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

ИНСТИТУТ КИБЕРНЕТИКИ

КАФЕДРА ВЫСШЕЙ МАТЕМАТИКИ

Лабораторная работа 1

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

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

Выполнил:
Студент 3-го курса
Маргулис А.П.

Группа: КМБО-01-17

МОСКВА 2020

Задание

Задание 1. Получить выборку, сгенерировав 100 псевдослучайных чисел распределенных по биномиальному закону с параметрами n и p .

Задание 2. Получить выборку, сгенерировав 100 псевдослучайных чисел распределенных по геометрическому закону с параметром p .

Задание 3. Получить выборку, сгенерировав 100 псевдослучайных чисел распределенных по закону Пуассона с параметром λ .

Следуя указаниям для всех выборок построить:

1) статистический ряд;

2) полигон относительных частот;

3) эмпирическую функцию распределения.

Найти:

1) выборочное среднее;

2) выборочную дисперсию;

3) выборочное среднее квадратическое отклонение;

4) моду;

5) медиану;

6) выборочный коэффициент асимметрии;

7) выборочный коэффициент эксцесса.

Все вычисления проводить с точностью до 0,00001 .

Краткие теоретические сведения

Биномиальное распределение:

- ряд распределения: $P(X = k) = C_n^k p^k q^{n-k}$
- математическое ожидание (среднее значение): *np*
- дисперсия: *npq*
- среднее квадратичное отклонение: $\sqrt{npq}$
- мода: *[(n+1)p]*, если *(n+1)p* – дробное; *(n+1)p–0,5* , если *(n+1)p* – целое
- медиана: *Round(np)*
- коэффициент асимметрии: $\frac{q-p}{\sqrt{npq}}$
- коэффициент эксцесса: $\frac{1-6pq}{npq}$

Геометрическое распределение:

- ряд распределения: $P(X = n) = q^n p$
- математическое ожидание (среднее значение): $\frac{q}{p}$, где $q = 1 - p$
- дисперсия: $\frac{q}{p^2}$, где $q = 1 - p$
- среднее квадратичное отклонение: $\sqrt{\frac{q}{p^2}}$
- мода: *0*
- медиана: $[\frac{-\ln 2}{\ln q}]$, если $\frac{\ln 2}{\ln q}$ – дробное; $\frac{-\ln 2}{\ln q} - \frac{1}{2}$, если $\frac{\ln 2}{\ln q}$ – целое
- коэффициент асимметрии: $\frac{2-p}{\sqrt{q}}$
- коэффициент эксцесса: $6 + \frac{p^2}{q}$

Распределение Пуассона:

- ряд распределения: $P(X = k) = \frac{\lambda^k}{k!} e^{-\lambda}$
- математическое ожидание (среднее значение): λ

- дисперсия: λ
- среднее квадратичное отклонение: $\sqrt{\lambda}$
- мода: $[\lambda]$
- медиана: $[\lambda + \frac{1}{3} - \frac{0.02}{\lambda}]$
- коэффициент асимметрии: $\frac{1}{\sqrt{\lambda}}$
- коэффициент эксцесса: $\frac{1}{\lambda}$

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

- **Эмпирическая функция распределения**

$$F_N^{Э}(x) = \sum_{x_i^* \leq x} w_i = \begin{cases} 0, & x < x_1^* \\ w_1, & x_1^* \leq x < x_2^* \\ w_1 + w_2, & x_2^* \leq x < x_3^* \\ w_1 + w_2 + w_3, & x_3^* \leq x < x_4^* \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots & \\ 1, & x \geq x_m^* \end{cases}$$

- **Выборочное среднее**

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

- **Выборочная дисперсия**

$$D_B = \sum_{i=1}^{m} (x_i^* - \bar{x})^2 w_i$$

- **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 квврадратическое отклонение**
- $\bar{\sigma} = \sqrt{D_B}$
-
- **Выборочная медиана**

$$\overline{M_e} = \begin{cases} x_i^*, & F_N^э(x_{i-1}^*) < 0{,}5 < F_N^э(x_i^*) \\ \frac{1}{2}(x_i^* + x_{i+1}^*), & F_N^э(x_i^*) = 0{,}5 \end{cases}$$

- **Выборочная мода** – это значение x_i, которому соответствует максимальная частота.

$$\begin{cases} \text{если } n_i = \max n_k > n_j, \mathrm{i} \neq \mathrm{j}, & \overline{M_0} = \{x_i^* | n_i = \max n_k\} \\ \text{если } n_i = n_i + 1 = \ldots = n_{i+j} = \max n_k, & \text{то } \overline{M_0} = \frac{1}{2}(x_i^* + x_{i+1}^*) \\ \text{если } n_i = n_j = \max n_k > n_l, i < \mathrm{l} < j, & \text{то } \overline{M_0} - \text{не существует} \end{cases}$$

- **Выборочный коэффициент асимметрии**

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

- **Выборочный коэффициент эксцесса**

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

Средства языка Python

В программе расчёта используются следующие средства языка Octave:

• функция binomial(n, p, size) - возвращает матрицу случайных значений из биномиального распределения с параметрами n и p , где n есть число испытаний, p – вероятность успеха, size – количество элементов.

• функция geometric(p, size) - возвращает матрицу случайных значений из геометрического распределения с параметром p, size – количество элементов.

• poisson(λ, size) - возвращает матрицу случайных значений из распределения Пуассона с параметром λ, size – количество элементов.

• sorted(x) - возвращает копию x с элементами, расположенными в порядке возрастания.

• Counter(x) – возвращает словарь, где ключи – это элементы, а значения – их количество.

• plot(x,y) - построение графика по координатам x,y

Результаты расчетов

Задание 1 (биномиальное распределение)

n = 20, p = 0,375.

