**Лабораторна робота № 2**

МОДЕЛЮВАННЯ ПРОСТИХ НЕЙРОННИХ МЕРЕЖ

***Мета роботи:*** дослідити можливості ППП *MATLAB* *моделювання простих нейронних мереж****.***

1. ТЕОРЕТИЧНІ ВІДОМОСТІ

Теоретичні відомості подані у окремому файлі **ШІ\_КБ\_ЛР-2\_теорія**

**2. ЗАВДАННЯ НА ЛАБОРАТОРНУ РОБОТУ**

**Завдання 2.1. Ознайомлення з графічним інтерфейсом користувача NNTool**

Нейронні мережі (NN - Neural Networks) широко використовуються для вирішення різноманітних завдань. Серед країн, що розвиваються областей застосування NN - обробка аналогових і цифрових сигналів, синтез і ідентифікація електронних ланцюгів і систем. Основи теорії і технології застосування NN широко представлені в пакеті MATLAB. У зв'язку з цим особливо слід відзначити останню версію пакета - MATLAB 6.0, де вперше представлений GUI (Graphical User Interface - графічний інтерфейс користувача) для NN - NNTool.

Прикладами застосування технології нейронних мереж для цифрової обробки сигналів є: фільтрація, оцінка параметрів, детектування, ідентифікація систем, розпізнавання образів, реконструкція сигналів, аналіз часових рядів і стиснення. Згадані види обробки застосовні до різноманітних видів сигналів: звукових, відео, мовним, зображення, передачі повідомлень, геофізичних, локаційним, медичних вимірювань (кардіограми, енцефаллограмми, пульс) і іншим.

В даній статті дано опис NNTool і показана техніка його застосування в ряді завдань синтезу ланцюгів і цифрової обробки сигналів.

Обробка сигналів в технологіях NN виконується за допомогою яких NN без пам'яті, або NN c пам'яттю. І в тому і в іншому випадках ключовим елементом є NN без пам'яті. Подібна роль визначається тією обставиною, що при використанні нейронів з певними функціями активації (передавальними характеристиками) NN є універсальним аппроксіматором. Останнє означає, що в заданому діапазоні зміни вхідних змінних NN може з заданою точністю відтворювати (моделювати) довільну безперервну функцію цих змінних.

Нижче обговорюються питання, що відносяться до так званим NN прямого поширення, тобто без зворотних зв'язків. Чудовим властивістю таких NN є їх стійкість.

Після того як структура NN обрана, повинні бути встановлені її параметри. Вибір структури NN і типів нейронів - самостійний і вельми непросте питання, який тут ми обговорювати не будемо. Що ж стосується значень параметрів, то, як правило, вони визначаються в процесі вирішення деякої оптимізаційної задачі. Ця процедура в теорії NN називається навчанням.

Графічний інтерфейс користувача NNTool дозволяє вибирати структури NN з великого переліку і надає множину алгоритмів навчання для кожного типу мережі.

У роботі розглянуті наступні питання, що стосуються роботи з NNTool:

• призначення графічних керуючих елементів;

• підготовка даних;

• створення нейронної мережі;

• навчання мережі;

• прогін мережі.

Всі етапи роботи з мережами проілюстровані прикладами розв'язання простих завдань. Передбачається, що читач знайомий з основами теорії NN і її термінологією.

**Керуючі елементи NNTool**

Щоб запустити NNTool, необхідно виконати однойменну команду в командному вікні MATLAB:

>> nntool

після цього з'явиться головне вікно NNTool, іменоване "Вікном управління мережами і даними" (Network / Data Manager) (рис. 1).

