**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №5

по дисциплине «**Вычислительная математика**»

Автор: Баянов Равиль Динарович

Факультет: ПИиКТ

Группа: P3234

Преподаватель: Перл О. В.

Изображение выглядит как черный, темнота

Автоматически созданное описание

Санкт-Петербург, 2024

Оглавление

[Описание метода 3](#_Toc160999556)

[Блок-схема 4](#_Toc160999557)

[Исходный код метода на языке программирования Python 5](#_Toc160999558)

[Примеры работы программы 6](#_Toc160999559)

[Вывод 9](#_Toc160999560)

# Описание метода

Метод Адамса-Башфорта – это явный, многошаговый метод для решения задачи Коши. Данный метод не является самостоятельным, так как нужно с помощью одношагового метода рассчитать разгоночные точки. Чаще всего для этой подзадачи используется метод Рунге-Кутты в силу того, что порядок точности одношагового метода должен быть выше или совпадать с порядком точности многошагового метода. Метод Адамса-Башфорта использует интерполяционный полином Лагранжа для аппроксимации решения задачи Коши с точностью до 4 порядка. Не будем рассматривать вычисление точек методом Рунге-Кутты сразу перейдём к вычислению точек в методе Адамса-Башфорта.

Пусть у нас заданы условия: и , для которых надо найти решение.

Экстраполяционная формула Адамса выглядит так:

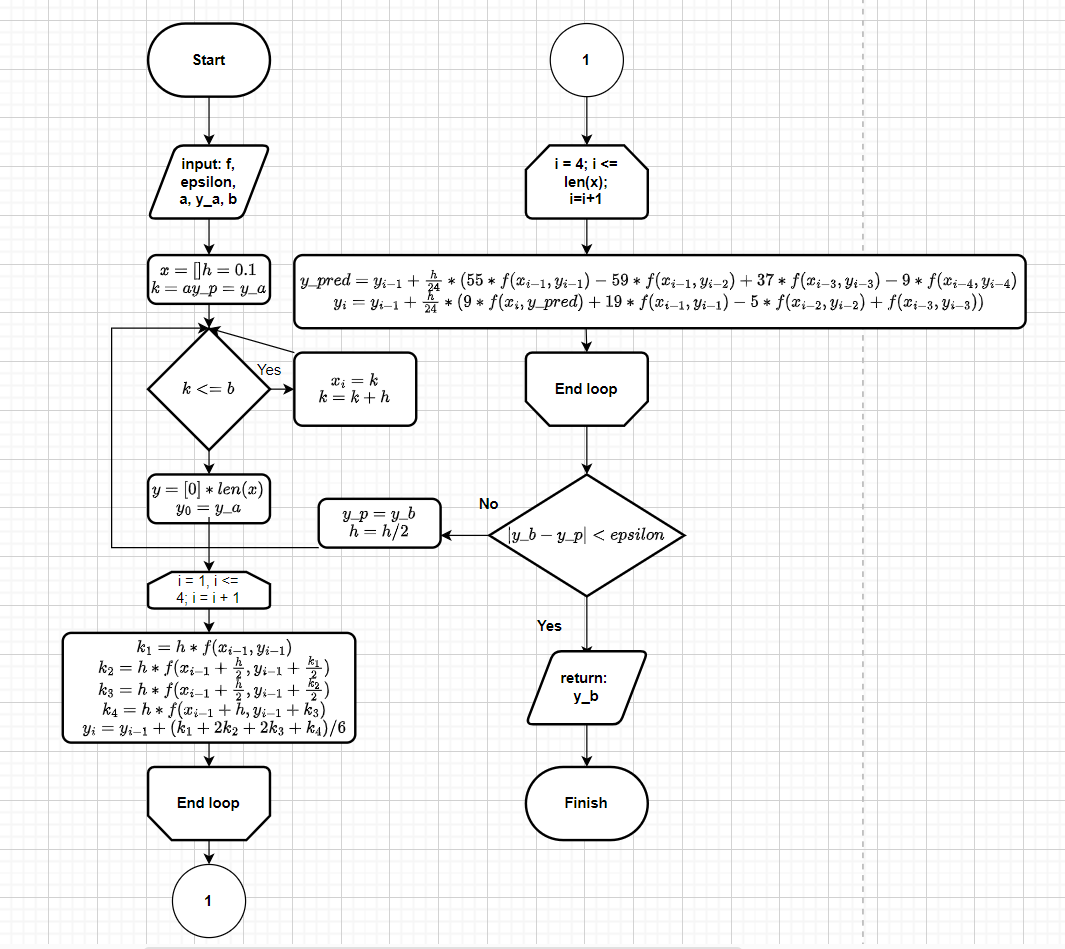
Формула метода Адамса-Башфорта:

Таким образом, получив первые 4 точки методом Рунге-Кутты, мы можем перейти к методу Адамса-Башфорта.

На практике неявный метод Адамса-Мультона используется вместе с явным методом Адамса\_Башфорта. Формула метода Адамса-Мультона выглядит так:

После вычисления начальных точек делаем прогноз для величин yi+1 с помощью явного метода Адамса-Башфорта. Затем корректируем величину yi+1 по формуле Адамса-Мультона. А теперь превратим этот метод в блок-схему и в код.

