МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

ИНСТИТУТ КИБЕРНЕТИКИ

КАФЕДРА ВЫСШЕЙ МАТЕМАТИКИ

Лабораторная работа 2

по курсу «**Теория вероятностей и математическая статистика, часть 2**»

Тема: **Первичная обработка выборки из непрерывной генеральной совокупности**

Выполнил:
Студент 3-го курса
Жолковский Д.А.

Группа: КМБО-01-16

МОСКВА 2019

Лабораторная работа по Математической статистике № 2
«Первичная обработка выборки из непрерывной генеральной совокупности»

Задание 1. Получить выборку, сгенерировав 200 псевдослучайных чисел, распределенных по нормальному закону с параметрами

$a = (-1)^V \cdot 0{,}1 \cdot V$ и σ^2, где $\sigma = 0{,}01 \cdot V + 1$

Задание 2. Получить выборку, сгенерировав 200 псевдослучайных чисел, распределенных по показательному закону с параметром λ.

$\lambda = 2 + (-1)^V \cdot 0{,}01 \cdot V$

Задание 3. Получить выборку, сгенерировав 200 псевдослучайных чисел, распределенных равномерно на отрезке [*a, b*].

$a = (-1)^V \cdot 0{,}05 \cdot V$, $b = a + 0{,}05 \cdot V + 1$

V – номер варианта.

Для каждого Задания:

Построить:

1) группированную выборку (интервальный вариационный ряд) и ассоциированный статистический ряд;

2) гистограмму относительных частот;

3) график эмпирической функции распределения.

Найти:

1) выборочное среднее;

2) выборочную дисперсию с поправкой Шеппарда;

3) выборочное среднее квадратическое отклонение;

4) выборочную моду;

5) выборочную медиану;

6) выборочный коэффициент асимметрии;

7) выборочный коэффициент эксцесса.

Составить таблицы:

1) сравнения относительных частот и теоретических вероятностей попадания в интервалы;

2) сравнения рассчитанных характеристик с теоретическими значениями.

V=26 – номер варианта.

Вычисления проводить с точностью до 0,00001.

Краткие теоретические сведения

При построении группированной выборки (интервального вариационного ряда) число интервалов $[a_0, a_1]$, $(a_1, a_2]$,…, $(a_{m-1}, a_m]$ определяется по формуле Стерджеса $m = 1 + [log_2 N]$, $a_0 = x_{(1)}$, $a_m = x_{(N)}$, $d = a_m - a_0$, $a_k - a_{k-1} = d/m$.

Интервальный ряд (группированная выборка) имеет вид:

$[a_{i-1}, a_i]$	$[a_0, a_1]$	…	$(a_{m-1}, a_m]$
$\boldsymbol{n_i}$	$\boldsymbol{n_1}$	…	$\boldsymbol{n_m}$
$\boldsymbol{w_i}$	$\boldsymbol{w_1}$	…	$\boldsymbol{w_m}$

Ассоциированный статистический ряд:

x_i^*	x_1^*	…	x_m^*
$\boldsymbol{n_i}$	$\boldsymbol{n_1}$	…	$\boldsymbol{n_m}$
$\boldsymbol{w_i}$	$\boldsymbol{w_1}$	…	$\boldsymbol{w_m}$

$x_i^* = \frac{a_{i-1} + a_i}{2}$ – середина интервала $(a_{i-1}, a_i]$Полигон

Эмпирическая функция распределения

$$F_N^{Э}(x; x_1, x_2, \dots, x_N) = \sum_{x_i \leqq x} \frac{1}{N}$$

Выборочное среднее

$$\bar{x} = \sum_{i=1}^{m} x_i^* w_i$$

Выборочная дисперсия с поправкой Шеппарда

$$S_B^2 = \sum_{i=1}^{m} (x_i^* - \bar{x})^2 w_i \; - \frac{h^2}{12}, \text{где } h = (a_m - a_0)/m$$

Выборочный момент k-ого порядка

$$\overline{\mu_k} = \sum_{i=1}^{m} (x_i^*)^k w_i$$

Выборочный центральный момент k-ого порядка

$$\overline{\mu_k^o} = \sum_{i=1}^{m} (x_i^* - \bar{x})^k w_i$$

Выборочное среднее квадратическое отклонение

$$\bar{\sigma} = \sqrt{S_B^2}$$

Выборочная медиана

$$\overline{M_e} = \begin{cases} a_{k-1} - \dfrac{h}{w_k}\left(\dfrac{1}{2} - \sum_{i=1}^{k-1} w_i\right), & \sum_{i=1}^{k-1} w_i < 0{,}5 < \sum_{i=1}^{k} w_i \\ a_k, & \sum_{i=1}^{k} w_i = 0{,}5 \end{cases}$$

Выборочная мода

$$\overline{M_0} = a_{k-1} - h\frac{w_k - w_{k-1}}{2w_k - w_{k-1} - w_{k+1}}$$

Выборочный коэффициент асимметрии

$$\overline{\alpha_s} = \frac{\overline{\mu_3^o}}{\overline{\sigma^3}}$$

Выборочный коэффициент эксцесса

$$\overline{\varepsilon_k} = \frac{\overline{\mu_4^o}}{\overline{\sigma^4}} - 3$$

Нормальное распределение

Характеристика	Значение
Вероятность	$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-a)^2}{2\sigma^2}}$
Математическое ожидание	a
Дисперсия	σ^2
Среднее квадратичное отклонение	σ
Мода	a
Медиана	A
Коэффициент асимметрии	0
Коэффициент эксцесса	0

Показательное распределение

Характеристика	Значение
Вероятность	$\begin{cases} 0, x < 0 \\ \lambda e^{-\lambda x}, x \geq 0 \end{cases}$
Математическое ожидание	λ^{-1}
Дисперсия	λ^{-2}
Среднее квадратичное отклонение	λ^{-1}
Мода	0
Медиана	$\frac{\ln 2}{\lambda}$
Коэффициент асимметрии	2
Коэффициент эксцесса	6

Равномерное распределение на отрезке $[a,b]$

Характеристика	Значение
Вероятность	$\begin{cases} 0, x \notin (a,b) \\ \frac{1}{b-a}, x \in (a,b) \end{cases}$
Математическое ожидание	$\frac{a+b}{2}$
Дисперсия	$\frac{(b-a)^2}{12}$
Среднее квадратичное отклонение	$\frac{b-a}{2\sqrt{3}}$
Мода	$\frac{a+b}{2}$
Медиана	$\frac{a+b}{2}$
Коэффициент асимметрии	0
Коэффициент эксцесса	$-\frac{6}{5}$

Средства высокоуровневого интерпретируемого языка программирования Python, которые использованы в программе расчета

scipy.stats – модуль библиотеки scipy для работы со статистикой, в том числе и распределениями. Объектами класса являются типы распределений, методами – все методы, проходимые в курсе.

numpy – библиотека для эффективной работы с числами / массивами.

itertools.islice – метод библиотеки itertools для быстрого / эффективного взятия “слайсов”.

math.log2 – метод библиотеки math для взятия двоичного логарифма.

collections.Counter – класс словарь библиотеки collections, отличающийся от стандартного словаря возможностью автоматически считать частоту объектов.

matplotlib.pyplot – библиотека для построения графиков.

functools.reduce – метод reduce библиотеки functools.

copy.deepcopy – метод для копирования сложных структур.

random.choice – метод для выбора из предложенных вариантов.

collections.namedtuple – именованная структура.

np.unique – возвращение уникальных элементов массива.

np.asarray – представление структуры в виде numpy массива.

distribution.rvs – генерация рандомной выборки.

abs – модуль числа.

range – создание последовательности.

zip – произведение множеств.

.append – добавление элемента в конец списка.

matplotlib.rc – доступ к полям в графике.

plt.bar – построение диаграммы.

plt.show – вывод графика.

plt.savefig – сохранение графика.

plt.clf – очистка поля для построения нового графика.

sorted – сортировка структуры.

list – преобразование входной структуры в список.

islice – срез структуры.

plt.subplots – создание “подграфиков”.

ax.plot – редактирование конкретной части графика.

ax.set_title – заголовок графика.

ax.legend – заголовок всей картинки.

ax.set_ylabel – название оси y.

ax.set_xlabel – название оси x.

ax.set_xlim – масштаб оси x.

ax.set_ylim – масштаб оси y.

fig.tight_layout – автоматический подбор оптимального масштаба.

ax.set_axisbelow – добавление подсетки.

ax.minorticks_on – отображение подсетки.

distribution.stats – рассчет теоретической статистики.

distribution.median – теоретическая медиана.

distribution.std – теоретическое отклонение.

np.array – создание numpy массива.

