Báo cáo giải tích số:

Bài toán nội suy bằng đa thức,Sơ đồ Horner và ứng dụng

Nguyễn Đức Trung-Kstn Toán Tin K60-MSSV:20153975

Phần 1:Nội suy bằng đa thức:

- Mở đầu:
  - ✓ Vấn đề:
    - o Quan tâm đến giá trị $f(x)$tại $x=x*$ hoặc muốn phân tích hàm $f(x)$
    - o (Chỉ biết giá trị hàm f tại $x=x_{i} \forall i=\bar{0,n})$hoặc (biết công thức của $f(x)$ nhưng công thức phức tạp)
  - ✓ Ý tưởng:
    - o Thay hàm f bởi hàm F đơn giản hơn vẫn thỏa mãn f "cách" F đủ gần.
    - o Việc thay f bởiF gọi là xấp xỉ hàm.
  - ✓ Xap xi ham là 1 chủ dề lớn ;trong phần đầu này,ta quan tâm đến ham F la đa thức (do dễ tính theo sđ Hoocne; tính chất đẹp như khả vi vô hạn lần trên hợp các khoảng của $mathbb{R}$…)va quan tam den bài toán nội suy :tìm đa thức xấp xỉ hàm mà đi qua các điểm đã cho.
- Phát biểu định lí Weierstrass
  - ✓ Phát biểu: Tập các đa thức từ [a,b] vào ℝ là trù mật trong $C([a,b];\mathbb{R})$ đối với chuẩn $\|.\|_\infty$

$$(\| f \|_\infty = \sup_{t\in[a,b]} | f(t) | \quad)$$

  - ✓ Kết quả này cho biết : có thể xấp xỉ f bởi đa thức với độ chính xác tùy ý.

- Giải quyết bài toán nội suy bằng đa thức.Nêu cách xây dựng đa thức nội suy Lagrange
  - ✓ Phát biểu bài toán: Cho bộ dữ liệu{x_{i},y_{i}}(i=0;1;…;n) với x_{i}đôi một phân biệt.Tìm đa thức bậc không quá n $P_{n}(x)$ sao cho

$P_{n}(x_{i})= y_{i} \forall i=\bar{0,n}$.Một đa thức như vậy luôn tồn tại và là duy nhất.

- ✓ Giải: $P_n(x)=a_0x^n+a_1x^{n-1}+...+a_n$ Từ $P_n(x_i)=y_i\forall i=0;1;...;n$ ta lập hệ pt(viết dưới dạng ma trận):Ax=y trong đó

$$\mathbf{A}=\begin{pmatrix} x_0^n & \dots & 1 \\ x_1^n & \dots & 1 \\ \vdots & \ddots & \\ x_n^n & \dots & 1 \end{pmatrix}$$

$$\mathbf{x}=\begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix}$$

$$\mathbf{y}=\begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}$$

Theo kq quen biết;A khả nghịch.Vậy hệ có nghiệm duy nhất $x=A^{-1}y$ .Chứng tỏ P_n xác định duy nhất.

- ✓ Cách trên ko tốt để tìm đa thức nội suy vì hệ thiết lập được là không ổn định.
- ✓ Để ý rằng bản chất cách trên là chọn 1 cơ sở $\{q_0(x);q_1(x);...;q_n(x)\}$ của không gian po_n các đa thức bậc không quá n trên trường thực(hiển nhiên là ko gian này có chiều là n+1),ta có biểu diễn $P_n=a_oq_0+...+a_nq_n$ và từ các dữ kiện $P_n(x_i)=y_i$ thiết lập được hệ :Ax=y trong đó

$$A=\begin{pmatrix} q_o(x_0) & q_1(x_0) & \dots & q_n(x_0) \\ . & & & . \\ . & & & . \\ q_o(x_n) & q_1(x_n) & \dots & q_n(x_n) \end{pmatrix}$$

$$\mathbf{x}=\begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix}$$

$$\mathbf{y}=\begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}$$

suy ra x=$A^{-1}y$.(Dĩ nhiên;bộ {q_i}(i=0,..,n) là cở sở của không gian po_n nếu và chỉ nếu A(như trên) khả nghịch vì theo mệnh đề tồn tại và duy nhất ;mỗi cột i của A sẽ xác định duy nhất đa thức q_i(i=0,..,n)nên nếu một cột của A biểu thị tuyến tính theo các cột còn lại sẽ dẫn đến 1 đa thức q_i biểu thị tuyến tính theo các đa thức còn lại).Ý tưởng là đi tìm cơ sở của po_n sao cho A^{-1} dễ tính.Đơn giản nhất A là ma trận đơn vị(x=y hay $a_i = y_i \forall i = 0,..,n$).Khi đó từ các sự kiện :q_i nhận mọi x_{j}(j khác i)làm nghiệm; $q_i(x_i) = 1$; $\deg(q_i) \le \mathrm{n}$ xác định