# Блок-схема



# Исходный код метода на языке программирования Python

|  |
| --- |
| 1. **class** Result: 3. **def** first\_function(x: float, y: float): 4. **return** math.sin(x) 6. **def** second\_function(x: float, y: float): 7. **return** (x \* y) / 2 9. **def** third\_function(x: float, y: float): 10. **return** y - (2 \* x) / y 12. **def** fourth\_function(x: float, y: float): 13. **return** x + y 15. **def** default\_function(x: float, y: float): 16. **return** 0.0 18. *# How to use this function:* 19. *# func = Result.get\_function(4)* 20. *# func(0.01)* 21. **def** get\_function(n: int): 22. **if** n == 1: 23. **return** Result.first\_function 24. **elif** n == 2: 25. **return** Result.second\_function 26. **elif** n == 3: 27. **return** Result.third\_function 28. **elif** n == 4: 29. **return** Result.fourth\_function 30. **else**: 31. **return** Result.default\_function 33. *#* 34. *# Complete the 'solveByAdams' function below.* 35. *#* 36. *# The function is expected to return a DOUBLE.* 37. *# The function accepts following parameters:* 38. *# 1. INTEGER f* 39. *# 2. DOUBLE epsilon* 40. *# 3. DOUBLE a* 41. *# 4. DOUBLE y\_a* 42. *# 5. DOUBLE b* 43. *#* 44. **def** runge\_kutta(f, x0, y0, h): 45. k1 = h \* f(x0, y0) 46. k2 = h \* f(x0 + h / 2, y0 + k1 / 2) 47. k3 = h \* f(x0 + h / 2, y0 + k2 / 2) 48. k4 = h \* f(x0 + h, y0 + k3) 49. **return** y0 + (k1 + 2 \* k2 + 2 \* k3 + k4) / 6 50. **def** solve(f, eps, a, y\_a, b, h): 51. x = [] 52. k = a 53. **while** k <= b: 54. x.append(k) 55. k += h 56. y = [0] \* len(x) 57. y[0] = y\_a 58. **for** i **in** range(1, 4): 59. y[i] = Result.runge\_kutta(f, x[0], y[0], h) 60. **for** i **in** range(4, len(x)): 61. y[i] = y[i - 1] + h / 24 \* ( 62. 9 \* f(x[i], y[i - 1] + h / 24 \* ( 63. 55 \* f(x[i - 1], y[i - 1]) - 59 \* f(x[i - 2], y[i - 2]) + 37 \* f(x[i - 3], y[i - 3]) - 9 \* f( 64. x[i - 4], y[i - 4]))) + 19 \* f(x[i - 1], y[i - 1]) - 5 \* f(x[i - 2], y[i - 2]) + f(x[i - 3], 65. y[i - 3])) 66. **return** y[-1] 68. **def** solveByAdams(f, eps, a, y\_a, b): 69. f = Result.get\_function(f) 70. h = 0.1 71. y\_p = y\_a 72. **while** True: 73. y\_b = Result.solve(f, eps, a, y\_a, b, h) 74. **if** (abs(y\_b - y\_p) < eps): 75. **return** y\_b 76. y\_p = y\_b 77. h /= 2 78. **return** y\_b |

# Примеры работы программы

* 1. Первая функция:

Изображение выглядит как текст, снимок экрана, Шрифт, часы

Автоматически созданное описание

Изображение выглядит как График, линия, диаграмма, текст

Автоматически созданное описание

* 1. Вторая функция:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

* 1. Третья функция:

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

* 1. Четвёртая функция:

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

* 1. Четвёртая функция:

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

* 1. Default функция:

Изображение выглядит как снимок экрана, Шрифт, типография

Автоматически созданное описание

# Вывод

Метод Адамса-Башфорта – это численный метод для решения обыкновенных дифференциальных уравнений. Он основан на идее использования интерполяции для вычисления приближённого значения функции в следующей точке. Этот метод является явным и многошаговым.

Сравним наш метод с другими:

* Точность: Метод Адамса-Башфорта имеет более высокий порядок точности, чем простые методы, такие как метод Эйлера. Например, метод Адамса-Башфорта четвертого порядка имеет порядок точности 4, в то время как метод Эйлера имеет порядок точности 1. Однако, метод Рунге-Кутты также имеет высокий порядок точности, и он может быть более точным, чем метод Адамса-Башфорта в некоторых случаях.
* Устойчивость: Метод Адамса-Башфорта может быть неустойчивым при решении жестких дифференциальных уравнений, в то время как метод Рунге-Кутты и некоторые другие методы более устойчивы. Это означает, что метод Адамса-Башфорта может давать неверные результаты при решении некоторых типов уравнений.
* Вычислительная сложность: Метод Адамса-Башфорта требует вычисления дополнительных значений функции, что может увеличить вычислительную сложность метода. Метод Эйлера проще и требует меньше вычислений, но он менее точен. Метод Рунге-Кутты требует большего количества вычислений, но он более точен и устойчив.

Метод Адамса-Башфорта может быть неустойчивым в некоторых случаях, но требует не очень много вычислений по сравнению с другими методами.

Данный метод не стоит применять для жёстких функций, так как метод крайне неустойчив. Также для метода Адамса-Башфорта требуется наличие начальных условий.

Для метода Адамса-Башфорта порядка k, количество арифметических операций, необходимых для вычисления одного шага, пропорционально k. Это означает, что вычислительная сложность одного шага метода Адамса-Башфорта порядка k равна O(k).

Общая вычислительная сложность метода Адамса-Башфорта зависит от количества шагов, необходимых для решения дифференциального уравнения. Если N - количество шагов, необходимых для решения уравнения, то общая вычислительная сложность метода Адамса-Башфорта порядка k будет равна O(Nk).

Ошибка метода сильно зависит от начальных условий, от гладкости решения и от поведения дифференциального уравнения. В целом, метод Адамса-Башфорта имеет относительно небольшую численную ошибку за счёт высокого порядка точности.

Таким образом, метод Адамса-Башфорта один из многих методов для решения обыкновенных дифференциальных уравнений. Он очень часто используется вместе с методом Адамса-Мультона для корректировки ответа. Данный метод очень эффективен и точен.