МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ АВТОМОБІЛЬНО-ДОРОЖНІЙ УНІВЕРСИТЕТ

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних та самостійних робіт
з дисципліни «Основи програмування»
для студентів спеціальностей
132 «Матеріалознавство», 133 «Галузеве машинобудування»
за освітньо-кваліфікаційним рівнем бакалавра.
Розділ «Інтегрований пакет математичного моделювання Matlab»

Затверджено
методичною радою університету
протокол № 3 від 23 листопада 2022

Харків
ХНАДУ
2023

Укладачі: Костікова М. В.,
Шевченко В. О.

Кафедра інформатики та прикладної математики

ВСТУП

Matlab – одна з найстаріших, ретельно пророблених і перевірених часом систем автоматизації математичних розрахунків, побудована на розширеному представленні і застосуванні матричних операцій.

Однією із основних задач системи *Matlab* завжди було надання користувачам потужної мови програмування, орієнтованої на технічні і математичні розрахунки і здатної перевершити можливості традиційних мов програмування, котрі багато років використовувались для реалізації чисельних методів.

Сучасні системи автоматичного керування характеризуються високою складністю, підвищеними вимогами до показників швидкості, точності та стійкості. Для моделювання і проектування таких систем використовуються пакети програм, що дають можливість проводити моделювання аналіз і синтез систем керування. Одним із програмних комплексів призначених для моделювання систем керування є комплекс *Matlab*.

Початкові знання по роботі з пакетом *Matlab* можна отримати, якщо опрацювати теми, які надані у цих методичних вказівках. Тут розглянуто основи практичної роботи в середовищі системи комп'ютерної математики *Matlab*.

Дані методичні вказівки містять опис лабораторних робіт з дисципліни «Основи програмування». Їх матеріал відповідає робочій навчальній програмі з вказаної дисципліни для студентів спеціальностей 132 «Матеріалознавство», 133 «Галузеве машинобудування» за освітньо-кваліфікаційним рівнем бакалавра. Всі роботи методичних вказівок подано в єдиному форматі. Кожна лабораторна робота включає короткі теоретичні відомості для виконання роботи, перелік та порядок виконання робочих завдань і контрольні запитання, що можуть бути корисними як для самоперевірки, так і для контролю викладачем. Виконання завдань лабораторних робіт контролює викладач, який проводить заняття. Кожна з робіт розрахована на виконання за 4 академічні години в комп'ютерному класі. Методика їх виконання передбачає етап підготовки напередодні заняття. У ході підготовки потрібно вивчити теоретичні відомості, загальні рекомендації, скласти приблизний план дій з реалізації виконання кожного пункту завдан-

ня, включивши в нього необхідні команди і дані для роботи, відповісти на контрольні запитання. Виконання роботи здійснюється по пунктах, вказаних в завданні. Наприкінці кожного заняття студент повинен надати викладачу звіт з опрацьованої теми, що містить: номер лабораторної роботи, назву теми, мету роботи, контрольні запитання, відповіді на них та, при необхідності, тексти створених програм, додаткові записи та пред'явити виконані завдання на ПК.

ЛАБОРАТОРНА РОБОТА № 1
ОПЕРАЦІЇ З ЧИСЛАМИ, ВЕКТОРАМИ І МАТРИЦЯМИ В *MATLAB*

Мета роботи – знайомство з системою *Matlab*, запуск *Matlab*, робота в режимі діалогу з числами, векторами і матрицями.

Теоретичні відомості

Пакет *Matlab* був створений компанією *Math Works* більше десяти років тому. Система *Matlab* побудована на розширеному поданні та застосуванні матричних операцій. Це знайшло відображення і в самій назві системи – *MATrix LABoratory*, тобто матрична лабораторія. Однак синтаксис мови програмування системи продуманий настільки ретельно, що дана орієнтація майже не відчувається тими користувачами, яких не цікавлять безпосередньо матричні обчислення.

Незважаючи на те що спочатку *Matlab* призначалася виключно для обчислень, в процесі еволюції в *Matlab* з'явилися бібліотеки, які забезпечують унікальні для математичних пакетів функції. В системі *Matlab* також існують широкі можливості для програмування на мові *C*. Усередині пакета можна використовувати як процедури самої *MatLab*, так і стандартні процедури мови *C*, що робить цей пакет найпотужнішою підмогою при розробці додатків.

Таким чином, *Matlab* давно вже вийшла за рамки спеціалізованої матричної системи, перетворившись в одну з найбільш потужних універсальних інтегрованих систем комп'ютерної математики. Сьогодні система *Matlab* широко використовується в техніці, науці та освіті, але все-таки вона більше підходить для аналізу даних і ор-

ганізації обчислень, ніж для чисто математичних викладок. Проте, математичний пакет *Matlab* – інтелектуальний лідери в своєму класі, який визначає розвиток комп'ютерної математики.

Інтерфейс програми Matlab

Стандартний вид вікна *Matlab* складається з п'яти основних частин:

- командне вікно ***Command Window*** – область введення команд і виведення отриманих результатів;
- робоча область ***Workspace*** – область відображення всіх змінних, використовуваних Matlab в поточному сеансі роботи;
- вікно історії команд ***Command History*** – перелік команд, що виконувалися раніш;.
- вікно ***Current Folder*** – в ньому вказується папка, яка буде за замовчуванням використовуватися для всіх операцій з файлами;
- вікно ***Details*** – область відображення відомостей про вибрані файли.

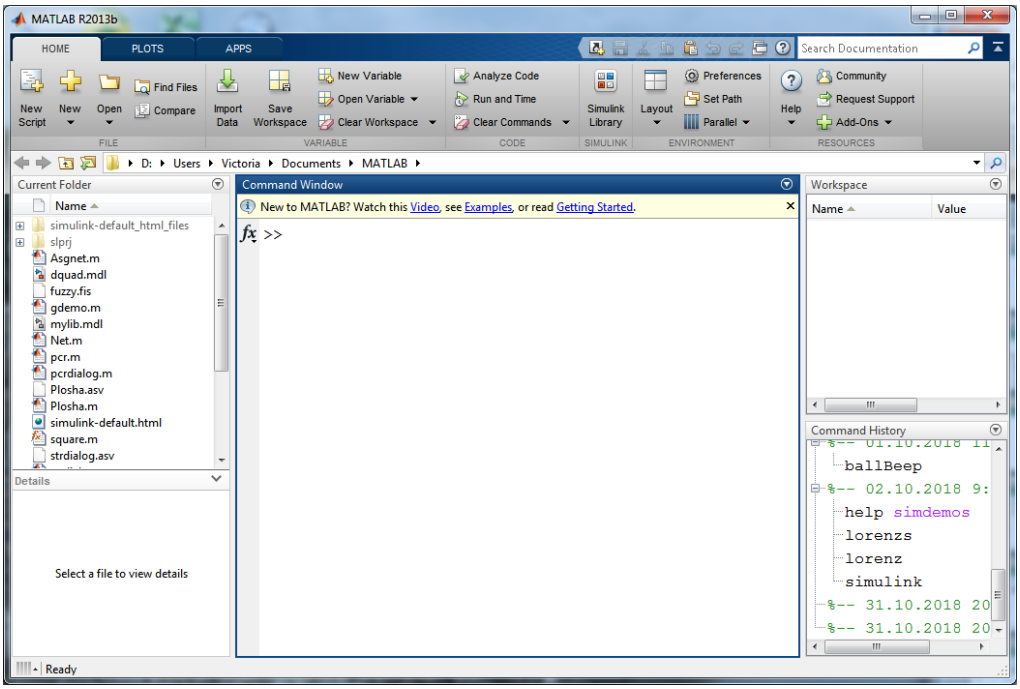


Рисунок 1.1 – Вид вікна Matlab

Вигляд вікна можна змінювати за допомогою команди ***Layout*** на вкладці ***Home***, а також використовуючи стандартні можливості управління вікнами.

Вид команд, що вводяться в командному вікні, досить нескладний. Наприклад, якщо ввести

x=2*pi

то на екран виводиться наступний результат:

x =

6.2832

При цьому виконана команда відображається у вікні історії команд, а в робочій області створюється змінна ***x***.

Якщо команда завершується крапкою з комою, то її результат не виводиться на екран. Наприклад, якщо ввести

x=2*pi;

то змінна *x* отримає таке ж значення, що і при виконанні команди **x=2*pi** (без крапки з комою), але результат не буде виведений на екран.

Якщо при виконанні команди не вказана змінна, якій повинен бути привласнений результат, то він за замовчуванням присвоюється змінній ***ans***.

Наприклад, якщо ввести

2*pi

то на екран виводиться наступний результат:

ans =

6.2832

Команди, що вводяться в командному вікні, відображаються у вікні історії команд.

Якщо при виконанні команди створюється нова змінна, то вона відображається в робочій області.

Нову команду в командному вікні можна вводити тільки в рядку введення, в якій знаходиться курсор. До цього рядка можна копіювати команди, виконані раніше, використовуючи звичайні способи копіювання. Крім того, команди, виконані раніше, можна викликати в рядок введення з вікна історії команд (подвійним клацанням або перетягуванням за допомогою миші), а також натисканням клавіші «Стрілка вгору».

Зі змінними можна працювати в робочій області. Подвійне клацання по імені змінної в робочій області викликає редактор у вигляді електронної таблиці, в якому можна змінювати значення змінної, будувати графіки (якщо змінна представляє собою матрицю) і

виконувати ряд інших операцій.

Крім того, для змінних в робочій області можна використовувати команди контекстного меню, що викликається правою кнопкою миші.

Імена змінних можуть починатися як з великої, так і з малої літери. Слід звернути увагу, що великі і малі літери в іменах змінних різняться. Наприклад, змінні з іменами ***x*** і ***X*** – різні.

Імена функцій (наприклад, **sin**), стандартних констант (наприклад, ***pi***) і інші зарезервовані слова вказуються малими літерами.

Представлення даних в *Matlab*

Основний тип даних, що використовуються в *Matlab* – дійсні числа (тип ***double***). Для управління форматом представлення даних на екрані використовується команда ***format***. Основні формати виведення даних на екран наступні: ***format short*** – вивід з точністю до чотирьох значущих цифр після коми (використовується за замовчанням); ***format long*** – повне представлення числа; ***format short e***, ***format long e*** – те ж, що ***format short*** і ***format long***, але дані виводяться в експоненційній формі з мантисою і порядком; ***format rat*** – вивід даних у вигляді раціонального дробу.

Обраний формат застосовується до тих пір, поки не буде введена команда ***format*** з іншим форматом.

Всі дані в *Matlab* розглядаються як матриці. Навіть звичайна змінна є матрицею розміром 1x1. Для одновимірних матриць (рядків і стовпців) застосовується також термін «масив».

У *Matlab* є зручні можливості для операцій з окремими елементами матриць, а також їх рядками і стовпцями, включаючи перегляд, додавання, видалення і т. д. Такі операції можуть виконуватися з окремими рядками і стовпцями, діапазонами і матрицями в цілому.

Алфавіт мови *Matlab*

В *Matlab* використовуються всі букви латинського алфавіту від *A* до *Z* і арабські цифри від 0 до 9, причому великі і малі літери – це різні змінні і константи. Крім букв і цифр використовуються всі

спеціальні символи клавіатури комп’ютера.

Таблиця 1.1 – **Арифметичні оператори**

Функція	Позначення (синтаксис)
Додавання	+ ($M1+M2$)
Віднімання	– ($M1–M2$)
Матричне множення	∗ ($M1∗M2$)
Поелементне множення масивів	.∗ ($M1.∗M2$)
Піднесення матриці до степеня	∧ ($M1∧ x$)
Поелементне піднесення масиву до степеня	.∧ ($M1.∧ x$)
Ділення матриць зліва направо	/ ($M1$ / $M2$)
Поелементне ділення масивів зліва направо	./ ($M1$ ./ $M2$)
Ділення матриць справа наліво	\ ($M1$ \ $M2$)
Поелементне ділення масивів справа наліво	.\ ($M1$.\ $M2$)

Таблиця 1.2 – **Елементарні математичні функції**

Математичний запис	Запис в *Matlab*
$\lvert x \rvert$ – модуль числа	abs(x)
e^x – експонента числа	exp(x)
ln x– натуральний логарифм	log(x)
lg x– десятковий логарифм	log10(x)
$\sqrt{x}$ – квадратний корінь	sqrt(x)
$\sqrt[n]{x}$ – корінь ступеня n	x^(1/n)
arccos x – арккосинус	acos(x)
arcctg x – арккотангенс	acot(x)
arcsin x – арксинус	asin(x)
arctg x – арктангенс	atan(x)
cos x – косинус	cos(x)
ctg x – котангенс	cot(x)
sin x – синус	sin(x)
tg x – тангенс	tan(x)

Пріоритет операцій

У математичних виразах оператори мають певний пріоритет виконання:

1. Математичні функції.
2. Логічні оператори.

3. Піднесення до степеня.

4. Множення і ділення.

5. Додавання і віднімання.

Для зміни пріоритету операцій потрібно використовувати круглі дужки.

Приклади.

14/5*2=5.6 – операції одного пріоритету виконуються зліва направо;

27^1/3=9 – піднесення до степеня має найвищий пріоритет;

27^(1/3)=3 – дужки змінюють послідовність операцій.

Робочі завдання

Завдання 1. Робота в режимі калькулятора.

Робота в середовищі *Matlab* може здійснюватися в двох режимах:

1) у режимі калькулятора, коли обчислення виконуються безпосередньо після набору чергового оператора або команди MATLAB; при цьому значення результатів обчислення можна присвоїти деяким змінним або результати видаються безпосередньо на екран без присвоювання (як у звичайних калькуляторах);

2) шляхом виклику програми, складеної і записаної мовою MATLAB, що містить усі необхідні команди, які забезпечують введення даних, організацію обчислень і виведення результатів на екран (програмний режим).

В обох режимах користувачу доступні практично всі обчислювальні можливості системи, у тому числі з виведення інформації в графічній формі. Програмний режим дозволяє зберігати розроблені обчислювальні алгоритми і, таким чином, повторювати обчислення з іншими початковими даними.

Особливістю *Matlab* як калькулятора є можливість використання імен змінних для запису проміжних результатів у пам’ять комп’ютера. Для цього застосовується операція присвоювання, що вводиться знаком рівності (=) відповідно до схеми <ім’я змінної> = <вираз> [;].

Ім’я змінної може мати до 19 символів і не повинне збігатися з іменами функцій і процедур системи і системних змінних. При

цьому система розрізняє в змінних великі та малі літери. Так, імена *amenu*, *Amenu*, *aMenu* у *Matlab* позначають різні змінні.

Вираз праворуч від знака присвоювання може бути просто числом, арифметичним виразом, рядком символів (тоді ці символи потрібно обмежити апострофами) або символьним виразом. Якщо вираз не закінчується символом «;», то після натискання клавіші *Enter* у командному вікні з'явиться результат виконання у вигляді: <ім'я змінної> = <результат>

Обчислити вираз (25+17)*7. Це можна зробити так. Спочатку ввести послідовність 25+17 і натиснути клавішу *Enter*. Результат буде представлений у вигляді: *ans* = 42. Далі ввести послідовність *ans**7 і натиснути *Enter*. Після цього з'явиться наступний результат: *ans* = 294.

```
>> 25+17

ans =

    42

>> ans*7

ans =

   294
```

Щоб не видавати проміжний результат обчислення 25+17, досить після запису цього виразу додати символ «крапка з комою». Тоді результат буде мати такий вигляд:

```
>> 25+17;
>> ans*7

ans =

   294
```

Завдання 2. Обчислити значення арифметичних виразів:

$2+3\cdot\sin w^2$, при $w=0{,}7$;

$\pi-\cos^2 d$, при $d=-0{,}3$;

$$\frac{\sqrt{k1+k2}}{\left|s-e^{s}\right|}, \qquad \text{при } k1=4{,}63;\ k2=0{,}71;\ s=4;$$

$$\lg\frac{2{,}5\cdot b7}{1{,}7}+\sqrt[5]{4}, \qquad \text{при } b7=0{,}5;$$

$$cz-\mathrm{tg}^{3}(q-x7)^{2}, \qquad \text{при } cz=0{,}7;\ q=0{,}2;\ x7=-0{,}8.$$

Завдання 3. Введення векторів.

Вектор в *Matlab* – это одномерный числовой массив. Вихідні значення векторів можна задавати з клавіатури шляхом поелементного введення. Для цього в рядку треба спочатку вказати ім'я вектора, потім поставити знак присвоювання =, потім ліву квадратну дужку, а за нею ввести задані значення елементів вектора, відокремлюючи їх одне від одного пробілами або комами. Завершується рядок правою квадратною дужкою. Наприклад, введення рядка *V* = [1.2 -0.3 1.2e-5] задає вектор *V*, що містить три елементи зі значеннями 1.2, -0.3, 1.2e-5:

```
>> V=[1.2 -0.3 1.2e-5]

V =

    1.2000   -0.3000    0.0000
```

Після введення вектора система виводить його на екран. Те, що в наведеному прикладі останній елемент виведений як нуль, зумовлено встановленим форматом *Short*, відповідно до якого дані виводяться на екран. Якщо ви встановите довгий формат виведення даних, можна побачити фактичне значення останнього елемента:

```
>> format long
>> V

V =

   1.200000000000000  -0.300000000000000   0.000012000000000
```

Однак для відображення значень вектора *V* найбільше підходить короткий формат експонренційної форми:

```
>> format short e
>> V

V =

   1.2000e+00  -3.0000e-01   1.2000e-05
```

У такий спосіб вводять вектори-рядки. Вектор-стовпець вводиться аналогічно, але значення елементів у переліку відокремлюються знаком «крапка з комою».

Введіть вектор *V* як вектор стовпець. Результат має бути таким:

```
>> V=[1.2;-0.3;1.2e-5]

V =

   1.2000e+00
  -3.0000e-01
   1.2000e-05
```

Matlab дає користувачу можливість скороченого введення вектора, значення елементів якого є арифметичною прогресією. Якщо позначити: *nz* – початкове значення цієї прогресії (значення першого елемента вектора); *kz* – кінцеве значення прогресії (значення останнього елемента вектора); *h* – розмірність прогресії (крок), то вектор можна ввести за допомогою короткого запису *V*=*nz*:*h*:*kz*. Наприклад, введення рядка *V*=-0.1:0.3:1.4 приводить до наступного результату:

```
>> format short
>> v=-0.1:0.3:1.4

v =

   -0.1000    0.2000    0.5000    0.8000    1.1000    1.4000
```

Якщо середня величина (різниця прогресії) не зазначена, то вона за замовчуванням приймається рівною одиниці. Наприклад, команда

```
>> -2.1:5
```

приводить до формування такого вектора:

```
ans =

  Columns 1 through 5

   -2.1000   -1.1000   -0.1000    0.9000    1.9000

  Columns 6 through 8

    2.9000    3.9000    4.9000
```

Завдання 4. Введення матриць.

Введення значень елементів матриці здійснюється в *Matlab* у квадратних дужках по рядках. При цьому елементи рядка матриці відокремлюються один від одного пробілом або комою, а рядки відокремлюються один від одного знаком «крапка з комою» (;).

Введите матрицу $A=\begin{bmatrix} 2 & 4 & 6 & 8 & 10 \\ 5.5 & 6.3 & 6.8 & 8 & 8.6 \end{bmatrix}$. Ваш результат:

```
>> A=[2 4 6 8 10;5.5 6.3 6.8 8 8.6]

A =

    2.0000    4.0000    6.0000    8.0000   10.0000
    5.5000    6.3000    6.8000    8.0000    8.6000
```

Завдання 5. Витягання і вставка частин матриць.

Звертання до будь-якого елемента визначеної матриці в *Matlab* здійснюється вказуванням після імені матриці (у дужках, через кому) двох цілих додатних чисел, що визначають відповідно номер рядка і стовпця матриці, на перетині яких розташований цей елемент.

Нехай маємо деяку матрицю *A*:

```
>> A=[1 2 3 4;5 6 7 8;9 10 11 12]

A =

     1     2     3     4
     5     6     7     8
     9    10    11    12
```

Одержати значення елемента цієї матриці, розташованого на

перетині другого рядка з третім стовпцем, можна в такий спосіб:

```
>> A(2,3)

ans =

     7
```

Якщо потрібно вставити на це місце деяке число, наприклад π, то це можна зробити так:

```
>> A(2,3)=pi

A =

    1.0000    2.0000    3.0000    4.0000
    5.0000    6.0000    3.1416    8.0000
    9.0000   10.0000   11.0000   12.0000
```

Іноді потрібно створити меншу матрицю з більшої, формуючи її шляхом витягання з останньої елементів її декількох рядків і стовпців, чи, навпаки, вставити меншу матрицю таким чином, щоб вона стала визначеною частиною матриці більшого розміру. Це робиться в *Matlab* за допомогою знака двокрапки (:).

Приклади.

Нехай потрібно створити вектор *V*1, що складається з елементів третього стовпця попередньої матриці *A*. Для цього можна зробити так:

```
V1 =

    3.0000
    3.1416
   11.0000
```

Створити вектор *V*2, що складається з елементів другого рядка матриці *A*, можна в такий спосіб:

```
>> V2=A(2,:)

V2 =

    5.0000    6.0000    3.1416    8.0000
```

Припустимо, що необхідно з матриці *A* створити матрицю *B* розміром (2*2), що складається з елементів лівого нижнього кута матриці *A*. Тоді команда така:

```
>> B=A(2:3,1:2)

B =

     5     6
     9    10
```

Аналогічно можна «вставити» матрицю *B* у верхню частину матриці *A*:

```
>> A(1:2,2:3)=B

A =

     1     5     6     4
     5     9    10     8
     9    10    11    12
```

Завдання 6. Дії над векторами.

Розрізняють дві групи дій над векторами: векторні дії, що передбачені векторним численням у математиці; дії з перетворення елементів – це дії, що перетворюють елементи вектора, але не є операціями з векторами, що існують у математиці. Розглянемо векторні дії над векторами.

Додавання векторів. Складати (підсумовувати) можна тільки вектори однакового типу, тобто обидва вектори повинні бути або векторами-рядками, або векторами-стовпцями і мати однакову довжину (однакову кількість елементів). Якщо *X* і *Y* саме такі вектори, то їхню суму можна одержати, якщо ввести команду *Z=X+Y*. Аналогічно за допомогою арифметичного знака мінус (-) здійснюється віднімання векторів, що мають однаковий тип: *Z=X-Y*.

Самостійно виконайте векторне додавання та віднімання векторів *x* = [1 2 3] та *y* = [4 5 6].

Транспонування вектора здійснюється застосуванням знака апострофа, що записується відразу за записом імені транспонованого вектора: *X'*.

Самостійно транспонуйте вектор *x*.

Множення вектора на число здійснюється в *Matlab* за допомогою знака арифметичного множення (*) у такий спосіб: *Z=X*r* або *Z=r*X*, де *r* – деяке дійсне число.

Самостійно помножте вектор y на число 2.

Множення двох векторів визначено в математиці тільки для векторів однакової довжини тоді, коли один з векторів-співмножників є рядком, а другий – стовпцем. Інакше кажучи, якщо вектори *X* та *Y* є рядками, то математичний зміст мають тільки дві форми множення цих векторів: *U*=*X'***Y* і *V*=*X***Y'*. Причому в першому випадку результатом буде квадратна матриця (діада), а в другому – число. У *Matlab* множення векторів здійснюється з використанням звичайного знака множення (*), що записується між співмножниками-векторами.

Самостійно помножте вектор y на число 2.

Дії з поелементного перетворення векторів.

У мові *Matlab* є ряд операцій, що перетворять заданий вектор в інший того ж розміру і типу, але вони не є математичними операціями з вектором як математичним об'єктом. Ці операції перетворюють елементи вектора як елементи звичайного одновимірного масиву чисел. До таких операцій належать всі елементарні математичні функції, що залежать тільки від одного аргументу. У мові *Matlab* запис виду *Y*=sin(*X*), де *X* – деякий відомий вектор, приводить до формування нового вектора *Y*, що має той же тип і розмір, але елементи якого дорівнюють синусу відповідних елементів вектора-аргументу *X*.

Крім вказаних операцій у *Matlab* передбачено кілька операцій поелементного перетворення, що здійснюються за допомогою знаків звичайних арифметичних дій. Ці операції застосовуються до векторів однакового типу і розміру. Результатом їх є вектор того ж типу і розміру. Розглянемо поелементні операції з векторами.

Додавання числа до кожного елемента вектора. Здійснюється за допомогою знака плюс (+). Протилежна дія – віднімання числа від кожного елемента вектора (-).

До кожного елемента вектора *x* = [1, 2, 3, 4, 5] додайте число 2, з кожного елемента вектора *y* = [-2, 1, 4, 0, 5] відніміть число 3.

Поелементне множення векторів. Виконується за допомогою сукупності знаків крапка та зірочка (.*), що записується між іменами векторів, які треба перемножити. Результатом буде вектор, кожен елемент якого є добутком відповідних елементів векторів-співмножників.

Виконайте поелементне множення векторів *x* та *y*.

Поелементне ділення векторів. Здійснюється за допомогою сукупності знаків крапка та слеш (./). Результат – вектор, кожен елемент якого є часткою від ділення відповідного елемента першого вектора на відповідний елемент другого вектора.

Виконайте поелементне ділення векторів *x* та *y*.

Поелементне ділення векторів у зворотному напрямку (справа наліво). Здійснюється за допомогою сукупності знаків крапка та зворотний слеш (.\). Результатом є вектор, кожен елемент якого є часткою від ділення відповідного елемента другого вектора на відповідний елемент першого вектора.

Виконайте поелементне ділення векторів *x* та *y* у зворотньому напрямку.

Поелементне піднесення до степеня. Здійснюється за допомогою сукупності знаків «.^». Результат – вектор, кожен елемент якого є відповідним елементом першого вектора, піднесеним до степеня, показником якого є значення відповідного елемента другого вектора.

Виконайте поелементне піднесення до степеня векторів *x* та *y*.

Завдання 7. Розрахунок значень складних математичних функцій

Обчислити значення функції $y=a\cdot e^{-hx}\square\sin x$ для значень аргументу *x* від 0 до 10 з кроком 1. Результат виконання:

```
>> a=3;h=0.5;
>> x=0:1:10;
>> y=a*exp(-h*x).*sin(x)

y =

  Columns 1 through 6

         0    1.5311    1.0035    0.0945   -0.3073   -0.2361

  Columns 7 through 11

   -0.0417    0.0595    0.0544    0.0137   -0.0110
```

Самостійно обчислити функцію $y = e^{-2x}\cdot\cos x^2 + \lg 2x$ для зна-

чень аргументу *x* від 0,1 до 4 з кроком 0,5.

