Санкт-Петербургский Политехнический
Университет им. Петра Великого

Институт прикладной математики и механики

Кафедра прикладной математики

Лабораторная работа №5

3 курс, группа 33631/2

Студент Д. А. Плаксин

Преподаватель Баженов А. Н.

Санкт-Петербург
2019 г.

Содержание

1. Список иллюстраций **3**

2. Список таблиц **3**

3. Постановка задачи **4**

4. Теория **4**

5. Реализация **4**

6. Результаты **5**

7. Выводы **9**

8. Список литературы **9**

9. Приложения **9**

1 Список иллюстраций

1 Графики двумерного нормального распределения(2) при $p = 0.0$ 5
2 Графики двумерного нормального распределения(2) при $p = 0.5$ 6
3 График двумерного нормального распределения (2) при $p = 0.9$ 7
4 Графики смеси двумерных нормальных распределений 8

2 Список таблиц

1 Результаты для двумерного нормального распределения (2) при $p = 0.0$ . . 5
2 Результаты для двумерного нормального распределения (2) при $p = 0.5$ . . 6
3 Результаты для двумерного нормального распределения (2) при $p = 0.9$ . . 7
4 Результаты для смеси двумерных нормальных распределений 8

3 Постановка задачи

Необходимо построить выборки объёмом $20, 60, 100, 1000$ для двумерного нормального распределения с коэффициентами корреляции $\rho = 0, 0.5, 0.9$

Вычислить коэффициент корреляции Пирсона, Спирмана и квадрантный коэффициент корреляции для каждой выборки. Эти же вычисления повторить для смеси двумерных нормальных распределений [4]:

$$f(x, y) = 0.9N(x, y, 0, 0, 1, 1, 0.9) + 0.1N(x, y, 0, 0, 10, 10, -0.9) \tag{1}$$

На графике изобразить точки выборки и эллипс равновероятности.

4 Теория

1. Двумерное нормально распределение [5]:

$$N(x, y, 0, 0, 1, 1, \rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} e^{-\frac{1}{2(1-\rho^2)}(x^2 - 2\rho xy + y^2)} \tag{2}$$

2. Коэффициент корреляции Пирсона [6]:

$$r_{xy} = \left(\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})\right)\left(\sum_{i=1}^{n}(x_i - \overline{x})^2 \sum_{i=1}^{n}(y_i - \overline{y})^2\right)^{-\frac{1}{2}} \tag{3}$$

3. Коэффициент корреляции Спирмана [7]:

$$\rho_n = 1 - \frac{6}{n^3 - n}\sum_{i=1}^{n} d_i^2 \tag{4}$$

4. Квадрантный коэффициент корреляции [8]:

$$\hat{q} = \frac{1}{n}\sum_{i=1}^{n} sign(x_i - med\, x) sign(y_i - med\, y) \tag{5}$$

5 Реализация

Работы была выполнена на языке *Python*3.7. Для генерации выборок использовался модуль [1]. Для построения графиков использовалась библиотека matplotlib [2]. Функции распределения обрабатывались при помощи библиотеки scipy.stats [3]

