**Các công thức nội suy trung tâm**

**Đào Thanh Tùng**

**Bài toán chung**:Hàm số y= f(x) xác định trên đoạn [a;b] nào đó, ta chưa biết biểu thức cụ thể của f(x). Nhưng bằng một cách nào đó ta thu được một bảng số có gồm các điểm thuộc đoạn [a;b] tương ứng với giá trị  của hàm số tại điểm đó, vậy làm thế nào để từ các điểm trên ta có thể tính được giá trị tại một điểm bất kỳ  . Hoặc cũng có thể ta đã xác định được việc biên đổi quy luật của hàm y=f(x) nhưng quy luật quá phức tạp nên ta cũng không thể tính giá trị tại  một cách dễ dàng được. Do vậy, ta tìm cách xây dựng hàm F(x) đơn giản hơn f(x) mà khi tính giá trị của F(x) và f(x) thì sự sai lệch là không lớn. Hàm F(x) tìm được thỏa mãn điều trên thường sẽ là đa thức và gọi là đa thức nội suy.

Ta đã được làm quen với đa thức nội suy tiến, lùi. Đa thức nội suy tiến hoặc lùi đều có tính chất một phía. Vậy tại sao ta không đưa ra đa thức nào đó có thể sử dụng cả tiến và lùi để hiệu quả đạt được cao hơn.

Đa thức mà có thể khai thác được cả 2 vấn đề tiến và lùi gọi là đa thức nội suy trung tâm (công thức nội suy trung tâm) .

**I – Những công thức nội suy trung tâm được xây dựng**

**1 – Công thức nội suy Gauss I**

**2 – Công thức nội suy Gauss II**

**3 – Công thức nội suy Sterling**

**4 – Công thức nội suy Bessel**

**II – Những vấn đề khởi đầu**

\*\* Các công thức nội suy trung tâm nêu trên sẽ chỉ dùng cho việc các mốc nội suy là *cách đều* với bước là h>0.

\*\* Công thức nội suy trung tâm được xây dựng dựa trên công thức nội suy Newton có mốc cách đều.

\*\* Về bản chất công thức nội suy trung tâm cũng được xây dựng dựa trên khai triển Taylor – Maclaurin.

\*\* Từ n+1 mốc ban đầu thì đa thức nội suy F(x) tìm được là duy nhất!!!

**1 – Sai phân**

Cho hàm số f(x) xác định trên đoạn [a;b]

 là số gia của đối số ,  = h : bước nhảy



Biểu thức  gọi là *sai phân* cấp một của hàm y=f(x) tại x.

*Sai phân của sai phân cấp một là sai phân cấp hai* ,

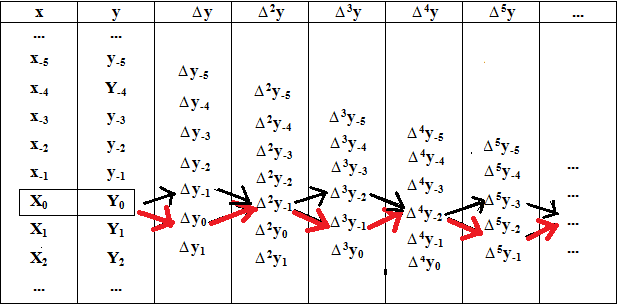
*Sai phân của sai phân cấp m-1 gọi là sai phân cấp m* ,

**2 - Nhắc lại một số tính chất của sai phân**

* Delta là toán tử tuyến tính
* Giá trị của hàm f(x) được biểu diễn qua sai phân các cấp của nó
* Sai phân cấp m của hàm f(x) được biểu diễn qua các giá trị liên tiếp của nó
* Nếu f(x) có đạo hàm liên tục đến cấp m trên đoạn [x,x+mh] thì ta có:

 với 

**3 – Bảng sai phân**



**III – Công thức nội suy Gauss**

Bài toán:

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



Và các giá trị tương ứng của hàm y=f(x)

