Национальный исследовательский университет ИТМО
Факультет информационных технологий и программирования
Прикладная математика и информатика

Численное дифференцирование и интегрирование

Отчёт по лабораторной работе №1

Работу выполнили:
Гуревич Михаил
Трохан Александр

Преподаватель:
Москаленко Мария Александровна

Санкт-Петербург
2023

Лабораторная работа №1

Выполнили: Гуревич Михаил и Трохан Александр, M32001

✅ Полный код лабораторной работы с комментариями можно найти на Github, а отчёт в Notion.

Цель работы

Изучить методы численного дифференцирования и интегрирования.

Постановка задачи

- Реализовать методы численного дифференцирования: метод правой разностной производной, метод левой разностной производной и метод центральной разностной производной.
- Для двух произвольно выбранных функций вычислить аналитические производные и сравнить результаты с численными производными.
- Определить зависимость среднеквадратичного отклонения метода от размера шага (h). Построить график.
- Реализовать методы численного дифференцирования: формула прямоугольников, формула трапеций и формула Симпсона.
- Для двух произвольно выбранных функций вычислить аналитические определённые интегралы и сравнить результаты с численным интегрированием.
- Определить зависимость среднеквадратичного отклонения метода от размера шага (h). Построить график.

Ход работы

Дифференцирование

Реализуем приведённые в Теории методы численного дифференцирования (а именно метод правой разностной производной, метод левой разностной производной и метод центральной разностной производной) с помощью **Python**:

```
def right_diff(f, x, h):
    return (f(x + h) - f(x)) / h

def left_diff(f, x, h):
    return (f(x) - f(x - h)) / h

def central_diff(f, x, h):
    return (f(x + h) - f(x - h)) / (2 * h)

def central_diff_leftmost(f, x, h):  # для самой левой точки
    return (-3 * f(x) + 4 * f(x + h) - f(x + 2 * h)) / (2 * h)

def central_diff_rightmost(f, x, h):  # для самой правой точки
    return (3 * f(x) - 4 * f(x - h) + f(x - 2 * h)) / (2 * h)
```

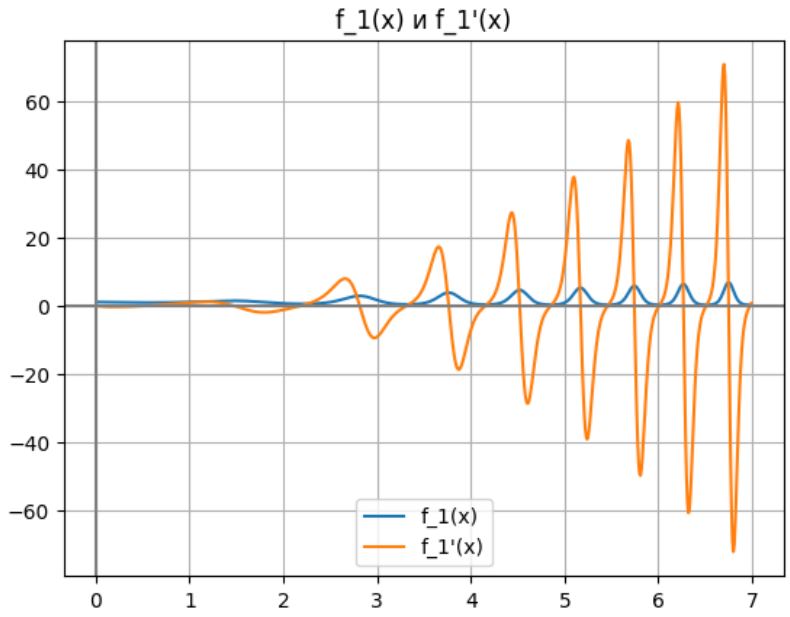
Выберем две произвольные функции. В нашем случае это будут функции $f_1(x) = x^{\sin(x^2)}$ и $f_2(x) = x^4 \cos(10x^4)$.

Далее рассмотрим работу с каждой из этих функций по отдельности.

