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 № 5

з дисципліни «Методи оптимізації та планування експерименту»

Виконав:

студент ІІ курсу ФІОТ

групи ІО-83

Малашкін В’ячеслав

Залікова книжка № 8318

Варіант: 316

Перевірив:

ст. вик.

Регіда П. Г.

Київ – 2020

Код програми:

from random import randint

import numpy

from scipy.stats import t,f

"""Функція дл виводу матриці та функції відгуку"""

def matrix():

columns = ["№", "x1", "x2", "x3", "x1\*x2", "x1\*x3", "x2\*x3", "x1\*x2\*x3", "x1^2", "x2^2", "x3^2"]

for i in range(11):

s = "{:<7}| "

if i >= 5:

s = "{:<9}| "

if i >= 6:

s = "{:<8}| "

if i >= 9:

s = "{:<7}| "

if i >= 10:

s = "{:<5}| "

print(s.format(columns[i]), end="")

for i in range(m):

print("Yi{:d} |".format(i+1), end="")

print(" Ys | Ye |", end="")

print()

for i in range(n):

print(" {:2d} ".format(i), end="")

for j in range(1, 11):

s = ""

if j >= 1 and j < 4:

s = " {:6.2f} "

if j >= 4 and j < 7:

s = " {:7.2f} "

if j == 7:

s = " {:8.2f} "

if j > 7 and j < 11:

s = " {:6.2f} "

print(s.format(cmb(x\_nat[i])[j]), end="")

for j in y[i][:-1]:

print(" {:3d} ".format(j), end="")

print(" {:6.2f} {:6.2f} "

.format(y[i][-1],

sum([cmb(x\_nat[i])[j] \* b[j] \* stud[j] for j in range(11)])), end="")

print()

print("\n\t\t\t\t\tФункція відгуку:\nY = ", end="")

if stud[0] != 0:

print("{:.5f}".format(b[0]), end="")

for i in range(1, 11):

if stud[i] != 0:

print(" + {:.5f}\*{}".format(b[i], columns[i]), end="")

print()

"""Функція для генерації у"""

def geny(n, m, y\_max, y\_min):

mat\_y = [[randint(y\_min, y\_max) for j in range(m)]for i in range(n)]

for elem in mat\_y:

elem.append(sum(elem) / len(elem))

return mat\_y

def kohren(mat\_y, m, n):

s = []

for i in range(n):

ks = 0

for j in range(m):

ks += (mat\_y[i][-1] - mat\_y[i][j]) \*\* 2

s.append(ks / m)

gp = max(s) / sum(s)

fisher = table\_fisher(0.95, n, m, 1)

gt = fisher/(fisher+(m-1)-2)

return gp < gt

def cmb(arr):

return [1, \*arr,

arr[0]\*arr[1],

arr[0]\*arr[2],

arr[1]\*arr[2],

arr[0]\*arr[1]\*arr[2],

arr[0]\*arr[0],

arr[1]\*arr[1],

arr[2]\*arr[2]]

"""Функція для підрахунку коефіціентів b"""

def get\_b(lmaty):

a00 = [[],

[x\_nat\_mod[0]], [x\_nat\_mod[1]], [x\_nat\_mod[2]],

[x\_nat\_mod[0], x\_nat\_mod[1]],

[x\_nat\_mod[0], x\_nat\_mod[2]],

[x\_nat\_mod[1], x\_nat\_mod[2]],

[x\_nat\_mod[0], x\_nat\_mod[1], x\_nat\_mod[2]],

[x\_nat\_mod[0], x\_nat\_mod[0]],

[x\_nat\_mod[1], x\_nat\_mod[1]],

[x\_nat\_mod[2], x\_nat\_mod[2]]]

def calcxi(n, listx):

sumxi = 0

for i in range(n):

lsumxi = 1

for j in range(len(listx)):

lsumxi \*= listx[j][i]

sumxi += lsumxi

return sumxi

a0 = [15]

for i in range(10):

a0.append(calcxi(n, a00[i + 1]))

a1 = [calcxi(n, a00[i] + a00[1]) for i in range(len(a00))]

