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 №4

З дисципліни «Методи оптимізації та планування»

ВИКОНАB:

Студент ІO-93

Губенко Владислав

ПЕРЕВІРИВ:

Регіда П.Г.

Київ 2021 р.

Вивчити основні поняття, визначення, принципи теорії планування

експерименту, на основі яких вивчити побудову формалізованих алгоритмів проведення експерименту і отримання формалізованої моделі об’єкта.

Закріпити отримані знання практичним їх використанням при написанні програми, що реалізує завдання на лабораторну роботу.



**Лістинг програми:**

import random import numpy as np

import sklearn.linear\_model as lm from scipy.stats import f, t

from numpy.linalg import solve

def regression(x, b):

y = sum([x[i] \* b[i] for i in range(len(x))]) return y

def dispersion(y, y\_aver, n, m): res = []

for i in range(n):

s = sum([(y\_aver[i] - y[i][j]) \*\* 2 for j in range(m)]) / m res.append(round(s, 3))

return res

def planing\_matrix\_interaction\_effect(n, m): x\_normalized = [[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]]

y = np.zeros(shape=(n, m), dtype=np.int64) for i in range(n):

for j in range(m):

y[i][j] = random.randint(y\_min, y\_max)

for x in x\_normalized: x.append(x[1] \* x[2])

x.append(x[1] \* x[3])

x.append(x[2] \* x[3])

x.append(x[1] \* x[2] \* x[3])

x\_normalized = np.array(x\_normalized[:len(y)])

x = np.ones(shape=(len(x\_normalized), len(x\_normalized[0])), dtype=np.int64)

for i in range(len(x\_normalized)): for j in range(1, 4):

if x\_normalized[i][j] == -1: x[i][j] = x\_range[j - 1][0]

else:

x[i][j] = x\_range[j - 1][1]

for i in range(len(x)):

x[i][4] = x[i][1] \* x[i][2]

x[i][5] = x[i][1] \* x[i][3]

x[i][6] = x[i][2] \* x[i][3]

x[i][7] = x[i][1] \* x[i][3] \* x[i][2]

Y3')

print(f'\nМатриця планування для n = {n}, m = {m}:') print('\nЗ кодованими значеннями факторів:')

print('\n X0 X1 X2 X3 X1X2 X1X3 X2X3 X1X2X3 Y1 Y2

print(np.concatenate((x, y), axis=1)) print('\nНормовані значення факторів:\n') print(x\_normalized)

return x, y, x\_normalized

def find\_coef(X, Y, norm=False):

skm = lm.LinearRegression(fit\_intercept=False) skm.fit(X, Y)

B = skm.coef\_

if norm == 1:

print('\nКоефіцієнти рівняння регресії з нормованими X:') else:

print('\nКоефіцієнти рівняння регресії:') B = [round(i, 3) for i in B]

print(B) return B

def bs(x, y, y\_aver, n):

res = [sum(1 \* y for y in y\_aver) / n] for i in range(7):

b = sum(j[0] \* j[1] for j in zip(x[:, i], y\_aver)) / n res.append(b)

return res

def kriteriy\_studenta2(x, y, y\_aver, n, m): S\_kv = dispersion(y, y\_aver, n, m) s\_kv\_aver = sum(S\_kv) / n

s\_Bs = (s\_kv\_aver / n / m) \*\* 0.5 Bs = bs(x, y, y\_aver, n)

ts = [round(abs(B) / s\_Bs, 3) for B in Bs]

return ts

def kriteriy\_studenta(x, y\_average, n, m, dispersion): dispersion\_average = sum(dispersion) / n

s\_beta\_s = (dispersion\_average / n / m) \*\* 0.5

beta = [sum(1 \* y for y in y\_average) / n] for i in range(3):

b = sum(j[0] \* j[1] for j in zip(x[:,i], y\_average)) / n beta.append(b)

t = [round(abs(b) / s\_beta\_s, 3) for b in beta] return t

def kriteriy\_fishera(y, y\_average, y\_new, n, m, d, dispersion):

