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(ЦЕНТРАЛЬНІЙ ОРТОГОНАЛЬНИЙ КОМПОЗИЦІЙНИЙ ПЛАН)»

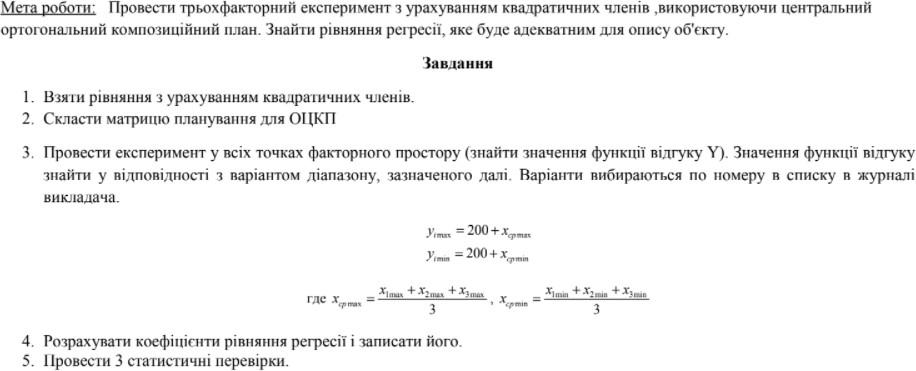
Виконав: Студент групи ІВ-92

Слободяник О.К.

Варіант 223

Перевірив: Ас. Регіда П.Г.

Київ 2021 р.





# Лістинг програми:

**from** random **import** randint  
**from** numpy.linalg **import** det  
**from** copy **import** deepcopy  
**from** scipy.stats **import** t  
  
  
**def** Naturalize(MatrixOfPlan, MinMaxArr, flag):  
 result = []  
 **for** i **in** range(len(MatrixOfPlan)):  
 **if** i < 8:  
 result.append(MinMaxArr[1]) **if** MatrixOfPlan[i] == 1 **else** result.append(MinMaxArr[0])  
 **else**:  
 x0 = (max(MinMaxArr) + min(MinMaxArr)) / 2  
 dx = x0 - min(MinMaxArr)  
 value = **None  
 if** flag == 1:  
 value = MatrixOfPlan[i] \* dx + x0 **if** i == 8 **or** 9 **else** x0  
 **elif** flag == 2:  
 value = MatrixOfPlan[i] \* dx + x0 **if** i == 10 **or** 11 **else** x0  
 **elif** flag == 3:  
 value = MatrixOfPlan[i] \* dx + x0 **if** i == 12 **or** 13 **else** x0  
 result.append(value)  
 **return** result  
  
  
**def** Сocharan(y\_arr, y\_avg, m, N):  
 dispersion = []  
 **for** i **in** range(len(y\_arr[0])):  
 current\_sum = 0  
 **for** j **in** range(len(y\_arr)):  
 current\_sum += (y\_arr[j][i] - y\_avg[j]) \*\* 2  
 dispersion.append(current\_sum / len(y\_arr))  
  
 print(**'dispersion:'**, dispersion)  
  
 gp = max(dispersion) / sum(dispersion)  
 print(**'Gp ='**, gp)  
  
 *# Рівень значимості q = 0.05  
 # f1 = m - 1  
 # f2 = N  
  
 # За таблицею Gт = 0.3346* **if** gp < 0.3346:  
 print(**'Дисперсія однорідна'**)  
 **return** dispersion  
 **else**:  
 print(**'Дисперсія неоднорідна'**)  
 **return None  
  
  
def** Students(plan1x0, plan1x1, plan1x2, plan1x3, y\_avg\_arr, dispersion, m):  
 *# Оцінка значимості коефіцієнтів регресії згідно критерію Стьюдента* s2b = sum(dispersion) / 15  
 s2bs\_avg = s2b / 15 \* m  
 sb = s2bs\_avg \*\* (1 / 2)  
  
 beta\_arr = [  
 sum([y\_avg\_arr[i] \* plan1x0[i] **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x1[i] **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x2[i] **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x3[i] **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x1[i] \* plan1x2[i] **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x1[i] \* plan1x3[i] **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x2[i] \* plan1x3[i] **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x1[i] \* plan1x2[i] \* plan1x3[i] **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x1[i] \*\* 2 **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x2[i] \*\* 2 **for** i **in** range(15)]) / 15,  
 sum([y\_avg\_arr[i] \* plan1x3[i] \*\* 2 **for** i **in** range(15)]) / 15  
 ]  
  
 print(**'beta:'**, beta\_arr)  
 t\_arr = [abs(beta\_arr[i]) / sb **for** i **in** range(11)]  
 print(**'t:'**, t\_arr)  
  
