

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 №5

по дисциплине:
“Моделирование”

по теме:
“Генерирование случайных величин с часто используемыми законами
распределения”

Выполнил: студент гр. 242

Фокин А.М.

Проверил: Анастасьев А. А.

Цель работы:

Составить подпрограммы генерирования случайных величин, подчиненных распределению, указанному в варианте задания (таб. 5). 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 $k = 25$.

19	Распределение Вейбулла	Гамма-распределение
----	------------------------	---------------------

Программа реализует следующие операции:

– Генерация выборки для распределения Вейбулла по формуле обратного преобразования:

```
return lam_weibull * (-np.log(1 - u)) ** (1 / k_weibull)
```

– Генерация выборки для гамма-распределения через произведение равномерных случайных чисел:

```
prod *= np.random.rand()
x = -math.log(prod) * beta_gamma
```

– Построение гистограммы и наложение теоретической плотности:

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

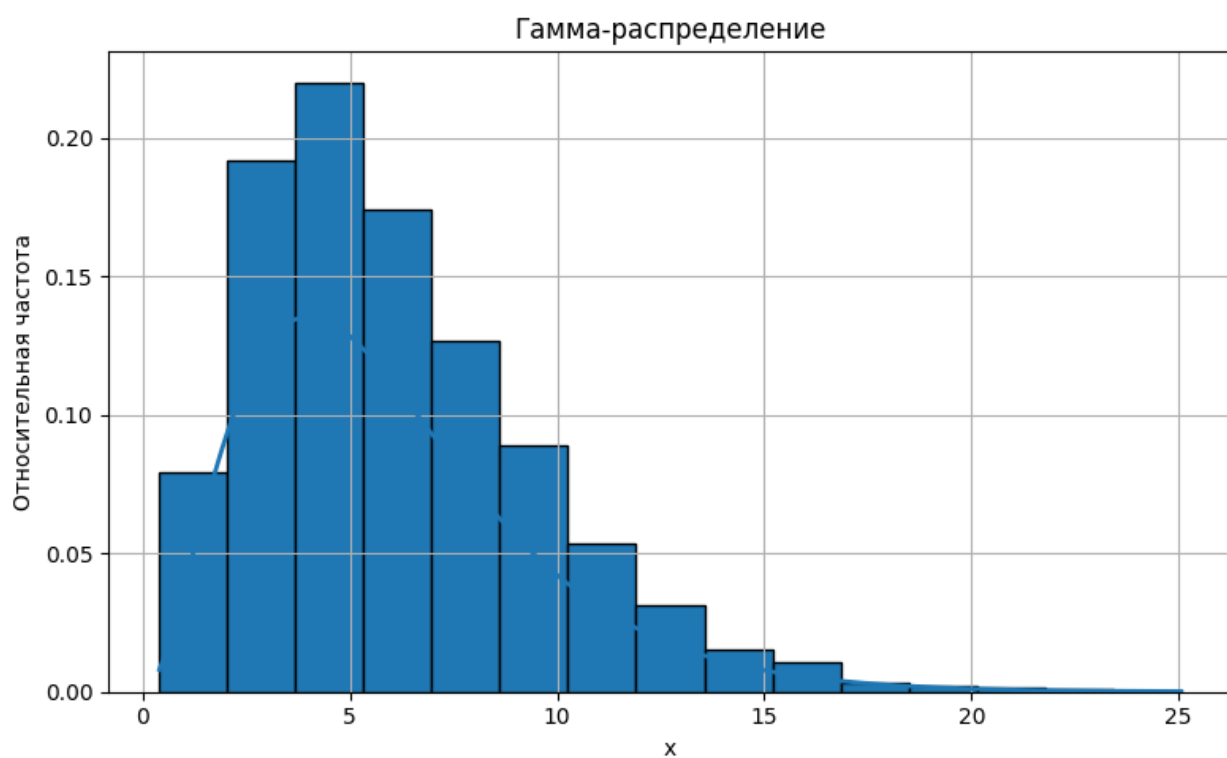
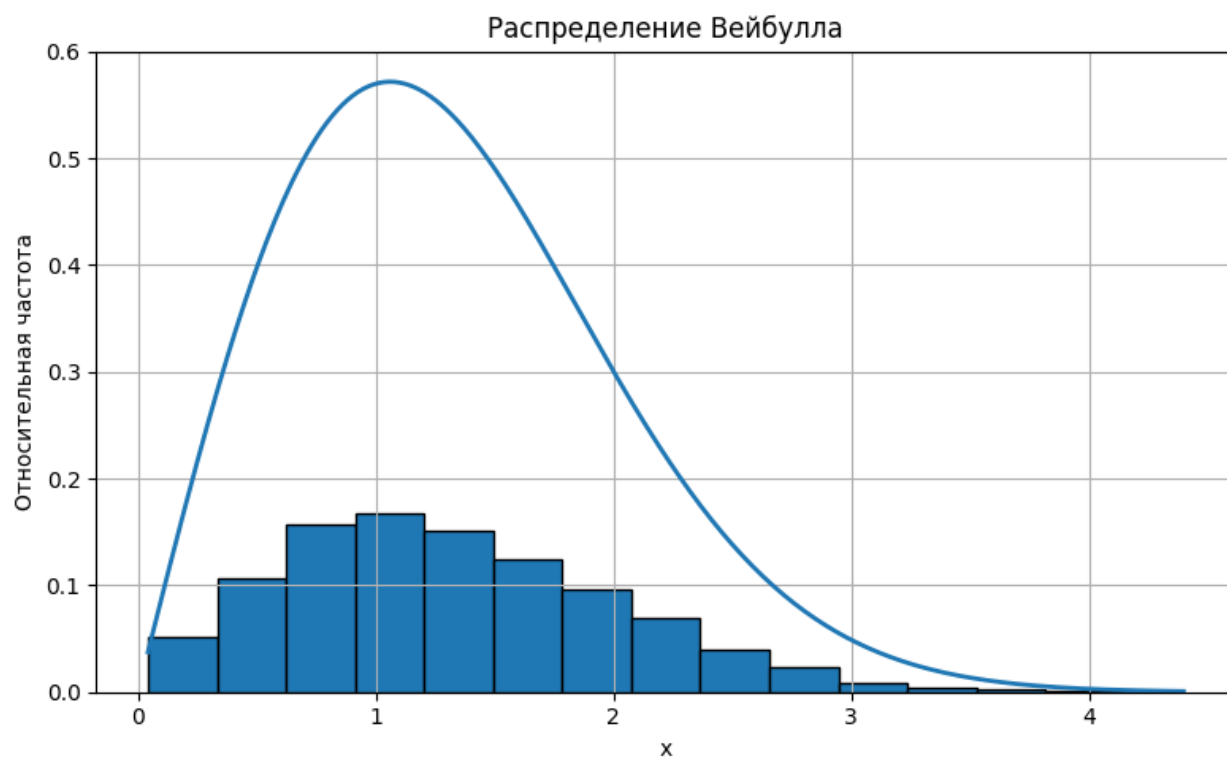
– Проверка соответствия распределений критерием Колмогорова и χ^2 :

```
D = max(D_plus, D_minus)
chi2 += (obs - exp) ** 2 / exp
```

Итог: реализованы подпрограммы генерации случайных величин для Вейбулла и Гамма-распределения, построена гистограмма, вычислены статистики и проверено соответствие теоретическому закону.

