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 №1.2

з дисципліни
«Інтелектуальні вбудовані системи» на тему «Дослідження автокореляційної і взаємноюкореляційної функцій випадкових сигналів»

Виконав :

Студент групи ІП-84

Валигін Андрій Олександрович,
номер залікової книжки: 8503

Перевірив:

Регіда Павло Геннадійович

Київ 2021

Завдання

Для згенерованого випадкового сигналу з Лабораторної роботи N 1 відповідно до заданого варіантом (Додаток 1) розрахувати його автокореляційної функцію. Згенерувати копію даного сигналу і розрахувати взаімнокорреляціонную функцію для 2-х сигналів. Розробити відповідну програму і вивести отримані значення і графіки відповідних параметрів.

Теоретичні відомості

Значення автокореляційної функції фізично представляє зв'язок між значенням однієї і тієї ж величини, тобто для конкретних моментів t_k, τ_s, значення $R_{xx}(t,\tau)$ оцінюється друге змішаним центральним моментом 2-х перетинів випадкових процесів $x(t_k), x(t_k+\tau_s)$

$$R_{xx}(t,\tau_s)=\lim_{N\to\infty}\frac{1}{N-1}\sum_{i=1}^{N}(\overbrace{x_i(t_k)-M_x(t_k)}^{\overset{0}{x}(t_k)})\cdot(\overbrace{x_i(t_k+\tau_s)-M_x(t_k+\tau_s)}^{\overset{0}{x}(t_k+\tau_s)})$$

для кожного конкретного інтервалу потрібно проходити по всім t_k (перетинах).

Центральні значення можна замінити:

$$\overset{0}{x}(t_k),\ \overset{0}{x}(t_k,\tau_s),\ \text{тобто їх}\ M_x=0$$

$$\left[\begin{array}{l} R_{xx}(t,\tau)=\lim\limits_{N\to\infty}\dfrac{1}{N-1}\sum\limits_{i=1}^{N}\overset{0}{x}_i(t)\cdot\overset{0}{x}_i(t+\tau) \\ R_{xx}(t,\tau)=\lim\limits_{N\to\infty}\dfrac{1}{N-1}\sum\limits_{i=1}^{N}\overset{0}{x}_i(t)\cdot\overset{0}{x}_i(t+\tau) \end{array}\right]$$

Обчислення кореляційної функції $R_{xx}(t,\tau)$ є відносно складним, оскільки необхідно попереднє обчислення математичного очікування M_x для виконання кількісної оцінки, іноді виповнюється ковариационной функцією:

$$C_{xx}(t,\tau)=\lim_{N\to\infty}\frac{1}{N-1}\sum_{i=1}^{N}x_i(t)\cdot x_i(t+\tau)$$

У завданнях управління частіше використовується нормована кореляційна функція:

$$S_{xx}(t,\tau)=\frac{R_{xx}(t,\tau)}{D_x(t)}<1$$

Варіант №3

Число гармонік в сигналі: 8

Гранична частота, ωгр: 1100

Кількість дискретних відліків, N: 256

Лістинг програми

```
const option = {
    maintainAspectRatio: false,
    elements: {
        point: {
            radius: 0
        }
    },
    scales: {
        yAxes: [{
            stacked: true,
            gridLines: {
                display: true,
                color: "rgba(255,99,132,0.2)"
            }
        }],
        xAxes: [{
            gridLines: {
                display: false
            }
        }]
    }
};


const conf = (name, len, arr) => ({
    type: 'line',
```

```
    data: {
        labels: new Array(len).fill(0).map((_, i) => i),
        datasets: [{
            lineTension: 0,
            label: name,
            backgroundColor: window.chartColors = '#fff',
            borderColor: window.chartColors = '#07c',
            borderWidth: 2,
            fill: false,
            data: arr,
        }]
    },
    options: option,
})


const newGraphic = (color, name, arr) => ({
    lineTension: 0,
    label: name,
    backgroundColor: window.chartColors = '#fff',
    borderColor: window.chartColors = color,
    borderWidth: 2,
    fill: false,
    data: arr,
})


const harmonicsAmount = 8
const frequency = 1100
const N = 256
```

```
const createArray = len => new Array(len).fill(0)

const average = arr => arr.reduce((a, c) => a + c) / arr.length

const mathAverageTrick = arr => Math.sqrt(average(arr.map(num => Math.pow((num - average(arr)), 2))))



const main = () => {

    let array = createArray(N)

    const harmArray = createArray(harmonicsAmount)

    harmArray.forEach((_, i) => array = array.map((item, j) => item + (Math.random() * Math.sin(((frequency / harmonicsAmount) * (i + 1)) * j + Math.random()))))

    return array

}



const correlation = (sig1, sig2) => {

    const a1 = average(sig1)

    const a2 = average(sig2)

    const mat1 = mathAverageTrick(sig1)

    const mat2 = mathAverageTrick(sig2)

    const len = parseInt(sig1.length / 2)

    const array = createArray(len)

    return array.map((_, i) => (array.reduce((a, _, j) => a + (sig1[j] - a1) * (sig2[i + j] - a2) / (len - 1))) / (Math.sqrt(mat1) * Math.sqrt(mat2)))

}



const autoCorrelation = sig => correlation(sig, sig)



//Usage

window.onload = function() {

    const ctx = document.querySelector('#myChart').getContext('2d')
```

```
    const ctx2 = document.querySelector('#secChart').getContext('2d')
    const ctx3 = document.querySelector('#thirdChart').getContext('2d')


    const sig1 = main()
    const sig2 = main()


    const graphic = new Chart(ctx, conf('1.0', N, sig1))
    console.log('Середнє значення', average(sig1))
    console.log('Дисперсія', mathAverageTrick(sig1))


    const secConf = conf('2.0', N / 2, autoCorrelation(sig1))
    secConf.data.datasets.push(newGraphic('red', '2.1', sig1))
    const graphic2 = new Chart(ctx2, secConf)



    const thirdConf = conf('3.0', N / 2, correlation(sig1, sig2))
    thirdConf.data.datasets.push(newGraphic('red', '3.1', sig1))
    thirdConf.data.datasets.push(newGraphic('orange', '3.2', sig2))
    const graphic3 = new Chart(ctx3, thirdConf)
}
```

Скріншоти

https://drewmeltpool.github.io/realTimeSys/lab1/

Висновок

Ознайомився з принципами побудови автокореляційної функці та кореляційної функції. Вивичив та дослідив їх основні параметри за допомогою JavaScript. Побудував відповідні графіки та написав програму.