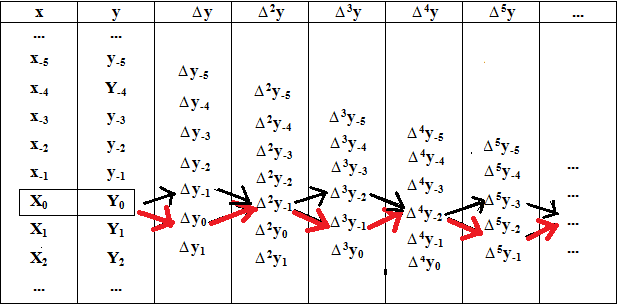
Các giá trị trên tương ứng với việc ta có bảng sau: **Bảng (1)**

|  |  |
| --- | --- |
| X | Y |
| X-n | Y-n |
| X-n+1 | Y-n+1 |
| ... | ... |
| X-1 | Y-1 |
| X0 | Y0 |
| X1 | Y1 |
| ... | ... |
| Xn | Yn |

 **Yêu cầu**: từ bảng trên hãy xây dựng đa thức nội suy P(x) bậc  2n sao cho

  (2)

* *Bảng sai phân*: **Bảng (2)**



1. **Công thức nội suy Gauss thứ I**

Đa thức nội suy được tìm trong dạng



 (3)



 Để tìm ta sử dụng điều kiện (2)

Thế dần  ta được:

Cho  thì 

Cho  thì ta có 



Cho  ta có :







Vậy ta có được **công thức tổng quát** sau:

  (4)

Xuất phát từ mốc  ; Đặt 

Thay (4) vào (3) ta được:





 (5)



Công thức (5) gọi là công thức nọi suy **Gauss thứ nhất**. Nhìn vào bảng sai phân trung tâm ta thấy công thức (5) có hệ số được tính theo đường gấp khúc in đậm màu đỏ và được gọi là công thức ‘’ *2 tiến một lùi* ‘’

**Ví dụ \*\*\*:** cho các mốc nội suy và giá trị của hàm số y=f(x) tương ứng sau:

(x;y)={(1,3);(2,7);(3,13);(4,17);(5,23)}

Tìm đa thức nội suy P(x) của f(x) ?

Ta có bảng sai phân sau:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | y |  |  |  |  |
| 1 | 3 |  |  |  |  |
| 4 |  |  |  |
| 2 | 7 |  | 2 |  |  |
| 6 |  | -4 |  |
| 3 | 13 |  | -2 |  | 8 |
| 4 |  | 4 |  |
| 4 | 17 |  | 2 |  |  |
| 6 |  |  |  |
| 5 | 23 |  |  |  |  |
|  |  |  |  |

Từ bảng sai phân trung tâm trên ta có:











Thay các giá trị vừa tính được vào đa thức P(x) ta được:

P(x)==> 

* Ta đã tìm được đa thức thỏa mãn

\*\* Công thức sai số:

Ta sử dụng sai phân cấp 2n và sai số tương tự như đa thức nội suy Newton có mốc cách đều nên ta có công thức:



**\*\* Thuật toán:**

* Input: + Vector X,Y tương ứng với các mốc và giá trị

+ Sai số mong muốn sẽ đạt được : eps

+ điểm cần tính giá trị x1

* Output: + hàm P sau khi đã rút gọn.

+ sai số đã tính được đối với điểm nhập vào x1

+ Giá trị của hàm tại điểm x1

* Thực hiện:

+ B1: Tìm mốc nội suy gần giá trị cần tính nhất x[a]

+ B2: Lập và hoàn thiện bảng sai phân với 2 vector đã cho X,Y

+ B3: Coi mốc x[a] là gốc: tính hệ số b[i] của đa thức F[x] tới hết mốc ngoài cùng gần nhất.

+ B4: thực hiện vòng lặp: (tới khi hết hệ số)

++ lần lượt gán hệ số vào đa thức

++ so sánh sai số: nếu sai số đạt được thì dừng vòng lặp.

+ In ra màn hình kết quả.

1. **Công thức nội suy Gauss thứ II**

Xuất phát từ mốc  , từ bảng sai phân trung tâm ta có thể lấy hệ số của đa thức theo đường gấp khúc màu đen và được gọi là công thức ‘’ *một tiến một lùi* ‘’.

