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»

Выполнил студент: __*Клименко Алексей Константинович*_____________

фамилия, имя, отчество

Группа: ______________________*ИУ7-45Б*___________________________

Проверил, к.п.н.: ____

подпись, дата

Оценка _________________________________ Дата _______________

2020 г.

Цель работы

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

Код программы

```
from typing import List, Callable as func
from math import exp, sin, cos, pi

from numpy import linspace, array
from numpy.linalg import solve

from matplotlib import pyplot as plt
from math import cos, pi


# Возвращает значение полинома Лежандра n-го порядка
def legendre(n: int, x: float) -> float:
    if n < 2: return [1, x][n]
    P1, P2 = legendre(n - 1, x), legendre(n - 2, x)
    return ((2 * n - 1) * x * P1 - (n - 1) * P2) / n


# возвращает значение производной полинома Лежандра
def legendre_prime(n: int, x: float) -> float:
    P1, P2 = legendre(n - 1, x), legendre(n, x)
    return n / (1 - x * x) * (P1 - x * P2)


# Нахождение корней полинома Лежандра n-го порядка
def legendre_roots(n: int, eps: float = 1e-12) -> List[float]:
    roots = [cos(pi * (4 * i + 3) / (4 * n + 2)) for i in range(n)]
    for i, root in enumerate(roots):  # уточнение корней
        root_val = legendre(n, root)
        while abs(root_val) > eps:
            root -= root_val / legendre_prime(n, root)
            root_val = legendre(n, root)
        roots[i] = root
    return roots


# Метод Гаусса для численного интегрирования на [-1; 1]
def gauss_integrate_norm(f: func, n: int) -> float:
    t = legendre_roots(n)
    T = array([[t_i**k for t_i in t] for k in range(n)])

    int_tk = lambda k: 2 / (k + 1) if k % 2 == 0 else 0
    b = array([int_tk(k) for k in range(n)])
    A = solve(T, b)  # решение системы линейных уравнений

    return sum(A_i * f(t_i) for A_i, t_i in zip(A, t))
```

```python
# Метод Гаусса для произвольного промежутка [a; b]
def gauss_integrate(f: func, a: float, b: float, n: int) -> float:
    mean, diff = (a + b) / 2, (b - a) / 2
    g = lambda t: f(mean + diff * t)
    return diff * gauss_integrate_norm(g, n)


# Метод Симпсона для промежутка [a; b]
def simpson_integrate(f: func, a: float, b: float, n: int) -> float:
    h, res = (b - a) / n, 0
    for i in range(0, n, 2):
        x1, x2, x3 = i * h, (i + 1) * h, (i + 2) * h
        f1, f2, f3 = f(x1), f(x2), f(x3)
        res += f1 + 4 * f2 + f3
    return h / 3 * res


def composite_integrate(f: func, a1: float, b1: float, a2: float, b2: float, \
    method_1: func, method_2: func, n1: int, n2: int) -> float:
    F = lambda y: method_1(lambda x: f(x, y), a1, b1, n1)
    return method_2(F, a2, b2, n2)


def function_integrator(f: func, a: float, b: float, \
    c: float, d: float, n: int, m: int) -> float:
    return composite_integrate(f, a, b, c, d, gauss_integrate, simpson_integrate, n, m)


def function(t: float, n: int, m: int) -> float:
    L_R = lambda theta, phi: 2 * cos(theta) / (1 - sin(theta)**2 * cos(phi)**2)
    f = lambda theta, phi: (1 - exp(-t * L_R(theta, phi))) * cos(theta) * sin(theta)

    return 4 / pi * function_integrator(f, 0, pi / 2, 0, pi / 2, n, m)


tao = linspace(0.05, 10, 100)
eps = [function(t, 3, 4) for t in tao]
plt.plot(tao, eps)

plt.grid()
plt.show()
```

Результаты работы

1. Алгоритм вычисления n корней полинома Лежандра n-ной степени:

$$x_i = \cos\left(\frac{(4i+3)\pi}{4n+2}\right), \quad i := \overline{0, n-1}, \quad \epsilon := 10^{-12}$$