полный код программы приведен в приложении 1

Результат работы программы:



===== RESTART: D:\мусорка\учеба\моделирования\lab5-Fokin-242.py =====

Распределение Вейбулла

Объем выборки: 2000

Выборочное среднее: 1.3195368677817574

Выборочная дисперсия: 0.4699913772245796

Критерий Колмогорова-Смирнова D: 0.012977718012416206

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

Гипотеза H_0 не отвергается при уровне значимости 0.05

Статистика χ^2 : 6.7227912963352106

Количество интервалов: 15

Ожидаемые частоты по интервалам: [94.296 220.82 302.637 329.978 308.836 255.87
1 190.602 128.809 79.42
44.846 23.253 11.093 4.876 1.977 0.74]

Гамма-распределение

Объем выборки: 2000

Выборочное среднее: 6.081203818403981

Выборочная дисперсия: 11.994489222739864

Критерий Колмогорова-Смирнова D: 0.0179642814220643

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

Гипотеза H_0 не отвергается при уровне значимости 0.05

Статистика χ^2 : 5.045599331243933

Количество интервалов: 15

Ожидаемые частоты по интервалам: [165.694 393.3 433.818 358.14 254.203 164.45
9 99.937 58.046 32.584
17.809 9.527 5.008 2.594 1.327 0.672]

Распределение Вейбулла

Объем выборки: 2000

Выборочное среднее: 1.3195368677817574

Выборочная дисперсия: 0.4699913772245796

Критерий Колмогорова-Смирнова D: 0.012977718012416206

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

Гипотеза H_0 не отвергается при уровне значимости 0.05

Статистика χ^2 : 6.7227912963352106

Количество интервалов: 15

Ожидаемые частоты по интервалам: [94.296 220.82 302.637
329.978 308.836 255.871 190.602 128.809 79.42
44.846 23.253 11.093 4.876 1.977 0.74]