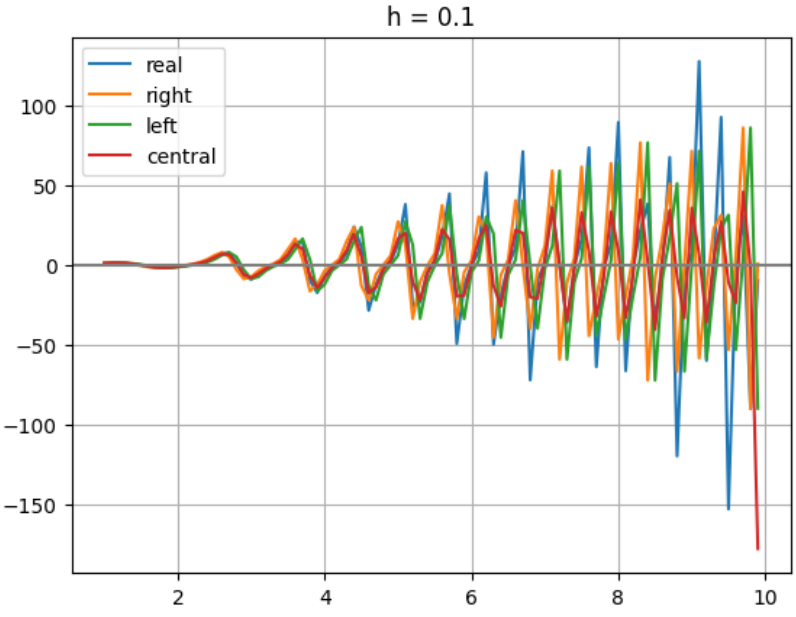
Функция $f_1(x) = x^{\sin(x^2)}$

Аналитическая производная функции: $x^{(\sin(x^2)-1)}(2x^2\cos(x^2)\ln(x) + \sin(x^2))$.

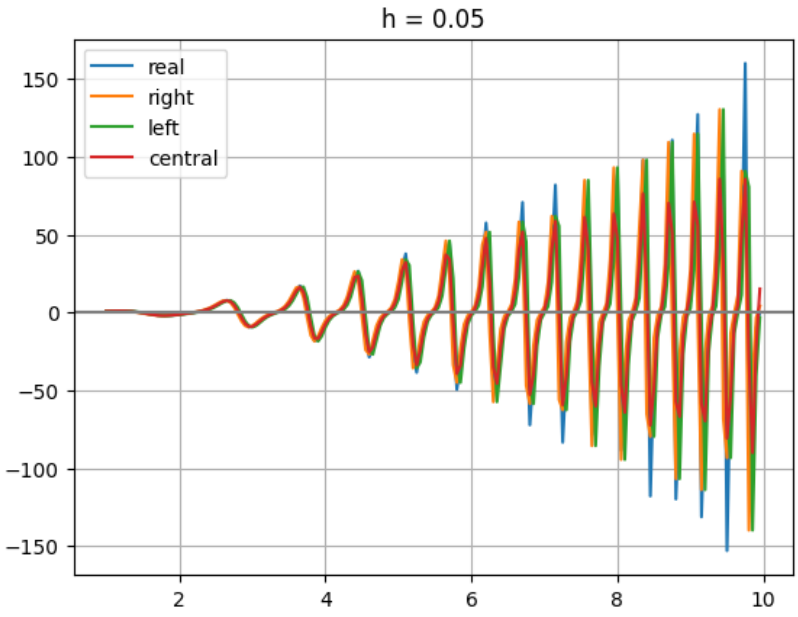
График функции и её производной:

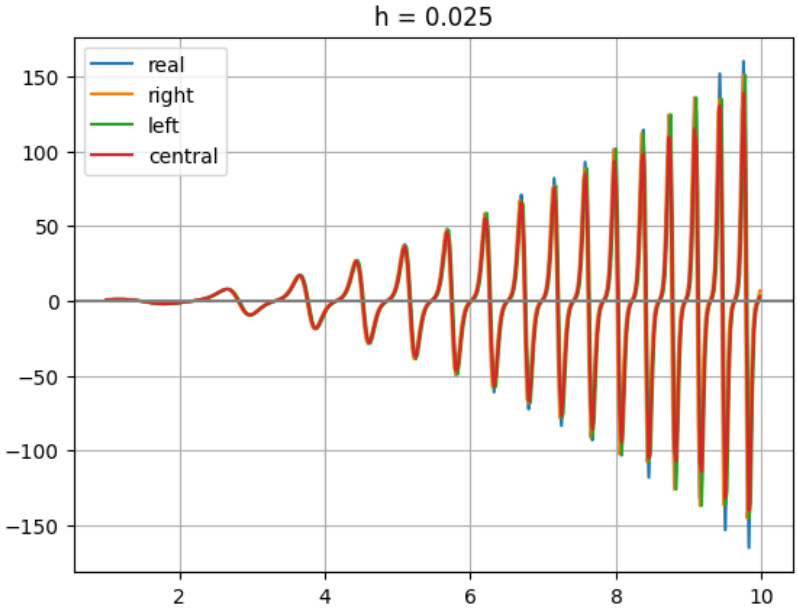


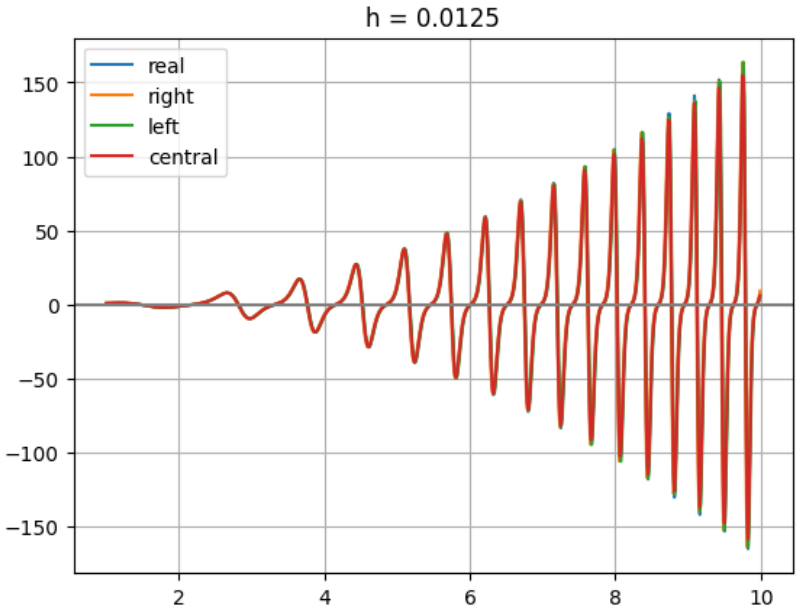
Найдём численное значение производной в узлах сетки на отрезке $[1; 10]$ при шаге $h = 0.1$. Для удобства представим эти значения в виде графика. На графике покажем реальное значение производной в этой точке, и значения производной вычисленное всеми тремя методами:

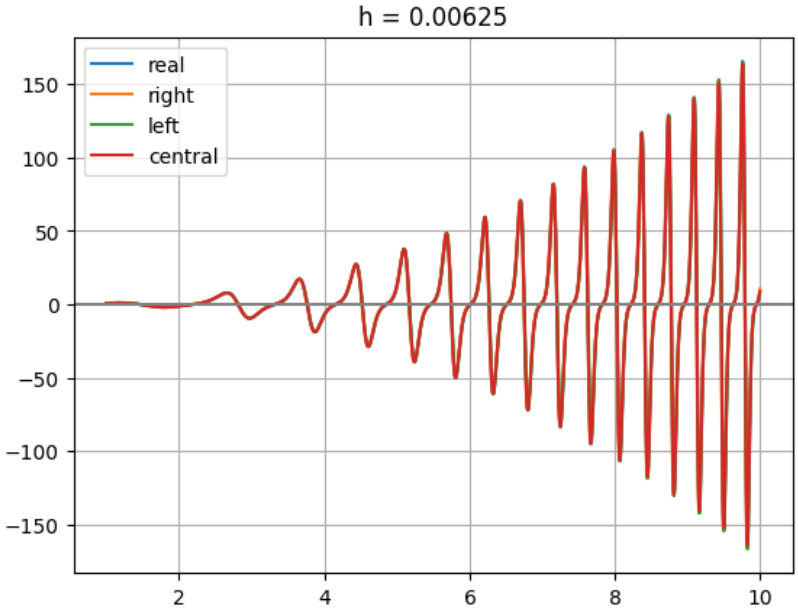


Теперь проведём аналогичные вычисления для других размеров шага, которые меньше в 2, 4, 8 и 16 раз:





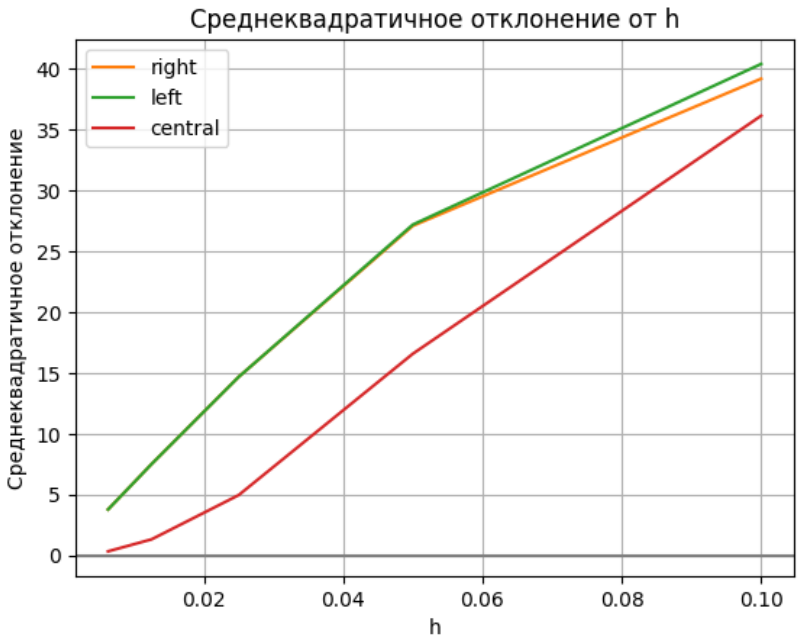




Теперь для каждого размера шага определим среднеквадратичное отклонение. Для этого реализуем функцию для вычисления среднеквадратичного отклонения:

```
def standard_deviation(real, approx):
    return np.sqrt(np.mean(np.power(real - approx, 2)))
```

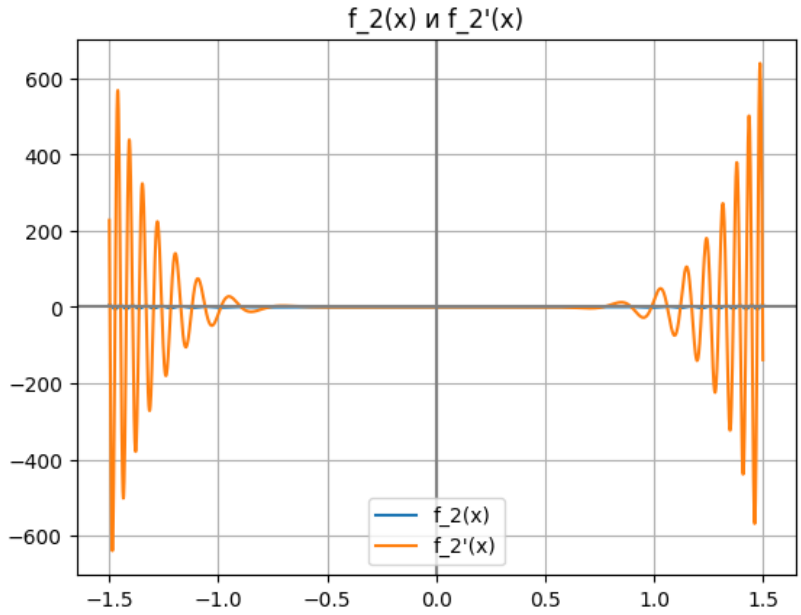
Представим вычисленные значения в виде графика:



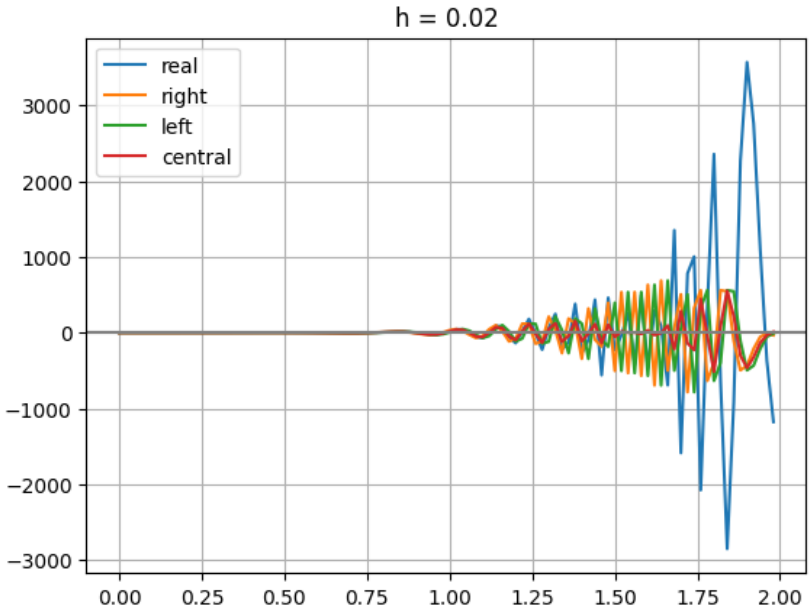
Функиця $f_2(x) = x^4 \cos(10x^4)$

Аналитическая производная функции: $4x^3 \cos(10x^4) - 40x^7 \sin(10x^4)$.

