**龙格现象**





从以上结果可以看到，用三次样条插值和线性分段插值，不会出现多项式插值是出现的Runge现象，插值效果明显提高。

进一步说，为了提高插值精度，用三次样条插值和线性分段插值是可以增加插值节点的办法来满足要求，而用多项式插值函数时，节点数的增加必然会使多项式的次数增加，这样会引起数值不稳定，所以说这两种插值要比多项式插值好的多。而且在给定节点数的条件下，三次样条插值的精度要优于线性分段插值，曲线的光滑性也要好一些。

程序：龙贝格积分：

**%作者：QQ2842305604 哈工大资源分享站**

syms f x;

f=input('请输入积分函数f=');

A=input('请输入积分区间[a,b]=');

e=input('请输入误差限e=');

x=A(1);

fa=eval(f);

x=A(2);

fb=eval(f);

T(1)=(A(2)-A(1))\*(fa+fb)/2;

m=1;

x=(A(1)+A(2));

t=T(1)/2+(A(2)-A(1))\*eval(f)/2;

while abs(t-T(1))>e

t=T(1);

new=0;

for i=1:(2^m)

x=A(1)+(2\*i-1)\*(A(2)-A(1))/(2^(m+1));

new=new+eval(f);

end

T(m+1)=T(m)/2+(A(2)-A(1))\*new/(2^(m+1));

for i=m:(-1):1

L(m,i)=T(i);

end

for p=m:(-1):1

T(p)=((4^(m+1-p))\*T(p+1)-T(p))/(4^(m+1-p)-1);

end

m=m+1;

end

fprintf('数值积分结果为T(1)=%.8f\n',T(1));

fprintf('经过了%d次迭代\n',m);

**结论：**

*Romberg*积分通常要求被积函数在积分区间上没有奇点。如有奇点，且奇点为第一间断点，那么采用例3的方法，还是能够求出来的，否则，必须采用其它的积分方法。当然，*Romberg*积分的收敛速度还是比较快的。

结果分析和讨论：

进行具体的积分时，精度取*R=1e-8*。

1. 求积分。精确解*I= 24999676*。函数表为

1.0e+007 \*

4.70101520000000 3.05022950000000 2.63753307500000

2.49996760000000 2.49996760000000 0

2.49996760000000 0 0

由函数表知*Romberg*积分给出的结果为**2.4999676×107**，与精确没有误差，精度很高。

1. 求积分。

直接按前面方法进行积分，会发现系统报错，出现了0为除数的现象。出现这种情况的原因就是当x=0时，被积函数分母出现了0，如果用一个适当的小数（最好不要小于程序给定的最小误差值，但是不能小于机器的最大精度）来代替，可以避免这个问题。本实验取，可得函数表为：

0.92073548319659 0.93979327500190 0.94451351171417 0.94569085359489 0.94598501993743

0.94614587227034 0.94608692395160 0.94608330088846 0.94608307538495 0

0.94608299406368 0.94608305935092 0.94608306035138 0 0

0.94608306038722 0.94608306036726 0 0 0

0.94608306036718 0 0 0 0

故该函数的积分为**0.94608306036718**，取8位有效数字。

1. 积分函数表为：

0.42073549240395 0.33406972582924 0.31597536075922 0.31168023948094 0.31062036680949 0.31035626065456

0.30518113697100 0.30994390573588 0.31024853238818 0.31026707591900 0.31026822526959 0

0.31026142365354 0.31026884083167 0.31026831215439 0.31026830189296 0 0

0.31026895856465 0.31026830376269 0.31026830173008 0 0 0

0.31026830119484 0.31026830172211 0 0 0 0

0.31026830172262 0 0 0 0 0

故该函数的积分为**0.31026830172262**，取8位有效数字。

**多项式最小二乘法源程序**

**%%作者：QQ2842305604 哈工大资源分享站**

*clear*

%%%给定测量数据点(s,f)

*s=[3 4 5 6 7 8 9];*

*f=[2.01 2.98 3.50 5.02 5.47 6.02 7.05];*

%%%计算给定的数据点的数目

*n=length(f);*

%%%给定需要拟合的数据的最高次多项式的次数

*m=10;*

%%%程序主体

*for k=0:m;*

*g=zeros(1,m+1);*

*for j=0:m;*

*t=0;*

*for i=1:n;*%计算内积(fai(si),fai(si))

*t=t+fai(s(i),j)\*fai(s(i),k);*

*end*

*g(j+1)=t;*

*end*

*A(k+1,:)=g;*%法方程的系数矩阵

*t=0;*

*for i=1:n;*%计算内积(f(si),fai(si))

