МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

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

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА №5

по дисциплине

“ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА”

Вариант «Метод Милна»

Студент:

Чернова Анна Ивановна

Группа P32301

Преподаватель:

Перл Ольга Вячеславовна

Санкт-Петербург, 2023

1. Описание реализованного метода, расчетные формулы

Решение в каждой точке находится в 2 этапа: на первом этапе осуществляется прогноз, на втором – коррекция полученного значения. Если полученное значение существенно отличается от спрогнозированного, то проводится еще один этапа коррекции и т. д.

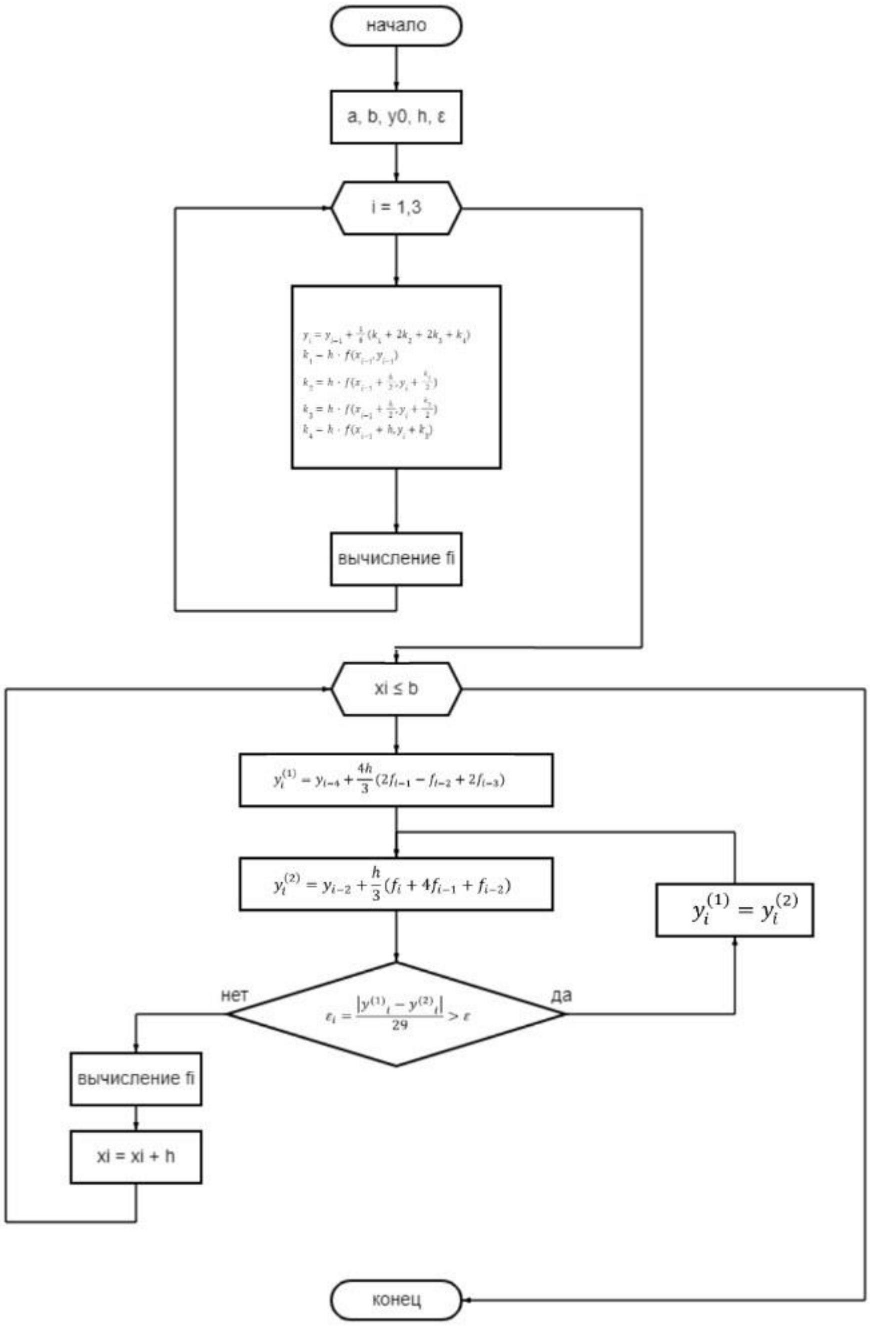
Для начала счета необходимо получить решения в первых трех точках с помощью одношаговых методов, например, методом Рунге-Кутта.