6 Результаты

Рис. 1: Графики двумерного нормального распределения(2) при $p = 0.0$

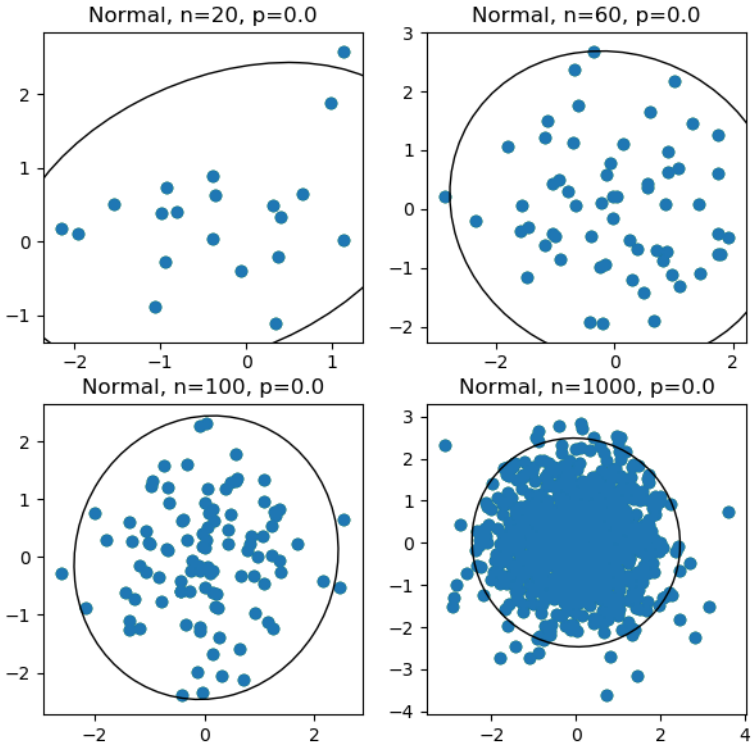

Таблица 1: Результаты для двумерного нормального распределения (2) при $p = 0.0$

Normal $n = 20$, $p = 0.0$			
	Pearson	Spearman	Quad
E	0.18892	0.14541	0.06000
E^2	0.05409	0.04186	0.02800
D	0.01840	0.02071	0.02440

Normal $n = 60$, $p = 0.0$			
	Pearson	Spearman	Quad
E	-0.04642	-0.05109	-0.03333
E^2	0.01080	0.00965	0.00667
D	0.00865	0.00704	0.00556

.

Normal $n = 100$, $p = 0.0$			
	Pearson	Spearman	Quad
E	-0.03469	-0.02805	-0.03200
E^2	0.00531	0.00539	0.00864
D	0.00411	0.00461	0.00762

Normal $n = 1000$, $p = 0.0$			
	Pearson	Spearman	Quad
E	0.00805	0.01039	0.00760
E^2	0.00094	0.00083	0.00094
D	0.00088	0.00073	0.00088