.reshape – изменение размера массива numpy.

np.round – округление массива numpy.

string.ascii_lowercase – вывод англ алфавита.

.to_csv – сохранение таблицы в формате csv.

str – преобразование структуры в строку.

pd.DataFrame – создание таблицы.

._asdict – представление структуры в виде хэш таблицы.

.format – форматирование строки.

enumerate – умножение входящего множества на множество натуральных чисел.

Результаты расчетов с комментариями

Задание 1) Распределение по нормальному закону

a=-0.9

σ =1.09

Полученная выборка

0.0054	-1.206	-1.0945	-0.6038	-1.9171	0.6559	3.1601	2.3215	-0.2131	1.69
-3.2762	-2.1668	-0.9616	-0.0637	-0.0018	-3.4326	1.1465	-1.8645	-1.4254	-0.51
-2.6086	2.2747	0.2823	-1.3921	-1.74	1.0894	-0.318	-0.3374	-0.0297	-2.8329
1.7613	1.4387	-0.2434	-2.1334	-1.0177	-2.4202	-1.2955	-0.8613	-1.4735	-0.4311
-1.2338	-0.31	-2.4996	-0.6525	0.866	-1.1478	-2.3337	0.0815	-2.3772	-1.7441
-1.3118	-2.1854	-2.4218	-1.9215	-0.7592	-2.1776	1.5536	-1.2426	-0.5593	-0.3703
-0.8892	-0.6531	-5.1531	-2.3126	0.5019	-1.3681	-1.8703	-1.6759	-1.566	-0.2172
1.1098	-0.1361	-2.112	-2.1059	0.6242	-2.1045	-1.6247	-1.1468	-0.1162	-2.0852
-0.5836	-2.1708	-0.406	0.4	-1.0297	-2.2474	-1.103	-0.007	0.2898	0.1545
-1.025	-0.6954	0.0422	-2.7492	-1.0332	-0.0959	-0.8491	-0.6216	1.112	0.5114
-3.4971	-1.1621	-2.1053	-4.5448	-2.7498	-0.2763	-1.7919	-1.5537	0.6847	-4.3275
-0.1745	-1.6504	-1.0327	-1.8224	-1.7821	-0.3839	-1.4182	-2.6145	-0.8222	-1.6261
-1.4556	0.0053	-1.1956	-0.8238	-1.0639	-1.2467	-1.4377	-2.1662	0.6975	0.5
-0.7559	-1.384	0.3893	1.9015	0.4301	-0.7149	-0.426	-0.7909	-0.6294	1.3118
-1.1112	-2.8299	-1.9588	-2.0703	-3.5493	0.3809	-1.77	-3.6225	-1.6602	-0.4452
-1.7775	-0.9727	0.1616	-1.9573	-2.0215	-2.1535	-0.0089	-0.5554	-1.0469	-1.2181
-1.8538	-0.2608	-0.3394	0.0405	-0.581	-1.5098	0.2395	-2.6143	-0.1581	-2.4969
-2.1744	-2.2356	-0.1128	-2.1151	-1.8309	-1.8739	-2.1511	-0.5134	-1.3735	-1.755
-0.702	-1.4616	0.1132	0.7608	-1.9689	-1.1977	-1.548	-2.3208	-0.0779	-0.5909
-2.8815	-1.4314	-1.7223	-2.5419	-0.0671	0.8109	-1.169	-2.0622	1.9061	-3.2561

Упорядоченная выборка

-5.1531	-4.5448	-4.3275	-3.6225	-3.5493	-3.4971	-3.4326	-3.2762	-3.2561	-2.8815
-2.8329	-2.8299	-2.7498	-2.7492	-2.6145	-2.6143	-2.6086	-2.5419	-2.4996	-2.4969
-2.4218	-2.4202	-2.3772	-2.3337	-2.3208	-2.3126	-2.2474	-2.2356	-2.1854	-2.1776
-2.1744	-2.1708	-2.1668	-2.1662	-2.1535	-2.1511	-2.1334	-2.1151	-2.112	-2.1059
-2.1053	-2.1045	-2.0852	-2.0703	-2.0622	-2.0215	-1.9689	-1.9588	-1.9573	-1.9215
-1.9171	-1.8739	-1.8703	-1.8645	-1.8538	-1.8309	-1.8224	-1.7919	-1.7821	-1.7775
-1.77	-1.755	-1.7441	-1.74	-1.7223	-1.6759	-1.6602	-1.6504	-1.6261	-1.6247
-1.566	-1.5537	-1.548	-1.5098	-1.4735	-1.4616	-1.4556	-1.4377	-1.4314	-1.4254
-1.4182	-1.3921	-1.384	-1.3735	-1.3681	-1.3118	-1.2955	-1.2467	-1.2426	-1.2338
-1.2181	-1.206	-1.1977	-1.1956	-1.169	-1.1621	-1.1478	-1.1468	-1.1112	-1.103
-1.0945	-1.0639	-1.0469	-1.0332	-1.0327	-1.0297	-1.025	-1.0177	-0.9727	-0.9616
-0.8892	-0.8613	-0.8491	-0.8238	-0.8222	-0.7909	-0.7592	-0.7559	-0.7149	-0.702
-0.6954	-0.6531	-0.6525	-0.6294	-0.6216	-0.6038	-0.5909	-0.5836	-0.581	-0.5593
-0.5554	-0.5134	-0.51	-0.4452	-0.4311	-0.426	-0.406	-0.3839	-0.3703	-0.3394
-0.3374	-0.318	-0.31	-0.2763	-0.2608	-0.2434	-0.2172	-0.2131	-0.1745	-0.1581
-0.1361	-0.1162	-0.1128	-0.0959	-0.0779	-0.0671	-0.0637	-0.0297	-0.0089	-0.007
-0.0018	0.0053	0.0054	0.0405	0.0422	0.0815	0.1132	0.1545	0.1616	0.2395
0.2823	0.2898	0.3809	0.3893	0.4	0.4301	0.5	0.5019	0.5114	0.6242
0.6559	0.6847	0.6975	0.7608	0.8109	0.866	1.0894	1.1098	1.112	1.1465
1.3118	1.4387	1.5536	1.69	1.7613	1.9015	1.9061	2.2747	2.3215	3.1601

Группированная выборка (интервальный вариационный ряд)

(-0.9965, 0.0427)	(-2.0356, -0.9965)	(-3.0748, -2.0356)	(-4.1139, -3.0748)	(-5.1531, -4.1139)	(0.0427, 1.0818)	(1.0818, 2.121)	(2.121, 3.1601)
57	63	36	6	3	21	11	3

Ассоциированный статистический ряд

-4.6335	-3.5944	-2.5552	-1.5161	-0.4769	0.5622	1.6014	2.6405
3.0	6.0	36.0	63.0	57.0	21.0	11.0	3.0
0.015	0.03	0.18	0.315	0.285	0.105	0.055	0.015

Проверка выполнения условия была выполнена в программе. $\sum_{i=1}^{n} w_i = 1$, где n – количество x_i