được $q_i\left(x\right) = \dfrac{\prod\limits_{\substack{j \ne i \\ j=0}}^{n}(x - x_j)}{\prod\limits_{\substack{j \ne i \\ j=0}}^{n}\left(x_i - x_j\right)}$ (bộ cơ sở Lagrange)

- ✓ Điều thú vị là ta có thể "tình cờ" nhận được đa thức nội suy Lagrange:

Từ $\begin{vmatrix} P(x) & 1 & .. & x^n \\ y_0 & 1 & .. & x_o^n \\ ... & ... & & ... \\ y_n & 1 & .. & x_n^n \end{vmatrix}$ =0;Khai triển Laplace theo cột 1 ta được điều cần tìm.

Phần 2:Sơ đồ Hoocner và ứng dụng:

- Xây dựng sơ đồ.Nêu lợi ích của việc tính bằng sơ đồ

  - ✓ Giả sử $p\left(x\right) = \sum\limits_{i=0}^{n} a_i x^{n-i} = (...((\mathrm{a}_0\, x + a_1)x + a_2)x... + a_n)$ $\left(a_i \in \mathbb{C}, x \in \mathbb{C}\right)$ (1)

Khi x=c tính: $\begin{matrix} b_0 = a_0; \\ \mathrm{b}_i = a_i + c.b_{i-1} \forall i = \overline{1n} \end{matrix}$ (2) thì $p\left(c\right) = b_n$

(dễ thấy khi nhìn vào cách viết số 2)

Bảng mô tả quá trình trên gọi là sđ Hoocne:

Hệ số của P	a_0	a_1	...	a_n
		cb_0	...	cb_{n-1}
C	b_0	b_1	...	$b_n = p\left(c\right)$

- ✓ Sơ đồ Horner giúp giảm khối lượng tính toán.Cụ thể:

- Tính tuần tự: khoảng $\frac{n^2}{2}$ phép nhân; n phép cộng.
- Tính bằng sơ đồ Horner: n phép nhân; n phép cộng.

- Các ứng dụng:
  1. Tính giá trị đa thức tại 1 điểm
     - ✓ Thuật toán:

```
function [v,q]=horner1(p,c)%tinh p tai x=c va dua ra thuong p/(x-c)
m=length(p);
v=p(1);q= [zeros(1,m-1)];q(1)=p(1);
for i=2:m-1
    v=v*c+p(i);q(i)=v;
end
if m>1
    v=v*c+p(m);
end
end
```

  2. Nhân, chia đa thức ("thuật toán" hóa cách nhân chia đa thức thông thường) (divide-multy)
     - ✓ Lí thuyết:

Xét $f(x) = a_0.x^m + ... + a_m$ và $g(x) = b_0.x^n + ... + b_n$
Đặt $q(x) = c_0.x^{m-n} + ... + c_{m-n}; r(x) = d_{m-n+1}.x^{n-1} + ... + d_m$
Viết lại
$g(x) = g(x) = b_0.x^n + ... + b_n + b_{n+1}.x^{-1} + ... + b_m.x^{-(m-n)}; q(x) = c_0.x^{m-n} + ... + c_{m-n} + c_{m-n+1}.x^{-1} + ... + c_m.x^{-n}; r(x) = d_0.x^m + ... + d_{m-n}.x^n + d_{m-n+1}.x^{n-1} + ... + d_m$ trong đó $c_{m-n+1} = ... = c_m = b_{n+1} = ... = b_m = d_0 = ... = d_{m-n} = 0$
từ $f(x) = g(x).q(x) + r(x)$ suy ra $\forall i = \overline{0,m}$ thì $a_i = \sum_{k=0}^{i} c_k b_{i-k} + d_i \Rightarrow c_0 = \frac{a_0}{b_0}; c_i = \frac{a_i - \sum_{k=0}^{i-1} c_k b_{i-k}}{b_0} \forall i = \overline{0, m-n}$ và $d_i = a_i - \sum_{k=0}^{i} c_k b_{i-k} \forall i = \overline{m-n+1, m}$

