|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  | | --- | --- | --- | |  |  |  | | Министерство образования и науки РОССИЙСКОЙ ФЕДЕРАЦИИ | | |  Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА – Российский технологический университет»** |
|
|
|

ИНСТИТУТ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

КАФЕДРА ВЫСШЕЙ МАТЕМАТИКИ

Лабораторная работа 3

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

**ВАРИАНТ 46**

Тема: \_\_\_\_\_\_\_ **Проверка статистических гипотез с помощью** \_\_\_\_\_

\_\_\_\_\_\_\_ **критерия хи-квадрат и критерия Колмогорова**\_\_\_\_\_

Выполнил:

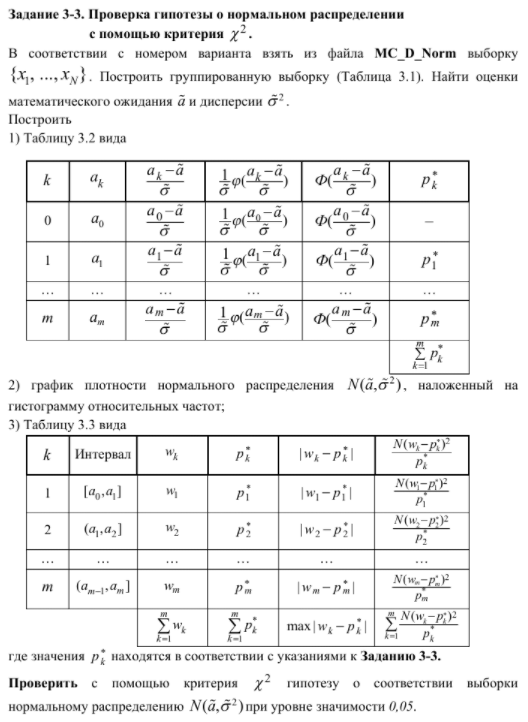
Студент 3-го курса

Успенский А.А.

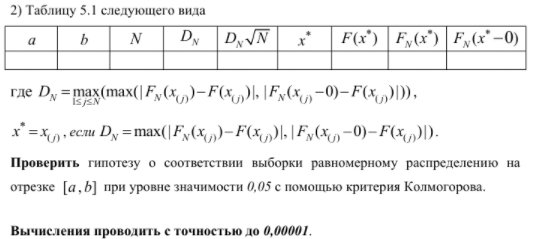
Группа: КМБО-03-19

МОСКВА – 2022

# **Задание**

****

****

**Краткие теоретические сведения**

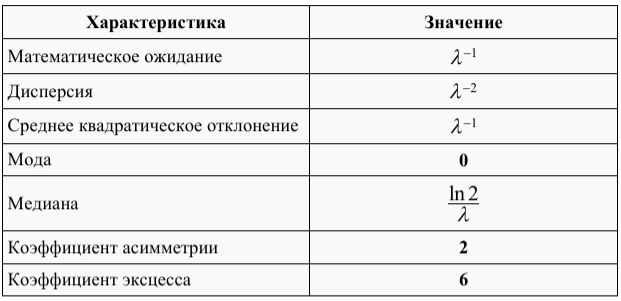
# **Нормальное распределение**



***Плотность***

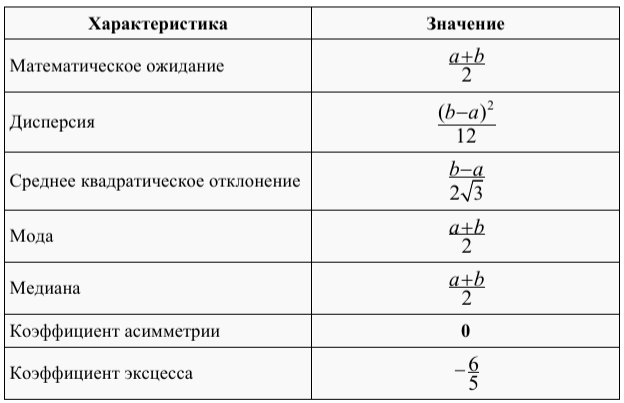
***Функция распределение***

## **Показательное распределение**



***Плотность***

***Функция распределение***

**Равномерное распределение на отрезке**

***Плотность***

***Функция распределение***

Ряд распределения - структурная группировка с целью выделения характерных свойств и закономерностей изучаемой совокупности.

Математическое ожидание – понятие среднего значения случайной величины в теории вероятностей.

Дисперсия – отклонение величины от ее математического ожидания.

Среднеквадратическое отклонение – показатель рассеивания значений случайной величины относительно ее математического ожидания.

Мода – значение во множестве наблюдений, которое встречается наиболее часто.

Медиана – возможное значение признака, которое делит вариационный ряд выборки на две равные части.

Коэффициент асимметрии используется для проверки распределения на симметричность, а также для грубой предварительной проверки на нормальность.

Если плотность распределения симметрична, то выборочный коэффициент асимметрии равен нулю, если левый хвост распределения тяжелее – больше нуля, легче – меньше.

Коэффициент эксцесса используется для проверки на нормальность.

Нормальное распределение имеет нулевой эксцесс. Если хвосты распределения «легче», а пик острее, чем у нормального распределения, то коэффициент эксцесса положительный; если хвосты распределения

«тяжелее», пик «приплюснутый», чем у нормального распределения, то отрицательный.

**Математическое ожидание**:

**Дисперсия с поправкой Шеппарда**:

**Среднее квадратическое отклонение**:

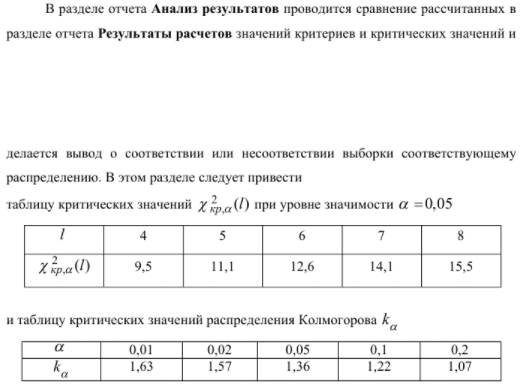
**Мода**:

**Медиана**:

**Выборочный момент k-го порядка**:

**Коэффициент асимметрии**:

**Коэффициент эксцесса**:



# **Средства языка программирования**

Для расчёта статистических исследований я использую язык Python. В программе расчёта используются следующие библиотеки:

**NumPy** — это библиотека языка Python, добавляющая поддержку  
больших многомерных массивов и матриц, вместе с большой библиотекой  
высокоуровневых (и очень быстрых) математических функций для операций  
с этими массивами.

**Pandas –** это библиотека для обработки и анализа данных Работа pandas с данными строится поверх библиотеки NumPy, являющейся инструментом более низкого уровня. Предоставляет специальные структуры данных и операции для манипулирования числовыми таблицами и временны́ми рядами.

**Matplotlib –** это библиотека для визуализации данных. Построение графиков диаграмм и гистограмм.

**Scipy.stats –** этот модуль содержит большое количество вероятностных распределений, а также растущую библиотеку статистических функций.