Vậy ta có :



 (6)



Công thức (6) gọi là công thức nội suy **Gauss thứ hai**.

Thay  để đưa đa thức p(x) trở về dạng ban đầu: biểu thức (3)







Thì ta sẽ có:

**\*\* Thuật toán:** tương tự với Gauss1

1. **So sánh công thức nội suy Gauss I và Gauss II**

Công thức nội suy Gauss I bắt đầu tại , sử dụng , rồi tới  ,..., tới mốc , rồi kết thúc tại mốc  .

Công thức Gauss II bắt đầu tại , sử dụng , rồi tới  ,..., tới mốc , rồi kết thúc tại mốc  .

Do vậy, trong những bài toán tính giá trị tại điểm cho trước, việc sử dụng Gauss I hoặc Gauss II sẽ có những lợi thế khác nhau

**Ví dụ**: ta vẫn có 2n+1 mốc nội suy cách đều được sắp xếp theo thứ tự:



* Giả sử cần tính giá trị tại điểm:  với  thì:

+ Khi sử dụng công thức nội suy Gauss I ta chỉ cần sử dụng đến mốc .

+ Còn khi sử dụng công thức nội suy Gauss II thì sau khi sử dụng mốc  ta còn phải sử dụng thêm mốc  mới có thể kết thúc bài toán.

* Nên sử dụng công thức Gauss I
* Tương tự như vậy với bài toán cần tính giá trị tại  với  thì:

+ Khi sử dụng công thức nội suy Gauss I ta chỉ cần sử dụng đến mốc .

+ Còn khi sử dụng công thức nội suy Gauss II thì sau khi sử dụng mốc  ta còn phải sử dụng thêm mốc  mới có thể kết thúc bài toán.

* Nên sử dụng công thức Gauss II

**IV – Công thức nội suy Sterling**

1. **Ý tưởng xây dựng**

Lấy *trung bình cộng* hai công thức nội suy Gauss thứ nhất (5) và công thức nội suy Gauss thứ 2 (6) ta được công thức nội suy **Sterling.**





 (7)



Như vậy công thức Sterling ta sử dụng cá sai phân theo sơ đồ **Bảng 2**

Gọi các hệ số  lần lượt là hệ số của p(x) tức là ta viết lại p(x) như sau:







Như vậy, các hệ số sẽ được tính theo công thức tổng quát sau:



***Có thể nói****: công thức nội suy Stirling chính là trung bình cộng 1 cách “hình thức” giữa 2 công thức Gauss I và Gauss II*

1. **Ưu điểm của công thức**
2. **Ưu điểm trong việc tính toán**

Từ các hệ số của công thức nội suy Sterling ta có thể thấy được:









Mặt khác ta có:







Như vậy có thể thấy, tất các các sai phân cấp “ lẻ ” trong các hệ số của đa thức đều có thể biến đổi về sai phân cấp “chẵn” một cách “hệ thống”.

Và các sai phân cấp “chẵn” đều có thể biến đổi về các sai phân cấp “chẵn” - thấp hơn nó một cách “hệ thống”.

Do đó, trong việc tính toán các hệ số của đa thức nội suy thì: “Ta chỉ cần tính các sai phân cấp chẵn của hàm số f(x) ” là ta có thể tính được toàn bộ các hệ số của đa thức nội suy và xây dựng được đa thức.

1. **Ưu điểm trong sử dụng đa thức sội suy**

Ta có thể thấy trong công thức nội suy Gauss sẽ có 2 công thức Gauss I và Gauss II, và mỗi công thức sẽ có ưu điểm riêng trong việc sử dụng nó để dùng vào bài toán tính giá trị.

Ngay từ ý tưởng xây dựng công thức nội suy Sterling ta có thể thấy, công thức này đã tích hợp lại được cả 2 công thức Gauss I và Gauss II vào bên trong nó:

Xuất phát từ mốc  , rồi sử dụng đồng thời  ,..., cứ như vậy dùng đến  rồi kết thúc.

Từ đó, ta thấy được khi sử dụng công thức nội suy Sterling trong việc tính toán giá trị sẽ không cần phải xem sét việc điểm cần tính là nằm phía bên nào so với mốc nằm ở trung tâm là .

