

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Методи наукових досліджень

Лабораторна робота №5

«Проведення трьохфакторного експерименту
при використанні рівняння регресії з урахуванням квадратичних членів
(центральний ортогональний композиційний план)»

Виконав:
студент 2 курсу, групи ІВ-91
Коренюк Андрій Олександрович
Залікова книжка № ІВ-9115
Варіант: 14

Перевірив: ас. Рєгіда П.Г.

Мета: провести трьохфакторний експеримент з урахуванням квадратичних членів, використовуючи центральний ортогональний композиційний план. Знайти рівняння регресії, яке буде адекватним для опису об'єкту.

Завдання

| № _{варіанта} | x_1 | | x_2 | | x_3 | |
|-----------------------|-------|-----|-------|-----|-------|-----|
| | min | max | min | max | min | max |
| 114 | -10 | 1 | -6 | 6 | -1 | 10 |

Лістинг програми

configuration.py

```
from math import ceil, floor
```

```
"""Довірча ймовірність  $p = 0.95$  (критерій значимості 0.05)"""
```

```
variant = {"n": 114, "x1min": -10, "x1max": 1, "x2min": -6, "x2max": 6, "x3min": -1, "x3max": 10}
```

```
x_min_average = (variant["x1min"] + variant["x2min"] + variant["x3min"]) / 3
```

```
x_max_average = (variant["x1max"] + variant["x2max"] + variant["x3max"]) / 3
```

```
y_min = ceil(200 + x_min_average)
```

```
y_max = floor(200 + x_max_average)
```

```
x1_average = (variant["x1min"] + variant["x1max"]) / 2
```

```
x2_average = (variant["x2min"] + variant["x2max"]) / 2
```

```
x3_average = (variant["x3min"] + variant["x3max"]) / 2
```

```
del_x1 = variant["x1max"] - x1_average
```

```
del_x2 = variant["x2max"] - x2_average
```

```
del_x3 = variant["x3max"] - x3_average
```

```
x0 = [1, 1, 1, 1]
```

```
x1 = [-1, -1, 1, 1]
```

```
x2 = [-1, 1, -1, 1]
```

```
x3 = [1, -1, -1, 1]
```

```
nx0 = [1, 1, 1, 1]
```

```
nx1 = [variant["x1min"] if x1[i] == -1 else variant["x1max"] for i in range(4)]
```

```
nx2 = [variant["x2min"] if x2[i] == -1 else variant["x2max"] for i in range(4)]
```

```
nx3 = [variant["x3min"] if x3[i] == -1 else variant["x3max"] for i in range(4)]
```

```
sp_x0 = [1, 1, 1, 1]
```

```
sp_x1 = [-1, -1, 1, 1]
```

```
sp_x2 = [-1, 1, -1, 1]
```

```
sp_x3 = [-1, 1, 1, -1]
```

```
sp_nx0 = [1, 1, 1, 1]
```

```
sp_nx1 = [variant["x1min"] if sp_x1[i] == -1 else variant["x1max"] for i in range(4)]
```

```
sp_nx2 = [variant["x2min"] if sp_x2[i] == -1 else variant["x2max"] for i in range(4)]
```

```
sp_nx3 = [variant["x3min"] if sp_x3[i] == -1 else variant["x3max"] for i in range(4)]
```

```
tp_x0 = [1, 1, 1, 1, 1, 1, 1]
```

```
tp_x1 = [-1.215, 1.215, 0, 0, 0, 0, 0]
```

```

tp_x2 = [0, 0, -1.215, 1.215, 0, 0, 0]
tp_x3 = [0, 0, 0, 0, -1.215, 1.215, 0]

tp_nx0 = [1, 1, 1, 1, 1, 1, 1]
tp_nx1 = [tp_x1[i] * del_x1 + x1_average for i in range(7)]
tp_nx2 = [tp_x2[i] * del_x2 + x2_average for i in range(7)]
tp_nx3 = [tp_x3[i] * del_x3 + x3_average for i in range(7)]

```

lab_5.py

```

from experiment import Experiment
from linear_without_interaction import LinearWithoutInteractionModel
from linear_with_interaction import LinearWithInteractionModel
from square_central_orthogonal import SquareCentralOrthogonalModel
from configuration import *
from copy import deepcopy
import logs

```

```

x = list()
nx = list()

```

```

def get_factor_lines(x, N):
    lines = list()
    for i in range(N):
        lines.append([x[0][i], x[1][i], x[2][i], x[3][i]])
    return lines

```

```

def extend_view(step, view, N):
    if step == 2:
        for i in range(N):
            view[i].append(view[i][1] * view[i][2])
            view[i].append(view[i][1] * view[i][3])
            view[i].append(view[i][2] * view[i][3])
            view[i].append(view[i][1] * view[i][2] * view[i][3])
    elif step == 3:
        for i in range(N):
            view[i].append(view[i][1] * view[i][1])
            view[i].append(view[i][2] * view[i][2])
            view[i].append(view[i][3] * view[i][3])

```

```

def linear_model_without_interaction():
    global x
    global nx
    logs.comment(0, [])
    N, K, m = 4, 4, 3
    logs.comment(3, [N, K, m])

    x = [deepcopy(x0), deepcopy(x1), deepcopy(x2), deepcopy(x3)]
    nx = [deepcopy(nx0), deepcopy(nx1), deepcopy(nx2), deepcopy(nx3)]

    logs.comment(4, [])
    # Складання плану експерименту
    x_lines = get_factor_lines(x, N)
    nx_lines = get_factor_lines(nx, N)

    # Виконання експерименту
    experiment = Experiment(y_min, y_max, m, N)
    lm_without = LinearWithoutInteractionModel(K, N)
    experiment.do()

```

```

# Перевірка критерія Кохрена
experiment.check_kohren()
logs.comment(13, [experiment.m, experiment.f1, experiment.f2, experiment.Gp])
logs.show_plan(0, 0, nx_lines, experiment)
logs.show_plan(1, 0, x_lines, experiment)

# Пошук коефіцієнтів
logs.comment(5, [])
lm_without.find_nature_cfs(nx, experiment.y_average)
logs.comment(7, [round(el, 4) for el in lm_without.A])

logs.comment(6, [])
lm_without.find_encoded_cfs(x, experiment.y_average)
logs.comment(8, [round(el, 4) for el in lm_without.B])

logs.show_natured_checking_matrix(6, 0, nx_lines, lm_without, experiment)
logs.show_encoded_checking_matrix(7, 0, x_lines, lm_without, experiment)

# Перевірка критерія Стюдента
experiment.check_student(lm_without.K, lm_without.A, lm_without.B)
logs.comment(14, [experiment.f3, [round(el, 4) for el in experiment.t]])
logs.comment(15, [round(el, 4) for el in lm_without.A])
logs.comment(16, [round(el, 4) for el in lm_without.B])

# Перевірка критерія Фішера
is_suitable = experiment.check_fisher(lm_without, nx_lines)
logs.comment(21, [experiment.f3, experiment.f4, experiment.Fp])

# Передача даних
if is_suitable:
    logs.comment(22, [])
    logs.comment(25, [])
    logs.show_natured_checking_matrix(6, 0, nx_lines, lm_without, experiment)
    logs.show_encoded_checking_matrix(7, 0, x_lines, lm_without, experiment)
    del x, nx
else:
    logs.comment(23, [])
    logs.comment(24, [])

# Видалення зайвого
del lm_without, experiment
return is_suitable

def linear_model_with_interaction():
    global x
    global nx
    logs.comment(1, [])
    N, K, m = 8, 8, 3
    logs.comment(3, [N, K, m])

    x[0].extend(sp_x0)
    x[1].extend(sp_x1)
    x[2].extend(sp_x2)
    x[3].extend(sp_x3)
    nx[0].extend(sp_nx0)
    nx[1].extend(sp_nx1)
    nx[2].extend(sp_nx2)
    nx[3].extend(sp_nx3)

    logs.comment(4, [])

```

```

# Складання плану експерименту
x_lines = get_factor_lines(x, N)
nx_lines = get_factor_lines(nx, N)
x_views = deepcopy(x_lines)
nx_views = deepcopy(nx_lines)

extend_view(2, x_views, N)
extend_view(2, nx_views, N)

# Виконання експерименту
experiment = Experiment(y_min, y_max, m, N)
lm_with = LinearWithInteractionModel(K, N)
experiment.do()

# Перевірка критерія Кохрена
experiment.check_kohren()
logs.comment(13, [experiment.m, experiment.f1, experiment.f2, experiment.Gp])
logs.show_plan(2, 1, nx_views, experiment)
logs.show_plan(3, 1, x_views, experiment)

# Пошук коефіцієнтів
logs.comment(5, [])
lm_with.find_nature_cfs(nx, experiment.y_average)
logs.comment(9, [round(el, 4) for el in lm_with.A])

logs.comment(6, [])
lm_with.find_encoded_cfs(x, experiment.y_average)
logs.comment(10, [round(el, 4) for el in lm_with.B])

logs.show_natured_checking_matrix(8, 1, nx_views, lm_with, experiment)
logs.show_encoded_checking_matrix(9, 1, x_views, lm_with, experiment)

# Перевірка критерія Стюдента
experiment.check_student(lm_with.K, lm_with.A, lm_with.B)
logs.comment(14, [experiment.f3, [round(el, 4) for el in experiment.t]])
logs.comment(17, [round(el, 4) for el in lm_with.A])
logs.comment(18, [round(el, 4) for el in lm_with.B])

# Перевірка критерія Фішера
is_suitable = experiment.check_fisher(lm_with, nx_lines)
logs.comment(21, [experiment.f3, experiment.f4, experiment.Fp])

# Передача даних
if is_suitable:
    logs.comment(22, [])
    logs.comment(25, [])
    logs.show_natured_checking_matrix(8, 1, nx_views, lm_with, experiment)
    logs.show_encoded_checking_matrix(9, 1, x_views, lm_with, experiment)
    del x, nx
else:
    logs.comment(23, [])
    logs.comment(24, [])

# Видалення зайвого
del x_views, nx_views, lm_with, experiment
return is_suitable

def square_central_orthogonal_model():
    global x
    global nx
    logs.comment(2, [])

```

```

N, K, m = 15, 11, 3
logs.comment(3, [N, K, m])

x[0].extend(tp_x0)
x[1].extend(tp_x1)
x[2].extend(tp_x2)
x[3].extend(tp_x3)
nx[0].extend(tp_nx0)
nx[1].extend(tp_nx1)
nx[2].extend(tp_nx2)
nx[3].extend(tp_nx3)

logs.comment(4, [])
# Складання плану експерименту
x_lines = get_factor_lines(x, N)
nx_lines = get_factor_lines(nx, N)
x_views = deepcopy(x_lines)
nx_views = deepcopy(nx_lines)

extend_view(2, x_views, N)
extend_view(2, nx_views, N)
extend_view(3, x_views, N)
extend_view(3, nx_views, N)

# Виконання експерименту
experiment = Experiment(y_min, y_max, m, N)
sq_co = SquareCentralOrthogonalModel(K, N)
experiment.do()

# Перевірка критерія Кохрена
experiment.check_kohren()
logs.comment(13, [experiment.m, experiment.f1, experiment.f2, experiment.Gp])
logs.show_plan(4, 2, nx_views, experiment)
logs.show_plan(5, 2, x_views, experiment)

# Пошук коефіцієнтів
logs.comment(5, [])
sq_co.find_nature_cfs(experiment.m, nx, experiment.y)
logs.comment(11, [round(el, 4) for el in sq_co.A])

logs.comment(6, [])
sq_co.find_encoded_cfs(experiment.m, x, experiment.y)
logs.comment(12, [round(el, 4) for el in sq_co.B])

logs.show_natured_checking_matrix(10, 2, nx_views, sq_co, experiment)
logs.show_encoded_checking_matrix(11, 2, x_views, sq_co, experiment)
# Перевірка критерія Стьюдента
experiment.check_student(sq_co.K, sq_co.A, sq_co.B)
logs.comment(14, [experiment.f3, [round(el, 4) for el in experiment.t]])
logs.comment(19, [round(el, 4) for el in sq_co.A])
logs.comment(20, [round(el, 4) for el in sq_co.B])

# Перевірка критерія Фішера
is_suitable = experiment.check_fisher(sq_co, nx_lines)
logs.comment(21, [experiment.f3, experiment.f4, experiment.Fp])

# Передача даних
if is_suitable:
    logs.comment(22, [])
    logs.comment(25, [])
    logs.show_natured_checking_matrix(10, 2, nx_views, sq_co, experiment)
    logs.show_encoded_checking_matrix(11, 2, x_views, sq_co, experiment)

```

```

    del x, nx
else:
    logs.comment(23, [])
    logs.comment(24, [])

# Видалення зайвого
del x_views, nx_views, sq_co, experiment
return is_suitable

def main():
    while True:
        if linear_model_without_interaction():
            break
        elif linear_model_with_interaction():
            break
        elif square_central_orthogonal_model():
            break
        print(logs.comment(26, []))

if __name__ == "__main__":
    main()

```

experiment.py

```

from random import randint
from math import sqrt

import criterion_tables as ct

class Experiment:
    m = 0
    N = 0
    y_min, y_max = 0, 0
    y = list()
    y_average = list()
    S2_dis = list()
    f1, f2, f3, f4 = 0, 0, 0, 0
    Gp = 0
    t = list()
    Fp = 0
    d = 0
    s2b = 0

    def __init__(self, y_min, y_max, m, N):
        self.y_min = y_min
        self.y_max = y_max
        self.m = m
        self.N = N

    def __del__(self):
        del self.m, self.N, self.y_min, self.y_max
        del self.y, self.y_average, self.S2_dis
        del self.f1, self.f2, self.f3, self.f4
        del self.Gp, self.t, self.Fp, self.d, self.s2b

    def do(self):
        self.y = list()
        for i in range(self.N):
            self.y.append([randint(self.y_min, self.y_max) for _ in range(self.m)])