Гистограмма относительных частот

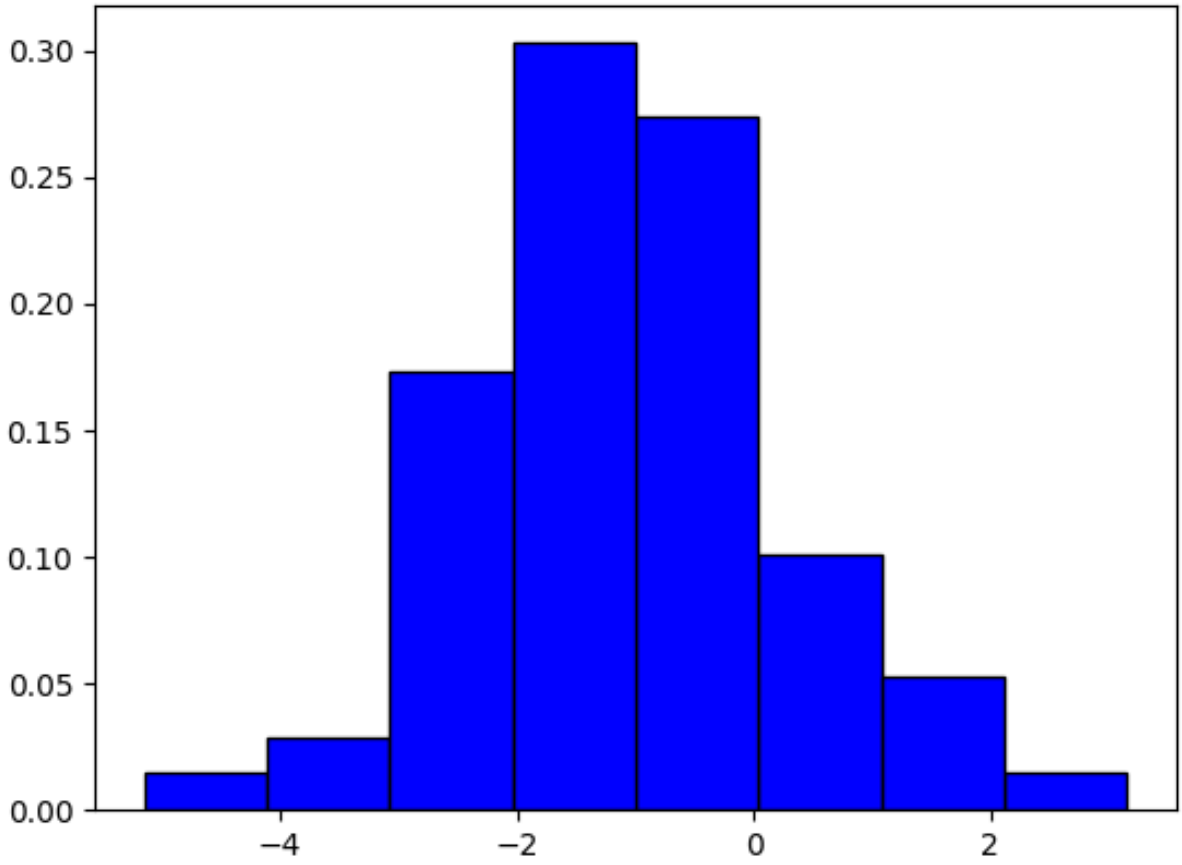
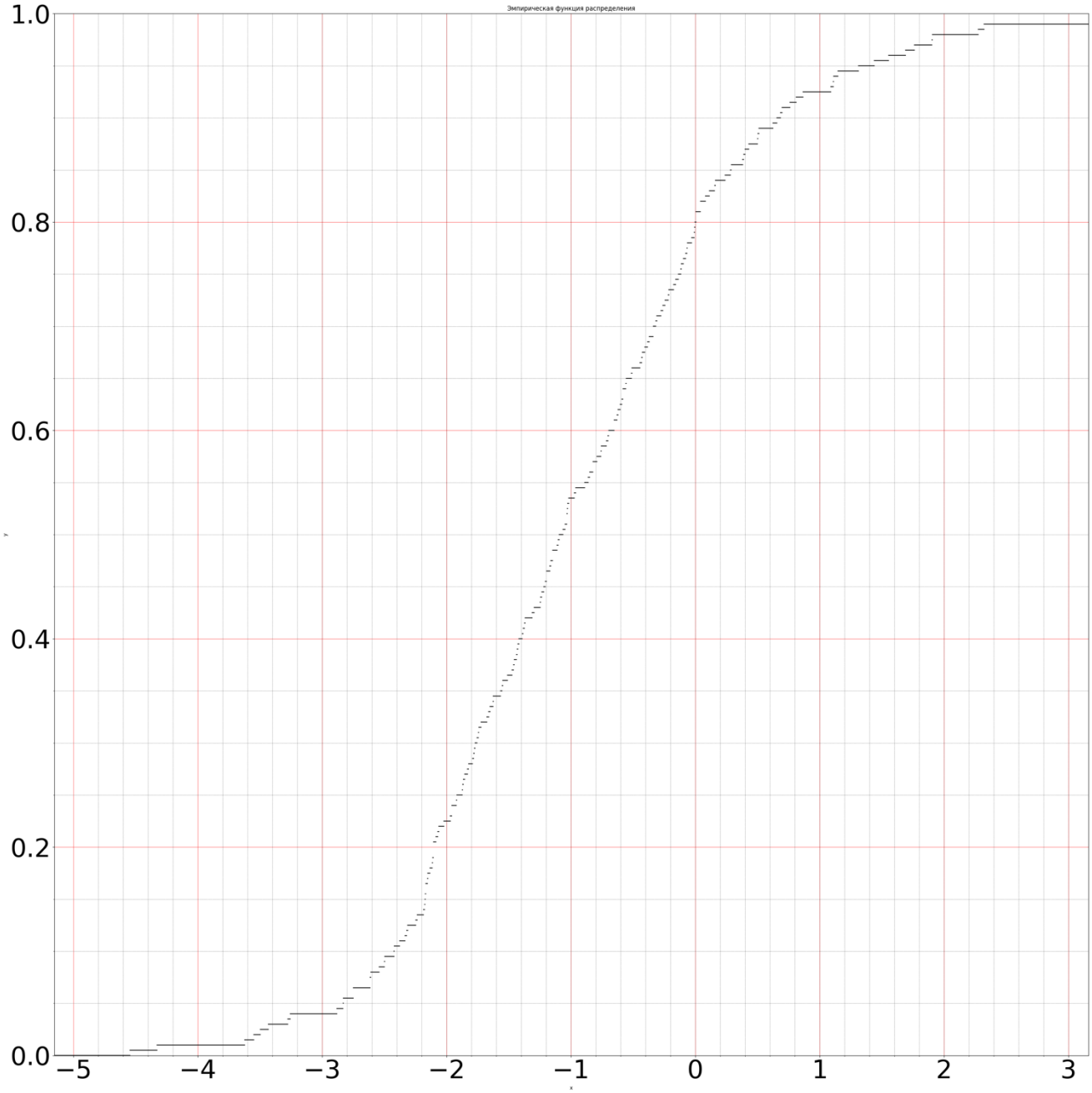


График эмпирической функции распределения



Результаты расчетов требуемых характеристик

выборочное среднее	-1.064
выборочная дисперсия с поправкой Шеппарда	1.7304
выборочное среднее квадратическое отклонение	1.3154
выборочная мода	-1.1854
выборочная медиана	-1.1284
выборочный коэффициент асимметрии	0.2281
выборочный коэффициент эксцесса	0.6529

Задание 2) Распределение по показательному закону

λ=1.91

Полученная выборка

1.0926	0.3663	0.0317	0.2599	0.2411	0.4506	0.2668	0.0253	0.3684	0.7742
0.1104	0.2941	0.4839	0.1207	0.1796	0.4361	0.0163	0.2491	0.7134	0.4086
0.5232	0.2328	0.2874	0.1906	0.9341	0.5523	0.4436	0.1724	0.4748	0.1172
1.0998	0.2663	0.3862	0.9662	0.031	0.1079	0.4423	0.2553	0.9253	0.0608
0.1252	0.3662	0.8639	0.2983	0.1004	0.1178	1.2647	0.018	0.4618	0.692
0.1396	2.4545	0.077	0.6364	0.1461	0.0444	0.1894	0.9661	0.3049	1.0175
0.5783	0.7796	0.6476	0.8734	0.0249	0.4115	0.589	0.6861	0.3013	0.3788
0.0975	1.1705	0.1082	0.4533	0.1225	0.0597	0.2345	0.6517	0.1113	0.2879
0.1945	0.4529	0.0221	0.337	0.1012	0.0823	0.012	0.2909	0.4029	0.1889
0.3592	0.4121	0.0294	0.0062	1.1717	1.2459	0.2138	0.2652	0.1494	0.0318
0.5029	0.5117	0.0193	0.0464	0.2866	0.125	0.1668	0.0919	0.0005	0.7517
0.1558	1.2171	0.4748	0.6094	0.4376	0.7227	0.1663	0.177	0.4588	0.2379
0.3717	0.378	0.8172	1.366	0.0407	1.143	0.308	0.6124	0.7319	0.4992
0.0168	0.5839	0.0376	0.3691	0.381	0.0834	2.8035	0.2937	0.6344	1.1279
0.5517	0.6034	0.0879	0.2323	0.3147	0.2311	0.1187	0.0687	1.4426	0.126
0.0879	0.1811	0.3239	0.1664	0.7125	0.3544	0.2097	0.882	0.4564	0.0138
0.0192	0.9744	0.1587	0.6887	0.3837	0.3764	0.0814	1.1794	0.4267	0.1523
0.2022	0.613	0.4328	0.5475	1.2378	0.0593	0.0999	0.6679	0.6207	0.4355
0.0547	0.1618	0.0087	0.2	0.634	0.5455	1.6948	0.9511	0.63	0.2899
0.3215	0.5637	0.4734	0.3473	1.3455	0.7006	0.5935	0.229	0.586	0.0099

