Министерство науки и высшего образования Российской Федерации
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 Адамса”

Группа: P32312

Выполнил: Обляшевский С.А.

Преподаватель:
Перл О. В.

Санкт-Петербург, 2023

Описание метода, расчетные формулы:

Задача Коши - одна из основных задач теории дифференциальных уравнений (обыкновенных и с частными производными); состоит в нахождении решения (интеграла) дифференциального уравнения, удовлетворяющего так называемым начальным условиям (начальным данным).

В данной лабораторной работе я рассмотрел решение задачи Коши при помощи метода Адамса. Этот метод является многошаговым, т. е. использует для вычисления очередного значения искомого решения не одно, а несколько значений, которые уже вычислены в предыдущих точках.

Так как входные данные для лабораторной работы содержат всего одну точку, а метод Адамса может работать при кол-ве точек не менее 4, я также реализовал метод Рунге – Кутта для вычисления 3 дополнительных стартовых точек. Я взял самый распространенный метод Рунге – Кутта 4 порядка точности.

Метод Рунге – Кутта 4 порядка точности:

Пусть есть дифференциальное уравнение первого порядка $y' = f(x, y), y(x_0) = y_0$, тогда приближенное значение в следующих точках вычисляется по итерационной формуле: $y_{(n+1)} = y_n + \frac{\mathrm{h}}{6}(k_1 + 2k_2 + 2k_3 + k_4)$, где $k_1 = f(x_n, y_n)$, $k_2 = f\left(x_n + \frac{\mathrm{h}}{2}, y_n + \frac{\mathrm{h}}{2}k_1\right)$, $k_3 = f\left(x_n + \frac{\mathrm{h}}{2}, y_n + \frac{\mathrm{h}}{2}k_2\right)$, $k_4 = f(x_n + \mathrm{h}, y_n + \mathrm{h}k_3)$, где h – величина шага. Этот метод имеет четвертый порядок точности, ошибка на одном шаге составляет $O(h^5)$, ошибка на конечном интервале интегрирования - $O(h^4)$.

Метод Адамса:

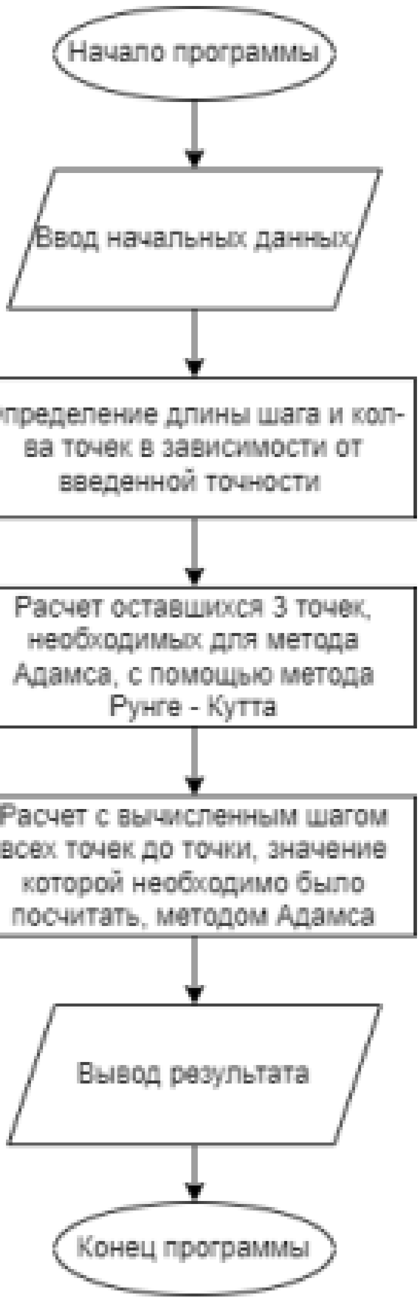
Пусть есть дифференциальное уравнение первого порядка $y' = f(x, y), y(x_0) = y_0$, для которого надо найти решение на сетке с постоянным шагом $x_n - x_0 = nh$, тогда расчетная формула метода Адамса выглядит следующим образом:

$$y_{n+1} = y_n + h \sum_{\lambda=-1}^{k-1} v_{-\lambda} f(x_{n-\lambda}, y_{n-\lambda})$$

где ϑ - некоторая постоянная

Локальная погрешность методов Адамса k порядка равна $O(h^k)$.

