Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №2.2
з дисципліни
«Інтелектуальні вбудовані системи»
на тему
«ДОСЛІДЖЕННЯ АЛГОРИТМУ ШВИДКОГО ПЕРЕТВОРЕННЯ ФУР'Є З ПРОРІДЖУВАННЯМ ВІДЛІКІВ СИГНАЛІВ У ЧАСІ »

Виконав: Перевірив:
студент групи ІП-84 викладач
Сімонов Павло Ігорович Регіда Павло Геннадійович
номер залікової книжки: 8421

Київ 2021

Основні теоретичні відомості

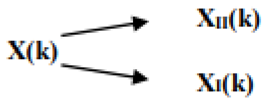
Швидкі алгоритми ПФ отримали назву схеми Кулі-Тьюкі. Всі ці алгоритми використовують регулярність самої процедури ДПФ і те, що будь-який складний коефіцієнт W_N^{pk} можна розкласти на прості комплексні коефіцієнти.

$$W_N^{pk} = W_N^1 W_N^2 W_N^3$$

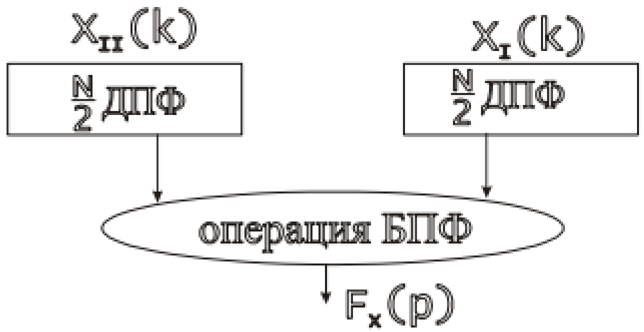
Для стану таких груп коефіцієнтів процедура ДПФ повинна стати багаторівневою, не порушуючи загальних функціональних зв'язків графа процедури ДПФ. Існують формальні підходи для отримання регулярних графів ДПФ. Всі отримані алгоритми поділяються на 2 класи:

1) На основі реалізації принципу зріджені за часом $\mathbf{X_к}$

2)на основі реалізації принципу зріджені відліків шуканого спектру **F(p).**

Найпростіший принцип зріджені - поділу на парні/непарні пів-послідовності, які потім обробляють паралельно. А потім знаходять алгоритм, як отримати шуканий спектр. Якщо нам вдасться ефективно розділити, а потім алгоритм 

отримання спектра, то ми можемо перейти від N ДПФ до N/2 ДПФ.



Розглянемо формальний висновок алгоритму ШПФ, який реалізує в одноразовому застосуванні принцип проріджування по часу:

$$F_x(p)=\sum_{k=0}^{N-1}X(k)W_N^{pk}=\sum_{k=0}^{N-2}X_{II}(k)W_N^{pk}+\sum_{k=1}^{N-2}X_I(k)W_N^{pk}$$

$$X_{II}(k)\to X(2k^*);\; X_I(k)\to X(2k^*+1);\; k^*=0;\frac{N}{2}-1$$

$$F_x(p)=\sum_{k^*=0}^{\frac{N}{2}-1}X(2k^*)W_N^{pk}+\sum_{k^*=0}^{\frac{N}{2}-1}X(2k^*+1)W_N^{p(2k^*+1)}$$

$$W_N^{p2k^*}=e^{-j\frac{2\pi}{N}p2k^*}=e^{-j\frac{2\pi}{N/2}pk^*}=W_{\frac{N}{2}}^{pk^*}$$

У цій першій сумі з'явилися коефіцієнти в 2 рази менше. У другій сумі з'явився множник, який не залежить від k * тобто він може бути винесений за знак суми.

$$W_N^{p(2k^*+1)}=W_N^{p2k^*}\cdot W_N^p=W_{\frac{N}{2}}^{pk^*}W_N^p$$

$$F_x(p)=\underbrace{\sum_{k^*=0}^{\frac{N}{2}-1}X(2k^*)W_{\frac{N}{2}}^{pk^*}}_{F_{II}(p^*)}+W_N^p\underbrace{\sum_{k^*=0}^{\frac{N}{2}-1}X(2k^*+1)W_{\frac{N}{2}}^{pk^*}}_{F_I(p^*)}$$

Завдання

Для згенерованого випадкового сигналу з Лабораторної роботи N 1 відповідно до заданого варіантом (Додаток 1) побудувати його спектр, використовуючи процедуру швидкого перетворення Фур'є з проріджуванням відліків сигналу за часом. Розробити відповідну програму і вивести отримані значення і графіки відповідних параметрів.

