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

з дисципліни «Методи оптимізації та планування експерименту»

Виконав:

студент ІІ курсу ФІОТ

групи ІО-82

Рикун Олег Віталійович

Залікова книжка № 8218

Варіант: 216

Перевірив:

ст. вик.

Регіда П. Г.

Київ – 2020

Код програми

from random import randint

import math

from numpy import linalg

mas\_y = []

ser\_func\_vid = []

mas\_a = []

mas\_mx = []

s =[]

x1\_min = -10 + 1 # додав одиницю для того, щоб отримати ціле число для генерації масиву

x1\_max = 50 - 1 # відняв одиницю для того, щоб отримати ціле число для генерації масиву

x2\_min = 25

x2\_max = 65

x3\_min = -10

x3\_max = 15

n = 4

m = 3

mas\_x = [[x1\_min, x2\_min, x3\_min],

[x1\_min, x2\_max, x3\_max],

[x1\_max, x2\_min, x2\_max],

[x1\_max, x2\_max, x3\_min]]

# for i in range(n):

# print(mas\_x[i])

y\_min = 200 + (x1\_min + x2\_min + x3\_min) / 3

y\_max = 200 + (x1\_max + x2\_max + x3\_max) / 3

print("Ymin = {}".format(y\_min))

print("Ymax = {}".format(y\_max) + "\n")

for i in range(n):

b = []

mas\_y.append(b)

for j in range(m):

b.append(randint(y\_min, y\_max))

mas\_y.append(b)

ser\_func\_vid.append(sum(b) / 3)

print("Матриця:")

for i in range(n):

print(mas\_y[i])

print("\n" + "Середні значення функції відгуку y: ")

for i in range(n):

print(ser\_func\_vid[i])

print("\n" + "Середні значення функції відгуку mx: ")

mx1 = (mas\_x[0][0] + mas\_x[1][0] + mas\_x[2][0] + mas\_x[3][0]) / 4

mx2 = (mas\_x[0][1] + mas\_x[1][1] + mas\_x[2][1] + mas\_x[3][1]) / 4

mx3 = (mas\_x[0][2] + mas\_x[1][2] + mas\_x[2][2] + mas\_x[3][2]) / 4

print("mx1: {}".format(mx1))

print("mx2: {}".format(mx2))

print("mx3: {}".format(mx3) + "\n")

my = sum(ser\_func\_vid) / 4

print("Середнє значення функції відгуку my: {}".format(my) + "\n")

print("Числа для знаходження коефіціентів регресійного рівняння:")

a1 = (mas\_x[0][0] \* ser\_func\_vid[0] + mas\_x[1][0] \* ser\_func\_vid[1] + mas\_x[2][0] \* ser\_func\_vid[2] + mas\_x[3][0] \* ser\_func\_vid[3]) / 4

a2 = (mas\_x[0][1] \* ser\_func\_vid[0] + mas\_x[1][1] \* ser\_func\_vid[1] + mas\_x[2][1] \* ser\_func\_vid[2] + mas\_x[3][1] \* ser\_func\_vid[3]) / 4

a3 = (mas\_x[0][2] \* ser\_func\_vid[0] + mas\_x[1][2] \* ser\_func\_vid[1] + mas\_x[2][2] \* ser\_func\_vid[2] + mas\_x[3][2] \* ser\_func\_vid[3]) / 4

print("a1: {}".format(a1))

print("a2: {}".format(a2))

print("a3: {}".format(a3))

a11 = (math.pow(mas\_x[0][0], 2) + math.pow(mas\_x[1][0], 2) + math.pow(mas\_x[2][0], 2) + math.pow(mas\_x[3][0], 2)) / 4

a22 = (math.pow(mas\_x[0][1], 2) + math.pow(mas\_x[1][1], 2) + math.pow(mas\_x[2][1], 2) + math.pow(mas\_x[3][1], 2)) / 4

a33 = (math.pow(mas\_x[0][2], 2) + math.pow(mas\_x[1][2], 2) + math.pow(mas\_x[2][2], 2) + math.pow(mas\_x[3][2], 2)) / 4

print("a11: {}".format(a11))

print("a22: {}".format(a22))

print("a33: {}".format(a33))

a12 = (mas\_x[0][0]\*mas\_x[0][1] + mas\_x[1][0]\*mas\_x[1][1] + mas\_x[2][0]\*mas\_x[2][1] + mas\_x[3][0]\*mas\_x[3][1]) / 4

a13 = (mas\_x[0][0]\*mas\_x[0][2] + mas\_x[1][0]\*mas\_x[1][2] + mas\_x[2][0]\*mas\_x[2][2] + mas\_x[3][0]\*mas\_x[3][2]) / 4

a23 = (mas\_x[0][1]\*mas\_x[0][2] + mas\_x[1][1]\*mas\_x[1][2] + mas\_x[2][1]\*mas\_x[2][2] + mas\_x[3][1]\*mas\_x[3][2]) / 4

print("a12: {}".format(a12))

print("a23: {}".format(a23))

print("a13: {}".format(a13) + "\n")

mas\_znam = [[1, mx1, mx2, mx3],

[mx1, a11, a12, a13],

[mx2, a12, a22, a23],

[mx3, a13, a23, a33]]

mas\_chis0 = [[my, mx1, mx2, mx3],

[a1, a11, a12, a13],

[a2, a12, a22, a23],

[a3, a13, a23, a33]]

mas\_chis1 = [[1, my, mx2, mx3],

[mx1, a1, a12, a13],

[mx2, a2, a22, a23],

[mx3, a3, a23, a33]]