Блок – схема:



Начало программы
Ввод начальных данных
Определение длины шага и кол-ва точек в зависимости от введенной точности
Расчет оставшихся 3 точек, необходимых для метода Адамса, с помощью метода Рунге - Кутта
Расчет с вычисленным шагом всех точек до точки, значение которой необходимо было посчитать, методом Адамса
Вывод результата
Конец программы

Листинг численного метода:

Метод Рунге - Кутта:

```
@Override
public Dot calculateIntegral(Function function, Dot startDot, double
calculatedDot, double eps) {
    double k1 = step * function.dy(startDot.getX(), startDot.getY());
    double k2 = step * function.dy(startDot.getX() + step / 2,
startDot.getY() + k1 / 2);
    double k3 = step * function.dy(startDot.getX() + step / 2,
startDot.getY() + k2 / 2);
    double k4 = step * function.dy(startDot.getX() + step, startDot.getY() +
k3);
    double y = startDot.getY() + (k1 + 2 * k2 + 2 * k3 + k4) / 6;
    double x = startDot.getX() + step;
    return new Dot(x, y);
}
```

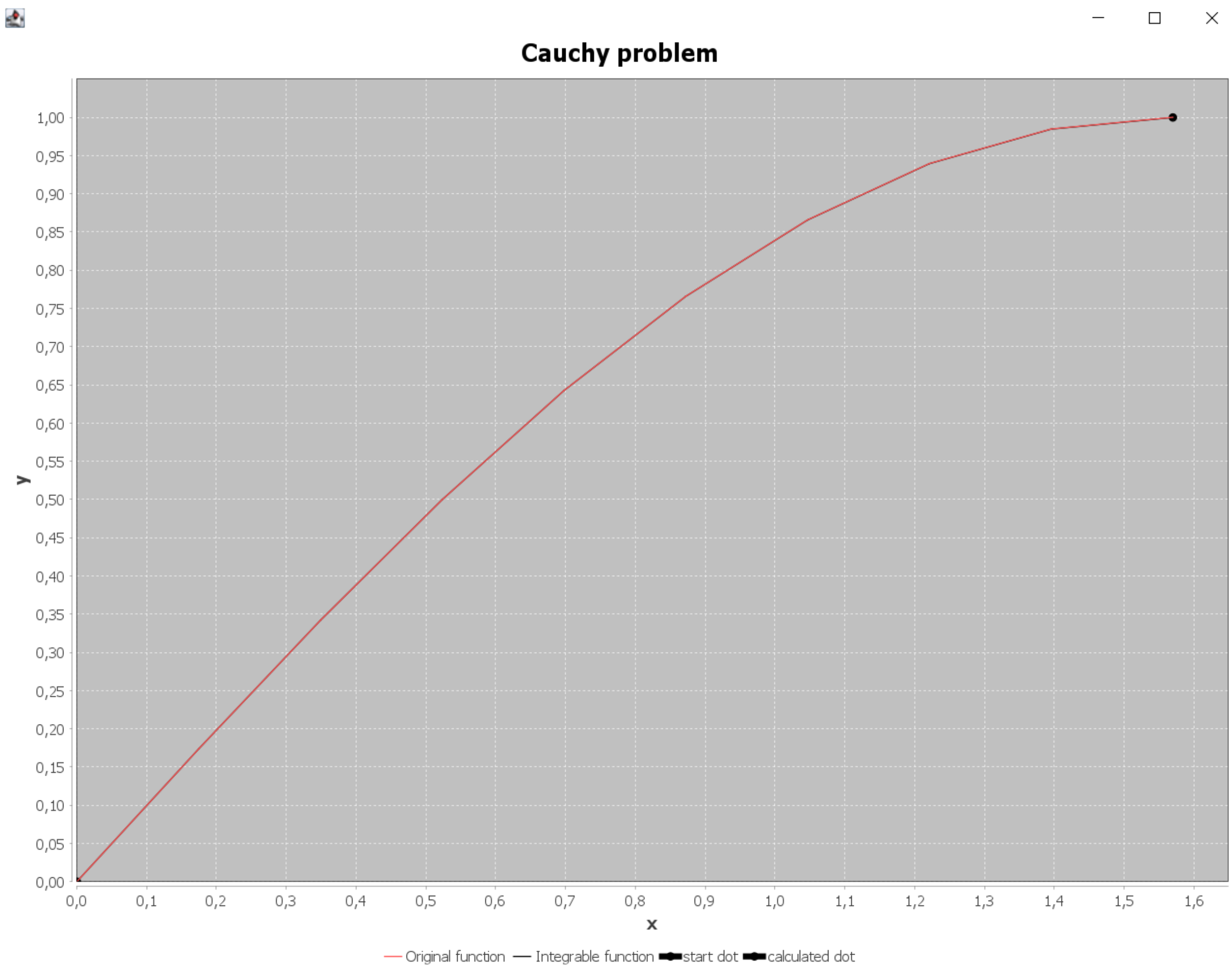
Метод Адамса:

```
@Override
public Dot[] calculateIntegral(Function function, Dot startDot, double
