НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ПРИЛАДОБУДІВНИЙ ФАКУЛЬТЕТ

КАФЕДРА КОМП'ЮТЕРНО-ІНТЕГРОВАНИХ ОПТИЧНИХ ТА НАВІГАЦІЙНИХ СИСТЕМ

«До захисту допущено»

Завідувач кафедри

_____ Бурау Н.І.

«___»_____2023 р.

Дипломна робота

на здобуття ступеня бакалавра за спеціальністю 151 – автоматизація та комп'ютерно інтегровані технології

на тему: «Методичне та алгоритмічне забезпечення до комп'ютерних практикумів з дисципліни «Інтегровані пакети прикладних програм»

Виконав:
студент групи ПГ-91
Галушко Вадим Олегович _____

Керівник:
К.т.н,
Півторак Діана Олександрівна _____

Рецензент:
Доцент, к.т.н.,
Шевченко Вадим Володимирович
_____

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ 2023

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Приладобудівний факультет

Кафедра комп'ютерно-інтегрованих оптичних та навігаційних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 151 – Автоматизація та комп'ютерно-інтегровані технології

Освітньо-професійна програма - Комп'ютерно - інтегровані технології та системи навігації і керування

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Надія БУРАУ

«___»_____20__ р.

ЗАВДАННЯ

на дипломну роботу студенту

Галушку Вадиму Олеговичу

1. Тема роботи «Методичне та алгоритмічне забезпечення для дисципліни "Інтегровані пакети прикладних програм"»,
керівник роботи Півторак Діана Олександрівна, к.т.н., доцент, затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом роботи 5 червня 2023 р.

3. Вихідні дані до роботи_ Тематика комп'ютерних практикумів: 1. *Ознайомлення з пакетом MatLab. 2. Моделювання динамічних систем з використанням пакета SimuLink. Створення S-моделей, використовуючи розділ Sources. 3. Створення S-моделей, використовуючи розділ Math Operation. 4. Створення S-моделей, використовуючи розділ Continuous. 5. Розв'язок рівнянь у символьному вигляді. 6. Розв'язування задач за допомогою каталогу Control System ToolBox. 7. Розв'язування задач за допомогою каталогу Signal Processing ToolBox. 8. Розв'язування задач за допомогою каталогу Image Processing ToolBox. 9.Розв'язування задач за допомогою каталогу WaveLet ToolBox.*_____

4. Зміст роботи *Кожний комп'ютерний практикум повинен містити: 1. Мета роботи. 2 Теоретичні відомості. 3. Завдання до виконання лабораторної роботи. 4. Контрольні запитання. Література.*_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) презентація _____

6. Консультанти розділів роботи*

	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 3 квітня 2023 року _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з пакетом MatLab	11.04.2023 р.	
2.	Моделювання динамічних систем з використанням пакета SimuLink. Створення S-моделей, використовуючи розділ Sources.	17.04.2023 р.	
3.	Створення S-моделей, використовуючи розділ Math Operation.	24.04.2023 р.	
4.	Створення S-моделей, використовуючи розділ Continuous.	01.05.2023 р.	
5.	Розв’язок рівнянь у символьному вигляді	08.05.2023 р.	
6.	Розв’язування задач за допомогою каталогу Control System ToolBox.	15.05.2023 р.	
7.	Розв’язування задач за допомогою каталогу Signal Processing ToolBox	22.05.2023 р.	
8.	Розв’язування задач за допомогою каталогу Image Processing ToolBox.	29.05.2023 р.	
9.	Розв’язування задач за допомогою каталогу WaveLet ToolBox	05.06.2023 р.	
10.	Оформлення пояснювальної записки. Підготовка до захисту	05.06.2023 р.	

Студент Вадим ГАЛУШКО

Керівник Діана ПІВТОРАК

* Якщо визначені консультанти. Консультантом не може бути зазначено керівника дипломної роботи.

Анотація

В даній дипломній роботі було розроблено методичне та алгоритмічне забезпечення до комп’ютерних практикумів з дисципліни «Інтегровані пакети програм».

В кожному окремому комп’ютерному практикумі були наведені короткі теоретичні відомості про програмне середовище MatLab та візуальне середовище моделювання SimuLink, а також каталоги Toolbox, які використовуються в даних практикумах, а саме: Control System Toolbox, Signal Processing Toolbox, Image Processing Toolbox, WaveLet Toolbox, Symbolic Math Toolbox. Проведено ознайомлення з можливостями пакету MatLab та SimuLink, а також з каталогами Toolbox. До кожного комп’ютерного практикуму наведені приклади кодів та схему у SimuLink. Кожен комп'ютерний практикум включає в себе набір завдань, які студент повинен виконати самостійно, а також питання, які допоможуть студенту перевірити його розуміння вивченого матеріалу.

Ключові слова: MatLab, SimuLink, Control System Toolbox, Signal Processing Toolbox, Image Processing Toolbox, WaveLet Toolbox, Symbolic Math Toolbox.

Annotation

In this thesis, we developed methodological and algorithmic support for computer workshops in the discipline "Integrated Software Packages".

In each individual computer workshop, brief theoretical information about the MatLab software environment and the SimuLink visual modeling environment was provided, as well as the Toolbox catalogs used in these workshops, namely: Control System Toolbox, Signal Processing Toolbox, Image Processing Toolbox, WaveLet Toolbox, Symbolic Math Toolbox. The students were familiarized with the capabilities of MatLab and SimuLink, as well as with the Toolbox catalogs. For each computer workshop, sample codes and a diagram in SimuLink are provided. Each computer workshop includes a set of tasks that the student must complete independently, as well as questions to help the student check his or her understanding of the material studied.

Keywords: MatLab, SimuLink, Control System Toolbox, Signal Processing Toolbox, Image Processing Toolbox, WaveLet Toolbox, Symbolic Math Toolbox.

Зміст

ПЕРЕЛІК ВИКОРИСТАНИХ СКОРОЧЕНЬ

ПІД - ідентифікатор процесу багатозадачної операційної системи;

АЧХ – амплітудно-частотна характеристика;

ФЧХ - фазо-частотна характеристика;

ЛАЧХ – логарифмічно амплітудно-частотна характеристика;

ЛФЧХ – логарифмічно фазо-частотна характеристика;

ЛАФЧХ - логарифмічно амплітудно-фазо частотна характеристика;

АФЧХ - амплітудно-фазочастотна характеристика;

SISO - система с одним входом і одним виходом;

MIMO - системи зв'язку з рознесеними передавальними і приймальними антенами;

АЦП - аналого-цифровий перетворювач;

WT – хвильове перетворення;

DWT – дискретне хвильове перетворення;

CWT – безперервне хвильове перетворення.

ВСТУП

У сучасному світі інформаційних технологій важко переоцінити значення комп’ютерної підтримки процесу навчання, особливо в таких дисциплінах, як «Інтегровані пакети прикладних програм». Дана дипломна робота включає в себе комп’ютерні практикуми, що охоплюють різні розділи пакету MatLab, включаючи S-моделювання в пакеті SimuLink, розв’язок рівнянь у символьному вигляді та використання спеціалізованих каталогів для розв’язування задач у галузях контролю, обробки сигналів, обробки зображень та вейвлет-аналізу.

Зв'язування MatLab та SimuLink дозволяє створити потужні інтегровані системи моделювання, аналізу та синтезу різних типів систем. Розділи комп'ютерних практикумів, які пропонуються у даній дипломній роботі, дозволяють студентам ознайомитися з різними інструментами, що входять до складу MatLab та SimuLink, та навчитися їх використовувати для вирішення практичних задач.

Метою даної дипломної роботи є розроблення методичного та алгоритмічного забезпечення для дисципліни «Інтегровані пакети прикладних програм», що включатиме розроблення комплексу комп’ютерних практикумів з використанням пакету MatLab та SimuLink. В роботі будуть розглянуті основні поняття та інструменти, що використовуються в пакеті MatLab і SimuLink та їх розділах, а також будуть наведені приклади задач з використанням каталогів Control System ToolBox, Signal Processing ToolBox, Image Processing ToolBox та WaveLet ToolBox. Результатом роботи буде створення повноцінного методичного та алгоритмічного забезпечення до комп’ютерних практикумів для студентів, які вивчають дисципліну «Інтегровані пакети прикладних програм».

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 1
ОЗНАЙОМЛЕННЯ З ПАКЕТОМ MATLAB

Мета роботи

Ознайомитись з інтерфейсом та базовими можливостями пакета MatLab, навчитись працювати з командним вікном, створювати скрипти, використовувати вбудовані функції, будувати графіки та розв'язувати прості математичні задачі. Також метою є зрозуміти основні принципи роботи з матрицями та векторами в MatLab і вміти виконувати матричні операції.

1.1. Теоретичні відомості

1.1.1. Ознайомлення з інтерфейсом MatLab та використання командного вікна MatLab для виконання простих арифметичних операцій

MatLab - це система автоматизації математичних розрахунків, яка базується на матричних операціях. Вона широко використовується у складних математичних розрахунках, зокрема при моделюванні статичних та динамічних систем. За допомогою MatLab можна отримувати математичні описи моделей, аналізувати їх властивості та синтезувати елементи керування. Також доступні інші інструменти, такі як оптимізація та аналіз у часовій та частотній областях.

Інтерфейс MatLab складається з різних вікон та панелей, які дозволяють виконувати різноманітні завдання.

Основні елементи інтерфейсу MatLab:

- **Command Window** – вікно команд, де можна виконувати команди MatLab та отримувати відповіді на них.
- **Current Folder** – панель, яка відображає поточну теку та файли, які в ній містяться.

- **Workspace** – панель, яка відображає змінні, які були введені в **Command Window** або завантажені з файлу.
- **Editor** – вікно редактора, яке дозволяє редагувати та зберігати скрипти та функції MatLab.
- **Figure Window** – вікно, яке відображає графіки та інші візуальні елементи, створені в MatLab.

Наприклад, у вікні **Command Window** можна виконати просту арифметичну операцію, таку як додавання двох чисел (рис. 1.1).

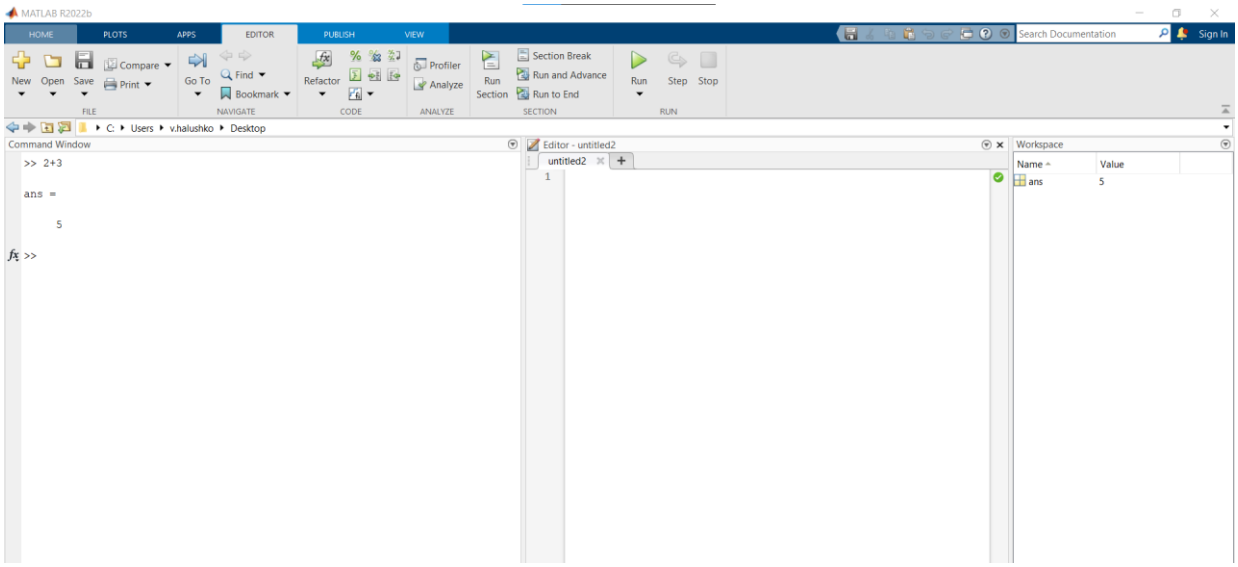


Рис. 1.1 Основне вікно системи MatLab

Ознакою того, що MatLab готовий до сприйняття і виконання чергової команди, є поява в останньому рядку текстового поля командного вікна знака запрошення (>>), праворуч якого миготить вертикальна риска [1].

Якщо оператор не містить знака присвоювання (=), тобто є просто записом деякої послідовності дій над числами і змінними, значення результату присвоюється спеціальній системній змінній з ім'ям **ans**. Отримане значення змінної **ans** можна використовувати в наступних операторах обчислень, використовуючи це ім’я **ans.** При цьому варто пам’ятати, що значення системної змінної **ans** змінюється після дії чергового оператора без знаку присвоювання [1].

У загальному випадку форма подання результату в командне вікно має вид [1]:

<Ім'я змінної> = <результат>.

При роботі з MatLab діє найпростіший редактор, команди якого перераховано нижче у вигляді комбінацій клавіш [3]:

- <®> або Ctrl+b – переміщення курсору вправо на один символ;
- ← або Ctrl+f – переміщення курсору вліво на один символ;
- Ctrl+→ або Ctrl+r – переміщення курсору вправо на одне слово;
- Ctrl+← або Ctrl+I – переміщення курсору вліво на одне слово;
- Home або Ctrl+a – переміщення курсору на початок рядку;
- Del або Ctrl+d – видалення символу справа від курсору;
- Backspace або Ctrl+h – видалення символу зліва від курсору;
- Ctrl+k – видалення до кінця рядку;
- Ins – включення або виключення режиму вставки;
- PgUp – перегортання сторінок сесії вгору;
- PgDn - перегортання сторінок сесії вниз;
- Esc – очищення рядку вводу;
- Ctrl+c – дозволяє зупинити виконання скрипта чи функції.

Для зручності роботи з MatLab також можна використовувати допоміжні інструменти, такі як довідка, підказки та автодоповнення, які дозволяють швидко знаходити необхідні команди та функції.

У верхній частині вікна (рис. 1.2) розміщений рядок вкладок, в якому містяться меню ***Home***, ***Plots***, ***Apps, Editor, Publish, View***. Щоб відчинити одне з меню потрібно встановити на ньому курсор миші і клацнути на нього лівою кнопкою.

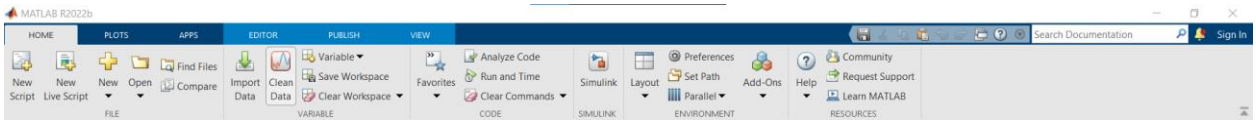


Рис. 1.2 Верхнє вікно MatLab з головним меню та панеллю інструментів

Головне меню в MatLab містить такі опції [4]:

- **File** (Файл): – робота з файлами: створення нового файлу, вибір існуючого файлу з каталогу, закриття файлу, збереження поточного файлу, збереження файлу зі зміною імені, друк документа і вихід в Windows.
- **Home** (Домашня): опції для роботи з вмістом та форматуванням коду в редакторі, включаючи вирівнювання коду, виконання коду, вибір кольорової схеми та розміру шрифту.
- **Plots** (Графіки): опції для створення та редагування графіків, включаючи вибір типу графіку, налаштування осей та маркерів, додавання легенди.
- **Apps** (Додатки): опції для доступу до додатків та інструментів MatLab, таких як SimuLink, Signal Processing Toolbox, Image Processing Toolbox та ін.
- **Editor** (Редактор): виконання основних операцій редагування (відміна операції, копіювання виділених частин документа в буфер з їх подальшим видаленням і без нього, перенесення виділених частин, їх стирання).
- **Debug** (Налагодження): опції для налагодження та відлагодження коду, включаючи встановлення точок зупинки, крок за кроком виконання коду, перегляд значень змінних.
- **View** (Вид): опції для налаштування параметрів відображення в MatLab, включаючи розмір та масштаб, налаштування графічного інтерфейсу.
- **Help** (Довідка): керування засобами довідкової системи.

Робочий простір (**Workspace**) - це область у пам'яті комп'ютера, де зберігаються всі змінні та об'єкти (рис. 1.3), які були створені та завантажені в MatLab. Він відображається у вигляді таблиці в головному вікні MatLab та містить інформацію про ім'я змінної, її значення та тип даних.

Робочий простір може бути корисним для візуалізації та аналізу змінних та об'єктів, що використовуються у програмі. Він дозволяє легко переглядати

значення змінних, додавати нові та видаляти існуючі, а також зберігати їх у файл. У робочому просторі також можна використовувати функції для аналізу та обробки даних, такі як сортування, фільтрування, обчислення статистичних характеристик та багато інших.

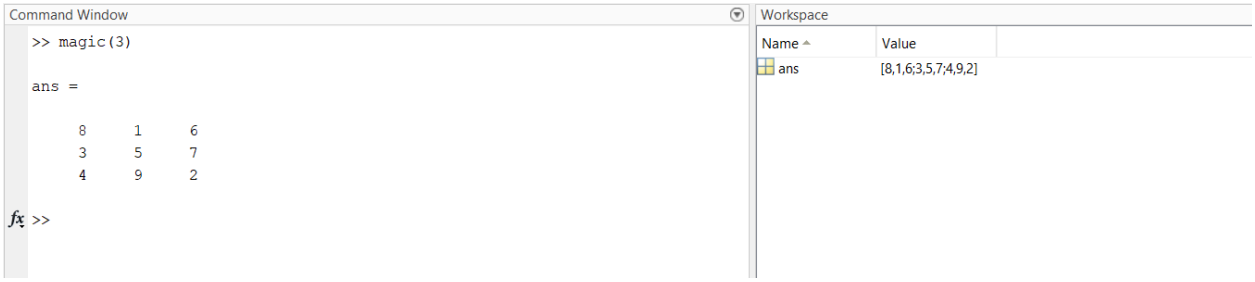


Рис. 1.3 Робочий простір (**Workspace**)

У MatLab дані розглядаються як матриці та вектори. Наприклад, числа можуть бути збережені як матриці розмірністю 1x1, або як вектори розмірністю 1xn або nx1, де n - кількість чисел.

Крім того, дані можуть бути збережені в різних форматах, таких як числа з плаваючої комою, цілі числа, символьні вирази тощо. MatLab також підтримує роботу з багатовимірними масивами даних, що дозволяє зберігати і обробляти велику кількість інформації.

Для роботи з даними в MatLab можна використовувати різноманітні функції та команди, які дозволяють виконувати різні операції, такі як обробка даних, статистичний аналіз, візуалізація результатів тощо.

В MatLab можливі два режими роботи: функціональний та інтерактивний режим.

- У функціональному режимі, користувач може написати скрипт на мові програмування MatLab, зберегти його як файл з розширенням .m та запустити його в командному вікні або з іншого скрипта.
- У інтерактивному режимі користувач може вводити команди в командне вікно та отримувати відповіді в режимі реального часу. Крім того, користувач може працювати з даними в робочому просторі та

використовувати графічний інтерфейс MatLab для візуалізації даних та результатів обчислень.

Під час роботи з програмою MatLab результати обчислень без графічного відображення виводяться у вікно командного рядка. У разі необхідності виводу результатів у вигляді тексту, можна використовувати спеціально створюване вікно.

У MatLab можна заблокувати вивід результату в командному вікні, якщо в кінці рядка вводу поставити знак крапка з комою «;». Значення змінної, якій результат присвоюється, буде зберігатись у робочій області [7].

При роботі з масивами у MatLab визначені оператори поелементного виконання операцій. Перед символом операції вводиться крапка «.». Символ присвоєння в MatLab позначається знаком рівності «=», а рівність як оператор відношення в умовах вводиться як подвійна рівність «==».

Коментарі в програмі MatLab вводяться з використанням символу «%», який повинен розміщуватись в першій позиції рядка. Коментарі це текст, в якому не потрібно включати символи операцій [1, 2].

Також в MatLab можна використовувати такі основні математичні дії:

- Додавання: «+»
- Віднімання: «-«
- Множення: «*»
- Ділення: «/»
- Піднесення до степеня: «^»
- Квадратний корінь: «sqrt()»
- Абсолютна величина: «abs()»
- Експоненціальна функція: «exp()»
- Логарифм: «log()»
- Синус: «sin()»
- Косинус: «cos()»
- Тангенс: «tan()»

1.1.2. Робота з векторами та матрицями

Робота з масивами даних є одним з найбільш важливих аспектів роботи в MatLab. Масиви в MatLab можуть містити дані різних типів, таких як числа, рядки, логічні значення та інші.

Створення масивів в MatLab може бути здійснене різними способами, зокрема:

- введення масиву вручну через командне вікно (рис. 1.4);
- завантаження масиву з файлу або бази даних;
- створення масиву за допомогою вбудованих функцій, наприклад, **ones()**, **zeros()**, **eye()**, **rand()** та інших.

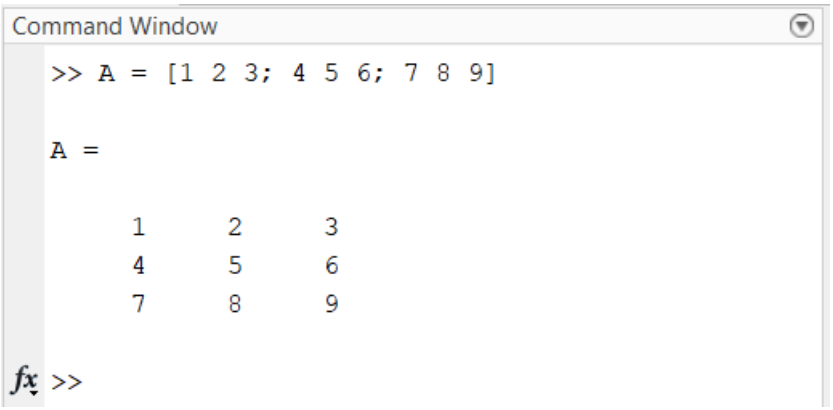


Рис. 1.4 Звичайне створення масиву 3x3 в командному вікні MatLab

Для оголошення та індексування масиву в MatLab здійснюється з використанням квадратних дужок **[ ]**, в які вказують номери елементів масиву, які потрібно витягнути.

Наприклад, якщо **A** - це масив, то звернення до елементів комірки масиву здійснюється за допомогою круглих дужок: **A(1)** поверне перший елемент масиву, а **A(2,;)** поверне підмасив, який складається з елементів масиву другого рядка (рис. 1.5). Також можна використовувати логічне індексування, коли для витягування підмасиву використовуються логічні вирази.

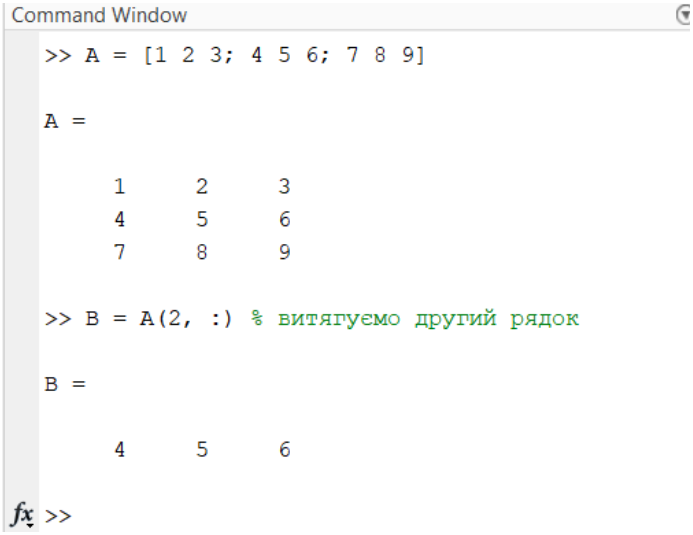


Рис. 1.5 Індексування та витягування підмасиву

Операції над масивами в MatLab зазвичай виконуються поелементно. Наприклад, оператор «+» виконує додавання елементів масивів, а оператор «*» виконує множення елементів масивів (рис. 1.6).

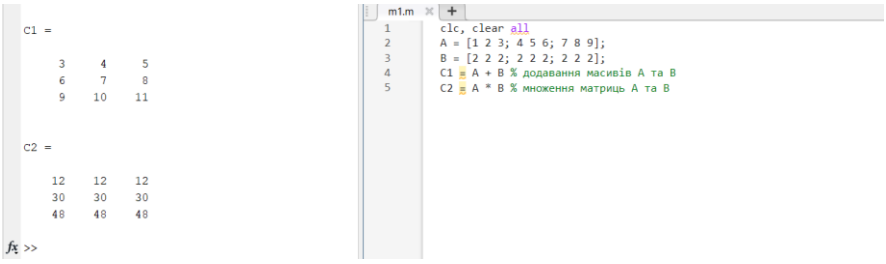


Рис. 1.6 Приклади операцій над масивами

Якщо розміри масивів не співпадають, то можна використовувати функції для розкладання та об'єднання масивів, наприклад, **reshape()**, **cat()**, **vertcat()**, **horzcat()** та інші.

Функція **reshape()** дозволяє змінити розміри масиву без зміни його елементів. Наприклад, якщо створено масив розміром 4x4 (рис. 1.7) та необхідно перетворити його в масив розміром 2x8.

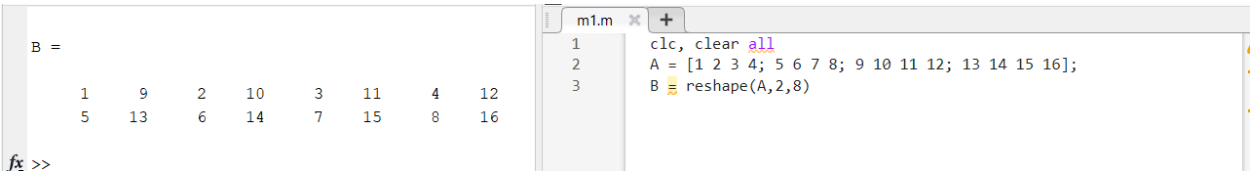


Рис. 1.7 Функція **reshape()**

Функції **cat(), vertcat() та horzcat()** дозволяють об'єднати масиви вздовж різних вимірів. Наприклад, якщо існує два масиви та необхідно об'єднати їх вздовж горизонтальної вісі, тоді використовують функцію **horzcat()** (рис. 1.8).

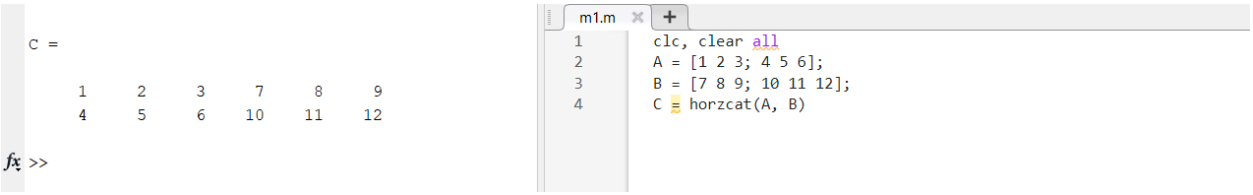


Рис. 1.8 Приклад застосування функції **horzcat()**

Якщо необхідно об'єднати два масиви вздовж вертикальної вісі, тоді використовується функція **vertcat()** (рис. 1.9).

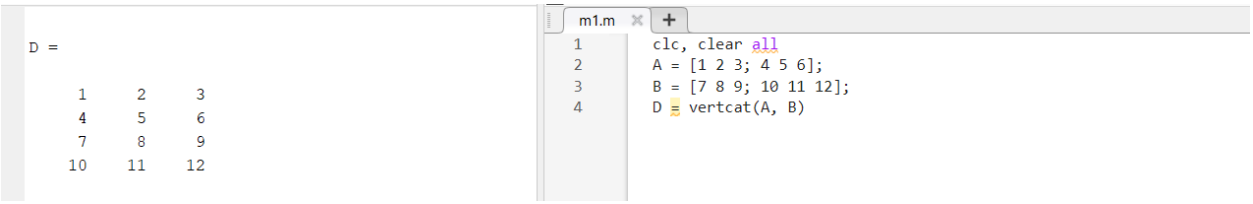


Рис. 1.9 Приклад застосування функції **vertcat()**

Загалом, робота з масивами в MatLab є досить простою та зручною, а вбудовані функції дозволяють виконувати різноманітні операції над масивами з високою швидкістю та точністю.

У MatLab є багато вбудованих функцій для роботи з масивами та матрицями, які допомагають виконувати різноманітні операції, а саме:

- Розкладання матриці: **LU**, **QR**, **SVD**, **eig**, **chol**.
- Обернення матриці: **inv**.
- Додавання матриць: **+**.
- Множення матриць: *****.
- Транспонування матриць: **'**.
- Елементарні функції: **sin**, **cos**, **tan**, **log**, **exp**, **sqrt**.

Для використання функцій в MatLab необхідно спочатку задати матриці, над якими будуть виконуватись операції. Наприклад, для додавання двох матриць можна скористатись кодом, який наведено на рис. 1.10.

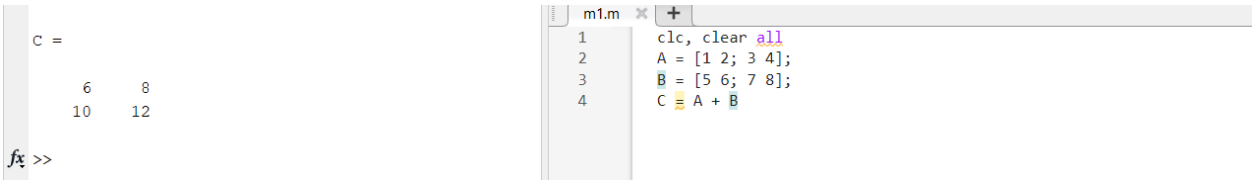


Рис. 1.10 Приклад додавання двох матриць

Функція **magic(N)** в MatLab створює «магічну» матрицю розміром NxN, де кожен рядок, кожен стовпець і кожна діагональна лінія мають однакову суму. На рис. 1.11 наведено приклад використання функції **magic.**

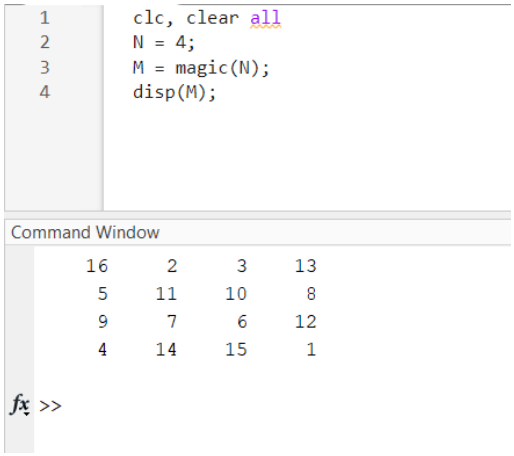


Рис. 1.11 Приклад використання функції **magic**

З прикладу, який наведено на рис.1.11 видно, що кожен рядок, кожен стовпець і кожна діагональна лінія матриці M мають однакову суму, що робить її «магічною».

Використання функцій для роботи з масивами та матрицями значно спрощує роботу з числовими даними в MatLab і дозволяє ефективно вирішувати складні задачі.

1.1.3 Побудова графіків з використанням вбудованих функцій

У MatLab є багато функцій для побудови графіків. Основна функція для побудови 2D графіків - це **plot()**. Вона використовується для побудови лінійних графіків та графіків з точок. Функція **plot()** приймає два або більше вектори, що містять координати точок графіка.

Основні функції, які використовуються при побудові та оформленні графіків представлені в табл.1.1 [5].

Таблиця 1.1

Основні функції, які використовуються при оформленні графіків

Функція	Опис функції
plot	Використовується для побудови лінійних графіків. Вона приймає вектори або матриці даних для вісей X та Y і будує графік.
xlabel, ylabel	Використовуються для задання підпису осі X та осі Y відповідно.
title	Використовується для задання заголовку графіку.
legend	Використовується для додавання легенди до графіку, яка пояснює значення ліній або маркерів на графіку.
xlim та ylim	Використовуються для налаштування меж значень по вісях X та Y відповідно.
grid	Використовується для додавання сітки на графік, що полегшує орієнтацію та вимірювання значень.
hold on, hold off	Використовуються для утримання графічного вікна для наступного графіку. hold on зберігає поточні графіки на вікні, дозволяючи додати новий графік без заміни, тоді як hold off вимикає режим утримання.
subplot	Використовується для створення багатокомірних графіків, де кілька графіків можуть бути розміщені у вигляді сітки.

Функція	**Опис функції**
figure	Використовується для створення нового графічного вікна або вибору існуючого вікна для наступних графіків.
colormap	Використовується для задання колірної мапи для графіків, дозволяючи візуально відображати різні значення.
box	Використовується для відображення або приховування рамки графіку. За замовчуванням рамка відображається, але команда box off може використовуватися для її вимкнення.
axis	Використовується для налаштування масштабу осей графіку. Наприклад, команда axis([xmin xmax ymin ymax]) встановлює межі для осей X та Y.
text	Використовується для додавання тексту на графік. Вона приймає координати та текст, який слід відобразити.
grid minor	Використовується для відображення додаткової сітки з меншими підрозділами на графіку.
axis equal	Використовується для встановлення однакового масштабу для осей X та Y, забезпечуючи пропорційність відображення на графіку.
axis tight	Використовується для автоматичного налаштування масштабу осей таким чином, щоб графік займав всю доступну площу.
polar	Використовується для побудови графіків у полярних координатах. Вона приймає радіальні значення і відображає їх відповідно до відстані від початку координат та кута.
loglog	Використовується для побудови графіків в логарифмічному масштабі.
semilogx	Використовується для побудови графіків з лінійною шкалою по осі тYа логарифмічною шкалою по осі X.

Функція	Опис функції
stem	Використовується для побудови графіків з використанням стовпчиків, які вказують значенням на вісі Y. Ця функція особливо корисна для відображення дискретних даних або дискретних подій.
linspace	Використовується для генерації рівномірно розподілених значень між двома вказаними крайніми точками.

Приклад побудови графіка функції $y(x) = x^2$ показано на рис.1.12.

```
1  clc, clear all
2  x = linspace(0, 10);
3  y = x.^2;
4  plot(x, y);
```

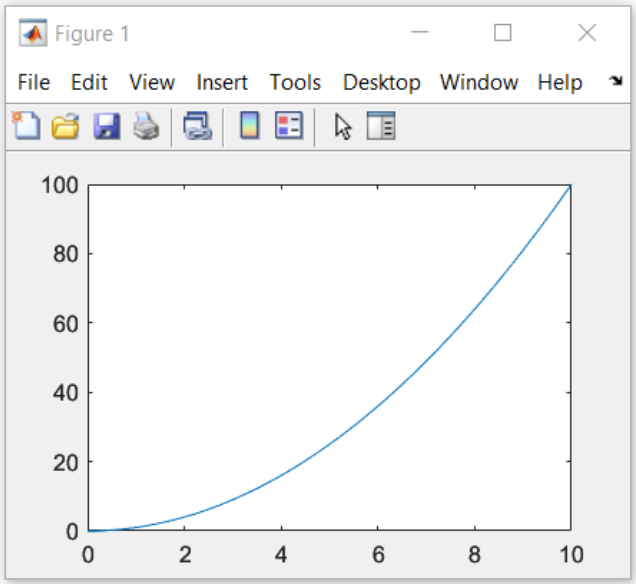


Рис 1.12 Приклад побудови графіка функції $y(x) = x^2$

Функція **linspace()** створює вектор з рівномірно розподіленими значеннями в заданому діапазоні.

У прикладі вектор **x** містить 100 точок, рівномірно розподілені між 0 та 10. За допомогою оператора «двокрапка» - «:», можна створювати вектори з заданим кроком, наприклад,

x = 0:0.1:10.

Якщо існує дві функції *f(x)* та *g(x)*, які мають значення на різних шкалах, можна використати **plotyy()** для того, щоб побудувати графіки цих функцій на двох різних шкалах.

Наприклад, необхідно побудувати графіки функцій *f(x) = sin(x)* та *g(x) = 10cos(x)* на двох різних шкалах, можна використати код, який показано на рис. 1.13.

```
x = 0:0.1:10;
f = sin(x);
g = 10*cos(x);

[ax,h1,h2] = plotyy(x,f,x,g,'plot');

xlabel('x');
ylabel(ax(1),'sin(x)');
ylabel(ax(2),'10cos(x)');

title('Графіки функцій на двох різних шкалах');

set(h1,'LineWidth',2);
set(h2,'LineWidth',2);

grid on;

legend('sin(x)', '10cos(x)');
```

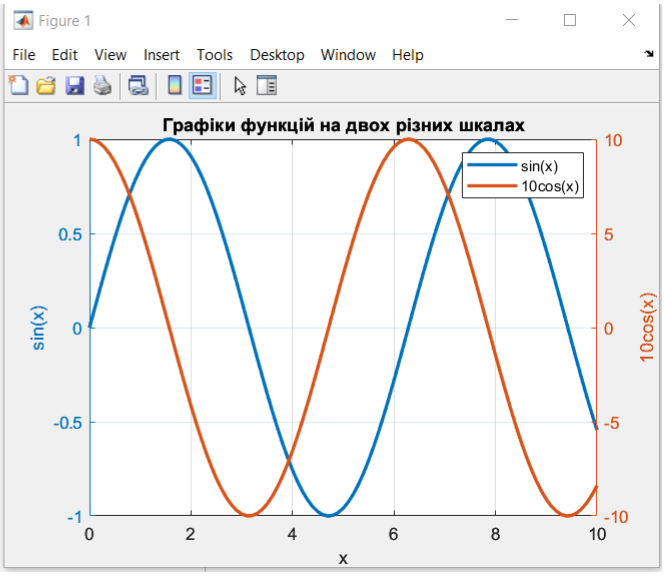


Рис 1.13 Приклад побудови графіків двох функцій з різними шкалами на вісі Y з використанням функції **plotyy()**

Код, наведений на рис.1.13 створить графіки функцій *f(x)* та *g(x)* на двох різних шкалах. Перший графік буде побудований по лівій шкалі, а другий - по правій. Додати підписи осей до графіку можна за допомогою функцій **xlabey** та **ylabel,** заголовок – функції **title**. За допомогою функції **legend** додається пояснення до кожної з ліній, якій функції вона відповідає.

Приклад побудови графіків в полярній системі координат наведено на рис. 1.14.

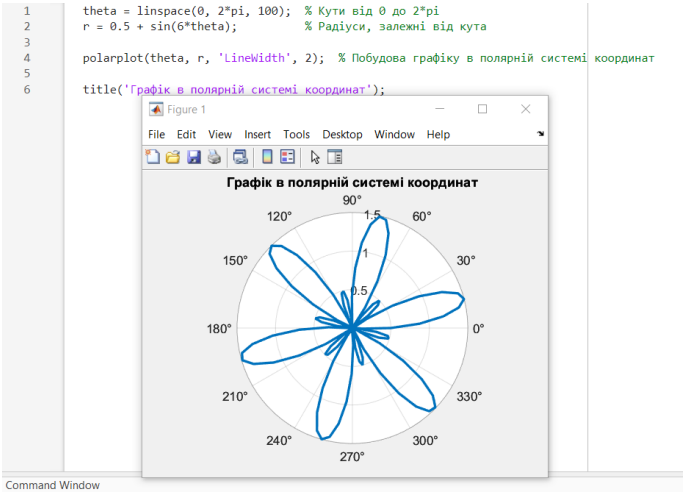


Рис. 1.14 Приклад побудови графіку в полярній системі координат

У прикладі, який наведено на рис.1.14 використання функції **polarplot** для побудови графіку в полярній системі координат. Кути **theta** - від 0 до 2π і відповідні радіуси **r**, залежні від кута.

