**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 взаємно-

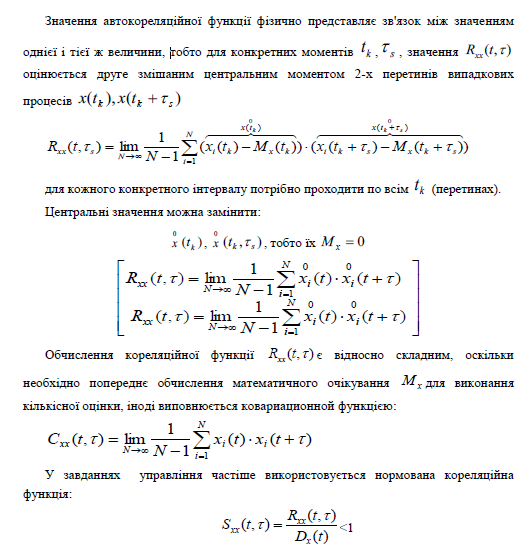
кореляційної функцій випадкових сигналів»

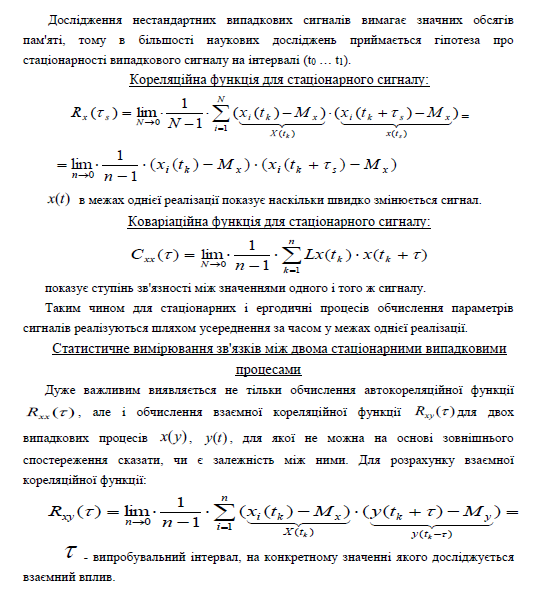
Виконав: Перевірив:

студент групи ІП-84 викладач  
Кучін Владислав Дмитрович Регіда Павло Геннадійович  
номер залікової книжки: 8415

Київ 2021

**Основні теоретичні відомості**





**Завдання на лабораторну роботу**

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

**Варіант-15**

Число гармонік в сигналі: 14.

Гранична частота: 2500.

Кількість дискретних відліків: 64

**Лістинг програми**

import numpy as np

import matplotlib.pyplot as plp

W\_MAX = 2500

HARMONICS\_QUANTITY = 14

DISCRETE\_SAMPLES\_NUMBER = 64

def calculate\_value(amplitude, phase, frequency, time):

return amplitude \* np.sin(frequency \* time + phase)

def create\_signal(w\_max, harmonics\_quantity, discrete\_samples\_number):

sum\_values = np.zeros(discrete\_samples\_number)

for w in range(1, harmonics\_quantity + 1):

phase = np.random.uniform(-np.pi / 2, np.pi / 2)

amplitude = np.random.uniform(0.0, 1000.0)

for t in range(discrete\_samples\_number):

sum\_values[t] += calculate\_value(amplitude, phase, w\_max / harmonics\_quantity \* w, t)

return sum\_values

def create\_ccr(signal, copy\_signal, max\_step):

auto\_correlation\_values = np.zeros(max\_step)

math\_expectation = np.average(signal)

std = np.std(signal)

for i in range(max\_step):

value = 0

for j in range(len(signal) - i):

value += signal[j] \* copy\_signal[j + i]

offset\_math\_expectation = np.average(copy\_signal[i:])

offset\_std = np.std(copy\_signal[i:])

auto\_correlation\_values[i] = (value / (len(signal) - i) - math\_expectation \* offset\_math\_expectation) / \

(std \* offset\_std)

return auto\_correlation\_values

def create\_acr(signal, max\_step):

return create\_ccr(signal, signal, max\_step)

SIGNAL = create\_signal(W\_MAX, HARMONICS\_QUANTITY, DISCRETE\_SAMPLES\_NUMBER)

COPY\_SIGNAL = create\_signal(W\_MAX, HARMONICS\_QUANTITY, DISCRETE\_SAMPLES\_NUMBER)

acr = create\_acr(SIGNAL, 32)

ccr = create\_ccr(SIGNAL, COPY\_SIGNAL, 32)

plp.subplot(221)

plp.plot(SIGNAL)

plp.title('Згенерований сигнал')

plp.subplot(222)

plp.plot(COPY\_SIGNAL)

plp.title('Копія сигналу')

plp.subplot(223)

plp.plot(acr)

plp.title('Автокореляційна функція')

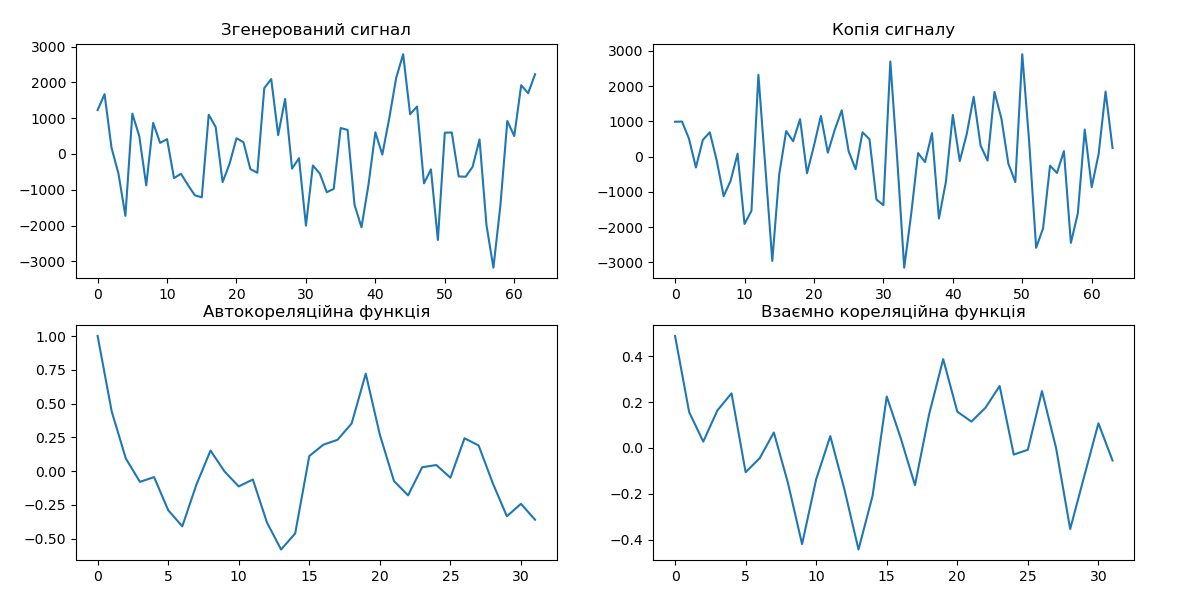
plp.subplot(224)

plp.plot(ccr)

plp.title('Взаємно кореляційна функція')

plp.show()

**Результати роботи програми**



**Висновки**

Під час даної лабораторної роботи ми вивчили, що таке автокореляційна та взаємно кореляційна функції, вирахували кожну з них на попередньо згенерованих сигналах.