Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

# Расчетная работа №2

по дисциплине «Теория вероятностей и математическая статистика»

**Работу выполнил:**
Ильин В.П.
Группа:
35300901/10005
**Преподаватель:**
Куляшова З.В.

Санкт-Петербург
2023

# 1. Оценка полных характеристик распределения

Рассмотрим вывод `part1.py`:

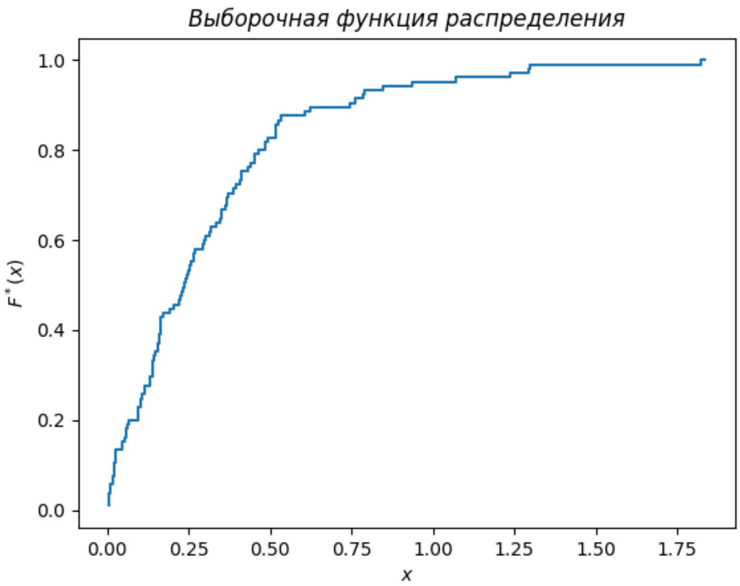



Рис. 1.1: Выборочная функция распределения

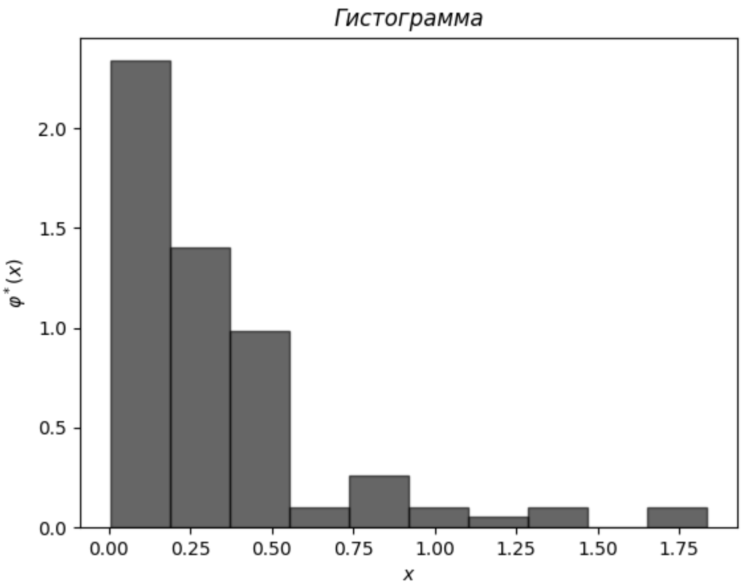



Рис. 1.2: Гистограмма $\varphi^*(x)$

# 2. Частные характеристики распределения

Будем искать характеристики для двух выборок: помимо полной, выделим ее подпоследовательность длиной 20. Найдем точечные оценки (`part21.py`):