Застосування функції **plot** для побудови графіку залежності y від x наведені на рис.1.15, та **loglog** - для побудови графіку у логарифмічному масштабі (рис.1.16) - та **semilogx (**рис.1.17, а**)** і **semilogy (**рис.1.17, б**)** - для побудови графіку у полулогарифмічному масштабі - рис. 1.17.

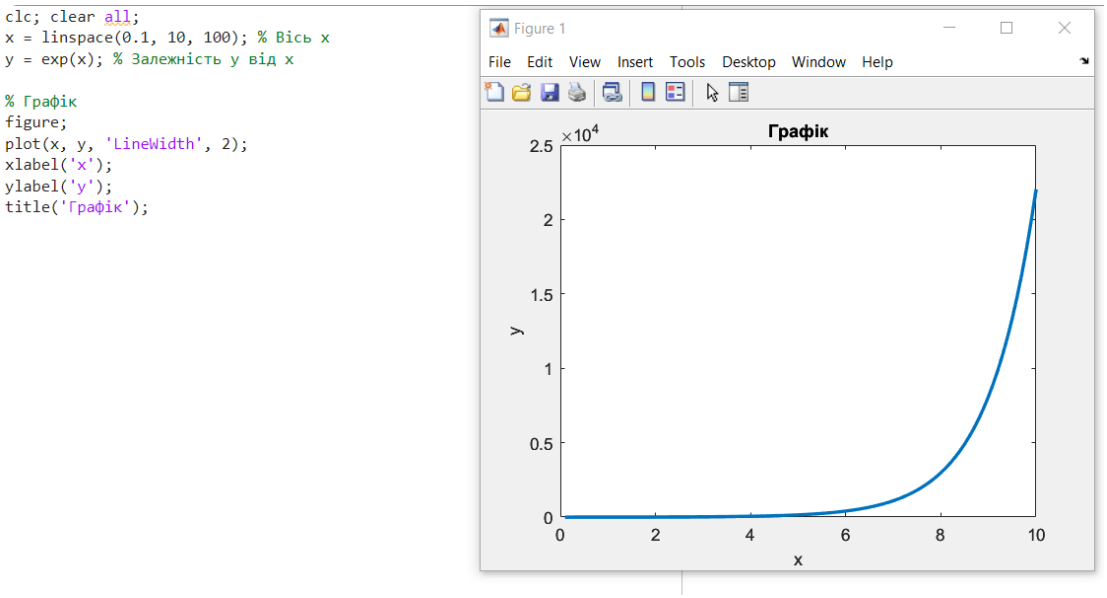


Рис. 1.15 Приклад використання функції

```
% Генеруємо дані
x = logspace(1, 10, 100); % Масив значень для осі x
y = logspace(1, 100, 100); % Масив значень для осі y

% Графік у логарифмічному масштабі» при використанні функції loglog
loglog(x, y, 'r-', 'LineWidth', 2);

% Налаштування осей та міток
xlabel('X-axis');
ylabel('Y-axis');
title('Графік у логарифмічному масштабі» при використанні функції loglog');
```

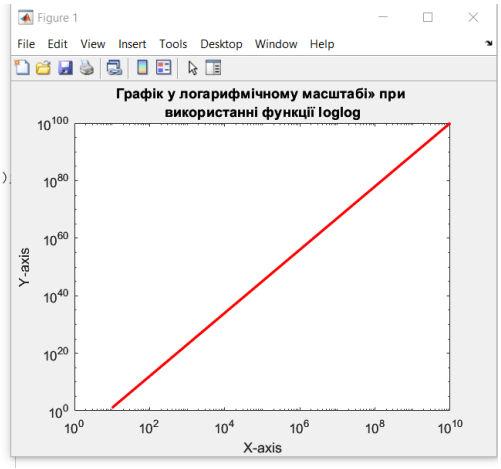
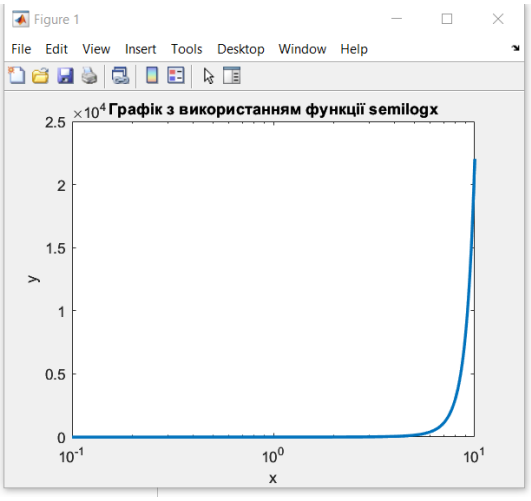


Рис. 1.16 Приклад використання функції **loglog**

```
clc; clear all;
x = linspace(0.1, 10, 100); % Вісь x
y = exp(x); % Залежність y від x


% Графік з використанням функції semilogx
figure;
semilogx(x, y, 'LineWidth', 2);
xlabel('x');
ylabel('y');
title('Графік з використанням функції semilogx');

% Графік з використанням функції semilogy
figure;
semilogy(x, y, 'LineWidth', 2);
xlabel('x');
ylabel('y');
title('Графік з використанням функції semilogy');
```

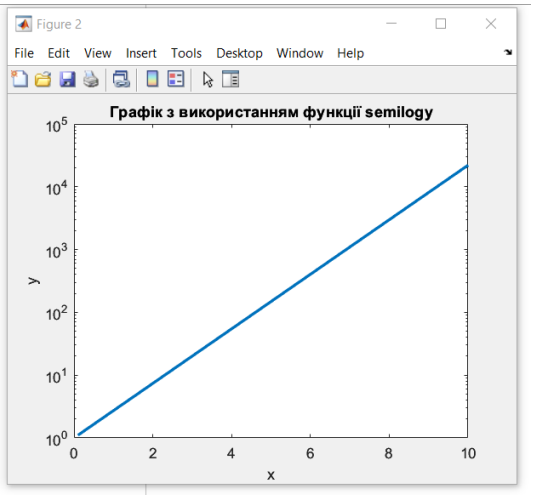


а)

```
clc; clear all;
x = linspace(0.1, 10, 100); % Вісь x
y = exp(x); % Залежність y від x


% Графік з використанням функції semilogx
figure;
semilogx(x, y, 'LineWidth', 2);
xlabel('x');
ylabel('y');
title('Графік з використанням функції semilogx');

% Графік з використанням функції semilogy
figure;
semilogy(x, y, 'LineWidth', 2);
xlabel('x');
ylabel('y');
title('Графік з використанням функції semilogy');
```



б)

Рис. 1.17 Приклади використання функцій: а) **semilogx; б) semilogy**

1.2. Завдання до виконання комп'ютерного практикуму

Завдання 1. Обчисліть арифметичний вираз відповідно до варіанту згідно з табл.1.2.

Таблиця 1.2

Варіанти індивідуальних завдань

Варіант	**Завдання**
1	$\frac{\left(16\frac{5}{8}+15{,}08\right)*\frac{9}{10}}{\frac{5}{11}-\frac{4\frac{3}{5}}{0{,}12}}$
2	$\frac{\frac{9}{10}*9}{\frac{5}{11}-\frac{4\frac{3}{5}}{0{,}12}+\frac{5}{8}*\frac{81}{9}*(4-3)}$
3	$\frac{0{,}02+\frac{20}{0{,}1}-13\frac{3}{2}}{12\frac{4}{3}*(\frac{4}{7}+\frac{\frac{5}{8}}{4}*\frac{4}{10})}$
4	$\frac{444{,}4+(55{,}7*\frac{50}{4}+28*0{,}0001)}{5*(12{,}4+\frac{12}{7}*61{,}0001)}$
5	$\frac{0{,}2*\frac{8}{7}+81{,}8*(2000/1222{,}4)}{\frac{9}{10-91}}$
6	$99{,}9*\frac{\frac{8}{9}*100{,}4-\frac{88{,}5}{81{,}6}}{5+99{,}9}$
7	$\frac{81}{7}+71{,}9*\frac{7{,}4+\frac{6{,}6}{1{,}1}}{6{,}4}+9{,}4$
8	$\frac{4{,}4+(550{,}7*\frac{1}{4}+2*0{,}0003)}{5*(12{,}4+\frac{12}{7}*\frac{661{,}0001}{1})}$

Варіант	Завдання
9	$\dfrac{\left(188\frac{5}{8}-15{,}08\right)*\frac{9}{100}}{\frac{5}{11}-\frac{4\frac{3}{5}}{0{,}12}+78*(10{,}1-6{,}8)}$
10	$\dfrac{0{,}11*\frac{10}{7}+11{,}8*(3000/1242{,}4)}{\frac{90}{11-91}+\frac{51}{3}}$

Завдання 2. Виконати операції з матрицями згідно власного варіанту згідно з табл.1.3:

2.1. Видалити з матриці **A** третій стовпець, а з матриці **B** другий рядок (табл.1.3). В матриці **A** змінити значення елемента з координатами (2, 1) на номер свого варіанту за списком. Розрахувати суму рядків, стовпців та діагоналі матриці **B**.

Таблиця 1.3

Варіанти індивідуальних завдань

Варіант	Матриці
1.	$A=\begin{bmatrix}7&2&5\\10&4&6\\3&5&9\end{bmatrix}, B=\begin{bmatrix}5&9&1\\0&11&2\\1&3&9\end{bmatrix}$
2.	$A=\begin{bmatrix}1&2&3\\1&2&4\\10&8&9\end{bmatrix}, B=\begin{bmatrix}5&7&1\\0&0&2\\2&3&9\end{bmatrix}$
3.	$A=\begin{bmatrix}3&2&5\\3&9&2\\3&6&9\end{bmatrix}, B=\begin{bmatrix}1&3&2\\1&3&2\\1&3&10\end{bmatrix}$
4.	$A=\begin{bmatrix}7&2&5\\10&3&6\\2&5&4\end{bmatrix}, B=\begin{bmatrix}0&9&11\\10&1&5\\1&4&12\end{bmatrix}$
5.	$A=\begin{bmatrix}1&2&9\\6&4&6\\3&9&9\end{bmatrix}, B=\begin{bmatrix}9&9&1\\8&11&2\\7&3&6\end{bmatrix}$

Варіант	Матриці
6.	$A=\begin{bmatrix}7 & 12 & 5\\ 20 & 4 & 6\\ 3 & 15 & 9\end{bmatrix}, B=\begin{bmatrix}5 & 9 & 11\\ 2 & 11 & 2\\ 1 & 13 & 19\end{bmatrix}$
7.	$A=\begin{bmatrix}8 & 2 & 5\\ 9 & 3 & 6\\ 3 & 5 & 4\end{bmatrix}, B=\begin{bmatrix}5 & 8 & 8\\ 0 & 11 & 2\\ 8 & 3 & 9\end{bmatrix}$
8.	$A=\begin{bmatrix}1 & 0 & 1\\ 10 & 1 & 6\\ 0 & 5 & 0\end{bmatrix}, B=\begin{bmatrix}5 & 9 & 1\\ 8 & 5 & 6\\ 1 & 3 & 5\end{bmatrix}$
9.	$A=\begin{bmatrix}2 & 2 & 2\\ 10 & 14 & 6\\ 13 & 2 & 9\end{bmatrix}, B=\begin{bmatrix}5 & 7 & 9\\ 8 & 3 & 2\\ 1 & 3 & 9\end{bmatrix}$
10.	$A=\begin{bmatrix}17 & 21 & 15\\ 0 & 4 & 16\\ 3 & 3 & 9\end{bmatrix}, B=\begin{bmatrix}7 & 9 & 7\\ 0 & 0 & 2\\ 0 & 3 & 9\end{bmatrix}$

2.2 Видалити перший рядок і третій стовпець «магічної» матриці **M** розміром 3×3. Отриману матрицю скласти з матрицею **A** в якій заздалегідь необхідно обернути рядки в стовпці, а стовпці в рядки (табл.1.3).

2.3 Об'єднати в одну матрицю «магічну» матрицю та матрицю **A** розміром 3×3, матрицю **B =A · 5**, матрицю C і матрицю **D=C+4** таким чином, щоб отримана матриця мала розмірність 6×6. В одержаній матриці замінити елемент з координатами (3, 4) на 100 (табл.1.3).

2.4 Обчислити результат добутку елементу матриці **M** з координатами (2, 3) на синус значень першого стовпця цієї ж матриці (табл.1.4).

Таблиця 1.4

Варіанти індивідуальних завдань

Варіанти	Завдання	Матриця
1, 3	2.2	$A=\begin{bmatrix}7 & 2 & 5\\ 10 & 4 & 6\\ 3 & 5 & 9\end{bmatrix}, B=\begin{bmatrix}5 & 9 & 1\\ 0 & 11 & 2\\ 1 & 3 & 9\end{bmatrix}$
2, 4	2.3	$A=\begin{bmatrix}1 & 5 & 3 & 4\\ 5 & 6 & 0 & 10\\ 2 & 0 & 1 & 8\end{bmatrix}, B=\begin{bmatrix}5 & 7 & 1\\ 0 & 0 & 2\\ 2 & 3 & 9\end{bmatrix}$

Варіанти	Завдання	Матриця
3, 6	2.4	$A = \begin{bmatrix} 3 & 0 & 1 \\ 4 & 2 & 9 \\ 19 & 8 & 3 \end{bmatrix}$, $\mathrm{M} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 0 & 9 \\ 12 & 0 & 11 & 8 \end{bmatrix}$
4, 8	2.5	$A = \begin{bmatrix} 2 & 2 & 2 \\ 10 & 14 & 6 \\ 13 & 2 & 9 \end{bmatrix}$ $C = \begin{bmatrix} 5 & 2 & 7 \\ 7 & 4 & 12 \\ 1 & 3 & 9 \end{bmatrix}$.
5, 10	2.6	$\mathrm{M} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 0 & 9 \\ 12 & 0 & 11 & 8 \end{bmatrix}$

Завдання 3. Виконати операції з матрицями згідно власного варіанту згідно з табл.1.5:

Таблиця 1.5

Варіанти індивідуальних завдань

Варіанти	Завдання
1, 3	Створити матрицю **A** з нульовими елементами розміром 5×5 та одиничну матрицю B розміром 2×3. Замінити верхню центральну частину матриці **A** елементами матриці **B**.
2, 4	Створити матрицю **A** з випадковими елементами розміром 4×5. Переставити стовпці відносно вертикальної осі, а рядки переставити відносно горизонтальної осі.
3, 6	Створити матрицю **A** розміром 3×6 з випадковими, що розподілені по закону Гауса. Повернути цю матрицю проти годинникової стрілки на 90° .
4, 8	Створити матрицю **A** розміром 2×4 з одиницями по головній діагоналі та всіма іншими елементами – нулями. Змінити розмір матриці на 8×1.
5, 10	Створити матрицю **A,** в якій головною діагоналлю є вектор **V = − [2 3 6 4]** . Обернути отриману матрицю.

Завдання 4. Побудувати та оформити графічне оформлення графіків функції f(x) на відрізку [a;b] з кроком h відповідно до табл.1.6.

Таблиця 1.6

Варіанти індивідуальних завдань

Варіант	**Функція f(x)**	**Параметри**		
		a	**b**	**h**
1	$\frac{x^2}{1+0.25\sqrt{x}}$	0	1,6	0,16
2	$\frac{\cos(\pi x^2)}{\sqrt{1-3x}}$	-1	0	0,1
3	$\sqrt{1+4x} * \sin(\pi x)$	0,1	0,8	0,05
4	$\frac{e^{\frac{x}{3}}}{1+x^2}$	1,4	2,2	0,2
5	$e^{2x}+x^2-1$	0,35	3,35	0,35
6	$(e+x) * \sin(\pi\sqrt{x-1})$	1,9	2,9	0,1
7	$\sqrt{2+3x} * tg\frac{\pi x^3}{2}$	0,1	0,7	0,07
8	$\sqrt{2+3x} * \ln(1+3x^2)$	-0,1	0,9	0,1
9	$\sqrt[3]{x^2+3} * \cos(\frac{\pi x}{2})$	1	2,5	0,15
10	$(4+7x) * \sin(\pi\sqrt[3]{1+x})$	0	9	0,9

1.3. Контрольні запитання

1. Які команди використовуються для введення змінних в командному вікні MatLab?
2. Які арифметичні операції можна виконувати в командному вікні MatLab?
3. Як можна створити одновимірний масив в MatLab?
4. Як можна використовувати індексування для доступу до елементів масиву в MatLab?

5. Як можна витягнути підмасив з матриці в MatLab?
6. Як можна виконати множення двох матриць в MatLab?
7. Які функції використовуються для побудувави графіка в MatLab?
8. Як можна побудувати 3D графік в MatLab?
9. Як можна побудувати два графіки з різною шкалою на осі Y в MatLab в одному графічному вікні?

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 2
МОДЕЛЮВАННЯ ДИНАМІЧНИХ СИСТЕМ З ВИКОРИСТАННЯМ ПАКЕТА SIMULINK. СТВОРЕННЯ S-МОДЕЛЕЙ, ВИКОРИСТОВУЮЧИ РОЗДІЛ SOURCES ТА SINK

Мета роботи

Ознайомитись з основними принципами моделювання динамічних систем та роллю пакета SimuLink, навчитись створювати S-моделі, використовуючи блоки розділу Sources та Sink.

2.1. Теоретичні відомості

Пакет SimuLink дозволяє здійснювати дослідження (моделювання у часі) поводження динамічних нелінійних систем. Утворення чисельної моделі досліджуваної системи здійснюється шляхом графічного складання у спеціальному вікні схеми з’єднань елементарних візуальних блоків, що містяться в бібліотеках SimuLink.

Кожний блок фактично являє собою математичну програму. Лінії з’єднання блоків перетворюються на зв’язки між цими програмами, які дозволяють визначити послідовність виклику програм і пересилання інформації. У результаті такого складання утворюється програмна модель, яку надалі називатимемо S-моделлю і яка зберігається у файлі з розширенням **.mdl.** Такий процес утворення обчислювальних програм прийнято називати візуальним програмуванням [1].

SimuLink надає графічний інтерфейс, що дозволяє користувачам створювати моделі за допомогою блоків та з'єднувальних ліній. Це спрощує процес розроблення і робить його інтуїтивно зрозумілим навіть для звичайних користувачів.

Перед початком роботи з пакетом SimuLink варто на панелі інструментів командного вікна програми MatLlab відкрити вкладку з інструментами **Home,**

знайти значок  (рис. 2.1) та натиснути на нього. Відповідно, на екрані з'явиться вікно SimuLink Start Page (рис. 2.2)

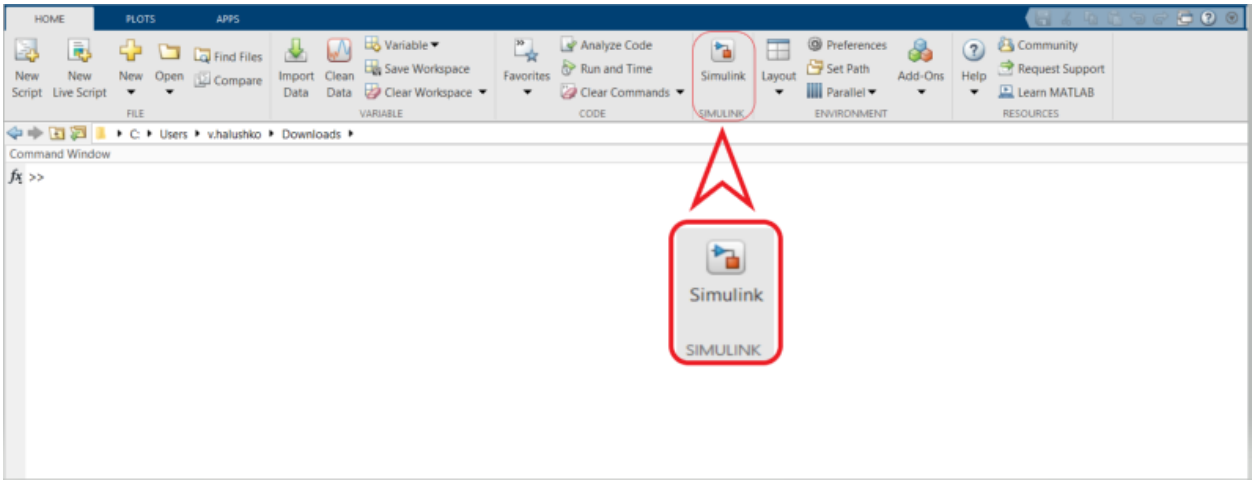


Рис. 2.1 Командне вікно програми MatLab

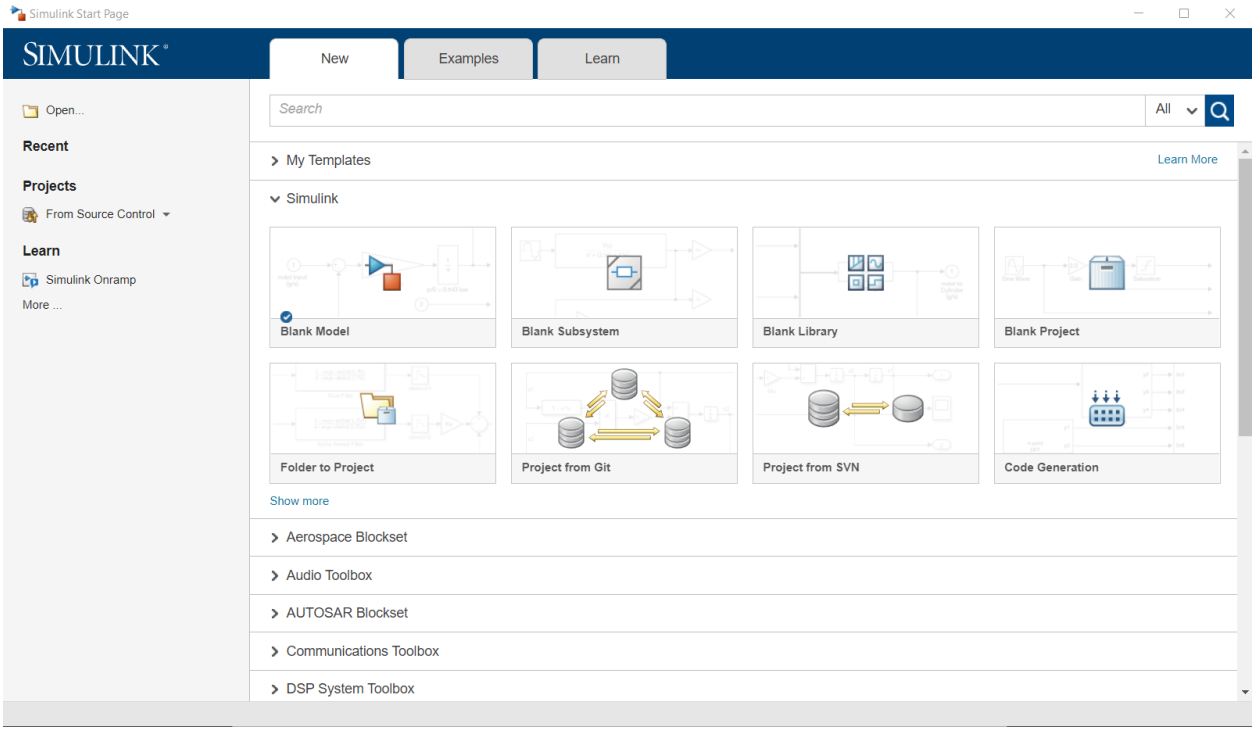
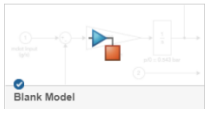


Рис. 2.2 Вікно SimuLink Start Page

В вікні SimuLink Start Page вибираємо вкладення SimuLink та натискаємо на позначку 

для створення порожньої моделі Blank Model. Після чого перед нами з'являється редактор SimuLink (рис. 2.3).

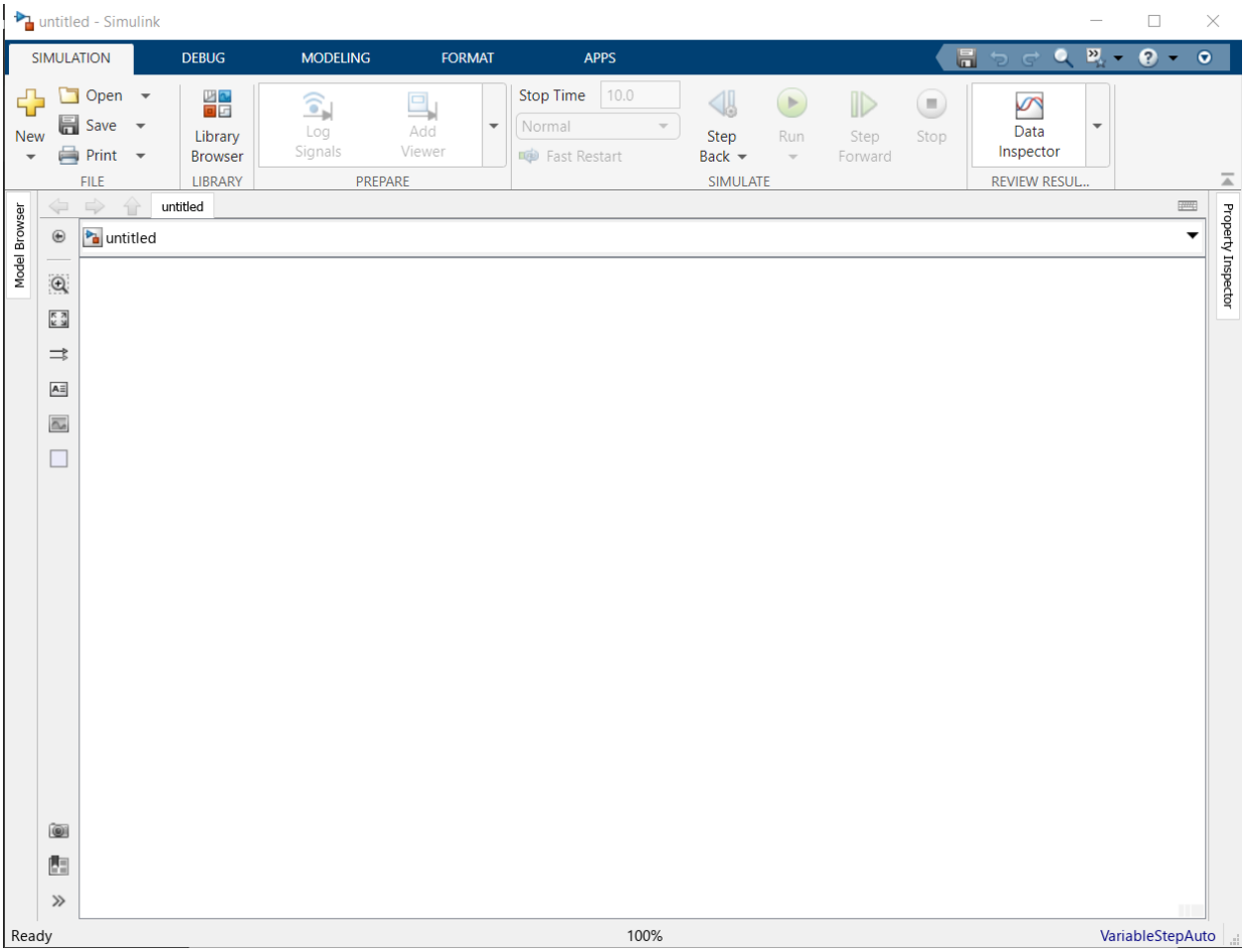


Рис. 2.3 Вікно редактора SimuLink

В головному пакеті SimuLink існує кілька вкладень (tabs) або розділів (рис 2.4), які надають доступ до різних функцій та інструментів. Основні вкладки включають Simulation, Debug, Modeling, Format і Apps.

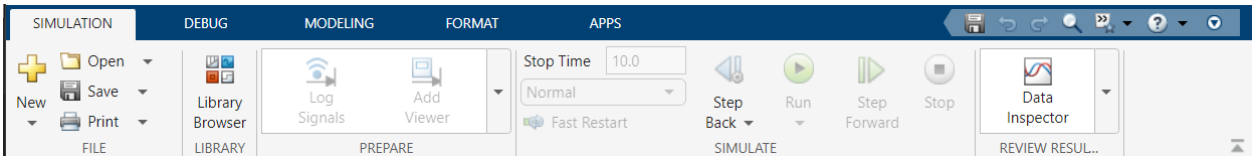


Рис. 2.4 Головне вікно пакету SimuLink

Вкладка «Simulation» (Симуляція). Розділ надає доступ до інструментів, пов'язаних з налаштуванням та виконанням симуляцій моделей SimuLink. Можна налаштувати параметри симуляції, запустити симуляцію, аналізувати результати та відтворювати сигнали.

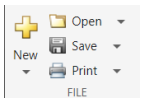
Вкладка «Debug» (Відлагодження). Цей розділ надає інструменти для відлагодження моделей SimuLink. Можна встановлювати точки зупину, крокувати по моделі, відстежувати значення сигналів та аналізувати потік виконання програми.

Вкладка «Modeling» (Моделювання). Розділ містить інструменти для редагування та модифікації моделей SimuLink. Можна додавати блоки, змінювати їх параметри, з'єднувати їх, організовувати підсистеми та налаштовувати поведінку моделі.

Вкладка «Format» (Форматування). Цей розділ надає інструменти для форматування та оформлення моделей SimuLink. Можна налаштовувати вигляд блоків, з'єднувальних ліній, текстових надписів та інших елементів моделі, щоб поліпшити їх зрозумілість та організацію.

Вкладка «Apps» (Додатки). Розділ містить набір додаткових інструментів та додатків, які можна використовувати з SimuLink. Він може включати спеціалізовані пакети, бібліотеки, інструменти аналізу та інші додатки, як SimuLink Coder, SimuLink Control Design, SimuLink Design Verifier, SimuLink Real-Time і багато інших. Ці додатки розширюють можливості SimuLink і надають спеціалізовані функції для певних областей застосування, таких як генерація коду, проектування регуляторів, верифікація моделей та виконання симуляцій в реальному часі.

Ці вкладки та їх функціональність дозволяють зручно і ефективно розробляти, моделювати та аналізувати системи за допомогою SimuLink. Вони допомагають керувати різними аспектами роботи з моделями, симуляціями та відлагодженням.

Для створення і збереження нової моделі потрібно перейти в вкладеннння Simulation та знайти набір інструментів з підписом File 

та відповідно створити нову S-модель або ж зберегти її натиснувши на вкладення Save (рис.2.5).

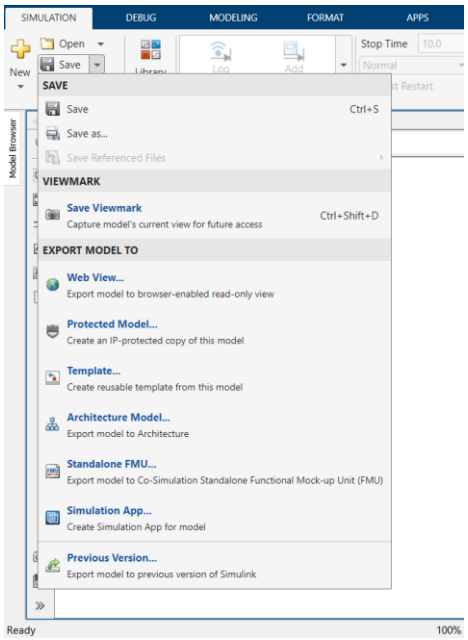


Рис. 2.5 Вікно збереження S-моделі

Натиснувши на вкладення Save, з'явиться вікно збереження файлу, в якому в полі «Ім'я файлу(File name)» потрібно назвати дану модель, яка матиме тип файлу **.slx.** Наприклад, файл зі створеною моделлю матиме назву та розширенням – «*new_model.slx*.».

Для виклику браузера бібліотеки SimuLink шляхом натиснення на відповідну піктограму 

у лінійці інструментів командного вікна. Виникне вікно SimuLink Library Browser (рис. 2.6), яке містить перелік основних розділів бібліотеки SimuLink; більш зручним є користування вікном Library: SimuLink (рис. 2.7), яке викликається за допомогою контекстного меню. В ньому представлені графічні позначення розділів бібліотеки [1].

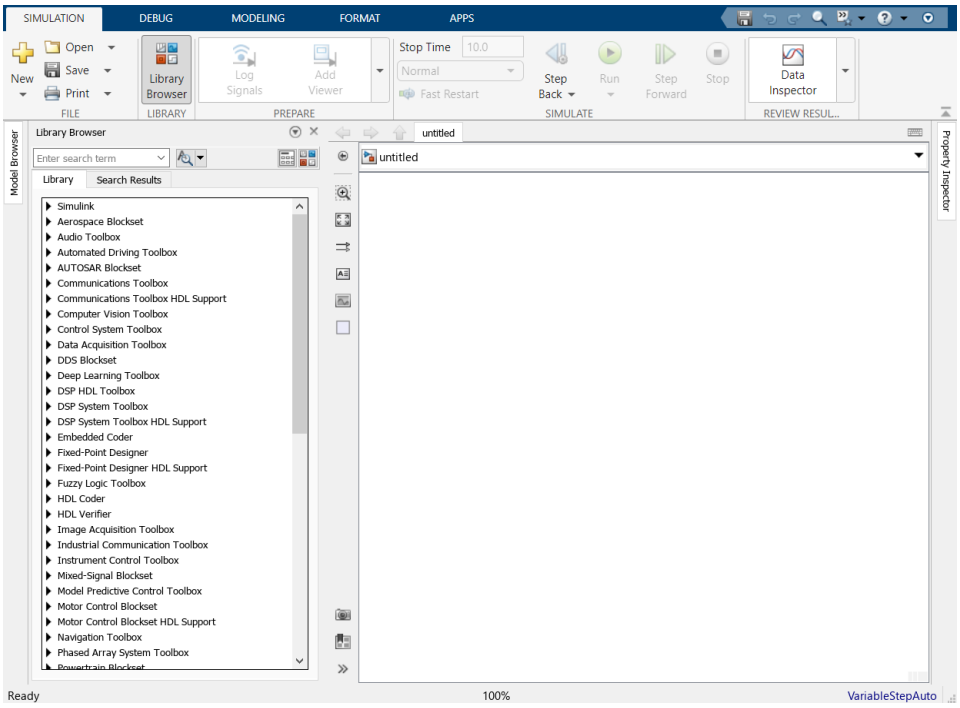


Рис. 2.6 Вікно браузера бібліотеки SimuLink

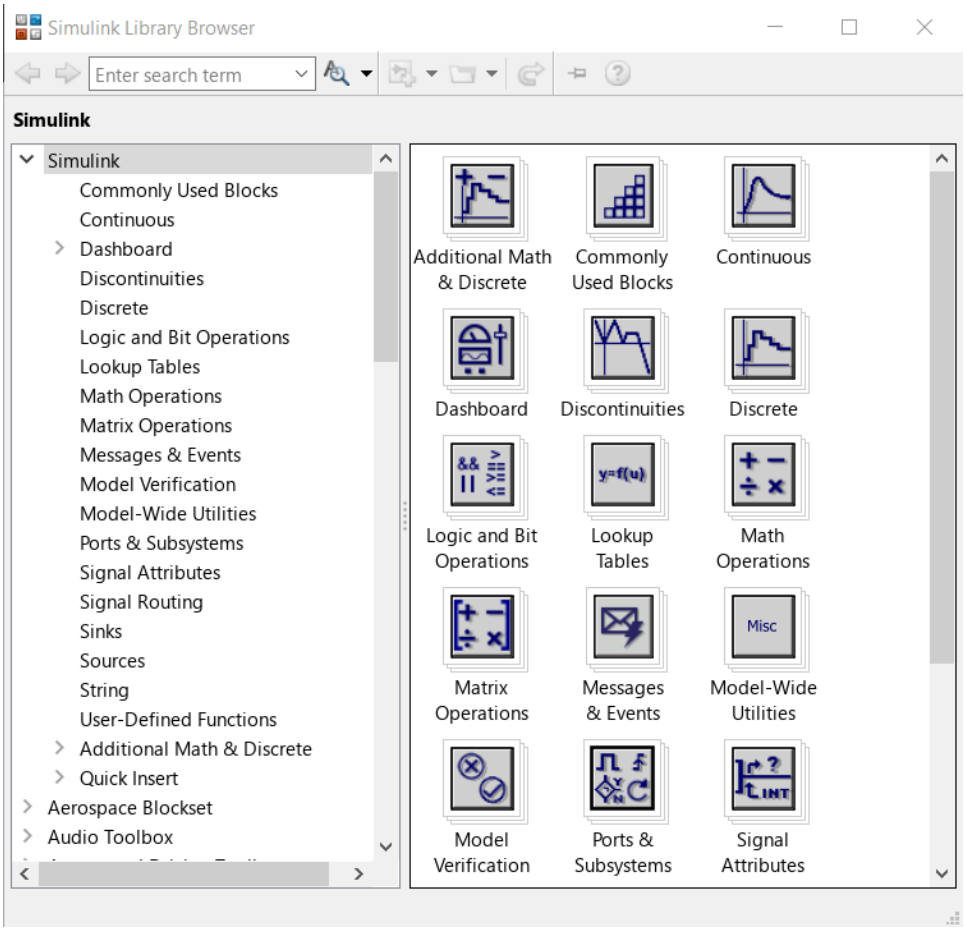


Рис. 2.7 Вікно Library Browser

В основі блок-схем S-моделей лежать елементарні блоки, які дозволяють зв’язати блок-схему з середовищем MatLab і забезпечити функціонування в ньому S-моделі як програми. Ці блоки містяться у головній бібліотеці пакету SimuLink, яка має ту саму назву. Бібліотека блоків SimuLink є набором візуальних об'єктів, використовуючи які можна з’єднуючи окремі модулі між собою лініями функціонального зв'язку, складати функціональну блок-схему будь-якого пристрою [1].

Бібліотека блоків складається з 17 розділів. Десять з них є головними і не можуть змінюватися користувачем [1]:

- **Sources** (Джерела);
- **Sinks** (Приймачі);
- **Continuous** (Лінійні неперервні елементи);
- **Discrete** (Дискретні елементи);
- **Discontinuities** (Розривні елементи)
- **Signal Routing** (Пересилання сигналів);
- **Math Operations** (Математичні операції);
- **Logic & Bit Operations** (Логічні та бітові операції);
- **User-defined Functions** (Функції, що визначаються користувачем);
- **Ports & Subsystems** (Порти та підсистеми).

2.1.1 Розділ Sinks (приймачі)

Розділ «Sinks» (приймачі) в SimuLink включає в себе блоки, які використовуються для відображення, збереження або виведення результатів симуляції моделі. Ці блоки допомагають спостерігати та аналізувати вихідні дані моделі.

На рис. 2.8 зображені блоки, які місяться в розділі Sinks бібліотеки.

