уМИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Поволжский государственный университет телекоммуникаций и информатики»

Факультет информатики и вычислительной техники

Кафедра ПрИ

ОТЧЁТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

Дисциплина: Численные методы

Выполнил: студент
ПрИ-21 Морзюков
М.А.
Проверил(а):
Осанов В.А.

Самара 2024

Цель работы: изучить методы решения задачи Коши, такие как метод Эйлера, метод Эйлера-Коши, метод Рунге-Кутта 4-го порядка, реализовать программными средствами.

Вариант 11

№ варианта	$F_1(x,y_1,y_2)$	$F_2(x,y_1,y_2)$	$y_1(a)$	$y_2(a)$	a	b
11	$arctg\left(\frac{1}{1+y_1^2} + y_2^2 \right)$	$sin(y_1 y_2)$	1	1	1	4

Метод Эйлера: это численный метод, который используется для приближенного решения обыкновенных дифференциальных уравнений. Основная идея метода заключается в том, чтобы на каждом шаге вычислять новое значение функции, используя текущее значение и производную (или правую часть уравнения).

```java
public class Euler {
    public static void solve(double a, double b, int n, Function2[] TFMas, double[] y) {
        double h = (b - a) / n;
        double x = a;
        double[][] TFunZnach = new double[n + 1][2];
        TFunZnach[0][0] = y[0];
        TFunZnach[0][1] = y[1];

        for (int i = 0; i < n; i++) {
            double xNext = x + h;

            double[] yNext = new double[2];
            yNext[0] = y[0] + h * TFMas[0].apply(y[0], y[1]);
            yNext[1] = y[1] + h * TFMas[1].apply(y[0], y[1]);
            TFunZnach[i + 1][0] = yNext[0];
            TFunZnach[i + 1][1] = yNext[1];
            x = xNext;
            y = yNext;
        }

        for (int i = 0; i <= n; i++) {
            double xi = a + i * h;
            System.out.printf("x = %.2f, y1 = %.6f, y2 = %.6f%n", xi, TFunZnach[i][0], TFunZnach[i][1]);
        }
    }
}
```

Метод Эйлера-Коши: это улучшенная версия метода Эйлера, которая использует среднее значение производной на шаге интегрирования для вычисления следующего значения функции. Это позволяет улучшить точность приближения.

```
public class EulerCauchy {
    public static void solve(double a, double b, int n, Function2[] TFMas, double[] y) {
        double h = (b - a) / n;
        double x = a;
        double[][] TFunZnach = new double[n + 1][2];
        TFunZnach[0][0] = y[0];
        TFunZnach[0][1] = y[1];

        for (int i = 0; i < n; i++) {
            double xNext = x + h;
            double[] yNext = new double[2];
            yNext[0] = y[0] + h * TFMas[0].apply(y[0] + h / 2 * TFMas[0].apply(y[0], y[1]), y[1]);
            yNext[1] = y[1] + h * TFMas[1].apply(y[0] + h / 2 * TFMas[0].apply(y[0], y[1]), y[1]);
            TFunZnach[i + 1][0] = yNext[0];
            TFunZnach[i + 1][1] = yNext[1];
            x = xNext;
            y = yNext;
        }

        for (int i = 0; i <= n; i++) {
            double xi = a + i * h;
            System.out.printf("x = %.2f, y1 = %.6f, y2 = %.6f%n", xi, TFunZnach[i][0], TFunZnach[i][1]);
        }
    }
}
```