Завдання 8. Дії з матрицями.

Для поелементного перетворення матриці придатні всі алгебраїчні функції. Кожна така функція формує матрицю того ж розміру, що і задана, кожен елемент якої обчислюється як зазначена функція від відповідного елемента заданої матриці. Крім того, у *Matlab* визначені операції поелементного множення матриць однакового розміру (за допомогою символів «.*», що поміщаються між іменами матриць, що «перемножуються»), поелементного ділення в обох напрямках («./» і «.\»), поелементного зведення в ступінь (символ «.^»), коли кожен елемент першої матриці підноситься до степеня, показник якого дорівнює значенню відповідного елемента другої матриці.

Оригінальною в мові *Matlab* є операція додавання до матриці числа. Вона записується так: *A*+*x* або *x*+*A* (де *A* – матриця, а *x* – число). Такої операції немає в математиці. У *Matlab* вона еквівалентна сукупності операцій *A*+*x***E*, де *E* – позначення матриці тих же розмірів, що і матриця *A*, яка складається тільки з одиниць.

До матричних дій з матрицями належать такі операції, що використовуються в матричному численні в математиці.

Базові дії з матрицями – додавання, віднімання, транспонування, множення матриці на число, множення матриці на матрицю, піднесення матриці до цілого степеня – здійснюються в мові *MATLAB* за допомогою звичайних знаків арифметичних операцій.

Умови, за яких ці операції можливі:

1) при додаванні або відніманні матриць вони повинні мати однакові розміри;

2) при множенні матриць число стовпців першого множника повинне збігатися з числом рядків другого множника.

Невиконання цих умов приведе до появи в командному вікні повідомлення про помилку.

Функція обернення матриці – *inv*(*A*) – обчислює матрицю, обернену заданій матриці *A*. Вихідна матриця *A* повинна бути квадратною, і її визначник не повинен дорівнювати нулю.

Піднесення матриці до степеня позначається в *Matlab* за допомогою комбінації знаків «.^» (наприклад, *A*.^*n*). При цьому n повин-

не бути цілим (додатним або від'ємним) числом. Ця матрична операція еквівалентна множенню матриці *A* на себе *n* раз (якщо *n* – додатне) або множенню оберненої матриці на себе (в разі від'ємного *n*).

Оригінальними в мові *Matlab* є дві нові, невідомі в математиці функції ділення матриць. При цьому вводяться поняття ділення матриць зліва направо і ділення матриць справа наліво. Перша операція записується за допомогою знака «/», а друга – за допомогою знака «\», що ставиться між іменами матриць, які діляться одна на одну.

Виконаємо дії з поелементного перетворення матриць.

Введіть матриці $A=\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ -2 & 3 & 1 & 6 & 0 \end{bmatrix}$ та $B=\begin{bmatrix} -1 & 3 & 5 & -2 & 1 \\ 5 & 6 & 7 & 8 & 9 \end{bmatrix}$.

Сформуйте матрицю, кожен елемент якої обчислюється як функція синус відповідного елемента заданої матриці *A*.

Виконайте поелементне множення матриць *A* та *B*.

Виконайте поелементне ділення матриць (використати два види ділення).

Виконайте поелементне піднесення до степеня матриць *A* та *B*.

Додайте до матриці *A* число 2.

Розглянемо матричні дії з матрицями.

Введіть матриці $A=\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 11 \end{bmatrix}$ та $B=\begin{bmatrix} 0 & -1 & -2 & -3 & -4 \\ 5 & 6 & 7 & 8 & 9 \end{bmatrix}$.

Виконайте додавання матриць *A* та *B*.

Віднімить з матриці *A* матрицю *B*.

Помножте матрицю *A* на число 5.

Транспонуйте матрицю А.

Помножте матриці (тому що при множенні матриць число стовпців першого множника повинне збігатися з числом рядків другого множника, то необхідно в даному разі виконати наступні дії: *C=A'*B* або *C=A*B'*).

Обчислить матрицю, обернену матриці *C*.

Виконайте піднесення матриці *A* до степеня 2.

Завдання 9. Знаходження коренів системи лінійних алгебраїчних рівнянь.

Операція *B*/*A* еквівалентна такій послідовності дій: *B***inv*(*A*).

Тут функція *inv* здійснює обернення матриці. Її зручно використовувати для розв'язання матричного рівняння *X***A*=*B*.

Аналогічно операція *A**B* рівнозначна сукупності операцій *inv*(*A*)**B*, що є розв'язанням матричного рівняння *A***X*=*B*.

Наприклад, треба знайти корені системи лінійних алгебраїчних рівнянь:

$x_1 + 2x_2 + 3x_3 = 14$

$2x_1 - x_2 - 5x_3 = -15$

$x_1 - x_2 - x_3 = -4$

У системі *Matlab* це можна зробити так:

```
>> A=[1 2 3;2 -1 -5;1 -1 -1]

A =

     1     2     3
     2    -1    -5
     1    -1    -1

>> B=[14;-15;-4]

B =

    14
   -15
    -4

>> x=A\B

x =

     1
     2
     3
```

Самостійно знайти корені системи лінійних алгебраїчних рівнянь:

$7{,}9x_1 + 5{,}6x_2 + 5{,}7x_3 - 7{,}2x_4 = 6{,}68$

$8{,}5x_1 - 4{,}8x_2 + 0{,}8x_3 + 3{,}5x_4 = 9{,}95$

$4{,}3x_1 + 4{,}2x_2 - 3{,}2x_3 + 9{,}3x_4 = 8{,}6$

$3{,}2x_1 -1{,}4x_2 - 8{,}9x_3 + 3{,}3x_4 = 1$

Завдання 16. Завершити роботу із системою, для чого можна використовувати команди *quit, exit* чи комбінацію клавіш *Ctrl+Q*.

Завдання 17. Письмово відповісти на контрольні запитання.

Контрольні запитання

1. У яких режимах може здійснюватися робота в середовищі *Matlab*?
2. З яких частин складається основне вікно системи *Matlab*?
3. Які формати виведення даних є в *Matlab*?
4. Перелічити елементарні функції у мові *Matlab* і вказати їх призначення.
5. Записати порядок виконання операцій у виразах *Matlab* згідно пріоритету.
6. Що таке системна змінна, яке її призначення?
7. В якому випадку результат виконання команди в *Matlab* не виводиться на екран?
8. Опишіть поелементне введення вектора-рядка.
9. Як вводиться вектор-стовпець?
10. Як ввести вектор, значення елементів якого є арифметичною прогресією?
11. Опишіть введення матриць.
12. Чим в *Matlab* відрізняються векторні дії над векторами та дії з поелементного перетворення векторів?
13. Які дії відносяться до поелементного перетворення матриць?
14. Які операції належать до матричних дій з матрицями?
15. Назвіть умови до виконання матричних дій з матрицями.

ЛАБОРАТОРНА РОБОТА № 2
ГРАФІЧНА ВІЗУАЛІЗАЦІЯ ОБЧИСЛЕНЬ У СИСТЕМІ *MATLAB*. ПОБУДОВА НАЙПРОСТІШИХ ГРАФІКІВ

Мета роботи – засвоїти прийоми побудови графіків функцій у системі *Matlab*.

Теоретичні відомості

Основною функцією, що забезпечує побудову графіків на екрані дисплея, є функція ***plot***. Загальна форма звертання до цієї процедури така:

***plot*(*x*1,*y*1,*s*1,*x*2,*y*2,*s*2,...)**.

Тут ***x*1**, ***y*1** – задані вектори, елементами яких є масиви значень аргументу (*x*1) і функції (*y*1), що відповідають першій кривій графіка; ***x*2**, ***y*2** – масиви значень аргументу і функції другої кривої і т. д. При цьому передбачається, що значення аргументу відкладаються уздовж горизонтальної осі графіка, а значення функції – уздовж вертикальної осі. Змінні ***s*1**, ***s*2**, ... є символьними (їх вказувати не обов'язково). Кожна з них може містити три спеціальних символи, що визначають тип лінії, що з'єднує окремі точки графіка, тип точок графіка і колір лінії. Якщо змінні s не вказані, то тип лінії за замовчуванням – відрізок прямої, тип точки – піксель. Якщо колір ліній і точок не вказано, він вибирається автоматично з таблиці кольорів (білий виключається).

Значеннями параметра ***s*** у процедурі ***plot*** можуть бути описані далі символи.

Колір лінії: ***Y*** – жовтий; ***M*** – фіолетовий; ***C*** – блакитний; ***R*** – червоний; ***G*** – зелений; ***B*** – синій; ***W*** – білий; ***K*** – чорний.

Тип крапки: **.** – крапка; **0** – коло; ***X*** – хрест; **+** – плюс; ***** – зірочка; ***S*** – квадрат; ***D*** – ромб; ***V*** – трикутник (вершиною донизу); ^ – трикутник (вершиною догори); < – трикутник (вершиною вліво); > – трикутник (вершиною вправо); ***P*** – п'ятикутник; ***H*** – шестикутник.

Тип лінії: **-** – суцільна; **:** – подвійний пунктир; **-.** – штрихпунктир; **--** – штрихова.

Графіки в *Matlab* завжди виводяться в окремому (графічному) вікні, що називається ***Figure*** (тобто рисунок, фігура).

Команда ***grid*** дозволяє задавати побудову сітки або скасовувати її:

1) ***grid on*** додає сітку до поточного графіка;

2) ***grid off*** відключає сітку;

3) ***grid*** послідовно робить включення і відключення сітки.

Цінною особливістю графіків, побудованих у системі *Matlab*, є

те, що сітка координат завжди відповідає «цілим» крокам зміни, що робить графіки «читабельними», тобто за графіком можна робити «відлік» значення функції при будь-якому заданому значенні аргументу і навпаки. Такої властивості немає в більшості графічних пакетів-додатків до мов програмування високого рівня.

Заголовок графіка виводиться за допомогою процедури ***title***. Якщо після звертання до процедури ***plot*** ввести команду ***title***, то зверху у вікні графіка з'явиться текст, записаний між апострофами:

***title*('<текст>')**.

Аналогічно можна вивести пояснення до графіка, що розміщаються уздовж горизонтальної осі (функція ***xlabel***) і уздовж вертикальної осі (функція ***ylabel***).

Команда ***text*(*x,y,* '<текст>')** дозволяє розмістити вказаний у ній текст у поле графіка, при цьому початок тексту розміщується в точці з координатами ***x*** и ***y***. Значення цих координат повинні бути вказані в одиницях величин, що відкладаються по осях графіка, і знаходитися в діапазоні зміни цих величин, що не зовсім зручно.

Більш зручним для розміщення тексту в полі графіка є використання команди ***gtext*('<текст>')**, за допомогою якої в активному графічному вікні висвітлюється перехрестя. Переміщуючи його за допомогою миші, можна вказати місце початку виводу вказаного тексту.

Щоб створити кілька графічних вікон, у кожному з яких розміщуються відповідні графіки, можна скористатися командою ***figure***, що створить нове графічне вікно, залишаючи попереднє.

Крім того, існує ще ряд команд, які призначені для створення графічних вікон і управління ними: ***gcf*** – одержати дескриптор графічного вікна ***figure***; ***cls*** – очистити графічне вікно; **shg** – показати раніше згорнуте графічне вікно; ***close*** – закрити графічне вікно; ***refresh*** – обновити графічне вікно.

Для того щоб кілька послідовних графіків відобразити в одному графічному вікні, можна використовувати команду ***hold on***. Тоді кожен наступний графік буде будуватися в тому ж попередньо відкритому графічному вікні, тобто нова лінія буде додаватися до раніше побудованого графіка. Команда ***hold off*** виключає режим збереження графічного вікна, встановлений попередньою командою. Команда ***hold*** працює як перемикач, послідовно включаючи

режим продовження графічних побудов і відключаючи його.

В одному графічному вікні, але на окремих графічних полях можна побудувати кілька графіків, використовуючи процедуру ***subplot***. Звертання до цієї процедури повинне передувати звертанню до процедури ***plot*** і виглядати так:

***subplot*(*m,n,p*)**,

де ***m*** вказує, на скільки частин поділяється графічне вікно по горизонталі, ***n*** – по вертикалі, а ***p*** є номером підвікна, у якому буде будуватися графік. Підвікна нумеруються зліва направо, зверху вниз (так, як читають текст книги).

Matlab має також інші графічні функції.

Функція ***bar*** зручна для перегляду результатів за певний період часу, порівняння результатів із різних наборів даних та демонстрації вкладу окремих елементів у загальну суму. За умовчанням ***bar*** представляє кожен елемент вектора чи матриці як одну смугу, отже висота смуги пропорційна значенню елемента. ***Bar*** функції можуть бути представлені наступними форматами:

1) ***bar*(*X,Y*)** будує стовпчастий графік елементів масиву ***Y*** у позиціях, зумовлених вектором ***X*** з упорядкованими в порядку зростання значень елементами. Якщо ***X*** і ***Y*** – двовимірні масиви однакового розміру, то стовпці будуються попарно один на одному;

2) ***bar*(*Y*)** будує графік значень елементів одновимірного масиву ***Y***;

3) ***bar*(*X,Y,WIDTH*)** чи ***bar*(*Y,WIDTH*)** – команда аналогічна раніше розглянутим, але з указанням ширини стовпців (при ***WIDTH*>1** стовпці перекриваються). За замовчуванням задана ***WIDTH* = 0,8**.

Якщо функція задана своїми значеннями при дискретних значеннях аргументу і невідомо, як вона може змінюватися в проміжках між значеннями аргументу, зручніше представляти графіка такої функції у вигляді окремих вертикальних ліній для кожного із заданих значень аргументу. Таким чином, вигляд графіка функції ***y*(*x*)** – її представлення дискретними одиницями (відліками). Цей вид графіка застосовується, наприклад, для опису квантування сигналів. Кожна одиниця представляється вертикальною рискою з колом угорі, причому висота риски відповідає у-координаті точки. Це можна зробити, застосовуючи процедуру ***stem***, звертання до якої

аналогічне звертанню до процедури ***plot***:

1) ***stem*(*Y*)** будує графік функції з ординатами у векторі ***Y*** у вигляді відліків;

2) ***stem*(*X,Y*)** будує графік відліків з ординатами у векторі ***Y*** і абсцисами у векторі ***X***;

3) ***stem*(...,*'filled'*)** будує графік функції з зафарбованими маркерами;

4) ***stem*(...,*'LINESPEC'*)** дає побудови, аналогічні наведеним командам, але зі специфікацією ліній ***'LINESPEC'***, що подібна специфікації для функції ***plot***.

Ще одна корисна інженеру функція – ***hist*** (побудова графіка гістограми заданого вектора). Стандартне звертання до неї має вигляд:

***hist*(*y,x*)**,

де ***y*** – вектор, гістограму якого потрібно побудувати, ***x*** – вектор, що визначає інтервали зміни першого вектора, всередині яких підраховується число елементів вектора ***y***. Ця функція виконує дві операції:

1) підраховує число елементів вектора **y**, значення яких попадають в діапазон, визначений вектором **x**;

2) будує стовпчасту діаграму підрахованих чисел елементів вектора **y** як функцію визначених вектором **x** діапазонів.

Функція ***comet*(*x,y*)** будує це анімований графік, у якому коло (голова комети) відстежує точки даних екрані. Тіло комети є хвостовий сегмент, що йде за головою. Хвіст є суцільною лінією, що окреслює всю функцію. Це може бути корисним при побудові просторових траєкторій, щоб показати, як траєкторія змінюється в часі.

Matlab має кілька функцій, що дозволяють будувати графіки в логарифмічному масштабі.

Функція ***logspace***, яка має формат ***x*=*logspace*(*d*1,*d*2,*n*)** формує вектор-рядок ***x***, що містить ***n*** рівновіддалених одна від одної у логарифмічному масштабі точок, що покривають діапазон від 10^{d1} до 10^{d2} .

Функція ***loglog*** повністю аналогічна функції ***plot***, але графіки по обох осях будуються в логарифмічному масштабі. Для побудови графіків, що використовують логарифмічний масштаб тільки по одній з координатних осей, користуються процедурами ***semilogx*** і ***semilogy***. Перша процедура будує графіки з логарифмічним масш-

табом уздовж горизонтальної осі, друга – уздовж вертикальної осі. Звертання до останніх трьох процедур аналогічне звертанню до функції ***plot***.

Пояснення у вигляді відрізків ліній з довідковими написами, розташовані всередині графіка або біля нього, називаються легендою. Для створення легенди використовуються різні варіанти команди ***legend***:

1) ***legend*(*string*1,*string*2,*string*3,...)** додає до поточного графіка легенду у вигляді рядків, зазначених у списку параметрів;

2) ***legend off*** усуває раніше виведену легенду;

3) ***legend*** перемальовує поточну легенду, якщо така є;

4) ***legend*(...,*Pos*)** поміщує легенду в точно визначене місце, специфіковане параметром ***Pos***: ***Pos*=0** - краще місце, вибирається автоматично; ***Pos*=1** – верхній правий кут; ***Pos*=2** – верхній лівий кут; ***Pos*=3** – нижній лівий кут; ***Pos*=4** - нижній правий кут; ***Pos*=-1** – праворуч від графіка.

Щоб перенести легенду, треба встановити на ній курсор, натиснути ліву кнопку миші і перетягти легенду в необхідну позицію.

Подвійним клацанням можна встановити режим редагування легенди.

Для зміни масштабу двовимірних графіків використовуються команди класу ***zoom***:

1) ***zoom*** переключає стан режиму інтерактивної зміни масштабу для поточного графіка;

2) ***zoom*(*FACTOR*)** встановлює масштаб згідно з коефіцієнтом ***FACTOR***;

3) ***zoom on*** вмикає режим інтерактивної зміни масштабу для поточного графіка;

4) ***zoom off*** вимикає режим інтерактивної зміни масштабу для поточного графіка;

5) ***zoom out*** забезпечує повний перегляд, тобто встановлює стандартний масштаб графіка;

6) ***zoom xon*** або ***zoom yon*** вимикає режим зміни масштабу тільки по осі ***x*** або по осі ***y***;

7) ***zoom reset*** запам’ятовує поточний масштаб як масштаб за замовчуванням для даного графіка;

8) ***zoom*(*FIG,OPTION*)** застосовується до графіка, заданого дескриптором ***FIG***, при цьому ***OPTION*** може бути будь-яким з перерахованих вище аргументів.

Команда ***zoom*** дозволяє керувати масштабуванням графіка за допомогою миші. Для цього треба підвести курсор миші до потрібного місця графіка. Якщо команда ***zoom*** включена (***on***), то натискання лівої кнопки збільшує масштаб удвічі, а правої – зменшує вдвічі. При натиснутій лівій кнопці миші можна виділити пунктирним чорним прямокутником потрібну ділянку графіка – при відпусканні кнопки він з'явиться у збільшеному вигляді й у тому масштабі, що відповідає прямокутнику, тобто в повну величину графічного вікна буде розгорнуте зображення, яке потрапило в прямокутник.

Команда ***zoom***, таким чином, виконує функцію лупи, що дозволяє спостерігати в збільшеному виді окремі фрагменти складних графіків. Однак варто враховувати, що для спостереження фрагментів графіків при великому збільшенні вони повинні бути задані великою кількістю точок. Інакше вигляд окремих фрагментів і тим більше окремих точок буде істотно відрізнятися від справжнього.

Робочі завдання

Завдання 1. Побудувати графік функції $y = 3\sin(x + \pi/3)$ на проміжку від -3π до $+3\pi$ із кроком $\pi/100$.

Спочатку треба сформувати масив значень аргументу ***x***: ***x*=-3*pi:pi/100:3*pi**, потім обчислити масив відповідних значень функції ***y*=3*sin(*x* + *pi*/3)** і, нарешті, побудувати графік залежності ***y*(*x*)**. В цілому, у командному вікні ця послідовність команд буде виглядати так:

```
» x=-3*pi:pi/100:3*pi;
» y=3*sin(x+pi/3);
» plot(x,y) ),grid
» title('Function y=3*sin(x + pi/3) ')
» xlabel('x');ylabel('y')
```

В результаті на екрані з'явиться додаткове вікно з графіком.

Завдання 2. В новому графічному вікні побудувати графік

функції $y(x)$, що задана параметричними формулами $x = 4e^{-0,05t}\sin(t)$, $y = 0,2e^{-0,1t}\sin(2t)$.

Виберемо діапазон зміни параметра t від 0 до 50 із кроком 0,1. Тоді, набираючи сукупність операторів:

```
» t=0:0.1:50;
» x=4*exp(-0.05*t).*sin(t);
» y=0.2*exp(-0.1*t).*sin(2*t);
» plot(x,y)
» title('Параметрична функція x=4*exp(-0.05t)*sin(t); y=0.2*exp(-0.1t)*sin(2t)')
» grid
```

Одержимо необхідний графік.

Завдання 3. В новому графічному вікні побудувати графіки двох функцій **sin(*x*)** і **cos(*x*)**, дані яких містяться в матриці ***Y***, а значення **x** зберігаються у векторі ***X***. Приклад виконання:

```
» % Побудова графіків двох функцій з вектором значень X
» X=[0 1 2 3 4 5];
» Y=[sin(X);cos(X)];
» plot(X,Y)
```

Завдання 4. В новому графічному вікні побудувати графіки трьох функцій з різним стилем представлення кожної з них. Тут графік функції $y1=\sin(x)$ будується суцільною фіолетовою лінією, графік $y2=\sin^2(x)$ будується штрихпунктирною лінією з крапками у вигляді знака «плюс» червоного кольору, а графік $y3=\sin^3(x)$ будується штриховою лінією з кружками чорного кольору.

```
» % Графіки трьох функцій зі специфікацією ліній кожного графіка
» x=-2*pi:0.1*pi:2*pi;
» y1=sin(x);
» y2=sin(x).^2;
» y3=sin(x).^3;
» plot(x,y1, '-m',x,y2, '-.+r',x,y3, '--ok')
```

За допомогою процедури ***plot*** можлива побудова графіка функції лінією, колір якої відрізняється від кольору вузлових точок. Так, якщо треба побудувати графік функції лінією синього кольору з червоними точками, то спочатку треба задати побудову графіка з

точками червоного кольору (без лінії), а потім графіка тільки лінії синього кольору (без точок).

Завдання 5. В нових графічних вікнах побудувати графіки вектора без аргументів двома способами: 1) за допомогою функції ***plot***; 2) за допомогою функції ***bar***.

Перший спосіб:

```
» x=[1 3 2 9 6 8 4 6];
» plot(x)
» grid
» title('Графік вектора X')
» ylabel('Значення елементів')
» xlabel('Номер елемента')
```

Другий спосіб:

```
» bar(x)
» title('Графік вектора X')
» xlabel('Номер елемента')
» ylabel('Значення елементів')
```

Завдання 6. Виконати приклади побудови графиків функцій різними способами.

6.1. В новому графічному вікні побудувати графік функції $\boldsymbol{y = e^{-x*x}}$ у вигляді стовпчастої діаграми для x в діапазоні від –2.9 до +2.9 з кроком 0,2. Порядок виконання:

```
» x =-2.9:0.2:2.9;
» bar(x,exp(-x.*x))
» title('Стовпчаста діаграма функції y =exp(-x^2) ')
» xlabel('Аргумент x')
» ylabel('Значення функції y')
```

6.2. В новому графічному вікні побудувати графік функції $\boldsymbol{y = \sin(x^2)\cdot e^{-x}}$ у вигдяді дискретних відліків функції для x в діапазоні від 0 до 4 з кроком 0,1. Порядок виконання:

```
» % Графік дискретних відліків функції
» x=0:0.1:4;
» y=sin(x.^2).*exp(-x);
» stem(x,y)
```

6.3. В новому графічному вікні побудувати гістограму для

діапазону зміни випадкових величин від -2,9 до +2,9 з кроком 0,1. Візьмемо загальне число елементів вектора цих випадкових величин 10000. Порядок виконання:

```
» x=-2.9:0.1:2.9;
» y=randn(10000,1);
» hist(y,x)
» ylabel('Кількість з 10000')
» xlabel('Аргумент')
» title('Гістограма нормального розподілу')
```

Проаналізувавши результати виконання наведених команд, можна зробити висновок, що функція *randn* досить правильно відображає нормальний гаусовий закон розподілу випадкової величини.

6.4. В новому графічному вікні побудувати графік залежності $\boldsymbol{y(x)}$ у вигляді траєкторії послідовного руху комети для $\boldsymbol{x = 4e^{-0.05t}\cdot\sin(t)}$; $\boldsymbol{y=0.2e^{-0.1t}\cdot\sin(2t)}$ для t в діапазоні від 0 до 50 з кроком 0,1. Порядок виконання:

```
» t=0:0.1:50;
» x=4*exp(-0.05*t).*sin(t);
» y=0.2*exp(-0.1*t).*sin(2*t);
» comet(x,y)
```

Завдання 7. В новому графічному вікні побудувати графіки амплітудно-частотної і фазо-частотної характеристик ланки, що описуються передавальною функцією $W(p)=\dfrac{p+4}{p^2+4\cdot p+100}$.

Для цього треба, по-перше, створити вектори-поліноми чисельника $P1$=[1 4] і знаменника $P2$=[1 4 100] передавальної функції.

По-друге, визначити корені цих двох поліномів:

```
» P1=[1 4];P2=[1 4 100];
» roots(P1)
ans =
-4
» roots(P2)
ans =
-2.0000e+000     +9.7980e+000i
-2.0000e+000      -9.7980e+000i
```

По-третє, задати діапазон зміни частоти таким, щоб він охоплював усі знайдені корені:

» om0=1e-2;omk=1e2;

Тепер потрібно задати кількість точок майбутнього графіка:

» n=41;

і сформувати масив точок по частоті ***OM=logspace*(-2,2,41)**, де значення -2 і +2 відповідають десятковим порядкам початкового (*om*0) і кінцевого (*омк*) значень частоти.

