Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №4

з дисципліни МОПЕ

на тему:

# «Проведення трьохфакторного експерименту

# при використанні рівняння регресії з урахуванням ефекту взаємодії.»

Виконав:

студент групи ІВ-93

Сержан Д. О.

Варіант: 321

Перевірив:

Регіда П.Г

Київ 2021

Варіант



Код програми

**import** math  
**import** numpy **as** np  
**from** numpy.linalg **import** solve  
**from** scipy.stats **import** f, t  
**from** functools **import** partial  
**from** random **import** randint  
**from** prettytable **import** PrettyTable  
  
**while True**:  
 **def** cohren\_teor(f1, f2, q=0.05):  
 q1 = q / f1  
 fisher\_value = f.ppf(q=1 - q1, dfn=f2, dfd=(f1 - 1) \* f2)  
 **return** fisher\_value / (fisher\_value + f1 - 1)  
  
  
 fisher\_teor = partial(f.ppf, q=1 - 0.05)  
 student\_teor = partial(t.ppf, q=1 - 0.025)  
  
 X1min = 15  
 X1max = 45  
 X2min = -15  
 X2max = 45  
 X3min = 15  
 X3max = 20  
 Xmax\_average = (X1max + X2max + X3max) / 3 *# Xcp(max)* Xmin\_average = (X1min + X2min + X3min) / 3 *# Xcp(min)* y\_max = round(200 + Xmax\_average)  
 y\_min = round(200 + Xmin\_average)  
 *# матриця ПФЕ* x0\_factor = [1, 1, 1, 1, 1, 1, 1, 1]  
 x1\_factor = [-1, -1, 1, 1, -1, -1, 1, 1]  
 x2\_factor = [-1, 1, -1, 1, -1, 1, -1, 1]  
 x3\_factor = [-1, 1, 1, -1, 1, -1, -1, 1]  
 x1x2\_factor = [a \* b **for** a, b **in** zip(x1\_factor, x2\_factor)]  
 x1x3\_factor = [a \* b **for** a, b **in** zip(x1\_factor, x3\_factor)]  
 x2x3\_factor = [a \* b **for** a, b **in** zip(x2\_factor, x3\_factor)]  
 x1x2x3\_factor = [a \* b \* c **for** a, b, c **in** zip(x1\_factor, x2\_factor, x3\_factor)]  
 m = 3 *# кількість повторень кожної комбінації* y1, y2, y3 = [], [], []  
 **for** i **in** range(0, 8):  
 y1.append(randint(y\_min, y\_max)) *# заповнення у* y2.append(randint(y\_min, y\_max))  
 y3.append(randint(y\_min, y\_max))  
 Y\_row1 = [y1[0], y2[0], y3[0]] *# перший рядок у* Y\_row2 = [y1[1], y2[1], y3[1]] *# другий рядок у* Y\_row3 = [y1[2], y2[2], y3[2]] *# ...* Y\_row4 = [y1[3], y2[3], y3[3]]  
 Y\_row5 = [y1[4], y2[4], y3[4]]  
 Y\_row6 = [y1[5], y2[5], y3[5]]  
 Y\_row7 = [y1[6], y2[6], y3[6]]  
 Y\_row8 = [y1[7], y2[7], y3[7]] *# восьмий рядок у* Y\_average1 = np.average(Y\_row1) *# середній у першого рядка* Y\_average2 = np.average(Y\_row2) *# ...* Y\_average3 = np.average(Y\_row3)  
 Y\_average4 = np.average(Y\_row4)  
 Y\_average5 = np.average(Y\_row5)  
 Y\_average6 = np.average(Y\_row6)  
 Y\_average7 = np.average(Y\_row7)  
 Y\_average8 = np.average(Y\_row8) *# середній у восьмого рядка* Y\_average = [round(Y\_average1, 3), round(Y\_average2, 3), round(Y\_average3, 3), round(Y\_average4, 3),  
 *# округлення до* round(Y\_average5, 3), round(Y\_average6, 3), round(Y\_average7, 3), round(Y\_average8, 3)] *# тисячних* x0 = [1, 1, 1, 1, 1, 1, 1, 1]  
 x1 = [15, 15, 45, 45, 15, 15, 45, 45] *# заміна -1 на x1(min) 1 на x1(max)* x2 = [-25, 10, -25, 10, -25, 10, -25, 10] *# заміна -1 на x2(min) 1 на x2(max)* x3 = [-45, 50, 50, -45, 50, -45, -45, 50] *# заміна -1 на x3(min) 1 на x3(max)* x1x2 = [a \* b **for** a, b **in** zip(x1, x2)]  
 x1x3 = [a \* b **for** a, b **in** zip(x1, x3)]  
 x2x3 = [a \* b **for** a, b **in** zip(x2, x3)]  
 x1x2x3 = [a \* b \* c **for** a, b, c **in** zip(x1, x2, x3)]  
 list\_for\_solve\_b = [x0\_factor, x1\_factor, x2\_factor, x3\_factor, x1x2\_factor, x1x3\_factor, x2x3\_factor,  
 x1x2x3\_factor]  
 list\_for\_solve\_a = list(zip(x0, x1, x2, x3, x1x2, x1x3, x2x3, x1x2x3))  
