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 №2.1-2.2

з дисципліни
«Інтелектуальні вбудовані системи»

на тему
“ДОСЛІДЖЕННЯ ПАРАМЕТРІВ АЛГОРИТМУ ДИСКРЕТНОГО ПЕРЕТВОРЕННЯ ФУР’Є", “ДОСЛІДЖЕННЯ АЛГОРИТМУ ШВИДКОГО ПЕРЕТВОРЕННЯ ФУР'Є З ПРОРІДЖУВАННЯМ ВІДЛІКІВ СИГНАЛІВ У ЧАСІ”

Виконав:

студент групи ІП-83

Кухаренко Олександр Олександрович
номер залікової книжки: 8312

Перевірив:

ас. Регіда П. Г.

Київ 2021

Завдання:
Заліковка 8312
Варіант 12
Число гармонік в сигналі n = 8
Гранична частота, $\omega_{гр}$ = 1200
Кількість дискретних відліків, N = 256

Для згенерованого випадкового сигналу з Лабораторної роботи N 1 відповідно до заданого варіантом побудувати його спектр, використовуючи процедуру дискретного перетворення Фур'є. Розробити відповідну програму і вивести отримані значення і графіки відповідних параметрів.

Для згенерованого випадкового сигналу з Лабораторної роботи N 1 відповідно до заданого варіантом 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.

Лістинг програми:

1) Процедура дискретного перетворення Фур'є

```
const getDFT = signals => {
  const result = [];
  for (let i = 0; i < signals.length; i++) {
    let sum = math.complex();
    for (let j = 0; j < signals.length; j++) {
      const arg = (2 * Math.PI * i * j) / signals.length;
      const w = math.complex(math.cos(arg), -math.sin(arg));
      sum = math.add(sum, math.multiply(w, signals[j]));
    }
    result.push(sum);
  }
  return result;
};
```

2) Процедура швидкого перетворення Фур'є

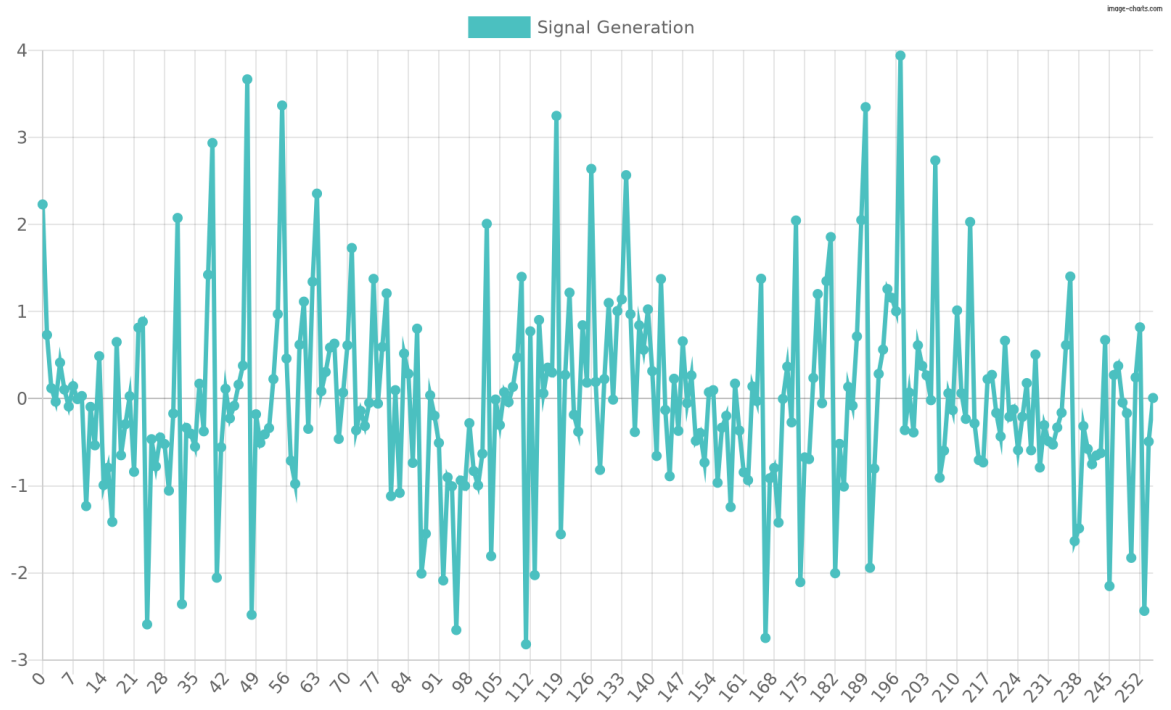
```
const getFFT = signals => {
  if (signals.length === 1) {
    return signals;
  }
  const result = [];
  const evens = getFFT(signals.filter((value, index) => !(index % 2)));
  const odds = getFFT(signals.filter((value, index) => index % 2));
  for (let i = 0; i < signals.length / 2; i++) {
    const x = -2 * Math.PI * (i / signals.length);
    const root = math.complex(math.cos(x), math.sin(x));
    result[i] = math.add(evens[i], math.multiply(root, odds[i]));
    result[i + signals.length / 2] = math.subtract(evens[i], math.multiply(root, odds[i]));
  }
  return result;
};
```