Формула прогноза: $y_i = y_{i-4} + \frac{4h}{3}(2f_{i-1} - f_{i-2} + 2f_{i-3})$

Формула коррекции: $y_i = y_{i-2} + \frac{h}{3}(f_i + 4f_{i-1} + f_{i-2})$

Критерий окончания: $\varepsilon_i = \frac{\left|y^{(1)}{}_i - y^{(2)}{}_i\right|}{29} \leq \varepsilon$

2. Блок-схема численного метода



3. Листинг реализованного метода

```
private double doRungeKuttaMethod(double x, double y) {
    double k1 = h * equation.calcFunction(x, y);
    double k2 = h * equation.calcFunction(x + h / 2, y + k1 / 2.);
    double k3 = h * equation.calcFunction(x + h / 2, y + k2 / 2.);
    double k4 = h * equation.calcFunction(x + h, y + k3);
    return y + (k1 + 2 * k2 + 2 * k3 + k4) / 6.;
}
```

```java
public void doMilneMethod() {
    double epsilon = equation.getEpsilon();
    ArrayList<Point> points = equation.getPoints();
    ArrayList<Double> function = equation.getFunc();
    double leftBorder = equation.getLeftBound();
    double y1 = doRungeKuttaMethod(leftBorder, equation.getStartY());
    double y2 = doRungeKuttaMethod(leftBorder + h, y1);
    double y3 = doRungeKuttaMethod(leftBorder + 2 * h, y2);
    points.add(new Point(leftBorder + h, y1));
    points.add(new Point(leftBorder + 2 * h, y2));
    points.add(new Point(leftBorder + 3 * h, y3));
    double f0 = equation.calcFunction(leftBorder, equation.getStartY());
    double f1 = equation.calcFunction(leftBorder + h, y1);
    double f2 = equation.calcFunction(leftBorder + 2 * h, y2);
    double f3 = equation.calcFunction(leftBorder + 3 * h, y3);
    function.add(f0);
    function.add(f1);
    function.add(f2);
    function.add(f3);
    int i = 4;
    for (double x = leftBorder + 4 * h; x <= equation.getRightBound(); x +=
h) {
        double yPrediction = points.get(i - 4).getY() + 4 * h * (2 *
function.get(i - 1) - function.get(i - 2) + 2 * function.get(i - 3)) / 3.;
        while (true) {
            double yCorrection = points.get(i - 2).getY() + h *
(function.get(i - 2) + 4 * function.get(i - 1) + equation.calcFunction(x,
yPrediction)) / 3;
            double currentEpsilon = Math.abs(yCorrection - yPrediction) /
29;
            if (currentEpsilon <= epsilon) {
                points.add(new Point(x, yCorrection));
                function.add(equation.calcFunction(x, yCorrection));
                break;
            } else {
                yPrediction = yCorrection;
            }
        }
        i++;
    }
}
```

4. Примеры результата работы программы

Выберите функцию из списка. Введите число от 1 до 3

1. y' = -3 * y
2. y' = y * (x^2 + 1)
3. y' = e^(x - y)

