МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №4

з дисципліни «Системи реального часу»

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

Виконав:
студент гр. ІП-83
Бойко Андрій

Перевірив:
Регіда П.Г.

Київ 2021

Завдання

Мета роботи - ознайомлення з принципами реалізації прискореного спектрального аналізу випадкових сигналів на основі алгоритму швидкого перетворення Фур'є, вивчення та дослідження особливостей даного алгоритму з використанням засобів моделювання і сучасних програмних оболонок.

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.

Програмний код

utils.js

```
const complexAdd = (a, b) => [a[0] + b[0], a[1] + b[1]];

const complexSubtract = (a, b) => [a[0] - b[0], a[1] - b[1]];

const complexMultiply = (a, b) => [a[0] * b[0] - a[1] * b[1],
   a[0] * b[1] + a[1] * b[0]];

const complexMagnitude = c => Math.sqrt(c[0] * c[0] + c[1] * c[1]);


const mapExponent = {},
        exponent = (k, N) => {
           const x = -2 * Math.PI * (k / N);

           mapExponent[N] = mapExponent[N] || {};
           mapExponent[N][k] = mapExponent[N][k] || [Math.cos(x), Math.sin(x)];

           return mapExponent[N][k];
        };
```

```
const convert = arr => arr.map(([x, y]) => Math.sqrt(Math.pow(x, 2) +
Math.pow(y, 2)));

module.exports = {
  complex: {
    add: complexAdd,
    subtract: complexSubtract,
    multiply: complexMultiply,
    magnitude: complexMagnitude,
  },
  exponent,
  convert,
};
```

fft.js

```
const { complex, exponent } = require('./utils');

function fft(vector) {
  const X = [];
  const N = vector.length;

  if (N === 1) {
    if (Array.isArray(vector[0])) {
      return [[vector[0][0], vector[0][1]]];
    } else {
      return [[vector[0], 0]];
    }
  }

  const even = (__, ix) =>  ix % 2 === 0;

  const odd = (__, ix) => ix % 2 === 1;

  const X_evens = fft(vector.filter(even)),
        X_odds  = fft(vector.filter(odd));
```

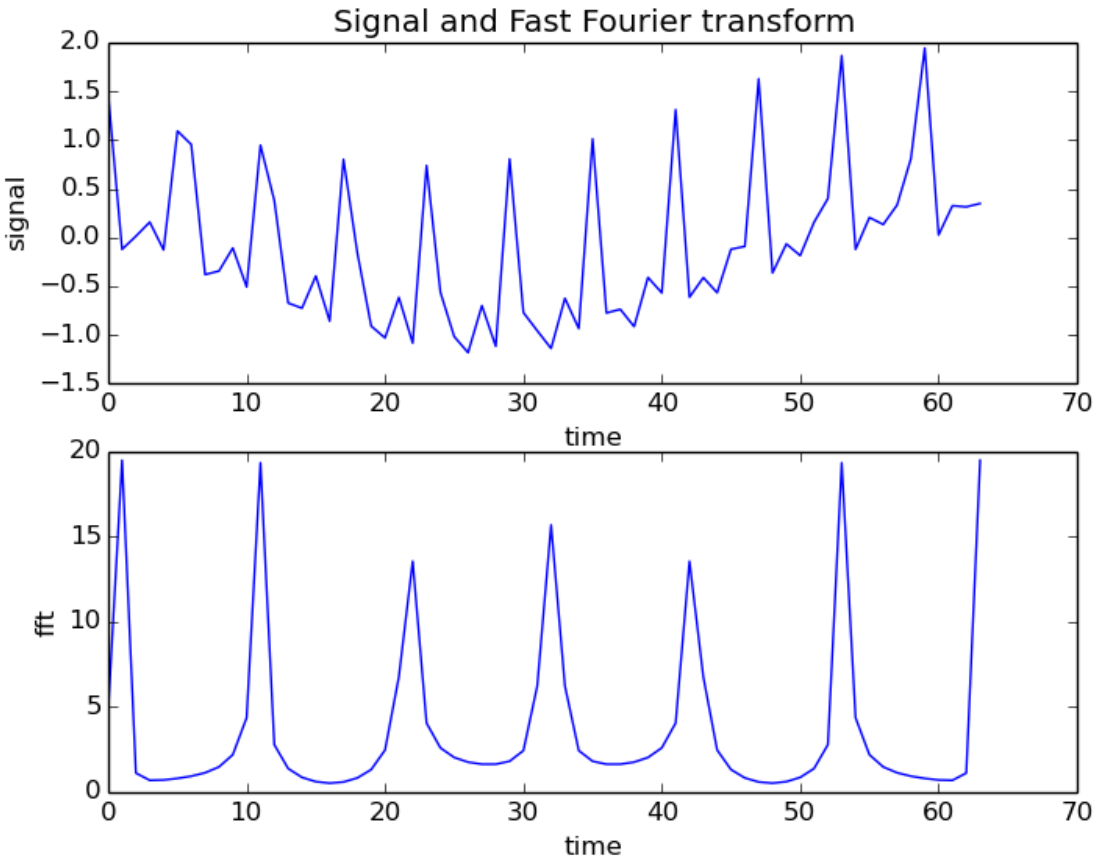
```
    for (let k = 0; k < N / 2; k++) {
        const t = X_evens[k],
                e = complex.multiply(exponent(k, N), X_odds[k]);

        X[k] = complex.add(t, e);
        X[k + (N / 2)] = complex.subtract(t, e);
    }

    return X;
}

module.exports = {
    fft,
};
```

index.js

```
const { generateSignal } = require('../../1/generateSignal');
const plt = require('matplotnode');
const path = require('path');
const { convert } = require('../utils');
const { fft } = require('../fft');

const harmonics = 6;
const frequency = 1200;
const discreteCalls = 64;

const signal = generateSignal(
    harmonics,
    frequency,
    discreteCalls,
);

const spectrumDft = fft(signal);

plt.subplot('211');
plt.title('Signal and Fast Fourier transform');
plt.plot([...Array(discreteCalls).keys()], signal);
```

```
plt.xlabel('time');
plt.ylabel('signal');
plt.legend();

plt.subplot('212');
plt.plot([...Array(discreteCalls).keys()], convert(spectrumDft));
plt.xlabel('time');
plt.ylabel('fft');
plt.legend();


const currentDir = path.join(__dirname, '/2.2.png');
plt.save(currentDir);
```

Результати виконання програми



Висновки

Під час виконання лабораторної роботи, було розкладено сигнал за допомогою оптимізованого алгоритму трансформації Фур’є, швидкість виконання якого для великої кількості дискретних відліків набагато вища за дискретне перетворення.