Лабораторная работа №1
Математический анализ
Вариант 16

Григорьев Даниил, ИСУ: 465635

Группа P3116, поток: Мат Ан Прод 11.3

21 апреля 2025

Задание

Вычислить приближённо интеграл с погрешностью $\epsilon = 0,00001$ методами: прямоугольников, трапеций, Симпсона. Интеграл:

$$\int_{\frac{\pi}{4}}^{\frac{\pi}{2}} lnsinx\,dx$$

Разбиение: $h = \frac{b-a}{n}$, узлы: $x_i = a + \left(i + \frac{1}{2}\right) h$, $i = 0, \ldots, n-1$

Формула:

$$I_n = h \sum_{i=0}^{n-1} f(x_i)$$

Погрешность:

$$|\Delta| \leq \frac{b-a}{24} h^2 \cdot \sup_{x \in [a,b]} |f''(x)|$$

Метод прямоугольников

```
import numpy as np

def f(x):
    return np.log(np.sin(x))

def f2(x):
    return -1 / (np.sin(x)**2) - 1 / (np.tan(x)**2)


a = np.pi / 4
b = np.pi / 2
eps = 1e-5


def rectangles_method(f, f2, a, b, eps):
    n = 1
    while True:
        h = (b - a) / n
        x = a + h / 2
        total = 0
        for _ in range(n):
```

```
            total += f(x) * h
            x += h
        x_vals = np.linspace(a + 1e-6, b - 1e-6, 1000)
        M2 = np.max(np.abs(f2(x_vals)))
        error = (b - a) / 24 * h**2 * M2

        if error < eps:
            return total, n, h, error
        n *= 2
I_rect, n_rect, h_rect, err_rect = rectangles_method(f, f2, a, b, eps)
print(I_rect)
```

Выводится строка:
-0.08641215676281004 - приближенное численное значение интеграла по методу прямоугольников с погрешностью $\epsilon = 0,00001$

Метод трапеций

Узлы: $x_i = a + ih$, где $h = \frac{b-a}{n}$, $i = 0, \ldots, n$

Формула:

$$I_n = \frac{h}{2}\left(f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)\right)$$

Погрешность:

$$|\Delta| \leq \frac{b-a}{12}h^2 \cdot \sup_{x\in[a,b]} |f''(x)|$$

```
import numpy as np

def f(x):
    return np.log(np.sin(x))

def f2(x):
    return -1 / (np.sin(x)**2) - 1 / (np.tan(x)**2)


a = np.pi / 4
b = np.pi / 2
eps = 1e-5


def trapezia_method(f, f2, a, b, eps):
    n = 1
    while True:
        h = (b - a) / n
        x = np.linspace(a, b, n + 1)
        y = f(x)
        total = h * (y[0] + 2 * np.sum(y[1:-1]) + y[-1]) / 2

        x_vals = np.linspace(a + 1e-6, b - 1e-6, 1000)
        M2 = np.max(np.abs(f2(x_vals)))
        error = (b - a) / 12 * h**2 * M2

        if error < eps:
            return total, n, h, error
        n *= 2
I_trap, n_trap, h_trap, err_trap = trapezia_method(f, f2, a, b, eps)
print(I_trap)
```

Выводится строка:
-0.08641686294215971 - приближенное численное значение интеграла по методу трапеций с погрешностью $\epsilon = 0,00001$

Метод Симпсона

Требует чётное число разбиений $n = 2m$. Формула:

$$I_n = \frac{h}{3}\left(f(x_0) + 2\sum_{j=1}^{m-1} f(x_{2j}) + 4\sum_{j=1}^{m} f(x_{2j-1}) + f(x_n)\right)$$

Погрешность:

$$|\Delta| \leq \frac{b-a}{180}h^4 \cdot \sup_{x\in[a,b]} |f^{(4)}(x)|$$

```
import numpy as np

def f(x):
    return np.log(np.sin(x))

def f2(x):
    return -1 / (np.sin(x)**2) - 1 / (np.tan(x)**2)
def f4(x):
    sinx = np.sin(x)
    cosx = np.cos(x)
    return (
        6 * cosx**2 / sinx**4 +
        8 * cosx**2 / sinx**2 +
        2 / sinx**2 +
        2 / np.tan(x)**2
    )

a = np.pi / 4
b = np.pi / 2
eps = 1e-5


def simpson_method(f, f4, a, b, eps):
    n = 2
    while True:
        h = (b - a) / n
        x = np.linspace(a, b, n + 1)
        y = f(x)
        total = h/3 * (y[0] + 4 * np.sum(y[1:n:2]) + 2 * np.sum(y[2:n-1:2]) + y[n])

        x_vals = np.linspace(a + 1e-4, b - 1e-4, 1000)
        M4 = np.max(np.abs(f4(x_vals)))
        error = (b - a) / 180 * h**4 * M4

        if error < eps:
            return total, n, h, error
        n *= 2
I_simp, n_simp, h_simp, err_simp = simpson_method(f, f4, a, b, eps)
print(I_simp)
```

Выводится число:
-0.08641385377900195 - численное приближенное значение интеграла вычисленное методом Симпсона с погрешностью $\epsilon = 0,00001$