Метод Рунге-Кутта: это более сложный численный метод, который использует несколько промежуточных вычислений для получения более точного результата. Он обладает более высокой точностью, чем методы Эйлера и Эйлера-Коши. Этот метод обладает высокой точностью (четвертый порядок аппроксимации), что делает его предпочтительным для большинства задач, требующих точных решений.

```
public class RungeKutta {
    public static void solve(double a, double b, int n, Function2[] TFMas, double[] y) {
        double h = (b - a) / n;
        double x = a;
        double[][] TFunZnach = new double[n + 1][2];
        TFunZnach[0][0] = y[0];
        TFunZnach[0][1] = y[1];

        for (int i = 0; i < n; i++) {
            double xNext = x + h;
            double k1_0 = h * TFMas[0].apply(y[0], y[1]);
            double k1_1 = h * TFMas[1].apply(y[0], y[1]);

            double k2_0 = h * TFMas[0].apply(y[0] + k1_0 / 2, y[1] + k1_1 / 2);
            double k2_1 = h * TFMas[1].apply(y[0] + k1_0 / 2, y[1] + k1_1 / 2);

            double k3_0 = h * TFMas[0].apply(y[0] + k2_0 / 2, y[1] + k2_1 / 2);
            double k3_1 = h * TFMas[1].apply(y[0] + k2_0 / 2, y[1] + k2_1 / 2);

            double k4_0 = h * TFMas[0].apply(y[0] + k3_0, y[1] + k3_1);
            double k4_1 = h * TFMas[1].apply(y[0] + k3_0, y[1] + k3_1);

            double[] yNext = new double[2];
            yNext[0] = y[0] + (k1_0 + 2 * k2_0 + 2 * k3_0 + k4_0) / 6;
            yNext[1] = y[1] + (k1_1 + 2 * k2_1 + 2 * k3_1 + k4_1) / 6;

            TFunZnach[i + 1][0] = yNext[0];
            TFunZnach[i + 1][1] = yNext[1];

            x = xNext;
            y = yNext;
        }

        for (int i = 0; i <= n; i++) {
            double xi = a + i * h;
            System.out.printf("x = %.2f, y1 = %.6f, y2 = %.6f%n", xi, TFunZnach[i][0], TFunZnach[i][1]);
        }
```