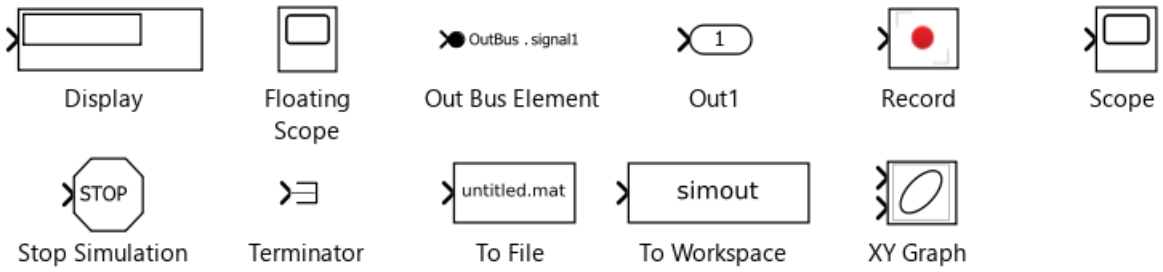


Рис. 2.8 Піктограми блоків розділу Sinks(приймачі)

Опис та функції блоків розділу Sinks [8]:

Display – відображає значення вхідних даних.

Floating Scope and Scope Viewer – відображення сигналів, згенерованих під час симуляції, без сигнальних ліній.

Out Bus Element – передає шину, нешинний сигнал або повідомлення на вихідний порт компонента моделі.

Out1 – з'єднує сигнали від системи до місця призначення за межами системи.

Record, XY Graph – записує данні до робочої області, до файлу або до робочої області та файлу.

Scope – відображає сигнали в часовій області.

Stop Simulation – зупиняє симуляцію, коли вхідний сигнал стає ненульовим. Симуляція завершує поточний часовий крок перед зупинкою.

Terminator – закриває блоки, вихідні порти яких не з'єднані з іншими блоками.

To file – записує дані вхідного сигналу у MAT-файл.

To Workspace – записує дані, підключені до його вхідного порту, у робочу область з моделі SimuLink.

2.1.2 Розділ Sources (джерела)

Блоки, що входять у розділ Sources (Джерела), призначені для формування сигналів, які забезпечують керування роботою S-моделі в цілому

або окремих її частин. Усі блоки-джерела мають по одному виходу і не мають входів [2].

Ці блоки дозволяють задати значення сигналу, яке може бути постійним, змінюваним в часі або генерованим за певним законом. Використання блоків з розділу Sources дозволяє створити початкові умови, вхідні сигнали або сигнали зовнішніх джерел для моделі.

Вони є важливим елементом при моделюванні динамічних систем, оскільки дозволяють задати початкові умови та вхідні сигнали для виконання симуляції. На рис. 2.9 зображено вікно розділу Sources бібліотеки з джерелами сигналів.

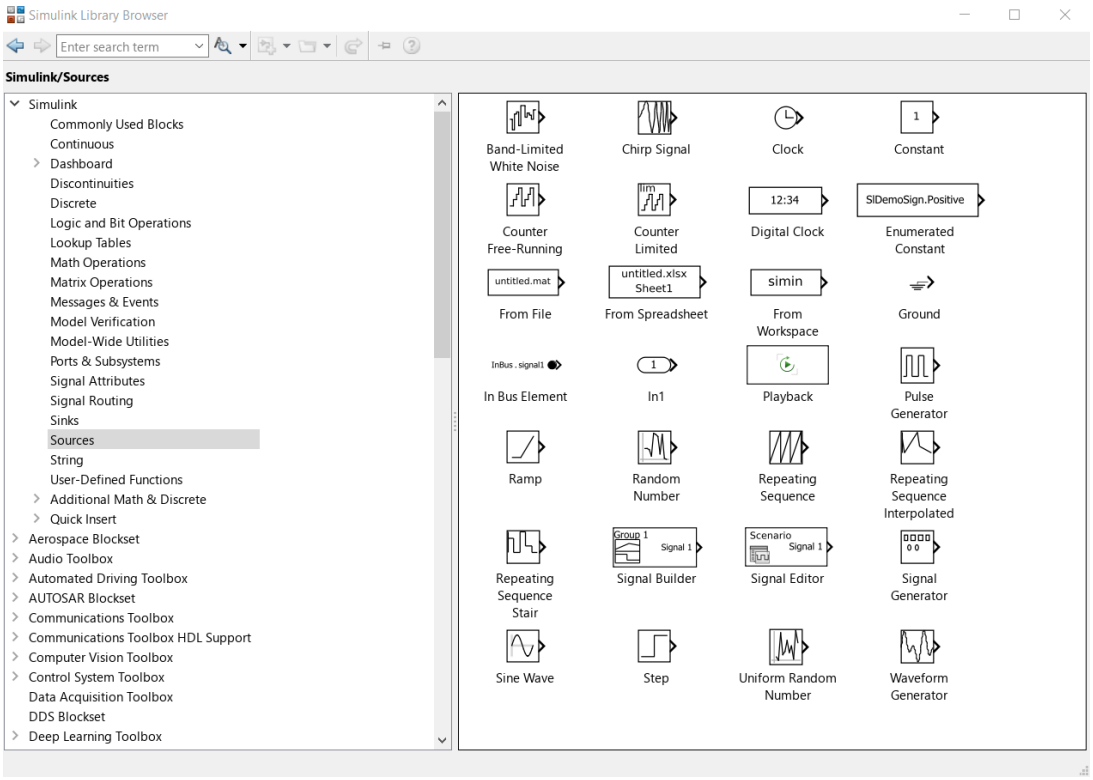


Рис. 2.9 Вікно розділу Sources

Для відображення стислого опису блоків та переліку його параметрів необхідно навести мишку на блок, який цікавить, та натиснути правою кнопкою на даний блок, а потім обрати *Block parameters* – в результаті відобразиться опис блоку. У даному випадку параметри блока доступні тільки для читання [2].

Приклад відображення параметрів блока Step представлений на рис.2.10.

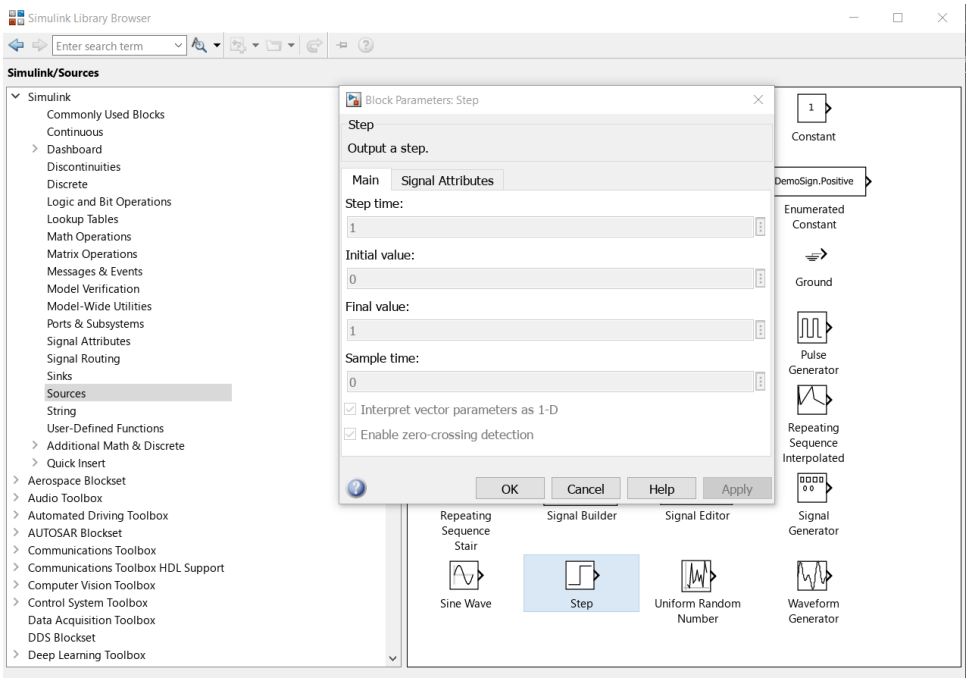


Рис. 2.10 Вікно параметрів блока Step

Для того, щоб почати роботу з блоками SimuLink, потрібно їх перетягнути з бібліотеки в робочу область редактору (рис. 2. 11).

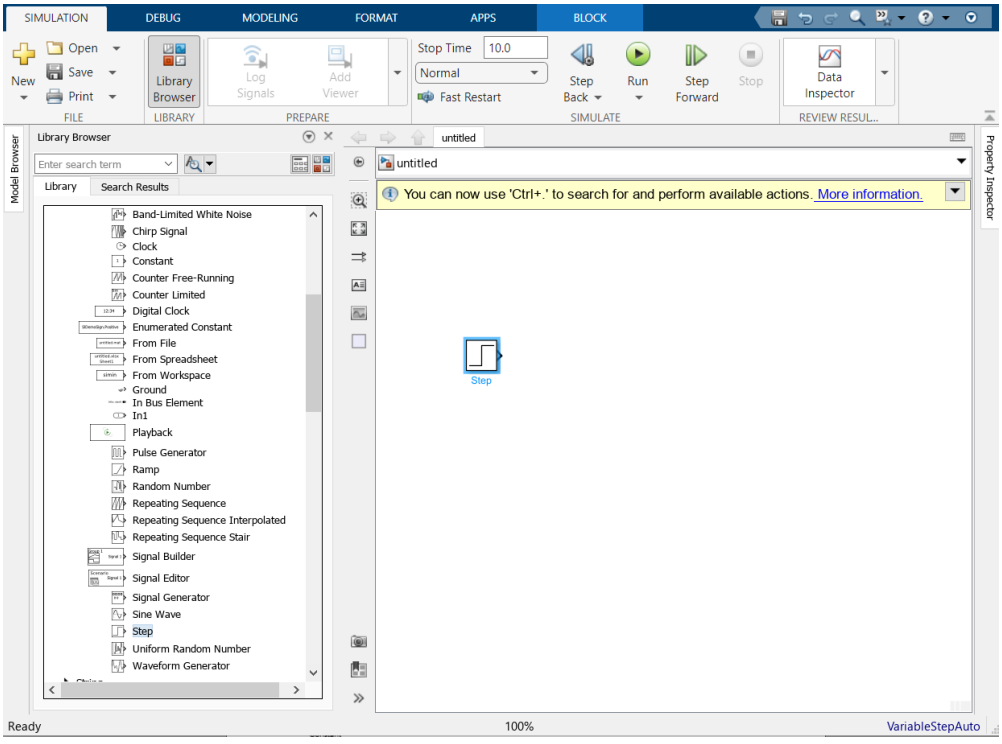


Рис. 2.11 Робота з блоками Sources в робочій області SimuLink

Зображення всіх блоків, які належать розділу Sources бібліотеки SimuLink представлені на рис. 2.12.

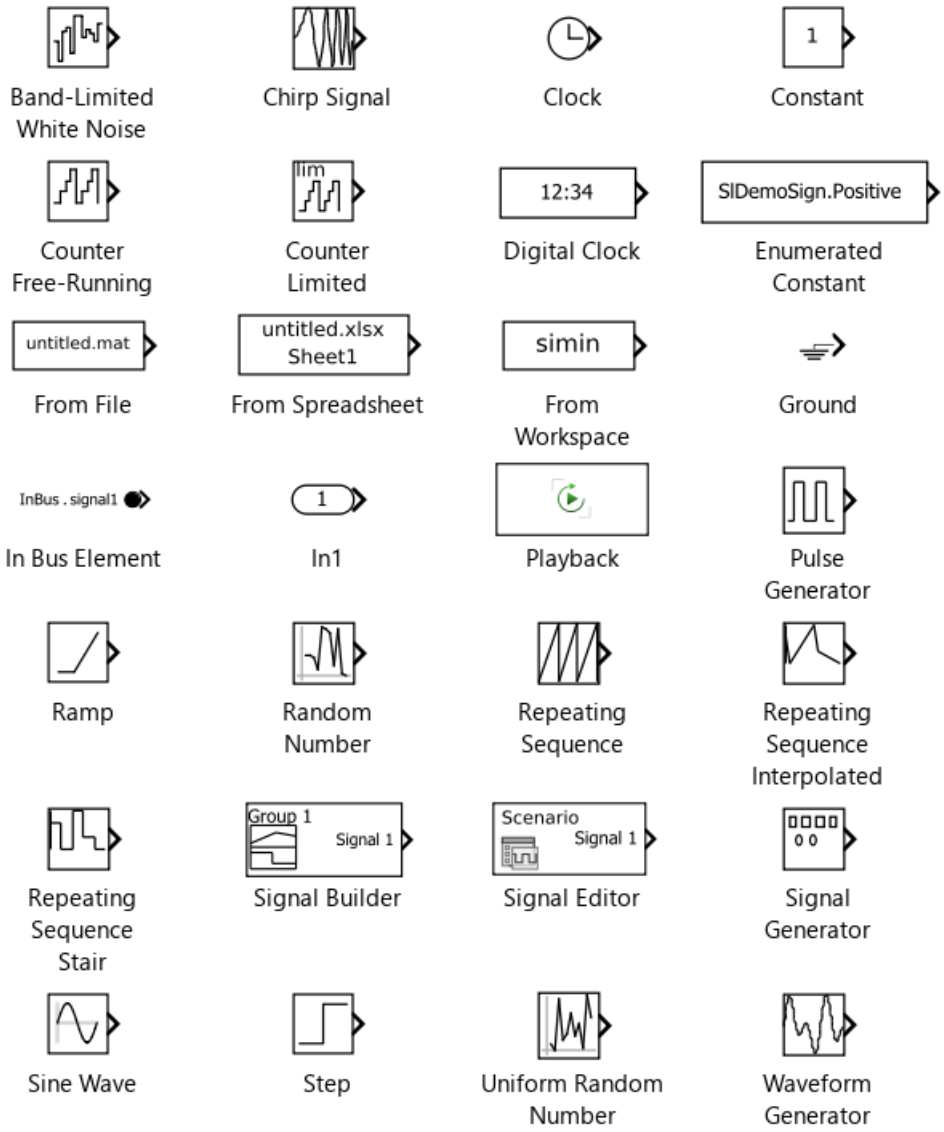


Рис. 2.12 Піктограми блоків розділу Sources

Опис та функції блоків розділу Sources [9]:

Band-Limited White Noise – генерує нормально розподілені випадкові числа, які підходять для використання в безперервних або гібридних системах.

Chirp Signal – генерує синусоїду, частота якої лінійно зростає з часом. Блок використовується для спектрального аналізу нелінійних систем. Блок генерує скалярний або векторний вихід.

Clock – виводить поточний час симуляції на кожному кроці симуляції. Цей блок корисний для інших блоків, яким потрібен час моделювання.

Constant – генерує дійсний або комплексний сигнал постійної величини (рис. 2.13). Забезпечує вхідний сигнал постійної величини.

Counter Free-Running – блок вільного запуску лічильника підраховує до досягнення максимального значення 2Гбіт, потім лічильник переповнюється до нуля і починає відлік знову.

Counter Limited – рахує до тих пір, поки не буде досягнута вказана верхня межа. Потім лічильник повертається до нуля і починає зворотній відлік. Лічильник завжди ініціалізується нулем.

Digital Clock – виводить час симуляції лише на вказаному інтервалі дискретизації. В інший час блок утримує вихідні дані на попередньому значенні.

Enumerated Constant – виводить скаляр, масив або матрицю перерахованих значень.

From File – завантажує дані в модель SimuLink з MAT-файлу і надає дані у вигляді сигналу або невіртуальної шини на виході блоку.

From Spreadsheet – зчитує дані з електронних таблиць Microsoft Excel (на всіх платформах) або CSV (лише для платформи Microsoft Windows зі встановленим Microsoft Office) і виводить дані у вигляді сигналу.

From Workspace – зчитує дані в SimuLink-модель з робочого простору і надає дані у вигляді сигналу або невіртуальної шини на виході блоку

Ground – з’єднується з блоками, вхідні порти яких не з’єднані з іншими блоками.

In Bus Element – використовується блок In Bus Element, для вибору елементу шини, шину, нешинний сигнал або повідомлення, пов’язане з портом.

Inport – зв’язує сигнали ззовні системи з системою.

Playback - блок для завантаження вхідних даних для моделювання. Блок Playback підтримує завантаження реальних або складних сигналів з

фіксованими або змінними розмірами, дискретних і неперервних сигналів, а також повідомлень.

Pulse Generator – генерує прямокутні імпульси через рівні проміжки часу (рис. 2.14).

Ramp – генерує сигнал, який починається у вказаний час і значення та змінюється із заданою швидкістю (рис. 2.15).

Random Number – може згенерувати повторювану послідовність, використовуючи будь-який блок випадкових чисел з тим самим невід'ємним початковим значенням і параметрами.

Repeating Sequence – виводить періодичний скалярний сигнал, форма якого задається за допомогою параметрів Time values і Output values.

Repeating Sequence Interpolated – виводить періодичну дискретну послідовність. Між точками даних блок використовує метод, який вказується у параметрі Lookup Method, щоб визначити вихідні дані.

Repeating Sequence Stair – виводить і повторює послідовність сходинок, яка задається за допомогою параметра Вектор вихідних значень.

Signal Editor – відображає, створює і редагує взаємозамінні сценарії, які містять сигнали.

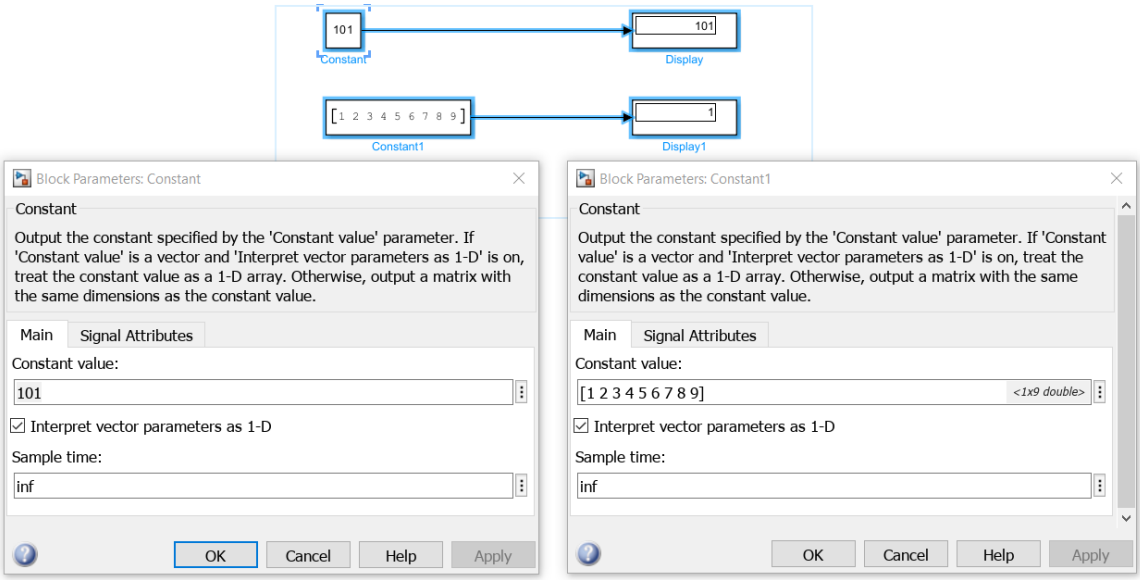
Signal Generator – створює одну з чотирьох різних форм сигналу: синус, квадрат, пилкоподібний, випадковий.

Sine Wave – виводить синусоїдальну форму сигналу (рис.2.16). Блок може працювати в режимі, заснованому на часі, або в режимі, заснованому на відліках.

Step – забезпечує крок між двома визначеними рівнями за вказаний час (рис. 2.17).

Uniform Random Number – генерує рівномірно розподілені випадкові числа на вказаному вами інтервалі (рис. 2.18).

Waveform Generator – виводить осцилограми на основі позначень сигналів (рис.2.19), які вводиться в таблиці Waveform Definition (Визначення осцилограми).



а) б)

Рис. 2.13 Блок Constant при:

а) одному значенню параметра постійного впливу; б) при більш ніж одному значенню параметра постійного впливу

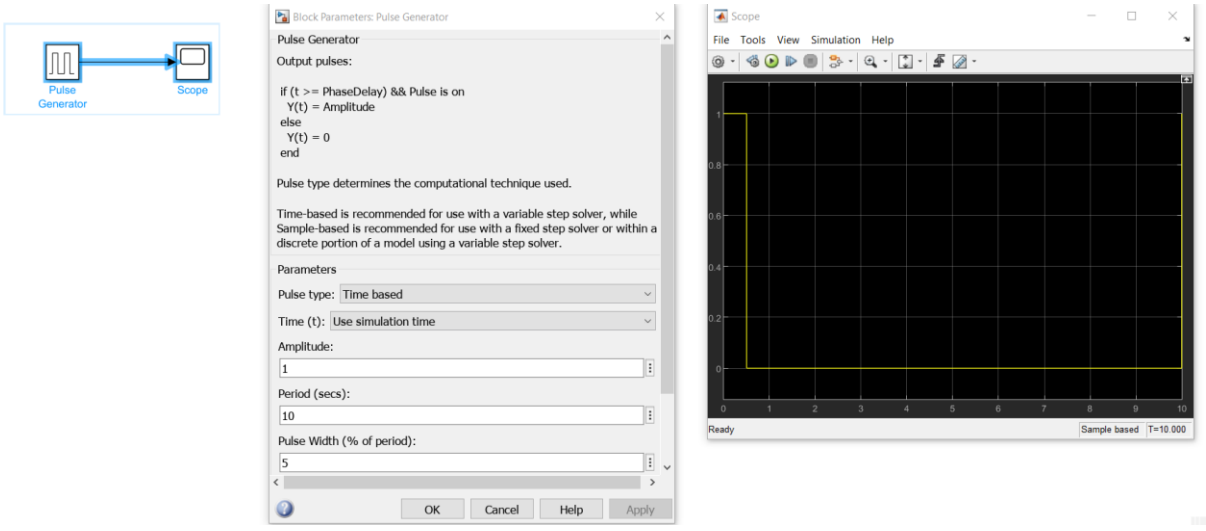


Рис. 2.14 Блок Pulse Generator (генератор прямокутних імпульсів)

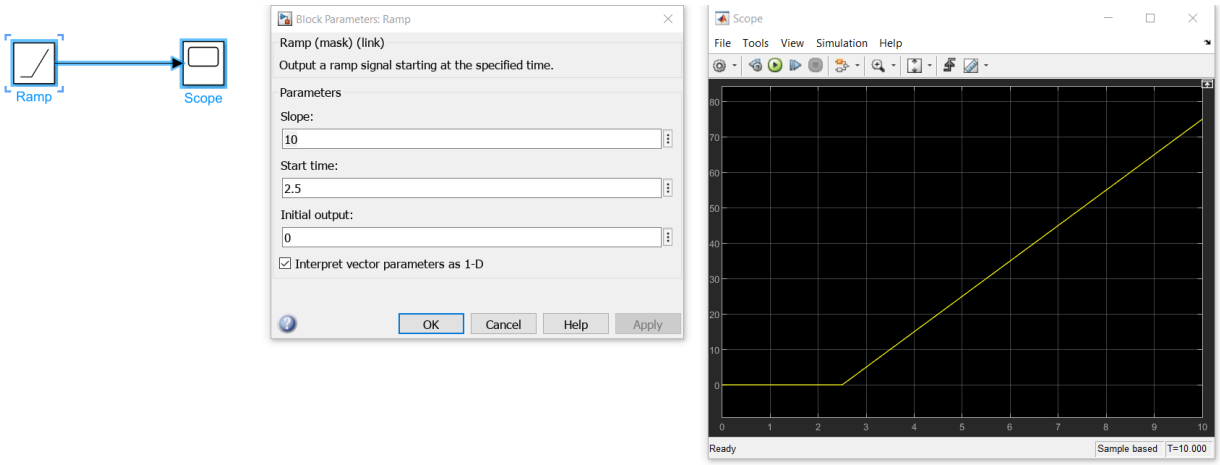


Рис. 2.15 Блок Ramp (джерело лінійно-наростаючого вливу)

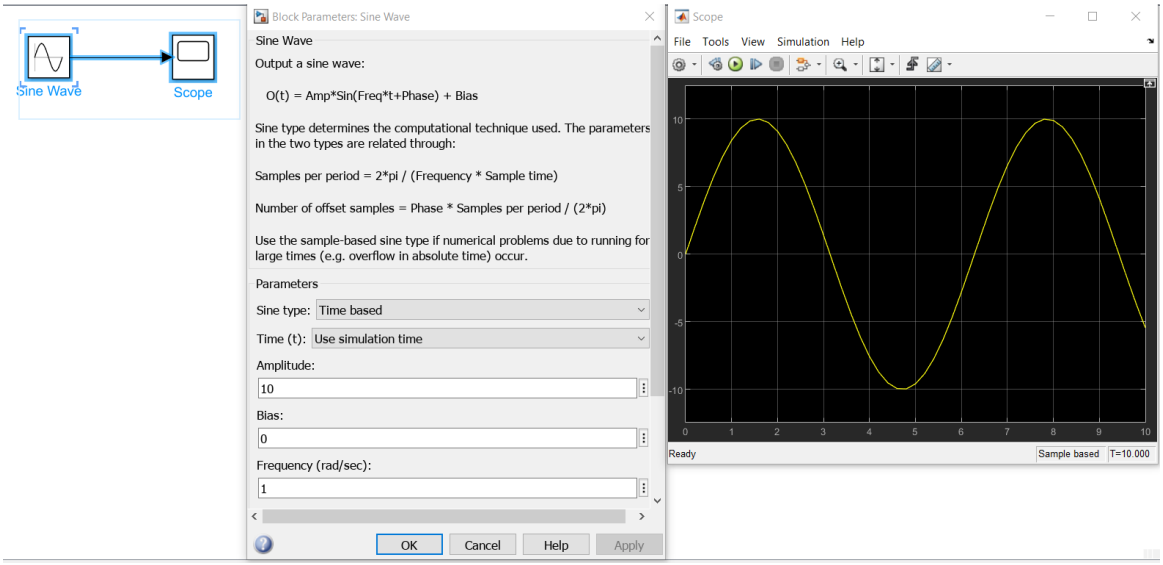


Рис. 2.16 Блок Sine Wave (генерує синусоїдальну форму сигналу)

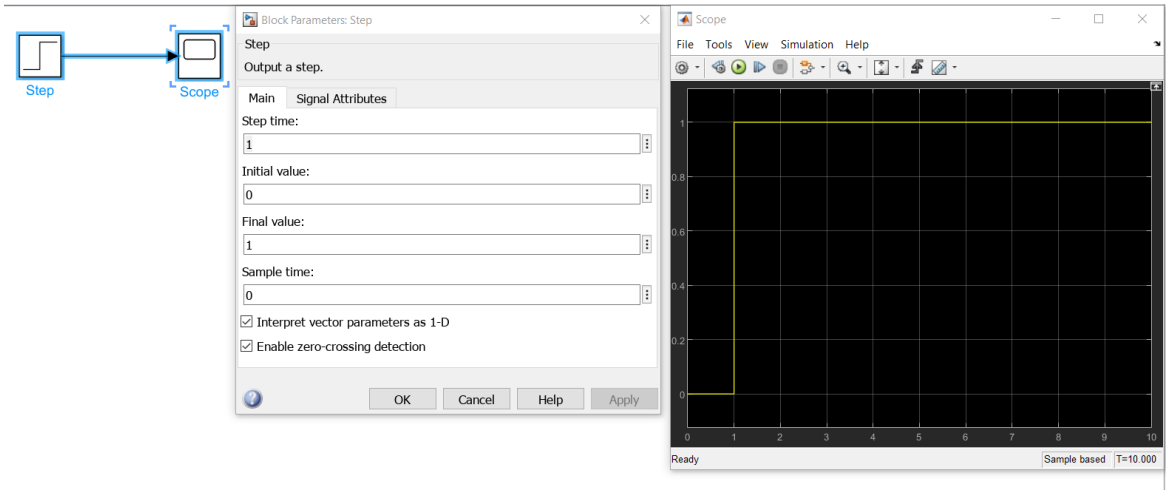


Рис. 2.17 Блок Step (джерело ступінчатого сигналу)

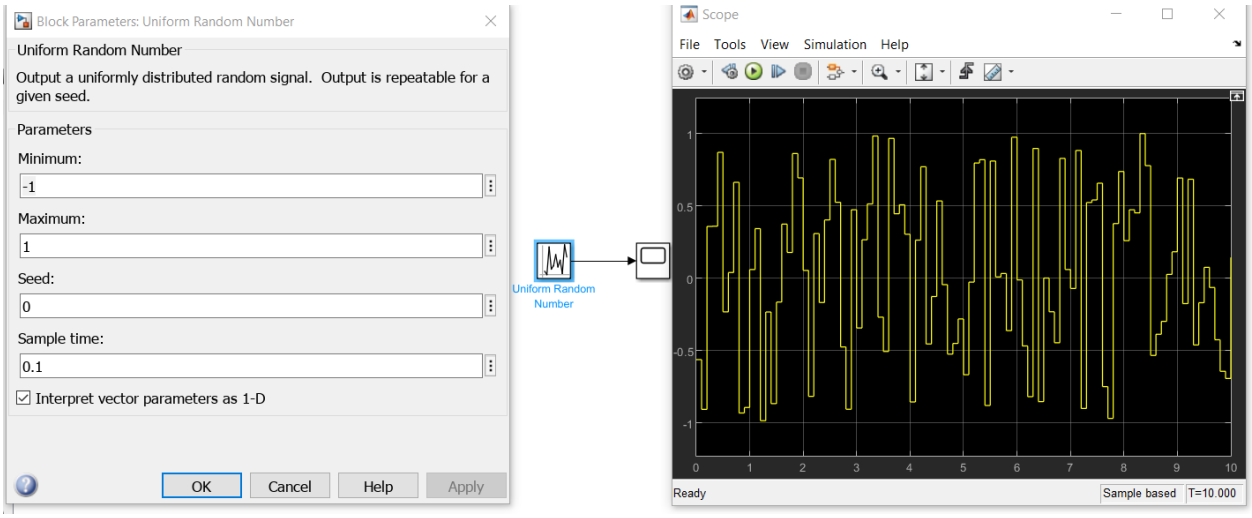


Рис. 2.18 Блок Uniform Random Number (джерело рівномірно розподілених випадкових чисел)

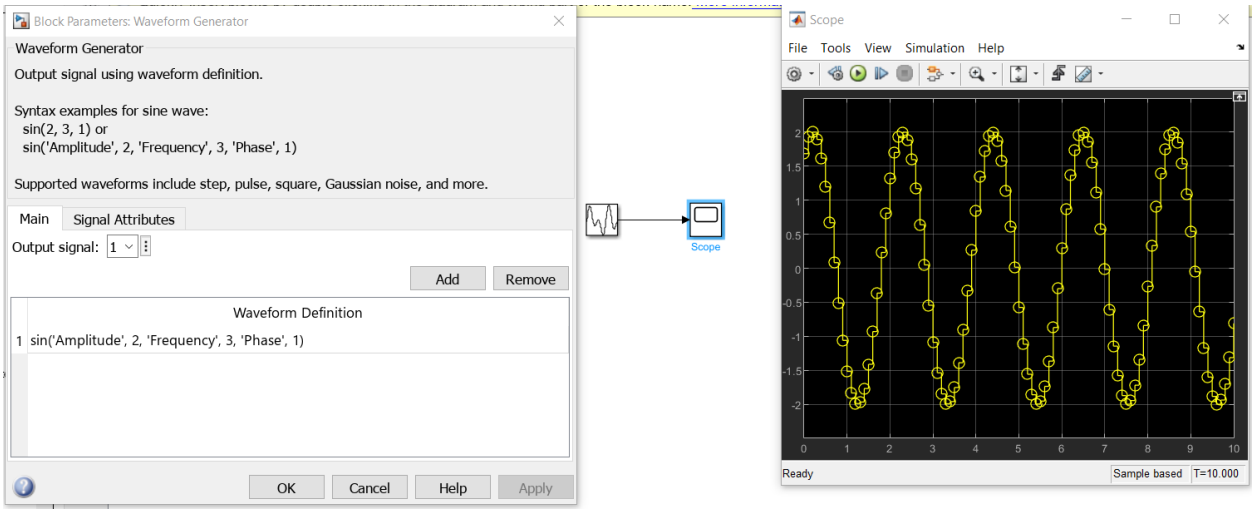


Рис. 2.19 Блок Waveform Generator (виводить осцилограми на основі позначень сигналів)

2.2. Завдання на виконання комп'ютерного практикуму

Завдання 1. Створити S-моделі, користуючись різноманітними джерела, відкалібрувати параметри джерел (відповідно до свого варіанту завдання) і вивести сигнали на виходах з моделей, використовуючи блок з розділу Sink.

1.1 Використати блок Constant, значення параметру Constant value (табл. 2.1);

1.2 Використати блок Pulse Generator (табл. 2.2);

1.3 Використати блок Sine Wave (табл. 2.3);

1.4 Використати блок Step (табл. 2.4);

1.5 Використати блок Waveform Generator (табл. 2.5).

Завдання 2. Змінити варіант (парні номери змінюють номер варіанту на непарний).

Завдання 3. Порівняти отримані графіки та зробити висновки.

Таблиця 2.1

Варіанти індивідуальних завдань

№	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Constant value	14	8	9	11	2	7	77	18	25	21

Таблиця 2.2

Варіанти індивідуальних завдань

№	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Amplitude	5	5,5	6	6,5	7	7,5	8	8,5	9	9,5
Period	1	1,5	2	2,5	3	3,5	4	4,5	5	5,5
Phase	2	2,5	3	3,5	4	4,5	5	5,5	6	6,5

Таблиця 2.3

Варіанти індивідуальних завдань

№	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Amplitude	15	15,5	16	16,5	17	17,5	18	18,5	19	19,5
Period	11	11,5	12	12,5	13	13,5	14	14,5	15	15,5
Phase	12	12,5	13	13,5	14	14,5	15	15,5	16	16,5

Таблиця 2.4

Варіанти індивідуальних завдань

№	1	2	3	4	5	6	7	8	9	10
Step time	17	5,5	18	16,56	19	13,5	18,5	18,5	12	17,5
Invalid value	18	1,05	14	12,6	13,5	11,5	14	19	18	15,5
Final value	1	18,5	13	13,6	14,5	18,5	15	5,15	19	16

Таблиця 2.5

Варіанти індивідуальних завдань

№	1	2	3	4	5	6	7	8	9	10
Amplitude	3	1	3,5	6,5	7	5	8	18	4,5	5,5
Period	2	4	5	6	7	8	9	10	12	14
Phase	1	2	3	4	5	6	7	8	9	10

2.3. Контрольні запитання

1. Що таке пакет SimuLink і яку роль він відіграє в моделюванні динамічних систем?
2. Які основні компоненти включає розділ Sources в SimuLink?
3. Яке призначення блоку «Constant» в розділі Sources? Як його використовувати для створення постійних сигналів?
4. Які параметри можна налаштувати для блоку «Constant»?
5. Які блоки в розділі Sources використовуються для генерації змінних сигналів?
6. Що таке блок «Step»? Яке його призначення в розділі Sources?
7. Як використовувати блок «Step» для генерації сигналу зі сталим рівнем до певного часу, а потім різко змінити його?

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 3
СТВОРЕННЯ S-МОДЕЛЕЙ, ВИКОРИСТОВУЮЧИ РОЗДІЛ MATH OPERATION

Мета роботи

Ознайомитись з розділом Math Operation бібліотеки SimuLink та поглибити з побудови S-моделей та їх можливостей, а також визначити їх потенційне значення для рішення практичних завдань з використанням розділу Math Operation.

3.2. Теоретичні відомості

Розділ Math Operation містить 37 блоків, що реалізують різні математичні компоненти. Можливість завдання безлічі математичних компонентів з описаними користувачами властивостями має важливе значення для виконання доступного для користувача математичного моделювання як простих і складних систем і пристроїв. Цей розділ дозволяє ефективно реалізувати багато математичних операцій, такі як додавання, віднімання, множення і ділення, використовувати прості математичні функції, пошук мінімуму або максимуму, рішення рівнянь, а також є можливість визначення коренів рівнянь [10].

Розділ Math Operation зазвичай містить такі основні функції та операції:

- Арифметичні операції. До них входять додавання (+), віднімання (-), множення (*), ділення (/) та інші арифметичні операції, які дозволяють виконувати розрахунки з числовими значеннями.
- Логічні операції. Ці операції включають умови порівняння, такі як рівність (==), нерівність (!=), більше (>), менше (<), більше або дорівнює (>=), менше або дорівнює (<=) і логічні оператори, такі як І (AND), АБО (OR) та НЕ (NOT), що дозволяють здійснювати логічні перевірки та прийняття рішень на основі умов.

- Математичні функції. Розділ Math Operation також надає доступ до різних математичних функцій, таких як sin, cos, tan, exp, log і багато інших. Ці функції дозволяють виконувати складні математичні обчислення та операції над числовими значеннями.

- Умовні вирази. За допомогою розділу Math Operation можна використовувати умовні вирази, що базуються на математичних операціях. Наприклад, можна встановити умови для виконання певної дії або вибору варіанту дії в залежності від значень змінних.

На рис. 3.1 зображено вікно розділу бібліотеки Math Operation з блоками, які до неї входять.

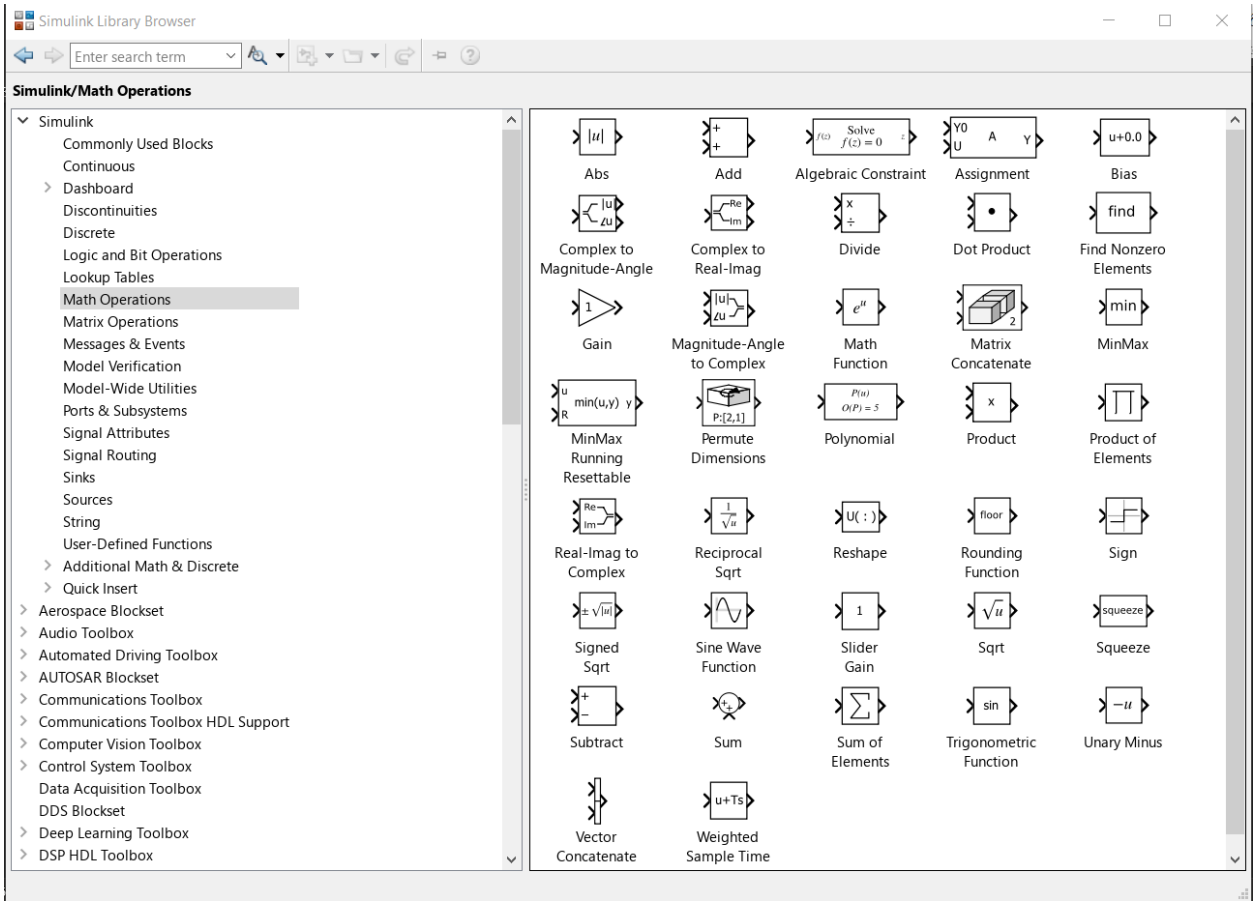


Рис. 3.1 Вікно розділу бібліотеки Math Operation

Назви блоків, що входять до розділу Math Operation та опис функцій, що ними виконуються [11]:

Abs - виводить абсолютне значення вхідних даних.

