cắt bỏ

.4. Cài đặt sai phân hữu hạn

Dựa trên định nghĩa, một xấp xỉ Taylor cho một hàm số $f(x)$ xung quanh một điểm a

$$f(x) \approx f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n \qquad (53)$$

Và nếu ta xấp xỉ một hàm của $x+h$ xung quanh một điểm x

$$f(x+h) \approx f(x) + \frac{f'(x)}{1!}(h) + \frac{f''(x)}{2!}(h)^2 + \cdots = \sum_{n=0}^{\infty} \frac{f^{(n)}(x)}{n!}(h)^n \tag{54}$$

Ta có thể tìm được đạo hàm cấp một bằng cách lấy hiệu của khai triển hàm $f(x+h)$ và $f(x-)$

$$f(x+h) - f(x-h) \approx 2\left[hf'(x) + \frac{h^3}{3!}f'''(x) + \ldots\right] \tag{55}$$

Và dễ dàng thấy được

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} - \frac{1}{h}\left(\frac{h^3}{3!}f'''(x) + \ldots\right) \approx \frac{f(x+h) - f(x-h)}{2h} - \frac{h^2}{6}f'''(x) - \ldots \tag{56}$$

Để xác định độ chính xác của phép xấp xỉ, ta cố định x và thay đổi h, ta có thể giả sử rằng lỗi cắt bỏ như sau:

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} + O(h^2) \tag{57}$$

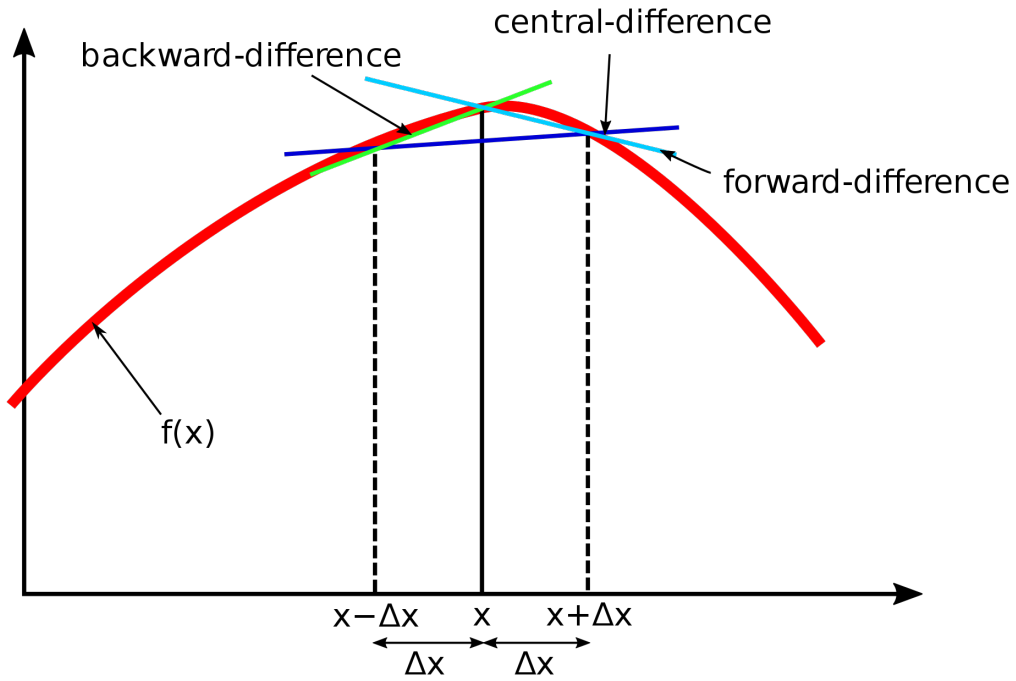
vì hầu như h nhỏ hơn 1, nên số mũ của h càng nhỏ thì phép xấp xỉ càng chính xác.

Các phép tính gần đúng có thể được thực hiện theo cách "trung tâm", dựa trên việc tính các điểm $f(x+h)$ và $f(x-h)$ với x nằm ở tâm của hai điểm này. Hình (1) minh họa ý tưởng này.

.4.1. Sai phân hữu hạn lùi

Phương pháp sai phân hữu hạn lùi (finite backward difference) sử dụng giá trị hạn tại x và $x-h$. Hàm dưới đây cài đặt phương pháp bằng MATLAB.

```
1 function bfd = backwardFiniteDifference(f, x, h, m, n)
2     % Backward Finite Difference Calculation
```



Hình 1: *Minh họa sai phân tiến, sai phần lùi, và sai phân trung tâm*

```
% Inputs:
%   f: function handle
%   x: evaluation point
%   h: step size
%   m: derivative order (1 for first derivative, 2
    for second)
%   n: accuracy order (1 for O(h), 2 for O(h^2))
% Output:
%   bfd: backward finite difference approximation
%   of the derivative

% Validate inputs
if m > 2 || n > 2
    error('m and n must be <= 2');
end
```

```
17
18      coefficients = [
19          NaN, NaN,  1, -1;% First derivative, O(h)
20          NaN, 3/2, -2, 1/2;% Second derivative, O(h^2)
21      ];
22      coefficients(:, :, 2) = [
23          NaN,  1, -2,  1;% First derivative, O(h^2)
24           2, -5,  4, -1;% Second derivative, O(h^2)
25      ];
26
27      % Extract coefficients for the
28      % specified derivative order and accuracy
29      c = coefficients(n, :, m);
30      c = c(~isnan(c));
31
32      % Calculate the backward finite difference
33      bfd = 0;
34      for i = 1:length(c)
35          bfd = bfd + c(i) * f(x - (i-1) * h);
36      end
37      bfd = bfd / (h^m);
38  end
```

.4.2. Sai phân hữu hạn tiến

Phương pháp sai phân hữu hạn tiến (finite forward difference) sử dụng giá trị hạn tại x và $x + h$. Hàm dưới đây cài đặt phương pháp bằng MATLAB.

```
1  function ffd = forwardFiniteDifference(f, x, h, m, n)
2      % Forward Finite Difference Calculation
```

```
3     % Similar validation and structure as
          backwardFiniteDifference function
4     % Inputs:
5     %   f: function handle
6     %   x: evaluation point
7     %   h: step size
8     %   m: derivative order (1 for first derivative, 2
          for second)
9     %   n: accuracy order (1 for O(h), 2 for O(h^2))
10    % Output:
11    %   ffd: forward finite difference approximation
12    %   of the derivative
13
14    if m > 2 || n > 2
15        error('m and n must be <= 2');
16    end
17
18    coefficients = [
19        -1,  1, NaN, NaN;% First derivative, O(h)
20        -3/2, 2, -1/2, NaN;% Second derivative, O(h^2)
21    ];
22    coefficients(:, :, 2) = [
23         1, -2,  1, NaN;% First derivative, O(h^2)
24         2, -5,  4, -1;% Second derivative, O(h^2)
25    ];
26
27    c = coefficients(n, :, m);
28    c = c(~isnan(c));
29
```

```
    ffd = 0;
    for i = 1:length(c)
        ffd = ffd + c(i) * f(x + (i-1) * h);
    end
    ffd = ffd / (h^m);
end
```

