**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)»**
**(МГТУ им. Н. Э. Баумана)**

ФАКУЛЬТЕТ ИУ «Информатика и системы управления»

КАФЕДРА ИУ-7 «Программное обеспечение эвм и информационные технологии»

# ОТЧЕТ

## По лабораторной работе №1

## «Исследование функций и плотностей распределения случайных величин»

## По курсу «Моделирование»

| | |
|---|---|
| Студент | Т. А. Казаева |
| Группа | ИУ7-76Б |
| Преподаватель | И. В. Рудаков |

*2022 г.*

## 1. Задание

Исследовать функцию и плотность распределения:

- равномерного;
- нормального.

Разработать программу для построения графиков этих распределений. Реализовать графический интерфейс, позволяющий задавать параметры для построения графиков.

# 2. Математическая формализация

## 2.1 Непрерывное равномерное распределение

Говорят, что случайная величина $X$ имеет непрерывное равномерное распределение на отрезке $[a, b]$, если её функция плотности имеет вид (2.1):

$$f_X(x) = \begin{cases} \dfrac{1}{b-a}, & x \in [a, b] \\ 0, & x \notin [a, b] \end{cases}. \tag{2.1}$$

Обозначается: $X \sim U[a, b]$.
Функция распределения равномерной случайной величины $X \sim U[a, b]$ (2.2):

$$F_X(x) \equiv \mathbb{P}(X \leqslant x) = \begin{cases} 0, & x < a \\ \dfrac{x-a}{b-a}, & a \leqslant x < b \\ 1, & x \geqslant b \end{cases} \cdot \tag{2.2}$$

## 2.2 Нормальное распределение

Говорят, что случайная величина $X$ имеет нормальное распределение с параметрами $\mu$ и $\sigma^2$ ($\sigma^2 > 0$), если её функция плотности имеет вид (2.3):

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\dfrac{(x-\mu)^2}{2\sigma^2}}, x \in \mathbb{R} \tag{2.3}$$

Обозначается: $X \sim N(\mu, \sigma^2)$.

Функция распределения нормальной случайной величины $X \sim N(\mu, \sigma^2)$:

$$\frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right], \tag{2.4}$$

где

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt. \tag{2.5}$$

# 3. Результат

На рисунке 3.1 представлен графический интерфейс приложения.