 N = 8 *# кількість повторення дослідів* list\_bi = [] *# b(i)* **for** k **in** range(N):  
 S = 0  
 **for** i **in** range(N):  
 S += (list\_for\_solve\_b[k][i] \* Y\_average[i]) / N  
 list\_bi.append(round(S, 5))  
 Disp1 = 0 *# дисперсії* Disp2 = 0  
 Disp3 = 0  
 Disp4 = 0  
 Disp5 = 0  
 Disp6 = 0  
 Disp7 = 0  
 Disp8 = 0  
 **for** i **in** range(m):  
 Disp1 += ((Y\_row1[i] - np.average(Y\_row1)) \*\* 2) / m *# ((y-ycp)^2)/3* Disp2 += ((Y\_row2[i] - np.average(Y\_row2)) \*\* 2) / m  
 Disp3 += ((Y\_row3[i] - np.average(Y\_row3)) \*\* 2) / m  
 Disp4 += ((Y\_row4[i] - np.average(Y\_row4)) \*\* 2) / m  
 Disp5 += ((Y\_row5[i] - np.average(Y\_row5)) \*\* 2) / m  
 Disp6 += ((Y\_row6[i] - np.average(Y\_row6)) \*\* 2) / m  
 Disp7 += ((Y\_row7[i] - np.average(Y\_row7)) \*\* 2) / m  
 Disp8 += ((Y\_row8[i] - np.average(Y\_row8)) \*\* 2) / m  
 sum\_dispersion = Disp1 + Disp2 + Disp3 + Disp4 + Disp5 + Disp6 + Disp7 + Disp8 *# сума дисперсій* disp\_list = [round(Disp1, 3), round(Disp2, 3), round(Disp3, 3), round(Disp4, 3), round(Disp5, 3), round(Disp6, 3),  
 round(Disp7, 3), round(Disp8, 3)]  
 pt1 = PrettyTable() *# формування таблиці* column\_names1 = [**"X0"**, **"X1"**, **"X2"**, **"X3"**, **"X1X2"**, **"X1X3"**, **"X2X3"**, **"X1X2X3"**, **"Y1"**, **"Y2"**, **"Y3"**, **"Y"**, **"S^2"**] *# назви* pt1.add\_column(column\_names1[0], x0\_factor) *# запис назв* pt1.add\_column(column\_names1[1], x1\_factor)  
 pt1.add\_column(column\_names1[2], x2\_factor)  
 pt1.add\_column(column\_names1[3], x3\_factor)  
 pt1.add\_column(column\_names1[4], x1x2\_factor)  
 pt1.add\_column(column\_names1[5], x1x3\_factor)  
 pt1.add\_column(column\_names1[6], x2x3\_factor)  
 pt1.add\_column(column\_names1[7], x1x2x3\_factor)  
 pt1.add\_column(column\_names1[8], y1)  
 pt1.add\_column(column\_names1[9], y2)  
 pt1.add\_column(column\_names1[10], y3)  
 pt1.add\_column(column\_names1[11], Y\_average)  
 pt1.add\_column(column\_names1[12], disp\_list)  
 print(pt1, **"\n"**)  
 *# рівняння регресії з ефектом взаємодії* print(**"y = {} + {}\*x1 + {}\*x2 + {}\*x3 + {}\*x1x2 + {}\*x1x3 + {}\*x2x3 + {}\*x1x2x3 \n"**.format(list\_bi[0], list\_bi[1],  
 list\_bi[2], list\_bi[3],  
 list\_bi[4], list\_bi[5],  
 list\_bi[6], list\_bi[7]))  
 pt2 = PrettyTable() *# створення таблиці* pt2.add\_column(column\_names1[0], x0)  
 pt2.add\_column(column\_names1[1], x1)  
 pt2.add\_column(column\_names1[2], x2)  
 pt2.add\_column(column\_names1[3], x3)  
 pt2.add\_column(column\_names1[4], x1x2)  
 pt2.add\_column(column\_names1[5], x1x3)  
 pt2.add\_column(column\_names1[6], x2x3)  
 pt2.add\_column(column\_names1[7], x1x2x3)  
 pt2.add\_column(column\_names1[8], y1)  
 pt2.add\_column(column\_names1[9], y2)  
 pt2.add\_column(column\_names1[10], y3)  
 pt2.add\_column(column\_names1[11], Y\_average)  
 pt2.add\_column(column\_names1[12], disp\_list)  
 print(pt2, **'\n'**)  
 list\_ai = [round(i, 5) **for** i **in** solve(list\_for\_solve\_a, Y\_average)]  
 print(**"y = {} + {}\*x1 + {}\*x2 + {}\*x3 + {}\*x1x2 + {}\*x1x3 + {}\*x2x3 + {}\*x1x2x3"**.format(list\_ai[0], list\_ai[1],  
 list\_ai[2], list\_ai[3],  
 list\_ai[4], list\_ai[5],  
 list\_ai[6], list\_ai[7]))  
 Gp = max(Disp1, Disp2, Disp3, Disp4, Disp5, Disp6, Disp7, Disp8) / sum\_dispersion *# експерементальне* F1 = m - 1  
 N = len(y1)  
 F2 = N  
 Gt = cohren\_teor(F1, F2) *# теоретичне* **def** cohren(g\_prac, g\_teor):  
 **return** g\_prac < g\_teor  
  