1. **Ưu điểm các bước sử dụng mốc (xét việc cố định sử dụng 2n+1 mốc)**

Ta dễ thấy 2 công thức Gauss I và Gauss II sử dụng lần lượt các mốc theo thứ tự sau:

Gauss I: 

Gauss II: 

Như vậy, cả 2 công thức Gauss I và Gauss II đều phải mất tổng 2n lần sử dụng các mốc để giải quyết vấn đề.

Nhưng ta xét với công thức nội suy Sterling ta sẽ thấy việc công thức đã bắt đầu từ  sau đó sử dụng đồng thời 2 mốc với :



Do đó ta chỉ phải mất n lần sử dụng công thức, đã giảm đi 1 nửa so với 2 công thức Gauss I và Gauss II. Như vậy tốc độ giải quyết bài toán cũng đã nhanh hơn khi sử dụng công thức Sterling.

1. **Sai số của công thức nội suy Stirling**

Do sử dụng bảng sai phân cấp 2n nên

 (8)

Trong đó ta có: 

Sai số của đa thức nội suy, nếu  chưa biết ta cũng có thể thay:

 ta được

 (9)

**Thử với ví dụ trên:** vẫn dựa vào bảng sai phân trung tâm đã lập được ở trên ta thay vào công thức nội suy Stirling ta được kết quả sau.



Thế trực tiếp:  ta thu được:



Rút gọn biểu thức trên: 

**Hoàn toàn trùng khớp!!!**

Ví Dụ 8: giáo trình GT Số/147

**\*\* Thuật toán:**

* Input: + Vector X,Y tương ứng với các mốc và giá trị

+ Sai số mong muốn sẽ đạt được : eps

+ điểm cần tính giá trị x1

* Output: + hàm P sau khi đã rút gọn.

+ sai số đã tính được đối với điểm nhập vào x1

+ Giá trị của hàm tại điểm x1

* Thực hiện:

+ B1: Tìm mốc nội suy gần giá trị cần tính nhất x[a]

+ B2: Lập và hoàn thiện bảng sai phân (chỉ gồm các sai phân cấp lẻ)

+ B3: Coi mốc x[a] là gốc: tính hệ số b[i] của đa thức  với hệ số chẵn và lẻ tính riêng rồi gép với phần chứa ẩn t

+ B4: thực hiện vòng lặp: (tới khi hết hệ số)

++ lần lượt gán hệ số vào đa thức

++ so sánh sai số: nếu sai số đạt được thì dừng vòng lặp.

+ thay biến x cho t

+ In ra màn hình kết quả.

**V – Công thức nội suy Bessel**

1. **Ý tưởng xây dựng công thức**

Cho  mốc nội suy các đều sau:



Xuất phát từ mốc  sử dụng công thức nội suy Gauss thứ 2 (6) :



 (10)



Xuất phát từ mốc , với  thì



Viết lại công thức (10) với mốc  , nghĩa là thay  bởi  trong công thức (10) ta được





 (11)

***Trung bình cộng*** giữa công thức (11) và công thức nội suy Gauss thứ 1 (5) ta được:







 (12)



Trong đó: 

Gọi các hệ số  lần lượt là hệ số của p(x) tức là ta viết lại p(x) như sau:









Như vậy, các hệ số sẽ được tính theo công thức tổng quát sau:



Có thể nói: công thức nội suy Bessel chính là trung bình cộng cả về “hình thức” và “tính toán” của 2 công thức Gauss I và Gauss II

1. **Ưu điểm của công thức**

Về cơ bản công thức nội suy Bessel cũng có 3 ưu điểm tương tự như công thức nội suy Stirling đã kể trên. Nhưng công thức nội suy Bessel chỉ cần tính sai phân cấp “chẵn”.

**Thuật toán:** Tương tự với công thức Sterling, chỉ khác phần vòng lặp tính hệ số và gán

* Input: + Vector X,Y tương ứng với các mốc và giá trị

+ Sai số mong muốn sẽ đạt được : eps

+ điểm cần tính giá trị x1

* Output: + hàm P sau khi đã rút gọn.

