КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет радіофізики,електроніки та комп'ютерних систем

Кафедра квантової радіофізики та наноелектроніки

Ю.В. Пустовіт

МЕТОДИЧНІ ВКАЗІВКИ З КУРСУ "ЛАБОРАТОРІЯ З ОПТИЧНОЇ, МІКРОХВИЛЬОВОЇ ТА ЦИФРОВОЇ ТЕХНІКИ"

Київ – 2025

Методичні вказівки з курсу “Лабораторія з оптичної мікрохвильової та цифрової техніки”/ Пустовіт Ю.В. – Київ: Видавнича лабораторія факультету радіофізики, електроніки та комп’ютерних систем Київського національного університету імені Тараса Шевченка, 2025. – 30 с

Рецензенти: кандидат-фізико-математичних наук, асистент кафедри комп’ютерної інженерії, факультету радіофізики, електроніки і комп’ютерних систем Коновалов А. М.;
доктор філософії (PhD) зі спеціальності 105 Прикладна фізика та наноматеріали, асистент кафедри квантової радіофізики Науково-навчального інституту високих технологій Чумак Г.Л.

Посібник складений відповідно до навчальної програми курсу «Лабораторія з оптичної мікрохвильової та цифрової техніки», що викладається на факультеті радіофізики, електроніки та комп’ютерних систем. В посібнику наведено опис трьох лабораторних робіт. Опис кожної роботи містить мету, теоретичні відомості з прикладами програм, завдання до роботи, методичні вказівки щодо виконання роботи та контрольні питання для самостійної роботи студентів.

Ухвалено Вченою радою факультету радіофізики,
електроніки та комп’ютерних систем
Київського національного університету імені Тараса Шевченка
(протокол №10 від 26 травня 2025 р.)

Зміст

Лабораторна робота №1
“Використання бібліотеки matplotlib для візуалізації експериментальних даних”

Мета роботи: Ознайомитись з можливостями Google Colab та бібліотеки matplotlib для візуалізації експериментально одержаних даних.

Теоретичні відомості

Бібліотека matplotlib - бібліотека на мові програмування Python створена для візуалізації даних. Використовуючи **matplotlib** можна створювати високоякісні рисунки різних форматів. Matplotlib є гнучким, легко конфігурованим пакетом, який разом з NumPy, SciPy і IPython надає можливості подібні до MATLAB. Перевагою є безкоштовність, сумісність з іншими бібліотеками Python та простота використання.

Важливою концепцією у бібліотеці matplotlib є концепція ієрархії об'єктів. “Ієрархія” у нашому випадку означає, що існує певна деревовидна структура об'єктів бібліотеки matplotlib, які лежать в основі побудови кожного графіка

Найвищий в ієрархії об'єктів - об'єкт Figure контейнер для графіки, який може вміщувати багато об'єктів Axes (позначає окремий графік або діаграму). Далі знаходяться менші об'єкти, такі як окремі лінії, легенда, текст та підписи вісей (рис.1.1). Майже будь який елемент графіку є об'єктом, яким можна керувати.

Інтерфейс **matplotlib.pyplot** дозволяє суттєво спростити роботу з об'єктами та використовувати для побудови графіків високорівневі функції.

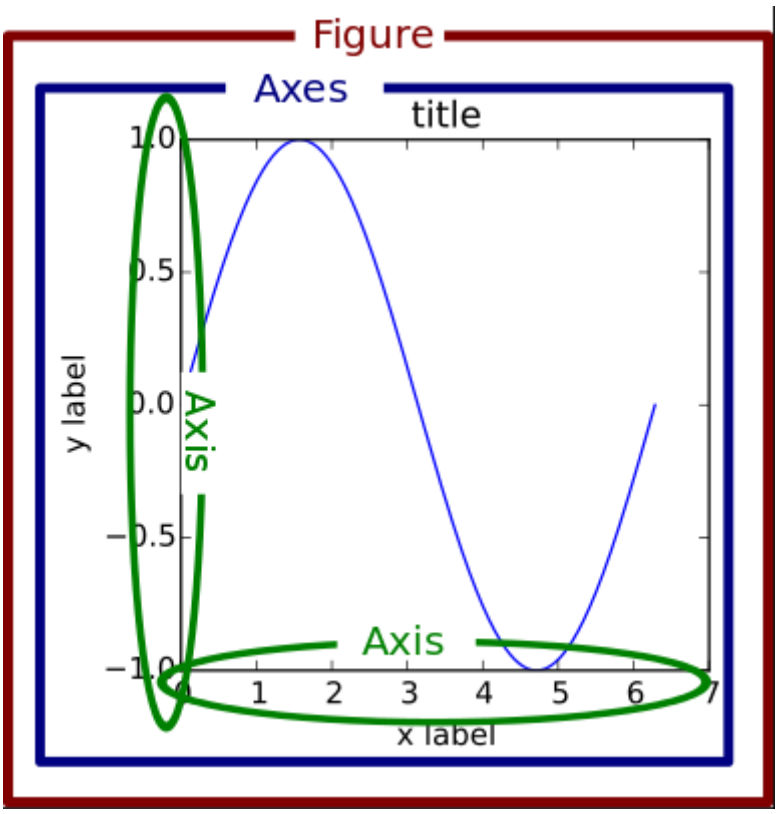


Рис.1.1. Ієрархія об'єктів в mathplotlib

Бібліотека numpy

У мові програмування Python не реалізовані масиви, тому для роботи з масивами використовується бібліотека **numpy** для підключення бібліотеки слід використовувати

```
import numpy as np
```

Для доступу до окремих елементів двовимірного масиву **arr** можна використовуємо ***arr[i,j]***

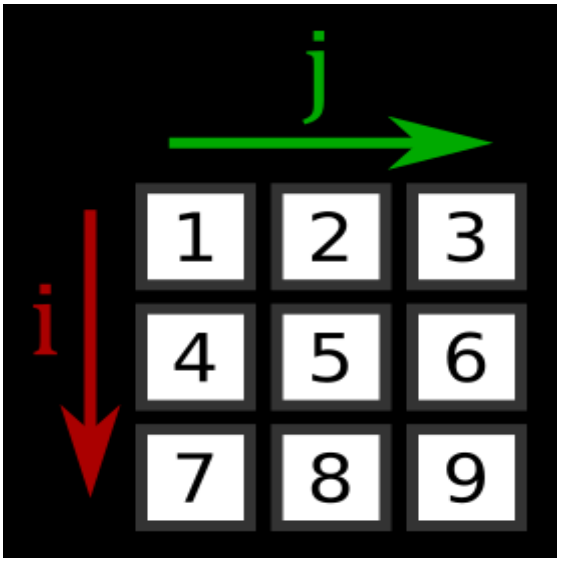


Рис.1.2. представлення 2 вимірного масиву

Приклад

```
import numpy as np
a2 = np.array([[1, 2, 3], # створюємо масив розмірністю 3 на 3
               [4, 5, 6],
               [7, 8, 9]])
a2[1,2] # поверне 6
```

Перерізи масивів

Можливим є створення масивів меншої розмірності використовуючи перерізи

Приклад

```
print(a2[2]) # [7, 8, 9] поверне масив меншої розмірності, що відповідає 3ьому рядку матриці
```

Для виділення стовпчиків слід використовувати **:** - вибирає всі елементи по даній розмірності

```
print(a2[:, 1])  # поверне [2, 5, 8]
```

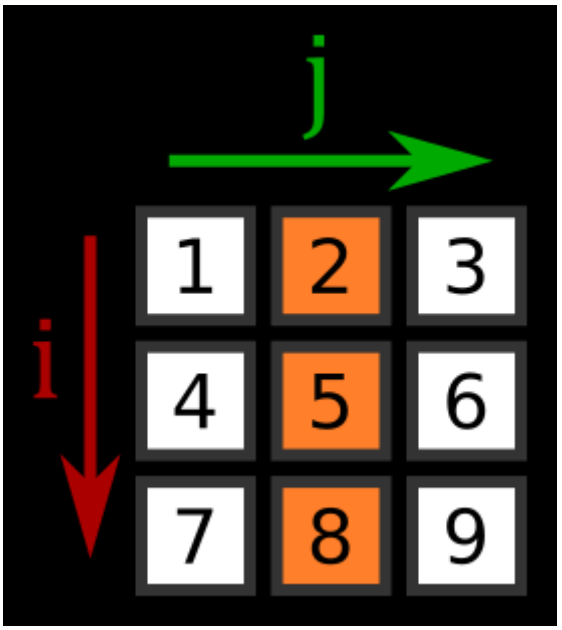


Рис.1.3. Результат зрізу `arr[:, 1]`

Також можливо використовувати **:** для виділення проміжків, зокрема, `arr[start_point:end_point:step]`, де `start_point` - початкова точка, `end_point` - кінцева точка, `step` - крок, за замовчуванням дорівнює 1

