Nội suy trung tâm: Phương pháp nội suy Gauss I và Gauss II

Nguyễn Đức Anh

Viện Toán Ứng dụng và Tin học,

Đại học Bách Khoa Hà Nội

Ngày 23 tháng 1 năm 2022

Outline

1. Nội suy trung tâm
2. Phương pháp Gauss I
   - Công thức toán học
   - Thuật toán
3. Phương pháp Gauss II
   - Công thức toán học
   - Thuật toán
4. Thuật toán với bộ dữ liệu lớn
5. Nhận xét

Outline

Nội suy trung tâm

- Công thức nội suy Newton (tiến và lùi) mang tính chất "một phía".
- Để tính các giá trị tại giữa bảng, công thức một phía sẽ gặp hạn chế.
- Một số phương pháp nội suy trung tâm:
    - Phương pháp Gauss I và Gauss II.
    - Phương pháp Stirlin.
    - Phương pháp Bessel.

Nội suy trung tâm

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	$\dots$
$\dots$	$\dots$					
x_{-2}	y_{-2}					
		Δy_{-2}				
x_{-1}	y_{-1}		$\Delta^2 y_{-2}$			
		Δy_{-1}		$\Delta^3 y_{-2}$		
x_0	y_0		$\Delta^2 y_{-1}$		$\Delta^4 y_{-2}$	$\dots$
		Δy_0		$\Delta^3 y_{-1}$		
x_1	y_1		$\Delta^2 y_0$			
		Δy_1				
x_2	y_2					
$\dots$	$\dots$					

Bảng 1: Bảng sai phân trung tâm

Bài toán

Giả sử có $2n+1$ mốc nội suy cách đều và được xếp thứ tự:

$$x_{-n}, \dots, x_{-1}, x_0, x_1, \dots, x_n$$

và các giá trị tương ứng y_i cho trong bảng số:

$$y_i = f(x_i), \quad i = 0, \pm 1, \dots, \pm n. \tag{1}$$

Xây dựng đa thức nội suy $P(x)$ bậc $\leq 2n$ sao cho:

$$y_i = P(x_i), \quad i = 0, \pm 1, \dots, \pm n. \tag{2}$$

- Ý tưởng: Xuất phát từ mốc trung tâm, nạp lần lượt các mốc trái, phải.
- Điều kiện: Trường hợp mốc cách đều, đã được sắp thứ tự. Và số mốc thường là lẻ.

Outline

- Phương pháp Gauss I xuất phát từ mốc trung tâm và nạp các mốc theo thứ tự một mốc bên phải rồi đến một mốc bên trái:

$$x_0 \to x_1 \to x_{-1} \to x_2 \to x_{-2} \dots$$

- Đa thức nội suy có dạng:

$$\begin{aligned} P(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + a_3(x - x_{-1})(x - x_0)(x - x_1) + \dots \\ + a_{2n-1}(x - x_{-(n-1)}) \dots (x - x_0) \dots (x - x_{n-1}) \\ + a_{2n}(x - x_{-(n-1)}) \dots (x - x_0) \dots (x - x_n). \end{aligned}$$

- Đặt $h = x_i - x_{i-1}$ với $i = 1, \ldots, 2n$.
- Với $x = x_0$ thì $a_0 = y_0$.
- Với $x = x_1$ thì:

$$y_1 = P(x_1) = y_0 + a_1(x_1 - x_0)$$

Suy ra

$$a_1 = \frac{y_1 - y_0}{x_1 - x_0} = \frac{\Delta y_0}{h}.$$

- Với $x = x_{-1}$ thì:

$$\begin{aligned} y_{-1} = P(x_{-1}) &= y_0 + \frac{\Delta y_0}{h}(x_{-1} - x_0) + a_2(x_{-1} - x_0)(x_{-1} - x_1) \\ &= y_0 - \Delta y_0 + a_2 \times h \times 2h \end{aligned}$$