Користуючись функцією ***polyval***, можна обчислити спочатку вектор **ch** комплексних значень чисельника частотної передавальної функції (ЧПФ), що відповідає заданій передавальній функції, якщо як аргумент функції ***polyval*** використовувати сформований вектор частот *OM*, елементи якого помножені на уявну одиницю. Аналогічно обчислюється комплекснозначний вектор ***zn*** знаменника ЧПФ.

Вектор значень амплітудно-частотної характеристики (АЧХ) можна знайти, обчислюючи модулі векторів чисельника і знаменника ЧПФ і ділячи поелементно отримані вектори. Щоб знайти вектор значень фазово-частотної характеристики (ФЧХ), треба розділити поелементно комплекснозначні вектори чисельника і знаменника ЧПФ і визначити вектор аргументів елементів отриманого вектора. Щоб фази представити в градусах, отримані результати треба помножити на 180 і розділити на π.

Нарешті, для побудови графіка АЧХ у логарифмічному масштабі, досить застосувати функцію ***loglog***, а для побудови ФЧХ зручніше скористатися функцією ***semilogx***.

У цілому послідовність дій може бути такою:

```
» figure
» OM=logspace(-2,2,41);
» ch=polyval(P1,i*OM);
» zn=polyval(P2,i*OM);
» ACH=abs(ch)./abs(zn);
» loglog(OM,ACH)
» grid
» title('Графік амплітудно-частотної характеристики')
» xlabel('Частота (рад/с) ')
» ylabel('Відношення амплітуд')
```

```
» FCH=angle(ch./zn)*180/pi;
» semilogx(OM,FCH)
» grid
» title('Фазово-частотна характеристика')
» xlabel('Частота (рад/с) ')
» ylabel('Фаза (градуси) ')
```

Завдання 8. Помістити графіки із Завдання 7 в різних підвікнах одного графічного вікна. Порядок виконання:

```
» figure
» subplot(2,1,1)
» loglog(OM, AFC, ‘k’); grid
» title('Amplitude-Frequency Characteristic')
» ylabel('Amplitude')
» subplot(2,1,2)
» semilogx(OM, PFC, 'k'); grid
» title('Phase-Frequency Characteristic')
» xlabel('Frequncy (rad/s) '); ylabel('Phase (degrees) ')
```

Завдання 9. Самостійно побудувати в одному новому графічному вікні графіки чотирьох функцій, використовуючи команду ***hold on***, з легендою, розташованої у верхньому лівому куті графіка. Рядками легенди є назви графіків.

1. Напівкубічна парабола. Рівняння в параметричній формі: $x = t$, $y = at^2$. Параметри: $a = 2$; $-2 \le t \le 2$; $\Delta t = 0{,}5$. Графік функції – суцільна лінія синього кольору.

2. Локон Ан’єзі. Рівняння $y = \dfrac{a^3}{a^2 + x^2}$ Параметри: $a = 2$; $-10 \le x \le 10$; $\Delta x = 0{,}1$. Графік функції – штрихова фіолетова лінія.

3. Строфоїда. Рівняння в параметричній формі: $y = at\dfrac{t^2 - 1}{t^2 + 1}$, $x = a\dfrac{t^2 - 1}{t^2 + 1}$. Параметри: $a = 2$; $-10 \le t \le 10$; $\Delta t = 0{,}1$. Графік функції – вузлові зелені точки у виді зірочок.

4. Звичайна циклоїда. Рівняння в параметричній формі: $x = a(t - \sin(t))$, $y = a(1 - \cos(t))$. Параметри: $a = 2$; $-6{,}28 \le t \le 6{,}28$; $\Delta t = 0{,}5$.

Графік функції – вузлові точки типу хрест блакитного кольору.

Графічне вікно зберегти в окремому файлі з ім'ям **Здн_9**.

Завдання 10. Побудувати в новому графічному вікні графік функції $\frac{\sin(x^5)}{x^5+2^{-52}}$, де $-5 \le x \le 5$; $\Delta x = 0{,}01$. При цьому:

1. Перед виконанням завдання на ПК ввести текстовий коментар: *Завдання 10*.
2. Для запису числа 2^{-52} використати ім'я системної змінної *eps*.
3. Графік функції – суцільна червона лінія.
4. Встановити титульний напис (заголовок графіка): *Графік функції sin(x^5)/(x^5+2^(-52))*.
5. Встановити осьові написи. Біля осі ***x*** – *Вісь абсцис*; біля осі ***y*** – *Вісь ординат*.
6. Розмістити напис (*Крива загасаючого коливання*) під кривою графіка з позиції (-2.5, -0.3).
7. Вивести текст (*Двовимірний графік*) з позиції(-1, 1.1). Позиціонування тексту виконати за допомогою миші.
8. Змінити масштаб графіка. У повний розмір графічного вікна розгорнути зображення частини графіка (розглянути фрагмент графіка на інтервалі: по осі ***x*** – [1; 3], по осі ***y*** – [-0.1; 0.15]).

Графічне вікно зберегти в окремому файлі з ім'ям **Здн_10**.

Завдання 11. Побудувати в одному новому графічному вікні три графіки в різних підвікнах. При цьому графічне вікно по вертикалі містить одну частину, по горизонталі графічне вікно поділяється на три підвікна.

Побудову графіка функції $\sin(5t)$, де $0 \le t \le 2p$, $\Delta t = \pi/50$ здійснити:

- перше підвікно – відрізками прямих, параметр ***S*** – за замовчуванням;
- друге підвікно – стовпчастою діаграмою;
- третє підвікно – дискретними відліками функції.

Графічне вікно зберегти в окремому файлі з ім'ям **Здн_11**.

Завдання 12. Письмово відповісти на контрольні запитання.

Контрольні запитання

1. Описати призначення і формат команди ***plot***.
2. Описати призначення і формати команди ***bar***.
3. Описати призначення і формати команди ***stem***.
4. Записати стандартне звертання до функції, за допомогою якої виконують побудову графіка гістограми заданого вектора.
5. Описати формати команд оформлення графіків: установка титульного напису; установка осьових написів; створення легенди; введення тексту в будь-яке місце графіка з указанням координати точки, з якої починається текст; позиціонування тексту у визначеному місці графіка за допомогою миші; вмикання і вимикання сітки; зміна масштабу графіка.
6. Описати формати команд комбінування графіків: накладення графіків один на один; розбивка графічного вікна.

ЛАБОРАТОРНА РОБОТА № 3
ОПЕРАТОРИ УПРАВЛІННЯ ОБЧИСЛЮВАЛЬНИМ ПРОЦЕСОМ В *MATLAB*. РІШЕННЯ ЗАДАЧ ЗА ФАХОМ

Мета роботи – вивчити оператори організації обчислювального процесу, придбати навички рішення задач за фахом в середовищі *Matlab*.

Теоретичні відомості

Бульдозери є навісним устаткуванням на гусеничних і колісних тракторах. За характером роботи бульдозери можуть бути циклічної і безперервної дії. Найширше використовуються бульдозери циклічної дії. Їх робочий процес складається з наступних операцій:

- зарізання з утворенням перед відвалом призми ґрунту L_p;
- переміщення призми ґрунту перед відвалом на необхідну відстань $L_{пер}$;
- розвантаження ґрунту шляхом підйому відвала;

- повернення в забій (холостий хід).

Виконання вказаних операцій супроводжується поворотами машини, перемиканням її передач, підйомом і опусканням відвала $t_{оо}$.

Експлуатаційна продуктивність бульдозера – це кількість ґрунту, що розробляється і переміщується відвалом в одиницю часу. Експлуатаційна продуктивність машини дозволяє оцінити вплив конструктивних і технологічних параметрів її робочого процесу на продуктивність і ефективність використання машини в конкретних умовах роботи.

Експлуатаційна продуктивність бульдозера визначається за формулою (3.1):

$$П = \frac{3600 \cdot V_{гр} \cdot k_n \cdot k_В}{\left(\frac{L_р}{v_р} + \frac{L_{пер}}{v_{пер}} + \frac{L_{х.х.}}{v_{х.х.}}\right) + 2 \cdot \left(t_п + t_{п.п.}\right) + t_{оо}}, \ \frac{м^3}{год}, \qquad (3.1)$$

де $V_{гр}$ – обсяг призми волочіння, м^3; k_n – коефіцієнт, що враховує втрати у бічні валяння; $k_в$ – коефіцієнт, що враховує використання машини за часом ($k_в$ = 0,85); $L_р$, $L_{пер}$, $L_{х.х.}$ – довжина шляху різання, переміщення ґрунту і холостого ходу відповідно, м; $v_р$, $v_{пер}$, $v_{х.х.}$ – швидкості руху при різанні, переміщенні ґрунту і холостому ході відповідно, м/хв; $t_{п.п.}$, $t_п$, $t_{оо}$ – час на переключення передач, поворот бульдозера, опускання відвалу, с.

При цьому: час, за який бульдозер розвертається, $t_п = 2\text{-}10$ с; час на перемикання передач $t_{п.п.} = 4\text{-}5$ с; час на піднімання та опускання відвалу $t_{оо} = 2\text{-}3$ с.

Об’єм ґрунту перед відвалом $V_{гр}$ обчислюється за формулою (3.2):

$$V_{гр} = \frac{0{,}6 \cdot H^2 \cdot l}{k_р}, \qquad (3.2)$$

де H – висота відвалу, м; l – довжина відвалу, м; $k_р$ – коефіцієнт, що враховує розрихлення ґрунту ($k_р = 1{,}25\text{-}1{,}30$).

Коефіцієнт втрати ґрунту k_n обчислюється за формулою (3.3.):

$$k_n = 1 - 0{,}005 \cdot L_{\text{пер}}, \qquad (3.3)$$

де $L_{\text{пер}}$ – шлях переміщення ґрунту, м.

Шлях переміщення ґрунту $L_{\text{пер}}$ обчислюється за формулою (3.4):

$$L_{\text{пер}} = L_{\text{зах}} - L_{\text{р}}, \qquad (3.4)$$

де $L_{\text{зах}}$ – довжина захватки, м; $L_{\text{р}}$ – довжина шляху різання, м.

Шлях різання $L_{\text{р}}$ обчислюється за формулою (3.5):

$$L_{\text{р}} = \frac{v_{\text{пер}}}{l - h_{\text{сл}}}, \text{ м}, \qquad (3.5)$$

де $\nu_{\text{пер}}$ – швидкості руху при переміщенні ґрунту; l – довжина відвалу, м; $h_{\text{сл}}$ – товщина шару ґрунту, що зрізається, м.

Автогрейдер – самохідна планувальна машина безперервної дії для вирізування, переміщення і планування ґрунту. Автогрейдери застосовують для профілізації ґрунтових доріг, зведення земляного полотна висотою до 0,75 м, планування поверхні земляного полотна різної висоти, розрівнювання ґрунту при роботі скреперів, грейдер-елеваторів, доставці ґрунту насипу, що відсипається автомобілями-самоскидами, вирізування ґрунтового корита, розрівнювання шарів підстави дорожнього одягу, планування ґрунтових майданчиків, прибирання снігу з проїжджої частини автомобільних доріг.

Робочий процес автогрейдера складається з таких операцій:

1) операція зарізу – вирізування щільного ґрунту, переміщення його уздовж відвала;

2) операція переміщення – захоплення рихлого ґрунту з валика, переміщення ґрунту уздовж відвала в новий, переміщення убік валика;

3) операція планування – розрівнювання одного або декількох поряд укладених валиків шаром заданої товщини з плануванням всіх нерівностей.

При зведенні земляного полотна автогрейдер рухається по резерву і переміщає ґрунт в насип перпендикулярно напряму свого

руху;

Експлуатаційна продуктивність автогрейдера – це кількість ґрунту, що розробляється, переміщується і відсипається в одиницю часу (м^3/год).

Визначення експлуатаційної продуктивності автогрейдера при зведенні земляного полотна дозволяє оцінити вплив конструктивних і технологічних параметрів робочого процесу на продуктивність і ефективність використання цієї машини в конкретних умовах.

Експлуатаційна продуктивність автогрейдера розраховується за формулою (3.6):

$$\text{П} = \frac{3600 \cdot L \cdot F_{\text{Н}} \cdot K_{\text{В}}}{\left(\frac{n_{\text{з}}}{v_{\text{з}}} + \frac{n_{\text{п}}}{v_{\text{п}}} + \frac{n_{\text{о}}}{v_{\text{о}}}\right) + 2 \cdot t_{\text{пов}} \cdot \left(n_{\text{з}} + n_{\text{п}} + n_{\text{о}}\right)}, \ \frac{\text{м}^3}{\text{год}}, \tag{3.6}$$

де L – довжина робочої ділянки, м; $F_{\text{Н}}$ – площа перетину насипу, м^2; $K_{\text{в}}$ – коефіцієнт використання машини за часом ($K_{\text{в}}$ = 0,85); $n_{\text{з}}$ – число проходів при зарізанні ґрунту; $n_{\text{п}}$ – число проходів при переміщенні ґрунту; $n_{\text{о}}$ – число обробних проходів автогрейдера (16 - 20); $v_{\text{з}}$ – швидкість автогрейдера при зарізі ґрунту (1-а передача), м/с; $v_{\text{п}}$ – швидкість автогрейдера при переміщенні ґрунту (2-а передача), м/с; $v_{\text{о}}$ – швидкість автогрейдера при плануванні ґрунту (обробних проходах) (2 - 3-я передачі), м/с; $t_{\text{пов}}$ – час одного повороту автогрейдера (30 - 35), с.

Число проходів автогрейдера при зарізанні ґрунту $n_{\text{з}}$ обчислюється за формулою (3.7):

$$n_{\text{з}} = \frac{F_{\text{Н}} \cdot K_{\text{пп}}}{2 \cdot f_{\text{з}}}, \tag{3.7}$$

де $F_{\text{Н}}$ – площа перетину насипу, м^2; $K_{\text{пп}}$ – коефіцієнт перекриття проходів при переміщенні (приймати як 1,15); $f_{\text{з}}$ = 0,12.

Число проходів автогрейдера при переміщенні ґрунту $n_{\text{п}}$ обчислюється за формулою (3.8):

$$n_{\text{п}} = n_{\text{з}} \frac{l_{\text{о}}}{l_{\text{п}}} \cdot K_{\text{пп}} \qquad (3.8)$$

де $n_{\text{з}}$ – число проходів при зарізанні ґрунту; $l_{\text{о}}$ – середня довжина переміщення ґрунту з резерву в насип, м (при висоті насипу $h_{\text{Н}}$ = 0,75 м $l_{\text{о}}$ = 8 м); $l_{\text{п}}$ – відстань переміщення ґрунту за один прохід автогрейдера (приймати 2,2 м); $K_{\text{пп}}$ – коефіцієнт перекриття проходів при переміщенні (приймати як 1,15).

Робочі завдання

Завдання 1. Вивчити конструкції бульдозерів різних типів, виконати приклад розрахунку продуктивності бульдозера П при різних значеннях шару ґрунту $h_{\text{сл}}$, що зрізається, і різної дальності переміщення $L_{\text{пер}}$. За наслідками розрахунків побудувати графіки:

1) П = f ($h_{\text{сл}}$), коли $L_{\text{зах}}$ = *const*;

2) П = f ($L_{\text{зах}}$), коли $h_{\text{сл}}$ = *const*.

Дані для розрахунку продуктивності бульдозерів приведені в табл. 3.1.

Таблиця 3.1 – **Дані для розрахунку продуктивності бульдозерів**

Номер варіанту	Марка бульдозера	Марка трактора	Висота відвалу H, м	Довжина відвалу l, м	Коефіцієнт розпушування грунту $k_{\text{р}}$	Товщина зрізаємого грунту $h_{\text{сл}}$, м	Довжина захватки $L_{\text{зах}}$, м
1	ДЗ-28	Т-130Г	1,00	3,94	1,25	0,05 0,10 0,15 0,20 0,25	25 30 35 40 45
2	ДЗ-118	ДЕТ-250М	0,37	4,31	1,30	0,10 0,15 0,20 0,25 0,30	25 30 35 40 45

Номер варіанту	Марка бульдозера	Марка трактора	Висота відвалу H, м	Довжина відвалу l, м	Коефіцієнт розпушування грунту $k_р$	Товщина зрізаємого грунту $h_{сл}$, м	Довжина захватки $L_{зах}$, м
Приклад	ДЗ-62	ДТ-75	0,80	2,56	1,27	0,02 0,05 0,10 0,12 0,15	25 30 35 40 45

Розрахунок продуктивності та побудову графіків розглянемо на прикладі бульдозера ДЗ-62.

1. Вводимо значення коефіцієнтів, що наведені в умові задачі:

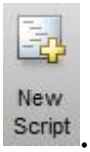
```
>> kb=0.85; vp=1.43; vper=1.59; vxx=197; tn=10; tnn=4.5; too=2;
```

2. Задаємо значення змінних з табл. 3.1:

Знаходимо об’єм ґрунту перед відвалом, коли $L_{зах}$ = *const*:

```
>> H=0.8; l=2.56; kp=1.27; Lzah=35;
>> hsl=0.02:0.01:0.15

hsl =

  Columns 1 through 7

    0.0200    0.0300    0.0400    0.0500    0.0600    0.0700    0.0800

  Columns 8 through 14

    0.0900    0.1000    0.1100    0.1200    0.1300    0.1400    0.1500
```

3. Послідовність команд для розрахунку продуктивності бульдозера напишемо у скрипт-файл. Для цього:

3.1. Відкриваємо вікно редактора скрипт-файлів за допомогою кнопки New Script 

.

3.2. У вікні редактора скрипт-файлів вводимо наступну послідовність команд:

```
Vgr=0.6*H^2/kp;
Lp=Vgr./(1*hsl);
```

```
Lper=Lzah-Lp;
Lxx=Lp+Lper;
kn=1-0.005*Lper;
P=3600*Vgr*kn*kp./((Lp/vp+Lper/vper+Lxx/vxx)+2*(tn+tnn)+too)
```

3.3. Зберігаємо створений скрипт-файл під ім'ям ***Example*1** та закриваємо вікно редактора скрипт-файлів.

4. У вікні *Command Window* набираємо ім'я створеного скрипт-файлу та отримаємо результат:

```
>> Example1

P =

  Columns 1 through 7

   39.3173   38.4687   38.0402   37.7818   37.6089   37.4851   37.3922

  Columns 8 through 14

   37.3198   37.2618   37.2143   37.1747   37.1412   37.1124   37.0875
```

5. Будуємо графік залежності П = f ($h_{сл}$), коли $L_{зах}$ = *const*:

```
>> subplot(2,2,1)
>> plot(hsl,P); grid
>> title('P=f(hsl) Lzax=const')
>> xlabel('Lzah'); ylabel('P')
```

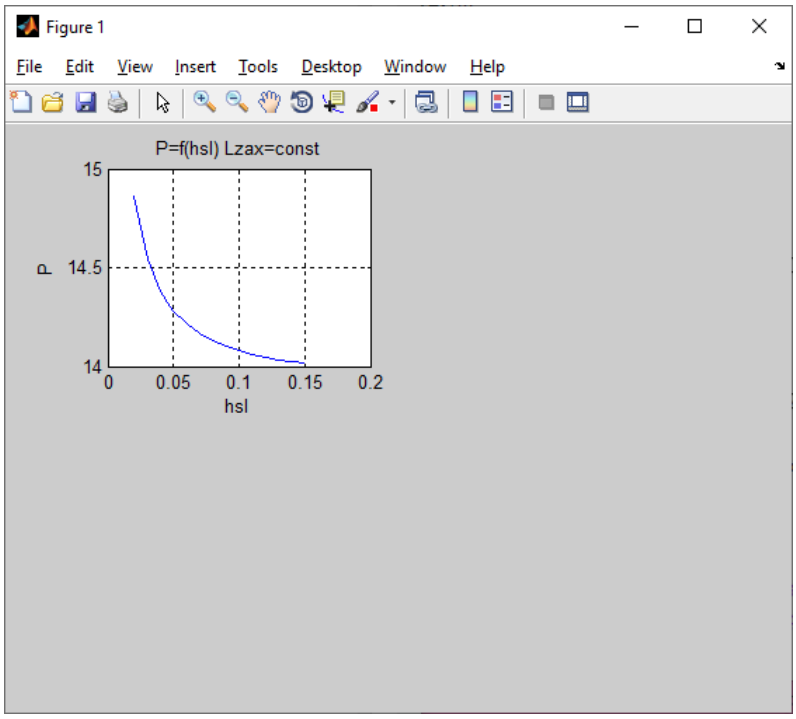


Рисунок 3.1 – Графік продуктивності бульдозера при $L_{зах}$ = *const*

6. Розраховуємо продуктивність бульдозера і будуємо графік залежності П = f ($L_{зах}$), коли $h_{сл}$ = *const*. Для цього:

6.1. Вводимо нові вихідні дані:

```
>> hsl=0.1;
>> Lzah=25:45;
```

6.2. Вводимо ім’я скрипт-файлу та отримаємо результат:

```
P =

  Columns 1 through 7

   44.7979   43.9541   43.1325   42.3321   41.5522   40.7920   40.0508

  Columns 8 through 14

   39.3278   38.6224   37.9339   37.2618   36.6054   35.9643   35.3379

  Columns 15 through 21

   34.7256   34.1271   33.5419   32.9695   32.4095   31.8615   31.3252
```

6.3. Будуємо графік у наступному підвікні:

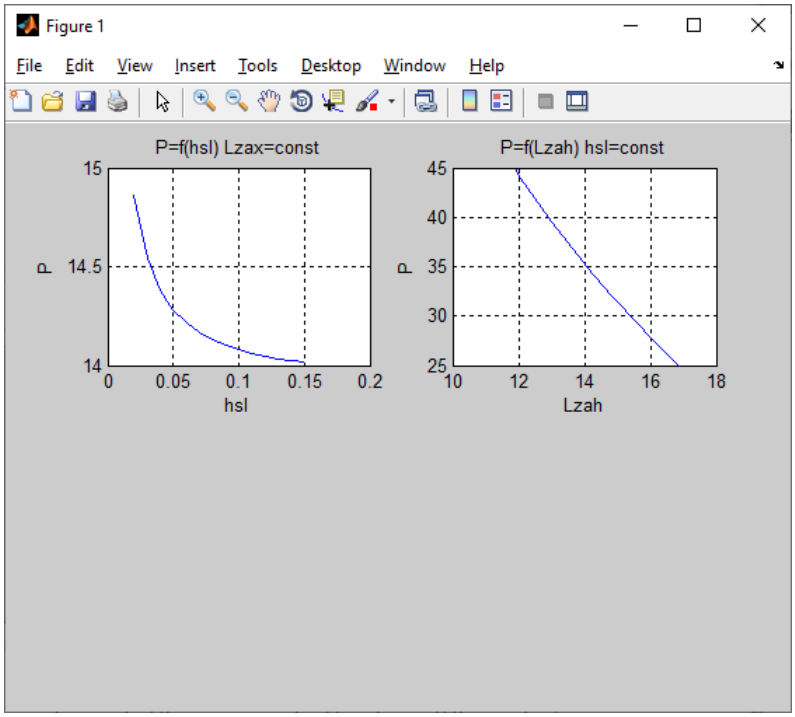


Рисунок 3.2 – Графік продуктивності бульдозера при $h_{сл}$ = *const*

Завдання 2. Використовуючи наданий приклад виконати розрахунки продуктивності бульдозера і побудувати графіки в

наступних підвікнах відповідно заданому варіанту (номер варіант надає викладач).

Завдання 3. Вивчити конструкції автогрейдерів різних типів. Виконати приклад розрахунку продуктивності автогрейдера П при різних значеннях довжини ділянки земельного полотна L, що відсипається, коефіцієнта використання машини за часом $K_в$ і висоти насипу $h_Н$. За даними розрахунків побудувати графіки:

1) П = f (L) при $K_в$ = 0,85, $h_Н$ = 0,75 м;
2) П = f ($K_в$) при L = 800 м, $h_Н$ = 0,75 м;
3) П = f ($h_Н$) при L = 800 м, $K_в$ = 0,85 м.

Дані для розрахунку продуктивності автогрейдерів приведені в табл. 3.2.

Таблиця 3.2 – **Дані для розрахунку продуктивності автогрейдерів**

Номер варіанту	Марка автогрейдера	Швидкість зарізу грунту, v_s	Швидкість переміщення грунту, v_n	Швидкість планування грунту, v_o	Коефіцієнт використання $K_В$	Довжина робочої ділянки L, м	Висота насипу $h_Н$, м
1	ДЗ-31А	1-а передача 0,97	2-а передача 2,10	3-а передача 2,22	0,20 0,40 0,60 0,80 0,85 0,90	200 400 600 800 1000 1200	0,30 0,45 0,60 0,75 0,90 0,80
2	ДЗ-98	1-а передача 0,97	2-а передача 1,38	3-а передача 1,75	0,20 0,40 0,60 0,80 0,85 0,90	200 400 600 800 1000 1200	0,30 0,45 0,60 0,75 0,90 0,80
Приклад	ДЗ-99	1-а передача 1,02	2-а передача 1,70	3-а передача 2,98	0,20 0,40 0,60 0,80 0,85 0,90	200 400 600 800 1000 1200	0,30 0,45 0,60 0,75 0,90 0,80

Розрахунок продуктивності та побудову графіків розглянемо на прикладі автогрейдера ДЗ-99.

Обчислимо продуктивність автогрейдера і побудуємо графік залежності П = f (L) при $K_в$ = 0,85, $h_Н$ = 0,75 м.

1. Вводимо значення коефіцієнтів, що наведені в умові задачі:

```
>> Knn=1.15; fz=0.12; lo=8; ln=2.2; no=18; tn=33;
```

2. Задаємо значення змінних з табл. 3.2:

```
>> vz=1.02; vn=1.7; vo=2.98; KB=0.85; hH=0.75;
>> L=200:200:1200

L =

     200     400     600     800    1000    1200
```

3. Створюємо новий скрипт-файл для обчислення продуктивності автогрейдера та зберігаємо його під ім'ям ***Example2***. Для визначення $F_Н$ використовувати рис. 3.3.

