Вычислительная математика

Раевский Григорий, группа 3.2

16.04.2024

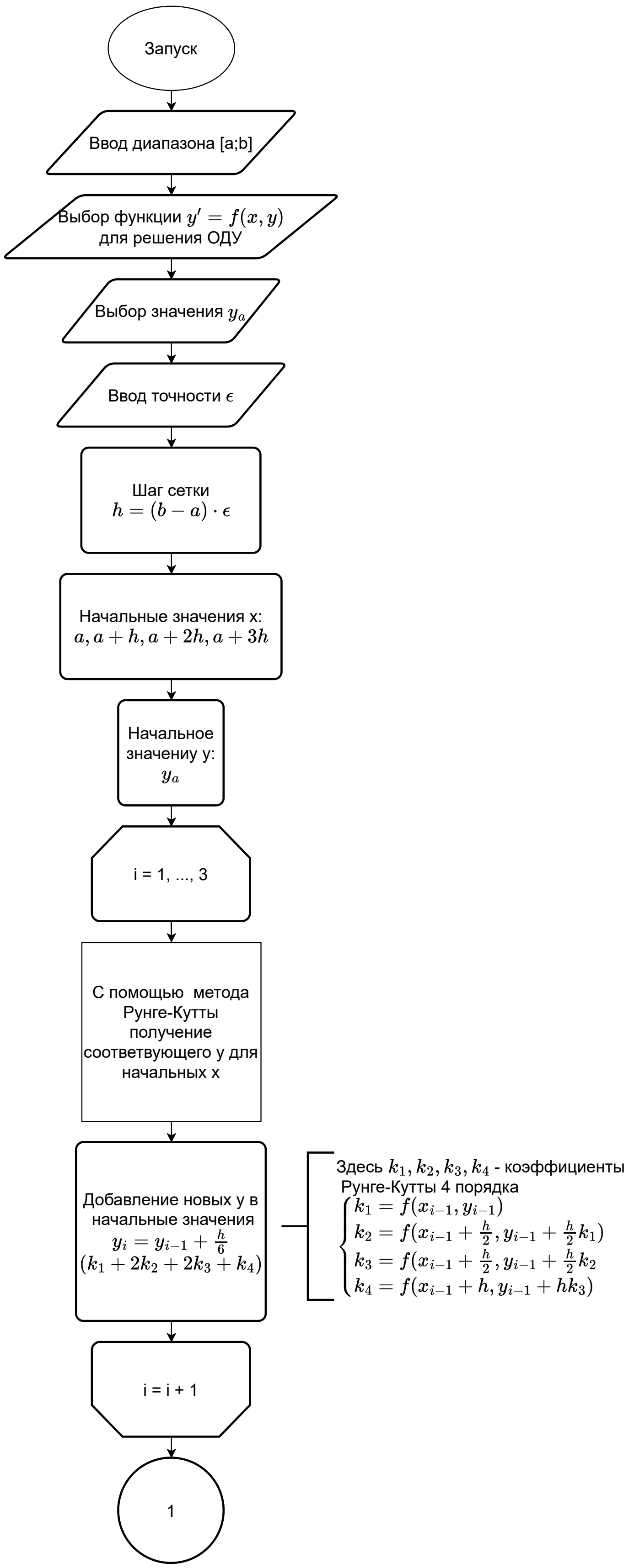
Отчет по лабораторной работе 5

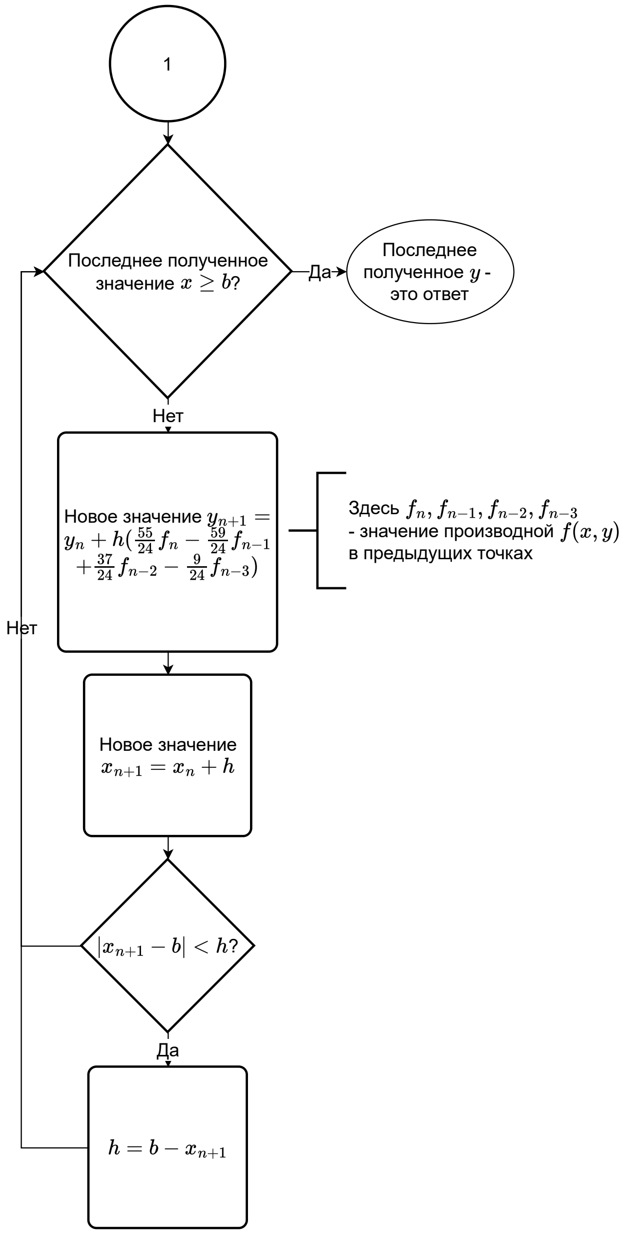
Содержание

Описание численного метода

Решение ОДУ с помощью метода Адамса — численный метод для решения дифференциальных уравнений и поиска значения функции на промежутке [a;b]. Метод Адамса основан на многошаговом приближении. Он использует информацию из предыдущих точек для предсказания значений в будущих. Чтобы метод смог работать, ему требуется несколько "разгонных"точек. В данном случае они находятся с помощью метода Рунге-Кутты 4 порядка. Формула Адаса-Башфорта, которая использует линейную комбинацию предыдущих значений производной функции $f(x, y)$ и констант, дает приближённое значение y на следующем шаге. Если разница между последовательными приближениями превышает точность ϵ, то происходит ее корректировка

Диаграмма


Запуск
Ввод диапазона [a;b]
Выбор функции $y' = f(x, y)$ для решения ОДУ