Робота з matplotlib

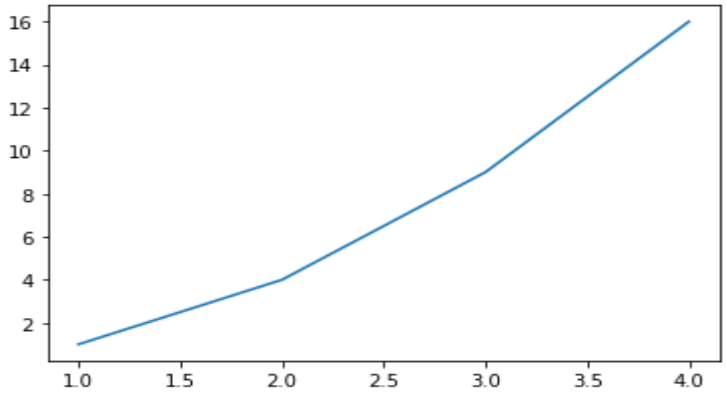
Для початку роботи слід підключити інтерфейс **pyplot**

```
import matplotlib.pyplot as plt
```

для побудови графіка по точкам можна використати процедуру **pyplot.plot(X,Y)** , де X - масив, який задає шкалу, Y -відповідне значення для кожної точки X

Приклад

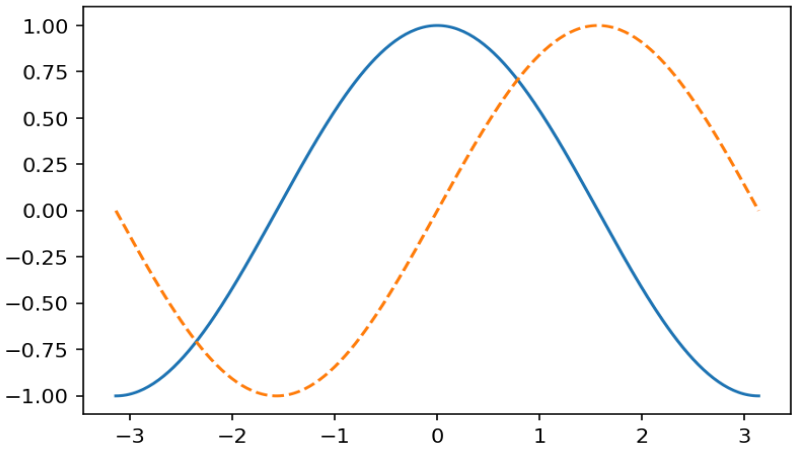
```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16])
plt.show()
```



Оскільки, найчастіше доводиться працювати з великими масивами даних для роботи з ними слід використовувати бібліотеку NumPy.

Приклад використання та результат роботи коду:

```
import numpy as np (підключаємо бібліотеку numpy)
matplotlib.pyplot.figure(dpi=200)/ обираємо якість зображення
X = np.linspace(-np.pi, np.pi, 256)/ функція повертає масив рівновіддалених точок кількістю 256, в межах між -п,п
C, S = np.cos(X), np.sin(X)(будуємо функцію cos та sin при цьому np.cos, np.sin - векторизовані функції, що рахують значення sin cos для кожної точки масива)
plt.plot(X,C)
plt.plot(X,S, '--',color='purple', linewidth=10)( )
plt.show()
```



Для створення більш інформативних графіків та їх модифікації слід використовувати наступні функції:

pyplot.xlabel (“підпис”) задає підпис вісі x

pyplot.ylabel (“підпис”) задає підпис вісі y

pyplot.axis([x_min, x_max, y_min, y_max,]) - задає діапазон значень, що відображається на рисунку

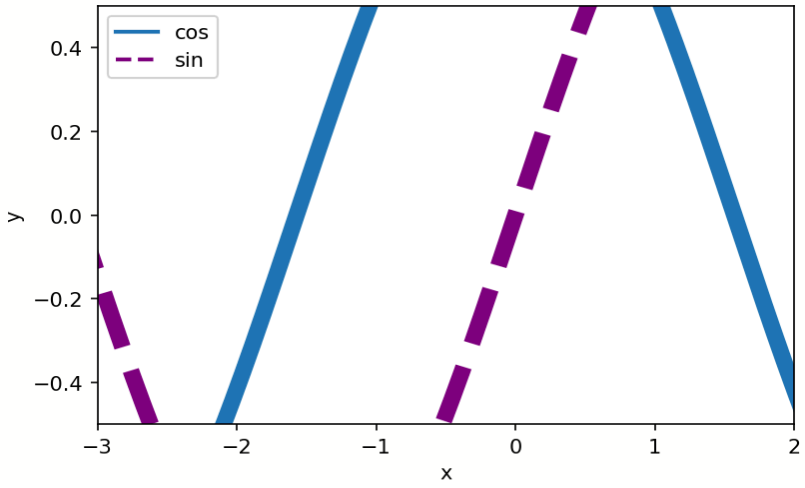
pyplot.legend() - додаємо легенду до рисунка, у цьому слід дещо змінити функцію **plot**(.... label='позначення')

Також за потреби розмістити на одному рисунку декілька графіків можна змінювати параметри кожного окремого графіка (ширина, колір та вид лінії, наявність чи відсутність маркерів таке інше) в функції **plot**.

Приклад коду та результати його виконання :

```
import numpy as np
plt.figure(dpi=150) / задаємо якість рисунку
X = np.linspace(-np.pi, np.pi, 256)
C, S = np.cos(X), np.sin(X)
plt.xlabel ('x')
plt.ylabel ('y')
plt.axis([-3, 2, -0.5, 0.5,])
plt.plot(X,C, label='cos', linewidth=8)
plt.plot(X,S, '--', label='sin',color='purple',
linewidth=8)/ змінили ширину лінії, колір да додали ярлик
для легенди
leg = plt.legend()
```

```
leg.get_lines()[0].set_linewidth(2)
leg.get_lines()[1].set_linewidth(2)
plt.show()
```



Для комбінування декількох рисунків одним з варіантів є використання **pyplot.subplots()**. Можливим є кілька підходів до створення композиції декількох рисунків:

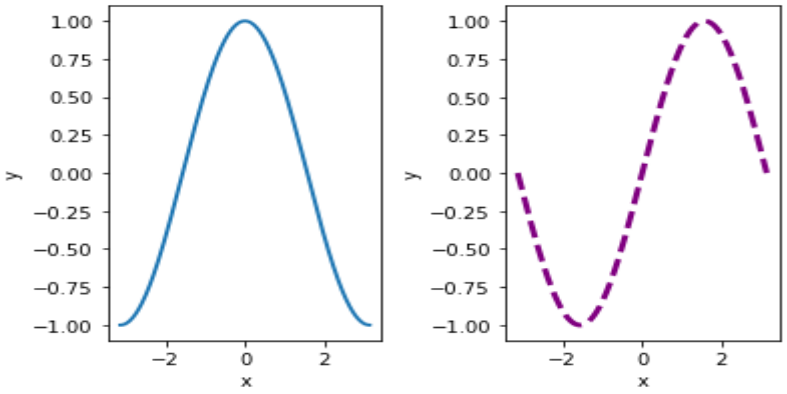
1)створювати об’єкти класу Axes та додавати їх до раніше створеного об'єкту Figure функцією figure.add_subplot(кількість рядків, кількість стовпців, номер рисунку), при цьому нумерація рисунків відбувається з лівого верхнього кута по рядкам. Після цього окремо модифікувати об’єкт класу Axes для отримання потрібного результату.

Приклад коду та результати його виконання

```
fig = plt.figure()
X = np.linspace(-np.pi, np.pi, 256)
C, S = np.cos(X), np.sin(X)

ax_1 = fig.add_subplot(1, 2, 1)
ax_2 = fig.add_subplot(1, 2, 2)
ax_1.set_xlabel ('x')
ax_2.set_ylabel ('y')
ax_2.set_xlabel ('x')
ax_1.set_ylabel ('y')
ax_1.plot(X,C, label='cos', linewidth=2)
```

```
ax_2.plot(X,S, '--', label='sin',color='purple',
linewidth=3)
plt.subplots_adjust(wspace = 0.35 )/ змінюємо відстань між
окремими графіками
plt.show()
```



2) використати функцію plt.subplots(nrows=, ncols=), що створює об’єкт класу Figure та масив об'єктів класу Axes, а потім працюючи з кожним з цих об'єктів окремо, звертаючись до них як до елементів масиву.