Add - виконує додавання або віднімання на своїх входах.

Algebraic Constraint- обмежує вхідний сигнал f(z) до z або 0 і виводить алгебраїчний стан z. Блок виводить значення, яке створює 0 або z на вході.

Assignment - призначає значення визначеним елементам сигналу.

Bias - додає зсув, або зміщення, до вхідного сигналу згідно з формулою Y = U + зсув, де U - вхід блоку, а Y – вихід.

Complex to Magnitude-Angle - виводить амплітуду та/або фазовий кут вхідного сигналу, залежно від налаштування параметра Output.

Complex to Real-Imag - виводить дійсну та/або уявну частину вхідного сигналу, залежно від налаштування параметра Output.

Divide - виводить результат ділення першого входу на другий.

Dot Product - генерує точковий добуток вхідних векторів.

Find Nonzero Elements - знаходить всі ненульові елементи вхідного сигналу і повертає лінійні індекси цих елементів.

Gain - множить вхідні дані на постійну величину (коефіцієнт підсилення). Вхідні дані та коефіцієнт підсилення можуть бути скаляром, вектором або матрицею.

Magnitude-Angle to Complex - перетворює входи амплітуди і фазового кута в комплексний вихід. Вхідні дані кута повинні бути в радіанах.

Math Function - виконує багато поширених математичних функцій.

Matrix Concatenate - об'єднує вхідні сигнали для створення нескалярного сигналу, який можна ітеративно обробляти за допомогою підсистеми, наприклад, підсистеми for-each, while-itterator або for-itterator.

MinMax - виводить або мінімальний, або максимальний елемент або елементи входів.

MinMax Running Resettable - виводить мінімальне або максимальне значення всіх минулих входів u. На вхід може подаватися скалярний, векторний або матричний сигнал.

Permute Dimensions - впорядковує розміри масиву в порядку, визначеному дипорядком вектора

Polynomial - створює поліноміальну функцію;

Product - використовується для обчислення ділення та множення елементів;

Product of Elements - вводить один скаляр, вектор або матрицю.

Real-Imag to Complex - перетворює реальні та/або уявні входи у вихідний сигнал комплексного значення.

Reciprocal Sqrt - обчислює значення, зворотне до квадратного кореня.

Reshape - змінює розмірність вхідного сигналу на розмірність, яка вказується за допомогою параметра Output dimensionality. Наприклад, за допомогою цього блоку можна змінити N-елементний вектор на матричний сигнал 1 на N або N на 1.

Rounding Function - округлює кожен елемент вхідного сигналу для отримання вихідного сигналу.

Sign - показує, що для векторних та матричних входів блок виводить вектор або матрицю, де кожен елемент є знаком відповідного вхідного елемента.

Signed Sqrt - квадратний корінь від модуля вхідного сигналу, помножений на знак вхідного сигналу.

Sine Wave Function - виводить синусоїдальну форму сигналу.

Slider Gain - виконує скалярний коефіцієнт підсилення, який можна змінювати під час моделювання.

Sqrt - обчислює квадратний корінь, квадратний корінь зі знаком або обернений квадратний корінь із вхідного сигналу.

Squeeze - видаляє з багатовимірного вхідного сигналу синглетні виміри.

Subrtact - обчислює різницю, виконує віднімання від значення верхнього входу «+» значення, що приходить на нижній вхід «-».

Sum - обчислює суму або різницю поточних значень сигналів, що надходять на відповідні входи блоку.

Sum of Elements - обчислення суми або різниці елементів.

Trigonometric Function - виконує загальні тригонометричні функції і виводить результат в радіанах або обертах за хвилину.

Unary Minus - встановлює негативне значення сигналу.

Vector Concatenate - об'єднує вхідні сигнали для створення нескалярного сигналу, який можна ітеративно обробити за допомогою підсистеми, наприклад, підсистеми for-each, while-iterator або for-iterator.

Weighted Sample Time Math - виводить одне з цих значень, залежно від заданої операції та контексту виконання функції, яка містить цей блок.

Ці блоки надають широкі можливості для виконання різних математичних операцій і обробки числових значень в рамках створення S-моделей.

Блоки **Sum**, **Add**, **Subtract** і **Sum of Elements** здійснюють підсумовування сигналів, що надаються до їхніх входів [1].

Блок Sum може використовуватися у двох режимах:

- додавання вхідних сигналів (у тому числі з різними знаками);
- підсумовування елементів вектора, що надходить на вхід блока.

Для керування режимами роботи блока використовується два параметри (рис. 3.2):

– Icon Shape (Форма зображення),

– List of signs (Список знаків).

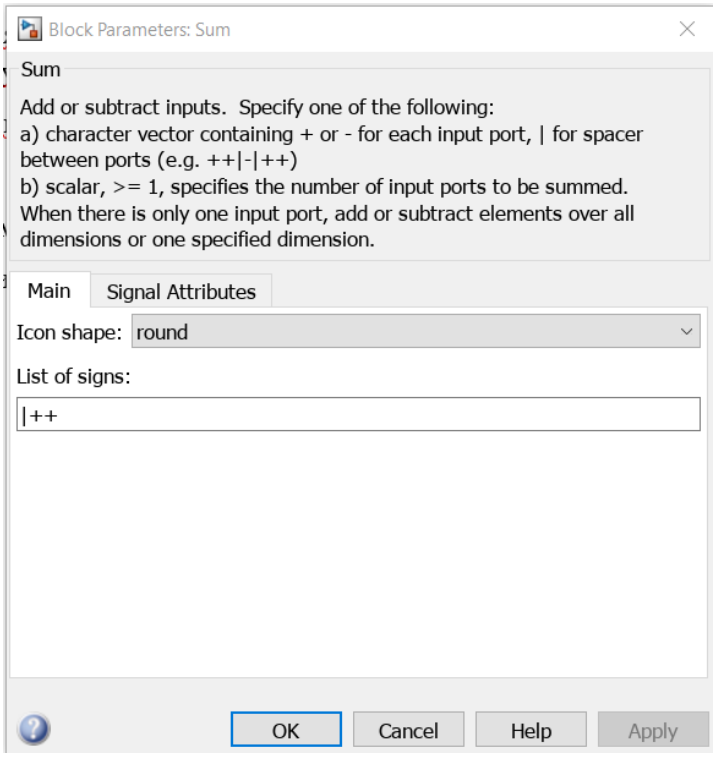


Рис. 3.2 Вікно налаштування блоку Sum

Перший параметр може набувати двох значень (рис.3.3): round (круглий) і rectangular (прямокутний).

Рис. 3.3 Круглий та прямокутний блок Sum та Add

Значення другого параметра можуть задаватися одним із трьох способів [1]:

- у вигляді послідовності знаків «+» або «–»; при цьому кількість знаків визначає кількість входів блока, а сам знак – полярність відповідного вхідного сигналу;

- у виді цілої додатної і більше одиниці константи; значення цієї константи визначає кількість входів блока, а усі входи вважаються додатними;
- у виді символу «1», який указує, що блок використовується в другому режимі.

На рис. 3.4 представлений приклад застосування блоку Sum.

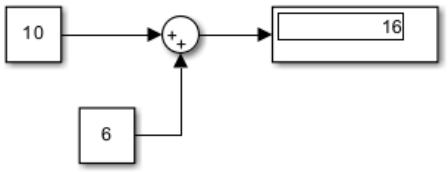
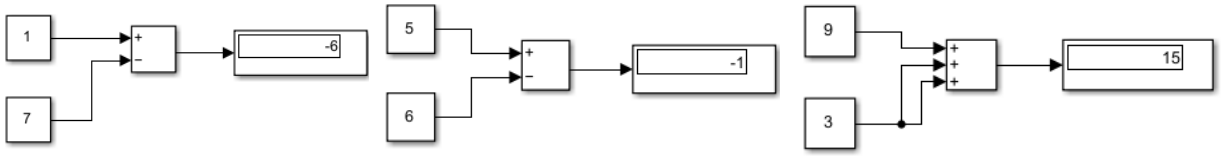


Рис. 3.4 Приклад застосування блоку Sum

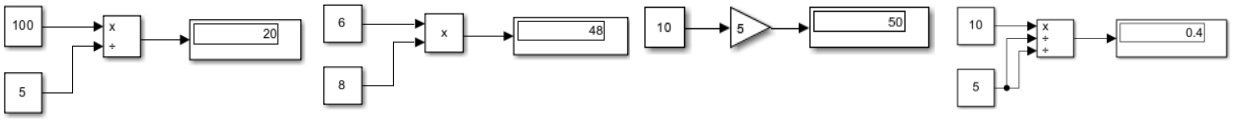
Як вже було вказано вище, окрім блоку Sum подібну операцію додавання двох вхідних сигналів виконують блоки Add, Subtract і Sum of Elements. Розглянемо детальніше ці блоки на прикладах, які зображено на рис 3.5.



а) б) в)

Рис. 3.5 Приклади застосування блоків:
а) Add; б) Subtract; в) Sum of Elements

На рис. 3.6 наведений приклад використання ділення та множення: Devide, Product, Gain, Product of Element:



а) б) в) г)

Рис. 3.6 Приклади застосування блоків:
а) Devide, б) Product, в) Gain, г) Product of Element

На рис. 3.7 наведений приклад застосування блоку Abs.

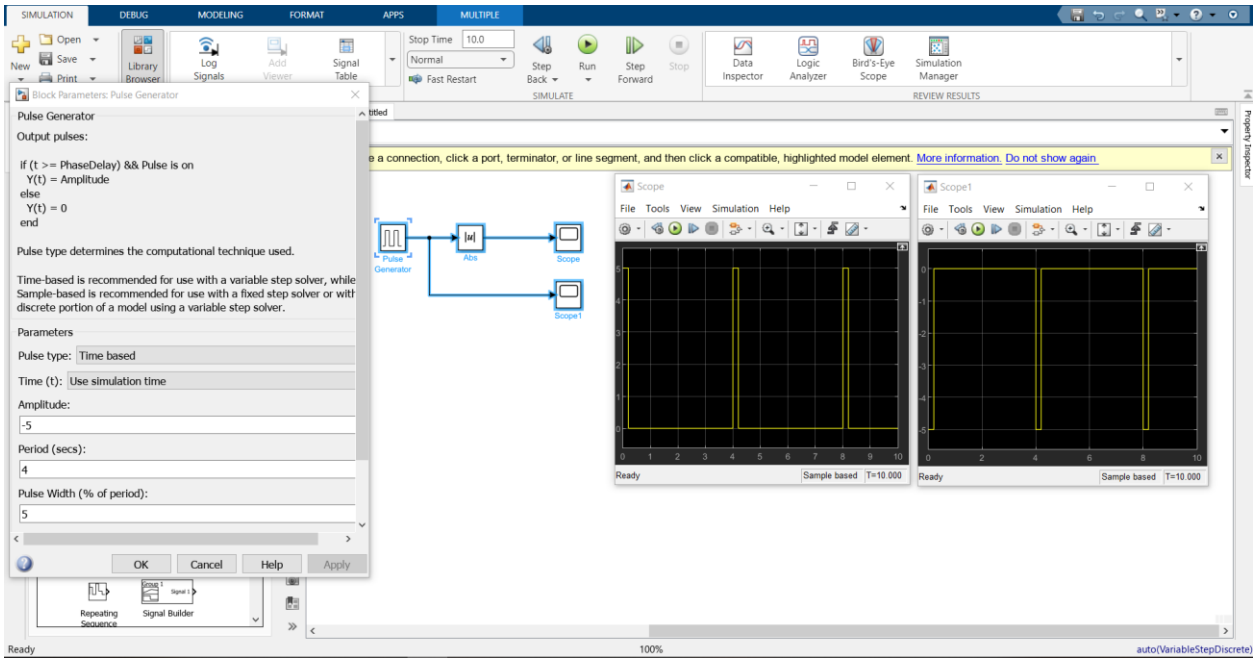


Рис. 3.7 Приклад застосування блоку Abs

На рис. 3.8 наведений приклад застосування блоку Sum при додаванні двох сигналів (ступінчастого та синусоїдального).

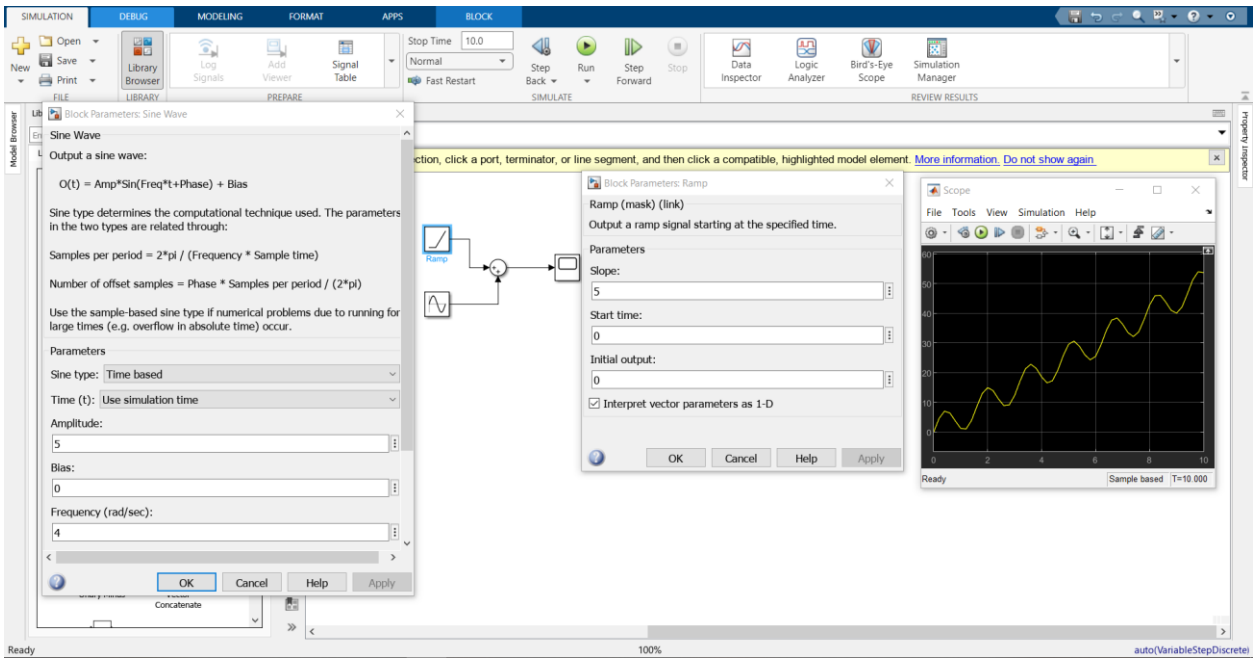


Рис. 3.8 Приклад застосування блоку Sum

На рис. 3.9 наведено приклади застосування блоку Trigonometric Function.

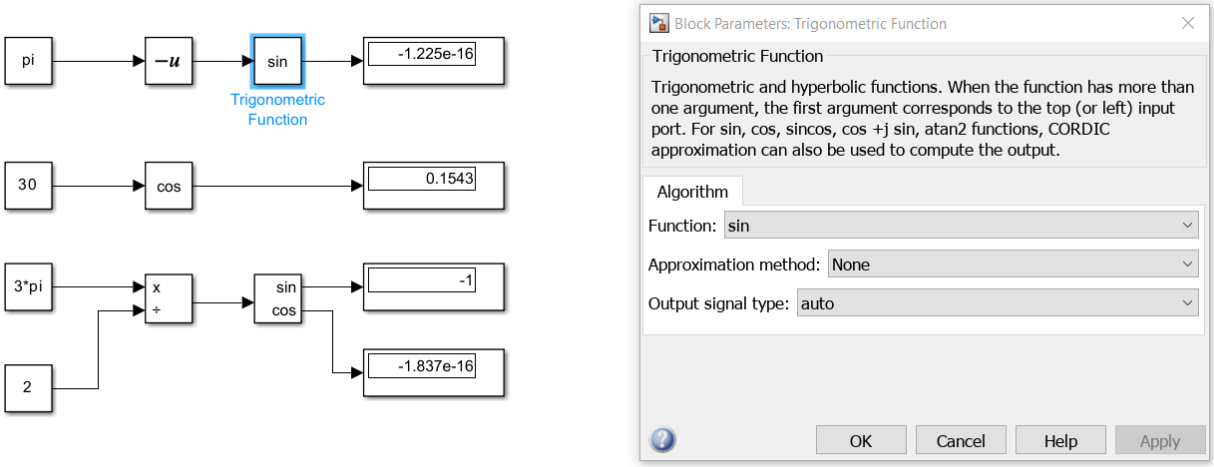


Рис. 3.9 Вікно з прикладом використання блоку Trigonometric Function

Блок Trigonometric Function виконує обчислення тригонометричної функції. До параметрів блоку відносяться налаштування [2]:

- **Function** – вид обчислюваної функції (вибирається зі списку): **sin, cos, tan, asin, acos, atan, atan2, sinh, cosh і tanh.**
- **Output signal type** – тип вихідного сигналу (вибирається зі списку): **Auto** – автоматичне визначення типу, **Real** – дійсний сигнал; **Complex** – комплексний сигнал.

На рис. 3.10 наведено приклад застосування блоку Math Function, а саме обчислення експоненти та натурального логарифму

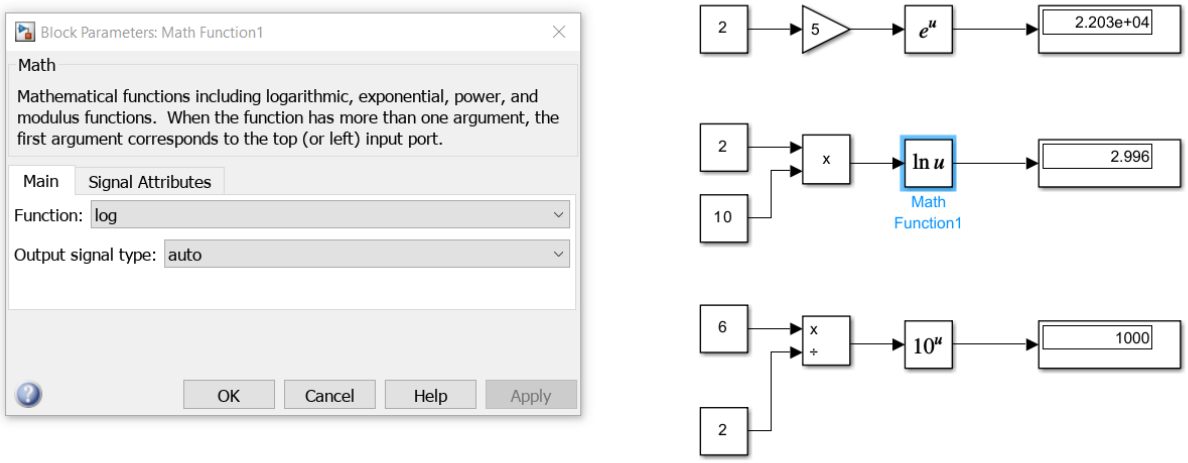


Рис. 3.10 Вікно з прикладом використання блоку Math Function

Блок Math Function має наступні параметри:

- **Function**. Вибір математичної функції, яку необхідно обчислити. Список доступних функцій може включати такі операції, як експонента (**exp**), натуральний логарифм (**log**), синус (**sin**), косинус (**cos**), тангенс (**tan**), квадратний корінь (**sqrt**) та багато інших. Можна вибрати функцію зі списку або ввести власну математичну формулу.
- **Sample time**. Інтервал часу між відліками вхідного сигналу блоку Math Function. Встановлення значення **Sample time** визначає частоту оновлення результату обчислення функції. Можна встановити специфічний інтервал часу або використовувати налаштування, що успадковується від вхідного сигналу.
- **Inputs**. Кількість та тип вхідних сигналів, які передаються до блоку Math Function. Можна визначити кілька вхідних сигналів, які будуть використовуватися як аргументи для обчислення обраної функції.
- **Outputs**. Кількість та тип вихідних сигналів, які представляють результат обчислення математичної функції. Можна вказати кілька вихідних сигналів, які будуть містити обчислені значення.

Додатковою можливістю блоку Math Function в SimuLink є можливість налаштування вхідних та вихідних розмірів сигналів, обмеження значень вхідних сигналів або використання векторів та матриць для багатовимірних обчислень.

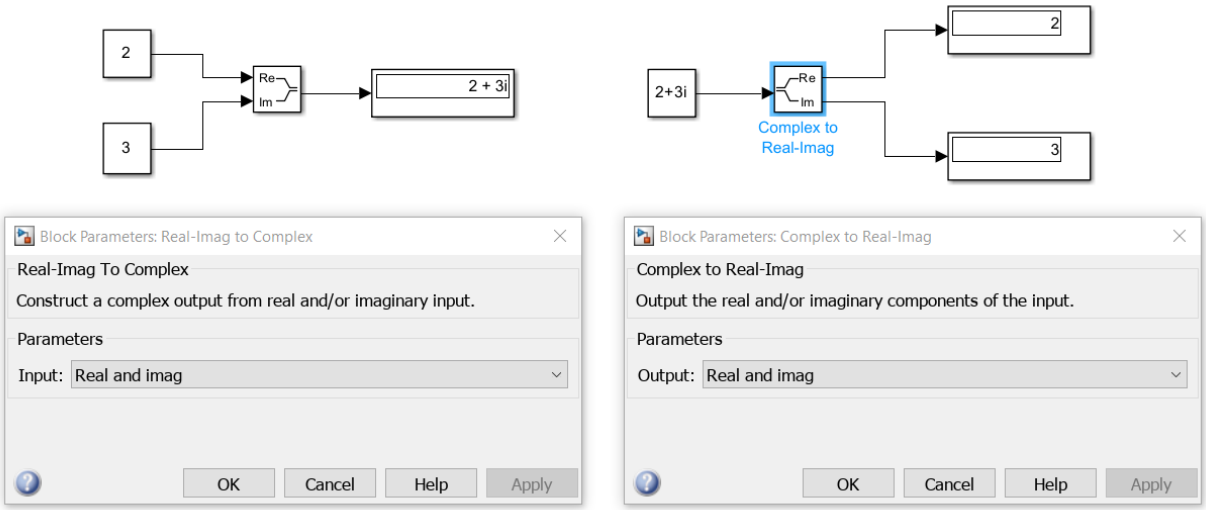
Блок «Complex to Real-Imag» в SimuLink виконує перетворення комплексного числа на його дійсну та уявну частини. До параметрів даного блоку відносяться:

- **Complex Input**. Вхідний комплексний сигнал, який представляється у форматі комплексного числа. Цей сигнал містить дійсну та уявну частини комплексного числа.
- **Real Output**. Вихідний сигнал, що представляє дійсну частину комплексного числа.
- **Imaginary Output**. Вихідний сигнал, що представляє уявну частину комплексного числа.

Блок «Real-Imag to Complex» виконує обернене перетворення, тобто об'єднання дійсної та уявної частин комплексного числа в одне комплексне число. До параметрів даного блоку відносяться:

- **Real Input**. Вхідний сигнал, що представляє дійсну частину комплексного числа.
- **Imaginary Input**. Вхідний сигнал, що представляє уявну частину комплексного числа.
- **Complex Output**. Вихідний комплексний сигнал, який представляється у форматі комплексного числа. Сигнал містить дійсну та уявну частини, які об'єднуються у комплексне число.

На рис.3.11 наведені приклади використання блоків Real-Imag to Complex та Complex to Real-Imag.



а) б)

Рис. 3.11 Вікно з прикладами використання блоків:

а) Real-Imag to Complex; б) Complex to Real-Imag

На рис.3.12 зображений приклад, що показує спосіб застосування блоку Matrix Concantenate:

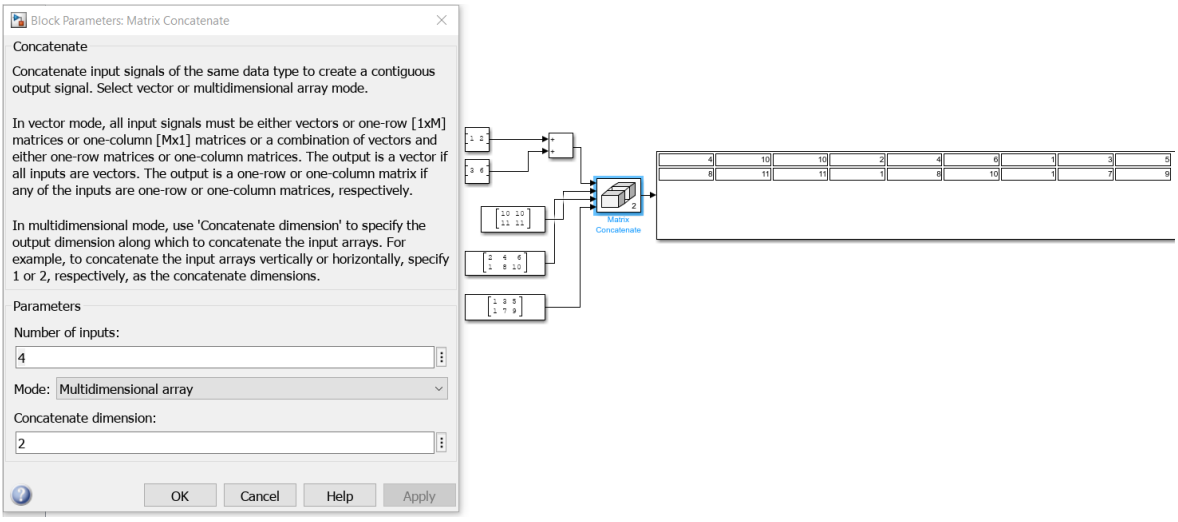


Рис. 3.12 Вікно з прикладами використання блоку Matrix Concatenate

На рис. 3.13 висвітлено приклад застосування блоку Vector Concatenate:

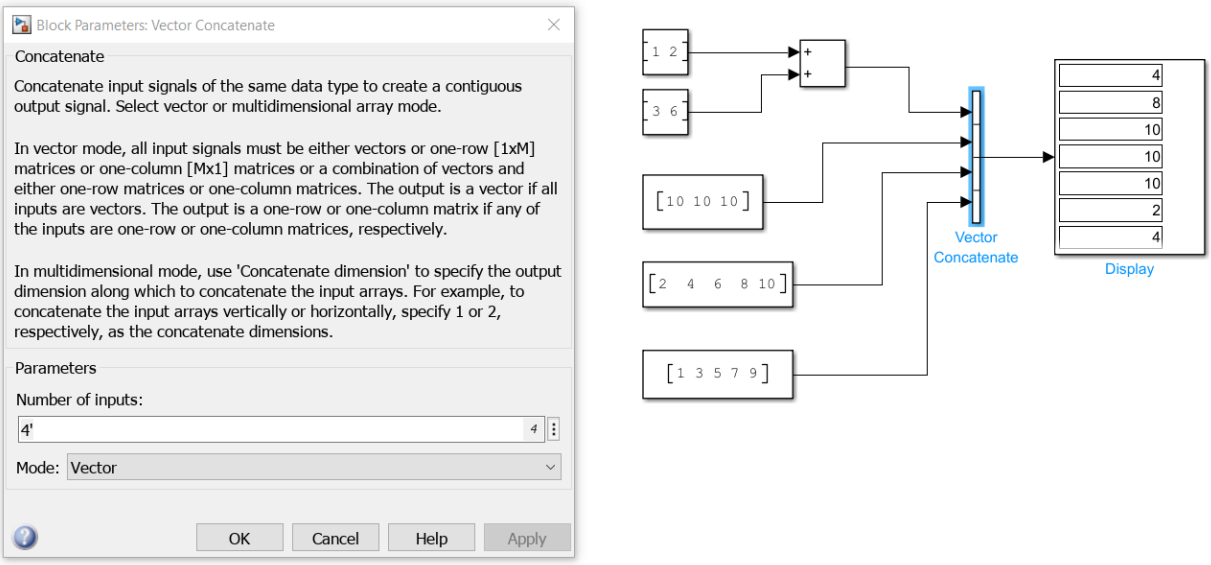


Рис. 3.13 Вікно з прикладами використання блоку Vector Concatenate

3.2. Завдання на виконання комп'ютерного практикуму

Завдання 1. Створити S-моделі за схемою (рис.3.14) відповідно до варіанту згідно табл.3.1, налаштувати параметрів блоків відповідно до варіанту (табл. 2.2 з комп'ютерного практикуму №2) та вивести сигнали з виходів моделей за допомогою блоку розділу Sinks: Scope.

Рис. 3.14 Схема моделі

Таблиця 3.1

Варіанти індивідуальних завдань

Варіант	Джерело 1	Джерело 2	Математична дія
1.	Sink	Pulse	Ділення

2.	Step	Ramp	Додавання
3.	Waveform	Sine Wave	Множення
4.	Pulse	Step	Віднімання
5.	Sine Wave	Sink	Додавання
6.	Pulse	Ramp	Множення
7.	Waveform	Sine Wave	Віднімання
8.	Step	Pulse	Ділення
9.	Ramp	Waveform	Множення
10.	Sine Wave	Step	Ділення

Завдання 2. Створити S-модель за схемою (рис. 3.15) відповідно до варіанту згідно табл.3.2 налаштувати параметр блоку Constant та MinMax, вивести значення з виходу моделі за допомогою блоку розділу Sinks: Display.

Рис. 3.15 Схема моделі

Таблиця 3.2

Варіанти індивідуальних завдань

Варіант	Масив значень	Значення з масиву
1	[1 2 3 4 5 6 7 8 9 10]	Мінімальне
2	[2 9 7 1 6 10 3 4 5 6]	Максимальне
3	[12 13 5 6 7 3 2 1 19]	Мінімальне
4	[2 4 5 6 7 10 3 1 8 5]	Максимальне
5	[14 15 16 3 7 9 5 11 3]	Мінімальне
6	[9 1 5 7 2 4 13 16 9 11]	Максимальне

Варіант	Масив значень	Значення з масиву
7	[8 4 3 1 10 17 18 9 16]	Мінімальне
8	[1 2 9 3 4 5 16 11 20 6]	Мінімальне
9	[6 5 2 3 31 7 4 9 5 3 2]	Максимальне
10	[10 20 21 22 6 4 7 3 9 1]	Мінімальне

Завдання 3. Створити S-модель за заданою функцією, використовуючи набір блоків з розділу Math Operation, щоб на виході в блоці Display отримати задану функцію відповідно до варіанту згідно табл.3.3. Звіритись та замінивши блок на Scope, отримати графік функції f(x).

Таблиця 3.1

Варіанти індивідуальних завдань

Варіант	Усталене значення блоку Constant	Функція	Аргумент x
1	0,1	$y = 2 * \sin(e^{2x} - \ln(2))$	x = ctgt+cost
2	0,2	$y = \sqrt{10 * \frac{x^5 + x^9}{\cos(2x)}}$	x=cost
3	0,3	$y = 5 * \sqrt{2 * \frac{2x^2 + 9x^9}{\sin(2x) + 1}}$	x=t
4	0,4	$y = 10 * \cos(e^{5x} + \log(2) + 2)$	x=sin2t
5	0,5	$y = 3 * \frac{x + x^2 + (\sin(x) + \cos(2x))}{2x + 10}$	x=arctgt
6	0,6	$y = \ln\lvert x\rvert + \sin(\frac{2x}{10} - e^{x^2})$	x=t-1
7	0,7	$y = \text{arctg}\left(e^{2\sqrt{x^2}}\right) - \sin 2x$	x=t+1
8	0,8	$y = \text{tg}\left(10e^{2+x-\sin x}\right) + 99$	x=t

Варіант	**Усталене значення блоку Constant**	**Функція**	**Аргумент x**
9	0,9	$y = \sqrt{ctg(\frac{5x^2}{\lvert x\rvert} + e^{\ln 10})}$	x=cost-sint
10	0,11	$y = 10\sin(e^{2*x^3}) - tg(\sqrt{\frac{\lvert x\rvert}{2}})$	x=sint+cost

3.3. Контрольні запитання

1. Які основні елементи складають S-модель?
2. Які операції можна виконувати з вхідними сигналами в S-моделі?
3. Які математичні операції можна використовувати для обробки сигналів у S-моделі?
4. За допомогою яких блоків можна додавати сигнали у S-моделі?
5. За допомогою яких блоків можна перемножити сигнали у S-моделі?
6. Які функції можна використовувати для обробки сигналів у розділі Math Operation?
7. Як можна виконувати математичні операції на спектрах сигналів у S-моделі?

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 4
СТВОРЕННЯ S-МОДЕЛЕЙ, ВИКОРИСТОВУЮЧИ РОЗДІЛ CONTINUOUS

Мета роботи

Ознайомитись з розділом Continuous бібліотеки SimuLink та поглибити розуміння S-моделей та їх можливостей, а також визначити їх потенційне значення для рішення практичних завдань з використанням розділу Continuous.

4.1. Теоретичні відомості

Створення S-моделей з використанням блоків розділу Continuous може бути використано для аналізу даних у багатьох галузях, таких як економіка, наука про матеріали, біологія, медицина та інші. Цей підхід може допомогти знайти корисну інформацію у неперервних даних та зробити правильні рішення на основі цієї інформації.

До розділу Continuous входять 16 блоків, які охоплюють різні аспекти аналізу неперервних змінних. На рис. 4.1 зображено вікно розділу бібліотеки Continuous з блоками, що входять до нього.

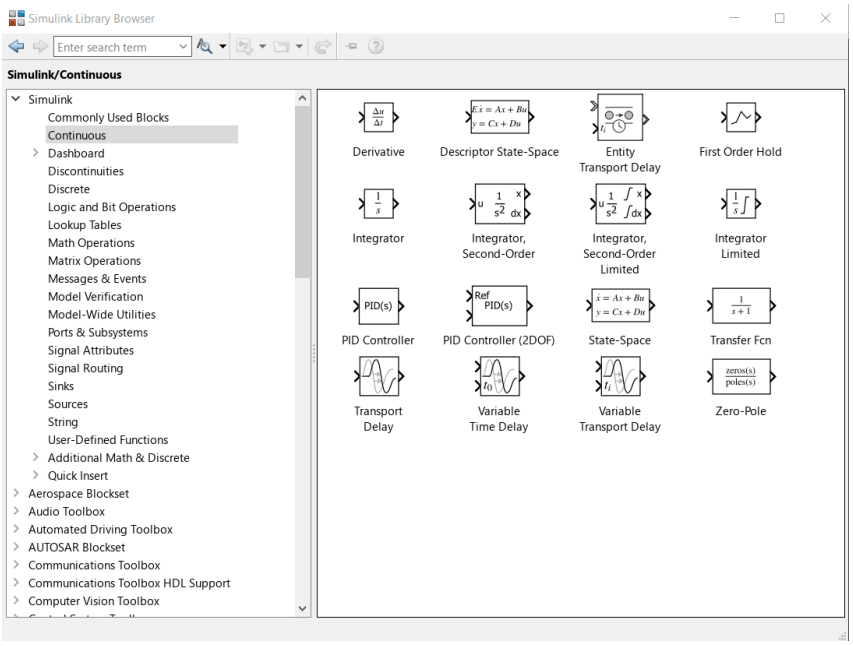


Рис. 4.1 Вікно розділу Continuous

Назви блоків, що входять до розділу Continuous [12]:

Derivative - апроксимує похідну вхідного сигналу u по відношенню до часу моделювання t. Цей блок отримує один вхід і генерує один вихід.

Descriptor State-Space - дозволяє моделювати лінійні неявні системи, які можна виразити у вигляді $E\dot{x} = Ax + Bu$, де E - матриця мас системи.

Entity Transport Delay - затримує об'єкт на певний проміжок часу, який називається транспортною затримкою. Першим вхідним параметром є об'єкт, який переміщується з точки A в точку B по рухомій поверхні постійної довжини, швидкість якої змінюється з часом.

First Order Hold - генерує неперервну кусково-лінійну апроксимацію вхідного сигналу. Використання блоку First Order Hold найкраще підходить для перетворення дискретного дискретного сигналу в неперервний сигнал, не викликаючи скидання розв'язувача.

Integrator - інтегрує вхідний сигнал у часі і видає результат у вигляді вихідного сигналу.

Integrator Limited - ідентичний до блоку **Integrator**, за винятком того, що вихід блоку обмежується на основі верхньої та нижньої меж насичення.

PID Controller - реалізує ПІД -регулятор (ПІД, ПІ, РД, тільки П або тільки І). Вихід блоку є зваженою сумою вхідного сигналу, інтеграла вхідного сигналу та похідної вхідного сигналу. Вагами є параметри пропорційного, інтегрального та похідного коефіцієнтів підсилення. Полюс першого порядку фільтрує дію похідної.

PID Controller (2DOF) - генерує вихідний сигнал на основі різниці між еталонним сигналом і виміряним виходом системи. Блок обчислює зважений сигнал різниці для пропорційної та похідної дії відповідно до заданих вагових коефіцієнтів (b і c). Вихід блоку є сумою пропорційної, інтегральної та похідної дії на відповідні сигнали різниці значень, де кожна дія зважується відповідно до параметрів підсилення П, І та Д. Полюсник першого порядку фільтрує похідну дію.

Second-Order Integrator - розв'язують задачу початкових значень другого порядку $\frac{d^2x}{dt^2} = u$, де u - вхід системи.

Second-Order Integrator Limited – ідентичний до блоку **Second-Order Inte**, за винятком того, що за замовчуванням він обмежує стани на основі вказаних верхньої та нижньої меж.

State-Space - реалізує систему, поведінку якої визначається як:

$$\begin{aligned} \dot{x} &= A\,x + B\,u \\ y &= C\,x + D\,u \\ x|_{t=t_0} &= x_0, \end{aligned}$$

де x — вектор стану, u — вхідний вектор, y — вихідний вектор, а x_0 — початкова умова вектора стану. Матриці A, B, C і D можна вказати як розріджені або щільні матриці.

Transfer Fcn - моделює лінійну систему за допомогою передатної функції лапласівської змінної s. Блок може моделювати системи з одним входом і одним виходом (SISO) та системи з одним входом і декількома виходами (SIMO).

Передатна функція має вигляд: $H(s) = \frac{y(s)}{u(s)} = \frac{num(s)}{den(s)}$, де u та y - вхід та вихід системи відповідно, num(s) та den(s) містять коефіцієнти чисельника та знаменника в порядку спадання s.