3) Генерація випадкового сигналу

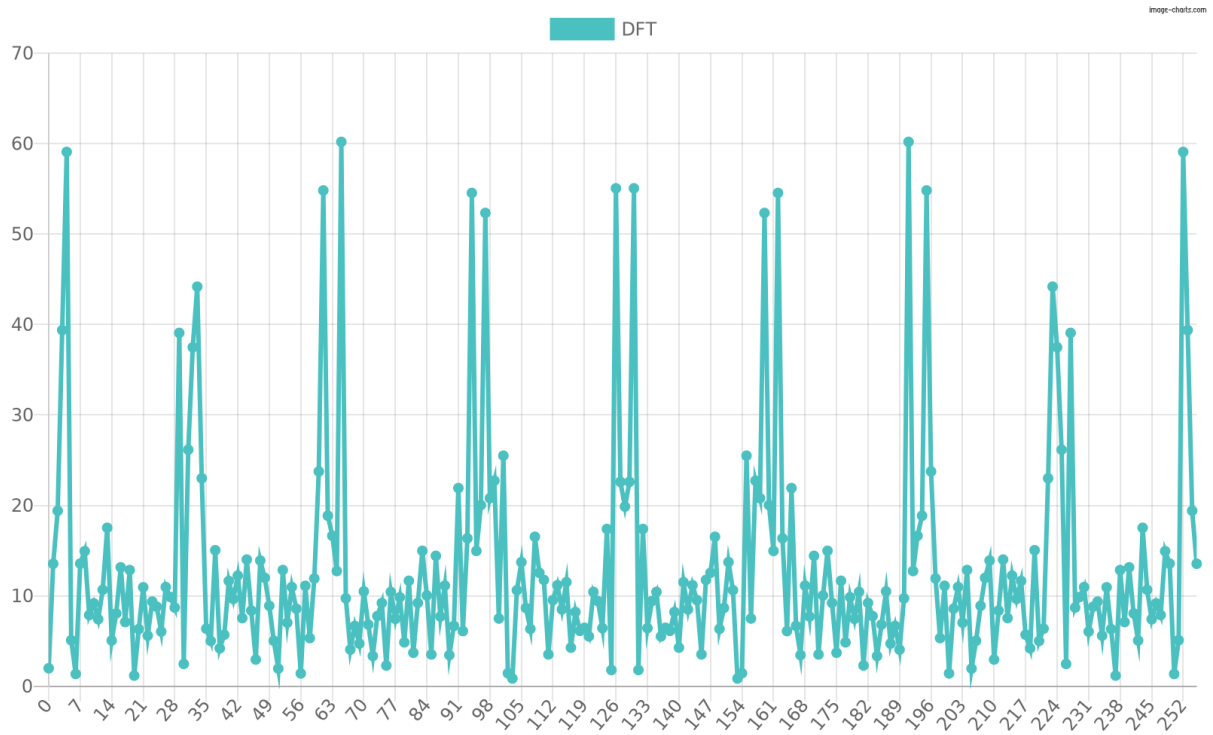
```
class SignalGenerator {
  constructor(signalHarmonics, frequency, disRepetitions) {
    this.signalHarmonics = signalHarmonics;
    this.disRepetitions = disRepetitions;
    this.minW = frequency / signalHarmonics;
    this.points = {};
  }

  setPoint(x, y) {
    const ay = this.points[x] || 0;
    this.points[x] = ay + y;
  }

  generateSignal() {
    for (let i = 1; i <= this.signalHarmonics; i++) {
      const wi = this.minW * i;
      for (let t = 0; t < this.disRepetitions; t++) {
        const x = Math.random() * Math.sin(wi * t + Math.random());
        this.setPoint(t, x);
      }
    }
    return { x: Object.keys(this.points), y: Object.values(this.points) };
  }
}
```