Неупорядоченная выборка (200 чисел):

11	11	8	10	10	3	8	10	5	9
6	8	5	7	6	3	8	7	8	12
11	4	3	4	11	7	6	10	7	11
12	9	5	6	4	8	10	5	9	9
6	10	10	4	6	7	11	7	5	6
9	5	6	5	6	9	7	6	10	10
10	8	7	9	8	6	8	9	11	9
8	4	8	9	9	9	9	8	8	6
7	8	7	8	13	6	5	8	11	6
7	6	9	10	8	11	10	10	7	11
11	8	5	12	6	6	5	11	8	7
7	10	9	7	6	4	6	4	8	7
10	6	10	5	7	8	10	4	10	10
15	8	8	6	6	7	6	6	9	6
7	9	5	9	5	8	6	1	8	12
6	8	8	7	5	5	7	8	13	7
7	8	9	6	5	9	8	9	7	8
11	8	9	5	8	9	2	7	8	10
9	6	3	8	10	11	7	9	11	6
6	9	10	5	8	6	7	2	13	7

Упорядоченная выборка (200 чисел):

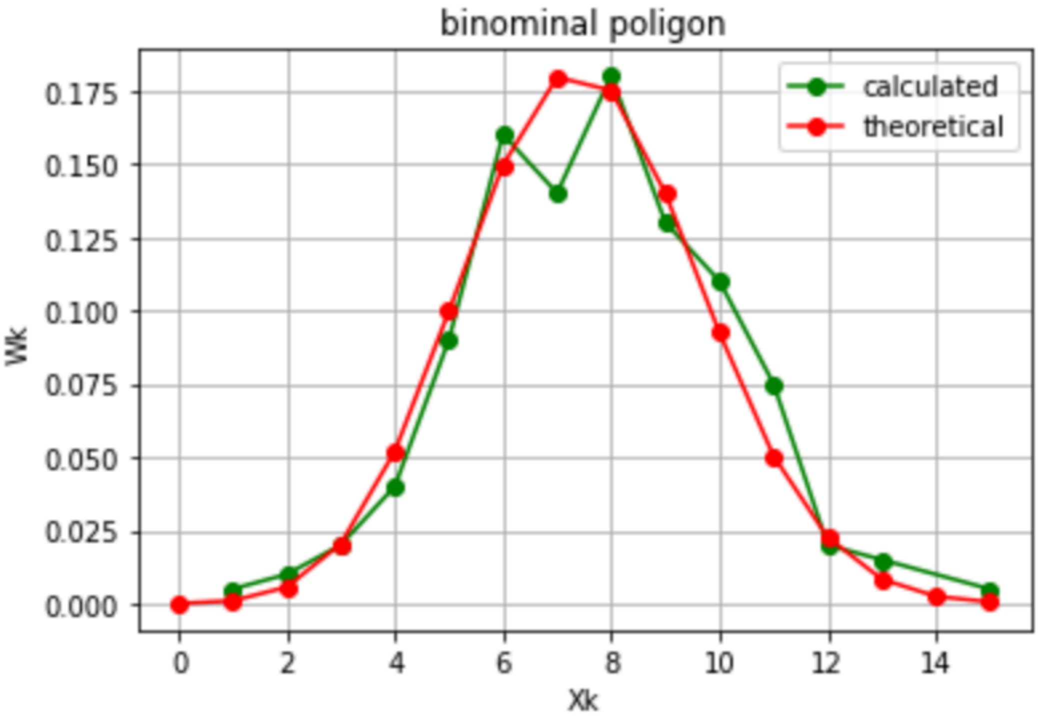
1	2	2	3	3	3	3	4	4	4
4	4	4	4	4	5	5	5	5	5
5	5	5	5	5	5	5	5	5	5
5	5	5	6	6	6	6	6	6	6
6	6	6	6	6	6	6	6	6	6
6	6	6	6	6	6	6	6	6	6
6	6	6	6	6	7	7	7	7	7
7	7	7	7	7	7	7	7	7	7
7	7	7	7	7	7	7	7	7	7
7	7	7	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8	8	9
9	9	9	9	9	9	9	9	9	9
9	9	9	9	9	9	9	9	9	9
9	9	9	9	9	10	10	10	10	10
10	10	10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	11	11	11
11	11	11	11	11	11	11	11	11	11
11	11	12	12	12	12	13	13	13	15

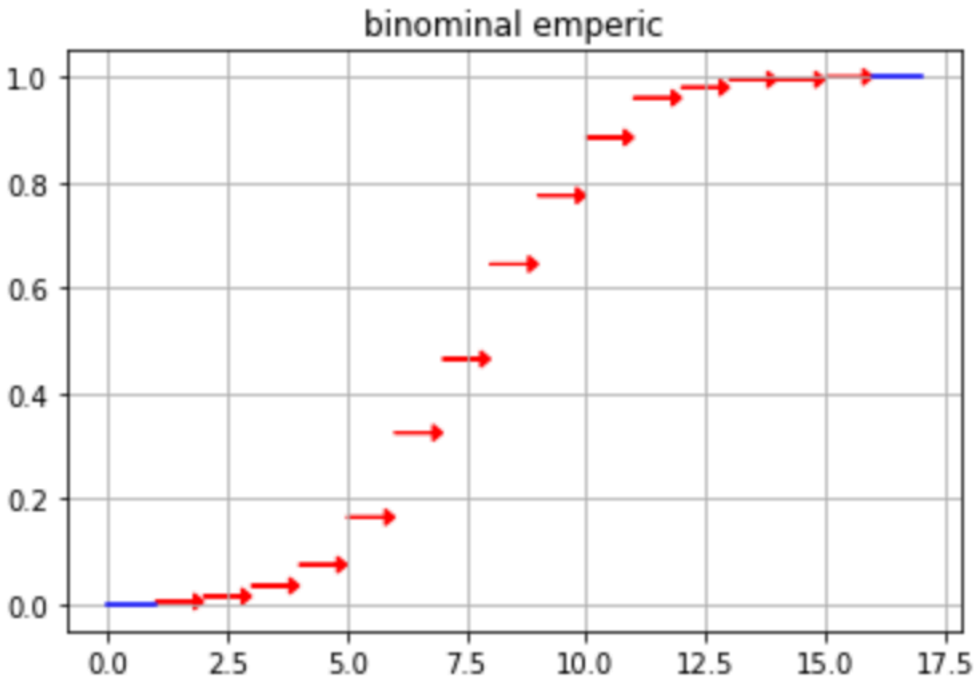
Статистический ряд:

Xk	Nk	Wk	Sk
1	1	0,005	0,005
2	2	0,01	0,015
3	4	0,02	0,035
4	8	0,04	0,075
5	18	0,09	0,165
6	32	0,16	0,325
7	28	0,14	0,465
8	36	0,18	0,645
9	26	0,13	0,775
10	22	0,11	0,885
11	15	0,075	0,96
12	4	0,02	0,98
13	3	0,015	0,995
15	1	0,005	1

