Университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №5
по дисциплине “Вычислительная математика”
Вариант: Метод Адамса

Выполнила:
Шевченко Д. П.,
группа P3230

Преподаватель:
Перл О.В.

Санкт-Петербург
2024

Оглавление

Описание численного метода

Метод Адамса — это многошаговый метод для интегрирования дифференциальных уравнений.
У нас есть уравнение $y' = f(x, y)$
Сначала нужно определить шаг h

$$h = \frac{b-a}{\varepsilon}$$

где a, b границы интервала
Определим $x = a + i * \varepsilon$
i = 0, … , h
Для решения методом Адамса нужно более 1 начальной точки (которая дана), для этого воспользуемся методом Рунге-Кутты и рассчитаем коэффициенты k_1, …,k_4 по формулам:

$$k_1 = \varepsilon * f(x_i, y_i)$$

$$k_2 = \varepsilon * f\left(x_i + \frac{\varepsilon}{2}, y_i + \frac{k_1}{2}\right)$$

$$k_3 = \varepsilon * f\left(x_i + \frac{\varepsilon}{2}, y_i + \frac{k_2}{2}\right)$$

$$k_4 = \varepsilon * f(x_i + \varepsilon, y_i + k_3)$$

Далее вычисляем следующие значения y по формуле:

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

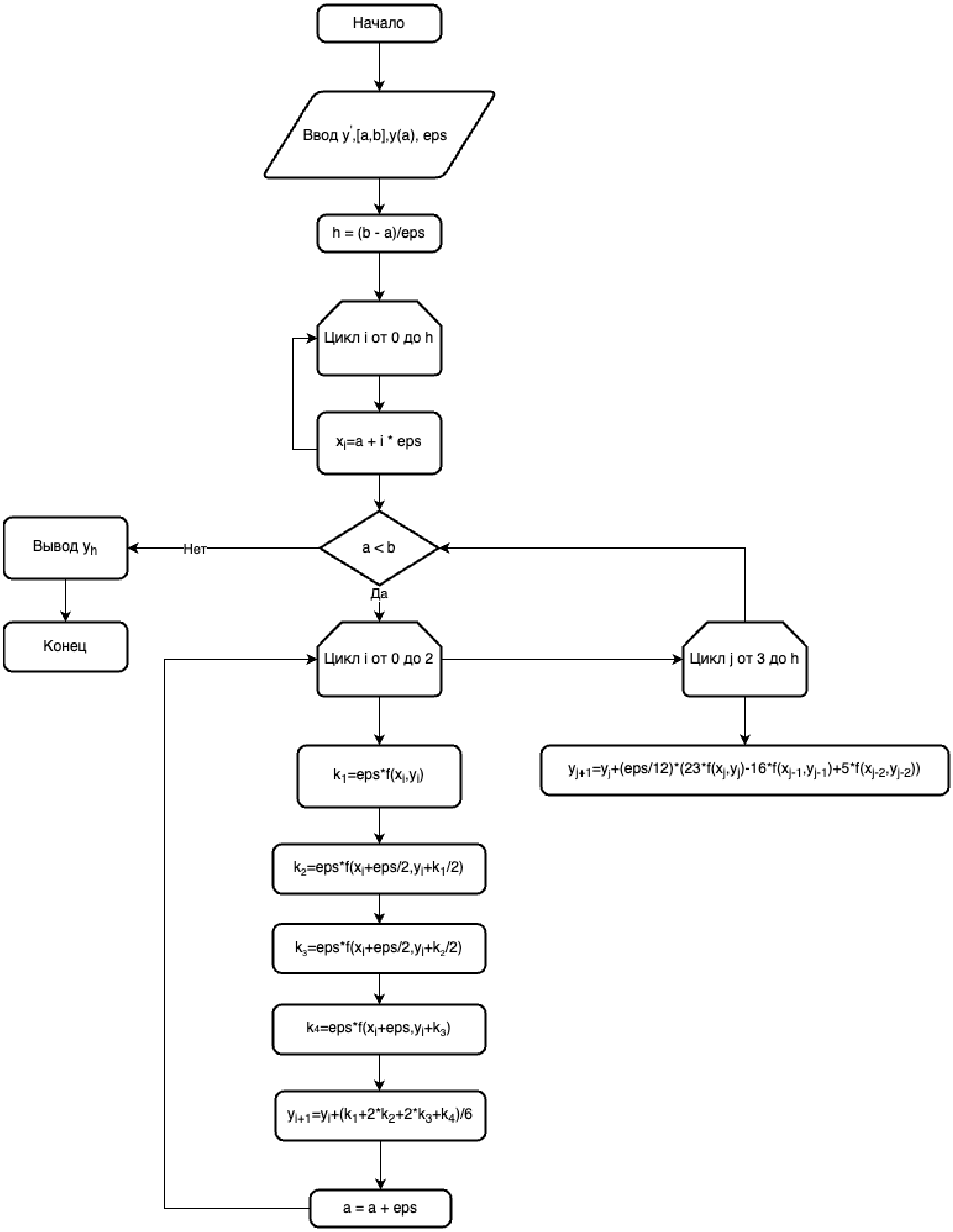
i = 0, …,2

Увеличиваем a на ε, после чего считаем следующие значения y для остальных точек по формуле:

$$y_{j+1} = y_j + \frac{\varepsilon}{12} * (23 * k_1 - 16 * k_2 + 5 * k_3)$$

j = 3, …, h
Выполняем алгоритм до тех пор, пока a не станет больше b.