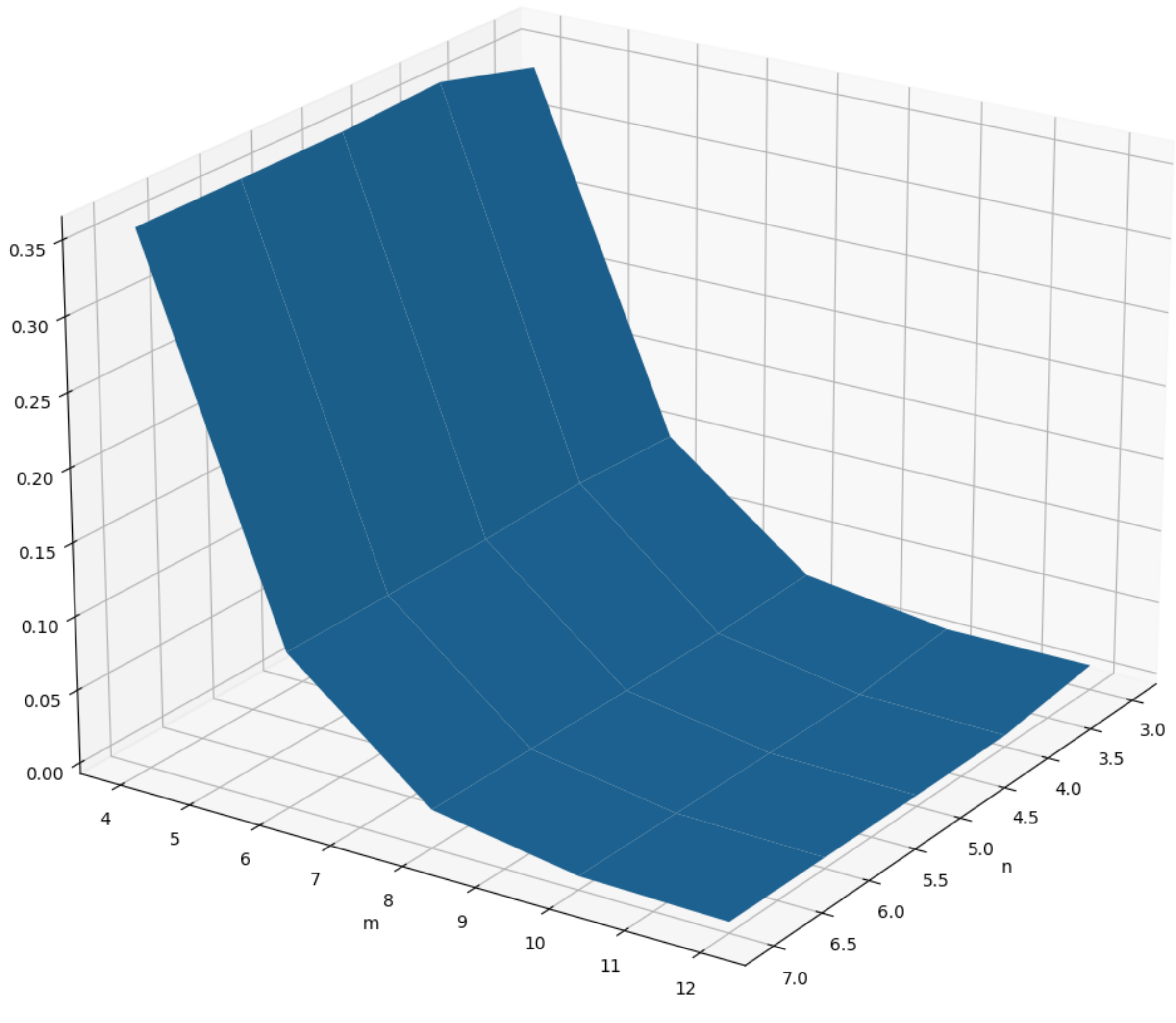
$$\textit{while} \ \left|P_n(x_i)\right| > \epsilon \ \ \textit{do}$$

$$x_i := x_i - \frac{P_n(x_i)}{P_n{}'(x_i)}$$

$$\textit{end}$$

Использован простой итеративный метод Ньютона.

