Задание 1. Проверять, может ли полином вида

$$f(z) = a_0 + a_1 z^{n_1} + a_2 z^{n_2}$$

аппроксимировать функции температуры.

Решение

❖ **Алгоритм**

1. Заданы n_1 и n_2

2. Обычным методом находим коэффициенты a_0, a_1, a_2

Используя метод наименьших квадратов:

$$\phi(a_0, a_1, a_2) = \sum_i \left(y_i - f(z_i)\right)^2 \rightarrow min$$

$$\begin{cases} \dfrac{\partial \phi}{\partial a_0} = -2\sum_i \left(y_i - a_0 - a_1 z_i^{n_1} - a_2 z_i^{n_2}\right) = 0 \\ \dfrac{\partial \phi}{\partial a_1} = -2\sum_i \left(y_i - a_0 - a_1 z_i^{n_1} - a_2 z_i^{n_2}\right) z_i^{n_1} = 0 \\ \dfrac{\partial \phi}{\partial a_2} = -2\sum_i \left(y_i - a_0 - a_1 z_i^{n_1} - a_2 z_i^{n_2}\right) z_i^{n_2} = 0 \end{cases}$$

$$\begin{cases} m a_0 + a_1 \sum_i z_i^{n_1} + a_2 \sum_i z_i^{n_2} = \sum_i y_i \\ a_0 \sum_i z_i^{n_1} + a_1 \sum_i z_i^{2n_1} + a_2 \sum_i z_i^{n_1+n_2} = \sum_i y_i z_i^{n_1} \\ a_0 \sum_i z_i^{n_2} + a_1 \sum_i z_i^{n_1+n_2} + a_2 \sum_i z_i^{2n_2} = \sum_i y_i z_i^{n_2} \end{cases}$$

Данная система уравнений можно записывается в следующим матричном виде

$$MA = B$$

$$M = \begin{pmatrix} m & \sum_i z_i^{n_1} & \sum_i z_i^{n_2} \\ \sum_i z_i^{n_1} & \sum_i z_i^{2n_1} & \sum_i z_i^{n_1+n_2} \\ \sum_i z_i^{n_2} & \sum_i z_i^{n_1+n_2} & \sum_i z_i^{2n_2} \end{pmatrix}$$

$$A = (a_0 \quad a_1 \quad a_2)^T$$

$$B = \left(\sum_i y_i \quad \sum_i y_i z_i^{n_1} \quad \sum_i y_i z_i^{n_2} \right)^T$$

Решением этого уравнения является: $A = M^{-1}B$

❖ **Код программы** (*Python*)

```
# Check function: P(z) = a0 + a1.z^n1 + a2.z^n2

from math import *

import numpy as np

import matplotlib.pyplot as plt


def f(z, a0, a1, n1, a2, n2):

        return a0 + a1 * pow(z, n1) + a2 * pow(z, n2)


# TEST DATA

x = [ 0.05,  0.10,  0.15,  0.20,  0.25,  0.30,  0.35,  0.40,  0.45,  0.50,  0.55,  0.60,  0.65,  0.70,
0.75, 0.80, 0.85, 0.90, 0.95, 1.00]

y = [30000, 29540, 28705, 28056, 27203, 25899, 23181, 20000, 19305, 18990, 18392, 17530, 16900, 13593,
11999, 9312, 6812, 4012, 3500, 3000]

x = np.asarray(x)

y = np.asarray(y)


# Settings graphic

plt.xlabel("X")

plt.ylabel("Y")

plt.grid(True)


# Draw source data

plt.scatter(x, y)

plt.plot(x, y)

# test symlog

# plt.plot(x, y - y.mean())

# plt.yscale('symlog', linthreshy=0.01)


# Setup n1, n2

n2 = 50.0

d = 100.0

n1 = n2 - d
```

```
err = 1000000000
newY = []
_N1 = -40.0
_N2 = 40.0

for i in range(0, int(d * 10) - 1):
        n1 = n1 + 0.1

        # Using zip to calculate sum of arrays
        m_11 = len(x)
        m_12 = np.sum(np.power(x, n1))
        m_13 = np.sum(np.power(x, n2))

        m_21 = m_12
        m_22 = np.sum(np.power(x, 2 * n1))
        m_23 = np.sum(np.power(x, n1 + n2))

        m_31 = m_13
        m_32 = m_23
        m_33 = np.sum(np.power(x, 2 * n2))

        b_1 = np.sum(y)
        b_2 = np.sum(ty * pow(tx, n1) for ty, tx in zip(y, x))
        b_3 = np.sum(ty * pow(tx, n2) for ty, tx in zip(y, x))

        M = np.array([[m_11, m_12, m_13],
                              [m_21, m_22, m_23],
                              [m_31, m_32, m_33]])

        B = np.array([b_1, b_2, b_3])

        # Solution
        A = np.linalg.solve(M, B)

        # Build up our figure
        vect = np.vectorize(f)
        new_err = np.sum(pow(ty - tz, 2.0) for ty, tz in zip(y, vect(x, A[0], A[1], n1, A[2], n2)))
        if new_err < err:
                err = new_err
```