Аналогічний результат можна одержати замінивши у попередньому прикладі

`fig, axes = plt.subplots(nrows = 1, ncols =2 )` та замінивши ax_1 ax_2 на axes[0] axes[1].

Слід також відзначити, що можна задати всі параметри одночасно використовуючи функцію Axes .set(), зокрема у попередньому прикладі можна замінити

```
ax_1.set_xlabel ('x')
ax_1.set_ylabel ('y')
```

на

```
ax_1.set(xlabel= 'x', ylabel ='y')
```

Хід виконання роботи

1) Використовуючи google colab зчитати експериментальні фотоемісійні спектри з файлу формату csv за допомогою функції з бібліотеки Numpy genfromtxt('Table1.csv', delimiter=',')

2) За допомогою matplotlib побудувати на одному рисунку графіки залежності для 3 різних значень енергії на одному рисунку.
3) Додати підписи до осей (**plt.xlabel** та **plt.ylabel)** та до графіку (**plt.title**)
4) Змінити діапазон значень використовуючи **plt.xlim** та **plt.ylim** та додати сітку (**plt.grid(True)**). Зберегти графік (**plt.savefig("graph1.png", dpi=100)**
5) Використовуючи pyplot.subplots() побудувати набір з 3 рисунків залежність при сталому куті, при сталій енергії та весь спектр цілком (використати **plt.imshow**).
6) Використовуючи ***numpy.linspace*** та ***numpy.meshgrid*** побудувати графік функції двох змінних. Використати plot_surface
7) Зберегти усі рисунки використовуючи plt.savefig().

Список рекомендованої літератури:

1. Python Ploting With Mathplotlib(Guide)/ URL: https://realpython.com/python-matplotlib-guide/ (дата звернення: 14.03.2025)
2. Jake VanderPlas. Python Data Science Handbook O'Reilly Media, Inc. Sebastopol, California, 2016, p.576
3. Jake VanderPlas. Python Data Science Handbook https://jakevdp.github.io/PythonDataScienceHandbook/ (дата звернення: 14.03.2025)
4. Matplotlib API Overview /URL https://matplotlib.org/3.3.3/api (дата звернення: 14.03.2025)

Лабораторна робота №2
“Використання перетворення Фур’є для обробки експериментальних даних”

Мета роботи: Ознайомитись з можливостями швидкого перетворення Фур’є для обробки зображень одержаних за допомогою просвічуючого електронного мікроскопа.

Теоретичні відомості

Суттєвий прогрес, який спостерігався в електроніці протягом останніх десятиліть, пов’язаний з мінітюаризацією. Якщо у 70-х роках минулого століття розмір окремого транзистора складав 10 мкм, то сучасні пристрої використовують технологію, з розміром транзистора порядку нм. Така мінітюаризація сприяла розвитку різних експериментальних методів, що дозволяють проводити дослідження нанометрових структур з надзвичайно високою роздільною здатністю.

Серед таких методів можна виділити, розроблені на початку 1980их років атомно-силову мікроскопію та скануючий тунельний мікроскоп, які дозволяють одержувати інформацію про властивості поверхні в залежності від положення зонда. Іншим досить великим класом методів, що дозволяють досліджувати нанорозмірні об’єкти, є методи пов'язані з оберненим простором, такі як рентгенівська дифракція . У цьому випадку використовуючи формулу Брега

$$2dsin(\alpha) = n\lambda ,$$

де ***d*** - стала ґратки, α - брегівський кут, ***n*** - порядок дифракційного максимума, ***λ*** - довжина хвилі, одержуємо інформацію за рахунок просторових частот ***(u,v)***, що пов'язані з кутом розсіяння. Після комп'ютерної обробки яких, можна одержувати інформацію про структуру досліджуваного зразка.

Серед цих методів слід окремо виділити електронну мікроскопію, яка дозволяє працювати як у координатному так і в оберненому просторі та одержувати інформацію про форму, кристалічну та мікроструктуру зразка.

Електронна мікроскопія

Існує велика кількість видів електронних мікроскопів — скануючі, растрові електронні, просвічуючий і таке інше. Для спрощення розглянемо принцип роботи просвічуючого мікроскопу.

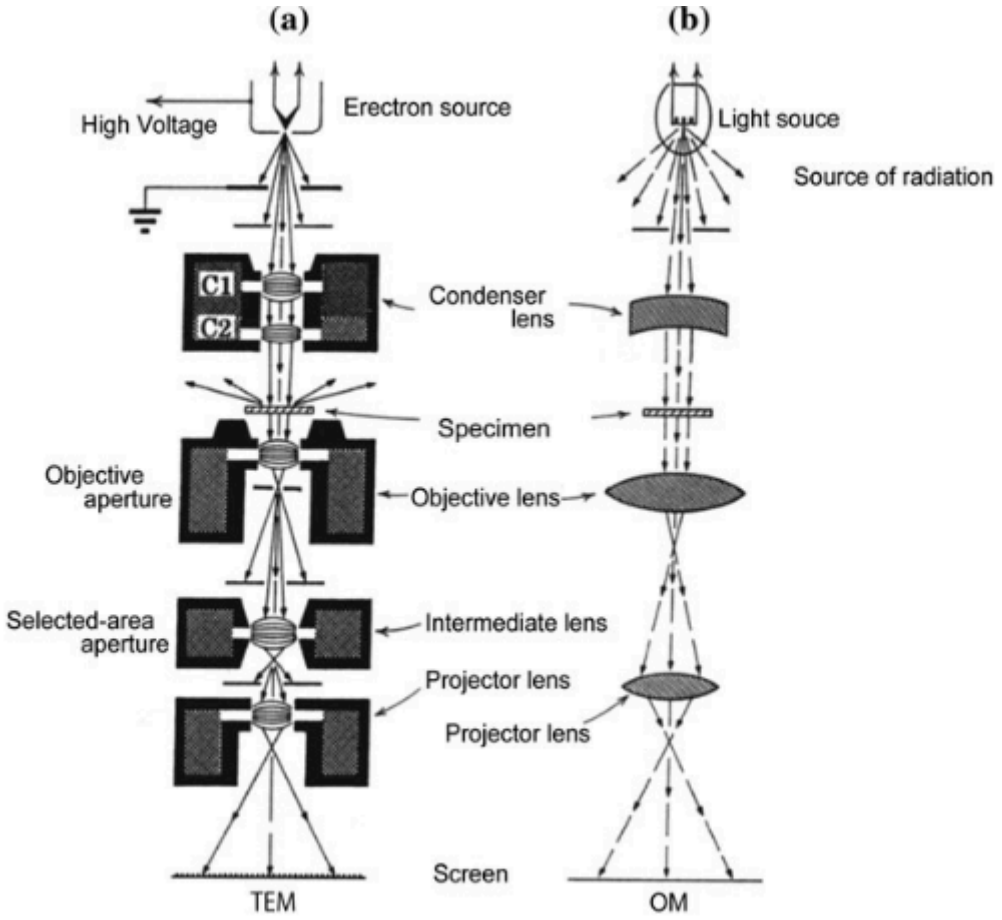


Рис.2.1. Порівняння схеми просвічуючого електронного мікроскопу (a) та оптичного (b)

Як можна бачити з Рис.2.1 принципова схема просвічуючого електронного мікроскопу схожа на оптичний мікроскоп. Використання електронів дозволяє суттєво підвищити роздільну здатність. Як відомо, роздільна здатність оптичного мікроскопу визначається довжиною хвилі випромінювання та використовуючи критерій Релея $D = 0.61\ \lambda/\alpha$, для оптичного мікроскопу роздільна здатність може досягати 100 нм. Для електронів з енергіями 100 кеВ, 200 кеВ, та 1 МеВ довжина хвилі де Бройля складатиме 0.00370, 0.00251, та 0.00087 нм, відповідно, теоретична роздільна здатність може досягати цих величин.

Просвічуючий електронний мікроскоп складається з джерела електронів — катода (вольфрамового, LaB_6). Емітовані електрони прискорюються до певних енергій (найчастіше в діапазоні від 80 до 200 кеВ), фокусуються за допомогою магнітних або електростатичних лінз та потрапляє на зразок, який досліджується. Частина електронів взаємодіє зі зразком та несе інформацію про зразок. Проходячи через систему збільшуючих лінз електрони потрапляють на детектор або екран, та дозволяють одержувати зображення мікроструктури.