Стоит описать некоторые команды для генерации и визуализации данных:

**sps.norm/expon/uniform.cdf(x, loc, scale)** – Функция распределения случайной непрерывной величины. Позволяет узнать значение функции в точке.;

**sps.norm/expon/uniform.pdf(x, loc, scale)** – Плотность распределения случайной непрерывной величины. Позволяет узнать значение плотности в точке;

**pd.DataFrame –** визуализация данных в виде таблицы. Используется для визуализации статистического ряда.

# **Результаты расчётов**

## **Задание 3-1**

### Выборка из файла MC\_D\_Binom.pdf:

### Сортированная выборка

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| 4 | 4 | 4 | 4 | 4 | 4 | 4 | 5 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 7 |
| 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 7 | 7 | 7 | 7 | 8 | 8 | 8 | 8 | 8 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 8 | 8 | 8 | 8 | 9 | 9 | 9 | 9 | 9 | 9 |

Статистический ряд:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 3 | 3 | 0,015 |
| 4 | 14 | 0,170 |
| 5 | 35 | 0,175 |
| 6 | 47 | 0,235 |
| 7 | 65 | 0,325 |
| 8 | 30 | 0,150 |
| 9 | 6 | 0,030 |
|  | 200 | 1.0 |

**Оценка**

### Полигон относительных частот и график полигона соответствующих теоритических вероятностей

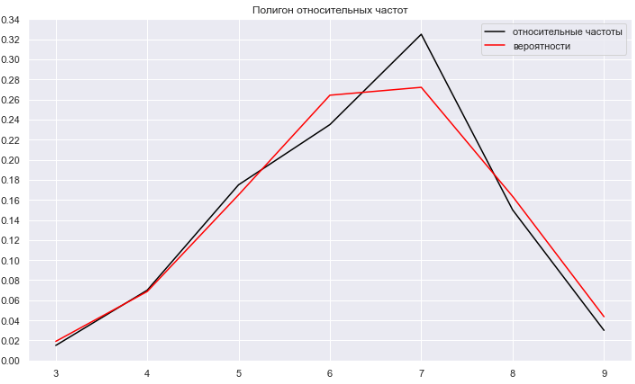


Таблица 1.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Xi | Wi | Pi | |Wi-Pi| |  |
| 3 | 0.015 | 0.01905 | 0.00405 | 0.17220 |
| 4 | 0.070 | 0.06867 | 0.00133 | 0.00515 |
| 5 | 0.175 | 0.16499 | 0.01001 | 0.12146 |
| 6 | 0.235 | 0.26428 | 0.02928 | 0.64880 |
| 7 | 0.325 | 0.27213 | 0.05287 | 2.05434 |
| 8 | 0.150 | 0.16346 | 0.01346 | 0.22167 |
| 9 | 0.030 | 0.04364 | 0.01364 | 0.85266 |
|  | 1.0 | 0.99622 | 0.05287 | 4.07628 |

**Оценка**

**Выборочное значение**

**Число степеней свободы: l** = 5

## **Задание 3-2**

### Выборка из файла MC\_D\_Exp.pdf:

### Сортированная выборка:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0.0004 | 0.00081 | 0.001 | 0.00174 | 0.00689 | 0.00755 | 0.01015 | 0.01631 | 0.02046 | 0.02206 |
| 0.02259 | 0.0309 | 0.04024 | 0.04749 | 0.05298 | 0.05637 | 0.05806 | 0.06479 | 0.076 | 0.07721 |
| 0.08229 | 0.08482 | 0.08694 | 0.09046 | 0.09811 | 0.09979 | 0.10912 | 0.13559 | 0.13592 | 0.13929 |
| 0.13962 | 0.14207 | 0.14308 | 0.14453 | 0.14595 | 0.14632 | 0.15558 | 0.15739 | 0.16015 | 0.17406 |
| 0.17531 | 0.17917 | 0.18265 | 0.18711 | 0.19192 | 0.1949 | 0.19552 | 0.20326 | 0.20652 | 0.20676 |
| 0.21235 | 0.21642 | 0.23564 | 0.24792 | 0.25107 | 0.25663 | 0.26168 | 0.26311 | 0.26468 | 0.27342 |
| 0.27799 | 0.27818 | 0.27854 | 0.28394 | 0.29945 | 0.30559 | 0.31931 | 0.33859 | 0.33862 | 0.34066 |
| 0.34861 | 0.37297 | 0.38188 | 0.40903 | 0.41095 | 0.41253 | 0.41267 | 0.41517 | 0.42116 | 0.42906 |
| 0.43331 | 0.43443 | 0.43471 | 0.43739 | 0.44408 | 0.44665 | 0.4485 | 0.45171 | 0.46023 | 0.46034 |
| 0.46522 | 0.46807 | 0.47997 | 0.48073 | 0.49628 | 0.51029 | 0.5122 | 0.52083 | 0.52571 | 0.52992 |
| 0.54301 | 0.54428 | 0.5561 | 0.56765 | 0.57252 | 0.59787 | 0.63074 | 0.65251 | 0.65431 | 0.66105 |
| 0.66364 | 0.6761 | 0.67741 | 0.69063 | 0.69219 | 0.6931 | 0.69646 | 0.71303 | 0.72536 | 0.7309 |
| 0.74501 | 0.76509 | 0.78006 | 0.81523 | 0.82593 | 0.83528 | 0.84122 | 0.84819 | 0.84825 | 0.85295 |
| 0.8717 | 0.89172 | 0.91889 | 0.92245 | 0.9272 | 0.92739 | 0.92776 | 0.94765 | 1.05088 | 1.06502 |
| 1.06742 | 1.09893 | 1.11188 | 1.12378 | 1.12611 | 1.17596 | 1.18896 | 1.19374 | 1.23128 | 1.23387 |
| 1.27399 | 1.28525 | 1.28583 | 1.31295 | 1.34678 | 1.35348 | 1.36475 | 1.3741 | 1.38866 | 1.40109 |
| 1.4026 | 1.40444 | 1.44526 | 1.46321 | 1.49746 | 1.50322 | 1.51726 | 1.52676 | 1.55681 | 1.57545 |
| 1.58386 | 1.60032 | 1.61129 | 1.62112 | 1.63059 | 1.69262 | 1.70317 | 1.71495 | 1.7599 | 1.76193 |
| 1.76315 | 1.76708 | 1.81542 | 1.8767 | 1.91352 | 1.91726 | 2.11917 | 2.12724 | 2.17253 | 2.17256 |
| 2.20956 | 2.2402 | 2.34382 | 2.43031 | 2.53646 | 2.60738 | 2.91082 | 2.94829 | 2.99722 | 3.17029 |

### Таблица 2.1 Группированная выборка

|  |  |  |
| --- | --- | --- |
| Интервалы |  |  |
| [0, 0.39664] | 73 | 0.455 |
| (0.39664, 0.79287] | 50 | 0.290 |
| (0.79287, 1.18911] | 24 | 0.100 |
| (1.18911, 1.58535] | 24 | 0.075 |
| (1.58535, 1.98158] | 15 | 0.055 |
| (1.98158, 2.37782] | 7 | 0.005 |
| (2.37782, 2.77405] | 3 | 0.005 |
| (2.77405, 3.17029] | 4 | 0.015 |
|  | 200 | 1.0 |

**Оценка**

### Таблица 2.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | *ak* | *f(ak, )* | *F(ak, )* |  |
| 0 | 0 | 1.27224 | 0 | 0 |
| 1 | 0.39664 | 0.76809 | 0.39627 | 0.39627 |
| 2 | 0.79287 | 0.46397 | 0.63531 | 0.29905 |
| 3 | 1.18911 | 0.28026 | 0.77971 | 0.14440 |
| 4 | 1.58535 | 0.16929 | 0.86694 | 0.08722 |
| 5 | 1.98158 | 0.10226 | 0.91962 | 0.05269 |
| 6 | 2.37782 | 0.06177 | 0.95145 | 0.03183 |
| 7 | 2.77405 | 0.03791 | 0.97067 | 0.02503 |
| 8 | 3.17029 | 0.02254 | 0.98229 | 0.02352 |
|  |  |  |  | 1.0 |

