**实验三 常微分方程的差分方法实验**

学号：20002462 姓名：刘子言

**一. 实验目的**

（1）深入理解常微分方程的差分方法的原理，学会用差分方法解决某些实际的常微分方程问题，比较这些方法解题的不同之处。

（2）熟悉Matlab编程环境，利用Matlab实现具体的常微分方程。

**二. 实验要求**

用Matlab软件实现欧拉方法、改进的欧拉方法、龙格-库塔方法和亚当姆斯方法，并用实例在计算机上计算。

1. **实验内容**

**3-1：用欧拉方法、改进的欧拉方法、四阶龙格-库塔方法求解初值问题：**

**在区间[0,1]上取*h*=0.1时的数值解，并与精确解进行比较。**

**1、设计思想**

1. **欧拉方法的设计思想**

对于一阶常微分方程的初值问题，设在区间的左端点列出方程：



并用差商代替其中的导数项，则有近似关系式：



若用的近似值代入上式右端，并记所得结果为，由此设计得到Euler格式：



1. **改进的欧拉方法的设计思想**

先用Euler格式（显式格式）求得一个初步近似值作为预报值，再用梯形格式（隐式格式）迭代得出较高精度的校正值，这样可以得出改进的Euler格式，再将其改写为平均化形式，便于编程实现：







1. **四阶龙格-库塔方法的设计思想**

设法在区间上预报四个点的斜率，然后将它们加权平均作为平均斜率，即可设计出更高精度的四阶Runge-Kutta格式：











其中直接求出，然后依次预报出，和。可以看出，这一格式每一步需4次计算f的函数值y。

1. **对应程序**
2. **常微分方程的函数代码：**

function z = liuziyan\_3\_1\_dEquation( x,y )

z = 2\*x./(3\*y.^2);

end

1. **求准确解的函数代码：**

function y = liuziyan\_3\_1\_solvef( x )

y = power(1+x.^2,1/3);

end

1. **欧拉方法：**

function E = liuziyan\_3\_1\_1\_Euler( f,a,b,N,ya )

% f——微分方程右端函数；N——区间等分的个数；

% a,b——自变量的取值区间[a,b]的端点；ya——表初值y(a)

% E=[x',y']是自变量X和解Y所组成的矩阵

h = (b-a)/N;

y = zeros(1,N+1);

x = zeros(1,N+1);

y(1) = ya;

x = a:h:b;

for i = 1:N

y(i+1) = y(i) + h\*feval(f,x(i),y(i));

end

E = [x',y'];

1. **改进的欧拉方法：**

function E = liuziyan\_3\_1\_2\_MendEuler( f,a,b,N,ya )

% f——微分方程右端函数；N——区间等分的个数；

% a,b——自变量的取值区间[a,b]的端点；ya——表初值y(a)

% E=[x',y']是自变量X和解Y所组成的矩阵

h = (b-a)/N;

y = zeros(1,N+1);

x = zeros(1,N+1);

y(1) = ya;

x = a:h:b;

for i = 1:N

y1 = y(i) + h\*feval(f,x(i),y(i));

y2 = y(i) + h\*feval(f,x(i+1),y1);

y(i+1) = (y1+y2)/2;

end

E = [x',y'];

1. **四阶龙格-库塔方法：**

function R = liuziyan\_3\_1\_3\_Rungkuta4( f,a,b,N,ya )

% R = [x',y']是自变量X和解Y所组成的矩阵

h = (b-a)/N;

y = zeros(1,N+1);

x = zeros(1,N+1);

y(1) = ya;

x = a:h:b;

for i = 1:N

k1 = feval(f,x(i),y(i));

k2 = feval(f,x(i)+h/2,y(i)+(h/2)\*k1);

k3 = feval(f,x(i)+h/2,y(i)+(h/2)\*k2);

k4 = feval(f,x(i)+h, y(i)+ h \*k3);

y(i+1) = y(i) + (h/6) \* (k1 + 2\*k2 + 2\*k3 + k4);

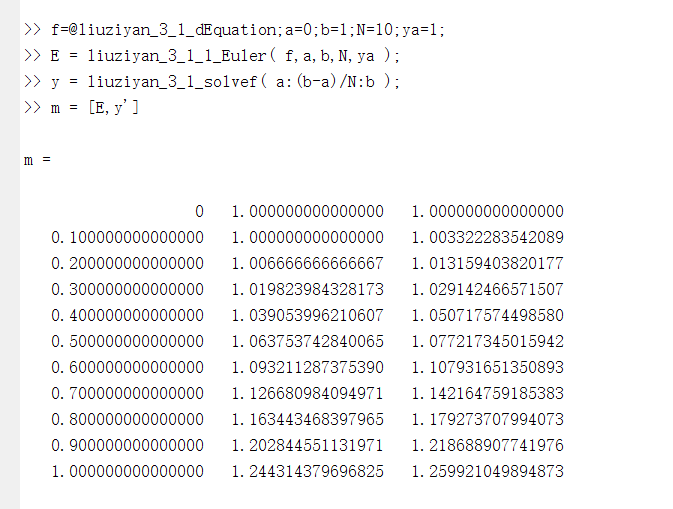
end

R = [x',y'];