mas\_chis2 = [[1, mx1, my, mx3],

[mx1, a11, a1, a13],

[mx2, a12, a2, a23],

[mx3, a13, a3, a33]]

mas\_chis3 = [[1, mx1, mx2, my],

[mx1, a11, a12, a1],

[mx2, a12, a22, a2],

[mx3, a13, a23, a3]]

znam = linalg.det(mas\_znam)

chis0 = linalg.det(mas\_chis0)

chis1 = linalg.det(mas\_chis1)

chis2 = linalg.det(mas\_chis2)

chis3 = linalg.det(mas\_chis3)

b0 = chis0 / znam

b1 = chis1 / znam

b2 = chis2 / znam

b3 = chis3 / znam

print("Коефіціенти b0, b1, b2 та b3:")

print("b0: {}".format(b0))

print("b1: {}".format(b1))

print("b2: {}".format(b2))

print("b3: {}".format(b3) + "\n")

print("Перевірка:")

y1 = b0 + b1 \* mas\_x[0][0] + b2 \* mas\_x[0][1] + b3 \* mas\_x[0][2]

y2 = b0 + b1 \* mas\_x[1][0] + b2 \* mas\_x[1][1] + b3 \* mas\_x[1][2]

y3 = b0 + b1 \* mas\_x[2][0] + b2 \* mas\_x[2][1] + b3 \* mas\_x[2][2]

y4 = b0 + b1 \* mas\_x[3][0] + b2 \* mas\_x[3][1] + b3 \* mas\_x[3][2]

print("y1: {}".format(y1))

print("y2: {}".format(y2))

print("y3: {}".format(y3))

print("y4: {}".format(y4) + "\n")

print("Перевірка одноріності за критерієм Кохрена:")

print("Дисперсії по рядках:")

s\_y1 = ((mas\_y[0][0] - ser\_func\_vid[0])\*\*2 + (mas\_y[0][1] - ser\_func\_vid[0])\*\*2 + (mas\_y[0][2] - ser\_func\_vid[0])\*\*2) / 3

s\_y2 = ((mas\_y[1][0] - ser\_func\_vid[1])\*\*2 + (mas\_y[1][1] - ser\_func\_vid[1])\*\*2 + (mas\_y[1][2] - ser\_func\_vid[1])\*\*2) / 3

s\_y3 = ((mas\_y[2][0] - ser\_func\_vid[2])\*\*2 + (mas\_y[2][1] - ser\_func\_vid[2])\*\*2 + (mas\_y[2][2] - ser\_func\_vid[2])\*\*2) / 3

s\_y4 = ((mas\_y[3][0] - ser\_func\_vid[3])\*\*2 + (mas\_y[3][1] - ser\_func\_vid[3])\*\*2 + (mas\_y[3][2] - ser\_func\_vid[3])\*\*2) / 3

s.append(s\_y1)

s.append(s\_y2)

s.append(s\_y3)

s.append(s\_y4)

print("s\_y1: {}".format(s\_y1))

print("s\_y2: {}".format(s\_y2))

print("s\_y3: {}".format(s\_y3))

print("s\_y4: {}".format(s\_y4) + "\n")

gp = max(s) / sum(s)

gt = 0.7679

print("Отримане значення Gp: {}".format(gp))

if gp < gt:

print("Дисперсія однорідна.\n")

else:

print("Дисперсія неоднорідна.\n")

print("Перевірка значимості коефіціентів за критерієм Стьюдента:")

s\_b = sum(s) / 4

s\_beta = math.sqrt(s\_b / 12)

beta0 = (ser\_func\_vid[0] \* 1 + ser\_func\_vid[1] \* 1 + ser\_func\_vid[2] \* 1 + ser\_func\_vid[3] \* 1) / 4

beta1 = (ser\_func\_vid[0] \* (-1) + ser\_func\_vid[1] \* (-1) + ser\_func\_vid[2] \* 1 + ser\_func\_vid[3] \* 1) / 4

beta2 = (ser\_func\_vid[0] \* (-1) + ser\_func\_vid[1] \* 1 + ser\_func\_vid[2] \* (-1) + ser\_func\_vid[3] \* 1) / 4

beta3 = (ser\_func\_vid[0] \* (-1) + ser\_func\_vid[1] \* 1 + ser\_func\_vid[2] \* 1 + ser\_func\_vid[3] \* (-1)) / 4

print("beta0: {}".format(beta0))

print("beta1: {}".format(beta1))

print("beta2: {}".format(beta2))

print("beta3: {}".format(beta3) + "\n")

t0 = abs(beta0) / s\_beta

t1 = abs(beta1) / s\_beta

t2 = abs(beta2) / s\_beta

t3 = abs(beta3) / s\_beta

print("t0: {}".format(t0))

print("t1: {}".format(t1))

print("t2: {}".format(t2))

print("t3: {}".format(t3))

print("Коефіціенти b1, b2 та b3 є незначними, тобто вони виключаюься з рівняння.\n")

print("Перевірка критерія Фішера:")

y1k = b0

y2k = y1k

y3k = y1k

y4k = y1k

s\_ad = 1.5 \* ((y1k - ser\_func\_vid[0])\*\*2 + (y2k - ser\_func\_vid[1])\*\*2 + (y3k - ser\_func\_vid[2])\*\*2 + (y4k - ser\_func\_vid[3]))

f = s\_ad / s\_b

ft = 4.1

print("Fp: {}".format(f))

if f < ft:

print("Рівняння регресії адекватно оригіналу при рівні значимості 0.05.")

else:

print("Рівняння регресії неадекватно оригіналу при рівні значимості 0.05")