### График плотности показательного распределения, наложенный на гистограмму относительных частот

### Таблица 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| k | Интервал | *Wk* |  | |Wi-Pi| |  |
| 1 | [0, 0.39664] | 0.455 | 0.39627 | 0.03127 | 0.49336 |
| 2 | (0.39664, 0.79287] | 0.290 | 0.29905 | 0.01095 | 0.10034 |
| 3 | (0.79287, 1.18911] | 0.100 | 0.14440 | 0.02440 | 0.82461 |
| 4 | (1.18911, 1.58535] | 0.075 | 0.08722 | 0.03278 | 2.46327 |
| 5 | (1.58535, 1.98158] | 0.055 | 0.05269 | 0.02231 | 1.89016 |
| 6 | (1.98158, 2.37782] | 0.005 | 0.03183 | 0.00317 | 0.06333 |
| 7 | (2.37782, 2.77405] | 0.005 | 0.01922 | 0.00422 | 0.18559 |
| 8 | (2.77405, 3.17029] | 0.015 | 0.01772 | 0.00229 | 0.05897 |
|  |  | 1.0 | 0.98839 | 0.03878 | 6.07963 |

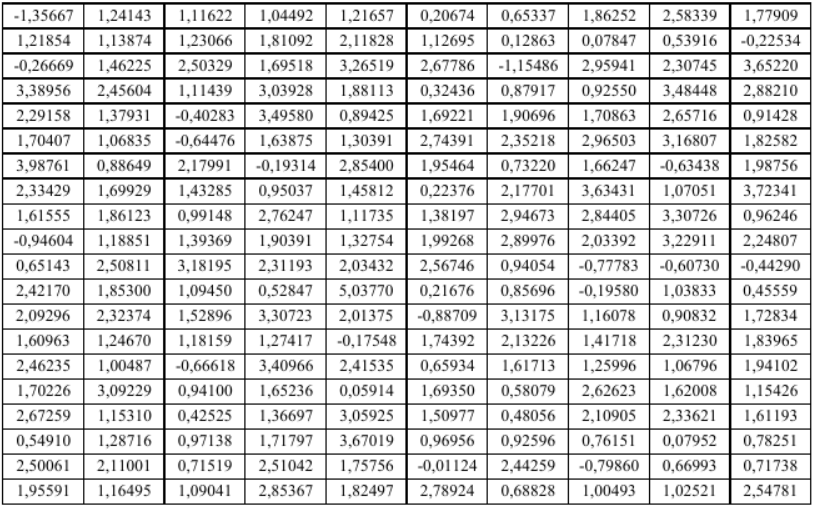
**Оценка**

**Выборочное значение**

**Число степеней свободы: l** = 6

## **Задание 3-3**

Выборка из файла MC\_D\_Norm.pdf



### Сортированная выборка:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -1.35667 | -1.15486 | -0.94604 | -0.88709 | -0.7986 | -0.77783 | -0.66618 | -0.64476 | -0.63438 | -0.6073 |
| -0.4429 | -0.40283 | -0.26669 | -0.22534 | -0.1958 | -0.19314 | -0.17548 | -0.01124 | 0.05914 | 0.07847 |
| 0.07952 | 0.12863 | 0.20674 | 0.21676 | 0.22376 | 0.32436 | 0.42525 | 0.45559 | 0.48056 | 0.52847 |
| 0.53916 | 0.5491 | 0.58079 | 0.65143 | 0.65337 | 0.65934 | 0.66993 | 0.68828 | 0.71519 | 0.71738 |
| 0.7322 | 0.76151 | 0.78251 | 0.85696 | 0.87917 | 0.88649 | 0.89425 | 0.90832 | 0.91428 | 0.9255 |
| 0.92596 | 0.94054 | 0.941 | 0.95037 | 0.96246 | 0.96956 | 0.97138 | 0.99148 | 1.00487 | 1.00493 |
| 1.02521 | 1.03833 | 1.04492 | 1.06796 | 1.06835 | 1.07051 | 1.09041 | 1.0945 | 1.11439 | 1.11622 |
| 1.11735 | 1.12695 | 1.13874 | 1.1531 | 1.15426 | 1.16078 | 1.16495 | 1.18159 | 1.18851 | 1.21657 |
| 1.21854 | 1.23066 | 1.24143 | 1.2467 | 1.25996 | 1.27417 | 1.28716 | 1.30391 | 1.32754 | 1.36697 |
| 1.37931 | 1.38197 | 1.39369 | 1.41718 | 1.43285 | 1.45812 | 1.46225 | 1.50977 | 1.52896 | 1.60963 |
| 1.61193 | 1.61555 | 1.61713 | 1.62008 | 1.63875 | 1.65236 | 1.66247 | 1.69221 | 1.6935 | 1.69518 |
| 1.69929 | 1.70226 | 1.70407 | 1.70863 | 1.71797 | 1.72834 | 1.74392 | 1.75756 | 1.77909 | 1.81092 |
| 1.82497 | 1.82582 | 1.83965 | 1.853 | 1.86123 | 1.86252 | 1.88113 | 1.90391 | 1.90696 | 1.94102 |
| 1.95464 | 1.95591 | 1.98756 | 1.99268 | 2.01375 | 2.03392 | 2.03432 | 2.09296 | 2.10905 | 2.11001 |
| 2.11828 | 2.13226 | 2.17701 | 2.17991 | 2.24807 | 2.29158 | 2.30745 | 2.31193 | 2.3123 | 2.32374 |
| 2.33429 | 2.33621 | 2.35218 | 2.41535 | 2.4217 | 2.44259 | 2.45604 | 2.46235 | 2.50061 | 2.50329 |
| 2.50811 | 2.51042 | 2.54781 | 2.56746 | 2.58339 | 2.62623 | 2.65716 | 2.67259 | 2.67786 | 2.74391 |
| 2.76247 | 2.78924 | 2.84405 | 2.85367 | 2.854 | 2.8821 | 2.89976 | 2.94673 | 2.95941 | 2.96503 |
| 3.03928 | 3.05925 | 3.09229 | 3.13175 | 3.16807 | 3.18195 | 3.22911 | 3.26519 | 3.30723 | 3.30726 |
| 3.38956 | 3.40966 | 3.48448 | 3.4958 | 3.63431 | 3.6522 | 3.67019 | 3.72341 | 3.98761 | 5.0377 |

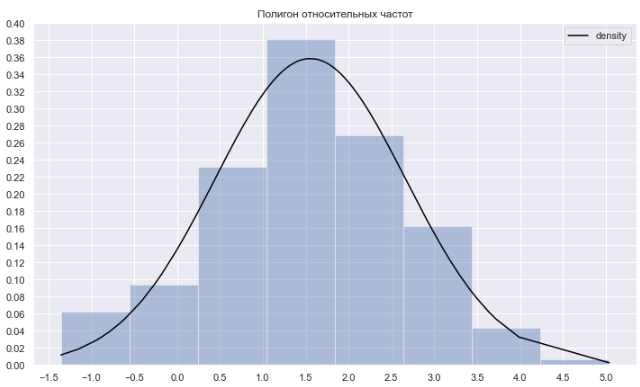
### Таблица 3.1 Группированная выборка

|  |  |  |
| --- | --- | --- |
| Интервалы |  |  |
| [-1.35667, -0.55737] | 10 | 0.050 |
| (-0.55737, 0.24192] | 15 | 0.075 |
| (0.24192, 1.04122] | 37 | 0.185 |
| (1.04122, 1.84052] | 61 | 0.305 |
| (1.84052, 2.63981] | 43 | 0.215 |
| (2.63981, 3.43911] | 26 | 0.130 |
| (3.43911, 4.22840] | 7 | 0.035 |
| (4.22840, 5.03770] | 1 | 0.005 |
|  | 200 | 1.0 |