S\_ad = m / (n - d) \* sum([(y\_new[i] - y\_average[i])\*\*2 for i in range(len(y))])

dispersion\_average = sum(dispersion) / n

return S\_ad / dispersion\_average

def check(X, Y, B, n, m, norm=False):

f1 = m - 1 f2 = n

f3 = f1 \* f2 q = 0.05

y\_aver = [round(sum(i) / len(i), 3) for i in Y] print('\nСереднє значення y:', y\_aver)

dispersion\_arr = dispersion(Y, y\_aver, n, m)

qq = (1 + 0.95) / 2

student\_cr\_table = t.ppf(df=f3, q=qq)

ts = kriteriy\_studenta2(X[:, 1:], Y, y\_aver, n, m)

temp\_cohren = f.ppf(q=(1 - q / f1), dfn=f2, dfd=(f1 - 1) \* f2) cohren\_cr\_table = temp\_cohren / (temp\_cohren + f1 - 1)

Gp = max(dispersion\_arr) / sum(dispersion\_arr)

print('Дисперсія y:', dispersion\_arr)

print(f'Gp = {Gp}')

if Gp < cohren\_cr\_table:

print(f'З ймовірністю {1-q} дисперсії однорідні.') else:

print("Необхідно збільшити кількість дослідів") m += 1

with\_interaction\_effect(n, m)

print('\nКритерій Стьюдента:\n', ts)

res = [t for t in ts if t > student\_cr\_table]

final\_k = [B[i] for i in range(len(ts)) if ts[i] in res] print('\nКоефіцієнти {} статистично незначущі, тому ми виключаємо їх з

рівняння.'.format(

[round(i, 3) for i in B if i not in final\_k]))

y\_new = []

for j in range(n):

y\_new.append(regression([X[j][i] for i in range(len(ts)) if ts[i] in res], final\_k))

print(f'\nЗначення "y" з коефіцієнтами {final\_k}') print(y\_new)

d = len(res) if d >= n:

print('\nF4 <= 0') print('')

return f4 = n - d

Fp = kriteriy\_fishera(Y, y\_aver, y\_new, n, m, d, dispersion\_arr)

Ft = f.ppf(dfn=f4, dfd=f3, q=1 - 0.05)

print('\nПеревірка адекватності за критерієм Фішера') print('Fp =', Fp)

print('Ft =', Ft)

if Fp < Ft:

print('Математична модель адекватна експериментальним даним') return True

else:

print('Математична модель не адекватна експериментальним даним') return False

def with\_interaction\_effect(n, m):

X, Y, X\_norm = planing\_matrix\_interaction\_effect(n, m)

y\_aver = [round(sum(i) / len(i), 3) for i in Y]

B\_norm = find\_coef(X\_norm, y\_aver, norm=True)

return check(X\_norm, Y, B\_norm, n, m, norm=True)

def planning\_matrix\_linear(n, m, x\_range): x\_normalized = np.array([[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]])

y = np.zeros(shape=(n,m)) for i in range(n):

for j in range(m):

y[i][j] = random.randint(y\_min,y\_max) x\_normalized = x\_normalized[:len(y)]

x = np.ones(shape=(len(x\_normalized), len(x\_normalized[0]))) for i in range(len(x\_normalized)):

for j in range(1, len(x\_normalized[i])): if x\_normalized[i][j] == -1:

x[i][j] = x\_range[j-1][0] else:

x[i][j] = x\_range[j-1][1]

print('\nМатриця планування:' )

print('\n X0 X1 X2 X3 Y1 Y2 Y3 ')

print(np.concatenate((x, y), axis=1))

return x, y, x\_normalized

def regression\_equation(x, y, n):

y\_average = [round(sum(i) / len(i), 2) for i in y]

