Київський національний університет імені Тараса
Шевченка
Факультет комп’ютерних наук і кібернетики

Звіт
з лабораторної роботи №1
з моделювання складних систем

Виконав:
Студент групи ІПС-32
Левицький Іван Олегович

Київ
2025

Варіант 8

Постановка Задачі:

Визначити модель в класі функцій:

$$y(t) = a_1 t^3 + a_2 t^2 + a_3 t + \sum_{i=4}^{k} a_i \sin(2\pi f_{i-3} t) + a_{k+1}$$

Для спостережуваної дискретної функції y(ti), i = 1,2,...,N,(відповідний файл f8.txt), ti+1 − ti = Δt = 0.01, інтервал спостереження [0,T], T = 5

1) Теоретична частина

Для цього нам потрібно виконати дискретне перетворення Фур'є для дискретної послідовності y(tm), m = 0,1,2,...,N − 1.

$$c_x(k) = \frac{1}{N} \sum_{m=0}^{N-1} x(m) e^{-i 2\pi k m / N}$$

Далі, ми визначаємо частоти з найбільшим вкладом в дискретне перетворення Фур'є. Для цього беремо момент де модуль набуває найбільшого значення(тобто локальний екстремум). І тоді виконаємо множення k * Δf = k * /T (де k* - локальні екстремуми, а f* - частоти з найбільшим вкладом).

Знайшовши все необхідне можна перейти до визначення невідомих параметрів ai, i = k + 1, де будемо застосовувати метод найменших квадратів. Для цього записуємо функціонал похибки

$$F(a_1, a_2, \ldots, a_{k+1}) = \frac{1}{2} \sum_{j=0}^{N-1} (a_1 t_j^3 + a_2 t_j^2 + a_3 t_j + \sum_{i=4}^{k} a_i sin(2\pi f_{i-3} t_j) + a_{k+1} - y(t_j))^2$$

ai, i = 1,2,...,k + 1 - беремо з умови

$$F(a_1, a_2, \ldots, a_{k+1}) -> \min_{a_1, a_2, \ldots, a_{k+1}}$$

Й тепер записуємо систему рівнянь:

$$\frac{\partial F(a_1, a_2, \ldots, a_{k+1})}{\partial a_j} = 0$$

Ця система є системою лінійних алгебраїчних рівнянь. Розв'язавши цю систему одним із відомих методів, знаходимо ai, i = 1,2,...,k + 1

2) Виконання:

Реалізовувати завдання будемо мовою програмування Python.

1) Для початку ініціалізуємо всі необхідні параметри:

```
observations = np.loadtxt('f8.txt')

T = 5
N = len(observations)
dt = 0.01
t = np.arange(0, N*dt, dt)
```

2) Потім виконуємо дискретне перетворення Фур'є, та визначаємо його за модулем

```
# 1. Discrete Fourier transform
f_transform = np.fft.fft(observations) / N
frequencies = np.fft.fftfreq(N, dt)

#abs Fourier transform
f_magnitude = np.abs(f_transform)
```