```

```

def do_more(self):
    for i in range(self.N):
        self.y[i].append([randint(self.y_min, self.y_max)])

def get_y(self):
    return self.y

def check_kohren(self):
    self.y_average = [0 for _ in range(self.N)]
    for i in range(self.N):
        self.y_average[i] = sum(self.y[i]) / self.m

    self.S2_dis = [0 for _ in range(self.N)]
    for i in range(self.N):
        for j in range(self.m):
            self.S2_dis[i] += (self.y[i][j] - self.y_average[i]) ** 2
    self.S2_dis[i] /= self.m

    self.Gp = max(self.S2_dis) / sum(self.S2_dis)
    self.f1 = self.m - 1
    self.f2 = self.N
    if not(ct.compare_kohren_with_table_value(self.f1, self.f2, self.Gp)):
        self.m += 1
        self.do_more()
    return self.check_kohren()

def check_student(self, K, A, B):
    self.s2b = sum(self.S2_dis) / self.N
    s2_b = self.s2b / (self.N * self.m)
    s_b = sqrt(s2_b)
    self.t = [0 for _ in range(K)]
    for i in range(K):
        self.t[i] = abs(B[i]) / s_b
    self.f3 = self.f1 * self.f2

    self.d = K
    for i in range(K):
        if not ct.compare_student_with_table_value(self.f3, self.t[i]):
            A[i] = 0
            B[i] = 0
            self.d -= 1

def check_fisher(self, model, nx_lines):
    y_for_fisher = [0 for _ in range(self.N)]
    for i in range(self.N):
        y_for_fisher[i] = model.calculate_with_nature_cfs(nx_lines[i])
    s2ad = 0
    for i in range(self.N):
        s2ad += (y_for_fisher[i] - self.y_average[i]) ** 2
    s2ad = self.m * s2ad / (self.N - self.d)
    self.f4 = self.N - self.d
    self.Fp = s2ad / self.s2b
    return ct.compare_phisher_with_table_value(self.f3, self.f4, self.Fp)

```

linear_without_interaction.py

```
from numpy.linalg import det
```

```
class LinearWithoutInteractionModel:
    K = 0
```



```

N = 0
A = list()
B = list()

def __init__(self, K, N):
    self.K = K
    self.N = N

def __del__(self):
    del self.K, self.N, self.A, self.B

def find_nature_cfs(self, nx, y_average):
    """nx - матрица натуральных значений x"""
    mx1, mx2, mx3, my = sum(nx[1]) / self.N, sum(nx[2]) / self.N, sum(nx[3]) / self.N, sum(y_average) / self.N
    a11, a22, a33 = 0, 0, 0
    a12, a13, a23 = 0, 0, 0
    a1, a2, a3 = 0, 0, 0
    for i in range(self.N):
        a11 += nx[1][i] ** 2
        a22 += nx[2][i] ** 2
        a33 += nx[3][i] ** 2
        a12 += nx[1][i] * nx[2][i]
        a13 += nx[1][i] * nx[3][i]
        a23 += nx[2][i] * nx[3][i]
        a1 += y_average[i] * nx[1][i]
        a2 += y_average[i] * nx[2][i]
        a3 += y_average[i] * nx[3][i]
    a11, a22, a33 = a11 / self.N, a22 / self.N, a33 / self.N
    a12, a13, a23 = a12 / self.N, a13 / self.N, a23 / self.N
    a1, a2, a3 = a1 / self.N, a2 / self.N, a3 / self.N
    a21 = a12
    a31 = a13
    a32 = a23

    main_det = det([[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a21, a22, a23], [mx3, a31, a32, a33]])
    A0 = det([[my, mx1, mx2, mx3], [a1, a11, a12, a13], [a2, a21, a22, a23], [a3, a31, a32, a33]]) / main_det
    A1 = det([[1, my, mx2, mx3], [mx1, a1, a12, a13], [mx2, a2, a22, a23], [mx3, a3, a32, a33]]) / main_det
    A2 = det([[1, mx1, my, mx3], [mx1, a11, a1, a13], [mx2, a21, a2, a23], [mx3, a31, a3, a33]]) / main_det
    A3 = det([[1, mx1, mx2, my], [mx1, a11, a12, a1], [mx2, a21, a22, a2], [mx3, a31, a32, a3]]) / main_det
    self.A = [A0, A1, A2, A3]

def find_encoded_cfs(self, x, y_average):
    """x - матрица натуральных значений факторов"""
    self.B = [0 for _ in range(self.K)]
    for i in range(self.N):
        self.B[0] += y_average[i] * x[0][i]
        self.B[1] += y_average[i] * x[1][i]
        self.B[2] += y_average[i] * x[2][i]
        self.B[3] += y_average[i] * x[3][i]

    for i in range(self.K):
        self.B[i] /= self.N

def calculate_with_nature_cfs(self, nxl):
    """nxl - nature x line"""
    return self.A[0] * nxl[0] + self.A[1] * nxl[1] + self.A[2] * nxl[2] + self.A[3] * nxl[3]

def calculate_with_encoded_cfs(self, xl):
    """xl - encoded x line"""
    return self.B[0] * xl[0] + self.B[1] * xl[1] + self.B[2] * xl[2] + self.B[3] * xl[3]

```

linear_with_interaction.py

```
from numpy.linalg import det
```

```
class LinearWithInteractionModel:
```

```
    K = 0
```

```
    N = 0
```

```
    A = list()
```

```
    B = list()
```

```
    def __init__(self, K, N):
```

```
        self.K = K
```

```
        self.N = N
```

```
    def __del__(self):
```

```
        del self.K, self.N, self.A, self.B
```

```
    def find_nature_cfs(self, nx, y_average):
```

```
        """nx - матриця натуральних значень x"""
```

```
        m00, m10, m20, m30, m40, m50, m60, m70, k0 = 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
        m01, m11, m21, m31, m41, m51, m61, m71, k1 = 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
        m02, m12, m22, m32, m42, m52, m62, m72, k2 = 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
        m03, m13, m23, m33, m43, m53, m63, m73, k3 = 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
        m04, m14, m24, m34, m44, m54, m64, m74, k4 = 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
        m05, m15, m25, m35, m45, m55, m65, m75, k5 = 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
        m06, m16, m26, m36, m46, m56, m66, m76, k6 = 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
        m07, m17, m27, m37, m47, m57, m67, m77, k7 = 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
    for i in range(self.N):
```

```
        m00 += self.N
```

```
        m10 += nx[1][i]
```

```
        m20 += nx[2][i]
```

```
        m30 += nx[3][i]
```

```
        m40 += nx[1][i] * nx[2][i]
```

```
        m50 += nx[1][i] * nx[3][i]
```

```
        m60 += nx[2][i] * nx[3][i]
```

```
        m70 += nx[1][i] * nx[2][i] * nx[3][i]
```

```
        k0 += y_average[i]
```

```
        m01 += nx[1][i]
```

```
        m11 += nx[1][i] ** 2
```

```
        m21 += nx[1][i] * nx[2][i]
```

```
        m31 += nx[1][i] * nx[3][i]
```

```
        m41 += (nx[1][i] ** 2) * nx[2][i]
```

```
        m51 += (nx[1][i] ** 2) * nx[3][i]
```

```
        m61 += nx[1][i] * nx[2][i] * nx[3][i]
```

```
        m71 += (nx[1][i] ** 2) * nx[2][i] * nx[3][i]
```

```
        k1 += y_average[i] * nx[1][i]
```

```
        m02 += nx[2][i]
```

```
        m12 += nx[1][i] * nx[2][i]
```

```
        m22 += nx[2][i] ** 2
```

```
        m32 += nx[2][i] * nx[3][i]
```

```
        m42 += nx[1][i] * (nx[2][i] ** 2)
```

```
        m52 += nx[1][i] * nx[2][i] * nx[3][i]
```

```
        m62 += (nx[2][i] ** 2) * nx[3][i]
```

```
        m72 += nx[1][i] * (nx[2][i] ** 2) * nx[3][i]
```

```
        k2 += y_average[i] * nx[2][i]
```

```
        m03 += nx[3][i]
```

```
        m13 += nx[1][i] * nx[3][i]
```

```
        m23 += nx[2][i] * nx[3][i]
```

```
        m33 += nx[3][i] ** 2
```

```
        m43 += nx[1][i] * nx[2][i] * nx[3][i]
```

```
        m53 += nx[1][i] * (nx[3][i] ** 2)
```

```
        m63 += nx[2][i] * (nx[3][i] ** 2)
```

```

m73 += nx[1][i] * nx[2][i] * (nx[3][i] ** 2)
k3 += y_average[i] * nx[3][i]
m04 += nx[1][i] * nx[2][i]
m14 += (nx[1][i] ** 2) * nx[2][i]
m24 += nx[1][i] * (nx[2][i] ** 2)
m34 += nx[1][i] * nx[2][i] * nx[3][i]
m44 += (nx[1][i] ** 2) * (nx[2][i] ** 2)
m54 += (nx[1][i] ** 2) * nx[2][i] * nx[3][i]
m64 += nx[1][i] * (nx[2][i] ** 2) * nx[3][i]
m74 += (nx[1][i] ** 2) * (nx[2][i] ** 2) * nx[3][i]
k4 += y_average[i] * nx[1][i] * nx[2][i]
m05 += nx[1][i] * nx[3][i]
m15 += (nx[1][i] ** 2) * nx[3][i]
m25 += nx[1][i] * nx[2][i] * nx[3][i]
m35 += nx[1][i] * (nx[3][i] ** 2)
m45 += (nx[1][i] ** 2) * nx[2][i] * nx[3][i]
m55 += (nx[1][i] ** 2) * (nx[3][i] ** 2)
m65 += nx[1][i] * nx[2][i] * (nx[3][i] ** 2)
m75 += (nx[1][i] ** 2) * nx[2][i] * (nx[3][i] ** 2)
k5 += y_average[i] * nx[1][i] * nx[3][i]
m06 += nx[2][i] * nx[3][i]
m16 += nx[1][i] * nx[2][i] * nx[3][i]
m26 += (nx[2][i] ** 2) * nx[3][i]
m36 += nx[2][i] * (nx[3][i] ** 2)
m46 += nx[1][i] * (nx[2][i] ** 2) * nx[3][i]
m56 += nx[1][i] * nx[2][i] * (nx[3][i] ** 2)
m66 += (nx[2][i] ** 2) * (nx[3][i] ** 2)
m76 += nx[1][i] * (nx[2][i] ** 2) * (nx[3][i] ** 2)
k6 += y_average[i] * nx[2][i] * nx[3][i]
m07 += nx[1][i] * nx[2][i] * nx[3][i]
m17 += (nx[1][i] ** 2) * nx[2][i] * nx[3][i]
m27 += nx[1][i] * (nx[2][i] ** 2) * nx[3][i]
m37 += nx[1][i] * nx[2][i] * (nx[3][i] ** 2)
m47 += (nx[1][i] ** 2) * (nx[2][i] ** 2) * nx[3][i]
m57 += (nx[1][i] ** 2) * nx[2][i] * (nx[3][i] ** 2)
m67 += nx[1][i] * (nx[2][i] ** 2) * (nx[3][i] ** 2)
m77 += (nx[1][i] ** 2) * (nx[2][i] ** 2) * (nx[3][i] ** 2)
k7 += y_average[i] * nx[1][i] * nx[2][i] * nx[3][i]

main_det = det([
    [m00, m10, m20, m30, m40, m50, m60, m70],
    [m01, m11, m21, m31, m41, m51, m61, m71],
    [m02, m12, m22, m32, m42, m52, m62, m72],
    [m03, m13, m23, m33, m43, m53, m63, m73],
    [m04, m14, m24, m34, m44, m54, m64, m74],
    [m05, m15, m25, m35, m45, m55, m65, m75],
    [m06, m16, m26, m36, m46, m56, m66, m76],
    [m07, m17, m27, m37, m47, m57, m67, m77]])

det0 = det([
    [k0, m10, m20, m30, m40, m50, m60, m70],
    [k1, m11, m21, m31, m41, m51, m61, m71],
    [k2, m12, m22, m32, m42, m52, m62, m72],
    [k3, m13, m23, m33, m43, m53, m63, m73],
    [k4, m14, m24, m34, m44, m54, m64, m74],
    [k5, m15, m25, m35, m45, m55, m65, m75],
    [k6, m16, m26, m36, m46, m56, m66, m76],
    [k7, m17, m27, m37, m47, m57, m67, m77]])

det1 = det([
    [m00, k0, m20, m30, m40, m50, m60, m70],
    [m01, k1, m21, m31, m41, m51, m61, m71],

```

```
[m02, k2, m22, m32, m42, m52, m62, m72],
[m03, k3, m23, m33, m43, m53, m63, m73],
[m04, k4, m24, m34, m44, m54, m64, m74],
[m05, k5, m25, m35, m45, m55, m65, m75],
[m06, k6, m26, m36, m46, m56, m66, m76],
[m07, k7, m27, m37, m47, m57, m67, m77]]])
```

```
det2 = det([
[m00, m10, k0, m30, m40, m50, m60, m70],
[m01, m11, k1, m31, m41, m51, m61, m71],
[m02, m12, k2, m32, m42, m52, m62, m72],
[m03, m13, k3, m33, m43, m53, m63, m73],
[m04, m14, k4, m34, m44, m54, m64, m74],
[m05, m15, k5, m35, m45, m55, m65, m75],
[m06, m16, k6, m36, m46, m56, m66, m76],
[m07, m17, k7, m37, m47, m57, m67, m77]])
```

```
det3 = det([
[m00, m10, m20, k0, m40, m50, m60, m70],
[m01, m11, m21, k1, m41, m51, m61, m71],
[m02, m12, m22, k2, m42, m52, m62, m72],
[m03, m13, m23, k3, m43, m53, m63, m73],
[m04, m14, m24, k4, m44, m54, m64, m74],
[m05, m15, m25, k5, m45, m55, m65, m75],
[m06, m16, m26, k6, m46, m56, m66, m76],
[m07, m17, m27, k7, m47, m57, m67, m77]])
```

```
det4 = det([
[m00, m10, m20, m30, k0, m50, m60, m70],
[m01, m11, m21, m31, k1, m51, m61, m71],
[m02, m12, m22, m32, k2, m52, m62, m72],
[m03, m13, m23, m33, k3, m53, m63, m73],
[m04, m14, m24, m34, k4, m54, m64, m74],
[m05, m15, m25, m35, k5, m55, m65, m75],
[m06, m16, m26, m36, k6, m56, m66, m76],
[m07, m17, m27, m37, k7, m57, m67, m77]])
```

```
det5 = det([
[m00, m10, m20, m30, m40, k0, m60, m70],
[m01, m11, m21, m31, m41, k1, m61, m71],
[m02, m12, m22, m32, m42, k2, m62, m72],
[m03, m13, m23, m33, m43, k3, m63, m73],
[m04, m14, m24, m34, m44, k4, m64, m74],
[m05, m15, m25, m35, m45, k5, m65, m75],
[m06, m16, m26, m36, m46, k6, m66, m76],
[m07, m17, m27, m37, m47, k7, m67, m77]])
```

```
det6 = det([
[m00, m10, m20, m30, m40, m50, k0, m70],
[m01, m11, m21, m31, m41, m51, k1, m71],
[m02, m12, m22, m32, m42, m52, k2, m72],
[m03, m13, m23, m33, m43, m53, k3, m73],
[m04, m14, m24, m34, m44, m54, k4, m74],
[m05, m15, m25, m35, m45, m55, k5, m75],
[m06, m16, m26, m36, m46, m56, k6, m76],
[m07, m17, m27, m37, m47, m57, k7, m77]])
```

```
det7 = det([
[m00, m10, m20, m30, m40, m50, m60, k0],
[m01, m11, m21, m31, m41, m51, m61, k1],
[m02, m12, m22, m32, m42, m52, m62, k2],
[m03, m13, m23, m33, m43, m53, m63, k3],
```

```

[m04, m14, m24, m34, m44, m54, m64, k4],
[m05, m15, m25, m35, m45, m55, m65, k5],
[m06, m16, m26, m36, m46, m56, m66, k6],
[m07, m17, m27, m37, m47, m57, m67, k7]])