Фур'є обробка цифрових зображень

Перетворення Фур'є зображення ***f(x,y)***,дозволяє розкласти сигнал по просторовим гармоніках ***F(u,v)***. Кожна компонента ***F(u,v)*** містить всі значення ***f(x,y)*** помножені на відповідні експоненти. Тому, за виключенням тривіальних випадків досить складно встановити однозначну відповідність між певними компонентами зображення та його Фур'є перетворенням. Проте, певні загальні співвідношення все ж таки існують. Зокрема компоненти, що змінюються найповільніше ***(u=v=0)*** пропорційні до середньої інтенсивності зображення. Низькі частоти відповідають повільнозмінними компонентам (слабкі варіації освітленості на цифрових зображеннях). Високі просторові частоти будуть відповідати краям об'єктів та різким змінам інтенсивності.

Техніка фільтрування у частотному просторі базується на модифікації (змінах) перетворення Фур'є для досягнення певних завдань (певної мети) та подальшого зворотнього Фур'є перетворення. Можливим є зміни фазових компонент та/або просторового спектру. Проте, найбільших результатів можна досягти, проводячи маніпуляції саме зі спектром. Тому для цифрового зображення ***f(x,y)*** розміром ***M*** на ***N***, рівняння фільтрації набуде вигляду

$$g(x, y) = F^{-1}[H(u, v)F(u, v)],$$

де F^{-1} -зворотне перетворення Фур'є, ***F(u,v)*** - дискретне Фур'є перетворення зображення ***f(x,y)***, ***H(u,v)*** - фільтр, функція фільтрації (filter function) , ***g(x,y)*** -відфільтроване зображення. У цьому випадку ***F,H*** та ***g*** є масивами такої ж розміру як і вхідне зображення. ***H(u,v)F(u,v)*** - поелементне множення відповідних масивів. Задання ***H(u,v)*** значно спрощується у випадку задання функцій симетричних відносно центру, що потребує центрування і ***F(u,v)***.

Найпростіший випадок фільтр низьких частот, який пропускає лише низькі частоти:

$$H_{LP}(u, v) = 1 \text{ якщо } D(u, v) \le D_0$$
$$= 0 \text{ якщо } D(u, v) > D_0 ,$$

де $D_0(u,v)$ - коло певного радусу, з центром у точці (0,0).

Та фільтр високих частот, який пропускає все крім низьких частот

$$H_{HP}(u,v) = 1 - H_{LP}(u,v)$$

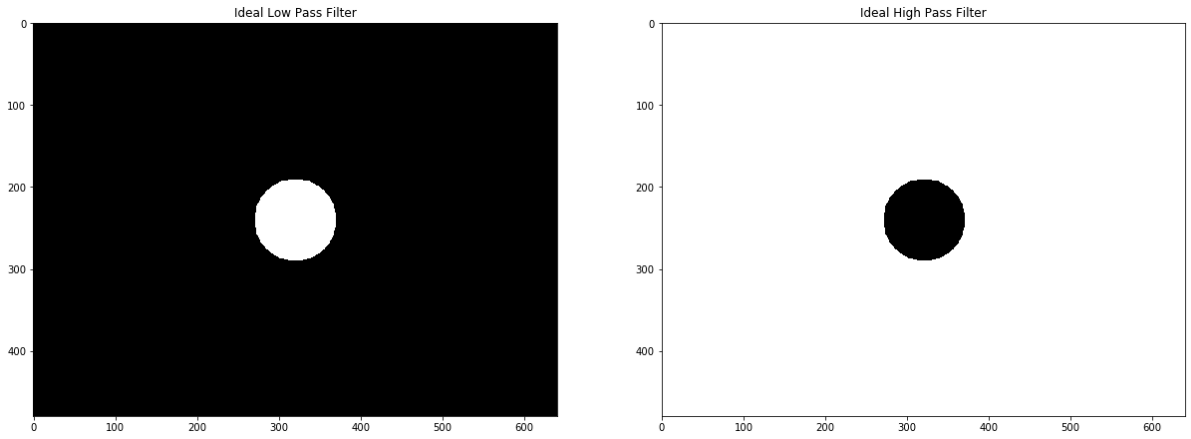


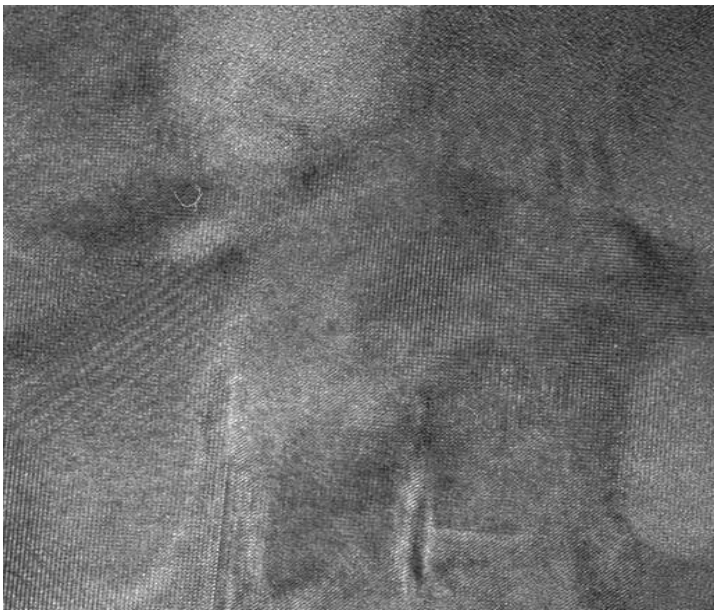
Рис.2.2. Ідеальні фільтри низьких(ліворуч) та високих (праворуч) частот

Хід виконання роботи

1) Перед початком виконання роботи імпортувати необхідні модулі

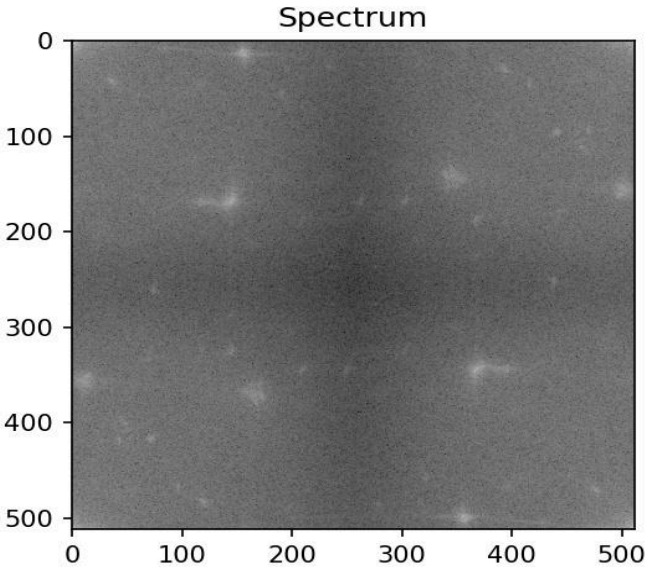
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

2) Завантажити одержані від викладача дані у середовище Google Colab та зчитати їх (`img_c1 = cv2.imread("Var1_42F_COPY.TIF", 0))`



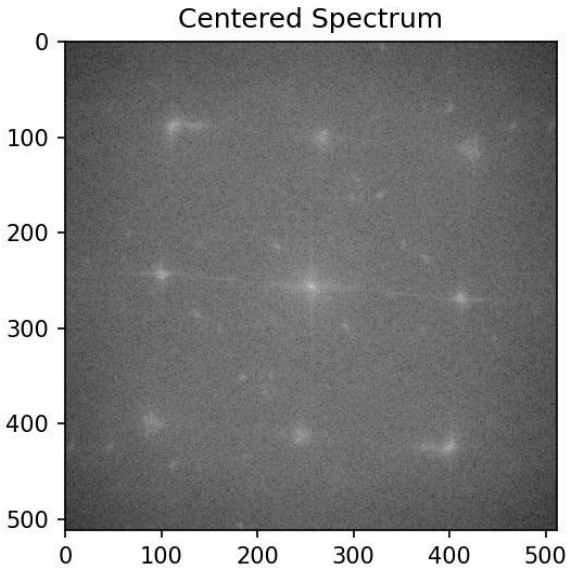
Приклад зображення одержаного за допомогою ТЕМ

3) Використовуючи функцію бібліотеки numpy **numpy.fft.fft2** одержати 2-вимірне Фур'є перетворення.