**Оценка**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| k | *ak* |  |  | *Ф()* |  |
| 0 | -1.35667 | -2.61675 | 0.01169 | 0.00444 | - |
| 1 | -0.55737 | -1.89803 | 0.05922 | 0.02885 | 0.02885 |
| 2 | 0.24192 | -1.17932 | 0.17896 | 0.11913 | 0.09029 |
| 3 | 1.04122 | -0.46060 | 0.32262 | 0.32254 | 0.20341 |
| 4 | 1.84052 | 0.25811 | 0.34697 | 0.60184 | 0.27930 |
| 5 | 2.63981 | 0.97682 | 0.22262 | 0.83567 | 0.23383 |
| 6 | 3.43911 | 1.69554 | 0.08521 | 0.95501 | 0.11934 |
| 7 | 4.22840 | 2.41425 | 0.01946 | 0.99212 | 0.03710 |
| 8 | 5.03770 | 3.13297 | 0.00265 | 0.99913 | 0.00788 |

### График плотности нормального распределения, наложенный на гистограмму относительных частот



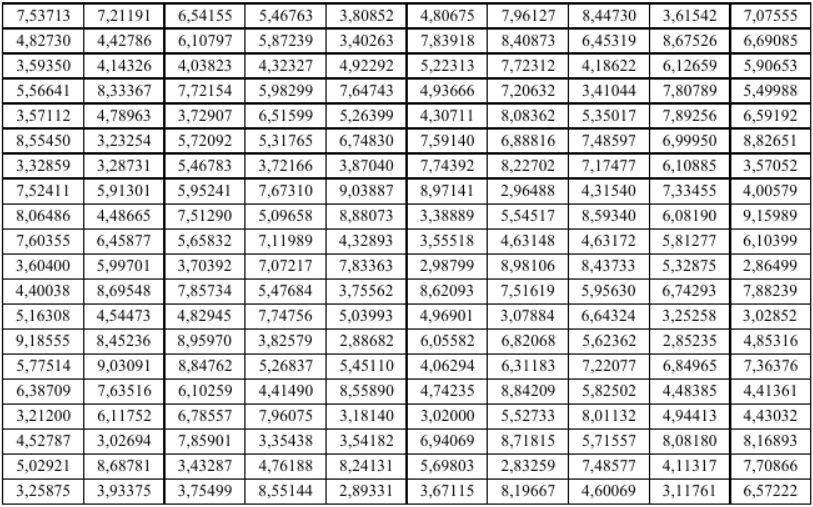
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| k | Интервал | *Wk* |  | |Wi-Pi| |  |
| 1 | [-1.35667, -0.55737] | 0.050 | 0.02885 | 0.05000 | 3.10102 |
| 2 | (-0.55737, 0.24192] | 0.075 | 0.09029 | 0.04615 | 0.51785 |
| 3 | (0.24192, 1.04122] | 0.185 | 0.20341 | 0.09471 | 0.33325 |
| 4 | (1.04122, 1.84052] | 0.305 | 0.27930 | 0.10159 | 0.47296 |
| 5 | (1.84052, 2.63981] | 0.215 | 0.23383 | 0.06430 | 0.30327 |
| 6 | (2.63981, 3.43911] | 0.130 | 0.11934 | 0.10383 | 0.19044 |
| 7 | (3.43911, 4.22840] | 0.035 | 0.03710 | 0.08434 | 0.02377 |
| 8 | (4.22840, 5.03770] | 0.005 | 0.00788 | 0.03210 | 0.21052 |
|  |  | 1.0 | 1.0 | 0.10383 | 5.15308 |

**Оценка**

**Выборочное значение**

## **Число степеней свободы: l** = 5

## **Задание 3-4**

Выборка из файла MC\_D\_Unif.pdf

### Сортированная выборка:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2.83259 | 2.85235 | 2.86499 | 2.88682 | 2.89331 | 2.96488 | 2.98799 | 3.02 | 3.02694 | 3.02852 |
| 3.07884 | 3.11761 | 3.1814 | 3.212 | 3.23254 | 3.25258 | 3.25875 | 3.28731 | 3.32859 | 3.35438 |
| 3.38889 | 3.40263 | 3.41044 | 3.43287 | 3.54182 | 3.55518 | 3.57052 | 3.57112 | 3.5935 | 3.604 |
| 3.61542 | 3.67115 | 3.70392 | 3.72166 | 3.72907 | 3.75499 | 3.75562 | 3.80852 | 3.82579 | 3.8704 |
| 3.93375 | 4.00579 | 4.03823 | 4.06294 | 4.11317 | 4.14326 | 4.18622 | 4.30711 | 4.3154 | 4.32327 |
| 4.32893 | 4.40038 | 4.41361 | 4.4149 | 4.42786 | 4.43032 | 4.48385 | 4.48665 | 4.52787 | 4.54473 |
| 4.60069 | 4.63148 | 4.63172 | 4.74235 | 4.76188 | 4.78963 | 4.80675 | 4.8273 | 4.82945 | 4.85316 |
| 4.92292 | 4.93666 | 4.94413 | 4.96901 | 5.02921 | 5.03993 | 5.09658 | 5.16308 | 5.22313 | 5.26399 |
| 5.26837 | 5.31765 | 5.32875 | 5.35017 | 5.4511 | 5.46763 | 5.46783 | 5.47684 | 5.49988 | 5.52733 |
| 5.54517 | 5.56641 | 5.62362 | 5.65832 | 5.69803 | 5.71557 | 5.72092 | 5.77514 | 5.81277 | 5.82502 |
| 5.87239 | 5.90653 | 5.91301 | 5.95241 | 5.9563 | 5.98299 | 5.99701 | 6.05582 | 6.0819 | 6.10259 |
| 6.10399 | 6.10797 | 6.10885 | 6.11752 | 6.12659 | 6.31183 | 6.38709 | 6.45319 | 6.45877 | 6.51599 |
| 6.54155 | 6.57222 | 6.59192 | 6.64324 | 6.69085 | 6.74293 | 6.7483 | 6.78557 | 6.82068 | 6.84965 |
| 6.88816 | 6.94069 | 6.9995 | 7.07217 | 7.07555 | 7.11989 | 7.17477 | 7.20632 | 7.21191 | 7.22077 |
| 7.33455 | 7.36376 | 7.48577 | 7.48597 | 7.5129 | 7.51619 | 7.52411 | 7.53713 | 7.5914 | 7.60355 |
| 7.63516 | 7.64743 | 7.6731 | 7.70866 | 7.72154 | 7.72312 | 7.74392 | 7.74756 | 7.80789 | 7.83363 |
| 7.83918 | 7.85734 | 7.85901 | 7.88239 | 7.89256 | 7.96075 | 7.96127 | 8.01132 | 8.06486 | 8.0818 |
| 8.08362 | 8.16893 | 8.19667 | 8.22702 | 8.24131 | 8.33367 | 8.40873 | 8.43733 | 8.4473 | 8.45236 |
| 8.55144 | 8.5545 | 8.5589 | 8.5934 | 8.62093 | 8.67526 | 8.68781 | 8.69548 | 8.71815 | 8.82651 |
| 8.84209 | 8.84762 | 8.88073 | 8.9597 | 8.97141 | 8.98106 | 9.03091 | 9.03887 | 9.15989 | 9.18555 |