     - ✓ Thuật toán chia đa thức:

```
function [q,d]=divide(f,g)%tim thuong q va phan du d khi chia f cho g
m=length(f)-1;
n=length(g)-1;
a=g(1);
q=zeros(1,m+1);
d=zeros(1,m+1);
f=(1/a)*f;%dau tien,ta chia ca 2 da thuc f,g cho he so cua hang tu bac cao nhat trong g(x) thu ???c f*,g*;nh
g=(1/a)*g;
q(1)=f(1);
g=[g zeros(1,m-n)];
for i=2:m-n+1
    t=0;
    for j=1:i-1
        t=t+q(j)*g(i-j+1);
    end
    q(i)=f(i)-t;
end
for i=m-n+2:m+1
    t=0;
    for j=1:i
        t=t+q(j)*g(i+1-j);
    end
    d(i)=f(i)-t;
end
q=q([1:m-n+1]);%?ó lý do vì sao ph?i nhân a vào d ? ?ây
for i=1:length(d)
    if d(i)~=0
        break
    end
end
d=a*d(i:length(d));
end
```

VD:

```
>> f=[1 2 3 4 5 6 7 8]

f =

     1     2     3     4     5     6     7     8

>> g=[12 13 15]

g =

    12    13    15

>> [q,d]=divide(f,g)

q =

   0.083333333333333   0.076388888888889   0.063078703703704   0.169511959876543   0.154180330504115   0.121081358774863


d =

   3.113237378365053   6.183779618377056

>> deconv(f,g)

ans =

   0.083333333333333   0.076388888888889   0.063078703703704   0.169511959876543   0.154180330504115   0.121081358774863
```

✓ Thuật toán nhân đa thức:

```
function f=multy(d,q)
% tim tich 2 da thuc:f=d.q
m=length(d)-1;n=length(q)-1;
d=[d zeros(1,n)];q=[q zeros(1,m)];f=zeros(1,m+n+1);
%chen them so 0 vao trong bieu dien hinh thuc cua d va q
%viec nay giup tinh he so f theo cong f(i)=\sum_{k=0}^{i}d(k).g(i-k)
for i=1:m+n+1
    for j=1:i
        f(i)=f(i)+d(j)*q(i+1-j);
    end
end
end
```

3. Tìm đạo hàm mọi cấp của đa thức tại 1 điểm(derivative)

✓ Lý thuyết

Khai triển Taylor tại c:

$f(x) = f(c) + f'(c).(x-c) + ... + \frac{f^{(m)}(c)}{m!}(x-c)^m(1)$

Từ đó, $\frac{f(x)-f(c)}{x-c} = q_1(x) \Rightarrow q_1(c) = f'(c)$; $\frac{q_i(x)-q_i(c)}{x-c} = q_{i+1}(x) \Rightarrow q_{i+1} = \frac{f^{(i+1)}(c)}{(i+1)!} \forall i = \overline{1,m}$

✓ Thuật toán:

```
function d=derivative(p,c)%tim dao ham moi cap cua p tai x=c;xuat ra vector d
m=length(p);
d=[1:1:m];
q=p;
for i=1:m
    [v,q]=horner1(q,c);%horner1(f,c) là chương trình tình f(c) và đa thức f/(x-c)
    d(i)=v*factorial(i-1);
end
d;
end
```

VD:

```
>> f=[1 2 3 4 5 6];
>> derivative(f,2)

ans =

   120   201   300   354   288   120
```

4. Rời rạc hóa đạo hàm:tìm tỉ hiệu
   - ✓ Input:
     - o Đa thức p(x) bậc n
     - o Bộ số (x_0;x_1;…;x_{n})
   - ✓ Output:
     - o Các tỷ hiệu dạng:p[x_0;x_1;…;x_{i}](i=0;1;…;n)
   - ✓ Thuật toán:
     - o Biểu diễn p(x) dưới dạng nội suy Newton: p(x)=p(x_0)+(x-x_0)p[x_0;x_1]+(x-x_0)(x-x_1)p[x_0;x_1;x_2]+…+(x-x_0)…(x-x_{n-1})p[x_0;…;x_{n}].
       =p(x_0)+(x-x_0)(p[x_0;x_1]+(x-x_1)(p[x_0;x_1;x_2]+(x-x_2)(…(p[x_0;…;x_{n-1}]+(x-x_{n})p[x_0;..;x_n])..)
     - o Dùng sơ đồ Hocner tính được p(x_0) và đa thức q_{1}(x)=$\dfrac{p(x)-p(x_o)}{x-x_o}$
     - o Dùng sơ đồ Hocner tính được $q_1(x_1)=p[x_0;x_1]$ và đa thức $q_2(x)=\dfrac{q_1(x)-q_1(x_1)}{x-x_1}$
     - o …

5. Kết hợp phương pháp Newton, tìm gần đúng nghiệm thực của pt đa thức
   - ✓ Sau bước lặp thứ n cuả phương pháp tiếp tuyến,ta nhận được x_n .Để tìm x_{n+1} cần tính $f(x_n), f'(x_n)$ .tại đây có thể áp dụng thuật toán ở 1. và 3.

