**数值分析计算实习题**

第二章

2-1

程序：

clear;clc;

x1=[0.2 0.4 0.6 0.8 1.0];

y1=[0.98 0.92 0.81 0.64 0.38];

n=length(y1);

c=y1(:);

for j=2:n %求差商

for i=n:-1:j

c(i)=(c(i)-c(i-1))/(x1(i)-x1(i-j+1));

end

end

syms x df d;

df(1)=1;d(1)=y1(1);

for i=2:n %求牛顿差值多项式

df(i)=df(i-1)\*(x-x1(i-1));

d(i)=c(i)\*df(i);

end

disp('4次牛顿插值多项式');

P4=vpa(collect((sum(d))),5) %P4即为4次牛顿插值多项式,并保留小数点后5位数

pp=csape(x1,y1, 'variational');%调用三次样条函数

q=pp.coefs;

disp('三次样条函数');

for i=1:4

S=q(i,:)\*[(x-x1(i))^3;(x-x1(i))^2;(x-x1(i));1];

S=vpa(collect(S),5)

end

x2=0.2:0.08:1.08;

dot=[1 2 11 12];

figure

ezplot(P4,[0.2,1.08]);

hold on

y2=fnval(pp,x2);

x=x2(dot);

y3=eval(P4);

y4=fnval(pp,x2(dot));

plot(x2,y2,'r',x2(dot),y3,'b\*',x2(dot),y4,'co');

title('4次牛顿插值及三次样条');

结果如下：

4次牛顿插值多项式

P4 = - 0.52083\*x^4 + 0.83333\*x^3 - 1.1042\*x^2 + 0.19167\*x + 0.98

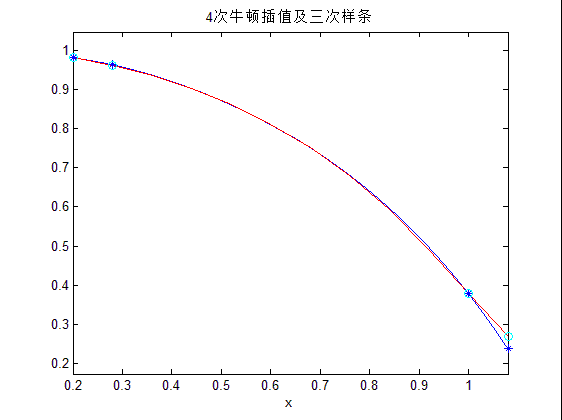
三次样条函数

x∈[0.2,0.4]时， S = - 1.3393\*x^3 + 0.80357\*x^2 - 0.40714\*x + 1.04

x∈[0.4,0.6]时，S = 0.44643\*x^3 - 1.3393\*x^2 + 0.45\*x + 0.92571

x∈[0.6,0.8]时，S = - 1.6964\*x^3 + 2.5179\*x^2 - 1.8643\*x + 1.3886

x∈[0.8,1.0]时，S =2.5893\*x^3 - 7.7679\*x^2 + 6.3643\*x - 0.80571

输出图如下

4次牛顿插值曲线

三次样条插值曲线

2-3（1）

程序：

clear;

clc;

x1=[0 1 4 9 16 25 36 49 64];

y1=[0 1 2 3 4 5 6 7 8];%插值点

n=length(y1);

a=ones(n,2);

a(:,2)=-x1';

c=1;

for i=1:n

c=conv(c,a(i,:));

end

q=zeros(n,n);

r=zeros(n,n+1);

for i=1:n

[q(i,:),r(i,:)]=deconv(c,a(i,:));%wn+1/(x-xk)

end

Dw=zeros(1,n);

for i=1:n

Dw(i)=y1(i)/polyval(q(i,:),x1(i));%系数

end

p=Dw\*q;

syms x L8;

for i=1:n

L8(i)=p(n-i+1)\*x^(i-1);

end

disp('8次拉格朗日插值');

L8=vpa(collect((sum(L8))),5)

xi=0:64;

yi=polyval(p,xi);

figure

plot(xi,yi,x1,y1,'r\*');

hold on

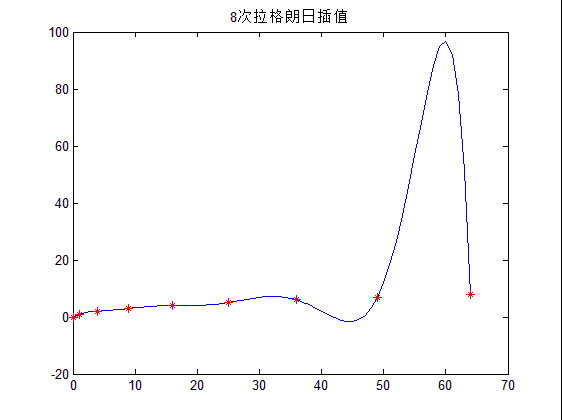
title('8次拉格朗日插值');

结果如下：

8次拉格朗日插值

L8 =- 3.2806e-10\*x^8 + 6.7127e-8\*x^7 - 5.4292e-6\*x^6 + 0.00022297\*x^5 - 0.0049807\*x^4 + 0.060429\*x^3 - 0.38141\*x^2 + 1.3257\*x

输出图如下：



第五章

4-1（3）

程序：

clc;

clear;

y= @(x) sqrt(x).\*log(x);

a=0;b=1;tol=1e-4;

p=quad(y,a,b,tol);

fprintf('采用自适应辛普森积分结果为： %d \n', p);