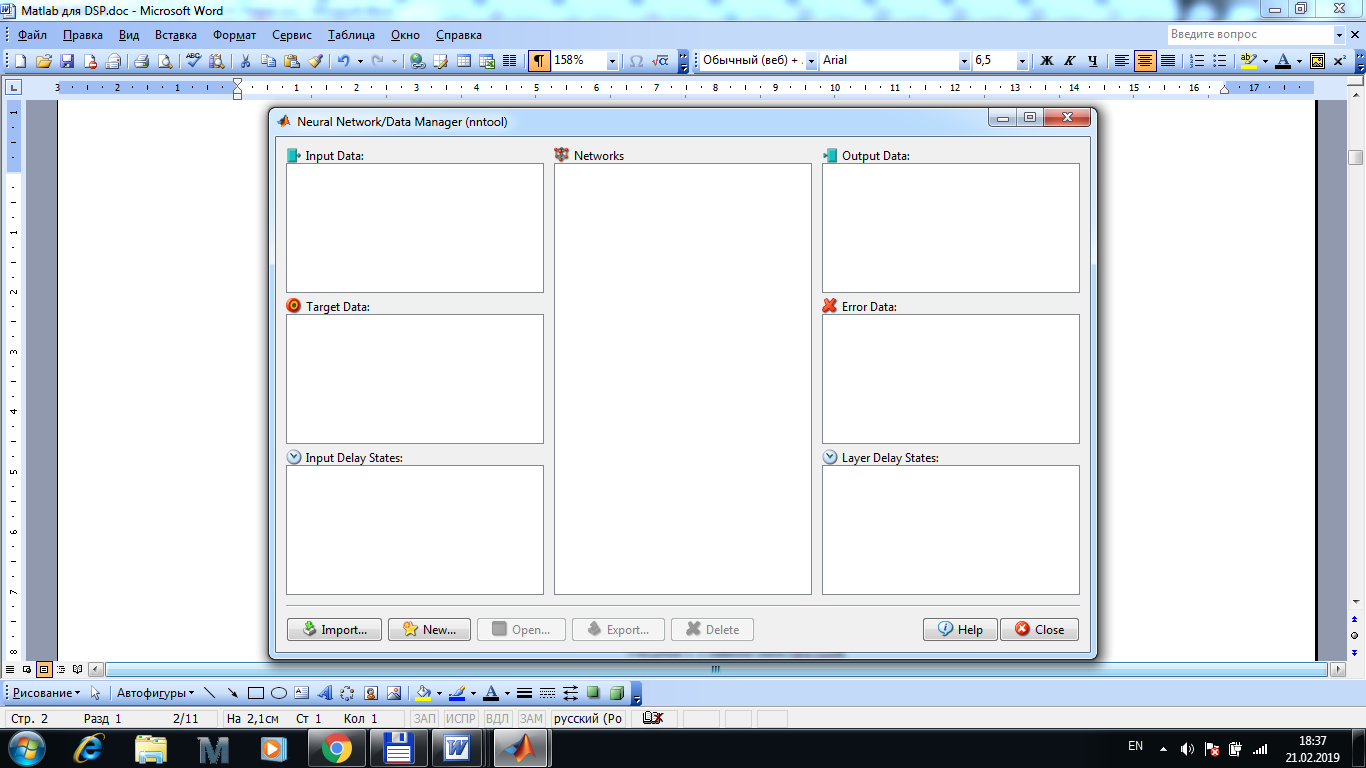


Рис 1. Головне вікно NNTool

Панель "Мережі і дані" (Networks and Data) має функціональні клавіші з наступними призначеннями:

• Допомога (Help) - короткий опис керуючих елементів цього вікна;

• Нові дані (New Data ...) - виклик вікна, що дозволяє створювати нові набори даних;

• Нова мережа (New Network ...) - виклик вікна створення нової мережі;

• Імпорт (Import ...) - імпорт даних з робочого простору MATLAB в простір змінних NNTool;

• Експорт (Export ...) - експорт даних з простору змінних NNTool в робочий простір MATLAB;

• Вид (View) - графічне відображення архітектури обраної мережі;

• Видалити (Delete) - видалення обраного об'єкта.

На панелі "Тільки мережі" (Networks only) розташовані клавіші для роботи виключно з мережами. При виборі покажчиком миші об'єкта будь-якого іншого типу, ці кнопки стають неактивними.

При роботі з NNTool важливо пам'ятати, що клавіші View, Delete, Initialize, Simulate, Train і Adapt (зображені на рис. 1 як неактивні) діють стосовно того об'єкту, який зазначений у даний момент виділенням. Якщо такого об'єкта немає, або над виділеним об'єктом неможливо зробити вказане дію, відповідна клавіша неактивна.

Розглянемо створення нейронної мережі за допомогою NNTool на прикладі.

**Завдання 2.2. Створити нейронну мережу, що виконує логічну функцію "І".**

Створення мережі

Виберемо мережу, що складається з одного персептрона з двома входами. У процесі навчання мережі на її входи подаються вхідні дані і проводиться зіставлення значення, отриманого на виході, з цільовим (бажаним). На підставі результату порівняння (відхилення отриманого значення від бажаного) обчислюються величини зміни ваг і зсуву, що зменшують це відхилення.