```

```

A0 = det0 / main_det
A1 = det1 / main_det
A2 = det2 / main_det
A3 = det3 / main_det
A4 = det4 / main_det
A5 = det5 / main_det
A6 = det6 / main_det
A7 = det7 / main_det

```

```

self.A = [A0, A1, A2, A3, A4, A5, A6, A7]

```

```

def find_encoded_cfs(self, x, y_average):
    """x - матриця натуральних значень факторів"""

```

```

    self.B = [0 for _ in range(self.K)]
    for i in range(self.N):
        self.B[0] += y_average[i] * x[0][i]
        self.B[1] += y_average[i] * x[1][i]
        self.B[2] += y_average[i] * x[2][i]
        self.B[3] += y_average[i] * x[3][i]
        self.B[4] += y_average[i] * x[1][i] * x[2][i]
        self.B[5] += y_average[i] * x[1][i] * x[3][i]
        self.B[6] += y_average[i] * x[2][i] * x[3][i]
        self.B[7] += y_average[i] * x[1][i] * x[2][i] * x[3][i]
    for i in range(self.K):
        self.B[i] /= self.N

```

```

def calculate_with_nature_cfs(self, nxl):

```

```

    """nxl - nature x line"""
    return self.A[0]*nxl[0] + self.A[1]*nxl[1] + self.A[2]*nxl[2] + self.A[3]*nxl[3] + \
        self.A[4]*nxl[1]*nxl[2] + self.A[5]*nxl[1]*nxl[3] + self.A[6]*nxl[2]*nxl[3] + \
        self.A[7]*nxl[1]*nxl[2]*nxl[3]

```

```

def calculate_with_encoded_cfs(self, xl):

```

```

    """xl - encoded x line"""
    return self.B[0]*xl[0] + self.B[1]*xl[1] + self.B[2]*xl[2] + self.B[3]*xl[3] + \
        self.B[4]*xl[1]*xl[2] + self.B[5]*xl[1]*xl[3] + self.B[6]*xl[2]*xl[3] + \
        self.B[7]*xl[1]*xl[2]*xl[3]

```

square_central_orthogonal.py

```

from numpy.linalg import det

```

```

class SquareCentralOrthogonalModel:

```

```

    K = 0
    N = 0
    A = list()
    B = list()

```

```

    def __init__(self, K, N):

```

```

        self.K = K
        self.N = N

```

```

    def __del__(self):

```

```

        del self.K, self.N, self.A, self.B

```

```

    def find_nature_cfs(self, m, nx, y):

```

```

        """nx - матриця натуральних значень x"""

```

```

nx.append([nx[1][i] * nx[2][i] for i in range(self.N)])
nx.append([nx[1][i] * nx[3][i] for i in range(self.N)])
nx.append([nx[2][i] * nx[3][i] for i in range(self.N)])
nx.append([nx[1][i] * nx[2][i] * nx[3][i] for i in range(self.N)])
nx.append([nx[1][i] ** 2 for i in range(self.N)])
nx.append([nx[2][i] ** 2 for i in range(self.N)])
nx.append([nx[3][i] ** 2 for i in range(self.N)])
self.A = self.find_cfs_core(m, nx, y)

```

```

def find_encoded_cfs(self, m, x, y):

```

```

    """x - матриця натуральних значень факторів"""
    x.append([x[1][i] * x[2][i] for i in range(self.N)])
    x.append([x[1][i] * x[3][i] for i in range(self.N)])
    x.append([x[2][i] * x[3][i] for i in range(self.N)])
    x.append([x[1][i] * x[2][i] * x[3][i] for i in range(self.N)])
    x.append([x[1][i] ** 2 for i in range(self.N)])
    x.append([x[2][i] ** 2 for i in range(self.N)])
    x.append([x[3][i] ** 2 for i in range(self.N)])
    self.B = self.find_cfs_core(m, x, y)

```

```

def find_cfs_core(self, m, x, y):

```

```

    m00, m10, m20, m30, m40, m50, m60, m70, m80, m90, m100, k0 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m01, m11, m21, m31, m41, m51, m61, m71, m81, m91, m101, k1 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m02, m12, m22, m32, m42, m52, m62, m72, m82, m92, m102, k2 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m03, m13, m23, m33, m43, m53, m63, m73, m83, m93, m103, k3 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m04, m14, m24, m34, m44, m54, m64, m74, m84, m94, m104, k4 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m05, m15, m25, m35, m45, m55, m65, m75, m85, m95, m105, k5 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m06, m16, m26, m36, m46, m56, m66, m76, m86, m96, m106, k6 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m07, m17, m27, m37, m47, m57, m67, m77, m87, m97, m107, k7 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m08, m18, m28, m38, m48, m58, m68, m78, m88, m98, m108, k8 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m09, m19, m29, m39, m49, m59, m69, m79, m89, m99, m109, k9 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    m010, m110, m210, m310, m410, m510, m610, m710, m810, m910, m1010, k10 = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

```

0

```

    for i in range(self.N):

```

```

        for j in range(m):
            m00 += x[0][i]
            m10 += x[1][i]
            m20 += x[2][i]
            m30 += x[3][i]
            m40 += x[4][i]
            m50 += x[5][i]
            m60 += x[6][i]
            m70 += x[7][i]
            m80 += x[8][i]
            m90 += x[9][i]
            m100 += x[10][i]
            k0 += y[i][j]
            m01 += x[0][i] * x[1][i]
            m11 += x[1][i] * x[1][i]
            m21 += x[2][i] * x[1][i]
            m31 += x[3][i] * x[1][i]
            m41 += x[4][i] * x[1][i]
            m51 += x[5][i] * x[1][i]
            m61 += x[6][i] * x[1][i]
            m71 += x[7][i] * x[1][i]
            m81 += x[8][i] * x[1][i]
            m91 += x[9][i] * x[1][i]
            m101 += x[10][i] * x[1][i]
            k1 += y[i][j] * x[1][i]
            m02 += x[0][i] * x[2][i]

```

```
m12 += x[1][i] * x[2][i]
m22 += x[2][i] * x[2][i]
m32 += x[3][i] * x[2][i]
m42 += x[4][i] * x[2][i]
m52 += x[5][i] * x[2][i]
m62 += x[6][i] * x[2][i]
m72 += x[7][i] * x[2][i]
m82 += x[8][i] * x[2][i]
m92 += x[9][i] * x[2][i]
m102 += x[10][i] * x[2][i]
k2 += y[i][j] * x[2][i]
m03 += x[0][i] * x[3][i]
m13 += x[1][i] * x[3][i]
m23 += x[2][i] * x[3][i]
m33 += x[3][i] * x[3][i]
m43 += x[4][i] * x[3][i]
m53 += x[5][i] * x[3][i]
m63 += x[6][i] * x[3][i]
m73 += x[7][i] * x[3][i]
m83 += x[8][i] * x[3][i]
m93 += x[9][i] * x[3][i]
m103 += x[10][i] * x[3][i]
k3 += y[i][j] * x[3][i]
m04 += x[0][i] * x[4][i]
m14 += x[1][i] * x[4][i]
m24 += x[2][i] * x[4][i]
m34 += x[3][i] * x[4][i]
m44 += x[4][i] * x[4][i]
m54 += x[5][i] * x[4][i]
m64 += x[6][i] * x[4][i]
m74 += x[7][i] * x[4][i]
m84 += x[8][i] * x[4][i]
m94 += x[9][i] * x[4][i]
m104 += x[10][i] * x[4][i]
k4 += y[i][j] * x[4][i]
m05 += x[0][i] * x[5][i]
m15 += x[1][i] * x[5][i]
m25 += x[2][i] * x[5][i]
m35 += x[3][i] * x[5][i]
m45 += x[4][i] * x[5][i]
m55 += x[5][i] * x[5][i]
m65 += x[6][i] * x[5][i]
m75 += x[7][i] * x[5][i]
m85 += x[8][i] * x[5][i]
m95 += x[9][i] * x[5][i]
m105 += x[10][i] * x[5][i]
k5 += y[i][j] * x[5][i]
m06 += x[0][i] * x[6][i]
m16 += x[1][i] * x[6][i]
m26 += x[2][i] * x[6][i]
m36 += x[3][i] * x[6][i]
m46 += x[4][i] * x[6][i]
m56 += x[5][i] * x[6][i]
m66 += x[6][i] * x[6][i]
m76 += x[7][i] * x[6][i]
m86 += x[8][i] * x[6][i]
m96 += x[9][i] * x[6][i]
m106 += x[10][i] * x[6][i]
k6 += y[i][j] * x[6][i]
m07 += x[0][i] * x[7][i]
m17 += x[1][i] * x[7][i]
m27 += x[2][i] * x[7][i]
```

```

m37 += x[3][i] * x[7][i]
m47 += x[4][i] * x[7][i]
m57 += x[5][i] * x[7][i]
m67 += x[6][i] * x[7][i]
m77 += x[7][i] * x[7][i]
m87 += x[8][i] * x[7][i]
m97 += x[9][i] * x[7][i]
m107 += x[10][i] * x[7][i]
k7 += y[i][j] * x[7][i]
m08 += x[0][i] * x[8][i]
m18 += x[1][i] * x[8][i]
m28 += x[2][i] * x[8][i]
m38 += x[3][i] * x[8][i]
m48 += x[4][i] * x[8][i]
m58 += x[5][i] * x[8][i]
m68 += x[6][i] * x[8][i]
m78 += x[7][i] * x[8][i]
m88 += x[8][i] * x[8][i]
m98 += x[9][i] * x[8][i]
m108 += x[10][i] * x[8][i]
k8 += y[i][j] * x[8][i]
m09 += x[0][i] * x[9][i]
m19 += x[1][i] * x[9][i]
m29 += x[2][i] * x[9][i]
m39 += x[3][i] * x[9][i]
m49 += x[4][i] * x[9][i]
m59 += x[5][i] * x[9][i]
m69 += x[6][i] * x[9][i]
m79 += x[7][i] * x[9][i]
m89 += x[8][i] * x[9][i]
m99 += x[9][i] * x[9][i]
m109 += x[10][i] * x[9][i]
k9 += y[i][j] * x[9][i]
m010 += x[0][i] * x[10][i]
m110 += x[1][i] * x[10][i]
m210 += x[2][i] * x[10][i]
m310 += x[3][i] * x[10][i]
m410 += x[4][i] * x[10][i]
m510 += x[5][i] * x[10][i]
m610 += x[6][i] * x[10][i]
m710 += x[7][i] * x[10][i]
m810 += x[8][i] * x[10][i]
m910 += x[9][i] * x[10][i]
m1010 += x[10][i] * x[10][i]
k10 += y[i][j] * x[10][i]

```

```

main_det = det([
    [m00, m10, m20, m30, m40, m50, m60, m70, m80, m90, m100],
    [m01, m11, m21, m31, m41, m51, m61, m71, m81, m91, m101],
    [m02, m12, m22, m32, m42, m52, m62, m72, m82, m92, m102],
    [m03, m13, m23, m33, m43, m53, m63, m73, m83, m93, m103],
    [m04, m14, m24, m34, m44, m54, m64, m74, m84, m94, m104],
    [m05, m15, m25, m35, m45, m55, m65, m75, m85, m95, m105],
    [m06, m16, m26, m36, m46, m56, m66, m76, m86, m96, m106],
    [m07, m17, m27, m37, m47, m57, m67, m77, m87, m97, m107],
    [m08, m18, m28, m38, m48, m58, m68, m78, m88, m98, m108],
    [m09, m19, m29, m39, m49, m59, m69, m79, m89, m99, m109],
    [m010, m110, m210, m310, m410, m510, m610, m710, m810, m910, m1010]
])

```

```

det0 = det([
    [k0, m10, m20, m30, m40, m50, m60, m70, m80, m90, m100],

```



```

[k1, m11, m21, m31, m41, m51, m61, m71, m81, m91, m101],
[k2, m12, m22, m32, m42, m52, m62, m72, m82, m92, m102],
[k3, m13, m23, m33, m43, m53, m63, m73, m83, m93, m103],
[k4, m14, m24, m34, m44, m54, m64, m74, m84, m94, m104],
[k5, m15, m25, m35, m45, m55, m65, m75, m85, m95, m105],
[k6, m16, m26, m36, m46, m56, m66, m76, m86, m96, m106],
[k7, m17, m27, m37, m47, m57, m67, m77, m87, m97, m107],
[k8, m18, m28, m38, m48, m58, m68, m78, m88, m98, m108],
[k9, m19, m29, m39, m49, m59, m69, m79, m89, m99, m109],
[k10, m110, m210, m310, m410, m510, m610, m710, m810, m910, m1010]
])

```

```

det1 = det([
[m00, k0, m20, m30, m40, m50, m60, m70, m80, m90, m100],
[m01, k1, m21, m31, m41, m51, m61, m71, m81, m91, m101],
[m02, k2, m22, m32, m42, m52, m62, m72, m82, m92, m102],
[m03, k3, m23, m33, m43, m53, m63, m73, m83, m93, m103],
[m04, k4, m24, m34, m44, m54, m64, m74, m84, m94, m104],
[m05, k5, m25, m35, m45, m55, m65, m75, m85, m95, m105],
[m06, k6, m26, m36, m46, m56, m66, m76, m86, m96, m106],
[m07, k7, m27, m37, m47, m57, m67, m77, m87, m97, m107],
[m08, k8, m28, m38, m48, m58, m68, m78, m88, m98, m108],
[m09, k9, m29, m39, m49, m59, m69, m79, m89, m99, m109],
[m010, k10, m210, m310, m410, m510, m610, m710, m810, m910, m1010]
])