Варіант

21

Число гармонік в сигналі, n - 14

Гранична частота, ωгр - 1800

Кількість дискретних відліків, N - 256

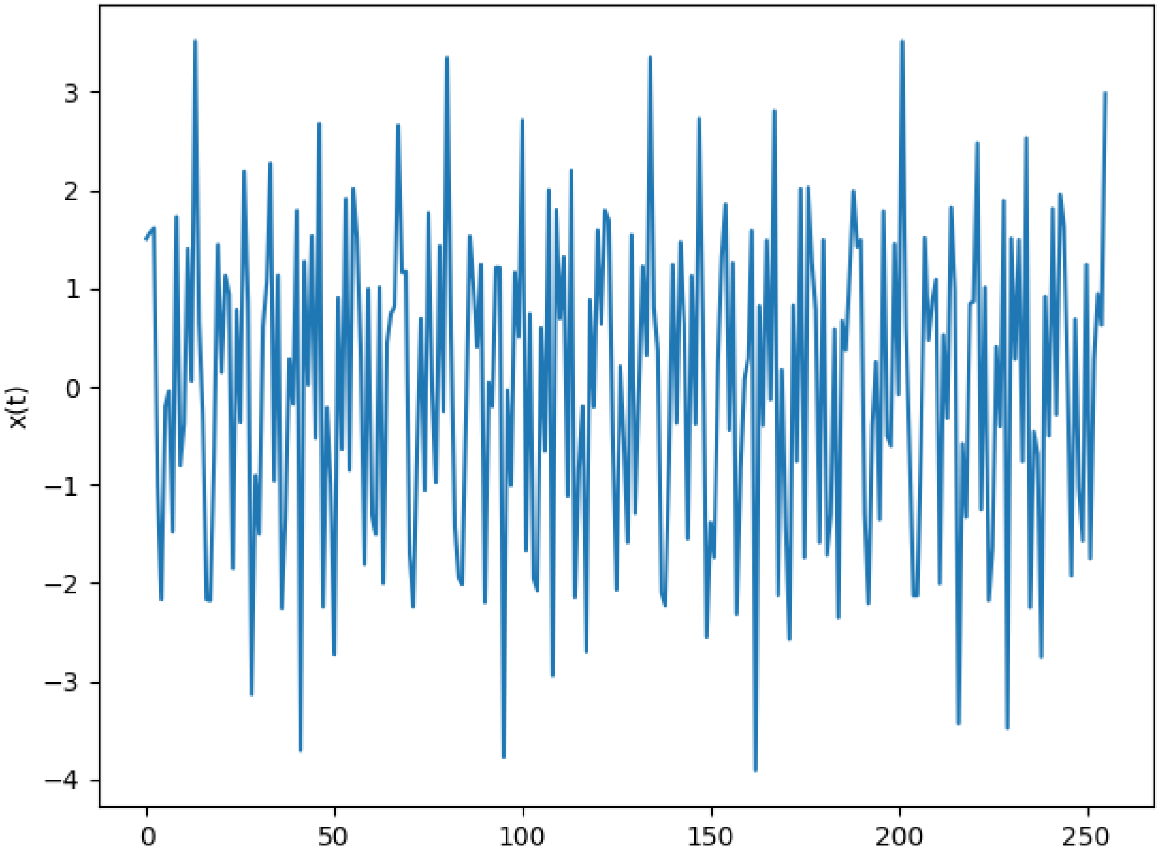
Лістинг програми

```
import random
import math
import time

import numpy as np
import matplotlib.pyplot as plt

n = 14
omega = 1800
N = 256  # 256
w = 0
i = 0
t = 0
x = np.zeros(N)
f = np.complex64(np.zeros(N))


def fun1(i, n, w, omega, arr):
    while i < n:
        w += omega / n
        i += 1
        t = 0
        A = random.random()
        fi = np.random.uniform(-np.pi / 2, np.pi / 2)
        while t < N:
            arr[t] += A * math.sin(w * t + fi)
            t += 1
```