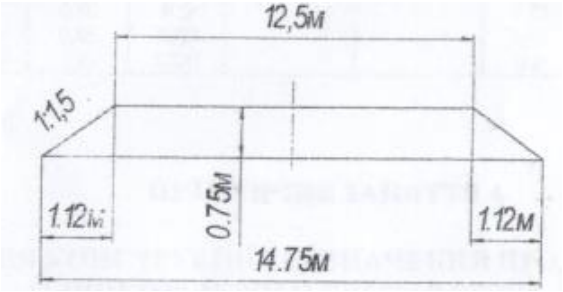


Рисунок 3.3 – Визначення площі земляного полотна автомобільної дороги 2-ої категорії $h_Н$ = 0,75 м

Перелік команд для скрипт-файлу:

```
FH=hH*1.12+hH*(14.75-2*1.12);
nz=FH*Knn/(2*fz);
nn=Knn*nz*lo/ln;
P=3600*L*FH*KB./(2*L*(no/vo+nn/vn+nz/vz)+2*tn*(nz+nn+no))
```

4. У вікні *Command Window* набираємо ім'я другого скрипт-файлу та отримаємо результат:

```
>> Example2

P =

   71.2855   79.4010   82.5330   84.1935   85.2222   85.9222
```

5. У новому графічному вікні будуємо графік залежності

П = f (L) при $K_в$ = 0,85, $h_Н$ = 0,75 м. Порядок виконання:

```
>> figure
>> subplot(2,3,1)
>> plot(P,L);grid
>> title('P=f(L) KB=const hH=const')
>> xlabel('L'); ylabel('P')
```

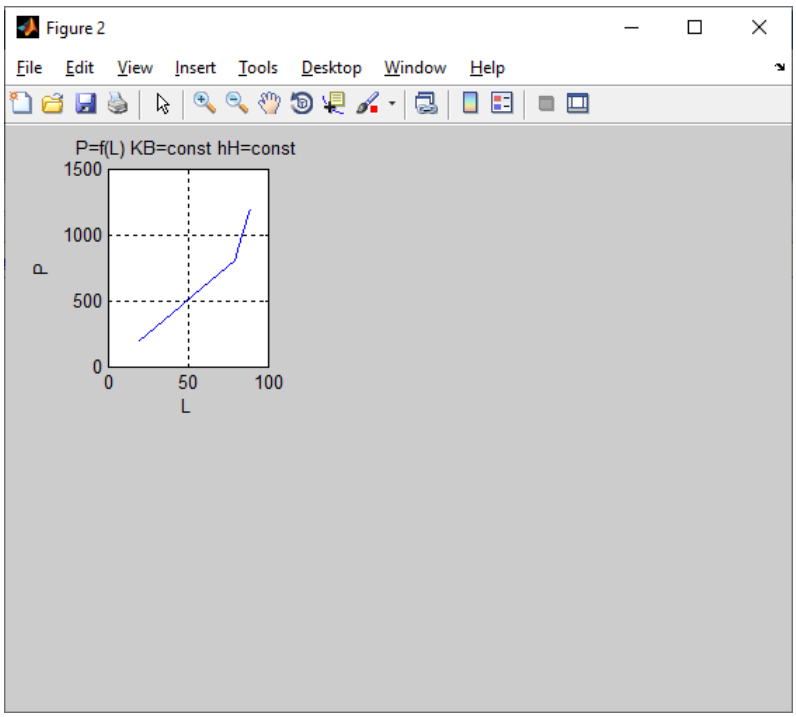


Рисунок 3.4 – Графік продуктивності автогрейдера при $K_в$ = 0,85, $h_Н$ = 0,75 м

Обчислюємо продуктивність автогрейдера і будуємо графік залежності П = f ($K_в$) при L = 800 м, $h_Н$ = 0,75 м.

1. Вводимо нові вихідні дані:

```
>> L=800;
>> KB=[0.2 0.4 0.6 0.8 0.85 0.9]

KB =

    0.2000    0.4000    0.6000    0.8000    0.8500    0.9000
```

2. Обчислюємо продуктивність автогрейдера за допомогою скрипт-файлу ***Example2*** та будуємо графік в наступному підвікні. Отримаємо наступні результати:

```
>> Example2

P =

   19.8102    39.6205    59.4307    79.2409    84.1935    89.1460
```

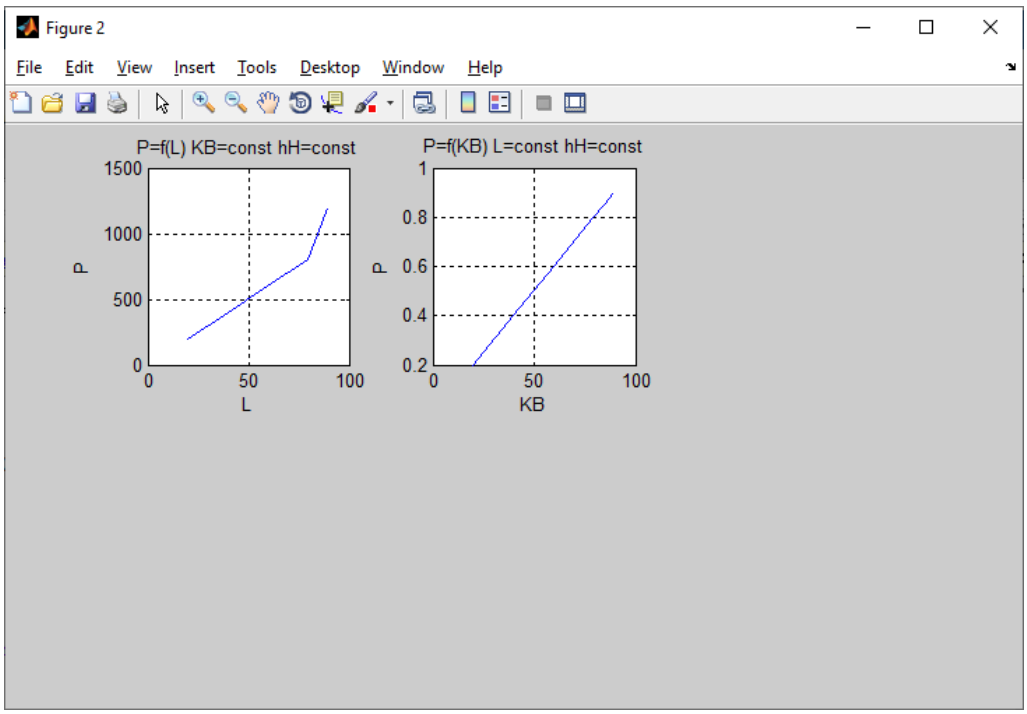


Рисунок 3.5 – Графік продуктивності автогрейдера при L = 800 м, $h_Н$ = 0,75 м

Обчислення продуктивності автогрейдера і побудову графіка залежності П = f ($h_Н$) при L = 800 м, $K_в$ = 0,85 м виконуємо аналогічно.

Вихідні дані:

```
>> KB=0.85;
>> hH=[0.3 0.45 0.6 0.75 0.9 0.8]

hH =

    0.3000    0.4500    0.6000    0.7500    0.9000    0.8000
```

Результати:

```
>> Example2

P =

   79.8218    82.1928    83.4319    84.1935    84.7089    84.3860
```

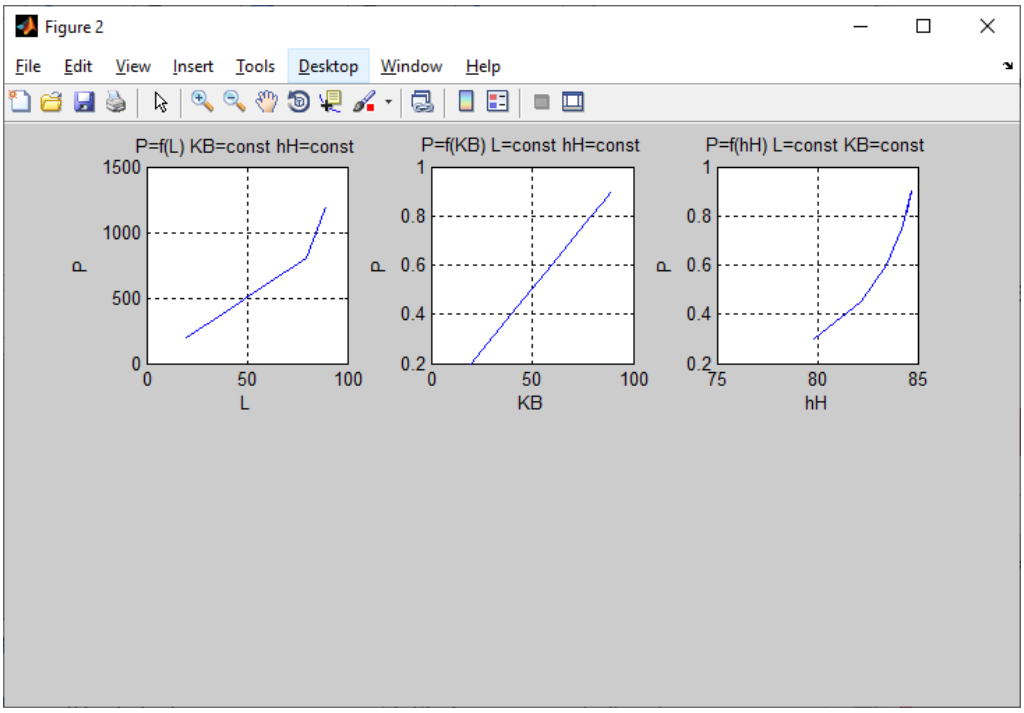


Рисунок 3.6 – Графік продуктивності автогрейдера при L = 800 м, $K_в$ = 0,85 м

Завдання 4. Використовуючи наданий приклад виконати розрахунки продуктивності автогрейдера і побудувати графіки в наступних підвікнах відповідно заданому варіанту (номер варіант надає викладач).

Завдання 5. Вивчити оператори управління обчислювальним процесом. Виконати приклади.

До операторів управління обчислювальним процесом відносять оператори умовних переходів (розгалуження обчислювального процесу) і оператори організації циклічних процесів. Однак, система *Matlab* побудована таким чином, що ці оператори можуть бути використані і при роботі *Matlab* у режимі калькулятора.

Всі оператори циклу й умовного переходу побудовані в *Matlab* у вигляді складного оператора, що починається службовим словом *if*, *while* або *for* і закінчується службовим словом end. Оператори між цими словами сприймаються системою як частини одного складного оператора. Тому натискання клавіші *Enter* для переходу до наступного рядку не приводить у даному випадку до виконання цих операторів. Виконання операторів починається тільки тоді, коли введена «закриваюча дужка» складного оператора у вигляді слова *end*, а потім натиснута клавіша *Enter*. Якщо кілька складних опе-

раторів такого типу вкладені один в один, обчислення починаються лише тоді, коли записаний кінець (*end*) найбільш «зовнішнього» складного оператора. З цього випливає можливість здійснення навіть у режимі калькулятора досить складних і об'ємних (що складаються з багатьох рядків і операторів) обчислень, якщо вони охоплені складним оператором.

Оператори умовного переходу. Оператор умовного переходу в загальному вигляді має таку конструкцію:

if <умова>
<оператори1>
else
<оператори2>
end

Працює він у таким чином: спочатку перевіряється, чи виконується вказана умова. Якщо так, то програма виконує сукупність операторів, що записана в розділі <оператори1>. У протилежному випадку виконується послідовність операторів розділу <оператори2>.

Скорочена форма умовного оператора має наступний вигляд:

if <умова>
<оператори>
end

Дія оператора аналогічна попередньому випадку, за винятком того, що при невиконанні заданої умови виконується оператор, що йде за оператором *end*.

Недолік цього оператора, що є наслідком відсутності оператора безумовного переходу, це те, що вся частина програми, яка виконується за умовою, повинна розміщатися всередині операторних дужок *if* та *end*.

Умовою в описаному операторі є вираз типу

<ім'я змінної1> <операція порівняння> <ім'я змінної2>

Операції порівняння в мові *Matlab* можуть бути такими: < (менше); > (більше); <= (менше або дорівнює); >= (більше або дорівнює); = = (дорівнює); ~ = (не дорівнює).

Припустима ще одна конструкція, яку називають повною формою оператора умовного переходу:

if <умова1>

<оператори1>
elseif <умова2>
<оператори2>
elseif <умова3>
<оператори3>
…
else
<оператори>
end

Оператор *elseif* виконується тоді, коли <умова1> не виконана. При цьому спочатку перевіряється <умова2>. Якщо вона істинна, виконуються <оператори2>, якщо ж ні, то <оператори2> ігноруються і відбувається перехід до наступного оператора *elseif*, тобто до перевірки <умови3>. Аналогічним чином при його виконанні обробляються <оператори3>, у протилежному випадку відбувається перехід до наступного оператора *elseif*. Якщо жодна з умов в операторах *elseif* не виконується (тобто всі умови хибні), працюють <оператори>, що йдуть за оператором *else*. Так можна забезпечити розгалуження програми у кількох напрямках.

Роботу оператора умовного переходу розглянемо на прикладі: треба вичислити значення y за формулою ***y=a*x***, якщо ***x>=0***, та ***y=b*x***, якщо ***x<0***, де ***a=3***, ***b=4***, ***x*** - може приймати будь-яке значення. для рішення цієї задачі скористаємося вікном редактора скрипт-файлів. В цьому вікні запишемо наступний програмний код для обчислення значення ***y*** за умовами нашої задачі:

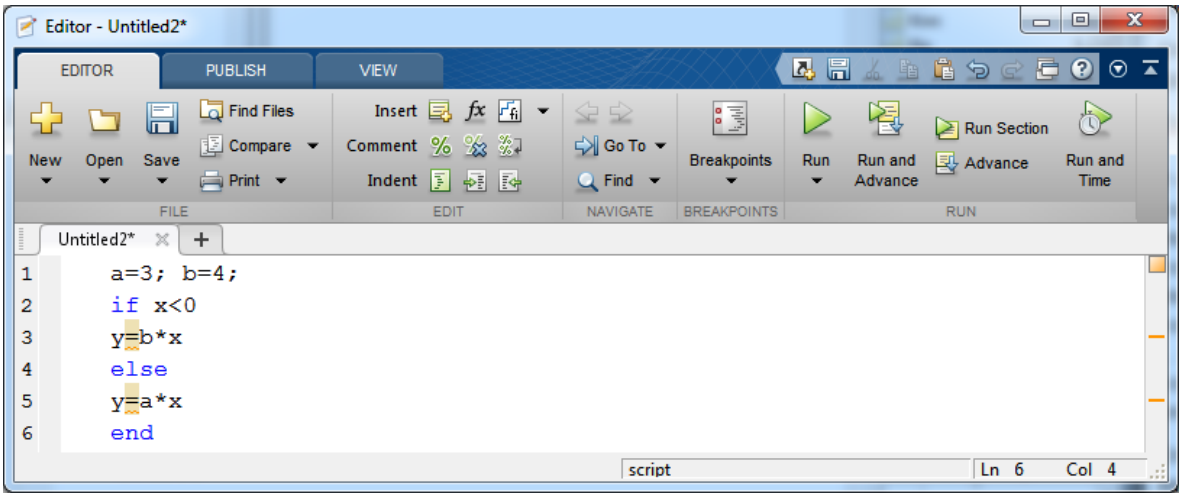


Рисунок 3.7 – Вікно редактора скрипт-файлів з програмним кодом рішення задачі обчислення ***y*** за заданою умовою

Створений скрипт-файл зберігаємо під ім'ям ***LR*3**.

Виконайте розрахунки значень **y** за наступними значеннями ***x***: 1,6; -0,7; 0,9; -0,1; 0,4; 0,5. Для цього у командному вікні системи *Matlab* задайте перше значення ***x*** і клацніть кнопку *Run* вікна редактора скрипт-файлів. В командному вікні з'явиться відповідь:

```
>> x=1.6;
>> LR7

y =

    4.8000
```

Самостійно виконайте обчислення усіх значень ***y***.

Оператори циклу. У мові *Matlab* є два різновиди операторів циклу – з передумовою та арифметичний.

Оператор циклу з передумовою має наступний вигляд:

while <умова>

<оператори>

end

Оператори всередині циклу обробляються лише в тому разі, якщо виконується умова, записана після слова *while*. При цьому серед операторів усередині циклу обов'язково повинні бути такі, що змінюють значення однієї із змінних, вказаних в умові циклу.

Арифметичний оператор циклу має наступний вигляд:

for <ім'я>=<НЗ>:<Ш>:<ДОЗ>

<оператори>

end

де <ім'я> – ім'я керуючої змінної циклу – лічильника циклу; <НЗ> – задане початкове значення цієї змінної; <Ш> – значення кроку, з яким вона повинна змінюватися; <ДОЗ> – кінцеве значення змінної циклу. У цьому разі <оператори> всередині циклу виконуються кілька разів (щоразу з новим значенням керуючої змінної) доти, поки значення керуючої змінної не вийде за межі інтервалу між <НЗ> і <ДОЗ>. Якщо параметр <Ш> не зазначений, за замовчуванням його значення приймається рівним одиниці.

Щоб достроково вийти з циклу (наприклад, при виконанні деякої умови), застосовують оператор *break*. Якщо в програмі зустрічається цей оператор, виконання циклу достроково при-

пиняється і починає виконуватися наступний після слова *end* оператор.

Наприклад, обчислити значення синуса при десяти значеннях аргументу від 0.2 до 2 із кроком 0.2. Обчислення виконаємо в режимі калькулятора у командному вікні системи *Matlab*:

```
>> for x=0.2:0.2:2
y=sin(x); disp([x, y])
end
    0.2000    0.1987

    0.4000    0.3894

    0.6000    0.5646

    0.8000    0.7174

    1.0000    0.8415

    1.2000    0.9320

    1.4000    0.9854

    1.6000    0.9996

    1.8000    0.9738

    2.0000    0.9093
```

Завдання 6. Письмово відповісти на контрольні запитання.

Контрольні запитання

1. Що таке бульдозер?
2. З яких операцій складається робочий процес бульдозерів?
3. Що називають експлуатаційною продуктивністю бульдозера?
4. Запишіть формулу, за якою визначається продуктивність бульдозера. Надайте пояснення складникам формули.
5. Що таке автогрейдер?

6. Де застосовуються автогрейдери?

7. З яких операцій складається робочий процес автогрейдерів?

8. Запишіть формулу, за якою визначається продуктивність автогрейдера. Надайте пояснення складникам формули.

9. Які оператори відносяться до операторів управління обчислювальним процесом?

10. Як побудовані оператори управління обчислювальним процесом в *Matlab*?

11. Запишіть загальний вигляд оператора умовного переходу Matlab. Поясніть, як він виконується.

12. Запишіть скорочену форму оператора умовного переходу Matlab. Чим скорочена форма відрізняється від загального вигляду оператора умовного переходу?

13. Запишіть повну форму оператора умовного переходу Matlab. Опишіть роботу оператора повної форми.

14. Які види операторів циклу існують в *Matlab*?

15. Запишіть формат оператора циклу з передумовою. Опишіть його роботу.

16. Запишіть формат арифметичного оператора циклу *Matlab*. Надайте йому пояснення.

17. Який оператор застосовують для дострокового виходу з циклу в *Matlab*?

ЛАБОРАТОРНА РОБОТА № 4
МОДЕЛЮВАННЯ НЕЧІТКИХ СИСТЕМ ТА НЕЙРОННИХ МЕРЕЖ ЗАСОБАМИ MATLAB

Мета роботи – придбання практичних навичок роботи в пакеті *Fuzzy Logic Toolbox* по створенню нечіткої моделі управління, дослідження можливостей *Matlab* з моделювання та навчання нейронних мереж.

Теоретичні відомості

Графічний інтерфейс *Fuzzy Logic Toolbox*

Пакет *Fuzzy Logic Toolbox* (пакет нечіткої логіки) – це сукупність прикладних програм, що дозволяють конструювати нечіткі експертні та/або керуючі системи. Даний пакет є одним з інструментальних засобів широко відомої і поширеної в нашій країні математичної системи *Matlab*.

До складу програмних засобів *Fuzzy Logic Toolbox* входять наступні основні програми, що дозволяють працювати в режимі графічного інтерфейсу:

1) редактор нечіткої системи виведення *Fuzzy Inference System Editor* (*FIS Editor* або *FIS*-редактор);

2) редактор функцій приналежності (*Membership Function Editor*);

3) редактор правил (*Rule Editor*);

4) програма перегляду правил (*Rule Viewer*);

5) програма перегляду поверхні системи нечіткого виведення (*Surface Viewer*).

Набір даних програм надає користувачеві максимальні зручності для створення, редагування та використання різних систем нечіткого виводу.

Редактор систем нечіткого виводу

Редактор систем нечіткого виводу *FIS* (або просто редактор *FIS*) є основним засобом, що використовується для створення або

редагування систем нечіткого виводу в графічному режимі. Редактор *FIS* може бути відкритий за допомогою введення функції *fuzzy* у вікні команд.

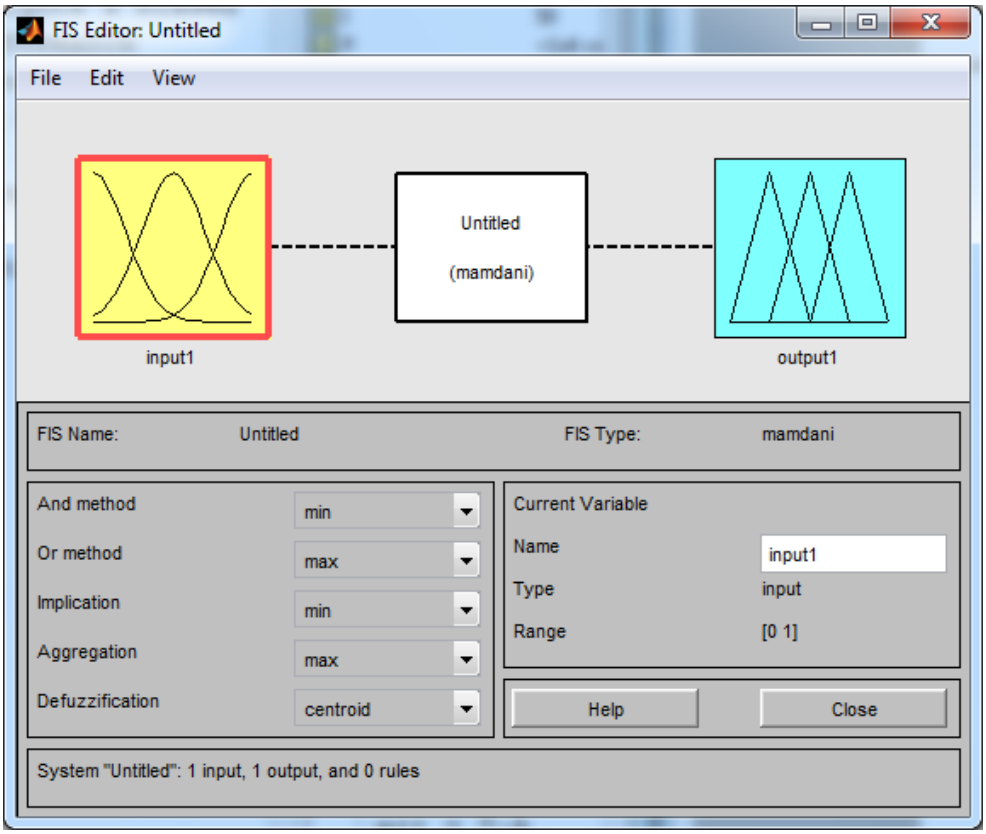


Рисунок 4.1 – Вікно редактора систем нечіткого виводу

Редактор *FIS* має головне меню, яке дозволяє користувачеві викликати інші графічні засоби роботи з системою нечіткого виводу *FIS*, завантажувати і зберігати структуру *FIS* в зовнішніх файлах і т. д.

Головне меню редактора містить позиції:

File – робота з файлами моделей (їх створення, збереження, зчитування і друк);

Edit – операції редагування (додавання і виключення вхідних і вихідних змінних);

View – перехід до додаткового інструментарію.

Пункт меню ***File*** (**Файл**) редактора *FIS* містить наступні операції:

New FIS... – дозволяє вибрати тип нової системи нечіткого виводу: *Mamdani* – типу Мамдані або *Sugeno* – типу Сугено. При цьому система нечіткого виводу не має ні вхідних, ні вихідних

змінних, а її ім'я задається за замовчуванням як *Untitled*;

Import – дозволяє завантажити в редактор *FIS* існуючу систему нечіткого виводу одним із таких способів: ***From Workspace...*** – з робочого простору програми *Matlab* або ***From File...*** – з зовнішнього файлу;

Export – дозволяє зберегти систему нечіткого виводу одним із таких способів: ***To Workspace...*** – в робочому просторі програми *Matlab* або ***To File...*** – в зовнішньому файлі;

Print – дозволяє роздрукувати на принтері систему нечіткого виводу;

Close – закриває редактор *FIS*.

Пункт меню ***Edit*** (**Редагування**) містить наступні операції:

Undo – скасовує виконання останньої дії;

Add Variable... – дозволяє додати в систему нечіткого виводу змінну одного з наступних типів: ***Input*** – вхідну змінну або ***Output*** – вихідну змінну;

Remove Selected Variable – видаляє обрану змінну з системи нечіткого виводу;

Membership Functions... – викликає редактор функцій приналежності;

Rules – викликає редактор правил нечіткого виводу.

Пункт меню ***View*** (**Вид**) містить наступні операції:

Rules – викликає програму перегляду правил нечіткого виводу;

Surface – викликає програму перегляду поверхні нечіткого виводу.

Редактор функцій приналежності

Редактор функцій приналежності призначений для завдання і редагування функцій приналежності системи нечіткого виведення в графічному режимі. Програму-редактор функцій приналежності можна відкрити трьома способами:

- через пункт меню ***View → Edit membership functions...***;
- подвійним клацанням лівої кнопки миші по іконці, що відображає відповідну змінну;
- натисканням комбінацію клавіш [***Ctrl*+2**].

