Практическая работа № 4

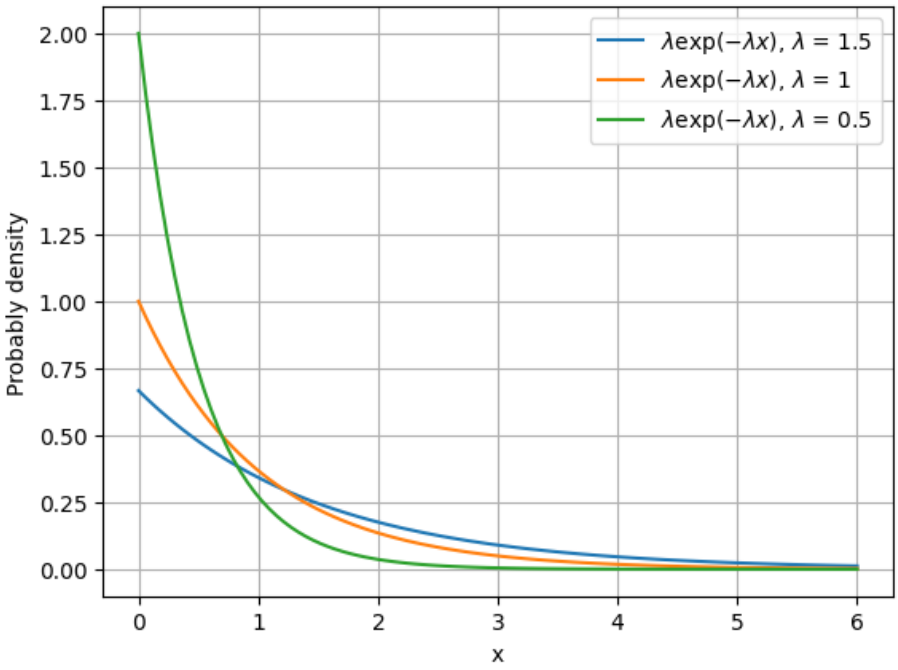
Валерий Сергеевич Верхотуров БСБО-05-20

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import expon
import random
from IPython.core.display_functions import display
```

Задание 1

Создать функцию, описывающую распределение плотности вероятности для экспоненциального закона распределения случайной величины. Построить графики плотности вероятности экспоненциального закона с заданными значениями параметра масштаба λ : $\lambda = 1.5$, $\lambda = 1$ и $\lambda = \frac{1}{2}$.

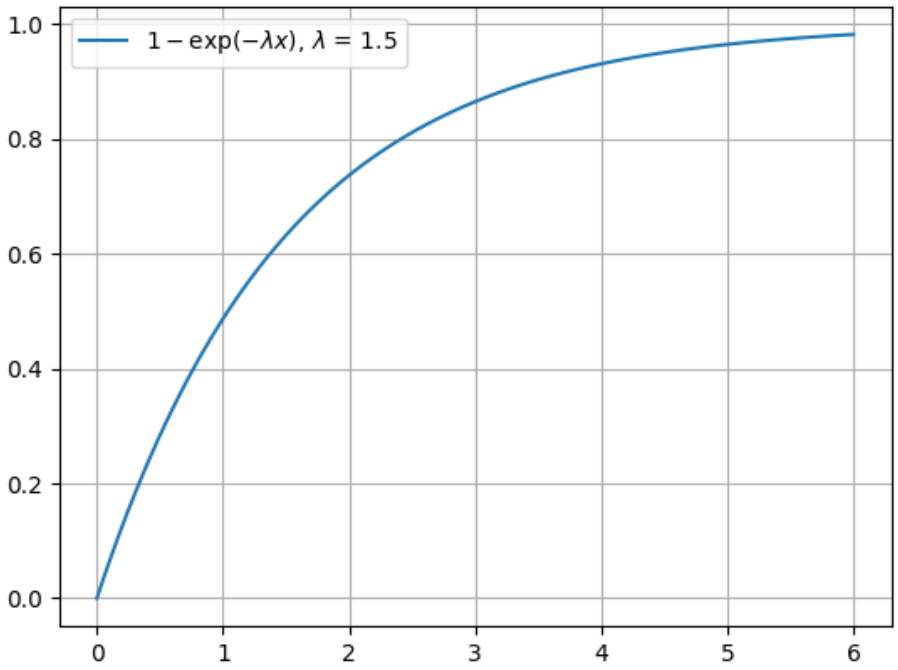
```
def task_1():
    scales = [1.5, 1, 0.5]
    x = np.linspace(0, 6, 100)
    for scale in scales:
        plt.plot(x, expon.pdf(x, scale=scale), label=f'$\lambda \exp(-\lambda x)$, $\lambda$
    plt.legend()
    plt.grid(True)
    plt.ylabel('Probably density')
    plt.xlabel('x')
    plt.show()