.4.3. Đạo hàm cấp một trung tâm

Và dựa trên sai phân hữu hạn tiến (finite forward difference) và sai phân hữu hạn lùi (finite backward difference), ta có sai phân trung tâm cho đạo hàm cấp một như sau:

```
function fcd = firstCenteredDerivative(f, x, h)
    % Calculate the first centered derivative
    fcd = (f(x+h) - f(x-h)) / (2*h);
end
```

.4.4. Đạo hàm cấp hai trung tâm

Và dựa trên sai phân hữu hạn tiến (finite forward difference) và sai phân hữu hạn lùi (finite backward difference), ta có sai phân trung tâm cho đạo hàm cấp hai như sau:

```
function scd = secondCenteredDerivative(f, x, h)
    % Calculate the second centered derivative
    scd = (f(x+h) - 2*f(x) + f(x-h)) / (h^2);
end
```

.5. Thiết lập thực nghiệm với hàm số cụ thể

Ta sử dụng hàm $sin(x)$ để tiến hành thực nghiệm.

```
% Define symbolic variable and function
syms x;
f(x) = sin(x^2);

% Convert symbolic function to MATLAB function
% for numerical evaluation
func1 = matlabFunction(f);

% Initialize parameters for step size variation
pow = 0.6; % h expansion factor
lim = 50;  % h highest exponent
i = 1:lim;
% Generate hs = [h^1, h^2, ..., h^lim]
hs = arrayfun(@(x) double(pow^x), i);

% Point of evaluation
x_eval = 2;
```

.6. Sai phân hữu hạn trung tâm

Ta tính toán giá trị đạo hàm đúng cho đạo hàm cấp một và cấp hai.

```
% Calculate the actual values of the first and
% second derivatives at x
% First derivative
df_actual   = double(subs(diff(f, 1), x, x_eval));
% Second derivative
d2f_actual  = double(subs(diff(f, 2), x, x_eval));

```

```
% Initialize arrays for errors and approximations
errors = zeros(10, length(hs));
approximations = zeros(1, 10);

% Table to store the minimum error for each method
minErrors = zeros(10, 2);
```

Lặp qua mỗi giá trị của h, ta tính toán độ lỗi cho các phương pháp sai phân hữu hạn.

```
% Loop over each h value to calculate errors for
   different differentiation methods
for j = 1:length(hs)
    h = hs(j); % Current step size

    % Calculate approximations for first and second
       derivatives
    %% Centered first derivative - O(h^2)
    approximations(1) = firstCenteredDerivative(func1,
        x_eval, h);

    %% Centered second derivative - O(h^2)
    approximations(2) = secondCenteredDerivative(func1
       , x_eval, h);

    %% Forward Finite first derivative - O(h)
    approximations(3) = forwardFiniteDifference(func1,
        x_eval, h, 1, 1);

    %% Forward Finite first derivative - O(h^2)
```

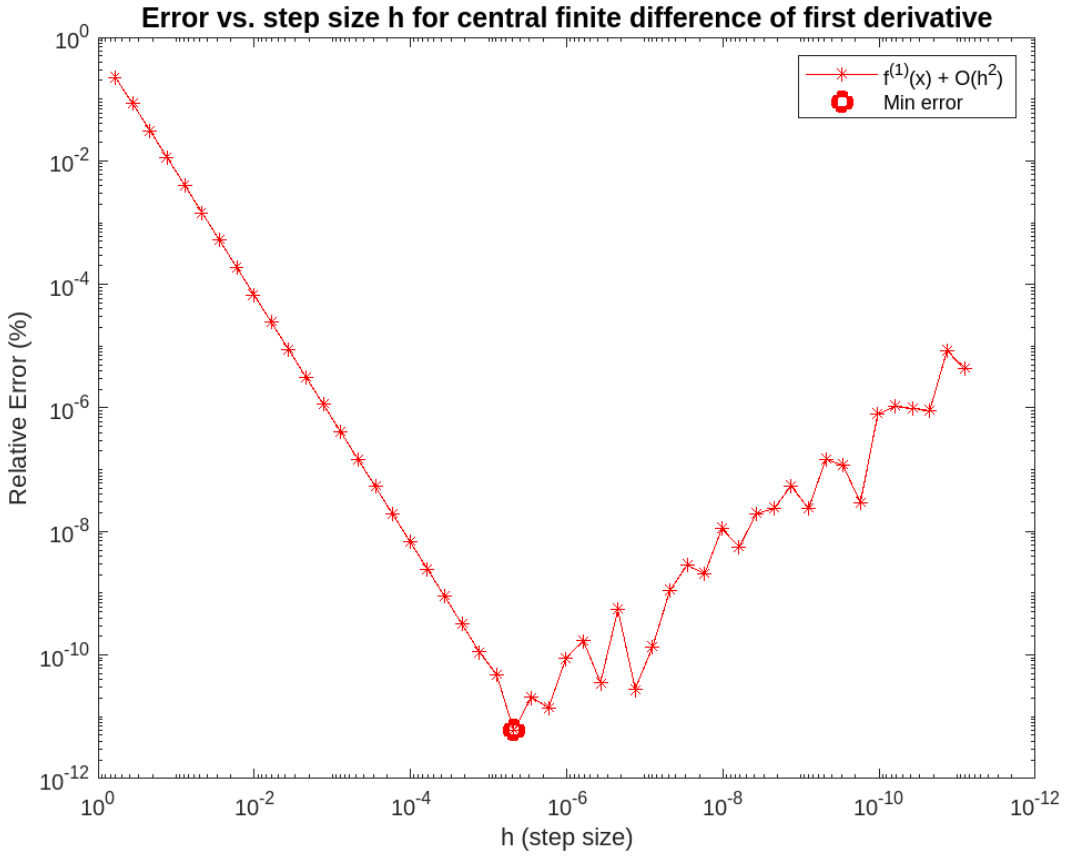
```
16      approximations(4) = forwardFiniteDifference(func1,
            x_eval, h, 1, 2);
17
18      %% Forward Finite second derivative - O(h)
19      approximations(5) = forwardFiniteDifference(func1,
            x_eval, h, 2, 1);
20
21      %% Forward Finite second derivative - O(h^2)
22      approximations(6) = forwardFiniteDifference(func1,
            x_eval, h, 2, 2);
23
24      %% Backward Finite first derivative - O(h)
25      approximations(7) = backwardFiniteDifference(func1
           , x_eval, h, 1, 1);
26
27      %% Backward Finite first derivative - O(h^2)
28      approximations(8) = backwardFiniteDifference(func1
           , x_eval, h, 1, 2);
29
30      %% Backward Finite second derivative - O(h)
31      approximations(9) = backwardFiniteDifference(func1
           , x_eval, h, 2, 1);
32
33      %% Backward Finite second derivative - O(h^2)
34      approximations(10) = backwardFiniteDifference(
           func1, x_eval, h, 2, 2);
35
36      % Calculate and store relative errors
37      % for each approximation
```

```
    for i = 1:10
        if mod(i, 2) == 1 % Odd indices for first derivative
            errors(i, j) = abs(approximations(i) / df_actual - 1);
        else % Even indices for second derivative
            errors(i, j) = abs(approximations(i) / d2f_actual - 1);
        end
    end
end
```

