Міністерство освіти і науки України Національний технічний університет України "Київський політехнічний інститут ім. Ігоря Сікорського" Факультет інформатики та обчислювальної техніки Кафедра обчислювальної техніки

Лабораторна робота 1.2
з дисципліни «Інтелектуальні вбудовані системи»

Виконав:

студент групи ІП-83

Подаш А.М.

Перевірив:

асистент Регіда П.Г.

Київ 2021

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:

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

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

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 ($t_0 \ldots t_1$).

<u>Кореляційна функція для стаціонарного сигналу:</u>

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

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

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

<u>Коваріаційна функція для стаціонарного сигналу:</u>

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

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.

<u>Статистичне вимірювання зв'язків між двома стаціонарними випадковими процесами</u>

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.

2.2. Завдання на лабораторну роботу

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.

Варіант №18

Число гармонік в сигналі n - 10

Гранична частота, ωгр - 1500

Кількість дискретних відліків, N - 256

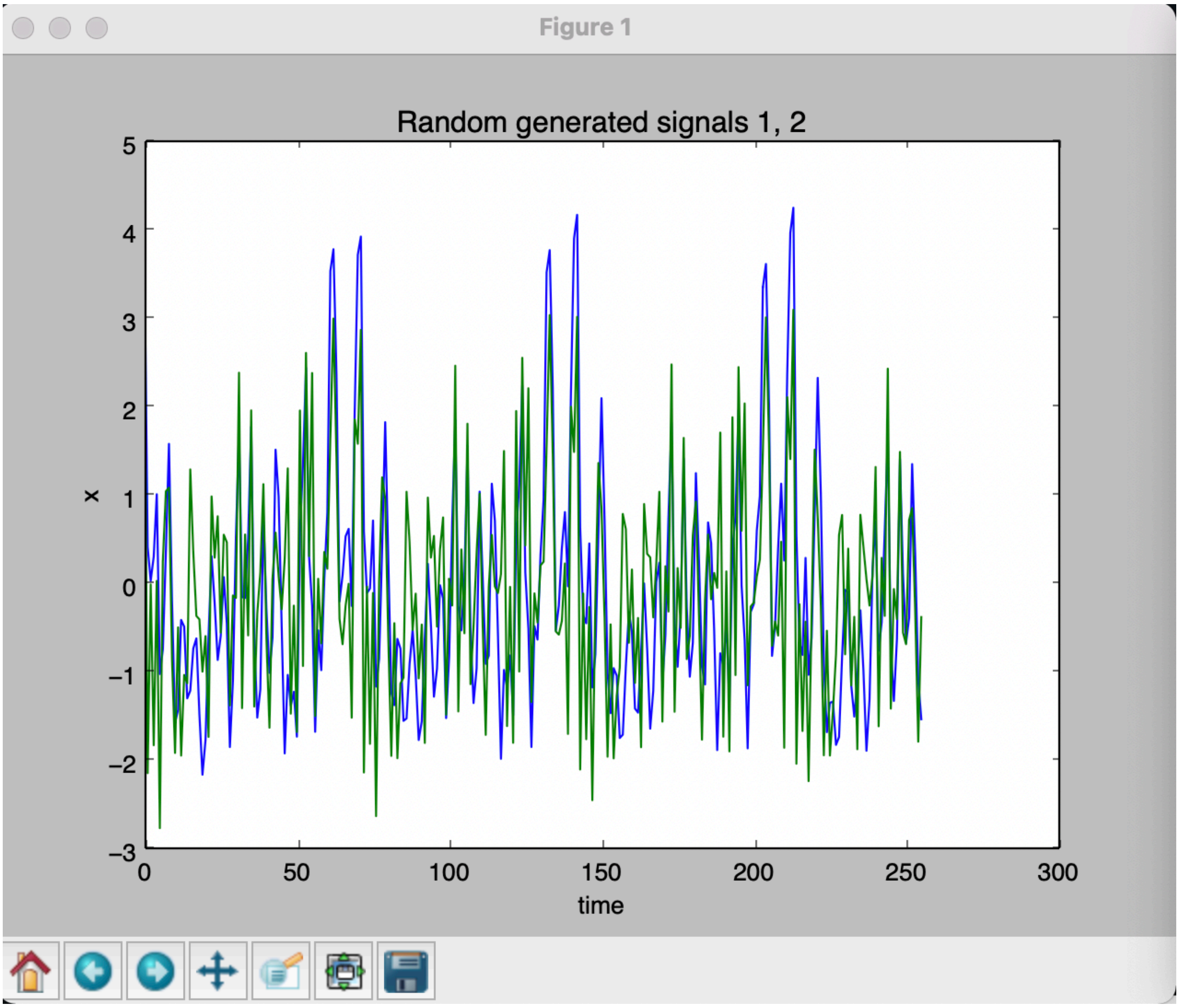
Лістинг програми

```
import matplotlib.pyplot as plt  # lib for graphs
import numpy as np  # lib for math operations

# constants
n = 10  # number of harmonics
w = 1500  # max frequency
N = 256  # number of descrete calls

# function for calculating random signal
def formula(a, w, t, phi):
    return a*np.sin(w*t+phi)

# function for generation array of signals
def generateSignals(n, w, N):
    signals = [0]*N  # array of signals
    w0 = w/n  # frequency
    for _ in range(n):
        a = np.random.rand()  # amplitude
        phi = np.random.rand()  # phase

        for t in range(N):
            signals[t] += formula(a, w0, t, phi)
        w0 += w0
    return signals

# correlation function
def correlation(signal1, signal2):
```

```
    Mx1 = np.average(signal1)  # math expectation
    Mx2 = np.average(signal2)  # math expectation
    sd1 = np.std(signal1)  # standart deviation == sqrt(dispersion)
    sd2 = np.std(signal2)  # standart deviation == sqrt(dispersion)
    length = len(signal1) // 2
    res = []

    for t in range(length):
        covarience = 0

        for l in range(length):
            covarience += (signal1[l]-Mx1)*(signal2[l + t]-Mx2) / (length-1)

        res.append((covarience / sd1 * sd2))

    return res

# autocorrelation function
def autocorrelation(signal):
    return correlation(signal, signal)

signals = generateSignals(n, w, N)
signals_copy = generateSignals(n, w, N)

print('Mx:', np.average(signals))  # Average
print('Dx:', np.var(signals))  # Dispersion

# plotting

# signals
plt.plot(signals)
plt.plot(signals_copy)
plt.xlabel('time')
plt.ylabel('x')
plt.title('Random generated signals 1, 2')
plt.figure()
```

```
# cross-correlation
plt.plot(correlation(signals, signals_copy))
plt.xlabel('time')
plt.ylabel('correlation')
plt.title('cross-correlation')
plt.figure()

# autocorrelation
plt.plot(autocorrelation(signals))
plt.xlabel('time')
plt.ylabel('correlation')
plt.title('autocorrelation')
plt.show()
```