1

Введите левую границу

2

Введите правую границу

4

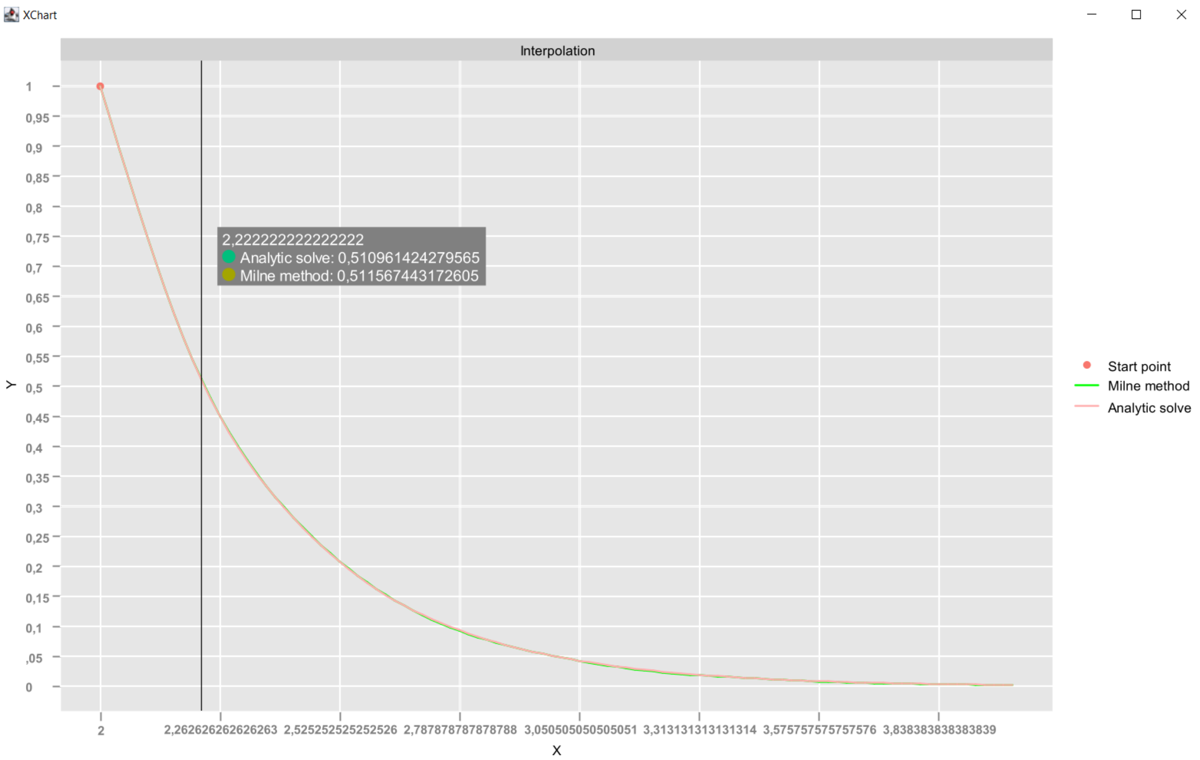
Введите шаг

0.2

Введите начальное значение y
1

Введите точность
0.01

```
   x   |   y
-------|-------
2,0000 | 1,0000
2,2000 | 0,5494
2,4000 | 0,3018
2,6000 | 0,1658
2,8000 | 0,0894
3,0000 | 0,0499
3,2000 | 0,0259
3,4000 | 0,0149
3,6000 | 0,0075
3,8000 | 0,0043
4,0000 | 0,0024
```



4. Вывод

В данной лабораторной работе были рассмотрены одношаговые и многошаговые методы численного дифференцирования. Они отличаются тем, что в одношаговых методах для вычисления текущего значения используется значение, полученное на предыдущем шаге, а в многошаговых методах для вычисления текущего значения используются результаты k предыдущих шагов. Также при использовании многошаговых методов первые несколько значений необходимо получить с помощью одношаговых методов. Многошаговые методы более точные, чем одношаговые.

Метод Милна относится к многошаговым методам и является одним из методов прогноза и коррекции. Для получения формул Милна используется первая интерполяционная формула Ньютона с разностями до третьего порядка.

Сравнивая метод Милна и метод Адамса (который тоже является многошаговым), можно сделать вывод, что они имеют одинаковую суммарную погрешность, однако метод Милна требует меньшего количества вычислений.