Федеральное агентство связи
Федеральное государственное бюджетное образовательное учреждение высшего образования «Сибирский государственный университет телекоммуникаций и информатики»

Кафедра ПМиК

Отчёт по курсовой работе на тему:

«Решение краевой задачи методом Рунге-Кутта IV порядка.»

Выполнил: студент группы ИА-831
Угольникова Екатерина Алексеевна
Проверил: ассистент кафедры ПМиК
Петухова Яна Владимировна

Новосибирск 2020

Содержание

1. Постановка задачи..3
2. Теоретическое обоснование..3
2.1. Краевая задача……………………………………………………………...3
2.2. Метод стрельбы для решения краевых задач...4
2.3. Метод Рунге-Кутта IV порядка..5
2.4. Метод двойного пересчета для оценки погрешности..................................6
3. Описание использованных функций..7
4. Результат выполнения программы..8
Список использованной литературы...9
Текст программы...10

1. Постановка задачи

Написать программу, реализующую решение краевой задачи методом Рунге-Кутта IV порядка.

Проверить работу программы на конкретном примере:

$$y'' = \frac{3e^x + 4y' + 2y}{9}$$
$$y(0) = 1$$
$$y(1) = 2{,}718281828$$

2. Теоретическое обоснование

2.1 Краевая задача

Краевая задача — это задача отыскания частного решения системы обыкновенных дифференциальных уравнений с дополнительными условиями, налагаемыми на значения исследуемых функций не менее чем в двух точках отрезка [a,b], чаще всего на концах этого отрезка.

Методы решения краевых задач подразделяются на точные аналитические, приближенные и численные. Численные методы решения дифференциальных уравнений подразделяются на три группы [1]:

1) методы сведения решения краевой задачи к последовательности решений задач Коши;
2) методы конечных разностей;
3) методы конечных элементов.

В контексте курсовой работы для решения краевой задачи применяется метод стрельбы, который относится к методам сведения решения краевой задачи к последовательности решений задач Коши.

2.2. Метод стрельбы для решения краевых задач

Дана краевая задача для дифференциального уравнения второго порядка

$$y''=f(x,y,y') \quad x\in[a,b] \tag{1.1}$$

С граничными условиями в виде

$$y(a)=\alpha, \tag{1.2}$$
$$y(b)=\beta. \tag{1.3}$$

Задача (1.1)-(1.3) приводится к системе уравнений первого порядка. Уравнение (1.1) перепишется в виде системы уравнений:

$$\begin{cases} y'=z, \\ z'=f(x,y,z). \end{cases} \tag{1.4}$$

Для решения (1.4) необходимо два начальных условия. Одно начальное условие задано (1.2). Для разрешимости (1.4) необходимо второе условие, соответствующее функции *z(x)* в координате *x=0*, η — произвольное значение [2].

$$y(a)=\alpha \tag{1.5}$$
$$z(a)=\eta \tag{1.6}$$

Задача (1.4)-(1.6) есть задача Коши для системы двух ДУ 1-го порядка. Решая эту задачу Коши каким-либо численным методом (в рамках курсовой работы это метод Рунге-Кутта IV порядка), получаем ее решение $y(x,\eta)$, зависящее от η, как от параметра. Так как η выбрано произвольно, то решение задачи Коши удовлетворяет условию краевой задачи в точке *a* и не удовлетворяет ее условию в точке *b*. Необходимо менять параметр η таким образом, чтобы решение задачи Коши в точке *b* совпадало с (1.3). Решение краевой задачи сводится к нахождению корня нелинейного алгебраического уравнения

$$(\eta)=y(b,\eta)-\beta=0. \tag{1.7}$$

Функция *(η)* задана в виде таблицы чисел, которая заполняется при решении серии задач Коши. Решение уравнения (1.7) можно искать методом дихотомии. Для ускорения сходимости к корню этого уравнения можно использовать другие методы, например метод секущих. Для этого делают два расчета с произвольными значениями η_1 и η_2 а следующие значения вычисляют по формуле

$$\eta^{k+1} = \eta^k - \frac{(\eta^k - \eta^{k-1})[y(b, \eta^k) - \beta]}{y(b, \eta^k) - y(b, \eta^{k-1})}. \tag{1.8}$$

Простота алгоритма метода стрельбы и возможность использования стандартных программ решения задач Коши позволяет успешно использовать его при решении как линейных, так и нелинейных ДУ.