Trực quan hóa kết quả cho sai phân trung tâm của đạo hàm cấp một bằng:

```
% Plotting the error for the first centered derivative approximation
figure;
loglog(hs, errors(1,:), 'r-*');
hold on;
[minError, idxMinError] = min(errors(1,:));
loglog(hs(idxMinError), errors(1,idxMinError), "ro", 'MarkerSize', 7, 'LineWidth', 3);
set(gca, 'XDir', 'reverse');
xlabel("h (step size)");
ylabel("Relative Error (%)");
title("Error vs. step size h for central finite difference of first derivative", 'FontSize', 12);
legend("f^{(1)}(x) + O(h^2)", "Min error");
minErrors(1, :) = [hs(idxMinError), errors(1, idxMinError)];
```

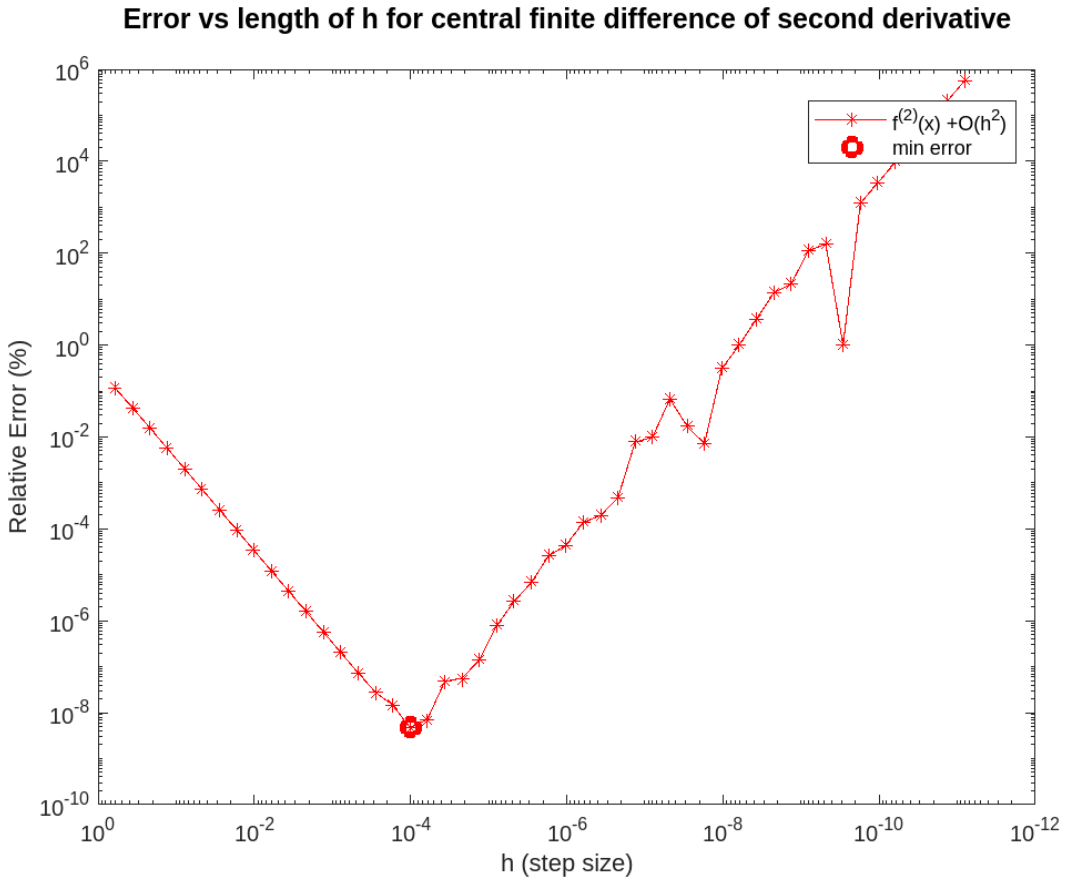
```
hold off;
```



Hình 2: *Sai phân trung tâm cho đạo hàm cấp một*

```
% Plotting the error for the second centered
   derivative approximation
figure;
loglog(hs,errors(2,:),'r-*')
hold on;
[~,idx] = min(errors(2,:));
loglog(hs(idx),errors(2,idx),"r o",'MarkerSize',7,'
   LineWidth',3)
set(gca,'XDir','reverse')
xlabel("h (step size)")
ylabel("Relative Error (%)")
```

```
10 [t,~] = title("Error vs length of h for central finite
       difference of second derivative",' ');
11 t.FontSize = 12;
12 legend("f^{(2)}(x) +O(h^2)","min error")
13 tabErrors(2,1)= hs(idx);
14 tabErrors(2,2)= errors(2,idx);
15 hold off
```