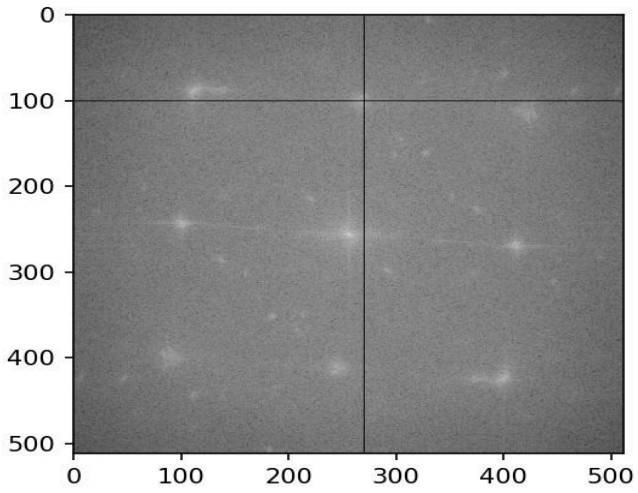


Примітка Щоб візуалізувати комплексне Фур'є перетворення використовувати у подальшому модуль (**np.abs**) та логарифмічну шкалу **np.log**

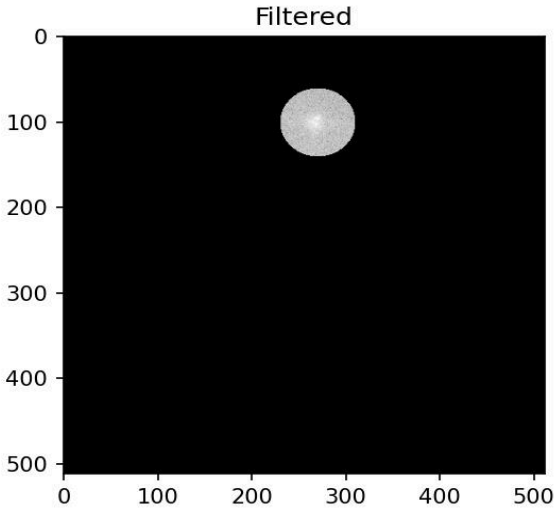
4) Використовуючи функцію numpy.fft.fftshift(), зсунути спектр, таким чином, щоб нульові частоти знаходились по центру



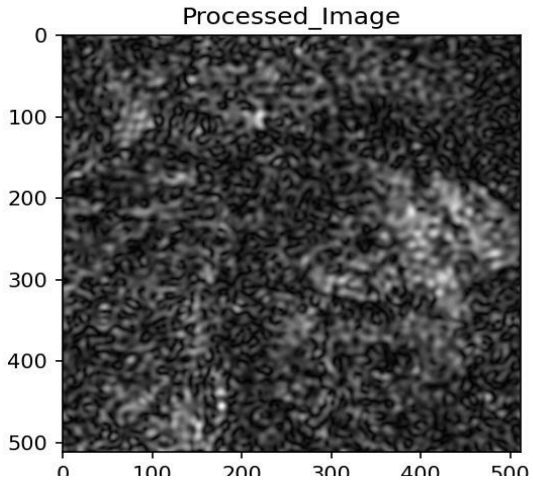
5) Обрати один з нецентральних дифракційних піків,



6) Використовуючи просторовий фільтр з заданими параметрами провести фільтрацію просторових гармонік у певному діапазоні частот (приклади реалізації див. додаток)



7) Провести зворотній зсув та зворотне перетворення частот, одержати зображення



8) Дослідити вплив на зображення фільтрування центрального піку за допомогою фільтра високих та низьких частот

Контрольні питання

1. Використовуючи критерій Релея оцінити роздільну здатність електронного мікроскопу з енергією електронів 60 keV.
2. Як пов'язані положення піків Фур’є перетворення з вектором оберненої ґратки ?
3. Чим буде обмежуватись максимальна товщина зразка, що можна досліджувати методом просвічуючої електронної мікроскопії?
4. Якщо центральний пік відповідає електронам, що пройшли через зразок, то яким електронам будуть відповідати нецентральні піки?
5. Роздільна здатність пов’язана з енергією електронів - чим вище енергія, тим краща роздільна здатність, чому енергії електронів, що використовуються зазвичай обмежуються 200 keV?

Список рекомендованої літератури:

1) Nobuo Tanaka/Electron Nanoimaging Basics of Imaging and Diffraction for TEMand STEM, Springer Japan KK, 2017, p.253
2) Rafael C. Gonzalez, Richard E. Woods/ Digital Image Processing, 3rd Edition Pearson, 2008, p.1022

Додаток

Приклад коду для реалізації фільтрів

```
import numpy as np
from math import sqrt
def distance(point1,point2):
return sqrt((point1[0]-point2[0])**2+point1[1]-point2[1])**2)

def idealFilterLP(D0,imgShape): // фільтр низьких частот
    base = np.zeros(imgShape[:2])
    rows, cols = imgShape[:2]
    center = (rows/2,cols/2)// кординати центру фільтра
    for x in range(cols):
        for y in range(rows):
            if distance((y,x),center) < D0:
                base[y,x] = 1
    return base

def idealFilterHP(D0,imgShape): // фільтр високих частот
    base = np.ones(imgShape[:2])
    rows, cols = imgShape[:2]
    center = (rows/2,cols/2)
    for x in range(cols):
        for y in range(rows):
            if distance((y,x),center) < D0:
                base[y,x] = 0
    return base
```

Лабораторна робота №3
“Частотно-часове представлення сигналів. Використання вейвлет перетворення для обробки сигналів.”

Мета роботи: Опанувати методи вейвлет-перетворення для аналізу сигналів з нестаціонарним спектром.

Теоретичні відомості

Перетворення Фур'є працює для випадків, коли частотний спектр є стаціонарним, тобто спектр сигналу не залежить від часу. Проте перетворення Фур'є не дозволяє визначити в який саме момент часу в сигналі з’явилася чи зникла складова тої чи іншої частоти, якщо спектр сигналу не є стаціонарним у часі. Перетворення Фур'є має гарну роздільну здатність у частотній області, але дуже погану у часовій. Для визначення складу сигналу можна використовувати певне часове “вікно”. Недоліком цього підходу є те, що мінімальна частота, яку можна фіксувати обмежується розміром вікна. Тобто, використання вікна дозволяє фіксувати зміну у часі високочастотних компонент, проте повільнозмінні сигнали на низьких частотах, не можна буде зафіксувати.

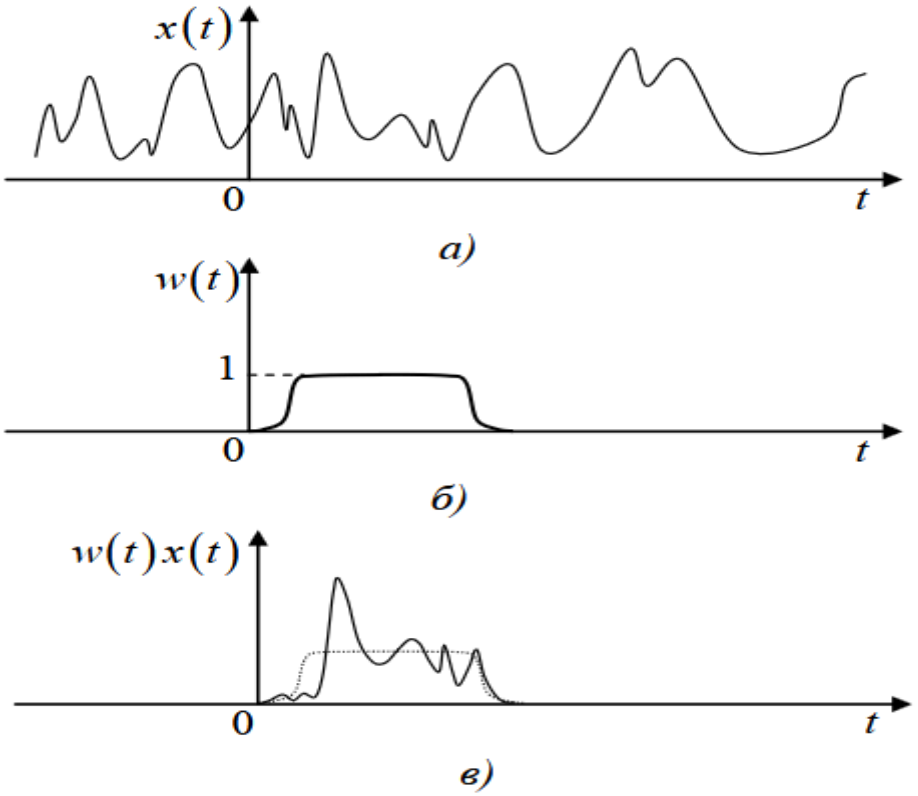


Рис.3.1 Використання вікна для визначення спектрального складу сигналу у певному діапазоні часу.

Для аналізу сигналів зі змінним спектром потрібне інше представлення сигналів – двовимірне частотно-часове представлення. Необхідно мати можливість знати одночасно і спектральний склад, і тривалості в сигналі складових на різних частотах. Для того, щоб кожна ділянка сигналу була аналізована у вікні з оптимальною шириною, треба мати можливість зміни тривалості (масштабування в часі) віконної функції. Тобто, в самому виразі для розкладу сигналу тривалість вікна повинна бути присутньою як параметр. В рамках класичного спектрального аналізу це зробити не можна, тож треба залучати розклади сигналу на складові, які відмінні від гармонічних.