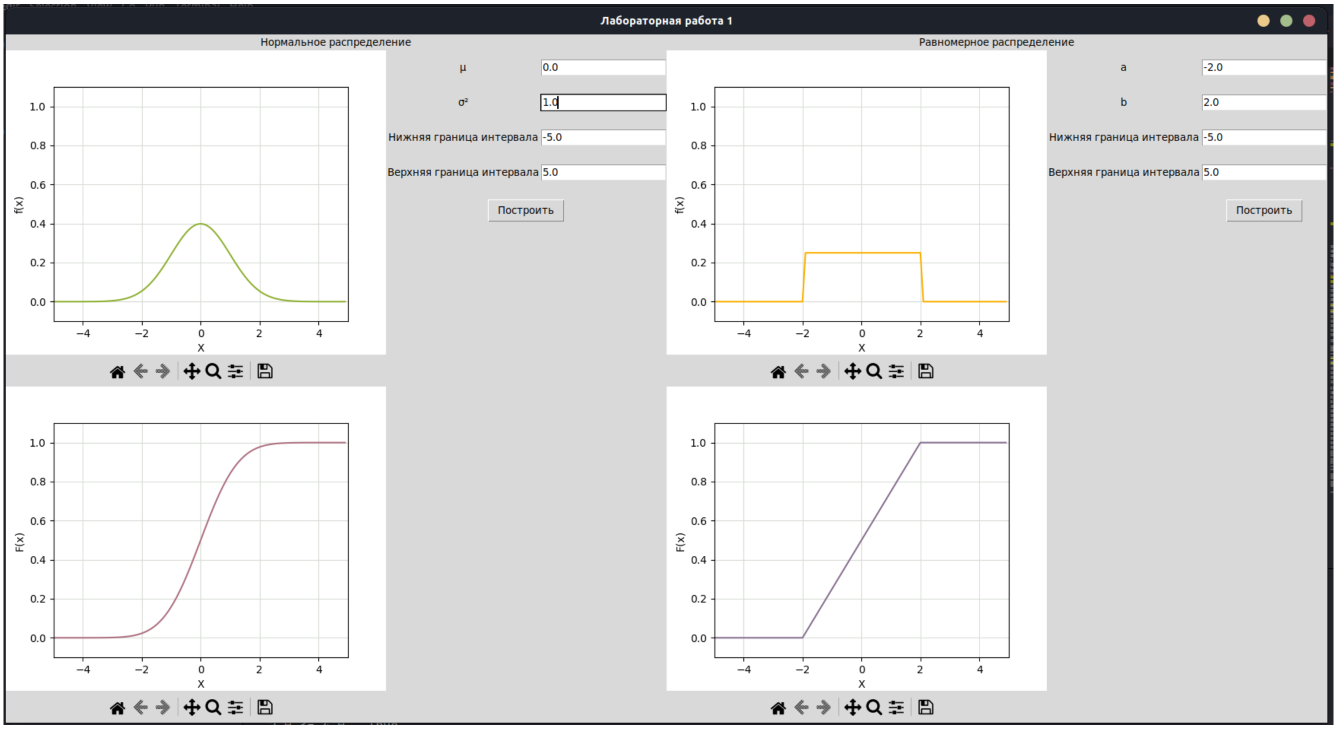



Рис. 3.1: Графический интерфейс приложения

Если пользователь ввёл некорректные данные, он увидит предупреждение с ошибкой и названием поля с некорректными данными (рисунок 3.2).

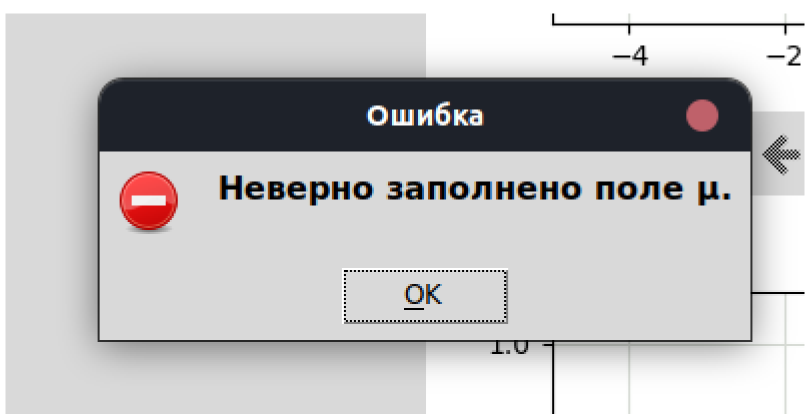
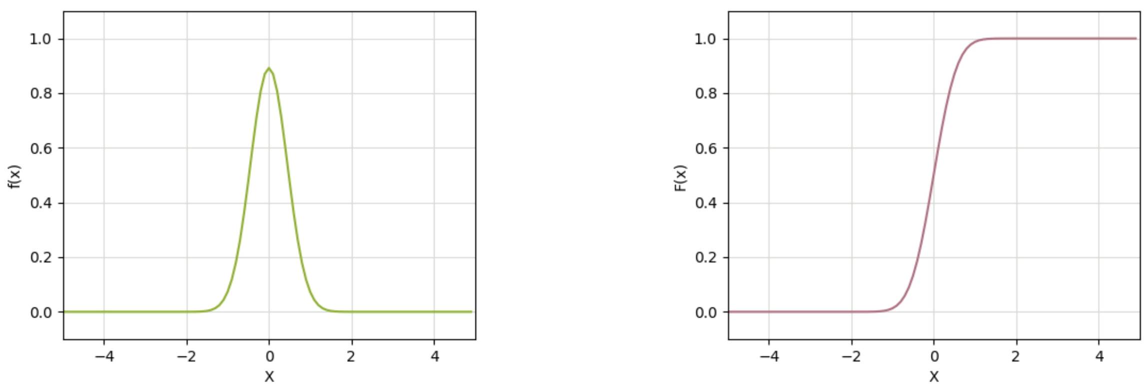