Упорядоченная выборка

0.0005	0.0062	0.0087	0.0099	0.012	0.0138	0.0163	0.0168	0.018	0.0192
0.0193	0.0221	0.0249	0.0253	0.0294	0.031	0.0317	0.0318	0.0376	0.0407
0.0444	0.0464	0.0547	0.0593	0.0597	0.0608	0.0687	0.077	0.0814	0.0823
0.0834	0.0879	0.0879	0.0919	0.0975	0.0999	0.1004	0.1012	0.1079	0.1082
0.1104	0.1113	0.1172	0.1178	0.1187	0.1207	0.1225	0.125	0.1252	0.126
0.1396	0.1461	0.1494	0.1523	0.1558	0.1587	0.1618	0.1663	0.1664	0.1668

0.1724	0.177	0.1796	0.1811	0.1889	0.1894	0.1906	0.1945	0.2	0.2022
0.2097	0.2138	0.229	0.2311	0.2323	0.2328	0.2345	0.2379	0.2411	0.2491
0.2553	0.2599	0.2652	0.2663	0.2668	0.2866	0.2874	0.2879	0.2899	0.2909
0.2937	0.2941	0.2983	0.3013	0.3049	0.308	0.3147	0.3215	0.3239	0.337
0.3473	0.3544	0.3592	0.3662	0.3663	0.3684	0.3691	0.3717	0.3764	0.378
0.3788	0.381	0.3837	0.3862	0.4029	0.4086	0.4115	0.4121	0.4267	0.4328
0.4355	0.4361	0.4376	0.4423	0.4436	0.4506	0.4529	0.4533	0.4564	0.4588
0.4618	0.4734	0.4748	0.4748	0.4839	0.4992	0.5029	0.5117	0.5232	0.5455
0.5475	0.5517	0.5523	0.5637	0.5783	0.5839	0.586	0.589	0.5935	0.6034
0.6094	0.6124	0.613	0.6207	0.63	0.634	0.6344	0.6364	0.6476	0.6517
0.6679	0.6861	0.6887	0.692	0.7006	0.7125	0.7134	0.7227	0.7319	0.7517
0.7742	0.7796	0.8172	0.8639	0.8734	0.882	0.9253	0.9341	0.9511	0.9661
0.9662	0.9744	1.0175	1.0926	1.0998	1.1279	1.143	1.1705	1.1717	1.1794
1.2171	1.2378	1.2459	1.2647	1.3455	1.366	1.4426	1.6948	2.4545	2.8035

Группированная выборка (интервальный вариационный ряд)

(0.0005, 0.3509)	(0.3509, 0.7012)	(0.7012, 1.0516)	(1.0516, 1.402)	(1.402, 1.7524)	(2.4532, 2.8035)
101	64	18	13	2	2

Ассоциированный статистический ряд

0.1757	0.526	0.8764	1.2268	1.5772	2.6284
101.0	64.0	18.0	13.0	2.0	2.0
0.505	0.32	0.09	0.065	0.01	0.01

Проверка выполнения условия была выполнена в программе.

$$\sum_{i=1}^{n} w_i = 1, \text{где } n - \text{количество } x_i$$

Гистограмма относительных частот

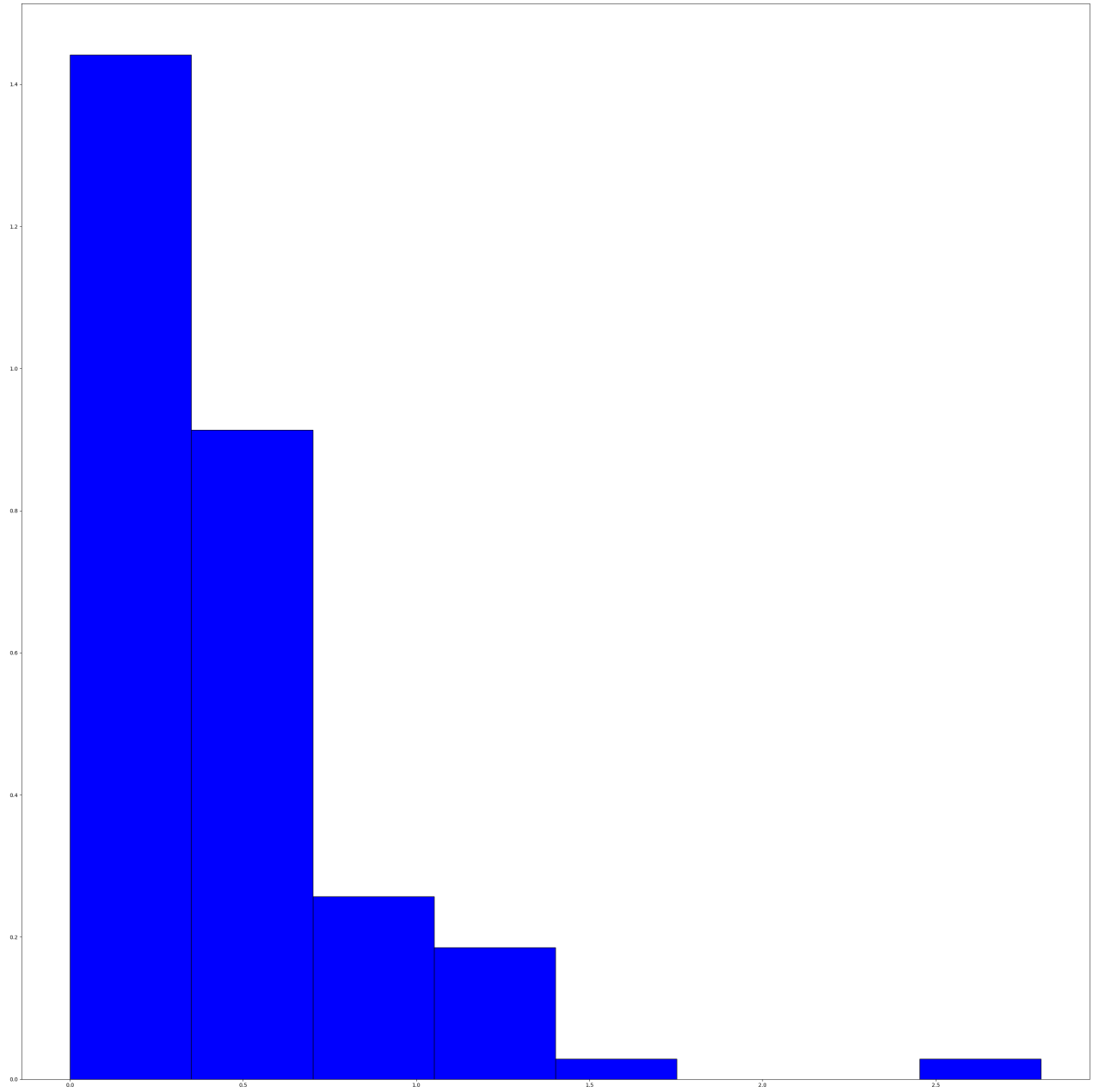
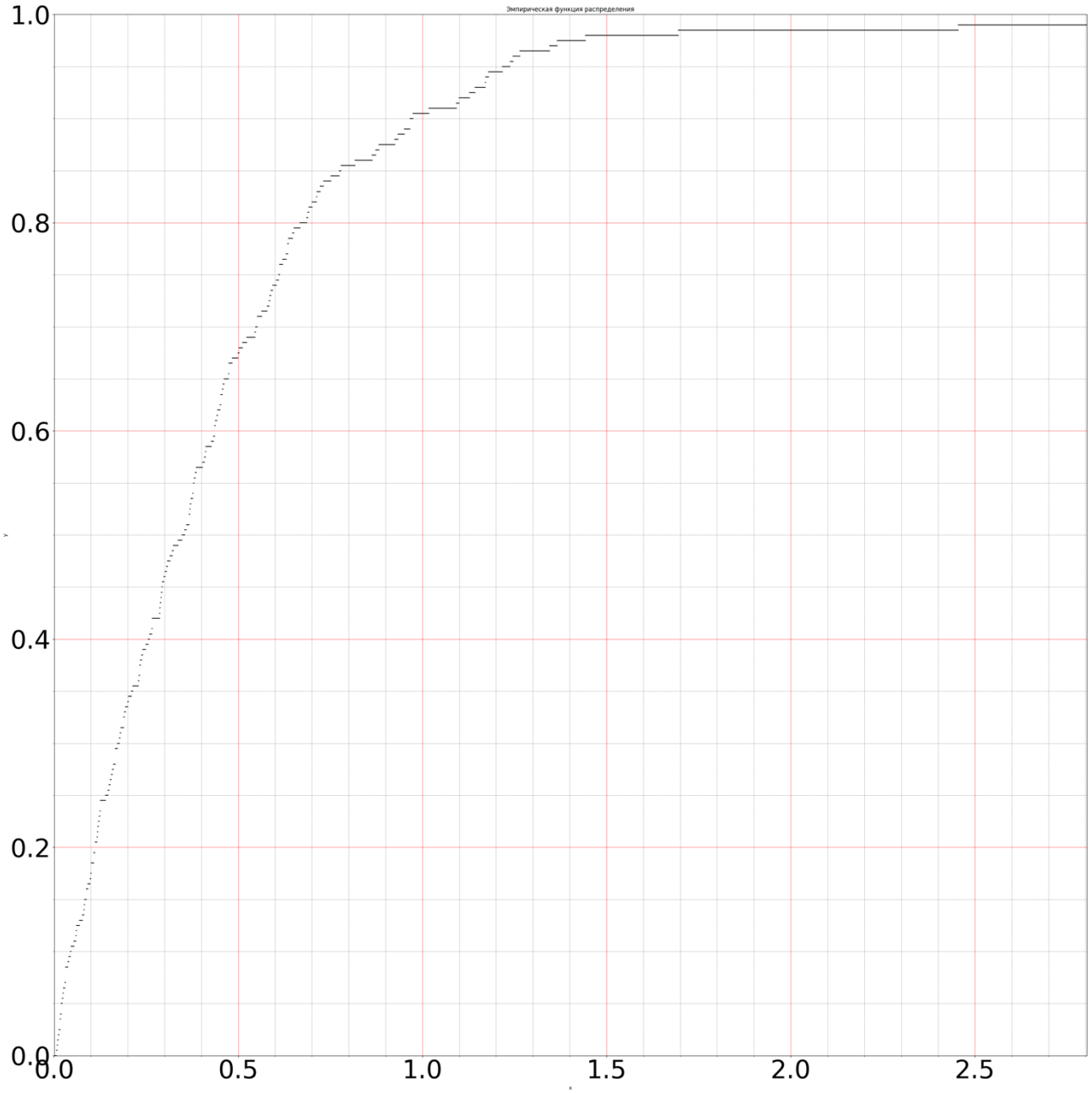