6. Đổi biến (doibien)
   - ✓ Lý thuyết:

Từ công thức Taylor (1) nếu đặt $y = x - c$,ta được:
$f(x) = g(y) = f(c) + f'(c)y + ... + \frac{f^{(m)}(c)}{m!}y^m(1)$

✓ Thuật toán:

```
function f=doibien(p,c)
m=length(p);
d=[1:m];f=[1:1:m];
for i=1:m
    [v,p]=horner1(p,c);
    d(i)=v;
end
for i=1:m
    f(i)=d(m+1-i);
end
end
```

VD:

```
>> f=[1 2 3 4 5];
>> doibien(f,1)

ans =

     1     6    15    20    15
```

7. Khai căn bậc n của 1 số thực(nếu căn bậc n tồn tại)

✓ Lý thuyết:

- Đưa bài toán về trường hợp số thực dương a vì khi n lẻ, $\sqrt[n]{a} = -\sqrt[n]{-a} \forall a \in \mathbb{R}$;khi n chẵn số thực dương a có hai căn bậc n trái dấu là $\pm\sqrt[n]{a}$
- nhận xét đơn giản sau để tìm số chữ số thập phân của $\lfloor \sqrt[n]{a} \rfloor$(trong bước 1 của thuật toán sẽ trình bày dưới đây):

  Nếu $\lfloor a \rfloor$ có k chữ số thập phân thì $\lfloor \sqrt[n]{a} \rfloor$ có $\left\lceil \frac{k}{n} \right\rceil$ chữ số thập phân.(để chứng minh điều này chỉ cần chú ý:
  $\lfloor a \rfloor$ có k chữ số thập phân ⇔ $10^{k-1} \le a < 10^k$)
- Ý tưởng phương pháp:
  Ta tìm nghiệm thực dương của phương trình $x^n = a$.Ý tưởng là tìm từng chữ số bắt đầu từ chữ số đầu tiên của phần nguyên của $\sqrt[n]{a}$

Giả sử $\sqrt[n]{a} = x_0x_1x_2..x_m, y_1\, y_2...$ (x_i, y_i là các chữ số, $x_o \neq 0$)

Đa thức $10^{n(m-1)}x^n - a$ có nghiệm thực dương là $\frac{\sqrt[n]{a}}{10^{m-1}} = x_o, x_1x_2...$nên f có nghiệm dương duy nhất trong [$x_0, x_0 + 1$].Nếu cho i=1:10 thì $x_0 + 1$ là vị trí đầu tiên f(i)>0.

Sau khi tìm được x_0,tìm x_1 như sau:

B1:gán g=doibien(f,x_0)

B2:gán $g = 10^n.g\left(\frac{x}{10}\right)$;g có nghiệm dương duy nhất và nghiệm đó trong $[x_1, x_1 + 1]$.

Tương tự tìm các chữ số còn lại.

✓ Thuật toán:

```
function y=extract1(a,n,d)%tinh x= can bac n (duong) cua 1 so duong a voi d chu so chinh xac ke tu cs
Digit=ceil(numel(num2str(floor(a)))/n);%tim so chu so cua phan nguyen cua can bac n cua a
p=[10^(n*(Digit-1)) zeros(1,n-1) -a];
s=zeros(1,d);%s luu cac chu so cua can bac n cua a
for i=1:d
    for k=1:10
        if tinh(p,k)>0
            break;
        end
    end
    s(i)=k-1;
    if i<d
        p=doibien(p,s(i));
        for j=1:n+1
            p(j)=p(j)*10^(j-1);
        end
    end
end
y=0;
for i=1:d
    y=y+s(i)*(10^(Digit-i));
end
end
```

VD:

```
>>  extract1(13579,5,15)-nthroot(13579,5)

ans =

     0

>> extract1(13579,5,15)

ans =

   6.707698872422500
```