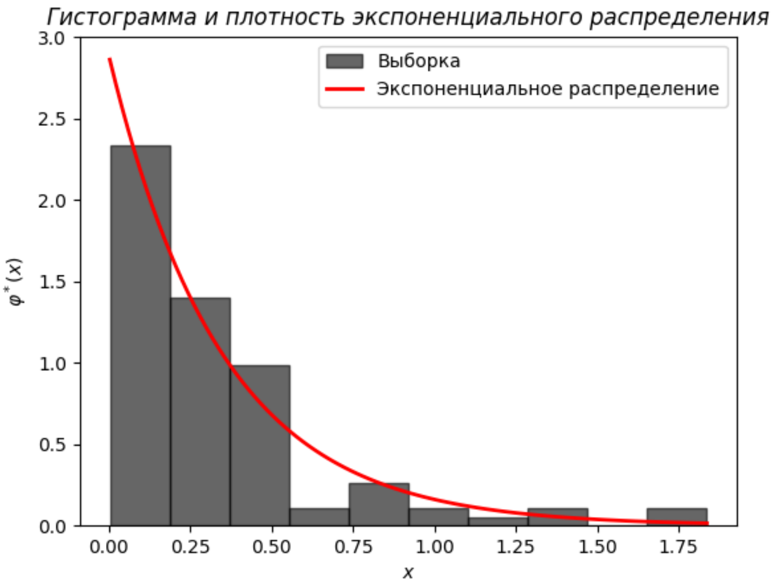
1. Полная выборка
   - Выборочное среднее $= 0.3326$
   - Выборочная медиана $= 0.2392$
   - Середина размаха $= 0.91975$
   - Второй центральный момент (дисперсия) $= 0.1198$
   - Третий центральный момент $= 0.092$
   - Четвертый центральный момент $= 0.1269$
   - Коэффициент асимметрии $= 2.2181$
   - Коэффициент эксцесса $= 5.8456$
   - Границы интерквантильного промежутка (для $P = 0.95$): $[-0.3641, 0.9111]$
2. Частичная выборка
   - Выборочное среднее $= 0.336$
   - Выборочная медиана $= 0.2364$
   - Середина размаха $= 0.6802$
   - Второй центральный момент (дисперсия) $= 0.0841$
   - Третий центральный момент $= 0.0493$
   - Четвертый центральный момент $= 0.0472$
   - Коэффициент асимметрии $= 2.0238$
   - Коэффициент эксцесса $= 3.6748$

Интервальные оценки с доверительной вероятностью $Q = 0.95$:

1. Полная выборка
   - Первый начальный момент $= 0.3326$
   - Интервальные оценки для первого начального момента: $(0.2653, 0.3999)$
   - Второй центральный момент $= 0.1209$
   - Интервальные оценки для второго центрального момента: $(0.0938, 0.1619)$
   - Интерквантильный промежуток (параметрический подход): $(0.257, 0.408)$
   - Интерквантильный промежуток (непараметрический подход): $(0.232, 0.251)$
2. Частичная выборка
   - Первый начальный момент $= 0.4596$
   - Интервальные оценки для первого начального момента: $(0.2144, 0.7047)$
   - Второй центральный момент $= 0.2744$
   - Интервальные оценки для второго центрального момента: $(0.1587, 0.5854)$

# 3. Идентификация закона распределения генеральной совокупности

Исходя из вида гистограммы, можно сделать предположение об экспоненциальном виде закона распределения.




Проверим гипотезу несколькими критериями (`part3.py`):

1. Критерий «Хи-квадрат»: $p = 0.1253 > \alpha = 0.05 \Rightarrow$ подходит;
2. Критерий типа Колмогорова-Смирного: $0.0656 < 0.1309$(крит) $\Rightarrow$ подходит;
3. Критерий «Омега-квадрат» Мизеса: $0.066 < 0.461$(крит) $\Rightarrow$ подходит.

# 4. Вывод

В ходе работы были оценены полные и частные характеристики распределения данной случайной величины. Было идентифицировано, что она подчиняется экспоненциальному закону распределения.