2.3. Метод Рунге-Кутта IV порядка

Популярным среди методов Рунге-Кутта является метод IV порядка. Соответствующие формулы имеют вид

$$\begin{cases} y_{i+1} = y_i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \\ k_1 = f(x_i, y_i), \\ k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_1\right), \\ k_3 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_2\right), \\ k_4 = f(x_i + h, y_i + hk_3). \end{cases} \tag{2.1}$$

Методы Рунге-Кутта можно использовать не только для решения ДУ первого порядка, но и для решения дифференциальных уравнений более высоких порядков. Любое дифференциальных уравнений m-го порядка можно свести к системе, состоящей из m уравнений 1-го порядка при помощи замен.

Для решения системы дифференциальных уравнений используются те же методы, что и для решения одного дифференциального уравнения 1-го порядка.

При этом необходимо соблюдать условие: на каждом шаге все уравнения системы надо решать параллельно.

Если рассматривается задача Коши для системы ДУ

$$\begin{cases} y_i' = f_i(x, y_{1,..}, y_n), \quad 1 \le i \le n, \\ y_i(a) = y_i^{(0)}. \end{cases} \qquad (2.2)$$

то формулы Рунге-Кутта IV порядка аналогичны скалярному случаю. Индекс i указывает номер компоненты вектора, индекс j — номер узла x_j, который имеет компоненты $y^{(j)}=(y_1^{(j)},..,y_n^{(j)})$. Формулы Рунге-Кутта IV порядка для СДУ следующие:

$$\begin{cases} y^{(j+1)} = y^j + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), 0 \le j \le m-1, \\ k_{1,i} = f_i(x_j, y_1^{(j)}, .., y_n^{(j)}), 1 \le i \le n, \\ k_{2,i} = f_i(x_j + \frac{h}{2}, y_1^{(j)} + \frac{h}{2} k_{1,1}, y_n^{(j)} + \frac{h}{2} k_{1,n}), \\ k_{3,i} = f_i(x_j + \frac{h}{2}, y_1^{(j)} + \frac{h}{2} k_{2,1}, y_n^{(j)} + \frac{h}{2} k_{2,n}), \\ k_{4,i} = f_i(x_j + h, y_1^{(j)} + hk_{2,1}, y_n^{(j)} + hk_{3,n}) \end{cases} \qquad (2.3)$$

[3].

2.4. Метод двойного пересчета для оценки погрешности

Оценка погрешности методов Рунге-Кутта очень затруднительна. Грубую оценку погрешности можно получить с помощью двойного пересчета по формуле

$$|y_n^* - y(x_n)| \approx \frac{|y_n^* - y_n|}{15} \qquad (3.1)$$

где $y(x_n)$ — значение точного решения, y_n^* и y_n — приближенные значения, полученные с нагом h и h/2.

При реализации методов Рунге-Кутта на ЭВМ с автоматическим выбором шага обычно в каждой точке x_i делают двойной пересчет — сначала с шагом h, затем с шагом h/2. Если полученные при этом значения y_i различаются в пределах допустимой точности, то шаг h для следующей точки удваивают, в противном случае берут половинный шаг [4].

Метод двойного пересчета при решении ДУ и СДУ практически единственный имеет возможность для оценки погрешностей, так как иные формулы очень сложны и требуют оценок различных производных.

3. Описание использованных функций

Для работы программы реализованы следующие функции:

- double f1(double x, double y, double z)

  Функция возвращает значение первой производной (y')

- double f2(double x, double y, double z)

  Функция возвращает значение второй производной (y'')

- double RungeKuttFour(double h,int N,double *yy,double YD_begin)

  Функция получает на вход шаг расчета, количество точек расчета, указатель на массив результатов расчета функции и значение начального условия (1.6). Производится решение системы СДУ из двух уракнений (1.4) по формулам (2.3). Для этого параллельно расчитываются коэффициенты и вычисляются значения для обоих уравнений в конкретной точке.

- double RK_DoubleRecount(double m)

  Функция получает на вход значение начального условия (1.6) для решения СДУ (1.4). Внутри функции организован двойной пересчет для метода Рунге-Кутта IV порядка.