Вейвлет-перетворення

Інженер-геофізик Д. Морле у кінці 70-х років XX століття зіткнувся з проблемою аналізу сигналів від сейсмодатчиків, сигнал яких містив як короткострокову високочастотну компоненту (сейсмічна активність) так і низькочастотну складову (спокійний стан земної кори) – протягом тривалого періоду. Використання вікна з Фур'є перетворенням дозволяє аналізувати або ВЧ складову, або НЧ складову, але не обидві одночасно.

Тому було запропоновано метод аналізу, в якому ширина віконної функції для низьких частот збільшувалась, а для високих - зменшувалась. Нове віконне перетворення одержувалось в результаті масштабування (стиснення) та зміщення у часі однієї материнської функції (скейлінг функції). Така материнська функція називалась вейвлетом.

Вейвлет-перетворення (від слова wavelet – маленька хвиля), введено в теорію сигналів у 1984 році в роботі геофізика Жана Морле та математика Александра Гросмана "Розклад функцій Харді у інтегровані з квадратом вейвлети сталої форми", в якій вони запропонували розклад сигналів за допомогою т.зв. фреймів.

Для випадку вейвлетів цей набір не обов'язково повинен бути ортогональним базисом, а може також бути напівортогональним, неортогональним базисом або переповненим словником функцій

$$x(t) = \sum_n c_n \psi_n(t) .$$

де c_n - коефіцієнт розкладу, який відповідає вкладу відповідної складової у сигнал;ψ_n - елемент з деякого набору функцій. В залежності від властивостей вейвлет-функцій ψ_n , на які розкладається сигнал, можна виділити такі класи вейвлет-перетворень:

1) Неперервне вейвлет-перетворення – виконується для неперервних та дискретних сигналів;

2) Розкладання сигналу по фреймам;

3) Ортогональні вейвлет-перетворення – клас перетворень сигналу, до якого належать: діадне ортогональне вейвлет-перетворення, вейвлет-перетворення з біортогональними базисами вейвлетів, розклад у вейвлет-пакети та вейвлети на інтервалі

Вимоги, щодо вейвлет функцій, які використовуються

Обмеженість . Квадрат норми функції має бути скінченним

$$||\psi||^2 = \int_{-\infty}^{+\infty} |\psi|^2 dt < \infty$$

Локалізованість. Вейвлет повинен бути локалізований поблизу нуля аргументу як у часовому, так і в частотному просторі. Для цього достатньо, щоб виконувались умови**:**

$$|\psi| \leq C(1 + |\psi|)^{-1-\epsilon}$$

$$\left|S_\psi(\omega)\right| \leq C(1 + |\omega|)^{-1-\epsilon}, \text{ при } \varepsilon > 0$$

Наприклад, дельта функція чи гармонічна функція не задоволняють умови локалізованості у часі та просторі

Нульове середнє. Графік материнської функції має осцилювати (бути знакозмінним) навколо нуля та мати нульову площу:

$$\int_{-\infty}^{+\infty} \psi(t)dt = 0$$

Автомодельність. Характерною особливістю вейвлет перетворення є його самоподібність. Всі вейвлети конкретного сімейства мають те ж саме число осциляцій, що і материнський вейвлет, оскільки отримуються з нього шляхом масштабних перетворень та зсуву. Також вейвлет повинен достаньо шкидко затухаючою функцією часової (просторової) змінної.

Вейвлет називається ортогональним, якщо сімейство $\{\psi_k\}$ представляє ортонормований базис функціонального простору $L_2(R)$. Вейвлети прийнято класифікувати за виглядом та особливостями

функції або за іменем вченого, який вперше запропонував той чи інший вейвлет. Відомі вейвлети Хаара, Симлета, Добеши, Коіфлетса, Гауса, Морле, Шенона, Мейера, біортогональний та зворотній біортогональний, частотний B-сплайновий,«мексиканський капелюх», «французький капелюх» та інші

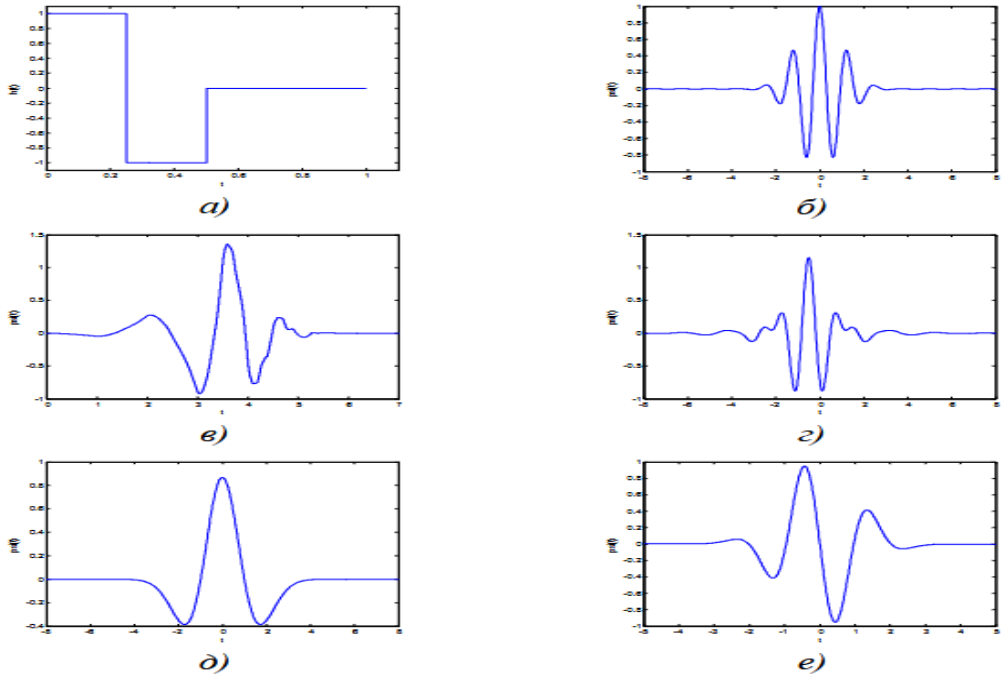


Рис.3.2. Приклади материнських вейвлет-функцій: а) функція Хаара, б) вейвлет Морле, в) вейвлет Добеші 4-го порядку, г) вейвлет Мейера, д) вейвлет «мексиканський капелюх», е) вейвлет Гауса 5-го порядку

Неперервне вейвлет-перетворення.

Вейвлет спектр (wavelet spectrum або time-scale-spectrum) ***$W_s(b,a)$*** - є двовимірним, на відміну від фур'є-спектра. Перший аргумент ***a*** у часовому масштабі аналогічний періоду осциляцій , тобто обернений частоті, другий ***b*** - аналогічний зміщенню сигнала по осі часу.

$$W_s(b,a) = (S(t),\psi_{ab}(t)) = \frac{1}{\sqrt{a}}\int_{-\infty}^{+\infty} S(t)\psi(\frac{t-b}{a})dt$$

$$S(t) = \frac{1}{C_\psi}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} W_s(b,a)\,\psi_{ab}(t)\frac{dadb}{a^2}$$

$W(b,a_0)$ - характеризує часову залежність при $a = a_0$

$W(b_0,a)$ - характеризує частотну залежність при $b = b_0$

Як показано на рис.3.3 процедура вейвлет перетворення починається з масштабу ***a=1*** і з кожною ітерацією значення збільшується. Це еквівалентно старту з високих частот і подальшого руху в бік низьких частот. Початкове значення ***a*** відповідає найбільш “стиснутому” вейвлету, при збільшенні ***a*** вейвлет “розширюється”.

Для кожної точки часового інтервалу вейвлет-функція знаходиться інтеграл добутку вейвлет-функції з певним значенням ***a*** та сигналу. При цьому результат інтегрування домножується на сталу ***1/|a|***, щоб сигнал для кожного масштабу мав би однакову енергію. Вейвлет потім зміщується праворуч у момент часу ***τ*** і процедура повторюється. Отримуєьться, ще одне значення, що відповідає даній точці частотно-часової площини.

Якщо в сигналі є спектральні компоненти відповідно поточному значенню a , добуток вейвлета з сигналом в інтервалі часу, де ця компонента присутня дає відносно велике значення, інакше добуток малий або дорівнює нулю. Зміна ширини вікна відбувається за рахунок зміни коефіцієнту ***a,*** при цьому збільшення ширин вікна виділяє все більш низькі частоти.