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

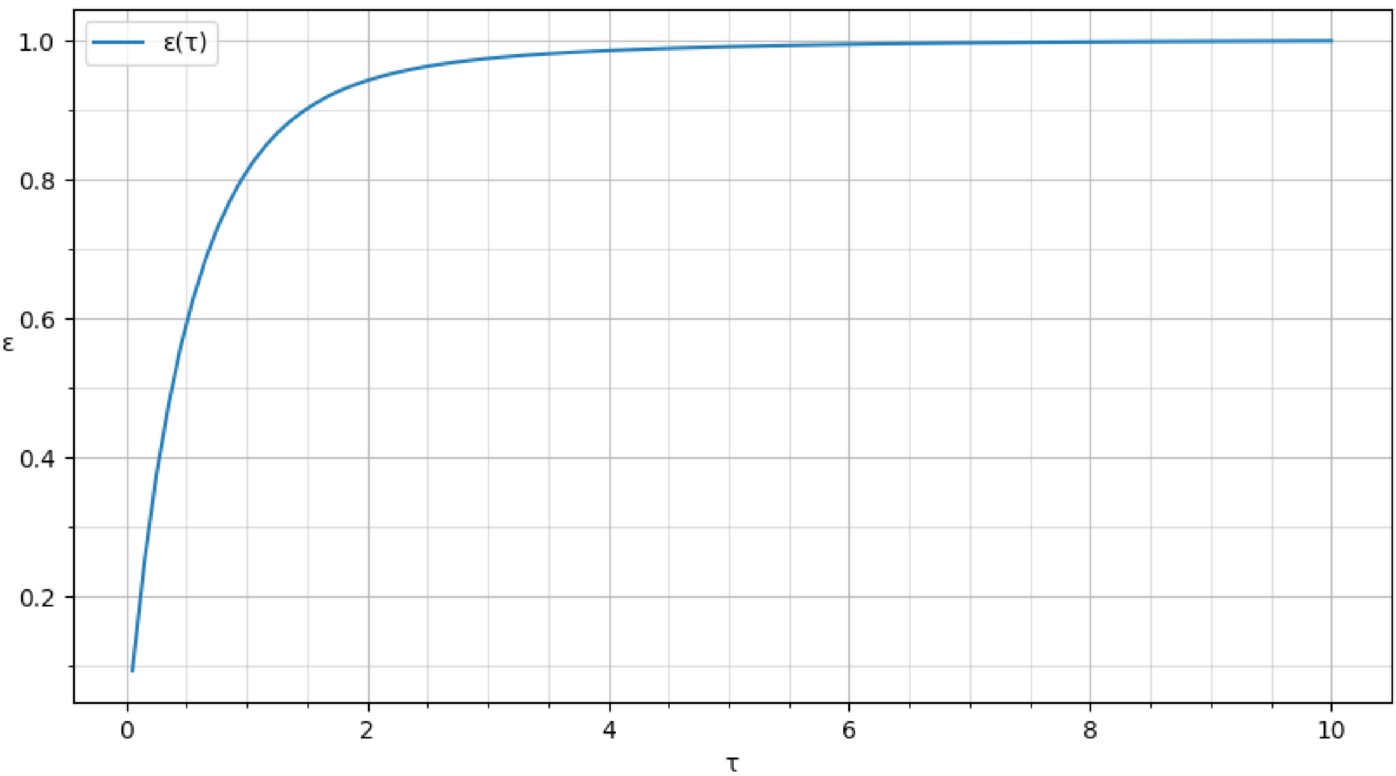


Приведённый выше график соответствует функции зависимости среднеквадратичной ошибки вычисления исходной функции (от двух параметров n и m для диапазона τ = [0.05; 10]). За эталон выбрана исходная функция при максимальных параметрах: n = 7 для метода Гаусса и m = 12 для формулы Симпсона.

По графику можно сказать, что зависимость ошибки от параметра n практически несущественна, что означает более высокую точность используемого метода Гаусса в сравнении с формулой Симпсона.

