

# 天津理工大学 实验报告

学院（系）名称：计算机科学与工程学院

姓名	王帆	学号	20152180	专业	计算机科学与技术
班级	15 计算机 1 班	实验项目	实验四 常微分方程的数值解法		
课程名称		数值计算方法		课程代码	0665026
实验时间		2017 年 5 月 23 日 第 3 - 4 节		实验地点	7-215
批改意见				成绩	
教师签字:					

## 一、 实验目的

掌握龙格-库塔法求解一阶常微分方程和一阶常微分方程组的初值问题，并能够熟练的将上述算法编程实现。

## 二、 实验环境

- 硬件环境：IBM-PC 或兼容机
- 软件环境：Windows 操作系统，VC6.0
- 编程语言：C 或 C++

## 三、 实验内容

1. 用欧拉法、改进欧拉法和四阶龙格-库塔法求解一阶常微分方程初值问题

$$\begin{cases} y' = x - y & 0 \leq x \leq 1 \\ y(0) = 0 \end{cases}$$

的解，计算中步长取为 0.1。

要求：

- (1) 步长  $h$  的值从键盘终端输入；
- (2) 将三种方法计算的每个节点的函数值打印输出，并和精确值进行比较（精确解对应的解析表达式为  $y(x)=x+e^{-x}-1$ ），根据计算结果分析这三种方法求解结果的精度差别。

2. 用四阶龙格-库塔法求解一阶常微分方程组初值问题

$$\begin{cases} y_1' = \frac{1}{y_2 - x} \\ y_2' = -\frac{1}{y_1} + 1 \\ y_1(0) = 1 \\ y_2(0) = 1 \end{cases}$$

$x \in [0, 2], h=0.1$ 。

要求：

- (1) 步长  $h$  的值从键盘终端输入；
- (2) 将每一步的计算结果打印输出。

#### 四、 实验要求

1. 每一个实验内容要求自己独立完成，不允许抄袭别人，否则按不及格处理；
2. 按照实验要求，根据自己的程序编写情况绘制相应的算法框图或描述算法步骤；
3. 按照实验内容和相应的要求书写实验报告；
4. 在实验过程部分，要求根据实验内容和要求书写每一个实验相应的算法步骤或框图、运行过程和运行结果的截图、运行结果分析、以及程序源代码。每一个实验要求书写下述内容：

- (1) 算法步骤描述或算法框图
- (2) 程序源代码
- (3) 运行结果（要求截图）
- (4) 运行结果分析
5. 在规定的时间内上交实验报告。

#### 五、 实验过程、

1. 用欧拉法、改进欧拉法和四阶龙格-库塔法求解一阶常微分方程初值问题

代码实现：

```
//三种方法求解微分方程
#include <iostream>
#include <cmath>
#include <cstdio>
#define MAX_NUM 100
using namespace std;
//全局变量
int num=0;
double h;
double f[MAX_NUM];
//原型声明
void print(int x);
void init();
```

```

void F(double h);
void OL(double h);
void NOL(double h);
void _4LK(double h);
//实现
int main(){
    init();
    OL(h);
    NOL(h);
    _4LK(h);
    return 0;
}
void init(){
    cout<<"请输入步长:";
    cin>>h;
    num=1/h;
    F(h);
}
void print(int x){
    switch(x){
        case 1:{
            cout<<"欧拉法:"<<endl;
            cout<<"xn\t  yn\t      精确解"<<endl;
            break;
        }
        case 2:{
            cout<<"改进欧拉法:"<<endl;
            cout<<"xn\t  yn\t      精确解"<<endl;
            break;
        }
        case 3:{
            cout<<"四阶龙格-库塔法:"<<endl;
            cout<<"xn\t  yn\t      精确解"<<endl;
            break;
        }
        default: break;
    }
    return;
}
void F(double h){
    double x;
    for(int i=1;i<num+1;i++){
        x=i*h;
        f[i]=x+exp(-x)-1;
    }
}