8. Tính tổ hợp chập k của n(hệ số nhị thức)
   - ✓ Xét đa thức $p(x)=x^n$;g(y)=doibien(p(x),1)

     Suy ra g(y)=$\sum_{k=0}^{n} C_n^k y^{n-k}$
9. Tính hoán vị của tập bội(permutation of multyset)(hay gọi là hệ số đa thức)(PerMulSet)
   - ✓ Lý thuyết:(trong phần này các số m,n nguyên dương;a_{i}nguyên ko âm)
     - o Ví dụ về tập bội:{a,a,a,b,b,c}.bội của a là 3,của b là 2,của c là 1.
     - o Tập bội có thể hiểu là tập gồm các phần tử có thể ko thỏa mãn đôi một khác nhau.
     - o Câu hỏi 1:có bao nhiêu hoán vị của các phần tử của 1 tập bội có n phần tử trong đó có m loại phần tử;phần tử loại i có bội là a_{i}?Trả lời: $\frac{n!}{a_1!a_2!...a_m!}$
     - o Tại sao gọi là hệ số đa thức?

       Xét đa thức m biến $\left(x_1+x_2+...+x_m\right)^n$ .các số hạng sau khi khai triển có dạng $x_1^{a_1}.x_2^{a_2}...x_m^{a_m}$ với $\sum_{i=1}^{m} a_i = n$ .Hệ số ứng với số hạng đó là $\frac{n!}{a_1!a_2!...a_m!}$ (liên hệ câu hỏi 1)
     - o Sau khi khai triển đa thức trên có bao nhiêu hạng tử?

       Tương đương với bài toán đếm số bộ các số nguyên ko âm $\left(a_1,...,\mathrm{a}_m\right)$ (có kể thứ tự)thỏa mãn phương trình $a_1+...+\mathrm{a}_m=n$

       Bằng cách thiết lập song ánh giữa mỗi bô số với 1 chuỗi nhị phân (n+m-1)bit gồm n bit 1,m-1 bit 0(trước bit 0 đầu tiên là a_1 bit 1;giữa bit 0 thứ i và thứ i+1 là a_{i+1} bit 1;sau bit 0 cuối cùng là a_m bit 1)Không khó khăn, ta đếm được có $\binom{n+m-1}{n}$ bộ.
     - o Thứ tự các bộ:

Với 2 bộ $a=(a_1,...,a_m);b=(b_1,...,b_m)$ ta nói $a<b \Leftrightarrow$ tồn tại i thỏa mãn: $a_j=b_j \forall j<i; a_i<b_i$ và a=b $\Leftrightarrow$ $a_i=b_i \forall i$ .dễ thấy 2 bộ m phần tử bất kì luôn so sánh được với nhau.

- ✓ Thuật toán: Tính hệ số đa thức như thế nào?
  - o Input: m,n nguyên dương.
  - o Output: các hệ số của đa thức m biến: $(x_1+x_2+...+x_m)^n$ sau khi khai triển.
  - o Thuật toán:

```
function vector=PerMulSet(m,n)
if m==1
    temp=[1];
elseif m==2
    temp=doibien([1 zeros(1,n)],1);
else
    temp=[];
    for i=0:n
        temp=[temp (factorial(n)/(factorial(i)*factorial(n-i)))*PerMulSet(m-1,n-i)];% de quy
    end
end
vector=temp;
end
```

VD:

```
>> PerMulSet(3,5)

ans =

     1     5    10    10     5     1     5    20    30    20     5    10    30    30    10    10    20    10     5     5     1
```

- ✓ Lưu ý:mỗi hệ số ứng với 1 bộ $a=(a_1,...,a_m)$,ta in ra các hệ số theo thứ tự các bộ a sắp xếp từ nhỏ đến lớn bởi quan hệ thứ tự được nghĩa ở trên.
- ✓ Sau khi in ra vector các hệ số,có nhu cầu muốn biết hệ số ở vị trí s ứng với bộ $a=(a_1,...,a_m)$ nào,ta sử dụng ý tưởng "tựa" 7. tìm từng a_i bắt đầu từ i=1.

  Tìm a_1 :đặt $S_k^1=\sum_{a_1=0}^{k}\binom{n-a_1+m-2}{m-2}$ .cho k chạy từ 0,chỉ số k nhỏ nhất làm cho $S_k^1 \geq s$ là a_1.

Sau khi tìm được $a_1,...,a_i$ ta tìm a_{i+1} bằng cách:đặt

$S_k^{i+1} = \sum_{a_{i+1}=0}^{k} \binom{n-\sum_{j=1}^{i+1} a_j + m - i - 2}{m-i-2}$;chỉ số k nhỏ nhất làm cho

$S_k^{i+1} \geq s - \sum_{j=1}^{i} S_{a_j-1}^{j}$ là a_{i+1} .