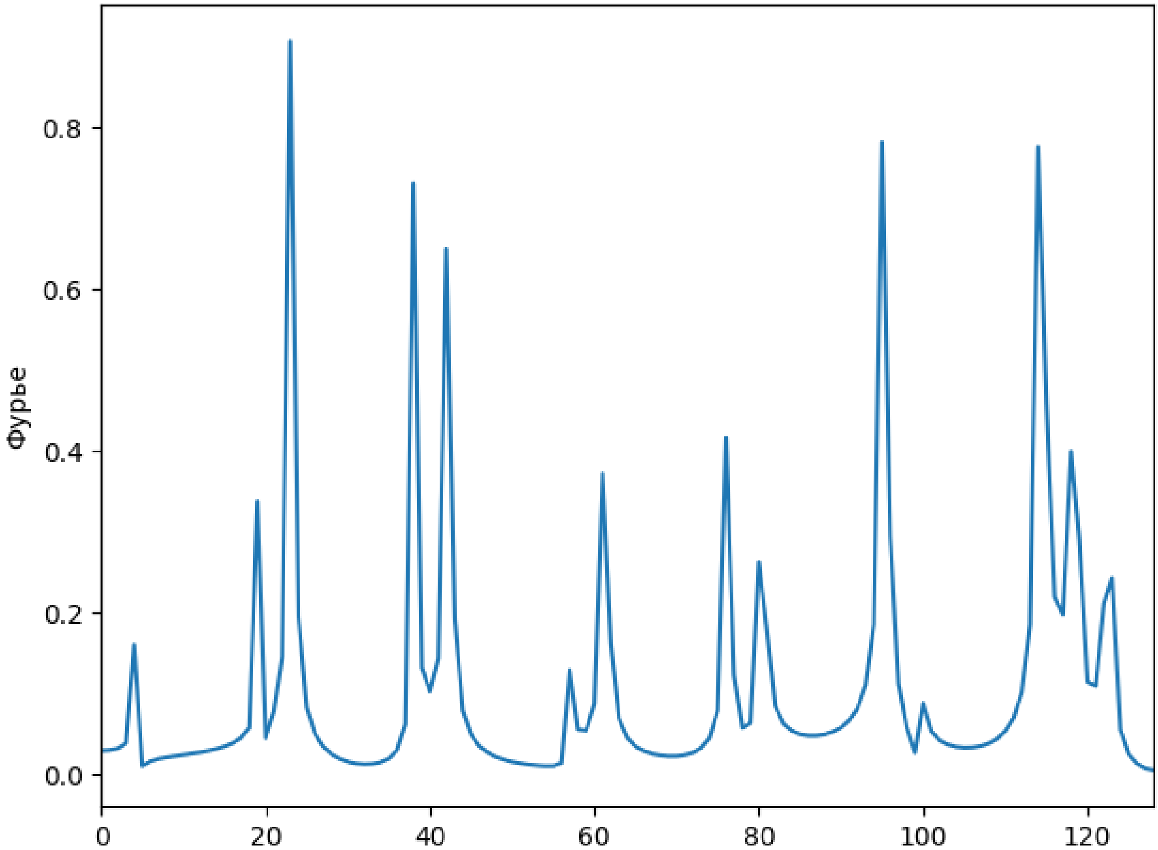
```
fun1(0, n, w, omega, x)


plt.plot(x)
plt.ylabel('x(t)')
plt.show()


p = 0
k = 0



while p < N:
    k = 0
    while k < N:
        f[p] += x[k] * (np.cos(2 * np.pi * p * k / N) - np.sin(2 * np.pi * p * k / N) * 1j)
        k += 1
    p += 1



def fft(x):
    V = len(x)
    if V <= 1:
        return x
    even = fft(x[0::2])
    odd = fft(x[1::2])
    T = [np.exp(-2j * np.pi * m / V) * odd[m] for m in range(V // 2)]
    return [even[m] + T[m] for m in range(V // 2)] + [even[m] - T[m] for m in range(V // 2)]
```