 *# f3 = f1\*f2 = 2\*15 = 30* f1 = m - 1  
 f2 = 15  
 f3 = f1 \* f2  
  
 b\_arr = []  
 **for** i **in** range(len(t\_arr)):  
 **if** t\_arr[i] > t.ppf(q=0.975, df=f3):  
 b\_arr.append(t\_arr[i])  
 **else**:  
 print(**f'Коефіцієнт b{**i**} приймаємо не значним'**)  
 b\_arr.append(0)  
  
 **return** b\_arr, s2b  
  
  
**def** Fisher(b\_arr, s2b, y\_avg, y\_res, m):  
 *# Критерій Фішера* d = len([i **for** i **in** b\_arr **if** i != 0]) *# кількість значимих коефіцієнтів* print(**f'd = {**d**}'**)  
 s2\_ad = m \* sum([(y\_res[i] - y\_avg[i]) \*\* 2 **for** i **in** range(15)]) / 15 - d  
 fp = s2\_ad / s2b  
 print(**f'Fp = {**fp**}'**)  
  
  
 *# Fт = 2.1* **if** fp > 2.1:  
 print(**'Рівняння регресії неадекватно оригіналу при рівні значимості 0.05'**)  
 **else**:  
 print(**'Рівняння регресії адекватно оригіналу при рівні значимості 0.05'**)  
  
  
**def** main(m):  
 N = 15  
  
 x1 = [-10, 10]  
 x2 = [-10, 2]  
 x3 = [-10, 4]  
  
 *# Величина зоряного плеча* l = 1.215  
  
 *# Матриця планування з нормованих значень* plan1x0 = [1 **for** \_ **in** range(N)]  
 plan1x1 = [-1, -1, 1, 1, -1, -1, 1, 1, -l, l, 0, 0, 0, 0, 0]  
 plan1x2 = [-1, 1, -1, 1, -1, 1, -1, 1, 0, 0, -l, l, 0, 0, 0]  
 plan1x3 = [1, -1, -1, 1, -1, 1, 1, -1, 0, 0, 0, 0, -l, l, 0]  
 print(**'x1:'**, plan1x1)  
 print(**'x2:'**, plan1x2)  
 print(**'x3:'**, plan1x3)  
 print(**'-'** \* 100)  
  
 *# Матриця планування з натуралізованих значень* plan2x1 = Naturalize(plan1x1, x1, 1)  
 plan2x2 = Naturalize(plan1x2, x2, 2)  
 plan2x3 = Naturalize(plan1x3, x3, 3)  
  
 *# Мультиплікативні значення факторів* plan2x4 = [plan2x1[i] \* plan2x2[i] **for** i **in** range(len(plan2x1))]  
 plan2x5 = [plan2x1[i] \* plan2x3[i] **for** i **in** range(len(plan2x1))]  
 plan2x6 = [plan2x2[i] \* plan2x3[i] **for** i **in** range(len(plan2x1))]  
 plan2x7 = [plan2x1[i] \* plan2x2[i] \* plan2x3[i] **for** i **in** range(len(plan2x1))]  
  
 *# Квадратичні значення факторів* plan2x8 = [plan2x1[i] \*\* 2 **for** i **in** range(len(plan2x1))]  
 plan2x9 = [plan2x2[i] \*\* 2 **for** i **in** range(len(plan2x1))]  
 plan2x10 = [plan2x3[i] \*\* 2 **for** i **in** range(len(plan2x1))]  
  
 print(**f'x1: {**plan2x1**}'**)  
 print(**f'x2: {**plan2x2**}'**)  
 print(**f'x3: {**plan2x3**}'**)  
 print(**f'x4: {**plan2x4**}'**)  
 print(**f'x5: {**plan2x5**}'**)  
 print(**f'x6: {**plan2x6**}'**)  
 print(**f'x7: {**plan2x7**}'**)  
 print(**f'x8: {**plan2x8**}'**)  
 print(**f'x9: {**plan2x9**}'**)  
 print(**f'x10: {**plan2x10**}'**)  
  
