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 «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 5

Тема Построение и программная реализация алгоритмов численного интегрирования.

Студент Алахов А.Г.

Группа ИУ7-42Б

Оценка (баллы) _______________

Преподаватель Градов В.М.

Москва.
2021 г

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

1 Исходные данные

Двукратный интеграл при фиксированном значении параметра τ:

$$\varepsilon(\tau) = \frac{4}{\pi} \int_0^{\pi/2} d\varphi \int_0^{\pi/2} [1 - \exp(-\tau \frac{l}{R})] \cos\theta \sin\theta \, d\theta ,$$

где $$\frac{l}{R} = \frac{2\cos\theta}{1 - \sin^2\theta \cos^2\varphi},$$

θ, φ - углы сферических координат.

Применяется метод последовательного интегрирования. По одному направлению использовать формулу Гаусса, а по другому - формулу Симпсона.

2 Код программы

Код программы представлен на листингах 1-2.

Листинг 1. functions.py

```
from math import fabs, sin, cos, pi, exp, sqrt
import numpy as np


def polynomial_func(polynomial, x):
    y = 0
    for i in range(len(polynomial)):
        y += polynomial[i] * x ** i
    return y


def dichotomy_method(polynomial, a, b):
    eps = 10 ** (-5)

    y1 = polynomial_func(polynomial, a)
    if fabs(y1) < eps:
        return a
    x = (b + a) / 2
    y = polynomial_func(polynomial, x)

    while fabs(y) > eps:
        if y1 * y <= 0:
            b = x
```

```python
        else:
            a = x
        x = (b + a) / 2
        y = polynomial_func(polynomial, x)

    return x


def Gauss(func, M, a, b, phi, tao):
    n = M + 1

    #Нахождение узлов t
    Legendre_pol_prev = np.array([0, 1])
    Legendre_pol = np.array([-0.5, 0, 1.5])

    t = [0, sqrt(1 / 3), 1]
    for i in range(3, n + 1):
        Legendre_pol_temp = Legendre_pol.copy()

        Legendre_pol_prev = np.append(Legendre_pol_prev, [0, 0])
        Legendre_pol = np.insert(Legendre_pol, 0, 0)

        Legendre_pol = (Legendre_pol * (2*i-1) -
                        Legendre_pol_prev * (i-1)) / i

        Legendre_pol_prev = Legendre_pol_temp.copy()

        for j in range(len(t) - 1):
            t[j] = dichotomy_method(Legendre_pol, t[j], t[j + 1])
        if i % 2 == 0:
            t.insert(0, 0)

    t.remove(1)
    i = 1
    while i < len(t):
        t.insert(0, -t[i])
        i += 2
    if n % 2 == 0:
        t.remove(0)

    #Нахождение коэффициентов A
    matr = []
    for k in range(n):
        matr.append((np.array(t) ** k).tolist())
        matr[k].append((1 - (-1) ** (k + 1)) / (k + 1))

    if n % 2:
        matr[0], matr[n // 2 + 1] = matr[n // 2 + 1][:], matr[0][:]

    A = []
    for i in range(n - 1, 0, -1):
        tmp = matr[i][i]
        for j in range(n + 1):
            matr[i][j] /= tmp

        for j in range(i):
            tmp = matr[j][i]
            for k in range(n + 1):
                matr[j][k] -= matr[i][k] * tmp
    matr[0][n] /= matr[0][0]

    for i in range(n):
        summ = matr[i][n]
        for j in range(len(A)):
```

```
            summm -= A[j] * matr[i][j]
        A.append(summ)

    F = 0
    for i in range(n):
        F += A[i] * func((b + a) / 2 + (b - a) / 2 * t[i], phi, tao)
    F *= (b - a) / 2

    return F


def get_arg(a, b, N, i):
    return (b - a) / N * i


def sequential_integration(func, N, M, a, b, tao):
    Integral = 0

    #Формула Симпсона
    for i in range(N // 2):
        Integral += Gauss(func, M, a, b, get_arg(b, a, N, 2 * i), tao)
        Integral += 4 * Gauss(func, M, a, b, get_arg(b, a, N, 2 * i + 1), tao)
        Integral += Gauss(func, M, a, b, get_arg(b, a, N, 2 * i + 2), tao)

    return Integral * (b - a) / N / 3
```