a2 = [calcxi(n, a00[i] + a00[2]) for i in range(len(a00))]

a3 = [calcxi(n, a00[i] + a00[3]) for i in range(len(a00))]

a4 = [calcxi(n, a00[i] + a00[4]) for i in range(len(a00))]

a5 = [calcxi(n, a00[i] + a00[5]) for i in range(len(a00))]

a6 = [calcxi(n, a00[i] + a00[6]) for i in range(len(a00))]

a7 = [calcxi(n, a00[i] + a00[7]) for i in range(len(a00))]

a8 = [calcxi(n, a00[i] + a00[8]) for i in range(len(a00))]

a9 = [calcxi(n, a00[i] + a00[9]) for i in range(len(a00))]

a10 = [calcxi(n, a00[i] + a00[10]) for i in range(len(a00))]

a = numpy.array([[a0[0], a0[1], a0[2], a0[3], a0[4], a0[5],

a0[6], a0[7], a0[8], a0[9], a0[10]],

[a1[0], a1[1], a1[2], a1[3], a1[4], a1[5],

a1[6], a1[7], a1[8], a1[9], a1[10]],

[a2[0], a2[1], a2[2], a2[3], a2[4], a2[5],

a2[6], a2[7], a2[8], a2[9], a2[10]],

[a3[0], a3[1], a3[2], a3[3], a3[4], a3[5],

a3[6], a3[7], a3[8], a3[9], a3[10]],

[a4[0], a4[1], a4[2], a4[3], a4[4], a4[5],

a4[6], a4[7], a4[8], a4[9], a4[10]],

[a5[0], a5[1], a5[2], a5[3], a5[4], a5[5],

a5[6], a5[7], a5[8], a5[9], a5[10]],

[a6[0], a6[1], a6[2], a6[3], a6[4], a6[5],

a6[6], a6[7], a6[8], a6[9], a6[10]],

[a7[0], a7[1], a7[2], a7[3], a7[4], a7[5],

a7[6], a7[7], a7[8], a7[9], a7[10]],

[a8[0], a8[1], a8[2], a8[3], a8[4], a8[5],

a8[6], a8[7], a8[8], a8[9], a8[10]],

[a9[0], a9[1], a9[2], a9[3], a9[4], a9[5],

a9[6], a9[7], a9[8], a9[9], a9[10]],

[a10[0], a10[1], a10[2], a10[3], a10[4], a10[5],

a10[6], a10[7], a10[8], a10[9], a10[10]]])

c0 = [calcxi(n, [lmaty])]

for i in range(len(a00) - 1):

c0.append(calcxi(n, a00[i + 1] + [lmaty]))

c = numpy.array([c0[0], c0[1], c0[2], c0[3], c0[4], c0[5],

c0[6], c0[7], c0[8], c0[9], c0[10]])

b = numpy.linalg.solve(a, c)

return b

""""""

def table\_student(prob, n, m):

x\_vec = [i\*0.0001 for i in range(int(5/0.0001))]

par = 0.5 + prob/0.1\*0.05

f3 = (m - 1) \* n

for i in x\_vec:

if abs(t.cdf(i, f3) - par) < 0.000005:

return i

def table\_fisher(prob, n, m, d):

x\_vec = [i\*0.001 for i in range(int(10/0.001))]

f3 = (m - 1) \* n

for i in x\_vec:

if abs(f.cdf(i, n-d, f3)-prob) < 0.0001:

return i

"""Критерій Стьюдента"""

def student(n, m, mat\_y):

disp = []

for i in mat\_y:

s = 0

for k in range(m):

s += (i[-1] - i[k]) \*\* 2

disp.append(s / m)

sbt = (sum(disp) / n / n / m) \*\* (0.5)

bs = []

for i in range(11):

ar = []

for j in range(len(mat\_y)):

ar.append(mat\_y[j][-1] \* cmb(xnorm[j])[i] / n)

bs.append(sum(ar))

t = [(bs[i] / sbt) for i in range(11)]

tt = table\_student(0.95, n, m)

st = [i > tt for i in t]

return st

"""Критерій Кохрена"""