Рис. 3.2: Графический интерфейс приложения

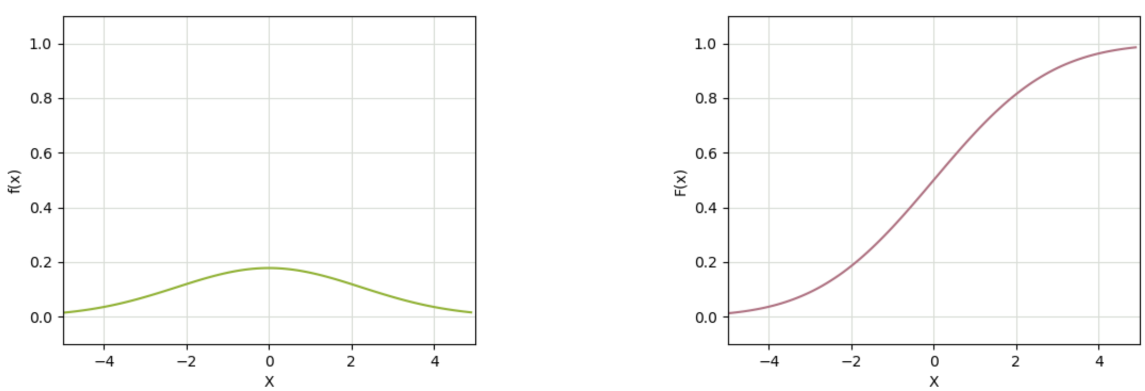
На рисунках 3.3 – 3.5 представлены полученные графики нормального распределения с различными вариациями коэффициентов.

1. Рисунок 3.3 – $\mu = 0.0,\ \sigma^2 = 0.2$
2. Рисунок 3.4 – $\mu = 0.0,\ \sigma^2 = 5.0$
3. Рисунок 3.5 – $\mu = -2.0,\ \sigma^2 = 0.5$