 x\_avg\_max = (max(plan2x1) + max(plan2x2) + max(plan2x3)) / 3  
 x\_avg\_min = (min(plan2x1) + min(plan2x2) + min(plan2x3)) / 3  
 print()  
 print(**f'x\_avg\_max = {**x\_avg\_max**}'**)  
 print(**f'x\_avg\_min = {**x\_avg\_min**}'**)  
 print(**'-'** \* 100)  
  
 *# Діапазон y* y\_max = int(200 + x\_avg\_max)  
 y\_min = int(200 + x\_avg\_min)  
 print(**f'y\_max = {**y\_max**}'**)  
 print(**f'y\_min = {**y\_min**}'**)  
 print()  
  
 y\_arr = [[randint(y\_min, y\_max) **for** \_ **in** range(N)] **for** \_ **in** range(m)]  
 **for** i **in** range(len(y\_arr)):  
 print(**f'y{**i + 1**}: {**y\_arr[i]**}'**)  
  
 y\_avg = []  
 **for** i **in** range(len(y\_arr[0])):  
 current\_sum = 0  
 **for** j **in** range(len(y\_arr)):  
 current\_sum += y\_arr[j][i]  
 y\_avg.append(current\_sum / len(y\_arr))  
 print(**'y average:'**, y\_avg)  
 print(**'-'** \* 100)  
  
 dispersion = Сocharan(y\_arr, y\_avg, m, N)  
 **if** dispersion:  
 mx1 = sum(plan2x1) / len(plan2x1)  
 mx2 = sum(plan2x2) / len(plan2x2)  
 mx3 = sum(plan2x3) / len(plan2x3)  
 mx4 = sum(plan2x4) / len(plan2x4)  
 mx5 = sum(plan2x5) / len(plan2x5)  
 mx6 = sum(plan2x6) / len(plan2x6)  
 mx7 = sum(plan2x7) / len(plan2x7)  
 mx8 = sum(plan2x8) / len(plan2x8)  
 mx9 = sum(plan2x9) / len(plan2x9)  
 mx10 = sum(plan2x10) / len(plan2x10)  
 my = sum(y\_avg) / len(y\_avg)  
  
 a1 = sum([y\_avg[i] \* plan2x1[i] **for** i **in** range(len(plan2x1))]) / len(plan2x1)  
 a11 = mx8  
 a12 = mx4  
 a13 = mx5  
 a14 = sum([plan2x1[i] \* plan2x4[i] **for** i **in** range(len(plan2x1))]) / len(plan2x1)  
 a15 = sum([plan2x1[i] \* plan2x5[i] **for** i **in** range(len(plan2x1))]) / len(plan2x1)  
 a16 = sum([plan2x1[i] \* plan2x6[i] **for** i **in** range(len(plan2x1))]) / len(plan2x1)  
 a17 = sum([plan2x1[i] \* plan2x7[i] **for** i **in** range(len(plan2x1))]) / len(plan2x1)  
 a18 = sum([plan2x1[i] \* plan2x8[i] **for** i **in** range(len(plan2x1))]) / len(plan2x1)  
 a19 = sum([plan2x1[i] \* plan2x9[i] **for** i **in** range(len(plan2x1))]) / len(plan2x1)  
  
 a2 = sum([y\_avg[i] \* plan2x2[i] **for** i **in** range(len(plan2x1))]) / len(plan2x2)  
 a21 = a12  
 a22 = mx9  
 a23 = mx6  
 a24 = sum([plan2x2[i] \* plan2x4[i] **for** i **in** range(len(plan2x2))]) / len(plan2x2)  
 a25 = sum([plan2x2[i] \* plan2x5[i] **for** i **in** range(len(plan2x2))]) / len(plan2x2)  
 a26 = sum([plan2x2[i] \* plan2x6[i] **for** i **in** range(len(plan2x2))]) / len(plan2x2)  
 a27 = sum([plan2x2[i] \* plan2x7[i] **for** i **in** range(len(plan2x2))]) / len(plan2x2)  
 a28 = sum([plan2x2[i] \* plan2x8[i] **for** i **in** range(len(plan2x2))]) / len(plan2x2)  
 a29 = sum([plan2x2[i] \* plan2x9[i] **for** i **in** range(len(plan2x2))]) / len(plan2x2)  
  