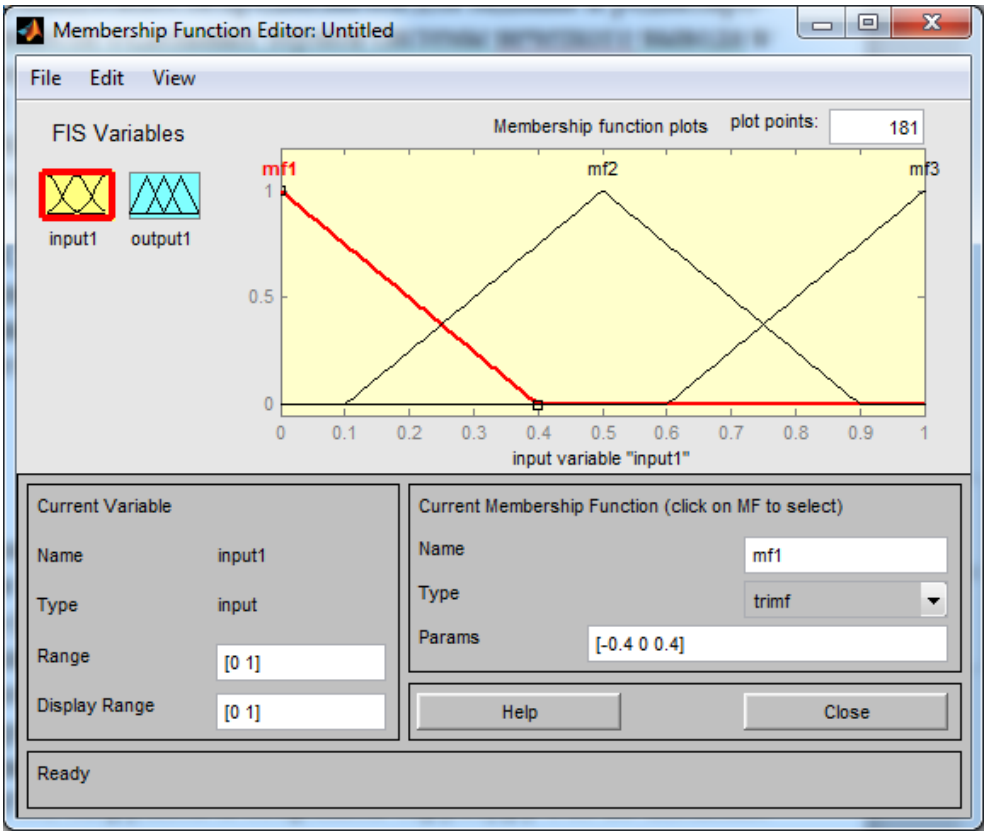


Рисунок 4.2 – Вікно редактора функцій приналежності

Редактор надає користувачеві не тільки можливість вибрати будь-яку з 11 вбудованих функцій приналежності, але і задати власну функцію приналежності. Для кожної функції приналежності можна змінити її ім’я, тип і параметри.

Для відображення графіків функцій приналежності слід вибрати необхідну змінну в лівій частині графічного інтерфейсу редактора під заголовком ***FIS Variables*** (**Змінні *FIS***).

Щоб вибрати потрібну функцію приналежності, слід клацнути на ній або її мітці в основному вікні з графіками функцій приналежності.

Редактор функцій приналежності має головне меню програми, яке дозволяє користувачеві викликати інші графічні засоби роботи з системою нечіткого виводу *FIS*, завантажувати і зберігати структуру *FIS* в зовнішніх файлах і т. д.

Пункт меню ***File*** (**Файл**) редактора функцій приналежності містить такі ж операції, що і відповідний пункт меню редактора *FIS*.

Пункт меню ***Edit*** (**Редагування**) містить наступні операції:

Undo – скасовує виконання останньої дії;

Add *MF*... – дозволяє додати вбудовану функцію приналежно-

сті для виділеної змінної;

Add Custom MF... – дозволяє додати власну функцію приналежності для окремої змінної;

Remove Current MF – дозволяє видалити окрему функцію приналежності;

Remove All MFs – дозволяє видалити всі функції приналежності для окремої змінної;

FIS Properties... – викликає редактор *FIS*;

Rules... – викликає редактор правил нечіткого виводу.

Пункт меню ***View*** (**Вид**) містить наступні операції;

Rules – викликає програму перегляду правил нечіткого виводу;

Surface – викликає програму перегляду поверхні нечіткого виводу.

Редактор правил системи нечіткого виводу

Редактор правил системи нечіткого виводу призначений для завдання і редагування окремих правил системи нечіткого виводу в графічному режимі. Редактор правил може бути відкритий за допомогою головного меню редактора *FIS* командою меню ***Edit*** → ***Rules...*** або натисканням комбінацію клавіш [***Ctrl*+3**].

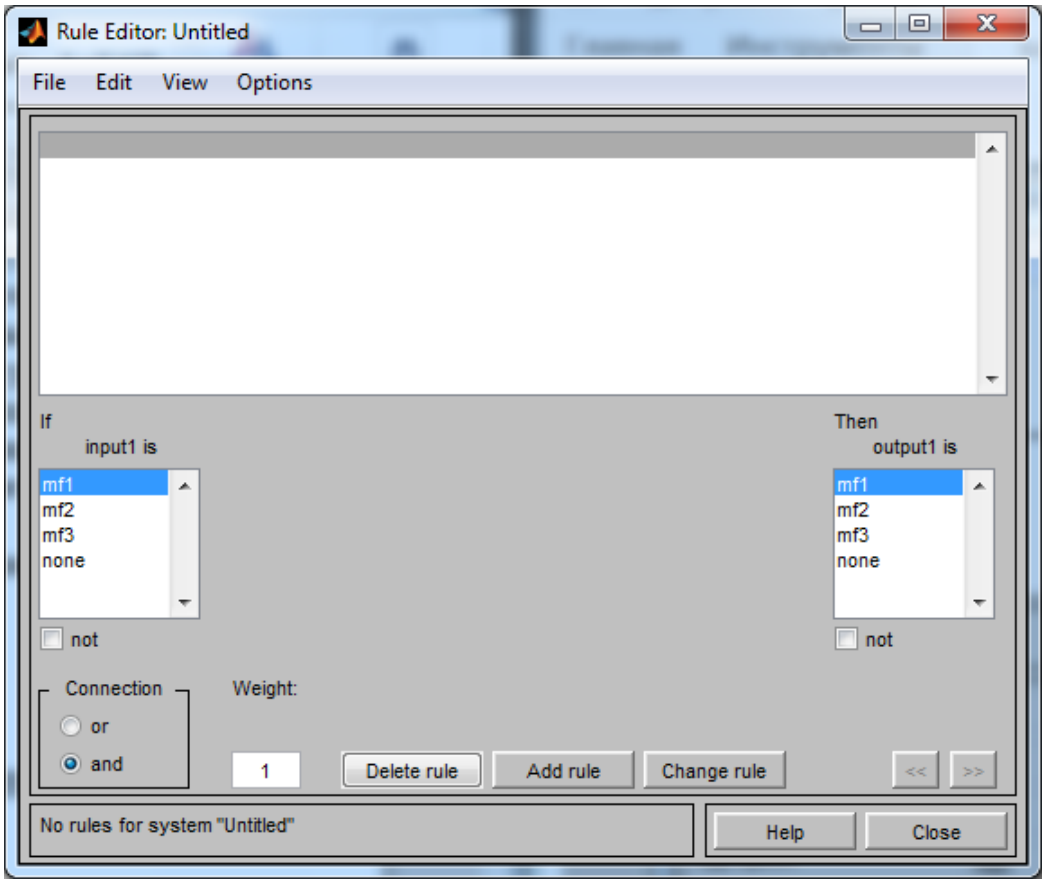


Рисунок 4.3 – Вікно редактора правил

Щоб використовувати цей редактор для створення правил, необхідно попередньо визначити все вхідні і вихідні змінні, для чого можна скористатися редактором системи нечіткого виведення FIS і редактором функцій приналежності. При цьому задати правила можна за допомогою вибору відповідних значень вхідних і вихідних змінних.

Пункт меню ***File*** (**Файл**) редактора правил містить такі ж операції, що і відповідний пункт меню редактора *FIS*.

Пункт меню ***Edit*** (**Редагування**) містить наступні операції:

Undo – скасовує виконання останньої дії;

FIS Properties... – викликає редактор *FIS*;

Membership Functions... – викликає редактор функцій приналежності.

Пункт меню ***View*** (**Вид**) містить наступні операції:

Rules – викликає програму перегляду правил;

Surface – викликає програму перегляду поверхні виводу.

Пункт меню ***Options*** (**Сервіс**) містить наступні операції:

Language – дозволяє вибрати мову для запису правил у формі

тексту: ***English*** (англійська), ***Deutsch*** (німецька) або ***Francais*** (французька);

Format – формат запису правил системи нечіткого виводу: ***Verbose*** (в формі тексту), ***Symbolic*** (в символічній формі) або ***Indexed*** (в цифровій формі).

Поля введення в середній частині графічного інтерфейсу редактора правил дозволяють задати нове правило в системі нечіткого виводу. Для цього необхідно виділити ім'я відповідної змінної, яка повинна бути попередньо визначена за допомогою редактора функцій приналежності. Якщо деяка змінна не входить в правило, то для неї слід вибрати значення ***none***. Якщо в умові правила використовується логічне заперечення, то слід зазначити відповідний прапорець з міткою ***not***.

Редактор правил дозволяє також задати логічні зв'язки для підумов правила (перемикач ***Connection***) і вагу правила (поле введення ***Weight***). Кнопки в нижній частині графічного інтерфейсу редактора правил, як випливає з їх назв, служать для видалення виділеного у вікні правила (***Delete rule***), додавання створеного правила в систему (***Add rule***) і внесення змін в виділене в вікні правило (***Change rule***). У правому нижньому кутку знаходяться кнопки виклику вбудованої довідкової системи *Matlab* (***Help***) і кнопка закриття редактора правил (***Close***).

Програма перегляду поверхні системи нечіткого виводу

Програма перегляду поверхні системи нечіткого виводу дозволяє переглядати поверхню системи нечіткого виводу і візуалізувати графіки залежності вихідних змінних від окремих вхідних змінних. Графічний інтерфейс програми перегляду правил може бути відкритий за допомогою головного меню редактора *FIS*, редактора функцій приналежності або редактора правил командою меню ***View*** → ***Surface*** або натисканням комбінації клавіш [***Ctrl*+6**].

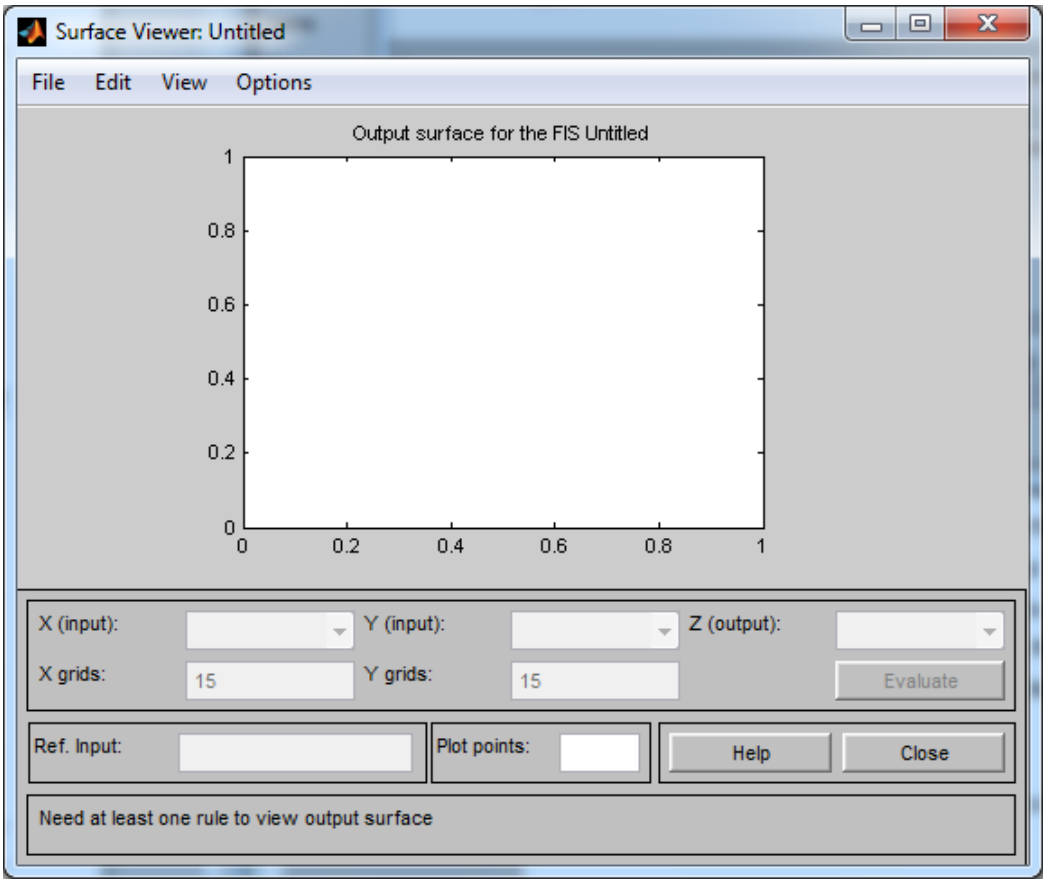


Рисунок 4.4 – Вікно програми перегляду поверхні

Програма перегляду поверхні виводу має головне меню, яке дозволяє користувачеві викликати інші графічні засоби роботи з системою нечіткого виводу *FIS*, завантажувати і зберігати структуру *FIS* в зовнішніх файлах і т. д.

Пункт меню ***File*** (**Файл**) програми перегляду поверхні містить такі ж операції, що і відповідний пункт меню редактора *FIS*.

Пункт меню ***Edit*** (**Редагування**) містить наступні операції:

Undo – скасовує виконання останньої дії;

FIS Properties... – викликає редактор *FIS*;

Membership Functions... – викликає редактор функцій приналежності;

Rules... – викликає програму редагування правил.

Пункт меню ***View*** (**Вид**) містить наступну операцію:

Rules – викликає програму перегляду правил.

Пункт меню ***Options*** (**Сервіс**) містить наступні операції:

Plot – дозволяє вибрати один з 8 стилів зображення графіка поверхні виводу;

Color Map – дозволяє вибрати одну з 4 колірних схем зобра-

ження графіка поверхні виводу;

Always evaluate – позначка галочкою цього пункту вкладеного меню призводить до автоматичного формування нової поверхні виведення щоразу, коли вносяться зміни в систему нечіткого виводу, що впливають на форму графіка поверхні виводу (такі як зміна кількості точок сітки графіка). Це значення прийнято за замовчуванням. Для його скасування необхідно зняти галочку у цього пункту вкладеного меню, клацнувши на цій позиції меню.

Програма перегляду поверхні виводу не дозволяє вносити зміни в систему нечіткого виводу і відповідну їй структуру *FIS*. Використовуючи головне меню програми, користувач може вибрати вхідні змінні і відповідні їм горизонтальні осі системи координат (*X* і *Y*), а також вихідну змінну, якій відповідає вертикальна вісь системи координат (*Z*).

Натиснувши і утримуючи ліву кнопку миші на осях графіка поверхні, за допомогою подальшого переміщення курсора миші в тому чи іншому напрямку можна змінити кут перегляду поверхні виводу. Якщо розглядається система нечіткого виводу з більш ніж двома вхідними змінними, то для невізуалізованих вхідних змінних слід задати деякі постійні значення (константи).

Базові відомості про штучні нейронні мережі

На рис. 4.5 наведений типовий приклад мережі з прямою передачею сигналу. Нейрони регулярним чином організовані в шари. Вхідний шар служить просто для введення значень вхідних змінних. Кожен з прихованих і вихідних нейронів з’єднаний з усіма елементами попереднього шару.

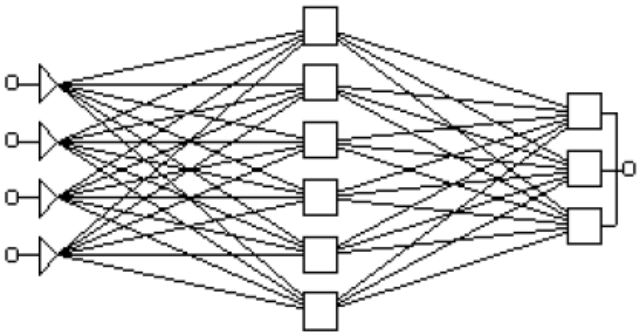


Рисунок 4.5 – Типовий приклад мережі з прямою передачею сигналу

При роботі (використанні) мережі на вхідні елементи подаються значення вхідних змінних, потім сигнали послідовно відпрацьовують нейрони проміжних і вихідного шарів. Кожен з них обчислює своє значення активації, беручи зважену суму виходів елементів попереднього шару і віднімаючи з неї порогове значення. Потім значення активації перетворюється за допомогою функції активації, і в результаті виходить вихід нейрона. Після того, як вся мережа відпрацює, вихідні значення елементів вихідного шару приймаються за вихід всієї мережі в цілому.

Спочатку мережа налаштовується. Процес налаштування мережі отримав назву "навчання мережі". Перед початком навчання зв'язкам присвоюються невеликі випадкові значення. Кожна ітерація процедури складається з двох фаз. Під час першої фази на мережу подається вхідний вектор шляхом установки в потрібне стан вхідних елементів. Потім вхідні сигнали поширюються по мережі, породжуючи деякий вихідний вектор. Для роботи алгоритму потрібно, щоб характеристика вхід-вихід нейроподібних елементів не була спадною і мала обмежену похідну. Зазвичай для цього використовують сигмоидну нелінійність.

Робочі завдання

Завдання 1. Розглянемо побудову експертної системи на прикладі задачі про чайові. Завдання може бути описане наступними виразами:

1. Якщо обслуговування погане або їжа підгоріла, то чайові - малі.
2. Якщо обслуговування добре, то чайові – середні.
3. Якщо обслуговування відмінне або їжа чудова, то чайові – щедрі.

Якість обслуговування та їжі будемо оцінювати за 10-бальною шкалою (0 – найгірша оцінка, 10 – найкраща). Припустимо, що малі чайові складають 5% вартості обіду, середні – 15%, щедрі – близько 25%.

Цій інформації досить для проектування нечіткої експертної системи. Ця система буде мати 2 входи – «сервіс» і «їжа», один вихід – «чаевые», три правила типу «ЯКЩО ..., ТО ...» (відповідно

до наведених вище) і по 3 значення (відповідно 0 балів, 5 і 10 балів, 5%, 15%, 25%) для центрів функцій приналежності входів і виходу. Побудуємо систему, використовуючи алгоритм виводу Мамдані.

Для цього необхідно виконати наступне:

1.1. Створення системи нечіткого виводу в редакторі *FIS*.

Командою ***fuzzy*** запускаємо *FIS*-редактор. За замовчуванням, вихідний алгоритм виводу – типу *Mamdani* (про що говорить напис в центральному білому блоці) і тут ніяких змін не потрібно.

Так як в системі повинно бути два входи, через пункт меню ***Edit → Add Variable → Input*** додаємо в систему другий вхід (у вікні редактора з'являється другий жовтий блок з ім'ям ***input*2**).

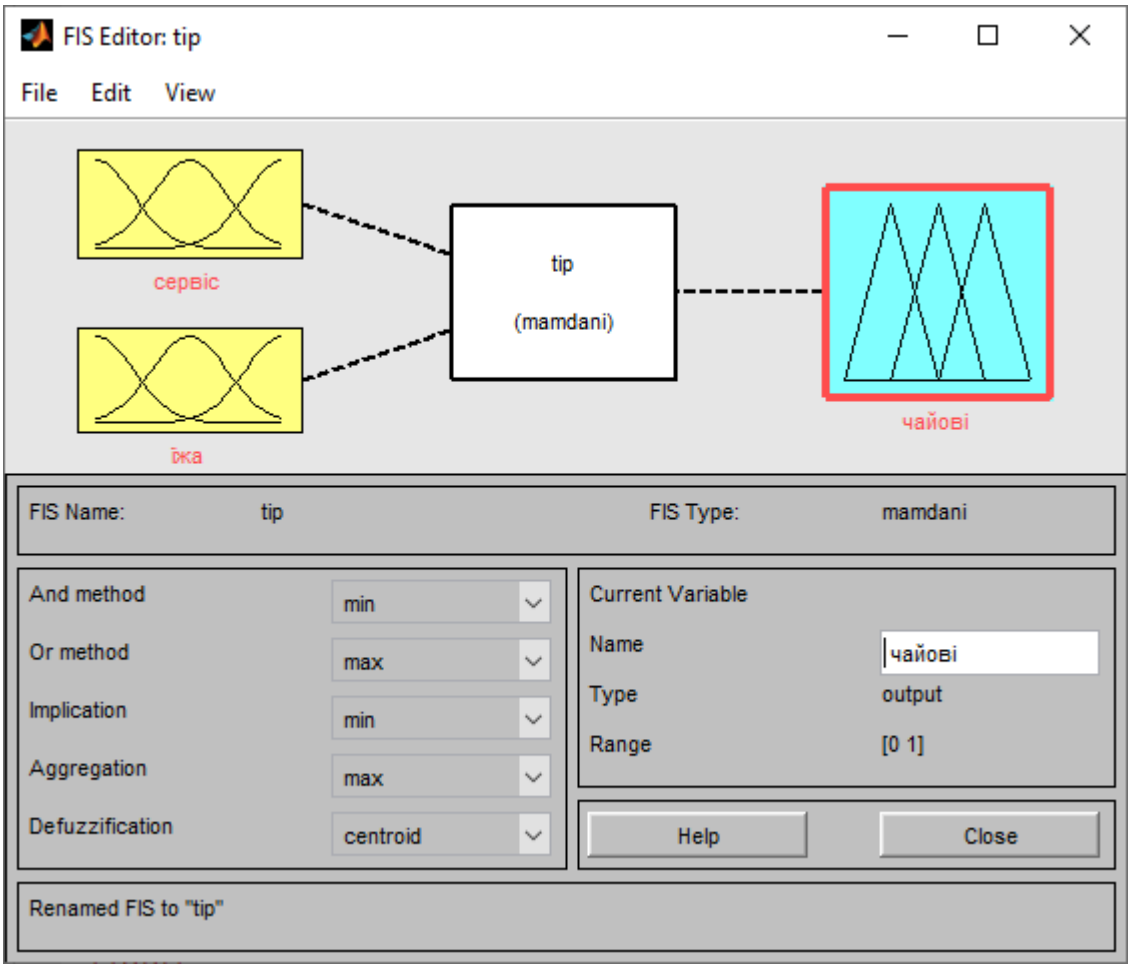


Рисунок 4.6 – Вікно *FIS*-редактора після завдання імен

Далі задамо нові імена блокам і створюваній системі. Роблячи одноразове клацання лівою кнопкою миші на блоці ***input*1**, в полі імені його ім'я змінюємо на **сервіс**. Завершується введення нового імені натисканням клавіші *Enter*. Аналогічним чином встанов-

люється ім’я **їжа** блоку ***input*2** і **чайові** – вихідного блоку ***output*1** (справа вгорі). Через пункт меню ***File*** → ***Export*** → ***To Workspace...*** (**Зберегти в робочому просторі...**) присвоюємо ім’я всій системі, наприклад, ***tip*** (по-англійськи – чайові).

1.2. Завдання функцій приналежності змінних.

Завдання і редагування функцій приналежності почнемо зі змінній **сервис**. Для цього відкриємо редактор функцій приналежності подвійним клацанням лівої кнопки миші по іконці **сервіс**. Далі в полях ***Range*** і ***Display Range*** встановимо діапазон зміни і відображення цієї змінної – від 0 до 10 (балів) і підтвердимо клавішею *Enter*. Потім через пункт меню ***Edit*** → ***Remov All MF*** видалимо всі функції приналежності, задані за замовчуванням, а за допомогою команди ***Edit*** → ***Add MF*** задамо нові функції приналежності типу гауса (*gaussmf*) із загальним числом 3. Після натискання кнопки ***OK*** відбудеться повернення у вікно редактора функцій приналежності.

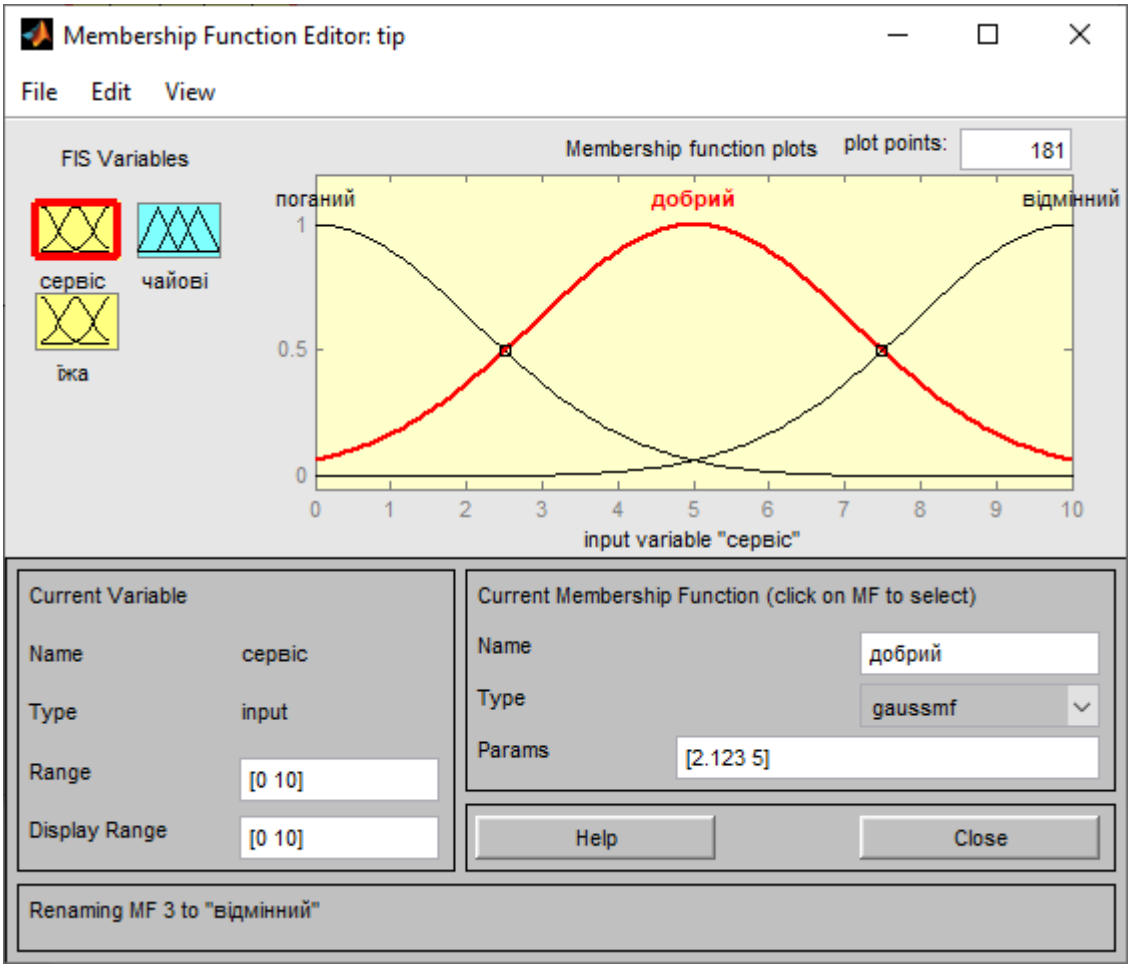


Рисунок 4.7 – Вікно редактора функцій приналежності із заданими функціями приналежності для змінної **сервіс**

У вікні редактора функцій приналежності скорегуємо розмах і положення заданих функцій, а також змінимо їх імена на **поганий**, **добрий** і **відмінний**.