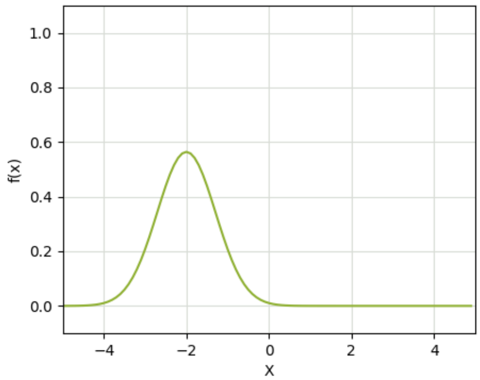

(a) Плотность вероятности (b) Функция распределения

Рис. 3.3: Нормальное распределение с параметрами $\mu = 0.0,\ \sigma^2 = 0.2$

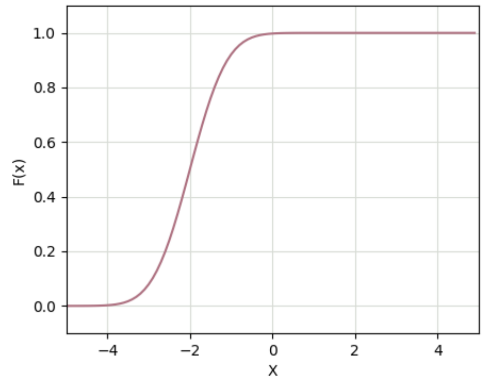



(a) Плотность вероятности (b) Функция распределения

Рис. 3.4: Нормальное распределение с параметрами $\mu = 0.0,\ \sigma^2 = 5.0$




(a) Плотность вероятности

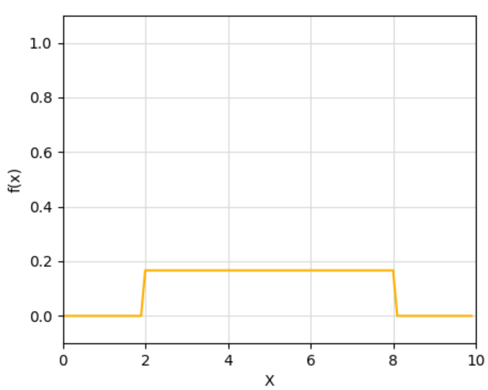



(b) Функция распределения

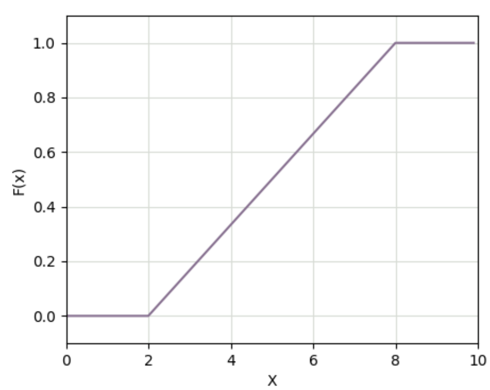
Рис. 3.5: Нормальное распределение с параметрами $\mu = -2.0,\ \sigma^2 = 0.5$

На рисунках 3.6 – 3.8 представлены полученные графики равномерного распределения с различными вариациями коэффициентов.

1. Рисунок 3.6 – $a = 2.0,\ b = 8.0$
2. Рисунок 3.7 – $a = 5.75,\ b = 7.0$
3. Рисунок 3.8 – $a = -5.0,\ b = -3.0$

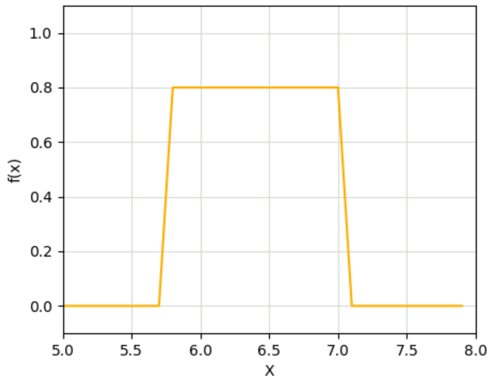



(a) Плотность вероятности

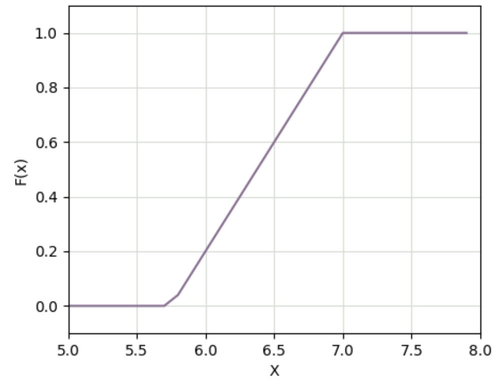



(b) Функция распределения

Рис. 3.6: Равномерное распределение с параметрами $a = 2.0,\ b = 8.0$

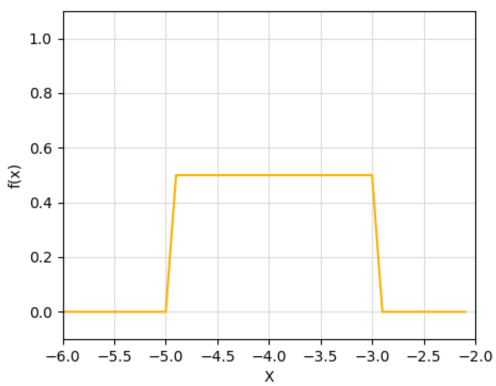



(a) Плотность вероятности

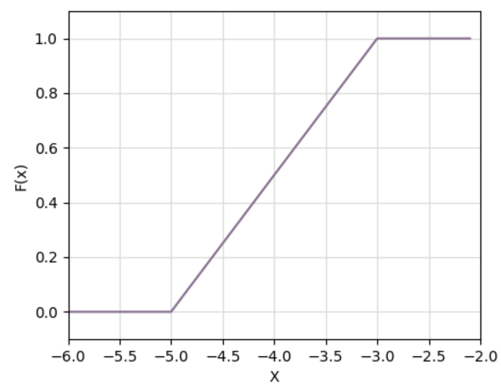



(b) Функция распределения

Рис. 3.7: Равномерное распределение с параметрами $a = 5.75,\ b = 7.0$




(a) Плотность вероятности




(b) Функция распределения

Рис. 3.8: Равномерное распределение с параметрами $a = -5.0,\ b = -3.0$

# 4. Программный код

Для реализации программы был выбран язык Python.