+ sai số đã tính được đối với điểm nhập vào x1

+ Giá trị của hàm tại điểm x1

* Thực hiện:

+ B1: Tìm mốc nội suy gần giá trị cần tính nhất x[a]

+ B2: Lập và hoàn thiện bảng sai phân (chỉ gồm các sai phân cấp chẵn)

+ B3: Coi mốc x[a] là gốc: tính hệ số b[i] của đa thức  với hệ số chẵn và lẻ tính riêng rồi gép với phần chứa ẩn t

+ B4: thực hiện vòng lặp: (tới khi hết hệ số)

++ lần lượt gán hệ số vào đa thức

++ so sánh sai số: nếu sai số đạt được thì dừng vòng lặp.

+ thay biến x cho t

+ In ra màn hình kết quả.

1. **Nghịch lý xuất hiện trên công thức Bessel**

Ta đều biết định lý duy nhất của đa thức nội suy: *nếu sử dụng với số mốc cố định thì đa thức nội suy thu được là duy nhất*. Nhưng trong công thức nội suy Bessel điều đó không còn đúng nữa.

Thật vậy, với 2n+2 mốc nội suy bất kì được đánh số theo thứ tự:



* Nếu sử dụng công thức với 2n+2 mốc như trên với công thức nội suy Bessel thì ta sẽ thu được một đa thức nội suy (1)
* Nếu sử dụng 2n+2 mốc như trên với công thức nội suy Gauss I, Gauss II hoặc Stirling thì:

+ Nếu ta bắt đầu từ mốc  thì ta sẽ sử dụng 2n+1 mốc và thừa ra mốc  và thu được đa thức nội suy (2)

+ Nếu ta bắt đầu từ mốc  thì ta sẽ sử dụng 2n+1 mốc và thừa ra mốc  và thù được đa thức nội suy (3)

Đa thức (2) và (3) đều được xây dựng từ 2n+1 mốc nhưng có sự khác nhau giữa 2 mốc  và  , còn 2n mốc còn lại được sử dụng hoàn toàn giống nhau. Do vậy, đa thức thu tuy về cơ bản là giống nhau nhưng vẫn có sự **khác nhau**.

Như vậy, ta có thể thấy đều sử dụng 2n+2 mốc cố định nhưng các đa thức thu được là khác nhau.

***Code:***

**Gass1 :**

function [R, DeltaY, a, P] = Gauss1(X, Y , eps, x1)

%INPUT: % 2 vector X,Y

% diem can tinh gia tri: x1

% sai so mong muon: eps

syms x;

m = length(X);

h = X(2) - X(1);

b = 1;

for k=1:m %% Tim vi tri moc gan x1 nhat-----------------------------------

if abs(X(k)- x1) <= h/2

b = k;

break;

end

end %%-------------------------------------------------------------------

if b <= m-b

n = b-1;

else

n = m-b;

end

Z = zeros(1,2\*n + 1);%% doi Y thanh Z, 2n+1 vi tri --------------------

for i = 1:(2\*n +1)

Z(i) = Y(b-1-(n-i));

end %% ----------------------------------------------------------------

DeltaY = zeros(length(Z), 2\*n + 1);%% Tinh sai phan DeltaY-------------

DeltaY(:,1) = Z;

for i = 1:(2\*n);

for j = 1:length(Z)-i

DeltaY(j,i+1) = DeltaY(j+1,i) - DeltaY(j,i);

end

end %% ----------------------------------------------------------------

a = zeros(1, 2\*n+1); %% Tinh he so a(i)--------------------------------

a(1) = Z(n+1);

fac = 1;

pow = 1;

for i = 1:n

fac = fac \* (2\*i-1);

pow = pow \* h;

a(2\*i) = DeltaY(n-i+2, 2\*i) / (fac \* pow);

fac = fac \* 2\*i;

pow = pow \* h;

a(2\*i+1) = DeltaY(n-i+1,2\*i+1) / (fac \* pow);

end %% ----------------------------------------------------------------

P = a(1);

F = 1;

G = DeltaY(1,2\*n+1)/(fac);

for i = 1:n %%% Gan vao da thuc P -------------------------------------

F = F \* (x - X(b + 1 - i));