 a3 = sum([y\_avg[i] \* plan2x3[i] **for** i **in** range(len(plan2x3))]) / len(plan2x3)  
 a31 = a13  
 a32 = a23  
 a33 = mx10  
 a34 = sum([plan2x3[i] \* plan2x4[i] **for** i **in** range(len(plan2x3))]) / len(plan2x3)  
 a35 = sum([plan2x3[i] \* plan2x5[i] **for** i **in** range(len(plan2x3))]) / len(plan2x3)  
 a36 = sum([plan2x3[i] \* plan2x6[i] **for** i **in** range(len(plan2x3))]) / len(plan2x3)  
 a37 = sum([plan2x3[i] \* plan2x7[i] **for** i **in** range(len(plan2x3))]) / len(plan2x3)  
 a38 = sum([plan2x3[i] \* plan2x8[i] **for** i **in** range(len(plan2x3))]) / len(plan2x3)  
 a39 = sum([plan2x3[i] \* plan2x9[i] **for** i **in** range(len(plan2x3))]) / len(plan2x3)  
  
 a4 = sum([y\_avg[i] \* plan2x4[i] **for** i **in** range(len(plan2x4))]) / len(plan2x4)  
 a41 = a14  
 a42 = a24  
 a43 = a34  
 a44 = sum([plan2x4[i] \*\* 2 **for** i **in** range(len(plan2x4))]) / len(plan2x4)  
 a45 = sum([plan2x4[i] \* plan2x5[i] **for** i **in** range(len(plan2x4))]) / len(plan2x4)  
 a46 = sum([plan2x4[i] \* plan2x6[i] **for** i **in** range(len(plan2x4))]) / len(plan2x4)  
 a47 = sum([plan2x4[i] \* plan2x7[i] **for** i **in** range(len(plan2x4))]) / len(plan2x4)  
 a48 = sum([plan2x4[i] \* plan2x8[i] **for** i **in** range(len(plan2x4))]) / len(plan2x4)  
 a49 = sum([plan2x4[i] \* plan2x9[i] **for** i **in** range(len(plan2x4))]) / len(plan2x4)  
  
 a5 = sum([y\_avg[i] \* plan2x5[i] **for** i **in** range(len(plan2x5))]) / len(plan2x5)  
 a51 = a15  
 a52 = a25  
 a53 = a35  
 a54 = a45  
 a55 = sum([plan2x5[i] \*\* 2 **for** i **in** range(len(plan2x5))]) / len(plan2x5)  
 a56 = sum([plan2x5[i] \* plan2x6[i] **for** i **in** range(len(plan2x5))]) / len(plan2x5)  
 a57 = sum([plan2x5[i] \* plan2x7[i] **for** i **in** range(len(plan2x5))]) / len(plan2x5)  
 a58 = sum([plan2x5[i] \* plan2x8[i] **for** i **in** range(len(plan2x5))]) / len(plan2x5)  
 a59 = sum([plan2x5[i] \* plan2x9[i] **for** i **in** range(len(plan2x5))]) / len(plan2x5)  
  
 a6 = sum([y\_avg[i] \* plan2x6[i] **for** i **in** range(len(plan2x6))]) / len(plan2x6)  
 a61 = a16  
 a62 = a26  
 a63 = a36  
 a64 = a46  
 a65 = a56  
 a66 = sum([plan2x6[i] \*\* 2 **for** i **in** range(len(plan2x6))]) / len(plan2x6)  
 a67 = sum([plan2x6[i] \* plan2x7[i] **for** i **in** range(len(plan2x6))]) / len(plan2x6)  
 a68 = sum([plan2x6[i] \* plan2x8[i] **for** i **in** range(len(plan2x6))]) / len(plan2x6)  
 a69 = sum([plan2x6[i] \* plan2x9[i] **for** i **in** range(len(plan2x6))]) / len(plan2x6)  
  
