Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут ім. Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

ЗВІТ

до лабораторної роботи №2.1

з дисципліни «Інтелектуальні вбудовані системи»

на тему «Дослідження параметрів алгоритму дискретного перетворення Фур’є»

Виконав:

студент групи ІП-83

Черевач А.М.

Перевірив:

асистент Регіда П.Г.

Київ - 2021

Основні теоретичні відомості

В основі спектрального аналізу використовується реалізація так званого дискретного перетворювача Фур'є (ДПФ) з неформальним (не формульним) поданням сигналів, тобто досліджувані сигнали представляються послідовністю відліків $x(k)$

$$F_x(p) = \sum_{k=0}^{N-1} x(k) \cdot e^{-jk\Delta tp\Delta\omega}$$

$$\omega \to \omega_p \to p\Delta\omega \to p \qquad \Delta\omega = \frac{2\pi}{T}$$

На всьому інтервалі подання сигналів **T**, **2π** - один період низьких частот. Щоб підвищити точність треба збільшити інтервал **T.**

$$t \to t_k \to k\Delta t \to k; \qquad \Delta t = \frac{T}{N} = \frac{1}{k_{зап.}} \cdot f'гр.$$

ДПФ - проста обчислювальна процедура типу звірки (тобто Σ-е парних множень), яка за складністю також має оцінку $\mathbf{N^2 + N}$. Для реалізації ДПФ необхідно реалізувати поворотні коефіцієнти ДПФ:

$$W_N^{pk} = e^{-jk\Delta t\Delta\omega p}$$

Ці поворотні коефіцієнти записуються в ПЗУ, тобто є константами.

$$W_N^{pk} = e^{-jk\frac{T}{N}p\frac{2\pi}{T}} = e^{-j\frac{2\pi}{N}pk}$$

W_N^{pk} не залежать від **T**, а лише від розмірності перетворення **N.** Ці коефіцієнти подаються не в експоненційній формі, а в тригонометричній.

$$W_N^{pk} = \cos\left(\frac{2\pi}{N}pk\right) - j\sin\left(\frac{2\pi}{N}pk\right)$$

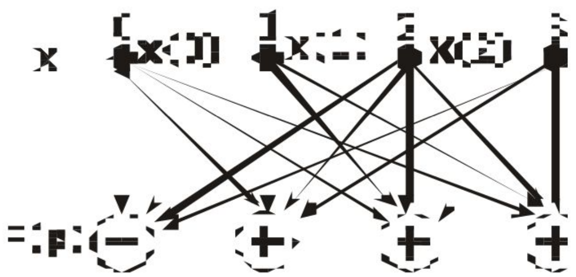
Ці коефіцієнти повторюються (тому і **p** до **N-1**, і **k** до **N-1**, а **(N-1) • (N-1)**) з періодом **N(2π)**.. Т.ч. в ПЗУ треба зберігати N коефіцієнтів дійсних і уявних частин. Якщо винести знак коефіцієнта можна зберігати **N/2** коефіцієнтів.

2π/N- деякий мінімальний кут, на який повертаються ці коефіцієнти. У ПЗУ окремо зберігаються дійсні та уявні частини компілюють коефіцієнтів. Більш загальна форма ДПФ представляється як:

$$F_x(p) = \sum_{k=0}^{N-1} x(k) \cdot W_N^{pk}$$

ДПФ дуже зручно представити у вигляді відповідного графа.

Приклад: граф 4-х точкового ДПФ. ($k = \overline{0,3}$; $p = \overline{0,3}$)



Коефіцієнти зручно представити у вигляді таблиці:

k \ p	0	1	2	3
0	W_4^0	W_4^0	W_4^0	W_4^0
1	W_4^0	W_4^1	W_4^2	W_4^3
2	W_4^0	W_4^2	W_4^0	W_4^2
3	W_4^0	W_4^3	W_4^2	W_4^1

Завдання

Для згенерованого випадкового сигналу з Лабораторної роботи N 1 відповідно до заданого варіантом (Додаток 1) побудувати його спектр, використовуючи процедуру

дискретного перетворення Фур'є. Розробити відповідну програму і вивести отримані значення і графіки відповідних параметрів.