Результат роботи програми



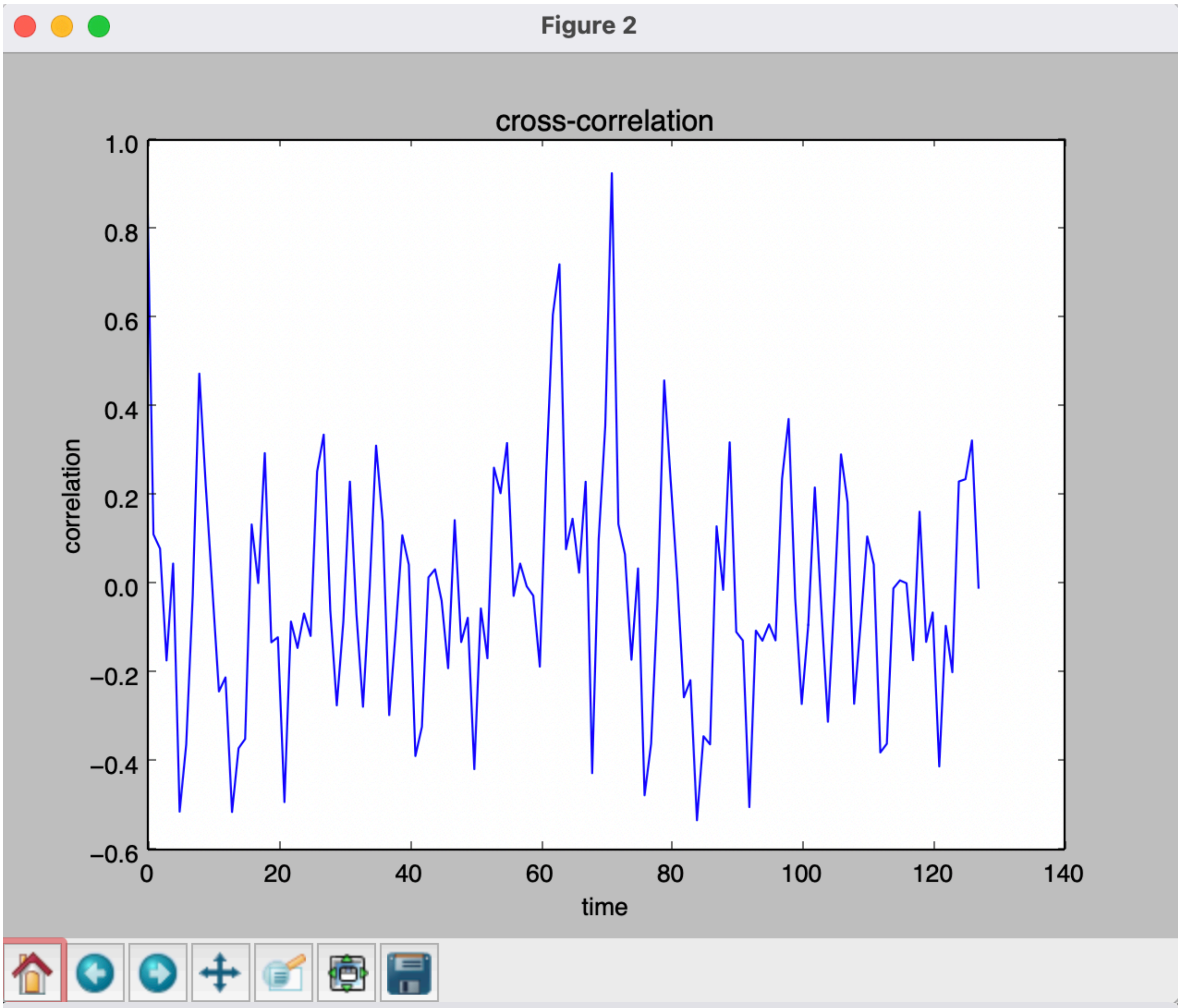


Figure 2
cross-correlation
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
correlation
0
20
40
60
80
100
120
140
time

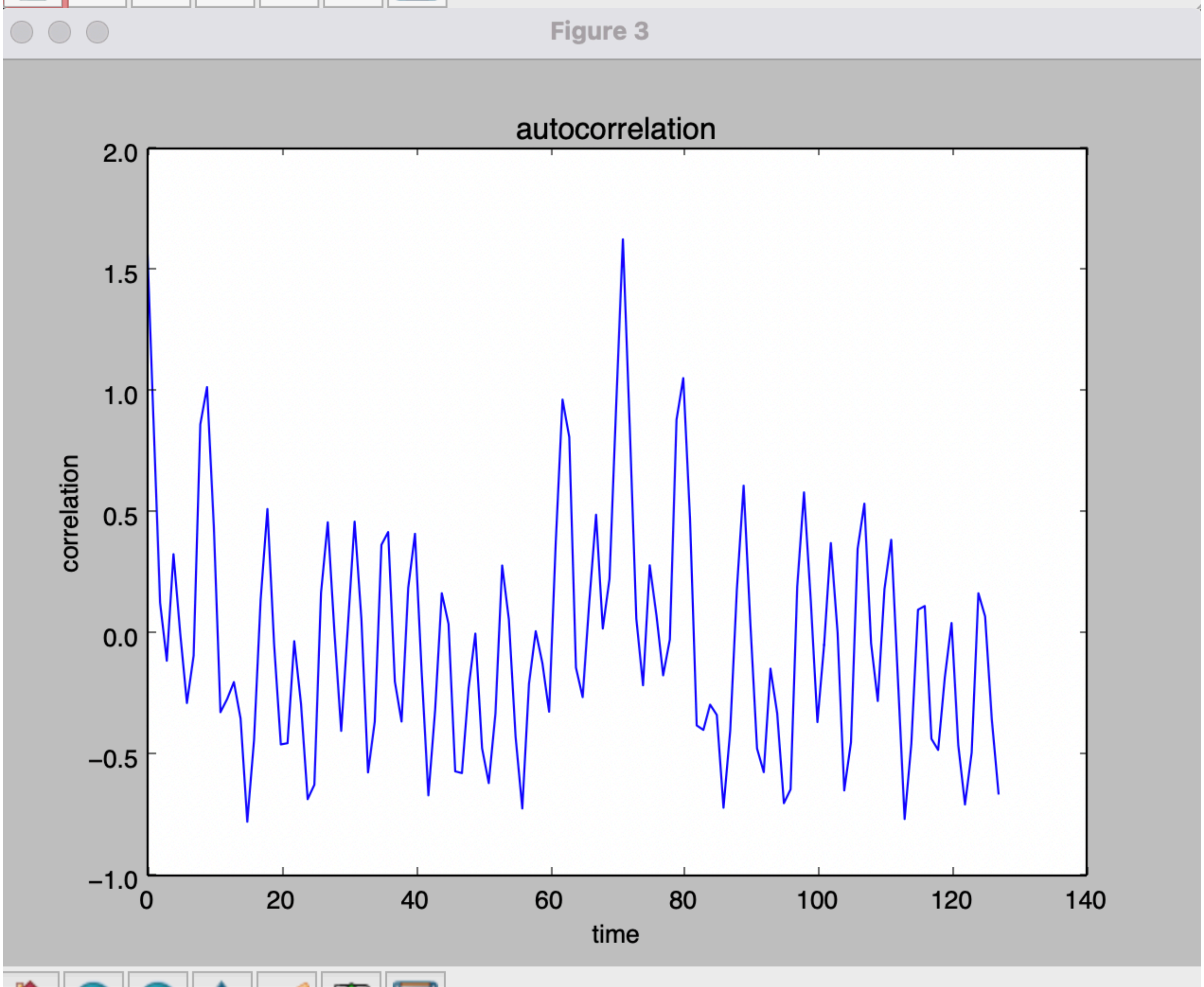


Figure 3
autocorrelation
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
correlation
0
20
40
60
80
100
120
140
time

```
MacBook-Pro:lab_1-2 antonpodas$ python lab_1-2.py
('Mx:', -0.031656775123002932)
('Dx:', 1.9521879103405422)
```

Висновки

Під час виконання лабораторної роботи, дослідив методи обчислення кореляції. Реалізована програма на мові Python.