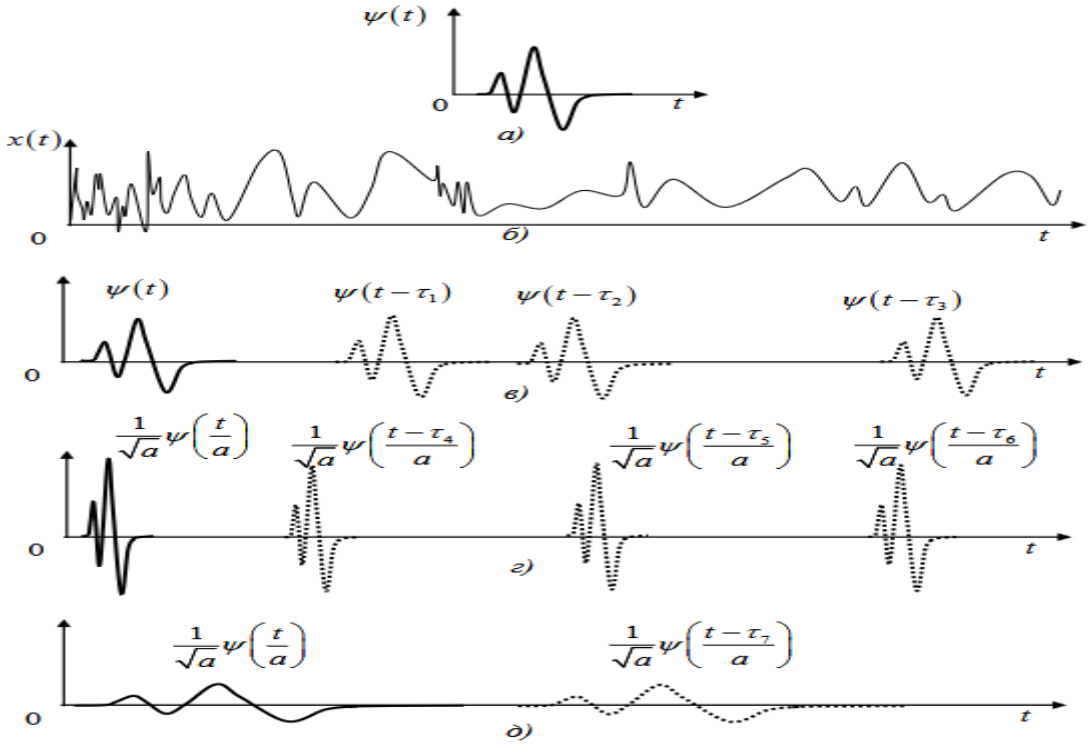


Рис.3.3 а) материнська вейвлет-функція, б) сигнал,
в) можливі зміщення немасштабованої материнської функції (a),
г) можливі зміщення при масштабах 0 a , д) можливі зміщення при масштабах 1

Дискретне вейвлет перетворення

При роботі з дискретними сигналами використовується дискретне вейвлет-перетворення (Discrete Wavelet Transform (DWT)). DWT це фактично CWT для певної сітки дискретних значень, особливістю є, що для вейвлет-перетворення найчастіше сітка обирається нерівномі рною.

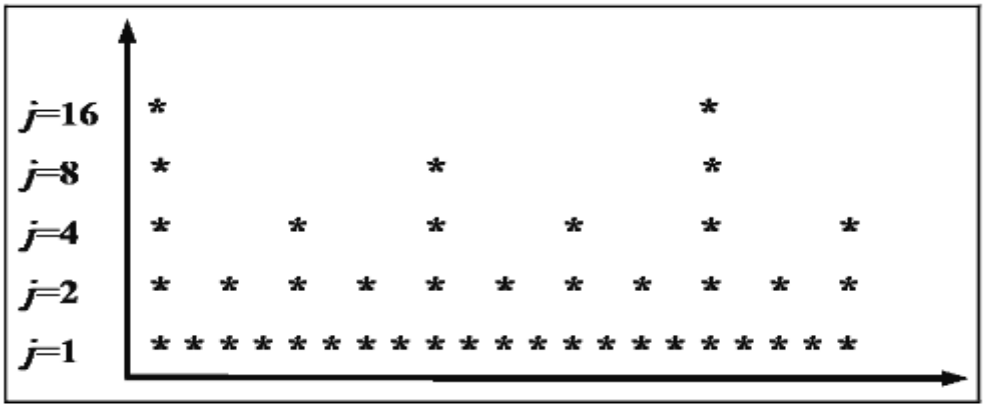


Рис.3.4. Сітка для розрахунку дискретного вейвлет перетвоерння

Дискретні значення обираються використовуючи формулу

$$a = a_0^{-m},\ b = ka_0^{-m},\ \boldsymbol{a_0 > 1\ m,k \in Z}$$

У загальному випадку, значення *a* може бути довільним, але зазвичай приймається рівним 2, при цьому перетворення називається діадним вейвлет-перетворенням. Для діадного перетворення розроблений швидкий алгоритм обчислень, аналогічний швидкому перетворенню Фурье, що зумовило його широке використання при аналізі дискретних функцій і масивів цифрових даних.

$$W(n,k) = W(t,a)\big|_{a=a_k, t=t_k} = \frac{1}{(\Delta a)^{k/2}} \int g * (n\Delta t - \frac{t}{(\Delta a)^k}) x(\tau) d\tau$$

DWT можна визначити як рекурсивне використання двох фільтрів зі скінченною імпульсною характеристикою. Один з фільтрів є ВЧ фільтром, другий НЧ, обидва фільтри мають частоту відсічки на половині частоти дискретизації. Фільтри використовуються до вхідної послідовності , щоб виділити низькочастотні та високочастотні компоненти $\boldsymbol{x_1(n)}$ та $\boldsymbol{x_2(n)}$.

$$x_1(n) = \sum_{k=0}^{L-1} c_k x(n-k)$$

$$x_2(n) = \sum_{k=0}^{L-1} d_k x(n-k)$$

де c_k та d_k коефіцієнти ВЧ та НЧ фільтрів. В результаті обробки фільтрами кількість елементів послідовності зменшується вдвічі, таким чином що $x_1(n)$ - відповідає низьким частотам, а $x_2(n)$ - відповідає верхньому частотному діапазоні.

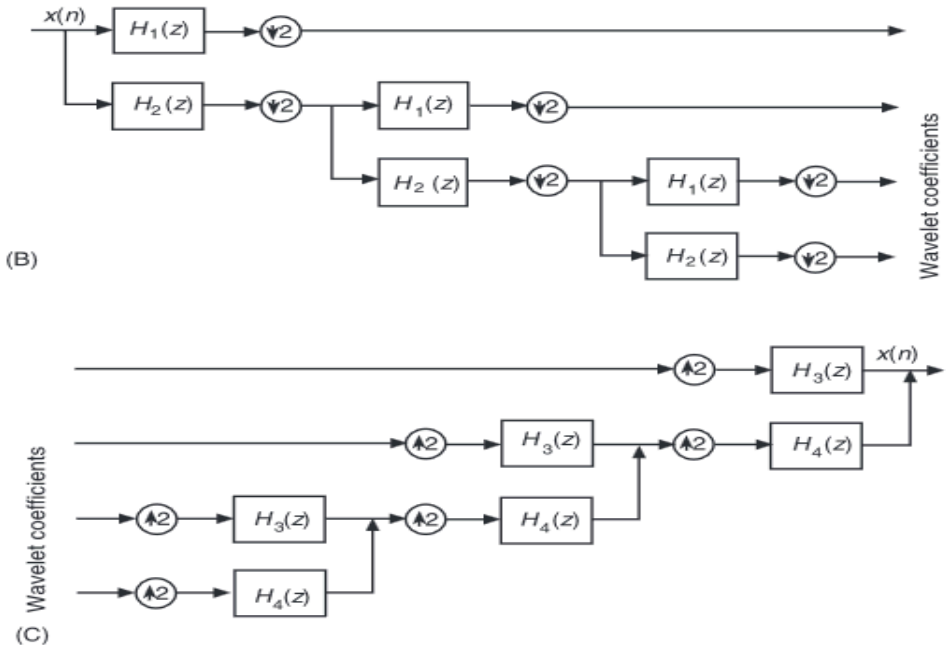
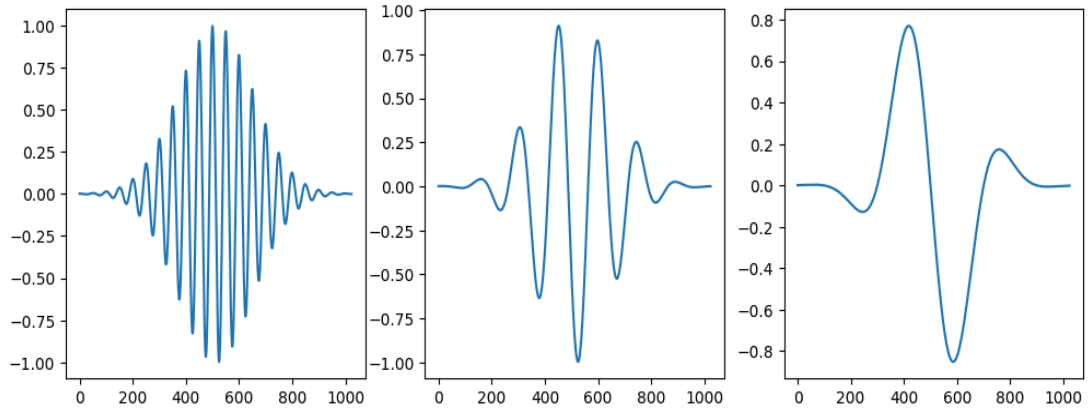
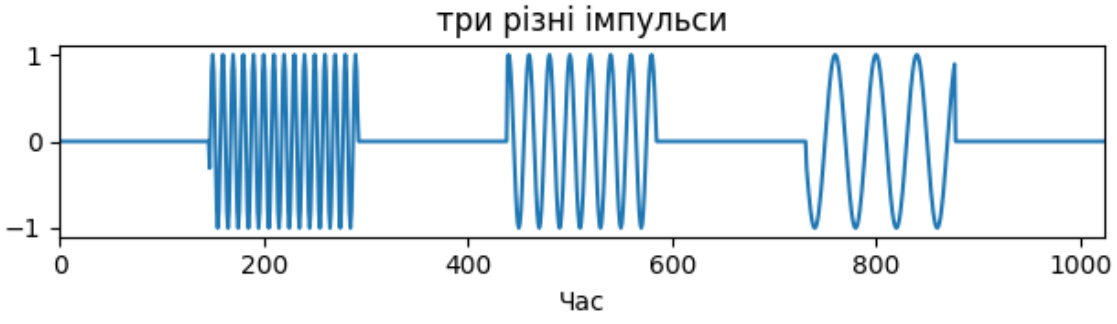