结果如下：

采用自适应辛普森积分结果为： -4.439756e-01

第九章

9-1

(a)程序：

clc;

clear;

a=1;b=2;%定义域

h=0.05;%步长

n=(b-a)/h;

y0=1;%初值

f= @(x,y) 1/x^2-y/x;%微分函数

Xn=linspace(a,b,n+1);%将定义域分为n等份

Yn=zeros(1,n);%结果矩阵

Yn(1)=y0;%赋初值

%以下根据改进欧拉公式求解

for i=1:n

xn=Xn(i);

xnn=Xn(i+1);

yn=Yn(i);

yp=yn+h\*f(xn,yn);

yc=yn+h\*f(xnn,yp);

yn=(yp+yc)/2;

Yn(i+1)=yn;

end

Xn=Yn;

%以下根据经典四阶R-K法公式求解

for i=1:n

xn=Xn(i);

yn=Yn(i);

k1=f(xn,yn);

k2=f(xn+h/2,yn+h/2\*k1);

k3=f(xn+h/2,yn+h/2\*k2);

k4=f(xn+h,yn+h\*k3);

yn=yn+h/6\*(k1+2\*k2+2\*k3+k4);

Yn(i+1)=yn;

end

disp(' 改进欧拉法 四阶经典R-K法');

disp([Xn' Yn'])

结果如下：

改进欧拉法 四阶经典R-K法

1 1

0.99887 0.99885

0.99577 0.9978

0.99114 0.99694

0.98532 0.99634

0.97857 0.99603

0.97111 0.99606

0.96311 0.99645

0.9547 0.99723

0.94598 0.99841

0.93705 1

0.92798 1.002

0.91883 1.0044

0.90964 1.0073

0.90045 1.0106

0.89129 1.0143

0.88218 1.0184

0.87315 1.0229

0.86421 1.0278

0.85538 1.0331

0.84665 1.0388

（b）程序：

clc;

clear;

a=0;b=1;%定义域

H=[0.1 0.025 0.01];%步长

y0=1/3;%初值

f= @(x,y) -50\*y+50\*x^2+2\*x;%微分函数

xi=linspace(a,b,11);

Y=1/3\*exp(-50\*xi)+xi.^2;%准确解

Ym=zeros(1,11);

for j=1:3

h=H(j);

n=(b-a)/h;

Xn=linspace(a,b,n+1);%将定义域分为n等份

Yn=zeros(1,n);%结果矩阵

Yn(1)=y0;%赋初值

for i=1:n

xn=Xn(i);

yn=Yn(i);

k1=f(xn,yn);

k2=f(xn+h/2,yn+h/2\*k1);

k3=f(xn+h/2,yn+h/2\*k2);

k4=f(xn+h,yn+h\*k3);

yn=yn+h/6\*(k1+2\*k2+2\*k3+k4);

Yn(i+1)=yn;

end

for k=1:11

m=0.1/h;

Ym(k)=Yn(1+(k-1)\*m);

end

delta=Ym-Y;

fprintf('步长为： %d \n', h);

disp(' 四阶经典R-K法 准确解 误差');

disp([Ym' Y' delta'])

end

结果如下：

步长为： 1.000000e-01

四阶经典R-K法 准确解 误差

0.33333 0.33333 0

4.6055 0.012246 4.5932

63.062 0.040015 63.022

864.05 0.09 863.96

11844 0.16 11843

1.6235e+05 0.25 1.6235e+05

2.2256e+06 0.36 2.2256e+06

3.0509e+07 0.49 3.0509e+07

4.1823e+08 0.64 4.1823e+08

5.7333e+09 0.81 5.7333e+09

7.8594e+10 1 7.8594e+10

步长为： 2.500000e-02

四阶经典R-K法 准确解 误差

0.33333 0.33333 0

0.013015 0.012246 0.00076894

0.040063 0.040015 4.82e-05

0.090037 0.09 3.6857e-05

0.16004 0.16 3.6723e-05

0.25004 0.25 3.6722e-05

0.36004 0.36 3.6722e-05

0.49004 0.49 3.6722e-05

0.64004 0.64 3.6722e-05

0.81004 0.81 3.6722e-05

1 1 3.6722e-05

步长为： 1.000000e-02

四阶经典R-K法 准确解 误差

0.33333 0.33333 0

0.012256 0.012246 9.5673e-06

0.040016 0.040015 7.8252e-07

0.090001 0.09 6.6347e-07

0.16 0.16 6.6226e-07

0.25 0.25 6.6225e-07

0.36 0.36 6.6225e-07

0.49 0.49 6.6225e-07

0.64 0.64 6.6225e-07

0.81 0.81 6.6225e-07

1 1 6.6225e-07

由结果可知，步长越小，结果越精确。

9-3

主程序：

clc;

clear;

options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4 1e-4]);

[X,Y] = ode45(@rigid,[0 0.004],[1 1 0],options);

subplot(1,2,1);

plot(X,Y(:,1),'-',X,Y(:,2),'-.',X,Y(:,3),'.');

title('四阶R-K方法');legend('y1', 'y2','y3');

[X,YY] = ode23t(@rigid,[0 0.004],[1 1 0],options);

subplot(1,2,2);

plot(X,YY(:,1),'-',X,YY(:,2),'-.',X,YY(:,3),'.');

title('梯形法');legend('y1', 'y2','y3');

rigid函数

function dy = rigid (~,y)

dy = zeros(3,1); % a column vector

dy(1) = -0.013\*y(1)-1000\*y(1)\*y(2);

dy(2) = -2500\*y(2) \* y(3);

dy(3) =-0.013\*y(1)-1000\*y(1)\*y(2)-2500\*y(2) \* y(3);

结果如图：