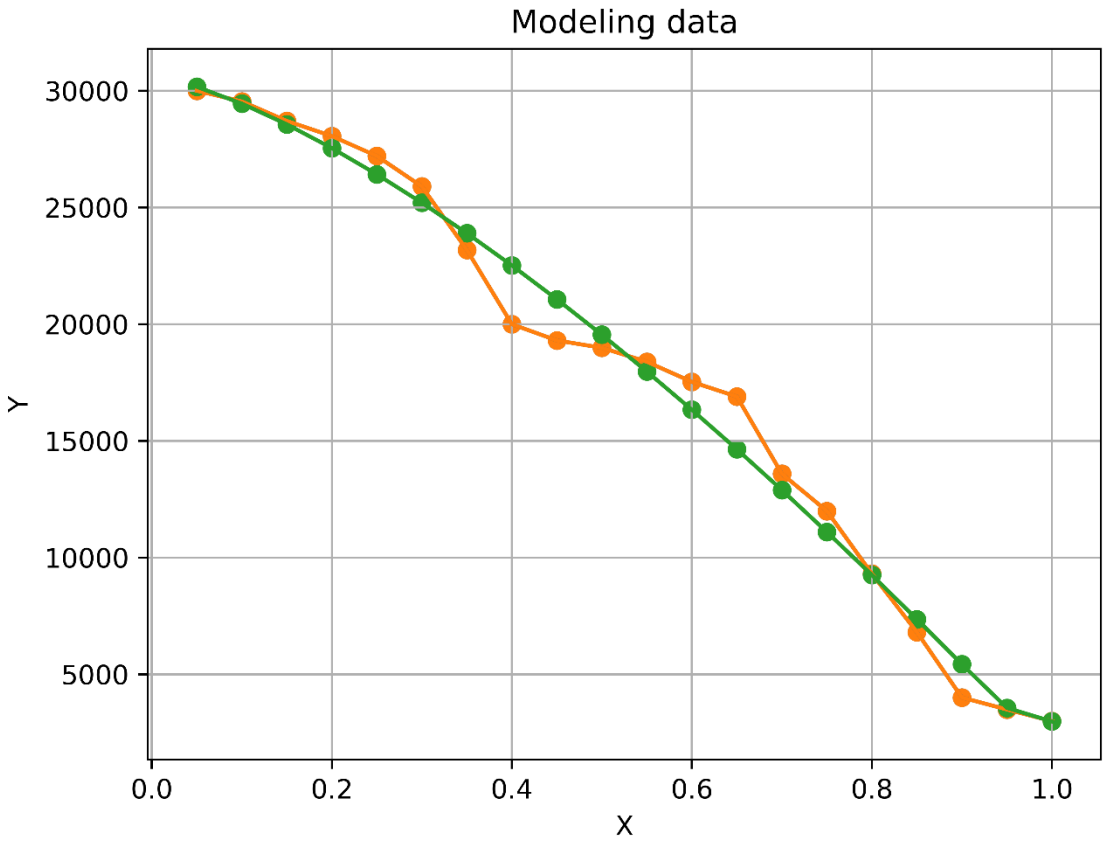
```
                newY = vect(x, A[0], A[1], n1, A[2], n2)
                _N1 = n1
                _N2 = n2

        # print("N1 = ", n1, '\tN2 = ', n2, '\terror: ', new_err)

# Draw source data
print("Output: N1 = ", _N1, '\tN2 = ', _N2, '\terror: ', err)

plt.title("Modeling data")
plt.scatter(x, y)
plt.plot(x, y)

plt.scatter(x, newY)
plt.plot(x, newY)

# plt.show()
plt.savefig('03.png', dpi=720)
```

