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
Кабір Лабіб Ахмед
№ залікової книжки: ІП-8416

Перевірив:
викладач
Регіда П.Г.

Київ 2021

Теоретичні відомості

2.1. Основні 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:

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

Дослідження нестандартних випадкових сигналів вимагає значних обсягів пам'яті, тому в більшості наукових досліджень приймається гіпотеза про стаціонарності випадкового сигналу на інтервалі ($t_0 \dots t_1$).

Кореляційна функція для стаціонарного сигналу:

$$R_x(\tau_s) = \lim_{N \to 0} \cdot \frac{1}{N-1} \cdot \sum_{i=1}^{N} \underbrace{(x_i(t_k) - M_x)}_{X(t_k)} \cdot \underbrace{(x_i(t_k + \tau_s) - M_x)}_{x(t_s)} =$$

$$= \lim_{n \to 0} \cdot \frac{1}{n-1} \cdot (x_i(t_k) - M_x) \cdot (x_i(t_k + \tau_s) - M_x)$$

$x(t)$ в межах однієї реалізації показує наскільки швидко змінюється сигнал.

Коваріаційна функція для стаціонарного сигналу:

$$C_{xx}(\tau) = \lim_{N \to 0} \cdot \frac{1}{n-1} \cdot \sum_{k=1}^{n} Lx(t_k) \cdot x(t_k + \tau)$$

показує ступінь зв'язності між значеннями одного і того ж сигналу.

Таким чином для стаціонарних і ергодичні процесів обчислення параметрів сигналів реалізуються шляхом усереднення за часом у межах однієї реалізації.

Статистичне вимірювання зв'язків між двома стаціонарними випадковими процесами

Дуже важливим виявляється не тільки обчислення автокореляційної функції $R_{xx}(\tau)$, але і обчислення взаємної кореляційної функції $R_{xy}(\tau)$ для двох випадкових процесів $x(y)$, $y(t)$, для якої не можна на основі зовнішнього спостереження сказати, чи є залежність між ними. Для розрахунку взаємної кореляційної функції:

$$R_{xy}(\tau) = \lim_{n \to 0} \cdot \frac{1}{n-1} \cdot \sum_{i=1}^{n} \underbrace{(x_i(t_k) - M_x)}_{X(t_k)} \cdot \underbrace{(y(t_k + \tau) - M_y)}_{y(t_k - \tau)} =$$

τ - випробувальний інтервал, на конкретному значенні якого досліджується взаємний вплив.

