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 №1

з дисципліни «Системи реального часу»

на тему *«Дослідження і розробка моделей випадкових сигналів. Аналіз їх характеристик»*

Виконав:
студент гр. ІП-83
Бойко Андрій

Перевірив:
Регіда П.Г.

Київ 2021

Основні теоретичні відомості

СРЧ обов'язково пов'язані з деякою зовнішнім середовищем. СРЧ забезпечує контроль за зміною параметрів зовнішнього середовища і в ряді випадків забезпечує управління параметрами середовища через деякі впливу на неї. Параметри середовища представляються деякою зміною фізичного середовища. При вимірах фізичного параметра ми отримуємо певний електричний сигнал на вході вимірювального датчика. Для подання такого електричного сигналу можна використовувати різні моделі. Найкращою моделлю досліджуваного сигналу є відповідна математична інтерпретація випадкового процесу. Випадковий сигнал або процес завжди представляється деякою функцією часу $x(t)$, значення якої не можна передбачити з точністю засобів вимірювання або обчислень, які б кошти моделі ми не використовували.

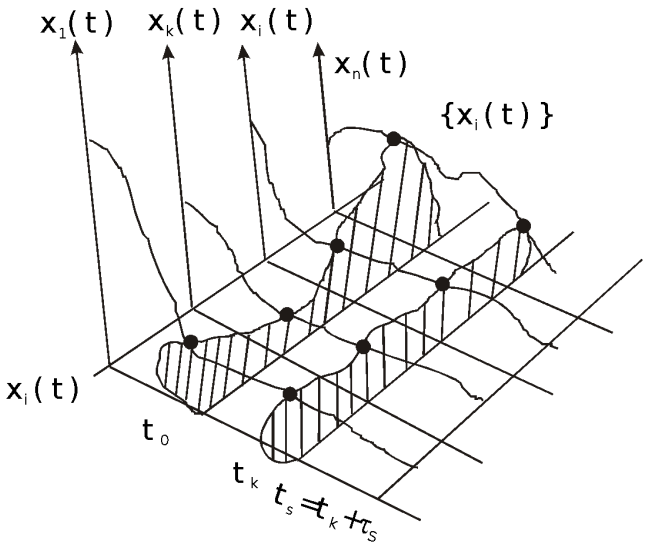
Для випадкового процесу його значення можна передбачити лише основні його характеристики: математичне сподівання $M_x(t)$, дисперсію $D_x(t)$, автокореляційну функцію $R_{xx}(t,\tau), R_{xy}(t,\tau)$.

Ці характеристики для випадкового нестаціонарного процесу теж є функціями часу, але вони детерміновані. Для оцінки цих характеристик використовуються СРВ, які повинні обробити значну кількість інформації; для отримання їх при нестаціонарному процесі необхідно мати безліч реалізацій цього процесу.

При наявності такого ансамблю реалізації можуть бути обчислені значення $M_x(t)$та інші для кожного конкретного часу t_k

Математичне сподівання $M_x(t)$для конкретного часу t_kвизначається першим початковим моментом, випадкової величини $x\left(t_k\right)$, ка називається перерізом випадкового процесу, її значення представлені у відповідному перерізі, усереднення проводиться по ансамблю:

$$M_x\left(t_к\right) = \lim \frac{1}{N} \cdot \sum_{i=1}^{N} x_i\left(t_k\right)$$



Аналогічним способом обчислюється і дисперсія $D_x(t)$, у якої конкретне t_k оцінюється 2-м центральним моментом у відповідності з $x\left(t_k\right)$.

Властивість ергодичності стаціонарного випадкового процессу

Багато досліджуваних випадкових процесів та сигналів є стаціонарними, тобто вони з плином часу не згасають і не розгойдуються, тобто можна виділити x_{max} і x_{min}, що є детермінантами. Випадковий процес $x(t)$називається стаціонарним, якщо його основні характеристики $M_x(t)$,$D_x(t)$,$R_{xx}(t,\tau)$ не залежать від часу їх зміни.

Для стаціонарного випадкового процесу $M_x = const$,$D_x = const$, а $R_{xx}(\tau)$ - залежить тільки від τ Для доказу того, що процес є стаціонарним зазвичай використовується вимірювання автокореляційної функції. Вона має вигляд:

$$R_{xx}(0) = D_x$$

Якщо $R_x(\tau) \to 0$, то це свідчить про те, що процес стационарен, має властивість ергодичності (інваріантності) або збереження енергії по відношенню до схеми обчислення його характеристик, тобто для стаціонарного сигналу можемо перейти при обчисленні характеристик від усереднення по ансамблю до усереднення за часом.