❖ Эксперименты

1) Замечание на рисуках:
   - Линия оранжевая – данная;
   - Линия зеленая – аппроксимированная;
2) **Вывод: полученная линия не имеет хорошее качество аппроксимации.**

1) **Экс. №1**

	n_1	n_2
Вход	-50	50
Выход	1.4	50

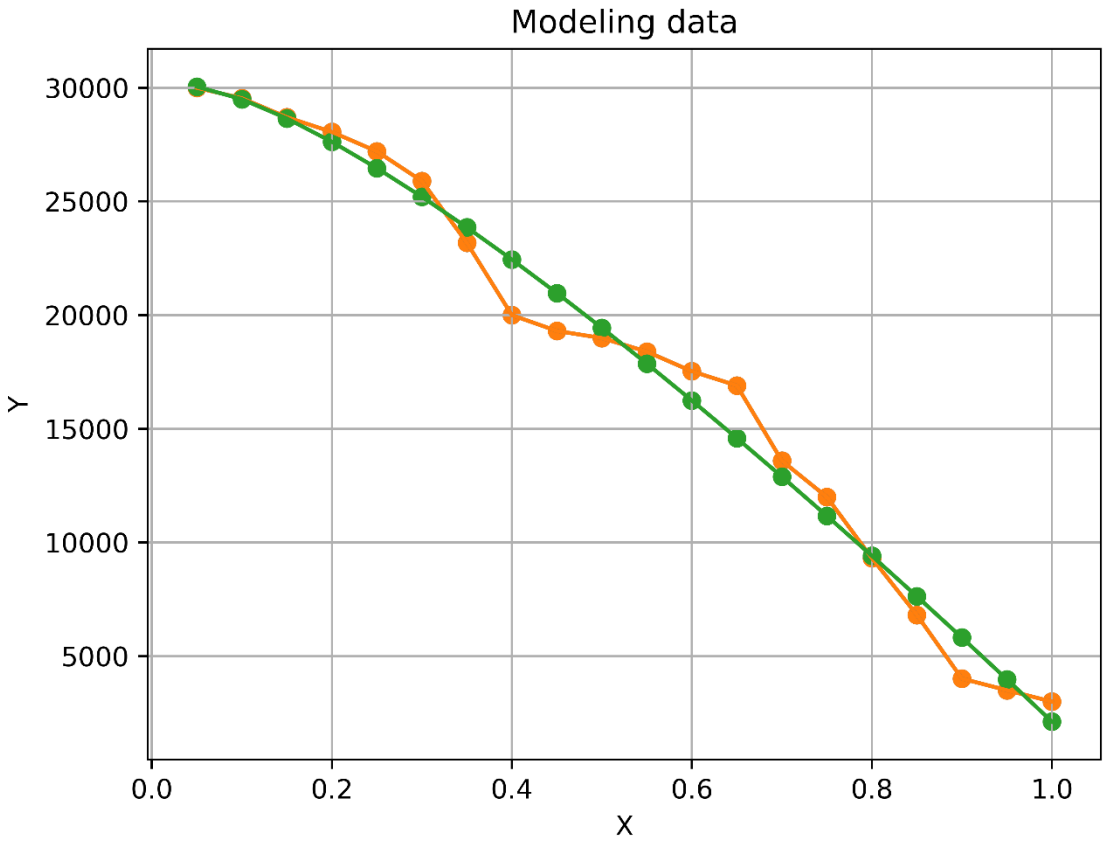