### Таблица 4.1 Группированная выборка

|  |  |  |
| --- | --- | --- |
| Интервалы |  |  |
| [2.83259, 3.62672] | 31 | 0.155 |
| (3.62672, 4.42083] | 23 | 0.115 |
| (4.42083, 5.21495] | 24 | 0.120 |
| (5.21495, 6.00907] | 29 | 0.145 |
| (6.00907, 6.80319] | 21 | 0.105 |
| (6.80319, 7.59731] | 21 | 0.105 |
| (7.59731, 8.39143] | 27 | 0.135 |
| (8.39143, 9.18555] | 24 | 0.120 |
|  | 200 | 1.0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| k | Интервал | *Wk* |  | |Wi-Pi| |  |
| 1 | [2.83259, 3.62672] | 0.155 | 0.125 | 0.030 | 1.44 |
| 2 | (3.62672, 4.42083] | 0.115 | 0.125 | 0.010 | 0.16 |
| 3 | (4.42083, 5.21495] | 0.120 | 0.125 | 0.005 | 0.04 |
| 4 | (5.21495, 6.00907] | 0.145 | 0.125 | 0.020 | 0.64 |
| 5 | (6.00907, 6.80319] | 0.105 | 0.125 | 0.020 | 0.64 |
| 6 | (6.80319, 7.59731] | 0.105 | 0.125 | 0.020 | 0.64 |
| 7 | (7.59731, 8.39143] | 0.135 | 0.125 | 0.010 | 0.16 |
| 8 | (8.39143, 9.18555] | 0.120 | 0.125 | 0.005 | 0.04 |
|  |  | 1.0 | 1.0 | 0.03 | 3.76 |

### График плотности равномерного распределения, наложенный на гистограмму относительных частот

**Выборочное значение**

## **Число степеней свободы: l** = 7

**Задание 3-5**

### График Эмпирической функции распределения FN(x) и график функции распределения F(x)

Таблица 5.1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *a* | *b* | *N* | *DN* | *DN* |  |  |  |  |
| 2.90 | 9.22 | 200 | 0.05684 | 0.80383 | 6.12659 | 0.51816 | 0.575 | 0.57 |

**Анализ результатов**

Таблица критических значений при уровне значимости

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *l* | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|  | 3,8 | 6,0 | 7,8 | 9,5 | 11,1 | 12,6 | 14,1 | 15,5 |

Таблица критических значений критерия Колмогорова

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0.01 | 0.02 | 0.05 | 0.1 | 0.2 |
|  | 1.63 | 1.57 | 1.36 | 1.22 | 1.07 |

**Проверка гипотез с помощью критерия**

**Биномиальное распределение**

Так как , то гипотеза о соответствии выборки биномиальному распределению **не противоречит** экспериментальным данным при уровне значимости =0.05.

**Показательное распределение**

Так как , то гипотеза о соответствии выборки показательному распределению **не противоречит** экспериментальным данным при уровне значимости =0.05.

**Нормальное распределение**

Так как , то гипотеза о соответствии выборки нормальному распределению **не противоречит** экспериментальным данным при уровне значимости =0.05.

**Равномерное распределение**

Так как , то гипотеза о соответствии выборки равномерному распределению на отрезке [2.80, 9.22] **не противоречит** экспериментальным данным при уровне значимости =0.05.

**Проверка гипотез с помощью критерия Колмогорова**

**Равномерное распределение**

*DN*

Так как , то гипотеза о соответствии выборки равномерному распределению на отрезке [2.80, 9.22] **не противоречит** экспериментальным данным при уровне значимости =0.05.

# **Список литературы**

1. Математическая статистика [Электронный ресурс]: метод. указания  
   по выполнению лаб. работ / А.А. Лобузов – М.: МИРЭА, 2017.
2. Гмурман В.Е. Теория вероятностей и математическая статистика. –  
   М.:Юрайт, 2020.
3. Ивченко Г.И., Медведев Ю.И. Математическая статистика. – М.: URSS, 2020.

# **Приложение**

1. **import** scipy.stats **as** sps
2. **import** numpy **as** np
3. **import** pandas **as** pd
4. **import** matplotlib.pyplot **as** plt
5. **import pylab**
6. **import** statistics
7. **import** cmath
8. **import** math
9. **import** seaborn **as** sns
10. **from matplotlib import ticker**
11. **from** math **import** factorial
12. **from** math **import** exp
13. **from** math **import** log
14. **from** math **import** inf
15. **from scipy import integrate**
17. **def** Xi(distr):
18. xi = []
19. **for** item **in** distr:
20. **if item not in xi:**
21. xi.append(item)
23. **return**(xi)
25. **def Freq(distr, xi, size):**
26. ni=[]
27. **for** j **in** range(len(xi)):
28. help = 0
29. **for** i **in** range(size):
30. **if distr[i]==xi[j]:**
31. help += 1
32. ni.append(help)
33. **return**(ni)
35. **def Rel\_freq(ni, size):**
36. wi = []
37. **for** i **in** range(len(ni)):
38. wi.append(ni[i]/size)
39. **return**(wi)
41. **def** Mean\_dis(xi ,wi):
42. X = 0
43. **for** i **in** range (len(xi)):
44. help = xi[i]\*wi[i]
45. **X += help**
46. **return** (X)
48. **def** Moment(xi,wi):
49. M = []
50. **for i in range (len(xi)):**
51. help = 0
52. **for** j **in** range (len(xi)):
53. help += pow(xi[j],i+1)\*wi[j]
54. M.append(help)
55. **return (M)**

58. **def** chi\_square(wi, pi, size):
59. chi = [i **for** i **in** range(len(wi))]
60. **for k in range (len(wi)):**
61. chi[k] = round((size\*(wi[k]-pi[k])\*\*2)/pi[k],5)
62. **return** (chi)
64. **def** Disp(Moment):
65. **D = Moment[1]- pow(Moment[0], 2)**
66. **return** (D)
68. **def** intervals(distr, size):
69. m = 1+int(log(size,2))
70. **d = distr[-1] - distr[0]**
71. a = [distr[0]]
72. **for** i **in** range (1,(m+1)):
73. a.append(d/m+a[i-1])
74. **return** a
76. **def** Freq\_inter(distr, inter, size):
77. ni=[]
78. **for** j **in** range(1,len(inter)):
79. count = 0
80. **h=0**
81. **for** i **in** range(size):
82. **if** j-1 == 0:
83. **if** distr[i]>=inter[j-1] **and** distr[i]<=inter[j]:
84. count +=1
85. **elif j == len(inter)-1:**
86. **if** distr[i]>inter[j-1] **and** distr[i]<=inter[j]:
87. count += 1
88. **if** distr[i]>inter[j]:
89. count += 1
90. **else:**
91. **if** distr[i]>inter[j-1] **and** distr[i]<=inter[j]:
92. count += 1
93. ni.append(count)
94. **return**(ni)
96. **def** Mid\_inter(inter):
97. mid = []
98. **for** i **in** range(len(inter)-1):
99. mid.append((inter[i]+inter[i+1])/2)
100. **return(mid)**
102. **def** read\_distr(files):
103. **with** open(files) **as** file:
104. array = [row.strip() **for** row **in** file]
105. **return (array)**
107. **def** Round\_array(array):
108. **for** i **in** range (len(array)):
109. array[i] = round(array[i],5)
110. **return array**
112. X\_kr = [0,0,0,0, 9.5, 11.1, 12.6, 14.1, 15.5]
114. sns.set(rc={'figure.figsize':(12,7)})
116. v3\_1 = 46
117. n3\_1 = 9
118. size = 200
119. distr3\_1 = read\_distr("binom.txt")
120. **for i, item in enumerate(distr3\_1):**
121. distr3\_1[i] = int(item)
122. distr3\_1.sort()
123. **print**(distr3\_1)
125. **df\_3\_1 = pd.DataFrame({'distr3\_1':distr3\_1})**
127. xi3\_1 = []
128. ni3\_1 = []
129. wi3\_1 = []
130. ***#xi - значение***
131. *#ni - кол-во значения*
132. *#sum(ni) = size*
133. *#wi = ni/size*
135. **xi3\_1 = Xi(distr3\_1)**
136. ni3\_1 = Freq(distr3\_1, xi3\_1, size)
137. wi3\_1 = Rel\_freq(ni3\_1, size)
139. Table3\_1\_1 = pd.DataFrame({'Xi':xi3\_1,
140. **'Ni':ni3\_1,**
141. 'Wi':wi3\_1})
142. Table3\_1\_1
144. *# вычисление теор вероятности*
145. **M3\_1 = Moment(xi3\_1,wi3\_1)**
146. pi3\_1 = [i **for** i **in** range(len(xi3\_1))]
147. p3\_1 = M3\_1[0]/n3\_1
149. **print**('p =', round(p3\_1, 5))