Клацанням лівої кнопки миші по іконці **їжа** переходимо у вікно редагування функцій приналежності для змінної **їжа**. Спочатку ставимо діапазон зміни цієї змінної від 0 до 10, а потім, поступаючи як раніше, задаємо дві трапецієподібні функції приналежності (*trampf*) з параметрами [0 0 1 3] і [7 9 10 10] відповідно і іменами **підгоріла** та **чудова**.

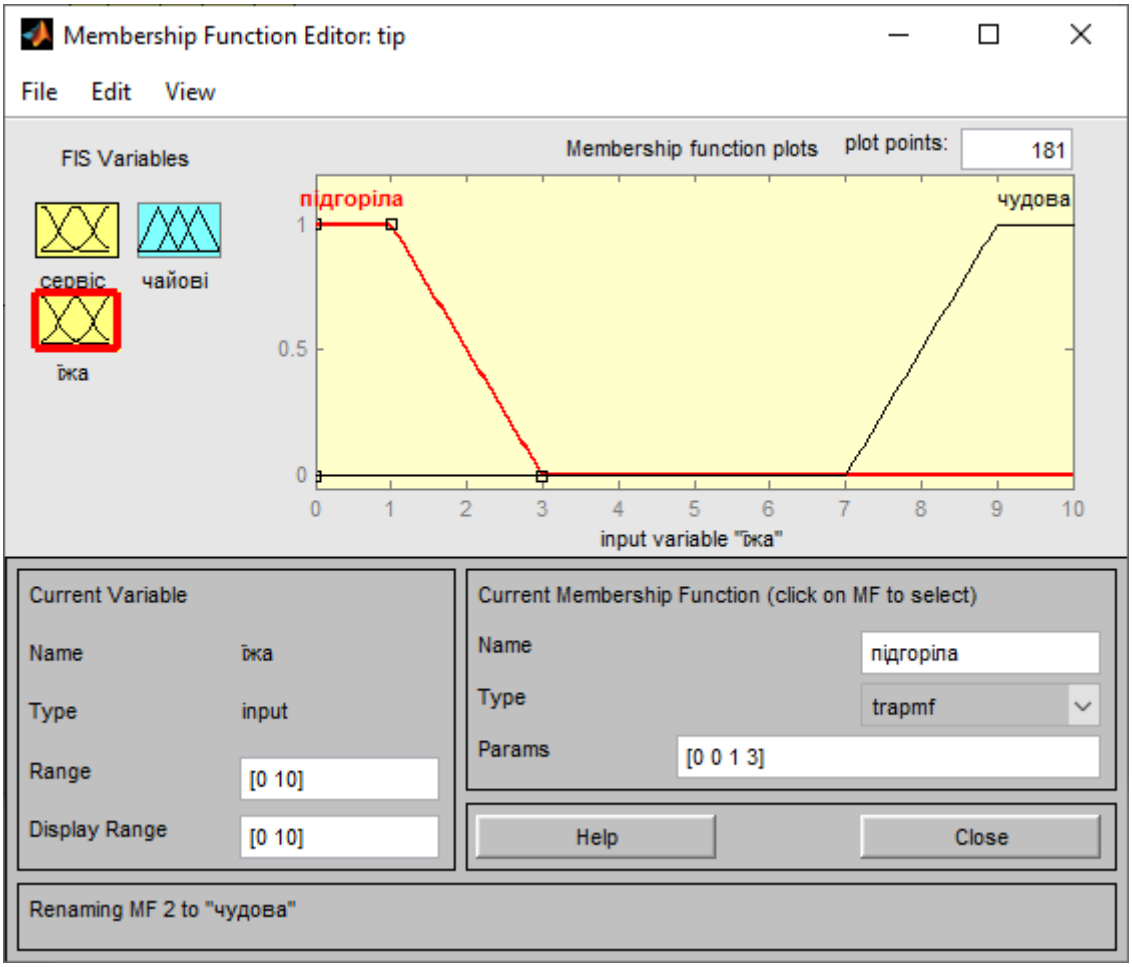


Рисунок 4.8 - Вікно редактора функцій приналежності з заданими функціями приналежності для змінної **їжа**

Для вихідної змінної **чайові** вказуємо спочатку діапазон зміни від 0 до 30, потім задаємо три функції приналежності трикутної форми (*trimf*) з іменами **малі**, **середні**, **щедрі**.

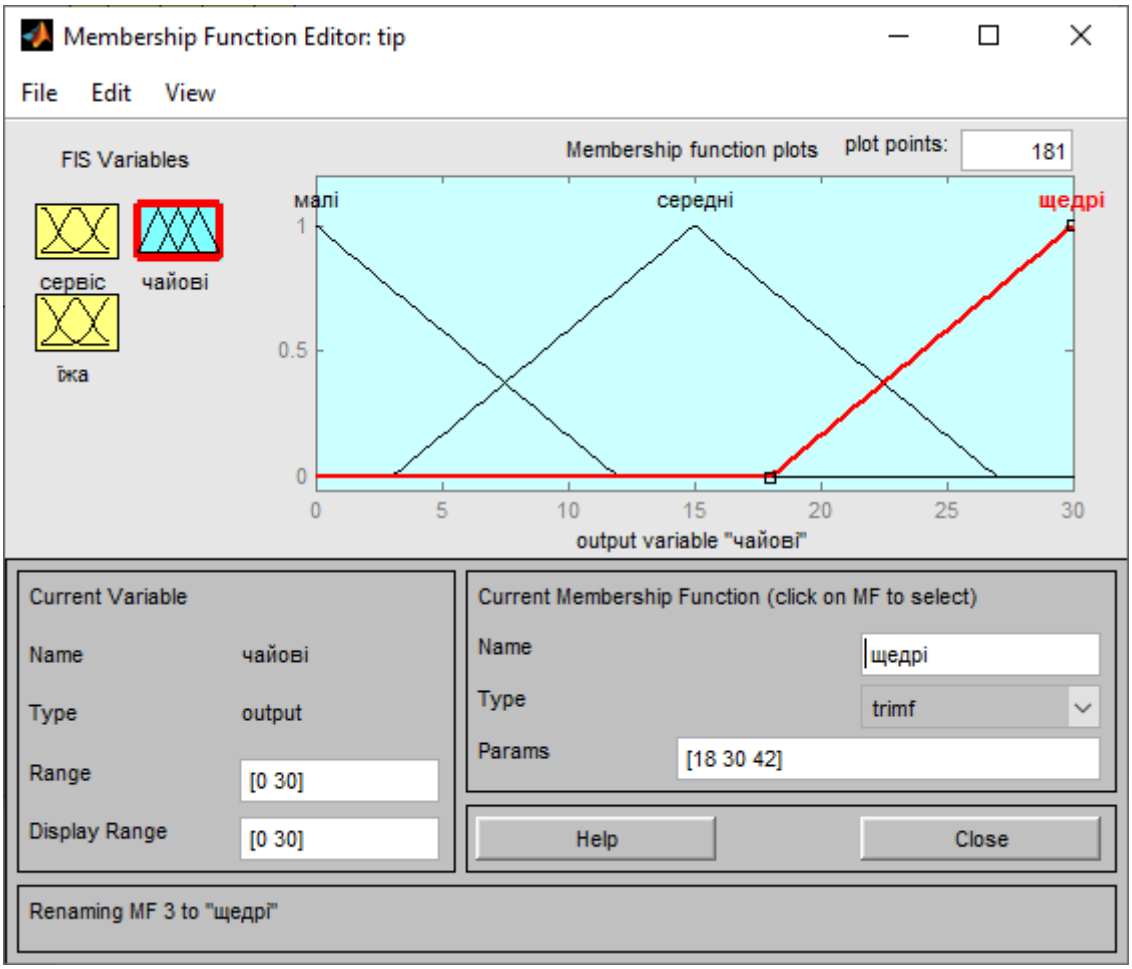


Рисунок 4.9 – Вікно редактора функцій приналежності з заданими функціями приналежності для змінної **чайові**

Після того, як всі функції приналежності визначені, закриваємо вікно редактора функцій приналежності.

1.3. Конструювання правил.

Запускаємо редактор правил системи нечіткого виводу за допомогою команди ***Edit*** → ***Rules...*** з вікна редактора *FIS*. Підсумковий набір правил для даної системи виглядає наступним чином:

1. If (сервіс is поганий) or (їжа is підгоріла) then (чайові is малі) (1)
2. If (сервіс is добрий) then (чайові is середні) (1)
3. If (сервіс is відмінний) or (їжа is чудова) then (чайові is щедрі) (1)

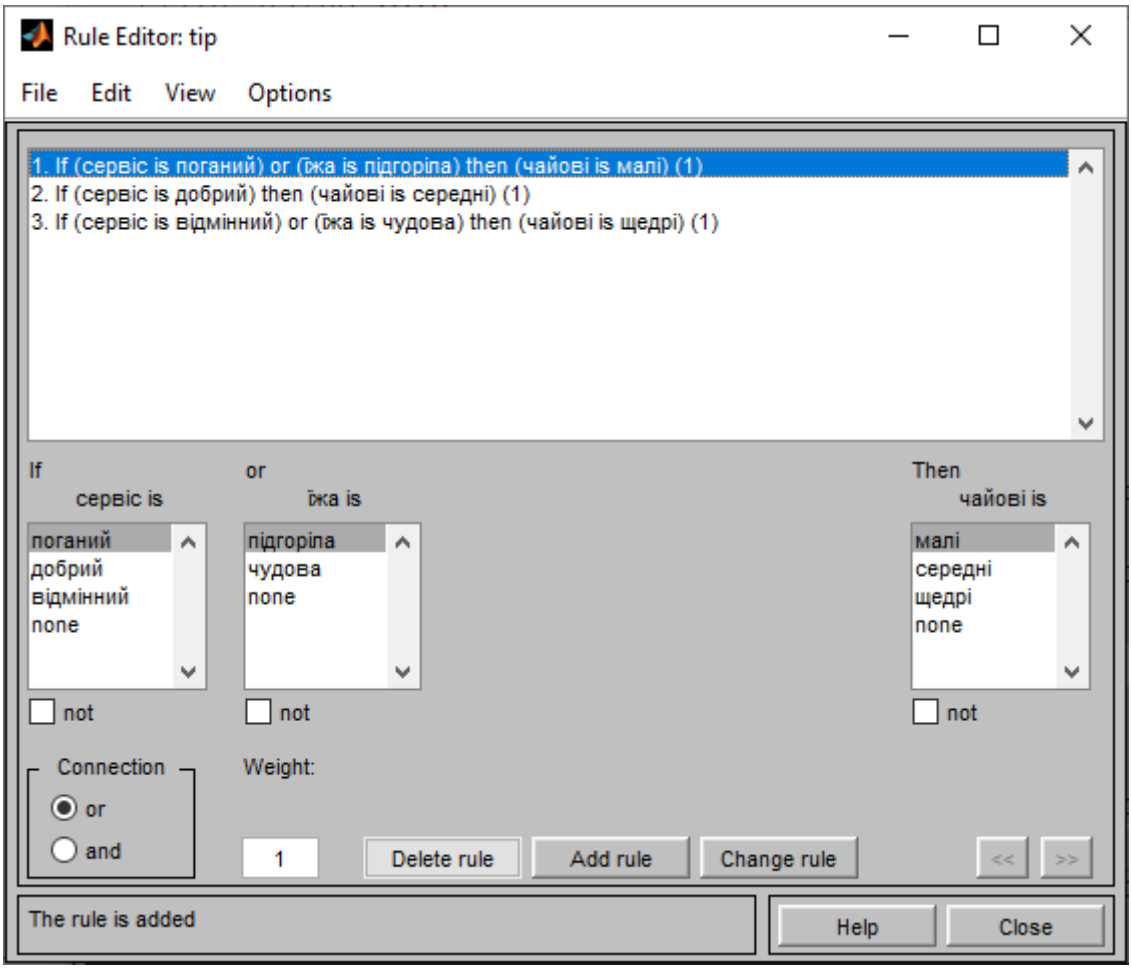


Рисунок 4.10 – Вікно редактора правил для даної системи

Одиниця в дужках після кожного правила вказує його вагу, тобто значимість правила. Дану вагу можна міняти, використовуючи відповідне поле в лівій нижній частині вікна редактора правил. На цьому конструювання експертної системи закінчено. Зберігаємо її в файл на диску під ім'ям ***tip***.

1.4. Перевірка роботи системи нечіткого виводу.

Для перевірки роботи створеної системи потрібно відкрити через пункт меню ***View*** → ***Rules...*** вікно перегляду правил і встановити значення змінних: **сервіс=0** (тобто нікудишній), **їжа=10** (тобто чудова). Отримана відповідь: **чайові=15** (тобто середні). Можна перевірити й інші варіанти.

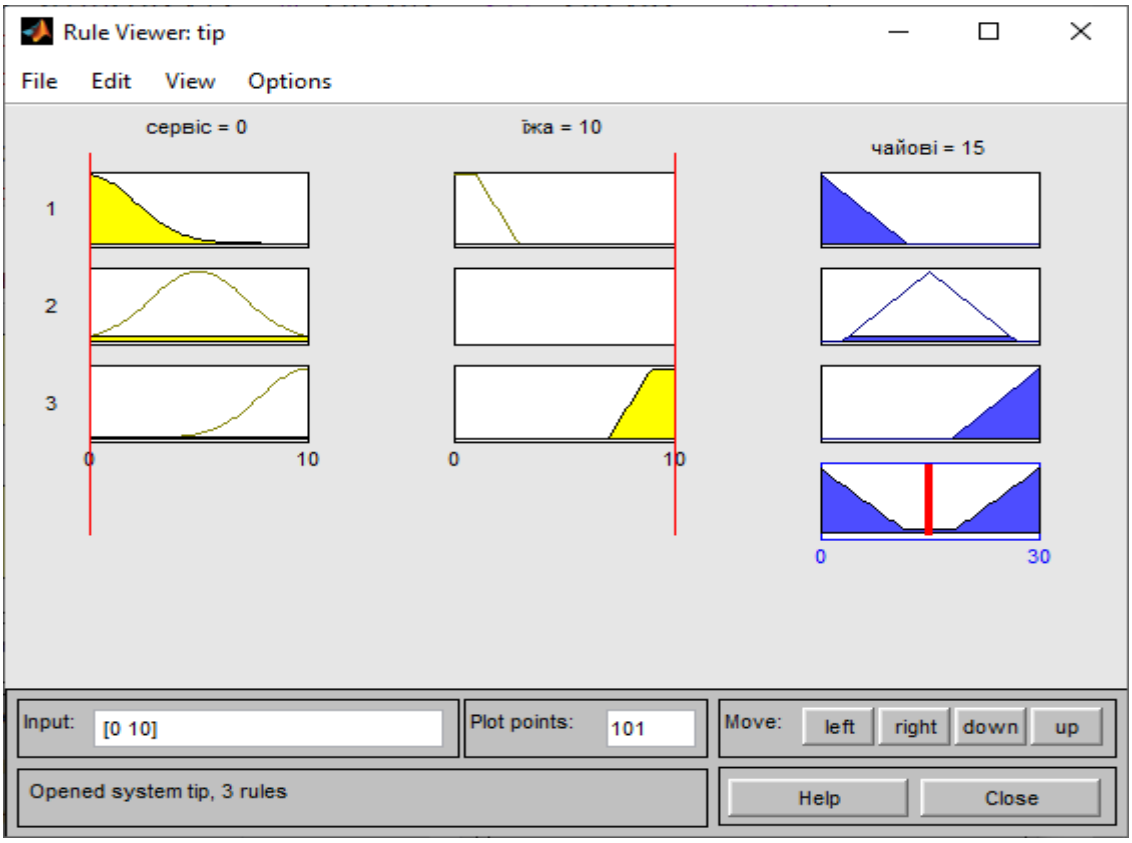


Рисунок 4.11 – Вікно перегляду правил даної системи

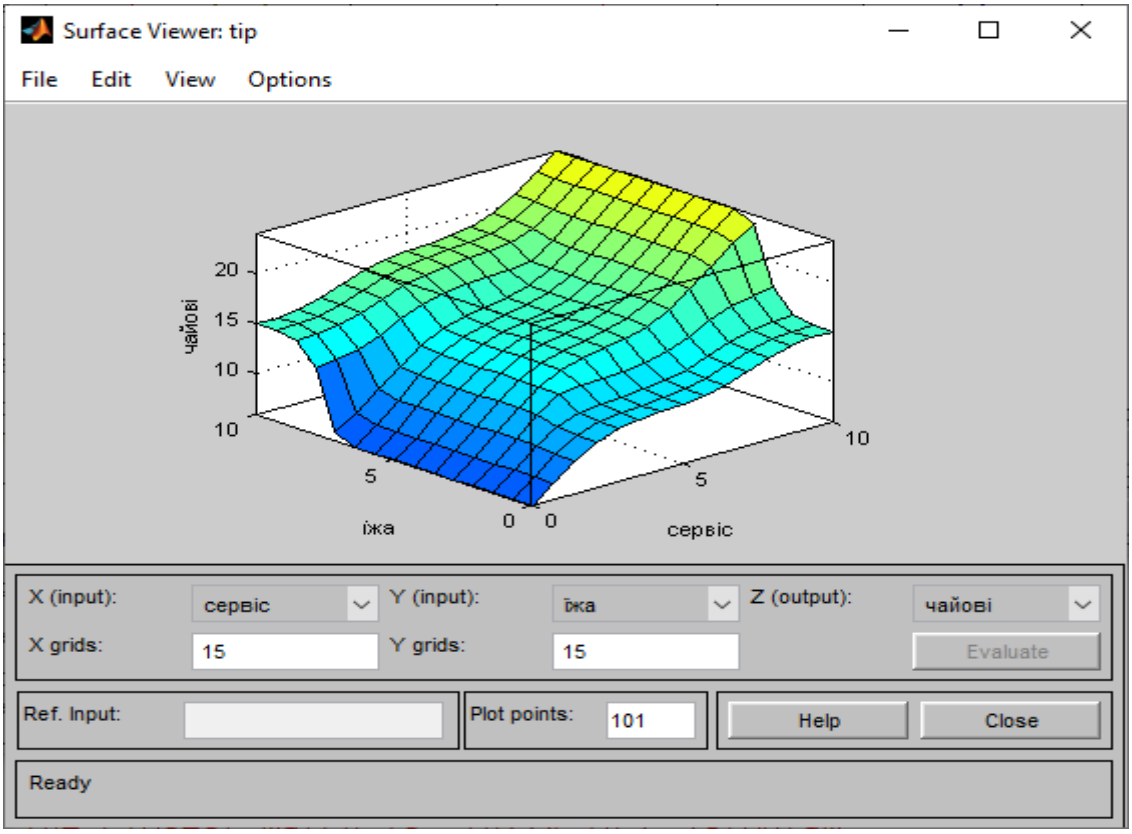


Рисунок 4.12 – Вікно перегляду поверхні даної системи

Підтвердженням зазначеної залежності вихідної змінної від вхідних може служити вид поверхні відгуку, який видається при виборі пункту меню ***View*** → ***Surface***; за допомогою мишки графік можна повертати в різні боки.

Завдання 2. Сконструюємо нечітку систему, яка буде показувати залежність між змінними x і y ($y = x^2$), задану за допомогою табл. 4.1.

Таблиця 4.1 – **Залежність x від y**

x	-1.00	-0.60	0.00	0.40	1.00
y	1.00	0.36	0.00	0.16	1.00

Для цього необхідно виконати наступне:

2.1. Відкриємо нове вікно *FIS*-редактора за допомогою пункту меню ***File*** → ***New FIS*** → ***Sugeno*** (нова система типу *Sugeno*), при цьому в блоці, який відображається білим квадратом у верхній частині вікна редактора з'явиться напис ***Untitled* (*sugeno*)**.

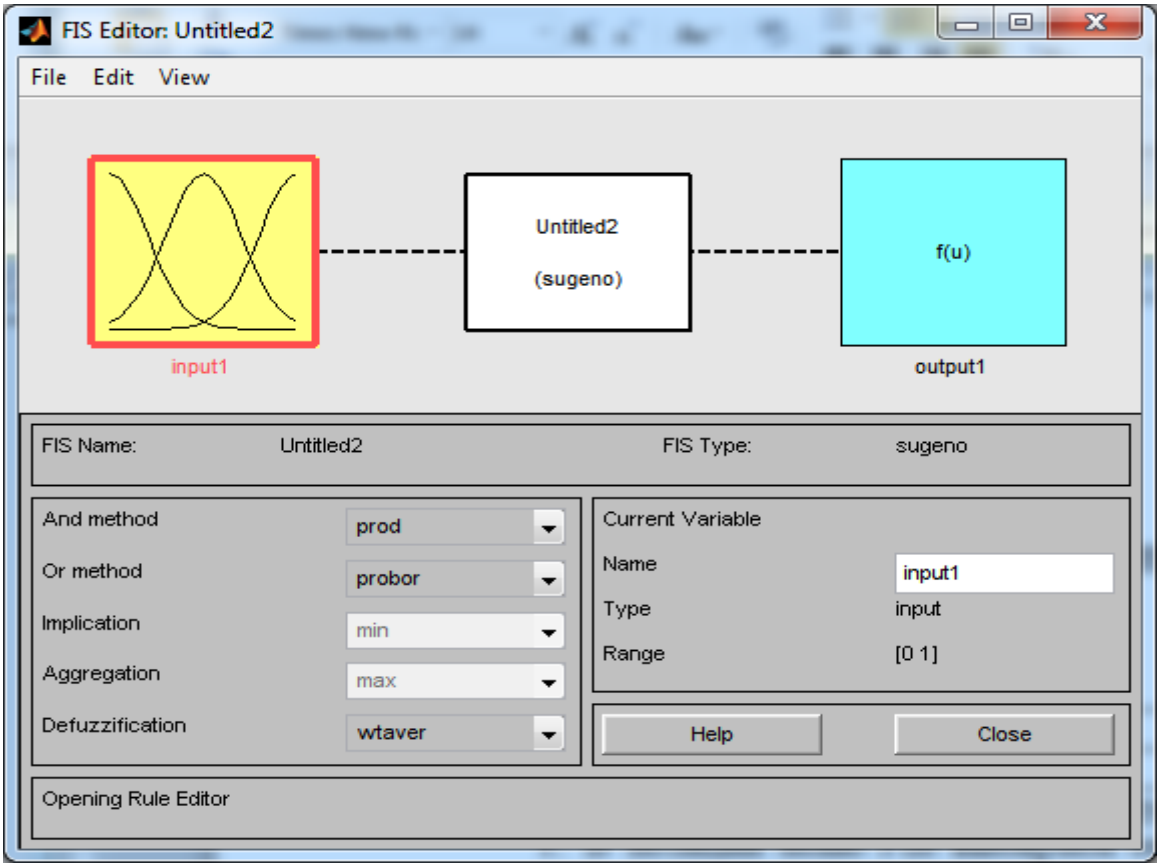


Рисунок 4.13 – Вікно *FIS*-редактора для створення системи нечіткого виводу типу Сугено

2.2. У вікні *FIS*-редактора задаємо ім’я входу ***x***, ім’я виходу – ***y***.

2.3. У редакторі функцій приналежності для вхідної змінної **x** виберемо гаусові функції приналежності *(gaussmf)*, а їх кількість задаємо рівним п’яти – по числу значень аргументу.

2.4. Встановимо діапазон зміни змінної ***x*** від -1 до 1.

2.5. Перейменуємо функції приналежності наступним чином:

*mf*1 – *bn*;

*mf*2 – *n*;

*mf*3 – *z*;

*mf*4 – *p*;

*mf*5 – *bp*.

