Runge-kutta法上机报告

计试81 白思雨 2186123935

一、算法原理

利用泰勒展开可以导出龙格-库塔法。m级龙格-库塔法的一般形式为：

$$\begin{cases} y_{i+1}=y_i+\lambda_1\mathcal{K}_1+\lambda_2\mathcal{K}_2+\cdots+\lambda_m\mathcal{K}_m \\ \mathcal{K}_1=hf(x_i,y_i) \\ \mathcal{K}_2=hf(x_i+\alpha_2h,y_i+\beta_{21}\mathcal{K}_1) \\ \mathcal{K}_3=hf(x_i+\alpha_3h,y_i+\beta_{31}\mathcal{K}_1+\beta_{32}\mathcal{K}_2) \\ \cdots \\ \mathcal{K}_m=hf(x_i+\alpha_mh,y_i+\beta_{m1}\mathcal{K}_1+\beta_{m2}\mathcal{K}_2+\cdots+\beta_{m,m-1}\mathcal{K}_{m-1}) \end{cases}$$

其中$\lambda_i,\alpha_i,\beta_{j,k}$均为常数，由待定系数法确定，确定的原则则是将局部截断误差$R[y]=y(x_{i+1})-y_{i+1}$在x_i处泰勒展开，适当选取h的系数，使得局部截断误差$R[y]$的阶尽可能高。

$$y(x_i+1)=y(x_i+h)=y(x_i)+y'(x_i)h+\frac{1}{2!}y''(x_i)h^2+\frac{1}{3!}y'''(x_i)h^3+\cdots$$

$$y''(x)=f_x{}'+f'{}_yy'=f_x{}'+f'{}_yf$$

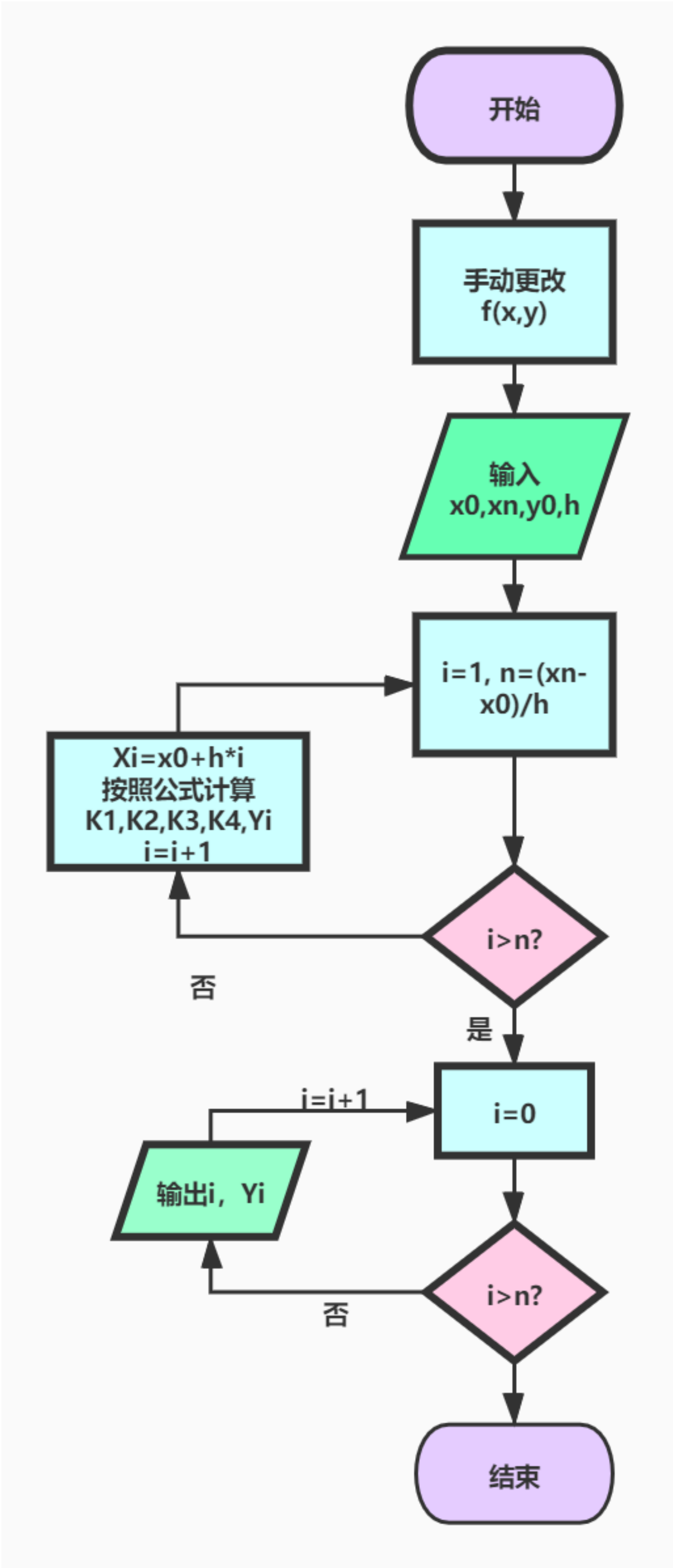
$$y''(x)=f_{xx}{}''+2f_{xy}{}''f+f_{yy}{}''f^2+f_x{}'f_y{}'+f_{yy}{}'^2f$$