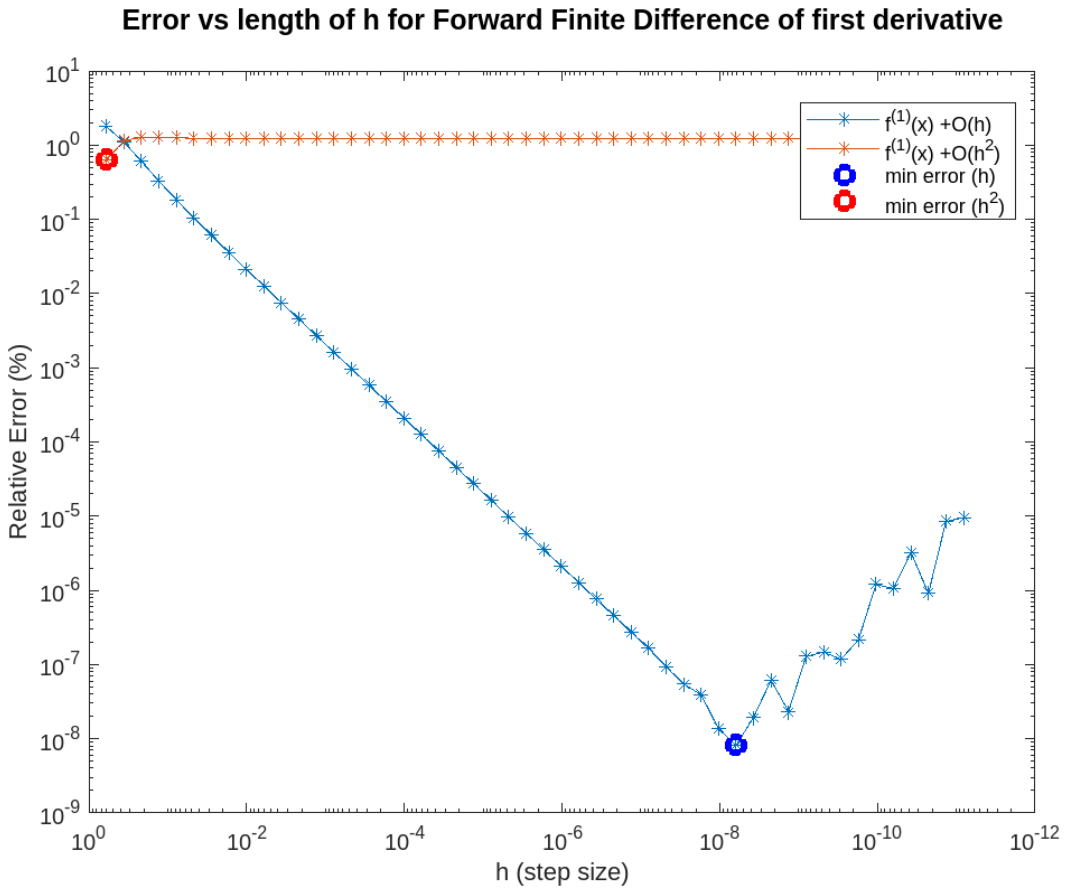


Hình 3: *Sai phân trung tâm cho đạo hàm cấp hai*

.7. Phân tích sai phân tiến hữu hạn

```
1 figure;
2 loglog(hs,errors(3:4,:),"-*")
3 hold on
```

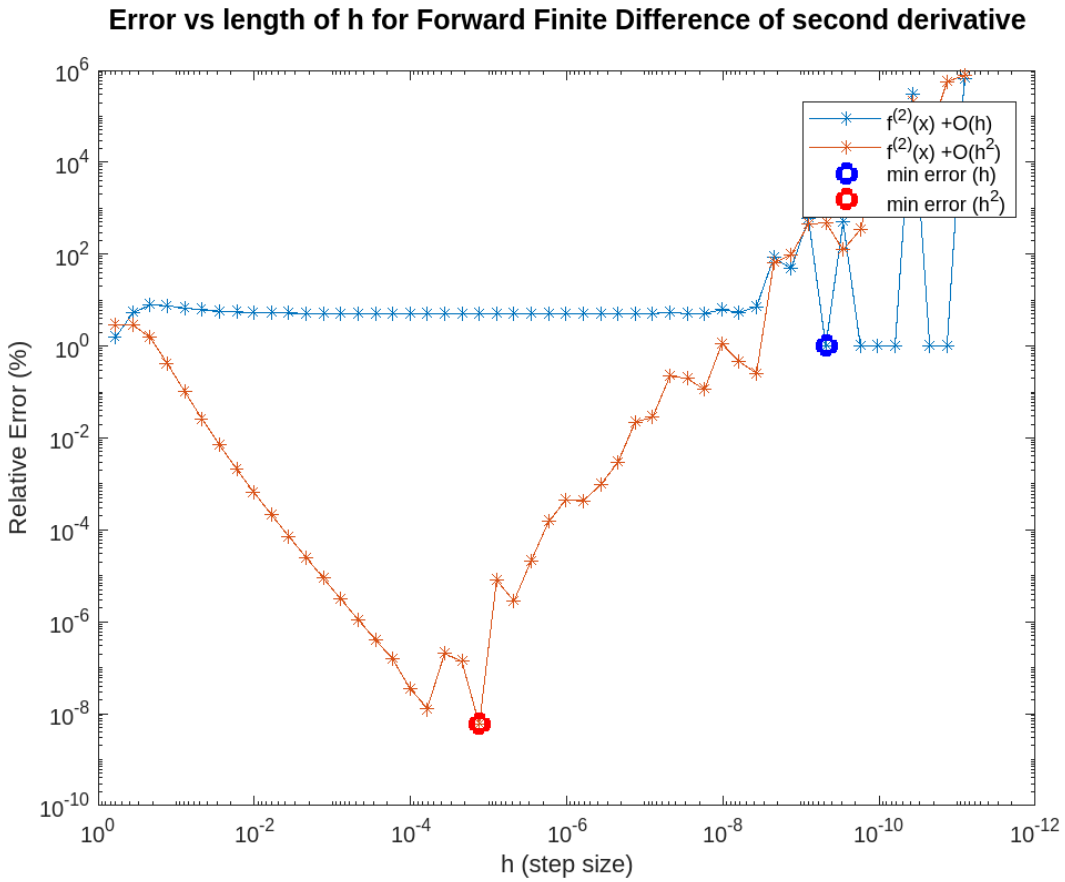
```
[~,idx] = min(errors(3,:));
loglog(hs(idx),errors(3,idx),"b o",'MarkerSize',7,'
   LineWidth',3)
tabErrors(3,1)= hs(idx);
tabErrors(3,2)= errors(3,idx);


[~,idx] = min(errors(4,:));
loglog(hs(idx),errors(4,idx),"r o",'MarkerSize',7,'
   LineWidth',3)
tabErrors(4,1)= hs(idx);
tabErrors(4,2)= errors(4,idx);
set(gca,'XDir','reverse')
xlabel("h (step size)")
ylabel("Relative Error (%)")
[t,~] = title("Error vs length of h for Forward Finite
    Difference of first derivative",' ');
t.FontSize = 12;
legend("f^{(1)}(x) +O(h)","f^{(1)}(x) +O(h^2)","min
   error (h)","min error (h^2)")
hold off
```



Hình 4: *Sai phân tiến cho đạo hàm cấp một*

```
figure;
loglog(hs,errors(5:6,:),"-*")
hold on
[~,idx] = min(errors(5,:));
loglog(hs(idx),errors(5,idx),"b o",'MarkerSize',7,'
   LineWidth',3)
tabErrors(5,1)= hs(idx);
tabErrors(5,2)= errors(5,idx);

[~,idx] = min(errors(6,:));
loglog(hs(idx),errors(6,idx),"r o",'MarkerSize',7,'
   LineWidth',3)
```