mx1 = sum(x[:, 1]) / n

mx2 = sum(x[:, 2]) / n

mx3 = sum(x[:, 3]) / n

my = sum(y\_average) / n

a1 = sum([y\_average[i] \* x[i][1] for i in range(len(x))]) / n a2 = sum([y\_average[i] \* x[i][2] for i in range(len(x))]) / n a3 = sum([y\_average[i] \* x[i][3] for i in range(len(x))]) / n

a12 = sum([x[i][1] \* x[i][2] for i in range(len(x))]) / n

a13 = sum([x[i][1] \* x[i][3] for i in range(len(x))]) / n

a23 = sum([x[i][2] \* x[i][3] for i in range(len(x))]) / n

a11 = sum([i \*\* 2 for i in x[:, 1]]) / n a22 = sum([i \*\* 2 for i in x[:, 2]]) / n a33 = sum([i \*\* 2 for i in x[:, 3]]) / n

X = [[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a12, a22, a23], [mx3, a13, a23, a33]]

Y = [my, a1, a2, a3]

B = [round(i, 2) for i in solve(X, Y)]

print('\nРівняння регресії:')

print(f'y = {B[0]} + {B[1]}\*x1 + {B[2]}\*x2 + {B[3]}\*x3')

return y\_average, B

def linear(n, m): f1 = m - 1

f2 = n

f3 = f1 \* f2 q = 0.05

x, y, x\_norm = planning\_matrix\_linear(n, m, x\_range)

y\_average, B = regression\_equation(x, y, n) dispersion\_arr = dispersion(y, y\_average, n, m)

temp\_cohren = f.ppf(q=(1 - q / f1), dfn=f2, dfd=(f1 - 1) \* f2) cohren\_cr\_table = temp\_cohren / (temp\_cohren + f1 - 1)

Gp = max(dispersion\_arr) / sum(dispersion\_arr)

print('\nПеревірка за критерієм Кохрена:\n') print(f'Розрахункове значення: Gp = {Gp}'

f'\nТабличне значення: Gt = {cohren\_cr\_table}') if Gp < cohren\_cr\_table:

print(f'З ймовірністю {1-q} дисперсії однорідні.') else:

print("Необхідно збільшити ксть дослідів") m += 1

linear(n, m)

qq = (1 + 0.95) / 2

student\_cr\_table = t.ppf(df=f3, q=qq)

student\_t = kriteriy\_studenta(x\_norm[:,1:], y\_average, n, m, dispersion\_arr)

print('\nТабличне значення критерій Стьюдента:\n', student\_cr\_table) print('Розрахункове значення критерій Стьюдента:\n', student\_t) res\_student\_t = [temp for temp in student\_t if temp > student\_cr\_table] final\_coefficients = [B[student\_t.index(i)] for i in student\_t if i in

res\_student\_t]

print('Коефіцієнти {} статистично незначущі.'.

format([i for i in B if i not in final\_coefficients]))

y\_new = []

for j in range(n):

y\_new.append(regression([x[j][student\_t.index(i)] for i in student\_t if i in res\_student\_t], final\_coefficients))

print(f'\nОтримаємо значення рівння регресії для {m} дослідів: ') print(y\_new)

d = len(res\_student\_t) f4 = n - d

Fp = kriteriy\_fishera(y, y\_average, y\_new, n, m, d, dispersion\_arr) Ft = f.ppf(dfn=f4, dfd=f3, q=1 - 0.05)

print('\nПеревірка адекватності за критерієм Фішера:\n') print('Розрахункове значення критерія Фішера: Fp =', Fp) print('Табличне значення критерія Фішера: Ft =', Ft)

if Fp < Ft:

print('Математична модель адекватна експериментальним даним') return True

else:

print('Математична модель не адекватна експериментальним даним') return False

def main(n, m):

main\_1 = linear(n, m) if not main\_1:

interaction\_effect = with\_interaction\_effect(n, m) if not interaction\_effect:

main(n, m)

if name == ' main ':

x\_range = ((10, 60), (-35, 10), (-30, 45))

y\_max = 200 + int(sum([x[1] for x in x\_range]) / 3) y\_min = 200 + int(sum([x[0] for x in x\_range]) / 3)

main(8, 3)

# Результат виконання роботи:

Матриця планування:

X0 X1 X2 X3 Y1 Y2 Y3

[[ 1. 10. -35. -30. 223. 237. 221.]

