

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
**«Рязанский государственный радиотехнический университет  
Имени В. Ф. Уткина»**

Факультет вычислительной техники  
Кафедра вычислительной и прикладной математики

Отчёт по практической работе №3

по дисциплине:  
«Моделирование»

по теме:  
«Генерирование случайных величин с заданным законом распределения»

Выполнил: студент гр. 242

Фокин А.М.

Проверил: Анастасьев А. А.

Рязань 2025

### Цель работы:

Составить подпрограмму генерирования случайных величин в соответствии с вариантом задания, определяемым таблицей 3. По полученной с помощью подпрограммы выборке построить и проанализировать гистограмму частот и статистическую функцию распределения, оценить матожидание и дисперсию случайной величины. Соответствие эмпирических данных теоретическому распределению проверить с помощью критерия Пирсона или критерия Колмогорова. Объем выборки случайных величин не менее 1000. Количество интервалов разбиения  $k = 15$  или  $k = 25$ . Теоретическая часть для данного практического занятия представлена в учебнике [1] на стр. 65–76

№ вар.	Закон распределения	Способ построения
19.	$f(x) = \sqrt{R^2 - (x-a)^2}$ ; $R = \sqrt{\pi/2}$ ; $a = -2$ .	Метод отбора

### Программа реализует следующие операции:

– Генерация выборки методом отбора:

```
data = sample_rejection(n)
```

– Расчёт выборочного среднего и дисперсии:

```
sample_mean = np.mean(data)  
sample_var = np.var(data, ddof=1)
```

– Построение гистограммы и теоретической плотности:

```
plt.bar(bin_centers, rel_freq, ...)  
plt.plot(xs, ys, ...)
```

– Проверка соответствия распределения (Колмогоров):

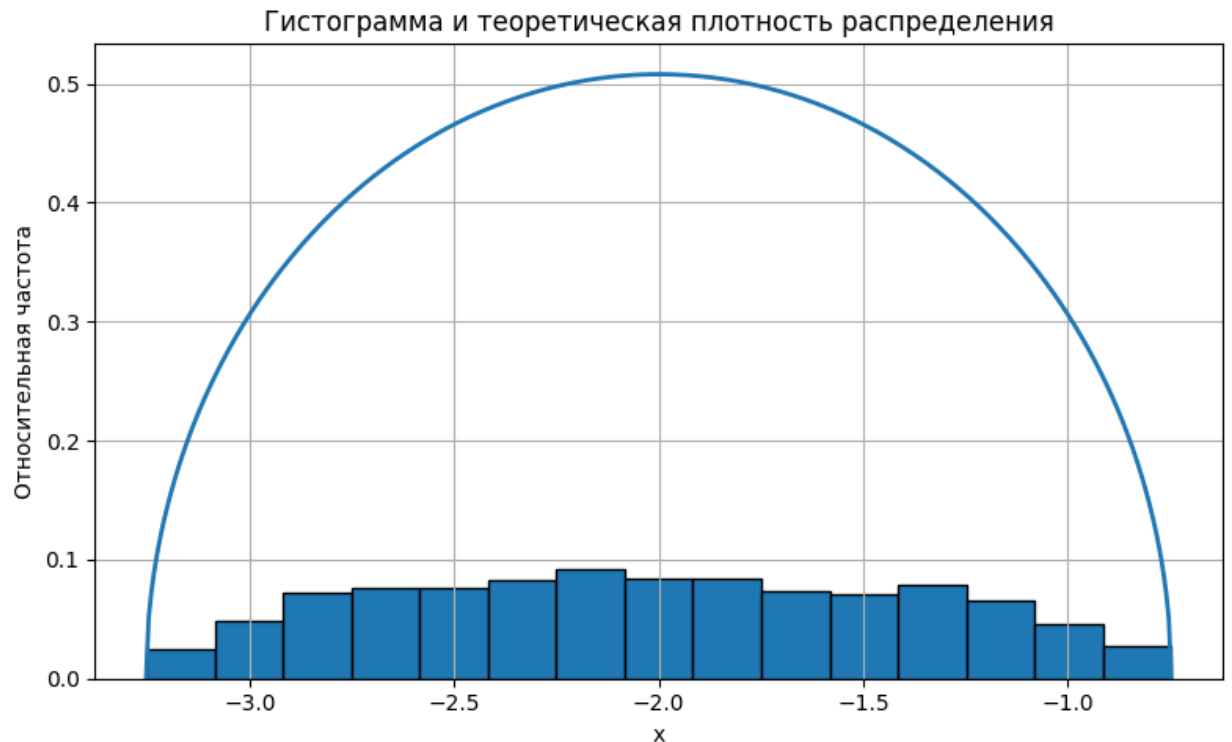
```
D = max(D_plus, D_minus)
```

– Проверка критерием  $\chi^2$ :

```
chi2 += (obs - exp) ** 2 / exp
```

*полный код программы приведен в приложении 1*

## Результат работы программы:



```
===== RESTART: D:\мусорка\учеба\моделирования\lab3-Fokin-242.py =====
Объем выборки: 2000
Выборочное среднее: -2.0107770646791443
Выборочная дисперсия: 0.3871362699770522
Критерий Колмогорова-Смирнова D: 0.01951319744155433
Критическое значение D ( $\alpha=0.05$ ): 0.03041052449399714
Гипотеза  $H_0$  не отвергается при уровне значимости 0.05
Статистика  $\chi^2$ : 11.663080217974553
Количество интервалов: 15
Ожидаемые частоты по интервалам: [ 57.261 101.266 126.23 143.397 155.429 163.47
7 168.12 169.639 168.12
163.477 155.429 143.397 126.23 101.266 57.261]
```