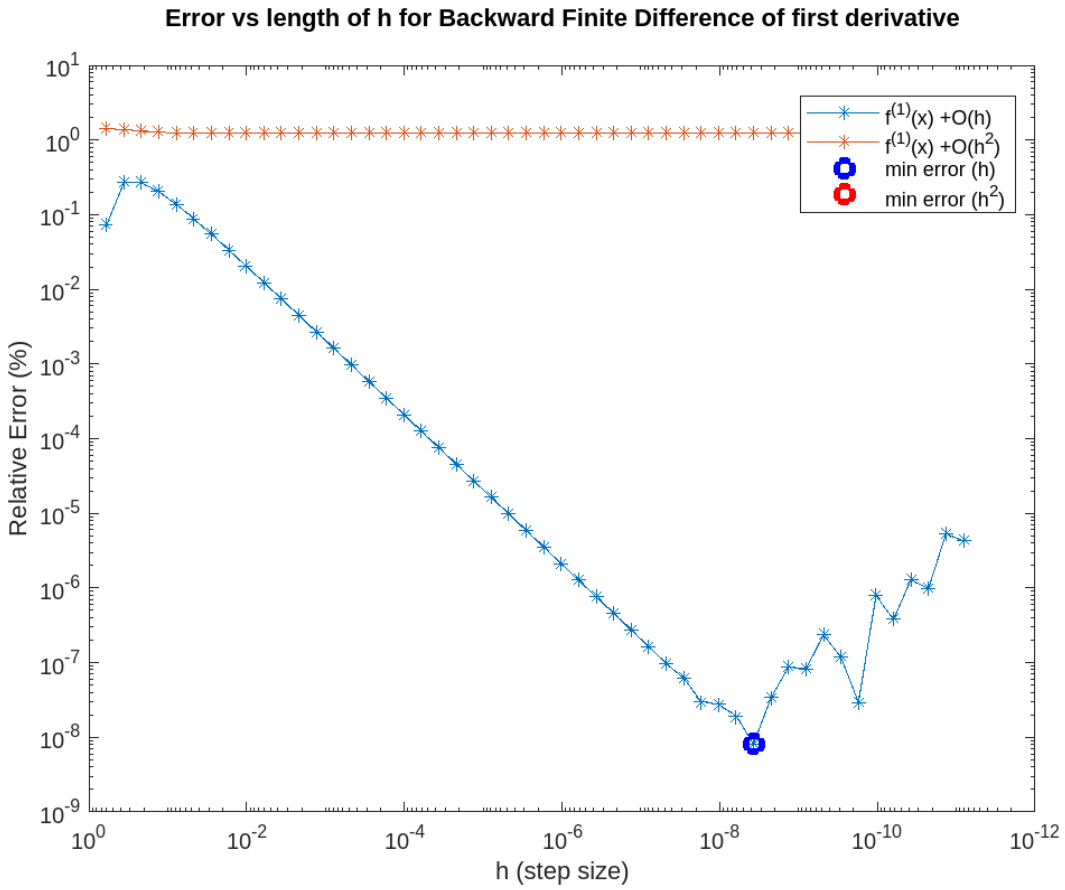
```
set(gca,'XDir','reverse')
xlabel("h (step size)")
ylabel("Relative Error (%)")
[t,s] = title("Error vs length of h for Forward Finite
    Difference of second derivative", ' ');
t.FontSize = 12;
legend("f^{(2)}(x) +O(h)","f^{(2)}(x) +O(h^2)","min
   error (h)","min error (h^2)")
tabErrors(6,1)= hs(idx);
tabErrors(6,2)= errors(6,idx);
hold off
```



Hình 5: *Sai phân tiến cho đạo hàm cấp hai*

.8. Phân tích sai phân lùi hữu hạn

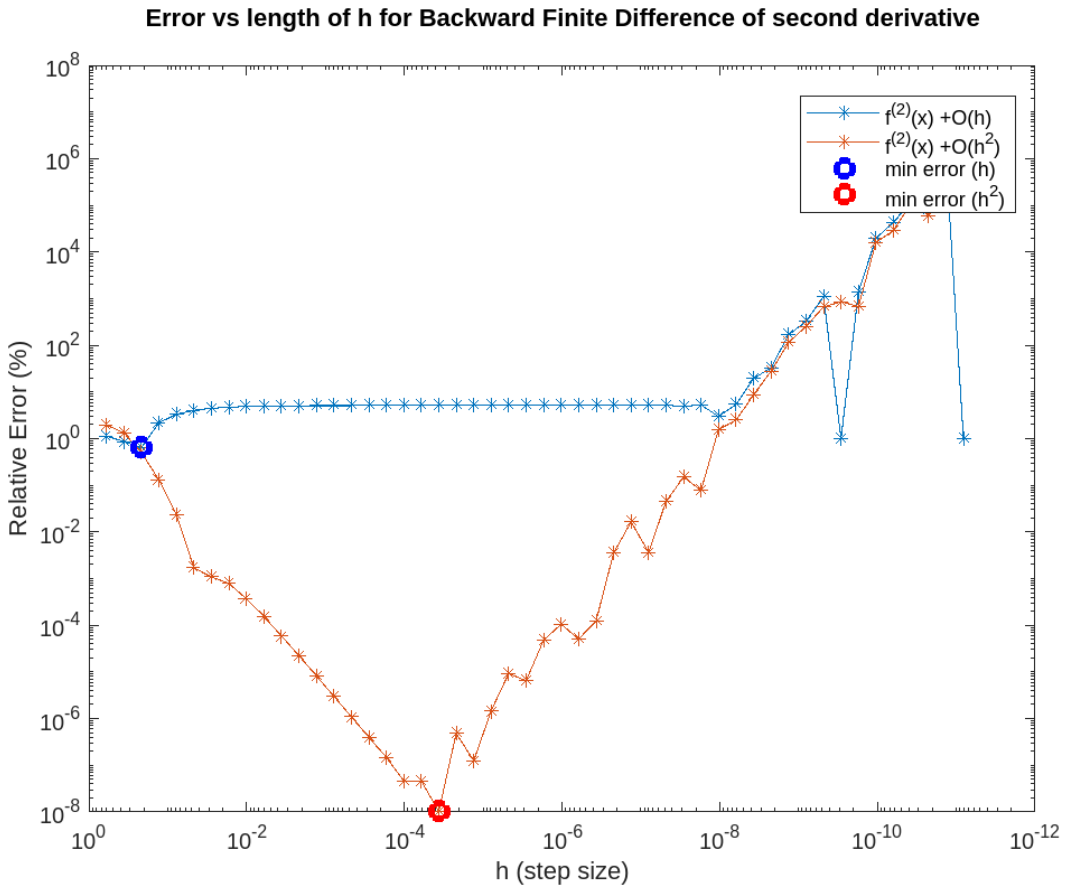
```
figure;
loglog(hs,errors(7:8,:),"-*")
hold on
[~,idx] = min(errors(7,:));
loglog(hs(idx),errors(7,idx),"b o",'MarkerSize',7,'
   LineWidth',3)
tabErrors(7,1)= hs(idx);
tabErrors(7,2)= errors(7,idx);

[~,idx] = min(errors(8,:));
loglog(hs(idx),errors(8,idx),"r o",'MarkerSize',7,'
   LineWidth',3)
set(gca,'XDir','reverse')
xlabel("h (step size)")
ylabel("Relative Error (%)")
title("Error vs length of h for Backward Finite
   Difference of first derivative",' ')
legend("f^{(1)}(x) +O(h)","f^{(1)}(x) +O(h^2)","min
   error (h)","min error (h^2)")
tabErrors(8,1)= hs(idx);
tabErrors(8,2)= errors(8,idx);
hold off
```