task_1()
```



Задание 2

Описать функцию обратного преобразования для экспоненциального закона с параметром $\lambda = 1.5$ и построить её график.

```
def task_2():
    scale = 1.5

    x = np.linspace(0, 6, 100)

    # Cumulative distribution function
    plt.plot(x, expon.cdf(x, scale=scale), label=f'$1-\exp(-\lambda x)$, $\lambda$ = {scale}
    plt.legend()
    plt.grid(True)

task_2()
```



Задания 3, 4

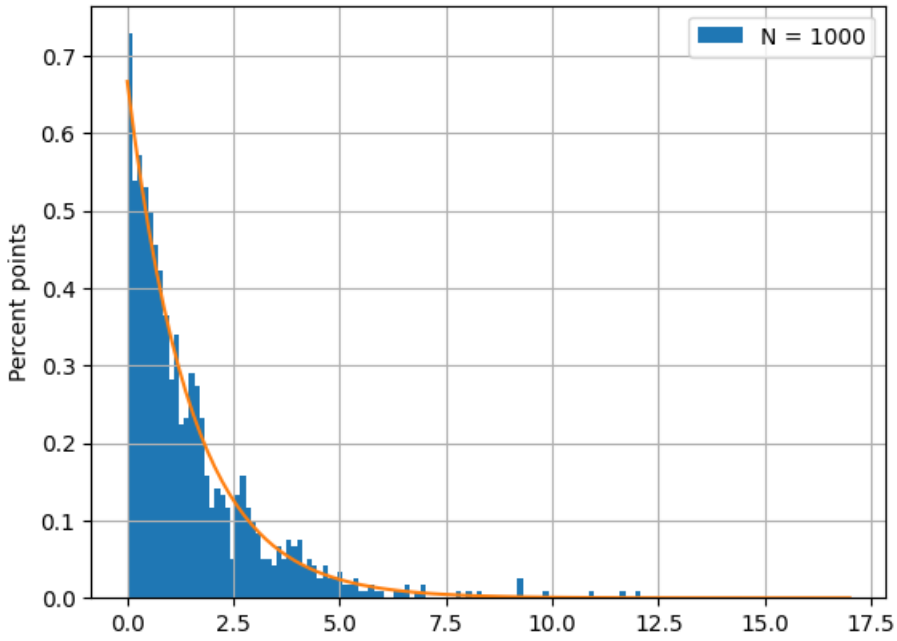
С помощью известной функции обратного преобразования (см. задание 2) провести моделирование экспоненциального закона распределения с помощью метода обратной функции. Для моделирования равномерно распределенной случайной величины использовать встроенный генератор случайных чисел `np.random.rand()`. Число экспериментов для моделирования принять равными $N = 10^3, 10^4\, 10^5\, 10^6$.

Построить гистограммы относительных частот для полученных последовательностей случайных чисел (см. задание 3) на 100 интервалах и рассчитать для них значения среднеквадратического отклонения от функции плотности вероятности экспоненциального закона распределения.

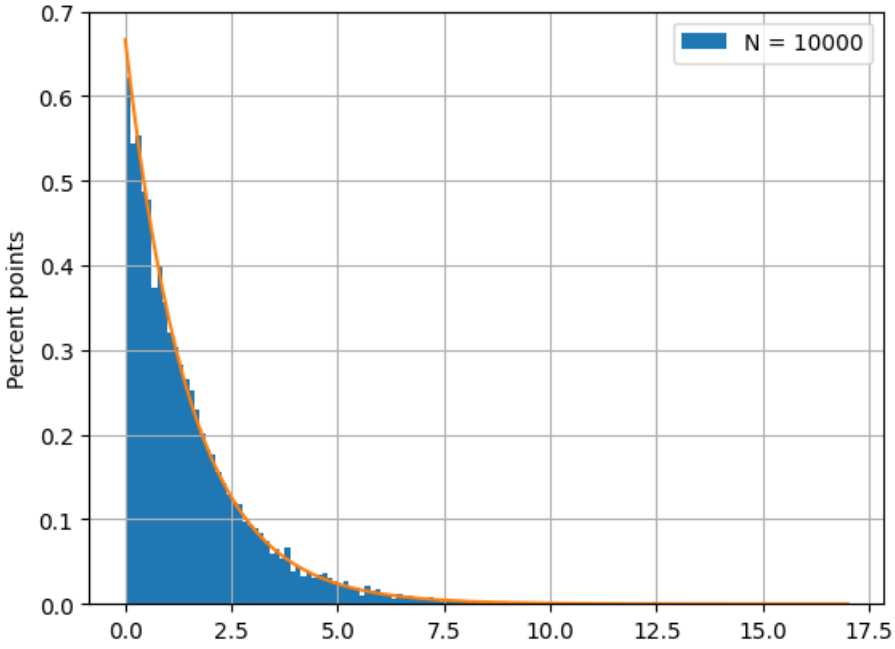
```
def task_3_4():
    scale = 1.5
    experiments = [10 ** 3, 10 ** 4, 10 ** 5, 10 ** 6]
    parts = 100

