МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет електроніки та комп’ютерних технологій

Звіт

Про виконання лабораторної роботи №3

З курсу “Цифрова обробка інформації”

“Обчислення дискретного перетворення Фур’є (ДПФ, DFT)”

Виконав:
студент групи Фес-21
Шавало А. А.

Перевірив

Вдовиченко В. М.

Львів-2024

Завдання: Реалізувати пряме та обернене дискретне перетворення Фур'є намові програмування С++.

Теоретичні відомості та реалізація:

$$X(k) = \sum_{n=0}^{N-1} x(n)W^{nk}, k = 0, 1, \ldots, N-1 \quad \text{ДПФ},$$

$$x(n) = \frac{1}{N} \sum_{n=0}^{N-1} X(k)W^{-nk} \quad \text{ОДПФ},$$

Вхідна послідовність x(n) повинна представлятися в комплексній формі (дійсна і уявначастини). Для послідовності дійсних чисел уявні частини покладаються рівними нулю.

Перетворення Фур'є використовується для аналізу сигналів, розкладаючи їх на складові частоти та амплітуди, тобто для переведення сигналу з часової області в частотну. Після застосування дискретного перетворення Фур'є ми отримуємо спектр того ж розміру, що і початковий сигнал.

Насправді, результат дискретного перетворення Фур'є є набором комплексних значень, які описують спектр сигналу. Для обчислення кожного значення спектра необхідно взяти всі відліки сигналу в часовій області, помножити їх на відповідні комплексні амплітуди обраних частот і підсумувати. Таким чином, частотні компоненти, які ми отримуємо на певній частотній сітці, є елементами ряду Фур'є, в який розкладено початковий сигнал.

Існує також обернене дискретне перетворення Фур'є, яке дозволяє відновити сигнал з частотної області в часову. В ідеальному випадку, якщо кількість елементів ряду Фур'є була б нескінченною, ми могли б повністю відновити сигнал без втрат. Однак у реальних обчисленнях це неможливо, тому кількість елементів завжди обмежена, що призводить до помилок при перетвореннях.

Ці спотворення, викликані обмеженою кількістю членів ряду Фур'є, називаються ефектом Гібса. На практиці це призводить до появи небажаних коливань та викидів поблизу різких змін у сигналі, що виглядає як розмиття чи хвилеподібні коливання в околицях різких стрибків.

Лістинг коду:

```
import numpy as np


def dft(data, inv):
    N = len(data)
    WN = 2 * np.pi / N
    if inv == 1:
        WN = -WN

    X_real = np.zeros(N)
    X_imag = np.zeros(N)

    for k in range(N):
        wk = k * WN
        for n in range(N):
            c = np.cos(n * wk)
            s = np.sin(n * wk)
            X_real[k] += data[n].real * c + data[n].imag * s
            X_imag[k] -= data[n].real * s - data[n].imag * c

        if inv == 1:
            X_real[k] /= N
            X_imag[k] /= N

    result = [complex(X_real[k], X_imag[k]) for k in range(N)]
    return result


def main():
    data_input = input("Введіть послідовність дійсних чисел через пробіл: ")
    data = list(map(float, data_input.split()))
    data = [complex(x, 0) for x in data]

    direct_result = dft(data, inv=0)

    inverse_result = dft(direct_result, inv=1)

    print("Результат прямого ДПФ:")
    for r in direct_result:
        print(f"{r.real:.3f} + {r.imag:.3f}j")

    print("\nРезультат оберненого ДПФ:")
    for r in inverse_result:
        print(f"{r.real:.3f} + {r.imag:.3f}j")


if __name__ == "__main__":
    main()
```

Результат програми:

```
Ввести послідовність : 4 2 1 4 6 3 5 2
Результат прямого ДПФ:
27.000 + 0.000j
-4.121 + 3.293j
4.000 + 1.000j
0.121 + -4.707j
5.000 + 0.000j
0.121 + 4.707j
4.000 + -1.000j
-4.121 + -3.293j
```