График функции и её производной:

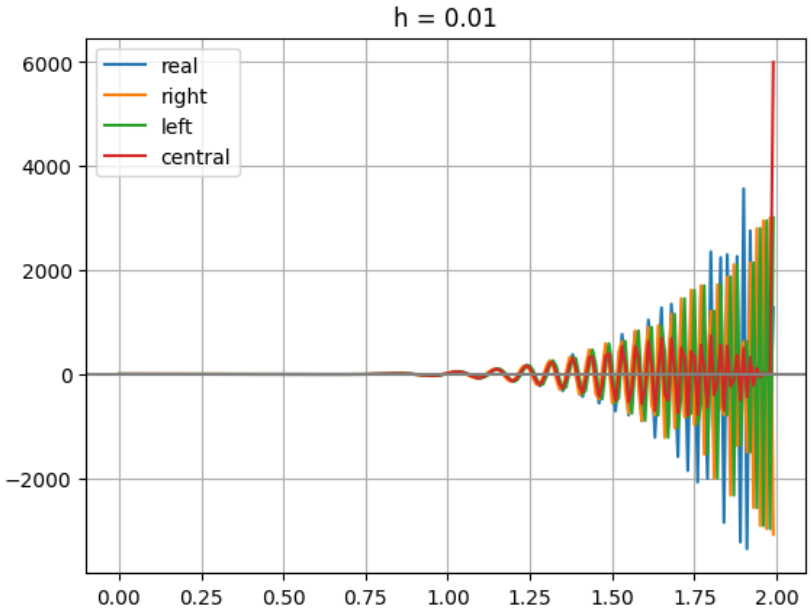


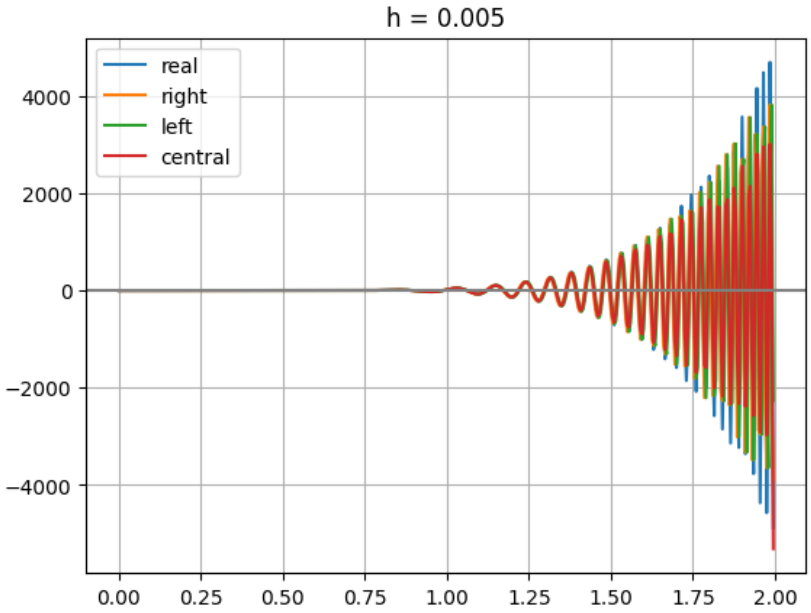
Найдём численное значение производной в узлах сетки на отрезке $[0; 2]$ при шаге $h = 0.02$. Для удобства представим эти значения в виде графика. На графике покажем реальное значение производной в этой точке, и значения производной вычисленное всеми тремя методами:

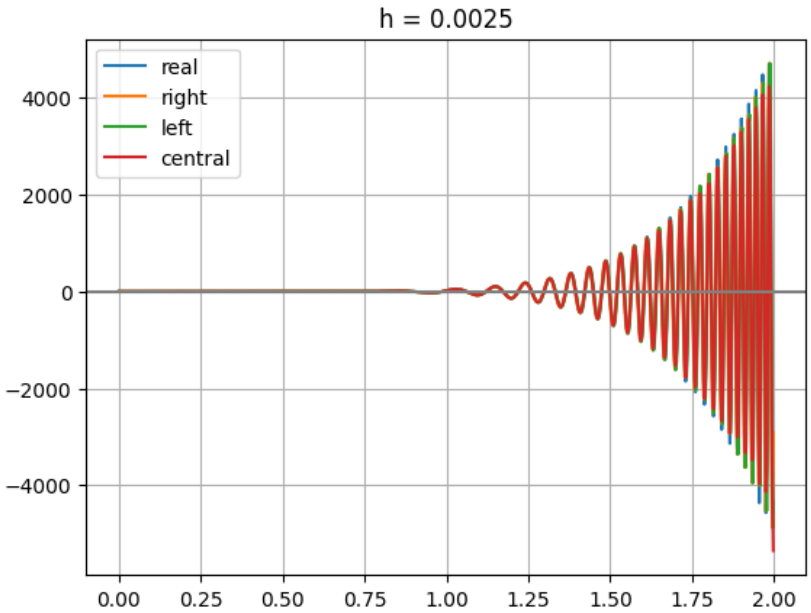


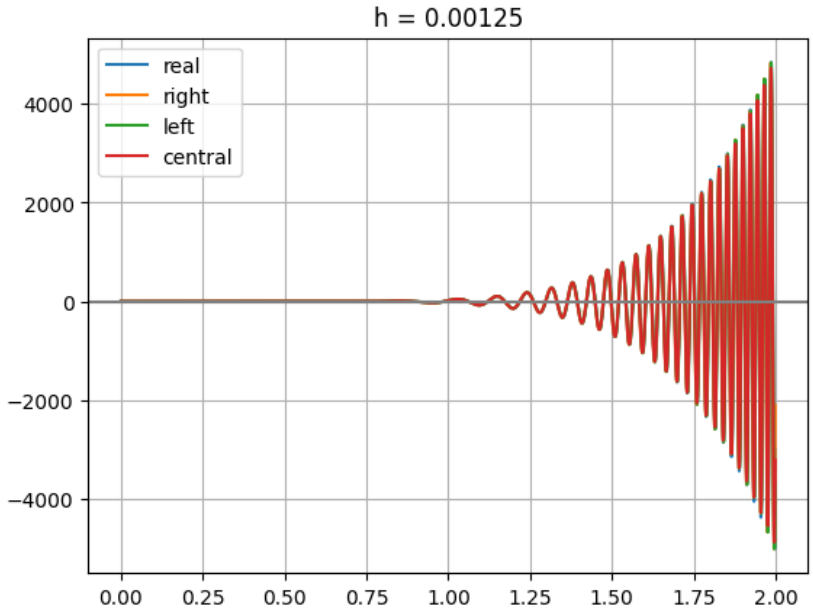
Можно заметить, что численные методы начинают давать очень сильное отклонение от реального результата при увеличении x.

Теперь проведём аналогичные вычисления для других размеров шага, которые меньше в 2, 4, 8 и 16 раз:

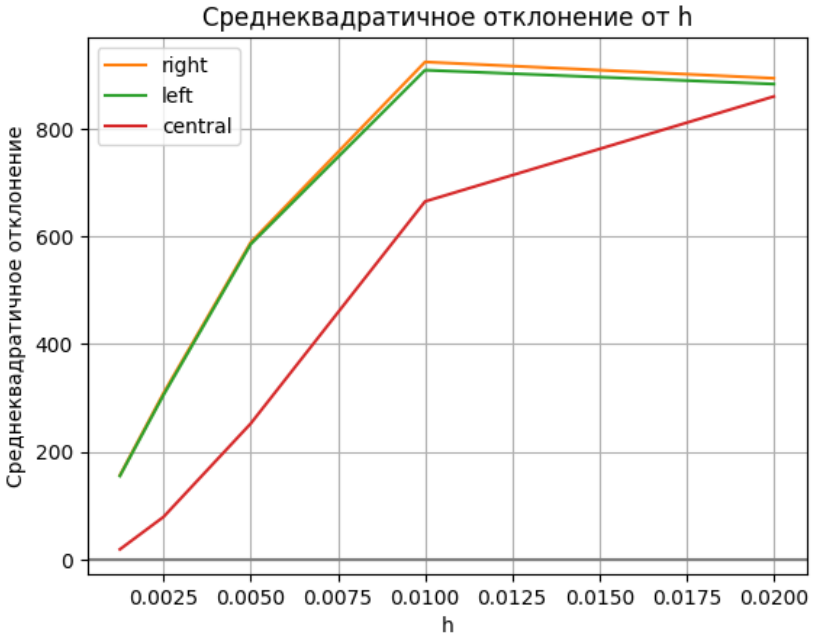



h = 0.005
real
right
left
central
4000
2000
0
−2000
−4000
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00


h = 0.0025
real
right
left
central
4000
2000
0
−2000
−4000
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00



Теперь для каждого размера шага определим среднеквадратичное отклонение. Представим вычисленные значения в виде графика:



Выводы по дифференцированию

Были реализованы три метода численного дифференцирования, и эти методы были применены для нахождения значения производной двух функций. Было установлено, что при большом шаге (т. е. при малом количестве узлов сетки) все три метода численного дифференцирования обладают большой погрешностью, о чём можно судить из того, что их среднеквадратичное отклонение от реального значения велико (около 20-25%). Было установлено, что при уменьшении шага отклонение всех трёх методов уменьшается и со временем стремится к нулю.

Можно отметить, что метод центральной разностной производной всегда имел меньшее отклонение относительно метода левой и правой разностной производной. Помимо этого, чем больше колебания функции на отрезке, тем менее точными являются все численные методы: это хорошо заметно при наименьшем шаге для функции $f_2(x)$ — в этом случае значение полученное любым из трёх методов сильно отличается от реального.