152. **for** k **in** range(len(xi3\_1)):
153. pi3\_1[k] = round((factorial(n3\_1)/(factorial(xi3\_1[k])\*factorial(n3\_1-xi3\_1[k])))\*
154. (p3\_1\*\*xi3\_1[k])\*(1-p3\_1)\*\*(n3\_1-xi3\_1[k]), 5)

157. *# Вычисление значения хи-квадрат*
158. chi\_square3\_1 = chi\_square(wi3\_1, pi3\_1, size)


162. *# Wi - Pi*
163. diff\_wp3\_1 = []
164. **for** i **in** range(len(wi3\_1)):
165. **diff\_wp3\_1.append(abs(wi3\_1[i]-pi3\_1[i]))**
167. Table3\_1\_2 = pd.DataFrame({'Xi':xi3\_1,
168. 'Wi':wi3\_1,
169. 'Pi':pi3\_1,
170. **'Wi-Pi':diff\_wp3\_1,**
171. 'chi\_square':chi\_square3\_1})
172. Table3\_1\_2
174. **print** ('Сумма wk:',round(sum(wi3\_1),5))
175. **print ('Сумма pk:',round(sum(pi3\_1),5))**
176. **print**('max |Wk-Pk|:',max(diff\_wp3\_1))
177. **print**('Хи квадрат =', sum(chi\_square3\_1))
179. **if** (sum(chi\_square3\_1)<= X\_kr[len(wi3\_1)-2]):
180. **print ('''гипотеза о соответствии выборки биномиальному распределению**
181. НЕ ПРОТИВОРЕЧИТ экспериментальным данным при уровне значимости 0,05''')
182. **else**:
183. **print** ('''гипотеза о соответствии выборки биномиальному распределению
184. ПРОТИВОРЕЧИТ экспериментальным данным при уровне значимости 0,05''')
186. fig, fir = plt.subplots()
187. fir.xaxis.set\_major\_locator(ticker.MultipleLocator(1))
188. fir.yaxis.set\_major\_locator(ticker.MultipleLocator(0.02))
189. pi3\_1 = [i **for** i **in** range(len(xi3\_1))]
191. **for** k **in** range(len(xi3\_1)):
192. pi3\_1[k] = (factorial(n3\_1)/(factorial(xi3\_1[k])\*factorial(n3\_1-
193. xi3\_1[k])))\*p3\_1\*\*xi3\_1[k]\*(1-p3\_1)\*\*(n3\_1-xi3\_1[k])
194. fir.plot(xi3\_1, wi3\_1, 'black',label='относительные частоты')
195. **fir.plot(xi3\_1, pi3\_1, 'red',label='вероятности')**
196. plt.title("Полигон относительных частот")
197. plt.grid(True)
198. plt.legend(loc='best')
200. **def Density3\_2(inter,lam):**
201. foo=[i **for** i **in** range(len(inter))]
202. **for** i **in** range (len(inter)):
203. foo[i] = lam\*exp(-lam\*inter[i])
204. **return**(foo)
206. **def** func3\_2(inter,lam):
207. func=[i **for** i **in** range(len(inter))]
208. **for** i **in** range (len(inter)):
209. func[i] = 1-exp(-lam\*inter[i])
210. **return(func)**
212. **def** Pi3\_2(func):
213. P=[i **for** i **in** range(len(func)-1)]
214. P[-1] = 1 - func[-1]
215. **for i in range(len(func)-2):**
216. P[i] = func[i+1] - func[i]
217. **return** P
219. v3\_2 = 46
220. **size = 200**
221. distr3\_2 = read\_distr("expon.txt")
223. *# делем из str - float*
224. **for** i **in** range(len(distr3\_2)):
225. **distr3\_2[i] = float(str(distr3\_2[i]).replace(",", "."))**
227. **for** i, item **in** enumerate(distr3\_2):
228. distr3\_2[i] = float(item)
230. **distr3\_2.sort()**
231. **print** (distr3\_2)
233. df\_3\_2 = pd.DataFrame({'distr3\_2':distr3\_2})
235. **xi3\_2 = []**
236. ni3\_2 = []
237. wi3\_2 = []
238. sk3\_2 = []
239. *#xi - значение*
240. ***#ni - кол-во значения***
241. *#sum(ni) = size*
242. *#wi = ni/size*
243. *#sk = sum(wj)*
245. **xi3\_2 = Xi(distr3\_2)**
246. ni3\_2 = Freq(distr3\_2, xi3\_2, size)
247. wi3\_2 = Rel\_freq(ni3\_2, size)