endDot, double eps) {
    double start = startDot.getX();
    double step = (endDot - start)/4;
    int n = 5;
    while (Math.pow(step, 4) > eps){
        step = (endDot - start)/++n;
    }
    oneStepMethod.setStep(step);
    Dot[] calculatedDots = new Dot[n + 1];
    calculatedDots[0] = startDot;
    for (int i = 1; i < 4; i++){
        calculatedDots[i] = oneStepMethod.calculateIntegral(function,
calculatedDots[i - 1], calculatedDots[i - 1].getX(), eps);
    }
    for (int i = 4; i < n + 1; i++){
        double newY = calculatedDots[i - 1].getY() + step * (55 *
function.dy(calculatedDots[i - 1].getX(), calculatedDots[i - 1].getY())
                - 59 * function.dy(calculatedDots[i - 2].getX(),
calculatedDots[i - 2].getY()) + 37 * function.dy(calculatedDots[i -
3].getX(), calculatedDots[i - 3].getY())
                - 9 * function.dy(calculatedDots[i - 4].getX(),
calculatedDots[i - 4].getY())) / 24;
        double newX = calculatedDots[i - 1].getX() + step;
        calculatedDots[i] = new Dot(newX, newY);
    }
    return calculatedDots;
}
```

Примеры и рез-ты работы:

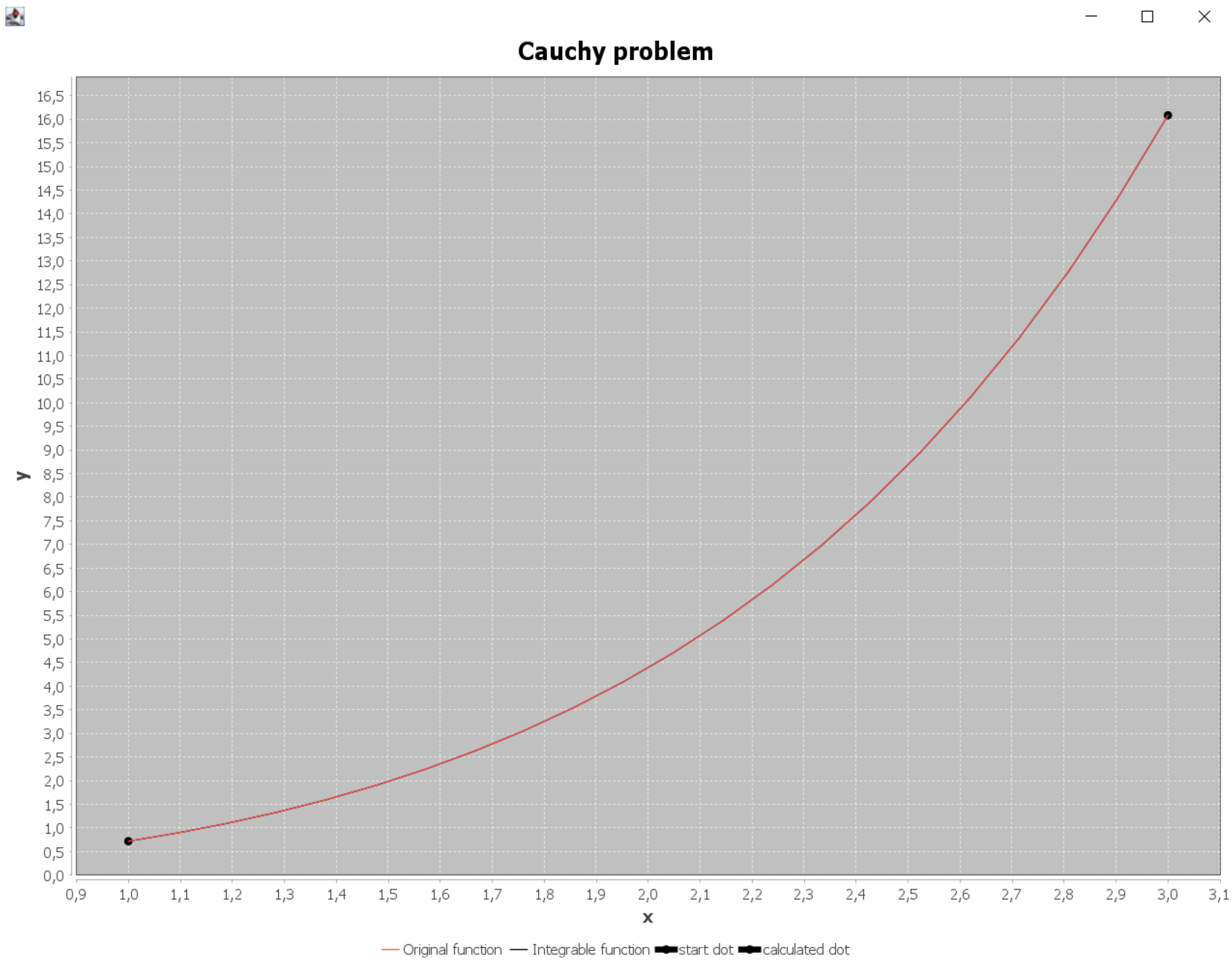
```
Выберите функцию:
1) y' = cos(x),
2) y' = x + y,
3) y' = (x * y)/2
1
Введите начальную точку:
0
Введите значение функции в начальной точке:
0
Введите конечную точку:
1.57
Введите точность:
0.001
```

Результат:



```
Выберите функцию:
1) y' = cos(x),
2) y' = x + y,
3) y' = (x * y)/2
2
Введите начальную точку:
1
Введите значение функции в начальной точке:
0.718
Введите конечную точку:
3
Введите точность:
0.0001
```

Результат:



Вывод:

Я рассмотрел численное интегрирование на примере методов Адамса и Рунге – Кутта.

Метод Рунге – Кутта является одношаговым и использует одну предыдущую точку для вычисления текущей, что делает его более простым в реализации. Этот метод по этой же причине занимает меньше памяти.

Метод Адамса же является многошаговым и требует несколько предыдущих точек для вычисления текущей, что, по сравнению с методом Рунге – Кутта делает его более сложным в реализации и затратным по памяти, но этот метод, в отличие от одношаговых, более стабильный и сохраняет высокую точность при увеличении шага интегрирования.