Transport Delay - затримує вхідний сигнал на певний проміжок часу. Можна використовувати цей блок для імітації часової затримки. На вхід цього блоку подається неперервний сигнал. На початку симуляції блок виводить параметр Початковий вихід до тих пір, поки час симуляції не перевищить параметр Час затримки. Після цього блок починає генерувати вхідний сигнал із затримкою. Під час симуляції блок зберігає точки входу та час симуляції у буфері. Розмір буфера задається параметром Початковий розмір буфера.

Variable Time Delay - дозволяє задавати керовану ззовні величину затримки.

Variable Transport Delay - у цьому режимі вихід блоку на поточному часовому кроці дорівнює значенню його входу даних (верхнього або лівого) на попередньому часовому кроці, що дорівнює поточному часу мінус транспортна затримка.

Zero-Pole - моделює систему, яка визначається за допомогою нулів, полюсів і коефіцієнта підсилення передатної функції Лапласа. Цей блок може моделювати системи з одним входом і одним виходом (SISO) та з одним входом і декількома виходами (SIMO).

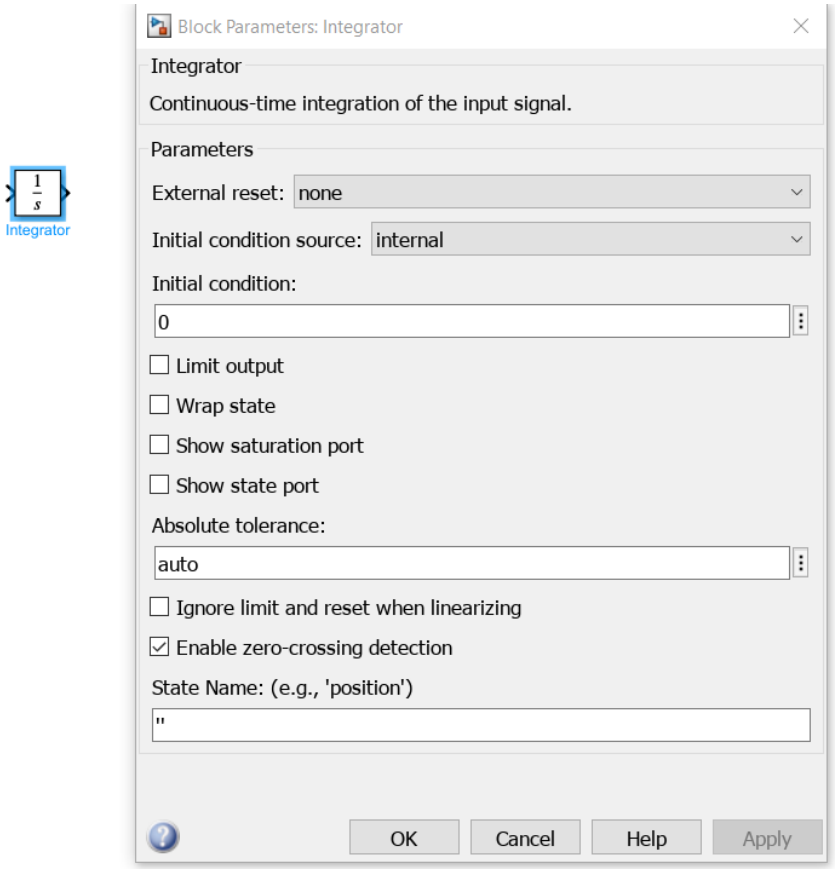


Рис. 4.2 Вікно налаштувань параметрів блоку **Integrator**

На рис. 4.2 зображений блок **Integrator** розділу Continuous та блок параметрів. Цей блок зазвичай використовується для моделювання фізичних систем, систем керування, систем зворотного зв'язку та інших динамічних

систем. Він дозволяє враховувати накопичення змінних в часі та визначати їх поведінку з урахуванням інтеграції.

Параметри блоку **Integrator** включають [2]:

- *External reset* (зовнішнє скидання/обнулення) – тип зовнішнього керуючого сигналу, що обирається зі списку: none – ні, rising – наростаючий, falling – спадаючий, either – будь-який.
- *Initial condition Source* – джерело початкового значення вихідного сигналу при інтегруванні. У списку можна вибрати внутрішнє (internal) або зовнішнє (external) джерело.
- *Initial condition* (початкові умови) – установка початкового значення вихідного сигналу при інтегруванні (у вигляді числа, за умовчанням 0).
- *Limit output* – включення / відключення обмеження вихідного сигналу.
- *Wrap state* – включення / відключення переносу циклічного стану.
- *Show saturation port* – керує відображенням порту, що виводить рівні обмеження вихідного сигналу.
- *Show state port* — керує відображенням порту стану системи.
- *Absolute tolerance* – абсолютна похибка (за замовчуванням автоматичний вибір – auto).
- *Enable zero crossing detections* – включення перевірки наявності переходів через нуль.

На рис 4.3 представлений спосіб використання блоку **Integrator** при вхідному імпульсному прямокутному сигналі (Pulse Generator).

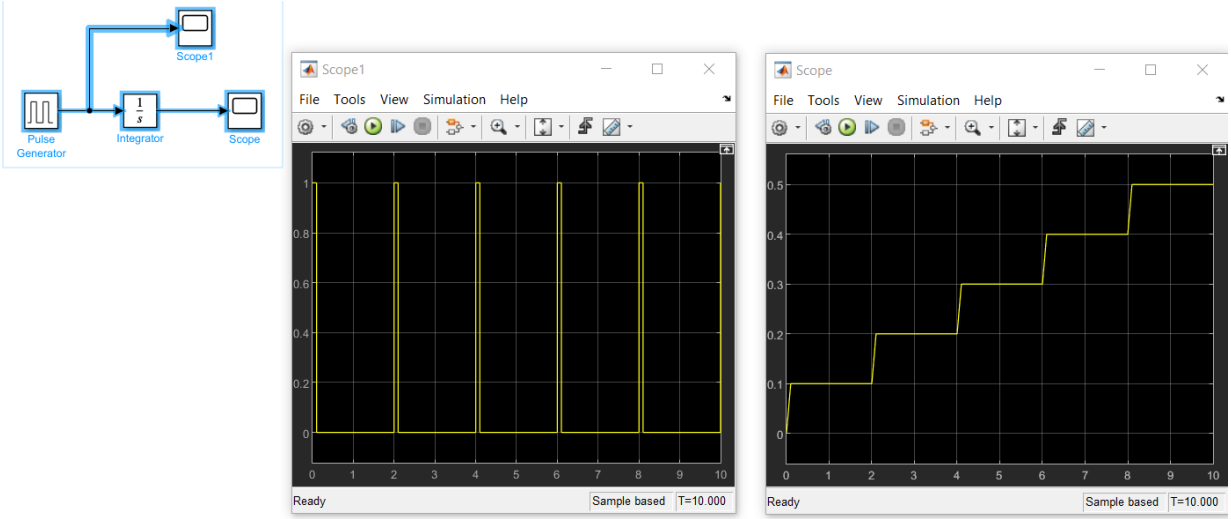


Рис. 4.3 Приклад застосування блоку **Integrator**

Блок **Derivative** в розділі Continuous використовується в системах моделювання для чисельного обчислення похідної функції від вхідного сигналу. На рис 4.4 зображений приклад використання блоку **Derivative** при вхідному синосуїдальному сигналі (Sine Wave)**.**

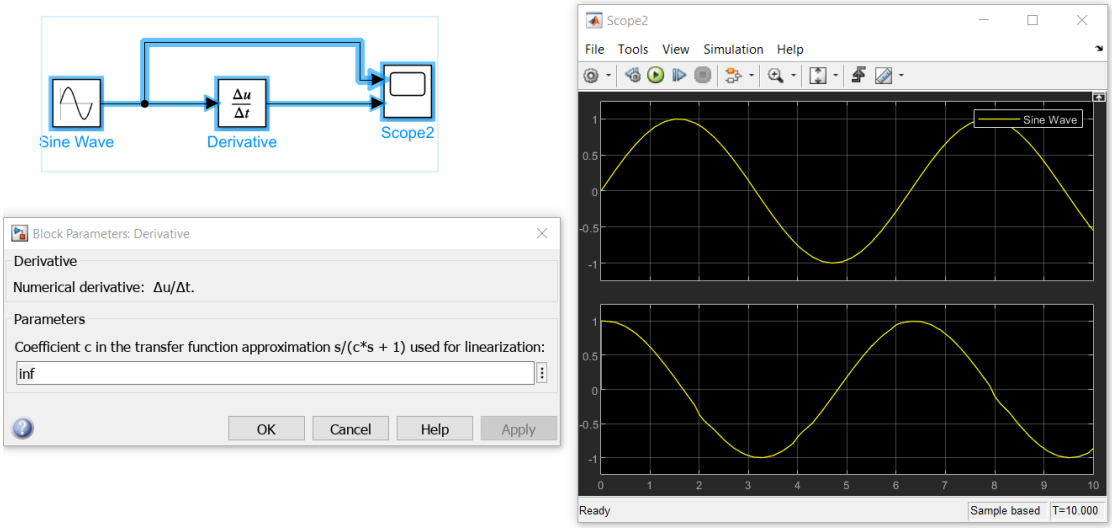


Рис. 4.4 Приклад застосування блоку **Derivative**

Блок **Transfer Fcn** використовується для моделювання систем на основі їх передатних функцій. На рис. 4.5 представлено вікно налаштувань параметрів блоку **Transfer Fcn.**

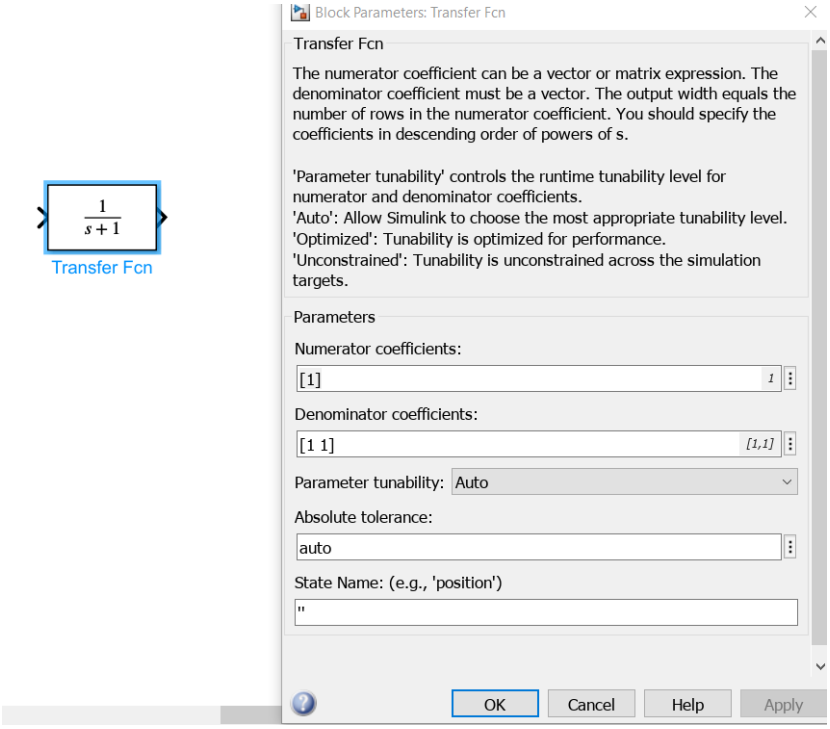


Рис. 4.5 Вікно налагодження параметрів блоку **Transfer Fcn**

Параметри блоку **Transfer Fcn** включають:

- *Numerator* (Чисельник): Введіть коефіцієнти чисельника у вигляді вектора або масиву коефіцієнтів.
- *Denominator* (Знаменник): Введіть коефіцієнти знаменника у вигляді вектора або масиву коефіцієнтів.
- *Sample Time* (Інтервал часу): Вказує інтервал часу між кроками обчислення вхідних та вихідних значень.
- *Input delay* (Затримка вхідного сигналу): Визначає затримку вхідного сигналу, яка враховується при моделюванні системи.
- *Output delay* (Затримка вихідного сигналу): Визначає затримку вихідного сигналу, яка враховується при моделюванні системи.

- *Input processing* (Обробка вхідного сигналу): Встановіть тип обробки вхідного сигналу, такий як «Element-wise» (елемент-за-елементом) або «Whole vector» (весь вектор).

На рис 4.6 зображений приклад використання блоку **Transfer Fcn** при вхідному ступінчастому сигналі (Step)**.**

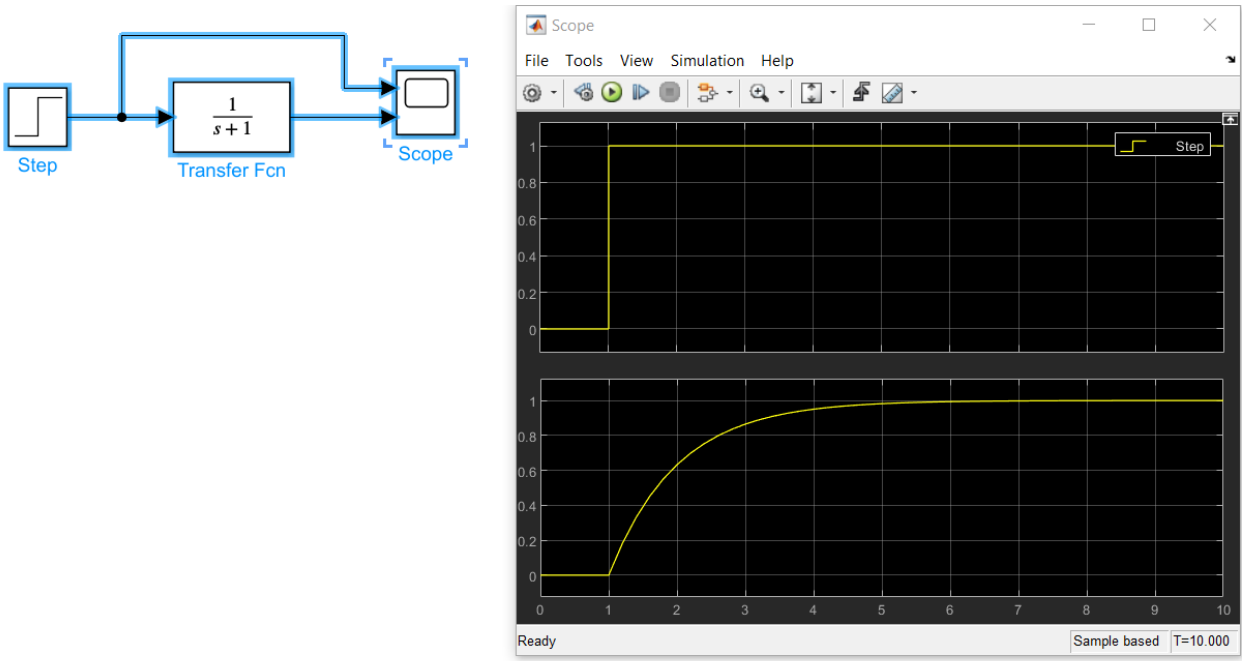


Рис. 4.6 Приклад застосування блоку **Transfer Fcn**

Блок **State-Space** використовується для моделювання лінійних станових просторових систем, які описуються векторним рівнянням стану. На рис. 4.7 представлено зображення блоку та вікно налаштування його параметрів**.**

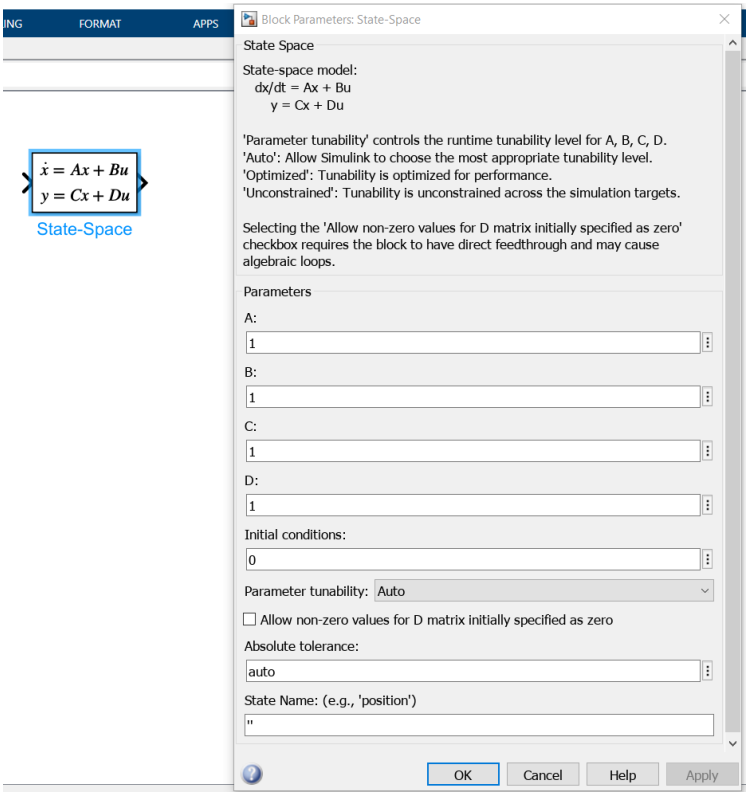


Рис. 4.7 Вікно Блоку **State-Space** та його параметрів блоку

Параметри блоку **State-Space** включають [13]:

- *A* – Коефіцієнт матриці A, яка представляє зв’язки між станами системи.
- *B* – Коефіцієнт матриці B, яка зображує зв’язки між вхідними сигналами та станами системи.
- *C* – Коефіцієнт матриці C, який висвітлює зв’язки між станами системи та вихідними сигналами.
- *D* – Коефіцієнт матриці D, що представляє зв’язки між вхідними сигналами та вихідними сигналами, якщо такі є.
- *Initial condition* – Задання вектору початкових умов.
- *Absolute tolerance* - Абсолютна похибка системи.

На рис 4.8 зображений приклад використання блоку **State-Space** при вхідному пилкоподібному сигналі (Repeating Sequence)**.**

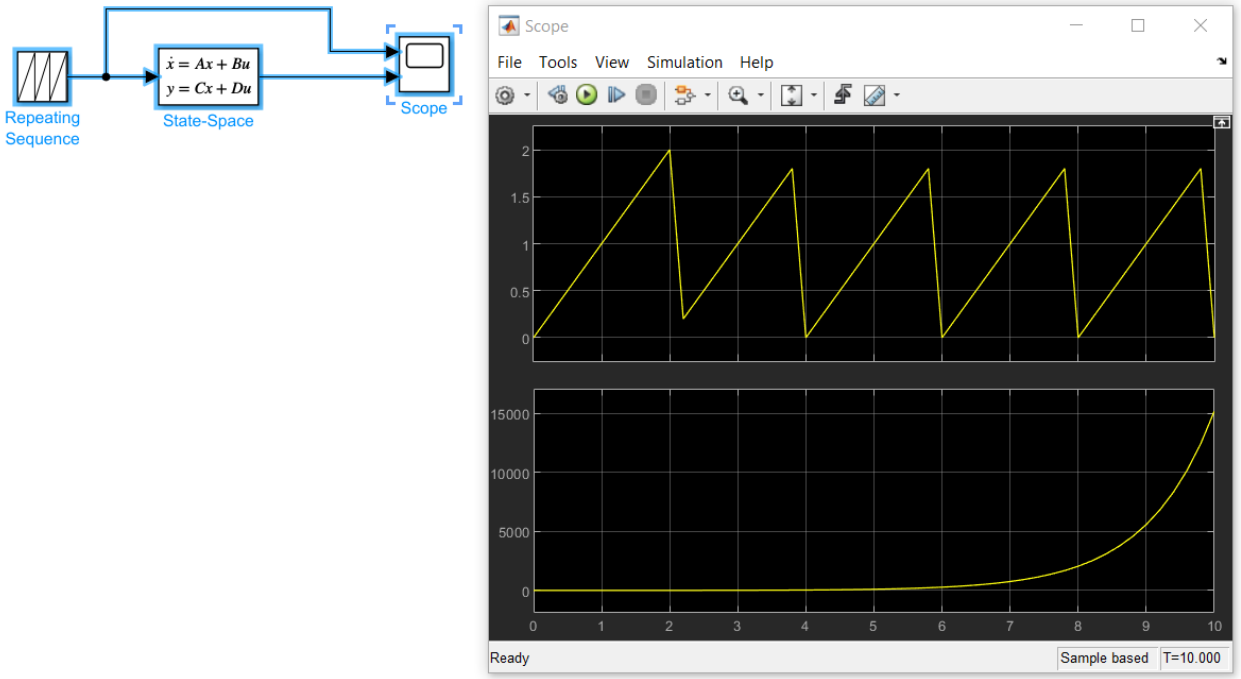


Рис. 4.8 Приклад застосування блоку **State-Space**

4.2. Завдання на виконання комп'ютерного практикуму

Завдання 1. Створити S-моделі відповідно до рис.4.9, застосувавши блоки з розділів Continuous та Sources (джерела). Налаштувати блоки згідно з варіантом відповідно до табл.4.1. Отримати сигнали на виходах за допомогою блоків розділу Sinks (приймачі).

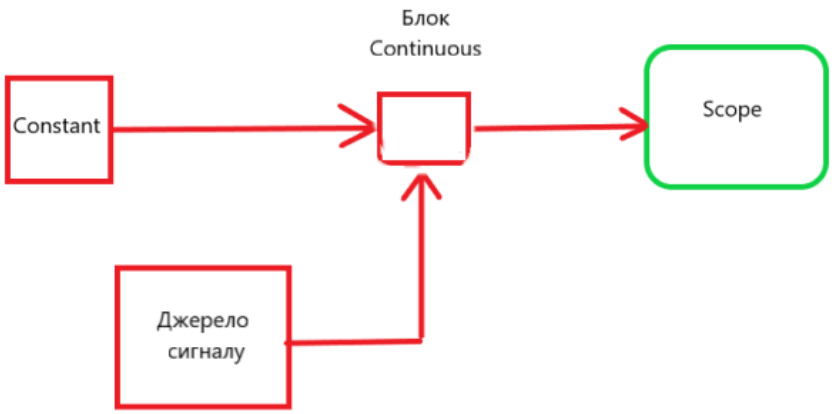


Рис. 4.9 Схема роподілу блоків

Таблиця 4.1

Варіанти індивідуальних завдань

Варіант	**Блоки розділу Sources**	**Налаштування блоку розділу Sources**			**Блоки розділу Continuous**
1.	Sine Wave	**Amplitude**	**Frequency**	**Phase**	
		0,4	0,4	20	Derivative
					Transfer Fcn
					State-Space
					Integrator
2.	Ramp	**Slope**	**Start time**	**Initial output**	
		0,5	0,4	20	Derivative
					Transfer Fcn
					State-Space
					Integrator
3.	Step	**Step time**	**Initial value**	**Final value**	
		1,5	0,4	20	Derivative
					Transfer Fcn
					State-Space
					Integrator
4.	Pulse Generator	**Amplitude**	**Period**	**Phase**	
		0,5	1,2	0	Derivative
					Transfer Fcn
					State-Space
					Integrator
5.	Ramp	**Slope**	**Start time**	**Initial output**	
		0,7	0,8	10	Derivative
					Transfer Fcn
					State-Space
					Integrator
6.	Sine Wave	**Amplitude**	**Frequency**	**Phase**	
		1,4	1,4	10	Derivative
					Transfer Fcn
					State-Space
					Integrator

Варіант	**Блоки розділу Sources**	**Налаштування блоку розділу Sources**			**Блоки розділу Continuous**
7.	Pulse Generator	**Amplitude**	**Period**	**Phase**	
		1,6	2,2	5	Derivative
					Transfer Fcn
					State-Space
					Integrator
8.	Step	**Step time**	**Initial value**	**Final value**	
		1,5	0,4	20	Derivative
					Transfer Fcn
					State-Space
					Integrator
9.	Sine Wave	**Amplitude**	**Frequency**	**Phase**	
		1,4	0,5	15	Derivative
					Transfer Fcn
					State-Space
					Integrator
10.	Step	**Step time**	**Initial value**	**Final value**	
		2,5	2,4	5	Derivative
					Transfer Fcn
					State-Space
					Integrator

Завдання 2. Змінити варіант та побудувати нові графіки (парні номери змінюють номери на непарні за принципом n-1, де n – номер варіанту; непарні номери змінюють номери на непарні за принципом n+1, де n – номер варіанту). Якщо студент має варіант 10, то він обирає 1 варіант та виконує вищезазначене розподілення завдання.

4.3. Контрольні запитання

1. Які основні компоненти складають S-модель?
2. Що таке Continuous (постійні) змінні в контексті S-моделей?

3. Які типи функцій можна використовувати для опису Continuous змінних в S-моделях?

4. Які основні оператори можна використовувати для обчислення Continuous змінних в S-моделях?

5. Які основні методи розв'язання диференціальних рівнянь використовуються в S-моделях?

6. Як впливає точність чисельного розв'язку диференціальних рівнянь на точність результатів S-моделей?

7. Які інструменти або програми можна використовувати для створення S-моделей з використанням Continuous розділу?

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 5
РОЗВ’ЯЗОК РІВНЯНЬ У СИМВОЛЬНОМУ ВИГЛЯДІ

Мета роботи

Ознайомлення та дослідження каталогу Symbolic Math Toolbox в процесі розв'язування рівнянь у символьному вигляді в системі MatLab, набуття практичних навичок у розв'язуванні рівнянь у символьному вигляді.

5.1. Теоретичні відомості

Розв'язок рівнянь є однією з ключових задач математики та прикладних наук, яка має велике значення у вирішенні різних завдань та проблем. Рівняння зустрічаються в різних галузях науки, включаючи фізику, інженерію, економіку та багато інших. У символьному розв'язку рівнянь відбувається ознаймолення зі змінними та символьними виразами, що надає змогу отримувати аналітичні розв'язки та проводити різні маніпуляції з ними.

Рівняння є основним математичним інструментом для вираження залежностей та знаходження невідомих значень. У загальному сенсі, рівняння - це висловлювання, в якому стверджується, що два вирази рівні між собою.

Змінні: Змінні - це символи або літери, які представляють невідомі значення. Зазвичай використовуються літери x, y, або z для позначення змінних.

- Степені: У рівняннях можуть зустрічатися степені, які вказують на піднесення числа або змінної до певної степені. Наприклад, x^2 означає x піднесене до квадрату.
- Коефіцієнти: Коефіцієнти - це числа, які множаться на змінні або степені. Наприклад, у рівнянні 2x + 3 = 0, 2 є коефіцієнтом перед x.

Оголосити символьну змінну в MatLab можна таким способом [14]:

- За допомогою команди **syms**:

Так можна оголосити відразу кілька символьних змінних синтаксис команди **syms**:

syms name1 [name2 ...] [options],

де *name1, [name2 ...]* - імена створюваних змінних. Можна контрулювати інтерпретацію змінних за допомогою функції *real*. При використанні функції *options* - для речових змінних, *unreal* - для комплексних.

Імена повинні починатися з літери і містити тільки букви та цифри. Використовуючи звичайні арифметичні операції та функції, можна створювати нові символьні вирази і функції.

Отже, таким чином оголошуємо змінні для функції f:

```
>> syms x y % створення символьного функції f
f = (x ^ 2 + 2 * y) / (2 * sin (x) -cos (2 * y))
```

В командному вікні спостерігаємо відображений символьний вираз функції f.

```
f =

-(x^2 + 2*y)/(cos(2*y) - 2*sin(x))
```

Для зручного відображення символьних виразів у більш зрозумілому та гарно оформленому вигляді. Функція **pretty** дозволяє відформатувати вирази, щоб зробити їх більш читабельними.

```
>> pretty(f)
         2
        x  + 2 y
- ------------------
  cos(2 y) - 2 sin(x)
```

Функція допомогає оформити вираз, розташовуючи степень та оператори у більш зрозумілому порядку.

При символьному розв'язанні рівнянь важливо враховувати арифметичні дії, які можна виконувати з символьними виразами. Додавання і віднімання, множення і ділення, піднесення до степеня, факторизація, заміна змінних та ін.

```
>> syms x y
f= (x^2+y^2)*x+y-x*y
```

В командному вікні відображається результат:

```
f =

y - x*y + x*(x^2 + y^2)
```

Symbolic Math Toolbox надає низку функцій для виконання алгебраїчних операцій над символьними змінними. Декілька основних функцій для цих операцій [6]:

- **simplify**. Ця функція спрощує символьний вираз шляхом застосування алгебраїчних та тригонометричних ідентичностей, заміни констант та інших спрощень. Приклад використання функції наведено на рис.5.1.

```
1  clc,clear all
2  syms x
3  expr = sin(x)^2+cos(x)^2;% Вираз, який потрібно спростити
4  disp('Спрощений вираз sin(x)^2+cos(x)^2');
5  simplified_expr=simplify(expr); % Спрощений вираз
6  disp(simplified_expr);
7
```

Command Window

```
Спрощений вираз sin(x)^2+cos(x)^2
1
```

Рис. 5.1 Приклад використання функції **simplify**

- **expand**. Функція розгортає вираз, розкриваючи скобки та виконуючи множення для отримання розкладу на елементарні частини. Приклад використання функції **expand** наведено на рис.5.2.

```
1  clc, clear
2  syms x
3  expr = (x + 1)^3; % Вираз, який потрібно розгорнути
4  expanded_expr = expand(expr); % Розгорнутий вираз
5  disp(expanded_expr);
```

Command Window

```
x^3 + 3*x^2 + 3*x + 1

fx >>
```

Рис. 5.2 Приклад використання функції **expand**

- **factor**. Ця функція факторизує символьний вираз на множники. Приклад використання функції **factor** наведено на рис.5.3.

```
clc, clear
syms x
expr = x^3 - 8; % Вираз, який потрібно факторизувати
factored_expr = factor(expr); % Факторизований вираз
disp(factored_expr);
```

```
Command Window
[x - 2, x^2 + 2*x + 4]

fx >>
```

Рис. 5.3 Приклад використання функції **factor**

- **collect**: Функція збирає однакові терміни разом у виразі, групуючи їх за певними змінними або степенями змінних. Приклад використання функції **collect** наведено на рис.5.4.

```
clc, clear
syms x y
expr = x^2*y + x*y + x + y; % Вираз, в якому потрібно зібрати однакові терміни
collected_expr = collect(expr, x); % Вираз зі зібраними однаковими термінами за змінною x
disp(collected_expr);
```

```
Command Window
y*x^2 + (y + 1)*x + y

fx >>
```

Рис. 5.4 Приклад використання функції **collect**

- **subs**: Функція замінює символьні змінні у виразі на задані значення або інші символьні вирази. Приклад використання функції **subs** наведено на рис.5.5.

```
clc, clear
syms x
expr = x^2 + x + 1; % Вираз, у якому потрібно замінити змінну
substituted_expr = subs(expr, x, 2); % Вираз зі заміненою змінною
disp(substituted_expr);
```

```
Command Window
   7

fx >>
```

Рис. 5.5 Приклад використання функції **subs**

- **solve**. Ця функція розв'язує символьні рівняння та системи рівнянь за заданими змінними. Приклад використання функції **solve** наведено на рис.5.6.

```
clc, clear
syms x
eqn = x^2 - 4; % Рівняння для розв'язання
sol = solve(eqn, x); % Розв'язок рівняння
disp(sol);
```

```
Command Window
  -2
   2

fx >>
```

Рис. 5.6 Приклад використання функції **solve**

Наприклад, знайти **x, y, z** з системи рівнянь (рис.5.7):

$$\begin{cases} 2x + 3y - z = 4 \\ x - y + 2z = 1 \\ 3x + 2y - 4z = -3 \end{cases}$$

```
clc, clear
syms x y z
eqn1 = 2*x + 3*y - z == 4;
eqn2 = x - y + 2*z == 1;
eqn3 = 3*x + 2*y - 4*z == -3;
sol = solve([eqn1, eqn2, eqn3], [x, y, z]);
disp(sol);
```

```
Command Window
    x: -1/5
    y: 2
    z: 8/5

fx >>
```

Рис. 5.7 Приклад використання функції **solve** при знаходженні **x, y, z** з системи рівнянь

Для визначення рівняння використовується подвійний знак рівності «==». Якщо не вказувати праву частину рівняння, то функція розв'язання припускає, що права частина рівняння дорівнює нулю [6].

- **diff**. Функція обчислює похідну символьного виразу за заданою змінною. Приклад використання функції **diff** наведено на рис.5.8.

```
clc, clear
syms x
expr = x^3 + 2*x^2 + x; % Вираз, для якого обчислюється похідна
derivative = diff(expr, x); % Похідна виразу за змінною x
disp(derivative);
```

```
Command Window
3*x^2 + 4*x + 1

fx >>
```

Рис. 5.8 Приклад використання функції **diff**

Функцію **diff** можна також використовувати для обчислення вищих похідних, вказавши порядок похідної як другий аргумент. Наприклад, для обчислення другої похідної виразу $x^3 + 4x^2 - 5x$ по змінній x можна скористатися кодом, який наведено на рис.5.9.

```
1  clc, clear
2  syms x
3  expression = x^3 + 4*x^2 - 5*x;
4  second_derivative = diff(expression, x, 2);
5  disp(second_derivative);
```

```
Command Window
   6*x + 8

fx >>
```

Рис. 5.9 Приклад використання функції **diff** для обчислення другої похідної

- **int.** Функція обчислює символьний інтеграл виразу за заданою змінною. Приклад використання функції **int** наведено на рис.5.10.

```
1  clc, clear
2  syms x
3  expr = x^2 + x + 1; % Вираз, для якого обчислюється інтеграл
4  integral = int(expr, x); % Інтеграл виразу за змінною x
5  disp(integral);
```

```
Command Window
   (x*(2*x^2 + 3*x + 6))/6

fx >>
```

Рис. 5.10 Приклад використання функції **int**

Також функція може бути використана для обчислення визначених інтегралів з тригонометричними функціями на заданому інтервалі (рис.5.11). Для цього потрібно передати нижню і верхню межі інтегрування як додаткові аргументи до функції int.

Приклад. Знаходження визначеного інтегралу $\int 2x \sin(x+3)$ на проміжку $[0; \frac{\pi}{2}]$:

```
clc, clear
syms x
expression = 2*x*sin(x+3);
integral = int(expression, x, 0, pi/2);
disp(integral);
```

```
Command Window
2*cos(3) - 2*sin(3) + pi*sin(3)

fx >>
```

Рис. 5.11 Приклад використання функції **int** для обчислення визначеного інтегралу на заданому інтервалі

- **limit**. Ця функція обчислює границю символьного виразу, коли змінна наближається до певного значення. Приклад використання функції **limit** наведено на рис.5.12.

```
clc, clear
syms x
expr = sin(x)/x; % Вираз, для якого обчислюється границя
limit_val = limit(expr, x, 0); % Границя виразу, коли x наближається до 0
disp(limit_val);
```

```
Command Window
1

fx >>
```

Рис. 5.12 Приклад використання функції **limit**

Функція **ezplot** в Symbolic Math Toolbox використовується для графічного відображення символьних функцій у 2D-просторі. Вона дозволяє швидко і просто побудувати графіки символьних виразів без необхідності вручну задавати значення змінних або створювати масиви точок.

Синтаксис використання функції **ezplot** виглядає наступним чином:

ezplot(expression),

де expression - символьний вираз графік якого потрібно побудувати.

Приклад використання функції **ezplot** для побудови графіку символьної функції наведено на рис.5.13 за допомогою наступного коду.

```
clc, clear
syms x
expression = sin(x);
ezplot(expression);
grid on
```

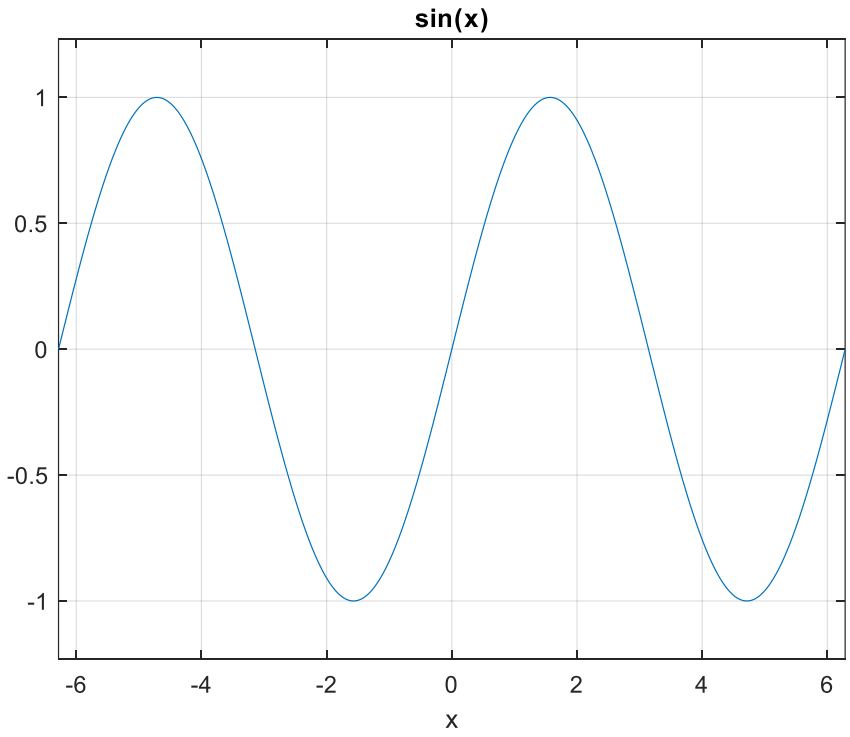


Рис. 5.13 Графік функції sin(x) при використанні функції **ezplot**

Функція **ezplot** також підтримує побудову графіків багатьох символьних функцій одночасно. Наприклад, можна побудувати графіки синуса та косинуса (рис.5.14) за допомогою наступного коду:

```
clc, clear
syms x
expression1 = sin(x);
expression2 = cos(x);
ezplot(expression1);
hold on;
ezplot(expression2);
hold off;
```

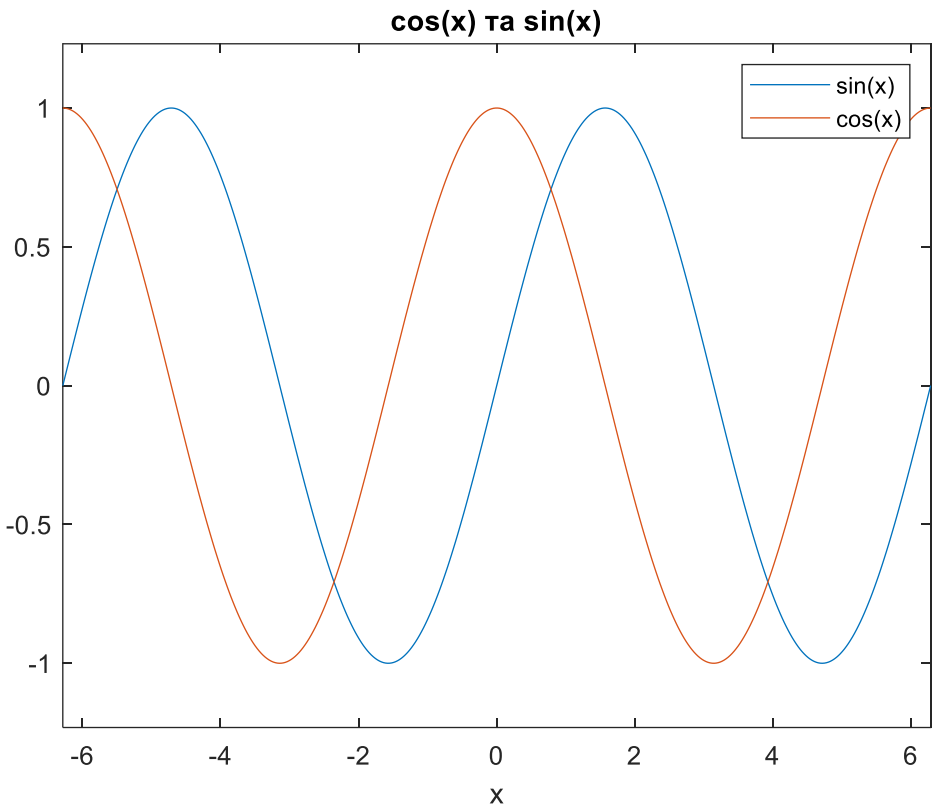


Рис. 5.14 Графіки синуса та косинуса за допомогою використання функцій **ezplot**

5.2. Завдання до виконання комп'ютерного практикуму

Завдання 1. За допомогою функції **subs** замініти символьні змінні у виразі (табл.5.1) на задані значення або інші символьні вирази.