Выбор значения y_a
Ввод точности ϵ
Шаг сетки $h = (b - a) \cdot \epsilon$
Начальные значения x: $a, a + h, a + 2h, a + 3h$
Начальное значениу y: y_a
i = 1, ..., 3
С помощью метода Рунге-Кутты получение соответвующего y для начальных x
Добавление новых y в начальные значения $y_i = y_{i-1} + \frac{h}{6}$ $(k_1 + 2k_2 + 2k_3 + k_4)$
Здесь k_1, k_2, k_3, k_4 - коэффициенты Рунге-Кутты 4 порядка
$\begin{cases} k_1 = f(x_{i-1}, y_{i-1}) \\ k_2 = f(x_{i-1} + \frac{h}{2}, y_{i-1} + \frac{h}{2}k_1) \\ k_3 = f(x_{i-1} + \frac{h}{2}, y_{i-1} + \frac{h}{2}k_2 \\ k_4 = f(x_{i-1} + h, y_{i-1} + hk_3) \end{cases}$
i = i + 1
1


1
Последнее полученное значение $x \geq b$?
Да
Последнее полученное y - это ответ
Нет
Новое значение $y_{n+1} = y_n + h(\frac{55}{24}f_n - \frac{59}{24}f_{n-1} + \frac{37}{24}f_{n-2} - \frac{9}{24}f_{n-3})$
Здесь $f_n, f_{n-1}, f_{n-2}, f_{n-3}$ - значение производной $f(x,y)$ в предыдущих точках
Нет
Новое значение $x_{n+1} = x_n + h$
$|x_{n+1} - b| < h$?
Да
$h = b - x_{n+1}$