$$\cdots$$

经典（标准）4级4阶R－K法：

$$\begin{cases} y_{i+1}=y_i+\dfrac{1}{6}(\mathcal{K}_1+2\mathcal{K}_2+2\mathcal{K}_3+\mathcal{K}_4) \\ \mathcal{K}_1=f(x_i,y_i) \\ \mathcal{K}_2=hf\left(x_i+\dfrac{1}{2}h,y_i+\dfrac{1}{2}\mathcal{K}_1\right) \\ \mathcal{K}_3=hf\left(x_i+\dfrac{1}{2}h,y_i+\dfrac{1}{2}\mathcal{K}_2\right) \\ \mathcal{K}_4=hf(x_i+h,y_i+\mathcal{K}_3) \end{cases}$$

二、程序框图



开始
手动更改
f(x,y)
输入
x0,xn,y0,h
i=1, n=(xn-x0)/h
Xi=x0+h*i
按照公式计算
K1,K2,K3,K4,Yi
i=i+1
i>n?
否
是
i=i+1
i=0
输出i，Yi
i>n?
否
结束

三、程序及使用说明(cpp)

```cpp
#include<iostream>
#include<cmath>
using namespace std;

double func1(double x, double y)
{
    return (-0.9 * y / (1 + 2 * x)); //这是被积函数，若计算其他函数积分则在此更换函数
}

double Y[1000],X[1000];
double x0, xn, yo, h;

int main()
{
    //输入基本条件，x0,xn,yo,h
    cout << "Please input x0, xn, yo and h:";
    cin >> x0 >> xn >> yo >> h;

    //计算分点个数，n 为分点的个数-1;
    int n = (xn - x0) / h;
    Y[0] = yo;
    X[0] = x0;

    //计算 yi(i=0,1...n)，并将其储存到 Y[1000]数组中
    for(int i = 1; i <= n; ++i)
    {
        X[i] = x0 + h * i;
        double K1 = h * func1(X[i - 1], Y[i - 1]);
        double K2 = h * func1(X[i - 1] + 0.5 * h, Y[i - 1] + 0.5 * K1);
        double K3 = h * func1(X[i - 1] + 0.5 * h, Y[i - 1] + 0.5 * K2);
        double K4 = h * func1(X[i - 1] + h, Y[i - 1] + K3);
        Y[i] = Y[i - 1] + (K1 + 2 * K2 + 2 * K3 + K4) / 6;
    }

    //输出 yi(i = 0,1...n)
    for(int i = 0; i <= n; ++i)
    {
        cout << "y" << i << ':' << Y[i] << endl;
    }

    return 0;
}
```

（说明：相关程序说明已在代码中注释写出，对于不同的函数积分需要更改代码中 func1.）

四、算例及计算结果

算例：例题9.1.1：

$$\begin{cases} y' = f(x,y) = \dfrac{-0.9y}{1+2x} \, (0 \leqslant x \leqslant 0.1) \\ y_0 = 1, x_0 = 0 \\ h = 0.02 \end{cases}$$

计算结果：

Microsoft Visual Studio 调试控制台

```
Please input x0,xn,yo and h:
0 0.1 1 0.02
y0:1
y1:0.982506
y2:0.96596
y3:0.950281
y4:0.935393
y5:0.921231

C:\Users\62613\source\repos\Project3\Debug\Project3.exe
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”
按任意键关闭此窗口...
```

与原题结果相符，实验正确。