```

```

}
void OL(double h){
    print(1);
    double xn[num+1],yn[num+1];
    xn[0]=0.0;
    yn[0]=0.0;
    for(int n=1;n<num+1;n++){
        yn[n]=yn[n-1]+h*(xn[n-1]-yn[n-1]);
        xn[n]=n*h;
    }
    for(int i=1;i<num+1;i++){
        printf("%.1f      %.6f      %.6f\n",xn[i],yn[i],f[i]);
    }
}
void NOL(double h){
    print(2);
    double xn[num+1],yn0[num+1],yn1[num+1];
    xn[0]=0.0;
    yn0[0]=0.0;
    yn1[0]=0.0;
    for(int n=1;n<num+1;n++){
        yn0[n]=yn1[n-1]+h*(xn[n-1]-yn0[n-1]);
        xn[n]=n*h;
        yn1[n]=yn1[n-1]+h/2*((xn[n-1]-yn1[n-1])+xn[n]-yn0[n]);
    }
    for(int i=1;i<num+1;i++){
        printf("%.1f      %.6f      %.6f\n",xn[i],yn1[i],f[i]);
    }
}
void _4LK(double h){
    print(3);
    double xn[num+1],yn0[num+1],yn1[num+1];
    double k1,k2,k3,k4;
    xn[0]=0.0;
    yn0[0]=0.0;
    yn1[0]=0.0;
    for(int n=1;n<num+1;n++){
        xn[n]=n*h;
        k1=xn[n-1]-yn1[n-1];
        k2=(xn[n-1]+h/2)-(yn1[n-1]+h/2*k1);
        k3=(xn[n-1]+h/2)-(yn1[n-1]+h/2*k2);
        k4=xn[n]-(yn1[n-1]+h*k3);
        yn1[n]=yn1[n-1]+h/6*(k1+2*k2+2*k3+k4);
    }
    for(int i=1;i<num+1;i++){

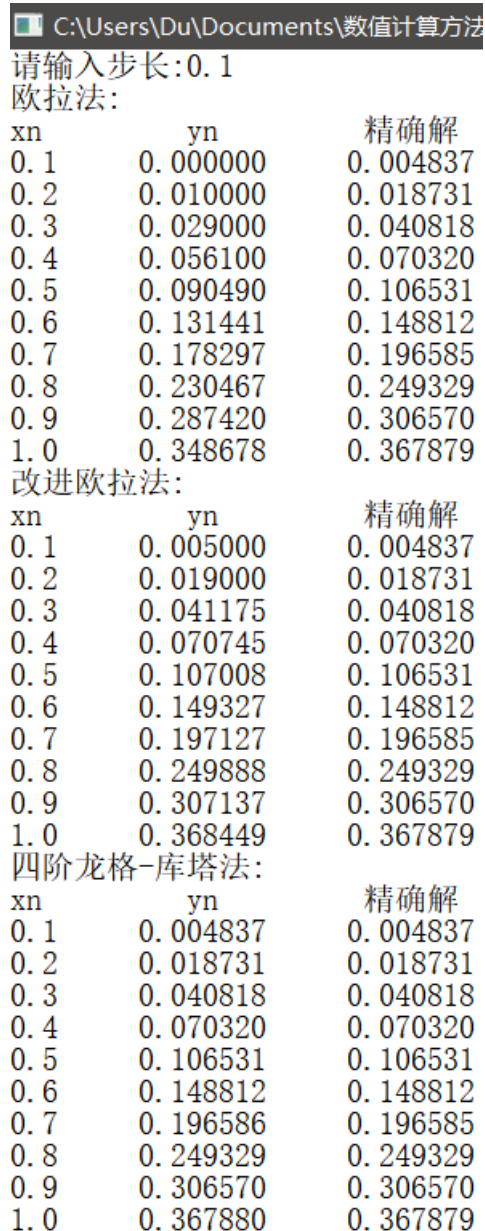
```

```

        printf("%.1f    %.6f    %.6f\n",xn[i],yn1[i],f[i]);
    }
}

```

运行结果:



```

C:\Users\Du\Documents\数值计算方法
请输入步长:0.1
欧拉法:
xn      yn      精确解
0.1      0.000000    0.004837
0.2      0.010000    0.018731
0.3      0.029000    0.040818
0.4      0.056100    0.070320
0.5      0.090490    0.106531
0.6      0.131441    0.148812
0.7      0.178297    0.196585
0.8      0.230467    0.249329
0.9      0.287420    0.306570
1.0      0.348678    0.367879
改进欧拉法:
xn      yn      精确解
0.1      0.005000    0.004837
0.2      0.019000    0.018731
0.3      0.041175    0.040818
0.4      0.070745    0.070320
0.5      0.107008    0.106531
0.6      0.149327    0.148812
0.7      0.197127    0.196585
0.8      0.249888    0.249329
0.9      0.307137    0.306570
1.0      0.368449    0.367879
四阶龙格-库塔法:
xn      yn      精确解
0.1      0.004837    0.004837
0.2      0.018731    0.018731
0.3      0.040818    0.040818
0.4      0.070320    0.070320
0.5      0.106531    0.106531
0.6      0.148812    0.148812
0.7      0.196586    0.196585
0.8      0.249329    0.249329
0.9      0.306570    0.306570
1.0      0.367880    0.367879