```

```

det2 = det([
[m00, m10, k0, m30, m40, m50, m60, m70, m80, m90, m100],
[m01, m11, k1, m31, m41, m51, m61, m71, m81, m91, m101],
[m02, m12, k2, m32, m42, m52, m62, m72, m82, m92, m102],
[m03, m13, k3, m33, m43, m53, m63, m73, m83, m93, m103],
[m04, m14, k4, m34, m44, m54, m64, m74, m84, m94, m104],
[m05, m15, k5, m35, m45, m55, m65, m75, m85, m95, m105],
[m06, m16, k6, m36, m46, m56, m66, m76, m86, m96, m106],
[m07, m17, k7, m37, m47, m57, m67, m77, m87, m97, m107],
[m08, m18, k8, m38, m48, m58, m68, m78, m88, m98, m108],
[m09, m19, k9, m39, m49, m59, m69, m79, m89, m99, m109],
[m010, m110, k10, m310, m410, m510, m610, m710, m810, m910, m1010]
])

```

```

det3 = det([
[m00, m10, m20, k0, m40, m50, m60, m70, m80, m90, m100],
[m01, m11, m21, k1, m41, m51, m61, m71, m81, m91, m101],
[m02, m12, m22, k2, m42, m52, m62, m72, m82, m92, m102],
[m03, m13, m23, k3, m43, m53, m63, m73, m83, m93, m103],
[m04, m14, m24, k4, m44, m54, m64, m74, m84, m94, m104],
[m05, m15, m25, k5, m45, m55, m65, m75, m85, m95, m105],
[m06, m16, m26, k6, m46, m56, m66, m76, m86, m96, m106],
[m07, m17, m27, k7, m47, m57, m67, m77, m87, m97, m107],
[m08, m18, m28, k8, m48, m58, m68, m78, m88, m98, m108],
[m09, m19, m29, k9, m49, m59, m69, m79, m89, m99, m109],
[m010, m110, m210, k10, m410, m510, m610, m710, m810, m910, m1010]
])

```

```

det4 = det([
[m00, m10, m20, m30, k0, m50, m60, m70, m80, m90, m100],
[m01, m11, m21, m31, k1, m51, m61, m71, m81, m91, m101],
[m02, m12, m22, m32, k2, m52, m62, m72, m82, m92, m102],
[m03, m13, m23, m33, k3, m53, m63, m73, m83, m93, m103],
[m04, m14, m24, m34, k4, m54, m64, m74, m84, m94, m104],
[m05, m15, m25, m35, k5, m55, m65, m75, m85, m95, m105],
[m06, m16, m26, m36, k6, m56, m66, m76, m86, m96, m106],

```

```

[m07, m17, m27, m37, k7, m57, m67, m77, m87, m97, m107],
[m08, m18, m28, m38, k8, m58, m68, m78, m88, m98, m108],
[m09, m19, m29, m39, k9, m59, m69, m79, m89, m99, m109],
[m010, m110, m210, m310, k10, m510, m610, m710, m810, m910, m1010]
])

```

```

det5 = det([
[m00, m10, m20, m30, m40, k0, m60, m70, m80, m90, m100],
[m01, m11, m21, m31, m41, k1, m61, m71, m81, m91, m101],
[m02, m12, m22, m32, m42, k2, m62, m72, m82, m92, m102],
[m03, m13, m23, m33, m43, k3, m63, m73, m83, m93, m103],
[m04, m14, m24, m34, m44, k4, m64, m74, m84, m94, m104],
[m05, m15, m25, m35, m45, k5, m65, m75, m85, m95, m105],
[m06, m16, m26, m36, m46, k6, m66, m76, m86, m96, m106],
[m07, m17, m27, m37, m47, k7, m67, m77, m87, m97, m107],
[m08, m18, m28, m38, m48, k8, m68, m78, m88, m98, m108],
[m09, m19, m29, m39, m49, k9, m69, m79, m89, m99, m109],
[m010, m110, m210, m310, m410, k10, m610, m710, m810, m910, m1010]
])

```

```

det6 = det([
[m00, m10, m20, m30, m40, m50, k0, m70, m80, m90, m100],
[m01, m11, m21, m31, m41, m51, k1, m71, m81, m91, m101],
[m02, m12, m22, m32, m42, m52, k2, m72, m82, m92, m102],
[m03, m13, m23, m33, m43, m53, k3, m73, m83, m93, m103],
[m04, m14, m24, m34, m44, m54, k4, m74, m84, m94, m104],
[m05, m15, m25, m35, m45, m55, k5, m75, m85, m95, m105],
[m06, m16, m26, m36, m46, m56, k6, m76, m86, m96, m106],
[m07, m17, m27, m37, m47, m57, k7, m77, m87, m97, m107],
[m08, m18, m28, m38, m48, m58, k8, m78, m88, m98, m108],
[m09, m19, m29, m39, m49, m59, k9, m79, m89, m99, m109],
[m010, m110, m210, m310, m410, m510, k10, m710, m810, m910, m1010]
])

```

```

det7 = det([
[m00, m10, m20, m30, m40, m50, m60, k0, m80, m90, m100],
[m01, m11, m21, m31, m41, m51, m61, k1, m81, m91, m101],
[m02, m12, m22, m32, m42, m52, m62, k2, m82, m92, m102],
[m03, m13, m23, m33, m43, m53, m63, k3, m83, m93, m103],
[m04, m14, m24, m34, m44, m54, m64, k4, m84, m94, m104],
[m05, m15, m25, m35, m45, m55, m65, k5, m85, m95, m105],
[m06, m16, m26, m36, m46, m56, m66, k6, m86, m96, m106],
[m07, m17, m27, m37, m47, m57, m67, k7, m87, m97, m107],
[m08, m18, m28, m38, m48, m58, m68, k8, m88, m98, m108],
[m09, m19, m29, m39, m49, m59, m69, k9, m89, m99, m109],
[m010, m110, m210, m310, m410, m510, m610, k10, m810, m910, m1010]
])

```

```

det8 = det([
[m00, m10, m20, m30, m40, m50, m60, m70, k0, m90, m100],
[m01, m11, m21, m31, m41, m51, m61, m71, k1, m91, m101],
[m02, m12, m22, m32, m42, m52, m62, m72, k2, m92, m102],
[m03, m13, m23, m33, m43, m53, m63, m73, k3, m93, m103],
[m04, m14, m24, m34, m44, m54, m64, m74, k4, m94, m104],
[m05, m15, m25, m35, m45, m55, m65, m75, k5, m95, m105],
[m06, m16, m26, m36, m46, m56, m66, m76, k6, m96, m106],
[m07, m17, m27, m37, m47, m57, m67, m77, k7, m97, m107],
[m08, m18, m28, m38, m48, m58, m68, m78, k8, m98, m108],
[m09, m19, m29, m39, m49, m59, m69, m79, k9, m99, m109],
[m010, m110, m210, m310, m410, m510, m610, m710, k10, m910, m1010]
])

```

```

det9 = det([
    [m00, m10, m20, m30, m40, m50, m60, m70, m80, k0, m100],
    [m01, m11, m21, m31, m41, m51, m61, m71, m81, k1, m101],
    [m02, m12, m22, m32, m42, m52, m62, m72, m82, k2, m102],
    [m03, m13, m23, m33, m43, m53, m63, m73, m83, k3, m103],
    [m04, m14, m24, m34, m44, m54, m64, m74, m84, k4, m104],
    [m05, m15, m25, m35, m45, m55, m65, m75, m85, k5, m105],
    [m06, m16, m26, m36, m46, m56, m66, m76, m86, k6, m106],
    [m07, m17, m27, m37, m47, m57, m67, m77, m87, k7, m107],
    [m08, m18, m28, m38, m48, m58, m68, m78, m88, k8, m108],
    [m09, m19, m29, m39, m49, m59, m69, m79, m89, k9, m109],
    [m010, m110, m210, m310, m410, m510, m610, m710, m810, k10, m1010]
])

```

```

det10 = det([
    [m00, m10, m20, m30, m40, m50, m60, m70, m80, m90, k0],
    [m01, m11, m21, m31, m41, m51, m61, m71, m81, m91, k1],
    [m02, m12, m22, m32, m42, m52, m62, m72, m82, m92, k2],
    [m03, m13, m23, m33, m43, m53, m63, m73, m83, m93, k3],
    [m04, m14, m24, m34, m44, m54, m64, m74, m84, m94, k4],
    [m05, m15, m25, m35, m45, m55, m65, m75, m85, m95, k5],
    [m06, m16, m26, m36, m46, m56, m66, m76, m86, m96, k6],
    [m07, m17, m27, m37, m47, m57, m67, m77, m87, m97, k7],
    [m08, m18, m28, m38, m48, m58, m68, m78, m88, m98, k8],
    [m09, m19, m29, m39, m49, m59, m69, m79, m89, m99, k9],
    [m010, m110, m210, m310, m410, m510, m610, m710, m810, m910, k10]
])

```

```

cfs0 = det0 / main_det
cfs1 = det1 / main_det
cfs2 = det2 / main_det
cfs3 = det3 / main_det
cfs4 = det4 / main_det
cfs5 = det5 / main_det
cfs6 = det6 / main_det
cfs7 = det7 / main_det
cfs8 = det8 / main_det
cfs9 = det9 / main_det
cfs10 = det10 / main_det

```

```

return [cfs0, cfs1, cfs2, cfs3, cfs4, cfs5, cfs6, cfs7, cfs8, cfs9, cfs10]

```

```

def calculate_with_nature_cfs(self, nxl):

```

```

    """nxl - nature x line"""
    return self.A[0]*nxl[0] + self.A[1]*nxl[1] + self.A[2]*nxl[2] + self.A[3]*nxl[3] + \
        self.A[4]*nxl[1]*nxl[2] + self.A[5]*nxl[1]*nxl[3] + self.A[6]*nxl[2]*nxl[3] + \
        self.A[7]*nxl[1]*nxl[2]*nxl[3] + self.A[8]*(nxl[1] ** 2) + self.A[9]*(nxl[2] ** 2) + \
        self.A[10]*(nxl[3] ** 2)

```

```

def calculate_with_encoded_cfs(self, xl):

```

```

    """xl - encoded x line"""
    return self.B[0]*xl[0] + self.B[1]*xl[1] + self.B[2]*xl[2] + self.B[3]*xl[3] + \
        self.B[4]*xl[1]*xl[2] + self.B[5]*xl[1]*xl[3] + self.B[6]*xl[2]*xl[3] + \
        self.B[7]*xl[1]*xl[2]*xl[3] + self.B[8]*(xl[1] ** 2) + self.B[9]*(xl[2] ** 2) + \
        self.B[10]*(xl[3] ** 2)

```

criterion_tables.py

```

"""Таблиця для критерія Кохрена"""

```

```

base_kohren = [
    [9985, 9750, 9392, 9057, 8772, 8534, 8332, 8159, 8010, 7880, 7341, 6602, 5813, 5000],

```

```

[9669, 8709, 7977, 7457, 7071, 6771, 6530, 6333, 6167, 6025, 5466, 4748, 4031, 3333],
[9065, 7679, 6841, 6287, 5892, 5598, 5365, 5175, 5017, 4884, 4366, 3720, 3093, 2500],
[8412, 6838, 5981, 5440, 5063, 4783, 4564, 4387, 4241, 4118, 3645, 3066, 2513, 2000],
[7808, 6161, 5321, 4803, 4447, 4184, 3980, 3817, 3682, 3568, 3135, 2612, 2119, 1667],
[7271, 5612, 4800, 4307, 3974, 3726, 3535, 3384, 3259, 3154, 2756, 2278, 1833, 1429],
[6798, 5157, 4377, 3910, 3595, 3362, 3185, 3043, 2926, 2829, 2462, 2022, 1616, 1250],
[6385, 4775, 4027, 3584, 3286, 3067, 2901, 2768, 2659, 2568, 2226, 1820, 1446, 1111],
[6020, 4450, 3733, 3311, 3029, 2823, 2666, 2541, 2439, 2353, 2032, 1655, 1308, 1000],
[5410, 3924, 3264, 2880, 2624, 2439, 2299, 2187, 2098, 2020, 1737, 1403, 1000, 833],
[4709, 3346, 2758, 2419, 2159, 2034, 1911, 1815, 1736, 1671, 1429, 1144, 889, 667],
[3894, 2705, 2205, 1921, 1735, 1602, 1501, 1422, 1357, 1303, 1108, 879, 675, 500],
[3434, 2354, 1907, 1656, 1493, 1374, 1286, 1216, 1160, 1113, 942, 743, 567, 417],
[2929, 1980, 1593, 1377, 1237, 1137, 1061, 1002, 958, 921, 771, 604, 457, 333],
[2370, 1576, 1259, 1082, 968, 887, 827, 780, 745, 713, 595, 462, 347, 250],
[1737, 1131, 895, 766, 682, 623, 583, 552, 520, 497, 411, 316, 234, 167],
[998, 632, 495, 419, 371, 337, 312, 292, 279, 266, 218, 165, 120, 83],
]
column_kohren_f1 = {(1,): 0, (2,): 1, (3,): 2, (4,): 3, (5,): 4, (6,): 5, (7,): 6, (8,): 7, (9,): 8,
                    (range(10, 14)): 9, (range(14, 26)): 10, (range(26, 91)): 11, (range(91, 145)): 12}
COLUMN_KOHREN_F1_ELSE = 13

row_kohren_f2 = {(2,): 0, (3,): 1, (4,): 2, (5,): 3, (6,): 4, (7,): 5, (8,): 6, (9,): 7,
                (range(10, 12)): 8, (range(12, 14)): 9, (range(14, 18)): 10, (range(18, 23)): 11,
                (range(23, 28)): 12, (range(28, 36)): 13, (range(36, 51)): 14, (range(51, 81)): 15,
                (range(81, 121)): 16}

ROW_KOHREN_F2_ELSE = 16

"""Таблиця для t-критерія Стьюдента"""
base_student_f3 = {(1,): 12.706, (2,): 4.303, (3,): 3.182, (4,): 2.776, (5,): 2.571, (6,): 2.447, (7,): 2.365,
                  (8,): 2.306, (9,): 2.262, (10,): 2.228, (11,): 2.201, (12,): 2.179, (13,): 2.160, (14,): 2.145,
                  (15,): 2.131, (16,): 2.120, (17,): 2.110, (18,): 2.101, (19,): 2.093, (20,): 2.086,
                  (range(21, 25)): 2.069, (range(25, 30)): 2.060, (range(30, 40)): 2.042, (range(40, 60)): 2.021,
                  (range(60, 81)): 2.000, (range(81, 111)): 1.980, (range(111, 121)): 1.960}
T_STUDENT_ELSE = 1.960

"""Таблиця для F-критерія Фішера"""
base_phisher = [
[164.4, 199.5, 215.7, 224.6, 230.2, 234.0, 244.9, 249.0, 254.3],
[18.5, 19.2, 19.2, 19.3, 19.3, 19.3, 19.4, 19.4, 19.5],
[10.1, 9.6, 9.3, 9.1, 9.0, 8.9, 8.7, 8.6, 8.5],
[7.7, 6.9, 6.6, 6.4, 6.3, 6.2, 5.9, 5.8, 5.6],
[6.6, 5.8, 5.4, 5.2, 5.1, 5.0, 4.7, 4.5, 4.4],
[6.0, 5.1, 4.8, 4.5, 4.4, 4.3, 4.0, 3.8, 3.7],
[5.5, 4.7, 4.4, 4.1, 4.0, 3.9, 3.6, 3.4, 3.2],
[5.3, 4.5, 4.1, 3.8, 3.7, 3.6, 3.3, 3.1, 2.9],
[5.1, 4.3, 3.9, 3.6, 3.5, 3.4, 3.1, 2.9, 2.7],
[5.0, 4.1, 3.7, 3.5, 3.3, 3.2, 2.9, 2.7, 2.5],
[4.8, 4.0, 3.6, 3.4, 3.2, 3.1, 2.8, 2.6, 2.4],
[4.8, 3.9, 3.5, 3.3, 3.1, 3.0, 2.7, 2.5, 2.3],
[4.7, 3.8, 3.4, 3.2, 3.0, 2.9, 2.6, 2.4, 2.2],
[4.6, 3.7, 3.3, 3.1, 3.0, 2.9, 2.5, 2.3, 2.1],
[4.5, 3.7, 3.3, 3.1, 2.9, 2.8, 2.5, 2.3, 2.1],
[4.5, 3.6, 3.2, 3.0, 2.9, 2.7, 2.4, 2.2, 2.0],
[4.5, 3.6, 3.2, 3.0, 2.8, 2.7, 2.4, 2.2, 2.0],
[4.4, 3.6, 3.2, 2.9, 2.8, 2.7, 2.3, 2.1, 1.9],
[4.4, 3.5, 3.1, 2.9, 2.7, 2.6, 2.3, 2.1, 1.9],
[4.4, 3.5, 3.1, 2.9, 2.7, 2.6, 2.3, 2.1, 1.9],
[4.3, 3.4, 3.1, 2.8, 2.7, 2.6, 2.2, 2.0, 1.8],
[4.3, 3.4, 3.0, 2.8, 2.6, 2.5, 2.2, 2.0, 1.7],
[4.2, 3.4, 3.0, 2.7, 2.6, 2.5, 2.2, 2.0, 1.7],
[4.2, 3.3, 3.0, 2.7, 2.6, 2.4, 2.1, 1.9, 1.7],