Отже, перед створенням мережі необхідно заготовити набір навчальних і цільових даних. Складемо таблицю істинності для логічної функції "І", де P1 і Р2 - входи, а А - бажаний вихід (табл. 1).

Таблиця 1. Таблиця істинності логічної функції "І"



Щоб задати матрицю, що складається з чотирьох векторів-рядків, скористаємося кнопкою New Data. У вікні слід провести зміни, показані на рис. 2, і натиснути клавішу "Створити" (Create).

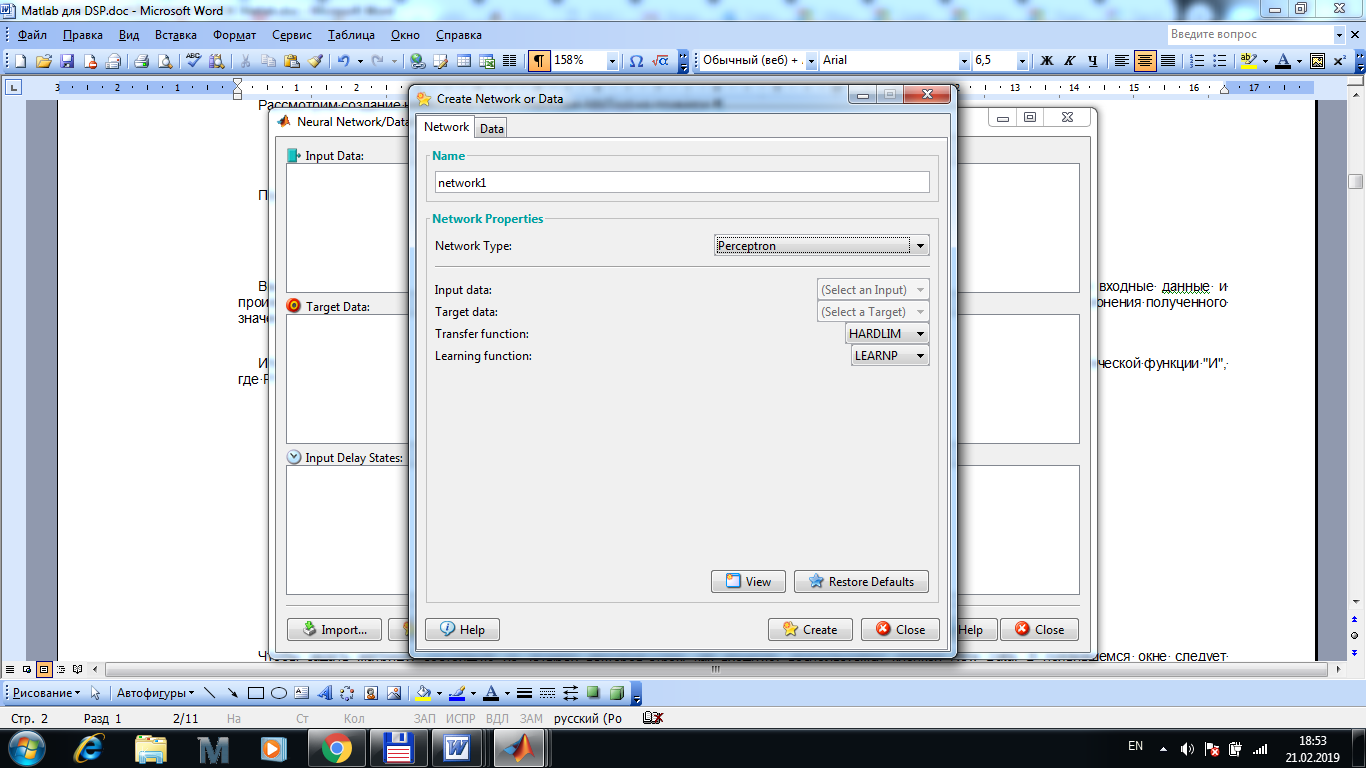
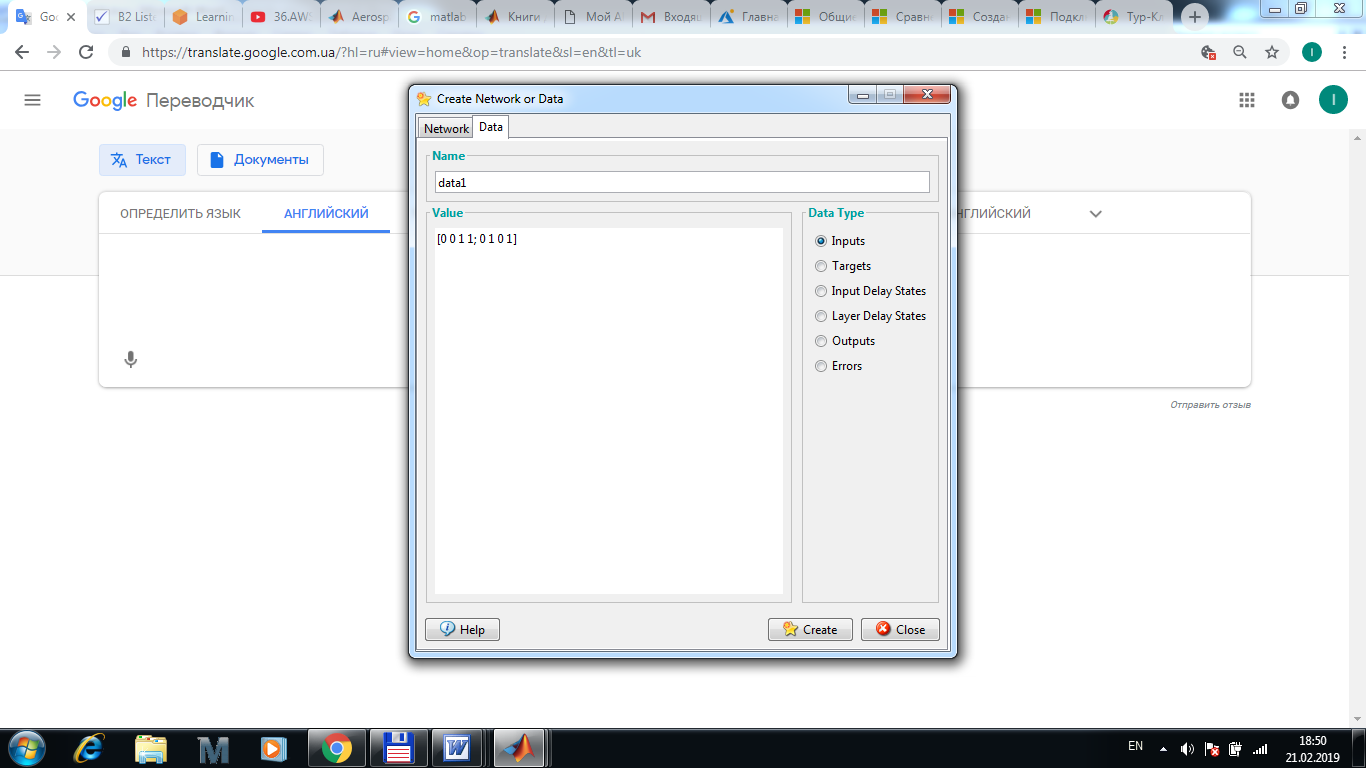
 