Эмпирическая функция распределения и ее график:

$$F_{200}^{Э}(x) = \sum_{\mathrm{x_i} \leq x} w_i = \begin{cases} 0, & x < 1 \\ 0.005, & 1 \leq x < 2 \\ 0.035, & 2 \leq x < 3 \\ 0.075, & 3 \leq x < 4 \\ 0.165, & 4 \leq x < 5 \\ 0.325, & 5 \leq x < 6 \\ 0.465, & 6 \leq x < 7 \\ 0.645, & 7 \leq x < 8 \\ 0.775, & 8 \leq x < 9 \\ 0.885, & 9 \leq x < 10 \\ 0.96, & 11 \leq x < 12 \\ 0.98, & 12 \leq x < 13 \\ 0.995, & 13 \leq x < 15 \\ 1, & x \geq 15 \end{cases}$$





Выборочное среднее = 7.68
Выборочная дисперсия = 5.417600000000007
Выборочный момент 3-ого порядка = 578.1600000000001
Выборочный момент 4-ого порядка = 5496.320000000001
Выборочное среднее квадратическое отклонение = 2.3275738441561864
Выборочная мода = 8
Выборочная медиана = 8
Выборочный коэффициент асимметрии = 0.0022241805841271612
Выборочный коэффициент эксцесса = -2.896367624820266

Задание 2 (геометрическое распределение)

p = 0,375

Неупорядоченная выборка (200 чисел):

1	2	0	1	0	0	4	0	0	1
0	0	0	3	0	1	0	0	2	1
0	0	8	0	3	0	0	2	0	0
1	0	0	0	0	2	8	0	0	0
2	0	1	3	2	2	1	0	2	1
0	5	1	0	0	0	0	0	6	1
2	2	2	2	0	1	0	2	3	3
1	0	1	1	0	2	1	5	1	4
4	2	1	2	1	4	1	1	0	0
2	7	2	4	0	1	1	0	1	0
1	0	0	1	0	8	0	5	1	0
0	1	1	0	0	1	2	0	0	0
0	6	0	2	0	3	2	0	5	1
1	0	0	0	0	2	0	0	0	0
11	1	4	2	0	0	0	1	0	0
1	3	3	2	1	2	0	1	3	1
1	0	2	1	0	0	3	1	8	5
2	0	0	0	1	4	0	2	0	1
4	0	0	4	2	3	4	3	0	1
1	1	0	3	6	2	0	0	2	1

Упорядоченная выборка (200 чисел):

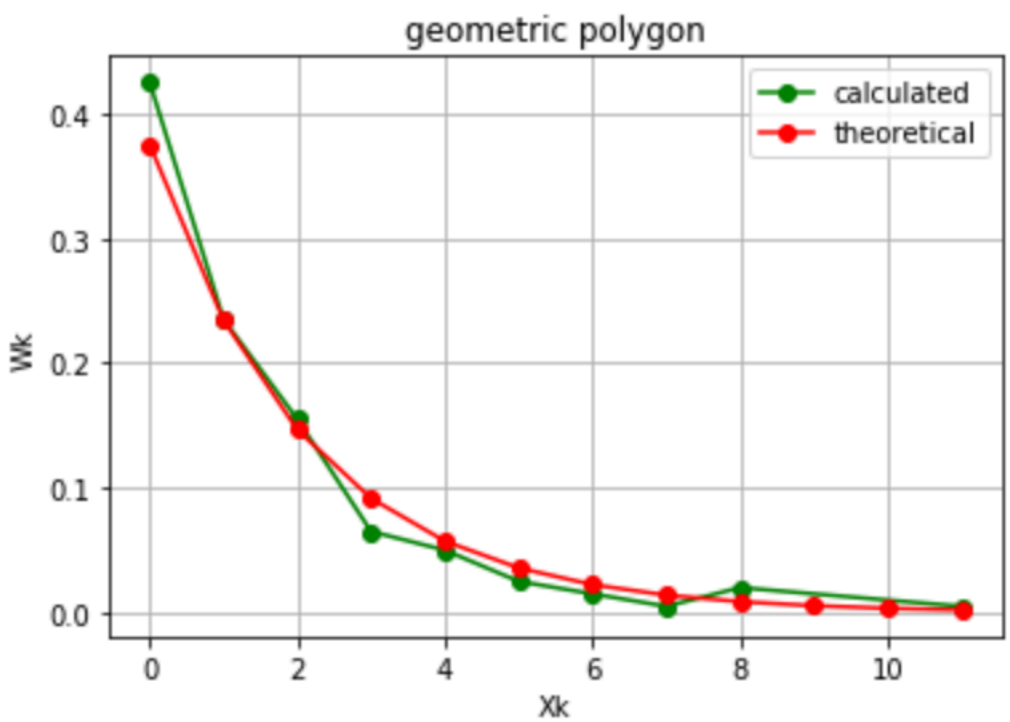
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2
2	2	2	3	3	3	3	3	3	3
3	3	3	3	3	3	4	4	4	4
4	4	4	4	4	4	5	5	5	5
5	6	6	6	7	8	8	8	8	11

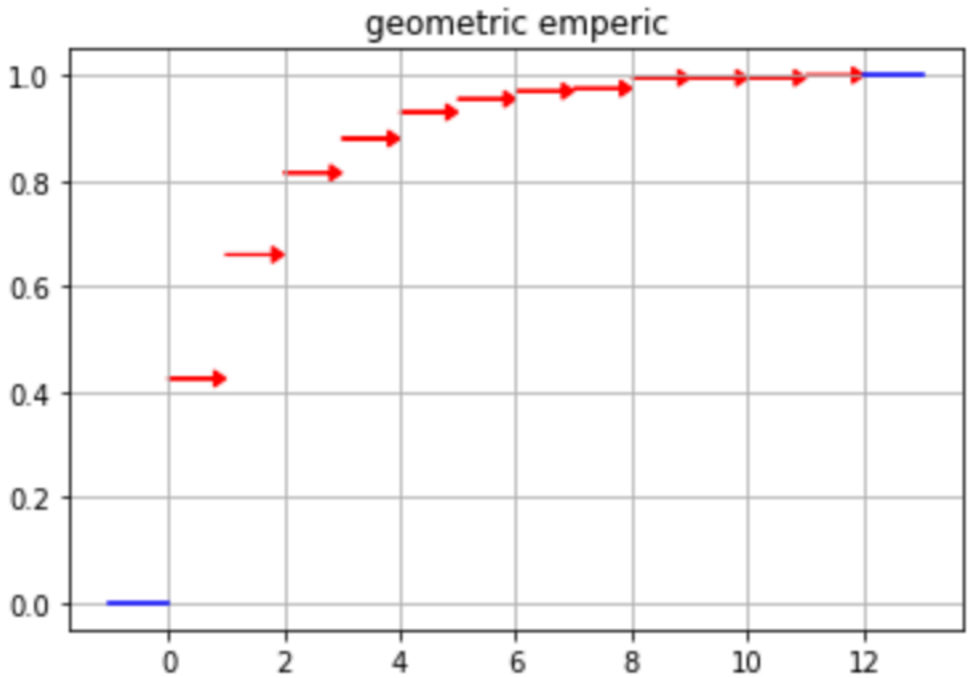
Статистический ряд:

Xk	Nk	Wk	Sk
0	85	0,425	0,425
1	47	0,235	0,66
2	31	0,155	0,815
3	13	0,065	0,88
4	10	0,05	0,93
5	5	0,025	0,955
6	3	0,015	0,97
7	1	0,005	0,975
8	4	0,02	0,995
11	1	0,005	1

Эмпирическая функция распределения и ее график:

$$F_{200}^{Э}(x) = \sum_{x_i \le x} w_i = \begin{cases} 0, & x < 0 \\ 0.425, & 0 \le x < 1 \\ 0.66, & 1 \le x < 2 \\ 0.815, & 2 \le x < 3 \\ 0.88, & 3 \le x < 4 \\ 0.93, & 4 \le x < 5 \\ 0.955, & 5 \le x < 6 \\ 0.97, & 6 \le x < 7 \\ 0.975, & 7 \le x < 8 \\ 0.995, & 8 \le x < 11 \\ 1, & x \ge 11 \end{cases}$$





Выборочное среднее = 1.4049999999999998
Выборочная дисперсия = 3.560975000000001
Выборочный момент 3-ого порядка = 31.405
Выборочный момент 4-ого порядка = 222.97499999999997
Выборочное среднее квадратическое отклонение = 1.8870545832063261
Выборочная мода = 0
Выборочная медиана = 1
Выборочный коэффициент асимметрии = 0.30167104769736
Выборочный коэффициент эксцесса = -2.3759431804499984

Задание 3 (распределение Пуассона)

λ=0,65

Неупорядоченная выборка (200 чисел):

0	0	2	0	1	0	0	0	0	1
1	1	0	1	0	0	2	1	1	1
1	0	0	1	3	0	0	0	0	0
0	1	1	0	1	0	1	1	1	2
1	0	1	1	1	1	0	1	1	0
1	1	1	0	0	0	0	1	2	0
0	1	2	0	2	0	0	0	0	0
0	2	0	0	1	2	0	0	0	3
0	0	1	1	1	1	1	1	1	0
0	1	1	0	0	2	0	1	2	0
1	3	0	0	0	1	0	0	1	1
0	0	0	3	1	0	0	0	1	0
0	1	0	0	1	1	0	1	1	1
0	1	1	0	0	1	0	0	0	1
1	0	1	1	1	2	0	0	1	0
1	2	0	0	1	0	1	0	1	1
1	0	1	0	0	1	1	1	1	0
3	1	0	0	3	0	0	1	0	0
0	0	0	0	1	1	0	2	2	0
0	1	3	0	2	1	0	1	0	0

Упорядоченная выборка (200 чисел):

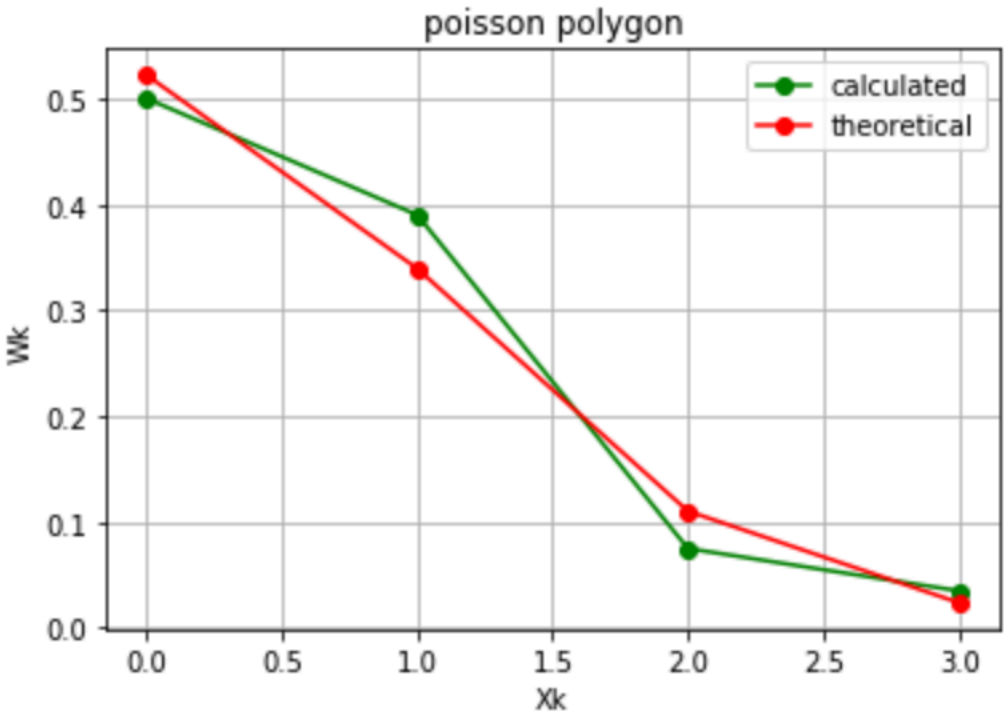
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	2	2
2	2	2	2	2	2	2	2	2	2
2	2	2	3	3	3	3	3	3	3