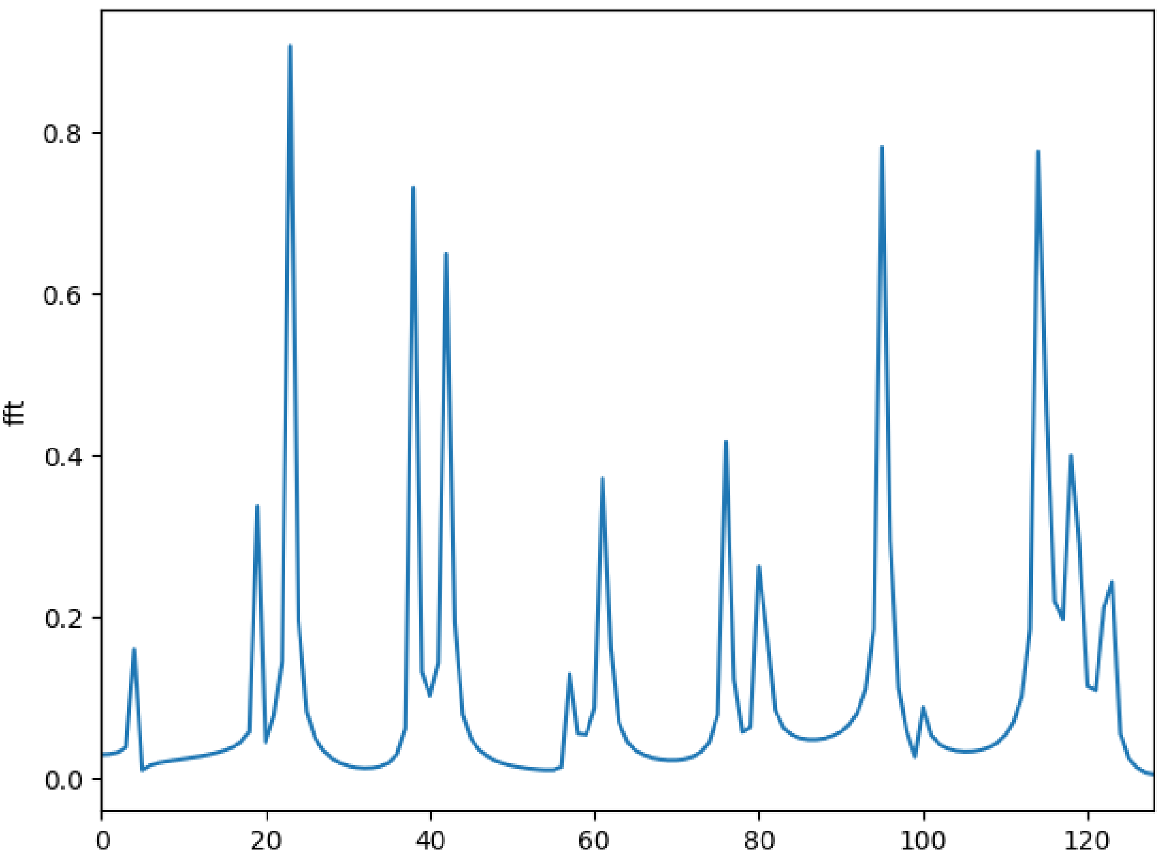
```
A = np.zeros(N)
B = np.zeros(N)
y = fft(x)
A = np.abs(f)
B = np.abs(y)
A = 2 * A / N
B = 2 * B / N

plt.plot(A)
plt.ylabel("Фурье")
plt.xlim(0, 128)  # 128
plt.show()

plt.plot(B)
plt.ylabel("fft")
plt.xlim(0, 128)  # 128
plt.show()
```

Результати роботи програми



x(t)
3
2
1
0
−1
−2
−3
−4
0
50
100
150
200
250


Фурье
0.0
0.2
0.4
0.6
0.8
0
20
40
60
80
100
120


fft
0.0
0.2
0.4
0.6
0.8
0
20
40
60
80
100
120

Висновки

У цій лабораторній роботі ми вивчили алгоритм Швидко Перетворення Фур'є з проріджуванням відлків сігналів у часі. Зробили відповідний код який демонструє роботу цього алгоритму і порівняли з графіком з лабораторної роботи 2.1