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

Звіт
з лабораторної роботи №1
з моделювання складних систем
Варіант 1

Виконала:
Студентка групи ІПС-32
Клевчук Марія Вячеславівна

Київ
2025

1. Постановка задачі

Нехай маємо дискретну функцію ў (t_i), i = 1, 2, …, N, що подана у вигляді значень сигналів у відповідний момент часу t_i.

Необхідно 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 ў (t_i), i = 1, 2, …, N, $t_{i+1} - t_i = \Delta t = 0.01$, інтервал спостереження [0, T], T = 5.

Для цього нам необхідно використати перетворення ряду Фур'є, для того, щоб розрахувати спектр частот для сигналів змінних у часі:

$$c_x(k) = \frac{1}{N}\sum_{m=0}^{N-1} x(m) e^{-i2\pi km/N}$$

Після, потрібно обчислити модуль ДПФ $|c_y(k)|$, та знайти момент у який приймається найбільше значення, тобто локальний максимум. Частоти, що відповідають цим максимумам, є частотами з найбільшим вкладом. Позначимо такі моменти через k^*. Щоб знайти саме частоти із найбільшим вкладом, які позначимо через f^*, потрібно виконати множення $f^* = k^*$ f $= k^* / T$

Після знаходження частот із найбільшим вкладом можна знаходити невідомі параметри a_i, i = k+1. Для цього застосовуємо метод найменший квадратів.

Для цього записуємо функціонал похибки:

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

параметри a_i, i = 1, 2, …, k+1 знаходимо з умови

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

Для цього записуємо систему рівнянь, що є системою лінійних алгебраїчних рівнянь:

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

Розв'язавши цю систему одним з відомих методів, отримуємо значення параметрів a_i, i = 1, 2, …, k+1.

2. Хід роботи

Програма реалізації виконана на Python.

Здійснюю ініціалізацію необхідних параметрів: читаю значення функції ŷ (t_i), i = 1, 2, …, N; рахую N - кількість моментів в інтервалі спостереження, крок dt (Δt) = 0,01.

```
def main():  1 usage
    # Читаємо значення y(t) з файлу
    observations = read_data_file('f1.txt')

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

Виконую дискретне перетворення Фур'є за відомою формулою.

```
# Дискретне перетворення Фур'є
def DFT(signal, N):  1 usage
    dft = np.zeros(N, dtype=complex)
    for k in range(N):
        for n in range(N):
            dft[k] += signal[n] * np.exp(-2j * np.pi * k * n / N)
    return dft
```

Тепер необхідно знайти суттєву частоту (локальний максимум спектра, тобто графіка модуля ДПФ). Для цього знаходимо df (Δf) = 1/T. Щоб знайти саме частоти із найбільшим вкладом, які позначимо через *f**, потрібно виконати множення f* = k*df, k=0, 1, …, [N/2]-1.

```python
# Знаходження локального максимуму (суттєва частота)
def find_significant_frequencies(dft, dt, T, N):  1 usage
    df = 1 / T

