|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатики и систем управления

КАФЕДРА Теоретической информатики и компьютерных технологий

**Лабораторная работа № 2**

«Сравнительный анализ методов численного интегрирования»

по курсу «Численные методы»

Выполнила:

студент группы ИУ9-61Б

Яровикова Анастасия

Проверила:

Домрачева А. Б.

Москва, 2023

1. **Цель**

Целью данной работы является сравнение по быстродействию методов численного интегрирования:

1. Метод центральных прямоугольников
2. Метод трапеций
3. Метод Симпсона
4. **Постановка задачи**

**Дано:** Интеграл

где – подынтегральная функция, непрерывная на отрезке

**Найти:** Значение интеграла

При заданной точности

**Индивидуальный вариант:** ,

1. **Основные теоретические сведения**
   1. **Метод центральных прямоугольников**

Метод заключается в вычислении площади под графиком подынтегральной функции с помощью суммирования площадей прямоугольников, ширина которых определяется шагом разбиения (расстояние между узлами интегрирования), а высота – значением подынтегральной функции в узле интегрирования.

Пусть требуется определить значение интеграла функции на отрезке . Тогда отрезок разбивается на равных отрезков длиной . Получаем разбиение данного отрезка точками

Тогда приближенное значение интеграла на всем отрезке будет равно:

* 1. **Метод трапеций**

Метод заключается в вычислении площади под графиком подынтегральной функции с помощью суммирования площадей трапеций, высота которых определяется шагом разбиения (расстояние между узлами интегрирования), а высота – значением подынтегральной функции в узле интегрирования.

Пусть требуется определить значение интеграла функции на отрезке . Тогда отрезок разбивается на равных отрезков длиной . Получаем разбиение данного отрезка точками

Тогда приближенное значение интеграла на всем отрезке будет равно:

* 1. **Метод Симпсона**

Метод заключается в приближении функции на отрезке интерполяционным многочленом 2 степени функции

Тогда приближенное значение интеграла на всем отрезке будет равно:

* 1. **Уточнение значения интеграла по Ричардсону**

, где – порядок точности метода, – приближенное значение интеграла, вычисленного с помощью метода с шагом .

Для метода средних прямоугольников и метода трапеций .

Для метода Симпсона .

, где – некоторая константа, – шаг.

Считаем, что вычисления проводятся без вычислительной погрешности, тогда можно записать строгое равенство для шага

для шага

Из равенств получаем уточненное значение интеграла:

Где значение – уточнение по Ричардсону:

Данная величина используется для компенсации методологической погрешности численных методов интегрирования.  
 Чтобы построить процедуру приближенного вычисления интеграла с заданной точностью , используется правило Рунге:

1. **Реализация**

Листинг 1. Численное интегрирование

import math

eps = 0.01

def testFunc(x):

return x \* math.cos(x) \*\* 2

# return math.exp(x)

def richardsonFormula(I\_h, I\_h2, k):

return (I\_h - I\_h2) / (2 \*\* k - 1)

def rect(f, a, b, n):

h = (b - a) / n

s = sum(f(a + (i - 0.5) \* h) for i in range(1, n + 1))

return h \* s

def trap(f, a, b, n):

h = (b - a) / n

s = sum(f(a + i \* h) for i in range(1, n))

return h \* ((f(a) + f(b)) / 2 + s)

def simp(f, a, b, n):

h = (b - a) / n

s1 = sum(f(a + i \* h) for i in range(1, n))

s2 = sum(f(a + (i - 0.5) \* h) for i in range (1, n + 1))

s3 = sum(f(a + (i - 1) \* h) for i in range(1, n + 2))

s = s1 + 4 \* s2 + s3

return h / 6 \* s

def res(metd, k, a, b, f):

n = 1

R = 100

iter = 0

I\_h = 0

while not (abs(R) < eps):

n \*= 2

I\_h2 = I\_h

I\_h = metd(f, a, b, n)

R = richardsonFormula(I\_h, I\_h2, k)

iter += 1

print(f' iterations: {iter}')

print(f' res: {I\_h}')

print(f' res + Richardson: {I\_h + R}')

if \_\_name\_\_ == "\_\_main\_\_":

for i in range(1,5) :

print("\n-----------\nEPS = " + str(eps))

print("\nCentral rectangles method: ")

res(rect, 2, 0, math.pi, testFunc)

print("\nTrapezoids method: ")

res(trap, 2, 0, math.pi, testFunc)

print("\nSimpson`s method: ")

res(simp, 4, 0, math.pi, testFunc)

eps = eps / 10

1. **Результаты**

Для тестирования выбран интеграл:

В качестве были выбраны следующие значения:

Таблица 1 – Результаты программы

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Кол-во итераций | Значение интеграла | Значение интеграла с уточнением по Ричардсону |
|  | | | |
| Метод центральных прямоугольников | |  |  | | --- | --- | |  |  |   4 | 0.36725322733841864 | 0.36684480994171986 |
| Метод трапеций | 4 | 0.36604553395806505 | 0.3668565158148817 |
| Метод Симпсона | 2 | 0.3669524667895847 | 0.3668393717934952 |
|  | | | |
| Метод центральных прямоугольников | 6 | 0.36687538006255127 | 0.3668502538890432 |
| Метод трапеций | 6 | 0.36680006961565864 | 0.36685029927146423 |
| Метод Симпсона | 3 | 0.3668565158148816 | 0.3668501190832347 |
|  | | | |
| Метод центральных прямоугольников | 7 | 0.366856550324328 | 0.36685027374492024 |
| Метод трапеций | 8 | 0.3668471375817167 | 0.3668502751625873 |
| Метод Симпсона | 3 | 0.3668565158148816 | 0.3668501190832347 |

1. **Вывод**

В ходе выполнения лабораторной работы были рассмотрены 3 метода численного интегрирования: метод центральных прямоугольников, метод трапеций и метод Симпсона. Данные методы были реализованы на языке программирования Python.

Самым точным среди рассмотренных трех методов оказался метод Симпсона.

Анализируя оставшиеся два метода, приходим к выводу, что метод средних прямоугольников точнее метода трапеций.