P = P + a(2\*i) \* F;

F = F \* (x - X(b + i));

P = P + a(2\*i+1) \* F;

G = (DeltaY(1,2\*n+1)/(fac))\*F;

if abs(subs(G,x1)) <= eps %% Dieu kien sai so

saiso = vpa(abs(subs(G,x1)))

break;

else

saisofaid = vpa(abs(subs(G,x1)))

end

end %%% ---------------------------------------------------------------

P = expand(P)

R = vpa(subs(P,x1))

end

**Gauss2:**

function [R, DeltaY, a, P] = Gauss2(X, Y , eps, x1)

%INPUT: % 2 vector X,Y

% diem can tinh gia tri: x1

% sai so mong muon: eps

syms x;

m = length(X);

h = X(2) - X(1);

b = 1;

for k=1:m %% Tim vi tri moc gan x1 nhat-----------------------------------

if abs(X(k)- x1) <= h/2

b=k;

break;

end

end %%-------------------------------------------------------------------

if b <= m-b

n = b-1;

else

n = m-b;

end

Z = zeros(1,2\*n + 1);%% doi Y thanh Z, 2n+1 vi tri --------------------

for i = 1:(2\*n +1)

Z(i) = Y(b-1-(n-i));

end %% ----------------------------------------------------------------

DeltaY = zeros(length(Z), 2\*n + 1);%% Tinh sai phan DeltaY-------------

DeltaY(:,1) = Z;

for i = 1:(2\*n);

for j = 1:length(Z)-i

DeltaY(j,i+1) = DeltaY(j+1,i) - DeltaY(j,i);

end

end %% ----------------------------------------------------------------

a = zeros(1, 2\*n+1); %% Tinh he so a(i)--------------------------------

a(1) = Z(n+1);

fac = 1;

pow = 1;

for i = 1:n

fac = fac \* (2\*i-1);

pow = pow \* h;

a(2\*i) = DeltaY(n-i+1, 2\*i) / (fac \* pow);

fac = fac \* 2\*i;

pow = pow \* h;

a(2\*i+1) = DeltaY(n-i+1,2\*i+1) / (fac \* pow);

end %% ----------------------------------------------------------------

P = a(1);

F = 1;

fac = fac\*(2\*n+1);

pow = pow\*h;

for i = 1:n %%% Gan vao da thuc P -------------------------------------

F = F \* (x - X(b - 1 + i));

P = P + a(2\*i) \* F;

F = F \* (x - X(b - i));

P = P + a(2\*i+1) \* F;

G = (DeltaY(1,2\*n+1)/(fac))\*F;

if abs(subs(G,x1)) <= eps %% Dieu kien sai so

saiso = vpa(abs(subs(G,x1)))

break;

else

saisofaid = vpa(abs(subs(G,x1)))

end

end %%% ---------------------------------------------------------------

P = expand(P)

R = vpa(subs(P,x1))

end

**Stirling:**

function [R, DeltaY, a, P] = Stirling(X, Y , eps, x1)

%INPUT: % 2 vector X,Y

% diem can tinh gia tri: x1

% sai so mong muon: eps

syms x t;

m = length(X);

h = X(2) - X(1);

b = 1;

for k=1:m %% Tim vi tri moc gan x1 nhat-----------------------------------

if abs(X(k)- x1) <= h/2

b=k;

break;

end

end %%-------------------------------------------------------------------

if b<=m-b

n = b-1;

else

n = m-b;

end

Z = zeros(1,2\*n + 1);%% doi Y thanh Z, 2n+1 vi tri --------------------

for i = 1:(2\*n +1)

Z(i) = Y(b-1-(n-i));

end %% ----------------------------------------------------------------

DeltaY = zeros(length(Z), 2\*n + 1);%% Tinh sai phan DeltaY-------------

DeltaY(:,1) = Z;

for i = 2:2:(2\*n);

for j = 1:length(Z)-i

DeltaY(j,i+1) = DeltaY(j+2,i-1) + DeltaY(j,i-1) - 2\*DeltaY(j+1,i-1);

end

end %% ----------------------------------------------------------------

a = zeros(1, 2\*n+1); %% Tinh he so a(i)--------------------------------

a(1) = Z(n+1);

fac = 1;

for i = 1:n

fac = fac \* (2\*i-1);

a(2\*i) = (DeltaY(n-i+3, 2\*i-1) - DeltaY(n-i+1,2\*i-1)) / (2\*fac);

fac = fac \* 2\*i;

a(2\*i+1) = DeltaY(n-i+1,2\*i+1) / (fac);

end %% ----------------------------------------------------------------

t1 = (x1 - X(b))/(h);