Умови завдання

Варіант :16

n = 12, ω гр = 900, N = 256

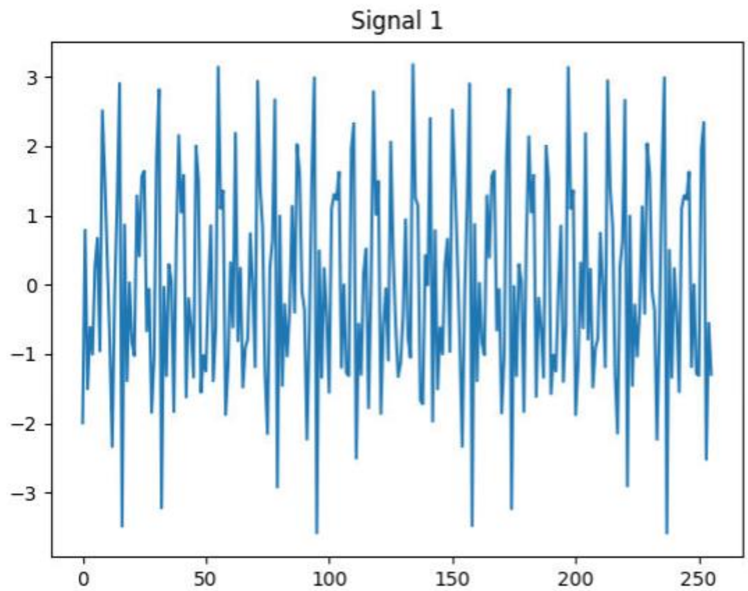
Вихідний код

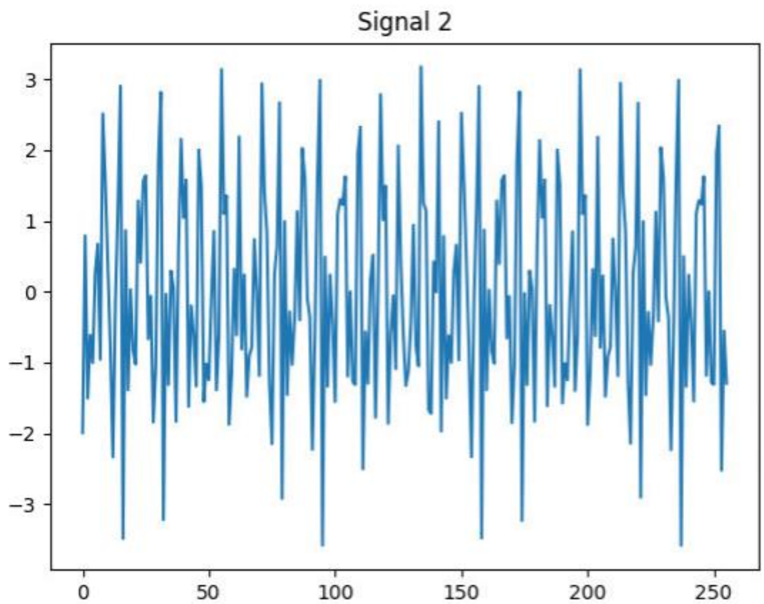
```
import numpy as np
import matplotlib.pyplot as plotter
n = 12
w = 900
N = 256
def generate_signal(amplitude, phase, frequency, time):
    return amplitude * np.sin(frequency * time + phase)  # Press Ctrl+F8 to toggle the breakpoint.
def signal_generator(harmonics, frequency_max, samples_amount):
    signal_collection = np.zeros(samples_amount)
    for f in range(1, harmonics + 1):
        phase = np.random.uniform((-np.pi / 2), (np.pi / 2))
```

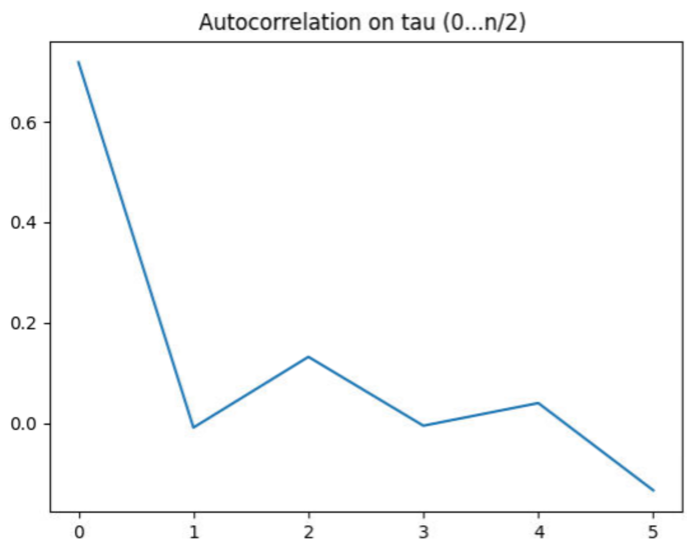
```
        amplitude = np.random.uniform(0, 1)
        frequency = (f * frequency_max) / harmonics
        for time in range(samples_amount):
            signal_collection[time] += generate_signal(amplitude, phase, frequency, 
time)
    return signal_collection
def  autocorrelation(signal, samples_amount, t_shift):
    collection = np.zeros(samples_amount - t_shift)

    for f in range(samples_amount - t_shift):
        collection[f] = signal[f] * signal[f + t_shift]
    expected_value = np.average(signal)
    sigma= np.sqrt(np.var(signal))
    covariation = np.average(collection) - expected_value * expected_value
    return covariation / sigma / sigma
def  autocorrelation_tau(signal, samples_amount):
    collection = np.zeros(int(samples_amount / 2))
    for tau in range(int(samples_amount / 2)):
      collection[tau] = autocorrelation(signal, samples_amount, tau)
    return collection
def  correlation_t(signal1, signal2, samples_amount, t_shift):
    collection = np.zeros(samples_amount - t_shift)

    for f in range(samples_amount - t_shift):
        collection[f] = signal1[f] * signal2[f + t_shift]
    covariation = np.average(collection) - np.average(signal1) *np.average(signal2)
    return covariation / np.sqrt(np.var(signal1)) / np.sqrt(np.var(signal2))
def  correlation_tau(signal1, signal2, samples_amount):
    collection = np.zeros(int(samples_amount / 2))
    for tau in range(int(samples_amount / 2)):
        collection[tau] = correlation_t(signal1, signal2, samples_amount, tau)
    return collection
signal = signal_generator(n, w, N)
signal_2 = signal_generator(n, w, N)
expected_value = np.average(signal)
variance = np.var(signal)
sigma= np.sqrt(variance)
autocorrelation_t = autocorrelation(signal, n, 7)
expected_value_2 = np.average(signal_2)
variance_2 = np.var(signal_2)
sigma_2= np.sqrt(variance_2)
# correlation = correlation(signal, signal_2, n)
plotter.plot(signal)
plotter.title('Signal 1')
plotter.show()
print('Expected Value for Signal 1 is', np.average(signal))
print('Variance for Signal 1 is', np.var(signal))
print('Autocorrelation for Signal 1 is', autocorrelation_t)
plotter.plot(signal)
```