Листинг 2. main.py

```
from functions import *
import pylab


def function(teta, phi, tao):
    return 4 / pi * (1 - exp(-tao * 2 * cos(teta) /
           (1 - (sin(teta) * cos(phi)) ** 2))) * cos(teta) * sin(teta)


def main():
    M = 4
    a = 0
    b = pi / 2

    #Исследование при различном кол-ве узлов в методе Симпсона
    dots = []
    for N in range(2, 7, 2):
        temp = [[], []]
        tao = 0.05
        while tao <= 10:
            temp[0].append(tao)
            temp[1].append(sequential_integration(function, N, M, a, b, tao))
            tao += 0.05
        dots.append(temp)

    pylab.figure(1)
    pylab.plot(dots[0][0], dots[0][1], label = '3 узла сетки в методе Симпмона')
    pylab.plot(dots[1][0], dots[1][1], label = '5 узлов сетки в методе Симпмона')
    pylab.plot(dots[2][0], dots[2][1], label = '7 узлов сетки в методе Симпмона')
    pylab.legend()
    pylab.ylabel('Epsilon')
    pylab.xlabel('Tao')

    #Исследование при различном кол-ве узлов в методе Гаусса
    dots = []
```

```
    for M in range(2, 7, 2):
        temp = [[], []]
        tao = 0.05
        while tao <= 10:
            temp[0].append(tao)
            temp[1].append(sequential_integration(function, N, M, a, b, tao))
            tao += 0.05
        dots.append(temp)

    N = 4
    pylab.figure(2)
    pylab.plot(dots[0][0], dots[0][1], label = '3 узла сетки в методе Гаусса')
    pylab.plot(dots[1][0], dots[1][1], label = '5 узлов сетки в методе Гаусса')
    pylab.plot(dots[2][0], dots[2][1], label = '7 узлов сетки в методе Гаусса')
    pylab.legend()
    pylab.ylabel('Epsilon')
    pylab.xlabel('Tao')

    pylab.show()


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

3 Результаты работы