Hay

$$\Delta y_0 - \Delta y_{-1} = a_2 \times 2h^2$$

Suy ra:

$$a_2 = \frac{\Delta y_0 - \Delta y_{-1}}{2h^2} = \frac{\Delta^2 y_{-1}}{2h^2}.$$

- Tổng quát:

$$a_{2i-1} = \frac{\Delta^{2i-1} y_{-(i-1)}}{(2i-1)!h^{2i-1}}$$

$$a_{2i} = \frac{\Delta^{2i} y_{-i}}{(2i)!h^{2i}}$$

$\Rightarrow$ Gặp vấn đề khi h nhỏ.

- Đặt $t = \dfrac{x - x_0}{h}$
- Với $a_{2i-1} = \dfrac{\Delta^{2i-1} y_{-(i-1)}}{(2i-1)!h^{2i-1}}$ và $(x - x_{-(i-1)}) \dots (x - x_0) \dots (x - x_{i-1})$ trở thành:

$$\frac{\Delta^{2i-1} y_{-(i-1)}}{(2i-1)!} \times (t+i-1)\dots(t+1)t(t-1)\dots(t-i+1).$$

- Với $a_{2i} = \dfrac{\Delta^{2i} y_{-i}}{(2i)!h^{2i}}$ và $(x - x_{-(i-1)}) \dots (x - x_0) \dots (x - x_i)$ trở thành:

$$\frac{\Delta^{2i} y_{-i}}{(2i)!} \times (t+i-1)\dots(t+1)t(t-1)\dots(t-i+1)(t-i).$$

- Tổng quát: Đa thức được tìm dưới dạng:

$$P(x) = P(x_0 + ht) = y_0 + \Delta y_0 t + \frac{\Delta^2 y_{-1}}{2!} t(t-1) + \frac{\Delta^3 y_{-1}}{3!}(t+1)t(t-1) + \ldots$$

trong đó:

$$a_{2i-1} = \frac{\Delta^{2i-1} y_{-(i-1)}}{(2i-1)!}, \quad a_{2i} = \frac{\Delta^{2i} y_{-i}}{(2i)!}.$$

Công thức Newton tiến:

$$P(t) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 + \frac{t(t-1)(t-2)}{3!}\Delta^3 y_0 + \frac{t(t-1)(t-2)(t-3)}{4!}\Delta^4 y_0 + \dots$$

Mặt khác, lại có:

$$\begin{aligned}
\Delta^2 y_0 &= \Delta^2 y_{-1} + \Delta^3 y_{-1} \\
\Delta^3 y_0 &= \Delta^3 y_{-1} + \Delta^4 y_{-1} \\
\Delta^4 y_0 &= \Delta^4 y_{-1} + \Delta^5 y_{-1}
\end{aligned}$$

Suy ra:

$$\begin{aligned}
P(t) = y_0 + t\Delta y_0 &+ \frac{t(t-1)}{2!}(\Delta^2 y_{-1} + \Delta^3 y_{-1}) + \frac{t(t-1)(t-2)}{3!}(\Delta^3 y_{-1} + \Delta^4 y_{-1}) \\
&+ \frac{t(t-1)(t-2)(t-3)}{4!}(\Delta^4 y_{-1} + \Delta^5 y_{-1}) + \dots
\end{aligned}$$

Tức là:

$$P(t) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}\Delta^3 y_{-1} + \frac{(t+1)t(t-1)(t-2)}{4!}\Delta^4 y_{-1} + \frac{(t+1)t(t-1)(t-2)(t-3)}{5!}\Delta^5 y_{-1} \dots$$

Tiếp tục áp dụng:

$$\Delta^4 y_{-1} = \Delta^4 y_{-2} + \Delta^5 y_{-2}$$
$$\Delta^5 y_{-1} = \Delta^5 y_{-2} + \Delta^6 y_{-2}$$
$$\dots$$

Cứ như thế, ta sẽ được công thức Gauss 1.

