Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана» (национальный исследовательский университет) (МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ФУНДАМЕНТАЛЬНЫЕ НАУКИ

КАФЕДРА ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И МАТЕМАТИЧЕСКАЯ ФИЗИКА (ФН11)

НАПРАВЛЕНИЕ ПОДГОТОВКИ МАТЕМАТИКА И КОМПЬЮТЕРНЫЕ НАУКИ (02.03.01)

О т ч е т

по лабораторной работе № 2

Название лабораторной работы:
Моделирование и обработка выборки из дискретного закона распределения.

Вариант № 9

Дисциплина:

Теория вероятности и математическая статистика

Студент группы ФН11-52Б ____________ (Подпись, дата) **Очкин Н.В.** (И.О. Фамилия)

Преподаватель ____________ (Подпись, дата) **Облакова Т.В.** (И.О. Фамилия)

Москва, 2024

Задание

1. Для заданных значений k, p и n смоделируйте выборку из биномиального закона распределения:

$$P(\xi = j) = p_j = C_k^j p^j (1-p)^{k-j}, j = \overline{0,k}.$$

2. Для полученной выборки постройте статистический ряд. Найдите эмпирическую функцию распределения $\tilde{F}_n(x)$. Постройте на одном рисунке графики $F(x)$ и $\tilde{F}_n(x)$. Вычислите статистику Колмогорова.

3. Вычислите выборочное среднее и выборочную дисперсию и сравните с истинными значениями этих характеристик.

Исходные данные

$$k = 8, \qquad p = 0.7, \qquad n = 140$$

Порядок выполнения работы

Найдем теоретический закон по формуле Бернулли(вектор P) и вычислим вектор кумулятивных вероятностей u

$$P = \quad [0.00007,\ 0.00122,\ 0.01,\ 0.04668,\ 0.13614,\ 0.25412,\ 0.29648,\ 0.19765,\ 0.05765]$$
$$u = \quad [0.00007,\ 0.00129,\ 0.01129,\ 0.05797,\ 0.1941,\ 0.44823,\ 0.7447,\ 0.94235,\ 1.]$$

Смоделируем вектор Y из n случайных чисел, по вектору Y разыграем вектор X в соответствии со следующим алгоритмом:

```
name: k
input: u, r
output: int

i = 0
for j in u length:
  if r is less than u_j:
      break
  i += 1

return i
```

$$X_j = k(u, Y_j)$$
$$j = 0, ..., n-1$$

$$Y = [0.22183838, 0.25624019, 0.05717201, \cdots, 0.13413202, 0.0200769, 0.15263197]$$
$$X = [5, 5, 3, 6, 6, 3, 5, 6, 4, 7, 5, 5, \cdots, 7, 6, 4, 7, 7, 4, 3, 4]$$

Построим статистический ряд (см. Приложение) и запишем результат в виде таблицы:

Значения случайной величины	Частоты	Относительные частоты	Накопленные частоты
0	0	0	0
1	1	0.00714286	0.00714286
2	2	0.01428571	0.02142857
3	8	0.05714286	0.07857143
4	17	0.12142857	0.2
5	37	0.26428571	0.46428571
6	37	0.26428571	0.72857143
7	31	0.22142857	0.95
8	7	0.05	1

Вектор накопленных частот содержит ненулевые значения эмпирической функции распределения, соответствующие значения теоретической функции распределения состовляют вектор u. Для вычисления статистики Колмогорова:

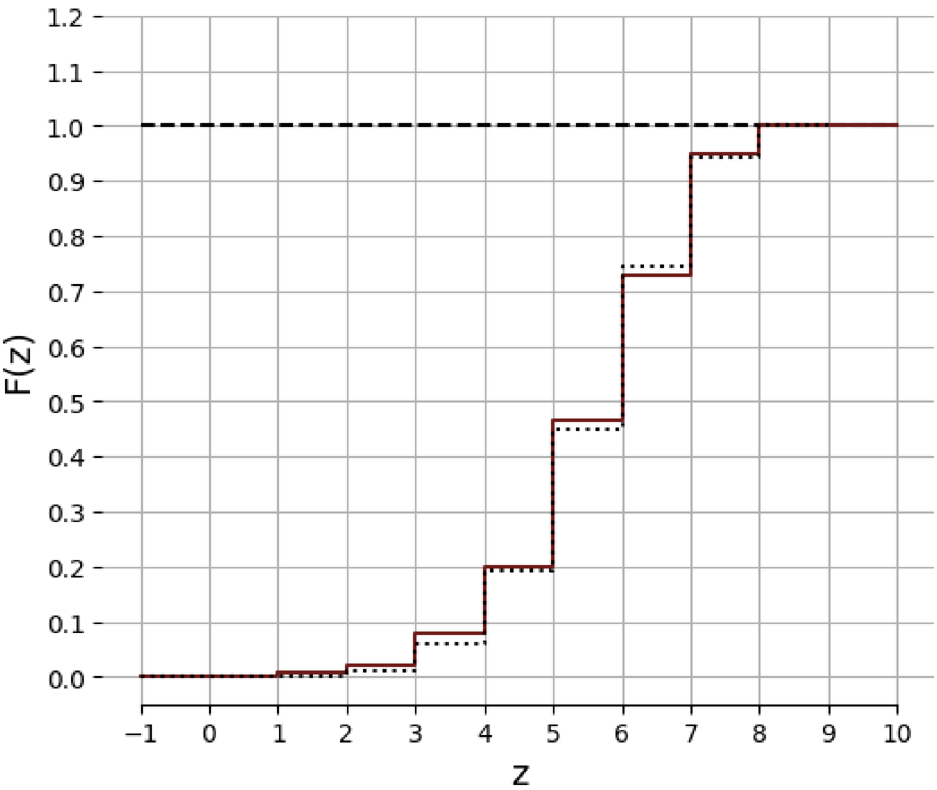
$$D_n = \sup_{x \in \mathbb{R}} \left| \tilde{F}_n(x) - F(x) \right|$$