*t=t+f(i)\*fai(s(i),k);*

*end*

*b(k+1,1)=t;*

*end*

*a=A\b*%求出多项式系数

*x=[s(1):0.01:s(n)]';*

*y=0;*

*for i=0:m;*

*y=y+a(i+1)\*fai(x,i);*

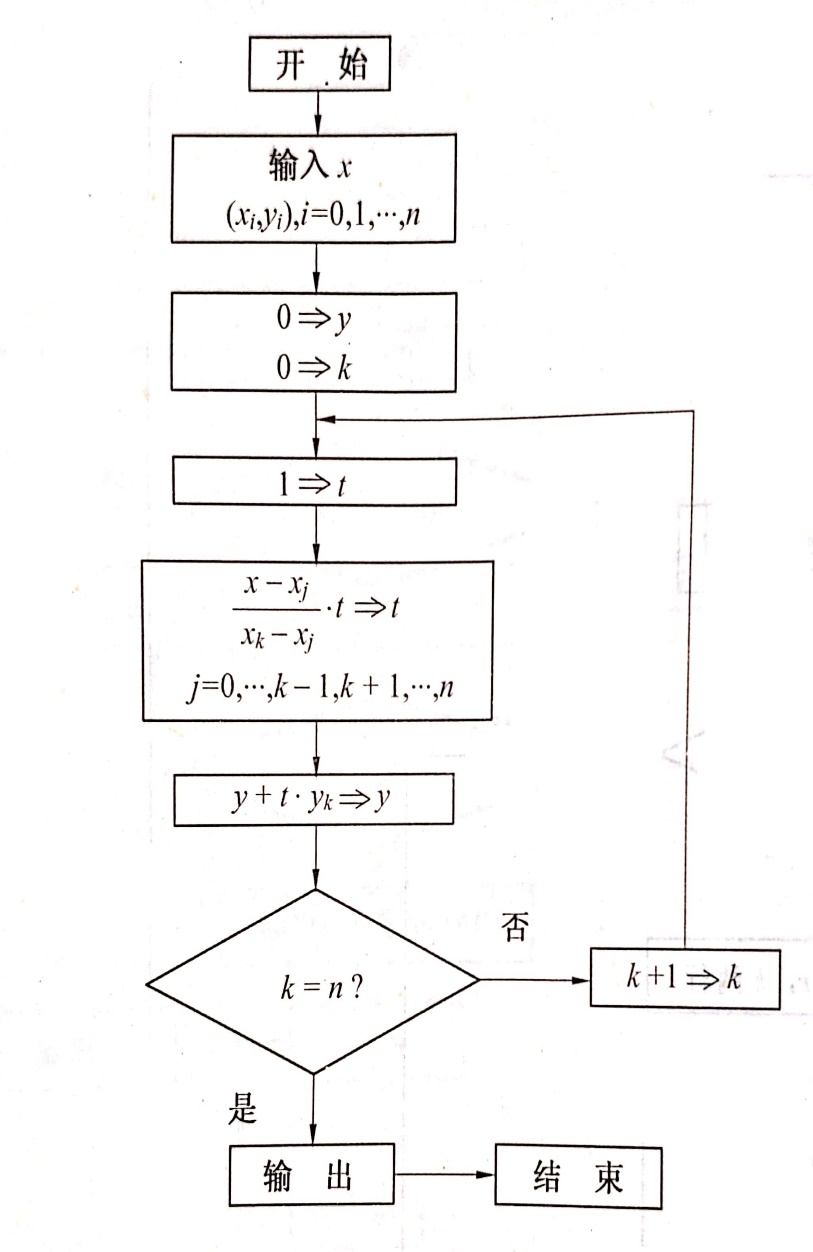
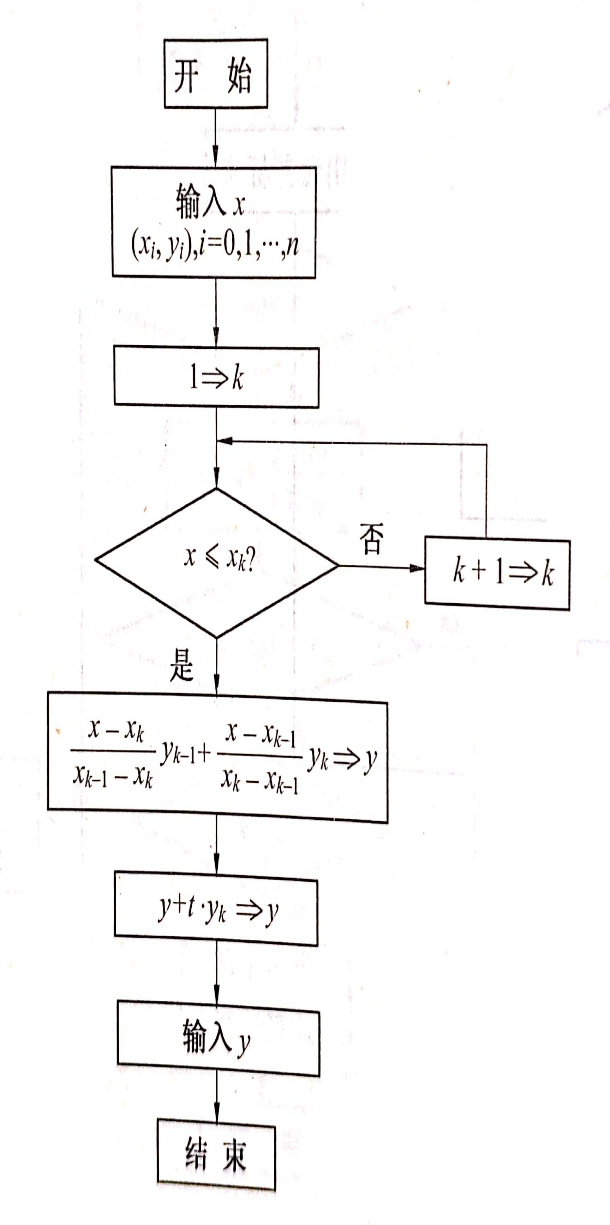
*end*

*plot(x,y)*%作出拟合成的多项式的曲线

*grid on*

*hold on*

*plot(s,f,'rx')* %在上图中标记给定的点



分别采用一次，二次和五次多项式来拟合数据得到相应的拟合多项式为：

y1=-0.38643+0.82750x；

y2=-1.03024+1.06893x-0.02012x2；

y5=-50.75309+51.53527x-19.65947x2+3.66585x3-0.32886x4+0.01137x5；

分别作出它们的曲线图，图中点划线为y1曲线，实线为y2曲线，虚线为y5曲线。’x’为给定的数据点。从图中可以看出并不是多项式次数越高越好，次数高了，曲线越能给定点处和实际吻合，但别的地方就很差了。因此，本例选用一次和两次的多项式拟合应该就可以了。

