Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
Московский государственный технический университет имени Н.Э. Баумана

Факультет: Информатика и системы управления
Кафедра: Теоретическая информатика и компьютерные технологии

Лабораторная работа №2
по курсу «Численные методы»
«Сравнительный анализ методов численного интегрирования»

Выполнила:
студентка группы ИУ9-62Б
Самохвалова П. С.

Проверила:
Домрачева А. Б.

Москва, 2023

Цель:

Обосновать быстродействие одного из алгоритмов: метода центральных прямоугольников, метода трапеций, метода Симпсона по количеству шагов, необходимых для достижения заданной цели ε.

Постановка задачи:

Дано:

$$\int_a^b f(x)\,dx$$

Найти:

$$I^* \approx I$$

Описание алгоритмов:

Метод центральных прямоугольников:

$$I^* = h\sum_{i=1}^{n} f(x_{i-0.5})$$

Метод трапеций:

$$I^* = h(\frac{f(a)+f(b)}{2} + \sum_{i=1}^{n-1} f(x_i))$$

Метод Симпсона:

$$I^* \approx \frac{h}{6}\sum_{i=1}^{n}(f(x_{i-1}) + 4f(x_{i-0.5}) + f(x_i))$$

Вычисление интегралов с учётом погрешности:

$$R = \frac{I^*_{\frac{h}{2}} - I^*_h}{2^k - 1}$$

Для метода прямоугольников и метода трапеций $k = 2$, для метода Симпсона $k = 4$

R - уточнение по Ричардсону

$|R| \leq \varepsilon$ - правило Рунге

Тестовый пример:

Вариант 21

$$I = \int_{0.25}^{2} \frac{1}{x + x^2}\, dx$$

$$\varepsilon = 0.001$$

Листинг 1. Методы численного интегрирования

```
import math


def f(x):
    return 1 / (x + x ** 2)


def rectangles_method(f, a, b, n):
    h = (b - a) / n
    return h * sum(f(a + (i - 0.5) * h) for i in range(1, n + 1))


def trapezoid_method(f, a, b, n):
    h = (b - a) / n
    return h * ((f(a) + f(b)) / 2 + sum(f(a + i * h) for i in range(1, n)))


def Simpson_method(f, a, b, n):
    h = (b - a) / n
    return h / 6 * sum((f(a + (i - 1) * h) + 4 * f(a + (i - 0.5) * h) +
                        f(a + i * h)) for i in range(1, n + 1))


epsilon = 0.001
a = 0.25
b = 2
methods = [rectangles_method, trapezoid_method, Simpson_method]
methods_names = ["Rectangles method", "Trapezoid method", "Simpson_method"]
for i in range(len(methods)):
    if methods[i] == Simpson_method:
        k = 4
    else:
```

```
        k = 2
    n = 1
    integral = 0
    r = math.inf
    while abs(r) >= epsilon:
        n *= 2
        integral_last = integral
        integral = methods[i](f, a, b, n)
        r = (integral - integral_last) / (2 ** k - 1)
    print(methods_names[i])
    print(n)
    print(integral)
    print(integral + r)
    print()
```

Результаты работы:

$$I = \int_{0.25}^{2} \frac{1}{x + x^2}\, dx \approx 1,20397$$

$$\varepsilon = 0.001$$

Таблица результатов работы программы с использованием перечисленных выше методов:

Метод	n	I^*	$I^* + R$
Метод прямоугольников	64	1.203499650700365	1.203968762824301
Метод трапеций	64	1.2049199981264607	1.2039774368604925
Метод Симпсона	8	1.2048584701623828	1.2043443424223774

n - число частей, на которые разбивается отрезок интегрирования

Выводы:

В результате выполнения лабораторной работы были рассмотрены методы численного интегрирования: метод центральных прямоугольников, метод трапеций, метод Симпсона. Была написала реализация каждого из методов на языке программирования python. Также был проведён сравнительный анализ данных методов по количеству шагов, необходимых для достижения заданной цели ε. В результате работы программы было получено, что метод центральных прямоугольников и метод трапеций достигают заданной цели ε при разбиении отрезка интегрирования на 64 равные части, а метод Симпсона - при разбиении отрезка интегрирования на 8 равных частей. Можно сделать вывод, что метод Симпсона достигает заданной точности быстрее, чем метод центральных прямоугольников и метод трапеций. А метод центральных прямоугольников и метод трапеций достигают заданной цели ε за одинаковое количество шагов.