- Kết luận:
  Trên đây là 1 số ứng dụng của sơ đồ Horner.Sơ đồ Horner còn có ứng dụng khác trong tính toán(ví dụ tính số Stirling loại 2 tham khảo trong Tian-Xiao He and Peter J.-S. Shiue,A note on Horner's Method, Illinois Wesleyan University…).do điều kiện thời gian,người viết chưa có cơ hội tìm hiểu.

Phần 3:Bài toán:Giải phương trình đa thức(tìm gần đúng tất cả nghiệm thực) và đưa ra số bội của nghiệm(P)

- Note:theo "Oliver Aberth,Introduction_to_precise_numerical methods,pp 80,2007",bài toán "cho đa thức thực hoặc phức bậc n>1;xác định số bội của các nghiệm" là bài toán không thể giải được.Trong bài này,ta đưa ra số bội của nghiệm theo nghĩa nếu 2 nghiệm "đủ gần"(cách nhau không quá …) thì coi chúng là trùng nhau.
- Ý tưởng bài toán tìm nghiệm:
  - ✓ Việc giải bài toán tìm tất cả nghiệm quy về bài toán tìm nghiệm dương.Cụ thể:
    - Cho đa thức $p(x) = \sum_{i=0}^{n} a_i x^{n-i}$ ($a_0 \neq 0 \quad a_i \in \mathbb{R}$)
    - Xét $p_1(x) = x^n + \sum_{k=1}^{n} \frac{a_i}{a_0} x^{n-k}$ .dễ thấy p(x) và $p_1(x)$ có cùng tập nghiệm.
    - +)Nếu $p_1(x)$ không có nghiệm suy ra $p_1(x)$>0 với mọi x.
    - +) $p_1(x)$ có nghiệm x=0 $\Leftrightarrow a_n = 0$

- +)Nếu $p_1(x)$ có nghiệm âm. Xét

$$p_2(y) = (-1)^n p_1(x) = (-x)^n + \sum_{i=1}^{n} \frac{a_i.(-1)^i}{a_0}(-x)^{n-i} = y^n + \sum_{i=1}^{n} \frac{a_i.(-1)^i}{a_0} y^{n-i}$$

- $p_2(x)$ có tập các nghiệm dương là S_2 thì tập các nghiệm âm của $p_1(x)$ là $s_1 = \{-x \mid x \in s_2\}$
- Như vậy bài toán tìm nghiệm thực của đa thức bất kì quy về bài toán tìm nghiệm thực dương của đa thức hệ số bậc cao nhất bằng 1.

✓ Nếu ta có thể biết trên 1 khoảng tùy ý; đa thức có nghiệm hay không thì ta có thể dùng phương pháp chia đôi hoặc Hocner(như trong phần khai căn) để tìm gần đúng nghiệm.

✓ Định lý sau đây giúp ta biết trên 1 khoảng mà 2 đầu mút thỏa mãn 1 điều kiện "nhẹ",đa thức đã cho có bao nhiêu nghiệm phân biệt

2.Định lý Sturm

2.1.Một số định nghĩa

Một đa thức $p(x)$ gọi là tự do bậc 2 nếu không tồn tại đa thức $q(x), r(x)$ sao cho $p = q^2.r$

Một dãy Sturm trên 1 gian(tức là 1 đoạn,1 khoảng hoặc 1 nửa khoảng) là một dãy gồm hữu hạn đa thức $P_0(x), p_1(x), \ldots, p_m(x)$ có bậc giảm dần và thỏa mãn các điều kiện:

(i)$P_0(x)$ là tự do bậc 2

(ii)nếu a là nghiệm của $p_0(x) \equiv p(x)$ thì $p_1(a).p'(a) > 0$

(iii)nếu a là nghiệm của $p_i(x)$ với$0 < i < m$ thì $p_{i-1}(a)p_{i+1}(a) < 0$

(iv)$p_m(x)$khác không và có dấu không đổi với mọi x thuộc gian đang xét

2.2.Mệnh đề:

cho trước 1 đa thức p(x) tự do bậc 2,ta có thể xây dựng 1 dãy Sturm(trên 1 gian)như sau $p_0(x) = p(x)$,$p_1(x) = p'(x)$, $p_i(x) = -rem(p_{i-2}, p_{i-1}) \forall m \geq i \geq 2$ trong đó rem(f,g)là phần dư của phép chia đa thức f cho g;m là số sao cho rem(p_{m-1}, p_m)=0

Kí hiệu trong bài này,dãy được xây dựng theo cách trên là OS(viết tắt:ordinary Sturm) CM:

2 tính chất đầu là hiển nhiên.