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	$\dots$
$\dots$	$\dots$					
x_{-2}	y_{-2}					
		Δy_{-2}				
x_{-1}	y_{-1}		$\Delta^2 y_{-2}$			
		Δy_{-1}		$\Delta^3 y_{-2}$		
x_0	y_0		$\Delta^2 y_{-1}$		$\Delta^4 y_{-2}$	$\dots$
		Δy_0		$\Delta^3 y_{-1}$		
x_1	y_1		$\Delta^2 y_0$			
		Δy_1				
x_2	y_2					
$\dots$	$\dots$					

Hình 1: Bảng sai phân với Gauss 1

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	$\Delta^5 y$
x_{-3}	y_{-3}					
		Δy_{-3}				
x_{-2}	y_{-2}		$\Delta^2 y_{-3}$			
		Δy_{-2}		$\Delta^3 y_{-3}$		
x_{-1}	y_{-1}		$\Delta^2 y_{-2}$		$\Delta^4 y_{-3}$	
		Δy_{-1}		$\Delta^3 y_{-2}$		$\Delta^5 y_{-3}$
x_0	y_0		$\Delta^2 y_{-1}$		$\Delta^4 y_{-2}$	
		Δy_0		$\Delta^3 y_{-1}$		
x_1	y_1		$\Delta^2 y_0$			
		Δy_1				
x_2	y_2					

Hình 2: Gauss 1 với số mốc chẵn

Thuật toán

Phương pháp Gauss I

- Hàm **saiphan**(a: vector):

```
ans ← empty list
n ← length(a)
for i = 1 → n − 1 do
    Append (a[i + 1] − a[i]) to list ans
end for
return ans
```

- Hàm **bangsaiphan**(x, y: vector):

```
n ← length(x)
table ← list[y]
for i = 1 → n − 1 do
    Append saiphan(table[i]) to table
end for
return table
```

Thuật toán

Phương pháp Gauss I

- Hàm **tichGauss1**(t: float, m: int):

  $ans \leftarrow 1$
  $c \leftarrow int\left(\frac{m}{2}\right)$
  if m%2 $== 1$ **then**
      for $i = -c \to c$ **do**
          $ans \leftarrow ans * (t + i)$
      end for
  else
      for $i = -c \to c - 1$ **do**
          $ans \leftarrow ans * (t + i)$
      end for
  end if
  return ans

Thuật toán I

Phương pháp Gauss I

- Hàm chính **Gauss1**(x, y: vector, $value$: float):

  $n \leftarrow length(x)$
  $h \leftarrow x[1] - x[0]$
  $table \leftarrow$ **bangsaiphan**(x, y)
  $fact \leftarrow 1$
  if n%2==1 **then**
      $t \leftarrow (value - x[int(n/2)])/h$
      $ans \leftarrow table[0][int(n/2)]$
      for $i = 1 \rightarrow n$ **do**
          $fact \leftarrow fact * i$
          $ans \leftarrow ans+$ **tichGauss1**(t, i)*$table[i][int((n-i)/2)]/fact$
      end for

Thuật toán II

Phương pháp Gauss I

else
    $t \leftarrow (value - x[int((n-1)/2)])/h$
    $ans \leftarrow table[0][int((n-1)/2)]$
    for $i = 1 \to n$ **do**
        $fact \leftarrow fact * i$
        $ans \leftarrow ans +$ **tichGauss1**(t, i)*$table[i][int((n-i-1)/2)]/fact$
    end for
end if
return ans

Để đưa ra đa thức nội suy, ta dùng phương pháp Horner.

