МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Кафедра вычислительной математики

Отчёт

Лабораторная работа

“Приближённое вычисление интегралов”

Вариант №5

Благодарный Артём Андреевич
студент 3 курса, 3 группы
специальности «Информатика»
дисциплина «Численные методы»

Минск, 2025

Постановка задачи

Для вычисления интеграла

$$\int_{0.35}^{1.35} 0.35e^{x} + 0.65\sin x \, dx$$

Для вычисления интеграла с точностью $\varepsilon = 10^{-5}$ необходимо:

1. Пользуясь выражением для погрешности интерполирования, определить шаг h в составной квадратурной формуле средних прямоугольников, которая обеспечит требуемую точность результата. (Симпсон)

2. Для СКФ из п.1 определить величину h шага разбиения исходного отрезка интегрирования, достаточного для достижения точности ε, по правилу Рунге.

3. Применить квадратурную формулу НАСТ Гаусса при указанном значении n. Оценить погрешность интегрирования через формулу остаточного члена $R_n(f)$, $n = 1$.

4. Провести сравнительный анализ полученных результатов.

Пункт 1.

Алгоритм решения

Квадратурная формула Симпсона:

$$\int_a^b f(x)dx \approx \frac{b-a}{6}\left(f(a) + 4f(\tfrac{a+b}{2}) + f(b)\right)$$

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 4f(x_{N-1}) + f(x_N)\right]$$

Формула имеет нечётное число узлов, поэтому применима теорема 14.3. остаток равен:

$$R_2(f) = \frac{1}{4!}\int_a^b f^{(4)}(\xi)(x-a)(x-\tfrac{a+b}{2})^2(x-b)dx.$$

Можно возможность применить теорему о среднем, что в итоге даёт:

$$R_2(f) = -\frac{f^{(4)}(\eta)}{2880}(b-a)^5.$$

$$|R| \le \frac{(b-a)^5}{180 \cdot N^4} \cdot \max|f^{(4)}(x)| \le \varepsilon$$

Можем вычислить N:

$$N \ge \left(\frac{(b-a)^5 \cdot \max|f^{(4)}(\eta)|}{180 \cdot \varepsilon}\right)^{1/4}$$

Найдём h:

$$h = \frac{b - a}{N} = \frac{1.35 - 0.35}{N}$$

Листинг кода

```
def f(x):
    return 0.35 * math.exp(x) + 0.65 * math.sin(x)

def true_integral(a, b):
```

```
    return (0.35 * math.exp(b) - 0.65 * math.cos(b)) - (0.35 * math.exp(a) - 0.65 * math.cos(a))

def simpsons_rule(a, b, epsilon):
    """
    Составной метод Симпсона:
    |E| ≤ ((b - a)^5) / (180 * N^4) * max|f^{(4)}(x)|
    где здесь f^{(4)}(x) = f(x) для нашей функции
    """
    # Оценка max|f^{(4)}(x)| на концах отрезка
    max_f4 = max(abs(f(a)), abs(f(b)))
    N = 2
    while True:
        error_est = ((b - a)**5) / (180 * N**4) * max_f4
        if error_est <= epsilon:
            break
        N += 2
    h = (b - a) / N
    integral_sum = f(a) + f(b)

    for i in range(1, N):
        x_i = a + i * h
        weight = 4 if i % 2 else 2
        integral_sum += weight * f(x_i)

    integral_approx = integral_sum * h / 3
    return integral_approx, N, h

a, b = 0.35, 1.35
epsilon = 1e-5

approx, N, h = simpsons_rule(a, b, epsilon)
true_val = true_integral(a, b)
error = abs(approx - true_val)
max_f4 = max(abs(f(a)), abs(f(b)))
error_est = ((b - a)**5) / (180 * N**4) * max_f4

print(f"Использовано разбиений N = {N}")
print(f"Величина шага h = {h}")
print(f"Приближённое значение интеграла: {approx}")
print(f"Истинное значение интеграла: {true_val}")
print(f"Оценка погрешности: {error_est:.6e}")
print(f"Истинная погрешность: {error:.6e}")
```

Результаты

- Использовано разбиений N = 6
- Величина шага h = 0.166666666666
- Приближённое значение интеграла: 1.3216688706368558
- Истинное значение интеграла: 1.3216632104766206
- Оценка погрешности: 8.506169e-06
- Истинная погрешность: 5.660160e-06

Истинная погрешность достигла степени 10^{-6}, что меньше требуемой погрешности. Это следует из того, что число разбиений было посчитано довольно грубо.

Пункт 2.