$$M_x = \lim_{N \to \infty} \frac{1}{N} \cdot \sum_{i=1}^{N} x_i(t_k) = \lim_{n \to \infty} \frac{1}{N} \sum_{k=0}^{n} x_i(t_k)$$

для $x(t_k)$перетину (одного перетину) в межах $x_i(t)$

(однієї $i - \text{той}$ реалізації)

$$D_x = \lim_{N \to 0} \cdot \frac{1}{N-1} \sum_{i=1}^{N} x_i(t_k) - M_x^{\ 2} = \lim_{n \to \infty} \frac{1}{n-1} \cdot \sum_{k=0}^{n}$$

в межах перетину $x(t_k)$ для однієї $x_i(t)$реалізації

Завдання

Мета роботи - ознайомлення з принципами генерації випадкових сигналів, вивчення та дослідження їх основних параметрів з використанням засобів моделювання і сучасних програмних оболонок.

Згенерувати випадковий сигнал по співвідношенню (див. нижче) відповідно варіантом по таблицею (Додаток 1) і розрахувати його математичне сподівання і дисперсію. Розробити відповідну програму і вивести отримані значення і графіки відповідних параметрів.

Генератор стаціонарного випадкового сигналу представлений як:

$$x^0(t) = \sum_{p=0}^{m} A_p \cdot sin(w_p \cdot t + \phi_p)$$

$p \to W_p$ - спектральні складові сигналу з частотою W_p, що змінюється від $p = 0, m$, W_m - верхня частотна складова; кількість складових від 6 до 10.

$A_p - random$ - амплітуда;

$\phi_p - random$- фаза.

Далі отриманий випадковий сигнал $x(t)$ представляється послідовністю дискретних відліків:

$x(t) \to \{x(t_k)\}, k = 0, \overline{N-1}$

$t \to t_k \to k \cdot \Delta t \to k$

Δt + вибирається як:

$$\Delta t = \frac{1}{k_{зап} \cdot f_{вн}} \qquad k_{зап} = 3 - 5$$

Варіант 1

Варіант	Число гармонік в сигналі n	Гранична частота, $\omega_{гр}$	Кількість дискретних відліків, N
1	6	1200	64

Програмний код

complexity.js

```
const { performance } = require('perf_hooks');
const { generateSignal } = require('./generateSignal');

const calculateTimeSignal = (harmonics, frequency, discreteCalls) => {

    const time = Array(discreteCalls).fill(0);

    return time.map((_, index) => {
        const startTime = performance.now();
        generateSignal(harmonics, frequency, index);
```

```
    const endTime = performance.now();

    return endTime - startTime;
  });
};
module.exports = { calculateTimeSignal };
```

generateSignal.js

```
'use strict';

const generateSignal = (harmonics, frequency, discreteCalls) => {
  const signals = Array(discreteCalls).fill(0);

  for (let i = 1; i <= harmonics; i++) {
    const wi = frequency / harmonics * i;
    const amplitude = Math.random();
    const phase = Math.random();
    for (let t = 0; t < discreteCalls; t++) {
      signals[t] += amplitude * Math.sin(wi * t + phase);
    }
  }
  return signals;
};

module.exports = { generateSignal };
```

statUtils.js

```
const mathExpectation = (arr = []) =>
  arr.reduce((acc, val) => (
```

```
    acc += val
  ), 0) / arr.length;

const mathDispersion = (mathExpectation, arr = []) =>
  arr.reduce((acc, val) => (
    acc += Math.pow(mathExpectation - val, 2)
  ), 0) / arr.length;



const correlation = (sig1, sig2) => {
  const len = sig1.length;
  if (len !== sig2.length) throw new Error();
  const mx1 = mathExpectation(sig1);
  const mx2 = mathExpectation(sig2);

  const result = [];
  for (let i = 0; i < len; i++) {
    const cr = (sig1[i] - mx1) * (sig2[i] - mx2);
    result.push(cr);
  }
  return result.reduce((acc, v) => acc + v, 0) / (len - 1);
};

const autoCorrelation = sig => {
  const mid = Math.floor(sig.length / 2);
  const sig_a = sig.slice(0, mid);

  const tauArr = [];
  const corrArr = [];
```

```
  for (let tau = 0; tau < mid; tau++) {
    const sig_b = sig.slice(tau, tau + mid);
    const corr = correlation(sig_a, sig_b);
    tauArr.push(tau);
    corrArr.push(corr);
  }

  return [tauArr, corrArr];
};

const crossCorrelation = (sig1, sig2) => {
  const mid = Math.floor(sig1.length / 2);
  const a = sig1.slice(0, mid);

  const tauArr = [];
  const corrArr = [];

  for (let tau = 0; tau < mid; tau++) {
    const b = sig2.slice(tau, tau + mid);
    const corr = correlation(a, b);
    tauArr.push(tau);
    corrArr.push(corr);
  }

  return [tauArr, corrArr];
};

module.exports = {
```