Таблиця 5.1

Варіанти індивідуальних завдань

Варіант	Вираз	Аргумент заміни
1	$\frac{x^2}{1+0.25\sqrt{x}}$	2x
2	$\frac{x^4}{1+0{,}5\sqrt[3]{x}}$	x-1
3	$e^{-x^2}(1+3x-x^2)$	x+1
4	$\frac{e^{\frac{x}{3}}}{1+x^2}$	10-x

Варіант	Вираз	Аргумент заміни
5	$e^{2x} + x^2 - 1$	5-x
6	$e^{-2x} + x^2 - x$	x+3
7	$\frac{x^3 + 2x}{\sqrt{1 + e^x}}$	3x
8	$x^3 - 3x + \frac{8}{\sqrt{1 + x^2}}$	$\frac{x}{4}$
9	$x^4 - \frac{0{,}8x}{\sqrt{1 + 2x}}$	x^2
10	$3x^3 + \frac{1}{x} + e^{2x^2}$	x-7

Завдання 2. За допомогою функції **int** розв'язати визначений інтеграл функції на заданому інтервалі (табл.5.2).

Таблиця 5.2

Варіанти індивідуальних завдань

Варіант	Інтеграл функції	Інтервал
1	$\frac{\cos(\pi x^2)}{\sqrt{1 - 3x}}$	$(\frac{\pi}{4}; \frac{\pi}{2})$
2	$\sqrt{1 + 4x} * \sin(\pi x)$	$(0; \frac{\pi}{2})$
3	$\sqrt{2 + 3x} * tg\frac{\pi x^3}{2}$	$(\frac{\pi}{2}; \frac{3\pi}{2})$
4	$(e + x) * \sin(\pi\sqrt{x - 1})$	$(0; \frac{\pi}{2})$
5	$\cos\left(\frac{\pi x}{2}\right) + x^2$	$(0; \frac{\pi}{4})$
6	$\sin(\pi\sqrt[3]{1 + x})$	$(\frac{\pi}{4}; \frac{\pi}{2})$
7	$\sqrt{x^2 + 1} * \cos(\frac{\pi x}{2})$	$(0; \frac{3\pi}{2})$
8	$3x + x^2 * \sin(x)$	$(\frac{\pi}{4}; \frac{\pi}{2})$

Варіант	Інтеграл функції	Інтервал
9	$x^3 * tg(\frac{\pi x}{3})$	$(0;\frac{\pi}{4})$
10	$\arccos\left(e^{-\sqrt{3x}}\right)$	$(\frac{\pi}{2};\frac{3\pi}{2})$

Завдання 3. За допомогою функції **ezplot** побудувати в графічному вікні графік функції згідно варіанту (табл.5.3).

Таблиця 5.3

Варіанти індивідуальних завдань

Варіант	Функція	Параметр a	Параметр b
1	$\frac{\cos(\pi x^3)}{\sqrt{10-5x}}$	0,2	4
2	$\sqrt{1+x} * \sin(\pi x) + 1$	0,1	1
3	$\sqrt{2+x} * tg\frac{\pi x^3}{4}$	0,4	4
4	$(e-x) * cos\ (\pi)$	0,5	5
5	$\cos\left(\frac{\pi x}{2}\right) - x^2 + 3$	1,1	2
6	$\sin\ (\pi\sqrt[3]{10+x}^2)$	1,6	5
7	$\sqrt{x^2-11} * arccos\ (\frac{\pi x}{2})$	0,5	4,2
8	$x - x^2 * \sin\ (x+1)$	1,6	2,9
9	$x^7 * cos\left(\frac{\pi x}{3}\right) + 10$	0,9	1,7
10	$\arccos\left(e^{-\sqrt{3x}}\right) + 1 - 1$	0,7	1,9

5.3. Контрольні запитання

1. Що таке символьний розв'язок рівняння?
2. Які методи можна використовувати для отримання символьного розв'язку рівняння?

3. Яка різниця між числовим та символьним розв'язком рівняння?

4. Які функції або методи можна використовувати в мові програмування або символьних обчислювальних системах для отримання символьного розв'язку рівняння?

5. Які основні принципи символьних обчислень застосовуються при розв'язанні рівнянь?

6. Які типи рівнянь можуть бути розв'язані у символьному вигляді?

7. Як можна використовувати символьні розв'язки рівнянь для виконання алгебраїчних операцій та обчислень?

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 6
РОЗВ'ЯЗУВАННЯ ЗАДАЧ ЗА ДОПОМОГОЮ КАТАЛОГУ CONTROL SYSTEM TOOLBOX

Мета роботи

Ознайомлення та дослідження можливостей застосування пакету при представленні та претворенні математичних моделей за допомогою Control System ToolBox для розв'язання різноманітних задач керування.

6.1. Теоретичні відомості

В теорії автоматичного управління (керування, регулювання) елементи автоматичних систем з точки зору їх динамічних властивостей зображують за допомогою динамічних ланок. Під динамічною ланкою розуміють математичну модель штучно виділеної частини системи, яка характеризується певним (найпростішим) алгоритмом передачі сигналу з входу ланки на її вихід (рис. 6.1) [14].

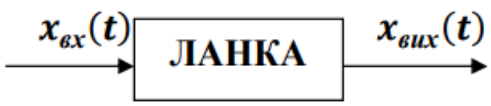


Рис. 6.1 Графічна схема динамічної ланки автоматизованої системи

Вхідну величину ланки позначають $x_{\text{вх}}(t)$ або, в загальному випадку, $x_1(t)$ а вихідну величину – позначають $x_{\text{вих}}(t)$ або $x_2(t)$ [14].

Вхідна і вихідна величини відповідають фізичним величинам, що зображують дію попередньої ланки на дану ланку ($x_{\text{вх}}(t)$) і дію даної ланки на наступну ($x_{\text{вих}}(t)$). Передача сигналу ланкою відбувається тільки в одному напрямі – сигнал передається з входу ланки на її вихід [14].

У теорії автоматичного керування однією з основних характеристик об'єкта є передатна функція.

Передатна функція ланки (системи) $W(p)$ (визначається як відношення зображень за Лапласом вихідної $Y(p)$ і вхідної $X(p)$ величин за нульових початкових умов [15]:

$$W(p) = \frac{Y(p)}{X(p)}. \quad (6.1)$$

Нехтуючи впливом зовнішнього збурення *L(t)*, у загальному вигляді рівняння динаміки (6.1) матиме вигляд [15]:

$$a_n \frac{d^n y(t)}{dt^n} + a_n \frac{d^{n-1} y(t)}{dt^{n-1}} + \cdots + a_1 \frac{dy(t)}{dt} + a_0 y(t)$$
$$= b_n \frac{d^m x(t)}{dt^m} + b_{n-1} \frac{d^{m-1} x(t)}{dt^{m-1}} + \cdots + b_1 \frac{dx(t)}{dt} + b_0 x(t). \quad (6.2)$$

Провівши перетворення за Лапласом рівняння (6.2) при нульових початкових умовах маємо:

$$a_n p^n Y(t) + a_{n-1} p^{n-1} Y(t) + \cdots + a_1 pY(p) + a_0 Y(p)$$
$$= b_m p^m X(t) + bp^{m-1} X(t) + \cdots + b_1 pX(p) + b_0 X(p). \quad (6.3)$$

Із вищезазначеного випливає, що передатна функція ланки має наступний вигляд:

$$W(p) = \frac{Y(p)}{X(p)} = \frac{b_m p^m + b_{m-1} p^{m-1} + \cdots + b_1 p + b_0}{a_n p^n + a_{n-1} p^{n-1} + \cdots + a_1 p + a_0}. \quad (6.4)$$

Для спрощення структурних схем використовують правила перетворення структурних схем. До найпростіших перетворень структурних схем, враховуючи правила визначення передатних функцій для типових з’єднань ланок [15]:

- **Послідовне** з'єднання (рис. 6.2):

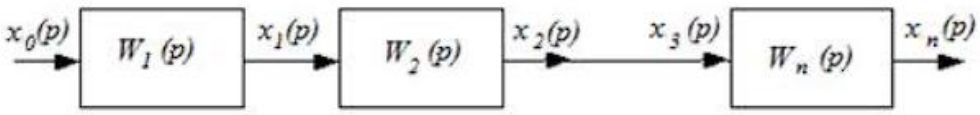


Рис. 6.2 Послідовне з'єднання динамічних ланок

Передатна функція $W(\mathrm{p})$ системи, утвореної послідовним з'єднанням, буде дорівнювати добутку їх передатних функцій $W_1(\mathrm{p})$ та $W_2(\mathrm{p})$.

$$\mathrm{W(p)} = W_1(\mathrm{p}) * W_2(\mathrm{p}). \quad (6.5)$$

- **Паралельне з'єднання** (рис.6.3):

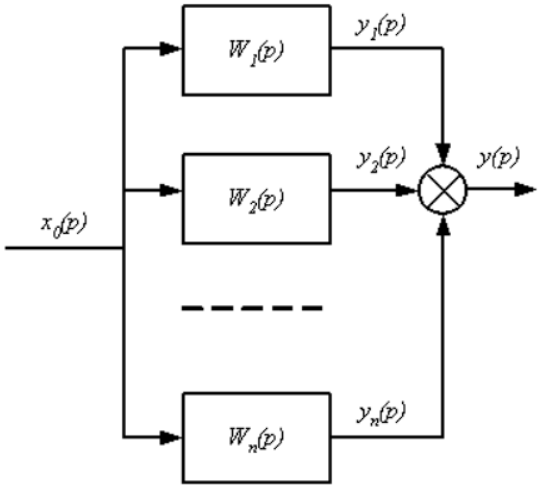


Рис. 6.3 Паралельне з'єднання динамічних ланок

Передатна функція W(p) системи, яка утворена паралельним з'єднанням буде дорівнювати сумі їх передатних функцій $W_1(\mathrm{p})$ та $W_2(\mathrm{p})$.

$$\mathrm{W}(\mathrm{p}) = W_1(\mathrm{p}) + W_2(\mathrm{p}) \qquad (6.6)$$

- **Ланка, охоплена зворотнім зв'язком** (рис. 6.4):

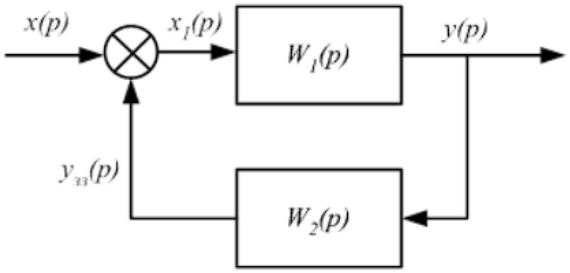


Рис. 6.3 Схема ланки охопленої зворотнім зв’язком

Для ланок охоплених зворотнім зв’язком передатна функція $W(p)$ системи визначається як:

$$W(p) = \frac{W_1(p)}{1 \pm W_1(p) * W_2(p)}. \qquad (6.7)$$

Зазначимо, що у формулі (6.7) стоїть знак «±», тому що в даному випадку, при додатному зворотному зв’язку стоїть знак «–», а у випадку від’ємного зворотного зв’язку – знак «+».

Для систем автоматичного керування визначають часові характеристики такі як:

- перехідна характристика *h(t);*
- вагова характеристика *w(t).*

Перехідна функція *h(t)* системи визначається як [15]:

$$h(t) = L^{-1}\left\{\frac{W(p)}{p}\right\}. \quad (6.8)$$

Ваговою функцією (характеристикою) *w(t)* визначається як [15]:

$$w(t) = L^{-1}\{W(p)\}. \quad (6.9)$$

Комплексна передатна функція (КПФ) ланки визначається як [14]:

$$W(jw) = \frac{X_{\text{вих}}(jw)}{X_{\text{вих}}(jw)}. \quad (6.10)$$

Зазвичай, комплексну передатну функцію $W(jw)$ визначають з передатної функції $W(p)$ (підстановкою $p=jw$ [13]:

$$W(jw) = W(p)|p = jw. \quad (6.11)$$

Тоді комплексна передатна функція матиме наступний вигляд:

$$W(jw) = P(w) + JQ(w). \quad (6.12)$$

де $P(w)$ та $Q(w)$ дійсна та уявна частина відповідно

До частотних характеристик відносяться [14]:

1. **Амплітудно-частотна характеристика (АЧХ)** визначається як:

$$|W(jw)| = \sqrt{P^2(w) + Q^2(w)}. \quad (6.13)$$

2. **Фазово-частотна характеристика (ФЧХ)** визначається як:

$$\varphi(w) = arctg\frac{Q(w)}{P(w)}. \quad (6.14)$$

3. **Логарифмічна амплітудно-частотна характеристика (ЛАЧХ)** визначається як:

$$L_{\text{ЛАФЧ}}(w) = 20\lg|W(jw)|, \quad (6.15)$$

4. **Логарифмічна фазово-частотна характеристика (ЛФЧХ)** визначається як:

$$L_{\text{ЛФЧХ}}(w) = \arg|W(jw)| \quad (6.16)$$

6.2. Можливості, які надає Toolbox користувачу

Control System Toolbox надає алгоритми та програми для систематичного аналізу, проектування та налаштування лінійних систем керування. Можна задати систему у вигляді передатної функції, простору станів, моделі з нульовим коефіцієнтом підсилення або частотної характеристики. Додатки та функції, такі як графік ступінчастої характеристики та графік Боде, дозволяють аналізувати та візуалізувати поведінку системи в часовій та частотній областях [16].

Також можна налаштовувати параметри компенсатора за допомогою методу формування петлі Боде. Інструментарій автоматично налаштовує як SISO (система с одним входом и одним вихідом), так і MIMO (системи зв'язку з рознесеними передавальними і приймальними антенами)-компенсатори, включаючи ПІД-регулятори.

Компенсатори можуть включати декілька блоків, що охоплюють декілька контурів зворотного зв'язку.

Перелік деяких основних функцій, що включає в себе Control System Toolbox [16]:

- **tf** - створення передатної функції;
- **ss** - створення просторової моделі стану системи;
- **zpk** - створення нуль-точка-коефіцієнтної форми передатної функції;
- **pid** - створення об'єкта ПІД-регулятора;
- **pole** і **zero** - визначення полюсів та нулів передатної функції системи;
- **tfdata** - отримання даних передатної функції;

- **step** – побудова перехідної характеристии системи автоматичного керування;
- **impulse** – побудова вагової характеристии системи автоматичного керування.
- **bode** - відображення логарифмічної амплітудно-частотної характеристики (ЛАЧХ) та логарифмічної фазо-частотної характеристики (ЛФЧХ);
- **nyquist** - побудова годографа Найквіста;
- **feedback** - створення зворотного зв'язку для системи.

Синтаксис функції **tf** у Control System Toolbox для створення передатної функції має наступний вигляд:

W= tf(num, den),

де num - вектор коефіцієнтів чисельника передатнаої функції; den - вектор коефіцієнтів знаменника передатної функції.

Обидва аргументи num і den мають векторну форму, яка відображає коефіцієнти передатної функції. Коефіцієнти передатної функції можуть бути виражені в будь-якій формі (числа, змінні, вирази).

Наприклад, для створення передатної функції

$$W(p) = \frac{p+1}{(p^2+3p+2)},$$

синтаксис функції матиме наступний вигляд:

```
num = [1, 1]; % Коефіцієнти чисельника: [1, 1]
den = [1, 3, 2]; % Коефіцієнти знаменника: [1, 3, 2]
W = tf(num, den); % Створення передатної функції W(s)
```

У результаті виконання даних команд в командному вікні з'явиться наступний текст:

```
W =

     s + 1
  -------------
  s^2 + 3 s + 2

Continuous-time transfer function.
```

Тобто за допомогою функції **tf** даний ToolBox дає можливість створити та визначити тип функції, в даному випадку це передатна функція неперервного часу (Continuous-time transfer function).

Синтаксис функції **ss** у Control System Toolbox для створення моделі простору стану (State-Space Model) має наступний вигляд:

$$sys = ss(A, B, C, D),$$

де A - матриця станів системи; B - матриця вхідних коефіцієнтів; C - матриця вихідних коефіцієнтів; D - матриця прямих коефіцієнтів.

Аргументи A, B, C, D мають бути матрицями або числами, які відображають коефіцієнти моделі. Розміри матриць мають бути відповідними для того, щоб забезпечити відповідність розмірності системи.

Отже, синтаксис матиме наступний вигляд:

```
A = [1, -2; 3, -4];
B = [1; 0];
C = [1, 1];
D = 0;
sys = ss(A, B, C, D) % Створення моделі просторового стану sys
```

У результаті виконання даних команд в командному вікні з'явиться наступний текст:

```
sys =

  A =
       x1  x2
   x1   1  -2
   x2   3  -4

  B =
       u1
   x1   1
   x2   0

  C =
       x1  x2
   y1   1   1

  D =
       u1
   y1   0

Continuous-time state-space model.
```

Функція **pole** в Control System Toolbox використовується для отримання полюсів системи керування. Функція **zero** для отримання нулів системи керування. Обидві функції мають один спільний синтаксис:

```
pole(sys)
zero(sys)
```

У результаті виконання даних команд у командному вікні з'явиться даний текст:

```
ans =

   -1.0000
   -2.0000


ans =

   -7.0000
```

Синтаксис функції **zpk** у Control System Toolbox для створення передатної функції у форматі нуль-полюс-коефіцієнт має наступний вигляд:

$$sys = zpk(z, p, k),$$

де z - нулі передатної функції; p - полюси передатної функції; k - коефіцієнт передатної функції.

Наприклад, для створення передатної функції

$$W(p) = \frac{p+1}{(p-2)(p+3)},$$

синтаксис функції матиме наступний вигляд:

```
z = [-1]; % Нулі: [-1]
p = [2, -3]; % Полюси: [2, -3]
k = 1; % Коефіцієнт: 1
sys = zpk(z, p, k) % Створення передатної функції W(s)
```

В результаті виконання даних команд у командному вікні з'явиться наступний текст:

```
sys =

     (s+1)
  -----------
  (s-2) (s+3)

Continuous-time zero/pole/gain model.
```

Синтаксис функції **pid** у Control System Toolbox для створення об'єкта ПІД-регулятора матиме наступний вигляд:

controller = pid(Kp, Ki, Kd),

де Kp - коефіцієнт пропорційної складової; Ki - коефіцієнт інтегральної складової; Kd - коефіцієнт диференціальної складової.

Коефіцієнти відповідають за внесок відповідної складової в роботу ПІД-регулятора. Зазвичай, коефіцієнти Kp, Ki, Kd встановлюються експериментально або ж на основі аналізу системи керування.

Наприклад, для створення об'єкта ПІД-регулятора із значеннями коефіцієнтів Kp = 1.2, Ki = 0.5 і Kd = 0.8, синтаксис матиме наступний вигляд:

```
Kp = 1.2; % Коефіцієнт пропорційної складової;
Ki = 0.5; % Коефіцієнт інтегральної складової;
Kd = 0.8; % Коефіцієнт диференціальної складової;
controller = pid(Kp, Ki, Kd); % Створення об'єкта PID-регулятора.
```

В результаті виконання даних команд у командному вікні з'явиться даний текст:

```
controller =

             1
  Kp + Ki * --- + Kd * s
             s

  with Kp = 1.2, Ki = 0.5, Kd = 0.8

Continuous-time PID controller in parallel form.
```

Синтаксис функції **step** у Control System Toolbox для відображення перехідної функції матиме такий вид:

[y,t]=step(sys),

де sys - об'єкт системи керування (наприклад, передатна функція, модель State-Space тощо); y - вектор, який містить значення вихідного сигналу імпульсної функції; t - вектор часу, який відповідає кожному значенню вектору y.

Наприклад, для відображення перехідної функції, синтаксис має бути наступним:

```
[y, t] = step(sys); % Перехідна функція
plot(t, y); % Побудувати графік відгуку системи
xlabel('Час');
ylabel('Вихідний сигнал');
title('Перехідна функція');
```

В результаті у графічному вікні отримано наступний графік перехідної функції (рис.6.4).

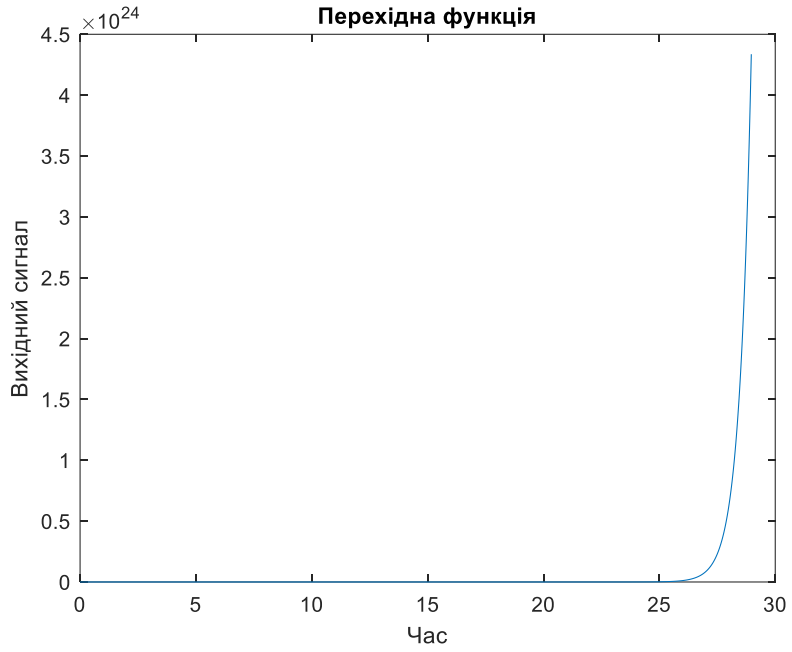


Рис. 6.4 Графік перехідної функції

Функція **impulse** в Control System Toolbox використовується для побудови графіка вагової характеристики. На рис.6.5 наведений приклад використання функції **impulse.**

```
sys = tf([1],[1, 2, 3]); % Створення передатної функції системи
impulse(sys); % Вагова характеристика
xlabel('Час');
ylabel('Вихідний сигнал');
title('Вагова характеристика');
```

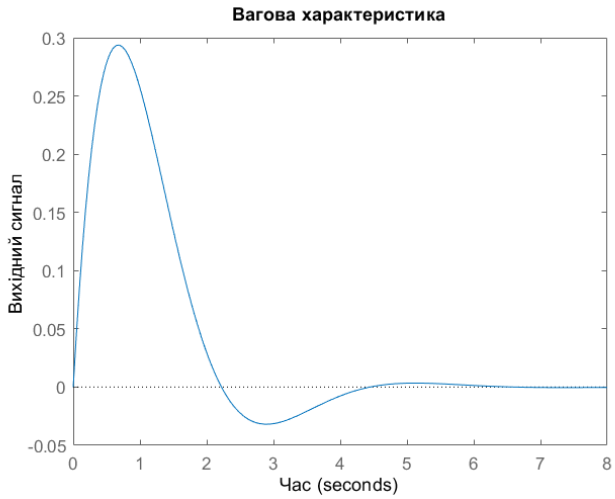


Рис. 6.5 Графік вагової функції

Функція **bode** в Control System Toolbox використовується для побудови графіка логарифмічних амплітудно-частотних та логарифмічних фазочастотних характеристик На рис. 6.6. представлений приклад використання функції **bode.**

```
sys=tf([1,1],[1,3,2]);%Створення передатної функції системи
bode(sys); %Побудова годографа логарифмічних амплітудно-частотних та логарифмічних фазочастотних характеристик
```

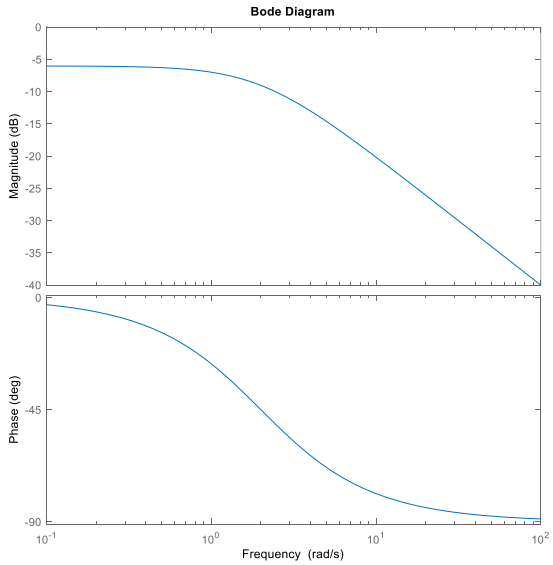


Рис. 6.6 Графік логарифмічних амплітудно-частотних та логарифмічних фазочастотних характеристик

Функція **nyquist** в Control System Toolbox використовується для годографа Найквіста системи керування. На рис.6.7 наведено приклад використання функції **nyquist.**

```
sys=tf([1,1],[1,3,2]);%Створення передатної функції системи
nyquist(sys);%Побудова годографа Найквіста
```

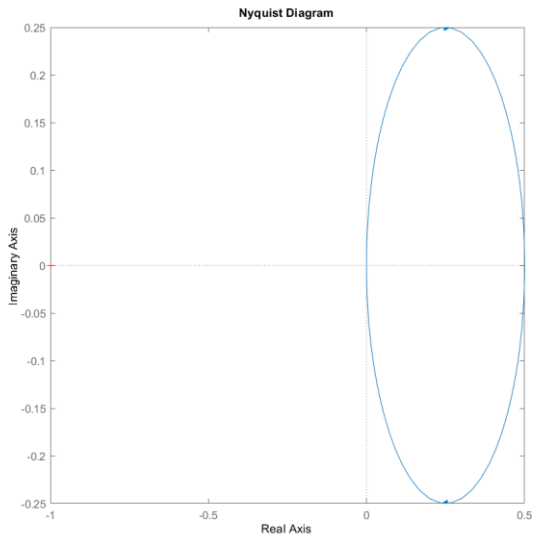


Рис. 6.7 Графік годограф Найквіста

6.3. Завдання до виконання комп'ютерного практикуму

Завдання 1. Створити передатну функцію за допомогою функції **tf** відповідно до варіанту, згідно з табл.6.1.

Завдання 2. Визначити полюси та нулі передатної функції за допомогою команд **pole** та **zero**.

Завдання 3. Створити **zpk** та **ss** модель системи.

Завдання 4. Створити об'єкт ПІД-регулятора із значеннями коефіцієнтів відповідно до варіанту згідно табл.6.2.

Завдання 5. Створити графіки перехідної, вагової, логарифмічних амплітудно-частотних та логарифмічних фазочастотних характеристик (табл.6.1) за допомогою команд: **step**, **impulse**, **bode**, **nyquist.**

Таблиця 6.1

Варіанти індивідуальних завдань

Варіант	Передатна функція
1	$W(p) = \frac{(p^3+1)(p^2+1)}{(p-2)(p+3)}$
2	$W(p) = \frac{1}{p(p^2+2p+3)}$
3	$W(p) = \frac{(p+5)(p^2-1)}{p(p-4)(p+4)}$
4	$W(p) = \frac{p-1}{p(p^2+2p-3)}$
5	$W(p) = \frac{(p-2)(p^2+1)}{p(p^2+4)(p-1)}$
6	$W(p) = \frac{1+p}{p^2(p^2+10p-5)}$
7	$W(p) = \frac{(p-5)(p+1)(p+2)}{p(p^2+2p+3)(p-1)}$
8	$W(p) = \frac{p(p+1)(p-2)}{p(p^2+p+6)(p+1)}$
9	$W(p) = \frac{p-1}{p(p^2+2p-3)}$
10	$W(p) = \frac{p(p-1)}{(p^2+p-1)(p-1)}$

Таблиця 6.2

Варіанти індивідуальних завдань

Варіант	**Коефіцієнт**		
	Kp	**Ki**	**Kd**
1	1.5	0.6	0.8
2	1.6	0.7	0.7
3	1.1	0.8	0.4

Варіант	**Коефіцієнт**		
	Kp	**Ki**	**Kd**
4	1.2	0.9	0.9
5	1.3	0.5	0.4
6	1.4	0.4	1
7	1.5	0.3	1.1
8	1.6	0.4	0.8
9	1.7	0.8	0.7
10	1.8	0.9	0.4

6.4. Контрольні запитання

1. Які існують типові з’єднання ланок?
2. Як визначається АЧХ та ФЧХ автоматизованої системи?
3. Що являє собою перехідна та вагова характеристики автоматизованої системи?
4. За допомогою яких функцій можна визначити перехідну та вагову характеристики автоматизованої системи?
5. Який синтаксис функції tf?
6. Який синтаксис функції ss?
7. Які функції та інструменти надає Control System Toolbox для розв'язування задач керування?

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 7
РОЗВ'ЯЗУВАННЯ ЗАДАЧ ЗА ДОПОМОГОЮ КАТАЛОГУ SIGNAL PROCESSING TOOLBOX

Мета роботи

Ознайомлення та дослідження можливостей застосування пакету Signal Processing Toolbox для генерації вихідних сигналів.

7.1. Теоретичні відомості

Електричний сигнал – електрична величина (напруга, струм та ін.), яка змінюється з часом [17].

Сигнали поділяють на такі дві групи: детерміновані та випадкові (стохастичні).

Детермінованими - це сигнали, значення яких у будь-який момент часу є точно відоме, тобто їх можна передбачити безпомилково. Такі сигнали не несуть нової інформації, проте їх використовують як тестові сигнали при дослідженні різних електронних кіл та пристроїв [17].

Випадковими називають такі сигнали, значення яких у будь-який момент часу неможливо передбачити абсолютно точно. Випадковими сигналами є різноманітні електромагнітні коливання атмосферного та промислового походження, а також сигнали інших передавальних станцій, які перешкоджають прийманню інформаційних сигналів. Отже, випадкові сигнали можна поділити на корисні та завади (шуми) [17].

Основні поняття, що стосуються сигналів:

- Амплітуда - це висота сигналу, якої він досягає, загасаючи (підстрибуючи вгору і вниз у своїй синусоїді). Таким чином, амплітуду також часто називають потужністю сигналу. Чим більша амплітуда сигналу на початку, тим далі він зможе поширюватися.

- Частота сигналу показує, скільки разів сигнал повторюється протягом однієї секунди. Одиницями частоти є цикли в секунду, або герци (скорочено Гц).
- Фаза - це положення хвилі в певний момент часу (мить) на циклі форми сигналу. Вона забезпечує вимірювання точного положення хвилі в межах її циклу, використовуючи або градуси (0°-360°), або радіани (0-2π).

Спектральний аналіз використовується для дослідження частотної складової сигналу або системи, надаючи інформацію про розподіл енергії сигналу чи системи в залежності від частоти сигналу.

Перетворення Фур'є (Fourier transaform) — інструмент спектрального аналізу неперіодичних сигналів. Основні відмінності перетворення Фур'є від ряду Фур'є, викликані припущенням, що період неперіодичного сигналу прямує до нескінченності:

– частота з дискретних відліків перетворюється в неперервний параметр перетворення;

– результатом розрахунків замість коефіцієнтів ряду є функція частоти (спектральна функція).

В результаті таких перетворень отримаємо функцію, яка розраховується за формулою [18]:

$$S(jw) = \int_{-\infty}^{+\infty} s(t) * e^{-jwt}\, dt. \qquad (7.1)$$

Функція $S(jw)$ ω отримала назву спектральна густина сигналу s(t), а формула (7.1) — пряме перетворення Фур'є. Обернене перетворення Фур'є розраховується за формулою [17]:

$$S(jw) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} S(jw) * e^{jwt}\, dw. \qquad (7.2)$$

На рис. 7.1,а наведено послідовність прямокутних імпульсів (τ — тривалість імпульсів, Т — їх період, А — амплітуда), а на рис. 7.1,б— спектр даної послідовності (дискретні відліки), як видно з рисунку, огинаючою

даного спектру є функція $\frac{\sin(x)}{x}$ (пунктирна лінія). Відстань між гармоніками спектру рівна $\frac{1}{T}$, тобто частоті послідовності імпульсів, ширина пелюсток спектру обернено пропорційна до тривалості імпульсу $\frac{1}{\tau}$ [18].

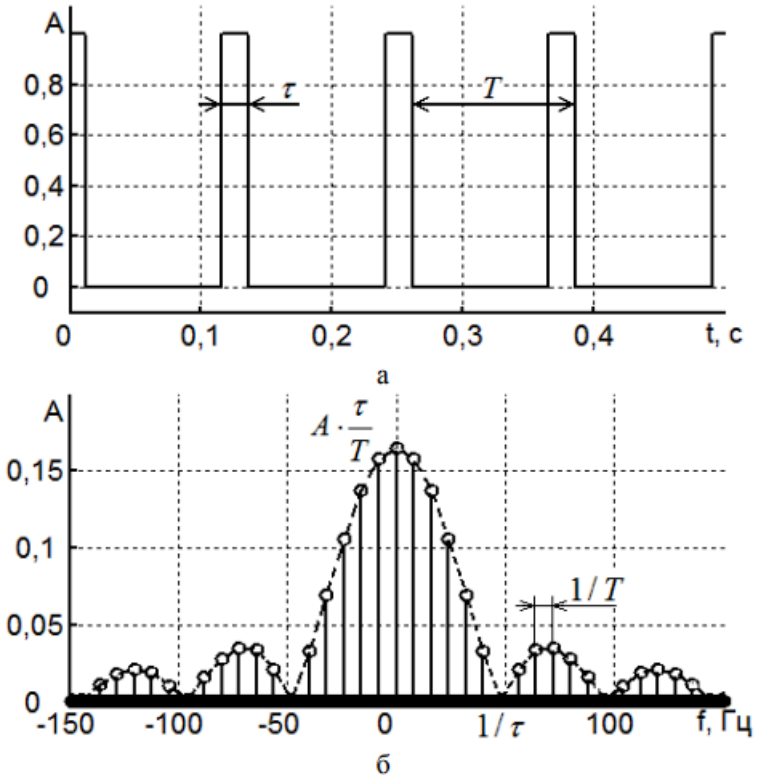


Рис. 7.1 а) Послідовність імпульсів(прямокутних);

б) їх спектр

Під фільтрацією розуміють будь-яке перетворення інформації (сигналів, результатів вимірювань і т.д.), при якому у вхідній послідовності оброблюваних даних змінюють певні співвідношення (динамічні або частотні) між різними компонентами цих даних. До операцій фільтрації відносять диференціювання, інтегрування, згладжування і поділ сигналів [19].

Цифрова фільтрація полягає в цифровому перетворенні послідовності числових відліків вхідного сигналу (рис. 7.2) [19].

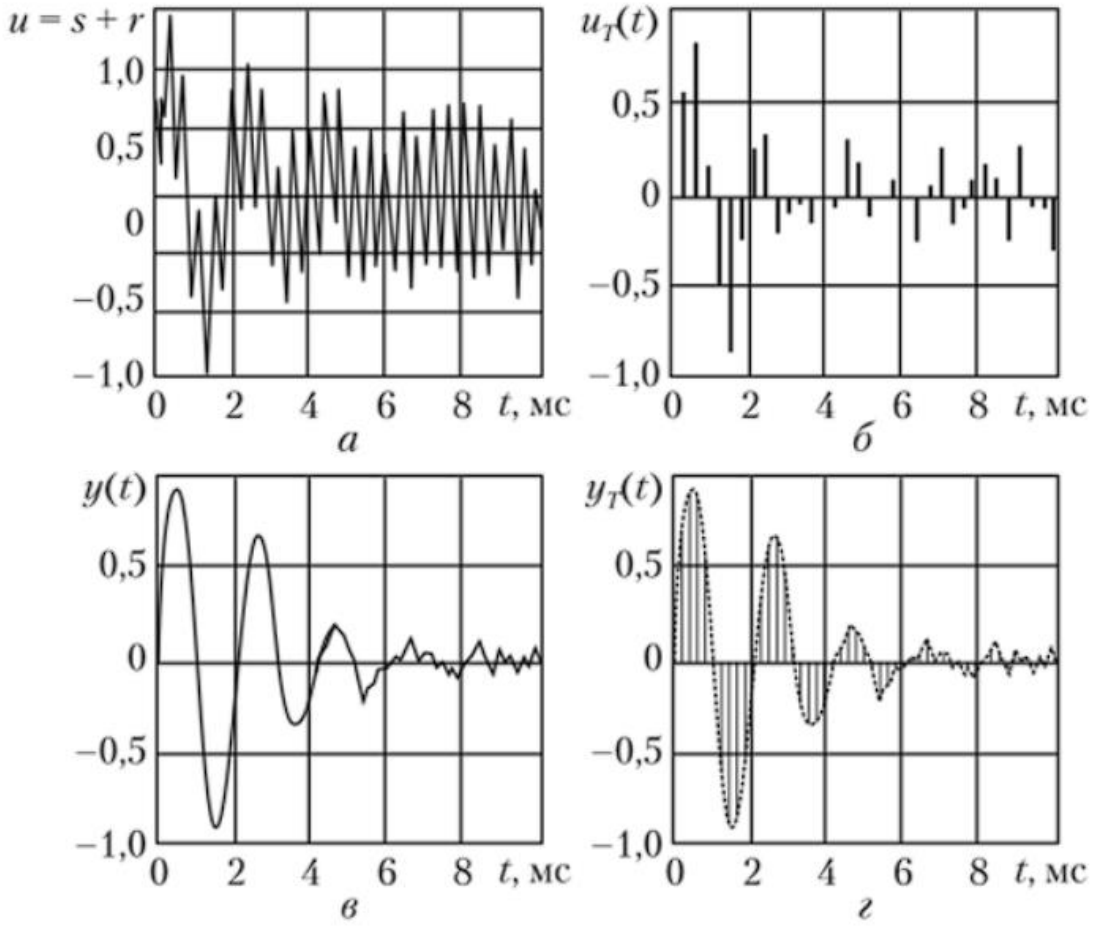


Рис. 7.2 До фільтрації сигналів:

а) вихідний сигнал з шумом; б) сигнал на виході аналогового фільтра;

в) дискретизований вихідний сигнал з шумом; г) сигнал на виході цифрового фільтру

7.2. Можливості, які надає Toolbox користувачу

Signal Processing Toolbox надає функції та програми для керування, аналізу, попередньої обробки та вилучення характеристик з рівномірно та нерівномірно дискретизованих сигналів. Toolbox включає в себе інструменти для проектування та аналізу фільтрів, повторної дискретизації, згладжування, детрендування та оцінки спектру потужності.

Toolbox підтримує широкий спектр операцій з обробки сигналів від генерації форми сигналу до проектування і реалізації фільтрів, параметричного моделювання і спектрального аналізу.

Реалізації фільтрів, параметричного моделювання та спектрального аналізу.