250. ***# Построение интервалов***
251. inter3\_2 = intervals(distr3\_2, size)
252. inter3\_2[0] = 0
254. interval\_all3\_2 = [[inter3\_2[0], inter3\_2[1]],
255. **[inter3\_2[1], inter3\_2[2]],**
256. [inter3\_2[2], inter3\_2[3]],
257. [inter3\_2[3], inter3\_2[4]],
258. [inter3\_2[4], inter3\_2[5]],
259. [inter3\_2[5], inter3\_2[6]],
260. **[inter3\_2[6], inter3\_2[7]],**
261. [inter3\_2[7], inter3\_2[8]],
262. ]
264. inter3\_2\_round = Round\_array(inter3\_2)
265. **interval\_all3\_2\_round = [**
266. [inter3\_2\_round[0], inter3\_2\_round[1]],
267. [inter3\_2\_round[1], inter3\_2\_round[2]],
268. [inter3\_2\_round[2], inter3\_2\_round[3]],
269. [inter3\_2\_round[3], inter3\_2\_round[4]],
270. **[inter3\_2\_round[4], inter3\_2\_round[5]],**
271. [inter3\_2\_round[5], inter3\_2\_round[6]],
272. [inter3\_2\_round[6], inter3\_2\_round[7]],
273. [inter3\_2\_round[7], inter3\_2\_round[8]],
274. ]
276. *# интервальная частота*
277. ni\_inter3\_2 = Freq\_inter(distr3\_2, inter3\_2, size)
279. *# интервальная отн частота*
280. **wi\_inter3\_2 = Rel\_freq(ni\_inter3\_2, size)**
282. *# центроиды интервалов*
283. x\_x3\_2 = Mid\_inter(inter3\_2)
285. **Table3\_2\_1 = pd.DataFrame({'Intervals':interval\_all3\_2\_round,**
286. 'Ni':ni\_inter3\_2,
287. 'Wi':wi\_inter3\_2})
288. Table3\_2\_1
290. ***# находим плотность и функцию***
291. f3\_2 = Density3\_2(inter3\_2, lam3\_2)
292. F3\_2 = func3\_2(inter3\_2, lam3\_2)
293. pi3\_2 = Pi3\_2(F3\_2)
295. **func3\_2\_1 = [i for i in range(len(inter3\_2))]**
296. func3\_2\_1[0] = 0
297. **for** i **in** range (1,len(inter3\_2)):
298. func3\_2\_1[i] = pi3\_2[i-1]
300. **Table3\_2\_2 = pd.DataFrame({'ak':inter3\_2,**
301. 'f(ak, lam)':f3\_2,
302. 'F(ak, lam)':F3\_2,
303. 'pi':func3\_2\_1})
304. Table3\_2\_2
306. diff\_wp3\_2 = []
307. **for** i **in** range(len(wi\_inter3\_2)):
308. diff\_wp3\_2.append(abs(wi\_inter3\_2[i]-pi3\_2[i]))
310. **chi\_square3\_2 = chi\_square(wi\_inter3\_2, pi3\_2, size)**
312. Table3\_2\_3 = pd.DataFrame({'Intervals':interval\_all3\_2\_round,
313. 'Wi':Round\_array(wi\_inter3\_2),
314. 'Pk':Round\_array(pi3\_2),
315. **'|Wk-Pk|':Round\_array(diff\_wp3\_2),**
316. 'X^2':chi\_square3\_2})
317. Table3\_2\_3
319. fig, fir = plt.subplots()
320. **fir.xaxis.set\_major\_locator(ticker.MultipleLocator(0.5))**
321. fir.yaxis.set\_major\_locator(ticker.MultipleLocator(0.05))
323. fir.plot(distr3\_2, dens3\_2, 'black',label='density')
324. sns.distplot(distr3\_2, bins = 8, norm\_hist=True, kde=False)
325. **plt.title("Полигон относительных частот")**
326. plt.grid(True)
327. plt.legend(loc='best')
329. **def** PHI(x):
330. **return (sps.norm.cdf(x))**
332. **def** phi(t):
333. **return** (sps.norm.pdf(t))
335. **def Pi3\_3(inter,a,std):**
336. P=[i **for** i **in** range(len(inter))]
337. P[0] = 0
338. P[1] = PHI((inter[1]-a)/std)
339. P[-1] = 1 - PHI((inter[-2]-a)/std)
340. **for i in range(2,len(inter)-1):**
341. P[i] = PHI((inter[i]-a)/std) - PHI((inter[i-1]-a)/std)
342. **return** P
344. distr3\_3 = read\_distr("norm.txt")
345. ***# делем из str - float***
346. **for** i **in** range(len(distr3\_3)):
347. distr3\_3[i] = float(str(distr3\_3[i]).replace(",", "."))
349. **for** i, item **in** enumerate(distr3\_3):
350. **distr3\_3[i] = float(item)**
352. distr3\_3.sort()
353. **print**(distr3\_3)
354. df3\_3 = pd.DataFrame({'df3\_3':distr3\_3})
356. inter3\_3 = intervals(distr3\_3, size)
358. interval\_all3\_3 = [
359. [inter3\_3[0], inter3\_3[1]],
360. **[inter3\_3[1], inter3\_3[2]],**
361. [inter3\_3[2], inter3\_3[3]],
362. [inter3\_3[3], inter3\_3[4]],
363. [inter3\_3[4], inter3\_3[5]],
364. [inter3\_3[5], inter3\_3[6]],
365. **[inter3\_3[6], inter3\_3[7]],**
366. [inter3\_3[7], inter3\_3[8]],
367. ]
369. inter3\_3\_round = Round\_array(inter3\_3)
370. **interval\_all3\_3\_round = [**
371. [inter3\_3\_round[0], inter3\_3\_round[1]],
372. [inter3\_3\_round[1], inter3\_3\_round[2]],
373. [inter3\_3\_round[2], inter3\_3\_round[3]],
374. [inter3\_3\_round[3], inter3\_3\_round[4]],
375. **[inter3\_3\_round[4], inter3\_3\_round[5]],**
376. [inter3\_3\_round[5], inter3\_3\_round[6]],
377. [inter3\_3\_round[6], inter3\_3\_round[7]],
378. [inter3\_3\_round[7], inter3\_3\_round[8]],
379. ]
381. *# интервальная частота*
382. ni\_inter3\_3 = Freq\_inter(distr3\_3, inter3\_3, size)
384. *# интервальная отн частота*
385. **wi\_inter3\_3 = Rel\_freq(ni\_inter3\_3, size)**
387. Table3\_3\_1 = pd.DataFrame({'Intervals':interval\_all3\_3\_round,
388. 'Ni':ni\_inter3\_3,
389. 'Wi':wi\_inter3\_3})
390. **Table3\_3\_1**
392. xi3\_3 = []
393. ni3\_3 = []
394. wi3\_3 = []
395. ***#xi - значение***
396. *#ni - кол-во значения*
397. *#wi = ni/size*
399. xi3\_3 = Xi(distr3\_3)
400. **ni3\_3 = Freq(distr3\_3, xi3\_3, size)**
401. wi3\_3 = Rel\_freq(ni3\_3, size)
403. *# вычисление теор вероятности*
404. M3\_3 = Moment(xi3\_3,wi3\_3)

407. a3\_3 = M3\_3[0]
408. **print**('Мат. ожидание: ', a3\_3)
409. *#дисперсия*
410. **var3\_3 = Disp(M3\_3)**
411. **print**('Дисперсия: ', var3\_3)
412. *#сред.квадратичное*
413. std3\_3 = var3\_3\*\*(1/2)
414. **print**('Ср.квадратичное: ',std3\_3)
416. *#(Ak-a)/q*
417. func3\_3\_1 = [i **for** i **in** range(len(inter3\_3))]
418. **for** i **in** range (len(inter3\_3)):
419. func3\_3\_1[i] = (inter3\_3[i]-a3\_3)/std3\_3
421. *#1/q\*phi*
422. func3\_3\_2 = [i **for** i **in** range(len(inter3\_3))]
423. **for** i **in** range (len(inter3\_3)):
424. func3\_3\_2[i] = (1/std3\_3)\*phi(func3\_3\_1[i])
426. *#PHI*
427. func3\_3\_3 = [i **for** i **in** range(len(inter3\_3))]
428. **for** i **in** range (len(inter3\_3)):
429. func3\_3\_3[i] = PHI(func3\_3\_1[i])
431. *#Pk*
432. pi3\_3 = Pi3\_3(inter3\_3,a3\_3,std3\_3)
433. func3\_3\_4 = [i **for** i **in** range(len(inter3\_3))]

