

ООPython

Задача 3. Численное интегрирование

Введение

В лекционном материале `lecture_7_numerical_integration.ipynb` рассматривалось представление первообразной F функции f в виде класса **Integral**, экземпляры которого являются функторами. В методе `__call__` данного класса реализованы вычисления по формуле трапеций для приближенного вычисления интеграла:

$$F(x; a, f) = \int_a^x f(t) dt$$
$$F(x; a, f) \approx F_{num} = h \left(\frac{f(t_0) + f(t_N)}{2} + \sum_{i=1}^{N-1} f(t_i) \right), t_i = a + ih \quad (1)$$

Существует множество формул различных порядков точности для расчета F_{num} .

Ограничимся следующими:

- формула правых прямоугольников: $F_{num} = h \sum_{i=1}^N f(t_i), t_i = a + ih$ (2)

- формула левых прямоугольников: $F_{num} = h \sum_{i=0}^{N-1} f(t_i), t_i = a + ih$ (3)

- формула средних прямоугольников: $F_{num} = h \sum_{i=0}^{N-1} f(t_i), t_i = a + \frac{h}{2} + ih$ (4)

- формула трапеций: $F_{num} = h \left(\frac{f(t_0) + f(t_N)}{2} + \sum_{i=1}^{N-1} f(t_i) \right), t_i = a + ih$ (5)

- формула Симпсона:

$$F_{num} = \frac{h}{3} (f(t_0) + f(t_{2m}) + 4\sigma_1 + 2\sigma_2)$$

$$\sigma_1 = \sum_{i=0}^{m-1} f(t_{2i+1}), \sigma_2 = \sum_{i=1}^{m-1} f(t_{2i}), 2m = N - \text{четное число отрезков разбиения}$$

(6)

Любую из вышеприведенных формул можно записать в общем виде:

$$F_{num} = h \sum_i \alpha_i f(t_i) \quad (7)$$

где $\{\alpha_i\}$ и $\{t_i\}$ - множества коэффициентов и точек расчетной сетки.

Задание

Определение классов

Для каждой из формул (2)-(6) реализовать соответствующий класс **IntegralName**, минимизировав суммарное число строк кода с помощью наследования. Каждый класс должен включать в себя как минимум следующее:

Поля:

- интегрируемая функция f
- левая граница отрезка a
- число отрезков разбиения N

Методы:

- конструктор
- «сеттеры» для f , a , N
- задать коэффициенты $\{\alpha_i\}$ и точки расчетной сетки $\{x_i\}$
- вычислить значение $F(x)$ по формуле (7) (реализовать через магический метод `__call__`)

Использование классов

Использовать созданные классы для приближенного вычисления интегралов на отрезке $[0, 2]$. Для каждой формулы (2)-(6) построить графики абсолютного значения

погрешности в зависимости от числа отрезков интегрирования $N_n = 2^{n+1}$, $n = 0, 1, \dots, 14$ для следующих подынтегральных функций:

- $f(x) = \frac{5}{2 + 3x^2}$
- $f(x) = \frac{2}{5 + \cos(x)}$
- $f(x) = \sqrt[3]{3 + 4x^2}$
- $f(x) = \frac{2}{\sqrt{\pi}} \exp(-x^2)$

- $f(x) = \frac{\ln(x+2)}{x+2}$

Использовать логарифмический масштаб по обеим осям. Графики погрешностей для каждой функции строить в отдельном окне. Задействовать библиотеку **SymPy** для аналитического вычисления интегралов. (**Аналогично Заданию 2**).

Подсказка: для ускорения расчета по формуле (7) вместо цикла можно использовать функцию **numpy.dot(a, b)**, вычисляющую скалярное произведение векторов **a** и **b**.