```

```

[4.2, 3.3, 2.9, 2.7, 2.5, 2.4, 2.1, 1.9, 1.6],
[4.1, 3.2, 2.9, 2.6, 2.5, 2.3, 2.0, 1.8, 1.5],
[4.0, 3.2, 2.8, 2.5, 2.4, 2.3, 1.9, 1.7, 1.4],
[3.9, 3.1, 2.7, 2.5, 2.3, 2.2, 1.8, 1.6, 1.3],
[3.8, 3.0, 2.6, 2.4, 2.2, 2.1, 1.8, 1.5, 1.0]
]

```

```

row_phisher_f3 = {(1,): 0, (2,): 1, (3,): 2, (4,): 3, (5,): 4, (6,): 5, (7,): 6, (8,): 7, (9,): 8, (10,): 9,
                  (11,): 10, (12,): 11, (13,): 12, (14,): 13, (15,): 14, (16,): 15, (17,): 16, (18,): 17, (19,): 18,
                  (20, 21): 19, (22, 23): 20, (24, 25): 21, (26, 27): 22, (28, 29): 23, (range(30, 36)): 24,
                  (range(36, 51)): 25, (range(51, 91)): 26, (range(91, 121)): 27}
ROW_PHISHER_F3_ELSE = 28

```

```

column_phisher_f4 = {(1,): 0, (2,): 1, (3,): 2, (4,): 3, (5,): 4, (range(6, 10)): 5, (range(10, 19)): 6,
                    (range(19, 25)): 7}
COLUMN_PHISHER_F4_ELSE = 8

```

```

def compare_kohren_with_table_value(f1, f2, Gp):

```

```

    """True, якщо дисперсія однорідна"""

```

```

    row = -1
    column = -1
    for key in row_kohren_f2.keys():
        if f2 in key:
            row = row_kohren_f2[key]
            break
    if row == -1:
        row = ROW_KOHREN_F2_ELSE

    for key in column_kohren_f1.keys():
        if f1 in key:
            column = column_kohren_f1[key]
            break
    if column == -1:
        column = COLUMN_KOHREN_F1_ELSE
    return Gp < (base_kohren[row][column]/1000)

```

```

def compare_student_with_table_value(f3, t_exp):

```

```

    """True, якщо коефіцієнт Bs є значущим."""

```

```

    t_teo = -1
    for key in base_student_f3.keys():
        if f3 in key:
            t_teo = base_student_f3[key]
            break

    if t_teo == -1:
        t_teo = T_STUDENT_ELSE

    return t_exp > t_teo

```

```

def compare_phisher_with_table_value(f3, f4, Fp):

```

```

    """True, якщо отримана математична модель адекватна експериментальним даним."""

```

```

    row = -1
    column = -1
    for key in row_phisher_f3.keys():
        if f3 in key:
            row = row_phisher_f3[key]
            break
    if row == -1:
        row = ROW_PHISHER_F3_ELSE

```

```

for key in column_phisher_f4.keys():
    if f4 in key:
        column = column_phisher_f4[key]
        break
if column == -1:
    column = COLUMN_PHISHER_F4_ELSE
return Fp <= base_phisher[row][column]

```

logs.py

```

from beautifultable import BeautifulTable

```

```

SYSTEM_MARK_L = "Action: "
SYSTEM_MARK_V = "View: "

```

```

text = {
    0: "Розглянемо лінійне рівняння регресії без взаємодії факторів:\n $y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + b_3 \cdot x_3$ ",
    1: "Розглянемо лінійне рівняння регресії із врахуванням взаємодії факторів:\n"
        " $y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + b_3 \cdot x_3 + b_{12} \cdot x_1 \cdot x_2 + b_{13} \cdot x_1 \cdot x_3 + b_{23} \cdot x_2 \cdot x_3 + b_{123} \cdot x_1 \cdot x_2 \cdot x_3$ ",
    2: "Розглянемо рівняння регресії із врахуванням квадратичних членів:\n"
        " $y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + b_3 \cdot x_3 + b_{12} \cdot x_1 \cdot x_2 + b_{13} \cdot x_1 \cdot x_3 + b_{23} \cdot x_2 \cdot x_3 + b_{123} \cdot x_1 \cdot x_2 \cdot x_3 +$  "
        " $+ b_{11} \cdot x_1 \cdot x_1 + b_{22} \cdot x_2 \cdot x_2 + b_{33} \cdot x_3 \cdot x_3$ ",
    3: "Маємо,  $N = \{0\}$ ,  $K = \{1\}$ ,  $m = \{2\}$ .",
    4: "Складаємо матрицю планування і проведемо експерименти",
    5: "Розраховуємо натуральні значення коефіцієнтів.",
    6: "Розраховуємо кодовані значення коефіцієнтів.",
    7: "Рівняння регресії має вигляд (нат. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3$ ",
    8: "Рівняння регресії має вигляд (код. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3$ ",
    9: "Рівняння регресії має вигляд (нат. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3 + \{4\} \cdot x_1 \cdot x_2 + \{5\} \cdot x_1 \cdot x_3 + \{6\} \cdot x_2 \cdot x_3 + \{7\} \cdot x_1 \cdot x_2 \cdot x_3$ ",
    10: "Рівняння регресії має вигляд (код. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3 + \{4\} \cdot x_1 \cdot x_2 + \{5\} \cdot x_1 \cdot x_3 + \{6\} \cdot x_2 \cdot x_3 + \{7\} \cdot x_1 \cdot x_2 \cdot x_3$ ",
    11: "Рівняння регресії має вигляд (нат. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3 + \{4\} \cdot x_1 \cdot x_2 + \{5\} \cdot x_1 \cdot x_3 + \{6\} \cdot x_2 \cdot x_3 + \{7\} \cdot x_1 \cdot x_2 \cdot x_3 +$  "
        " $+ \{8\} \cdot x_1 \cdot x_1 + \{9\} \cdot x_2 \cdot x_2 + \{10\} \cdot x_3 \cdot x_3$ ",
    12: "Рівняння регресії має вигляд (код. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3 + \{4\} \cdot x_1 \cdot x_2 + \{5\} \cdot x_1 \cdot x_3 + \{6\} \cdot x_2 \cdot x_3 + \{7\} \cdot x_1 \cdot x_2 \cdot x_3 +$  "
        " $+ \{8\} \cdot x_1 \cdot x_1 + \{9\} \cdot x_2 \cdot x_2 + \{10\} \cdot x_3 \cdot x_3$ ",
    13: "Дисперсія однорідна за критерієм Кохрена при  $m = \{0\}$ : \n"
        " $f_1 = \{1\}$ ,  $f_2 = \{2\}$ ,  $G_p = \{3\}$ ",
    14: "Перевіряємо нуль гіпотезу та корегуємо рівняння регресії: \n"
        " $f_3 = \{0\}$ ,  $t = \{1\}$ .",
    15: "Нове рівняння регресії має вигляд (нат. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3$ ",
    16: "Нове рівняння регресії має вигляд (код. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3$ ",
    17: "Нове рівняння регресії має вигляд (нат. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3 + \{4\} \cdot x_1 \cdot x_2 + \{5\} \cdot x_1 \cdot x_3 + \{6\} \cdot x_2 \cdot x_3 + \{7\} \cdot x_1 \cdot x_2 \cdot x_3$ ",
    18: "Нове рівняння регресії має вигляд (код. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3 + \{4\} \cdot x_1 \cdot x_2 + \{5\} \cdot x_1 \cdot x_3 + \{6\} \cdot x_2 \cdot x_3 + \{7\} \cdot x_1 \cdot x_2 \cdot x_3$ ",
    19: "Нове рівняння регресії має вигляд (нат. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3 + \{4\} \cdot x_1 \cdot x_2 + \{5\} \cdot x_1 \cdot x_3 + \{6\} \cdot x_2 \cdot x_3 + \{7\} \cdot x_1 \cdot x_2 \cdot x_3 +$  "
        " $+ \{8\} \cdot x_1 \cdot x_1 + \{9\} \cdot x_2 \cdot x_2 + \{10\} \cdot x_3 \cdot x_3$ ",
    20: "Нове рівняння регресії має вигляд (код. знач. коеф.): \n"
        " $y = \{0\} + \{1\} \cdot x_1 + \{2\} \cdot x_2 + \{3\} \cdot x_3 + \{4\} \cdot x_1 \cdot x_2 + \{5\} \cdot x_1 \cdot x_3 + \{6\} \cdot x_2 \cdot x_3 + \{7\} \cdot x_1 \cdot x_2 \cdot x_3 +$  "
        " $+ \{8\} \cdot x_1 \cdot x_1 + \{9\} \cdot x_2 \cdot x_2 + \{10\} \cdot x_3 \cdot x_3$ ",
    21: "Перевіряємо адекватність моделі. \n"
        " $f_3 = \{0\}$ ,  $f_4 = \{1\}$ ,  $G_p = \{2\}$ ",
    22: "Модель адекватна оригіналу.",
    23: "Модель не адекватна оригіналу.",

```

```

24: "Змінюємо рівняння регресії.\n",
25: "Виводимо результати.",
26: "Оскільки всі моделі не адекватні, то почнемо експерименти з початку.",
27: "Перевірка критерія Кохрена займає: {0}",
28: "Перевірка критерія Стьюдента займає: {0}",
29: "Перевірка критерія Фішера займає: {0}"

```

```

def comment(key, par):
    return print(SYSTEM_MARK_L + text[key].format(*par))

```

```

titles = {0: "Матриця планування експерименту (нат. знач. коеф., без взаємодії)",
1: "Матриця планування експерименту (код. знач. коеф., без взаємодії)",
2: "Матриця планування експерименту (нат. знач. коеф., із взаємодією)",
3: "Матриця планування експерименту (код. знач. коеф., із взаємодією)",
4: "Центральний ортогональний композиційний план (нат. знач. коеф.)",
5: "Центральний ортогональний композиційний план (код. знач. коеф.)",
6: "Перевірка знайдених коефіцієнтів (нат. знач. коеф., без взаємодії)",
7: "Перевірка знайдених коефіцієнтів (код. знач. коеф., без взаємодії)",
8: "Перевірка знайдених коефіцієнтів (нат. знач. коеф., із взаємодією)",
9: "Перевірка знайдених коефіцієнтів (код. знач. коеф., із взаємодією)",
10: "Перевірка знайдених коефіцієнтів (нат. знач. коеф., із квад. членами)",
11: "Перевірка знайдених коефіцієнтів (код. знач. коеф., із квад. членами)"}

```

```

x_headers = {
0: ["N°", "X0", "X1", "X2", "X3"],
1: ["N°", "X0", "X1", "X2", "X3", "X1·X2", "X1·X3", "X2·X3", "X1·X2·X3"],
2: ["N°", "X0", "X1", "X2", "X3", "X1·X2", "X1·X3", "X2·X3", "X1·X2·X3", "X1·X1", "X2·X2", "X3·X3"]
}

```

```

def show_plan(title_index, x_header_index, x_lines, experiment):
    print(SYSTEM_MARK_V + titles[title_index])
    plan = BeautifulTable()
    plan.max_table_width = 1000
    y_headers = [f"Y{i + 1}" for i in range(experiment.m)]
    plan.column_headers = [*x_headers[x_header_index], *y_headers]
    for i in range(experiment.N):
        plan.append_row([i + 1, *x_lines[i], *experiment.y[i]])
    print(plan, "\n")

```

```

def show_natured_checking_matrix(title_index, x_header_index, x_lines, model, experiment):
    print(SYSTEM_MARK_V + titles[title_index])
    natured_checking_matrix = BeautifulTable()
    natured_checking_matrix.max_table_width = 1000
    natured_checking_matrix.column_headers = [*x_headers[x_header_index], "Average Y[j]", "Exp-tal Y[j]"]
    y_average = experiment.y_average
    for i in range(experiment.N):
        y_exp = model.calculate_with_nature_cfs(x_lines[i])
        natured_checking_matrix.append_row([i + 1, *x_lines[i], y_average[i], y_exp])
    print(natured_checking_matrix, "\n")

```

```

def show_encoded_checking_matrix(title_index, x_header_index, x_lines, model, experiment):
    print(SYSTEM_MARK_V + titles[title_index])
    encoded_checking_matrix = BeautifulTable()
    encoded_checking_matrix.max_table_width = 1000
    encoded_checking_matrix.column_headers = [*x_headers[x_header_index], "Average Y[j]", "Exp-tal Y[j]"]
    y_average = experiment.y_average
    for i in range(experiment.N):

```

```

y_exp = model.calculate_with_encoded_cfs(x_lines[i])
encoded_checking_matrix.append_row([i+1, *x_lines[i], y_average[i], y_exp])
print(encoded_checking_matrix, "\n")

```

Результат виконання роботи

Примітка: оскільки для заданого варіанту більшість експериментів закінчується підтвердження адекватності лінійного рівняння регресії без взаємодії факторів, то робимо висновок, що ймовірність перевірити всю послідовність операції надзвичайно мала. Тому робимо припущення, що лінійне рівняння регресії без взаємодії факторів та лінійне рівняння регресії із взаємодією факторів виявились неадекватними для того, що продемонструвати, як проводяться обчислення при рівнянні регресії із квадратичними коефіцієнтами.

