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»

Загальні принципи організації експериментів з довільними значеннями факторів

ВИКОНАВ: Ралло Ігор
Студент 2 курсу
ФІОТ гр. ІО-92

ПЕРЕВІРИВ:
асистент
Регіда П.Г.

Київ 2021 р.

Мета: 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.

Варіант завдання:

216	-5	5	-2	5	-1	4

Лістинг програми:

```
from math import sqrt
from scipy.stats import f, t
from functools import partial
from random import randint
from numpy.linalg import solve

x1, x2, x3 = [-5, 5], [-2, 5], [-1, 4]
m, N, l = 3, 15, 1.215  # кількість повторень кожної
комбінації &  кількість повторення дослідів

x_avg = [(max(x1) + max(x2) + max(x3)) / 3, (min(x1) +
min(x2) + min(x3)) / 3]  # Xcp(max) & Xcp(min)
xo = [(min(x1) + max(x1)) / 2, (min(x2) + max(x2)) / 2,
(min(x3) + max(x3)) / 2]  # Xoi
delta_x = [max(x1) - xo[0], max(x1) - xo[1], max(x1) -
xo[2]]  # delta Xi

y_range = [200 + int(max(x_avg)), 200 + int(min(x_avg))]
# Yi(max) & Yi(min)

xn = [[+1, +1, +1, +1, +1, +1, +1, +1, +1, +1, +1, +1,
+1, +1, +1],  # нормовані значення факторів
      [-1, -1, -1, -1, +1, +1, +1, +1, -1.215, 1.215, 0,
0, 0, 0, 0],
      [-1, -1, +1, +1, -1, -1, +1, +1, 0, 0, -1.215,
```

```
1.215, 0, 0, 0],
      [-1, +1, -1, +1, -1, +1, -1, +1, 0, 0, 0, 0, -1.215,
1.215, 0]]

xx = [[int(x * y) for x, y in zip(xn[1], xn[2])],  #
нормовані значення факторів для ефекту взаємодії
      [int(x * y) for x, y in zip(xn[1], xn[3])],
      [int(x * y) for x, y in zip(xn[2], xn[3])]]

xxx = [int(x * y * z) for x, y, z in zip(xn[1], xn[2],
xn[3])]

x_xn = [[round(xn[j][i] ** 2, 3) for i in range(N)] for j
in range(1, m+1)]  # нормовані знач. факторів для квад.
членів

x = [[min(x1), min(x1), min(x1), min(x1), max(x1),
max(x1), max(x1), max(x1), round(-l * delta_x[0] + xo[0],
3),
      round(l * delta_x[0] + xo[0], 3), xo[0], xo[0],
xo[0], xo[0], xo[0]],  # натуральні значення факторів
     [min(x2), min(x2), max(x2), max(x2), min(x2),
min(x2), max(x2), max(x2), xo[1], xo[1],
      round(-l * delta_x[1] + xo[1], 3), round(l *
delta_x[1] + xo[1], 3), xo[1], xo[1], xo[1]],
     [min(x3), max(x3), min(x3), max(x3), max(x3),
min(x3), max(x3), min(x3), xo[2], xo[2], xo[2], xo[2],
      round(-l * delta_x[2] + xo[2], 3), round(l *
delta_x[2] + xo[2], 3), xo[2]]]

xx2 = [[round(x * y, 3) for x, y in zip(x[0], x[1])],  #
натуральні значення факторів для ефекту взаємодії
       [round(x * y, 3) for x, y in zip(x[0], x[2])],
       [round(x * y, 3) for x, y in zip(x[1], x[2])]]

xxx2 = [round(x * y * z, 3) for x, y, z in zip(x[0], x[1],
x[2])]

x_x = [[round(x[j][i] ** 2, 3) for i in range(N)] for j in
range(m)]  # натуральні значення факторів для квадрат.
```

```
членів

while True:
    y = [[round(randint(min(y_range), max(y_range)), 4)
for i in range(m)] for j in range(N)]  # формування Y
    arr_avg = lambda arr: round(sum(arr) / len(arr), 4)
    y_avg = list(map(arr_avg, y))  # середнє значення Y

    dispersions = [sum([((y[i][j] - y_avg[i]) ** 2) / m
for j in range(m)]) for i in range(N)]  # дисперсії по
рядках
    x_matrix = x + xx2 + [xxx2] + x_x  # повна матриця з
натуральними значеннями факторів
    norm_matrix = xn + xx + [xxx] + x_xn  # повна матриця
з нормованими значеннями факторів

    mx = list(map(arr_avg, x_matrix))  # середні значення
x по колонкам
    my = sum(y_avg) / N  # середнє значення Y_avg

    # ========================================
Форматування таблиці
==================================================

    table_factors_1 = ["X0", "X1", "X2", "X3"]
    table_factors_2 = ["X1X2", "X1X3", "X2X3", "X1X2X3",
"X1^2", "X2^2", "X3^2"]
    table_y = ["Y1", "Y2", "Y3"]
    other = ["#", "Y"]