Рис.3.5 Схема дискретного вейвлеит перетворення та оберненого дискретного вейвлет перетворення

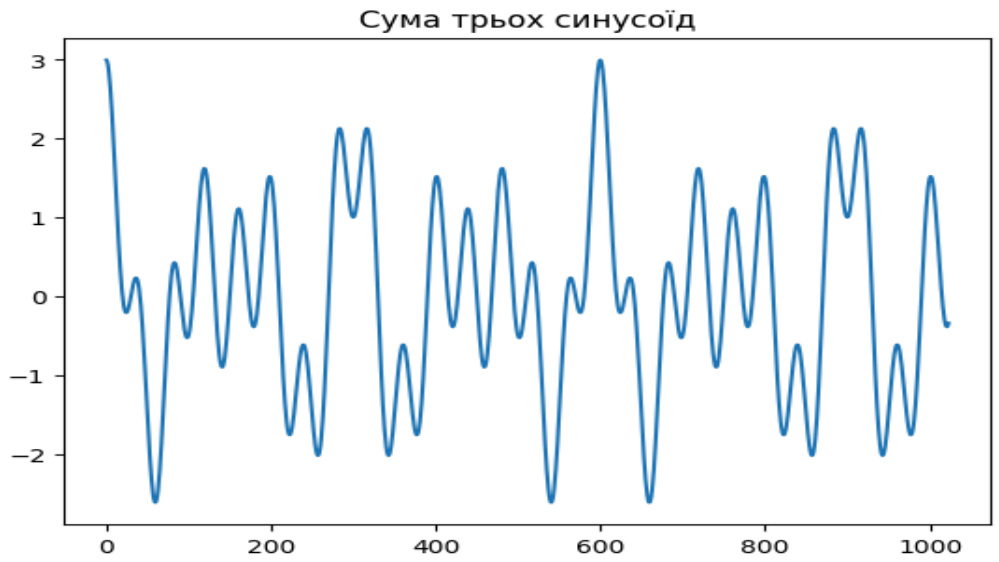
Хід виконання роботи

1) В середовищі *Google Colaboratory* використовуючи *pip install PyWawelets* встановити модуль для вейвлет-перетворення;
2) Підключити модуль *import pywt*
3) Використовуючи *pywt.cwt* знайти неперервне вейвлет перетворення наступних сигналів

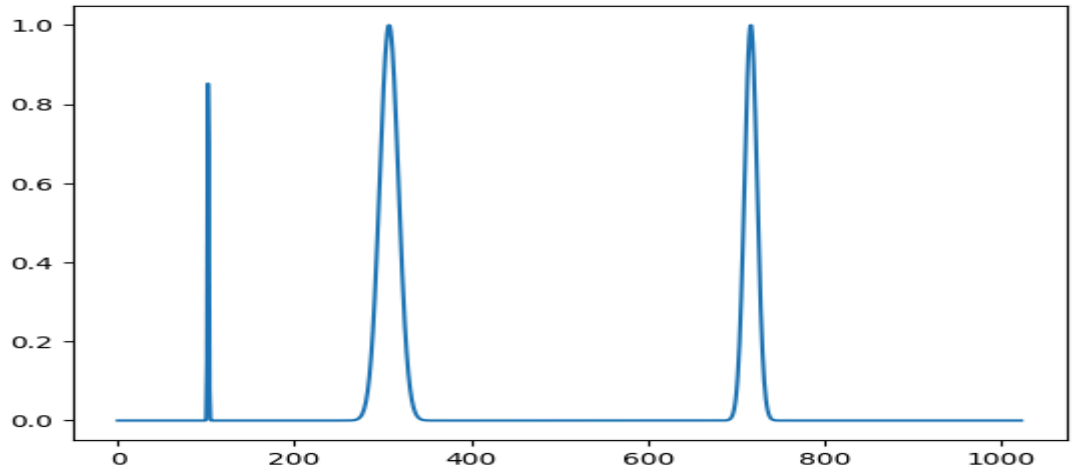
Добуток синусоїди та гаусової кривої при цьому змінювати частоту синусоїди та зсув зсунутої функції Гауса


Послідовність трьох імпульсів, кожен яких є синусоїдою обмеженою у часі


Сума трьох синусоїд з різними частотами. Змінити набір частот.



Сума декількох зсунутих гаусіанів. Дослідити вплив зміни напівширини гаусіана на вид вейвлет-перетворення сигнала.



Використовувати `wavelet='cmor'; wavelet='mexh'; wavelet='morl'`

Вказівка. Для того, щоб задати діапазон значень фунції використовувати *numpy.arange()* або *numpy.linspace()* для задання інтервалу зміни значень та векторизовані функції для розрахунків numpy.sin(), numpy.exp значень функції для кожної точки заданого діапазону.

4) Вибрати одну з вищенаведених функцій та додати шумів використовуючи numpy.random.normal. Провести вейвлет перетворення використовуючи pywt.dwt() та отримати 2 набори коефіцієнтів, що відповідають оцінці(НЧ коефіцієнти) та деталям(ВЧ коефіцієнти). **Додаткове завдання.** Провести знешумлення сигналу шляхом відкидання коефіцієнтів розкладу, що менше певного порогового значення. Для відновлення сигналу використати pywt.waverec() .
5) Поясніть результати отримані в пунктах 3-4.

Контрольні питання:

1) Для яких сигналів є сенс використовувати вейвлет-перетворення замість перетворення Фур'є?
2) Які вимоги до вейвлет-функцій? Наведіть приклади.
3) Поясніть, що таке неперервне вейвлет перетворення. Який фізичний зміст мають *a,b* коефіцієнти?
4) Які обмеження накладає використання перетворення Фур'є з певною функцією "вікна"?
5) Чим відрізняється неперервне вейвлет перетворення від дискретного?
6) Що таке діадне вейвлет перетворення?

Список рекомендованої літератури:

1) Теорія сигналів [Електронний ресурс] : навч. посіб. для студ. спеціальності 153 «Мікро- та наносистемна техніка» / КПІ ім. Ігоря Сікорського ; уклад.: А.О. Попов. – Електронні текстові данні (1 файл: 7399 Кбайт). – Київ : КПІ ім. Ігоря Сікорського, 2019. – 268 с.
2) DISCRETE WAVELET TRANSFORMS THEORY AND APPLICATIONS Edited by Juuso Olkkonen, IntechOpen, London, Greater London, United Kingdom, April 4, 2011, p. 270