ЛР 5. Интегрирование функций. Формулы трапеций, Симпсона

ПИН-21 Чендемеров Алексей

June 3, 2021

1 ЛР 5. Интегрирование функций. Формулы трапеций, Симпсона

[1]:
```
import numpy as np
import sympy as sp
import math
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams["figure.figsize"] = (15,10)
plt.rcParams['lines.linewidth'] = 2
```

1.1 Задание 1

Задайте функцию $f(x) = x^3$ на отрезке $[0, 1]$. Очевидно, определённый интеграл от функции $f(x)$ на этом отрезке равен $\frac{1}{4}$. Напишите программу, вычисляющую значение интеграла по формулам трапеций и Симпсона. Какую максимальную теоретическую ошибку мы при этом допускаем? Найдите реальное значение погрешности (абсолютное значение разности между теоретическим и аналитическим решением). Почему при вычислении интеграла по формуле Симпсона от данной функции ошибка равна нулю? Какие бы получились значения погрешностей для квадратичной и линейной функций (предположите и проведите численный эксперимент для $f_2(x) = x^2$, $f_1(x) = x/2$ на отрезке $[0, 1]$).

[2]:
```
from typing import TypedDict

x = sp.Symbol('x')
f = x ** 3
f1 = x / 2
f2 = x ** 2
f3 = x ** 4

def trap(f_sym, a, b, n=None):
    x = np.linspace(a, b) if n is None else np.linspace(a, b, n)
    I = 0
    f = sp.lambdify(sp.Symbol('x'), f_sym)
    df2 = sp.lambdify(sp.Symbol('x'), sp.diff(f_sym, sp.Symbol('x'), 2))
    for i in range(1, len(x)):
        I += (f(x[i-1]) + f(x[i])) * (x[i] - x[i-1]) / 2
```

```python
    M2 = max(abs(np.array(list(map(df2, x)))))
    R = (x[i] - x[i-1])** 2 * (b - a) * M2 / 12
    return dict(val=I, R=R, h=x[1] - x[0])

def quad(f_sym, a, b, n=None):
    x = np.linspace(a, b) if n is None else np.linspace(a, b, n)
    I = 0
    f = sp.lambdify(sp.Symbol('x'), f_sym)
    df4 = sp.lambdify(sp.Symbol('x'), sp.diff(f_sym, sp.Symbol('x'), 4))
    for i in range(1, len(x)):
        h = x[i] - x[i-1]
        I += h / 6 * (f(x[i-1]) + 4 * f(x[i-1] + h/2)  + f(x[i]))
    M4 = max(abs(np.array(list(map(df4, x)))))
    R = (x[i] - x[i-1]) ** 4 * (b - a) * M4 / 2880
    return dict(val=I, R=R, h=x[1] - x[0])


print("f = x ** 3")
print("trap", trap(f,0,1))
print("quad", quad(f,0,1))
print()

print("f1 = x / 2")
print("trap", trap(f1,0,1))
print("quad", quad(f1,0,1))
print()

print("f2 = x ** 2")
print("trap", trap(f2,0,1))
print("quad", quad(f2,0,1))
print()

print("f3 = x ** 4")
print("trap", trap(f3,0,1))
print("quad", quad(f3,0,1))
print()
```

```
f = x ** 3
trap {'val': 0.25010412328196585, 'R': 0.00020824656393169557, 'h':
0.02040816326530612}
quad {'val': 0.25000000000000006, 'R': 0.0, 'h': 0.02040816326530612}

f1 = x / 2
trap {'val': 0.25, 'R': 0.0, 'h': 0.02040816326530612}
quad {'val': 0.25, 'R': 0.0, 'h': 0.02040816326530612}

f2 = x ** 2
trap {'val': 0.33340274885464394, 'R': 6.94155213105652e-05, 'h':
```

```
0.02040816326530612}
quad {'val': 0.3333333333333333, 'R': 0.0, 'h': 0.02040816326530612}

f3 = x ** 4
trap {'val': 0.20013882526040358, 'R': 0.00041649312786339114, 'h':
0.02040816326530612}
quad {'val': 0.2000000014455544, 'R': 1.4455543796452592e-09, 'h':
0.02040816326530612}
```

Ошибка при вычислении по формуле Симпсона равна, т.к. формула Симпсона точна для любого многочлена 3 степени. Для оценки погрешности формулы Симпсона используется соотношение $|\Psi| \leq \frac{h^4(b-a)}{2880} M_4$, где $M_4 = max_{[a;b]}|f^{IV}(x)|$ (см. лекцию 6). Для любого многочлена 3 степени $f^{IV}(x) = 0$. Следовательно, $M_4 = 0$ и $|\Psi| = 0$

1.2 Задание 2

Используя соотношение $\int_0^1 \frac{1}{1+x^2} dx = arctg(1)$ найдите значение числа π с точностью 10^{-6} . В данном задании в процессе вычислений нельзя использовать встроенную константу pi для определения величины шага. Из каких соображений выбирался шаг для получения указанной точности?

[3]:
```python
x = sp.Symbol('x')
f = 1 / (1 + x**2)

val, R, h = None, None, None
n = 2
while True:
    ret = quad(f, 0, 1, n)
    val, R, h = ret['val'], ret['R'], ret['h']
    n += 1
    if R < 1e-6:
        break
pi = val * 4
err = abs(np.pi - pi)
print(f"Pi calculated with integral (h = {h}): {pi}")
print(f"Built-in pi: {np.pi}")
print(f"Eror: {err}")
```

```
Pi calculated with integral (h = 0.1): 3.141592652969785
Built-in pi: 3.141592653589793
Eror: 6.200080449048073e-10
```

Т.к. $arctg(1) = \frac{\pi}{4}$, то $\pi = 4 * arctg(1)$. Для нахождения значения с необходимой точностью, мы производим численное интегрирование с такой точностью, а из оценки точности интегрирования выбирается шаг.

1.3 Задание 3

Реализовать предыдущее задание, определяя точность методом Рунге. При численном вычислении интегралов последовательно с шагами h и $\frac{h}{2}$ можно сократить число арифметических операций. Заметим, что приближённое значение интеграла $I_{\frac{h}{2}}$ есть сумма, часть слагаемых которой возможно уже участвовало при вычислении I_h . Поэтому можно получить $I_{\frac{h}{2}}$, используя числовое значение I_h . Это позволяет избежать повторного суммирования части слагаемых.

[4]:
```
x = sp.Symbol('x')
f_sym = 1 / (1 + x**2)
f = sp.lambdify(x, f_sym)

a = 0
b = 1

h = 1
R = 1e100
while R >= 1e-6:
    Ih = h * (f(a)/2 + sum(map(f, np.arange(a+h, b-h, h))) + f(b) / 2)
    Ih2 = Ih/2 + h * sum(map(f, np.arange(a+h, b-h, 2*h)))
    R = abs(Ih - Ih2)
    h /= 2
pi = Ih * 4

print(f"Pi calculated with integral (h = {h}): {pi}")
print(f"Built-in pi: {np.pi}")
print(f"Eror: {R}")
```

```
Pi calculated with integral (h = 1.9073486328125e-06): 3.1415850241637644
Built-in pi: 3.141592653589793
Eror: 9.536770502194258e-07
```

Использовал оценку без использования точного значения (как в БДЗ-2)