График эмпирической функции распределения



Результаты расчетов требуемых характеристик

выборочное среднее	0.4577
выборочная дисперсия с поправкой Шеппарда	0.1453
выборочное среднее квадратическое отклонение	0.3812
выборочная мода	0.2555
выборочная медиана	0.3474
выборочный коэффициент асимметрии	2.5499
выборочный коэффициент эксцесса	9.6164

Задание 3) Равномерное распределение на отрезке [*a,b*]

a=-0.45 b=1.0

Полученная выборка

-0.1375	0.3458	0.5186	0.3008	0.5311	-0.4123	0.0358	-0.1347	0.2798	-0.3736
-0.2411	0.5288	0.5085	-0.2646	0.4524	0.3351	0.0628	-0.0162	0.2736	0.0627
-0.2175	-0.0179	0.191	-0.1138	-0.3769	0.4278	0.0351	0.5141	-0.2142	0.3623
0.3829	-0.1192	-0.2881	0.3501	0.0121	-0.1385	-0.3478	0.2789	0.3773	-0.1794
-0.3638	-0.0057	0.5406	-0.3063	-0.3029	-0.43	0.2227	0.2988	0.2504	-0.0918
0.4313	-0.1752	0.3455	0.3863	0.135	0.4417	-0.3221	0.515	-0.0787	0.501
0.5206	-0.1343	-0.184	0.0526	0.256	-0.1551	-0.1833	-0.3315	0.4645	-0.2128
0.3623	0.3531	0.2415	0.2188	0.4706	-0.292	0.1785	-0.1667	0.1977	0.3198
-0.4273	0.2412	0.0833	0.0899	-0.1286	-0.0931	0.2716	-0.3841	-0.3533	0.4872
0.0233	-0.4448	-0.1029	0.3905	0.44	-0.2303	0.3822	-0.1592	0.4866	0.3961
0.1102	-0.0921	-0.2191	-0.0773	0.2849	-0.2308	0.1324	-0.0104	-0.3817	0.0195
-0.1351	-0.2425	0.0634	-0.0008	-0.1504	-0.0849	0.3379	0.2422	-0.2818	-0.0576
-0.4295	-0.2044	0.0882	0.4028	-0.3545	0.3439	-0.1299	-0.2516	0.2704	0.4668
0.0969	0.3146	0.087	0.4708	-0.2746	-0.2073	-0.3209	-0.2805	-0.2201	-0.2765
0.2388	0.3004	0.186	0.3231	0.2982	-0.1472	-0.0894	0.4522	-0.2966	-0.1218
0.5391	0.5152	0.1833	0.347	0.4625	-0.1661	-0.418	0.183	0.2746	-0.1203
-0.0192	-0.1907	0.2291	0.0496	-0.3748	0.2398	-0.0715	-0.4383	0.2209	-0.4013
0.0658	0.3476	0.0959	0.503	0.13	-0.0575	0.3071	0.3219	0.0676	0.0208
-0.3178	-0.1229	0.0693	0.4062	-0.1186	0.2399	-0.2286	-0.1922	0.012	0.5048
0.2664	-0.2087	0.4571	0.4347	0.2758	-0.0968	-0.3369	-0.3135	-0.0334	0.4394

Упорядоченная выборка

-0.4448	-0.4383	-0.43	-0.4295	-0.4273	-0.418	-0.4123	-0.4013	-0.3841	-0.3817
-0.3769	-0.3748	-0.3736	-0.3638	-0.3545	-0.3533	-0.3478	-0.3369	-0.3315	-0.3221
-0.3209	-0.3178	-0.3135	-0.3063	-0.3029	-0.2966	-0.292	-0.2881	-0.2818	-0.2805
-0.2765	-0.2746	-0.2646	-0.2516	-0.2425	-0.2411	-0.2308	-0.2303	-0.2286	-0.2201
-0.2191	-0.2175	-0.2142	-0.2128	-0.2087	-0.2073	-0.2044	-0.1922	-0.1907	-0.184
-0.1833	-0.1794	-0.1752	-0.1667	-0.1661	-0.1592	-0.1551	-0.1504	-0.1472	-0.1385
-0.1375	-0.1351	-0.1347	-0.1343	-0.1299	-0.1286	-0.1229	-0.1218	-0.1203	-0.1192
-0.1186	-0.1138	-0.1029	-0.0968	-0.0931	-0.0921	-0.0918	-0.0894	-0.0849	-0.0787

-0.0773	-0.0715	-0.0576	-0.0575	-0.0334	-0.0192	-0.0179	-0.0162	-0.0104	-0.0057
-0.0008	0.012	0.0121	0.0195	0.0208	0.0233	0.0351	0.0358	0.0496	0.0526
0.0627	0.0628	0.0634	0.0658	0.0676	0.0693	0.0833	0.087	0.0882	0.0899
0.0959	0.0969	0.1102	0.13	0.1324	0.135	0.1785	0.183	0.1833	0.186
0.191	0.1977	0.2188	0.2209	0.2227	0.2291	0.2388	0.2398	0.2399	0.2412
0.2415	0.2422	0.2504	0.256	0.2664	0.2704	0.2716	0.2736	0.2746	0.2758
0.2789	0.2798	0.2849	0.2982	0.2988	0.3004	0.3008	0.3071	0.3146	0.3198
0.3219	0.3231	0.3351	0.3379	0.3439	0.3455	0.3458	0.347	0.3476	0.3501
0.3531	0.3623	0.3623	0.3773	0.3822	0.3829	0.3863	0.3905	0.3961	0.4028
0.4062	0.4278	0.4313	0.4347	0.4394	0.44	0.4417	0.4522	0.4524	0.4571
0.4625	0.4645	0.4668	0.4706	0.4708	0.4866	0.4872	0.501	0.503	0.5048
0.5085	0.5141	0.515	0.5152	0.5186	0.5206	0.5288	0.5311	0.5391	0.5406

Группированная выборка (интервальный вариационный ряд)

(-0.0753, 0.0479)	(-0.1985, -0.0753)	(-0.3217, -0.1985)	(-0.4448, -0.3217)	(0.0479, 0.171)	(0.171, 0.2942)	(0.2942, 0.4174)	(0.4174, 0.5406)
17	34	27	20	18	27	28	29

Ассоциированный статистический ряд

-0.3833	-0.2601	-0.1369	-0.0137	0.1094	0.2326	0.3558	0.479
20.0	27.0	34.0	17.0	18.0	27.0	28.0	29.0
0.1	0.135	0.17	0.085	0.09	0.135	0.14	0.145

Проверка выполнения условия была выполнена в программе. $\sum_{i=1}^{n} w_i = 1$, где n – количество x_i