Поэтому для направления интегрирования по методу Гаусса можно брать небольшие количества узлов сетки - 3-4, тогда как по второму направлению - 10-12.

3. График зависимости $\varepsilon(\tau)$ при n = 3, m = 12:



Контрольные вопросы

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

Теоритический порядок точности квадратурных формул численного интегрирования может не достигаться в случаях больших значений производной интегрируемой функции т.е. при наличии резких скачков функции.

2. *Построить формулу Гаусса численного интегрирования при одном узле.*

$$N = 1: \quad \int_a^b f(x)dx \approx \frac{b-a}{2} A_1 f(x_1)$$

$$P_1(t) = t \ \rightarrow \ t_1 = 0$$

$$x_1 = \frac{a+b}{2} + \frac{b-a}{2} t_1 = \frac{a+b}{2}$$
$$A_1 = 2$$

$$\int_a^b f(x)dx \approx (b-a)\, f\left(\frac{a+b}{2}\right)$$

3. *Построить формулу Гаусса численного интегрирования при двух узлах.*

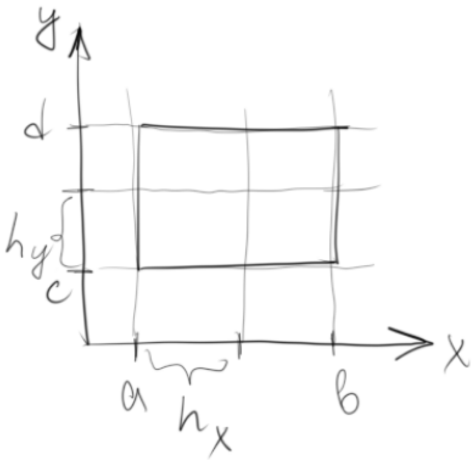
$$N = 2: \quad \int_a^b f(x)dx \approx \frac{b-a}{2}\left[\, A_1 f(x_1) + A_2 f(x_2) \,\right]$$

$$P_2(t) = 3t^2 - 1 \ \rightarrow \ t_1 = -\frac{1}{\sqrt{3}}, \ \ t_2 = \frac{1}{\sqrt{3}}$$

$$x_1 = \frac{a+b}{2} + \frac{b-a}{2} t_1 = \frac{a+b}{2} - \frac{b-a}{2\sqrt{3}}$$
$$x_2 = \frac{a+b}{2} + \frac{b-a}{2} t_2 = \frac{a+b}{2} + \frac{b-a}{2\sqrt{3}}$$
$$\begin{cases} A_1 + A_2 = 2 \\ A_1 t_1 + A_2 t_2 = 0 \end{cases} \ \rightarrow \ A_1 = A_2 = 1$$

$$\int_a^b f(x)dx \approx \frac{b-a}{2}\left[f\left(\frac{a+b}{2} - \frac{b-a}{2\sqrt{3}}\right) + f\left(\frac{a+b}{2} + \frac{b-a}{2\sqrt{3}}\right) \right]$$

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.*



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

$$I = \int_c^d \int_a^b f(x,y)\,dx\,dy = \int_c^d F(y)\,dy\,,\ \ F(y) = \int_a^b f(x,y)\,dx$$

$$F(y) = \frac{h_x}{2}\left(f(a,y) + 2f\left(\frac{a+b}{2},y\right) + f(b,y)\right)$$

$$\int_c^d F(y)\,dy = \frac{h_y}{2}\left(F(c) + 2F\left(\frac{c+d}{2}\right) + F(d)\right)$$

$$I = \frac{h_x h_y}{4}\left(\begin{array}{lll} f(a,c) & +\,2f\left(\frac{a+b}{2},c\right) & +\,f(b,c)\,+ \\ 2f\left(a,\frac{c+d}{2}\right) & +\,4f\left(\frac{a+b}{2},\frac{c+d}{2}\right) & +\,2f\left(b,\frac{c+d}{2}\right)\,+ \\ f(a,d) & +\,2f\left(\frac{a+b}{2},d\right) & +\,f(b,d) \end{array}\right)$$