данные удобно свести в таблицу:

Интервал	Эмпирическая функция распределения	Теоретическая функция распределения	Модуль разности
$(-\infty, 0]$	0	0	0
$(0, 1]$	0	0.00007	0.00007
$(1, 2]$	0.00714	0.00129	0.00585
$(2, 3]$	0.02143	0.01129	0.01014
$(3, 4]$	0.07857	0.05797	0.0206
$(4, 5]$	0.2	0.1941	0.0059
$(5, 6]$	0.46429	0.44823	0.01606
$(6, 7]$	0.72857	0.7447	0.01613
$(7, 8]$	0.95	0.94235	0.00765
$(8, +\infty)$	1	1	0

Из данных таблицы следует, что максимальное различие теоретической и эмпирической функций распределения наблюдается на полуинтервале $(3, 4]$. Заметим также, что значение статистики Колмогорова $D_{140} = 0.0206$ невелико, что говорит о приемлемом результате моделирования.

Изобразим совмещенные графики эмпирической и теоретической функций распределения:



где красной линией отмечен график эмпирической функции распределения; черной пунктирной - теоретической.

Выполнение работы завершим вычислением эмпирических и теоретических характеристик.

$$
\begin{aligned}
&\text{Выборочное среднее} &&= m\mu &&= \frac{1}{n} \cdot \sum_{k=0}^{n} X_k &&= 5.55 \\
&\text{Выборочная дисперсия} &&= S^2 &&= \frac{1}{n-1} \cdot \sum_{k=0}^{n} (X_k - m\mu)^2 &&= 1.90396 \\
&\text{Математическое ожидание} &&= m &&= \sum_{i=0}^{n} (i \cdot P_i) &&= 5.6 \\
&\text{Дисперсия} &&= D &&= \sum_{i=0}^{n} \left[(i - m)^2 \cdot P_i \right] &&= 1.68
\end{aligned}
$$

Поскольку абсолютная величина разности математического ожидания и выборочного среднего мала, а отношение выборочной дисперсии к ее теоретическому значению близко к единице, то результаты моделирования можно признать удовлетворительными.

Итог

В ходе проделанной лабораторной работы было проведено моделирование и обработка выборки из дискретного закона распределения. Для полученной выборки построен статистический ряд и эмпирическая функция распределения, вычислена статистика Колмогорова. На основании значений выборочного среднего и выборочной дисперсии был сделан вывод о степени качества моделирующей дискретный закон выборки.

Приложение

Программный код, с помощью которого была выполнена данная лабораторная работа.

Примечание.

Так как отчет был написан с использованием дистрибутива TeX и следующий код был отформатирован с использованием окружения LSTLISTING, в некоторых местах текст, написанный на русском языке, может иметь проблемы с выравниванием, пробелами, шрифтом и т.д. Это связано с тем, что библиотека LISTINGS, из которой мы берем окружение для форматирования кода, плохо работает с Unicode.

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

np.set_printoptions(suppress=True)

PRECISION = 5

k = 8   # Количествоиспытаний
p = 0.7 # Вероятностьуспехаводномиспытании
n = 140 # Объемвыборки

probs = [] # Вероятности

for j in range(k + 1):
    pj = math.comb(k, j) * p**j * (1 - p)**(k - j) # формулаБернулли
    probs.append(pj)

np.round(np.array(probs), PRECISION)

kumProbs = [sum(probs[:(i + 1)]) for i in range(k + 1)] # Кумулятивныевероятности

np.round(np.array(kumProbs), PRECISION)
```

```
Y = np.random.rand(n) # n случайныхвеличин
Y

# Повектору  Y разыгрываемвектор  X всоответствиисалгоритмом
def k_func(u, r):
    i: int = 0
    for j in range(len(u)):
        if r < u[j]:
            break
        i += 1
    return i

X = []

for Yj in Y:
    X.append(k_func(kumProbs, Yj))

print(X)