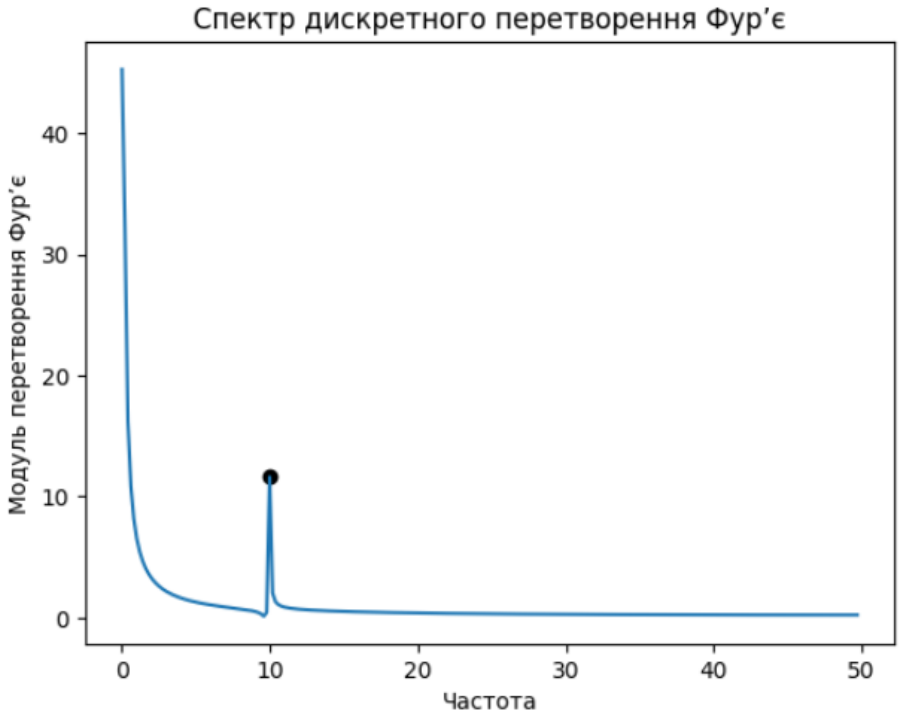
3) Тепер виводимо локальні максимуми(суттєві частоти), і потрібно зазначити що :N //2 - береться для того щоб визначити локальні максимуми лише для лівої частини тому що у дискретномуперетворенні Фур'є спектр є симетричним відносно нуля(дзеркальним) тому беремо лише одну частину.

```
# Local maxima
peaks, _ = find_peaks(f_magnitude[:N // 2])
peak_frequencies = frequencies[peaks]

print("Суттєві частоти:", peak_frequencies)
```

4) Виконуємо побудову графіку, і (Не забуваємо про :N // 2):

```
# 2. Building a graph of the Fourier transform module
plt.figure()
plt.plot(frequencies[:N // 2], f_magnitude[:N // 2])
plt.scatter(peak_frequencies, f_magnitude[peaks], color='gren')
plt.xlabel('Частота')
plt.ylabel('Модуль перетворення Фур'є')
plt.title('Спектр дискретного перетворення Фур'є')
plt.show()
```

```
Суттєві частоти: [9.98003992]
```




```
Знайдені параметри: [ 2.00151029 -1.21105126 -2.98421085  1.00041963  0.29103001]
```