[ 1. 10. 10. 45. 184. 214. 207.]

[ 1. 60. -35. 45. 223. 235. 202.]

[ 1. 60. 10. -30. 215. 186. 223.]

[ 1. 10. -35. 45. 215. 198. 219.]

[ 1. 10. 10. -30. 227. 216. 220.]

[ 1. 60. -35. -30. 200. 195. 203.]

[ 1. 60. 10. 45. 227. 188. 195.]]

Рівняння регресії:

y = 215.46 + -0.15\*x1 + -0.13\*x2 + -0.07\*x3

Перевірка за критерієм Кохрена:

Розрахункове значення: Gp = 0.2728798747991665

Табличне значення: Gt = 0.815948432359917

З ймовірністю 0.95 дисперсії однорідні.

Табличне значення критерій Стьюдента:

2.1199052992210112

Розрахункове значення критерій Стьюдента:

[90.121, 1.582, 1.226, 1.048]

Коефіцієнти [-0.15, -0.13, -0.07] статистично незначущі.

Отримаємо значення рівння регресії для 3 дослідів:

[215.46, 215.46, 215.46, 215.46, 215.46, 215.46, 215.46, 215.46]

Перевірка адекватності за критерієм Фішера:

Розрахункове значення критерія Фішера: Fp = 2.7933984178936244 Табличне значення критерія Фішера: Ft = 2.6571966002210865 Математична модель не адекватна експериментальним даним

Матриця планування для n = 8, m = 3:

З кодованими значеннями факторів:

X0 X1 X2 X3 X1X2 X1X3 X2X3 X1X2X3 Y1 Y2 Y3 [[ 1 10 -35 -30 -350 -300 1050 10500 193 222

211]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| [ | 1 | 10 | 10 | 45 | 100 450 450 4500 224 206 |
|  | 222] |  |  |  |  |
| [ | 1 | 60 | -35 | 45 | -2100 2700 -1575 -94500 213 215 |
|  | 233] |  |  |  |  |
| [ | 1 | 60 | 10 | -30 | 600 -1800 -300 -18000 207 229 |
|  | 231] |  |  |  |  |
| [ | 1 | 10 | -35 | 45 | -350 450 -1575 -15750 230 183 |
|  | 184] |  |  |  |  |
| [ | 1 | 10 | 10 | -30 | 100 -300 -300 -3000 200 189 |
|  | 207] |  |  |  |  |
| [ | 1 | 60 | -35 | -30 | -2100 -1800 1050 63000 195 214 |

202]

[ 1 60 10 45 600 2700 450 27000 211 204

201]]

Нормовані значення факторів:

[[ 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]]

Коефіцієнти рівняння регресії з нормованими X: [209.417, 3.5, 1.5, 1.083, -0.583, -1.167, -0.667, -7.75]

Середнє значення y: [208.667, 217.333, 220.333, 222.333, 199.0, 198.667, 203.667, 205.333]

Дисперсія y: [142.889, 64.889, 80.889, 118.222, 480.667, 54.889, 61.556, 17.556]

Gp = 0.47052391594399523

З ймовірністю 0.95 дисперсії однорідні.

Критерій Стьюдента:

[90.788, 1.517, 0.65, 0.47, 0.253, 0.506, 0.289, 3.36]

Коефіцієнти [3.5, 1.5, 1.083, -0.583, -1.167, -0.667] статистично незначущі, тому ми виключаємо їх з рівняння.

Значення "y" з коефіцієнтами [209.417, -7.75]

[217.167, 217.167, 217.167, 217.167, 201.667, 201.667, 201.667, 201.667]

Перевірка адекватності за критерієм Фішера

Fp = 0.5581327835842735

Ft = 2.741310828338778

Математична модель адекватна експериментальним даним

Process finished with exit code 0