    header_format = "+{0:=^3}" + "+{0:=^8}" *
(len(table_factors_1)) + "+{0:=^8s}" * (
        len(table_factors_2)) + "+{0:=^6s}" *
(len(table_y)) + "+{0:=^8s}"
    row_format = "|{:^3}" + "|{:^8}" *
(len(table_factors_1)) + "|{:^8}" * (len(table_factors_2))
+ "|{:^6}" * (
        len(table_y)) + "|{:^8}"
    separator_format = "+{0:-^3s}" + "+{0:-^8s}" *
(len(table_factors_1)) + "+{0:-^8s}" * (
```

```
        len(table_factors_2)) + "+{0:-^6s}" *
(len(table_y)) + "+{0:-^8s}"

    # ========================================= Нормальні
значення
=====================================================
    print(header_format.format("=") + "+\n" +
"|{:^132s}|\n".format("Матриця ПФЕ (нормальні значення
факторів)") +
          header_format.format("=") + "+\n" +
row_format.format(other[0], *table_factors_1,
*table_factors_2, *table_y,

other[1]) + "|\n" + header_format.format("=") + "+")

    for i in range(N):
        print("|{:^3}|".format(i + 1), end="")
        for j in range(4):
print("{:^+8}|".format(xn[j][i]), end="")
        for j in range(3):
print("{:^+8}|".format(xx[j][i]), end="")
        print("{:^+8}|".format(xxx[i]), end="")
        for j in range(m):
print("{:^+8}|".format(x_xn[j][i]), end="")
        for j in range(m): print("{:^6}|".format(y[i][j]),
end="")
        print("{:^8.2f}|".format(y_avg[i]))

    print(separator_format.format("-") + "+\n\n")

    # ========================================= Натуральні
значення
=====================================================
    print(header_format.format("=") + "+\n" +
"|{:^132s}|\n".format("Матриця ПФЕ (натуральні значення
факторів)") +
          header_format.format("=") + "+\n" +
row_format.format(other[0], *table_factors_1,
*table_factors_2, *table_y,
```

```
other[1]) + "|\n" + header_format.format("=") + "+")

    for i in range(N):
        print("|{0:^3}|{1:^+8}|".format(i + 1, xn[0][i]),
end="")
        for j in range(3): print("{:^
8}|".format(x[j][i]), end="")
        for j in range(3): print("{:^
8}|".format(xx2[j][i]), end="")
        print("{:^+8}|".format(xxx2[i]), end="")
        for j in range(m): print("{:^
8}|".format(x_x[j][i]), end="")
        for j in range(m): print("{:^
6}|".format(y[i][j]), end="")
        print("{:^8.2f}|".format(y_avg[i]))


    def a(first, second): return sum([x_matrix[first -
1][j] * x_matrix[second - 1][j] / N for j in range(N)])
    def find_a(num): return sum([y_avg[j] * x_matrix[num -
1][j] / N for j in range(N)])
    def check(b_lst, k):
        return b_lst[0] + b_lst[1] * x_matrix[0][k] +
b_lst[2] * x_matrix[1][k] + b_lst[3] * x_matrix[2][k] + \
               b_lst[4] * x_matrix[3][k] + b_lst[5] *
x_matrix[4][k] + b_lst[6] * x_matrix[5][k] + \
               b_lst[7] * x_matrix[6][k] + b_lst[8] *
x_matrix[7][k] + b_lst[9] * x_matrix[8][k] + \
               b_lst[10] * x_matrix[9][k]

    unknown = [[1, mx[0], mx[1], mx[2], mx[3], mx[4],
mx[5], mx[6], mx[7], mx[8], mx[9]],  # ліва частина
рівнянь з невідомими для пошуку коефіцієнтів b (приклад в
методі)
               [mx[0], a(1, 1), a(1, 2), a(1, 3), a(1, 4),
a(1, 5), a(1, 6), a(1, 7), a(1, 8), a(1, 9), a(1, 10)],
               [mx[1], a(2, 1), a(2, 2), a(2, 3), a(2, 4),
a(2, 5), a(2, 6), a(2, 7), a(2, 8), a(2, 9), a(2, 10)],
               [mx[2], a(3, 1), a(3, 2), a(3, 3), a(3, 4),
a(3, 5), a(3, 6), a(3, 7), a(3, 8), a(3, 9), a(3, 10)],
```

```
                [mx[3], a(4, 1), a(4, 2), a(4, 3), a(4, 4),
a(4, 5), a(4, 6), a(4, 7), a(4, 8), a(4, 9), a(4, 10)],
                [mx[4], a(5, 1), a(5, 2), a(5, 3), a(5, 4),
a(5, 5), a(5, 6), a(5, 7), a(5, 8), a(5, 9), a(5, 10)],
                [mx[5], a(6, 1), a(6, 2), a(6, 3), a(6, 4),
