Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Отчет по лабораторной работе №6

«Численное решение обыкновенных дифференциальных уравнений»

По дисциплине «Вычислительная математика»

Вариант 8

Выполнила: Иванова Мария Максимовна

Группа: P3208

Преподаватель: Машина Екатерина Алексеевна

Санкт-Петербург

~ 2024 ~

Цель работы:

решить задачу Коши для обыкновенных дифференциальных уравнений численными методами.

Рабочие формулы:

1. Модифицированный метод Эйлера

$$y_{i+1} = y_i + \frac{h}{2}\left[f(x_i, y_i) + f\left(x_{i+1}, y_i + hf(x_i, y_i)\right)\right],\ i = 0, 1 \dots$$

2. Метод Рунге Кутта IV порядка

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$

$$k_1 = h \cdot f(x_i, y_i)$$
$$k_2 = h \cdot f(x_i + \frac{h}{2}, y_i + \frac{k_1}{2})$$
$$k_3 = h \cdot f(x_i + \frac{h}{2}, y_i + \frac{k_2}{2})$$
$$k_4 = h \cdot f(x_i + h, y_i + k_3)$$

3. Метод Адамса

$$\Delta f_i = f_i - f_{i-1}$$
$$\Delta^2 f_i = f_i - 2f_{i-1} + f_{i-2}$$
$$\Delta^3 f_i = f_i - 3f_{i-1} + 3f_{i-2} - f_{i-3}$$

$$y_{i+1} = y_i + hf_i + \frac{h^2}{2}\Delta f_i + \frac{5h^3}{12}\Delta^2 f_i + \frac{3h^4}{8}\Delta^3 f_i$$