Гамма-распределение

Объем выборки: 2000

Выборочное среднее: 6.081203818403981

Выборочная дисперсия: 11.994489222739864

Критерий Колмогорова-Смирнова D: 0.0179642814220643

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

Гипотеза H_0 не отвергается при уровне значимости 0.05

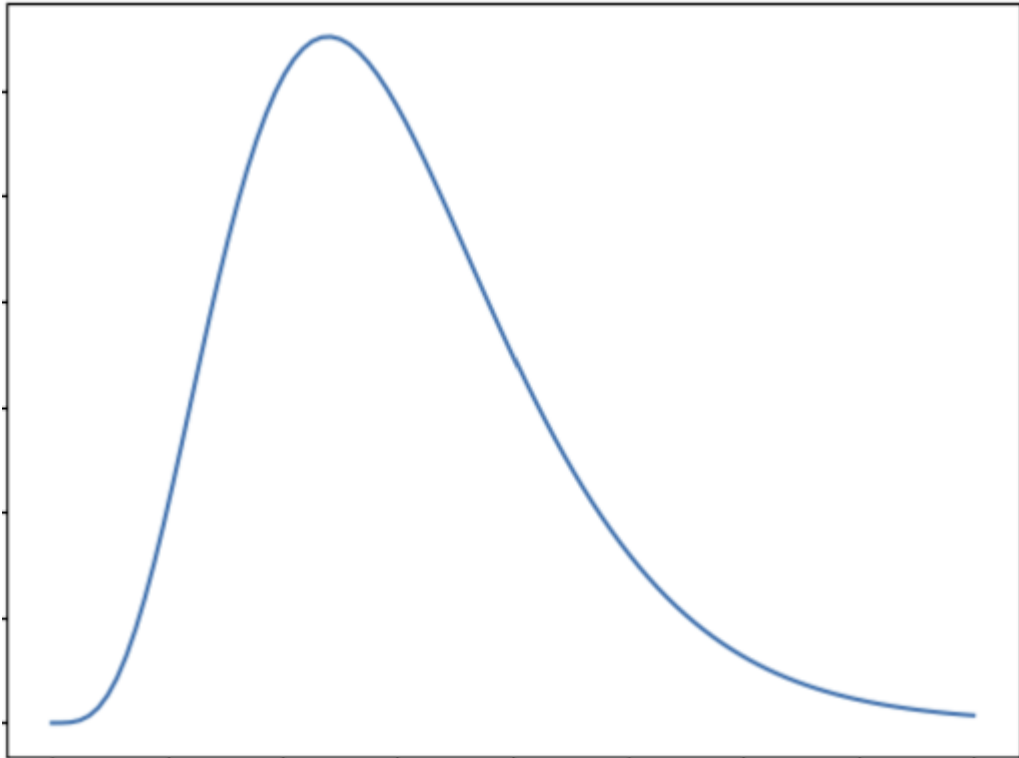
Статистика χ^2 : 5.045599331243933

Количество интервалов: 15

Ожидаемые частоты по интервалам: [165.694 393.3 433.818 358.14
254.203 164.459 99.937 58.046 32.584
17.809 9.527 5.008 2.594 1.327 0.672]

Ответ на вопрос

2. Как выглядит функция плотности гамма-распределения?



Общая форма гамма-распределения

$$f(x, \alpha, \beta) = \frac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}$$

Формула гамма-распределения

Где

α — параметр формы. Определяет «вид» распределения: чем больше α , тем более симметричным становится график.

β — параметр масштаба. Влияет на растяжение или сжатие распределения по оси X.

$\Gamma(\alpha)$ — гамма-функция, используется для нормализации, чтобы площадь под кривой была равна 1.

Источник:

<https://builtin.com/data-science/gamma-distribution>

Приложение 1 - код программы

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

n = 2000
k = 15

lam_weibull = 1.5
k_weibull = 2.0

alpha_gamma = 3.0
beta_gamma = 2.0

def weibull_generate(size):
    u = np.random.rand(size)
    return lam_weibull * (-np.log(1 - u)) ** (1 / k_weibull)

def weibull_pdf(x):
    if x < 0:
        return 0.0
    return (k_weibull / lam_weibull) * (x / lam_weibull) **
(k_weibull - 1) * math.exp(- (x / lam_weibull) ** k_weibull)

def weibull_cdf(x):
    if x < 0:
        return 0.0
    return 1 - math.exp(- (x / lam_weibull) ** k_weibull)

# --- Гамма ---
def gamma_generate(size):
    data = []
    for _ in range(size):
        prod = 1.0
        for _ in range(int(alpha_gamma)):
            prod *= np.random.rand()
        x = -math.log(prod) * beta_gamma
        data.append(x)
    return np.array(data)

def gamma_pdf(x):
    if x < 0:
        return 0.0
```

```

    return (x ** (alpha_gamma - 1) * math.exp(-x / beta_gamma))
    / (math.gamma(alpha_gamma) * (beta_gamma ** alpha_gamma))

def gamma_cdf(x):
    if x < 0:
        return 0.0
    total = 0
    for i in range(int(alpha_gamma)):
        total += ((x / beta_gamma) ** i) / math.factorial(i)
    return 1 - math.exp(-x / beta_gamma) * total

def analyze(data, pdf_func, cdf_func, title):
    sample_mean = float(np.mean(data))
    sample_var = float(np.var(data, ddof=1))

    counts, bin_edges = np.histogram(data, bins=k)
    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_func(bin_edges[i + 1]) - cdf_func(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(min(data), max(data), 400)
    ys = [pdf_func(x) for x in xs]
    plt.plot(xs, ys, linewidth=2)
    plt.xlabel('x')
    plt.ylabel('Относительная частота')
    plt.title(title)
    plt.grid(True)
    plt.tight_layout()
    plt.show()

    sorted_data = np.sort(data)
    emp_y = np.arange(1, n + 1) / n
    theo_y = np.array([cdf_func(x) for x in sorted_data])

```

```

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(title)
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))
print()

data_weibull = weibull_generate(n)
analyze(data_weibull, weibull_pdf, weibull_cdf, 'Распределение Вейбулла')

data_gamma = gamma_generate(n)
analyze(data_gamma, gamma_pdf, gamma_cdf, 'Гамма-распределение')

```