- void ShootingMethod()

Безтиповая функция, реализующая метод стрельбы и решение краевой задачи соответственно. В теле функции подсчитываются два начальных приближения, проводится проверка на соответствие полученных значений необходимому краевому условию и в случае соответствия, на экран выводится полученная точка, результат решения краевой задачи и производится выход из функции. Если же ни одно из полученных значений не подходит условию, высчитывается новое приближение по формуле (1.8). Далее новое приближение считается снова и снова, пока не будет выполнено условие соответствия этого приближения и искомого краевого условия.

- int main()

Главная функция проекта. Вызывает функцию ShootingMethod(), выводит информация о задаче, а так же результат вычислений.

4. Результат выполнения программы

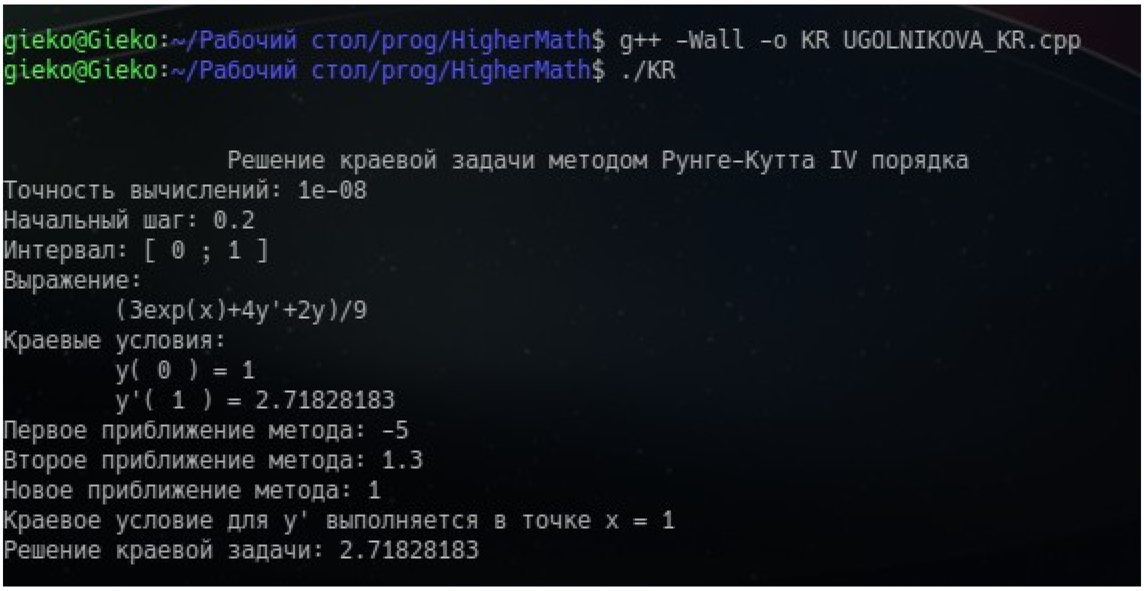

Рисунок 1. Компиляция и результат выполнения программы

Список использованной литературы

1. Мышенков В.И., Мышенков Е.В. Численные методы. Ч. 2. Численное решение обыкновенных дифференциальных уравнений: Учебное пособие для студентов специальности 073000. – М.:МГУЛ, 2005. – 109 с.: ил.
2. Крайнов А.Ю., Моисеева К.М. Численные методы решения краевых задач для обыкновенных дифференциальных уравнений : учеб. пособие. – Томск : STT, 2016. – 44 с.
3. Боглаев Ю. П. Вычислительная математика и программирование: Учеб. Пособие для студентов втузов. – М.: Высш. шк., 1990. – 544с.: ил.
4. Копченова Н. В., Марон И. А. Вычислительная математика в примерах и задачах: Учебное пособие. 2-е изд., стер. – СПБ.: Издательство «Лань», 2008. –368 с. – (Учебники для вузов. Специальная литература).

