龙格-库塔方法与亚当姆斯方法相结合求解偏微分方程

姓名：岳宇轩　　学号：19020011038

专业：19 慧与　　序号：14　　指导教师：高云

实验目的

1.实现龙格-库塔方法，并验证它的正确性；

2.先用龙格-库塔方法计算前 3 点，然后实现亚当姆斯预报-校正系统，并验证算法的正确性。

实验步骤

1.实现龙格-库塔方法，并用它启动

2.验证亚当姆斯预报-校正系统

实验原理

1. 四阶经典龙格-库塔方法：

$$
\begin{cases}
y_{n+1} = y_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\
K_1 = f(x_n, y_n) \\
K_2 = f(x_{n+\frac{1}{2}}, y_n + \frac{h}{2}K_1) \\
K_3 = f(x_{n+\frac{1}{2}}, y_n + \frac{h}{2}K_2) \\
K_3 = f(x_{n+1}, y_n + hK_3)
\end{cases}
$$

在 [xn,,xn+1] 区间上多预报几个点的斜率，然后将他们加权平均作为平均斜率，则可以构造出更高精度的公式，这就是龙格库塔方法的设计思想。

2. 亚当姆斯方法

亚当姆斯方法的设计思想是充分利用计算 yn+1 之前已得到一系列结点 xn.xn-1....上的斜率值来减少计算量。譬如，可以用 xn,xn-1 两点的斜率的加权平均作为区间[xn,xn+1]上的平均斜率，于是可以设计出如下二阶亚当姆斯格式:

$$y_{n+1} = y_n + \frac{h[3y_n' - y_{n-1}']}{2}$$

同理可得三、四阶亚当姆斯格式：

$$y_{n+1} = y_n + \frac{h[23y_n' - 16y_{n-1}' + 5y_{n-2}']}{12}$$

$$y_{n+1} = y_n + \frac{h[55y_n' - 59y_{n-1}' + 37y_{n-2}' - 9y_{n-3}']}{24}$$

同样，也可导出如下隐式的二阶、三阶和四阶亚当姆斯格式：

$$y_{n+1} = y_n + \frac{h[y_{n+1}' + y_n']}{2}$$

$$y_{n+1} = y_n + \frac{h[5y_{n+1}' + 8y_n' - y_{n-1}']}{12}$$

$$y_{n+1} = y_n + \frac{h[9y_{n+1}' + 19y_n' - 5y_{n-1}' + y_{n-2}']}{24}$$

将显式和隐式两种亚当姆斯格式相匹配，可构成下列亚当姆斯预报-校正系统

$$\begin{cases} \text{预报} & \bar{y}_{n+1} = y_n + \frac{h[55y_n' - 59y_{n-1}' + 37y_{n-2}' - 9y_{n-3}']}{24} \\ & \bar{y}_{n+1}' = f(x_{n+1}, \bar{y}_{n+1}) \\ \text{校正} & y_{n+1} = y_n + \frac{h[9y_{n+1}' + 19y_n' - 5y_{n-1}' + y_{n-2}']}{24} \\ & y_{n+1}' = f(x_{n+1}, y_{n+1}) \end{cases}$$

实验过程

①实现龙格库塔方法，验证正确性

选择微分方程 $y' = y - \frac{2x}{y}$ ，此方程有解析解 $y = \sqrt{1+2x}$ 。

实现微分方程的代码如下：

```
double f(double x, double y) {
    return y - 2 * x / y;
}
```

参数 x,y 即为微分方程中的 x,y

根据实验原理中四阶经典龙格-库塔公式，编写龙格-库塔方法函数如下：

```
double Runge_Kutta(double x0, double x1, double y0) {
    double K1, K2, K3, K4;
    double h = x1 - x0;

    K1 = f(x0, y0);
    K2 = f(x0 + h / 2.0, y0 + h / 2.0 * K1);
    K3 = f(x0 + h / 2.0, y0 + h / 2.0 * K2);
    K4 = f(x1, y0 + h * K3);

    return y0 + h / 6.0 * (K1 + 2 * K2 + 2 * K3 + K4);
}
```

参数 x0,x1,y0 即为公式中的 xn,xn+1,yn

②先用龙格-库塔方法计算前 3 点，然后实现亚当姆斯预报-校正系统，并验证算法的正确性。

实现代码如下：

```
//x 存储 x0-xn，y 存储预报值，real 存储校正值，y_dao 存储预报导数值，real_dao 存储校正导数值，true_value 存储真值，loss 存储校正值与真值的误差
double x[100], y[100], real[100], y_dao[100], real_dao[100], true_value[100], loss[100];

void Adams(double a, double b, double y0, double h) {

    //初始化
    x[0] = a;
    y[0] = y0;
    real[0] = y0;
    y_dao[0] = 0;
    real_dao[0] = 0;
    true_value[0] = y0;
    loss[0] = 0;

    int n = (int)((b - a) / h); //定义 n

    for (int i = 1; i <= n; i++)
    {
        x[i] = a + i * h; //计算 x
        true_value[i] = sqrt(1 + 2 * x[i]); //通过解析解计算真值

        if (i < 4)
        {
            //对于前三个结点，用龙格-库塔公式启动
            y[i] = Runge_Kutta(x[i - 1], x[i], real[i - 1]);
            real[i] = y[i];
            y_dao[i] = f(x[i - 1], y[i - 1]);
            real_dao[i] = y_dao[i];
        }
        else
        {
            //预报
            y[i] = real[i - 1] + h / 24.0 * (55 * real_dao[i - 1] - 59 * real_dao[i - 2] + 37 * real_dao[i - 3] - 9 * real_dao[i - 4]);
            y_dao[i] = f(x[i], y[i]);

            //校正
            real[i] = real[i - 1] + h / 24.0 * (9 * y_dao[i] + 19 * real_dao[i - 1] - 5 * real_dao[i - 2] + real_dao[i - 3]);
            real_dao[i] = f(x[i], real[i]);
        }
```

```
        loss[i] = abs(true_value[i] - real[i]); //计算误差值
    }
}
```