    x = np.linspace(0, 17, 100)

    for experiment in experiments:
        y = expon.ppf(
```

```
            [random.random() for _ in range(experiment)],
            scale=scale)
        print("First 10 elements of y:")
        display(y[:10])

        plt.hist(y, bins = parts, density = True, label = f'N = {experiment}')
        plt.legend()
        plt.ylabel('Percent points')
        plt.grid(True)
        plt.plot(x, expon.pdf(x, scale=scale), label=f'$\lambda \exp(-\lambda x)$, $\lambda$
        plt.show()

task_3_4()
```

```
First 10 elements of y:
array([1.19151229, 0.62089132, 2.96646161, 0.62114824, 1.49594465,
       1.61476598, 2.69697715, 1.10127575, 0.79341527, 0.84191712])
```

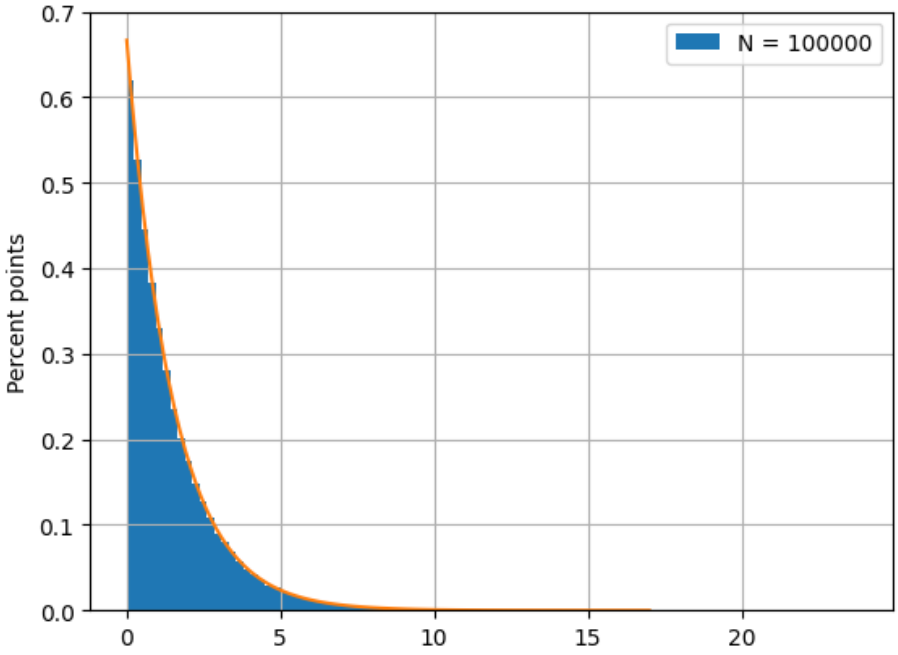