  
 print(**"\nGp = "**, Gp, **" Gt = "**, Gt)  
 **if** cohren(Gp, Gt):  
 print(**"Дисперсія однорідна!\n"**)  
  
 Dispersion\_B = sum\_dispersion / N  
 Dispersion\_beta = Dispersion\_B / (m \* N)  
 S\_beta = math.sqrt(abs(Dispersion\_beta))  
 beta0 = 0  
 beta1 = 0  
 beta2 = 0  
 beta3 = 0  
 beta4 = 0  
 beta5 = 0  
 beta6 = 0  
 beta7 = 0  
 **for** i **in** range(len(x0\_factor)):  
 beta0 += (Y\_average[i] \* x0\_factor[i]) / N  
 beta1 += (Y\_average[i] \* x1\_factor[i]) / N  
 beta2 += (Y\_average[i] \* x2\_factor[i]) / N  
 beta3 += (Y\_average[i] \* x3\_factor[i]) / N  
 beta4 += (Y\_average[i] \* x1x2\_factor[i]) / N  
 beta5 += (Y\_average[i] \* x1x3\_factor[i]) / N  
 beta6 += (Y\_average[i] \* x2x3\_factor[i]) / N  
 beta7 += (Y\_average[i] \* x1x2x3\_factor[i]) / N  
 beta\_list = [beta0, beta1, beta2, beta3, beta4, beta5, beta6, beta7]  
 t0 = abs(beta0) / S\_beta  
 t1 = abs(beta1) / S\_beta  
 t2 = abs(beta2) / S\_beta  
 t3 = abs(beta3) / S\_beta  
 t4 = abs(beta4) / S\_beta  
 t5 = abs(beta5) / S\_beta  
 t6 = abs(beta6) / S\_beta  
 t7 = abs(beta7) / S\_beta  
 t\_list = [t0, t1, t2, t3, t4, t5, t6, t7]  
  
 F3 = F1 \* F2  
 d = 0  
 T = student\_teor(df=F3)  
  