tính chất thứ 3 có thể được suy ra bằng quy nạp:p_i, p_{i+1} không có chung nghiệm $\forall i \geq 0$

tính chất 4,ta thấy nếu p_m có nghiệm trên gian đang xét thì p_{m-1}, P_m chung nghiệm(trái với trên).từ đó suy ra tính chất 4

2.3.Định lý Sturm

2.3.1.Phát biểu định lý

cho đa thức tự do bậc 2 $p(x)$.Xây dựng 1 dãy Sturm với $p_0 \equiv p$ và 1 khoảng (a, b) sao cho $p_i(a), p_i(b) \neq 0 \forall i$.Với 1 hằng số c,giả sử $\beta(c)$là số lần đổi dấu trong dãy $p_0(c), p_1(c), \ldots, p_m(c)$ thì số nghiệm của $p(x)$ trên (a,b) là $\beta(a) - \beta(b)$

2.3.2.Hệ qua

cho đa thức $p(x)$ và khoảng (a,b).Xây dựng dãy OS như trên cho $p(x)$.Nếu $p_i(a)p_i(b) \;\; = 0 \forall i = \overline{1, m}$ thì $p(x)$ có $\beta(a) - \beta(b)$ nghiệm phân biệt trên (a,b).

Phép chứng minh các định lí và hệ quả có trong [2]

Ta có thể khắc phục điều kiện "nhẹ" trên bằng cách: nếu a,b không thỏa mãn $p_i(a)p_i(b) \neq 0 \forall i = 1;m$ thì gán a=a-epsi;b=b+epsi(epsi rất nhỏ) cho đến khi nhận được a,b thỏa mãn.

- ✓ Định lí về miền nghiệm của đa thức:
  1. Định lý 1:Đặt $A = max\{|a_i|\}$ nghiệm phức của p(x) thỏa mãn:

     $|x| < 1 + \frac{A}{|a_0|}$ =R

     (cm:sgk-t32)
  2. Định lý 2: Nếu $a_i > 0 \forall i = 0;n$ thì $p(x) > 0 \forall x > 0$

     Nếu $a_0 > 0, a_k$ là hệ số âm đầu tiên tính từ $k = 1,2,3...$ đặt $B = max$ các trị tuyệt đối của các hệ số âm thì nghiệm dương của p(x) thỏa mãn: $x < 1 + \sqrt[k]{\frac{B}{a_0}}$ =M

- ✓ Thuật toán:
  - o Input:đa thức p(x) hệ số thực có ít nhất 1 hệ số khác 0.
  - o Output:tất cả nghiệm thực dương của đa thức
  - o Thuật toán 1:(Tìm nghiệm địa phương bằng "tựa" chia đôi)(PositiveRootsByDual)
    1) Tìm miền nghiệm dương của đa thức(theo 1 trong 2 định lí trên)
    2) Kiểm tra trên (a,b) p(x) có nghiệm không?Nếu có B3;không có kết thúc
    3) c=b;kiểm tra trên đoạn (a,c) có 1 nghiệm phân biệt hay không?Nếu đúng chuyển sang bước 5;sai sang bước 4
    4) d=c;c=(a+c)/2,Nếu trên đoạn (a,c) không có nghiệm thì gán a=c;c=d.Quay lại bước 3
    5) Tìm nghiệm bằng phương pháp "tựa" chia đôi (dưới đây,quy định độ dài khoảng chia ko quá 10^(-5) thì dừng,trả lại nghiệm gần đúng là điểm chính giữa khoảng chia)
    6) a=c;quay lại bước 2
    7) in ra tất cả nghiệm dương và kết thúc
  - o Code:
    1) Phương pháp "tựa" chia đôi:

```
function root=Dual(p,a,b)
while (b-a)>10^-5
    c=(a+b)/2;
    [dem,a,c]=check_roots(p,a,c);
    if dem==0
        a=c;
    else b=c;
    end
end
root=(a+b)/2;
end
```

2) Thuật toán 1:

```
function r=PositiveRootsByDual(p)
    a=0;b=PositiveUpperBound(p);%(a,b) la khoang chua nghiem
    r=[];%bien luu nghiem
    while check_roots(p,a,b)~=0
        c=b;
        while check_roots(p,a,c)~=1
            d=c;c=(a+c)/2;
            if check_roots(p,a,c)==0
                a=c;c=d;
            end
        end
        ro=Dual(p,a,c);r=[r ro];
        a=c;
    end
end
```