Action: Розглянемо рівняння регресії із врахуванням квадратичних членів:

$y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + b_3 \cdot x_3 + b_{12} \cdot x_1 \cdot x_2 + b_{13} \cdot x_1 \cdot x_3 + b_{23} \cdot x_2 \cdot x_3 + b_{123} \cdot x_1 \cdot x_2 \cdot x_3 + b_{11} \cdot x_1 \cdot x_1 + b_{22} \cdot x_2 \cdot x_2 + b_{33} \cdot x_3 \cdot x_3$

Action: Маємо, $N = 15$, $K = 11$, $m = 3$.

Action: Складаємо матрицю планування і проведемо експерименти

Action: Дисперсія однорідна за критерієм Кохрена при $m = 3$:

$f_1 = 2$, $f_2 = 15$, $G_p = 0.17819148936170212$

View: Центральний ортогональний композиційний план (нат. знач. коеф.)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Y1 | Y2 | Y3 |
|----|----|---------|-------|--------|---------|---------|---------|----------|---------|--------|---------|-----|-----|-----|
| 1 | 1 | -10 | -6 | 10 | 60 | -100 | -60 | 600 | 100 | 36 | 100 | 198 | 198 | 199 |
| 2 | 1 | -10 | 6 | -1 | -60 | 10 | -6 | 60 | 100 | 36 | 1 | 199 | 200 | 203 |
| 3 | 1 | 1 | -6 | -1 | -6 | -1 | 6 | 6 | 1 | 36 | 1 | 198 | 196 | 205 |
| 4 | 1 | 1 | 6 | 10 | 6 | 10 | 60 | 60 | 1 | 36 | 100 | 197 | 196 | 198 |
| 5 | 1 | -10 | -6 | -1 | 60 | 10 | 6 | -60 | 100 | 36 | 1 | 197 | 198 | 203 |
| 6 | 1 | -10 | 6 | 10 | -60 | -100 | 60 | -600 | 100 | 36 | 100 | 199 | 195 | 196 |
| 7 | 1 | 1 | -6 | 10 | -6 | 10 | -60 | -60 | 1 | 36 | 100 | 201 | 196 | 203 |
| 8 | 1 | 1 | 6 | -1 | 6 | -1 | -6 | -6 | 1 | 36 | 1 | 198 | 195 | 195 |
| 9 | 1 | -11.183 | 0.0 | 4.5 | -0.0 | -50.321 | 0.0 | -0.0 | 125.048 | 0.0 | 20.25 | 202 | 196 | 203 |
| 10 | 1 | 2.183 | 0.0 | 4.5 | 0.0 | 9.821 | 0.0 | 0.0 | 4.763 | 0.0 | 20.25 | 195 | 196 | 197 |
| 11 | 1 | -4.5 | -7.29 | 4.5 | 32.805 | -20.25 | -32.805 | 147.623 | 20.25 | 53.144 | 20.25 | 195 | 204 | 197 |
| 12 | 1 | -4.5 | 7.29 | 4.5 | -32.805 | -20.25 | 32.805 | -147.623 | 20.25 | 53.144 | 20.25 | 201 | 205 | 202 |
| 13 | 1 | -4.5 | 0.0 | -2.183 | -0.0 | 9.821 | -0.0 | 0.0 | 20.25 | 0.0 | 4.763 | 200 | 195 | 195 |
| 14 | 1 | -4.5 | 0.0 | 11.183 | -0.0 | -50.321 | 0.0 | -0.0 | 20.25 | 0.0 | 125.048 | 199 | 200 | 204 |
| 15 | 1 | -4.5 | 0.0 | 4.5 | -0.0 | -20.25 | 0.0 | -0.0 | 20.25 | 0.0 | 20.25 | 198 | 202 | 204 |

View: Центральний ортогональний композиційний план (код. знач. коеф.)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Y1 | Y2 | Y3 |
|----|----|--------|--------|--------|-------|-------|-------|----------|-------|-------|-------|-----|-----|-----|
| 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | 1 | 1 | 1 | 198 | 198 | 199 |
| 2 | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 1 | 199 | 200 | 203 |
| 3 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 198 | 196 | 205 |
| 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 197 | 196 | 198 |
| 5 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 | 197 | 198 | 203 |
| 6 | 1 | -1 | 1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 199 | 195 | 196 |
| 7 | 1 | 1 | -1 | 1 | -1 | 1 | -1 | -1 | 1 | 1 | 1 | 201 | 196 | 203 |
| 8 | 1 | 1 | 1 | -1 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | 198 | 195 | 195 |
| 9 | 1 | -1.215 | 0 | 0 | -0.0 | -0.0 | 0 | -0.0 | 1.476 | 0 | 0 | 202 | 196 | 203 |
| 10 | 1 | 1.215 | 0 | 0 | 0.0 | 0.0 | 0 | 0.0 | 1.476 | 0 | 0 | 195 | 196 | 197 |
| 11 | 1 | 0 | -1.215 | 0 | -0.0 | 0 | -0.0 | -0.0 | 0 | 1.476 | 0 | 195 | 204 | 197 |
| 12 | 1 | 0 | 1.215 | 0 | 0.0 | 0 | 0.0 | 0.0 | 0 | 1.476 | 0 | 201 | 205 | 202 |
| 13 | 1 | 0 | 0 | -1.215 | 0 | -0.0 | -0.0 | -0.0 | 0 | 0 | 1.476 | 200 | 195 | 195 |
| 14 | 1 | 0 | 0 | 1.215 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0 | 1.476 | 199 | 200 | 204 |
| 15 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 198 | 202 | 204 |

Action: Розраховуємо натуральні значення коефіцієнтів.

Action: Рівняння регресії має вигляд (нат. знач. коеф.):

$y = 197.7057 + -0.674 \cdot x_1 + -0.1519 \cdot x_2 + 0.44 \cdot x_3 + -0.0354 \cdot x_1 \cdot x_2 + 0.0262 \cdot x_1 \cdot x_3 + 0.0025 \cdot x_2 \cdot x_3 + 0.0025 \cdot x_1 \cdot x_2 \cdot x_3 + -0.0478 \cdot x_1 \cdot x_1 + 0.0069 \cdot x_2 \cdot x_2 + -0.0329 \cdot x_3 \cdot x_3$

Action: Розраховуємо кодовані значення коефіцієнтів.

Action: Рівняння регресії має вигляд (код. знач. коеф.):

$y = 200.5558 + -0.6938 \cdot x_1 + -0.1954 \cdot x_2 + 0.1459 \cdot x_3 + -0.7917 \cdot x_1 \cdot x_2 + 0.7917 \cdot x_1 \cdot x_3 + -0.2917 \cdot x_2 \cdot x_3 + 0.4583 \cdot x_1 \cdot x_2 \cdot x_3 + -1.4456 \cdot x_1 \cdot x_1 + 0.2479 \cdot x_2 \cdot x_2 + -0.994 \cdot x_3 \cdot x_3$

View: Перевірка знайдених коефіцієнтів (нат. знач. коеф., із квад. членами)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Average Y[j] | Exp-tal Y[j] |
|----|----|---------|-------|--------|---------|---------|---------|----------|---------|--------|---------|--------------|--------------|
| 1 | 1 | -10 | -6 | 10 | 60 | -100 | -60 | 600 | 100 | 36 | 100 | 198.333 | 198.566 |
| 2 | 1 | -10 | 6 | -1 | -60 | 10 | -6 | 60 | 100 | 36 | 1 | 200.667 | 201.05 |
| 3 | 1 | 1 | -6 | -1 | -6 | -1 | 6 | 6 | 1 | 36 | 1 | 199.667 | 197.887 |
| 4 | 1 | 1 | 6 | 10 | 6 | 10 | 60 | 60 | 1 | 36 | 100 | 197.0 | 197.788 |
| 5 | 1 | -10 | -6 | -1 | 60 | 10 | 6 | -60 | 100 | 36 | 1 | 199.333 | 198.357 |
| 6 | 1 | -10 | 6 | 10 | -60 | -100 | 60 | -600 | 100 | 36 | 100 | 196.667 | 198.258 |
| 7 | 1 | 1 | -6 | 10 | -6 | 10 | -60 | -60 | 1 | 36 | 100 | 200.0 | 199.428 |
| 8 | 1 | 1 | 6 | -1 | 6 | -1 | -6 | -6 | 1 | 36 | 1 | 196.0 | 195.579 |
| 9 | 1 | -11.183 | 0.0 | 4.5 | -0.0 | -50.321 | 0.0 | -0.0 | 125.048 | 0.0 | 20.25 | 200.333 | 199.265 |
| 10 | 1 | 2.183 | 0.0 | 4.5 | 0.0 | 9.821 | 0.0 | 0.0 | 4.763 | 0.0 | 20.25 | 196.0 | 197.579 |
| 11 | 1 | -4.5 | -7.29 | 4.5 | 32.805 | -20.25 | -32.805 | 147.623 | 20.25 | 53.144 | 20.25 | 198.667 | 201.159 |
| 12 | 1 | -4.5 | 7.29 | 4.5 | -32.805 | -20.25 | 32.805 | -147.623 | 20.25 | 53.144 | 20.25 | 202.667 | 200.684 |
| 13 | 1 | -4.5 | 0.0 | -2.183 | -0.0 | 9.821 | -0.0 | 0.0 | 20.25 | 0.0 | 4.763 | 196.667 | 198.911 |
| 14 | 1 | -4.5 | 0.0 | 11.183 | -0.0 | -50.321 | 0.0 | -0.0 | 20.25 | 0.0 | 125.048 | 201.0 | 199.266 |
| 15 | 1 | -4.5 | 0.0 | 4.5 | -0.0 | -20.25 | 0.0 | -0.0 | 20.25 | 0.0 | 20.25 | 201.333 | 200.556 |

View: Перевірка знайдених коефіцієнтів (код. знач. коеф., із квад. членами)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Average Y[j] | Exp-tal Y[j] |
|----|----|--------|--------|--------|-------|-------|-------|----------|-------|-------|-------|--------------|--------------|
| 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | 1 | 1 | 1 | 198.333 | 198.566 |
| 2 | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 1 | 200.667 | 201.05 |
| 3 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 199.667 | 197.887 |
| 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 197.0 | 197.788 |
| 5 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 | 199.333 | 198.357 |
| 6 | 1 | -1 | 1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 196.667 | 198.258 |
| 7 | 1 | 1 | -1 | 1 | -1 | 1 | -1 | -1 | 1 | 1 | 1 | 200.0 | 199.428 |
| 8 | 1 | 1 | 1 | -1 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | 196.0 | 195.579 |
| 9 | 1 | -1.215 | 0 | 0 | -0.0 | -0.0 | 0 | -0.0 | 1.476 | 0 | 0 | 200.333 | 199.265 |
| 10 | 1 | 1.215 | 0 | 0 | 0.0 | 0.0 | 0 | 0.0 | 1.476 | 0 | 0 | 196.0 | 197.579 |
| 11 | 1 | 0 | -1.215 | 0 | -0.0 | 0 | -0.0 | -0.0 | 0 | 1.476 | 0 | 198.667 | 201.159 |
| 12 | 1 | 0 | 1.215 | 0 | 0.0 | 0 | 0.0 | 0.0 | 0 | 1.476 | 0 | 202.667 | 200.684 |
| 13 | 1 | 0 | 0 | -1.215 | 0 | -0.0 | -0.0 | -0.0 | 0 | 0 | 1.476 | 196.667 | 198.911 |
| 14 | 1 | 0 | 0 | 1.215 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0 | 1.476 | 201.0 | 199.266 |
| 15 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 201.333 | 200.556 |

Action: Перевіряємо нуль гіпотезу та корегуємо рівняння регресії:

$f_3 = 30$, $t = [570.0323, 1.9718, 0.5553, 0.4148, 2.2501, 2.2501, 0.829, 1.3027, 4.1087, 0.7047, 2.8251]$.

Action: Нове рівняння регресії має вигляд (нат. знач. коеф.):

$y = 197.7057 + 0 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + -0.0354 \cdot x_1 \cdot x_2 + 0.0262 \cdot x_1 \cdot x_3 + 0 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_2 \cdot x_3 + -0.0478 \cdot x_1 \cdot x_1 + 0 \cdot x_2 \cdot x_2 + -0.0329 \cdot x_3 \cdot x_3$

Action: Нове рівняння регресії має вигляд (код. знач. коеф.):

$y = 200.5558 + 0 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + -0.7917 \cdot x_1 \cdot x_2 + 0.7917 \cdot x_1 \cdot x_3 + 0 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_2 \cdot x_3 + -1.4456 \cdot x_1 \cdot x_1 + 0 \cdot x_2 \cdot x_2 + -0.994 \cdot x_3 \cdot x_3$

Action: Перевіряємо адекватність моделі.

$f_3 = 30$, $f_4 = 10$, $F_p = 35.97540530635607$

Action: Модель не адекватна оригіналу.

Action: Змінюємо рівняння регресії.

Action: Оскільки всі моделі не адекватні, то почнемо експерименти з початку.

None

Action: Розглянемо рівняння регресії із врахуванням квадратичних членів:

$y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + b_3 \cdot x_3 + b_{12} \cdot x_1 \cdot x_2 + b_{13} \cdot x_1 \cdot x_3 + b_{23} \cdot x_2 \cdot x_3 + b_{11} \cdot x_1 \cdot x_1 + b_{22} \cdot x_2 \cdot x_2 + b_{33} \cdot x_3 \cdot x_3$

Action: Маємо, $N = 15$, $K = 11$, $m = 3$.