Hình 6: *Sai phân lùi cho đạo hàm cấp một*

```
figure;
loglog(hs,errors(9:10,:),"-*")
hold on
[~,idx] = min(errors(9,:));
loglog(hs(idx),errors(9,idx),"b o",'MarkerSize',7,'
   LineWidth',3)
tabErrors(9,1)= hs(idx);
tabErrors(9,2)= errors(9,idx);

[~,idx] = min(errors(10,:));
loglog(hs(idx),errors(10,idx),"r o",'MarkerSize',7,'
   LineWidth',3)
```

```
set(gca,'XDir','reverse')
xlabel("h (step size)")
ylabel("Relative Error (%)")
title("Error vs length of h for Backward Finite
   Difference of second derivative",' ')
legend("f^{(2)}(x) +O(h)","f^{(2)}(x) +O(h^2)","min
   error (h)","min error (h^2)")
hold off
```



Hình 7: *Sai phân lùi cho đạo hàm cấp hai*

Tên phương pháp	Độ lỗi nhỏ nhất	Độ lỗi tương đối nhỏ nhất
Sai phân hữu hạn cấp 1 $O(h^2)$	1.7058e-06	2.6423e-11
Sai phân hữu hạn cấp 2 $O(h^2)$	6.0936e-05	3.4290e-09
Sai phân hữu hạn tiến cấp 1 $O(h)$	6.1887e-09	8.0898e-09
Sai phân hữu hạn tiến cấp 1 $O(h^2)$	2.8430e-06	1.3405e-11
Sai phân hữu hạn tiến cấp 2 $O(h)$	2.8430e-06	7.7608e-06
Sai phân hữu hạn tiến cấp 2 $O(h^2)$	1.3162e-05	5.7523e-09
Sai phân hữu hạn lùi cấp 1 $O(h)$	3.7132e-09	8.0898e-09
Sai phân hữu hạn lùi cấp 1 $O(h^2)$	6.1409e-07	1.2357e-12
Sai phân hữu hạn lùi cấp 2 $O(h)$	2.8430e-06	1.5129e-05
Sai phân hữu hạn lùi cấp 2 $O(h^2)$	3.6562e-05	1.0195e-08