Toolbox містить такі категорії інструментів:

- побудова та аналіз графіків сигналів;
- спектральний аналіз;
- фільтрації сигналів;
- аналіз конструкцій фільтрів;
- перетворення Фур’є та інші перетворення.

Основними об'єктами, з якими працюють функції набору інструментів, є сигнали та системи.

Ці функції зосереджені на цифрових, або дискретних, сигналах і фільтрах, на відміну від аналоговим, або неперервним, сигналам. Основними типами фільтрів, які підтримує набір інструментів, є лінійний, незмінний у часі цифровий фільтр з одним входом і одним виходом.

Можна представляти лінійні, інваріантні в часі системи, використовуючи одну з декількох моделей (наприклад, передатна функція, простір станів, нульовий коефіцієнт підсилення та переріз другого порядку) і конвертувати їх.

Основні функцій для генерування форми сигналу в Toolbox наведені в табл.7.1.

Таблиця 7.1

Основні функцій для генерування форми сигналу

Функція	**Опис функції**
chirp	Косинус розгорнутої частоти
diric	Функція Діріхле або періодична синусоїда
gauspuls	Гауссово-модульований синусоїдальний радіочастотний імпульс
gmonopuls	Гауссів моноімпульс
pulstran	Імпульсний потік

Функція	Опис функції
rectpuls	Вибірковий аперіодичний прямокутник
sawtooth	Пилкоподібна або трикутна хвиля
sinc	Функція Sinc
square	Квадратна хвиля
tripuls	Вибірковий аперіодичний трикутник
vco	Генератор, керований напругою

Також пакет Signal Processing Toolbox можна знайти в розділі «APP», де його реалізовано в графічному середовищі. Щоб звернутися до нього, необхідно перейти по відповідному вкладенні та відкрити випадне вкладення (рис.7.3) та знайти в списку заголовків Signal Processing and Comunications (рис. 7.4).

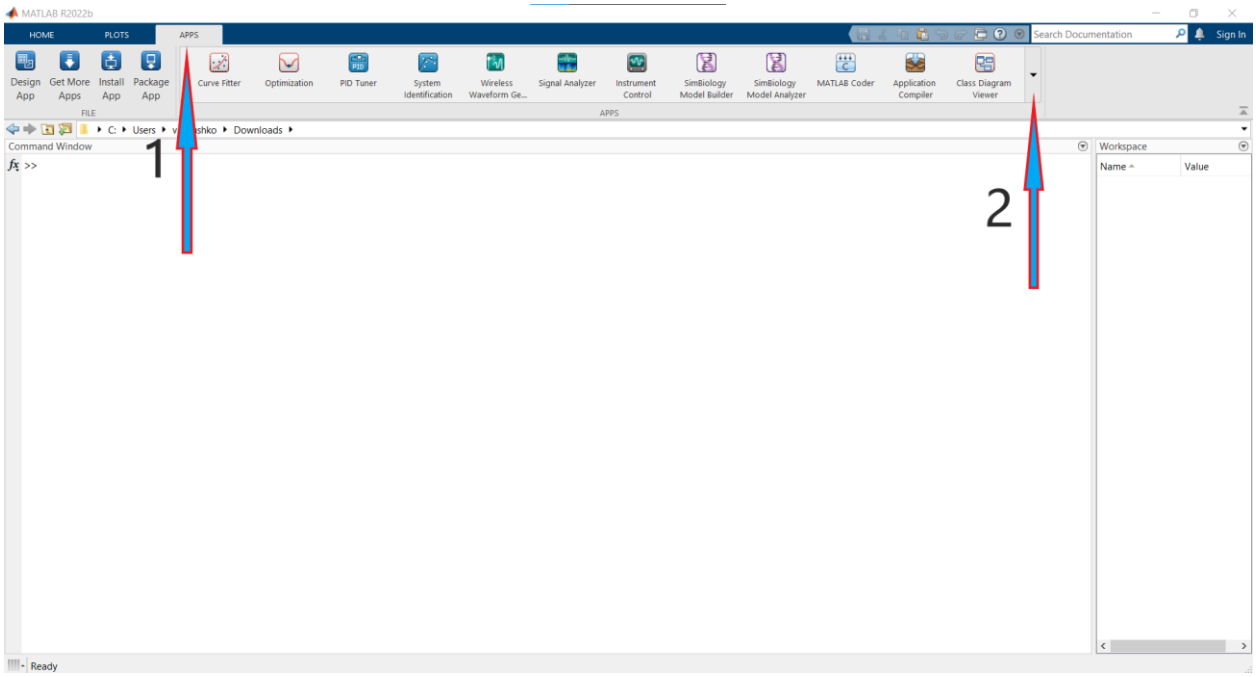


Рис. 7.3 Місцезнаходження Signal Processing Toolbox

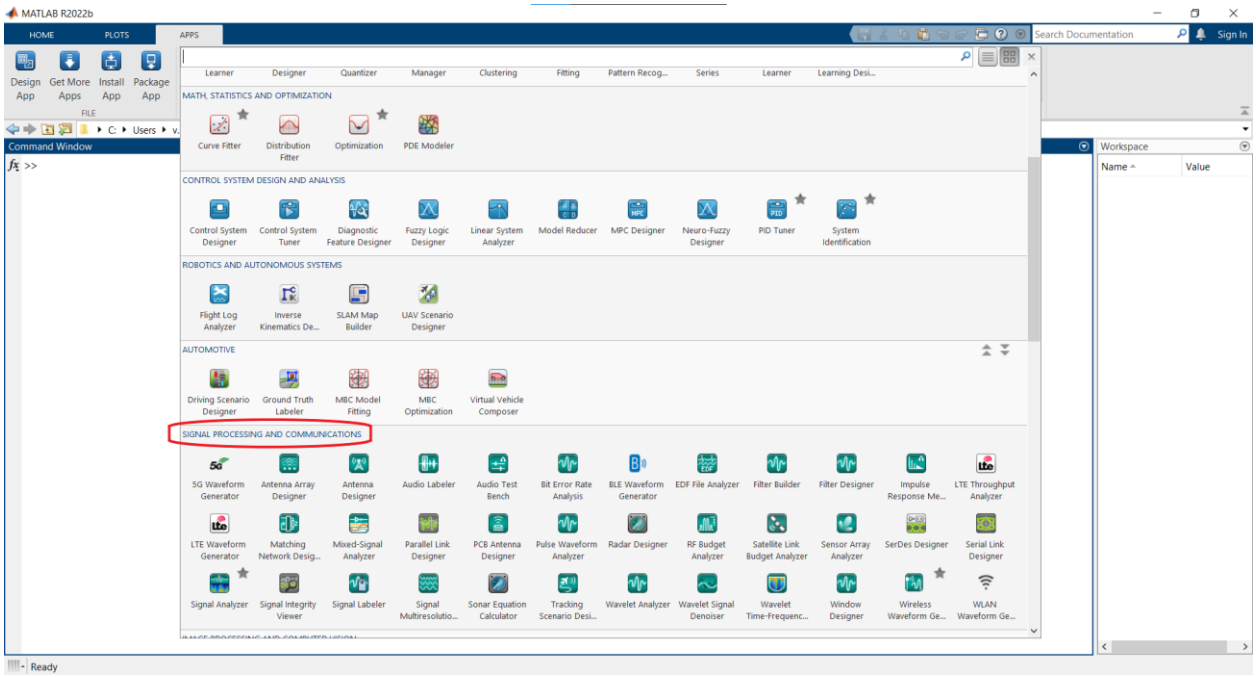
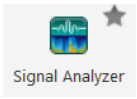


Рис. 7.4 Місцезнаходження Signal Processing and Comunications

Signal Processing and Comunications має досить великий набір функцій, які можна в подальшому використовувати, аналізуючи необхідний сигнал.

Потрібно знайти та запустити Signal Analyzer, натиснувши на відповідну піктограму 

, він дозволяє візуалізувати та аналізувати сигнали у часовому та частотному доменах. Після запуску з'явиться відповідне вікно виконання функції Signal Analyzer (рис. 7.5):

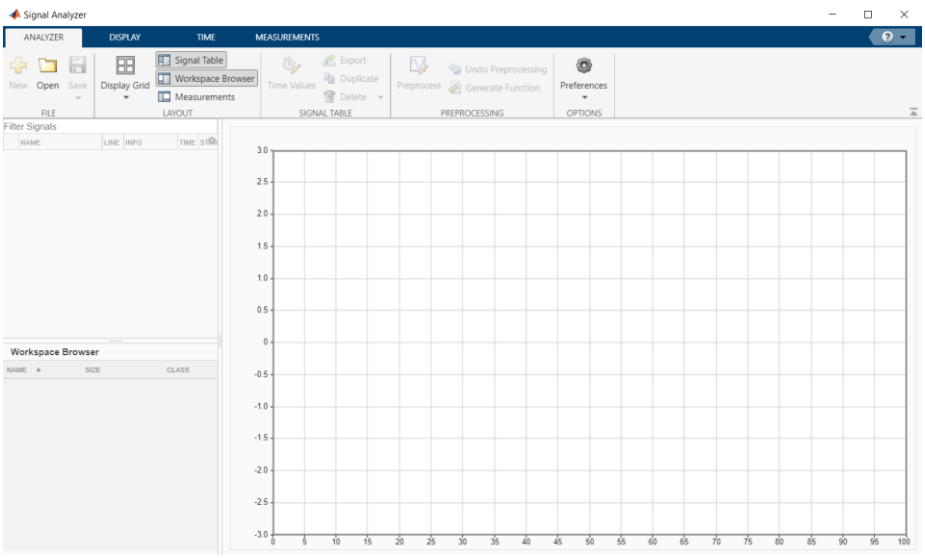


Рис. 7.5 Вікно інтерфейсу Signal Analyzer

При створенні вихідного сигналу в робочому полі Filter Signal з'явиться змінна вихідного сигналу та часу (рис.7.6), яку потрібно перетягнути на робочу область Signal Analyzer для відображення сигналу та подальшого його аналізу.

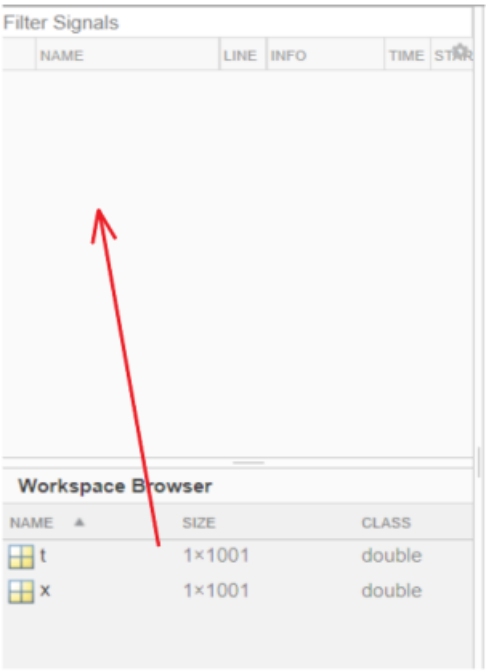


Рис.7.6 Додавання сигналу в робочу область Signal Analyzer

Наприклад, задамо відповідний сигнал:

```
% Створення сигналу
    Fs = 1000; % Частота дискретизації (Hz)
    t = 0:1/Fs:1; % Вектор часу (1 секунда)
    f = 10; % Частота сигналу (Hz)
    x = sin(2*pi*f*t); % Створення сигналу
    % Відкриття Signal Analyzer
    sa = dsp.SignalAnalyzer('SampleRate', Fs, 'TimeSpan', 1, 'ShowLegend', true);
    % Аналіз сигналу
    sa(x);
    % Налаштування відображення
    sa.ChannelNames = {'Signal'};
    sa.Title = 'Аналіз сигналу';
    sa.XLabel = 'Час (с)';
    sa.YLabel = 'Амплітуда';
    % Додавання горизонтальних ліній та текстової мітки
    line(sa.Axes, [0, 1], [0, 0], 'Color', 'r', 'LineWidth', 1.5);
    text(sa.Axes, 0.5, 0.2, '10 Hz', 'Color', 'r');
    % Закриття Signal Analyzer
    release(sa);
```

Запускаємо набраний код та переходимо до Signal Analyzer, виконавши вищезазначені дії, отримуємо графік вихідного сигналу, який наведено на рис.7.7.

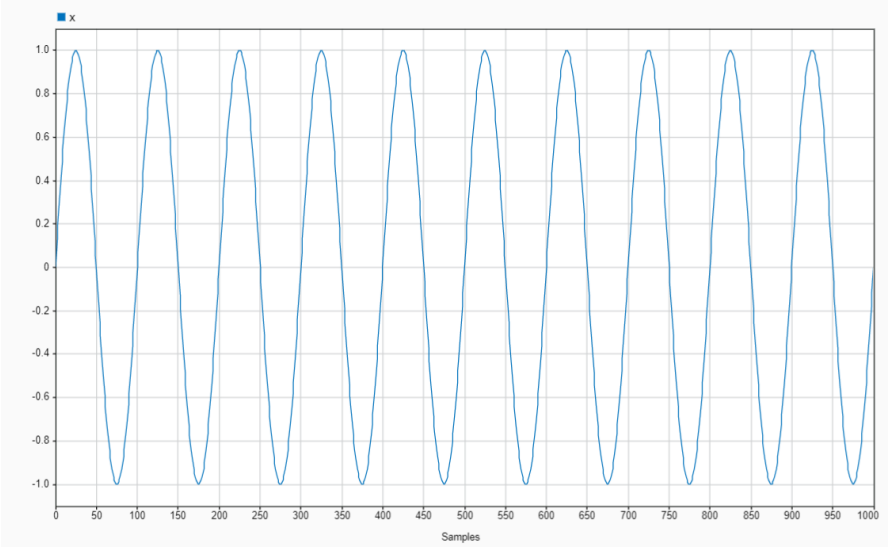


Рис. 7.7 Заданий вихідний сигнал

Проведемо спектральний аналіз сигналу натиснувши на піктограму 

(рис.7.8).

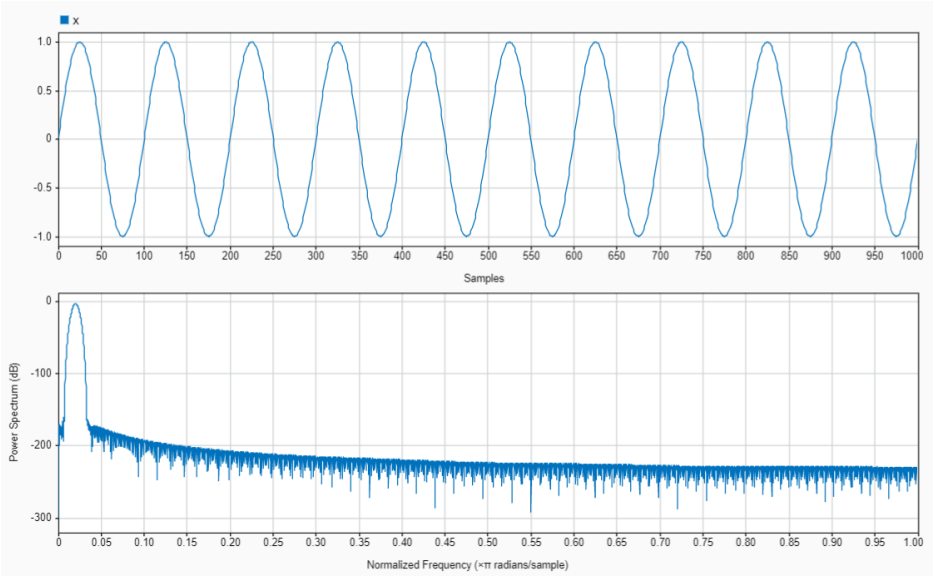


Рис. 7.8 Спектральний аналіз заданого вихідного сигналу

Проаналізувавши графік, можна спостерігати нормалізовану частоту цього сигналу, яка складає 0,01.

Нормалізована частота сигналу визначається відносно частоти дискретизації. У вище створеному коді, частота дискретизації задана як

Fs = 1000 Гц, а частота сигналу f = 10 Гц. Щоб визначити нормалізовану частоту, необхідно поділити фізичну частоту на частоту дискретизації:

$$F_{norm} = \frac{f}{F_S} = \frac{10}{1000} = 0{,}01 \quad (7.3)$$

Отже, нормалізована частота цього сигналу становить 0.01.

Нормалізована частота використовується для відносного вимірювання частоти сигналу у відношенні до частоти дискретизації, і вона немає фізичної одиниці вимірювання, такої як герц (Гц) для фізичної частоти.

Створюємо пульсовий сигнал за допомогою функції square та відтворюємо його. Потім відкриваємо Signal Analyzer за допомогою об'єкта dsp.SignalAnalyzer та передаємо сигнал для візуалізації за допомогою signalAnalyzer(x).

```
% Створення та відтворення пульсового сигналу
    fs = 1000; % Частота дискретизації
    t = 0:1/fs:1; % Вектор часу
    f = 1; % Частота пульса
    x = square(2*pi*f*t); % Пульсовий сигнал
    % Відкриття Signal Analyzer
    signalAnalyzer = dsp.SignalAnalyzer(fs);
    % Відображення пульсового сигналу
    signalAnalyzer(x);
    % Налаштування параметрів візуалізації
    signalAnalyzer.SampleRate = fs;
    signalAnalyzer.TimeSpan = 1; % Тривалість вікна аналізу
    % Відображення спектральних характеристик сигналу
    signalAnalyzer.SpectrumType = 'Power density';
    signalAnalyzer.SpectrumWindow = 'Hamming';
    signalAnalyzer.SpectrumUnits = 'dBW';
    % Відображення спектрограми сигналу
    signalAnalyzer.SpectrogramWindow = 'Hamming';
    signalAnalyzer.SpectrogramOverlapPercent = 75;
    % Виконання аналізу сигналу
    signalAnalyzer(x);
```

Аналогічно, після виконання команд запускається SignalAnalyzer та в графічному вікні з'являється заданий вихідний сигнал та відповідно вибрані спектральні характеристики Spectrum (рис.7.9).

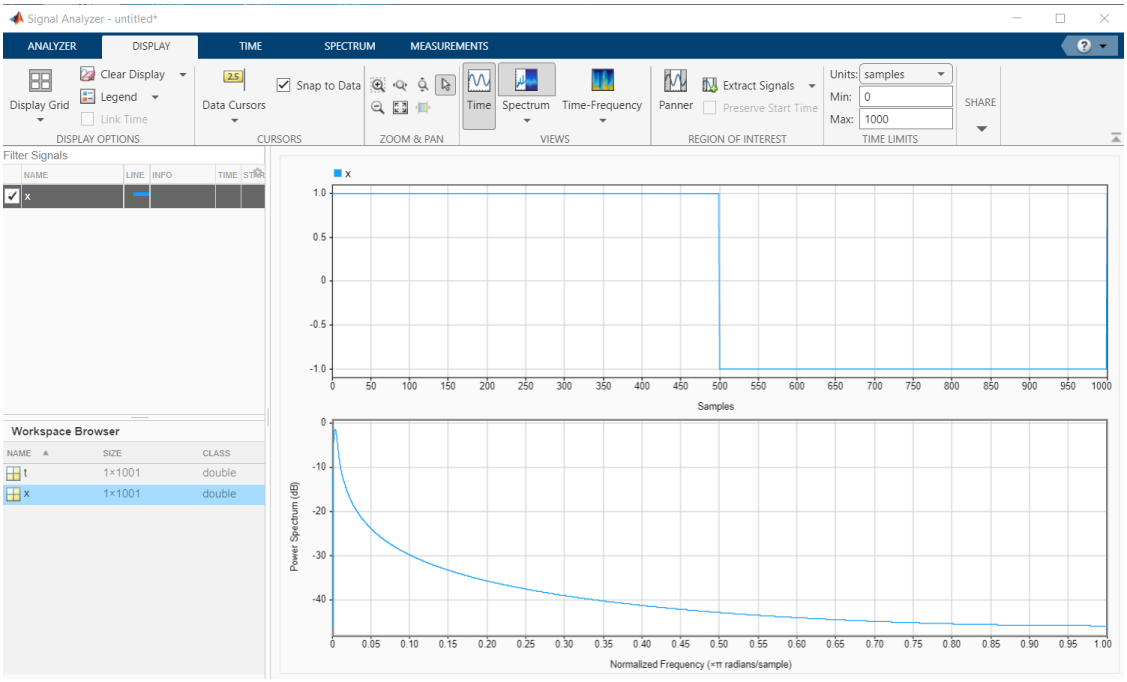


Рис. 7.9 Signal Analyzer для вхідного пульсового сигналу

На графіку можна спостерігати частоту сигналу, яка дорівнює 1 Гц та відповідно нормалізовану частоту вхідного пульсового сигналу, яка дорівнює $F_{norm} = 0{,}001$.

7.3. Завдання до виконання комп'ютерного практикуму

Завдання 1. Створити та проаналізувати синусоїдальний вихідний сигнал, виконати спектральний аналіз за допомогою Signal Analyzer відповідно до варіанту згідно табл.7.2.

Завдання 2. Створити та проаналізувати пульсовий вихідний сигнал, виконати спектральний аналіз за допомогою Signal Analyzer відповідно до варіанту згідно табл.7.3.

Таблиця 7.2

Варіанти індивідуальних завдань

Варіант	Вираз для створення сигналу	Частота дискретизації (Гц)	Частота сигналу (Гц)	Вектор часу (с)
1	$sin2\pi t$	1000	5	2

Варіант	Вираз для створення сигналу	Частота дискретизації (Гц)	Частота сигналу (Гц)	Вектор часу (с)
2	$sin\frac{3\pi}{2}t$	1001	10	4
3	$sin3\pi t$	1002	15	5
4	$sin4\pi t$	1003	20	9
5	$sin\frac{\pi}{2}t$	1004	25	3
6	$2sin2\pi t$	1005	30	6
7	$sin\frac{3\pi}{2}t$	1006	35	10
8	$sin2\pi t$	1007	40	12
9	$sin5\pi t$	1008	45	13
10	$sin\frac{\pi}{2}t$	1009	50	16

Таблиця 7.3

Варіанти індивідуальних завдань

Варіант	Вираз для створення сигналу	Частота дискретизації (Гц)	Частота сигналу (Гц)	Вектор часу (с)
1	$square2\pi$	1010	15	12
2	$square\frac{3\pi}{2}$	1011	10	4
3	$square3\pi$	1012	25	15
4	$square4\pi$	1013	10	9
5	$square\frac{\pi}{2}$	1014	15	3
6	$2square2\pi$	1015	5	16

Варіант	**Вираз для створення сигналу**	**Частота дискретизації (Гц)**	**Частота сигналу (Гц)**	**Вектор часу (с)**
7	$square\frac{3\pi}{2}$	1016	30	10
8	$square\pi$	1017	10	12
9	$square5\pi$	1018	15	13
10	$square\frac{\pi}{2}$	1019	20	16

7.4. Контрольні запитання

1. Що таке електричний сигнал?
2. Які бувають види сигналів?
3. Що таке амплітуда, частота та фаза сигналу?
4. Що таке перетворення Фур'є?
5. Що таке спектральний аналіз?
6. Для чого використовується спектральний аналіз в MatLab?
7. Для чого призначений Signal Processing Toolbox та сфера його застосування?

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 8
РОЗВ’ЯЗУВАННЯ ЗАДАЧ ЗА ДОПОМОГОЮ КАТАЛОГУ IMAGE PROCESSING TOOLBOX

Мета роботи

Ознайомлення та дослідження можливостей застосування пакету Image Processing Toolbox для розв'язування задач обробки зображень.

8.1. Теоретичні відомості

Методи обробки зображень (image processing) відіграють надзвичайно важливу роль у сучасній науці постійно вдосконалюються. Обробка зображень охоплює не лише покращення візуального сприйняття зображень, але й класифікацію об'єктів, що виконується під час аналізу зображень.

За допомогою методів обробки зображень, існує можливість застосовувати різноманітні алгоритми для розв'язання різних задач. Покращення зорового сприйняття зображень може включати фільтрацію шуму, видалення артефактів, розмиття, методи покращення контрастності та розширення динамічного діапазону.

Методи цифрової обробки широко застосовуються в промисловості, мистецтві, медицині, космосі. Вони застосовуються при керуванні процесами, автоматизації виявлення об'єктів, розпізнаванні образів і в багатьох інших. Цифрова передача зображень із космічних апаратів, цифрові канали передачі сигналів зображень вимагають забезпечення передачі все більших потоків інформації. Формування зображень, покращення якості та автоматизація обробки медичних зображень, включаючи зображення, що створюються електронними мікроскопами, рентгенівськими апаратами, томографами тощо, є предметом сучасних досліджень та розробок.

Комп'ютерна обробка зображень можлива після перетворення сигналу зображення з безперервної форми в цифрову форму. Ефективність обробки залежить від адекватності моделі, що описує зображення, необхідної для розроблення алгоритмів обробки.

8.2. Можливості, які надає Toolbox користувачу

Image Processing Toolbox надає повний набір еталонних стандартних алгоритмів та програм для обробки зображень, аналізу, візуалізації та розробки алгоритмів. Toolbox забезпечує сегментацію зображень, покращення зображень, зменшення шуму, геометричні перетворення та реєстрацію зображень, використовуючи традиційні методи обробки зображень. Інструментарій підтримує обробку 2D, 3D та зображень довільного розміру.

Фільтрація та покращення зображень: Image Processing Toolbox має широкий вибір фільтрів, включаючи медіанний, розмиття по Гаусу та ін. Надає користувачу використовувати функції для покращення контрастності, шумозниження та видалення артефактів.

Основні функції, доступні в Image Processing Toolbox наведені в табл.8.1 [21].

Таблиця 8.1

Функції, що доступні в Image Processing Toolbox

Функція	Опис функції
imread	Завантажує зображення з файлу.
imshow	Відображає зображення у вікні.
imwrite	Зберігає зображення у файлі з обраною форматом.
imadjust	Покращує контрастність зображення шляхом налаштування яскравості та контрастності.
imfilter	Застосовує фільтр до зображення для виконання операцій, таких як розмиття, збільшення чіткості або видалення шуму.
imresize	Змінює розмір зображення, збільшуючи або зменшуючи його
imrotate	Повертає зображення на заданий кут.
imcrop	Вирізає область зображення за вказаними координатами.
edge	Виявляє межі об'єктів на зображенні за допомогою різних алгоритмів, таких як Canny або Sobel.

Функція	Опис функції
hough	Виконує перетворення Хафа для виявлення ліній або колів на зображенні.
imhist	Побудова гістограми зображення для аналізу розподілу яскравості пікселів.

Image Processing Toolbox надає різноманітні фільтри для обробки зображень, які наведені в табл.8.2 [21].

Таблиця 8.2

Фільтри для обробки зображень

Розділ	Фільтри
Згладжування зображення (Smoothing Filters)	Average Filter (**imfilter**)
	Gaussian Filter (**imgaussfilt**)
	Median Filter (**medfilt2**)
	Bilateral Filter (**bilateralFilter**)
Підвищення гостроти зображення (Sharpening Filters)	Unsharp Masking (**imsharpen**)
	High Pass Filter (**imfilter**)
	Laplacian Filter (**fspecial** + **imfilter**).
Видалення шуму з зображення (Noise Removal Filters)	Wiener Filter (**wiener2**);
	Adaptive Median Filter (**medfilt2**)
	Non-local Means Denoising (**nlfilter**)
	Total Variation Denoising (**denoiseTV**)
Детекції меж зображення (Edge Detection Filters)	Sobel Filter (**imfilter**)
	Prewitt Filter (**imfilter**)
	Canny Edge Detector (**edge**)
	Roberts Filter (**imfilter**)
Розмиття зображення (Blur Filters)	Gaussian Blur (**imgaussfilt**)

Розділ	Фільтри
	Motion Blur (**fspecial** + **imfilter**)
	Average Blur (**imfilter**)
	Bokeh Blur (**imgaussfilt**)
Покращення контрастності (Contrast Enhancement Filters)	Histogram Equalization (**histeq**)
	Adaptive Histogram Equalization (**adapthisteq**)
	Contrast-Limited Adaptive Histogram Equalization (CLAHE) (**adapthisteq**)

Застосування Image Processing Toolbox для фільтрації та аналізу зображень та морфологічних операцій для покращення якості та виділення деяких об'єктів, використовуючи функції, які описані в табл. 8.1 та 8.2, створюємо код виконання програми :

```
% Завантаження та відображення зображення
image = imread('image.jpg');
figure;
imshow(image);
title('Оригінальне зображення');

% Конвертація зображення в градації сірого
grayImage = rgb2gray(image);

% Фільтрація зображення з використанням морфологічних операцій
se = strel('disk', 5);
filteredImage = imopen(grayImage, se);
figure;
imshow(filteredImage);
title('Зображення після морфологічної фільтрації');

% Визначення контурів об'єктів
edgeImage = edge(filteredImage, 'Canny');
figure;
imshow(edgeImage);
title('Контури об`єктів на зображенні');

% Визначення кількості об'єктів
[labeledImage, numObjects] = bwlabel(edgeImage);
disp('Кількість об`єктів:');
disp(numObjects);
```

В результаті отримуємо: оригінальне зображення (рис.8.2), зображення після морфологічної фільтрації (рис.8.3), контури об'єктів на зображенні (рис.8.4).

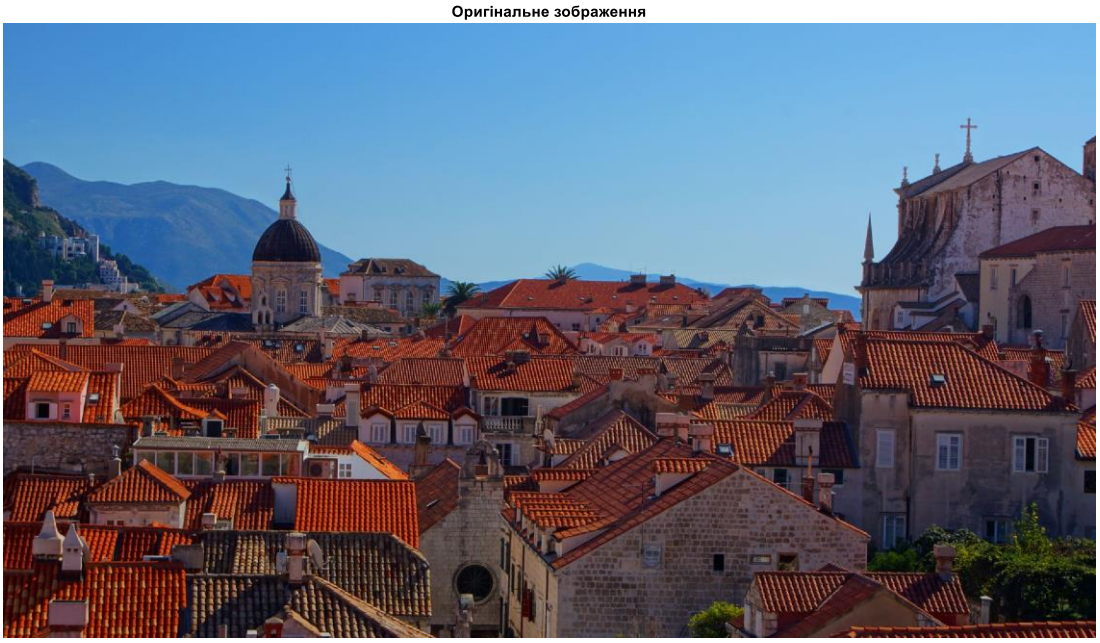


Рис. 8.2 Оригінальне зображення

Рис. 8.3 Зображення після морфологічної фільтрації

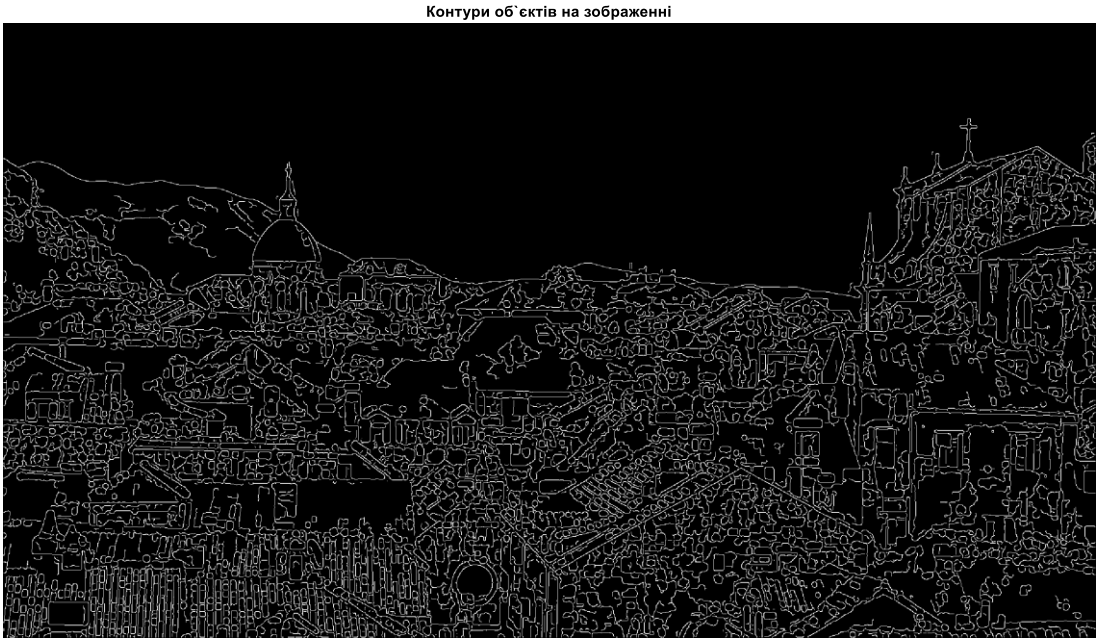


Рис. 8.4 Контури об'єктів на зображенні

Image Processing Toolbox також можна знайти в розділі програмного середовища MatLab - «APPS». Він має назву «Image Processing and Computer Vision» (рис.8.6).

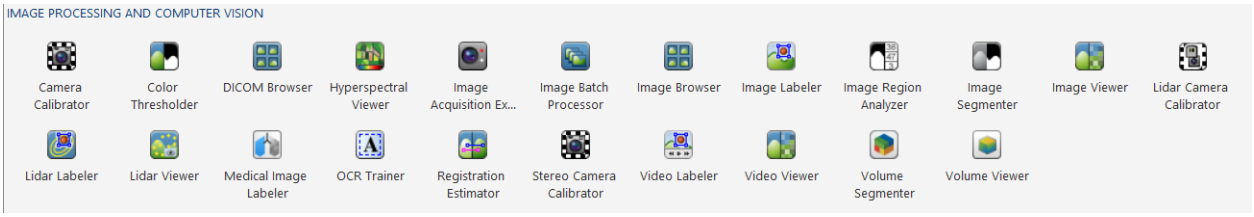


Рис. 8.6 Image Processing and Computer Vision

Потрібно знайти Color Thresholder та натиснути на відповідну піктограму, після чого з'явиться вікно інтерфейсу Color Thresholder (рис.8.7). Цей інструмент в Image Processing Toolbox, дозволяє визначити порогові значення для розпізнавання та виділення об'єктів на основі їх кольору.

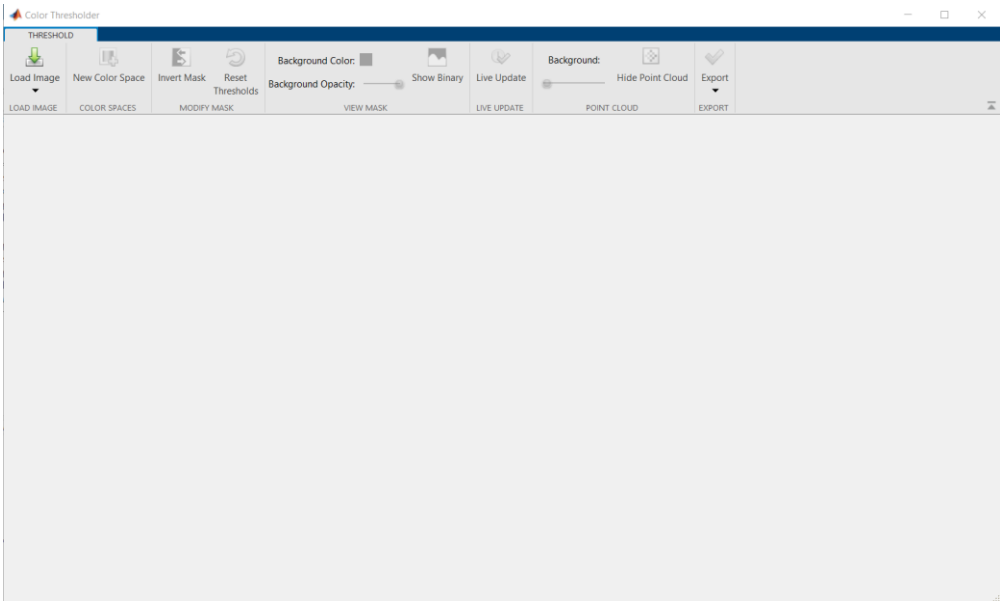


Рис. 8.7 Вікно меню інструменту Color Thresholder

Відповідно, для початку роботи з інструментом необхідно завантажити об'єкт, який потрібно проаналізувати, в даному випадку - це зображення в форматі «.**jpg».** Натиснувши на піктограму 

, Load Image > Load From File, завантажується попередньо обране зображення для аналізу (рис.8.8).

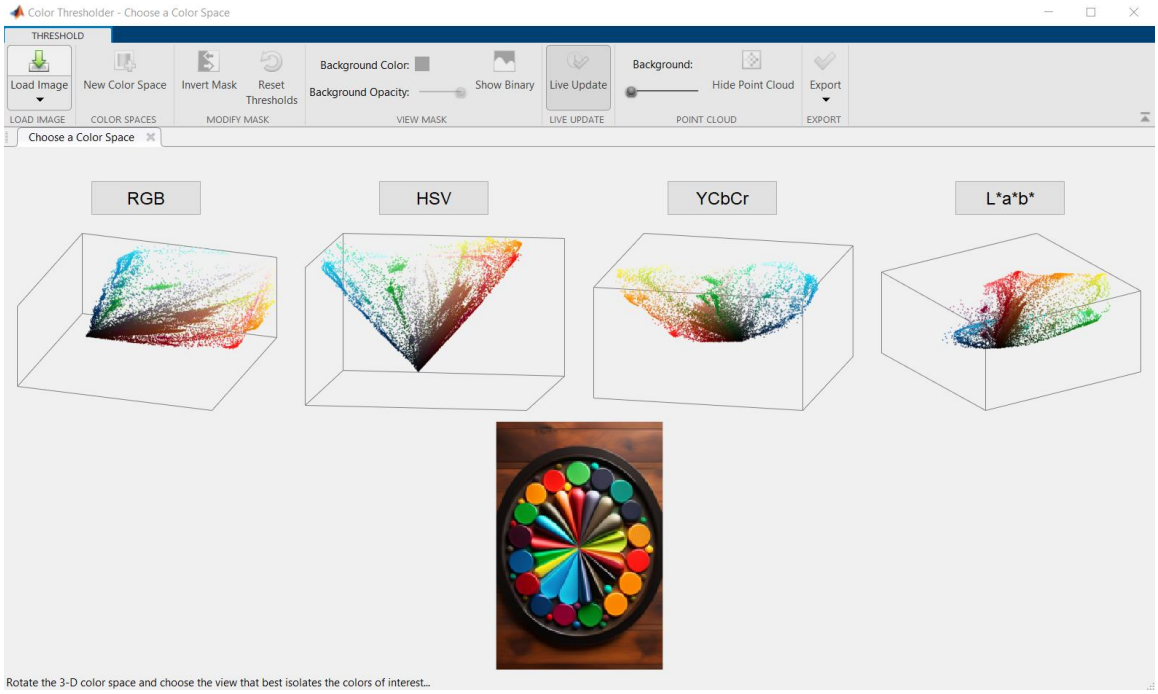


Рис. 8.8 Завантажене зображення

З'являються також, при завантаженні зображення різні кольорові простори, що використовуються для представлення кольору в комп'ютерному зорі та обробці зображень. Кожен з цих просторів має свої особливості і використовується для різних цілей:

RGB - це кольорова модель, в якій всі кольори представляються в поєднанні червоного, синього та зеленого. Ця модель використовується при утворенні зображення на екрані електронного пристрою.

