НаціональнийтехнічнийуніверситетУкраїни

«КиївськийполітехнічнийінститутіменіІгоряСікорського»

Факультет інформатики та обчислювальноїтехніки

Кафедра обчислювальноїтехніки

**Методи оптимізації та планування експерименту**

Лабораторна робота №5:

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННІ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ КВАДРАТИЧНИХ ЧЛЕНІВ(ЦЕНТРАЛЬНИЙ ОРТОГОНАЛЬНИЙ КОМПОЗИЦІЙНИЙ ПЛАН)»

Виконав:

студент групи ІВ-82

Іващенко Павло

Залікова книжка № 8213

Перевірив Регіда П. Г.

Київ 2020р.

**Лабораторна робота №5**

**Тема:**ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННІ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ КВАДРАТИЧНИХ ЧЛЕНІВ(ЦЕНТРАЛЬНИЙ ОРТОГОНАЛЬНИЙ КОМПОЗИЦІЙНИЙ ПЛАН).

**Мета:**провести трьохфакторний експеремент з урахуваннямквадратичнихчленів, використовуючи центральний ортогональний композиційний план. Знайти рівняннярегресії адекватнее об'єкту.

**Виконання:**

Варіант – 211.



1. Лістинг програми:

import random as r  
import numpy as np  
import pprint  
from scipy.stats import t, f  
import sklearn.linear\_model as lm  
from functools import partial  
  
  
  
x\_range = [[-6, 10], [-3, 5], [-4, 9]]  
x\_sered\_max = sum([x[1] for x in x\_range])/3  
x\_sered\_min = sum([x[0] for x in x\_range])/3  
  
x01 = (x\_range[0][1] - x\_range[0][0]) / 2  
x02 = (x\_range[1][1] - x\_range[1][0]) / 2  
x03 = (x\_range[2][1] - x\_range[2][0]) / 2  
delta\_x1 = x\_range[0][1] - x01  
delta\_x2 = x\_range[1][1] - x02  
delta\_x3 = x\_range[2][1] - x03  
  
y\_max = 200 + x\_sered\_max  
y\_min = 200 + x\_sered\_min  
  
def create\_plan\_matrix(n, m):  
 y = np.zeros(shape=(n, m))  
 for i in range(n):  
 for j in range(m):  
 y[i][j] = r.randint(int(y\_min), int(y\_max))  
 x\_matrix\_norm = [  
 [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, 1, 1, 1],  
 [1, -1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0, 0],  
 [1, 1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0, 0],  
 [1, 0, -1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0],  
 [1, 0, 1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0],  
 [1, 0, 0, -1.215, 0, 0, 0, 0, 0, 0, 1.4623],  
 [1, 0, 0, 1.215, 0, 0, 0, 0, 0, 0, 1.4623],  
 [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]  
 ]  
 print(**'Нормована матриця:'**)  
 pprint.pprint(x\_matrix\_norm)  
  
 x\_matrix = [[] for x in range(n)]  
 for i in range(len(x\_matrix)):  
 if i < 8:  
 x1 = x\_range[0][0] if x\_matrix\_norm[i][1] == -1 else x\_range[0][1]  
 x2 = x\_range[1][0] if x\_matrix\_norm[i][2] == -1 else x\_range[1][1]  
 x3 = x\_range[2][0] if x\_matrix\_norm[i][3] == -1 else x\_range[2][1]  
 else:  
 x1 = x\_matrix\_norm[i][1] \* delta\_x1 + x01  
 x2 = x\_matrix\_norm[i][2] \* delta\_x2 + x02  
 x3 = x\_matrix\_norm[i][3] \* delta\_x3 + x03  
 x\_matrix[i] = [1, float(format(x1, **'.2f'**)),  
 float(format(x2, **'.2f'**)),  
 float(format(x3, **'.2f'**)),  
 float(format(x1 \* x2, **'.2f'**)),  
 float(format(x1 \* x3, **'.2f'**)),  
 float(format(x2 \* x3, **'.2f'**)),  
 float(format(x1 \* x2 \* x3, **'.2f'**)),  
 float(format(x1 \*\* 2, **'.2f'**)),  
 float(format(x2 \*\* 2, **'.2f'**)),  
 float(format(x3 \*\* 2, **'.2f'**))]  
 print(**'Натуралізована матриця: '**)  
 pprint.pprint(x\_matrix)  
  
 print(**'Y :'**)  
 pprint.pprint(y)  
  
 y\_avr = np.zeros(n)  
 for i in range(len(y)):  
 for j in range(len(y[0])):  
 y\_avr[i] += y[i][j]/ m  
 return [x\_matrix\_norm, x\_matrix, y, y\_avr]  
  
