Министерство образования и науки 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 Информатика и системы управления (ИУ)

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

ЛАБОРАТОРНАЯ РАБОТА №5
« Построение и программная реализация алгоритмов численного интегрирования »

Студент группы ИУ7-41Б
Костев Дмитрий

Москва 2021

Цель работы

Получение навыков построения алгоритма вычисления двукратного интеграла с использованием квадратурных формул Гаусса и Симпсона.

Исходные данные

1. Количество узлов сетки N, M;
2. Значение параметра τ;
3. Методы для направлений при последовательном интегрировании.

Код программы

```
# Лабораторная работа №5
```

// main.py

```python
from math import pi

import plot
import integrator as inter


def main():
    sc = [0.05, 0.05, 10.0]

    ns, ms = [], []
    md1s, md2s = [], []
    ints = []

    end = '0'
    while end == '0':
        ns.append(int(input("Input N: ")))
        ms.append(int(input("Input M: ")))

        p = float(input("Enter parameter (tao): "))

        md1s.append(
            int(input("Outer integration mode (0 - Gauss, 1 - Simpson): ")))

        md2s.append(
            int(input("Inner integration mode (0 - Gauss, 1 - Simpson): ")))

        lm = [[0, pi / 2], [0, pi / 2]]

        ints.append(
            inter.Integrator(lm, [ns[-1], ms[-1]], [md1s[-1], md2s[-1]]))

        print("Result with {:.2f} as "
              "a parameter is {:.7f}".format(p, ints[-1](p)))

        end = input("Stop execution?: ")

    plot.plot(ints, sc, ns, ms, md1s, md2s)


if __name__ == "__main__":
    main()
```

// integrator.py

```
from math import cos, sin, exp, pi
from scipy.special import roots_legendre
from typing import Callable as Call


class Integrator(object):
    def __init__(self, lm: list[list[float]], n: list[int], fn: list[int]):
        self.lm = lm
        self.n = n
        self.f1 = Integrator.simpson if (fn[0]) else Integrator.gauss
        self.f2 = Integrator.simpson if (fn[1]) else Integrator.gauss

    def __call__(self, p: float) -> float:
        f = Integrator.__integrated(p)

        inner = lambda x: self.f2(
            lambda val1: f(x, val1),
            self.lm[1][0],
            self.lm[1][1],
            self.n[1])
        integ = lambda: self.f1(
            inner,
            self.lm[0][0],
            self.lm[0][1],
            self.n[0])

        return integ()

    @staticmethod
    def __integrated(p: float) -> Call[[float, float], float]:
        t = lambda x, y: 2 * cos(x) / (1 - sin(x) ** 2 * cos(y) ** 2)
        return lambda x, y: 4 / pi * (1 - exp(-p * t(x, y))) * cos(x) * sin(x)

    @staticmethod
    def simpson(f: Call[[float], float], a: float, b: float,
                n: int) -> float:
        if n < 3 or n % 2 == 0:
            raise Exception("Sorry, wrong n value")

        h = (b - a) / (n - 1.0)
        x = a
        res = 0.0

        for i in range((n - 1) // 2):
            res += f(x) + 4 * f(x + h) + f(x + 2 * h)
            x += 2 * h

        return res * h / 3

    @staticmethod
    def gauss(f: Call[[float], float], a: float, b: float,
              n: int) -> float:
        def p2v(p: float, c: float, d: float) -> float:
            return (d + c) / 2 + (d - c) * p / 2

        x, w = roots_legendre(n)
```

```
        return sum([(b - a) / 2 * w[i] * f(p2v(x[i], a, b)) for i in range(n)])
```

// plot.py

```
import matplotlib.pyplot as plt


def gen_label(n, m, md1, md2) -> str:
    f1s = "Gauss" if md1 == 0 else "Simpson"
    f2s = "Gauss" if md2 == 0 else "Simpson"

    return "N = {:}, M = {:}, Methods = {:}-{:}".format(n, m, f1s, f2s)


def plot(fs, sc, ns, ms, md1s, md2s):
    plt.clf()

    plt.title("Integration using meansquare method")
    plt.xlabel("Tao")
    plt.ylabel("Result")
    plt.grid(which='minor', color='k', linestyle=':')
    plt.grid(which='major', color='k')

    for i in range(len(fs)):
        x, y = [], []
        j = sc[0]
        while j < sc[2]:
            x.append(j)
            y.append(fs[i](j))
            j += sc[1]

        plt.plot(x, y, label=gen_label(ns[i], ms[i], md1s[i], md2s[i]))

    plt.legend()
    plt.savefig('points.png')
    plt.show()
```