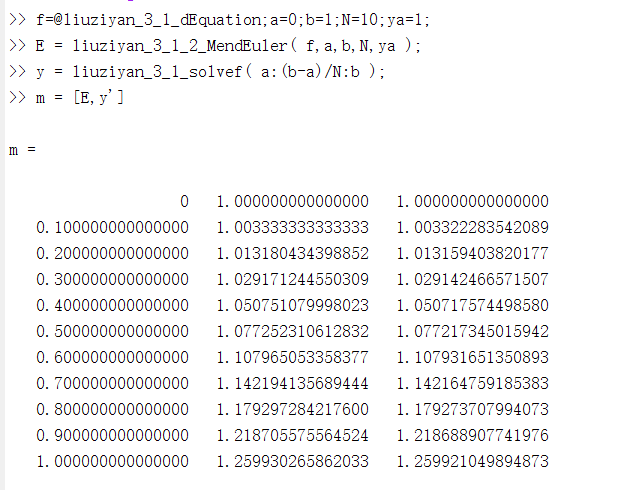
**3、实验结果**

实验3-1的运行结果如下：

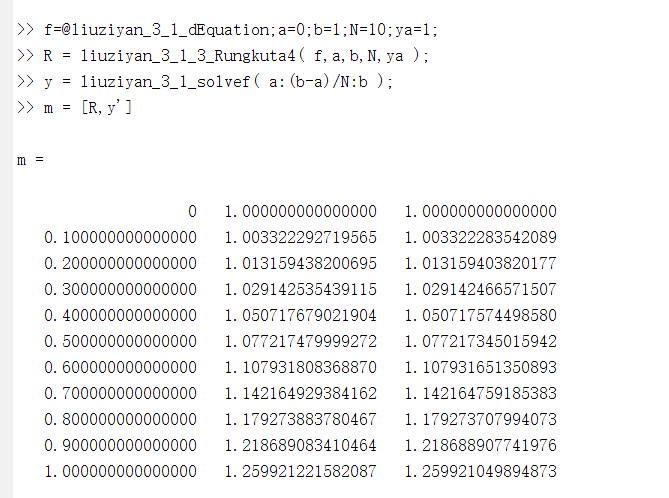
1. **欧拉方法：**



1. **改进的欧拉方法：**



1. **四阶龙格-库塔方法：**



由计算结果可知，改进的Euler方法比Euler方法计算结果更准确一些，但精度依旧有待提高；四阶的Runge-Kutta方法与前两种方法相比，精度要更高一些。

**3-2：分别用四阶亚当姆斯方法、改进的四阶亚当姆斯预估校正系统求解题3-1中的初值问题。提示：可用四阶龙格-库塔方法求出开头三步的值。**

1. **设计思想**
2. **四阶亚当姆斯方法的设计思想**

将四阶显式和隐式的Adams格式匹配在一起，即可生成四阶Adams预报校正系统：









由于这种方法是4步法，在计算时要用到前三步的信息，因此它不能自行启动，实际计算时，可利用四阶Runge-Kutta方法为其提供开头3步的值。

1. **改进的四阶亚当姆斯方法的设计思想**

利用误差估计式修改四阶Adams预报校正系统，即可导出下列改进的四阶Adams预报校正系统：













1. **对应程序**
2. **对应的常微分方程和求准确解的函数代码与实验3-1的相同：**

function z = liuziyan\_3\_1\_dEquation( x,y )

z = 2\*x./(3\*y.^2);

end

function y = liuziyan\_3\_1\_solvef( x )

y = power(1+x.^2,1/3);

end

1. **四阶亚当姆斯方法：**

function A1 = liuziyan\_3\_2\_1\_Adams4PC( f,a,b,N,ya )

% A1 = [x',y']是自变量X和解Y所组成的矩阵

if N < 4

return;

end

h = (b-a)/N;

y = zeros(1,N+1);

x = zeros(1,N+1);

y(1) = ya;

x = a:h:b;

F = zeros(1,4);

for i = 1:N

if i < 4 %用四阶Runge-Kutta法求前3个初始解

k1 = feval(f,x(i),y(i));

k2 = feval(f,x(i)+h/2,y(i)+(h/2)\*k1);

k3 = feval(f,x(i)+h/2,y(i)+(h/2)\*k2);

k4 = feval(f,x(i)+h, y(i)+ h \*k3);

y(i+1) = y(i) + (h/6)\*(k1+2\*k2+2\*k3+k4);

else

F = feval(f, x(i-3:i), y(i-3:i));

py = y(i) + (h/24)\*( F\*[-9, 37, -59, 55]'); %预报

p = feval(f, x(i+1), py);

F = [F(2) F(3) F(4) p];

y(i+1) = y(i) + (h/24)\*( F\*[1, -5, 19, 9]'); %校正

end

end

A1 = [x',y'];