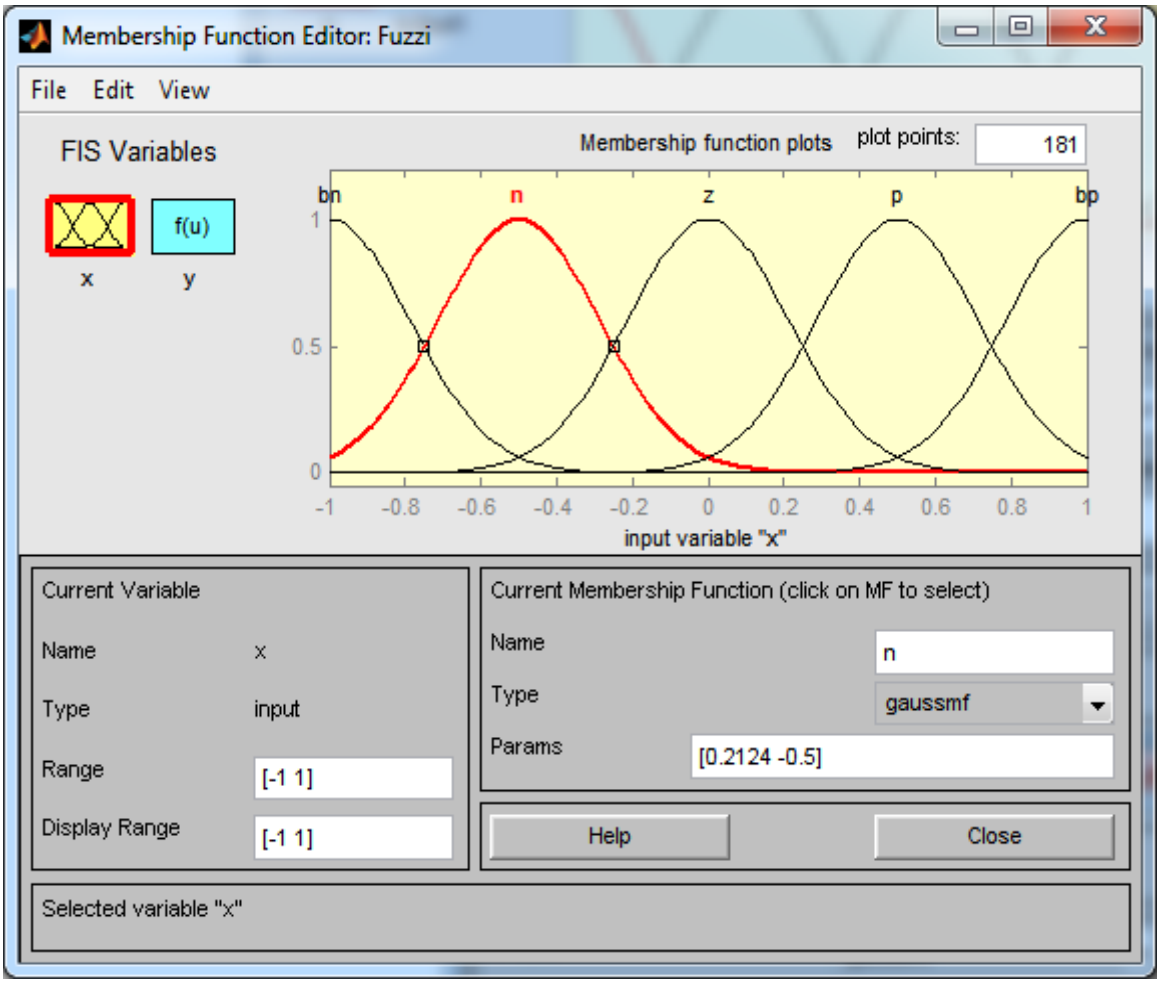


Рисунок 4.14 – Вікно редактора функцій приналежності з заданими функціями приналежності для змінної ***x***

2.6. Для вихідної змінної **y** виберемо постійні функції приналежності (*constant*) із загальним числом 4.

2.7. Змінимо імена вихідних функцій приналежності, задавши їх як відповідні числові значення ***y***, тобто 0, 0.16, 0.36, 1. Одночасно ці ж числові значення введемо в поле ***Params*** і закриємо вікно редактора функцій приналежності.

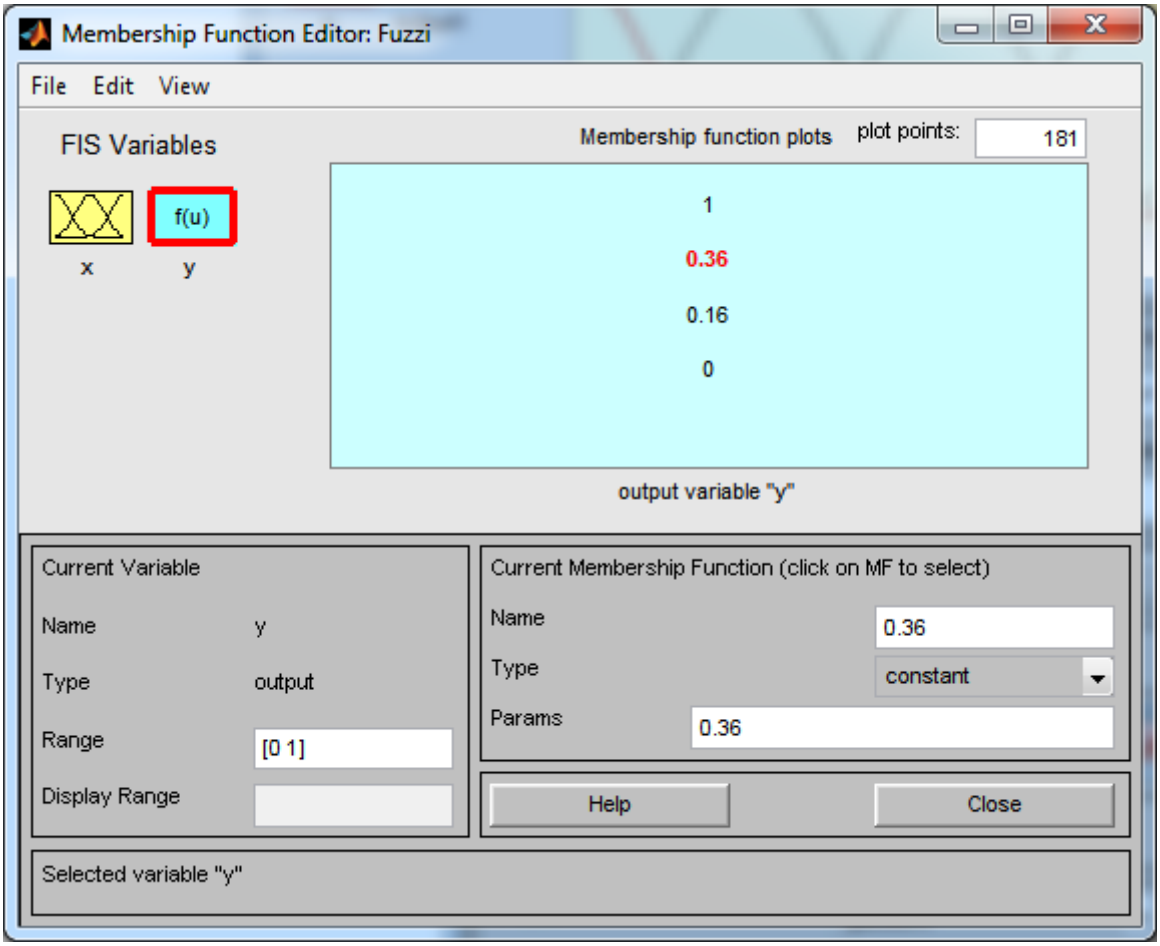


Рисунок 4.15 – Вікно редактора функцій приналежності з заданими функціями приналежності для змінної ***y***

2.8. У вікні редактора правил введемо такі правила:

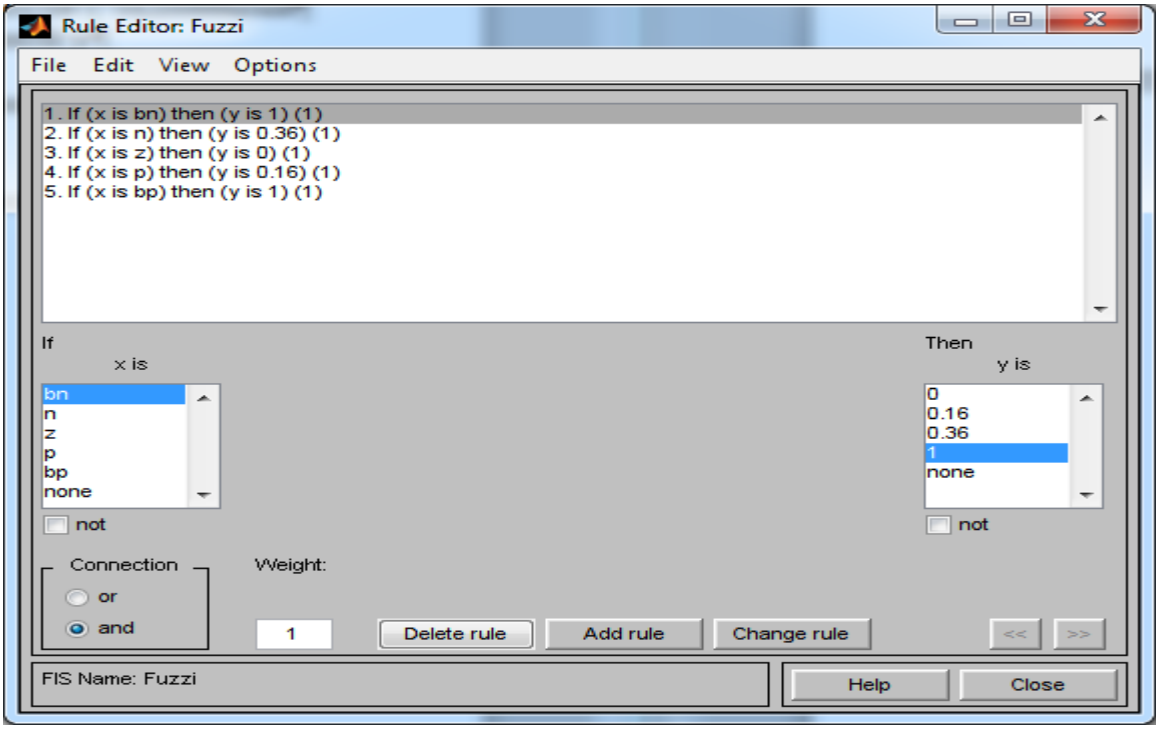


Рисунок 4.16 – Вікно редактора правил для даної системи

2.9. Збережемо побудовану систему в файл з ім'ям ***fuzzy***.

2.10. Перевіримо роботу системи. Для цього у вікні перегляду правил в поле ***Input*** задаємо ***x*=0.5** і натиснемо *Enter*. Значення ***y*** при цьому стане дорівнювати 0.198.

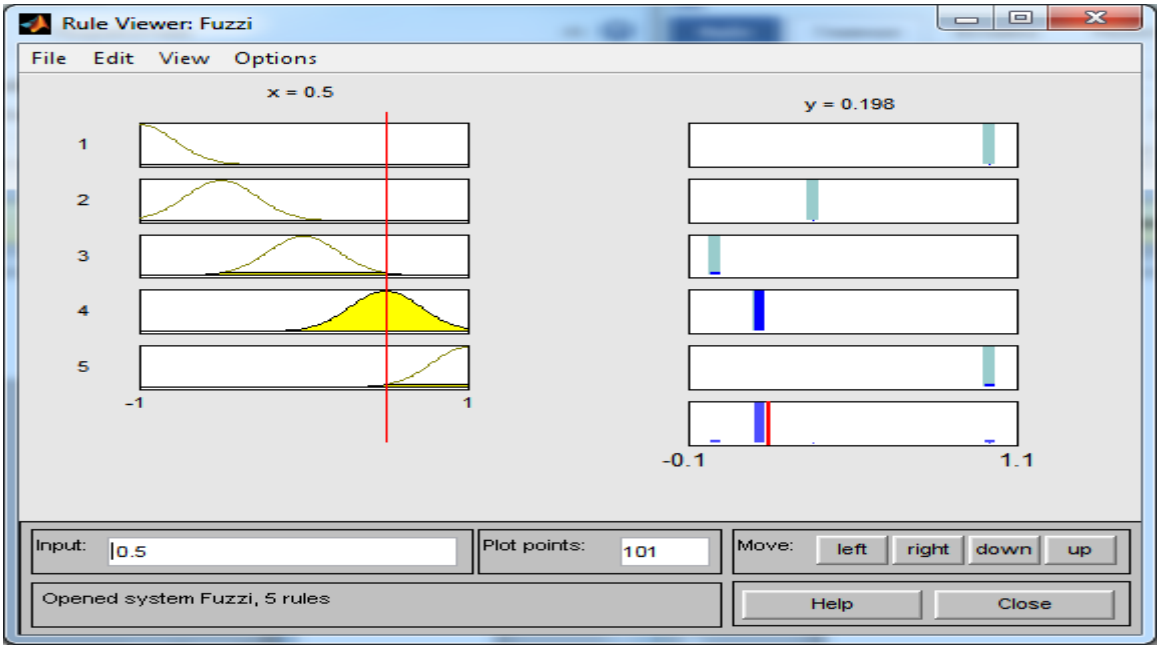


Рисунок 4.17 – Вікно перегляду правил даної системи

2.11. У вікні перегляду поверхні перевіримо залежність вихідної змінної від вхідної.

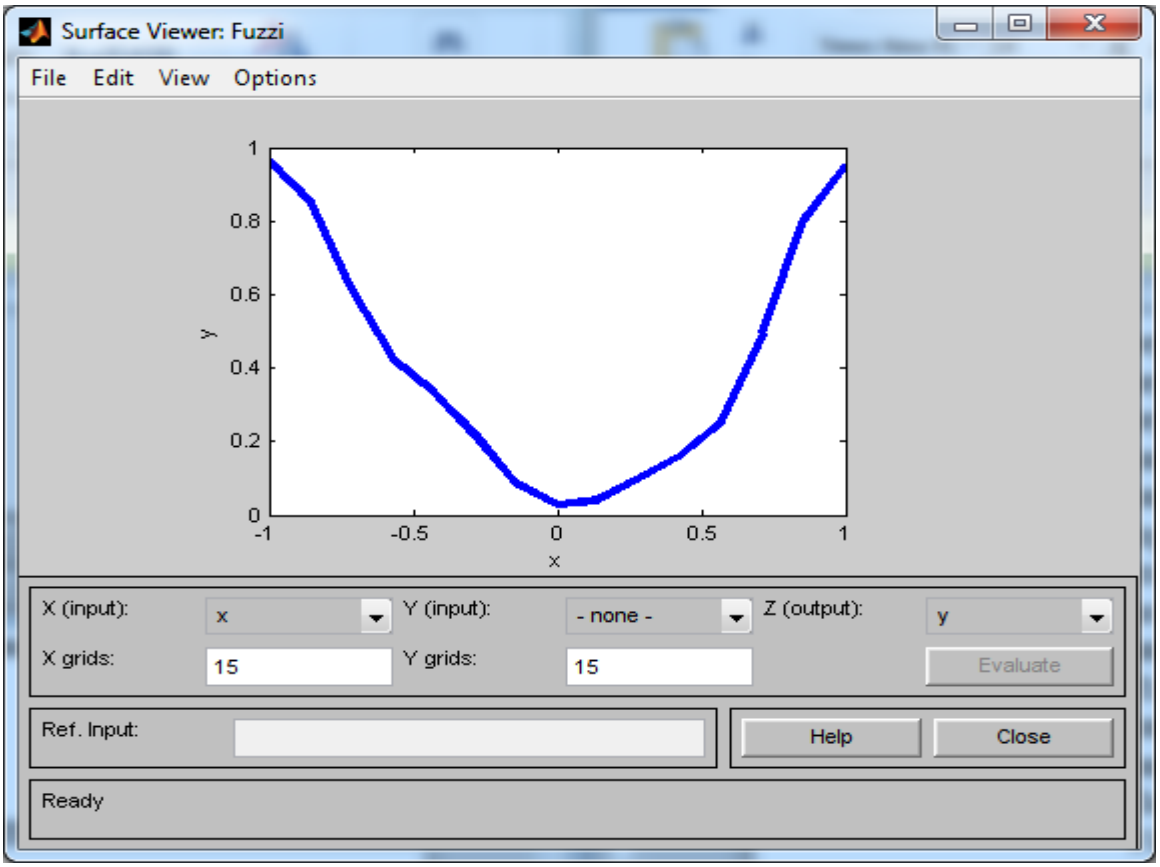


Рисунок 4.18 – Вікно перегляду поверхні даної системи

Таким чином, за допомогою побудованої моделі можна вирішувати задачу інтерполяції, тобто задачу, рішення якої і було потрібно знайти.

Завдання 3 (додаткове). Створити модель руху автомобіля по трасі.

Вихідні дані для моделювання:

1. Вхідні величини – відстань від автомобіля до перешкоди (змінюється в межах від 0 до 500); швидкість автомобіля (0 – 200).

2. Вихідна величина – сила гальмування (0 – 100).

Відстань і швидкість вимірюються в лінгвістичних змінних **дуже мала** (*VS*), **мала**(*S*), **середня** (*M*), **велика** (*B*), **дуже велика** (*VB*). Сила гальмування задається лінгвістичними змінними **близька до нуля** (*Z*), **чверть** (*OQ*), **половина** (*H*), **три чверті** (*TQ*), **повна**

(*FU*). Параметри вхідних та вихідної змінних представлені в табл. 4.2 – 4.5.

Таблиця 4.2 – **Змінна Швидкість**

Ім’я змінної (Name)	Тип функції приналежності (Type)	Діапазон зміни (Params)
Дуже мала	трапецеїдальна	[0 0 20 60]
Мала	трикутна	[20 60 100]
Середня	трикутна	[60 100 140]
Велика	трикутна	[100 140 180]
Дуже велика	трапецеїдальна	[140 180 200 200]

Таблиця 4.3 – **Змінна Відстань**

Ім’я змінної (Name)	Тип функції приналежності (Type)	Діапазон зміни (Params)
Дуже мала	трапецеїдальна	[0 0 50 150]
Мала	трикутна	[50 150 250]
Середня	трикутна	[150 250 350]
Велика	трикутна	[250 350 450]
Дуже велика	трапецеїдальна	[350 450 500 500]

Таблиця 4.4 – **Змінна Сила гальмування**

Ім’я змінної (Name)	Тип функції приналежності (Type)	Діапазон зміни (Params)
Близька до нуля	трапецеїдальна	[0 0 10 30]
Чверть	трикутна	[10 30 50]
Половина	трикутна	[30 50 70]
Три чверті	трикутна	[50 70 90]
Повна	трапецеїдальна	[70 80 100 100]

Таблиця 4.5 – **База правил для управління силою гальмування автомобіля**

Швидкість / Відстань	*VS*	*S*	*M*	*B*	*VB*
VS	*H*	*TQ*	*FU*	*FU*	*FU*
S	*OQ*	*H*	*TQ*	*FU*	*FU*
M	*Z*	*OQ*	*H*	*TQ*	*FU*
B	*Z*	*Z*	*OQ*	*H*	*TQ*
VB	*Z*	*Z*	*Z*	*OQ*	*H*

Побудувати нечітку систему виводу типу Мамдані, яка дозволяє виміряти силу гальмування автомобіля, необхідну для певного гальмівного шляху при певній швидкості.

Завдання 4. Робота з нейронною мережею (НМ) в командному режимі.

Виконаємо дослідження нейронної мережі для апроксимації обраної функції, використовуючи функції пакета ***Neural Networks Toolbox***. До складу пакету входить більше 160 різних функцій, що дають можливість створювати, навчати і досліджувати нейронні мережі.

Для виконання роботи необхідно запустити *Matlab* і перейти в командне вікно.

Створимо узагальнено-регресійну НМ (мережу типу *GRNN*) з ім'ям ***a***, що реалізує функціональну залежність між входом і виходом виду $\boldsymbol{y = x^2}$ на відрізку [-1, 1], використовуючи наступні експериментальні дані:

x = [-1 -0.8 -0.5 -0.2 0 0.1 0.3 0.6 0.9 1],

y = [1 0.64 0.25 0.04 0 0.01 0.09 0.36 0.81 1].

Перевірку якості відновлення наведеної залежності здійснимо, використовуючи дані контрольної вибірки $x1$ = [-0.9 -0.7 -0.3 0.4 0.8], яким відповідають значення $y1$ = [0.81 0.49 0.09 0.16 0.64].

Процедура створення та використання даної НМ описується наступним чином:

```
>> x = [-1 -0.8 -0.5 -0.2 0 0.1 0.3 0.6 0.9 1];
>> % Завдання вхідних значень
>> y = [1 0.64 0.25 0.04 0 0.01 0.09 0.36 0.81 1];
>> % Завдання цільових значень
>> a = newgrnn(x,y,0.01); % Створення НМ з відхиленням 0.01
>> Y1 = sim(a,[-0.9 -0.7 -0.3 0.4 0.8]) % Опитування НМ
Y1 =

    0.8200    0.6400    0.0400    0.0900    0.8100
```

Як видно, точність апроксимації в даному випадку вийшла не дуже високою.

Можна спробувати поліпшити якість апроксимації за рахунок

підбору величини відхилення, але в умовах прикладу прийнятний результат легко досягається при використанні мережі з радіальними базисними елементами типу ***newrbe***:

```
>> a = newrbe (x,y);
>> Y1 = sim(a,[-0.9 -0.7 -0.3 0.4 0.8]) % Опитування НМ
Y1 =

    0.8100    0.4900    0.0900    0.1600    0.6400
```

Розглянемо тепер задачу відновлення деякої, взагалі кажучи, невідомої залежності за наявними експериментальними даними з використанням лінійної НМ.

Нехай експериментальна інформація задана значеннями x = [1.0 1.5 3.0 -1.2], y = [0.5 1.1 3.0 -1.0].

Створимо вектори входу і цілей:

```
>> x = [1.0 1.5 3.0 -1.2];
>> y = [0.5 1.1 3.0 -1.0];
```

Тепер створимо лінійну нейронну мережу:

```
>> b = newlind(x,y); % Створення лінійної НМ з ім'ям b
```

Проведемо опитування мережі для значення входу, рівного 3.0 (цьому, згідно з експериментальними даними, відповідає цільове значення 3.0):

```
>> y1 = sim(b, 3.0) % Опитування мережі
```

Похибка відновлення за даними навчальної вибірки в даному випадку 10%. Відзначимо, що в умовах як першого, так і другого прикладу дати будь-яку оцінку граничної величини похибки апроксимації неможливо, особливо для значень входів, що виходять за межі діапазону входів навчальної послідовності. На жаль, це є характерною особливістю нейромережевих моделей. Для переважної більшості завдань, що вирішуються за допомогою апарату нейронних мереж, - не тільки для задач класифікації або прогнозу - будь-яких імовірнісних оцінок точності одержуваних рішень отримати не вдається.

Завдання 5. Використання *GUI*-інтерфейсу пакета нейронних мереж.

Для цього необхідно виконати наступне:

5.1. Створення нейронної мережі.

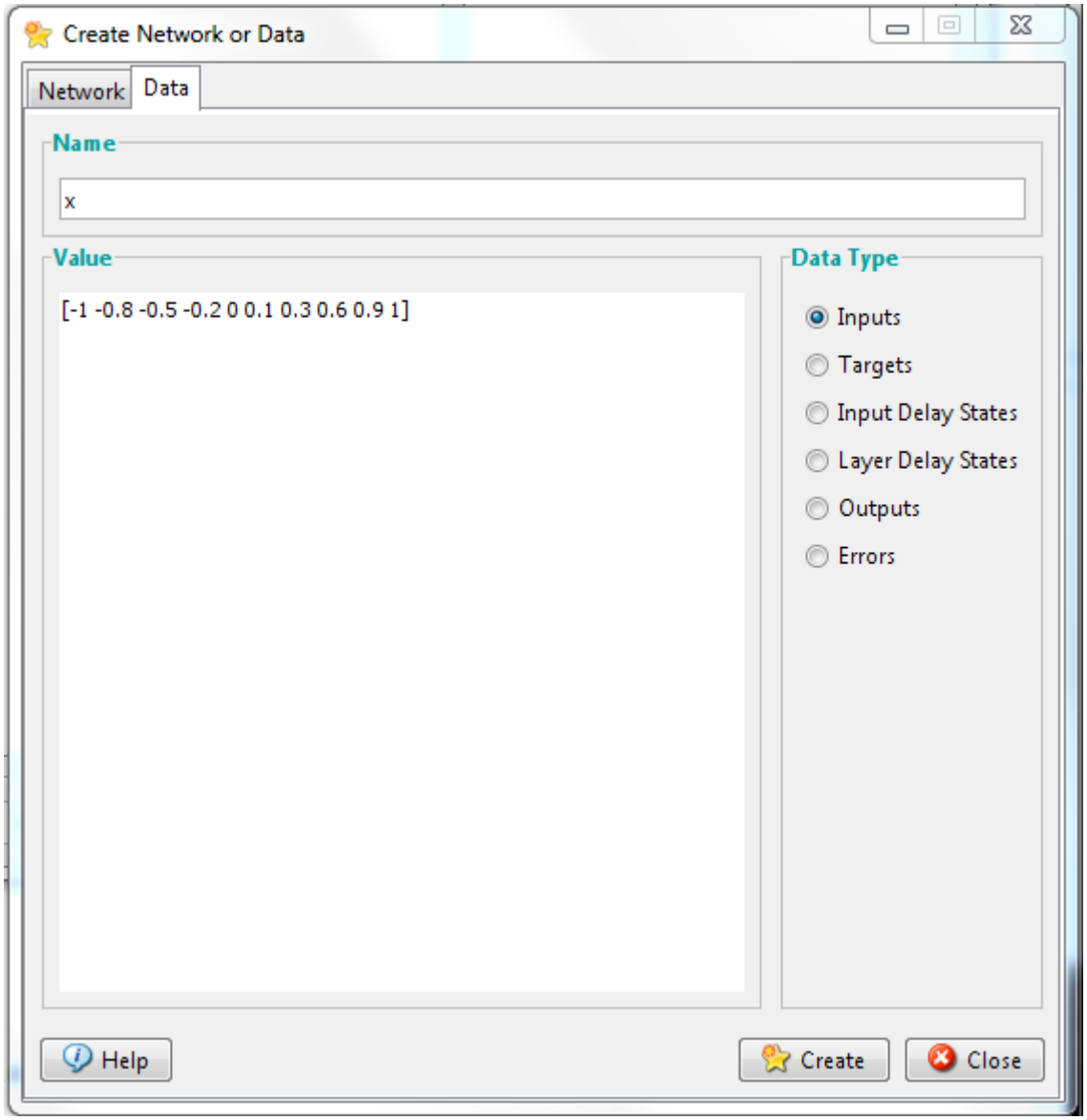


Рисунок 4.19 – Вікно *Create Network or Data*

Створимо, використовуючи графічний інтерфейс користувача, нейронну мережу для виконання операції $\boldsymbol{y} = \boldsymbol{x}^2$ при завданні векторів входу x = [-1 -0.8 -0.5 -0.2 0 0.1 0.3 0.6 0.9 1] і цілі y = [1 0.64 0.25 0.04 0 0.01 0.09 0.36 0.81 1].

Для відкриття основного вікна інтерфейсу необхідно в командному вікні *Matlab* ввести команду:

```
>> nntool
```

Виконання команди призведе до відкриття вікна створення нейронної мережі ***Network/Data Manager* (*nntool*)**.

Сформуємо послідовність входів і цілей в робочій області *GUI*-інтерфейсу, використовуючи вікно ***Create Network or Data***.

З цією метою спочатку натиснемо кнопку ***New***, далі відкриємо вкладку ***Data***, в поле **Name** введемо спочатку ім'я змінної ***x***, потім в області значень ***Value*** вектор значень [-1 - 0.8 - 0.5 - 0.2 0 0.1 0.3 0.6 0.9 1**]** і в області ***Data Type*** встановимо перемикач ***Inputs*** (вкажемо тип змінних – Входи).