Рис. 2: Графики двумерного нормального распределения(2) при $p = 0.5$

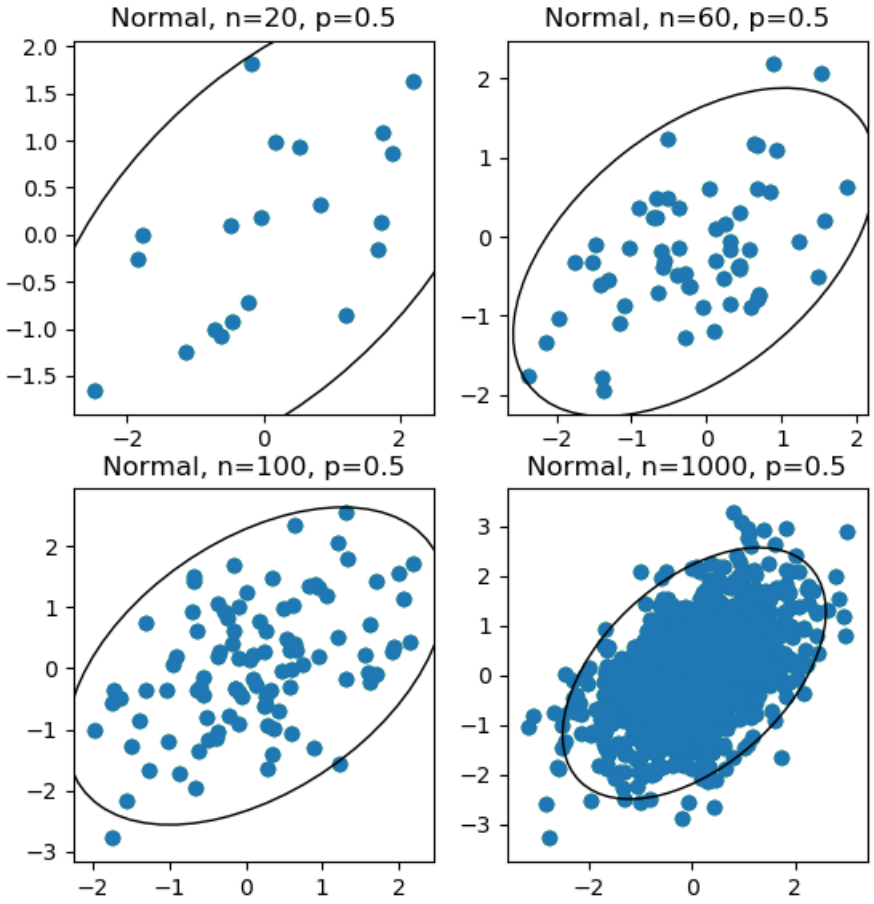

Таблица 2: Результаты для двумерного нормального распределения (2) при $p = 0.5$

Normal $n = 20$, $p = 0.5$			
	Pearson	Spearman	Quad
E	0.50363	0.52647	0.46000
E^2	0.27217	0.30152	0.22800
D	0.01853	0.02435	0.01640

Normal $n = 60$, $p = 0.5$			
	Pearson	Spearman	Quad
E	0.50847	0.47194	0.31333
E^2	0.26921	0.23710	0.12222
D	0.01067	0.01437	0.02404

.

Normal $n = 100$, $p = 0.5$			
	Pearson	Spearman	Quad
E	0.49628	0.47702	0.33200
E^2	0.25200	0.23489	0.12048
D	0.00570	0.00734	0.01026

Normal $n = 1000$, $p = 0.5$			
	Pearson	Spearman	Quad
E	0.49458	0.47938	0.33320
E^2	0.24515	0.23052	0.11207
D	0.00054	0.00071	0.00105