Гистограмма относительных частот

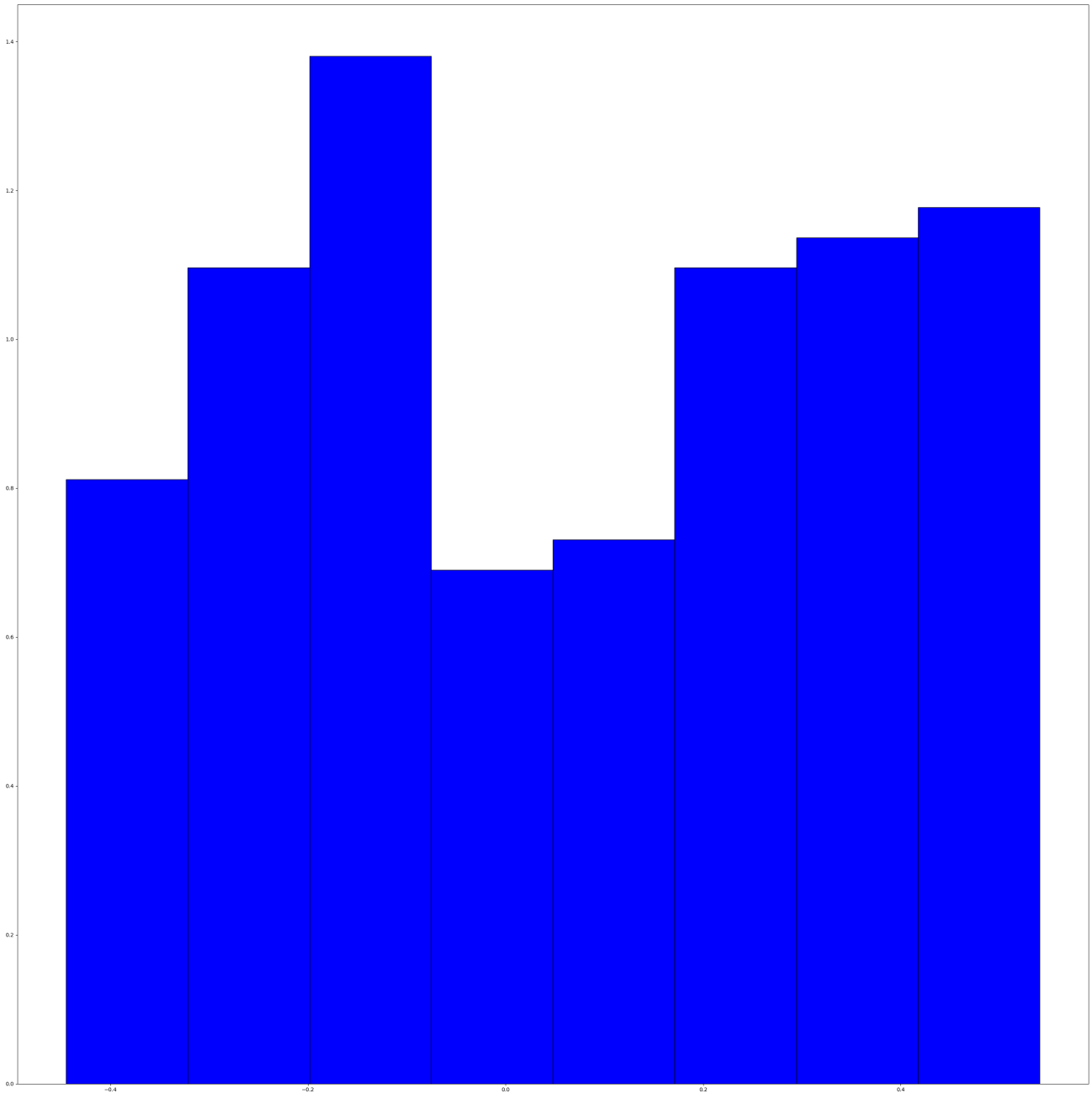
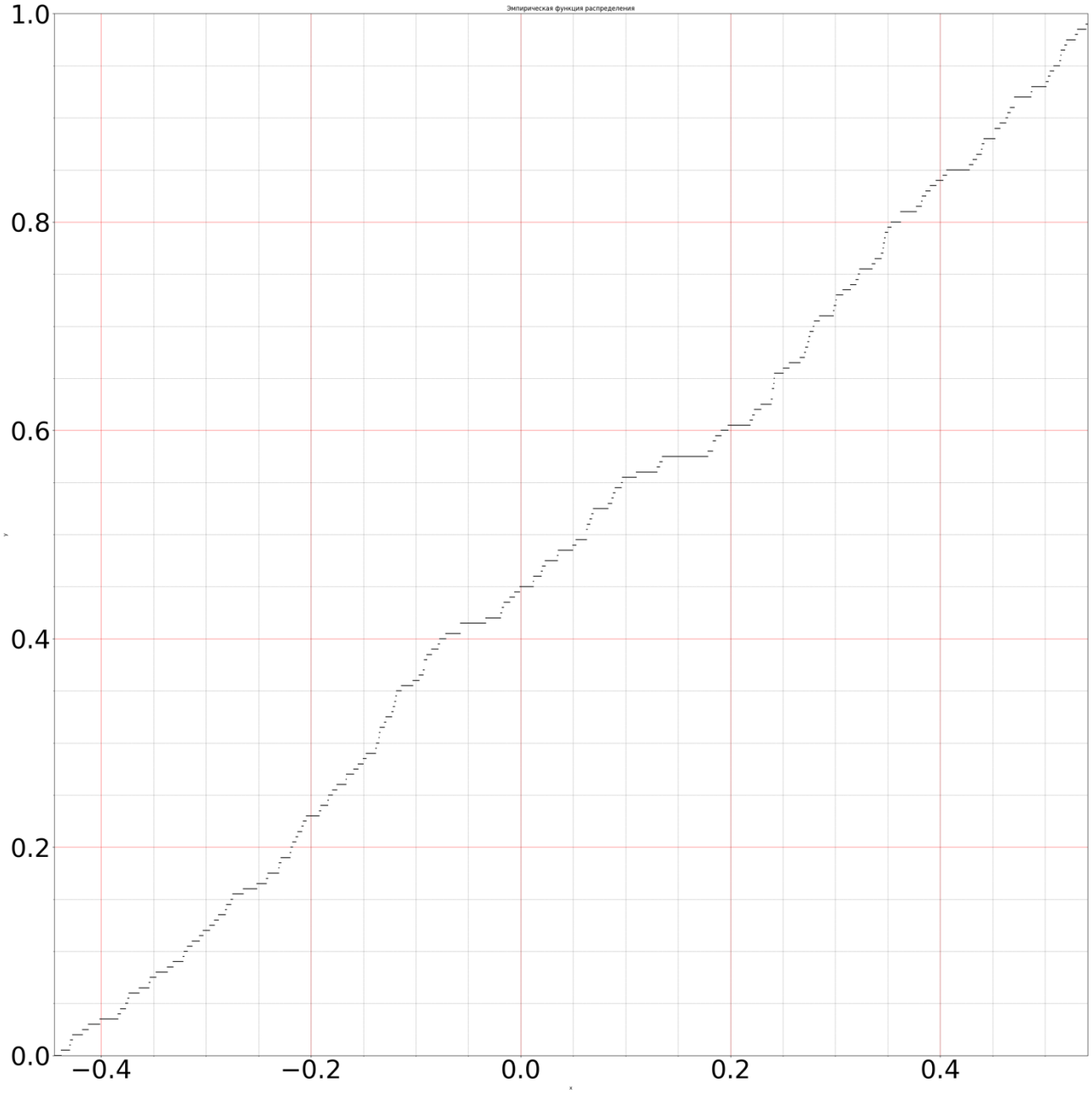


График эмпирической функции распределения



Результаты расчетов требуемых характеристик

выборочное среднее	0.0626
выборочная дисперсия с поправкой Шеппарда	0.0812
выборочное среднее квадратическое отклонение	0.285
выборочная мода	-0.1626
выборочная медиана	0.0615
выборочный коэффициент асимметрии	-0.0055
выборочный коэффициент эксцесса	-1.3026

Анализ результатов и выводы

1) Распределение по нормальному закону

Название показателя	Эксперимен-тальное значение	Теоретическое значение	Абсолютное отклонение	Относительное отклонение
Выборочное среднее	1.7304	1.4116	0.3188	0.2258
Выборочная дисперсия с поправкой Шеппарда	-1.064	-0.9	0.164	-0.1823
Выборочное среднее квадратичное отклонение	1.3154	1.1881	0.1273	0.1072
Выборочная мода	-1.1284	-0.9	0.2284	-0.2538
Выборочная медиана	0.6529	0.0	0.6529	inf
Выборочный коэффициент асимметрии	-1.1854	-0.9	0.2854	-0.3171
Выборочный коэффициент эксцесса	0.2281	0.0	0.2281	inf