Интегрирование

Реализуем приведённые в Теории методы численного интегрирования. Каждый метод использует различные формулы для вычисления площадей криволинейных трапеций, на которые разбивается исходный интеграл. Поэтому нами будут реализованы следующие формулы: формула левых, правых и средних прямоугольников, формула трапеций и формула Симпсона. Методы реализуем с помощью **Python**:

```
def left_rectangles(f, a, b, h):
    return np.sum(f(np.arange(a, b, h)) * h)

def right_rectangles(f, a, b, h):
    return np.sum(f(np.arange(a + h, b + h, h)) * h)

def central_rectangles(f, a, b, h):
    return np.sum(f(np.arange(a + h / 2, b + h / 2, h)) * h)

def trapezoidal(f, a, b, h):
    return h / 2 * (f(a) + f(b) + 2 * np.sum(f(np.arange(a + h, b, h))))

def simpson(f, a, b, h):
    return h / 3 * (f(a) + f(b) + 4 * np.sum(f(np.arange(a + h, b, 2 * h))) + 2 * np.sum(f(np.arange(a + 2 * h, b, 2 * h))))
```

Так как функции $f_1(x)$ и $f_2(x)$ не имеют первообразной, выраженной через элементарные функции, то выберем вместо них новые функции $f_3(x)$ и $f_4(x)$, такие что $f_3(x) = x\sin(x^2)$, $f_4(x) = x\cos(x^2) \cdot e^{\sin(x^2)}$.

Далее рассмотрим работу с каждой из этих функций по отдельности.

Функция $f_3(x) = x\sin(x^2)$

В данном случае будем рассматривать определённый интеграл $\int_0^{10} x\sin(x^2)dx$.

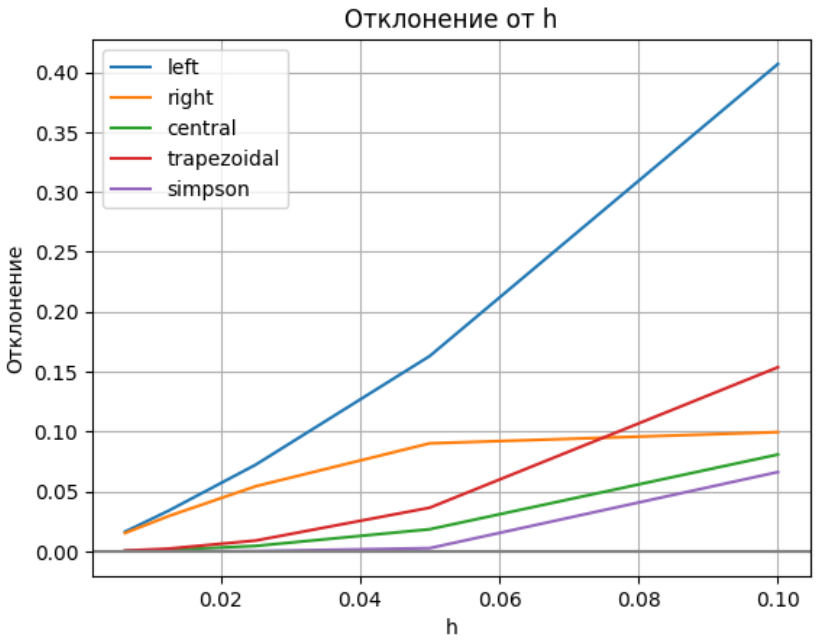
Первообразная к функции $f_3(x)$: $F_3(x) = -\frac{1}{2}\cos(x^2)$ (без учёта константы).

Анлитически вычисленный интеграл: $\int_0^{10} x\sin(x^2)dx = \left(-\frac{1}{2}\cos(x)\right)\Big|_0^{10} = \sin^2(50) \approx 0.068841$.

Найдём численное значение интеграла начиная с шага 0.1. Результаты приведены в таблице ниже:

h	реальное значение	формула левых прямоугольников	формула правых прямоугольников	формула центральных прямоугольников	формула трапеций	формула Симпсона
0.1	0.0688	0.4757	-0.0307	-0.0120	0.2225	0.0026
0.05	0.0688	0.2319	-0.0213	0.0504	0.1053	0.0662
0.025	0.0688	0.1411	0.0145	0.0643	0.0778	0.0687
0.0125	0.0688	0.1027	0.0394	0.0677	0.0711	0.0688
0.00625	0.0688	0.0852	0.0536	0.0686	0.0694	0.0688

Также построим график зависимости отклонения от шага h:



Функция $f_4(x) = x\cos(x^2)\cdot e^{\sin(x^2)}$

В данном случае будем рассматривать определённый интеграл $\int_5^{20} x\cos(x^2)\cdot e^{\sin(x^2)}dx$.

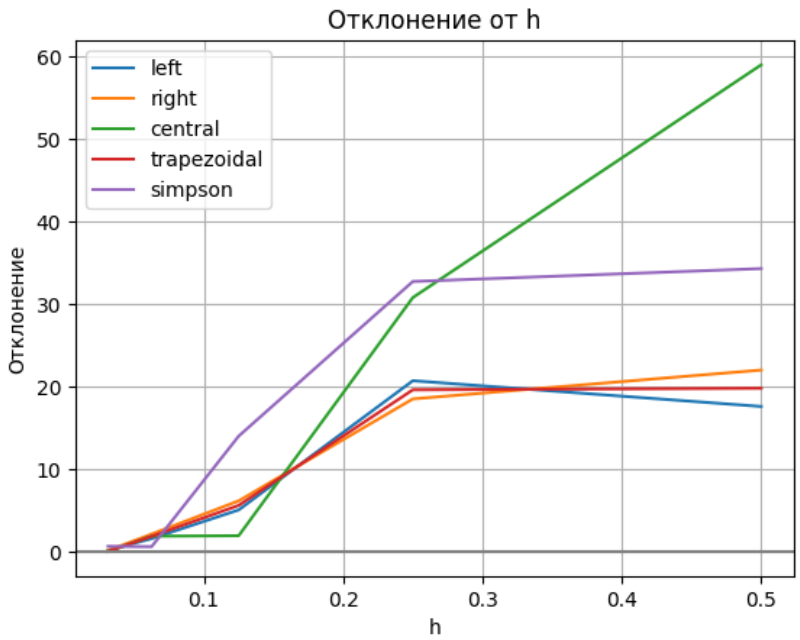
Первообразная к функции $f_4(x)$: $F_4(x) = \frac{1}{2}e^{\sin(x^2)}$ (без учёта константы).

Анлитически вычисленный интеграл: $\int_5^{20} x\cos(x^2)\cdot e^{\sin(x^2)}dx = \frac{1}{2}e^{\sin(x^2)}\Big|_5^{20} = \frac{1}{2}(e^{\sin(400)} - e^{\sin(25)}) \approx -0.22451$.

Найдём численное значение интеграла начиная с шага 0.05. Результаты приведены в таблице ниже:

h	реальное значение	формула левых прямоугольников	формула правых прямоугольников	формула центральных прямоугольников	формула трапеций	формула Симпсона
0.5	-0.2245	-17.8174	-22.2314	58.7998	-20.0244	-34.5363
0.25	-0.2245	20.4912	18.2842	-31.0247	19.3877	32.5251
0.125	-0.2245	-5.2667	-6.3702	1.6929	-5.8185	-14.2205
0.0625	-0.2245	-1.7869	-2.3387	1.6382	-2.0628	-0.8109
0.03125	-0.2245	-0.0744	-0.3503	-0.2232	-0.2123	0.4045

Также построим график зависимости отклонения от шага h:



Выводы по интегрированию

Были реализованы пять методов численного интегрирования, и эти методы были применены для нахождения значения определённых интегралов двух функций. Как и в случае с дифференцированием, было установлено, что среднеквадратичное отклонение прямо пропорционально величине шага h, а также при стремлении h к нулю среднеквадратичное отклонение тоже стремится к нулю.

Что касается самих методов, то можно отметить, что ни один из них не является однозначно точнее других. В случае функции $f_1(x)$ самым точным оказался метод, использующий формулу Симпсона, а самым неточным — формулу левых прямоугольников. Однако в случае с $f_2(x)$ ситуация практически полностью противоположна (формула Симпсона находится на втором месте с конца, а формула левых прямоугольников — самая точная). Из этого можно сделать вывод, что при случайном выборе функции и пределов интегрирования ни один из численных методов интегрирования не будет иметь гарантированного преимущества над другими.

Аналогично численным методам дифференцирования, существенные и частые флуктуации в значении функции ведут к более существенным погрешностям численных методов.

Выводы

Были реализованы различные методы численного дифференцирования и интегрирования, и эти методы были применены для нахождения значения производных определённых интегралов различных функций. Было установлено, что при стремлении величины шага h к нулю отклонение от реального результата всех реализованных численных методов также стремится к нулю.

Можно отметить, что чем быстрее меняется значение функции и чем больше абсолютная разность между двумя соседними значениями, тем менее точными оказываются численные методы.

Также было выявлено, что метод центральной разностной производной имеет меньшее отклонение относительно других реализованных методов численного дифференцирования. Однако для численного интегрирования лучшего метода выявлено не было.