a(6, 5), a(6, 6), a(6, 7), a(6, 8), a(6, 9), a(6, 10)],
                [mx[6], a(7, 1), a(7, 2), a(7, 3), a(7, 4),
a(7, 5), a(7, 6), a(7, 7), a(7, 8), a(7, 9), a(7, 10)],
                [mx[7], a(8, 1), a(8, 2), a(8, 3), a(8, 4),
a(8, 5), a(8, 6), a(8, 7), a(8, 8), a(8, 9), a(8, 10)],
                [mx[8], a(9, 1), a(9, 2), a(9, 3), a(9, 4),
a(9, 5), a(9, 6), a(9, 7), a(9, 8), a(9, 9), a(9, 10)],
                [mx[9], a(10, 1), a(10, 2), a(10, 3), a(10,
4), a(10, 5), a(10, 6), a(10, 7), a(10, 8), a(10, 9),
a(10, 10)]]
    known = [my, find_a(1), find_a(2), find_a(3),
find_a(4), find_a(5), find_a(6), find_a(7), find_a(8),
find_a(9), find_a(10)]   # знаходення відомих значень a1,
a2, ...

    b = solve(unknown, known)
    print(separator_format.format("-") + f"+\n\n\tОтримане
рівняння регресії при m={m}:\n"
                                        f"ŷ = {b[0]:.3f}
+ {b[1]:.3f}*X1 + {b[2]:.3f}*X2 + "
                                        f"{b[3]:.3f}*X3 +
{b[4]:.3f}*X1X2 + {b[5]:.3f}*X1X3 + "
                                        f"{b[6]:.3f}*X2X3
+ {b[7]:.3f}*X1X2X3 + {b[8]:.3f}*X11^2 + "

f"{b[9]:.3f}*X22^2 + {b[10]:.3f}*X33^2\n\n\tПеревірка:")
    for i in range(N): print("ŷ{} = {:.3f} ≈
{:.3f}".format((i + 1), check(b, i), y_avg[i]))

    # ========================================= Критерій
Кохрена
=========================================================
    def cochran(f1, f2, q=0.05):
        q1 = q / f1
```

```
        fisher = f.ppf(q=1 - q1, dfn=f2, dfd=(f1 - 1) *
f2)
        return fisher / (fisher + f1 - 1)


    f1, f2 = m - 1, N
    f3 = f1 * f2
    Gp = max(dispersions) / sum(dispersions)
    Gt = cochran(f1, f2)

    print("\nОднорідність дисперсії (критерій Кохрена): ")
    print(f"Gp = {Gp}\nGt = {Gt}")
    if Gp < Gt:
        print("\nДисперсія однорідна (Gp < Gt)")

        D_beta = sum(dispersions) / (N * N * m)
        Sb = sqrt(abs(D_beta))
        beta = [sum([(y_avg[j] * norm_matrix[i][j]) / N
for j in range(N)]) for i in range(len(norm_matrix))]

        t_list = [abs(i) / Sb for i in beta]
        student = partial(t.ppf, q=1-0.025)
        d, T = 0, student(df=f3)
        print("\nt табличне = ", T)

        for i in range(len(t_list)):
            if t_list[i] < T:
                b[i] = 0
                print("\tt{} = {} => коефіцієнт
незначимий, його слід виключити з рів-ня
регресії".format(i, t_list[i]))
            else:
                print("\tt{} = {} => коефіцієнт
значимий".format(i, t_list[i]))
                d += 1

        print("\nОтже, кіл-ть значимих коеф. d =", d,
"\n\n\tРів-ня регресії з урахуванням критерія
Стьюдента:\nŷ = ", end="")
        print("{:.3f}".format(b[0]), end="") if b[0] != 0
```

```
else None
        for i in range(1, 11):
            print(" + {:.3f}*{}".format(b[i],
(table_factors_1 + table_factors_2)[i]), end="") if b[i]
!= 0 else None
        print("\n\n\tПеревірка при підстановці в спрощене
рів-ня регресії:")
        for i in range(N): print("y`{} = {:.3f} ≈
{:.3f}".format((i + 1), check(b, i), y_avg[i]))

        f4 = N - d
        fisher_sum = sum([(check(b, i) - y_avg[i]) ** 2
for i in range(N)])
        D_ad = (m / f4) * fisher_sum

        fisher = partial(f.ppf, q=1-0.05)
        Fp = D_ad / sum(dispersions) / N
        Ft = fisher(dfn=f4, dfd=f3)
        print("\nКритерій Фішера:")
        if Fp > Ft:
            print("\tРівняння регресії неадекватне (Ft <
Fp).")
            break
        else:
            print("\tРівняння регресії адекватне (Ft >
Fp)!")
            break

    else:
        print("Дисперсія неоднорідна (Gp > Gt), збільшуємо
m, повторюємо операції")
        m += 1
```

