OOPython

Задача 3. Численное интегрирование

Введение

В лекционном материале **lecture_6_numerical_integration.ipynb** рассматривалось представление первообразной F функции f в виде класса **AntiderivativeNum**, экземпляры которого являются функторами. В методе **__call__** данного класса реализованы вычисления по формуле трапеций для приближенного вычисления первообразной функции f через интеграл:

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

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

- формула правых прямоугольников: $F_{num}=h\sum_{i=1}^{N}f\left(t_i\right),\ t_i=a+ih$ (2)
- формула левых прямоугольников: $F_{num}=h\sum_{i=0}^{N-1}f\left(t_i\right),\ t_i=a+ih$ (3)
- формула средних прямоугольников: $F_{num}=h\sum_{i=0}^{N-1}f\left(t_i\right),\ t_i=a+\frac{h}{2}+ih$ (4)
- формула трапеций: $F_{num}=h\left(\frac{f\left(t_0\right)+f\left(t_N\right)}{2}+\sum_{i=1}^{N-1}f\left(t_i\right)\right),\ t_i=a+ih$ (5)
- формула Симпсона:

$$F_{num}=\frac{h}{3}\left(f\left(t_0\right)+f\left(t_{2m}\right)+4\sigma_1+2\sigma_2\right)$$
$$\sigma_1=\sum_{i=0}^{m-1}f\left(t_{2i+1}\right),\ \sigma_2=\sum_{i=1}^{m-1}f\left(t_{2i}\right),\ 2m=N-\text{четное число отрезков разбиения}$$

(6)

Любую из вышеприведенных формул можно записать в общем виде:

$$F_{num}=h\sum_i\alpha_i f\left(t_i\right) \tag{7}$$

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

Задание

Определение классов

Для каждой из формул (2)-(6) реализовать соответствующий класс **Antiderivative***Name*, 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 $\Omega = [0; 2]$. Для каждой формулы (2)-(6) построить графики абсолютного значения погрешности в зависимости от числа отрезков интегрирования $N_n = 2^{n+1}$, $n = 0,1,...,14$ для следующих подынтегральных функций:

- $f(x) = \sin(x^2)$
- $f(x) = \cos(\sin(x))$
- $f(x) = \exp(\cos(\sin(x)))$
- $f(x) = \ln(x + 3)$
- $f(x) = \sqrt{x + 3}$

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

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