Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №3

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

ВИКОНАЛА:
студентка II курсу ФІОТ
групи ІВ-81
Яшан Оксана
Залікова – 8130
№ у списку: 29

ПЕРЕВІРИВ:
Регіда П.Г.

Київ – 2020

Варіант:

129	-25	75	-20	40	-20	-15

Код:

```
import scipy.stats
import numpy as np

m = 3
x1_min = -25
x1_max = 75
x2_min = -20
x2_max = 40
x3_min = -20
x3_max = -15

mx_min = (x1_min+x2_min+x3_min)/3
mx_max = (x1_max+x2_max+x3_max)/3
y_max = 200+mx_max
y_min = 200+mx_min
#створюємо array розміром (4,3)
#y_i = np.random.randint(y_min,y_max,(4,3))

matrix = np.ndarray(shape=(4,7),dtype = float)
matrix[0][0],matrix[1][0],matrix[2][0],matrix[3][0] = 1,1,1,1
matrix[0][1],matrix[1][1],matrix[2][1],matrix[3][1] = -1,-1,1,1
matrix[0][2],matrix[1][2],matrix[2][2],matrix[3][2] = -1,1,-1,1
matrix[0][3],matrix[1][3],matrix[2][3],matrix[3][3] = -1,1,1,-1


matrix_n = np.ndarray(shape=(4,6),dtype = float)
matrix_n[0][0],matrix_n[1][0],matrix_n[2][0],matrix_n[3][0] = x1_min,x1_min,x1_max,x1_max
matrix_n[0][1],matrix_n[1][1],matrix_n[2][1],matrix_n[3][1] = x2_min,x2_max,x1_min,x2_max
matrix_n[0][2],matrix_n[1][2],matrix_n[2][2],matrix_n[3][2] = x3_min,x3_max,x3_max,x3_min

mY_list = []
for i in range(4):
    for j in range(3,6):
        r = np.random.randint(y_min,y_max)
        matrix_n[i][j],matrix[i][j+1] = r,r
    mY_list.append(((matrix_n[i][3]+matrix_n[i][4]+matrix_n[i][5])/3).__round__(4))
mx1 = np.sum(matrix_n,axis=0)[0]/4
mx2 = np.sum(matrix_n,axis=0)[1]/4
mx3 = np.sum(matrix_n,axis=0)[2]/4
my = (sum(mY_list)/len(mY_list)).__round__(2)

a1 =
(matrix_n[0][0]*mY_list[0]+matrix_n[1][0]*mY_list[1]+matrix_n[2][0]*mY_list[2]+matrix_n[3][0]*mY_l
ist[3])/4
a2 =
(matrix_n[0][1]*mY_list[0]+matrix_n[1][1]*mY_list[1]+matrix_n[2][1]*mY_list[2]+matrix_n[3][1]*mY_l
ist[3])/4
a3 =
(matrix_n[0][2]*mY_list[0]+matrix_n[1][2]*mY_list[1]+matrix_n[2][2]*mY_list[2]+matrix_n[3][2]*mY_l
ist[3])/4

a11 = (matrix_n[0][0]**2+matrix_n[1][0]**2+matrix_n[2][0]**2+matrix_n[3][0]**2)/4
a22 = (matrix_n[0][1]**2+matrix_n[1][1]**2+matrix_n[2][1]**2+matrix_n[3][1]**2)/4
```

```
a33 = (matrix_n[0][2]**2+matrix_n[1][2]**2+matrix_n[2][2]**2+matrix_n[3][2]**2)/4

a12 = a21 =
(matrix_n[0][0]*matrix_n[0][1]+matrix_n[1][0]*matrix_n[1][1]+matrix_n[2][0]*matrix_n[2][1]+matrix_
n[3][0]*matrix_n[3][1])/4
a13 = a31 =
(matrix_n[0][0]*matrix_n[0][2]+matrix_n[1][0]*matrix_n[1][2]+matrix_n[2][0]*matrix_n[2][2]+matrix_
n[3][0]*matrix_n[3][2])/4
a23 = a32 =