Вывод: качество аппроксимации не очень хорошо!

На картинке количество аппроксимировано совпадающих точек – 6 / 20.

2) **Экс. №2**

	n_1	n_2
Вход	-49.0	1.0
Выход	0.9	1.0

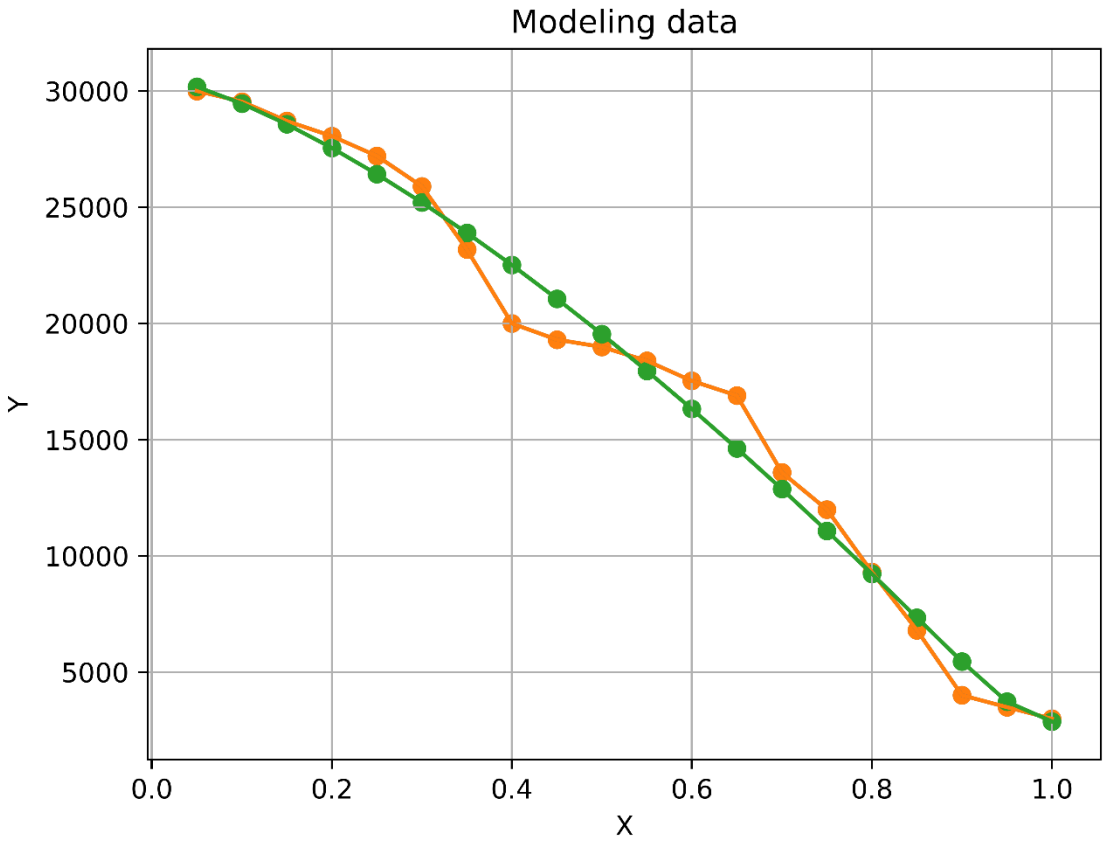


Вывод: качество аппроксимации не очень хорошо!