```

图 1 欧拉法、改进欧拉法、四阶龙格-库塔法求解运行结果

运行分析:

根据求解结果可得: 四阶龙格-库塔法的精度最高, 欧拉法最低。

## 2. 用四阶龙格-库塔法求解一阶常微分方程组初值问题

代码实现:

```

//四阶-龙格库塔法求解微分方程组
#include <iostream>

```

```

#include <cstdio>
using namespace std;
//全局变量
int num;
double h;
double k11,k12,k13,k14,k21,k22,k23,k24;
//原型声明
void init();
double f(double x,double y);
double f(double x);
void _4LK(double h, int num);
//实现
int main(){
    init();
    _4LK(h,num);
    return 0;
}
double f(double x,double y){
    return 1/(y-x);
}
double f(double x){
    return (-1/x)+1;
}
void init(){
    cout<<"请输入步长:";
    cin>>h;
    num=2.0/h;
}
void _4LK(double h,int num){
    double x[num+1],y1[num+1],y2[num+1];
    y1[0]=1.0;y2[0]=1.0;
    for(int n=1;n<num+1;n++){
        x[n]=n*h;
        k1=f(x[n-1],y2[n-1]);
        l1=f(y1[n-1]);

        k2=f(x[n-1]+h/2,y2[n-1]+h/2*l1);
        l2=f(y1[n-1]+h/2*k1);

        k3=f(x[n-1]+h/2,y2[n-1]+h/2*l2);
        l3=f(y1[n-1]+h/2*k2);

        k4=f(x[n],y2[n-1]+h*l3);
        l4=f(y1[n-1]+h*k3);
    }
}

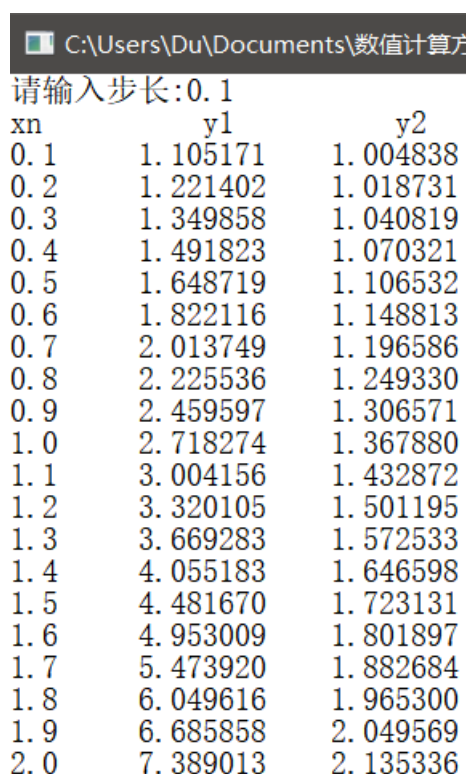
```

```

        y1[n]=y1[n-1]+h/6*(k1+2*k2+2*k3+k4);
        y2[n]=y2[n-1]+h/6*(l1+2*l2+2*l3+l4);
    }
    cout<<"xn\t    y1\t\tty2"<<endl;
    for(int i=1;i<num+1;i++){
        printf("%.1f    %.6f    %.6f\n",x[i],y1[i],y2[i]);
    }
}

```

运行结果：



xn	y1	y2
0.1	1.105171	1.004838
0.2	1.221402	1.018731
0.3	1.349858	1.040819
0.4	1.491823	1.070321
0.5	1.648719	1.106532
0.6	1.822116	1.148813
0.7	2.013749	1.196586
0.8	2.225536	1.249330
0.9	2.459597	1.306571
1.0	2.718274	1.367880
1.1	3.004156	1.432872
1.2	3.320105	1.501195
1.3	3.669283	1.572533
1.4	4.055183	1.646598
1.5	4.481670	1.723131
1.6	4.953009	1.801897
1.7	5.473920	1.882684
1.8	6.049616	1.965300
1.9	6.685858	2.049569
2.0	7.389013	2.135336

图 2 四阶龙格-库塔法求解运行结果

运行分析：

本题为微分方程组的求解问题，解决方案是将初值问题转化为一个一阶常微分方程组。因为两个函数（重载得到）需要相互调用，则使得计算结果更加精确。

## 六、 实验总结及心得体会

通过本次实验，我加深了对于数值方法求解微分方程的理解，其中包括对欧拉法、改进欧拉法以及龙格库塔法等数值方法。数值方法求解微分方程是一个重要的解决实际问题的方法，我在本次实验中将这三种方法进行编程实现也使我对他们能够灵活运用于实际问题中。