    # беремо тільки ліву половину спектра
    f_magnitude = np.abs(dft[:N//2])

    # шукаємо локальні максимуми
    peaks, _ = find_peaks(f_magnitude)

    if len(peaks) > 0:
        k_star = peaks[np.argmax(f_magnitude[peaks])]
        f_star = k_star * df
        return float(f_star)
    else:
        return None
```

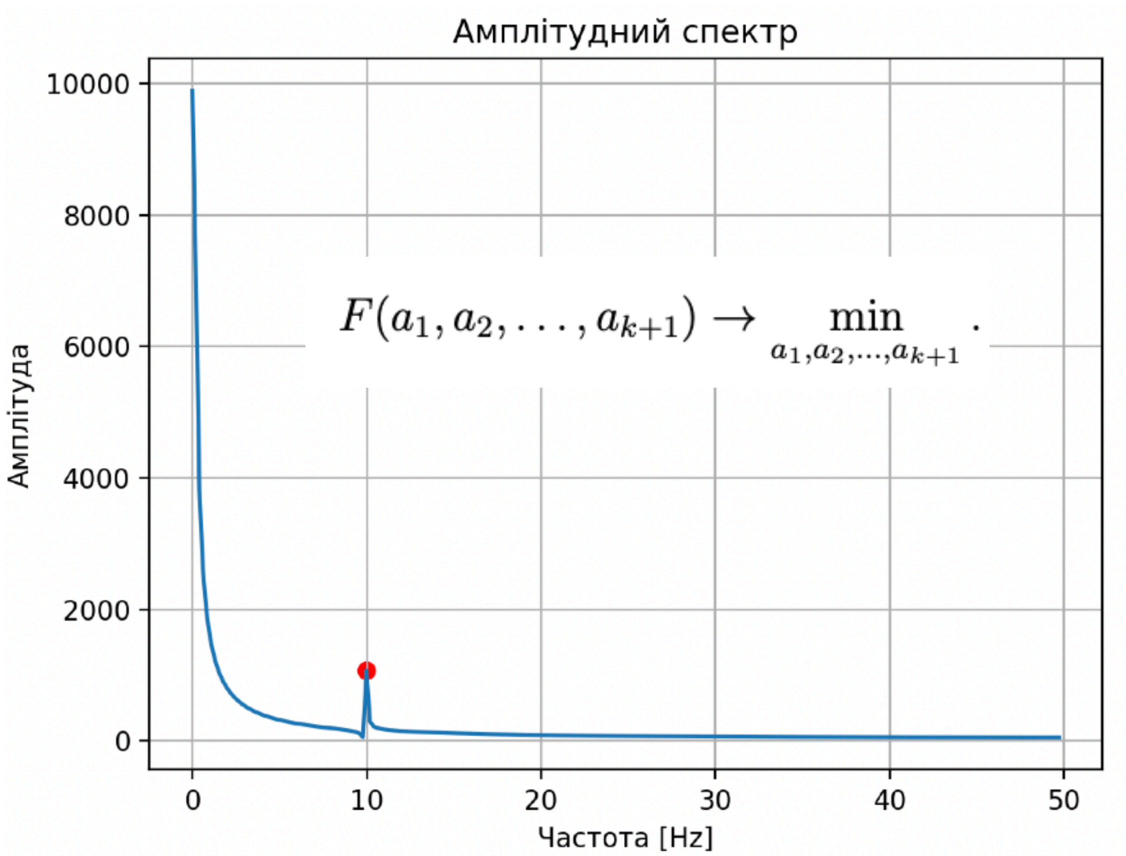
Викликаємо функції ДПФ та знаходження суттєвої частоти. Отримуємо результат 10.

```python
# ДПФ, пошук суттєвої частоти
spectrum = DFT(observations, N)
dominant_freq = find_significant_frequencies(spectrum, dt, T, N)
print("Суттєва частота:", dominant_freq)
```

```
Суттєва частота: 10.0
```

Тепер побудуємо графік спектру, беремо лише його половину, адже він симетричний. Бачимо пік на 10 Гц, тобто f=10.

```python
# Побудова графіка спектру
def plot_spectrum(spectrum, dt, dominant_freq):  1 usage
    magnitude = np.abs(spectrum)
    freqs = np.fft.fftfreq(len(spectrum), dt)
    plt.figure()
    half = len(spectrum)//2
    plt.plot( *args: freqs[:half], magnitude[:half])
    if dominant_freq is not None:
        # find nearest index in positive frequencies
        positive_mask = freqs[:half] >= 0
        pos_freqs = freqs[:half][positive_mask]
        pos_magnitude = magnitude[:half][positive_mask]
        idx = (np.abs(pos_freqs - dominant_freq)).argmin()
        plt.scatter( x: [pos_freqs[idx]],  y: [pos_magnitude[idx]], color='red')
    plt.xlabel("Частота [Hz]")
    plt.ylabel("Амплітуда")
    plt.title("Амплітудний спектр")
    plt.grid(True)
    plt.show()
```



Далі необхідно знайти невідомі параметри. Їх 5, адже маємо одну суттєву частоту (i = 1, k-3 = 1 -> k=4. a рахуються від 1 до k+1).
Зробимо це методом найменших квадратів. Виходимо з цієї формули (функціонал помилок):

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

параметри a_i, i = 1, 2, …, k+1 знаходимо з умови:

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

Будуємо систему рівнянь та розв'язуємо її:

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

Тобто беремо часткову похідну функціоналу помилок по кожному невідомому параметру, та розв'язуємо як систему рівнянь.

Реалізація в коді:

```python
# Знаходження невідомих параметрів
def fit_model(t, observations, peak_frequencies):  1 usage
    # Лінійна модель: y = a5 + a1 t^3 + a2 t^2 + a3 t + A1 sin(2π f1 t)
    ones_col = np.ones_like(t)
    t3_col = t**3
    t2_col = t**2
    t1_col = t

    if len(peak_frequencies) >= 1 and peak_frequencies[0] is not None:
        f1 = float(peak_frequencies[0])
        sin_col = np.sin(2 * np.pi * f1 * t)
        X = np.column_stack([t3_col, t2_col, t1_col, sin_col, ones_col])
    else:
        X = np.column_stack([t3_col, t2_col, t1_col, ones_col])