Отримані графіки:

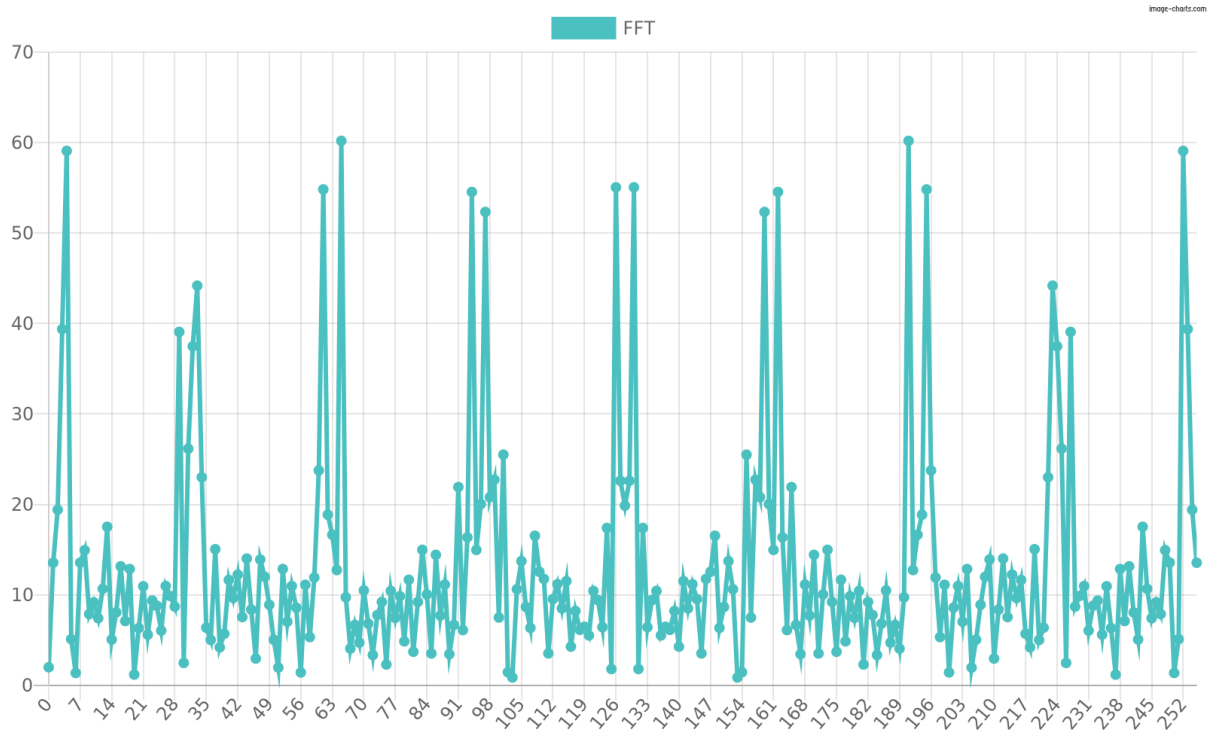
1) Генерація сигналу



2) Процедура дискретного перетворення Фур'є



3) Процедура швидкого перетворення Фур'є



Висновки:

Під час виконання лабораторних робіт 2.1 та 2.2 ми ознайомились з принципами реалізації спектрального аналізу випадкових сигналів на основі алгоритму перетворення Фур'є та на основі швидкого перетворення Фур’є, вивчили та дослідили особливості даного алгоритму з використанням засобів моделювання і сучасних програмних оболонок.