Блок схема


Начало
Ввод y',[a,b],y(a), eps
h = (b - a)/eps
Цикл i от 0 до h
x_i=a + i * eps
a < b
Нет
Вывод y_h
Конец
Да
Цикл i от 0 до 2
Цикл j от 3 до h
y_j+1=y_j+(eps/12)*(23*f(x_j,y_j)-16*f(x_j-1,y_j-1)+5*f(x_j-2,y_j-2))
k_1=eps*f(x_i,y_i)
k_2=eps*f(x_i+eps/2,y_i+k_1/2)
k_3=eps*f(x_i+eps/2,y_i+k_2/2)
k4=eps*f(x_i+eps,y_i+k_3)
y_i+1=y_i+(k_1+2*k_2+2*k_3+k_4)/6
a = a + eps

Код

```
def solveByAdams(f, epsilon, a, y_a, b):
    try:
        step_count = int((b - a) / epsilon)
        function = Result.get_function(f)
        x = []
        for i in range(step_count + 1):
            x.append(a + i * epsilon)
        y = [y_a] * (step_count + 1)
        while a < b:
            for i in range(3):
                k1 = epsilon * function(x[i], y[i])
                k2 = epsilon * function(x[i] + epsilon / 2, y[i] + k1 / 2)
                k3 = epsilon * function(x[i] + epsilon / 2, y[i] + k2 / 2)
                k4 = epsilon * function(x[i] + epsilon, y[i] + k3)
                y[i + 1] = y[i] + (k1 + 2 * k2 + 2 * k3 + k4) / 6
                a += epsilon
            for j in range(4, step_count):
                y[j + 1] = y[j] + (epsilon / 12) * (
                            23 * function(x[j], y[j]) - 16 * function(x[j - 
1], y[j - 1]) + 5 * function(x[j - 2], y[j - 2]))
        return y[step_count]
    except:
        None
```

Когда я загружала код на Code’n’Test, не заметила, что случайно цикл for j in range(4, step_count) вложила в цикл для вычисления k. Ошибки это не несет, лишь увеличивает количество одинаковых вычислений. Более корректный код представлен выше.

Примеры работы программы:

1) Корректный вывод

```
2
0.001
2
3
4
60.015704404645746
```

2) Вывод, когда a> b. Нужно было добавить обработку именно этого случая

```
2
0.001
3
2
1
None
```

3) Значение, посчитанное по первым трем точкам

```
2
0.1
1
1
1.3
1.1882717990261118
```

4) Выбрана слишком большая ε

```
3
0.1
1
1
1.02
None
```

5) Корректный вывод для 4 функции

```
4
0.01
1
2
4
72.95181141358091
```

Выводы

1) Примеры работы программы на странице 6
2) В сравнении с методом Эйлера, метод Адамса требует запоминания предыдущих значений функции, тем самым нужно иметь больше памяти, но зато является более точным и затрачивает меньше времени для выполнения. По сравнению с методом Рунге-Кутты, который является одношаговым, у метода Адамса более высокая погрешность при больших шагах интегрирования. Но одношаговые методы более трудоемкие, так как на каждом шаге несколько раз вычисляется правая часть ДУ, а метод Адамса более сложен в реализации.
3) Анализ применимости метода: эффективен для сложных дифференциальных уравнений, отлично подходит для решения задачи Коши с разрывными правыми частями, для ДУ нейтрального типа. Так как это метод высокого порядка, позволяет достичь более высокого уровня точности за меньшее количество итераций. Он может обрабатывать широкий спектр ОДУ, в том числе с переменными коэффициентами.
4) Анализ алгоритмической сложности:
   Сложность метода Адамса может быть представлена как O(n) – где n количество шагов(h), необходимых для нужной точности вычислений.
5) Анализ численной ошибки:
   Метод Адамса дает погрешность $O(h^k)$, где k – порядок метода. Точность можно повысить путем уменьшения шага h, но за счет этого вырастет объем вычислений.

   Таким образом, метод Адамса является хорошим инструментом для вычисления задач Коши и ОДУ. Но для применения данного метода, требуется более одной начальной точки, в результате чего нужно применять другие методы для вычисления нескольких начальных точек перед использованием метода Адамса. Многошаговый метод Адамса является более экономичным, чем одношаговые методы и не требует дополнительных вычислений, но при его использовании нельзя сразу же начать решать задачу без начальных точек.