```
First 10 elements of y:
array([0.10308248, 0.1177111 , 0.66239041, 3.29273003, 2.0367196 ,
       0.39031448, 1.7050926 , 1.57372237, 4.7967069 , 2.46183069])
```

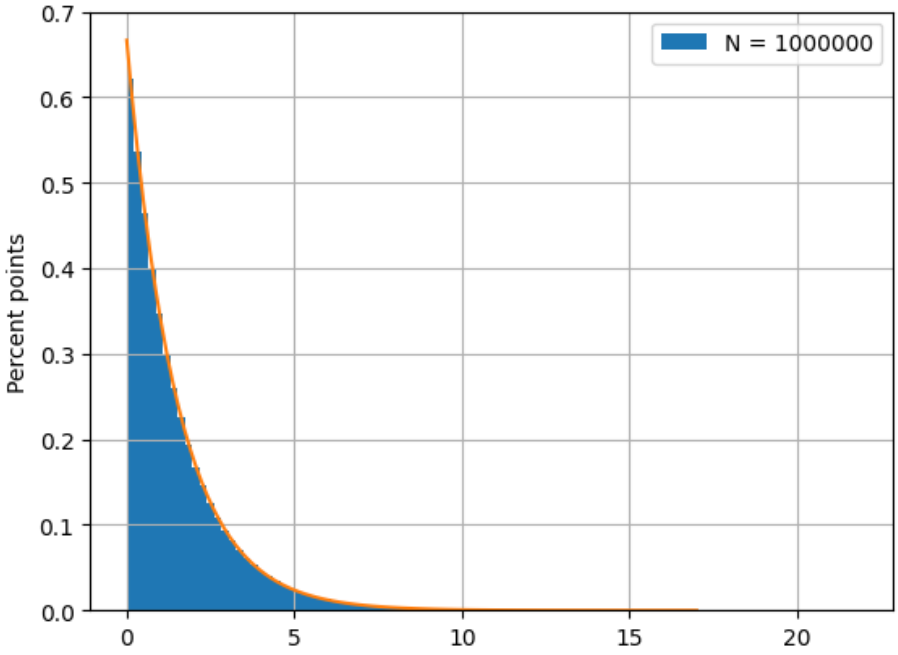


First 10 elements of y:

```
array([9.87944709e-02, 4.29043555e-01, 1.95060662e+00, 1.73311092e-01,
       1.65150441e+00, 5.38397527e-01, 2.61548273e-03, 4.04340477e+00,
       1.94115113e+00, 5.91114259e+00])
```




```
First 10 elements of y:

array([1.85615047, 5.32382698, 0.4857374 , 0.25849825, 0.43388257,
       2.31101558, 0.32442607, 6.16003118, 0.08853592, 0.96663018])
```


N = 1000000
Percent points
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20