Листинг кода

```
def rungeKutta(func, x, y, h):
    k1 = func(x, y)
    k2 = func(x + 0.5 * h, y + 0.5 * k1 * h)
    k3 = func(x + 0.5 * h, y + 0.5 * h * k2)
    k4 = func(x + h, y + h * k3)
    return y + (h / 6) * (k1 + 2 * k2 + 2 * k3 + k4)

def solveByAdams(f, epsilon, a, y_a, b):
    if a > b:
        raise ValueError("Interval is incorrect, solution is not possible")

    func = Result.get_function(f)
    h = (b - a) * epsilon

    x_val = [a + i * h for i in range(4)]
    y_val = [y_a]
    for i in range(3):
        y_val.append(Result.rungeKutta(func, x_val[i], y_val[i], h))

    while x_val[-1] < b:

        y_next = y_val[-1] + (h / 24) * (
            55 * func(x_val[-1], y_val[-1])
            - 59 * func(x_val[-2], y_val[-2])
            + 37 * func(x_val[-3], y_val[-3])
            - 9 * func(x_val[-4], y_val[-4])
        )
        x_next = x_val[-1] + h

        y_val.append(y_next), x_val.append(x_next)
        if abs(x_val[-1] - b) < h:
            h = b - x_val[-1]

    return y_val[-1]
```

Примеры работы программы

Стандартный случай

stdin:

```
1
0.001
0
0
3.14
```

stdout:

```
1.999998731659757
```

Метод корректно отрабатывает и приближается к 2, так как ОДУ $\sin x$ с начальным условием $y(0) = 0$ дает $y(x) = 2 - \cos x$ на интервале от 0 до π.

Разрыв

stdin:

```
3
0.01
0
0
1
```

stdout:

```
---
```

Метод Адамса не применим, так как функция не определена для $y = 0$.

Малый интервал

stdin:

```
4
0.0001
1
1
1.001
```

stdout:

```
1.0020015004989515
```

$y(b)$ очень близко к $y(a)$ из-за малого интервала, но метод Адамса отработал корректно.

Большой интервал

stdin:

```
2
0.01
0
1
10
```

stdout:

```
64153715375.41474
```

Из-за большого интервала шаг h может быть слишком большим, что может повлиять на точность. В этом случае метод Адамса выполнит большое количество итераций, прежде чем найдет результат.

Высокая точность

stdin:

```
4
0.000001
0
1
1
```

stdout:

```
3.4365636568818
```

Метод Адамса отработал корректно, но из-за высокой точности ему потребовалось большое количество итераций (999998), прежде чем он нашел решение.

Обратный интервал

stdin:

```
1
0.01
2
4
1
```

stdout:

```
ValueError: Interval is incorrect, solution is not possible
```

Метод Адамса не отработал корректно, так как $a > b (2 > 1)$.

Выводы

Программа работает стабильно на данных, если входные данные корректны (например, нет интервала с $a > b$ или имеется разрыв) и корректно находит приближенное значение функции в точке b с заданной точностью. Однако программа будет испытывать трудности, когда точность слишком высокая или интервал $[a; b]$ слишком мал.

По сравнению с методом Рунге-Кутты, метод Адамса (в данном случае, один из его типов — метод Адамса-Башфорта) является более точным на длинных интервалах, так как использует информацию из нескольких предыдущих шагов. Однако метод Рунге-Кутты является более универсальным и простым в реализации. Он так же не требует точек "разгона"для начала работы. Метод Адамса является особенно эффективным на гладких функциях, предсказывая и корректируя значения в процессе работы.Метод Милна же является более чувствительным к начальным условиям, он так же менее устойчив в случае работы со сложными функциями. Улучшенный метод эйлера хоть и требует меньших вычислительных ресурсов, может быть менее точным на длинных интервалах из-за накопления численной ошибки по сравнению с методом Адамса.

На точность работы метода Адамса в первую очередь влияет расчет "разгонных"точек с помощью метода Рунге-Кутты. Однако она может незначительно накапливаться из-за неточности в вычислении каждого нового значения функции в точке. Однако он хорошо подходит для различных задач, так как остается достаточно точным и обладает невысокой алгоритмической сложностью, $O(n)$, где n - число итераций цикла, обратно зависит от ϵ.