(matrix_n[0][1]*matrix_n[0][2]+matrix_n[1][1]*matrix_n[1][2]+matrix_n[2][1]*matrix_n[2][2]+matrix_
n[3][1]*matrix_n[3][2])/4

b0 = np.linalg.det(np.array([[my,mx1,mx2,mx3],
[a1,a11,a12,a13],[a2,a12,a22,a32],[a3,a13,a23,a33]]))/np.linalg.det(np.array([[1,mx1,mx2,mx3],
[mx1,a11,a12,a13],[mx2,a12,a22,a32],[mx3,a13,a23,a33]]))
b1 = np.linalg.det(np.array([[1,my,mx2,mx3],
[mx1,a1,a12,a13],[mx2,a2,a22,a32],[mx3,a3,a23,a33]]))/np.linalg.det(np.array([[1,mx1,mx2,mx3],
[mx1,a11,a12,a13],[mx2,a12,a22,a32],[mx3,a13,a23,a33]]))
b2 = np.linalg.det(np.array([[1,mx1,my,mx3],
[mx1,a11,a1,a13],[mx2,a12,a2,a32],[mx3,a13,a3,a33]]))/np.linalg.det(np.array([[1,mx1,mx2,mx3],
[mx1,a11,a12,a13],[mx2,a12,a22,a32],[mx3,a13,a23,a33]]))
b3 = np.linalg.det(np.array([[1,mx1,mx2,my],
[mx1,a11,a12,a1],[mx2,a12,a22,a2],[mx3,a13,a23,a3]]))/np.linalg.det(np.array([[1,mx1,mx2,mx3],
[mx1,a11,a12,a13],[mx2,a12,a22,a32],[mx3,a13,a23,a33]]))
print("    Матриця планування")
print("   x1      x2     x3      y1      y2      y3     ")
for i in range(3):
    for j in range(6):
        print("{:>6.1f}".format(matrix_n[i][j]), end="  ")
    print("\t")
print("\n","y = %.2f + %.2f * x1 + %.2f * x2+ %.2f * x3" %(b0,b1,b2,b3))
print("\nПеревірка:")
print((b0+b1*matrix_n[0][0]+b2*matrix_n[0][1]+b3*matrix_n[0][2]).__round__(3),"
",(b0+b1*matrix_n[1][0]+b2*matrix_n[1][1]+b3*matrix_n[1][2]).__round__(3),"   ",
(b0+b1*matrix_n[2][0]+b2*matrix_n[2][1]+b3*matrix_n[2][2]).__round__(3),"
",(b0+b1*matrix_n[3][0]+b2*matrix_n[3][1]+b3*+matrix_n[3][2]).__round__(3))

print(mY_list)


#Перевірка за Кохреном:
s2_y1 = ((matrix[0][4]-mY_list[0])**2+(matrix[0][5]-mY_list[0])**2+(matrix[0][6]-mY_list[0])**2)/3
s2_y2 = ((matrix[1][4]-mY_list[1])**2+(matrix[1][5]-mY_list[1])**2+(matrix[1][6]-mY_list[1])**2)/3
s2_y3 = ((matrix[2][4]-mY_list[2])**2+(matrix[2][5]-mY_list[2])**2+(matrix[2][6]-mY_list[2])**2)/3
s2_y4 = ((matrix[3][4]-mY_list[3])**2+(matrix[3][5]-mY_list[3])**2+(matrix[3][6]-mY_list[3])**2)/3

Gp = max(s2_y1,s2_y2,s2_y3,s2_y4)/(s2_y1+s2_y2+s2_y3+s2_y4)
q = 0.05
Gt = 0.7679
if(Gp < Gt):
    print(" Отже, дисперсія однорідна")
else:
    print("Дисперсія не однорідна - змініть m")

#Критерій Стьюдента
s2_b = (s2_y1+s2_y2+s2_y3+s2_y4)/4
s2_bb = s2_b/(4*m)
s_bb = np.sqrt(s2_bb)
n = 4
bb0 =
(mY_list[0]*matrix[0][0]+mY_list[1]*matrix[1][0]+mY_list[2]*matrix[2][0]+mY_list[3]*matrix[3][0])/
```

```
n
bb1 =
(mY_list[0]*matrix[0][1]+mY_list[1]*matrix[1][1]+mY_list[2]*matrix[2][1]+mY_list[3]*matrix[3][1])/
n
bb2 =
(mY_list[0]*matrix[0][2]+mY_list[1]*matrix[1][2]+mY_list[2]*matrix[2][2]+mY_list[3]*matrix[3][2])/