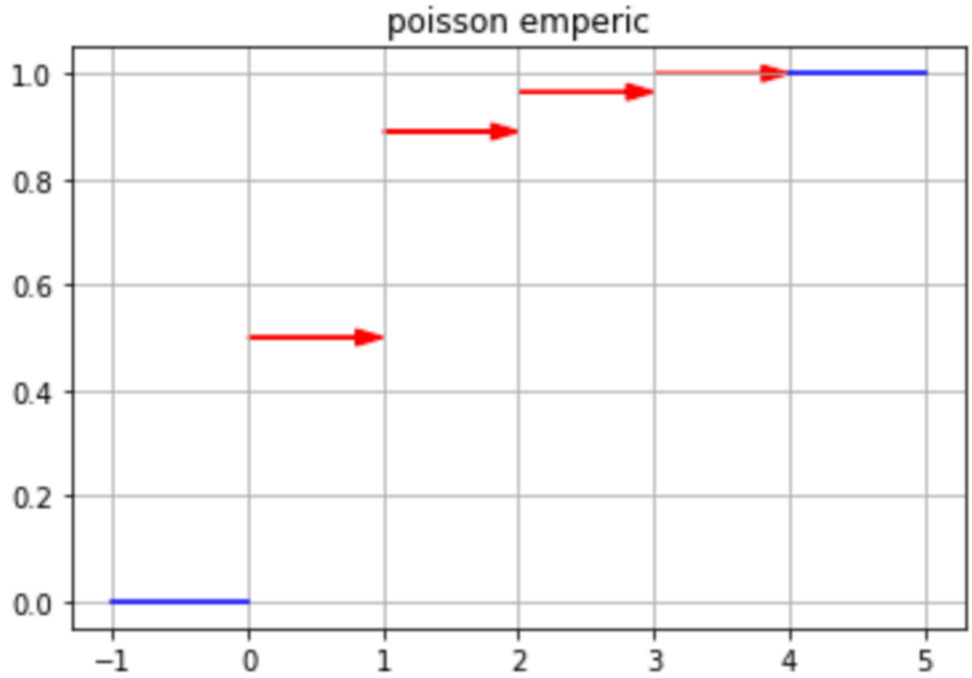
Статистический ряд:

Xk	Nk	Wk	Sk
0	100	0,5	0,5
1	78	0,39	0,89
2	15	0,075	0,965
3	7	0,035	1

Эмпирическая функция распределения и ее график:

$$F_{200}^{Э}(x) = \sum_{x_i \le x} w_i = \begin{cases} 0, & x < 0 \\ 0.5, & 0 \le x < 1 \\ 0.89, & 1 \le x < 2 \\ 0.965, & 2 \le x < 3 \\ 1, & x \ge 3 \end{cases}$$





Выборочное среднее = 0.645
Выборочная дисперсия = 0.5889749999999998
Выборочный момент 3-ого порядка = 1.935
Выборочный момент 4-ого порядка = 4.425000000000001
Выборочное среднее квадратическое отклонение = 0.767447066578536
Выборочная мода = 0
Выборочная медиана = 0.5
Выборочный коэффициент асимметрии = 2.5793944819997057
Выборочный коэффициент эксцесса = 8.817967718961837

Анализ результатов

Задание 1 (биномиальное распределение)

n = 20, p = 0,375

Таблица сравнения относительных частот и теоретических вероятностей:

j	w_j	P_j	$\lvert w_j - P_j \rvert$
1	0.05	0.0009926	0.0490074
2	0.01	0.0056579	0.0043421
3	0.02	0.0203684	0.0003684
4	0.04	0.0519396	0.019396
5	0.09	0.0997241	0.0097241
6	0.16	0.1495862	0.0104138
7	0.14	0.1795034	0.0395034
8	0.18	0.1750158	0.0049842
9	0.13	0.1400127	0.0100127
10	0.11	0.0924083	0.0175917

11	0.075	0.0504045	0.0245955
12	0.02	0.0226820	0.0026820
13	0.015	0.0083749	0.0066251
14	0	0.0025124	0.0025124
15	0.005	0.0006029	0.0043971

Таблица сравнения рассчитанных характеристик с теоретическими значениями:

Название показателя	Экспериментальное значение	Теоретическое значение	Абсолютное отклонение	Относительное отклонение
Выборочное среднее	7.68	7.5	0.18	2.4%
Выборочная дисперсия	5.4176	4.6875	0.7301	15.575466%
Выборочное среднее квадратичное отклонение	2.3275738	2.1650635	0.1625103	7.5060292%
Выборочная мода	8	7	1	14.285714%
Выборочная медиана	8	8	0	0%
Выборочный коэффициент асимметрии	0.0022241	0.11547	0.1132459	0.980738%
Выборочный коэффициент эксцесса	-2.896367	-0.086666	2.809701	3241.9876%

Задание 2 (геометрическое распределение)

p = 0,375

Таблица сравнения относительных частот и теоретических вероятностей:

j	w_j	P_j	$\lvert w_j - P_j \rvert$
0	0.425	0.375	0.05
1	0.235	0.234375	0.000625
2	0.155	0.1464843	0.0085157
3	0.065	0.0915527	0.0265527
4	0.05	0.0572204	0.0072204
5	0.025	0.0357627	0.0107627
6	0.015	0.0223517	0.0013002
7	0.005	0.0139698	0.0089698
8	0.005	0.0087311	0.0037311
9	0	0.0054569	0. 0054569
10	0.	0.0034106	0. 0034106
11	0.005	0.0021316	0.0028684

Таблица сравнения рассчитанных характеристик с теоретическими значениями:

Название показателя	Экспериментальное значение	Теоретическое значение	Абсолютное отклонение	Относительное отклонение
Выборочное среднее	1.404999	1.666667	0.261668	15.700076%
Выборочная дисперсия	3.560974	4.444445	0.883471	19.878095%
Выборочное среднее квадратичное отклонение	1.887054	2.1081851	0.2211311	10.489169%
Выборочная мода	0	0	0	-
Выборочная медиана	1	1	0	0%
Выборочный коэффициент асимметрии	0.301671	2.055480	1.753809	85.3235741%
Выб. коэфф. эксцесса	-2.375943	6.225	8.630943	137.9847%

Задание 3 (распределение Пуассона)

λ=0,65

Таблица сравнения относительных частот и теоретических вероятностей:

j	w_j	P_j	$\lvert w_j - P_j \rvert$
0	0.5	0.5220457	0.0220457
1	0.39	0.3393297	0.0506703
2	0.075	0.1102821	0.0352821
3	0.035	0.0238944	0.0111056