 a7 = sum([y\_avg[i] \* plan2x7[i] **for** i **in** range(len(plan2x7))]) / len(plan2x7)  
 a71 = a17  
 a72 = a27  
 a73 = a37  
 a74 = a47  
 a75 = a57  
 a76 = a67  
 a77 = sum([plan2x7[i] \*\* 2 **for** i **in** range(len(plan2x7))]) / len(plan2x7)  
 a78 = sum([plan2x7[i] \* plan2x8[i] **for** i **in** range(len(plan2x7))]) / len(plan2x7)  
 a79 = sum([plan2x7[i] \* plan2x9[i] **for** i **in** range(len(plan2x7))]) / len(plan2x7)  
  
 a8 = sum([y\_avg[i] \* plan2x8[i] **for** i **in** range(len(plan2x8))]) / len(plan2x8)  
 a81 = a18  
 a82 = a28  
 a83 = a38  
 a84 = a48  
 a85 = a58  
 a86 = a68  
 a87 = a78  
 a88 = sum([plan2x8[i] \*\* 2 **for** i **in** range(len(plan2x8))]) / len(plan2x8)  
 a89 = sum([plan2x8[i] \* plan2x9[i] **for** i **in** range(len(plan2x8))]) / len(plan2x8)  
  
 a9 = sum([y\_avg[i] \* plan2x9[i] **for** i **in** range(len(plan2x9))]) / len(plan2x9)  
 a91 = a19  
 a92 = a29  
 a93 = a39  
 a94 = a49  
 a95 = a59  
 a96 = a69  
 a97 = a79  
 a98 = a89  
 a99 = sum([plan2x9[i] \*\* 2 **for** i **in** range(len(plan2x9))]) / len(plan2x9)  
  
 a10 = sum([y\_avg[i] \* plan2x10[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a101 = sum([plan2x10[i] \* plan2x1[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a102 = sum([plan2x10[i] \* plan2x2[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a103 = sum([plan2x10[i] \* plan2x3[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a104 = sum([plan2x10[i] \* plan2x4[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a105 = sum([plan2x10[i] \* plan2x5[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a106 = sum([plan2x10[i] \* plan2x6[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a107 = sum([plan2x10[i] \* plan2x7[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a108 = sum([plan2x10[i] \* plan2x8[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a109 = sum([plan2x10[i] \* plan2x9[i] **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
 a1010 = sum([plan2x10[i] \*\* 2 **for** i **in** range(len(plan2x10))]) / len(plan2x10)  
  
 main\_determinant = [[1, mx1, mx2, mx3, mx4, mx5, mx6, mx7, mx8, mx9, mx10],  
 [mx1, a11, a21, a31, a41, a51, a61, a71, a81, a91, a101],  
 [mx2, a12, a22, a32, a42, a52, a62, a72, a82, a92, a102],  
 [mx3, a13, a23, a33, a43, a53, a63, a73, a83, a93, a103],  
 [mx4, a14, a24, a34, a44, a54, a64, a74, a84, a94, a104],  
 [mx5, a15, a25, a35, a45, a55, a65, a75, a85, a95, a105],  
 [mx6, a16, a26, a36, a46, a56, a66, a76, a86, a96, a106],  
 [mx7, a17, a27, a37, a47, a57, a67, a77, a87, a97, a107],  
 [mx8, a18, a28, a38, a48, a58, a68, a78, a88, a98, a108],  
 [mx9, a19, a29, a39, a49, a59, a69, a79, a89, a99, a109],  
 [mx10, a101, a102, a103, a104, a105, a106, a107, a108, a109, a1010]]  
  
 column\_to\_change = [my, a1, a2, a3, a4, a5, a6, a7, a8, a9, a10]  
 main\_determinant\_value = det(main\_determinant)  
  
 matrices = []  
 **for** i **in** range(len(main\_determinant[0])):  
 new\_matrix = deepcopy(main\_determinant)  
 **for** j **in** range(len(main\_determinant)):  
 new\_matrix[j][i] = column\_to\_change[j]  
 matrices.append(new\_matrix)  
  
 b\_list = []  
 **for** i **in** range(len(matrices)):  
 b\_list.append(det(matrices[i]) / main\_determinant\_value)  
 print(**'-'** \* 100)  
 print(**f'b: {**b\_list**}'**)  
  