Bảng 1: *Bản đánh giá độ lỗi ỏ nhất cho các phương pháp sai phân hữu hạn.*

.9. Phân tích độ lỗi

.9.1. Đánh giá độ lỗi nhất

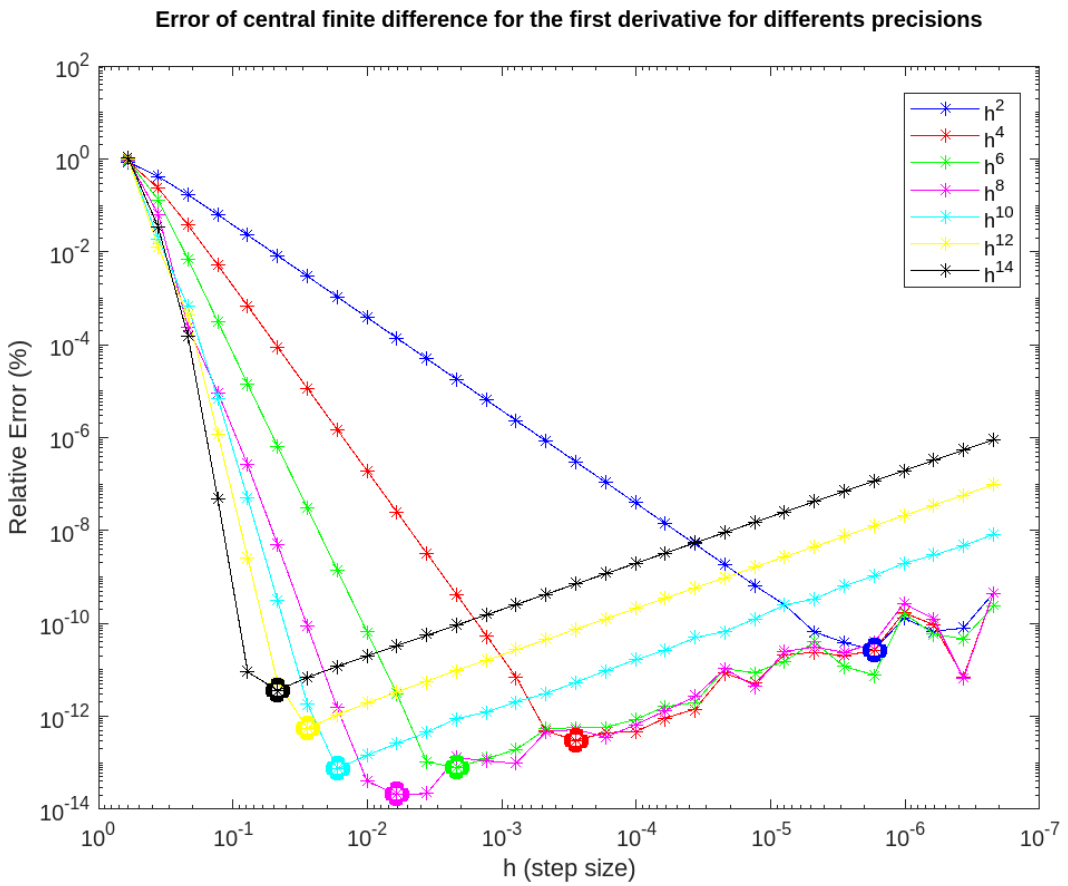
.9.2. Phân tích độ lỗi cho xấp xỉ đạo hàm cấp một

```
pre = 2:2:14; % Precision levels: 2, 4, 6, 8, 10, 12, 14
m = 1; % Order of derivative
hs = arr(1:30); % Assuming 'arr' is correctly defined elsewhere to generate hs

% Initialize error matrix
error = zeros(length(pre), length(hs));

for idx = 1:length(pre)
    n = pre(idx);
    n_coefs = 2*floor((m+1)/2)-1+n; p = (n_coefs-1)/2;
    A = power(-p:p, (0:2*p)');
```

```
12      b = zeros(2*p+1,1);
13      b(m+1) = factorial(m);
14      c = A\b; % Coefficients for the finite difference
            approximation
15
16      for j = 1:length(hs)
17          h = hs(j);
18          k = -p;
19          ffd_val = 0;
20          for cof = c'
21              ffd_val = ffd_val + cof * func1(x_eval + k
                    * h);
22              k = k + 1;
23          end
24          ffd_val = ffd_val / (h^m);
25
26          % Calculate and store the relative error for
                the current precision and step size
27          error(idx, j) = abs((ffd_val - df_actual) /
                df_actual);
28      end
29  end
30
31  figure
32  l(1) = loglog(hs,error(1,:),"b-*");
33  hold on
34  l(2) = loglog(hs,error(2,:),"r-*");
35  l(3) = loglog(hs,error(3,:),"g-*");
36  l(4) = loglog(hs,error(4,:),"m-*");
```

```
37 l(5) = loglog(hs,error(5,:),"c-*");
38 l(6) = loglog(hs,error(6,:),"y-*");
39 l(7) = loglog(hs,error(7,:),"k-*");
40 set(gca,'XDir','reverse');
41 [~,idx] = min(error(1,:));
42 loglog(hs(idx),error(1,idx),"b o",'MarkerSize',8,'
     LineWidth',3)
43 [~,idx] = min(error(2,:));
44 loglog(hs(idx),error(2,idx),"r o",'MarkerSize',8,'
     LineWidth',3)
45 [~,idx] = min(error(3,:));
46 loglog(hs(idx),error(3,idx),"g o",'MarkerSize',8,'
     LineWidth',3)
47 [~,idx] = min(error(4,:));
48 loglog(hs(idx),error(4,idx),"m o",'MarkerSize',8,'
     LineWidth',3)
49 [~,idx] = min(error(5,:));
50 loglog(hs(idx),error(5,idx),"c o",'MarkerSize',8,'
     LineWidth',3)
51 [~,idx] = min(error(6,:));
52 loglog(hs(idx),error(6,idx),"y o",'MarkerSize',8,'
     LineWidth',3)
53 [minV,idx] = min(error(7,:));
54 loglog(hs(idx),error(7,idx),"k o",'MarkerSize',8,'
     LineWidth',3)
55 xlabel("h (step size)")
56 ylabel("Relative Error (%)")
57 title("\fontsize{10}Error of central finite difference
     for the first derivative for differents precisions
```

```
    ',' ')
58  legend(l(1: 7),'h^2','h^4','h^6','h^8','h^{10}','h
    ^{12}','h^{14}');
59  hold off
```



Hình 8: *Phân tích độ lỗi của sai phân hữu hạn trung tâm cho đạo hàm cấp một*

Hình (8) minh họa độ lỗi của sai phân hữu hạn trung tâm cho đạo hàm cấp một với các bậc chính xác khác nhau. Bất chấp những phân tích đã đề cập về $O(h)$ so với $O(h^2)$. Khi cố gắng tăng bậc chính xác, chúng ta không thể tăng độ chính xác lên quá nhiều. Dựa trên hình vẽ trực quan, ta dễ dàng thấy được xuất hiện lỗi làm tròn, vì vậy ta sẽ cần đánh giá ít giá trị của h hơn và không giảm nó quá nhiều để đạt được xấp xỉ tốt nhất. Tuy nhiên, do độ chính xác hạn chế của độ chính xác kép của số học IEEE 754 (và các số trong máy tính nói

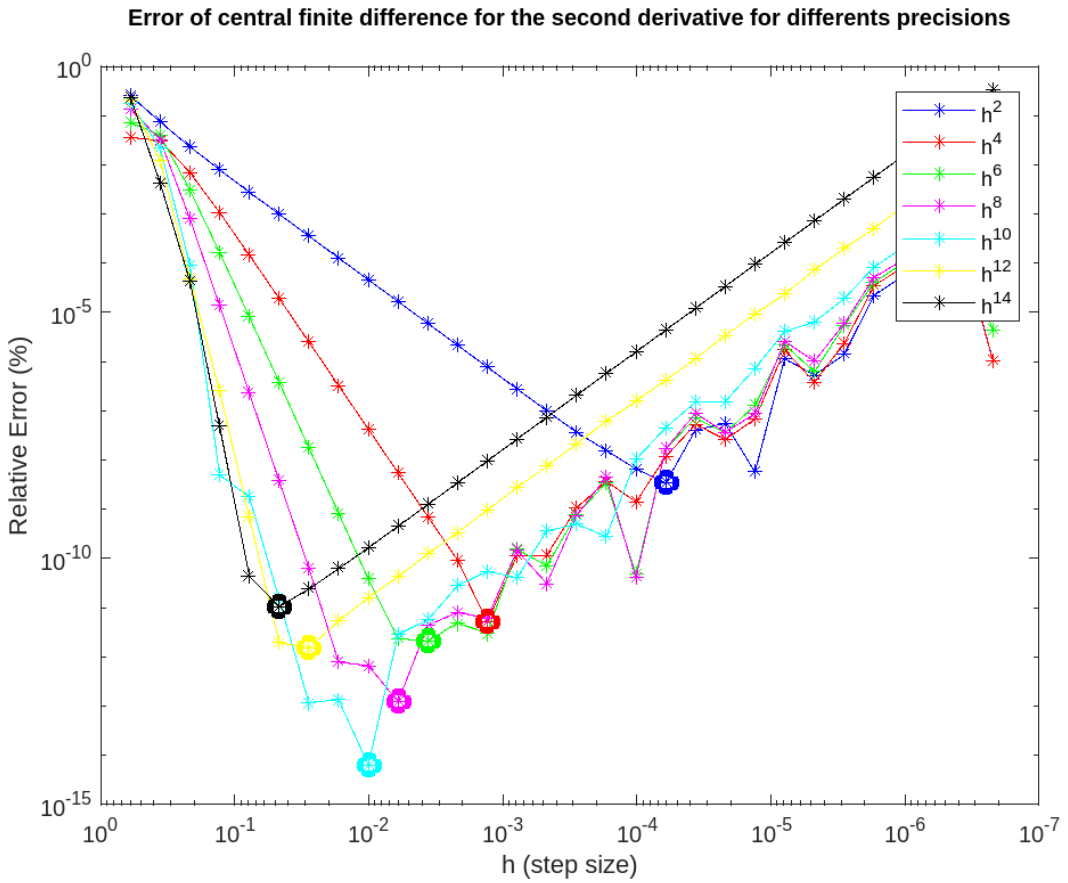
chung), khi mức độ chính xác tăng lên, cuối cùng chúng ta sẽ đạt đến lỗi làm tròn nhanh đến mức không có cách nào giảm h để giảm phần độ lỗi cắt bỏ của xấp xỉ Taylor.