n
bb3 =
(mY_list[0]*matrix[0][3]+mY_list[1]*matrix[1][3]+mY_list[2]*matrix[2][3]+mY_list[3]*matrix[3][3])/
n

t = [abs(bb0)/s_bb,abs(bb1)/s_bb,abs(bb2)/s_bb,abs(bb3)/s_bb]

f3 = (m-1)*n
# t_t = 2.306  # для значення f3 = 8, t табличне = 2,306
t_t = scipy.stats.t.ppf((1 + (1-q))/2, f3)
print("\nt табличне:", t_t)

if t[0] < t_t:
    b0 = 0
    print("t0<t_t; b0=0")
if t[1] < t_t:
    b1 = 0
    print("t1<t_t; b1=0")
if t[2] < t_t:
    b2 = 0
    print("t2<t_t; b2=0")
if t[3] < t_t:
    b3 = 0
    print("t3<t_t; b3=0")

print("\n","y = %.2f + %.2f * x1 + %.2f * x2+ %.2f * x3" %(b0,b1,b2,b3))
y1_exp = b0 + b1*matrix_n[0][0] + b2*matrix_n[0][1] + b3*matrix_n[0][2]
y2_exp = b0 + b1*matrix_n[1][0] + b2*matrix_n[1][1] + b3*matrix_n[1][2]
y3_exp = b0 + b1*matrix_n[2][0] + b2*matrix_n[2][1] + b3*matrix_n[2][2]
y4_exp = b0 + b1*matrix_n[3][0] + b2*matrix_n[3][1] + b3*matrix_n[3][2]

print(f"y1_exp = {b0:.2f}{b1:+.2f}*x11{b2:+.2f}*x12{b3:+.2f}*x13 "
      f"= {y1_exp:.2f}")
print(f"y2_exp = {b0:.2f}{b1:+.2f}*x21{b2:+.2f}*x22{b3:+.2f}*x23"
      f" = {y2_exp:.2f}")
print(f"y3_exp = {b0:.2f}{b1:+.2f}*x31{b2:+.2f}*x32{b3:+.2f}*x33 "
      f"= {y3_exp:.2f}")
print(f"y4_exp = {b0:.2f}{b1:+.2f}*x41{b2:+.2f}*x42{b3:+.2f}*x43"
      f" = {y4_exp:.2f}")

#Критерій Фішера
d = 2
f4 = n-d
s2_ad = ((y1_exp-mY_list[0])**2+(y2_exp-mY_list[1])**2+(y3_exp-mY_list[2])**2+(y4_exp-
mY_list[3])**2)/(m/n-d)

Fp = s2_ad/s2_b
Ft = scipy.stats.f.ppf(1-q, f4, f3)
print("\nFp:", Fp)
print("Ft:", Ft)
if Fp > Ft:
    print("Рівняння регресії не адекватно оригіналу при q = 0,05")
else:
    print("Рівняння регресії адекватно оригіналу при q = 0,05")
```

Результат:

```
    Матриця планування
   x1      x2      x3       y1       y2       y3
 -25.0   -20.0   -20.0   189.0   191.0   229.0
 -25.0    40.0   -15.0   210.0   196.0   220.0
  75.0   -25.0   -15.0   197.0   207.0   208.0

 y = 214.01 + -0.01 * x1 + 0.05 * x2+ 0.52 * x3

Перевірка:
203.018     208.638     203.975     204.689
[203.0, 208.6667, 204.0, 204.6667]
 Отже, дисперсія однорідна

t табличне: 2.3060041350333704
t1<t_t; b1=0
t2<t_t; b2=0
t3<t_t; b3=0

 y = 214.01 + 0.00 * x1 + 0.00 * x2+ 0.00 * x3
y1_exp = 214.01+0.00*x11+0.00*x12+0.00*x13 = 214.01
y2_exp = 214.01+0.00*x21+0.00*x22+0.00*x23 = 214.01
y3_exp = 214.01+0.00*x31+0.00*x32+0.00*x33 = 214.01
y4_exp = 214.01+0.00*x41+0.00*x42+0.00*x43 = 214.01

Fp: -1.6933941526154557
Ft: 4.458970107524511
Рівняння регресії адекватно оригіналу при q = 0,05
```