Листинг программы:

```
package org.example;

import static java.lang.Math.abs;

public class ImprovedEulerMethod {
    public double[] x = new double[20];
    public double[] y = new double[20];
    public double[] f= new double[20];

    public void solve(double x0, double y0, double a, double b, double h, int
functionChoice) {
        Functions functions = new Functions();
        x[0] = x0;
        y[0] = y0;
        f[0] = functions.f(functionChoice, x0, y0);
        int i = 0;
        double y_temp;
        while (a < b) {
            i++;
            a+=h;
```

```java
            x[i] = a;
            y_temp = y[i - 1] + h * functions.f(functionChoice, x[i - 1], y[i
- 1]);
            y[i] = y[i - 1] + h / 2 * (functions.f(functionChoice, x[i - 1],
y[i - 1]) + functions.f(functionChoice, x[i], y_temp));
            f[i] = functions.f(functionChoice, x[i], y[i]);
        }
    }

    public double[] getX() {
        return x;
    }

    public double[] getY() {
        return y;
    }

    public double[] getF(){
        return f;
    }


    public double RungeRule(int n, double h) {
        return abs(Math.pow(y[n], h) - Math.pow(y[n], h / 2)) / (Math.pow(2,
2) - 1);
    }
}
```

```java
package org.example;

import static java.lang.Math.abs;

public class RungeKuttaMethod {
    public double[] x = new double[20];
    public double[] y= new double[20];
    public double[] f= new double[20];

    public double[] solve(double x0, double y0, double a, double b, double h,
int functionChoice) {
        Functions functions = new Functions();
        x[0] = x0;
        y[0] = y0;
        f[0] = functions.f(functionChoice, x0, y0);
        int i = 1;
        double k1, k2, k3, k4;
        a+=h;
        while (a <= b) {
            x[i] = a;
            k1 = h * functions.f(functionChoice, x[i - 1], y[i - 1]);
            k2 = h * functions.f(functionChoice, x[i - 1] + h / 2, y[i - 1] +
k1 / 2);
            k3 = h * functions.f(functionChoice, x[i - 1] + h / 2, y[i - 1] +
k2 / 2);
            k4 = h * functions.f(functionChoice, x[i - 1] + h, y[i - 1] +
k3);
            y[i] = y[i-1]+1.0/6*(k1+2*k2+2*k3+k4);
            f[i] = functions.f(functionChoice, x[i], y[i]);
            a+=h;
            i++;
        }
        return f;
    }
```

```java
    public double[] getX() {
        return x;
    }

    public double[] getY() {
        return y;
    }

    public double RungeRule(int n, double h) {
        return abs(Math.pow(y[n], h) - Math.pow(y[n], h / 2)) / (Math.pow(2,
4) - 1);
    }

    public double[] getF() {
        return f;
    }
}
```

```java
package org.example;

public class AdamsMethod {
    public double[] x = new double[20];
    public double[] y = new double[20];
    public double[] f = new double[20];

    public void solve(double x0, double y0, double a, double b, double h, int
functionChoice) {
        Functions functions = new Functions();
        double df1 = 0;
        double df2 = 0;
        double df3 = 0;
        RungeKuttaMethod rungeKuttaMethod = new RungeKuttaMethod();
        rungeKuttaMethod.solve(x0, y0, a, a + 3 * h, h, functionChoice);
        for (int i = 0; i < 4; i++) {
            f[i] = rungeKuttaMethod.getF()[i];
            x[i] = rungeKuttaMethod.getX()[i];
            y[i] = rungeKuttaMethod.getY()[i];
        }
        int i = 3;
        a = a + 3 * h;
        while (a < b) {
            a += h;
            df1 = f[i] - f[i - 1];
            df2 = f[i] - 2 * f[i - 1] + f[i - 2];
            df3 = f[i] - 3 * f[i - 1] + 3 * f[i - 2] + f[i - 3];
            y[i + 1] = y[i] + h * f[i] + Math.pow(h, 2) / 2 * df1 + 5 *
Math.pow(h, 3) / 12 * df2 + 3 * Math.pow(h, 4) / 8 * df3;
            x[i+1] = a;
            f[i+1] = functions.f(functionChoice, x[i], y[i]);
            i++;
        }
    }

    public double[] getX() {
        return x;
    }

    public double[] getY() {
        return y;
    }
```

```
    public double[] getF() {
        return f;
    }

    public double getInaccuracy(int n, int functionChoice, double x0, double
y0) {
        double eps = 0;
        Functions functions = new Functions();
        double[] y_exact = new double[n];
        for (int i = 0; i < n; i++) {
            y_exact[i] = functions.exactY(functionChoice, x[i], x0, y0);
            if (Math.abs(y_exact[i] - y[i]) > eps) {
                eps = Math.abs(y_exact[i] - y[i]);
            }
        }
        return eps;
    }

}
```

Результат работы программы:

Выберите функцию:
1. y' = x+y
2. y' = x^2 + y + 1
3. y' = e^x

2
Выберите интервал (2 числа через пробел)
0 1
Введите начальные условия (x0 и y0 через пробел)
0 0
Выберите h:
0,2
Модифицированный метод Эйлера:

i	x_i	y_i	f(x_i, y_i)	y_exact
0	0,0000	0,0000	1,0000	0,0000
1	0,2000	0,2240	1,2640	0,2242
2	0,4000	0,5141	1,6741	0,5155
3	0,6000	0,9024	2,2624	0,9064
4	0,8000	1,4281	3,0681	1,4366
5	1,0000	2,1391	4,1391	2,1548

Погрешность:0.028414928046852344
Метод Рунге-Кутты 4 порядка:

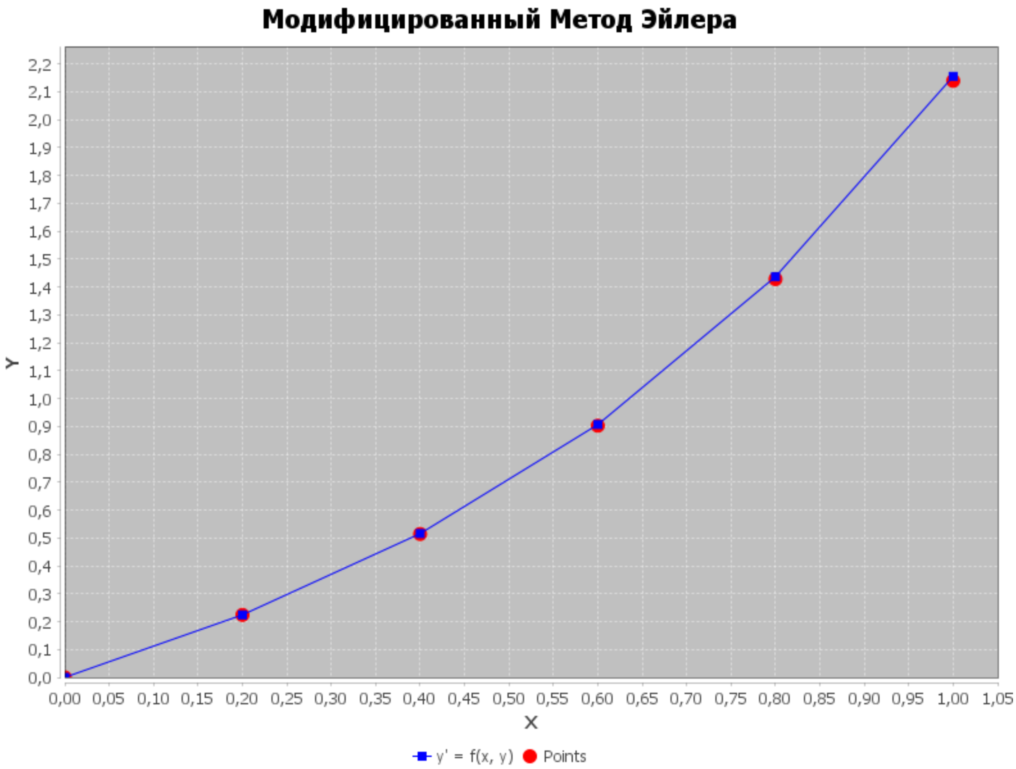
i	x_i	y_i	f(x_i, y_i)	y_exact
0	0,0000	0,0000	1,0000	0,0000
1	0,2000	0,2242	1,2642	0,2242
2	0,4000	0,5155	1,6755	0,5155
3	0,6000	0,9063	2,2663	0,9064
4	0,8000	1,4366	3,0766	1,4366
5	1,0000	2,1548	4,1548	2,1548

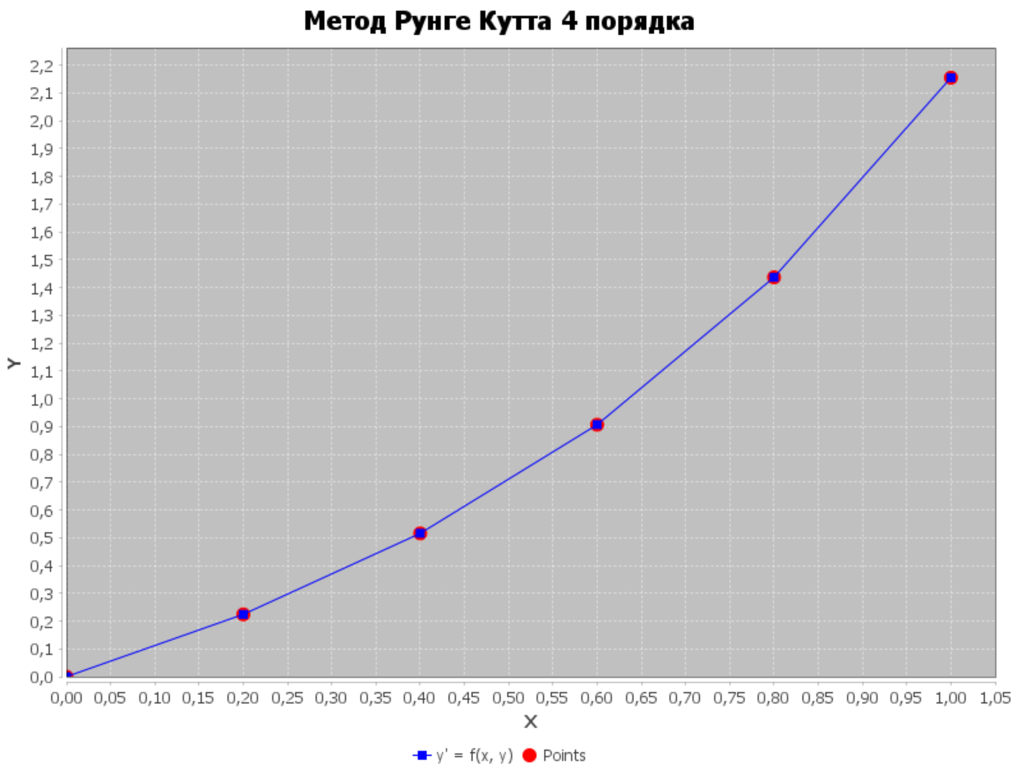
Погрешность:0.005744051120599103
Метод Адамса:

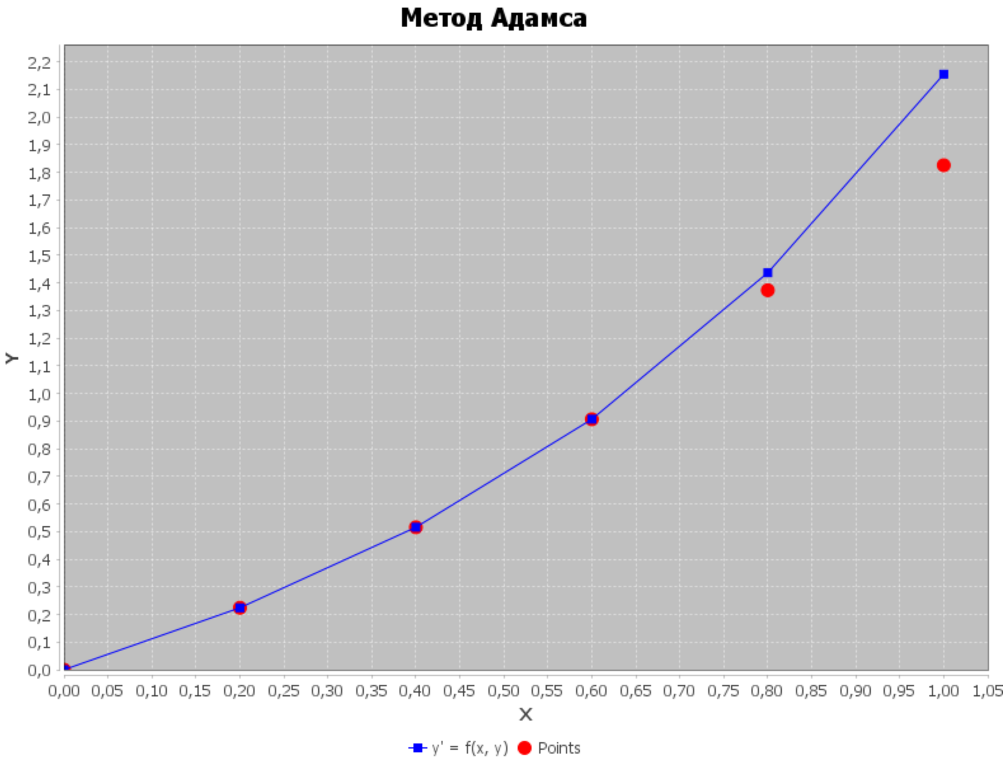
i	x_i	y_i	f(x_i, y_i)	y_exact
0	0,0000	0,0000	1,0000	0,0000
1	0,2000	0,2242	1,2642	0,2242

2	0,4000	0,5155	1,6755	0,5155
3	0,6000	0,9063	2,2663	0,9064
4	0,8000	1,3732	2,2663	1,4366
5	1,0000	1,8256	3,0132	2,1548

Погрешность:0.329242787224854







Вывод:

Во время выполнения работы мне удалось изучить методы решений дифференциальных уравнений.