**3-amaliy mashg’ulot**

**13.** Требуется построить модель зависимости значения выходного тока от изменения входного сопротивления четырехполюсника и оценить ее.

***i***

**U**

**C**

**R**

Сведения о входном сопротивлении и выходном токе четырехполюсника по выборочным данным приведены в таблице.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Номер сигнала  Показатели | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Сопротивления, Ом | 10 | 20 | 40 | 50 | 80 | 100 | 130 | 180 | 210 | 260 |
| Выходной ток, а | 11,2 | 10,3 | 8,8 | 5,8 | 4,6 | 3,4 | 2,6 | 2,9 | 1,3 | 3,2 |

Продолжение таблицы.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Номер сигнала  Показатели | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| Сопротивления, Ом | 320 | 380 | 450 | 500 | 600 | 700 | 800 | 850 | 900 | 950 |
| Выходной ток, а | 0,9 | 0,8 | 0,7 | 0,6 | 0,5 | 0,40 | 0,36 | 0,32 | 0,31 | 0,30 |

import numpy as np  
from matplotlib import pyplot as plt  
from numpy.polynomial.polynomial import Polynomial  
  
x = np.array([10, 20, 40, 50, 80, 100, 130, 180, 210, 260, 320, 380, 450, 500, 600, 700, 800, 850, 900, 950])  
y = np.array([11.2, 10.3, 8.8, 5.8, 4.6, 3.4, 2.6, 2.9, 1.3, 3.2, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.36, 0.32, 0.31, 0.3])  
n = len(x)  
  
x\_mean = np.mean(x)  
y\_mean = np.mean(y)  
  
Sx2 = np.sum(x \*\* 2) / (n - 1) - (n \* (x\_mean \*\* 2)) / (n - 1)  
Sy2 = np.sum(y \*\* 2) / (n - 1) - (n \* (y\_mean \*\* 2)) / (n - 1)  
  
sigma\_x = np.sqrt(np.sum((x - x\_mean) \*\* 2) / n)  
sigma\_y = np.sqrt(np.sum((y - y\_mean) \*\* 2) / n)  
  
Sx = np.mean(np.abs(x - x\_mean))  
Sy = np.mean(np.abs(y - y\_mean))  
Cx = (sigma\_x / Sx) \* 100  
Cy = (sigma\_y / Sy) \* 100  
  
numerator = n \* np.sum(x \* y) - np.sum(x) \* np.sum(y)  
denominator = np.sqrt((n \* np.sum(x \*\* 2) - (np.sum(x)) \*\* 2) \* (n \* np.sum(y \*\* 2) - (np.sum(y)) \*\* 2))  
Ryx = numerator / denominator  
  
degree = 2  
  
poly\_model = Polynomial.fit(x, y, deg=degree)  
  
coefficients = poly\_model.convert().coef  
  
equation\_terms = []  
for i, coeff in enumerate(coefficients):  
 sign = "+" if coeff >= 0 else "-"  
 equation\_terms.append(f"{sign} {abs(round(coeff, 5))} \* x^{i}")  
  
equation = "y = " + " ".join(equation\_terms).replace("+ -", "- ")  
  
y\_pred = poly\_model(x)  
  
differences = np.abs(y - y\_pred)  
closest\_indices = np.argsort(differences)[:3]  
  
plt.figure(figsize=(8, 6))  
plt.scatter(x, y, color='blue', label="Asosiy ma'lumotlar")  
plt.plot(x, y\_pred, color='red', label=f"Polynomial ({degree}-daraja) regressiya")  
plt.scatter(x[closest\_indices], y[closest\_indices], color='green', s=100, label="Eng yaqin 3 nuqta")  
plt.xlabel("X o'qi")  
plt.ylabel("Y o'qi")  
plt.title(f"Polynomial Regression (daraja={degree})")  
plt.legend()  
plt.grid()  
  
print("Korelyatsiya koeffitsiyenti:", Ryx)  
print("Polynomial Regressiya tenglamasi:", equation)  
print("Eng yaqin 3 ta nuqta:")  
for i in closest\_indices:  
 print(f"x = {x[i]}, y = {y[i]}, bashorat = {round(y\_pred[i], 3)}, farq = {round(differences[i], 3)}")  
  
plt.show()