Текст программы

```
#include <iostream>
#include <math.h>
using namespace std;

double a=3,b=4,c=2;
double EPS = 0.00000001;
double H=0.2,h=H/2;
double X_begin=0,X_end=1;
double Y_begin=1,YD_end=2.718281828459;
double YD_begin=-5;
double m1=YD_begin,m2=1,m3;
double *Y1,*Y2;
double shot[3];

double f1(double x, double y, double z) {
    return z;
}
double f2(double x, double y, double z) {
    return (a*exp(x)+b*z+c*y)/(a+b+c);
}
double RungeKuttFour(double h,int N,double *yy,double
YD_begin){
    double X=X_begin, Y=Y_begin, Z=YD_begin;
    double K[4],L[4];
    for (int i=0;i<N;i++) {
        K[0]=h*f2(X, Y, Z);
        L[0]=h*f1(X, Y, Z);
        K[1]=h*f2(X+h/2, Y+L[0]/2, Z+K[0]/2);
        L[1]=h*f1(X+h/2, Y+L[0]/2, Z+K[0]/2);
        K[2]=h*f2(X+h/2, Y+L[1]/2, Z+K[1]/2);
        L[2]=h*f1(X+h/2, Y+L[1]/2, Z+K[1]/2);
        K[3]=h*f2(X+h, Y+L[2], Z+K[2]);
        L[3]=h*f1(X+h, Y+L[2], Z+K[2]);
        Z=Z+(K[0]+2*K[1]+2*K[2]+K[3])/6;
        yy[i]=Y+(L[0]+2*L[1]+2*L[2]+L[3])/6;
```

```
        Y=yy[i];
        X+=h;
    }
    return Y;
}
double RK_DoubleRecount(double m) {
    int n1, n2;
    double result;
    do {
        n1=(X_end-X_begin)/H;
        n2=(X_end-X_begin)/h;
        Y1=new double[n1];
        Y2=new double[n2];
        RungeKuttFour(H, n1, Y1, m);
        result=RungeKuttFour(h, n2, Y2, m);
        h*=0.5;
        H=2*h;
    } while(abs(Y2[n2-1]-Y1[n1-1])>EPS);
    h=0.1;
    H=2*h;
    return result;
}
void ShootingMethod() {
    cout<<"Первое приближение метода: "<<m1<<endl;
    shot[0]=RK_DoubleRecount(m1);
    cout<<"Второе приближение метода: "<<m2<<endl;
    shot[1]=RK_DoubleRecount(m2);
    if(abs(shot[0]-YD_end)<EPS){
        cout<<"Краевое условие для y' выполняется в точке x =
"<<m1<<endl;
        printf("Решение краевой задачи: %0.8f\n\n", shot[0]);
        return;
    }
    else if(abs(shot[1]-YD_end)<EPS){
        cout<<"Краевое условие для y' выполняется в точке x =
"<<m2<<endl;
        printf("Решение краевой задачи: %0.8f\n\n", shot[1]);
        return;
```

```
    }
    else {
        m3=m2+(((m2-m1)*(YD_end-shot[1]))/((shot[1]-shot[0])));
        cout<<"Краевое условие для y' выполняется в точке x = "<<m3<<endl;
        shot[2]=RK_DoubleRecount(m3);
    }
    while (abs(shot[2]-YD_end)>=EPS) {
        m1=m2;
        m2=m3;
        shot[0]=shot[1];
        shot[1]=shot[2];
        m3=m2+(((m2-m1)*(YD_end-shot[1]))/((shot[1]-shot[0])));
        YD_begin=m3;
        cout<<"Новое приближение метода: "<<m3<<endl;
        shot[2]=RK_DoubleRecount(m3);
    }
    printf("Решение краевой задачи: %0.8f\n\n", shot[2]);
    cout<<endl;
}
int main() {
    cout<<"\n\n\t\tРешение краевой задачи методом Рунге-Кутта IV порядка"<<endl;
    cout<<"Точность вычислений: "<<EPS<<endl;
    cout<<"Начальный шаг: "<<H<<endl;
    cout<<"Интервал: [ "<<X_begin<<" ; "<<X_end<<" ]"<<endl;
    cout<<"Выражение:"<<endl;
    cout<<"\t("<<a<<"exp(x)+"<<b<<"y'+"<<c<<"y)/"<<a+b+c<<endl;
    cout<<"Краевые условия:"<<endl;
    cout<<"\ty( "<<X_begin<<" ) = "<<Y_begin<<endl;
    printf("\ty'( %.0f ) = %.8f \n",X_end,YD_end);
    ShootingMethod();
    return 0;
}
```