Результат работы программы

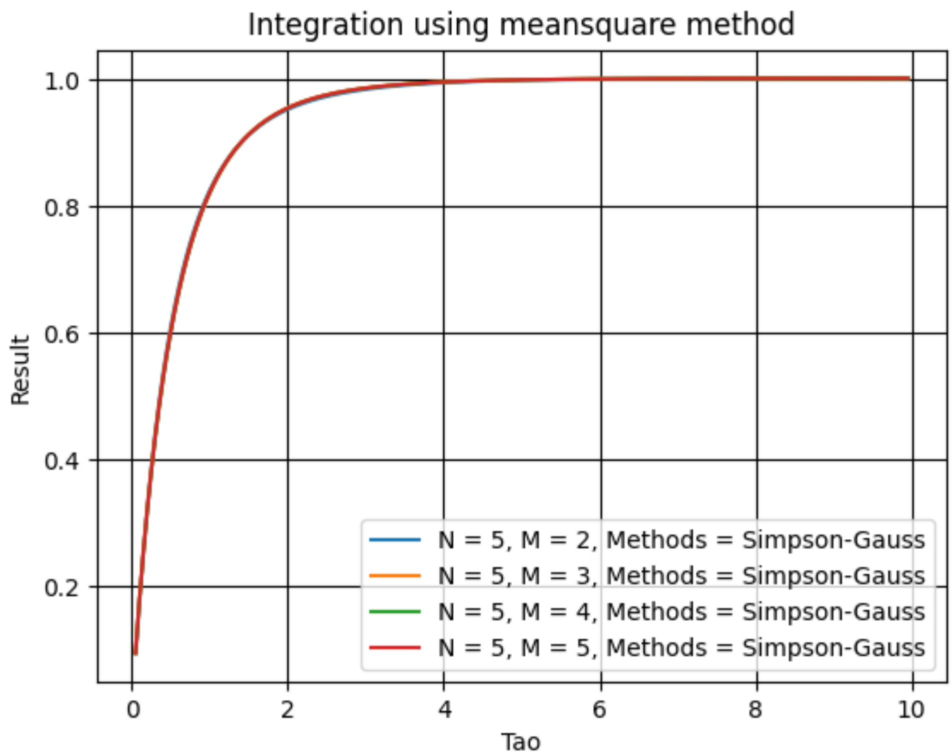
Алгоритм вычисления n корней полинома Лежандра n-ой степени
Все корни полинома лежат на интервале [-1, 1]. При этом стоит заметить, что интервалы [-1, 0] и [0, 1] – симметричны, так что при поиске корней достаточно рассмотреть интервал [0, 1].
Корни полинома можно вычислить итеративно по методу Ньютона

$$x_i^{(k+1)} = x_i^k - \frac{P_n(x_i)^{(k)}}{P_n'(x_i)^{(k)}}$$

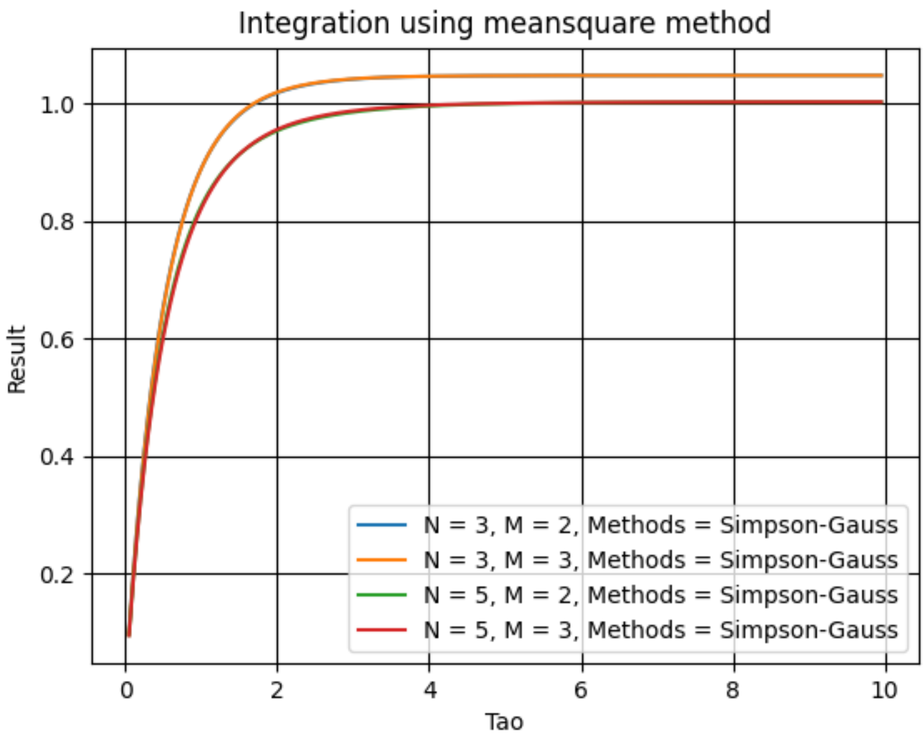
причем начальное приближение для i-го корня берется по формуле:

$$x_i^{(0)} = cos[\frac{\pi(4i-1)}{4n+2}]$$

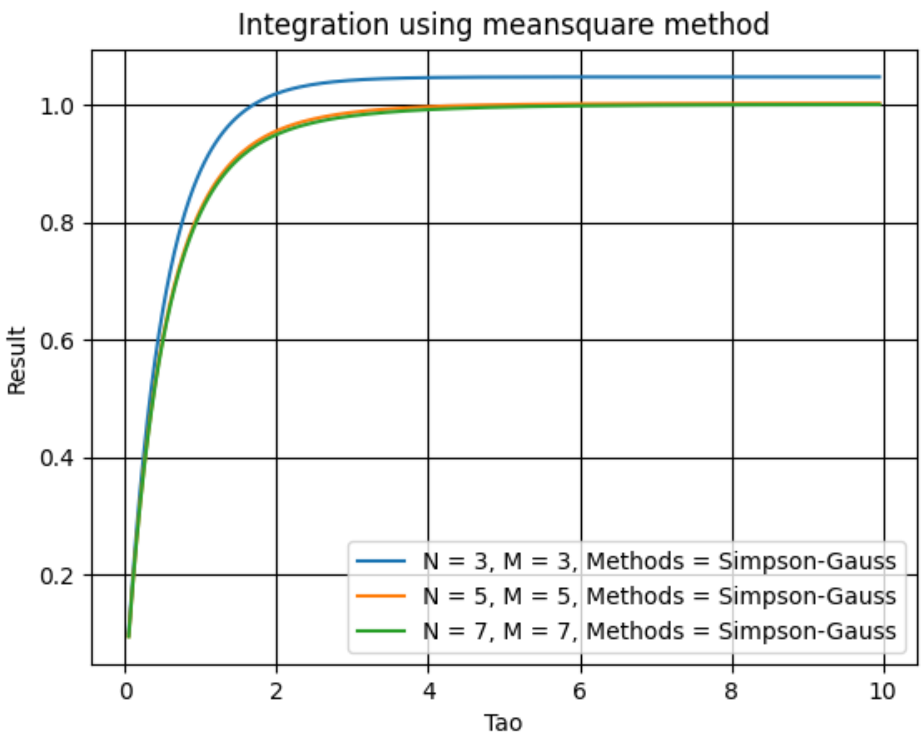
Влияние количества выбираемых узлов сетки по каждому направлению на точность расчетов
При использовании метода Симпсона в качестве «внешнего» метода интегрирования и при задании для него 5 узлов, метод Гаусса с различным количеством узлов будет давать одни и те же результаты.



Если для метода Симпсона задать меньшее количество узлов, получится расхождение с физическим смыслом - больший вклад будет вносить метод, являющийся «внешним».



Все измерения проводились для параметра $\tau = 1$.
График зависимости $\epsilon(\tau)$



Все измерения проводились для параметра $\tau = 1$ Как видно из графика, оптимальное значение достигнуто на сетке 5×5.

Ответы на вопросы

1. В каких ситуациях теоретический порядок квадратурных формул численного интегрирования не достигается?

Теоретический порядок квадратурных формул численного интегрирования не достигается в ситуациях, когда подынтегральная функция не имеет соответствующих производных. Порядок точности равен номеру последней существующей производной.

2. Построить формулу Гаусса численного интегрирования при одном узле.

$$\sum_{i=1}^{n} A_i = 2 \quad P_1(x) = x \Rightarrow x = 0$$

$$\int_a^b f(x)dx = \frac{b-a}{2} 2f(\frac{b+a}{2} + \frac{b-a}{2} \cdot 0) = (b-a)f(\frac{b+a}{2})$$

3. Построить формулу Гаусса численного интегрирования при двух узлах.

$$P_2(x) = \frac{1}{2}(3x^2 - 1) \ \Rightarrow x = \pm\frac{1}{\sqrt{3}}$$

$$\begin{cases} A_1 + A_2 = 2 \\ -\frac{1}{\sqrt{3}}A_1 + \frac{1}{\sqrt{3}}A_2 = 0 \end{cases} \Rightarrow A_2 = A_1 = 1$$

$$\int_{-1}^{1} f(f)df = f(-\frac{1}{\sqrt{3}}) + f(\frac{1}{\sqrt{3}})$$

$$\int_a^b f(x)dx = \frac{b-a}{2}(f(\frac{b+a}{2} - \frac{b-a}{2} \cdot \frac{1}{\sqrt{3}}) + f(\frac{b+a}{2} + \frac{b-a}{2} \cdot \frac{1}{\sqrt{3}}))$$

4. Получить обобщенную кубатурную формулу, на основе методе трапеций, с тремя узлами на каждом направлении.

$\int_c^d \int_a^b f(x,y)dxdy = \int_a^b dx \int_c^d f(x,y)dy = \int_a^b F(x)dx = h_x(\frac{1}{2}F_0 + F_1 + \frac{1}{2}F_2) = h_x h_y(\frac{1}{2}(\frac{1}{2}f(x_0,y_0) + f(x_0,y_1) + \frac{1}{2}f(x_0,y_2)) + \frac{1}{2}f(x_1,y_0) + f(x_1,y_1) + \frac{1}{2}f(x_1,y_2) + \frac{1}{2}(\frac{1}{2}f(x_2,y_0) + f(x_2,y_1) + \frac{1}{2}f(x_2,y_2)))$

$h_x = \frac{b-a}{2}, \ \ h_y = \frac{d-c}{2}$