Таблица сравнения рассчитанных характеристик с теоретическими значениями:

Название показателя	Экспериментальное значение	Теоретическое значение	Абсолютное отклонение	Относительное отклонение
Выборочное среднее	0.645	0,65	0.005	8.064516%
Выборочная дисперсия	0.58897	0,65	0.06103	9.389230%
Выборочное среднее квадратичное отклонение	0.76744	0.80622	0.03878	4.810101%
Выборочная мода	0	0	0	-

Выборочная медиана	0.5	0	0.5	-
Выборочный коэффициент асимметрии	2.57939	1.24034	1.33905	107.958302%
Выб. коэфф. эксцесса	8.81796	1.53846	7.2795	473.167973%

Вывод

В ходе лабораторной работы выяснилось, что полученные экспериментальным путем данные соответствуют заданным распределениям, если принимать в расчет отклонения от теоретического значения.

Экспериментальная оценка выборочных показателей может сильно отличаться от теоретического значения, в силу того, что выборки из 200 элементов недостаточно для проведения точных расчетов. С увеличением выборки точность будет улучшаться.

Список литературы

1. Математическая статистика [Электронный ресурс]: метод. указания по выполнению лаб. работ / А.А. Лобузов — М.: МИРЭА, 2017.
2. Боровков А. А. Математическая статистика. − СПб.: Лань, 2010. − 704 с.
3. Гмурман В.Е. Теория вероятностей и математическая статистика. − М.: Юрайт, 2013. − 479 с.
4. Кетков Ю.Л., Кетков Ю.Л., Шульц М.М. MATLAB 7: программирование, численные методы. − СПб.: БВХ-Петербург, 2005. − 752 с.

