**МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА**

**СПОРТУ УКРАЇНИ**

**Національний технічний університет України**

**“Київський політехнічний інститут”**

**Чисельні методи**

**Лабораторна робота №6**

**Варіант 17**



Виконав

студент групи ІП-22

Сочка Олександр Олександрович

**Київ – 2015**

**Теоретичні відомості**

***Формула трапеції***

Найпростіша інтерполяційна квадратурна формула одержується при   .

Тоді коефіцієнти при *; k = 0, 1, 2* обчислюються за формулою

; ;  .

На відрізку [a,b] одержуємо наступну формулу

.

Формула (1.6) має назву формули *трапецій.* Оцінка похибки цієї формули:



***Квадратурна формула Гауса***

При побудові квадратурних формул, що грунтуються на інтерполяційних формулах, використовувалися рівновіддалені вузли. Для побудови квадратурних формул Гауса вузли формуются іншим шляхом.

Побудуємо квадратурну формулу у вигляді



що буде точною для поліномів найбільш високого степеню при найменшій кількості вузлів. Коєфіцієнти  та вузли  визначимо за умови, щоб формула була точною для 2*m* функцій . З цієї умови та (9.16) одержуємо наступні 2*m* рівнянь

; .

Нехай



Тоді



Будемо шукати розв’язок цієї системи рівнянь за допомогою поліномів Лежандра. З цієї метою домножимо перше та наступні *m* рівнянь системи на ;  й складемо одержані *m* + 1 рівнянь:



За визначенням поліномів Лежандра



Якщо виконати ці ж дії із наступними *m* рівняннями, то одержимо для *k*-го рівняння



За умови ортогональності поліномів Лежандра



одержимо



Якщо в якості вузлів взяти корені поліномів Лежандра, то одержимо вузли для квадратурної формули Гаусса



У такому разі для визначення коефіцієнтів квадратурної формули одержуємо наступну систему лінійних рівнянь



Інтеграл буде дорівнювати

 .

У випадку відрізку довільної довжини  заміною змінної



приходимо до обчислення інтегралу на відрізку [-1,1].

Похибка квадратурної формули Гауса оцінюєтся нерівністю



**Постановка задачі**

Розв’язати нелінійне рівняння:

**Варіант 17**

**a = 2**

**b = 5**

**Результат**

Simpson's method

Segments: 262144

Calculated: -4.818569479756418

Real: -4.818635323909605

Diff: 6.584415318755532e-05

Gauss quadrature method:

N: 1

Calculated: -4.735573573809868

Real: -4.818635323909605

Diff: 0.08306175009973771

N: 2

Calculated: -4.921336842979301

Real: -4.818635323909605

Diff: 0.10270151906969538

N: 3

Calculated: -4.81467610699802

Real: -4.818635323909605

Diff: 0.003959216911585628

N: 4

Calculated: -4.818696141798503

Real: -4.818635323909605

Diff: 6.081788889744644e-05

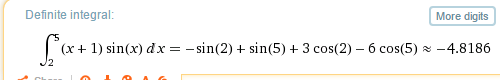
N: 5

Calculated: -4.818634810370236

Real: -4.818635323909605

Diff: 5.13539369251248e-07

**Результат в WolframAlpha**

****

**Висновок**

Методи трапецій та Сімпсона мають властивість накопичення похибки як збільшуючи, так і зменшуючи точність результату (за рахунок наявності апроксимуючих ліній як над, так і під графіком функцій, що призводить до наявності як додатніх, так і від’ємних площ).

Метод квадратурних формул Гауса дозволяє підвищити точність результату інтегрування (зафіксувавши кількість виконуваних операцій).

**Лістинг програми**

#!/usr/bin/python3

from math import \*

f = lambda x: (x+1) \* sin(x)

ddf = lambda x: 2\*cos(x) - (x + 1) \* sin(x)

integral = lambda x: sin(x) - (x + 1) \* cos(x)

a = 2

b = 5

rules = {

1: [(0, 2)],

2: [

(sqrt(1 / 3), 1),

(-sqrt(1 / 3), 1)

],

3: [

(0, 8 / 9),

(sqrt(3 / 5), 5 / 9),

(-sqrt(3 / 5), 5 / 9)

],

4: [

( sqrt(3 / 7 - 2 / 7 \* sqrt(6 / 5)), (18 + sqrt(30)) / 36),

(-sqrt(3 / 7 - 2 / 7 \* sqrt(6 / 5)), (18 + sqrt(30)) / 36),

( sqrt(3 / 7 + 2 / 7 \* sqrt(6 / 5)), (18 - sqrt(30)) / 36),

(-sqrt(3 / 7 + 2 / 7 \* sqrt(6 / 5)), (18 - sqrt(30)) / 36)

],

5: [

(0, 128 / 225),

( 1 / 3 \* sqrt(5 - 2 \* sqrt(10 / 7)), (322 + 13 \* sqrt(70)) / 900),

(-1 / 3 \* sqrt(5 - 2 \* sqrt(10 / 7)), (322 + 13 \* sqrt(70)) / 900),

( 1 / 3 \* sqrt(5 + 2 \* sqrt(10 / 7)), (322 - 13 \* sqrt(70)) / 900),

(-1 / 3 \* sqrt(5 + 2 \* sqrt(10 / 7)), (322 - 13 \* sqrt(70)) / 900)

]

}

def segs():

accuracy = 1e-3;

tmp\_max = ddf(a);

i = a

while i <= b:

tmp\_max = max(tmp\_max, abs(ddf(i)))

i += accuracy

mx = tmp\_max

residual\_part = lambda arg: (b - a) \*\* 3 \* mx / 12 / arg

segments = 2

while abs(residual\_part(segments)) > accuracy:

segments \*= 2

return segments

def simpson():

print('Simpson\'s method')

segments = segs()

print('Segments: ', segments)

h = (b - a) / segments

z = a

w = a + h

result = 0.

while w < b:

result += h / 6 \* (f(z) + 4\*f((z+w)/2) + f(w))

z += h

w += h

real = integral(b) - integral(a)

print('Calculated: ', result)

print('Real: ', real)

print('Diff: ', abs(real - result))

print()

print()

def gauss():

print("Gauss quadrature method:");

for n in rules:

result = 0

for x, w in rules[n]:

result += w \* f((b - a) / 2 \* x + (a + b) / 2)

result \*= (b - a) / 2

real = integral(b) - integral(a)

print("N: ", n)

print("Calculated: ", result)

print("Real: ", real)

print("Diff: ", abs(real - result))

print()

simpson()

gauss()