Action: Складаємо матрицю планування і проведемо експерименти

Action: Дисперсія однорідна за критерієм Кохрена при $m = 3$:

$f_1 = 2$, $f_2 = 15$, $G_p = 0.14726027397260275$

View: Центральний ортогональний композиційний план (нат. знач. коеф.)

View: Центральний ортогональний композиційний план (нат. знач. коеф.)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Y1 | Y2 | Y3 |
|---|----|-----|----|----|-------|-------|-------|----------|-------|-------|-------|-----|-----|-----|
| 1 | 1 | -10 | -6 | 10 | 60 | -100 | -60 | 600 | 100 | 36 | 100 | 202 | 205 | 202 |
| 2 | 1 | -10 | 6 | -1 | -60 | 10 | -6 | 60 | 100 | 36 | 1 | 202 | 199 | 195 |

| |
|--|
| 3 1 1 -6 -1 -6 -1 6 6 1 36 1 199 199 205 |
| 4 1 1 6 10 6 10 60 60 1 36 100 199 198 205 |
| 5 1 -10 -6 -1 60 10 6 -60 100 36 1 198 203 199 |
| 6 1 -10 6 10 -60 -100 60 -600 100 36 100 202 200 195 |
| 7 1 1 -6 10 -6 10 -60 -60 1 36 100 199 198 199 |
| 8 1 1 6 -1 6 -1 -6 -6 1 36 1 202 205 204 |
| 9 1 -11.183 0.0 4.5 -0.0 -50.321 0.0 -0.0 125.048 0.0 20.25 201 201 204 |
| 10 1 2.183 0.0 4.5 0.0 9.821 0.0 0.0 4.763 0.0 20.25 195 200 197 |
| 11 1 -4.5 -7.29 4.5 32.805 -20.25 -32.805 147.623 20.25 53.144 20.25 202 203 198 |
| 12 1 -4.5 7.29 4.5 -32.805 -20.25 32.805 -147.623 20.25 53.144 20.25 197 198 202 |
| 13 1 -4.5 0.0 -2.183 -0.0 9.821 -0.0 0.0 20.25 0.0 4.763 204 200 198 |
| 14 1 -4.5 0.0 11.183 -0.0 -50.321 0.0 -0.0 20.25 0.0 125.048 198 197 198 |
| 15 1 -4.5 0.0 4.5 -0.0 -20.25 0.0 -0.0 20.25 0.0 20.25 200 200 200 |

View: Центральний ортогональний композиційний план (код. знач. коеф.)

| № X0 X1 X2 X3 X1·X2 X1·X3 X2·X3 X1·X2·X3 X1·X1 X2·X2 X3·X3 Y1 Y2 Y3 |
|---|
| 1 1 -1 -1 1 1 -1 -1 1 1 1 1 202 205 202 |
| 2 1 -1 1 -1 -1 1 -1 1 1 1 1 202 199 195 |
| 3 1 1 -1 -1 -1 -1 1 1 1 1 1 199 199 205 |
| 4 1 1 1 1 1 1 1 1 1 1 1 199 198 205 |
| 5 1 -1 -1 -1 1 1 1 -1 1 1 1 198 203 199 |
| 6 1 -1 1 1 -1 -1 1 -1 1 1 1 202 200 195 |
| 7 1 1 -1 1 -1 1 -1 -1 1 1 1 199 198 199 |
| 8 1 1 1 -1 1 -1 -1 -1 1 1 1 202 205 204 |
| 9 1 -1.215 0 0 -0.0 -0.0 0 -0.0 1.476 0 0 201 201 204 |
| 10 1 1.215 0 0 0.0 0.0 0 0.0 1.476 0 0 195 200 197 |
| 11 1 0 -1.215 0 -0.0 0 -0.0 -0.0 0 1.476 0 202 203 198 |
| 12 1 0 1.215 0 0.0 0 0.0 0.0 0 1.476 0 197 198 202 |
| 13 1 0 0 -1.215 0 -0.0 -0.0 -0.0 0 0 1.476 204 200 198 |
| 14 1 0 0 1.215 0 0.0 0.0 0.0 0 0 1.476 198 197 198 |
| 15 1 0 0 0 0 0 0 0 0 0 0 200 200 200 |

Action: Розраховуємо натуральні значення коефіцієнтів.

Action: Рівняння регресії має вигляд (нат. знач. коеф.):

$y = 200.5977 + 0.2572 \cdot x_1 + 0.1523 \cdot x_2 + -0.2889 \cdot x_3 + 0.0317 \cdot x_1 \cdot x_2 + -0.0358 \cdot x_1 \cdot x_3 + -0.0064 \cdot x_2 \cdot x_3 + 0.0014 \cdot x_1 \cdot x_2 \cdot x_3 + 0.015 \cdot x_1 \cdot x_1 + 0.0189 \cdot x_2 \cdot x_2 + 0.0038 \cdot x_3 \cdot x_3$

Action: Розраховуємо кодовані значення коефіцієнтів.

Action: Рівняння регресії має вигляд (код. знач. коеф.):

$y = 199.2453 + -0.2133 \cdot x_1 + -0.2827 \cdot x_2 + -0.5154 \cdot x_3 + 1.25 \cdot x_1 \cdot x_2 + -1.0833 \cdot x_1 \cdot x_3 + -0.4167 \cdot x_2 \cdot x_3 + 0.25 \cdot x_1 \cdot x_2 \cdot x_3 + 0.4532 \cdot x_1 \cdot x_1 + 0.679 \cdot x_2 \cdot x_2 + 0.1145 \cdot x_3 \cdot x_3$

View: Перевірка знайдених коефіцієнтів (нат. знач. коеф., із квад. членами)

| | | | | | | | | | | | | | |
|----|---|---------|-------|--------|---------|---------|---------|----------|---------|--------|---------|---------|---------|
| 1 | 1 | -10 | -6 | 10 | 60 | -100 | -60 | 600 | 100 | 36 | 100 | 203.0 | 203.473 |
| 2 | 1 | -10 | 6 | -1 | -60 | 10 | -6 | 60 | 100 | 36 | 1 | 198.667 | 199.271 |
| 3 | 1 | 1 | -6 | -1 | -6 | -1 | 6 | 6 | 1 | 36 | 1 | 201.0 | 200.743 |
| 4 | 1 | 1 | 6 | 10 | 6 | 10 | 60 | 60 | 1 | 36 | 100 | 200.667 | 199.48 |
| 5 | 1 | -10 | -6 | -1 | 60 | 10 | 6 | -60 | 100 | 36 | 1 | 200.0 | 201.003 |
| 6 | 1 | -10 | 6 | 10 | -60 | -100 | 60 | -600 | 100 | 36 | 100 | 199.0 | 199.074 |
| 7 | 1 | 1 | -6 | 10 | -6 | 10 | -60 | -60 | 1 | 36 | 100 | 198.667 | 197.879 |
| 8 | 1 | 1 | 6 | -1 | 6 | -1 | -6 | -6 | 1 | 36 | 1 | 203.667 | 203.011 |
| 9 | 1 | -11.183 | 0.0 | 4.5 | -0.0 | -50.321 | 0.0 | -0.0 | 125.048 | 0.0 | 20.25 | 202.0 | 200.174 |
| 10 | 1 | 2.183 | 0.0 | 4.5 | 0.0 | 9.821 | 0.0 | 0.0 | 4.763 | 0.0 | 20.25 | 197.333 | 199.655 |
| 11 | 1 | -4.5 | -7.29 | 4.5 | 32.805 | -20.25 | -32.805 | 147.623 | 20.25 | 53.144 | 20.25 | 201.0 | 200.591 |
| 12 | 1 | -4.5 | 7.29 | 4.5 | -32.805 | -20.25 | 32.805 | -147.623 | 20.25 | 53.144 | 20.25 | 199.0 | 199.904 |
| 13 | 1 | -4.5 | 0.0 | -2.183 | -0.0 | 9.821 | -0.0 | 0.0 | 20.25 | 0.0 | 4.763 | 200.667 | 200.041 |
| 14 | 1 | -4.5 | 0.0 | 11.183 | -0.0 | -50.321 | 0.0 | -0.0 | 20.25 | 0.0 | 125.048 | 197.667 | 198.788 |
| 15 | 1 | -4.5 | 0.0 | 4.5 | -0.0 | -20.25 | 0.0 | -0.0 | 20.25 | 0.0 | 20.25 | 200.0 | 199.245 |

View: Перевірка знайдених коефіцієнтів (код. знач. коеф., із квад. членами)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Average Y[j] | Exp-tal Y[j] |
|----|----|--------|--------|--------|-------|-------|-------|----------|-------|-------|-------|--------------|--------------|
| 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | 1 | 1 | 1 | 203.0 | 203.473 |
| 2 | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 1 | 198.667 | 199.271 |
| 3 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 201.0 | 200.743 |
| 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 200.667 | 199.48 |
| 5 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 | 200.0 | 201.003 |
| 6 | 1 | -1 | 1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 199.0 | 199.074 |
| 7 | 1 | 1 | -1 | 1 | -1 | 1 | -1 | -1 | 1 | 1 | 1 | 198.667 | 197.879 |
| 8 | 1 | 1 | 1 | -1 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | 203.667 | 203.011 |
| 9 | 1 | -1.215 | 0 | 0 | -0.0 | -0.0 | 0 | -0.0 | 1.476 | 0 | 0 | 202.0 | 200.174 |
| 10 | 1 | 1.215 | 0 | 0 | 0.0 | 0.0 | 0 | 0.0 | 1.476 | 0 | 0 | 197.333 | 199.655 |
| 11 | 1 | 0 | -1.215 | 0 | -0.0 | 0 | -0.0 | -0.0 | 0 | 1.476 | 0 | 201.0 | 200.591 |
| 12 | 1 | 0 | 1.215 | 0 | 0.0 | 0 | 0.0 | 0.0 | 0 | 1.476 | 0 | 199.0 | 199.904 |
| 13 | 1 | 0 | 0 | -1.215 | 0 | -0.0 | -0.0 | -0.0 | 0 | 0 | 1.476 | 200.667 | 200.041 |
| 14 | 1 | 0 | 0 | 1.215 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0 | 1.476 | 197.667 | 198.788 |
| 15 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 200.0 | 199.245 |

Action: Перевіряємо нуль гіпотезу та корегуємо рівняння регресії:

f3 = 30, t = [642.6209, 0.6881, 0.9119, 1.6623, 4.0316, 3.494, 1.3439, 0.8063, 1.4616, 2.1899, 0.3692].

Action: Нове рівняння регресії має вигляд (нат. знач. коеф.):

$y = 200.5977 + 0 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 0.0317 \cdot x_1 \cdot x_2 + -0.0358 \cdot x_1 \cdot x_3 + 0 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_1 + 0.0189 \cdot x_2 \cdot x_2 + 0 \cdot x_3 \cdot x_3$

Action: Нове рівняння регресії має вигляд (код. знач. коеф.):

$y = 199.2453 + 0 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 1.25 \cdot x_1 \cdot x_2 + -1.0833 \cdot x_1 \cdot x_3 + 0 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_1 + 0.679 \cdot x_2 \cdot x_2 + 0 \cdot x_3 \cdot x_3$

Action: Перевіряємо адекватність моделі.

f3 = 30, f4 = 11, Fr = 5.723950269124205

Action: Модель не адекватна оригіналу.

Action: Змінюємо рівняння регресії.

Action: Оскільки всі моделі не адекватні, то почнемо експерименти з початку.

Action: Оскільки всі моделі не адекватні, то почнемо експерименти з початку.

Action: Розглянемо рівняння регресії із врахуванням квадратичних членів:

$$y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + b_3 \cdot x_3 + b_{12} \cdot x_1 \cdot x_2 + b_{13} \cdot x_1 \cdot x_3 + b_{23} \cdot x_2 \cdot x_3 + b_{123} \cdot x_1 \cdot x_2 \cdot x_3 + b_{11} \cdot x_1 \cdot x_1 + b_{22} \cdot x_2 \cdot x_2 + b_{33} \cdot x_3 \cdot x_3$$

Action: Маємо, $N = 15$, $K = 11$, $m = 3$.

Action: Складаємо матрицю планування і проведемо експерименти

Action: Дисперсія однорідна за критерієм Кохрена при $m = 3$:

$$f_1 = 2, f_2 = 15, G_p = 0.14823008849557526$$

View: Центральний ортогональний композиційний план (нат. знач. коеф.)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Y1 | Y2 | Y3 |
|----|----|---------|-------|--------|---------|---------|---------|----------|---------|--------|---------|-----|-----|-----|
| 1 | 1 | -10 | -6 | 10 | 60 | -100 | -60 | 600 | 100 | 36 | 100 | 197 | 205 | 200 |
| 2 | 1 | -10 | 6 | -1 | -60 | 10 | -6 | 60 | 100 | 36 | 1 | 199 | 196 | 204 |
| 3 | 1 | 1 | -6 | -1 | -6 | -1 | 6 | 6 | 1 | 36 | 1 | 200 | 199 | 205 |
| 4 | 1 | 1 | 6 | 10 | 6 | 10 | 60 | 60 | 1 | 36 | 100 | 196 | 203 | 200 |
| 5 | 1 | -10 | -6 | -1 | 60 | 10 | 6 | -60 | 100 | 36 | 1 | 198 | 203 | 198 |
| 6 | 1 | -10 | 6 | 10 | -60 | -100 | 60 | -600 | 100 | 36 | 100 | 200 | 202 | 201 |
| 7 | 1 | 1 | -6 | 10 | -6 | 10 | -60 | -60 | 1 | 36 | 100 | 205 | 197 | 201 |
| 8 | 1 | 1 | 6 | -1 | 6 | -1 | -6 | -6 | 1 | 36 | 1 | 200 | 198 | 204 |
| 9 | 1 | -11.183 | 0.0 | 4.5 | -0.0 | -50.321 | 0.0 | -0.0 | 125.048 | 0.0 | 20.25 | 195 | 202 | 204 |
| 10 | 1 | 2.183 | 0.0 | 4.5 | 0.0 | 9.821 | 0.0 | 0.0 | 4.763 | 0.0 | 20.25 | 203 | 203 | 200 |
| 11 | 1 | -4.5 | -7.29 | 4.5 | 32.805 | -20.25 | -32.805 | 147.623 | 20.25 | 53.144 | 20.25 | 205 | 197 | 197 |
| 12 | 1 | -4.5 | 7.29 | 4.5 | -32.805 | -20.25 | 32.805 | -147.623 | 20.25 | 53.144 | 20.25 | 203 | 204 | 200 |
| 13 | 1 | -4.5 | 0.0 | -2.183 | -0.0 | 9.821 | -0.0 | 0.0 | 20.25 | 0.0 | 4.763 | 200 | 204 | 201 |
| 14 | 1 | -4.5 | 0.0 | 11.183 | -0.0 | -50.321 | 0.0 | -0.0 | 20.25 | 0.0 | 125.048 | 203 | 202 | 201 |
| 15 | 1 | -4.5 | 0.0 | 4.5 | -0.0 | -20.25 | 0.0 | -0.0 | 20.25 | 0.0 | 20.25 | 201 | 204 | 200 |