# Приложение 1

```python
import numpy as np
from part1 import part1
from part21 import part21
from part23 import part23
from part3 import part3

nums = []
with open('data.txt') as f:
    for line in f.readlines():
        nums.append(float(line))
data = np.array(nums)
rnd = int(100 * np.random.rand())
random_data = np.array(nums[rnd:rnd + 20])

part1(data)
part21(data, random_data)
part23(data, random_data)
part3(data)
```

Рис. 4.1: `main.py`

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

def ecdf(data_array):
    n = len(data_array)
    x_sorted = np.sort(data_array)
    f = np.arange(1, n + 1) / n
    return x_sorted, f

def part1(data):
    x, y = ecdf(data)
    plt.step(x, y)
    plt.xlabel(r'$x$')
    plt.ylabel(r'$F^*(x)$')
    plt.title('Выборочная функция распределения', style='italic')
    plt.savefig('./fig/1.png')
    plt.show()
    plt.hist(data, bins=10, density=True, alpha=0.6, color='black', edgecolor='black')
    plt.xlabel(r'$x$')
    plt.ylabel(r'$\varphi^*(x)$')
    plt.title('Гистограмма', style='italic')
    plt.savefig('./fig/2.png')
    plt.show()
```

Рис. 4.2: `part1.py`

```python
import numpy as np

def find_data_characteristics(data_array):
    print("Выборочное среднее =", np.around(np.mean(data_array), 4))
    print("Выборочная медиана =", np.around(np.median(data_array), 4))
    print("Середина размаха =", np.around((np.max(data_array) + np.min(data_array)) / 2), 4)

    mu2 = np.mean((data_array - np.mean(data_array)) ** 2)  # второй центральный момент (дисперсия)
    mu3 = np.mean((data_array - np.mean(data_array)) ** 3)  # третий центральный момент
    mu4 = np.mean((data_array - np.mean(data_array)) ** 4)  # четвертый центральный момент
    print("Второй центральный момент (дисперсия) =", np.around(mu2, 4))
    print("Третий центральный момент =", np.around(mu3, 4))
    print("Четвертый центральный момент =", np.around(mu4, 4))
    print("Коэффициент асимметрии =", np.around(np.mean(mu3 / mu2 ** (3 / 2)), 4))
    print("Коэффициент эксцесса =", np.around(np.mean(mu4 / mu2 ** 2) - 3, 4))

    IQR = np.percentile(data_array, 75) - np.percentile(data_array, 25)  # интерквартильный размах
    q1 = np.percentile(data_array, 25)  # первый квартиль
    q3 = np.percentile(data_array, 75)  # третий квартиль
    Jp = [q1 - 1.5 * IQR, q3 + 1.5 * IQR]  # интерквантильный промежуток вероятности P = 0.95
    print("Границы интерквантильного промежутка (P = 0.95):", Jp)
    return


def part21(data, random_data):
    print("Основная выборка")
    find_data_characteristics(data)
    print()
    print("Случайная выборка")
    find_data_characteristics(random_data)
    print()
    return
```

Рис. 4.3: `part21.py`

```python
import numpy as np
from scipy import stats

def find_interval_chars_part(data_array):
    Q = 0.95
    n = len(data_array)
    mean = np.mean(data_array)
    var = np.var(data_array, ddof=1)
    std_error = stats.sem(data_array)
    t_value = stats.t.ppf((1 + Q) / 2, n - 1)

    # интервальные оценки для первого начального момента (среднего)
    ci_mean = (mean - t_value * std_error, mean + t_value * std_error)