- Hàm **horner**($poly$: vector, x: float):

```
n ← len(poly)
ans ← poly[n − 1]
for i = n − 2 → 0 do
    ans ← ans ∗ x + poly[i]
end for
```

- Hàm **mulConst**(a: vector, c: float):

```
ans ← empty list
for i = 0 → len(a) do
    Append c ∗ a[i] to ans
end for
return ans
```

- Hàm **sumHorner**(a: vector, b: vector):

```
if len(a) > len(b) then
    for i = 0 → len(b) do
        a[i] ← a[i] + b[i]
    end for
    return a
else
    for i = 0 → len(a) do
        b[i] ← a[i] + b[i]
    end for
    return b
end if
return ans
```

- Hàm **mulHorner**(a: vector, m: float, $case$: int):

```
c ← int(m/2)
ans ← empty list
if (m%2 == 1 and case == 1) or (m%2 == 0 and case == 2) then:
    for i = 0 → len(a) do
        if j == 0 then
            Append a[j] * c to ans
        else
            Append a[j − 1] + a[j] * c to ans
        end if
    end for
```

else
    for $i = 0 \to len(a)$ **do**
        if $j == 0$ **then**
            Append $a[j] * (-c)$ to *ans*
        else
            Append $a[j-1] + a[j] * (-c)$ to *ans*
        end if
    end for
end if
Append 1 to *ans*
return *ans*

- Hàm chính **Gauss1Horner**(x, y: vector, $value$: float):

```
n ← len(x), h ← x[1] − x[0]
table ← bangsaiphan(x, y)
fact ← 1
if n%2==1 then
    poly ← [0]
    ans ← [table[0][int(n/2)]]
    for i = 1 → n do
        fact ← fact ∗ i
        poly ← mulHorner(poly, i, 1)
        ans ← sumHorner(ans, mulConst(poly, table[i][int((n − i)/2)]/fact))
    end for
    result ← horner(ans, (value − x[int(n/2)])/h)
```

Thuật toán II

Phương pháp Gauss I

else
    $poly \leftarrow [0]$
    $ans \leftarrow [table[0][int((n-1)/2)]]$
    for $i = 1 \rightarrow n$ **do**
        $fact \leftarrow fact * i$
        $poly \leftarrow$ **mulHorner**$(poly, i, 1)$
        $ans \leftarrow$ **sumHorner**$(ans,$ **mulConst**$(poly, table[i][int((n-i-1)/2)]/fact))$
    end for
    $result \leftarrow$ **horner**$(ans, (value - x[int((n-1)/2)])/h)$
end if
return $ans, result$

- Tính giá trị tại $x = 345$ với bộ dữ liệu:

x	310	320	330	340	350	360	370
y	2.4914	2.5052	2.5185	2.5315	2.5441	2.5563	2.5687

- Tính giá trị tại $x = 34$ với bộ dữ liệu:

x	20	25	30	35	40	45
y	354	332	291	260	231	204

- Đảo ngược bảng giá trị trong ví dụ 1.

Outline

- Phương pháp Gauss II xuất phát từ mốc trung tâm và nạp các mốc theo thứ tự một mốc bên trái rồi đến một mốc bên phải:

$$x_0 \to x_{-1} \to x_1 \to x_{-2} \to x_2 \dots$$

- Đa thức nội suy có dạng:

$$\begin{aligned} P(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_{-1}) + a_3(x - x_{-1})(x - x_0)(x - x_1) + \\ a_4(x - x_{-2})(x - x_{-1})(x - x_0)(x - x_1) + \\ a_5(x - x_{-2})(x - x_{-1})(x - x_0)(x - x_1)(x - x_2) + \dots \end{aligned}$$

- Tương tự công thức Gauss 1, đặt $t = \dfrac{x - x_0}{h}$, ta sẽ có công thức Gauss 2 như sau:

$$P(x) = P(x_0 + ht) = y_0 + \Delta y_{-1} t + \frac{\Delta^2 y_{-1}}{2!} t(t+1) + \frac{\Delta^3 y_{-1}}{3!}(t+1)t(t-1) + \dots$$

trong đó:

$$a_{2i-1} = \frac{\Delta^{2i-1} y_{-i}}{(2i-1)!}, \quad a_{2i} = \frac{\Delta^{2i} y_{-i}}{(2i)!}.$$