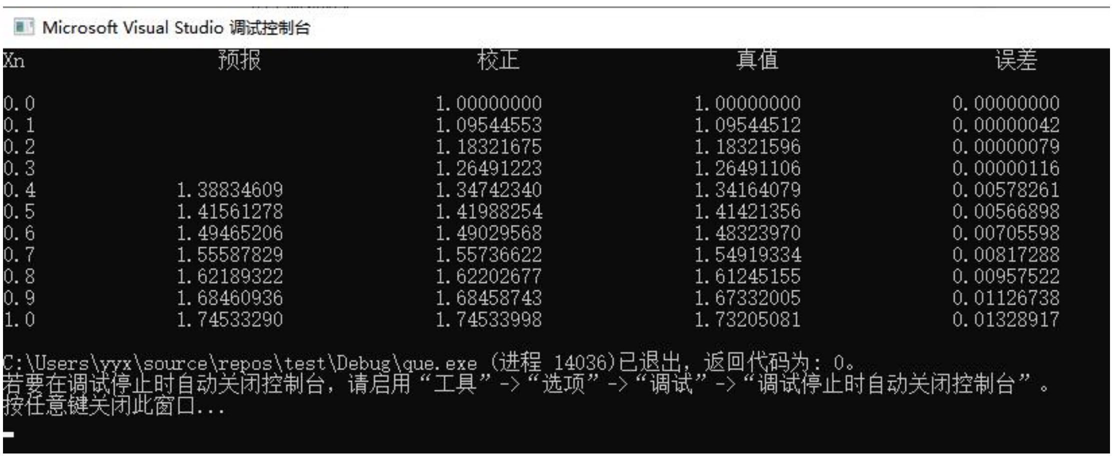
③主函数如下:

```
int main() {
    double a = 0, b = 1.0, y0 = 1.0, h = 0.1;
    Adams(a, b, y0, h);
    cout << "Xn\t\t    预报\t\t    校正\t\t    真值\t\t    误差" <<
endl << endl;
    int n = (int)((b - a) / h);
    for (int i = 0; i <= n; i++)
    {
        cout << fixed << setprecision(1) << x[i] << "\t\t";
        if (i < 4)
            cout << "          \t\t";
        else
            cout << fixed << setprecision(8) << y[i] << "\t\t";
        cout << fixed << setprecision(8) << real[i] << "\t\t" <<
true_value[i] << "\t\t" << loss[i] << endl;
    }
    return 0;
}
```

这里设置区间为[0,1]，步长 0.1

实验结果及分析

实验输出如图



```
Microsoft Visual Studio 调试控制台
Xn          预报            校正            真值            误差

0.0                     1.00000000      1.00000000      0.00000000
0.1                     1.09544553      1.09544512      0.00000042
0.2                     1.18321675      1.18321596      0.00000079
0.3                     1.26491223      1.26491106      0.00000116
0.4     1.38834609      1.34742340      1.34164079      0.00578261
0.5     1.41561278      1.41988254      1.41421356      0.00566898
0.6     1.49465206      1.49029568      1.48323970      0.00705598
0.7     1.55587829      1.55736622      1.54919334      0.00817288
0.8     1.62189322      1.62202677      1.61245155      0.00957522
0.9     1.68460936      1.68458743      1.67332005      0.01126738
1.0     1.74533290      1.74533998      1.73205081      0.01328917

C:\Users\yyx\source\repos\test\Debug\que.exe (进程 14036)已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用"工具"->"选项"->"调试"->"调试停止时自动关闭控制台"。
按任意键关闭此窗口...
```

分析：

①对于 x0, x1, x2, x3 来说，没有预报值，使用龙格-库塔启动得到的

结果作为校正值进行存储。可以看到龙格-库塔公式预测结果的误差分别为 0.00000042,0.00000079,0.00000116.误差较小，从而完成了实验目的 1：实现龙格-库塔方法，并验证它的正确性；

②对于使用亚当姆斯预报-校正系统进行预测的结点 x4-x10,误差分别为 0.00578261，0.00566898，0.00705598，0.00817288，0.00957522，0.01126738，0.01328917。可以看到误差较小，从而完成了实验目的二：先用龙格-库塔方法计算前 3 点，然后实现亚当姆斯预报-校正系统，并验证算法的正确性。

实验体会：

在这次的实验过程中，利用 c++编写了龙格-库塔算法和亚当姆斯算法的有关代码，增加了编程能力，也进一步理解了这两种常微分方程的计算方法。

龙格-库塔方法是显式的自开始方法，而且精度比较高，具有易于改变步长等优点。但使用龙格-库塔方法时，每一步需要多次计算函数 f(x,y)的值，计算量大。而且由于龙格-库塔方法的导出基于泰勒展开，所以要求函数具有较高的光滑性。对于光滑性不太好的解，最好采用低阶算法而将步长 h 取小。

亚当姆斯方法的计算量比龙格-库塔方法少，却具有同样的精度，但必须用龙格库塔方法提供开头几个函数值。