Варіант

Номер залікової книжки - **8525**

Варіант в таблиці - **25**

Число гармонік в сигналі n - **12**

Гранична частота, $\omega_{гр}$ - **2700**

Кількість дискретних відліків, N - **64**

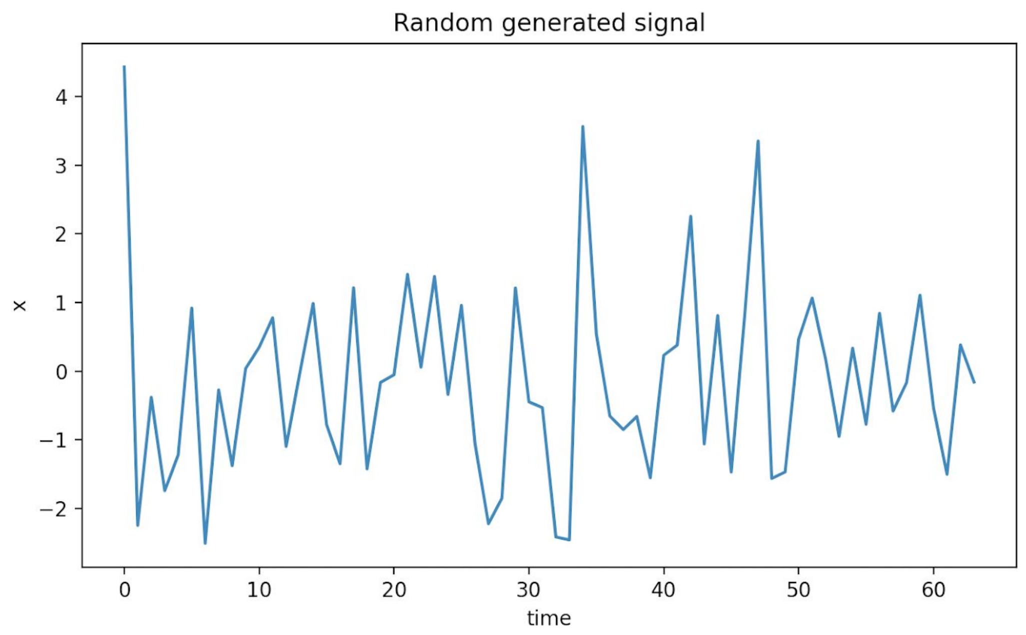
Лістинг програми

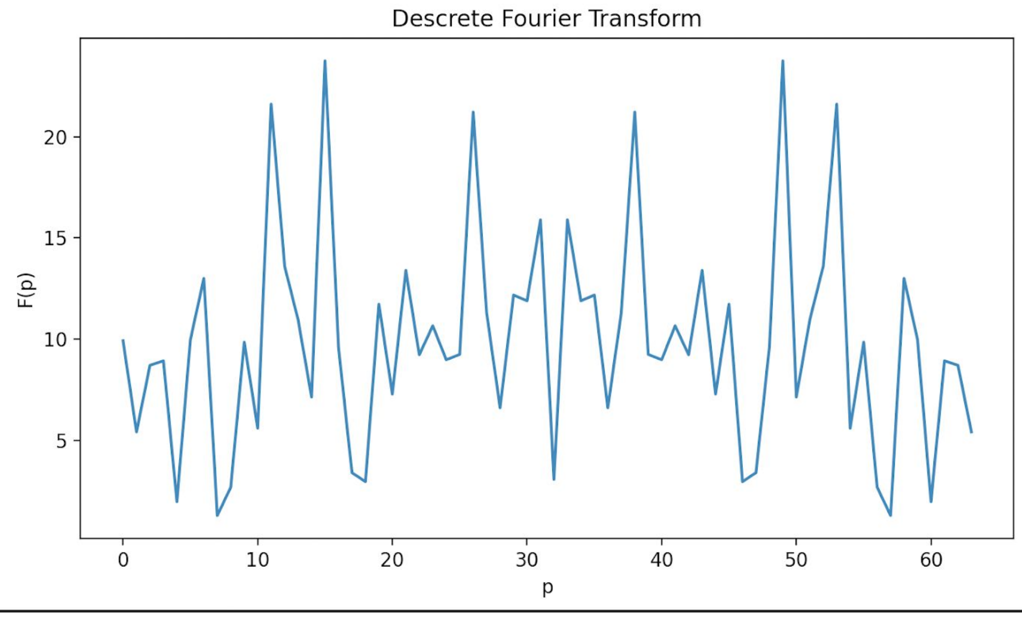
```
import matplotlib.pyplot as plt  # lib for graphs

import numpy as np  # lib for math operations

import math  # lib for math operations



# constants

n = 12  # number of harmonics

w = 2700  # max frequency

N = 64  # number of descrete calls



# function for calculating random signal

def formula(a, w, t, phi):
```

```python
    return a*np.sin(w*t+phi)


# function for generation array of signals
def generateSignals(n, w, N):
    signals = [0]*N  # array of signals
    w0 = w/n  # frequency
    for _ in range(n):

        for t in range(N):
            a = np.random.rand()  # amplitude
            phi = np.random.rand()  # phase
            signals[t] += formula(a, w0, t, phi)
        w0 += w0
    return signals


# function for calculating Discrete Fourier Transform coefficient
def dftCoeff(pk, N):
    exp = 2*math.pi*pk/N
    return complex(math.cos(exp), -math.sin(exp))


# function for calculating Discrete Fourier Transform
def dft(signals):
    N = len(signals)
    spectrum = []
    for p in range(N):
        sum = 0
```

```
        for k in range(N):

            sum+= signals[k] * dftCoeff(p*k, N)

        spectrum.append(abs(sum))


    return spectrum




signals = generateSignals(n, w, N)




# plotting


# signals

plt.plot(signals)

plt.xlabel('time')

plt.ylabel('x')

plt.title('Random generated signal')

plt.figure()


# dft

plt.plot(dft(signals))

plt.xlabel('p')

plt.ylabel('F(p)')

plt.title('Descrete Fourier Transform')

plt.show()
```

Результат роботи програми





Висновки

Під час виконання лабораторної роботи я дослідив принципи реалізації спектрального аналізу випадкових сигналів на основі алгоритму перетворення Фур'є. Було реалізовано програму обчислення спектру згенерованого сигналу на мові Phyton. Програма обчислює спектр за допомогою дискретного перетворення Фур’є і після цього виводить його графік.