  
 **def** student(t\_teor, t\_pr):  
 **return** t\_pr < t\_teor  
  
  
 print(**"t табличне = "**, T)  
 **for** i **in** range(len(t\_list)):  
 **if** student(t\_list[i], T):  
 beta\_list[i] = 0  
 print(**"Гіпотеза підтверджена, beta{} = 0"**.format(i))  
 **else**:  
 print(**"Гіпотеза не підтверджена.\nbeta{} = {}"**.format(i, beta\_list[i]))  
 d += 1  
 y\_1 = beta\_list[0] + beta\_list[1] \* x1[0] + beta\_list[2] \* x2[0] + beta\_list[3] \* x3[0] + beta\_list[4] \* x1x2[0] \  
 + beta\_list[5] \* x1x3[0] + beta\_list[6] \* x2x3[0] + beta\_list[7] \* x1x2x3[0]  
 y\_2 = beta\_list[0] + beta\_list[1] \* x1[1] + beta\_list[2] \* x2[1] + beta\_list[3] \* x3[1] + beta\_list[4] \* x1x2[1] \  
 + beta\_list[5] \* x1x3[1] + beta\_list[6] \* x2x3[1] + beta\_list[7] \* x1x2x3[1]  
 y\_3 = beta\_list[0] + beta\_list[1] \* x1[2] + beta\_list[2] \* x2[2] + beta\_list[3] \* x3[2] + beta\_list[4] \* x1x2[2] \  
 + beta\_list[5] \* x1x3[2] + beta\_list[6] \* x2x3[2] + beta\_list[7] \* x1x2x3[2]  
 y\_4 = beta\_list[0] + beta\_list[1] \* x1[3] + beta\_list[2] \* x2[3] + beta\_list[3] \* x3[3] + beta\_list[4] \* x1x2[3] \  
 + beta\_list[5] \* x1x3[3] + beta\_list[6] \* x2x3[3] + beta\_list[7] \* x1x2x3[3]  
 y\_5 = beta\_list[0] + beta\_list[1] \* x1[4] + beta\_list[2] \* x2[4] + beta\_list[3] \* x3[4] + beta\_list[4] \* x1x2[4] \  
 + beta\_list[5] \* x1x3[4] + beta\_list[6] \* x2x3[4] + beta\_list[7] \* x1x2x3[4]  
 y\_6 = beta\_list[0] + beta\_list[1] \* x1[5] + beta\_list[2] \* x2[5] + beta\_list[3] \* x3[5] + beta\_list[4] \* x1x2[5] \  
 + beta\_list[5] \* x1x3[5] + beta\_list[6] \* x2x3[5] + beta\_list[7] \* x1x2x3[5]  
 y\_7 = beta\_list[0] + beta\_list[1] \* x1[6] + beta\_list[2] \* x2[6] + beta\_list[3] \* x3[6] + beta\_list[4] \* x1x2[6] \  
 + beta\_list[5] \* x1x3[6] + beta\_list[6] \* x2x3[6] + beta\_list[7] \* x1x2x3[6]  
 y\_8 = beta\_list[0] + beta\_list[1] \* x1[7] + beta\_list[2] \* x2[7] + beta\_list[3] \* x3[7] + beta\_list[4] \* x1x2[7] \  
 + beta\_list[5] \* x1x3[7] + beta\_list[6] \* x2x3[7] + beta\_list[7] \* x1x2x3[7]  
 Y\_counted\_for\_Student = [y\_1, y\_2, y\_3, y\_4, y\_5, y\_6, y\_7, y\_8]  
 F4 = N - d  
 Dispersion\_ad = 0  
 **for** i **in** range(len(Y\_counted\_for\_Student)):  
 Dispersion\_ad += ((Y\_counted\_for\_Student[i] - Y\_average[i]) \*\* 2) \* m / (N - d)  
 Fp = Dispersion\_ad / Dispersion\_beta  
 Ft = fisher\_teor(dfn=F4, dfd=F3)  
  
  
 **def** fisher(f\_teor, f\_prac):  
 **return** f\_teor > f\_prac  
  
  
 **if** fisher(Ft, Fp):  
 print(**"Рівняння регресії адекватне!"**)  
 **break  
 else**:  
 print(**"Рівняння регресії неадекватне."**)  
 **break  
  
 else**:  
 print(**"Дисперсія неоднорідна. Спробуйте ще раз."**)  
 m += 1

Результат виконання