436. Table3\_3\_2 = pd.DataFrame({'Ak':Round\_array(inter3\_3),
437. '(Ak-a)/q':Round\_array(func3\_3\_1),
438. '1/q\*phi':Round\_array(func3\_3\_2),
439. 'PHI':Round\_array(func3\_3\_3),
440. **'Pk':Round\_array(pi3\_3)})**
441. Table3\_3\_2
443. diff\_wp3\_3 = []
444. **for** i **in** range(len(wi\_inter3\_3)):
445. **diff\_wp3\_3.append(abs(wi\_inter3\_3[i]-pi3\_3[i]))**
447. func3\_3\_5 = [i **for** i **in** range(len(pi3\_3)-1)]
448. **for** i **in** range(len(pi3\_3)-1):
449. func3\_3\_5[i] = pi3\_3[i+1]
451. chi\_square3\_3 = chi\_square(wi\_inter3\_3, func3\_3\_5, size)
453. Table3\_3\_3 = pd.DataFrame({'Intervals':interval\_all3\_3\_round,
454. 'Wi':Round\_array(wi\_inter3\_3),
455. **'Pk':Round\_array(func3\_3\_5),**
456. '|Wk-Pk|':Round\_array(diff\_wp3\_3),
457. 'X^2':chi\_square3\_3})
458. Table3\_3\_3
460. **fig, fir = plt.subplots()**
461. fir.xaxis.set\_major\_locator(ticker.MultipleLocator(0.5))
462. fir.yaxis.set\_major\_locator(ticker.MultipleLocator(0.02))
464. fir.plot(distr3\_3, dens3\_3, 'black',label='density')
465. **sns.distplot(distr3\_3, bins = 8, norm\_hist=True, kde=False)**
466. plt.title("Полигон относительных частот")
467. plt.grid(True)
468. plt.legend(loc='best')
470. **def Pi3\_4(inter):**
471. P=[i **for** i **in** range(len(inter)-1)]
472. **for** i **in** range (len(inter)-1):
473. P[i] = 1/(len(inter)-1)
474. **return** P
476. **def** Density3\_4(x,a,b):
477. **if** (a<=x **and** x<=b):
478. **return** (1/(b-a))
479. **else**:
480. **return 0**
482. a = 2.80
483. b = 9.22
485. **distr3\_4 = read\_distr("unif.txt")**
486. *# делем из str - float*
487. **for** i **in** range(len(distr3\_4)):
488. distr3\_4[i] = float(str(distr3\_4[i]).replace(",", "."))
490. **for i, item in enumerate(distr3\_4):**
491. distr3\_4[i] = float(item)
493. distr3\_4.sort()
494. **print** (distr3\_4)
496. inter3\_4 = intervals(distr3\_4, size)
498. interval\_all3\_4 = [
499. [inter3\_4[0], inter3\_4[1]],
500. **[inter3\_4[1], inter3\_4[2]],**
501. [inter3\_4[2], inter3\_4[3]],
502. [inter3\_4[3], inter3\_4[4]],
503. [inter3\_4[4], inter3\_4[5]],
504. [inter3\_4[5], inter3\_4[6]],
505. **[inter3\_4[6], inter3\_4[7]],**
506. [inter3\_4[7], inter3\_4[8]],
507. ]
509. inter3\_4\_round = Round\_array(inter3\_4)
510. **interval\_all3\_4\_round = [**
511. [inter3\_4\_round[0], inter3\_4\_round[1]],
512. [inter3\_4\_round[1], inter3\_4\_round[2]],
513. [inter3\_4\_round[2], inter3\_4\_round[3]],
514. [inter3\_4\_round[3], inter3\_4\_round[4]],
515. **[inter3\_4\_round[4], inter3\_4\_round[5]],**
516. [inter3\_4\_round[5], inter3\_4\_round[6]],
517. [inter3\_4\_round[6], inter3\_4\_round[7]],
518. [inter3\_4\_round[7], inter3\_4\_round[8]],
519. ]
521. *# интервальная частота*
522. ni\_inter3\_4 = Freq\_inter(distr3\_4, inter3\_4, size)
524. *# интервальная отн частота*
525. **wi\_inter3\_4 = Rel\_freq(ni\_inter3\_4, size)**
527. Table3\_4\_1 = pd.DataFrame({'Intervals':interval\_all3\_4\_round,
528. 'Ni':ni\_inter3\_4,
529. 'Wi':wi\_inter3\_4})
530. **Table3\_4\_1**
532. pi3\_4 = Pi3\_4(inter3\_4)
534. diff\_wp3\_4 = []
535. **for i in range(len(wi\_inter3\_4)):**
536. diff\_wp3\_4.append(abs(wi\_inter3\_4[i]- pi3\_4[i]))
538. chi\_square3\_4 = chi\_square(wi\_inter3\_4, pi3\_4, size)
540. **Table3\_4\_2 = pd.DataFrame({'Intervals':interval\_all3\_4\_round,**
541. 'Wi':Round\_array(wi\_inter3\_4),
542. 'Pk':Round\_array(pi3\_4),
543. '|Wk-Pk|':Round\_array(diff\_wp3\_4),
544. 'X^2':chi\_square3\_4})
545. **Table3\_4\_2**
547. xi3\_4 = Xi(distr3\_4)
548. fig, fir = plt.subplots()
549. fir.xaxis.set\_major\_locator(ticker.MultipleLocator(0.5))
550. **fir.yaxis.set\_major\_locator(ticker.MultipleLocator(0.01))**
552. fir.plot(xi3\_4, dens3\_4, 'black',label='density')
553. sns.distplot(distr3\_4, bins = 8, norm\_hist=True, kde=False)
554. plt.title("Полигон относительных частот")
555. **plt.grid(True)**
556. plt.legend(loc='best')
558. distr3\_5 = distr3\_4
559. a = a
560. **b = b**
561. **print** (distr3\_5)
562. df3\_5 = pd.DataFrame({'data3\_5':distr3\_5})
563. df3\_5.data3\_5.round(5)[150:200]
565. **inter3\_5 = intervals(distr3\_5, size)**
567. interval\_all3\_5 = [
568. [inter3\_5[0], inter3\_5[1]],
569. [inter3\_5[1], inter3\_5[2]],
570. **[inter3\_5[2], inter3\_5[3]],**
571. [inter3\_5[3], inter3\_5[4]],
572. [inter3\_5[4], inter3\_5[5]],
573. [inter3\_5[5], inter3\_5[6]],
574. [inter3\_5[6], inter3\_5[7]],
575. **[inter3\_5[7], inter3\_5[8]],**
576. ]
578. inter3\_5\_round = Round\_array(inter3\_5)
579. interval\_all3\_5\_round = [
580. **[inter3\_5\_round[0], inter3\_5\_round[1]],**
581. [inter3\_5\_round[1], inter3\_5\_round[2]],
582. [inter3\_5\_round[2], inter3\_5\_round[3]],
583. [inter3\_5\_round[3], inter3\_5\_round[4]],
584. [inter3\_5\_round[4], inter3\_5\_round[5]],
585. **[inter3\_5\_round[5], inter3\_5\_round[6]],**
586. [inter3\_5\_round[6], inter3\_5\_round[7]],
587. [inter3\_5\_round[7], inter3\_5\_round[8]],
588. ]
590. **interval\_all3\_5**
592. **def** F\_N(size):
593. array = []
594. **for** i **in** range(size):
595. **array.append((i+1)/size)**
596. **return**(array)

599. **def** F\_N0(size):
600. **array = []**
601. **for** i **in** range(size):
602. array.append((i)/size)
603. **return**(array)

606. **def** F3\_5(x,a,b):
607. **if** (x<a):
608. **return** 0
609. **elif** (a<=x **and** x<=b):
610. **return (x-a)/(b-a)**
611. **else**:
612. **return** 1
614. **def** D\_N(funcN, func0, func):
615. **Max = []**
616. **for** i **in** range(200):
617. res1 = abs(funcN[i]-func[i])
618. res2 = abs(func0[i]-func[i])
619. **if** res1 > res2:
620. **Max.append(res1)**
621. **else**:
622. Max.append(res2)
623. **for** i **in** range(200):
624. **if** max(Max) == Max[i]:
625. **help1 = i**
626. **print** (help1)
627. **return** (max(Max))
629. func3\_5\_1 = F\_N(size)
630. **func3\_5\_2 = F\_N0(size)**
632. func3\_5\_3 = []
633. **for** el **in** distr3\_5:
634. func3\_5\_3.append(F3\_5(el, a, b))
636. **print**('x\*', distr3\_5[114])
637. **print**('f(x\*)', func3\_5\_3[114])
638. **print**('f\_N(x\*)', func3\_5\_1[114])
639. **print**('f\_N0(x\*)', func3\_5\_2[114])
641. **if** ((dn \* math.sqrt(size)) <= 1.36):
642. **print** ('''гипотеза о соответствии выборки биномиальному распределению
643. НЕ ПРОТИВОРЕЧИТ экспериментальным данным при уровне значимости 0,05''')
644. **else**:
645. **print ('''гипотеза о соответствии выборки биномиальному распределению**
646. ПРОТИВОРЕЧИТ экспериментальным данным при уровне значимости 0,05''')
648. xi3\_5 = Xi(distr3\_5)
649. fig, fir = plt.subplots()
650. **fir.xaxis.set\_major\_locator(ticker.MultipleLocator(0.2))**
651. fir.yaxis.set\_major\_locator(ticker.MultipleLocator(0.1))
653. fir.plot(xi3\_5, func3\_5\_1, 'black',label='F\_N')
654. fir.plot(xi3\_5, func3\_5\_3, 'red',label='F')
655. **plt.title("Полигон относительных частот")**
656. plt.grid(True)
657. plt.legend(loc='best')