- thuật toán 2(Tìm nghiệm địa phương bằng Horner và sau khi tìm được nghiệm r,chia đa thức cho (x-r);tìm tiếp nghiệm với đa thức mới)(PositiveRootsByHorner)

```
function r=PositiveRootsByHorner(p)
m=length(p)-1;r=[];c=1;
while (c~=0)
    R=PositiveUpperBound(p);
    c=check_roots(p,0,R);
    if c~=0
        for i=0:ceil(R)
            a=check_roots(p,i,i+1);
            if a~=0
                ro=FindLocalRoot(p,i);
            end
        end
    else
        ro=0;
    end
    r=[r ro];
    [v,p]=horner1(p,ro);
end
r=r(1:length(r)-1);
end
```

- Bài toán đưa ra số bội của nghiệm:
  - ✓ Thực tế;sai số epsi ,để quy định khi 2 nghiệm cách nhau không quá epsi thì coi 2 nghiệm trùng nhau, đã ở trong phần tìm nghiệm địa phương(ví dụ:Phương pháp "tựa" chia đôi:khoảng chia bé hơn 2epsi thì coi chính giữa khoảng là nghiệm;pp Horner epsi=10^{-i};i là số chữ số sau dấu phẩy của nghiệm mà ta muốn tìm chính xác)
    - o Epsi ta chọn buộc phải phù hợp với khả năng tính chính xác của máy tính(như giới hạn tính của máy;sự làm tròn…)
  - ✓ Ý tưởng:
    - o Nhận xét 1:nếu Đa thức p(x) có nghiệm thực bội m suy ra p(x) và $p^{(i)}(x)$ có nghiệm chung với mọi i=1;2;…;m-1
    - o Nhận xét 2:nếu đa thức p(x) có nghiệm thực bội m thì $p^{(i)}(x)$và $p^{(i+1)}(x)$có nghiệm chung $\forall i = 1;2;...;m-2$
    - o Nhận xét 3:thuật chia Euclid tìm đa thức Ước chung lớn nhất của 2 đa thức:

      Nếu p(x)=q(x).g(x)+r(x) thì

      UCLN(p;g)=UCLN(g;r)
    - o Thuật toán:
      1) Tìm và lưu lại ước chung lớn nhất của p(x) với đạo hàm các cấp của nó.giả sử đến cấp m thì UCLN bằng

hằng số.như vậy;đa thức p(x) không có bội quá m(vì ngược lại vô lí)

2) Tìm và lưu lại UCLN của $p^{(i)}(x); p^{(i+1)}(x)$ .Giả sử đến cấp k thì UCLN=hằng số.Suy ra p(x) không có nghiệm bội quá k
3) Thay nghiệm tìm được của đa thức p(x) vào hệ 1)(hoặc 2) bên trên.Nếu nó nghiệm đúng t phương trình thì nó sẽ là nghiệm bội t+1.(Nghiệm đúng theo nghĩa "gần đúng" tức là giá trị hàm tại điểm đó ko vượt quá sai số cho trước)

- Chú ý:

1) Ta có thể thay ngay nghiệm vào đạo hàm các cấp của đa thức đã cho

- Kết hợp tất cả những gì đã trình bày trên,ta có chương trình tìm nghiệm(lưu vào 1 vector r) và in ra số bội của nghiệm(lưu vào 1 vector m) như sau:

```
function [r,m]=FindMulRoots(f)
r=[];p=f;
for i=1:length(p)
    if p(i)~=0
        Dau=i;
        break
    end
end
for i=length(p):-1:1
    if p(i)~=0
        Cuoi=i;
        if Cuoi~=length(p)
            r=[0];
        end
        break
    end
end
p=p([Dau:1:Cuoi]);
if p(1)<0
    p=-p;
end
r=[r PositiveRootsByDual(p)];
for i=1:length(p)
    p(i)=p(i)*(-1)^(i-1);
end
r=[r -PositiveRootsByDual(p)];
if isempty(r)
    m=[];
else
    m=zeros(1,length(r));
    [~,G]=PrintGcd(f);
    for i=1:length(r)
        for j=1:size(G,1)
            if abs(polyval(G(j,:),r(i)))>10^-8
                m(i)=j;  %10^-8 la sai so ta chon de
                break   %qui dinh 1 so la nghiem hay ko
            end
        end
```

Ví dụ:

```
>> p=poly([1.23456 1.23457 1.23458])

p =

   1.000000000000000  -3.703710000000000   4.572489254600000  -1.881686019601536

>> [r,m]=FindMulRoots(p)

r =

   1.234564292401075   1.234568161561629   1.234581395287514


m =

     1     1     1
```