def find\_coefs(x, y):  
 skm = lm.LinearRegression(fit\_intercept=False) # знаходимо коефіцієнти рівняння регресії  
 skm.fit(x, y)  
 B = skm.coef\_  
 print(**'Коефіціенти: '**)  
 print(B)  
 return B  
  
def perevirka(x, y, b):  
 y\_pract = np.zeros(len(y))  
 for i in range(len(x)):  
 for j in range(len(x[0])):  
 y\_pract[i] += b[j] \* x[i][j]  
 print(**"Перевірка:"**)  
 print(**"y - real :"** , y)  
 print(**'y - found:'**, y\_pract)  
  
  
def get\_new\_y(x, b):  
 y\_pract = np.zeros(len(y))  
 for i in range(len(x)):  
 for j in range(len(x[0])):  
 y\_pract[i] += b[j] \* x[i][j]  
 return y\_pract  
  
def get\_cohren\_critical(prob, f1, f2):  
 f\_crit = f.isf((1 - prob) / f2, f1, (f2 - 1) \* f1)  
 return f\_crit / (f\_crit + f2 - 1)  
  
def cohren\_crit(y, n, m):  
 y\_var = [np.var(i) for i in y]  
 Gp = max(y\_var)/sum(y\_var)  
 Gt = get\_cohren\_critical(0.95, m-1, n)  
 if(Gp < Gt):  
 print(**"Дисперсії однорідні"**)  
 return True  
 else:  
 print(**"Дисперсії не однорідні"**)  
 return False  
  
fisher\_teor = partial(f.ppf, q=1 - 0.05)  
  
student\_teor = partial(t.ppf, q=1 - 0.025)  
  
def kriteriy\_studenta(x, y, y\_aver, n, m, B):  
 d = 0  
 y\_var = [np.var(i) for i in y]  
 s\_kv\_aver = sum(y\_var) / n  
 s\_aver = (s\_kv\_aver/(n \* m))\*\* 0.5  
 b = np.zeros(len(x[0]))  
 for i in range(len(x[0])):  
 for j in range(len(y\_aver)):  
 b[i] += y\_aver[j] \* x[j][i] / n  
 ts = []  
 for bi in b:  
 ts.append(bi/s\_aver)  
 Stud\_teor = student\_teor(df = (m-1) \* n)  
 for i in range(len(ts)):  
 if ts[i] < Stud\_teor:  
 B[i] = 0  
 else:  
 d += 1  
 print(**"Коефіціенти після перевірки нуль гіпотези: "**)  
 print(B)  
 return [B, d]  
  
def kriteriy\_fishera(m, n, d, new\_y\_pract, y\_avr, y):  
 f4 = n - d  
 f3 = (m-1)\*n  
 y\_var = [np.var(i) for i in y]  
 Sa = (sum(y\_var)/ n)  
 Sad = m/(n-d)\* sum([(new\_y\_pract[i] - y\_avr[i])\*\*2 for i in range(len(y\_avr))])  
 pract = Sad/Sa  
 teor = fisher\_teor(dfn= f4, dfd=f3)  
 if pract > teor:  
 print(**"Практичне значення:"** , pract)  
 print(**"Теоретичне значення:"** , teor)  
 print(**"Рівняння регресії неадекватне"**)  
 return [False, False]  
 else:  
 print(**"Рівняння регресії адекватне"**)  
 return [True, True]  
  
  
  
  
if \_\_name\_\_ == **"\_\_main\_\_"**:  
 odnorid = False  
 adekvat = False  
 n = 15  
 m = 3  
 while not adekvat:  
 while not odnorid:  
 x\_matrix\_norm, x\_matrix, y, y\_avr = create\_plan\_matrix(n, m)  
 odnorid = cohren\_crit(y, n, m)  
 if odnorid == False:  
 m+=1  
 B = find\_coefs(x\_matrix, y\_avr)  
 perevirka(x\_matrix, y\_avr, B)  
 new\_B, d = kriteriy\_studenta(x\_matrix\_norm, y, y\_avr, n, m, B)  
 new\_y\_pract = get\_new\_y(x\_matrix, new\_B)  
 print(new\_y\_pract, y\_avr)  
 adekvat, odnorid = kriteriy\_fishera(m, n, 4, new\_y\_pract, y\_avr, y)

1. Результат виконання роботи програми:

/usr/bin/python3.7 /home/pavel/lab5\_mope/main.py

Нормована матриця:

[[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, 1, 1, 1],

[1, -1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0, 0],

[1, 1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0, 0],

[1, 0, -1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0],

[1, 0, 1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0],

[1, 0, 0, -1.215, 0, 0, 0, 0, 0, 0, 1.4623],

[1, 0, 0, 1.215, 0, 0, 0, 0, 0, 0, 1.4623],

[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]

Натуралізована матриця:

[[1, -6.0, -3.0, -4.0, 18.0, 24.0, 12.0, -72.0, 36.0, 9.0, 16.0],

[1, -6.0, -3.0, 9.0, 18.0, -54.0, -27.0, 162.0, 36.0, 9.0, 81.0],

[1, -6.0, 5.0, -4.0, -30.0, 24.0, -20.0, 120.0, 36.0, 25.0, 16.0],

[1, -6.0, 5.0, 9.0, -30.0, -54.0, 45.0, -270.0, 36.0, 25.0, 81.0],

[1, 10.0, -3.0, -4.0, -30.0, -40.0, 12.0, 120.0, 100.0, 9.0, 16.0],

[1, 10.0, -3.0, 9.0, -30.0, 90.0, -27.0, -270.0, 100.0, 9.0, 81.0],

[1, 10.0, 5.0, -4.0, 50.0, -40.0, -20.0, -200.0, 100.0, 25.0, 16.0],

[1, 10.0, 5.0, 9.0, 50.0, 90.0, 45.0, 450.0, 100.0, 25.0, 81.0],

[1, 5.57, 4.0, 6.5, 22.28, 36.2, 26.0, 144.82, 31.02, 16.0, 42.25],

[1, 10.43, 4.0, 6.5, 41.72, 67.8, 26.0, 271.18, 108.78, 16.0, 42.25],

[1, 8.0, 2.79, 6.5, 22.28, 52.0, 18.1, 144.82, 64.0, 7.76, 42.25],

[1, 8.0, 5.21, 6.5, 41.72, 52.0, 33.9, 271.18, 64.0, 27.2, 42.25],

[1, 8.0, 4.0, 3.46, 32.0, 27.7, 13.85, 110.8, 64.0, 16.0, 11.99],

[1, 8.0, 4.0, 9.54, 32.0, 76.3, 38.15, 305.2, 64.0, 16.0, 90.96],

[1, 8.0, 4.0, 6.5, 32.0, 52.0, 26.0, 208.0, 64.0, 16.0, 42.25]]

Y :

array([[197., 204., 201.],

[204., 195., 204.],

[199., 207., 198.],

[200., 203., 206.],

[204., 198., 198.],

[202., 208., 203.],

[207., 196., 201.],

[195., 200., 198.],

[202., 199., 205.],

[196., 205., 196.],

[204., 199., 208.],

[205., 205., 206.],

[199., 196., 207.],

[197., 201., 202.],

[205., 199., 202.]])

Дисперсії однорідні

Коефіціенти:

[ 2.03689485e+02 2.42047273e-01 -3.28500618e-01 3.19140526e-01

-1.51529276e-02 5.27215729e-03 -1.63220868e-02 -4.94002021e-03

-6.24320604e-02 1.83682010e-01 -4.42620556e-02]

Перевірка:

y - real : [200.66666667 201. 201.33333333 203. 200.

204.33333333 201.33333333 197.66666667 202. 199.

203.66666667 205.33333333 200.66666667 200. 202. ]

y - found: [200.6571278 200.99828941 201.26919844 202.99543563 199.97567087

204.49601381 201.17745713 198.86268201 203.6437541 199.21319313

201.26911017 202.86823412 202.71637527 200.06063534 201.79682277]

Коефіціенти після перевірки нуль гіпотези:

[ 2.03689485e+02 0.00000000e+00 0.00000000e+00 0.00000000e+00

0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00

-6.24320604e-02 1.83682010e-01 -4.42620556e-02]

[202.38687619 199.50984258 205.32578835 202.44875474 198.39122432

195.51419071 201.33013649 198.45310287 202.82168296 197.96696594

199.24913385 202.81991213 202.10204341 198.60666888 200.76267361] [200.66666667 201. 201.33333333 203. 200.

204.33333333 201.33333333 197.66666667 202. 199.

