НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
Факультет Программной инженерии и компьютерных технологий
Направление: Нейротехнологии и программная инженерия

Дисциплина: Вычислительная математика

Лабораторная работа № 4

“Метод Симпсона”

Выполнил студент
Рязанов Демид Витальевич
Группа P3221
Преподаватель: Перл Ольга Вячеславовна

г. Санкт-Петербург
2024

Содержание

Описание метода

Метод Симпсона – итерационный алгоритм для вычисления интеграла. На каждой итерации метода получается более точное приближение решения, и при достижении определенной точности алгоритм завершает свою работу.

Алгоритм:

1) Задаются отрезок $[a,b]$, ϵ , и количество разбиений n (четное число).

2) Отрезок $[a,b]$ разбивается на n равных частей $[a,x_1],[x_1,x_2],\ldots,[x_{n-1},b]$, $x_0=a$, $x_n=b$, каждая длиной $h=\frac{(b-a)}{n}$

3) Для каждого x_i считается значение подынтегральной функции $f(x_i)$

4) Считается ответ по формуле Симпсона

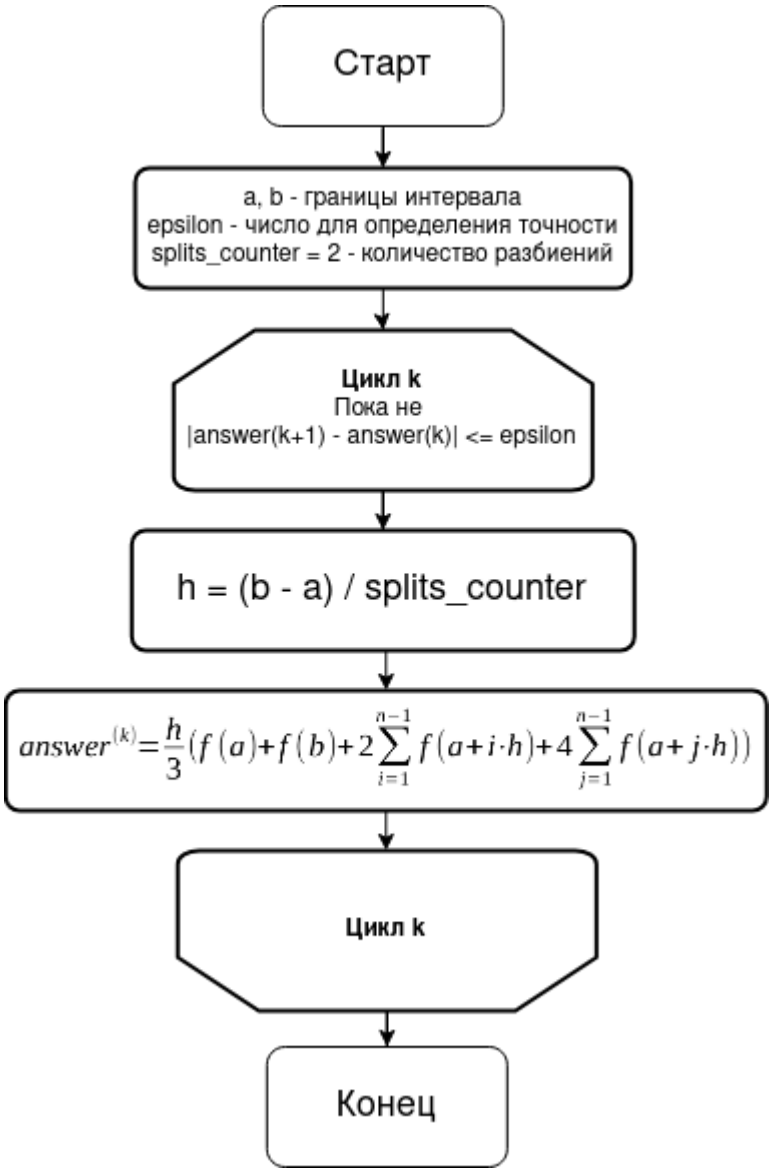
$$answer^{(k+1)}=\int_a^b f(x)dx=\frac{h}{3}(f(x_0)+f(x_n)+2\sum_{i=1}^{n-1} f(x_i)+4\sum_{j=1}^{n-1} f(x_j))$$

где $i=2t, t\in\mathbb{Z}$, $j=2p+1, p\in\mathbb{Z}$

5) Сравнивается с предыдущим ответом и, если $|answer^{(k)}-answer^{(k+1)}|\le\epsilon$, то поиск ответа завершается.

Иначе n увеличивается на 2 и повторяются пункты 2-4.

Блок-схема



$i=2t, j=2p+1 \qquad t, p \in \mathbb{Z}$

Исходный код

```
def calculate_integral(a, b, f, epsilon):
    function = Result.get_function(f)
    sign = 1
    if a > b:
        sign = -1


    def calculate_simpson_formula_for_n(n, left, right):
        h = float((right - left) / n)
        f_values = [function(left + i * h) for i in range(0, n + 1)]
        return h / 3 * (f_values[0] +
                        f_values[n] +
                        sum(x for i, x in enumerate(f_values[1:n]) if i % 2 == 0) * 2 +
                        sum(x for i, x in enumerate(f_values[1:n]) if i % 2 == 1) * 4)


    try:
        splits_counter = 2
        answer = calculate_simpson_formula_for_n(splits_counter, min(a, b),
max(a, b))
        while True:
            splits_counter *= 2
            new_answer = calculate_simpson_formula_for_n(splits_counter, min(a,
b), max(a, b))
            if abs(new_answer - answer) <= epsilon:
                return new_answer * sign
            answer = new_answer
    except:
        Result.has_discontinuity = True
        return 0
```

Примеры работы

Пример 1

Ввод	Вывод
0 1 3 0.01	2.3268280029296875

Пример 2

Ввод	Вывод
1 0 3 0.01	-2.3268280029296875

Пример 3

Ввод	Вывод
5 6 4 0.001	12.999471028645832

Пример 4

Ввод	Вывод
-3 3 1 0.01	Integrated function has discontinuity or does not defined in current interval

Пример 5

Ввод	Вывод
7 7 7 0.01	Function 7 not defined

Вывод

Метод Симпсона – итерационный метод, позволяющий вычислить интеграл. Суть метода в том, что подынтегральная функция заменяется интерполяционным полиномом. Лучше всего работает, когда интервал интегрирования небольшой. Не применим, когда интегрируемая функция имеет разрывы на интегрируемом участке. Сложность алгоритма O(n*k), где k – количество итераций. Этот метод имеет низкий порядок точности (максимальную степень полинома для которого метод даёт точное решение) – 3, это ниже чем у метода Гаусса и Чёбышева, но выше чем у методов прямоугольников и трапеций.