Рис. 3: График двумерного нормального распределения (2) при $p = 0.9$

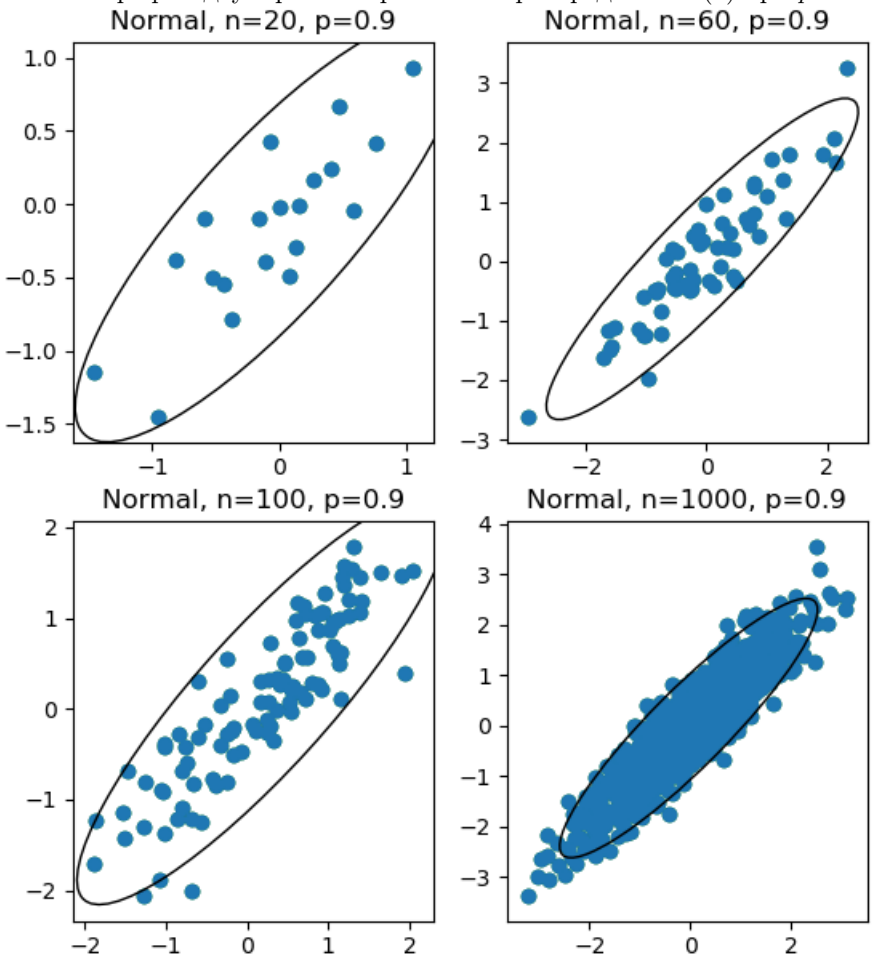

Таблица 3: Результаты для двумерного нормального распределения (2) при $p = 0.9$

Normal $n = 20$, $p = 0.9$			
	Pearson	Spearman	Quad
E	0.90154	0.85850	0.64000
E^2	0.81558	0.74275	0.44000
D	0.00281	0.00574	0.03040

Normal $n = 60$, $p = 0.9$			
	Pearson	Spearman	Quad
E	0.89761	0.88464	0.69333
E^2	0.80681	0.78457	0.48622
D	0.00112	0.00198	0.00551

.

Normal $n = 100$, $p = 0.9$			
	Pearson	Spearman	Quad
E	0.89624	0.88888	0.71600
E^2	0.80360	0.79094	0.51728
D	0.00037	0.00082	0.00462

Normal $n = 1000$, $p = 0.9$			
	Pearson	Spearman	Quad
E	0.89971	0.88953	0.71120
E^2	0.80951	0.79132	0.50603
D	0.00004	0.00005	0.00022

