# Lagrange

%拉格朗日插值算法（多个插值计算，即xh可以取多个同时计算）

function yh=lagrange(x,y,xh)

n = length(x);

m = length(xh);

x = x(:); %将行矩阵x变成列矩阵

y = y(:);

xh = xh(:);

yh = zeros(m,1);

c1 = ones(1,n-1);

c2 = ones(m,1);

for i=1:n

xp = x([1:i-1 i+1:n]);

yh = yh + y(i) \* prod((xh\*c1-c2\*xp')./(c2\*(x(i)\*c1-xp')),2); %prod(A)表示将A矩阵的各列元素乘起来得到一个行矩阵

end

**HERMITE**

%hermite.m

%求埃尔米特多项式和误差估计的MATLAB主程序

%输入的量:X是n+1个节点(x\_i,y\_i)(i = 1,2, ... , n+1)横坐标向量，Y是纵坐标向量，

%以f'(x\_i)=y'\_i(i = 1,2,...,n+1)为元素的向量Y1;

%x是以向量形式输入的m个插值点，M在[a,b]上满足｜f~(2n+2)(x)｜≤M

%注：f~(2n+2)(x)表示f(x)的2n+2阶导数

%输出的量：向量y是向量x处的插值，误差限R,2n+1阶埃尔米特插值多项式H\_k及其系数向量

%H\_c,误差公式wcgs及其系数向量Cw.

function[y,R,Hc,Hk,wcgs,Cw] = hermite(X,Y,Y1,x,M)

n = length(X);

m = length(x);

for t = 1 : m

z = x(t);

H = 0;

q = 1;

c1 = 1;

for k = 1 : n

s = 0;

V = 1;

for i = 1 : n

if k ~= i

s = s + (1/(X(k)-X(i)));

V = conv(V,poly(X(i)))/(X(k)-X(i));

end

end

h = poly(X(k));

g = ([0 1]-2 \* h \* s);%注意不要写成1-2\*h\*s，因为是多项式减法，低阶多项式前面必须用零填补，书上的错误，浪费我一晚上时间

G = g \* Y(k) + h \* Y1(k);

H = H + conv(G,conv(V,V));%hermite插值多项式

b = poly(X(k));

b2 = conv(b,b);

q = conv(q,b2);

end

Hc = H;

Hk = poly2sym(H);

Q = poly2sym(q);

for i = 1 : 2\*n

c1 = c1 \* i;

end

wcgs = M \* Q / c1;

Cw = M \* q / c1;

y(t) = polyval(Hc,x(t));

R(t) = polyval(Cw,x(t));

end

**NEWTON**

%保存文件名为New\_Int.m

%Newton基本插值公式

%x为向量，全部的插值节点

%y为向量，差值节点处的函数值

%xi为标量，是自变量

%yi为xi出的函数估计值

function yi=New\_Int(x,y,xi)

n=length(x);

m=length(y);

if n~=m

error('The lengths of X ang Y must be equal!');

return;

end

%计算均差表Y

Y=zeros(n);

Y(:,1)=y'; %将节点的函数值赋给Y的第一列

for k=1:n-1

for i=1:n-k

if abs(x(i+k)-x(i))<eps %避免相邻两个数相等

error('the DATA is error!');

return;

end

Y(i,k+1)=(Y(i+1,k)-Y(i,k))/(x(i+k)-x(i)) %构造差商表

end

end

%计算牛顿插值公式

yi=0;

for i=1:n

z=1;

for k=1:i-1

z=z\*(xi-x(k)); %得到因子（x-x(k))

end

yi=yi+Y(1,i)\*z

end

% New\_Int([0.40 0.55 0.65 0.80 0.90],[0.41075 0.57815 0.69675 0.88811

% 1.02652],0.596)