1. **改进的四阶亚当姆斯方法：**

function A2 = liuziyan\_3\_2\_2\_CAdams4PC( f,a,b,N,ya )

% A2 = [x',y']是自变量X和解Y所组成的矩阵

if N < 4

return;

end

h = (b-a)/N;

y = zeros(1,N+1);

x = zeros(1,N+1);

y(1) = ya;

x = a:h:b;

F = zeros(1,4);

for i = 1:N

if i < 4 %用四阶Runge-Kutta法求前3个初始解

k1 = feval(f,x(i),y(i));

k2 = feval(f,x(i)+h/2,y(i)+(h/2)\*k1);

k3 = feval(f,x(i)+h/2,y(i)+(h/2)\*k2);

k4 = feval(f,x(i)+h, y(i)+ h \*k3);

y(i+1) = y(i) + (h/6)\*(k1+2\*k2+2\*k3+k4);

elseif i==4

F = feval(f, x(i-3:i), y(i-3:i));

py = y(i) + (h/24)\*( F\*[-9, 37, -59, 55]'); %预报

p = feval(f, x(i+1), py);

F = [ F(2) F(3) F(4) p ];

y(i+1) = y(i) + (h/24)\*( F\*[1, -5, 19, 9]'); %校正

p = py;

c = y(i+1);

else

F = feval(f, x(i-3:i), y(i-3:i));

py = y(i) + (h/24)\*( F\*[-9, 37, -59, 55]'); %预报

my = py - 251\*(p-c)/270; %改进

m = feval(f, x(i+1), my);

F = [ F(2) F(3) F(4) m ];

cy = y(i) + (h/24)\*( F\*[1, -5, 19, 9]'); %校正

y(i+1) = cy + 19\*(py-cy)/270; %改进

p = py;

c = cy;

end

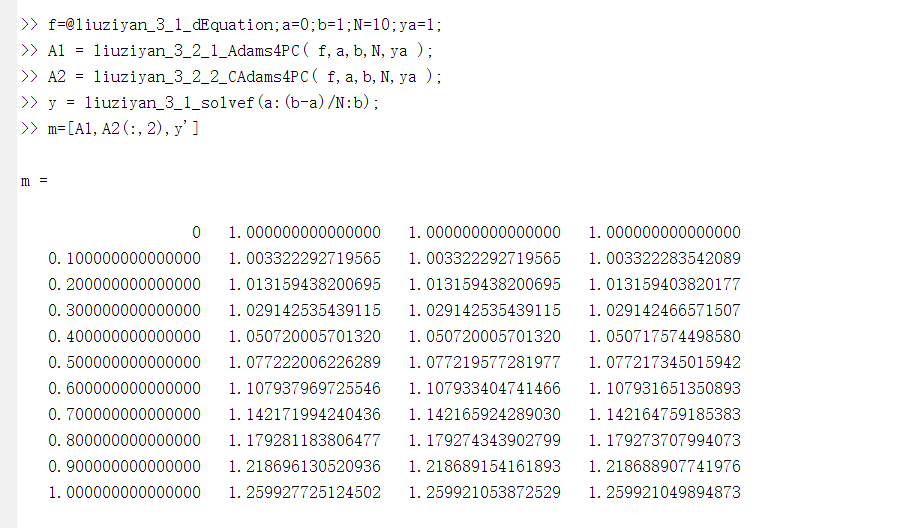
end

A2 = [x',y'];

**3、实验结果**

实验3-2的运行结果如下：

将四阶亚当姆斯方法、改进的四阶亚当姆斯方法结果与准确值放在一起显示：



以上4列从左到右依次为：离散节点值，四阶Adams预报校正系统所求解，改进的四阶Adams预报校正系统所求解，准确解。通过计算结果的比较分析可知，改进的四阶Adams预报校正系统效果比四阶Adams预报校正系统效果要好。

1. **实验体会**

对实验过程进行分析总结，对比求解常微分方程的不同方法，指出每种算法的设计要点及应注意的事项，以及自己通过实验所获得的对常微分方程的差分方法的理解。

答：

1. Euler法是一种显式算法，其计算量小，但精度很低；
2. 改进的Euler法也是一种显式格式，是将Euler格式与梯形格式相结合建立的预报校正系统，保证了计算量不会过大的同时还明显地改善了精度；
3. Runge-Kutta方法阶数不高，但是可以达到较高的精度，它在设计时用f(x,y)在某点上的值的线性组合来计算，避免计算函数f(x,y)的偏导数，从而提高了精度；
4. Adams预报校正系统相比同阶的单步法如Runge-Kutta法计算量大大减小，同时也能获得不错的精度；
5. 对于显式法和隐式法，一般来说同阶的隐式法比显式法精确，而且数值稳定性也较好，但在隐式公式中需要用迭代法求解，计算量较大。所以在实际计算中，很少单独使用显式公式或者隐式公式，而是将它们联合使用，先用显式公式求预测值，再用隐式公式对预测值进行校正，从而得出近似值。