.9.3. Phân tích độ lỗi cho xấp xỉ đạo hàm cấp hai

```
pre = 2:2:14; % Precision levels: 2, 4, 6, 8, 10, 12, 14
m = 1; % Order of derivative
hs = arr(1:30); % Assuming 'arr' is correctly defined elsewhere to generate hs

% Initialize error matrix
error = zeros(length(pre), length(hs));

for idx = 1:length(pre)
    n = pre(idx);
    n_coefs = 2*floor((m+1)/2)-1+n; p = (n_coefs-1)/2;
    A = power(-p:p, (0:2*p)');
    b = zeros(2*p+1,1);
    b(m+1) = factorial(m);
    c = A\b; % Coefficients for the finite difference approximation

    for j = 1:length(hs)
        h = hs(j);
        k = -p;
        ffd_val = 0;
        for cof = c'
```

```
21              ffd_val = ffd_val + cof * func1(x_eval + k
                    * h);
22              k = k + 1;
23          end
24          ffd_val = ffd_val / (h^m);
25
26          % Calculate and store the relative error for
               the current precision and step size
27          error(idx, j) = abs((ffd_val - d2f_actual) /
               d2f_actual);
28      end
29  end
30
31  figure
32  l(1) = loglog(hs,error(1,:),"b-*");
33  hold on
34  l(2) = loglog(hs,error(2,:),"r-*");
35  l(3) = loglog(hs,error(3,:),"g-*");
36  l(4) = loglog(hs,error(4,:),"m-*");
37  l(5) = loglog(hs,error(5,:),"c-*");
38  l(6) = loglog(hs,error(6,:),"y-*");
39  l(7) = loglog(hs,error(7,:),"k-*");
40  set(gca,'XDir','reverse');
41  [~,idx] = min(error(1,:));
42  loglog(hs(idx),error(1,idx),"b o",'MarkerSize',8,'
       LineWidth',3)
43  [~,idx] = min(error(2,:));
44  loglog(hs(idx),error(2,idx),"r o",'MarkerSize',8,'
       LineWidth',3)
```

```
45 [~,idx] = min(error(3,:));
46 loglog(hs(idx),error(3,idx),"g o",'MarkerSize',8,'
     LineWidth',3)
47 [~,idx] = min(error(4,:));
48 loglog(hs(idx),error(4,idx),"m o",'MarkerSize',8,'
     LineWidth',3)
49 [~,idx] = min(error(5,:));
50 loglog(hs(idx),error(5,idx),"c o",'MarkerSize',8,'
     LineWidth',3)
51 [~,idx] = min(error(6,:));
52 loglog(hs(idx),error(6,idx),"y o",'MarkerSize',8,'
     LineWidth',3)
53 [minV,idx] = min(error(7,:));
54 loglog(hs(idx),error(7,idx),"k o",'MarkerSize',8,'
     LineWidth',3)
55 xlabel("h (step size)")
56 ylabel("Relative Error (%)")
57 title("\fontsize{10}Error of central finite difference
     for the second derivative for differents precisions
     ",' ')
58 legend(l(1: 7),'h^2','h^4','h^6','h^8','h^{10}','h
     ^{12}','h^{14}');
59 hold off
```



Hình 9: *Phân tích độ lỗi của sai phân hữu hạn trung tâm cho đạo hàm cấp hai*

Hình (9) minh họa tương tự với độ lỗi sai phân hữu hạn trung tâm cho đạo hàm cấp hai với các bậc chính xác khác nhau. Nhận định tương tự với trường hợp đạo hàm cấp một cũng được rút ra.

.10. Kết luận

Phương pháp số sai phân hữu hạn có thể cho ta những xấp xỉ rất tốt về đạo hàm của một hàm, tuy nhiên chúng ta phải tính đến hai nguồn sai số mà ta đã đề cập ở phần đầu.

.11. Nhận xét về độ lỗi cắt bỏ

Độ lỗi cắt bỏ là một loại độ lỗi tự nhiên đối với bất kỳ phương pháp nào xuất phát từ Chuỗi Taylor không hội tụ hoặc bị cắt cụt (như trường hợp sai phân hữu hạn). Điều duy nhất ta có thể làm để giảm lỗi cắt ngắn này là chọn một h rất nhỏ. Nhưng việc chọn một bước rất nhỏ xung quanh x để xấp xỉ đạo hàm sẽ dẫn đến nguồn sai số thứ hai.

.12. Nhận xét về độ lỗi làm tròn

Độ chính xác của các số trong máy tính luôn là hữu hạn và không giống như các ngôn ngữ khác, bộ nhớ dành cho số học trong MATLAB bị hạn chế.

Độ chính xác kép được cung cấp bởi các con số trong MATLAB cho phép ta sử dụng h khá nhỏ trước khi phát sinh lỗi làm tròn. Nhưng phải lưu ý rằng, ta càng có nhiều số hạng sử dụng h và càng thực hiện nhiều thao tác với nó thì sai số làm tròn sẽ càng lớn.

.13. Các kết quả đạt được

Ta có thể kết luận rằng h tốt nhất để có được độ chính xác tốt nhất là h càng nhỏ càng tốt mà không gây ra lỗi làm tròn. Về mặt lý thuyết, h có thể được tính gần đúng bằng cách cộng phần dư Lagrange cùng với sai số làm tròn tương đương với h được sử dụng trong hàm để xấp xỉ. Tuy nhiên, ta cần đạo hàm sâu hơn đạo hàm mà chúng ta đang tính gần đúng để thu được h lý thuyết nói trên, vì vậy việc tính h này về mặt lý thuyết là không thực tế.

Kết quả quan trọng của báo cáo này trả lời câu hỏi sau: Nếu việc tăng độ chính xác của sai phân hữu hạn làm cho lỗi cắt cụt xuất hiện sớm hơn (khi chúng ta làm cho h nhỏ hơn), tại sao sai số lại nhỏ hơn khi chúng ta tăng các giá trị? các hệ số sử dụng h? Tức là khi chúng ta tăng độ chính xác của các công thức sai phân hữu hạn.