Результат виконання роботи:

Матриця ПФЕ (нормальні значення факторів)

#	X0	X1	X2	X3	X1X2	X1X3	X2X3	X1X2X3	X1^2	X2^2	X3^2	Y1	Y2	Y3	Y
1	+1	-1	-1	-1	+1	+1	+1	-1	+1	+1	+1	202	199	199	200.00
2	+1	-1	-1	+1	+1	-1	-1	+1	+1	+1	+1	202	200	198	200.00
3	+1	-1	+1	-1	-1	+1	-1	+1	+1	+1	+1	203	204	201	202.67
4	+1	-1	+1	+1	-1	-1	+1	-1	+1	+1	+1	204	200	198	200.67
5	+1	+1	-1	-1	-1	-1	+1	+1	+1	+1	+1	203	198	201	200.67
6	+1	+1	-1	+1	-1	+1	-1	-1	+1	+1	+1	198	200	202	200.00
7	+1	+1	+1	-1	+1	-1	-1	-1	+1	+1	+1	204	204	204	204.00
8	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	201	203	201	201.67
9	+1	-1.215	+0	+0	+0	+0	+0	+0	+1.476	+0	+0	200	200	202	200.67
10	+1	+1.215	+0	+0	+0	+0	+0	+0	+1.476	+0	+0	199	201	201	200.33
11	+1	+0	-1.215	+0	+0	+0	+0	+0	+0	+1.476	+0	204	202	201	202.33
12	+1	+0	+1.215	+0	+0	+0	+0	+0	+0	+1.476	+0	204	202	201	202.33
13	+1	+0	+0	-1.215	+0	+0	+0	+0	+0	+0	+1.476	203	199	199	200.33
14	+1	+0	+0	+1.215	+0	+0	+0	+0	+0	+0	+1.476	198	201	201	200.00
15	+1	+0	+0	+0	+0	+0	+0	+0	+0	+0	+0	202	201	200	201.00

Матриця ПФЕ (натуральні значення факторів)

#	X0	X1	X2	X3	X1X2	X1X3	X2X3	X1X2X3	X1^2	X2^2	X3^2	Y1	Y2	Y3	Y
1	+1	-5	-2	-1	10	5	2	-10	25	4	1	202	199	199	200.00
2	+1	-5	-2	4	10	-20	-8	+40	25	4	16	202	200	198	200.00
3	+1	-5	5	-1	-25	5	-5	+25	25	25	1	203	204	201	202.67

Матриця ПФЕ (натуральні значення факторів)