    # интервальные оценки для второго центрального момента (несмещенной выборочной дисперсии)
    chi2_value = stats.chi2.interval(Q, df=n - 1)
    ci_var = ((n - 1) * var / chi2_value[1], (n - 1) * var / chi2_value[0])

    print("Первый начальный момент: ", mean)
    print("Интервальные оценки для первого начального момента: ", ci_mean)
    print("Второй центральный момент: ", var)
    print("Интервальные оценки для второго центрального момента: ", ci_var)
    return Q, n, mean, std_error


def find_interval_chars(data_array):
    Q, n, mean, std_error = find_interval_chars_part(data_array)
    t_value = 2.2261
    ci_lower = mean - t_value * std_error
    ci_upper = mean + t_value * std_error
    k = 1
    sorted_data = np.sort(data_array)
    trimmed_data = sorted_data[k:n - k]
    alpha = 1 - Q
    tolerance = np.percentile(trimmed_data, [(1 - alpha) / 2 * 100, (1 + alpha) / 2 * 100])
    lower = tolerance[0]
    upper = tolerance[1]
    print(f"Интерквантильный промежуток (параметрический подход): ({ci_lower:.3f}, {ci_upper:.3f})")
    print(f"Интерквантильный промежуток (непараметрический подход): ({lower:.3f}, {upper:.3f})")
    return


def part23(data, random_data):
    print("Основная выборка")
    find_interval_chars(data)
    print()
    print("Случайная выборка")
    find_interval_chars_part(random_data)
    print()
    return
```

Рис. 4.4: part23.py

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats, integrate
from scipy.stats import skew, kurtosis, norm, chisquare, chi2, kstest, expon

def part3(data):
    mu, sigma = np.mean(data), np.std(data)
    x = np.linspace(np.min(data), np.max(data), 100)
    y = expon.pdf(x, scale=sigma)
    plt.hist(data, bins=10, density=True, alpha=0.6, color='black', edgecolor='black', label=r'Выборка')
    plt.plot(x, y, 'r-', lw=2, label=r'Экспоненциальное распределение')
    plt.xlabel(r'$x$')
    plt.ylabel(r'$\varphi^*(x)$')
    plt.title('Гистограмма и плотность экспоненциального распределения', style='italic')
    plt.legend()
    plt.savefig('./fig/3.png')
    plt.show()

    alpha = 0.05
    Xi = 0
    n = len(data)
    hist_new = plt.hist(data, bins=9)
    def get_p(start, end, l):
        return integrate.quad(lambda x: l * np.exp(-l * x), start, end)[0]
    for i in range(0, 9):
        nk = hist_new[0][i]
        start = hist_new[1][i]
        end = hist_new[1][i + 1]
        pk = get_p(start, end, 1 / sigma)
        Xi += (nk - n * pk) ** 2 / (n * pk)
    p_value = 1 - chi2.cdf(Xi, 8)
    print("Критерий Хи-квадрат: ", end='')
    if p_value > alpha:
        print("подходит.")
    else:
        print("не подходит.")

    print("Критерий типа Колмогорова-Смирнова: ", end='')
    mu, sigma = expon.fit(data)  # оценка параметров нормального распределения по выборке
    n = len(data)
    kstest_result = kstest(data, 'expon', args=(mu, sigma)) # статистика Колмогорова-Смирнова и p-значение
    D = kstest_result.statistic
    D_alpha = np.sqrt(-np.log(alpha / 2) / (2 * n)) - 1 / (6 * n) # критическое значение статистики Колмогорова-Смирнова
    if D <= D_alpha:
        print("подходит.")
    else:
        print("не подходит.")
    print("Критерий "омега-квадрат" Мизеса: ", end='')
    mu, std = expon.fit(data) # оцениваем параметры экспоненциального распределения по выборке
    data_sorted = np.sort(data)
    # значение эмпирической функции распределения (ECDF)
    ecdf = np.arange(1, len(data_sorted) + 1) / len(data_sorted)
    # значение теоретической функции распределения для экспоненциального распределения с оцененными параметрами
    cdf = expon.cdf(data_sorted, loc=mu, scale=std)
    omega_squared = np.sum((ecdf - cdf) ** 2) # значение статистики критерия Мизеса
    critical_value = 0.461 # критическое значение статистики Мизеса для уровня значимости 0.05 и n=9
    if omega_squared < critical_value:
        print("подходит.")
    else:
        print("не подходит.")
```

Рис. 4.5: `part3.py`