Введення завершимо натисканням кнопки ***Create*** (**Створити**).

Аналогічну операцію виконаємо для вектора ***y***, із зазначенням (за допомогою перемикача ***Targets*** в області ***Data Type***), що це - вектор цільових даних.

Створимо нову нейронну мережу. Для цього перейдемо на вкладку ***Network***.

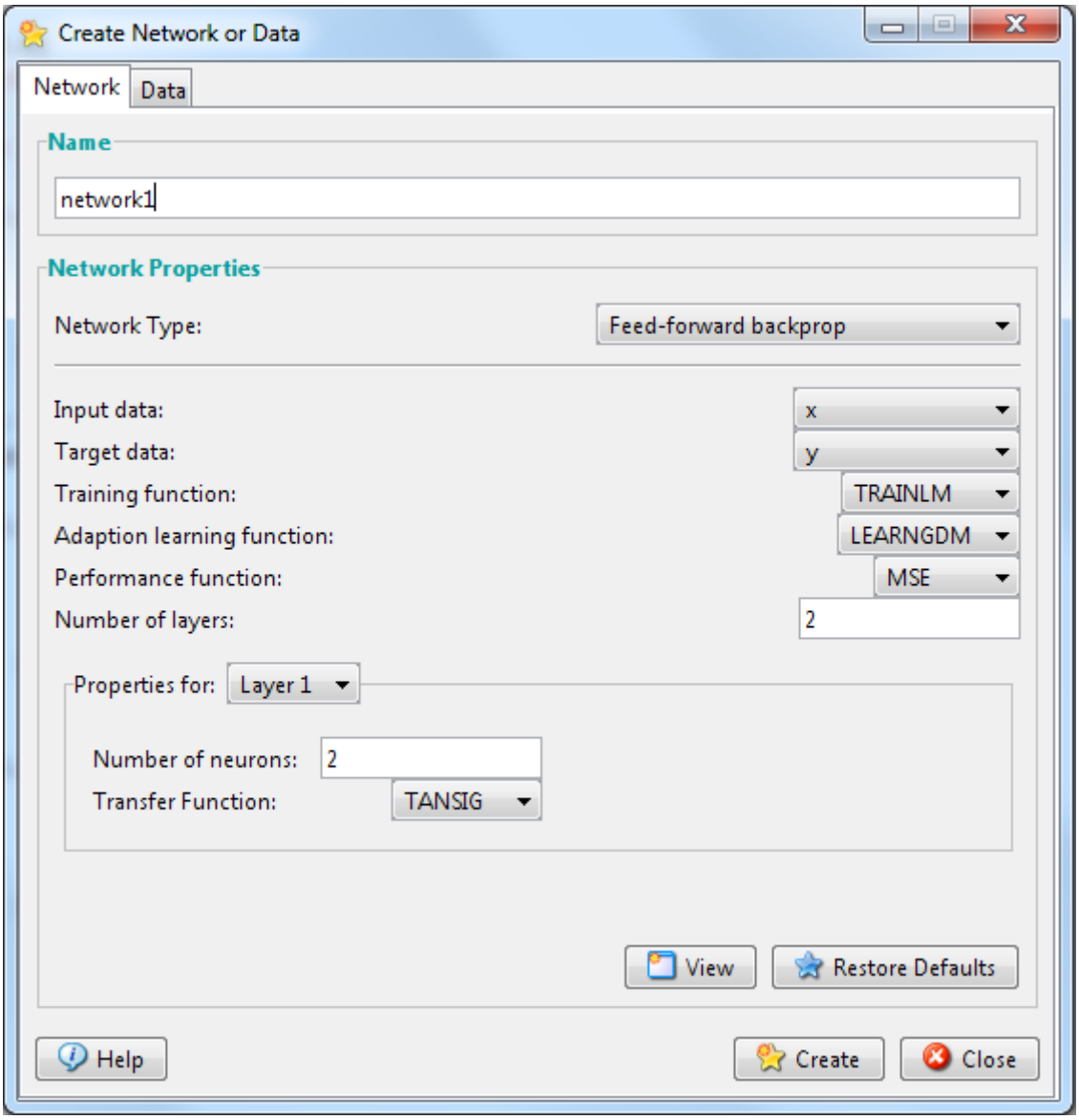


Рисунок 4.20 – Установки у вікні *Create Network or Data*

У вікні виберемо нейронну мережу типу ***feed-forward backprop*** з прямою передачею сигналу і з зворотним поширенням помилки.

При створенні мережі збережемо їй ім'я, що дається за замовчуванням (***network*1**).

Зазначимо вектор входу ***Input Data*** і вектор мети ***Target Data***.

Кількість нейронів (***Number of neurons***) першого шару (***Layer* 1**) встановимо рівним двом.

Решта установок при створенні мережі залишимо за замовчуванням.

Створення мережі завершимо натисканням кнопки ***Create***.

Після цього у вікні ***Network/Data Manager*** (***nntool***), в області ***Networks*** з'явиться ім'я нової створеної мережі - ***network*1**. Зазначимо це ім'я за допомогою мишки, що призведе до активізації всіх кнопок зазначеного вікна.

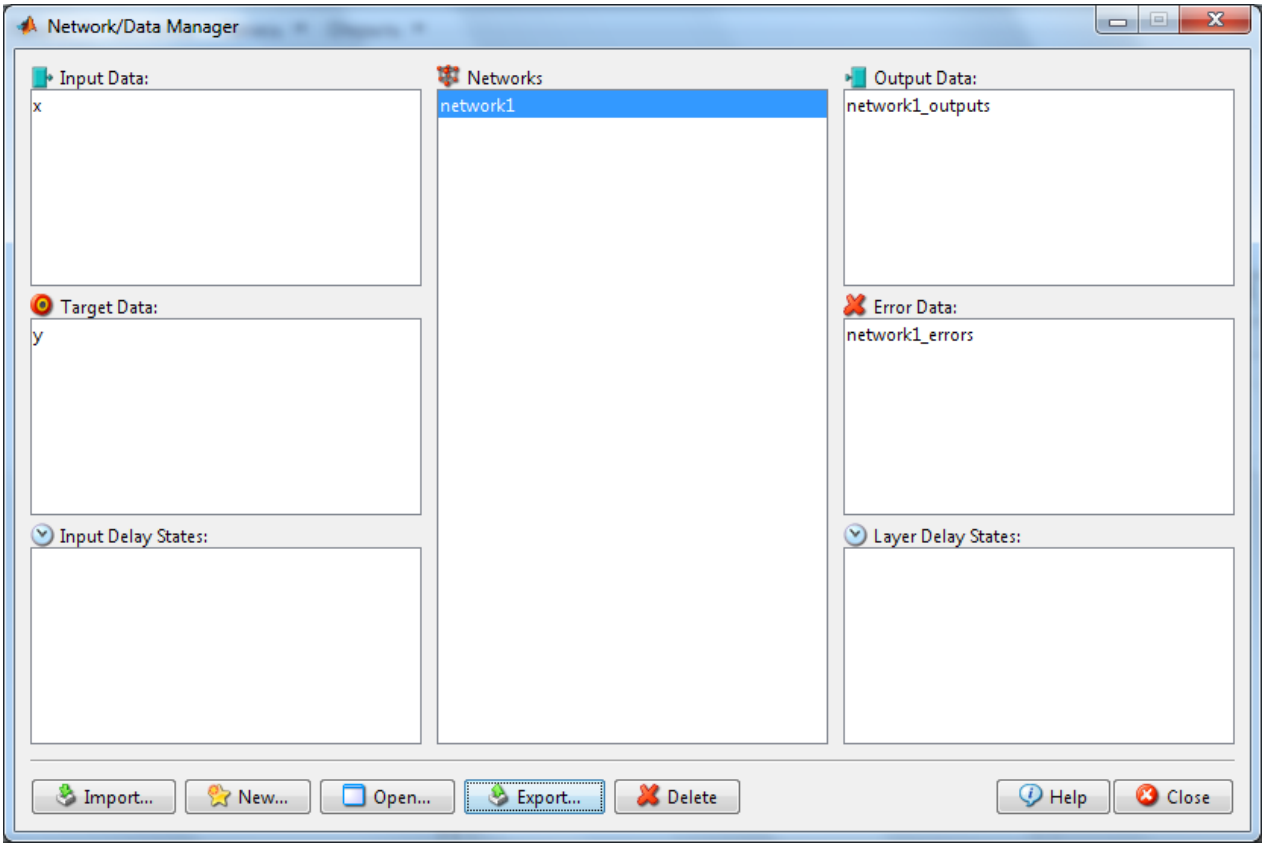


Рисунок 4.21 – Активізація всіх кнопок вікна *Network/Data Manager*

Виконаємо ініціалізацію мережі, для чого за допомогою кнопки ***Open*** відкриємо на діалоговій панелі ***Network: network*1** вкладку ***Reinitiablize Weights***.

Для введення встановлених діапазонів і ініціалізації ваг скористаємося кнопками **Set Input Ranges** (**Встановити діапазони**) та **Initialize Weights** (**Ініціалізувати ваги**). Якщо потрібно повернутися до колишніх діапазонів, слід вибрати кнопки **Revert Input Ranges** (**Повернути діапазони**) та **Revert Weights** (**Повернути ваги**), але в умовах прикладу це не потрібно.

5.2. Навчання нейронної мережі.

Для навчання створеної мережі вибирається закладка ***Train*** в панелі ***Network*: *network*1** і відкривається нова діалогова панель. Панель має дві закладки: ***Training Info*** (**Інформація про навчальні послідовності**) та ***Training Parameters*** (**Параметри навчання**).

Застосовуючи ці закладки, можна встановити імена послідовностей входу і цілі (на вкладці ***Training Info*** в лівій її частині необхідно вказати ***x*** і ***y***), а значення параметрів процедури навчання на вкладці ***Training Parameters*** в умовах прикладу збережемо за замовчуванням.

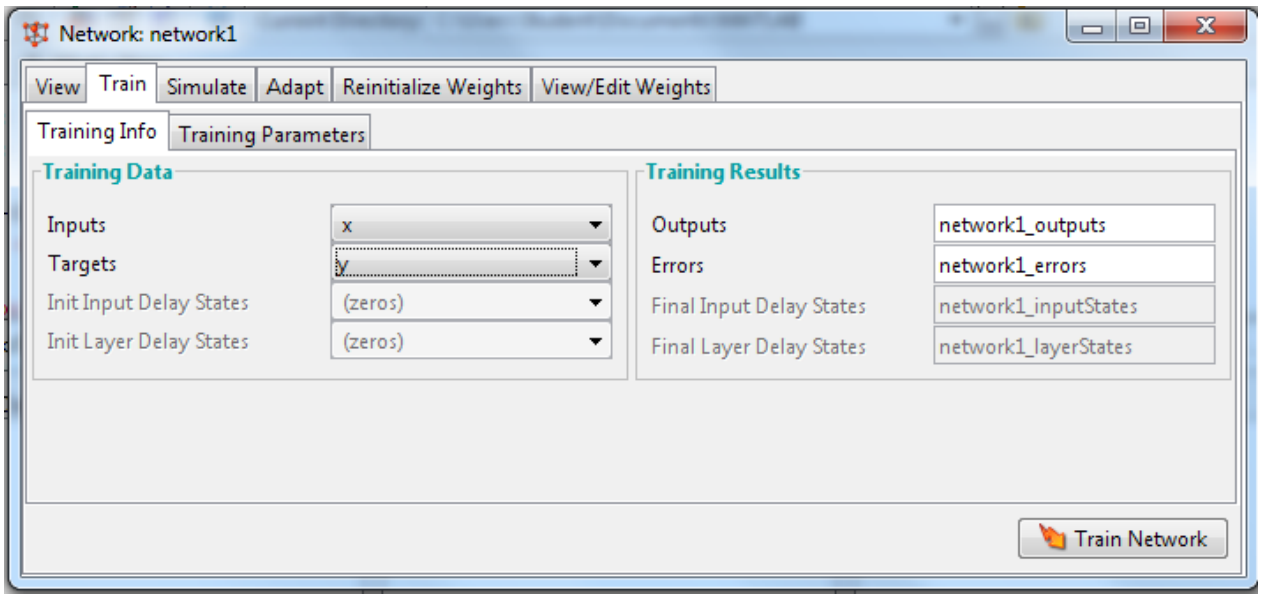


Рисунок 4.22 – Вкладка *Training Info* закладки *Train* в панелі *Network*: *network*1

Тепер натискання кнопки ***Train Network*** викликає навчання мережі. Якість навчання мережі на обраної навчальної послідовності відображається графіком (кнопка ***Performance***). На графіку до кінця процесу навчання помилка стає дуже малою.

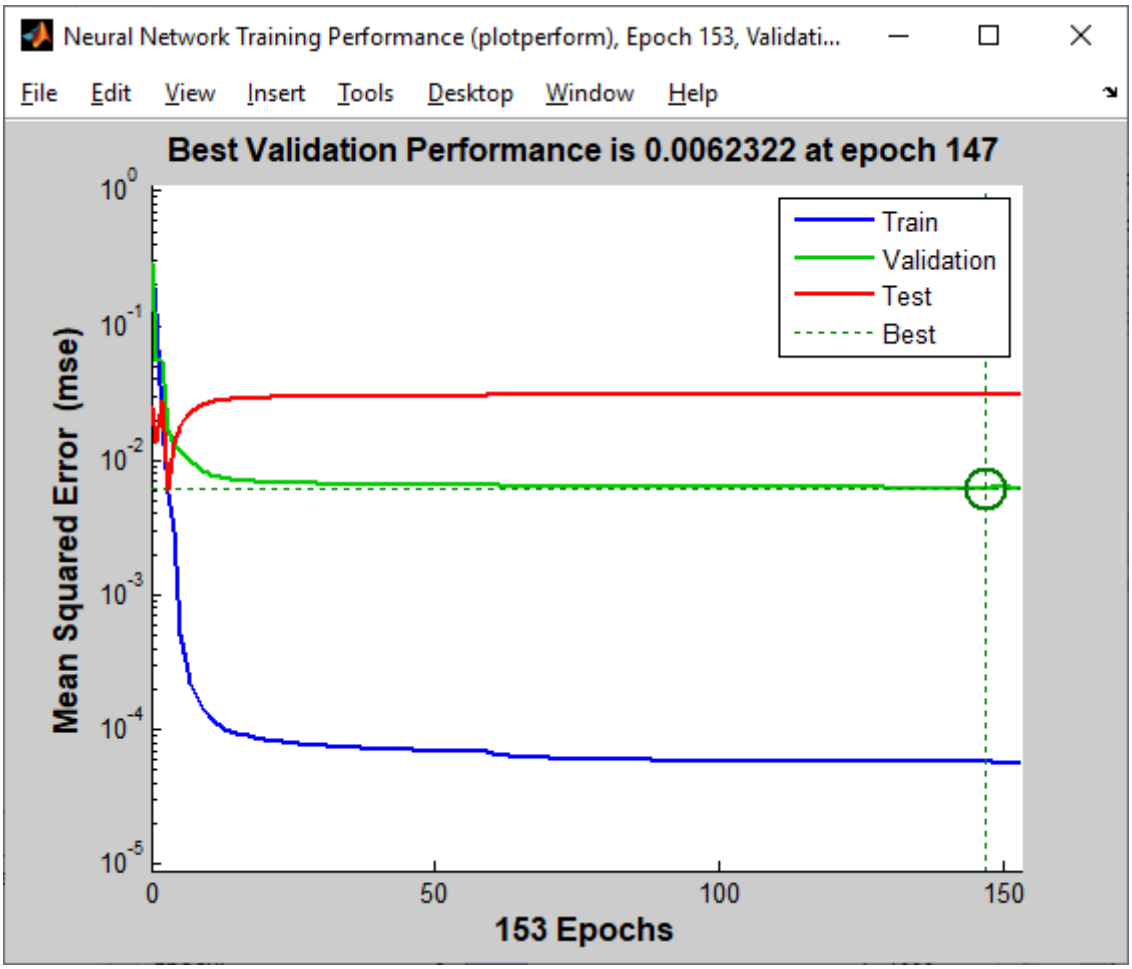


Рисунок 4.23 – Графік якості навчання мережі на обраної навчальної послідовності

Результати навчання можна переглянути у вікні ***Network/Data Manager*** **(*nntool*)**, активізуючи імена послідовностей виходів ***network*1*_outputs*** або помилок ***network*1*_errors*** і використовуючи кнопку ***Open***.

Слід зазначити, що в даному випадку точність апроксимації заданої функції вийшла не дуже високою, в чому можна переконатися, переглянувши значення помилок (***networkl_errors***) або виходів (***net-work*1_outputs**) мережі.

Зауважимо, що точність апроксимації тут можна було б підвищити, конструюючи мережу з великим числом нейронів, але при цьому необхідна і більш представницька навчальна вибірка.

5.3. Робота зі створеною мережею.

Для перегляду структурної схеми мережі необхідно в панелі ***Network*: *network*1** відкрити вкладку ***View***.

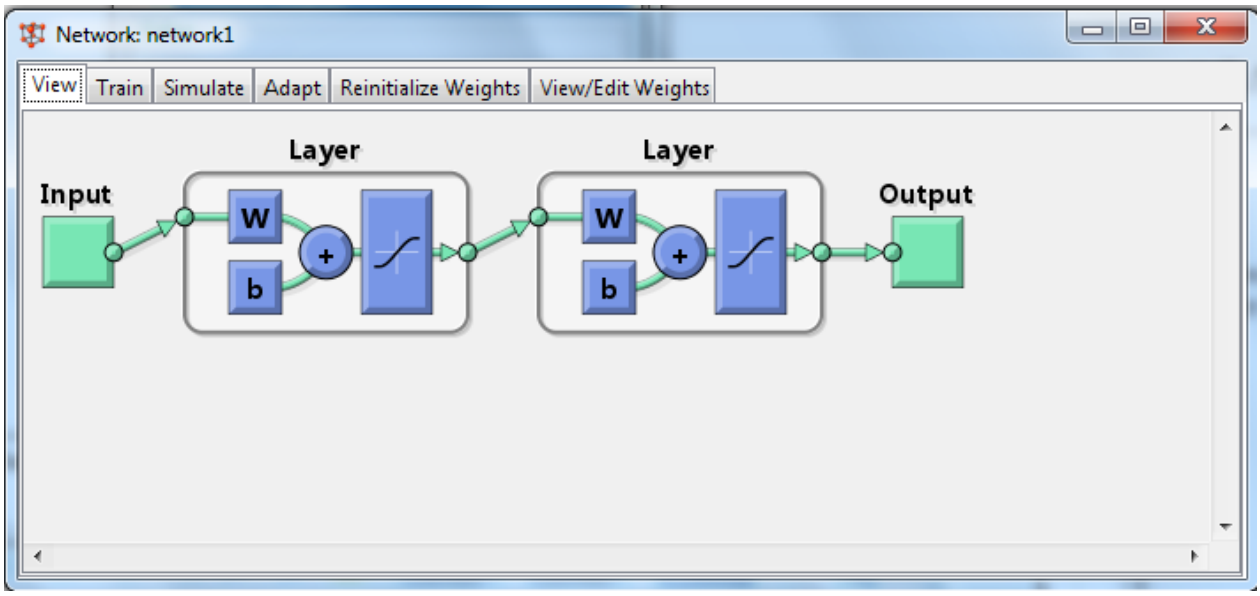


Рисунок 4.24 – Перегляд структурної схеми мережі

Експортуємо створену нейронну мережу в робочу область системи *Matlab*. Для цього у вікні ***Network/Data Manager* (*nntool*)** натискаємо кнопку ***Export*** і далі, у вікні ***Export from Network/Data Manager***, - кнопки ***Select All*** (**Вибрати все**) і ***Export*** і отримаємо інформацію про ваги і зсуви безпосередньо в робочому вікні системи, виконавши команди:

```
>> network1.IW{1,1}, network1.b{1}
>> network1.IW{2,1}, network1.b{2}
```

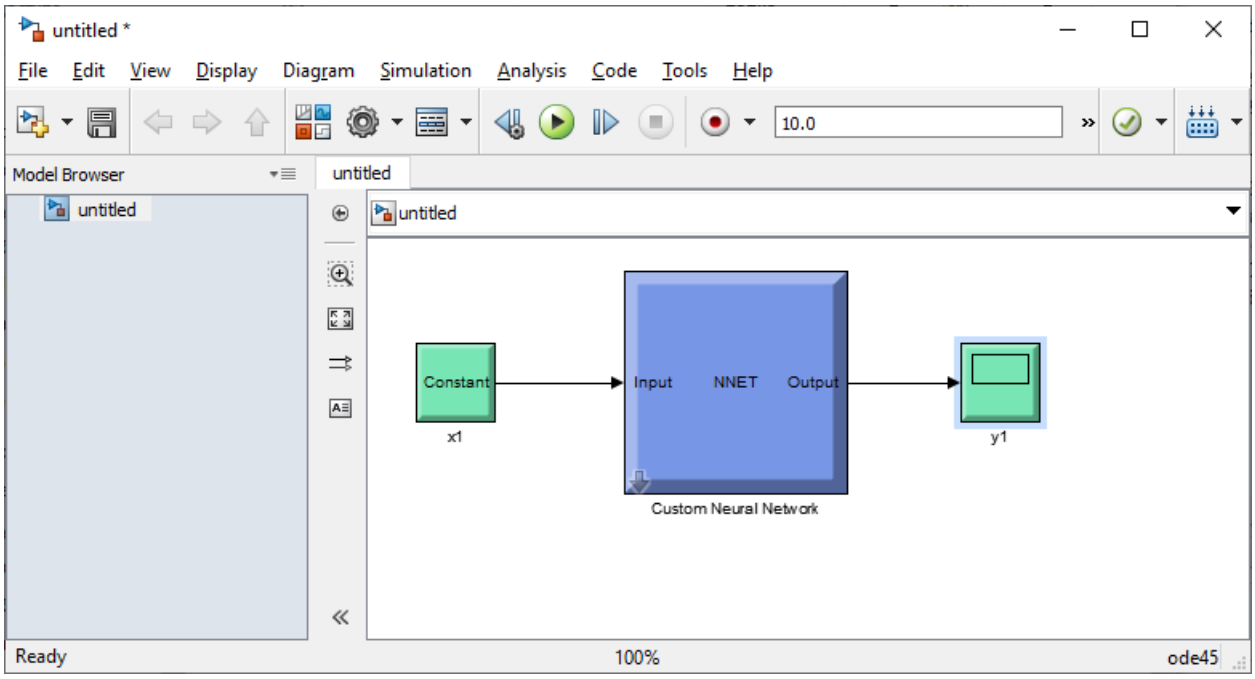


Рисунок 4.25 –Модель НМ побудувана в середовищі *Matlab*

Тепер можна побудувати модель НМ в середовищі *Matlab* і відобразити її схему, використовуючи команду:

```
>> gensim(network1)
```

Ця схема є в повній мірі функціональною схемою і може бути застосована для моделювання нейронної мережі.

Завдання 6. Вислати всі збережені файли та копію вікна *Command Window* в тестовому файлі.

Завдання 7. Письмово відповісти на контрольні запитання.

Контрольні запитання

1. Які програмні засоби входять в пакет *Fuzzy Logic Toolbox*?
2. Як запустити редактор систем нечіткого виводу? Яке призначення редактора систем нечіткого виводу?
3. Системи якого типу дозволяє створювати пакет Fuzzy Logic Toolbox?
4. Як відкрити вікно редактора функцій приналежності? Які установки можна виконати в цьому вікні?
5. Яким чином задаються правила роботи системи нечіткого виводу?
6. Як здійснюється перевірка роботи системи нечіткого виводу?
7. Опишіть процес навчання нейронної мережі.
8. Який пакет в середовищі *Matlab* призначений для створення нейронних мереж?
9. Яка функція проводить опитування нейронної мережі в середовищі *Matlab*?
10. Яка команда відкриває вікно створення нейронної мережі в середовищі *Matlab*?
11. Опишіть процес створення нейронної мережі за допомогою *GUI*-інтерфейсу.

ЛІТЕРАТУРА

1. Гоблик Н. М., Гоблик В. В. *MATLAB* в інженерних розрахунках. Комп'ютерний практикум: навчальний посібник. Львів: Видавництво Львівської політехніки, 2020. 192 с.

2. Гайдур Г. І., Бондаренко З. З., Марченко В. В., Чумак Н. С. Лабораторний практикум з навчальної дисципліни «Теорія інформації та кодування»: навчальний посібник для студентів вищих навчальних закладів. Київ: ДУТ, ННІЗІ, 2021. 50 с. URL: https://dut.edu.ua/uploads/l_2186_62744936.pdf (дата звернення: 28.09.2022).

3. Кузенко Л. М., Кузенко Д. В., Вантух З. З., Панюра Я. Й. Дорожньо-будівельні машини: навчальний посібник. Київ: Кондор, 2021. 236 с.

Навчальне видання

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних та самостійних робіт
з дисципліни «Основи програмування»
для студентів спеціальностей
132 «Матеріалознавство», 133 «Галузеве машинобудування»
за освітньо-кваліфікаційним рівнем бакалавра.
Розділ «Інтегрований пакет математичного моделювання Matlab»

Укладачі: КОСТІКОВА Марина Володимирівна
ШЕВЧЕНКО Вікторія Олександрівна

Відповідальний за випуск *А. І. Левтеров*

В авторській редакції

Комп'ютерна верстка *Н.В. Ольховської*

План 2023 р. Поз. 34
Підписано до друку 07.04.2023 р. Формат 60×84 1/16.
Гарнітура Times New Roman Cyr.
Ум. друк. арк. 4,3. Обл.-вид. арк. 4,8
Зам. № 21/23-В. Наклад сайт.

ВИДАВНИЦТВО
Харківського національного автомобільно-дорожнього університету
Видавництво ХНАДУ, 61002, Харків-МСП, вул. Ярослава Мудрого, 25.

Тел. /факс: (057)700-38-64; 707-37-03,
e-mail: rio@khadi.kharkov.ua

Свідоцтво Державного комітету інформаційної політики, телебачення та радіомовлення
України про внесення суб'єкта видавничої справи до Державного реєстру
видавців, виготівників і розповсюджувачів видавничої продукції,
серія № ДК №897 від 17.04 2002 р.