Приложение

```
# вариант №15
import matplotlib.pyplot as plt
from numpy.random import binomial, geometric, poisson
import pandas as pd
from collections import Counter
from math import factorial as fac
from math import exp, sqrt, floor, log

V = 15
n = 5+V%16
p = 0.3+0.005*V
lam = 0.5+0.01*V
size = 200
print(n, p, lam, size)

# создание df
def df_create(data, size):
    Wk = {}
    Sk = {}
    Nk = Counter(data)
    sum_Wk = 0
    for i in range(n):
        Wk[i] = Nk[i]/size
        sum_Wk += Wk[i]
        Sk[i] = sum_Wk

    df = pd.DataFrame(sorted(list(Nk.items())), columns=['Xk', 'Nk'])

    d_Wk = list()
    for i in range(len(Wk)):
        if Wk[i] != 0:
            d_Wk.append(Wk[i])
    df['Wk'] = d_Wk

    d_Sk = list()
    for i in sorted(list(Nk.keys())):
        d_Sk.append(Sk[i])
    df['Sk'] = d_Sk

    return df

# график знач. теор. бином. распр.
def comb(n, k):
    return (fac(n)/fac(k)/fac(n-k))

def binom_comp(n, k, p):
    return (comb(n,k)*(p**k)*((1-p)**(n-k)))
```

```
def plot_binom_theoretical(df):
    arr_x = list(range(0,(df['Xk'][len(df)-1])+1))
    arr_y = list()
    for i in range(df['Xk'][len(df)-1]+1):
        arr_y.append(binom_comp(n, i, p))
    plt.plot(arr_x, arr_y, marker='o', color='red', label='theoretical')

# график знач. теор. геом. распр.
def plot_geomm_theoretical(df):
    arr_x = list(range(df['Xk'][0], df['Xk'][len(df)-1]+1))
    arr_y = list()
    for i in range(df['Xk'][0], df['Xk'][len(df)-1]+1):
        arr_y.append(((1-p)**i)*p)
    plt.plot(arr_x, arr_y, marker='o', color='red', label='theoretical')

# график знач. теор. Пуассоновского распр.
def plot_poisson_theoretical(df):
    arr_x = list(range(df['Xk'][0], df['Xk'][len(df)-1]+1))
    arr_y = list()
    for i in range(df['Xk'][0], df['Xk'][len(df)-1]+1):
        arr_y.append((lam**i/fac(i)*exp(-lam)))
    plt.plot(arr_x, arr_y, marker='o', color='red', label='theoretical')

# график эмпир. ф-ии распр.
def plot_emperic(df):
    l = len(df)-1
    arr_x = list(range(df['Xk'][0],df['Xk'][l]+1))
    lst = list(df['Xk'])
    arr_y = []
    k=0
    for i in arr_x:
        if i in lst:
            arr_y.append(float(df.loc[df['Xk'] == i]['Sk']))
        else:
            arr_y.append(arr_y[k-1])
        k+=1
    for i in range(len(arr_x)):
        plt.arrow(arr_x[i], arr_y[i], 0.8, 0, head_width=0.03, head_length=0.2, color='red')
    plt.plot([df['Xk'][0], df['Xk'][0]-1], [0,0], color='blue')
    plt.plot([df['Xk'][l]+1, df['Xk'][l]+2], [1,1], color='blue')
    plt.grid(True)

# выборочное среднее
def sample_mean(df):
    size = 0
    for i in range (len(df)):
        size += df['Nk'][i]
    x = 0
    for i in range(len(df)):
        x += (df['Xk'][i])*df['Wk'][i]
    return x

# выборочниый момент К-го порядка
def sample_moment_k(df, k):
```

```
    size = 0
    for i in range (len(df)):
        size += df['Nk'][i]
    x = 0
    for i in range(len(df)):
        x += (df['Xk'][i]**k)*df['Wk'][i]
    return x

# выборочная дисперсия
def sample_dispersion(df):
    return (sample_moment_k(df, 2)-(sample_mean(df)**2))

# выборочное среднее квадратическое отклонение
def sample_quadratic_deviation(df):
    return (sqrt(sample_dispersion(df)))

# выборочный центральный момент k-ого порядка
def sample_centr_moment_k(df, k):
    x = 0
    sm = sample_mean(df)
    for i in range(len(df)):
        x += ((df['Xk'][i]-sm)**k)*df['Wk'][i]
    return x

# выборочная мода
def sample_fashion(df):
    max_c = max(df['Nk'])
    index = list()
    for i in range(len(df)):
        if df['Nk'][i] == max_c:
            index.append(i)
    check = True
    if len(index) > 1:
        for i in range(1,len(index)):
            if index[i]-index[i-1] != 1:
                check = False
    else:
        return (df['Xk'][index[0]])
    if check:
        return ((df['Xk'][index[0]]+df['Nk'][index[-1]])/2)
    else:
        return None

# выборочная медиана
def sample_median(df):
    for i in range(len(df)):
        if df['Sk'][i] == 0.5 :
            return ((df['Xk'][i]+df['Xk'][i+1])/2)
        elif df['Sk'][i] > 0.5 :
            return (df['Xk'][i])

# выборочный коэффициент асимметрии
def sample_asymmetry_coef(df):
    return (sample_centr_moment_k(df, 3) / sample_dispersion(df)**3)
```

```
# выборочный коэффициент эксцесса
def sample_kurtosis_coef(df):
    return ((sample_centr_moment_k(df, 4) / sample_dispersion(df)**4) - 3)

def print_all(df):
    print('Выборочное среднее =', sample_mean(df))
    print('Выборочная дисперсия =', sample_dispersion(df))
    print('Выборочный момент 3-ого порядка =', sample_moment_k(df, 3))
    print('Выборочный момент 4-ого порядка =', sample_moment_k(df, 4))
    print('Выборочное среднее квадратическое отклонение =', sample_quadratic_deviation(df))
    print('Выборочная мода =', sample_fashion(df))
    print('Выборочная медиана =', sample_median(df))
    print('Выборочный коэффициент асимметрии =', sample_asymmetry_coef(df))
    print('Выборочный коэффициент эксцесса =', sample_kurtosis_coef(df))

def print_binom_teor():
    print('BINOM teoreric')
    print('expected value =', n*p)
    print('dispersion =', n*p*(1-p))
    print('quadratic deviation =', sqrt(n*p*(1-p)))
    if type((n+1)*p)==int:
        print('fashion =', (n+1)*p - 0.5)
    else:
        print('fashion =', floor((n+1)*p))
    print('median =', round(n*p))
    print('asymmetry =', (1 - 2*p) / sqrt(n*p*(1-p)))
    print('kurtosis =', (1 - 6*(1-p)*p) / (n*p*(1-p)))

def print_geom_teor():
    print('GEOM teoreric')
    print('expected value =', (1-p)/p)
    print('dispersion =', (1-p)/p**2)
    print('quadratic deviation =', sqrt(1-p)/p)
    print('fashion =', 0)
    if log(2)%log(1-p) == 0:
        print('median =', -log(2)/log(1-p) - 0.5)
    else:
        print('median =', floor(-log(2)/log(1-p)))
    print('asymmetry =', (2-p)/sqrt(1-p))
    print('kurtosis =', 6+p**2/(1-p))

def print_poisson_teor():
    print('POISSON teoreric')
    print('expected value =', lam)
    print('dispersion =', lam)
    print('quadratic deviation =', sqrt(lam))
    print('fashion =', floor(lam))
    print('median =', floor(lam+1/3-0.02/lam))
    print('asymmetry =', 1/sqrt(lam))
    print('kurtosis =', 1/lam)