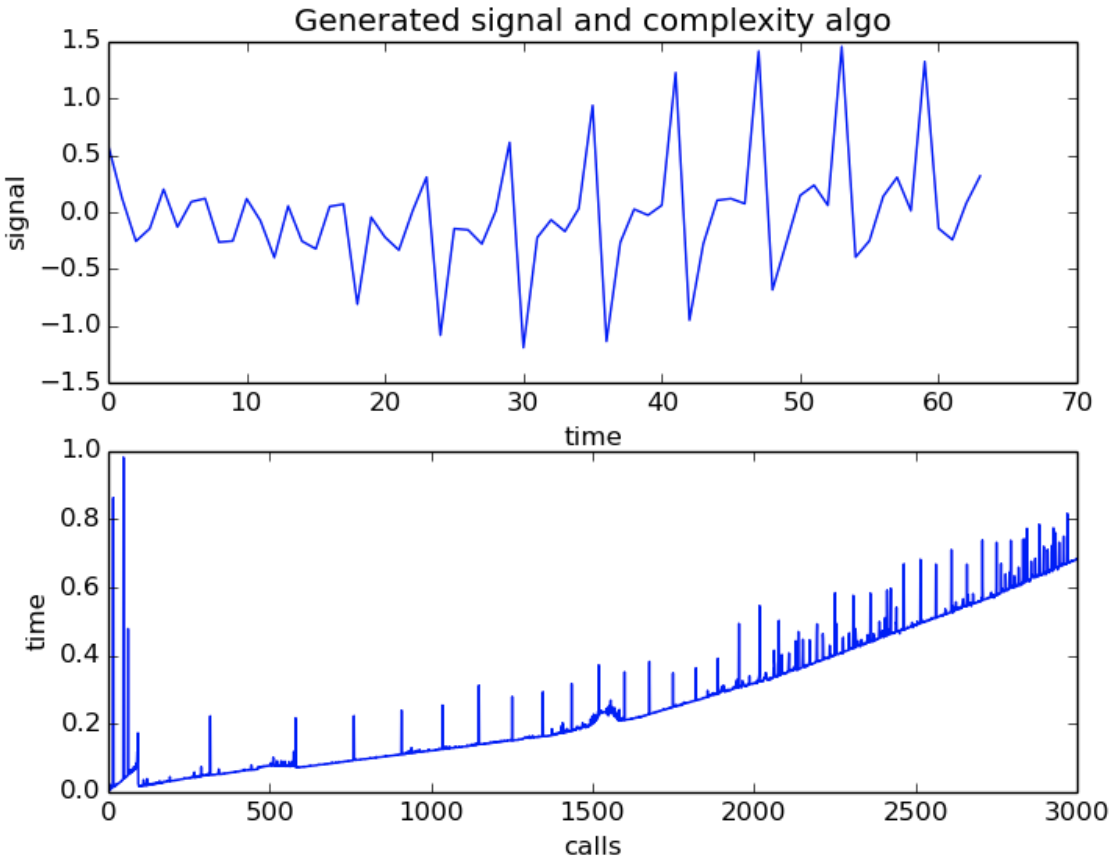
```
  mathExpectation,
  mathDispersion,
  autoCorrelation,
  crossCorrelation,
};
```

index.js

```
const { generateSignal } = require('../generateSignal');
const { mathExpectation, mathDispersion } = require('../statUtils');
const path = require('path');
const plt = require('matplotnode');
const { calculateTimeSignal } = require('../complexity');

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

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

const timeDiscretteCalls = 3000;

const time = calculateTimeSignal(
  harmonics,
  frequency,
  timeDiscretteCalls,
```

```
);


const expectation = mathExpectation(signal);


console.log('Math expectation = ' + expectation);

console.log('Math dispersion = ' + mathDispersion(expectation, signal));




plt.subplot('111');

plt.title('Generated signal and complexity algo');

plt.plot([...Array(discreteCalls).keys()], signal);

plt.xlabel('time');

plt.ylabel('signal');

plt.legend();


plt.subplot('112');

plt.plot([...Array(timeDiscretteCalls).keys()], time);

plt.xlabel('calls');

plt.ylabel('time');

plt.legend();


const currentDir = path.join(__dirname, '/1.1.png');

plt.save(currentDir);
```

Результати виконання програми




```
Math expectation = -0.014451229560399348
Math dispersion = 0.26360565478557213
```

Висновки

Під час виконання лабораторної роботи, було ознайомлено із принципами генерації випадкових сигналів, їх дослідження з використанням засобів моделювання. Було розроблено програму, що генерує сигнал з заданими параметрами та було досліджено його.