Интервал	w_i	p_i	$\lvert w_i - p_i \rvert$
[-5.15308, -4.11393]	0.015	0.0032	0.0118
(-4.11393, -3.07478]	0.03	0.0302	0.0002
(-3.07478, -2.03563]	0.18	0.136	0.044
(-2.03563, -0.99648]	0.315	0.2981	0.0169
(-0.99648, 0.042667]	0.285	0.3186	0.0336
(0.04266, 1.08181]	0.105	0.1661	0.0611
(1.08181, 2.12096]	0.055	0.0422	0.0128
(2.12096, 3.16012]	0.015	0.0052	0.0098
	$\sum_{i=1}^{n} w_i =$ 1	$\sum_{i=1}^{n} p_i =$ 0.9995	Δ_{max}=0.0611

2) Распределение по показательному закону

Название показателя	Эксперимен-тальное значение	Теоретическое значение	Абсолютное отклонение	Относительное отклонение
Выборочное среднее	0.1453	0.2741	0.1288	0.4699
Выборочная дисперсия с поправкой Шеппарда	0.4577	0.5236	0.0658	0.1258
Выборочное среднее квадратичное отклонение	0.3812	0.5236	0.1424	0.2719
Выборочная мода	0.3474	0.3629	0.0155	0.0428
Выборочная медиана	9.6164	6.0	3.6164	0.6027
Выборочный коэффициент асимметрии	0.2555	0.0	0.2555	inf
Выборочный коэффициент эксцесса	2.5499	2.0	0.5499	0.2749

Интервал	w_i	p_i	$\lvert w_i - p_i \rvert$
[0.00046, 0.35085]	0.505	0.4875	0.0175
(0.35085, 0.70123]	0.32	0.2496	0.0704
(0.70123, 1.05162]	0.09	0.1278	0.0378
(1.05162, 1.402]	0.065	0.0655	0.0005
(1.402, 1.75239]	0.01	0.0335	0.0235
(2.45316, 2.80354]	0.01	0.0045	0.0055
	$\sum_{i=1}^{n} w_i =$ 1	$\sum_{i=1}^{n} p_i =$ 0.9684	Δ_{max}=0.0704

3) Равномерное распределение на отрезке $[a,b]$

Название показателя	Эксперимен-тальное значение	Теоретическое значение	Абсолютное отклонение	Относительное отклонение
Выборочное среднее	0.0812	0.0833	0.0021	0.0255
Выборочная дисперсия с поправкой Шеппарда	0.0626	0.05	0.0126	0.2528
Выборочное среднее квадратичное отклонение	0.285	0.2887	0.0037	0.0129
Выборочная мода	0.0615	0.05	0.0115	0.2309
Выборочная медиана	-1.3026	-1.2	0.1026	-0.0855
Выборочный коэффициент асимметрии	-0.1626	0.275	0.4376	1.5911
Выборочный коэффициент эксцесса	-0.0055	0.0	0.0055	inf

Интервал	w_i	p_i	$\lvert w_i - p_i \rvert$
[-0.44483, -0.32166]	0.1	0.1232	0.0232
(-0.32166, -0.19849]	0.135	0.1232	0.0118
(-0.19849, -0.07531]	0.17	0.1232	0.0468
(-0.07531, 0.04785]	0.085	0.1232	0.0382
(0.04785, 0.17103]	0.09	0.1232	0.0332
(0.17103, 0.29420]	0.135	0.1232	0.0118
(0.2942, 0.41738]	0.14	0.1232	0.0168
(0.41738, 0.54055]	0.145	0.1232	0.0218
	$\sum_{i=1}^{n} w_i =$ 1	$\sum_{i=1}^{n} p_i =$ 0.9854	Δ_{max}=0.0468

Вывод: теоретические и экспериментальные в основном не сильно отличаются друг от друга, но были случаи, в которых достаточно большое относительное отклонение, но это из-за того, что взяли только 200 чисел.

Список использованной литературы

1. Лобузов А.А. Математическая статистика [Электронный ресурс]: Методические указания по выполнению лабораторных работ / под ред. Ю. И. Худака. Москва: Московский технологический университет (МИРЭА), 2017. 36 с.

2. Чернова Н. И. Математическая статистика: Учеб. пособие / Новосиб. гос. ун-т. Новосибирск, 2007. 148 с

Приложение (Листинг программы)

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

import sys
import argparse

from scipy import stats as st
import numpy as np
import pandas as pd

from copy import deepcopy
from collections import namedtuple
from collections import Counter
from functools import reduce
from copy import deepcopy
from random import choice
from itertools import islice
from math import log2
import string

import matplotlib.pyplot as plt
import matplotlib

def createParser():
    parser = argparse.ArgumentParser()
    parser.add_argument('-v', '--variant', default=9, type=int)
    parser.add_argument('-n', '--n', default=200, type=int)
    return parser

def findInterv(val: float, arr: list) -> int:
    test = lambda x, item: True if item[0] <= x <= item[1] else False
    for i, item in enumerate(arr):
        if test(val, item):
            return i
    return None

def expStats(sec_counter, first_counter, h):
    tmp = list(sec_counter.items())
    tmp.insert(0, 0.0)
    mean = reduce(lambda a, b: a + b[0] * b[1][1], tmp)
    s2 = reduce(lambda a, b: a + (b[0] - mean) ** 2 * b[1][1], tmp) - (h **
2) / 12
    ds = s2 ** 0.5

    ak, most_common = first_counter.most_common(1)[0]
    wn = sorted(first_counter.items(), key=lambda x: x[0])
    k = {val: i for i, (key, val) in enumerate(sorted(first_counter.items(),
key=lambda x: x[0]))}[most_common]
    wk = lambda k: sorted(sec_counter.items(), key=lambda x: x[0])[k][1][1]
```

```
    mode = ak[0] + h * ((wk(k) - wk(k-1)) / (2 * wk(k) - wk(k-1) - wk(k+1)))

    w = [val[1] for key, val in sorted(sec_counter.items(), key=lambda x: x)]
    sums = [reduce(lambda a, b: a + b, w[:i+1]) for i in range(len(w))]
    dsums = {el: i+1 for i, el in enumerate(sums)}
    try:
        prk = [(dsums[l], dsums[r]) for l, r in list(zip(sums, islice(sums,
1, None))) if l <= 0.5 < r][0][1]
        a = [key for key, val in sorted(first_counter.items(), key=lambda x:
(x[0]))]
        a = np.unique(np.asarray(a).reshape(-1, 1)).tolist()
        med = a[prk-1] + (h / w[prk-1]) * (0.5 - sums[prk-2])
    except:
        prk = 1
        a = [key for key, val in sorted(first_counter.items(), key=lambda x:
(x[0]))]
        a = np.unique(np.asarray(a).reshape(-1, 1)).tolist()
        med = a[prk-1] + (h / w[prk-1]) * (0.5)

    mk = lambda k: sum(map(lambda item: item[0] ** k * item[1][1],
sec_counter.items()))
    mck = lambda k: sum(map(lambda item: (item[0] - mean) ** k * item[1][1],
sec_counter.items()))
    skew, kurtosis = mck(3) / (ds ** 3), (mck(4) / (ds ** 4)) - 3

    return mean, s2, ds, mode, med, skew, kurtosis

class LabFitter3000(object):
    '''IN: distribution name, params, scipy class
    OUT: path to figs, experimental stats, theoretical stats'''
    def __init__(self, obj, N, name, *params):
        self.distribution = obj
        self.N = N
        self.distribution_name = name
        self.params = params
        self.rvs = None
        self.h = None
        self.first_counter = None
        self.sec_counter = None
        self.pathes = []
        self.experimental_stats = None
        self.theoretical_stats = None
        self.Stats = namedtuple('Stats', 'mean variance std mode med skew
kurtosis')

    def create_rvs(self):
        self.rvs = list(self.distribution.rvs(size=self.N))

    def create_first_counter(self):
        m = int(1 + log2(self.N))
        sr = sorted(self.rvs)
```