Объем выборки: 2000

Выборочное среднее: -2.0107770646791443

Выборочная дисперсия: 0.3871362699770522

Критерий Колмогорова-Смирнова D: 0.01951319744155433

Критическое значение D ( $\alpha=0.05$ ): 0.03041052449399714

Гипотеза  $H_0$  не отвергается при уровне значимости 0.05

Статистика  $\chi^2$ : 11.663080217974553

Количество интервалов: 15

Ожидаемые частоты по интервалам: [ 57.261 101.266 126.23

143.397 155.429 163.477 168.12 169.639 168.12

163.477 155.429 143.397 126.23 101.266 57.261]

### Ответ на вопрос

5. В каких случаях возможно применение метода обратных функций?

Метод обратных функций применяют, когда есть аналитическое выражение для функции распределения  $F(x)$  и её обратную функцию  $F^{-1}(x)$  можно найти явно.

То есть, если можно записать  $x = F^{-1}(u)$ , где  $u$  — случайное число из равномерного распределения  $(0,1)$ , тогда метод подходит.

Он используется для простых распределений — например, равномерного, экспоненциального, распределения Лапласа, где обратная функция легко вычисляется.

## Приложение 1 - код программы

```
import numpy as np
import math
import matplotlib.pyplot as plt

R = math.sqrt(math.pi / 2)
a = -2.0

def pdf(x):
    z = x - a
    inside = R * R - z * z
    if inside <= 0:
        return 0.0
    value = math.sqrt(inside)
    norm = 4.0 / (math.pi ** 2)
    return norm * value

def cdf(x):
    z = (x - a) / R
    if z <= -1:
        return 0.0
    if z >= 1:
        return 1.0
    part = z * math.sqrt(1.0 - z * z) + math.asin(z)
    return part / math.pi + 0.5

def sample_rejection(size):
    samples = []
    x_min = a - R
    x_max = a + R
    f_max = (4.0 / (math.pi ** 2)) * R
    while len(samples) < size:
        x = np.random.uniform(x_min, x_max)
        y = np.random.uniform(0.0, f_max)
        if y <= pdf(x):
            samples.append(x)
    return np.array(samples)

n = 2000
k = 15

data = sample_rejection(n)
```

```

sample_mean = float(np.mean(data))
sample_var = float(np.var(data, ddof=1))

counts, bin_edges = np.histogram(data, bins=k, range=(a - R, a +
R))
bin_centers = 0.5 * (bin_edges[:-1] + bin_edges[1:])
counts = counts.astype(float)
rel_freq = counts / n

theo_probs = []
for i in range(len(bin_edges) - 1):
    p = cdf(bin_edges[i + 1]) - cdf(bin_edges[i])
    theo_probs.append(p)
theo_probs = np.array(theo_probs)
expected = theo_probs * n

plt.figure(figsize=(8, 5))
plt.bar(bin_centers, rel_freq, width=(bin_edges[1] -
bin_edges[0]), align='center', edgecolor='black')
xs = np.linspace(a - R, a + R, 400)
ys = [pdf(x) for x in xs]
plt.plot(xs, ys, linewidth=2)
plt.xlabel('x')
plt.ylabel('Относительная частота')
plt.title('Гистограмма и теоретическая плотность распределения')
plt.grid(True)
plt.tight_layout()
plt.show()

sorted_data = np.sort(data)
emp_x = sorted_data
emp_y = np.arange(1, n + 1) / n
theo_y = np.array([cdf(x) for x in emp_x])
D_plus = np.max(emp_y - theo_y)
D_minus = np.max(theo_y - (np.arange(0, n) / n))
D = max(D_plus, D_minus)
D_crit = 1.36 / math.sqrt(n)

chi2 = 0.0
for obs, exp in zip(counts, expected):
    if exp > 0:
        chi2 += (obs - exp) ** 2 / exp

```

```
print('Объем выборки:', n)
print('Выборочное среднее:', sample_mean)
print('Выборочная дисперсия:', sample_var)
print('Критерий Колмогорова-Смирнова D:', D)
print('Критическое значение D ( $\alpha=0.05$ ):', D_crit)
if D > D_crit:
    print('Гипотеза  $H_0$  отвергается при уровне значимости 0.05')
else:
    print('Гипотеза  $H_0$  не отвергается при уровне значимости 0.05')
print('Статистика  $\chi^2$ :', chi2)
print('Количество интервалов:', k)
print('Ожидаемые частоты по интервалам:', np.round(expected, 3))
```