# Строимстатистическийряд
def findFreq(data, k):
    values = np.arange(k + 1)
    counting = {}
    for value in values:
        counting[value] = 0

    for el in data:
        counting[el] += 1

    return [counting[el] for el in values]

values  = np.arange(k + 1)
freq    = findFreq(X, k)
relFreq = np.array(freq) / n
kumFreq = np.array([sum(relFreq[:(i + 1)]) for i in range(len(relFreq))])

print(f'Значения случайнойвеличины : {values}')
print(f'Частоты:                      {freq}')
print(f'Относительные частоты:        {relFreq}')
print(f'Накопленные частоты:          {kumFreq}')

def CDF(z, values, kumFreq):
    if z <= values[0]:
        return 0

    if len(values) > 1:
        for i in range(1, len(values)):
            prev = values[i - 1]
            curr = values[i]

            if prev < z <= curr:
                return kumFreq[i - 1]

    if z > values[-1]:
        return 1

        def buildCDF(data,
        cdf, values, kumFreq,
        theoretical_cdf_y_values):
```

```python
RED    = '#6F1D1B'

# Define font sizes
SIZE_DEFAULT = 14
SIZE_LARGE   = 16
SIZE_TICKS   = 10
plt.rc("font", weight="normal")           # controls default font
plt.rc("font", size=SIZE_DEFAULT)         # controls default text sizes
plt.rc("axes", titlesize=SIZE_LARGE)      # fontsize of the axes title
plt.rc("axes", labelsize=SIZE_DEFAULT)    # fontsize of the x and y labels
plt.rc("xtick", labelsize=SIZE_DEFAULT)   # fontsize of the tick labels
plt.rc("ytick", labelsize=SIZE_DEFAULT)   # fontsize of the tick labels

_, ax = plt.subplots(
   figsize=(6, 5)
)

# Generate a range of x values
x_values = np.linspace(-1, np.max(data) + np.min(data) + 1, 100)

# Evaluate the function for each x value (empirical)
cdf_y_values = [cdf(x, values, kumFreq) for x in x_values]

xticks = [i for i in range(-1, int(np.max(data) + np.min(data)) + 1 + 1)]
yticks = np.arange(0, 1.2 + 0.1, 0.1)

# Hide the all but the bottom spines (axis lines)
ax.spines["right"].set_visible(False)
ax.spines["left"].set_visible(False)
ax.spines["top"].set_visible(False)

# Only show ticks on the left and bottom spines
ax.yaxis.set_ticks_position("left")
ax.xaxis.set_ticks_position("bottom")
ax.spines["bottom"].set_bounds(min(xticks), max(xticks))

# plot y = 1 line
plt.plot(x_values, np.full_like(x_values, 1), label='y = 1', linestyle='--',
                                                              color='black')

# Plot cdf(x) (empirical)
plt.step(x_values, cdf_y_values, label='empirical(x)', color=RED)

# Plot the theoretical distribution function
x_values = np.arange(-1, k + 2)
plt.step(x_values, theoretical_cdf_y_values, label='theoretical(x)',
                   color='black', linestyle='dotted')

# axis names
plt.xlabel('z')
plt.ylabel('F(z)')

plt.xticks(xticks)
plt.yticks(yticks)

# Adjust the font size of the tick labels
plt.tick_params(axis='both', which='major', labelsize=SIZE_TICKS)

plt.grid(True)
```

```python
plt.show()

kumProbs = [0] + [0] + kumProbs
buildCDF(X,
         CDF, values, kumFreq, # эмпирическая
         kumProbs)             # теоретическая

empirVals = []
theoretVals = []
diffs = []
print(f'значение: ', end='')
for i in range(0, k + 2):
    print(f'({i-1}, {i}]',end=', ')
    empirVal   = CDF(i, values, kumFreq)
    theoretVal = kumProbs[i + 1]
    diff = abs(empirVal - theoretVal)

    empirVals.append(empirVal)
    theoretVals.append(theoretVal)
    diffs.append(diff)

print(f'\n эмпирфункцияраспр  : {np.round(np.array(empirVals), PRECISION)}')
print(f'теор функцияраспр :  {np.round(np.array(theoretVals), PRECISION)}')
print(f'модуль разности:     {np.round(np.array(diffs), PRECISION)}')

D = max(diffs)
print(f'\nD = {D}')

# эмпирическая

# выборочноесреднее
overlineX = np.round((1 / n) * np.sum(X), PRECISION)
m1 = np.mean(X)

# выборочнаядисперсия
S2 = np.round(1 / (n - 1) * np.sum((X - overlineX)**2), PRECISION)

print(f'выборочное среднее: {overlineX} ({m1})')
print(f'выборочная дисперсия: {S2}')

# теоретическая

# матожидание
m = sum([i * probs[i] for i in range(k + 1)])

# дисперсия
d = sum([(i - m)**2 * probs[i] for i in range(k + 1)])

print(f'мат ожидание: {m}')
print(f'дисперсия: {d}')
```