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	$\dots$
$\dots$	$\dots$					
x_{-2}	y_{-2}					
		Δy_{-2}				
x_{-1}	y_{-1}		$\Delta^2 y_{-2}$			
		Δy_{-1}		$\Delta^3 y_{-2}$		
x_0	y_0		$\Delta^2 y_{-1}$		$\Delta^4 y_{-2}$	$\dots$
		Δy_0		$\Delta^3 y_{-1}$		
x_1	y_1		$\Delta^2 y_0$			
		Δy_1				
x_2	y_2					
$\dots$	$\dots$					

Hình 3: Bảng sai phân với Gauss 2

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	$\Delta^5 y$
x_{-2}	y_{-2}					
		Δy_{-2}				
x_{-1}	y_{-1}		$\Delta^2 y_{-2}$			
		Δy_{-1}		$\Delta^3 y_{-2}$		
x_0	y_0		$\Delta^2 y_{-1}$		$\Delta^4 y_{-2}$	
		Δy_0		$\Delta^3 y_{-1}$		$\Delta^5 y_{-2}$
x_1	y_1		$\Delta^2 y_0$		$\Delta^4 y_{-1}$	
		Δy_1		$\Delta^3 y_0$		
x_2	y_2		$\Delta^2 y_1$			
		Δy_2				
x_3	y_3					

Hình 4: Gauss 2 với số mốc chẵn

Thuật toán

Phương pháp Gauss II

- Hàm **tichGauss2**(t: float, m: int):

  $ans \leftarrow 1$
  $c \leftarrow int\left(\frac{m}{2}\right)$
  if m%2 == 1 **then**
      for $i = -c \rightarrow c$ **do**
          $ans \leftarrow ans * (t + i)$
      end for
  else
      for $i = -c + 1 \rightarrow c$ **do**
          $ans \leftarrow ans * (t + i)$
      end for
  end if
  return ans

- Hàm **Gauss2**(x, y: vector, $value$: float):

```
n ← length(x)
h ← x[1] − x[0]
table ← bangsaiphan(x, y)
fact ← 1
if n%2==1 then
    t ← (value − x[int(n/2)])/h
    ans ← table[0][int(n/2)]
    for i = 1 → n do
        fact ← fact ∗ i
        ans ← ans+ tichGauss2(t, i)*table[i][int((n − i − 1)/2)]/fact
    end for
```

Thuật toán II

Phương pháp Gauss II

else
    $t \leftarrow (value - x[int(n/2)])/h$
    $ans \leftarrow table[0][int(n/2)]$
    for $i = 1 \rightarrow n$ **do**
        $fact \leftarrow fact * i$
        $ans \leftarrow ans +$ **tichGauss2**(t, i)*$table[i][int((n-i)/2)]/fact$
    end for
end if
return ans

- Hàm chính **Gauss2Horner**(x, y: vector, $value$: float):

```
n ← len(x), h ← x[1] − x[0]
table ← bangsaiphan(x, y)
fact ← 1
if n%2==1 then
    poly ← [0]
    ans ← [table[0][int(n/2)]]
    for i = 1 → n do
        fact ← fact ∗ i
        poly ← mulHorner(poly, i, 2)
        ans ← sumHorner(ans, mulConst(poly, table[i][int((n − i − 1)/2)]/fact))
    end for
    result ← horner(ans, (value − x[int(n/2)])/h)
```

Thuật toán II

Phương pháp Gauss II

```
else
    poly ← [0]
    ans ← [table[0][int(n/2)]]
    for i = 1 → n do
        fact ← fact * i
        poly ← mulHorner(poly, i, 2)
        ans ← sumHorner(ans, mulConst(poly, table[i][int((n − i)/2)]/fact))
    end for
    result ← horner(ans, (value − x[int(n/2)])/h)
end if
return ans, result
```