#	X0	X1	X2	X3	X1X2	X1X3	X2X3	X1X2X3	X1^2	X2^2	X3^2	Y1	Y2	Y3	Y
1	+1	-5	-2	-1	10	5	2	-10	25	4	1	202	199	199	200.00
2	+1	-5	-2	4	10	-20	-8	+40	25	4	16	202	200	198	200.00
3	+1	-5	5	-1	-25	5	-5	+25	25	25	1	203	204	201	202.67
4	+1	-5	5	4	-25	-20	20	-100	25	25	16	204	200	198	200.67
5	+1	5	-2	4	-10	20	-8	-40	25	4	16	203	198	201	200.67
6	+1	5	-2	-1	-10	-5	2	+10	25	4	1	198	200	202	200.00
7	+1	5	5	4	25	20	20	+100	25	25	16	204	204	204	204.00
8	+1	5	5	-1	25	-5	-5	-25	25	25	1	201	203	201	201.67
9	+1	-6.075	1.5	1.5	-9.113	-9.113	2.25	-13.669	36.906	2.25	2.25	200	200	202	200.67
10	+1	6.075	1.5	1.5	9.113	9.113	2.25	+13.669	36.906	2.25	2.25	199	201	201	200.33
11	+1	0.0	-2.753	1.5	-0.0	0.0	-4.13	-0.0	0.0	7.579	2.25	204	202	201	202.33
12	+1	0.0	5.753	1.5	0.0	0.0	8.63	+0.0	0.0	33.097	2.25	204	202	201	202.33
13	+1	0.0	1.5	-2.753	0.0	-0.0	-4.13	-0.0	0.0	2.25	7.579	203	199	199	200.33
14	+1	0.0	1.5	5.753	0.0	0.0	8.63	+0.0	0.0	2.25	33.097	198	201	201	200.00
15	+1	0.0	1.5	1.5	0.0	0.0	2.25	+0.0	0.0	2.25	2.25	202	201	200	201.00

```
    Отримане рівняння регресії при m=3:
ŷ = 200.737 + -0.022*X1 + 0.014*X2 + 0.167*X3 + -0.004*X1X2 + 0.034*X1X3 + -0.005*X2X3 + 0.010*X1X2X3 + -0.015*X11^2 + 0.070*X22^2 + -0.049*X33^2

    Перевірка:
ŷ1 = 200.519 ≈ 200.000
ŷ2 = 200.332 ≈ 200.000
ŷ3 = 202.624 ≈ 202.667
ŷ4 = 200.437 ≈ 200.667
ŷ5 = 200.723 ≈ 200.667
ŷ6 = 200.243 ≈ 200.000
ŷ7 = 203.494 ≈ 204.000
ŷ8 = 201.348 ≈ 201.667
ŷ9 = 200.195 ≈ 200.667
ŷ10 = 200.771 ≈ 200.333
ŷ11 = 201.388 ≈ 202.333
ŷ12 = 203.237 ≈ 202.333
ŷ13 = 200.101 ≈ 200.333
ŷ14 = 200.208 ≈ 200.000
ŷ15 = 201.046 ≈ 201.000

Однорідність дисперсії (критерій Кохрена):
Gp = 0.19858156025577178
Gt = 0.7410730084501662

Дисперсія однорідна (Gp < Gt)

t табличне =  2.0422724563012373
    t0 = 933.4362380294681 => коефіцієнт значимий
    t1 = 0.8029363765480707 => коефіцієнт незначимий, його слід виключити з рів-ня регресії
    t2 = 2.578573743660417 => коефіцієнт значимий
    t3 = 1.6724370183773598 => коефіцієнт незначимий, його слід виключити з рів-ня регресії
    t4 = 0.5156899946221868 => коефіцієнт незначимий, його слід виключити з рів-ня регресії
    t5 = 0.3094263738282604 => коефіцієнт незначимий, його слід виключити з рів-ня регресії
    t6 = 1.1345427422787075 => коефіцієнт незначимий, його слід виключити з рів-ня регресії
    t7 = 0.10316275303433389 => коефіцієнт незначимий, його слід виключити з рів-ня регресії
    t8 = 681.2154054317175 => коефіцієнт значимий
    t9 = 682.8899905193207 => коефіцієнт значимий
    t10 = 680.9109146560928 => коефіцієнт значимий

Отже, кіл-ть значимих коеф. d = 5

    Рів-ня регресії з урахуванням критерія Стьюдента:
ŷ = 200.737 + 0.014*X2 + -0.015*X1^2 + 0.070*X2^2 + -0.049*X3^2

    Перевірка при підстановці в спрощене рів-ня регресії:
y`1 = 200.558 ≈ 200.000
y`2 = 199.819 ≈ 200.000
y`3 = 202.130 ≈ 202.667
y`4 = 201.391 ≈ 200.667
y`5 = 199.819 ≈ 200.667
y`6 = 200.558 ≈ 200.000
y`7 = 201.391 ≈ 204.000
y`8 = 202.130 ≈ 201.667
y`9 = 200.243 ≈ 200.667
y`10 = 200.243 ≈ 200.333
y`11 = 201.117 ≈ 202.333
y`12 = 203.027 ≈ 202.333
y`13 = 200.543 ≈ 200.333
y`14 = 199.286 ≈ 200.000
y`15 = 200.805 ≈ 201.000

Критерій Фішера:
    Рівняння регресії адекватне (Ft > Fp)!
```