def kohren(mat\_y, m, n):

s = []

for i in range(n):

ks = 0

for j in range(m):

ks += (mat\_y[i][-1] - mat\_y[i][j]) \*\* 2

s.append(ks / m)

gp = max(s) / sum(s)

fisher = table\_fisher(0.95, n, m, 1)

gt = fisher/(fisher+(m-1)-2)

return gp < gt

"""Критерій Фішера"""

def fisher(b\_0, x\_mod, n, m, d, mat\_y):

if d == n:

return True

disp = []

for i in mat\_y:

s = 0

for k in range(m):

s += (i[-1] - i[k]) \*\* 2

disp.append(s / m)

sad = sum([(sum([cmb(x\_nat[i])[j] \* b\_0[j] for j in range(11)]) - mat\_y[i][-1]) \*\* 2 for i in range(n)])

sad = sad \* m / (n - d)

fp = sad / sum(disp) / n

ft = table\_fisher(0.95, n, m, d)

return fp < ft

l = 1.215

x\_min = [-8, -6, -10]

x\_max = [8, 3, 7]

x\_0 = [(x\_min[0]+x\_max[0])/2,

(x\_min[1]+x\_max[1])/2,

(x\_min[2]+x\_max[2])/2]

x\_l = [l\*(x\_max[0] - x\_0[0]) + x\_0[0],

l\*(x\_max[1] - x\_0[1]) + x\_0[1],

l\*(x\_max[2] - x\_0[2]) + x\_0[2]]

x\_cp\_min = sum(x\_min) / 3

x\_cp\_max = sum(x\_max) / 3

ymin = round(200 + x\_cp\_min)

ymax = round(200 + x\_cp\_max)

xnorm = [[-1, -1, -1],

[-1, 1, 1],

[1, -1, 1],

[1, 1, -1],

[-1, -1, 1],

[-1, 1, -1],

[1, -1, -1],

[1, 1, 1],

[-l, 0, 0],

[l, 0, 0],

[0, -l, 0],

[0, l, 0],

[0, 0, -l],

[0, 0, l],

[0, 0, 0]]

x\_nat = [[x\_min[0], x\_min[1], x\_min[2]],

[x\_min[0], x\_min[1], x\_max[2]],

[x\_min[0], x\_max[1], x\_min[2]],

[x\_min[0], x\_max[1], x\_max[2]],

[x\_max[0], x\_min[1], x\_min[2]],

[x\_max[0], x\_min[1], x\_max[2]],

[x\_max[0], x\_max[1], x\_min[2]],

[x\_max[0], x\_max[1], x\_max[2]],

[-x\_l[0], x\_0[1], x\_0[2]],

[x\_l[0], x\_0[1], x\_0[2]],

[x\_0[0], -x\_l[1], x\_0[2]],

[x\_0[0], x\_l[1], x\_0[2]],

[x\_0[0], x\_0[1], -x\_l[2]],

[x\_0[0], x\_0[1], x\_l[2]],

[x\_0[0], x\_0[1], x\_0[2]]]

n = 15

m = 3

"""Якщо критерій Фішера поверне False, то цикл почнеться спочатку"""

while True:

"""Якщо критерій Фішера поверне Кохрена, то цикл почнеться спочатку"""

while True:

print("Варіант: 316\nm = {0}\nn = {1}\n".format(m, n))

x\_nat\_mod = [[x\_nat[i][j] for i in range(15)] for j in range(3)]

y = geny(n, m, ymax, ymin)

matymod = [y[i][-1] for i in range(len(y))]

kohren\_flag = kohren(y, 3, 15)

print("Дисперсія {}однорідна, з ймовірністю = {:.2}"

.format("" if kohren\_flag else "не ", 0.95))

if kohren\_flag:

break

else:

m += 1

b = get\_b(matymod)

stud = student(n, m, y)

d = sum(stud)

fisher\_ = fisher(b, x\_nat\_mod, n, m, d, y)

print("Рівняння {}адекватне, з ймовірністю = {:.2f}\n"

.format("" if fisher\_ else "не ", 0.95))

matrix()

if fisher\_:

break

Результати програми:

