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

ЗВІТ

Лабораторна робота №2.2

з дисципліни
«Інтелектуальні вбудовані системи»

на тему
«Дослідження алгоритму швидкого перетворення фур'є з проріджуванням відліків сигналів у часі»

Виконав:
Василиненко Д.Д.

Студент групи ІП-84

Перевірив:

Регіда Павло Геннадійович

Київ 2021

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.

Основні теоретичні відомості

Швидкі алгоритми ПФ отримали назву схеми Кулі-Тьюкі. Всі ці алгоритми використовують регулярність самої процедури ДПФ і те, що будьякий складний коефіцієнт можна розкласти на прості комплексні коефіцієнти. 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 2)на основі реалізації принципу зріджені відліків шуканого спектру F(p). Найпростіший принцип зріджені - поділу на парні/непарні півпослідовності, 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. При переході до процедури БПФ з зрідженим за часом від класичного ДПФ вже з'являється кілька рівнів перетворення даних і на кожному рівні ми в деякій послідовності поділяємо знову на парні і непарні відліки. У БПФ з зрідженим за часом розмірність замінних перетворень ДПФ знизу вгору. Загальна кількість рівнів заміщення в БПФ: m = log^2(N) , N-ціла ступінь 2. Тут мова йде про найпростішому БПФ з основою 2, тобто в базові операції є 2-х точкове БПФ. На кожному рівні БПФ з зрідженим за часом треба здійснити додаткову перестановку відліків на парні і непарні, що призводить до загального недоліку алгоритму - необхідність спеціальної перестановки відліків досліджуваного сигналу перед перетворенням. Для реалізації ШПФ з зрідженим за часом треба здійснювати двійкову инверсную перестановку відліків або просто обчислити адреси. Базова операція БПФ може виконуватися в один такт по паралельній схемі з 4-ма множниками; в 2 такти по послідовно-паралельною схемою з 2 множниками; в 4 такти по послідовній схемі з одним помножувачем і декількома регістрами.

Завдання по варіанту (4):

- Число гармонік в сигналі - 12
- Гранична частота - 2400
- Кількість дискретних відліків - 1024

Вихідний код:

f.py

```
import math

def fCoef(pk, N):
  arg = 2 * math.pi * pk / N
  return complex(math.cos(arg), -math.sin(arg))

def f(signal):
  N = len(signal)
  if N == 1: return signal
  length = int(N / 2)
  spectre = [None] * N
  evens = f(signal[::2])
  odds = f(signal[1::2])
  for p in range(length):
    w = fCoef(p, N)
    product = odds[p] * w
    spectre[p] = evens[p] + product
    spectre[p + length] = evens[p] - product
  return spectre
```

rsg.py

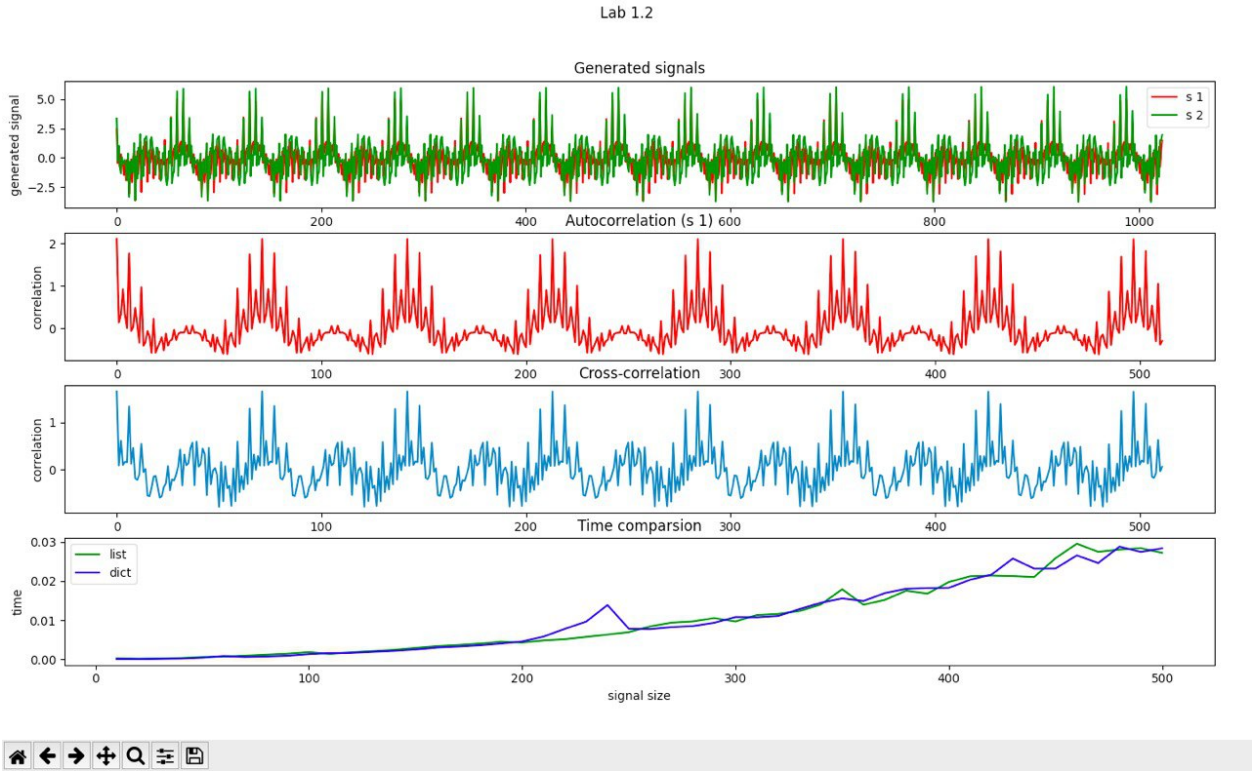
```
import random
import math
import time

def getRandomSignal(harmonicsAmount, limitFrequency, N):
    signal = [0] * N
```

```
    for i in range (harmonicsAmount):
        w = (limitFrequency / harmonicsAmount) * (i+1)
        A = random.random()
        Fi = random.random()
        for t in range(N):
            signal[t] += (A * math.sin(w * t + Fi))
    return signal
```

index.py

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import rsg
import f

n = 12
W = 2400
N = 1024
time = range(N)
signal = rsg.getRandomSignal(n, W, N)

s = f.f(signal)

fig, (ax1, ax2) = plt.subplots(2, 1)
fig.suptitle('Lab 2.2')

ax1.plot(signal, linewidth=0.8)
ax1.set_title('Generated signal')
ax1.set(xlabel='time', ylabel='generated signal')

ax2.plot(s, color='r', linewidth=0.8)
ax2.set_title('Fast Fourier transform')
ax2.set(xlabel='p', ylabel='F(p)')

fig.savefig("lab2.2.png")

plt.show()
```

Результати роботи програми



Висновки

Під час виконання даної лабораторної роботи ми ознайомилися з принципами реалізації прискореного спектрального аналізу випадкових сигналів на основі алгоритму швидкого перетворення Фур'є, вивчили та дослідили особливості даного алгоритму з використанням засобів моделювання і сучасних програмних оболонок.