**2-amaliyot ishi**

**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  
import matplotlib.pyplot as plt  
from scipy.stats import norm  
from numpy.polynomial.polynomial import Polynomial  
from sklearn.linear\_model import LinearRegression  
  
# Berilgan ma'lumotlar  
x\_values = np.array([10, 20, 40, 50, 80, 100, 130, 180, 210, 260, 320, 380, 450, 500, 600, 700, 800, 850, 900, 950])  
y\_values = 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])  
  
degree = 2  
poly\_model = Polynomial.fit(x\_values, y\_values, deg=degree)  
y\_poly\_pred = poly\_model(x\_values)  
  
# Polynomial koeffitsiyentlarini olish  
coefficients = poly\_model.convert().coef  
equation\_terms = [f"{'+' if coeff >= 0 else '-'} {abs(round(coeff, 5))} \* x^{i}" for i, coeff in  
 enumerate(coefficients)]  
equation\_poly = "y = " + " ".join(equation\_terms).replace("+ -", "- ")  
  
# Chiziqli regressiya modeli  
x\_values\_reshaped = x\_values.reshape(-1, 1)  
lin\_reg = LinearRegression()  
lin\_reg.fit(x\_values\_reshaped, y\_values)  
y\_lin\_pred = lin\_reg.predict(x\_values\_reshaped)  
  
# Chiziqli regressiya tenglamasi  
equation\_lin = f"y = {lin\_reg.coef\_[0]:.4f}x + {lin\_reg.intercept\_:.4f}"  
  
# 📌 Eng yaqin nuqtalarni topish  
diff\_poly = np.abs(y\_values - y\_poly\_pred)  
closest\_poly = x\_values[np.argsort(diff\_poly)[:5]], y\_values[np.argsort(diff\_poly)[:5]]  
  
diff\_lin = np.abs(y\_values - y\_lin\_pred)  
closest\_lin = x\_values[np.argsort(diff\_lin)[:5]], y\_values[np.argsort(diff\_lin)[:5]]  
  
# 📌 Gauss o‘rtacha qiymatiga eng yaqin 5 ta nuqtani topish  
mu, sigma = np.mean(y\_values), np.std(y\_values)  
diff\_gauss = np.abs(y\_values - mu)  
closest\_gauss = x\_values[np.argsort(diff\_gauss)[:5]], y\_values[np.argsort(diff\_gauss)[:5]]  
  
# В.И. Романовский kriteriyasi bo‘yicha R ni hisoblash  
B = len(x\_values) - 3  
R = np.sum((x\_values + y\_values) \*\* 2 - B) / (np.sum(y\_values) \* np.sqrt(2 \* B)) if B > 0 else None  
  
# 📌 Grafikni chizish  
plt.figure(figsize=(10, 5))  
  
# Regressiya grafikasi  
plt.subplot(1, 2, 1)  
plt.scatter(x\_values, y\_values, color='blue', label="Asosiy ma'lumotlar")  
plt.plot(x\_values, y\_poly\_pred, color='red', label=f"Polynomial ({degree}-daraja) regressiya")  
plt.plot(x\_values, y\_lin\_pred, color='orange', linestyle="dashed", label="Chiziqli regressiya")  
plt.scatter(closest\_poly[0], closest\_poly[1], color='green', s=100, label="Eng yaqin (Poly)")  
plt.scatter(closest\_lin[0], closest\_lin[1], color='purple', s=100, label="Eng yaqin (Linear)")  
plt.xlabel("Kirish signali (x)")  
plt.ylabel("Chiqish signali (y)")  
plt.legend()  
plt.title("Polynomial & Chiziqli Regression")  
  
# 📌 Gauss taqsimoti grafikasi  
gauss\_x = np.linspace(min(y\_values), max(y\_values), 100)  
gauss\_y = norm.pdf(gauss\_x, mu, sigma)  
  
plt.subplot(1, 2, 2)  
plt.plot(gauss\_x, gauss\_y, color='green', label="Gauss taqsimoti")  
plt.axvline(mu, color='red', linestyle='dashed', label=f"O'rtacha: {mu:.2f}")  
  
# 📌 Gaussga eng yaqin 5 ta nuqtani chizish  
for y in closest\_gauss[1]:  
 plt.axvline(y, color='purple', linestyle='dotted', alpha=0.8)  
  
plt.scatter(closest\_gauss[1], np.zeros\_like(closest\_gauss[1]), color='purple', s=100, label="Gaussga yaqin 5 nuqta")  
plt.xlabel("Chiqish signali (y)")  
plt.ylabel("Ehtimollik zichligi")  
plt.legend()  
plt.title("Gauss taqsimoti")  
  
plt.tight\_layout()  
plt.show()  
  
# 📌 Natijalarni chiqarish  
print("Polynomial Regression tenglamasi:", equation\_poly)  
print("Chiziqli Regression tenglamasi:", equation\_lin)  
print(f"B = {B}, R = {R if R is not None else 'Nomalum'}")  
  
print("\n📌 Polynomial Regressiya chizig‘iga eng yaqin nuqtalar (x, y):")  
for x, y in zip(\*closest\_poly):  
 print(f"({x}, {y})")  
  
print("\n📌 Chiziqli Regressiya chizig‘iga eng yaqin nuqtalar (x, y):")  
for x, y in zip(\*closest\_lin):  
 print(f"({x}, {y})")  
  
print("\n📌 Gauss o‘rtacha qiymatiga eng yaqin nuqtalar (x, y):")  
for x, y in zip(\*closest\_gauss):  
 print(f"({x}, {y})")