Алгоритм решения

Пусть $Q_n = Q$ — функционал СКФ Симпсона. Вычислим приближённое значение интеграла $S(f)$ по соответствующей составной формуле дважды, на двух разных разбиениях с числом отрезков N_1 и N_2, $N_2 > N_1$. Полученные приближения обозначим соответственно:

$$q_1 = Q^{N_1}(f) \quad \text{и} \quad q_2 = Q^{N_2}(f).$$

Пусть имеет место разложение остатка составной квадратурной формулы:

$$R^N(f) = S(f) - Q^N(f) = Kh^p + o(h^p), \quad h = \frac{b-a}{N},$$

где константа K не зависит от h, p — показатель точности КФ. Для Симпсона p = 4. Таким образом, $K\mathrm{h}^{\mathrm{p}}$ представляет собой главную часть погрешности, которую можно найти, если известно значение константы K.

Можно записать приближённые равенства:

$$S(f) - q_1 \approx Kh_1^p,$$
$$S(f) - q_2 \approx Kh_2^p.$$

Отсюда, исключая $S(f)$, имеем:

$$K \approx \frac{q_2 - q_1}{h_1^p - h_2^p} = \frac{\delta}{h_1^p - h_2^p} =: \tilde{K},$$

где $\delta = q_1 - q_2$

Откуда получаем оценку погрешности для обеих квадратурных сумм:

$$R^{N_i}(f) \approx Kh_i^p \approx \tilde{K}h_i^p = \tilde{R}_i,$$

где

$$\tilde{R}_i = \frac{\delta h_i^p}{h_1^p - h_2^p}.$$

Таким образом имеем следующий алгоритм:

1. Выбираем N_1 и N_2. Пусть $N_1 = 5$, $N_2 = 2N_1$;
2. Вычисляем q_1, q_2;
3. Вычисляем $\tilde{R}$. В нашем случае формула примет следующий вид:

$$\tilde{R} = \frac{q_2 - q_1}{2^p - 1}$$

4. Если $\tilde{R} \leq \varepsilon$, завершаем алгоритм, возвращаем q_2, h_2, N_2. Иначе – идём дальше;
5. Полагаем $N_2 \leftarrow 2{*}N_2$, $q1 \leftarrow q2$, $q2 \leftarrow Q^{2*N_2}(f)$;
6. Переходим к шагу 3.

Листинг кода

```
import math

def f(x):
    return 0.35 * math.exp(x) + 0.65 * math.sin(x)

def simpson_integral(a, b, n):
    """
    Составной метод Симпсона.
    n - количество разбиений (должно быть четным).
    """
    if n % 2 != 0:
        raise ValueError("Количество разбиений n должно быть чётным для метода Симпсона")

    h = (b - a) / n
    s = f(a) + f(b)

    for i in range(1, n):
        x_i = a + i * h
        weight = 4 if i % 2 != 0 else 2
        s += weight * f(x_i)

    return s * h / 3

def main():
    a = 0.35
    b = 1.35
    eps = 1e-5
    p = 4  # порядок точности для метода Симпсона
    N_1 = 4  # начальное количество разбиений (четное)
    N_2 = 2 * N_1

    q_1 = simpson_integral(a, b, N_1)
    q_2 = simpson_integral(a, b, N_2)

    while True:
        R = (q_2 - q_1) / (2**p - 1)

        if abs(R) <= eps:
            break

        N_1 = N_2
        q_1 = q_2
        N_2 *= 2
        q_2 = simpson_integral(a, b, N_2)

    h_final = (b - a) / N_2
    integral_approx = q_2
```

```
    integral_exact = 0.35 * (math.exp(b) - math.exp(a)) - 0.65 * (math.cos(b) -
math.cos(a))
    error_true = abs(integral_approx - integral_exact)

    print(f"Шаг h: {h_final}")
    print(f"Количество разбиений: {N_2}")
    print(f"Приближенное значение интеграла: {integral_approx}")
    print(f"Истинное значение интеграла: {integral_exact}")
    print(f"Оценка погрешности по Рунге R: {R}")
    print(f"Истинная погрешность: {error_true}")

main()
```

Результаты

- Шаг h: 0.125
- Количество разбиений: 8
- Приближенное значение интеграла: 1.3216650021307377
- Истинное значение интеграла: 1.3216632104766204
- Оценка погрешности по Рунге R: 1.788631695657609e-06
- Истинная погрешность: 1.7916541172890987e-06

Истинная погрешность достигла степени 10^{-6}, что меньше требуемой погрешности. Погрешность в этом пункте несколько меньше погрешности, посчитанной в прошлый раз. Это связано с бóльшим число разбиений и тем, что правило Рунге позволяет получить апостериорную оценку, которая является более точной, поскольку учитывает поведение функции.

Пункт 3.

Алгоритм решения

Выпишем квадратурную формулу Гаусса:

$$I(f)=\int_{-1}^{1} f(x)dx \approx \sum_{k=0}^{n} A_k f\left(x_k\right)$$

Системой многочленов ортогональных по весу $p(x) = 1$ на отрезке $[-1, 1]$ является система многочленов Лежандра:

$$L_n(x)=\frac{1}{2^n n!}\frac{d^n}{dx^n}\left((x^2-1)^n\right)$$

где в нашем случае n = 1.

Узлами x_k в формуле будут являться корни многочлена Лежандра, определяемые из соотношения для каждого конкретного значения n

$$P_{n+1}(x)=0. \quad P_1(x)=x \quad x_1=0$$

Коэффициент A_1 можно найти по формуле:

$$A_1=\frac{2}{(1-x_1^2)[P_1'(x_1)]^2} \quad P_1'(x)=1, \; A_1=\frac{2}{(1-0^2)\cdot 1^2}=2.$$

Для перехода к произвольному отрезку $[a, b]$ необходимо использовать линейное преобразование:

$$x=\frac{b-a}{2}x'+\frac{a+b}{2}, \quad x'\in[-1,1].$$

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

$$w_1=\frac{b-a}{2}A_1=(b-a)$$

Остаток квадратурной формулы для промежутка $[a, b]$ выражается через остаток на отрезке $[-1, 1]$ следующим образом:

$$R_n(f)=\left(\frac{b-a}{2}\right)^{2n+3} R_n(f).$$

где $R_n(f)$ в правой части:

$$|R_1(f)|\leq\frac{(b-a)^3}{12}\max_{x\in[a,b]}|f''(x)|$$

В нашем случае $f^{(2)}(\eta)$ оценим сверху как max$|f^{(2)}(\eta)|$ на [-1, 1].

Листинг кода

```
import numpy as np
from scipy.optimize import minimize_scalar

def f(x):
    return 0.35 * np.exp(x) + 0.65 * np.sin(x)

def f4(x):
    return f(x)

a = 0.35
b = 1.35
n = 1

x0 = a
x1 = (a + b) / 2
x2 = b

f0 = f(x0)
f1 = f(x1)
f2 = f(x2)

h = (b - a) / 2
simpson_integral = (h / 3) * (f0 + 4 * f1 + f2)

true_integral = 0.35 * (np.exp(b) - np.exp(a)) - 0.65 * (np.cos(b) - np.cos(a))

error = abs(simpson_integral - true_integral)

res = minimize_scalar(lambda x: -abs(f4(x)), bounds=(a, b), method='bounded')
max_f4 = abs(f4(res.x))

Rn = ((b - a)**5 / 2880) * max_f4

print(f"x0 = {x0}, x1 = {x1}, x2 = {x2}")
print(f"f(x0) = {f0:.10f}, f(x1) = {f1:.10f}, f(x2) = {f2:.10f}")
print(f"Приближённое значение интеграла (Симпсон, n=1): {simpson_integral:.10f}")
print(f"Истинное значение интеграла: {true_integral:.10f}")
print(f"Абсолютная погрешность: {error:.10e}")
print(f"Оценка остаточного члена Rn: {Rn:.10e}")
```

Результаты

- x0 = 0.35, x1 = 0.8500000000000001, x2 = 1.35
- f(x0) = 0.7195572169, f(x1) = 1.3072086615, f(x2) = 1.9843191183
- Приближённое значение интеграла (Симпсон, n=1): 1.3221184969
- Истинное значение интеграла: 1.3216632105
- Абсолютная погрешность: 4.5528639773e-04
- Оценка остаточного члена Rn: 6.8899660487e-04

Пункт 4.

Сравнительный анализ полученных результатов

Метод	Количество разбиений N	Шаг h	Оценка погрешности	Истинная погрешность
Симпсон	6	0.166666666666	8.506169e-06	5.660160e-06

Правило Рунге	8	0.125	1.788631e-06	1.791654e-06
НАСТ Гаусса	1	-	6.889966e-04	4.55286e-04

Сравнивая представленные методы по точности, можно сделать вывод, что правило Рунге обеспечивает наименьшую погрешность порядка 1E-6, при наибольшем количестве узлов.

По итогам сравнения трёх подходов к численному интегрированию функции на отрезке видно, что **метод Симпсона** даёт наилучший баланс между числом разбиений и точностью. Уже при N=6 его теоретическая оценка погрешности ∼6×10^−6, что подтверждает надежность O(h^4) аппроксимации. При увеличении числа отрезков до 8 и применении **правила Рунге** мы получили истинную ошибку около 1.79×10−6.

В то же время **простая формула Симпсона** с одним сегментом (N=1) показала, что для грубых приближений её порядка точности O(h^4) недостаточно: истинная ошибка 4.55×10−4 больше ошибки полученной методом Симпсона.