Рис. 4: Графики смеси двумерных нормальных распределений

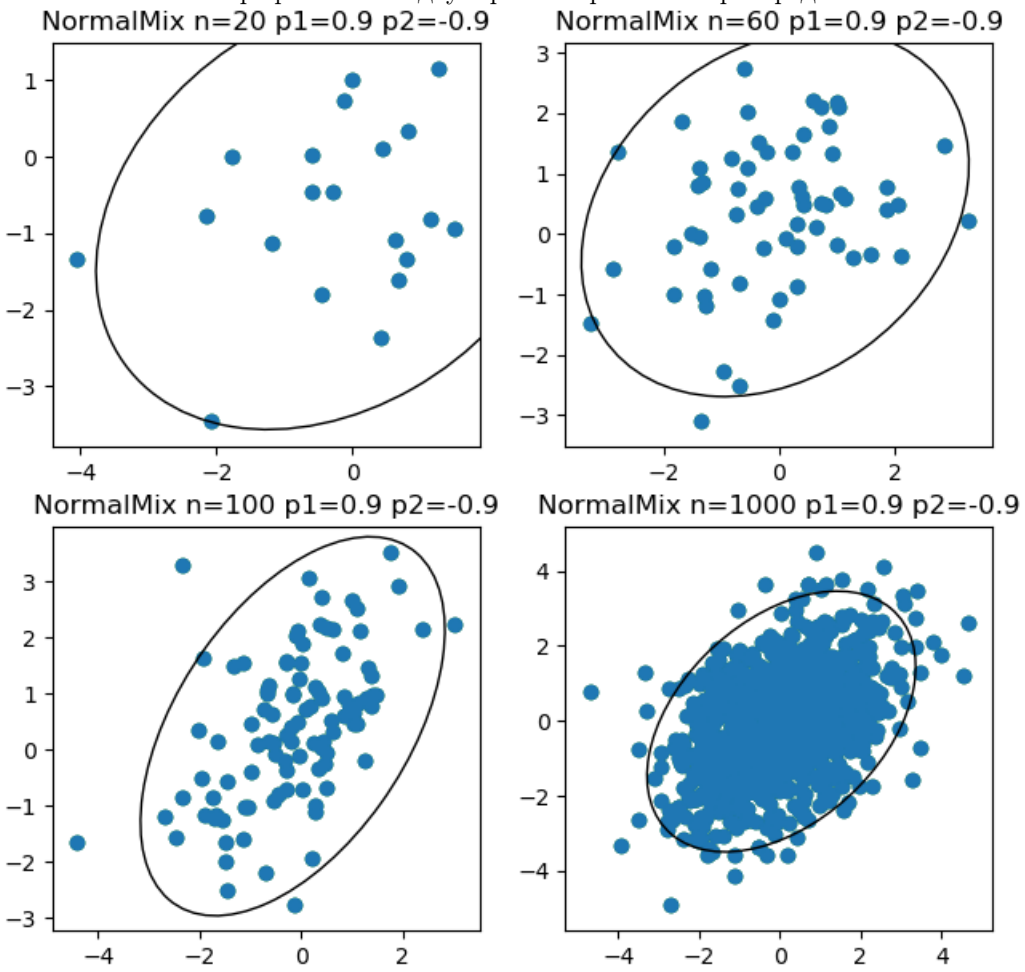

Таблица 4: Результаты для смеси двумерных нормальных распределений

NormalMix $n = 20,\ p_1 = 0.9,\ p_2 = -0.9$			
	Pearson	Spearman	Quad
E	0.90154	0.85850	0.64000
E^2	0.12784	0.14056	0.18000
D	0.03603	0.04226	0.11240

NormalMix $n = 60,\ p_1 = -0.9,\ p_2 = -0.9$			
	Pearson	Spearman	Quad
E	0.34444	0.33502	0.24000
E^2	0.13330	0.12744	0.08711
D	0.01466	0.01521	0.02951

NormalMix $n = 100,\ p_1 = 0.9,\ p_2 = -0.9$			
	Pearson	Spearman	Quad
E	0.42503	0.39751	0.26400
E^2	0.18615	0.16254	0.07584
D	0.00550	0.00453	0.00614

NormalMix $n = 1000,\ p_1 = 0.9,\ p_2 = -0.9$			
	Pearson	Spearman	Quad
E	0.38948	0.37427	0.25080
E^2	0.15242	0.14103	0.06380
D	0.00073	0.00095	0.00090

7 Выводы

По таблицам 1, 2, 3, 4, видно, что, при увеличении объёма выборки, подсчитанные коэффициенты корреляции стремятся к теоретическим.

Ближе всех к данному коэффициенту корреляции находится коэффициент Пирсона.

По графикам видно, что при уменьшении корреляции эллипс равновероятности стремится к окружности, а при увеличении растягивается.

8 Список литературы

[1] Модуль numpy - https://physics.susu.ru/vorontsov/language/numpy.html

[2] Модуль matplotlib - https://matplotlib.org/users/index.html

[3] Модуль scipy - https://docs.scipy.org/doc/scipy/reference/

[4] http://stu.sernam.ru/book_stat3.php?id=55

[5] Двумерное нормальное распределение: https://en.wikipedia.org/wiki/Multivariate_normal_distribution

[6] Коэффициент корреляции Пирса: http://statistica.ru/theory/koeffitsient-korrelyatsii/

[7] Коэффициент корреляции Спирмана: http://economic-definition.com/Exchange_Terminology/Koefficient_korrelyacii_Correlation_coefficient__eto.html

[8] Квадрантный коэффициент корреляции: https://www.researchgate.net/profile/Pavel_Smirnov8/publication/316973167_Robastnye_metody_i_algoritmy_ocenivania_korrelacionnyh_harakteristik_dannyh_na_osnove_novyh_vysokoeffektivnyh_i_bystryh_robastnyh_ocenok_masstaba/links/591b019d458515695282-8a52/Robastnye-metody-i-algoritmy-ocenivania-korrelacionnyh-harakteristik-dannyh-na-osnove-novyh-vysokoeffektivnyh-i-bystryh-robastnyh-ocenok-masstaba.pdf#page=81

9 Приложения

Код отчёта: https://github.com/MisterProper9000/MatStatLabs/blob/master/MatStatLab5/MatStatLab5.tex

Код лабораторной: https://github.com/MisterProper9000/MatStatLabs/blob/master/MatStatLab5/MatStatLab5.py