203.66666667 205.33333333 200.66666667 200. 202. ]

Практичне значення: 3.513648427671494

Теоретичне значення: 2.125558760875511

Рівняння регресії неадекватне

Нормована матриця:

[[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, 1, 1, 1],

[1, -1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0, 0],

[1, 1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0, 0],

[1, 0, -1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0],

[1, 0, 1.215, 0, 0, 0, 0, 0, 0, 1.4623, 0],

[1, 0, 0, -1.215, 0, 0, 0, 0, 0, 0, 1.4623],

[1, 0, 0, 1.215, 0, 0, 0, 0, 0, 0, 1.4623],

[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]

Натуралізована матриця:

[[1, -6.0, -3.0, -4.0, 18.0, 24.0, 12.0, -72.0, 36.0, 9.0, 16.0],

[1, -6.0, -3.0, 9.0, 18.0, -54.0, -27.0, 162.0, 36.0, 9.0, 81.0],

[1, -6.0, 5.0, -4.0, -30.0, 24.0, -20.0, 120.0, 36.0, 25.0, 16.0],

[1, -6.0, 5.0, 9.0, -30.0, -54.0, 45.0, -270.0, 36.0, 25.0, 81.0],

[1, 10.0, -3.0, -4.0, -30.0, -40.0, 12.0, 120.0, 100.0, 9.0, 16.0],

[1, 10.0, -3.0, 9.0, -30.0, 90.0, -27.0, -270.0, 100.0, 9.0, 81.0],

[1, 10.0, 5.0, -4.0, 50.0, -40.0, -20.0, -200.0, 100.0, 25.0, 16.0],

[1, 10.0, 5.0, 9.0, 50.0, 90.0, 45.0, 450.0, 100.0, 25.0, 81.0],

[1, 5.57, 4.0, 6.5, 22.28, 36.2, 26.0, 144.82, 31.02, 16.0, 42.25],

[1, 10.43, 4.0, 6.5, 41.72, 67.8, 26.0, 271.18, 108.78, 16.0, 42.25],

[1, 8.0, 2.79, 6.5, 22.28, 52.0, 18.1, 144.82, 64.0, 7.76, 42.25],

[1, 8.0, 5.21, 6.5, 41.72, 52.0, 33.9, 271.18, 64.0, 27.2, 42.25],

[1, 8.0, 4.0, 3.46, 32.0, 27.7, 13.85, 110.8, 64.0, 16.0, 11.99],

[1, 8.0, 4.0, 9.54, 32.0, 76.3, 38.15, 305.2, 64.0, 16.0, 90.96],

[1, 8.0, 4.0, 6.5, 32.0, 52.0, 26.0, 208.0, 64.0, 16.0, 42.25]]

Y :

array([[197., 205., 196.],

[208., 198., 203.],

[196., 201., 196.],

[196., 198., 204.],

[197., 201., 206.],

[203., 207., 197.],

[203., 201., 205.],

[206., 199., 207.],

[195., 203., 205.],

[199., 208., 202.],

[198., 197., 201.],

[197., 202., 198.],

[195., 204., 200.],

[197., 202., 198.],

[204., 203., 199.]])

Дисперсії однорідні

Коефіціенти:

[ 1.96597526e+02 -4.80588256e-02 -2.50932898e-01 1.61478905e-01

2.69946884e-02 -1.16242788e-02 -1.57568666e-02 5.75316264e-04

5.00245122e-02 6.55377550e-02 -1.18893619e-03]

Перевірка:

y - real : [199.33333333 203. 197.66666667 199.33333333 201.33333333

202.33333333 203. 204. 201. 203.

198.66666667 199. 199.66666667 199. 202. ]

y - found: [199.340878 203.01865846 197.70095431 199.38102329 201.33217509

202.23310822 202.85300959 202.7135575 198.78013325 202.66661992

200.01757206 201.03288362 200.39130847 200.44321959 200.42823196]

Коефіціенти після перевірки нуль гіпотези:

[ 1.96597526e+02 0.00000000e+00 0.00000000e+00 0.00000000e+00

0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00

5.00245122e-02 6.55377550e-02 -1.18893619e-03]

[198.96922544 198.89194459 200.01782952 199.94054867 202.17079422

202.09351337 203.2193983 203.14211745 199.14765808 203.03756415

200.25743539 201.53148935 200.8334437 200.73955341 200.79746649] [199.33333333 203. 197.66666667 199.33333333 201.33333333

202.33333333 203. 204. 201. 203.

198.66666667 199. 199.66666667 199. 202. ]

Рівняння регресії адекватне

**Висновок:**Отже, у ході виконання лабораторної роботи № 5 провели трьохфакторний експеримент при використанні рівняння з урахуванням квадратичних членів. Склали матрицю планування, знайшли коефіцієнти рівняння регресії, провели 3 статистичні перевірки.Була написана текстова програма, результати наведені вище. Результати співпадають із калькулятором. Кінцева мета роботи досягнута!