1. Описать алгоритм вычисления n корней полинома Лежандра n-ой степени P (x) n при реализации формулы Гаусса.

Для вычисления коэффициентов полинома Лежандра m-ой степени использовалось рекуррентное соотношение:

$$P_m(x) = \frac{1}{m}[(2m-1)\,x\,P_{m-1}(x) - (m-1)\,P_{m-2}(x)] \quad .$$

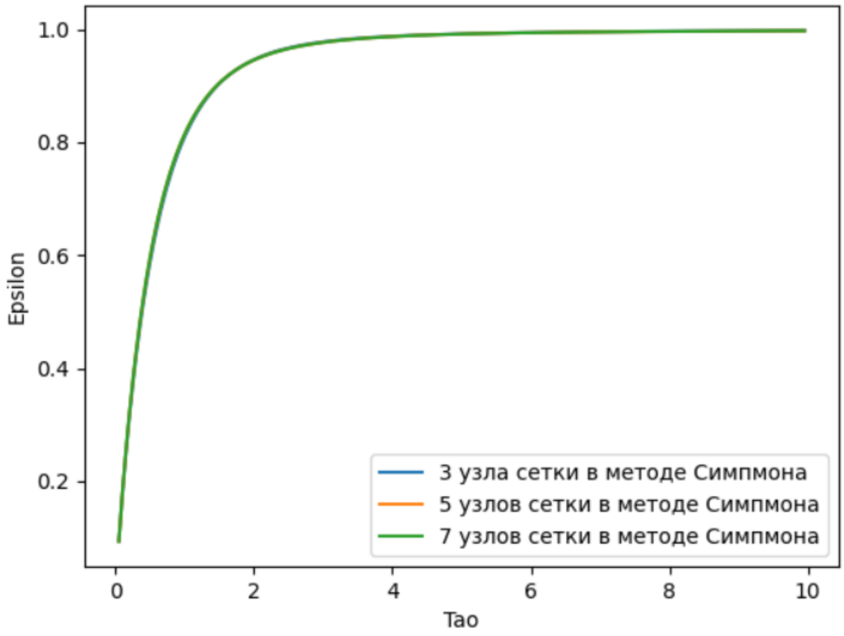
После нахождения полинома m-ой степени находились корни этого полинома на промежутке [0; 1], т.к. полиномы Лежандра чётных степеней явл. чётными, а нечётных степеней – нечётными.

Для нахождения интервала, в котором следует искать корень использовалось следующее свойство полиномов Лежандра: если рассматривать корни полинома степени m как делящие интервал [-1; 1] в m + 1 подынтервалы, каждый подынтервал будет содержать ровно один ноль из полинома степени m + 1.

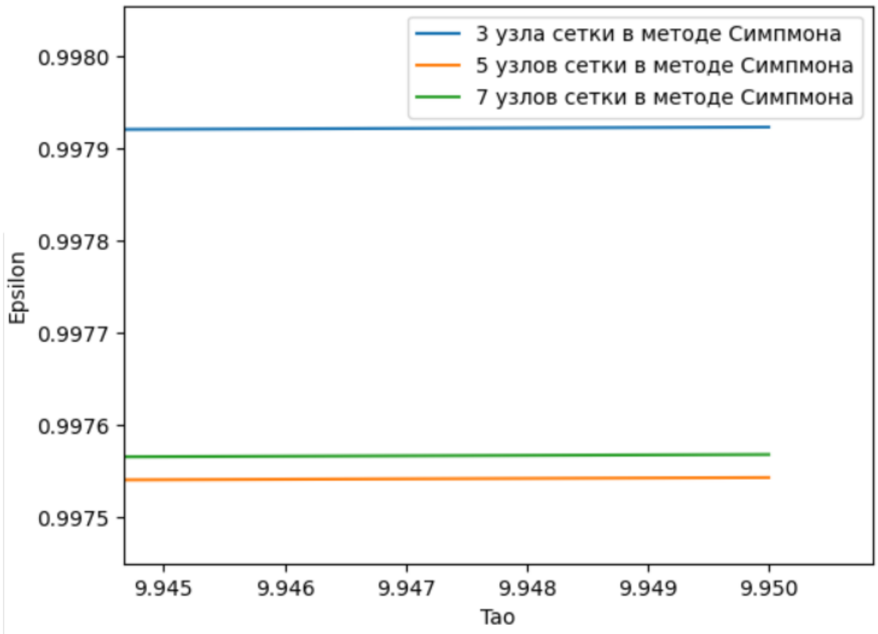
Для нахождения корней использовался метод половинного деления, идея которго состоит в следующем: если на концах отрезка значения функции имеют разный знак, то корень находится внутри этого отрезка. Изначально мы точно знаем, что отрезок содержит корень, поэтому делим отрезок пополам и проверяем, какой половине принадлежит корень. Таким образом мы итеративно уточняем значение корня, пока он не достигнет заранее указанной точности.

2. Исследовать влияние количества выбираемых узлов сетки по каждому направлению на точность расчетов.

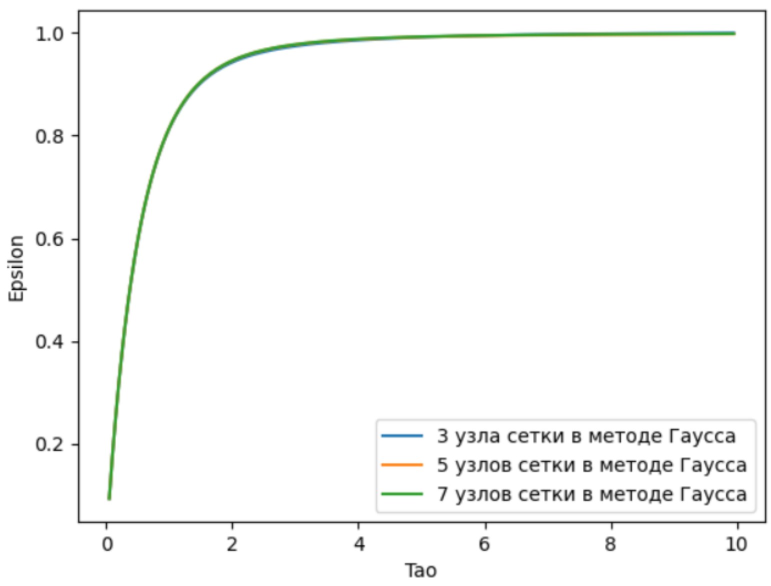
1) Исследование при различном кол-ве узлов в методе Симпсона (кол-во узлов в методе Гаусса равно 5):



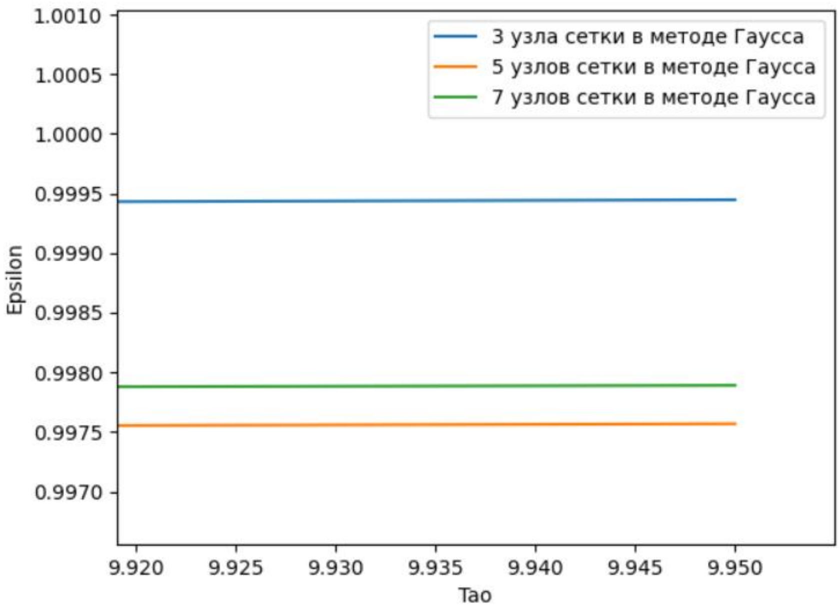
При приближении:



2) Исследование при различном кол-ве узлов в методе Гаусса (кол-во узлов в методе Симпсона равно 5):



При приближении:



При приближении видно, что уменьшение количества узлов в методе Гаусса уменьшает точность вычислений сильнее, чем изменение кол-ва узлов в методе Симпсона.

3. Построить график зависимости $\varepsilon(\tau)$ в диапазоне изменения τ = 0.05-10. Указать при каком количестве узлов получены результаты.

Данный пункт выполнен в процессе исследования из пункта 2.

4 Вопросы при защите лабораторной работы

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

Если подынтегральная функция не имеет соответствующих производных, то теоретический порядок точности не достигается. Так, если на отрезке интегрирования не существуют 3-я и 4-я производные, то порядок точности формулы Симпсона будет только 2-ой.

2. Построить формулу Гаусса численного интегрирования при одном узле.

A1 = 2

P1(t) = t => t1 = 0

$$\int_a^b f(x)dx = \frac{b-a}{2} * A1 * f(\frac{b+a}{2})$$

3. Построить формулу Гаусса численного интегрирования при двух узлах.

P2(t) = (3 * t^2 – 1) / 2 => t1 = √(1/3); t2 = -√(1/3)

$$\begin{cases} A1 + A2 = 2 \\ A1 * \sqrt{\frac{1}{3}} + A2 * (-\sqrt{\frac{1}{3}}) = 0 \end{cases} \Rightarrow A1 = A2 = 1$$

$$\int_a^b f(x)dx = \frac{b-a}{2} * (A1 * f\left(\frac{b+a}{2} + \frac{b-a}{2} * \sqrt{\frac{1}{3}}\right) + A2 * f\left(\frac{b+a}{2} + \frac{b-a}{2} * (-\sqrt{\frac{1}{3}})\right))$$

4. Получить обобщенную кубатурную формулу, аналогичную (6.6) из лекции №6, для вычисления двойного интеграла методом последовательного интегрирования на основе формулы трапеций с тремя узлами по каждому направлению.

$$\int_c^d \int_a^b f(x,y)dxdy = hx * \left(\frac{F0 + F2}{2} + F1\right)$$
$$= hx * hy * (\frac{f(x0,y0) + f(x0,y2) + f(x2,y0) + f(x2,y2)}{4} + \frac{f(x0,y1) + f(x2,y1) + f(x1,y0) + f(x1,y2)}{2} + f(x1,y1))$$