HSV - це циліндрична модель, яка відображає кольори за допомогою трьох параметрів: відтінок (H), насиченість (S) та значення (V).

YCbCr - це простір кольору, що розділяє яскравість (Y) від хроматичних (Cb та Cr) компонентів.

Lab* - це колірний простір, що відображає кольори на основі яскравості (L*) та двох колірних компонентів a* та b*.

Натиснувши на один із підписів кольорового простору з'являється вікно аналізу зображення за даною просторовою моделлю (RGB) (рис.8.9).

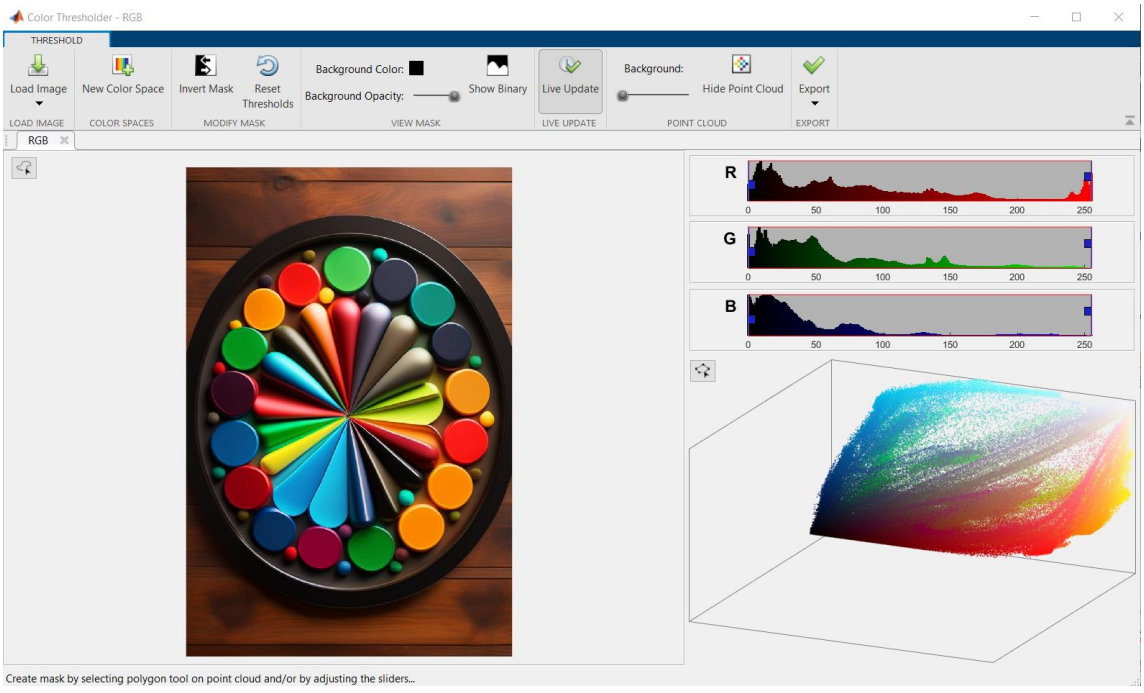


Рис. 8.9 RGB – кольорова модель зображення

За допомогою аналізу даної моделі можна оперувати значеннями при подальшому використанню даного зображення, а саме для відображення зображення на екрані, друку, обробки та аналізу кольорових даних.

8.3. Завдання до виконання комп’ютерного практикуму

Завдання 1. Завантажити заздалегіть надане викладачем зображення, виконати фільтрацію, аналіз зображення та морфологічні операції для покращення якості та виділення деяких об'єктів зображення відповідно до варіанту згідно табл. 8.4.

Завдання 2. Виконати аналіз кольорової моделі попередньо обраного індивідуального зображення (надається викладачем) відповідно до варіанту згідно табл.8.5.

Таблиця 8.4

Варіанти індивідуальних завдань

Варіант	**Розділ фільтра**	**Фільтр**
1	Smoothing Filters	imfilter
2	Sharpening Filters	imsharpen
3	Noise Removal Filters	medfilt2
4	Edge Detection Filters	edge
5	Blur Filters	imgaussfilt
6	Contrast Enhancement Filters	histeq
7	Smoothing Filters	imgaussfilt
8	Sharpening Filters	imfilter
9	Noise Removal Filters	denoiseTV
10	Contrast Enhancement Filters	adapthisteq

Таблиця 8.5

Варіанти індивідуальних завдань

Варіант	Кольоровий простір	Варіант	Кольоровий простір
1	RGB	6	HSV
2	HSV	7	YCbCr
3	YCbCr	8	RGB
4	Lab*	9	HSV
5	RGB	10	Lab*

8.4. Контрольні запитання

1. Які можна визначити можливості Image Processing Toolbox?
2. Який перелік функцій, що доступні в Image Processing Toolbox?
3. Які є фільтри для обробки зображень?
4. Що виконує функція imread?
5. Що виконує функція edge?
6. Які особливості використання функції imfilter?
7. Що виконує інструмент Color Thresholder?
8. Які існують кольорові простори?

КОМП'ЮТЕРНИЙ ПРАКТИКУМ 9
РОЗВ’ЯЗУВАННЯ ЗАДАЧ ЗА ДОПОМОГОЮ КАТАЛОГУ WAVELET TOOLBOX

Мета роботи

Ознайомлення та дослідження можливостей застосування пакету Wavelet Toolbox для розв'язування задач обробки зображень.

9.1. Теоретичні відомості

Цифрова обробка сигналів – це область обчислювальної техніки, що динамічно розвивається і охоплює як технічні, так і програмні засоби [22].

Засоби попередньої обробки призначені для обробки вхідних сигналів, які спостерігаються в загальному випадку на тлі випадкових шумів і перешкод різної фізичної природи і представлених у вигляді дискретних цифрових відліків, з метою виявлення і виділення (селекції) корисного сигналу і оцінки характеристик виявленого сигналу [22].

Аналіз сигналу у часовій області дозволяє досліджувати зміни параметрів сигналу з плином часу. Це означає, що можна визначити, як змінюються значення сигналу відносно часу. Аналіз надає інформацію про еволюцію сигналу та його динаміку.

Математичний опис сигналу носить назву математичної моделі. У цифровому дослідженні сигналів основними математичними інструментами є дискретне перетворення Фур'є і Wavelet-перетворення.

Wavelet - перетворення сигналів є узагальненням спектрального аналізу, типовим представником якого є класичне перетворення Фур'є. Термін «Wavelet» в перекладі з англійської означає «маленька (коротка) хвиля». Wavelet – це узагальнена назва сімейств математичних функцій певної форми, які локальні в часі і по частоті, і в яких всі функції виходять з однієї базової та породжуються за допомогою її зрушень і розтягувань по осі часу [22].

Перетворення такого типу проводить аналіз тимчасових функцій в межах коливань системи, обмежених за часом і частотою.

Зазвичай Wavelet - перетворення (WT) класифікується на два основних типи: дискретне Wavelet - перетворення (DWT) і безперервне Wavelet - перетворення (CWT).

DWT використовується для перетворення та кодування сигналів, CWT – для аналізу сигналів. Wavelet - перетворення в даний час приймаються на озброєння для величезного числа різноманітних застосувань, незмінно заміняючи звичайне перетворення Фур'є [22].

Область застосування Wavelet - перетворення досить різноманітна та включає в себе: сигнальний аналіз, обробка зображень та стиск даних (стиснення даних у певних форматах, такі як звукові файли, зображення та відео).

Дискретне перетворення Фур'є застосовується для розкладання сигналу на частотні компоненти, перетворюючи сигнал з часової області в частотну область, в якій можна визначити частоти, які є складовими сигналу та які інтенсивності мають.

Математично процес аналізу Фур'є представлений перетворенням Фур'є

$$F(w) = \int_{-\infty}^{\infty} f(t)e^{-iwt}dt, \qquad (9.1)$$

яке є сумою по всьому часу сигналу *f(t)*, помноженого на комплексну експоненту. Результатами цього перетворення є коефіцієнти Фур'є *F(w)*, множеннякіх на синусоїду відповідної частоти дасть синусну компоненту вихідного сигналу. Графічно цей процес виглядає як наведено на рис.9.1 [22]:

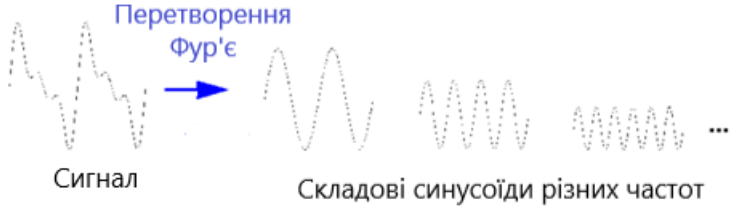


Рис. 9.1 Перетворення Фур'є на синусні компоненти

9.2. Можливості, які надає Toolbox користувачу

Wavelet Toolbox надає програми та функції для частотно-часового аналізу сигналів і багатомасштабного аналізу зображень. За допомогою цього можна зашумлювати та стискати дані, а також виявляти аномалії, точки зміни та перехідні процеси.

Wavelet Toolbox надає можливість виконувати такі завдання:

- Wavelet-декомпозиція;
- Wavelet-реконструкція;
- Wavelet-фільтрація;
- Детекція країв;
- Стиснення даних;
- Апроксимація та розпізнавання.

Основні функції каталогу Wavelet Toolbox наведені в табл.9.1 [22]:

Таблиця 9.1

Функції каталогу Wavelet Toolbox

Функція	Опис функції
wavedec	Декомпозицію сигналу на різних рівнях роздільності
waverec	Реконструкцію сигналу
dwt	Дискретне Wavelet - перетворення (DWT) сигналу
idwt	Інверсне дискретне Wavelet-перетворення для відновлення сигналу
cwt	Безперервне Wavelet-перетворення (CWT) сигналу, отримуючи спектрограму залежності амплітуди від часу та частоти
icwt	Інверсне безперервне Wavelet - перетворення для відновлення сигналу з його спектрограми
wdenoise	Wavelet - фільтри для шумопониження сигналу
wthresh	Порогову фільтрацію сигналу для видалення менш значущих компонент

Функція	Опис функції
wrcoef	Отримання окремих деталей або апроксимацій сигналу на певному рівні декомпозиції
wenergy	Обчислення енергії сигналу у кожному рівні декомпозиції для аналізу його частотних компонент
wscalogram	Wavelet - шкалограма для аналізу часово-частотних характеристик сигналу
wvarchg	Зміну варіації для виявлення змін в сигналі
wdencmp	Декомпозиції сигналу для подальшого відновлення з використанням порогової фільтрації

Також є можливість вирішення задач за допомогою програмного графічного інтерфейсу Wavelet Signal Denoiser, знайшовши його в вкладенні «APPS» в підрозділі Wavelet Toolbox та натиснувши на відповідну піктограму з'являється графічне вікно (рис.9.2).

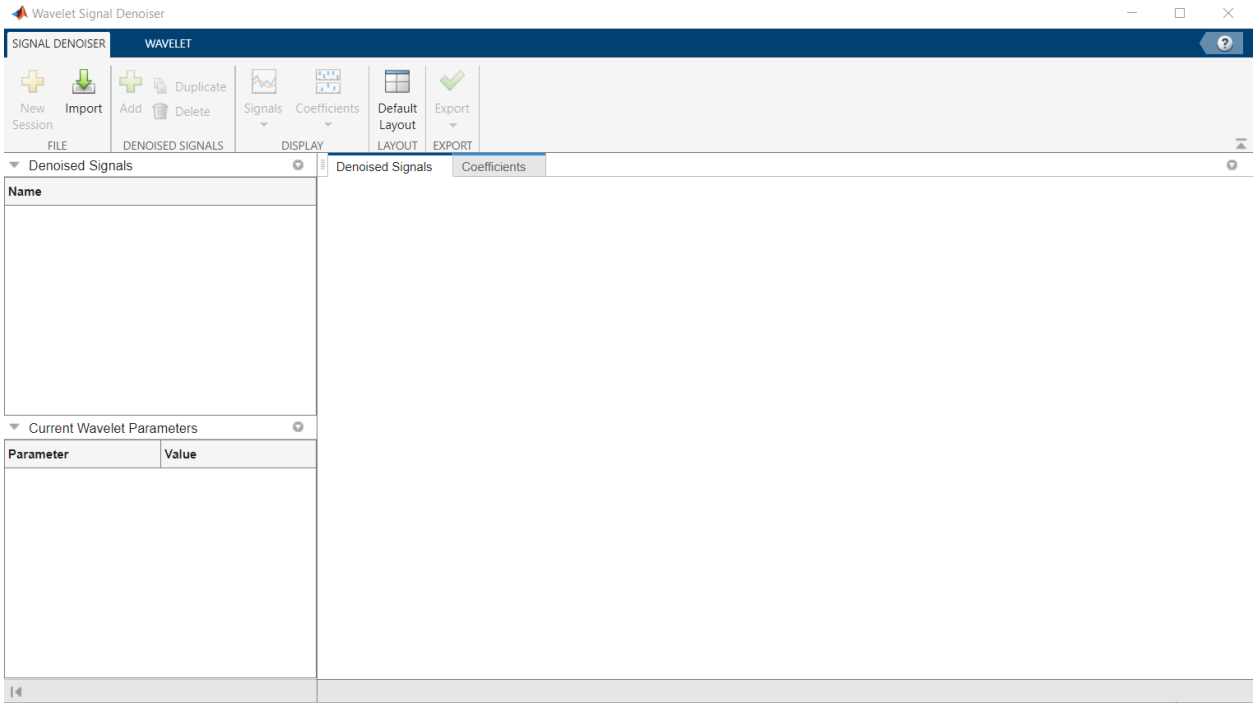


Рис. 9.2 Графічне меню Wavelet Signal Denoiser

Для початку роботи з інтерфейсом Wavelet Signal Denoiser потрібно імпортувати змінну workspace (рис.9.3), тобто графік сигналу для проведення знешумлення сигналу. Створивши сигнал з шумом в графічному вікні програмного інтерфейсу з'являється попердньо імпортований сигнал (denoised signals) (рис.9.4) та коефіцієнти (coefficients) (рис.9.5).

Код виконання програми:

```
clc, clear
% Створення часового вектору
t = 0:0.01:5;

% Створення чистого сигналу
clean_signal = sin(2*pi*t);

% Створення шуму
noise = 0.2 * randn(size(t));

% Додавання шуму до сигналу
noisy_signal = clean_signal + noise;

% Відображення шумного сигналу
plot(t, noisy_signal);
title('Сигнал з шумом');
```

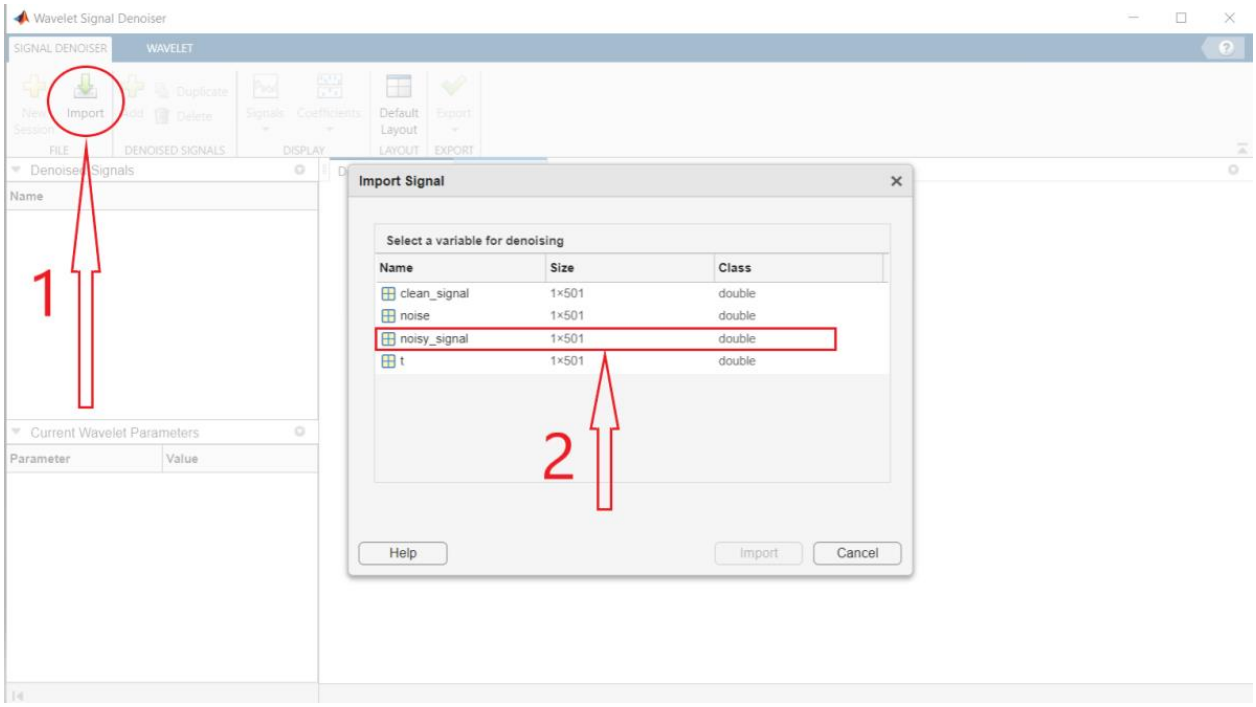


Рис. 9.3 Імпортування сигналу в графічний додаток Wavelet Signal Denoiser

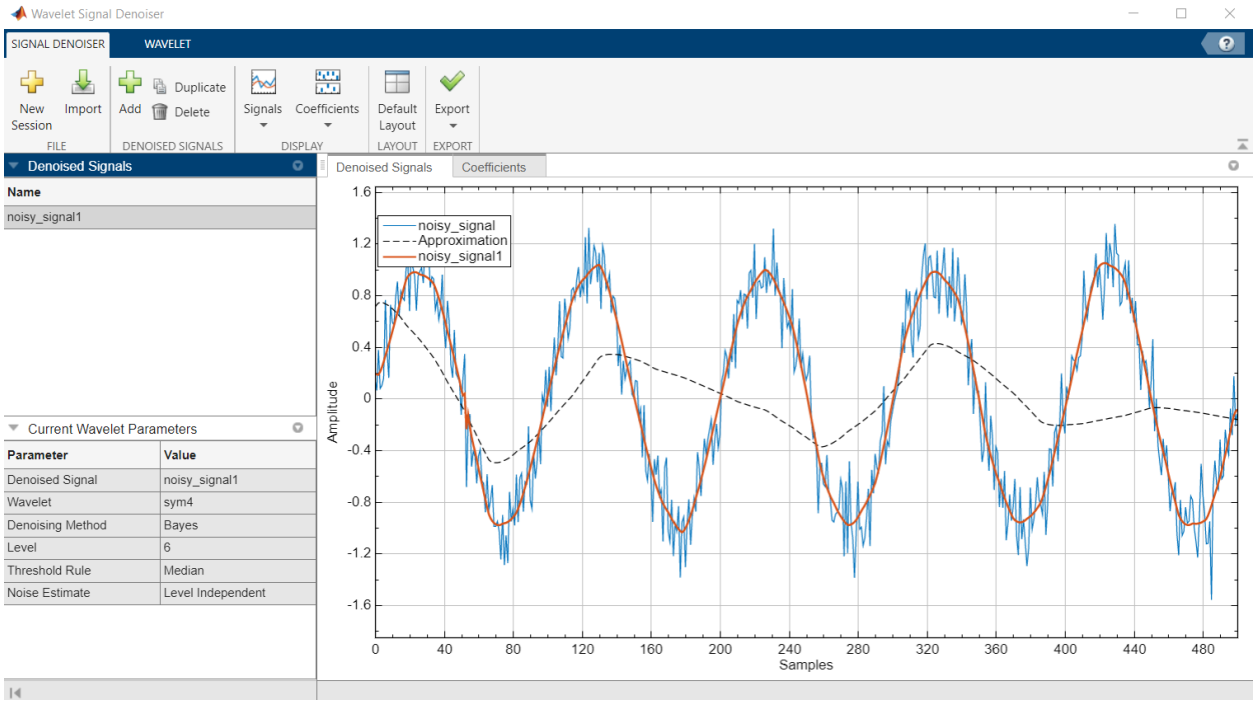


Рис. 9.4 Імпортований сигнал

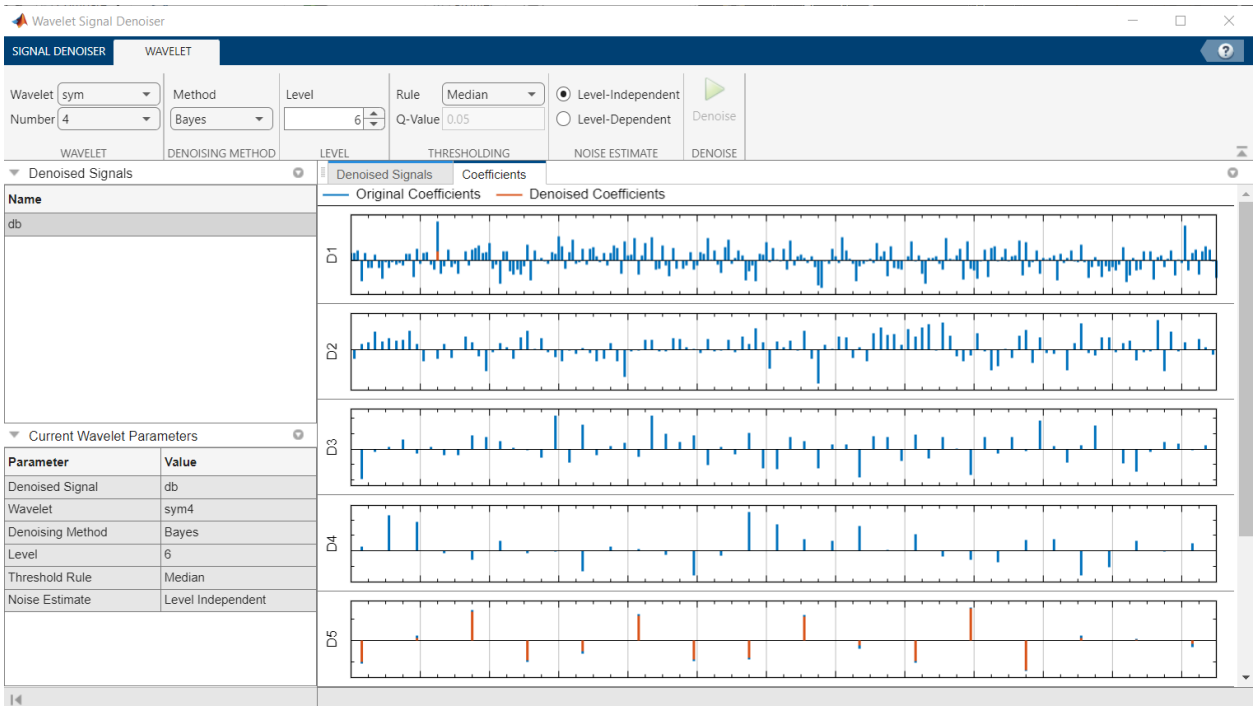


Рис. 9.5 Коефіцієнти сигналу

Програмний інтерфейс Wavelet Signal Denoiser включає різноманітні такі функції та команди (рис.9.6):

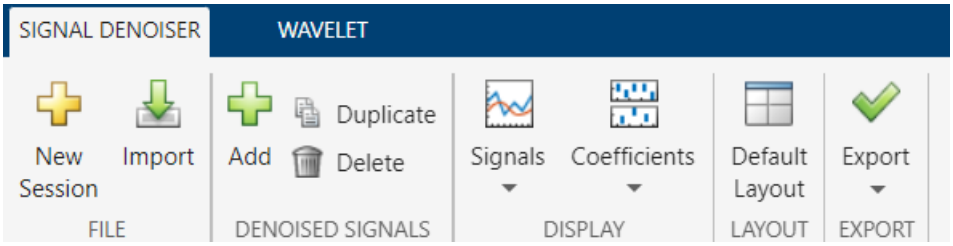


Рис. 9.6 Головне меню інтерфейсу

- **New Session** – створення нового аналізу;
- **Import** – імпортування вхідного сигналу для аналізу;
- **Add** – додати новий вхідний до існуючого;
- **Signals** – відображення ампроксимованого та оригінального вхідного сигналу (імпортований сигнал);
- **Coefficients** – відображення Wavelet-коефіцієнтів на графіках сигналів з шумом та знешумлено;
- **Deffult Layout** – компонування за замовчуванням;
- **Export** – експорт файлу до workspace або вигляді окремого коду.

Програмний інтерфейс Wavelet Signal Denoiser автоматично виконує операції з вхідним сигналом та будує в графічному вікні графік апроксимації та знешумленого сигналу.

Натиснувши на вкладення Wavelet з'являється меню налаштувань Wavelet Signal Denoiser (рис.9.7).

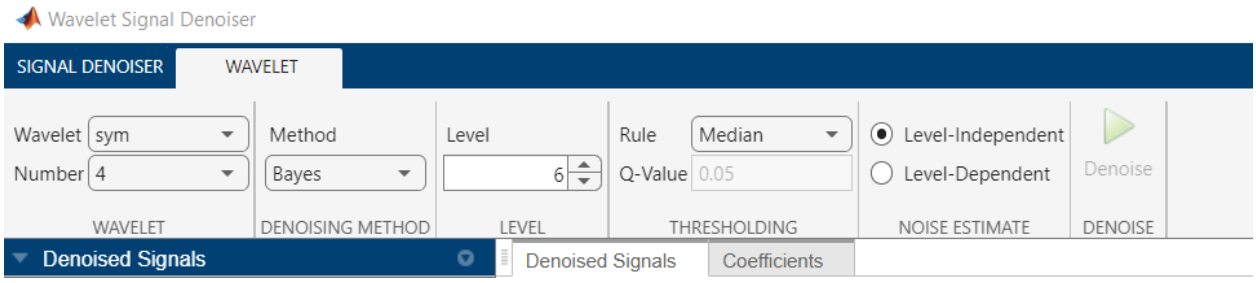


Рис. 9.7 Меню вкладення Wavelet

В графічному вікні Wavelet розміщенні наступні параметри налаштувань:

- **Wavelet** - вибір типу вейвлету, який буде використовуватися для шумопониження;

- **Number** (Рівень шуму) - параметр, який дозволяє вказати рівень шуму у вхідному сигналі;
- **Method** (Метод порогування) - вибір методу;
- **Level** - параметр, який визначає порогове значення для методу порогування. Це важливий параметр, який впливає на ефективність шумопониження;
- **Rule** - вибір порогового значення для методу.

У випадному списку **Wavelet** знаходяться наступні вейвлейти:

- Sym - гладкий вейвлейт з хорошою кількістю нулів;
- Db - ортогональний вейвлейт зі скінченною довжиною фільтра;
- Fk - гладкий вейвлейт, який базується на апроксимаційних властивостях Фейєра та Коровкіна;
- Bior - забезпечує більш гнучкий контроль над збереженням сигналу та шумопониженням;
- Coif - вейвлет є змінним вейвлетом, який пропонує більш різноманітний спектр коефіцієнтів.

У випадному списку **Method** знаходяться наступні методи порогування:

- Bayes - статистична модель для визначення оптимального порогового значення для шумопониження;
- BlockJs - блочну структуру для порогового рівняння;
- FDR - принцип контролю кількості помилок типу I (фальшивих відкриттів) при відновленні сигналу;
- Minimax - критерій максимального ризику для вибору порогового значення;
- SURE - оцінка безперехресного ризику Стейна для вибору оптимального порогового значення;
- Universal Threshold - універсальне порогове значення, яке залежить від статистичних властивостей шуму та сигналу.

9.3. Завдання до виконання комп'ютерного практикуму

Завдання 1. Використовуючи програмний інтерфейс Wavelet Signal Denoiser, провести знешумлення сигналу, (синусоїдальну функцію для вхідного сигналу взяти відповідно до варіанту згідно табл.7.2) налаштувавши параметри вкладення Wavelet відповідно до варіанту згідно табл.9.2.

Таблиця 9.2

Варіанти індивідуальних завдань

Варіант	Wavelet	Number	Method	Level
1	Sym	2	Bayes	1
2	Db	3	BlockJs	2
3	Coif	4	FDR	3
4	Bior	5	Minimax	4
5	Fk	2	SURE	5
6	Sym	3	BlockJs	6
7	Db	6	FDR	1
8	Fk	7	Bayes	2
9	Bior	8	Minimax	7
10	Coif	5	SURE	8

9.4. Контрольні запитання

1. Що таке цифрова обробка сигналів?
2. Які існують основні математичні інструменти?
3. Чим характеризується дискретне перетворення Фур'є?
4. Що таке Wavelet-перетворення?
5. Які можливості Toolbox є основними?
6. У чому особливості роботи з інтерфейсом Wavelet Signal Denoiser?
7. Які існують параметри налаштувань Wavelet?

ВИСНОВКИ

У даній дипломній роботі було розроблено методичне та алгоритмічне забезпечення для дисципліни «Інтегровані пакети прикладних програм». Головною метою було ознайомлення з пакетом MatLab та моделюванням динамічних систем з використанням каталогів Toolbox і пакета SimuLink. Загалом, викорстання цих додатків у MatLab надає зручні та потужні засоби для моделювання динамічних систем, що дозволяють зрозуміти їхню поведінку та здійснити різноманітний аналіз, що може бути корисним для розроблення та оптимізації рішень в різних галузях. MatLab є потужним інструментом для розв'язання складних завдань у сфері інформаційних технологій. MatLab та SimuLink допомагають виконувати різні функції і оптимізувати робочі процеси, що надає змогу користувачам зручно та ефективно працювати з різними типами даних та задачами.

У першому комп'ютерному практикумі було проведено ознайомлення з пакетом MatLab, його основними можливостями та інтерфейсом, також розглянуто різні функції та команди MatLab, що дозволяють працювати з числовими даними та графічними зображеннями.

У другому комп'ютерному практикумі було розглянуто процес моделювання динамічних систем з використанням пакета SimuLink. Було створено прості S-моделі з використанням розділів Sources та Sinks, що дозволяє задавати вхідні сигнали для системи та відображати їх вихідні дані. Розглянуто різні типи вхідних сигналів та їх вплив на поведінку системи.

У третьому комп'ютерному практикумі було проведено ознайомлення з розділом Math Operation пакета SimuLink для створення S-моделей. Головною метою якого є ознайомлення з функціями та операціями, доступними в цьому розділі, а також навчання їх використанню для моделювання та обробки сигналів у системах. Виконання даного практикуму поглибить розуміння принципів роботи розділу Math Operation та розвитку навичок створення S-

моделей із використанням математичних операцій. Отримані знання та навички можуть бути корисними в подальшій роботі з моделюванням динамічних систем та обробкою сигналів.

У четверному комп'ютерному практикумі було проведене ознайомлення з розділом Continuous пакета SimuLink для створення S-моделей динамічних систем. Розглянуті основні блоки та функціонал розділу. Використання різних блоків та функціоналу дозволяє створювати складні моделі та досліджувати їх поведінку в різних умовах. Це дозволяє вирішувати задачі контролю, оптимізації та аналізу систем з неперервними динаміками.

У п'ятому комп'ютерному практикумі було проведено ознайомлення та дослідження каталогу Symbolic Math Toolbox в процесі розв'язування рівнянь у символьному вигляді в системі MatLab. Головною метою є ознайомлення з функціями та можливостями символьного розв'язування рівнянь та їх використання для аналізу та моделювання систем. Використання символьних методів надає змогу користувачу здобувати точні аналітичні розв'язки рівнянь, що дозволяє зробити більш детальний аналіз та прогнозування поведінки системи.

У шостому комп'ютерному практикумі було проведено ознайомлення з каталогом Control System Toolbox в пакеті MatLab для розв'язування задач з теорії автоматичного керування. Розглянуто також можливості, які надає Toolbox користувачу, а саме функції для визначення та розрахунку вагової та перехідної характеристик, вивчено різні функції та методи пакету, включаючи створення передаточних функцій, розв'язання рівнянь стану та проведення аналізу стійкості систем.

У сьомому комп'ютерному практикумі було розглянуто використанням каталогу Signal Processing Toolbox. Під час роботи особлива увага була звернена на поняття електричного сигналу та типів, на які він поділяється, вивчено різні функції каталогу, включаючи фільтрацію сигналів, спектральний аналіз, кореляцію, дискретне перетворення Фур'є та багато

інших. Було досліджено вплив різних параметрів та налаштувань на результати обробки сигналів.

У восьмому комп'ютерному практикумі було проведено ознайомлення з каталогом Image Processing Toolbox для розв'язування задач обробки зображень. Наведено теоретичні відомості щодо методів обробки сигналів та їх значення у світі сучасних технологій, вивчено різні функції та методи каталогу Image Processing Toolbox, включаючи фільтрацію, сегментацію, морфологічну обробку, видалення шуму, виявлення контурів та багато інших. А також досліджено вплив різних параметрів та налаштувань на результати обробки зображень.

У дев'ятому комп'ютерному практикумі було проведено ознайомлення з каталогом Wavelet Toolbox при розв'язуванні задач з вейвлет-аналізу. Було розглянуто питання цифрової обробки сигналів та аналізу вихідних сигналів, включаючи дискретні вейвлет-перетворення, зворотні перетворення, вейвлет-фільтри, компресію сигналів та зображень, детекцію країв та багато інших. Досліджено вплив різних параметрів та налаштувань на результати вейвлет-аналізу. Використання функцій та методів пакету дозволяє проводити якісний аналіз сигналів та зображень за допомогою вейвлет-перетворень, виконувати компресію для зменшення розміру сигналів та зображень з мінімальною втратою інформації.

Список використаних джерел

1. Довідник з MATLAB / Електронний навчальний посібник з курсового і дипломного проектування. – К.: НТУУ "КПІ", 2013. – 132 с.

2. Комп’ютерне моделювання процесів і систем. Практикум [Електронний ресурс] : навч. посіб. для студ. спеціальності 151 «Автоматизація та комп’ютерно-інтегровані технології» / Д.О. Півторак, Ю.Ф. Лазарєв, С.Л. Лакоза ; КПІ ім. Ігоря Сікорського, 2020. - 207 с.

3. Методичні вказівки до лабораторної роботи на тему: «Вступ до системи MATLAB» [Електронний ресурс]. — Режим доступа: https://ua.kursoviks.com.ua/metodychki/5021-metodichni-vkazivki-do-laboratornoi-roboti-na-temu-vstup-do-sistemi-MatLab

4. Інформаційні технології-2: конспект лекцій [Електронний ресурс]: навч. посіб. для студ. спеціальності 171 «Електроніка», освітньої програми «Електронні системи» / КПІ ім. Ігоря Сікорського ; уклад.: К. С. Клен. – Електронні текстові данні (1 файл: 6,045 Мбайт). – Київ: КПІ ім. Ігоря Сікорського, 2019. – 154 с

5. Побудова графіків MatLab [Електронний ресурс]. — Режим доступа: https://jak.koshachek.com/articles/pobudova-grafikiv-MatLab.html

6. SimuLink [Електронний ресурс]. — Режим доступа: http://inmad.vntu.edu.ua/portal/static/F2E71C59-8441-47BF-BD35-4D7503890976.pdf

7. Математичне моделювання на ЕОМ (Частина 2) [Електронний ресурс] : методичні вказівки до виконання комп’ютерних практикумів для студентів напряму підготовки 6.051003 «Приладобудування» / НТУУ «КПІ» ; уклад.: Ю. Ф. Лазарєв, Д. О. Півторак, С. Л. Лакоза. – Електронні текстові данні (1 файл: 1,08 Мбайт). – Київ : НТУУ «КПІ», 2015. – 86 с.

8. Sinks – MathWorks [Електронний ресурс]. — Режим доступа: https://www.mathworks.com/help/SimuLink/sinks.html?s_tid=CRUX_lftnav

9. Sources – MathWorks [Електронний ресурс]. — Режим доступа: https://www.mathworks.com/help/SimuLink/sources.html?s_tid=CRUX_lftnav

10. Розділ Math (математичні блоки) [Електронний ресурс]. — Режим доступа: http://um.co.ua/1/1-1/1-105314.html

11. Math Operations – MathWorks [Електронний ресурс]. — Режим доступа: https://www.mathworks.com/help/SimuLink/math-operations.html?s_tid=CRUX_lftnav

12. Continuous – MathWorks [Електронний ресурс]. — Режим доступа: https://www.mathworks.com/help/SimuLink/continuous.html?s_tid=CRUX_lftnav

13. State-Space – MathWorks [Електронний ресурс]. — Режим доступа: https://www.mathworks.com/help/SimuLink/slref/statespace.html

14. Теорія автоматичного управління. Практикум [Електронний ресурс] : навч. посіб. для студ. спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» / Н.І. Бурау, Д.О. Півторак; КПІ ім. Ігоря Сікорського, 2021. - 57 с.

15. Теорія автоматичного управління. Теорія лінійних систем автоматичного управління. Лабораторний практикум [Електронний ресурс] : навч. посіб. для студ. спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» / С.А. Мураховський, Д.О. Півторак; КПІ ім. Ігоря Сікорського, 2022. - 94 с.

16. Subcategories (Control System Toolbox) – MathWorks [Електронний ресурс]. — Режим доступа: https://www.mathworks.com/help/control/index.html?s_tid=hc_product_card

17. Тема 1. Основні поняття теорії сигналів [Електронний ресурс]. — Режим доступа: https://org2.knuba.edu.ua/mod/book/tool/print/index.php?id=23040

18. Спектральний аналіз [Електронний ресурс]. — Режим доступа: https://kivra.kpi.ua/wp-content/uploads/file/discipline/AOTI/AOTI_2_2.pdf

19. Основи теорії цифрової фільтрації [Електронний ресурс]. — Режим доступа: https://stud.com.ua/171398/tehnika/osnovi_teoriyi_tsifrovoyi_filtratsiyi

20. Компютерне моделювання систем та процесів. Методи обчислень. Частина 1 : навчальний посібник / Р.Н. Квєтний, І.В. Богач, О.Р. Бойко та ін. – Вінниця: ВНТУ, 2013. – 191 с.

21. Reference List- MATLAB & SimuLink - MathWorks [Електронний ресурс]. — Режим доступа: https://www.mathworks.com/help/images/referencelist.html?type=function&listtype=cat&category=index&blocktype=all&capability=&s_tid=CRUX_topnav

22. Цифрова обробка сигналів [Електронний ресурк]: навч. посіб. для студ. спеціальності 151 «Автоматизація та комп’ютерно-інтегровані технології» / М.О. Рибальченко, О.П. Єгоров, В.Б.Зворикін. - Дніпро: НМетАУ 2018. - 79 с.