    params, *_ = np.linalg.lstsq(X, observations, rcond=None)
    params = np.round(params).astype(int)
    model_values = X @ params
    return params, model_values
```

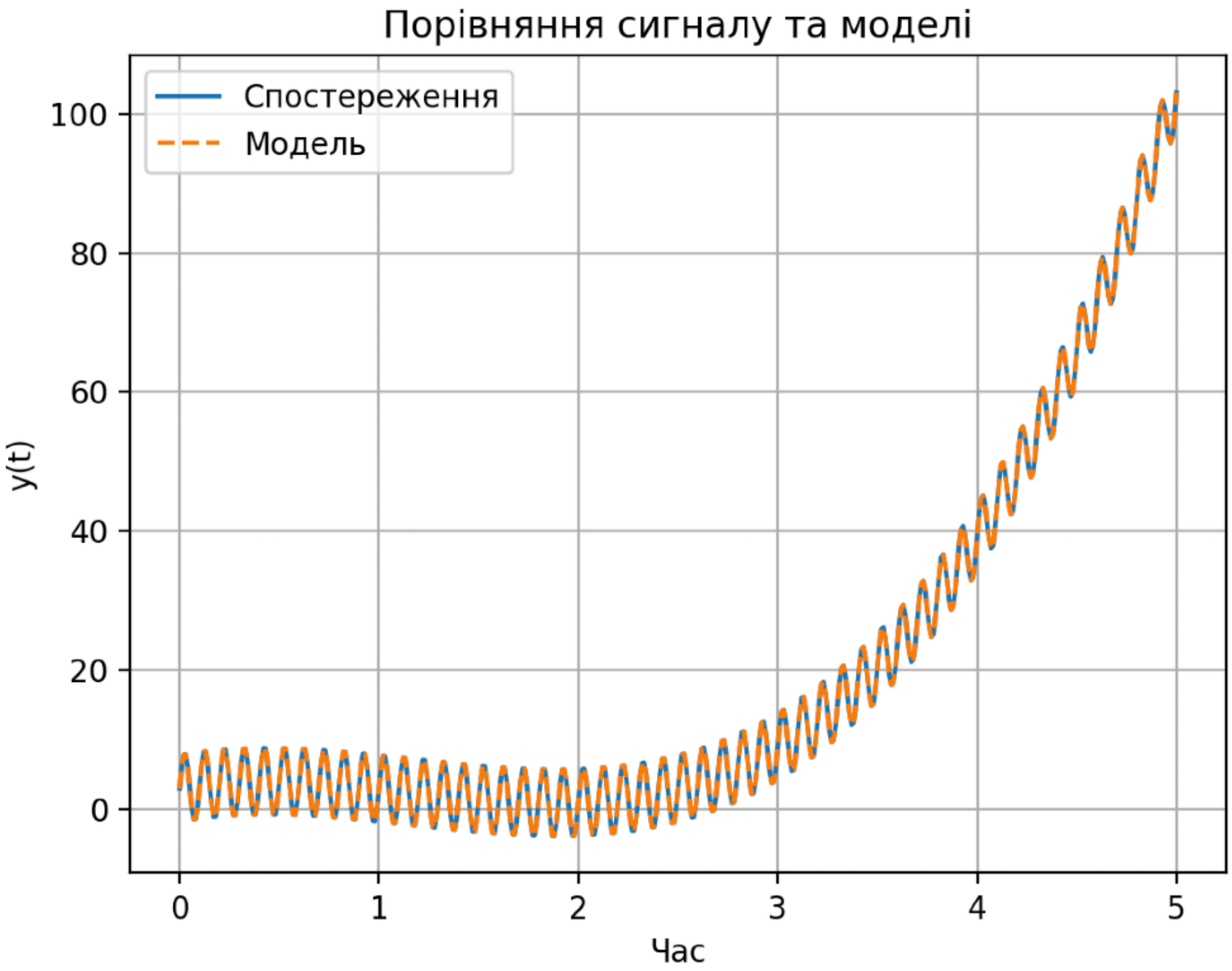
Викликаю функцію та отримую результати:

```python
# Параметри, знайдені методом найменших квадратів
params, model = fit_model(t, observations, [dominant_freq] if dominant_freq is not None else [])
print("Оцінені параметри:", params)
```

```
Оцінені параметри: [ 2 -7  5  5  3]
```

Зі знайденими f=10 та параметрами a_i, i = 1, 2, …, 5 отримуємо
y(t) = 2*t3 - 7*t2 + 5*t + 5*sin(2π*10t) + 3.

Тепер будую графіки ў(t) та y(t) та порівнюємо їх. Бачимо, що апроксимована функція дуже близька до моделі.


Порівняння сигналу та моделі
Спостереження
Модель
y(t)
Час
0
20
40
60
80
100
0
1
2
3
4
5