```
        f, l = sr[0], sr[-1]
        d = abs(l - f)
        self.h = d / m
        interv = [f + self.h * i for i in range(m)]
        interv.append(l)
        interv = list(zip(interv, islice(interv, 1, None)))

        self.first_counter = Counter()
        for val in sr:
            self.first_counter[interv[findInterv(val, interv)]] += 1

    def create_second_counter(self):
        self.sec_counter = {(key[1] + key[0]) / 2: (val, val / self.N)
                            for key, val in self.first_counter.items()}

    def hist(self, show: bool = False):#, sec_counter: dict, h: float, show:
bool = False, path: str = 'Data/hist.png'):
        matplotlib.rc('xtick', labelsize=10)
        matplotlib.rc('ytick', labelsize=10)
        tmp = {key: val[1] / self.h for key, val in self.sec_counter.items()}
        plt.bar(list(tmp.keys()),
                list(tmp.values()),
                color='b',
                edgecolor='black',
                width=self.h)
        if show:
            plt.show()
        else:
            path = 'Data/' + self.distribution_name + '_hist.png'
            plt.savefig(path)
            plt.clf()

    def cdf(self, show: bool = False, sizex: int = 30, sizey: int = 30):#,
sr: list, N: int, sizex: int = 30, sizey: int = 30, show: bool = False, path:
str = 'Data/cdf.png'):

        matplotlib.rc('xtick', labelsize=50)
        matplotlib.rc('ytick', labelsize=50)
        sr = sorted(self.rvs)
        items = list(zip(sr, islice(sr, 1, None)))
        delta = 0.00001
        Xlist = [[x * delta for x in range(int(item[0] / delta), int(item[1]
/ delta))]
                 for item in items]
        Ylist = [[i / self.N] * len(Xlist[i]) for i in range(self.N-1)]

        ###############################################################

        fig, ax = plt.subplots(figsize=(sizex, sizey))

        for X, Y in zip(Xlist, Ylist):
```

```
            ax.plot(X, Y, label='', color='black')
        ax.set_title('Эмпирическая функция распределения')
        ax.legend(loc='upper left')
        ax.set_ylabel('y')
        ax.set_xlabel('x')
        ax.set_xlim(xmin=sr[0], xmax=sr[-1])
        ax.set_ylim(ymin=0, ymax=1)
        fig.tight_layout()
        if sizex != 5:
            ax.set_axisbelow(True)
            ax.minorticks_on()
            ax.grid(which='major', linestyle='-', linewidth='0.5',
color='red')
            ax.grid(which='minor', linestyle=':', linewidth='0.5',
color='black')
        else:
            ax.grid()
        if show:
            plt.show()
        else:
            path = 'Data/' + self.distribution_name + '_cdf.png'
            plt.savefig(path)
            plt.clf()

    def create_theoretical_stats(self, mode_generator):
        mean, variance, skew, kurtosis =
self.distribution.stats(moments='mvsk')
        med = self.distribution.median()
        std = self.distribution.std()
        mode = mode_generator(*self.params)

        self.theoretical_stats = self.Stats(mean=float(mean),
                                            variance=float(variance),
                                            std=std,
                                            mode=mode,
                                            med=med,
                                            skew=float(skew),
                                            kurtosis=float(kurtosis))

    def create_experimental_stats(self, exp_stats_generator):
        mean, variance, std, mode, med, skew, kurtosis =
exp_stats_generator(self.sec_counter,

self.first_counter,

self.h)
        self.experimental_stats = self.Stats(mean=mean,
                                             variance=variance,
                                             std=std,
                                             mode=mode,
                                             med=med,
```

```
                                                skew=skew,
                                                kurtosis=kurtosis)

    def save_rvs(self, shape: int = 20, dec = 4):
        rvs = np.array(self.rvs)
        rvs = rvs.reshape(shape, -1)
        rvs = np.round(rvs, dec)
        df_rvs = pd.DataFrame(rvs,

columns=list(string.ascii_lowercase)[:rvs.shape[1]])
        df_rvs.to_csv('Data/' + self.distribution_name + '_rvs.csv',
                      sep=';',
                      encoding='utf-8',
                      index=False)

        sorted_rvs = np.array(sorted(self.rvs))
        sorted_rvs = sorted_rvs.reshape(shape, -1)
        sorted_rvs = np.round(sorted_rvs, dec)
        df_sorted_rvs = pd.DataFrame(sorted_rvs,

columns=list(string.ascii_lowercase)[:sorted_rvs.shape[1]])
        df_sorted_rvs.to_csv('Data/' + self.distribution_name +
'_sorted_rvs.csv',
                             sep=';',
                             encoding='utf-8',
                             index=False)

    def save_dict(self):
        res = {str((round(key[0], 4), round(key[1], 4))): [val]
               for key, val in self.first_counter.items()}
        df = pd.DataFrame(res)
        df.to_csv('Data/' + self.distribution_name + '_first_counter.csv',
                  sep=';',
                  encoding='utf-8')
        res = {round(key, 4): val
               for key, val in self.sec_counter.items()}
        df = pd.DataFrame(res)
        df.to_csv('Data/' + self.distribution_name + '_second_counter.csv',
                  sep=';',
                  encoding='utf-8')

    def save_stats(self, dec = 4):
        experimental = self.experimental_stats._asdict()
        df = pd.DataFrame(list(np.round(list(experimental.values()), dec)),
                          index=experimental.keys(),
                          columns=[['values']])
        df.to_csv('Data/' + self.distribution_name +
'_experimental_stats.csv',
                  sep=';',
                  encoding='utf-8')
```

```
        theoretical = self.theoretical_stats._asdict()
        df = pd.DataFrame(list(np.round(list(theoretical.values()), dec)),
                          index=theoretical.keys(),
                          columns=[['values']])
        df.to_csv('Data/' + self.distribution_name +
'_theoretical_stats.csv',
                  sep=';',
                  encoding='utf-8')

        all_stats = list(zip(self.experimental_stats._asdict().keys(),
list(zip(self.experimental_stats, self.theoretical_stats))))
        result = {key: (round(e, dec), round(t, dec), round(abs(e-t), dec),
round(abs(e-t)/t, dec))
                  for key, (e, t) in all_stats}
        df = pd.DataFrame(list(result.values()),
                          index=list(result.keys()))
        df.to_csv('Data/' + self.distribution_name + '_all_stats.csv',
                  sep=';',
                  encoding='utf-8')

    def create_and_save_p(self, dec = 4):
        cur_r = self.distribution
        cur_x = sorted(list(self.first_counter.keys()))
        cur_w = [self.sec_counter[key][1]
                 for key in sorted(list(self.sec_counter.keys()))]

        p = [cur_r.cdf(b) - cur_r.cdf(a) for a, b in cur_x]
        s = sum(p)
        wp = [abs(a - b) for a, b in zip(cur_w, p)]
        m = max(wp)
        new_w = deepcopy(cur_w)
        new_w.append(1.0)
        p.append(s)
        wp.append(m)
        new_x = deepcopy(cur_x)
        new_x.append('-')

        df = pd.DataFrame(np.round(np.array([new_w, p, wp]).T, 4),
                          index=new_x)
        df.to_csv('Data/' + self.distribution_name + '_wp' + '.csv',
                  sep=';',
                  encoding='utf-8')


def fitter(obj, mode_generator):
    obj.create_rvs()
    obj.create_first_counter()
    obj.create_second_counter()
    obj.hist()
    obj.cdf()
    obj.create_theoretical_stats(mode_generator)
```

```
    obj.create_experimental_stats(expStats)
    obj.save_rvs()
    obj.save_dict()
    obj.save_stats()
    obj.create_and_save_p()
    return obj


if __name__ == '__main__':
    parser = createParser()
    namespace = parser.parse_args(sys.argv[1:])
    variant = namespace.variant
    N = namespace.n

    mode = {'norm': lambda a, sgm: a,
            'expon': lambda lmbd: 0,
            'uniform': lambda a, b: sum([a, b])/2}

    Obj = namedtuple('Obj', 'name params distribution')

    mu = (-1) ** variant * 0.1 * variant
    sgm = (0.01 * variant + 1) ** 2

    lmbd = 2 + (-1) ** variant * 0.01 * variant

    a = (-1) ** variant * 0.05 * variant
    b = a + 0.05 * variant + 1

    print('mu = {0}\nsgm = {1}\nlmbd = {2}\na = {3}\nb = {4}'.format(mu,
                                                                     sgm **
(1/2),
                                                                     lmbd,
                                                                     a,
                                                                     b))


    Data = [Obj(name='norm',
                params=[mu, sgm],
                distribution=st.norm(mu, sgm)),
            Obj(name='expon',
                params=[1/lmbd],
                distribution=st.expon(scale=1/lmbd)),
            Obj(name='uniform',
                params=[a, b],
                distribution=st.uniform(a, b))]

    result = []
    for obj in Data:
        result.append(LabFitter3000(obj.distribution, N, obj.name,
*obj.params))

    result = [fitter(lf, mode[Data[i].name]) for i, lf in enumerate(result)]
```