На листинге 4.1 представлен программный код абстрактного класса Distribution.

Листинг 4.1: Абстрактный класс Distribution

```
from abc import ABCMeta, abstractmethod

class AbstractDistribution(object):
    __metaclass__ = ABCMeta

    @abstractmethod
    def CDF(self, x):
        pass

    @abstractmethod
    def PDF(self, x):
        pass
```

На листинге 4.2 представлен программный код класса Gaussian (нормальное распределение), реализующий абстрактный класс Distribution.

Листинг 4.2: Класс Gaussian

```
from probability.distribution.distribution import
   AbstractDistribution

import math
from scipy import special


class Gaussian(AbstractDistribution):
    __slots__ = ["mean",
                 "deviation",
                 "variance"]

    def __init__(self, mean, variance):
```

```
        if variance <= 0.0:
            raise ValueError("Variance can't non-positive")

        self.mean = mean
        self.variance = variance
        self.deviation = math.sqrt(variance)

    def CDF(self, x) -> float:
        return 0.5 * special.erfc(-(x-self.mean)/(self.deviation*math.sqrt(2)))

    def PDF(self, x) -> float:
        m = self.deviation * math.sqrt(2*math.pi)
        e = math.exp(-math.pow(x-self.mean, 2) / (2 * self.variance))
        return e / m
```

На листинге 4.3 представлен программный код класса Uniform (равномерное распределение), реализующий абстрактный класс Distribution.

Листинг 4.3: Класс Uniform

```
from probability.distribution.distribution import AbstractDistribution

class Uniform(AbstractDistribution):
    __slots__ = ["min",
                 "max"]

    def __init__(self, min, max):
        self.min = min
        self.max = max


    def CDF(self, x) -> float:
        if x < self.min:
            return 0
        if x > self.max:
            return 1
```

```
    return (x - self.min) / (self.max - self.min)

    def PDF(self, x) -> float:
        if x < self.min:
            return 0
        if x > self.max:
            return 0

        return 1 / (self.max - self.min)
```

При запуске приложения на координатных плоскостях изображены графики со следующими параметрами: нормальное распределение – коэффициенты стандартного нормального распределения ($\mu = 0.0,\ \sigma^2 = 1.0$), равномерное распределение – $a = -2.0,\ b = 2.0$. На листинге 4.4 представлена часть программного кода построения графиков с коэффициентами по умолчанию.

Листинг 4.4: Часть программного кода построения графиков

```

figCDFGaussian = Figure(figsize=(5, 4), dpi=100)
figPDFGaussian = Figure(figsize=(5, 4), dpi=100)

figCDFUniform = Figure(figsize=(5, 4), dpi=100)
figPDFUniform = Figure(figsize=(5, 4), dpi=100)

xs = np.arange(-5, 5, .1)

u = Uniform(-2, 2)
ysPDFUniform = []
ysCDFUniform = []

g = Gaussian(0, 1)
ysPDFGaussian = []
ysCDFGaussian = []

for x in xs:
    ysPDFGaussian.append(g.PDF(x))
    ysCDFGaussian.append(g.CDF(x))
```

```

    ysPDFUniform.append(u.PDF(x))
    ysCDFUniform.append(u.CDF(x))

axCDFGaussian = figCDFGaussian.add_subplot()
lineCDFGaussian, = axCDFGaussian.plot(xs, ysCDFGaussian,
                                          color='xkcd:mauve')

axPDFGaussian = figPDFGaussian.add_subplot()
linePDFGaussian, = axPDFGaussian.plot(xs, ysPDFGaussian,
                                       color='xkcd:avocado')

canvasPDFGaussian = FigureCanvasTkAgg(figPDFGaussian,
                                       master=root)
canvasPDFGaussian.draw()

canvasCDFGaussian = FigureCanvasTkAgg(figCDFGaussian,
                                       master=root)
canvasCDFGaussian.draw()
```