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
from matplotlib.patches import Ellipse
import sys

linalg = np.linalg

selection = [20, 60, 100, 1000]
cor = [0, 0.5, 0.9]




def E(x):
    return np.mean(x)
```

```
def D(x):
    return np.var(x)


def sqE(x):
    length = len(x)
    sum = 0
    for x_i in x:
        sum += x_i ** 2
    return sum / length


def pearson(x, y):
    res = stats.pearsonr(x, y)
    return res[0]


def spearman(x, y):
    res = stats.spearmanr(x, y)
    return res[0]


def quad(x, y):
    length = len(x)
    med_x = np.median(x)
    med_y = np.median(y)
    sum = 0
    for i in range(0, length):
        sum = sum + np.sign(x[i] - med_x) * np.sign(y[i] - med_y)
    return sum / length


cor_coef_dict = {
    "Pearson": pearson,
    "Spearman": spearman,
    "Quad": quad
}



def mix_normal_dist(p, N):
    cov1 = [[1, p], [p, 1]]
    cov2 = [[100, -p], [-p, 100]]
    mean = [0, 0]
    return 0.9 * np.random.multivariate_normal(mean, cov1, N) + 0.1 * np.
    random.multivariate_normal(mean, cov2, N)


def normal_dist(p, N):
    cov = [[1, p], [p, 1]]
    mean = [0, 0]
    return np.random.multivariate_normal(mean, cov, N)



dist_dict = {
    "Normal": normal_dist,
    "NormalMix": mix_normal_dist
}


def eigsorted(cov):
    vals, vecs = np.linalg.eigh(cov)
    order = vals.argsort()[::-1]
```

```
    return vals[order], vecs[:, order]


def dist_ellips(x, y, ax):
    nstd = 2.5
    #ax = plt.subplot(111)

    cov = np.cov(x, y)
    vals, vecs = eigsorted(cov)
    theta = np.degrees(np.arctan2(*vecs[:, 0][::-1]))
    w, h = 2 * nstd * np.sqrt(vals)
    ell = Ellipse(xy=(np.mean(x), np.mean(y)),
                  width=w, height=h,
                  angle=theta, color='black')
    ell.set_facecolor('none')
    ax.add_artist(ell)
    plt.scatter(x, y)



def research(p, N, dist, ax):
    pears = []
    spear = []
    quads = []
    for i in range(0, 10):
        data = dist(p, N)

        pears.append(pearson(data[:, 0], data[:, 1]))
        spear.append(spearman(data[:, 0], data[:, 1]))
        quads.append(quad(data[:, 0], data[:, 1]))

    plt.scatter(data[:, 0], data[:, 1], c='green')
    dist_ellips(data[:, 0], data[:, 1], ax)
    print("name;", end="")
    for name in cor_coef_dict:
        print("%-12s;" % name, end="")
    print()
    print("\t\tE   ;%.5lf;%.5lf;%.5lf" % (E(pears), E(spear), E(quads)))
    print("\t\tE^2 ;%.5lf;%.5lf;%.5lf" % (sqE(pears), sqE(spear), sqE(quads))
    )
    print("\t\tD   ;%.5lf;%.5lf;%.5lf" % (D(pears), D(spear), D(quads)))
    #print("\t\tE   : pears = %.5lf, spear = %.5lf, quad = %.5lf" % (E(pears)
    , E(spear), E(quads)))
    #print("\t\tE^2 : pears = %.5lf, spear = %.5lf, quad = %.5lf" % (sqE(
    pears), sqE(spear), sqE(quads)))
    #print("\t\tD   : pears = %.5lf, spear = %.5lf, quad = %.5lf" % (D(pears)
    , D(spear), D(quads)))
    print()



def draw(dist_name, p):
    plt.figure(p)
    sector = 1
    for N in selection:
        ax = plt.subplot(220 + sector)

        sector += 1
        if(dist_name == "NormalMix"):
            plt.title(dist_name + " n=%i" % N + " p1=%.1f" % p + " p2=%.1f" %
     (-p))
            print(dist_name + ", n=%i" % N + ", p1=%.1f" % p + ", p2=%.1f" %
    (-p))
        else:
            plt.title(dist_name + ", n=%i" % N + ", p=%.1f" % p)
            print(dist_name + ", n=%i" % N + ", p=%.1f" % p)
```

```
        research(p, N, dist_dict[dist_name], ax)

    plt.show()




f = open('out.csv', 'w')
sys.stdout = f

for p in cor:
    draw("Normal", p)
    print()

draw("NormalMix", 0.9)
```