В результаті отримали 1 значущий вклад частоти 9.98 Гц

5) Тепер приступаємо до визначення параметрів найменших квадратів

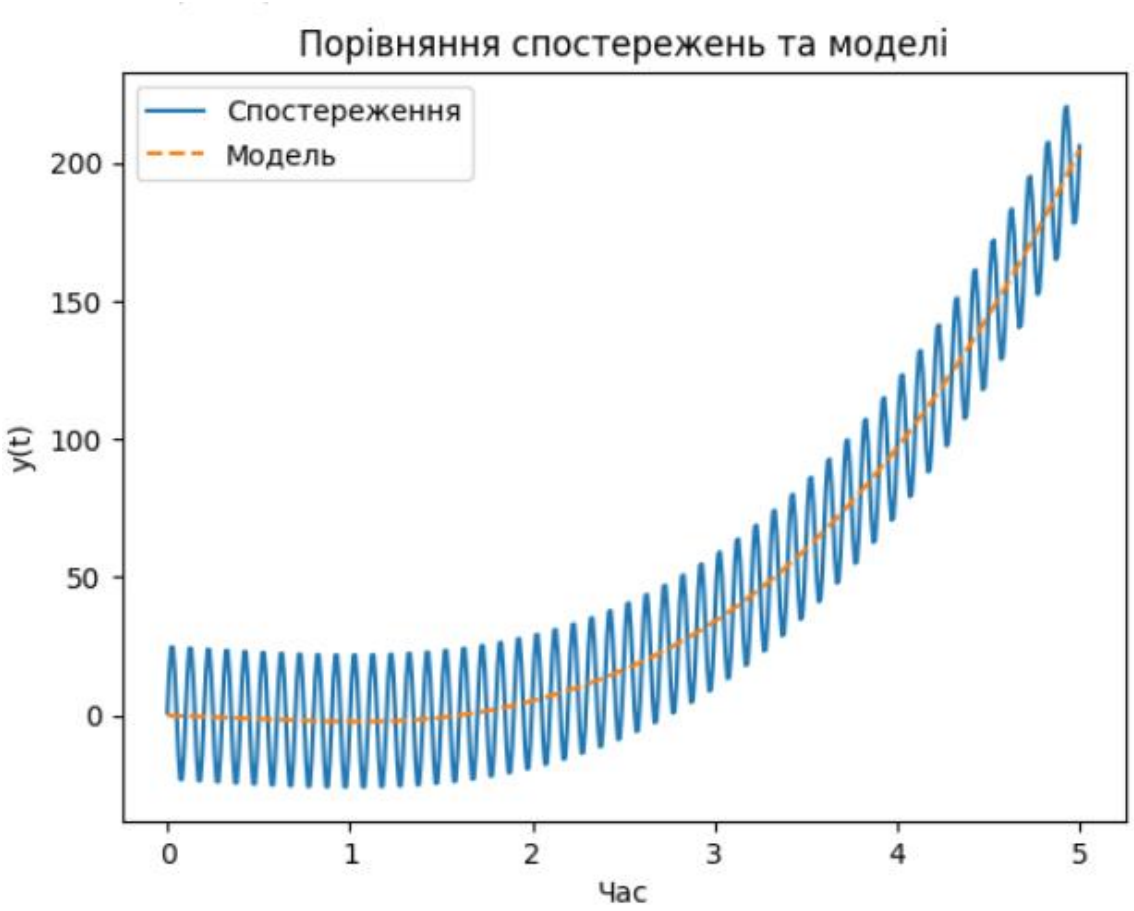
```
# 3. Least squares method
def model(t, a1, a2, a3, *params):
    k = len(params) // 2
    y = a1 * t**3 + a2 * t**2 + a3 * t
    for i in range(k):
        fi = params[i]
        ai = params[k + i]
        y += ai * np.sin(2 * np.pi * fi * t - 3 * t)
    return y

# Initialization of parameters
initial_guess = [1, 1, 1] + [1] * (len(peak_frequencies) * 2)

# 4. Parameter selection using the least squares method
params, covariance = curve_fit(model, t, observations, p0=initial_guess)

print("Знайдені параметри:", params)
```

І отримуємо такий результат:

```
Знайдені параметри: [ 2.00151029 -1.21105126 -2.98421085  1.00041963  0.29103001]
```

6) Й тепер можемо приступити до побудови графіка апроксимованої функції, і порівняємо його з моделлю

```
# 5. Calculating values using the model
fitted_values = model(t, *params)

plt.figure()
plt.plot(t, observations, label='Спостереження')
plt.plot(t, fitted_values, label='Модель', linestyle='--')
plt.xlabel('Час')
plt.ylabel('y(t)')
plt.legend()
plt.title('Порівняння спостережень та моделі')
plt.show()
```



І отримуємо таку апроксимовану функцію:

$$y(t) \approx 2.0015\,t^3 - 1.2111\,t^2 - 2.9842\,t + 1.0004\sin(2\pi \cdot 0.2910\,t).$$

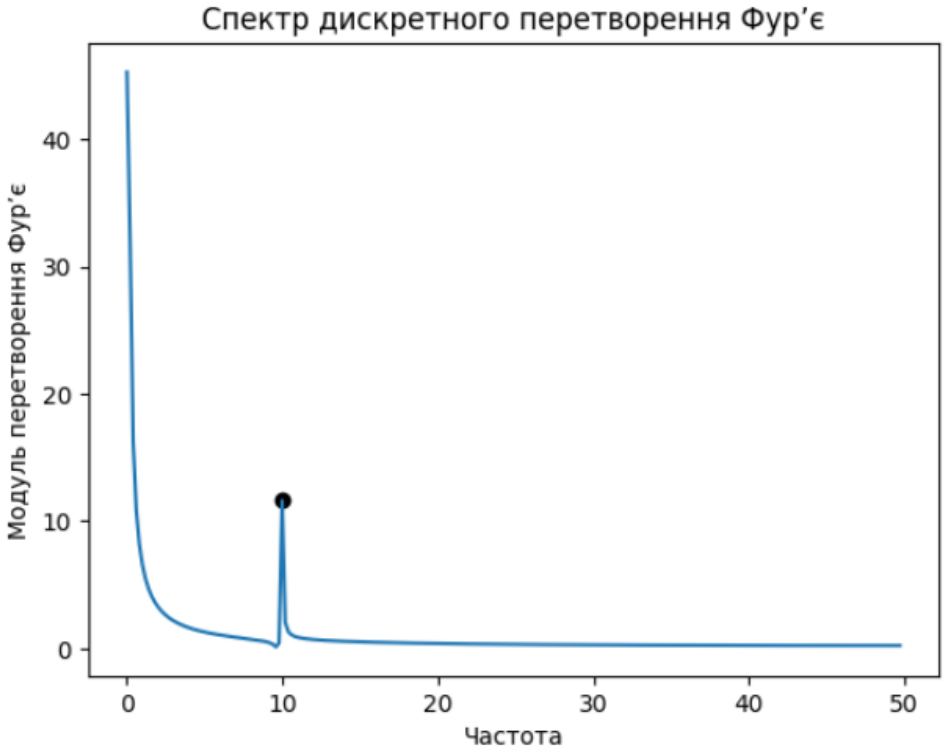
Повний код нашої програми:

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import find_peaks
from scipy.optimize import curve_fit

# load data
observations = np.loadtxt('f8.txt')

T = 5
N = len(observations)
dt = 0.01
t = np.arange(0, N*dt, dt)

# 1. Discrete Fourier transform
f_transform = np.fft.fft(observations) / N
frequencies = np.fft.fftfreq(N, dt)

#abs Fourier transform
f_magnitude = np.abs(f_transform)

# Local maxima
peaks, _ = find_peaks(f_magnitude[:N // 2])
peak_frequencies = frequencies[peaks]

print("Суттєві частоти:", peak_frequencies)

# 2. Building a graph of the Fourier transform module
plt.figure()
plt.plot(frequencies[:N // 2], f_magnitude[:N // 2])
plt.scatter(peak_frequencies, f_magnitude[peaks], color='gren')
plt.xlabel('Частота')
plt.ylabel('Модуль перетворення Фур’є')
plt.title('Спектр дискретного перетворення Фур’є')
plt.show()

# 3. Least squares method
def model(t, a1, a2, a3, *params):
    k = len(params) // 2
    y = a1 * t**3 + a2 * t**2 + a3 * t
    for i in range(k):
        fi = params[i]
```

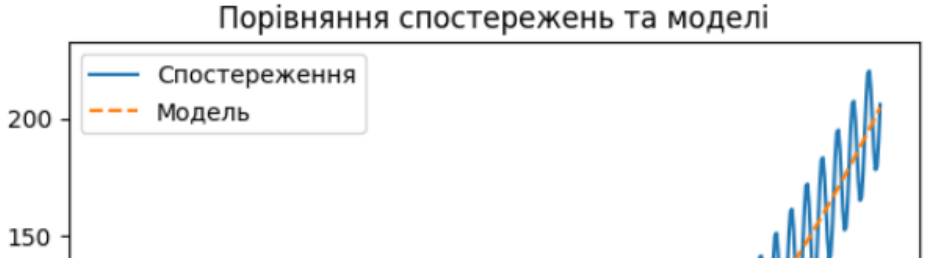
```
    for i in range(k):
        fi = params[i]
        ai = params[k + i]
        y += ai * np.sin(2 * np.pi * fi * t - 3 * t)
    return y

# Initialization of parameters
initial_guess = [1, 1, 1] + [1] * (len(peak_frequencies) * 2)

# 4. Parameter selection using the least squares method
params, covariance = curve_fit(model, t, observations, p0=initial_guess)

print("Знайдені параметри:", params)

# 5. Calculating values using the model
fitted_values = model(t, *params)

plt.figure()
plt.plot(t, observations, label='Спостереження')
plt.plot(t, fitted_values, label='Модель', linestyle='--')
plt.xlabel('Час')
plt.ylabel('y(t)')
plt.legend()
plt.title('Порівняння спостережень та моделі')
plt.show()
```

І весь вивід нашої програми:

```
Суттєві частоти: [9.98003992]
```




```
Знайдені параметри: [ 2.00151029 -1.21105126 -2.98421085  1.00041963  0.29103001]
```




```
Знайдені параметри: [ 2.00151029 -1.21105126 -2.98421085  1.00041963  0.29103001]
```