F = t;

P = a(1) + a(2) \* F;

G = t^2;

P = P + a(3) \* G;

for i = 2:n %%% Gan vao da thuc P -------------------------------------

F = F \* (t^2 - (i-1)^2);

P = P + a(2\*i) \* F;

G = G \* (t^2 - (i-1)^2);

P = P + a(2\*i+1) \* G;

L = vpa((DeltaY(1,2\*n+1)/(fac))\*G);

if abs(subs(L,t1)) <= eps %% Dieu kien sai so

saiso = vpa(abs(subs(L,t1)))

break;

else

saisofaid = vpa(abs(subs(L,t1)))

end

end %%% ---------------------------------------------------------------

P = expand(P)

R = vpa(subs(P,t1))

P = subs(P,t,(x - X(b))/(h));

P = expand(P)

end

**Bessel**

function [R, DeltaY, a, P] = Bessel(X, Y , eps, x1)

%INPUT: % 2 vector X,Y

% diem can tinh gia tri: x1

% sai so mong muon: eps

syms x t;

m = length(X);

h = X(2) - X(1);

b = 1;

for k = 1:m %% Tim vi tri moc gan x1 nhat---------------------------------

if abs(X(k) - x1) < h

b = k;

break;

end

end %%-------------------------------------------------------------------

if b <= m - b

n = b - 1;

else

n = (m - b - 1);

end

Z = zeros(1,2\*n + 2);%% doi Y thanh Z, 2n+1 vi tri --------------------

for i = 1:(2\*n + 2)

Z(i) = Y(b-1-(n-i));

end %% ----------------------------------------------------------------

DeltaY = zeros(length(Z), 2\*n + 2); %% Tinh sai phan DeltaY------------

DeltaY(:,1) = Z;

for i = 1:2\*n+1

DeltaY(i,2)=DeltaY(i+1,1) - DeltaY(i,1);

end

for i = 3:2:(2\*n+1);

for j = 1:length(Z)-i

DeltaY(j,i+1) = DeltaY(j+2,i-1) + DeltaY(j,i-1) - 2\*DeltaY(j+1,i-1);

end

end %% ----------------------------------------------------------------

a = zeros(1, 2\*n+2);

a(1) = (Z(n+1)+Z(n+2))/2;

a(2) = DeltaY(n+1,2);

fac = 1;

for i = 1:n %% Tinh he so a(i) ----------------------------------------

fac = fac \* (2\*i);

a(2\*i+1) = (DeltaY(n-i+3, 2\*i) - DeltaY(n-i+1,2\*i)) / (2\*fac);

fac = fac \* (2\*i+1);

a(2\*i+2) = DeltaY(n-i+1,2\*i+2) / (fac);

end %% ----------------------------------------------------------------

t1 = (x1 - X(b))/(h);

F = 1;

P = a(1);

G = t - 1/2;

P = P + a(2) \* G;

for i = 1:n %%% Gan he so vao da thuc ---------------------------------

F = F \* (t - i)\*(t + (i-1));

P = P + a(2\*i+1) \* F;

G = G\*(t - i)\*(t - (i-1));

P = P + a(2\*i+2) \* G;

L = (DeltaY(1,2\*n+2)/(fac))\*F;

if abs(subs(L,t1)) <= eps %% Dieu kien sai so ---------------------

saiso = vpa(abs(subs(L,t1)))

break;

else

saisofaid = vpa(abs(subs(L,t1)))

end

end %%% ---------------------------------------------------------------

P = expand(P)

R = vpa(subs(P,t1))

P = subs(P,t,(x - X(b))/(h));

P = expand(P)

end

**The End**