МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Ничипорук Роман Олегович

Численное интегрирование

Отчет по лабораторной работе № 2

Вариант 10

Численные методы

студента 3-го курса 4-ой группы

Преподаватель:
Левчук Е.А.

2024

ОГЛАВЛЕНИЕ

ГЛАВА 1

Введение

Лабораторная работа была выполнена с использованием языка програмирования Python.

При реализации функций были применены следующие модули:

- math - модуль предоставляет обширный функционал для работы с числами;

ГЛАВА 2

Задание 1

2.1 Условие

1. Написать программу, которая вычисляет интеграл с заданной точностью с помощью указанной составной квадратурной формулы с автоматическим выбором шага по правилу Рунге.

2. Вывести приближенное значения интеграла, а также величину шага.

3. Сравнить приближенное значение интеграла с точным, вычисленным аналитически.

$$\int_1^4 cosh(2x)sinh(2x)dx, \quad \epsilon = 10^{-7}$$

Используя квадратурную формулу трапеций.

2.2 Использованная теория

Случай неравноотстоящих узлов

Составная квадратурная формула трапеций на равномерной сетке:

$$\int_a^b f(x)dx \approx h\sum_{i=1}^{n} \frac{f_{i-1}+f_i}{2} \tag{2.1}$$

Правило Рунге:

$$\Theta = \frac{1}{2^p - 1} \tag{2.2}$$

Под p понимается порядок погрешности использованного численного метода.

2.3 Результаты

```
Approximate value of the integral: 555378.4940027511
Exact value of the integral: 555378.4940026
Step: 3.5762786865234375e-07
```

Листинг 2.1 terminal

ГЛАВА 3

Задание 2

3.1 Условие

1. Написать программу, которая находит приближенное значение интеграла с помощью квадратурной формулы НАСТ, а также с помощью формулы средних прямоугольников при том же самом количестве узлов.

2. Вывести полученные приближенные значения для разного количества узлов (по своему усмотрению).

3. На основании этих значений сделать вывод об эффективности формулы НАСТ по сравнению с формулой средних прямоугольников.

$$\int_{-1}^{1} \frac{sin^2 x}{\sqrt{1-x^2}} dx$$

3.2 Использованная теория

Окончательный вид квадратурной формулы НАСТ:

$$\int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}} dx \approx \frac{\pi}{n+1} \sum_{i=1}^{n} f\left(cos(\frac{\pi(2i+1)}{2(n+1)}) \right) \tag{3.1}$$

Составная квадратурная формула средних прямоугольников:

$$\int_{a}^{b} f(x) dx \approx h \sum_{i=1}^{n} f_{i=\frac{1}{2}}, \quad f_{i=\frac{1}{2}} = f\left(\frac{x_{i-1}+x_i}{2} \right) \tag{3.2}$$

3.3 Результаты

```
Number of nodes HADA  formula of average rectangles
1        1.3258405438706218    0.0
2        1.215331965220616   0.5308131749057596
```

```
3         1.2191791924349824    0.6840243090928864
4         1.219108723084689   0.765672740928688
5         1.2191095193883368    0.8185010588855125
6         1.2191095132828884    0.8562918338693903
7         1.2191095133167376    0.8850590614974976
8         1.2191095133165957    0.9079074434691199
9         1.2191095133165961    0.926624842076715
10        1.2191095133165957    0.9423236429738184
11        1.2191095133165961    0.9557372342198867
12        1.2191095133165961    0.9673714384305825
13        1.2191095133165963    0.9775879708909896
14        1.2191095133165957    0.9866532843411431
15        1.2191095133165961    0.9947685854754702
16        1.2191095133165961    1.0020890427680509
17        1.2191095133165963    1.0087365046448786
18        1.219109513316596   1.0148081702727039
19        1.219109513316596   1.0203826524160085
20        1.2191095133165961    1.0255243115805603
```

Листинг 3.1 terminal

ГЛАВА 4

Листинг программы

```python
from Lab_2 import utility

utility.run_task1()
utility.run_task2()
```

Листинг 4.1 main.py

```python
import math
import formulas


def f1(x):
    return math.sinh(2*x)*math.cosh(2*x)


def f2_numerator(x):
    return math.pow(math.sin(x), 2)


def f2(x):
    return f2_numerator(x) / math.sqrt(1 - math.pow(x, 2))


def calculate_error_using_runge_rule(i1, i2, p):
    return abs(i1 - i2) / (pow(2, p) - 1)


def run_task1():
    exact_integral = 555378.4940026
    a = 1
    b = 4
    eps = 1e-7
    p = 2
    N = 1
    error = 1
    while error > eps:
        i1 = formulas.calculate_using_trapezoidal_formula(a, b, N)
        N *= 2
        i2 = formulas.calculate_using_trapezoidal_formula(a, b, N)
        error = calculate_error_using_runge_rule(i1, i2, p)
    step_size = (b - a) / N
    print("Approximate value of the integral:", formulas.calculate_using_trapezoidal_formula(a, b, N))
    print("Exact value of the integral:", exact_integral)
    print("Step:", step_size)


def run_task2():
    a = -1.0
```

```
    b = 1.0
    print("Number of nodes\tHADA\tformula of average rectangles")
    for n in range(1, 21):
        nystrom_result = formulas.calculate_using_hada(a, b, n)
        rectangle_result = formulas.calculate_using_rectangle_formula(a, b, n)
        print(n, "\t\t\t", nystrom_result, "\t", rectangle_result)
```

Листинг 4.2 utility.py

```
iimport math

import utility


def calculate_using_trapezoidal_formula(a, b, n):
    h = (b - a) / n
    integral = 0.5 * (utility.f1(a) + utility.f1(b))
    for i in range(1, n):
        integral += utility.f1(a + i * h)
    return h * integral


def calculate_using_hada(a, b, n):
    sum_ = 0.0
    for i in range(n + 1):
        sum_ += utility.f2_numerator(math.cos(math.pi * (2 * i + 1) / 2 / (n +
    1)))
    return sum_ * math.pi / (n + 1)


def calculate_using_rectangle_formula(a, b, N):
    h = (b - a) / N
    sum_ = 0.0
    for i in range(N):
        sum_ += utility.f2(a + i * h + h / 2.0)
    return h * sum_
```

Листинг 4.3 formulas.py

ГЛАВА 5

Вывод

В задании 1 точное значение интеграла совпадает с приближенным, при этом программе потребовалось 23 шага.

В задании 2 приближенные значения, полученные по формуле НАСТ больше приближенны к точному значению, чем значения, полученные по формуле средних прямоугольников. Это свидетельсвует об большей эффективности формулы НАСТ для вычисления данного интеграла.