Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет Інформатики та Обчислювальної Техніки

Кафедра Автоматизованих Систем Обробки Інформації та Управління

Лабораторна робота №7

з дисципліни «Спеціальні розділи математики»

на тему

«Чисельне інтегрування функцій»

Виконав:

студент гр. ІС-02

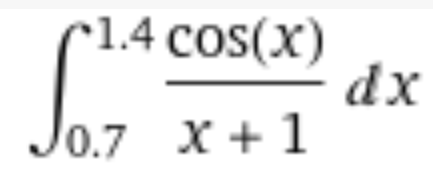
Плостак Ілля

Викладач:

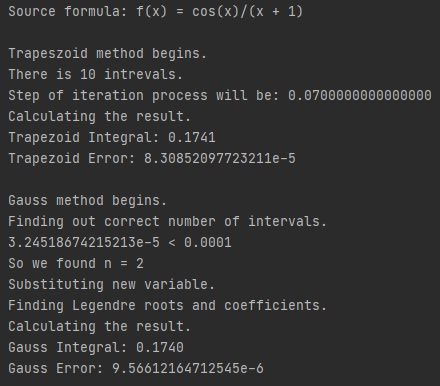
доц. Рибачук Л.В.

Київ – 2021

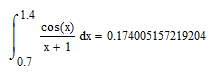
**1. Постановку задачі у вигляді вихідного рівняння**



**2. Обчислення інтегралу за допомогою формули трапеції або Сімпсона, та квадратурної формули Гауса:**



**3. Перевірочний розрахунок інтегралу за допомогою програми Mathcad:**



**4. Висновки:**

В цій лабораторній роботі ми порівняли три методи приблизного обчислення визначених інтегралів: метод трапецій, метод Сімпсона та метод квадратурних формул Гауса. З цих методів найточнішим є метод квадратурних формул Гауса, який дозволяє досягти максимальної для даного числа вузлів інтегрування алгебраїчної точності. Наприклад, для двох вузлів у методі Гауса ми отримали на порядок точніший результат аніж для дев’яти вузлів у методі трапецій. Це відбувається через оптимальне розміщення вузлів інтегрування у методі Гауса на відміну від рівномірного розподілу вузлів.

**5. Лістинг програми:**

[main.py](https://github.com/feedblackg44/kpilabs2/blob/master/CHM/Lab7/main.py):

from sympy import \*  
import functions as fc  
import math  
import numpy.polynomial.legendre as leg  
  
x = Symbol('x')  
  
f = cos(x) / (x + 1)  
n = 4  
e = 10\*\*-n  
a = 0.7  
b = 1.4  
MathCadIntegral = 0.174005157219204  
  
print("Source formula: f(x) =", f, "\n")  
if a < b:  
 trap = fc.trapIntegral(f, a, b, e)  
 print("Trapezoid Integral:", round(trap, n))  
 print("Trapezoid Error:", abs(MathCadIntegral - trap), "\n")  
 gauss = fc.gaussIntegral(f, a, b, e)  
 print("Gauss Integral:", round(gauss, n))  
 print("Gauss Error:", abs(MathCadIntegral - gauss))  
elif a == b:  
 print("Value of integral is 0.")  
else:  
 print("A mistake happend :D")

[functions.py](https://github.com/feedblackg44/kpilabs2/blob/master/CHM/Lab7/functions.py):

from sympy import \*  
from math import ceil  
import numpy.polynomial.legendre as leg  
  
x = Symbol('x')  
  
def trapIntegral(f, a, b, e):  
 print("Trapeszoid method begins.")  
 maxSecond = maxVal(f.diff(x).diff(x), a, b)  
 n = ceil(sqrt((maxSecond\*((b-a)\*\*3))/(abs(e)\*12)))  
 print("There is", n, "intrevals.")  
 f0 = lambdify(x, f, modules="sympy")  
 integral = (b - a) / (2 \* n)  
 summ = f0(a)  
 step = (b - a) / n  
 print("Step of iteration process will be:", step)  
 i = a + step  
 print("Calculating the result.")  
 while i < b:  
 summ += 2\*f0(i)  
 i += step  
 summ += f0(b)  
 integral \*= summ  
 return integral  
  
def maxVal(f, a, b):  
 e = 0.000001  
 f0 = lambdify(x, f)  
 solutions = []  
 i = a  
 while i <= b:  
 solutions.append(abs(f0(i)))  
 i += e  
 solutions.append(abs(f0(b)))  
 return max(solutions)  
  
def gaussIntegral(f, a, b, e):  
 print("Gauss method begins.")  
 n = 0  
 e1 = 1  
 print("Finding out correct number of intervals.")  
 while e1 >= e:  
 n += 1  
 f0 = f  
 for i in range(0, 2 \* n):  
 f0 = f0.diff(x)  
 e1 = ((factorial(n)\*\*4 \* (b-a)\*\*(2\*n+1) \* maxVal(f0, a, b))/((2\*n+1)\*(factorial(2\*n))\*\*3))  
 print(e1, "<", e)  
 print("So we found n =", n)  
 t = Symbol('t')  
 f1 = f.subs(x, (b + a) / 2 + t \* (b - a) / 2)  
 f1 = lambdify(t, f1, modules="sympy")  
 print("Substituting new variable.")  
 multip = (b - a) / 2  
 print("Finding Legendre roots and coefficients.")  
 x\_arr, A\_arr = leg.leggauss(n)  
 f\_arr = [f1(x\_arr[i]) for i in range(len(x\_arr))]  
 summ = 0  
 print("Calculating the result.")  
 for i in range(len(f\_arr)):  
 summ += A\_arr[i]\*f\_arr[i]  
 return summ \* multip