Outline

Thuật toán với bộ dữ liệu lớn

- Giống như công thức Newton, khi có nhiều mốc nội suy, ta cần chọn những mốc nội suy phù hợp.
- Trích xuất dữ liệu phù hợp với yêu cầu:

$$x_0, x_1, x_2, \ldots, x_N \longrightarrow x_i, x_{i+1}, \ldots, x_{i+n}.$$

Thuật toán với bộ dữ liệu lớn

- Hàm **chiadoi**(x: vector, $value$: float):

```
h ← abs(x[1] − x[0])
low ← 0
high ← len(x)
while low ≤ high do:
    mid ← int((low + high)/2)
    if abs(x[mid] − value) < h then
        return mid
    else
        if x[mid] > value then
            high = mid
        else
            low = mid
        end if
    end if
```

Thuật toán với bộ dữ liệu lớn I

- Hàm **GaussBig**(x: vector, y: vector, $value$: float, num: int)

  $n \leftarrow len(x)$
  if $num \geq n$ **then**:
      return "No enough nodes"
  end if
  $index \leftarrow int((value - x[0])/(x[1] - x[0]))$
  Kiểm tra có đủ mốc với hai mốc trung tâm $x[index]$ hoặc $x[index + 1]$.
  Chọn mốc $x[index]$ hoặc $x[index + 1]$ làm mốc trung tâm x_{index} và lưu $index$ phù hợp.

Thuật toán với bộ dữ liệu lớn II

if $num\%2 == 1$ **then**

    $x_{new} = x[index - int(num/2) : index + int(num/2) + 1]$

    $y_{new} = y[index - int(num/2) : index + int(num/2) + 1]$

else

    $x_{new} = x[index - int(num/2) + 1 : index + int(num/2) + 1]$

    $y_{new} = y[index - int(num/2) + 1 : index + int(num/2) + 1]$

end if

$ans = \mathbf{gauss1}(x_{new}, y_{new}, value)$

return ans

- Tính giá trị hàm số $f(x) = x^7 - 4x^3 + x - 2$, biết giá trị hàm số tại $x = 1, 2, \dots, 40$ tại
  - $x = 2.5$ với 7 mốc nội suy
  - $x = 2.5$ với 6 mốc nội suy
  - $x = 21.4$ với 7 mốc nội suy
  - $x = 21.4$ với 8 mốc nội suy
  - $x = 21.4$ với 9 mốc nội suy
- Cho bảng dữ liệu nhiệt độ Trái đất từ năm 1880 đến năm 2020 với bước nhảy 5 năm. Ước lượng nhiệt độ của Trái đất tại một thời điểm nào đó trong khoảng thời gian từ 1880 đến 2020.
- Đảo ngược data dữ liệu nhiệt độ Trái đất.

Outline

Nhận xét

- Về sai số, công thức Gauss được phát triển từ công thức Newton, do đó sai số của công thức Gauss là:

$$R_n(x) \approx \frac{\Delta^{n+1} y_0}{(n+1)!} \prod_{k=0}^{n} (t-k)$$

- Thuật toán đã xây dựng có độ phức tạp tối ưu cỡ $O(5n^2/2)$.
- Về mặt ứng dụng, công thức Gauss không được sử dụng nhiều trong các thực tiễn. Tuy nhiên, đây là công thức quan trọng trong việc phát triển ra các công thức nội suy như Sterling hay Bessel.

Tài liệu tham khảo

- Lê Trọng Vinh. "Giáo trình Giải tích số". In: *NXB Khoa học và Kỹ thuật* (2007)
- Graham de Vahl Davis. "Numerical Methods in Engineering and Science". In: *Springer, Dordrecht* (1986), pp. 274–338

Xin chân thành cảm ơn!