 y\_list = []  
 **for** i **in** range(len(plan2x1)):  
 y = b\_list[0] + b\_list[1] \* plan2x1[i] + b\_list[2] \* plan2x2[i] + b\_list[3] \* plan2x3[i] +\  
 b\_list[4] \* plan2x4[i] + b\_list[5] \* plan2x5[i] + b\_list[6] \* plan2x6[i] + b\_list[7] \* plan2x7[i] +\  
 b\_list[8] \* plan2x8[i] + b\_list[9] \* plan2x9[i] + b\_list[10] \* plan2x10[i]  
 y\_list.append(y)  
 print(**f'y = {**y**}; y avg = {**y\_avg[i]**}'**)  
 print(**'-'** \* 100)  
  
 t\_arr, s2b = Students(plan1x0, plan1x1, plan1x2, plan1x3, y\_avg, dispersion, m)  
  
 b\_arr = []  
 **for** i **in** range(len(b\_list)):  
 b = b\_list[i] **if** t\_arr[i] != 0 **else** 0  
 b\_arr.append(b)  
 print(**'-'** \* 100)  
  
 y\_res = []  
 **for** i **in** range(N):  
 y = b\_arr[0] + b\_arr[1] \* plan1x1[i] + b\_arr[2] \* plan1x2[i] + b\_arr[3] \* plan1x3[i] +\  
 b\_arr[4] \* plan1x1[i] \* plan1x2[i] + b\_arr[5] \* plan1x1[i] \* plan1x3[i] +\  
 b\_arr[6] \* plan1x2[i] \* plan1x3[i] + b\_arr[7] \* plan1x1[i] \* plan1x2[i] \* plan1x3[i] +\  
 b\_arr[8] \* plan1x1[i] \*\* 2 + b\_arr[9] \* plan1x2[i] \*\* 2 + b\_arr[10] \* plan1x3[i] \*\* 2  
 print(**f'ŷ = {**y**}'**)  
 y\_res.append(y)  
  
 Fisher(b\_arr, s2b, y\_avg, y\_res, m)  
 **else**:  
 main(m+1)  
 exit()  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 main(m=3)

# Результат виконання програми:

x1: [-1, -1, 1, 1, -1, -1, 1, 1, -1.215, 1.215, 0, 0, 0, 0, 0]

x2: [-1, 1, -1, 1, -1, 1, -1, 1, 0, 0, -1.215, 1.215, 0, 0, 0]

x3: [1, -1, -1, 1, -1, 1, 1, -1, 0, 0, 0, 0, -1.215, 1.215, 0]

----------------------------------------------------------------------------------------------------

x1: [-10, -10, 10, 10, -10, -10, 10, 10, -12.15, 12.15, 0.0, 0.0, 0.0, 0.0, 0.0]

x2: [-10, 2, -10, 2, -10, 2, -10, 2, -4.0, -4.0, -11.290000000000001, 3.290000000000001, -4.0, -4.0, -4.0]

x3: [4, -10, -10, 4, -10, 4, 4, -10, -3.0, -3.0, -3.0, -3.0, -11.505, 5.505000000000001, -3.0]

x4: [100, -20, -100, 20, 100, -20, -100, 20, 48.6, -48.6, -0.0, 0.0, -0.0, -0.0, -0.0]

x5: [-40, 100, -100, 40, 100, -40, 40, -100, 36.45, -36.45, -0.0, -0.0, -0.0, 0.0, -0.0]

x6: [-40, -20, 100, 8, 100, 8, -40, -20, 12.0, 12.0, 33.870000000000005, -9.870000000000003, 46.02, -22.020000000000003, 12.0]

x7: [400, 200, 1000, 80, -1000, -80, -400, -200, -145.8, 145.8, 0.0, -0.0, 0.0, -0.0, 0.0]

x8: [100, 100, 100, 100, 100, 100, 100, 100, 147.6225, 147.6225, 0.0, 0.0, 0.0, 0.0, 0.0]

x9: [100, 4, 100, 4, 100, 4, 100, 4, 16.0, 16.0, 127.46410000000002, 10.824100000000007, 16.0, 16.0, 16.0]

x10: [16, 100, 100, 16, 100, 16, 16, 100, 9.0, 9.0, 9.0, 9.0, 132.36502500000003, 30.305025000000008, 9.0]