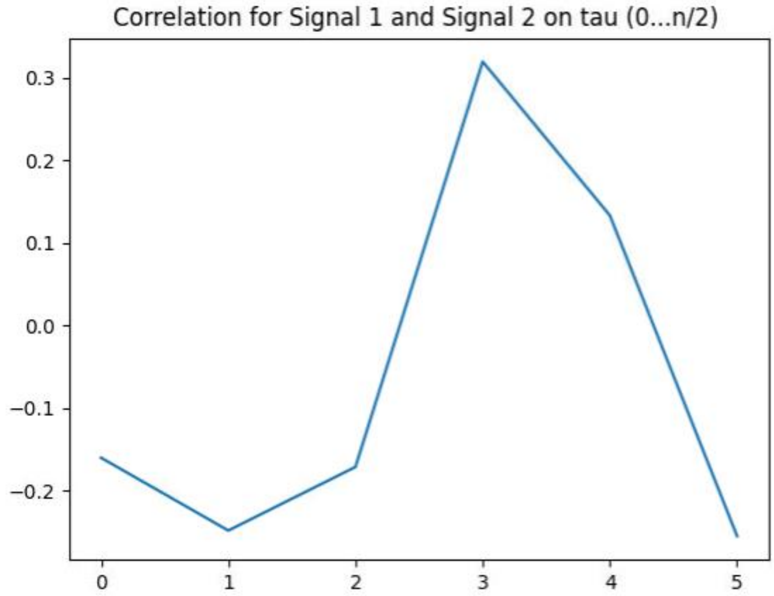
```
        plotter.title('Signal 2')
        plotter.show()
        print('Expected Value for Signal 2 is', expected_value_2)
        print('Variance for Signal 2 is', variance_2)
        autocorrelation_eachtau = autocorrelation_tau(signal, n)
        plotter.plot(autocorrelation_eachtau)
        plotter.title('Autocorrelation on tau (0...n/2) ')
        plotter.show()
        correlation_eachtau = correlation_tau(signal, signal_2, n)
        plotter.plot(correlation_eachtau)
        plotter.title('Correlation for Signal 1 and Signal 2 on tau (0...n/2) ')
        plotter.show()
```

Результати роботи програми


Signal 1
3
2
1
0
−1
−2
−3
0
50
100
150
200
250


Signal 2
3
2
1
0
−1
−2
−3
0
50
100
150
200
250


Autocorrelation on tau (0...n/2)
0.6
0.4
0.2
0.0
0
1
2
3
4
5



Висновки

У ході виконання лабораторної роботи ми ознайомилися з принципами побудови автокорелляційної і взаємної кореляційної функцій, вивчили та дослідили їх основні параметри з використанням засобів моделювання та сучасних програмних оболонок.