На картинке количество аппроксимировано совпадающих точек – 4 / 20.

3) Экс. №3

	n_1	n_2
Вход	-70.0	30.0
Выход	1.4	30.0

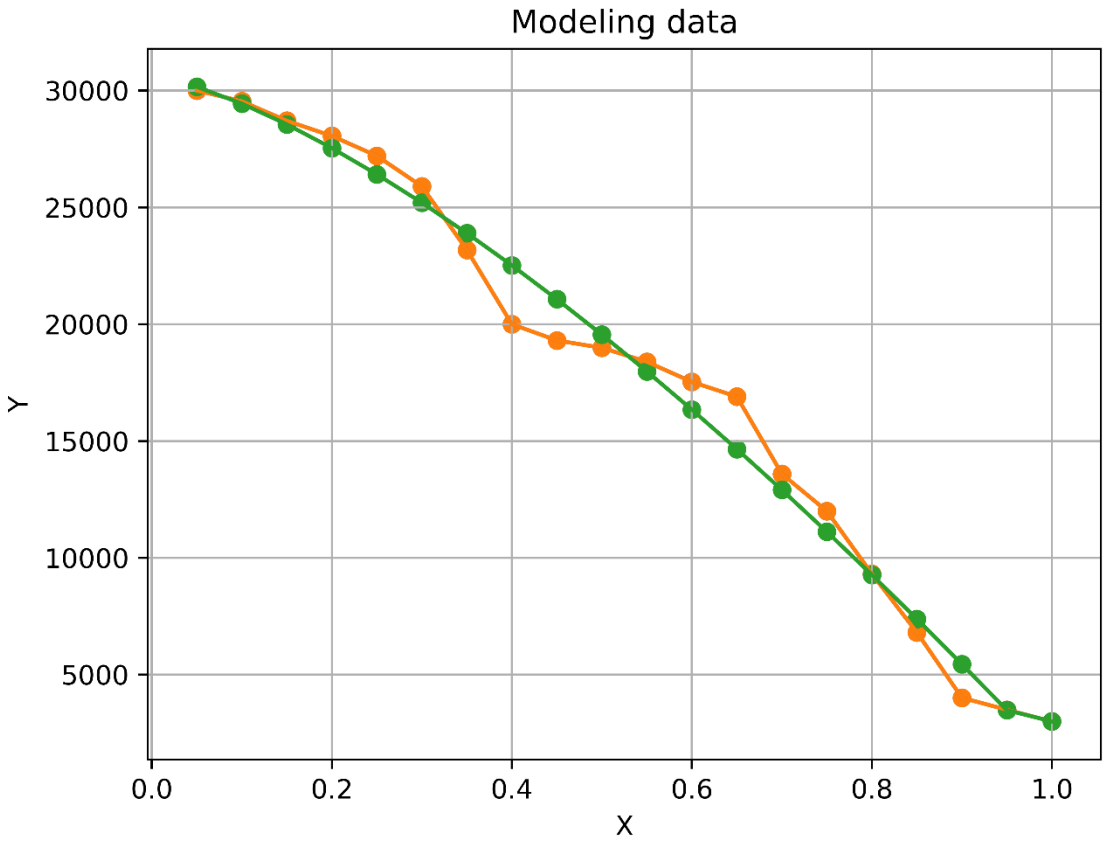


Вывод: качество аппроксимации не очень хорошо!

На картинке количество аппроксимировано совпадающих точек – 5 / 20.

4) Экс. №4

	n_1	n_2
Вход	-100.0	100.0
Выход	1.39	100.0



Вывод: качество аппроксимации не очень хорошо!

На картинке количество аппроксимировано совпадающих точек – 6 / 20.