x\_avg\_max = 6.9816666666666665

x\_avg\_min = -11.648333333333333

----------------------------------------------------------------------------------------------------

y\_max = 206

y\_min = 188

y1: [196, 192, 202, 196, 202, 204, 195, 202, 193, 195, 194, 192, 196, 197, 190]

y2: [197, 188, 190, 200, 192, 191, 191, 199, 189, 196, 197, 206, 203, 205, 191]

y3: [199, 201, 205, 191, 196, 197, 197, 204, 203, 205, 189, 202, 190, 191, 205]

y average: [197.33333333333334, 193.66666666666666, 199.0, 195.66666666666666, 196.66666666666666, 197.33333333333334, 194.33333333333334, 201.66666666666666, 195.0, 198.66666666666666, 193.33333333333334, 200.0, 196.33333333333334, 197.66666666666666, 195.33333333333334]

----------------------------------------------------------------------------------------------------

dispersion: [4.296296296296326, 21.518518518518515, 23.740740740740687, 35.29629629629634, 11.185185185185146, 18.51851851851846, 5.518518518518516, 25.074074074074076, 18.85185185185185, 15.62962962962966, 40.74074074074078, 63.1851851851853, 56.6296296296297, 64.18518518518526, 32.296296296296326]

Gp = 0.14698897370653105

Дисперсія однорідна

----------------------------------------------------------------------------------------------------

b: [196.71120401534088, 0.05491463477730062, 0.19772987413335222, 0.01046214021839, 0.010714285714285425, -0.03194444444444449, 0.00496031746031805, -0.0012896825396825546, 0.0031257248790564096, 0.005546441736556871, 0.008683123944796877]

y = 196.86767476670656; y avg = 197.33333333333334

y = 193.89009364710918; y avg = 193.66666666666666

y = 198.2433243550026; y avg = 199.0

y = 197.04436278323865; y avg = 195.66666666666666

y = 195.4783649927899; y avg = 196.66666666666666

y = 198.27940342102593; y avg = 197.33333333333334

y = 194.29930079558596; y avg = 194.33333333333334

y = 202.3217196759886; y avg = 201.66666666666666

y = 195.45390259937852; y avg = 195.0

y = 197.69957822446696; y avg = 198.66666666666666

y = 195.400573587757; y avg = 193.33333333333334

y = 197.41957390275496; y avg = 200.0

y = 197.2662763909348; y avg = 196.33333333333334

y = 196.2205377662436; y avg = 197.66666666666666

y = 196.11531309096424; y avg = 195.33333333333334

----------------------------------------------------------------------------------------------------

beta: [196.8, 0.6747777777777741, 0.6066666666666644, -0.3142222222222207, 0.4666666666666648, -1.000000000000002, 0.11111111111110858, -0.2888888888888914, 143.78714944444442, 143.75434444444446, 143.81995444444445]

t: [81.56065508574133, 0.27965100402874005, 0.25142342861796857, 0.13022444251494766, 0.1934026373984373, 0.4144342229966539, 0.04604824699962707, 0.11972544219903415, 59.59031555691196, 59.57672004222657, 59.60391107159737]

Коефіцієнт b1 приймаємо не значним

Коефіцієнт b2 приймаємо не значним

Коефіцієнт b3 приймаємо не значним

Коефіцієнт b4 приймаємо не значним

Коефіцієнт b5 приймаємо не значним

Коефіцієнт b6 приймаємо не значним

Коефіцієнт b7 приймаємо не значним

----------------------------------------------------------------------------------------------------

ŷ = 196.72855930590129

ŷ = 196.72855930590129

ŷ = 196.72855930590129

ŷ = 196.72855930590129

ŷ = 196.72855930590129

ŷ = 196.72855930590129

ŷ = 196.72855930590129

ŷ = 196.72855930590129

ŷ = 196.71581828855048

ŷ = 196.71581828855048

ŷ = 196.71939181129343

ŷ = 196.71939181129343

ŷ = 196.7240222599863

ŷ = 196.7240222599863

ŷ = 196.71120401534088

d = 4

Fp = 0.4014752900191738

Рівняння регресії адекватно оригіналу при рівні значимості 0