View: Центральний ортогональний композиційний план (код. знач. коеф.)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Y1 | Y2 | Y3 |
|----|----|--------|----|----|-------|-------|-------|----------|-------|-------|-------|-----|-----|-----|
| 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | 1 | 1 | 1 | 197 | 205 | 200 |
| 2 | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 1 | 199 | 196 | 204 |
| 3 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 200 | 199 | 205 |
| 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 196 | 203 | 200 |
| 5 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 | 198 | 203 | 198 |
| 6 | 1 | -1 | 1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 200 | 202 | 201 |
| 7 | 1 | 1 | -1 | 1 | -1 | 1 | -1 | -1 | 1 | 1 | 1 | 205 | 197 | 201 |
| 8 | 1 | 1 | 1 | -1 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | 200 | 198 | 204 |
| 9 | 1 | -1.215 | 0 | 0 | -0.0 | -0.0 | 0 | -0.0 | 1.476 | 0 | 0 | 195 | 202 | 204 |
| 10 | 1 | 1.215 | 0 | 0 | 0.0 | 0.0 | 0 | 0.0 | 1.476 | 0 | 0 | 203 | 203 | 200 |

| |
|--|
| 11 1 0 -1.215 0 -0.0 0 -0.0 -0.0 0 1.476 0 205 197 197 |
| 12 1 0 1.215 0 0.0 0 0.0 0.0 0 1.476 0 203 204 200 |
| 13 1 0 0 -1.215 0 -0.0 -0.0 -0.0 0 0 1.476 200 204 201 |
| 14 1 0 0 1.215 0 0.0 0.0 0.0 0 0 1.476 203 202 201 |
| 15 1 0 0 0 0 0 0 0 0 0 0 201 204 200 |

Action: Розраховуємо натуральні значення коефіцієнтів.

Action: Рівняння регресії має вигляд (нат. знач. коеф.):

$y = 201.9335 + -0.0494 \cdot x_1 + 0.0038 \cdot x_2 + -0.0004 \cdot x_3 + -0.0057 \cdot x_1 \cdot x_2 + -0.0152 \cdot x_1 \cdot x_3 + -0.0044 \cdot x_2 \cdot x_3 + -0.0007 \cdot x_1 \cdot x_2 \cdot x_3 + -0.0199 \cdot x_1 \cdot x_1 + -0.0198 \cdot x_2 \cdot x_2 + -0.0049 \cdot x_3 \cdot x_3$

Action: Розраховуємо кодовані значення коефіцієнтів.

Action: Рівняння регресії має вигляд (код. знач. коеф.):

$y = 201.9584 + 0.3371 \cdot x_1 + 0.1437 \cdot x_2 + 0.1283 \cdot x_3 + -0.2917 \cdot x_1 \cdot x_2 + -0.4583 \cdot x_1 \cdot x_3 + -0.0417 \cdot x_2 \cdot x_3 + -0.125 \cdot x_1 \cdot x_2 \cdot x_3 + -0.6011 \cdot x_1 \cdot x_1 + -0.714 \cdot x_2 \cdot x_2 + -0.1495 \cdot x_3 \cdot x_3$

View: Перевірка знайдених коефіцієнтів (нат. знач. коеф., із квад. членами)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Average Y[j] | Exp-tal Y[j] |
|----|----|---------|-------|--------|---------|---------|---------|----------|---------|--------|---------|--------------|--------------|
| 1 | 1 | -10 | -6 | 10 | 60 | -100 | -60 | 600 | 100 | 36 | 100 | 200.667 | 200.225 |
| 2 | 1 | -10 | 6 | -1 | -60 | 10 | -6 | 60 | 100 | 36 | 1 | 199.667 | 199.922 |
| 3 | 1 | 1 | -6 | -1 | -6 | -1 | 6 | 6 | 1 | 36 | 1 | 201.333 | 201.142 |
| 4 | 1 | 1 | 6 | 10 | 6 | 10 | 60 | 60 | 1 | 36 | 100 | 199.667 | 200.186 |
| 5 | 1 | -10 | -6 | -1 | 60 | 10 | 6 | -60 | 100 | 36 | 1 | 199.667 | 199.218 |
| 6 | 1 | -10 | 6 | 10 | -60 | -100 | 60 | -600 | 100 | 36 | 100 | 201.0 | 201.262 |
| 7 | 1 | 1 | -6 | 10 | -6 | 10 | -60 | -60 | 1 | 36 | 100 | 201.0 | 200.815 |
| 8 | 1 | 1 | 6 | -1 | 6 | -1 | -6 | -6 | 1 | 36 | 1 | 200.667 | 201.179 |
| 9 | 1 | -11.183 | 0.0 | 4.5 | -0.0 | -50.321 | 0.0 | -0.0 | 125.048 | 0.0 | 20.25 | 200.333 | 200.661 |
| 10 | 1 | 2.183 | 0.0 | 4.5 | 0.0 | 9.821 | 0.0 | 0.0 | 4.763 | 0.0 | 20.25 | 202.0 | 201.48 |
| 11 | 1 | -4.5 | -7.29 | 4.5 | 32.805 | -20.25 | -32.805 | 147.623 | 20.25 | 53.144 | 20.25 | 199.667 | 200.73 |
| 12 | 1 | -4.5 | 7.29 | 4.5 | -32.805 | -20.25 | 32.805 | -147.623 | 20.25 | 53.144 | 20.25 | 202.333 | 201.079 |
| 13 | 1 | -4.5 | 0.0 | -2.183 | -0.0 | 9.821 | -0.0 | 0.0 | 20.25 | 0.0 | 4.763 | 201.667 | 201.582 |
| 14 | 1 | -4.5 | 0.0 | 11.183 | -0.0 | -50.321 | 0.0 | -0.0 | 20.25 | 0.0 | 125.048 | 202.0 | 201.893 |
| 15 | 1 | -4.5 | 0.0 | 4.5 | -0.0 | -20.25 | 0.0 | -0.0 | 20.25 | 0.0 | 20.25 | 201.667 | 201.958 |

View: Перевірка знайдених коефіцієнтів (код. знач. коеф., із квад. членами)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Average Y[j] | Exp-tal Y[j] |
|----|----|--------|----|----|-------|-------|-------|----------|-------|-------|-------|--------------|--------------|
| 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | 1 | 1 | 1 | 200.667 | 200.225 |
| 2 | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 1 | 199.667 | 199.922 |
| 3 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 201.333 | 201.142 |
| 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 199.667 | 200.186 |
| 5 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 | 199.667 | 199.218 |
| 6 | 1 | -1 | 1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 201.0 | 201.262 |
| 7 | 1 | 1 | -1 | 1 | -1 | 1 | -1 | -1 | 1 | 1 | 1 | 201.0 | 200.815 |
| 8 | 1 | 1 | 1 | -1 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | 200.667 | 201.179 |
| 9 | 1 | -1.215 | 0 | 0 | -0.0 | -0.0 | 0 | -0.0 | 1.476 | 0 | 0 | 200.333 | 200.661 |
| 10 | 1 | 1.215 | 0 | 0 | 0.0 | 0.0 | 0 | 0.0 | 1.476 | 0 | 0 | 202.0 | 201.48 |

| |
|--|
| 11 1 0 -1.215 0 -0.0 0 -0.0 -0.0 0 1.476 0 199.667 200.73 |
| 12 1 0 1.215 0 0.0 0 0.0 0.0 0 1.476 0 202.333 201.079 |
| 13 1 0 0 -1.215 0 -0.0 -0.0 -0.0 0 0 1.476 201.667 201.582 |
| 14 1 0 0 1.215 0 0.0 0.0 0.0 0 0 1.476 202.0 201.893 |
| 15 1 0 0 0 0 0 0 0 0 0 0 201.667 201.958 |

Action: Перевіряємо нуль гіпотезу та корегуємо рівняння регресії:

$f_3 = 30$, $t = [523.5411, 0.8738, 0.3724, 0.3325, 0.7561, 1.1881, 0.108, 0.324, 1.5583, 1.851, 0.3876]$.

Action: Нове рівняння регресії має вигляд (нат. знач. коеф.):

$y = 201.9335 + 0 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 0 \cdot x_1 \cdot x_2 + 0 \cdot x_1 \cdot x_3 + 0 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_1 + 0 \cdot x_2 \cdot x_2 + 0 \cdot x_3 \cdot x_3$

Action: Нове рівняння регресії має вигляд (код. знач. коеф.):

$y = 201.9584 + 0 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 0 \cdot x_1 \cdot x_2 + 0 \cdot x_1 \cdot x_3 + 0 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_2 \cdot x_3 + 0 \cdot x_1 \cdot x_1 + 0 \cdot x_2 \cdot x_2 + 0 \cdot x_3 \cdot x_3$

Action: Перевіряємо адекватність моделі.

$f_3 = 30$, $f_4 = 14$, $F_p = 0.9196402821996418$

Action: Модель адекватна оригіналу.

Action: Виводимо результати.

View: Перевірка знайдених коефіцієнтів (нат. знач. коеф., із квад. членами)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Average Y[j] | Exp-tal Y[j] |
|----|----|---------|-------|--------|---------|---------|---------|----------|---------|--------|---------|--------------|--------------|
| 1 | 1 | -10 | -6 | 10 | 60 | -100 | -60 | 600 | 100 | 36 | 100 | 200.667 | 201.933 |
| 2 | 1 | -10 | 6 | -1 | -60 | 10 | -6 | 60 | 100 | 36 | 1 | 199.667 | 201.933 |
| 3 | 1 | 1 | -6 | -1 | -6 | -1 | 6 | 6 | 1 | 36 | 1 | 201.333 | 201.933 |
| 4 | 1 | 1 | 6 | 10 | 6 | 10 | 60 | 60 | 1 | 36 | 100 | 199.667 | 201.933 |
| 5 | 1 | -10 | -6 | -1 | 60 | 10 | 6 | -60 | 100 | 36 | 1 | 199.667 | 201.933 |
| 6 | 1 | -10 | 6 | 10 | -60 | -100 | 60 | -600 | 100 | 36 | 100 | 201.0 | 201.933 |
| 7 | 1 | 1 | -6 | 10 | -6 | 10 | -60 | -60 | 1 | 36 | 100 | 201.0 | 201.933 |
| 8 | 1 | 1 | 6 | -1 | 6 | -1 | -6 | -6 | 1 | 36 | 1 | 200.667 | 201.933 |
| 9 | 1 | -11.183 | 0.0 | 4.5 | -0.0 | -50.321 | 0.0 | -0.0 | 125.048 | 0.0 | 20.25 | 200.333 | 201.933 |
| 10 | 1 | 2.183 | 0.0 | 4.5 | 0.0 | 9.821 | 0.0 | 0.0 | 4.763 | 0.0 | 20.25 | 202.0 | 201.933 |
| 11 | 1 | -4.5 | -7.29 | 4.5 | 32.805 | -20.25 | -32.805 | 147.623 | 20.25 | 53.144 | 20.25 | 199.667 | 201.933 |
| 12 | 1 | -4.5 | 7.29 | 4.5 | -32.805 | -20.25 | 32.805 | -147.623 | 20.25 | 53.144 | 20.25 | 202.333 | 201.933 |
| 13 | 1 | -4.5 | 0.0 | -2.183 | -0.0 | 9.821 | -0.0 | 0.0 | 20.25 | 0.0 | 4.763 | 201.667 | 201.933 |
| 14 | 1 | -4.5 | 0.0 | 11.183 | -0.0 | -50.321 | 0.0 | -0.0 | 20.25 | 0.0 | 125.048 | 202.0 | 201.933 |
| 15 | 1 | -4.5 | 0.0 | 4.5 | -0.0 | -20.25 | 0.0 | -0.0 | 20.25 | 0.0 | 20.25 | 201.667 | 201.933 |

View: Перевірка знайдених коефіцієнтів (код. знач. коеф., із квад. членами)

| № | X0 | X1 | X2 | X3 | X1·X2 | X1·X3 | X2·X3 | X1·X2·X3 | X1·X1 | X2·X2 | X3·X3 | Average Y[j] | Exp-tal Y[j] |
|---|----|----|----|----|-------|-------|-------|----------|-------|-------|-------|--------------|--------------|
| 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | 1 | 1 | 1 | 200.667 | 201.958 |
| 2 | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 1 | 199.667 | 201.958 |
| 3 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 201.333 | 201.958 |
| 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 199.667 | 201.958 |
| 5 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 | 199.667 | 201.958 |
| 6 | 1 | -1 | 1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 201.0 | 201.958 |
| 7 | 1 | 1 | -1 | 1 | -1 | 1 | -1 | -1 | 1 | 1 | 1 | 201.0 | 201.958 |

| |
|--|
| 8 1 1 1 -1 1 -1 -1 -1 1 1 1 200.667 201.958 |
| 9 1 -1.215 0 0 -0.0 -0.0 0 -0.0 1.476 0 0 200.333 201.958 |
| 10 1 1.215 0 0 0.0 0.0 0 0.0 1.476 0 0 202.0 201.958 |
| 11 1 0 -1.215 0 -0.0 0 -0.0 -0.0 0 1.476 0 199.667 201.958 |
| 12 1 0 1.215 0 0.0 0 0.0 0.0 0 1.476 0 202.333 201.958 |
| 13 1 0 0 -1.215 0 -0.0 -0.0 -0.0 0 0 1.476 201.667 201.958 |
| 14 1 0 0 1.215 0 0.0 0.0 0.0 0 0 1.476 202.0 201.958 |
| 15 1 0 0 0 0 0 0 0 0 0 0 201.667 201.958 |

Висновки

В ході виконання лабораторної роботи було розглянуто проведення трьохфакторного експерименту при використанні рівняння регресії з урахуванням квадратичних членів (центральний ортогональний композиційний план).

Вказане вище рівняння регресії застосовується в тому випадку, коли рівняння регресії без урахування взаємодії факторів та рівняння регресії з урахуванням взаємодії факторів є неадекватними за критерієм Фішера.