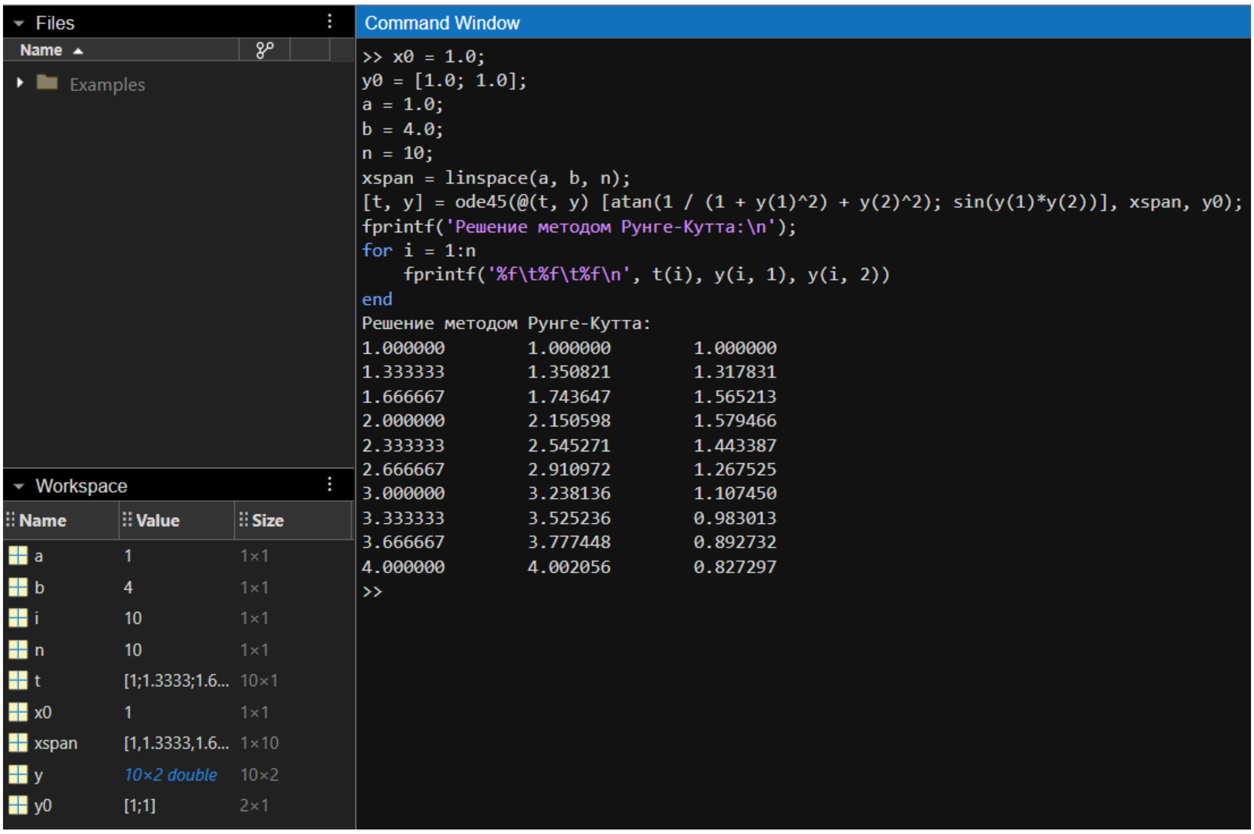
Результат программы:

```
Решение методом Эйлера:
x = 1,00, y1 = 1,000000, y2 = 1,000000
x = 1,30, y1 = 1,294838, y2 = 1,252441
x = 1,60, y1 = 1,623434, y2 = 1,552053
x = 1,90, y1 = 1,987676, y2 = 1,726836
x = 2,20, y1 = 2,367619, y2 = 1,640821
x = 2,50, y1 = 2,737413, y2 = 1,437816
x = 2,80, y1 = 3,079892, y2 = 1,223803
x = 3,10, y1 = 3,382964, y2 = 1,047645
x = 3,40, y1 = 3,643037, y2 = 0,930114
x = 3,70, y1 = 3,868612, y2 = 0,856810
x = 4,00, y1 = 4,070439, y2 = 0,805147

Решение методом Эйлера-Коши:
x = 1,00, y1 = 1,000000, y2 = 1,000000
x = 1,30, y1 = 1,288327, y2 = 1,273512
x = 1,60, y1 = 1,616962, y2 = 1,561719
x = 1,90, y1 = 1,980873, y2 = 1,659225
x = 2,20, y1 = 2,353395, y2 = 1,527418
x = 2,50, y1 = 2,709142, y2 = 1,328412
x = 2,80, y1 = 3,033250, y2 = 1,141399
x = 3,10, y1 = 3,317702, y2 = 1,001975
x = 3,40, y1 = 3,565103, y2 = 0,911320
x = 3,70, y1 = 3,784853, y2 = 0,849430
x = 4,00, y1 = 3,984265, y2 = 0,802134

Решение методом Рунге-Кутта:
x = 1,00, y1 = 1,000000, y2 = 1,000000
x = 1,30, y1 = 1,313580, y2 = 1,284757
x = 1,60, y1 = 1,662890, y2 = 1,532998
x = 1,90, y1 = 2,028762, y2 = 1,596937
x = 2,20, y1 = 2,390041, y2 = 1,507004
x = 2,50, y1 = 2,732432, y2 = 1,355663
x = 2,80, y1 = 3,046629, y2 = 1,199758
x = 3,10, y1 = 3,328177, y2 = 1,066242
x = 3,40, y1 = 3,578048, y2 = 0,962735
x = 3,70, y1 = 3,800834, y2 = 0,885556
x = 4,00, y1 = 4,001874, y2 = 0,827594
```

Проверка:




```
>> x0 = 1.0;
y0 = [1.0; 1.0];
a = 1.0;
b = 4.0;
n = 10;
xspan = linspace(a, b, n);
[t, y] = ode45(@(t, y) [atan(1 / (1 + y(1)^2) + y(2)^2); sin(y(1)*y(2))], xspan, y0);
fprintf('Решение методом Рунге-Кутта:\n');
for i = 1:n
    fprintf('%f\t%f\t%f\n', t(i), y(i, 1), y(i, 2))
end
Решение методом Рунге-Кутта:
1.000000	1.000000	1.000000
1.333333	1.350821	1.317831
1.666667	1.743647	1.565213
2.000000	2.150598	1.579466
2.333333	2.545271	1.443387
2.666667	2.910972	1.267525
3.000000	3.238136	1.107450
3.333333	3.525236	0.983013
3.666667	3.777448	0.892732
4.000000	4.002056	0.827297
>>
```