# биномиальное
# data_binom = binomial(n, p, size)
```

```
# [11, 11,  8, 10, 10,  3,  8, 10,  5,  9,
#   6,  8,  5,  7,  6,  3,  8,  7,  8, 12,
#  11,  4,  3,  4, 11,  7,  6, 10,  7, 11,
#  12,  9,  5,  6,  4,  8, 10,  5,  9,  9,
#   6, 10, 10,  4,  6,  7, 11,  7,  5,  6,
#   9,  5,  6,  5,  6,  9,  7,  6, 10, 10,
#  10,  8,  7,  9,  8,  6,  8,  9, 11,  9,
#   8,  4,  8,  9,  9,  9,  9,  8,  8,  6,
#   7,  8,  7,  8, 13,  6,  5,  8, 11,  6,
#   7,  6,  9, 10,  8, 11, 10, 10,  7, 11,
#  11,  8,  5, 12,  6,  6,  5, 11,  8,  7,
#   7, 10,  9,  7,  6,  4,  6,  4,  8,  7,
#  10,  6, 10,  5,  7,  8, 10,  4, 10, 10,
#  15,  8,  8,  6,  6,  7,  6,  6,  9,  6,
#   7,  9,  5,  9,  5,  8,  6,  1,  8, 12,
#   6,  8,  8,  7,  5,  5,  7,  8, 13,  7,
#   7,  8,  9,  6,  5,  9,  8,  9,  7,  8,
#  11,  8,  9,  5,  8,  9,  2,  7,  8, 10,
#   9,  6,  3,  8, 10, 11,  7,  9, 11,  6,
#   6,  9, 10,  5,  8,  6,  7,  2, 13,  7]
# sort_data_binom = sorted(data_binom)
# [1, 2, 2, 3, 3, 3, 3, 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, 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, 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, 8, 8, 8, 8, 8, 8, 9,
#  9, 9, 9, 9, 9, 9, 9, 9, 9, 9,
#  9, 9, 9, 9, 9, 9, 9, 9, 9, 9,
#  9, 9, 9, 9, 9, 10, 10, 10, 10, 10,
#  10, 10, 10, 10, 10, 10, 10, 10, 10, 10,
#  10, 10, 10, 10, 10, 10, 10, 11, 11, 11,
#  11, 11, 11, 11, 11, 11, 11, 11, 11, 11,
#  11, 11, 12, 12, 12, 12, 13, 13, 13, 15]
# df_binom = df_create(data_binom, size)
# df_binom.to_excel("output_binom.xlsx", sheet_name='Sheet_name_1', index=False)

df_binom = pd.read_excel('output_binom.xlsx')
df_binom

df_binom.plot(style='o-', x='Xk', y='Wk', c='green', label='calculated')
plot_binom_theoretical(df_binom)
plt.legend()
plt.ylabel('Wk')
plt.grid(True)
plt.title('binominal polygon')
```

```
plot_emperic(df_binom)
plt.title('binominal emperic')

print_all(df_binom)
print()
print_binom_teor()

# геометрическое
# data_geometric = geometric(p, size)
# data_geometric
# [2, 3, 1, 2, 1, 1, 5, 1, 1, 2,
#  1, 1, 1, 4, 1, 2, 1, 1, 3, 2,
#  1, 1, 9, 1, 4, 1, 1, 3, 1, 1,
#  2, 1, 1, 1, 1, 3, 9, 1, 1, 1,
#  3, 1, 2, 4, 3, 3, 2, 1, 3, 2,
#  1, 6, 2, 1, 1, 1, 1, 1, 7, 2,
#  3, 3, 3, 3, 1, 2, 1, 3, 4, 4,
#  2, 1, 2, 2, 1, 3, 2, 6, 2, 5,
#  5, 3, 2, 3, 2, 5, 2, 2, 1, 1,
#  3, 8, 3, 5, 1, 2, 2, 1, 2, 1,
#  2, 1, 1, 2, 1, 9, 1, 6, 2, 1,
#  1, 2, 2, 1, 1, 2, 3, 1, 1, 1,
#  1, 7, 1, 3, 1, 4, 3, 1, 6, 2,
#  2, 1, 1, 1, 1, 3, 1, 1, 1, 1,
#  12, 2, 5, 3, 1, 1, 1, 2, 1, 1,
#  2, 4, 4, 3, 2, 3, 1, 2, 4, 2,
#  2, 1, 3, 2, 1, 1, 4, 2, 9, 6,
#  3, 1, 1, 1, 2, 5, 1, 3, 1, 2,
#  5, 1, 1, 5, 3, 4, 5, 4, 1, 2,
#  2, 2, 1, 4, 7, 3, 1, 1, 3, 2]

# sort_data_geometric = (sorted(data_geometric))
# print(sort_data_geometric)
# [1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 2, 2, 2, 2, 2,
#  2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
#  2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
#  2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
#  2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
#  2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
#  3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
#  3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
#  3, 3, 3, 4, 4, 4, 4, 4, 4, 4,
#  4, 4, 4, 4, 4, 4, 5, 5, 5, 5,
#  5, 5, 5, 5, 5, 5, 6, 6, 6, 6,
#  6, 7, 7, 7, 8, 9, 9, 9, 9, 12]
```

```
# df_geometric = df_create(data_geometric, size)
# df_geometric.to_excel("output_geometric.xlsx", sheet_name='Sheet_name_1', index=False)

df_geometric = pd.read_excel('output_geometric.xlsx')
df_geometric

df_geometric.plot(style='o-', x='Xk', y='Wk', c='green', label='calculated')
plot_geomm_theoretical(df_geometric)
plt.title('geometric polygon')
plt.legend()
plt.ylabel('Wk')
plt.grid(True)

plot_emperic(df_geometric)
plt.title('geometric emperic')

print_all(df_geometric)
print()
print_geom_teor()

# Пуассон
# data_poisson = poisson(lam, size)
# data_poisson

# [0, 0, 2, 0, 1, 0, 0, 0, 0, 1,
#  1, 1, 0, 1, 0, 0, 2, 1, 1, 1,
#  1, 0, 0, 1, 3, 0, 0, 0, 0, 0,
#  0, 1, 1, 0, 1, 0, 1, 1, 1, 2,
#  1, 0, 1, 1, 1, 1, 0, 1, 1, 0,
#  1, 1, 1, 0, 0, 0, 0, 1, 2, 0,
#  0, 1, 2, 0, 2, 0, 0, 0, 0, 0,
#  0, 2, 0, 0, 1, 2, 0, 0, 0, 3,
#  0, 0, 1, 1, 1, 1, 1, 1, 1, 0,
#  0, 1, 1, 0, 0, 2, 0, 1, 2, 0,
#  1, 3, 0, 0, 0, 1, 0, 0, 1, 1,
#  0, 0, 0, 3, 1, 0, 0, 0, 1, 0,
#  0, 1, 0, 0, 1, 1, 0, 1, 1, 1,
#  0, 1, 1, 0, 0, 1, 0, 0, 0, 1,
#  1, 0, 1, 1, 1, 2, 0, 0, 1, 0,
#  1, 2, 0, 0, 1, 0, 1, 0, 1, 1,
#  1, 0, 1, 0, 0, 1, 1, 1, 1, 0,
#  3, 1, 0, 0, 3, 0, 0, 1, 0, 0,
#  0, 0, 0, 0, 1, 1, 0, 2, 2, 0,
#  0, 1, 3, 0, 2, 1, 0, 1, 0, 0]
# sort_data_poisson = (sorted(data_poisson))
# print(sort_data_poisson)

# [0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
#  1, 1, 1, 1, 1, 1, 1, 1, 2, 2,
#  2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
#  2, 2, 2, 3, 3, 3, 3, 3, 3, 3]

# df_poisson = df_create(data_poisson, size)
# df_poisson.to_excel("output_poisson.xlsx", sheet_name='Sheet_name_1', index=False)

df_poisson = pd.read_excel('output_poisson.xlsx')
df_poisson

df_poisson.plot(style='o-', x='Xk', y='Wk', c='green', label='calculated')
plot_poisson_theoretical(df_poisson)
plt.legend()
plt.ylabel('Wk')
plt.grid(True)
plt.title(‘poisson polygon')

plot_emperic(df_poisson)
plt.title('poisson emperic')

print_all(df_poisson)
print()
print_poisson_teor()

# теор. значения распр.
def comb(n, k):
   return (fac(n)/fac(k)/fac(n-k))

def binom_comp(n, k, p):
   return (comb(n,k)*(p**k)*((1-p)**(n-k)))

arr_binom_teor = []
arr_geom_teor = []
arr_poisson_teor = []

for i in range(df_binom['Xk'][0], df_binom['Xk'][len(df_binom)-1]+1):
   arr_binom_teor.append(binom_comp(n, i, p))
print('binom')
for i in range(len(arr_binom_teor)):
   print(i+1, arr_binom_teor[i])

for i in range(df_geometric['Xk'][0], df_geometric['Xk'][len(df_geometric)-1]+1):
   arr_geom_teor.append(((1-p)**i)*p)
print('geom')
```

```
for i in range(len(arr_geom_teor)):
    print(i+1, arr_geom_teor[i])

for i in range(df_poisson['Xk'][0], df_poisson['Xk'][len(df_poisson)-1]+1):
    arr_poisson_teor.append((lam**i/fac(i)*exp(-lam)))
print('poisson')
for i in range(len(arr_poisson_teor)):
    print(i, arr_poisson_teor[i])
```