Рис. 2 Рис. 3

Перейдіть у закладку Data та введіть вхідні дані, як показано на рис.3

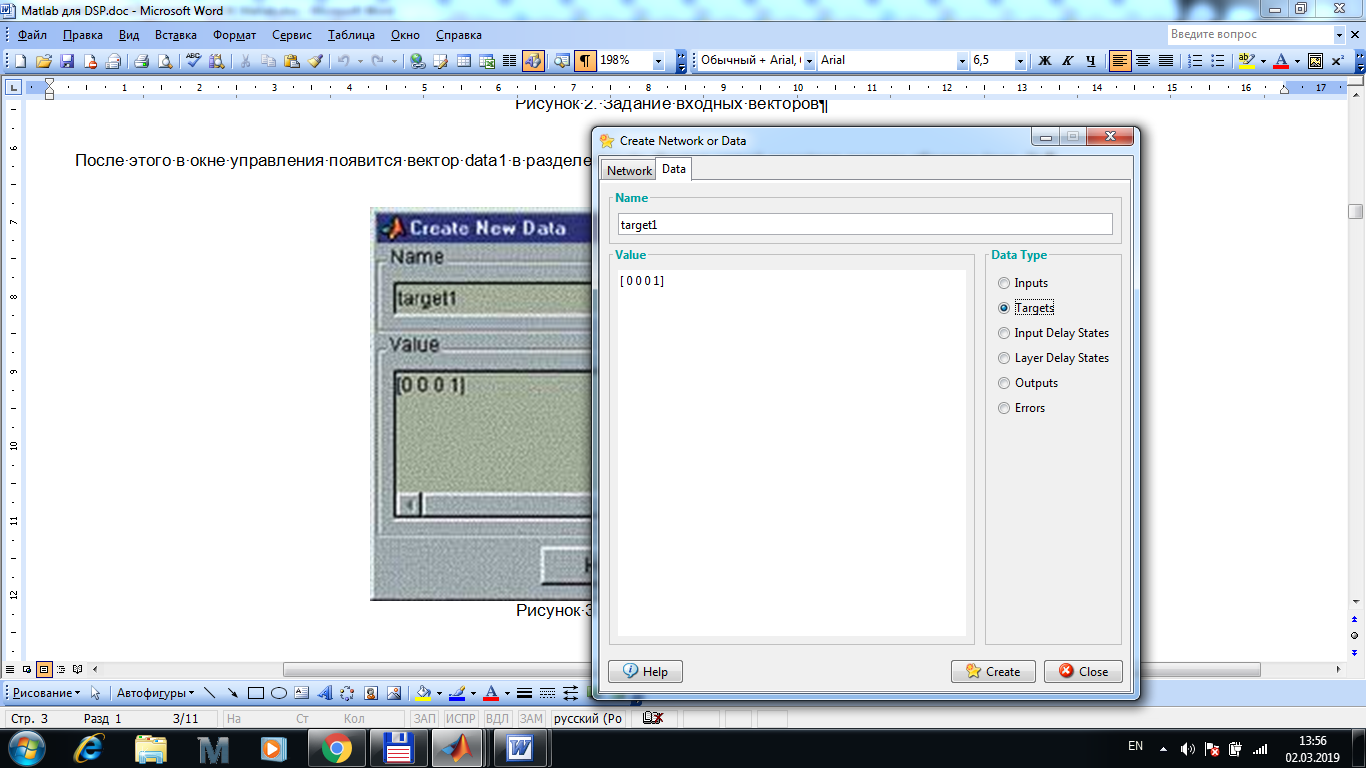
Дані в полі "Значення" (Value) можуть бути представлені будь-яким зрозумілим для MATLAB виразом. Наприклад, попереднє визначення вектора цілей можна еквівалентно замінити рядком виду

bitand ([0 0 1 1], [0 1 0 1]).

Після натискання на Create в розділі Input Data основного вікна з'явиться вектор data1.

***Зображення вікна з даними*** ***занесіть у бланк звіту (рис.1)***

Введіть вектор цілей, як показано на рис. 4.

 Рис. 4

Після натискання на Create в розділі Targets з'явиться вектор target1.

***Зображення вікна з цілями*** ***занесіть у бланк звіту (рис.2)***

Перейдемо до закладки Network

Поля цього вікна несуть наступні смислові навантаження:

• Ім’я мережі (Назва мережі) - це ім'я об'єкта створеної мережі.

• Тип мережі (Network Type) - визначає тип мережі та контекст вибору типу, що вказує на різні параметри в частині, розташованій нижче цього пункту. Таким чином, для різних типів сітки вікно змінює своє зміст.

• Вхідні діапазони (діапазони вхідних даних) - матриця з числом рядків, рівних числу входів. Кожен рядок представляє собою вектор з двома елементами: перший - мінімальне значення сигналу, який буде поданий на відповідний вхід в мережу за умов навчання, другого - максимального. Для того, щоб уникнути цих знань забезпечити випуск списку "Вибрати з входу", можна автоматично змінити необхідні дані, вказуючи на вхідний змінні.

• Кількість нейронів (Кількість нейронів) - число нейронів в слої.

• Передаточна функція (Функція передачі) - в цьому пункті вибирається передаточна функція (функція активації) нейронів.

• Функція навчання (функція навчання) - функція, що відповідає за оновлення і зміну мереж в процесі підготовки.

За допомогою кнопки "Вид" можна поглянути на архітектуру створюваної мережі (рис. 5). Так, ми можемо переконатись, що всі дії були зроблені вірно. На рис. 5 зображена персональна мережа з вихідним блоком, реалізовуючи передаточну функцію з жорстким обмеженням. Кількість нейронів в одному рівні, що символічно відображає величину візуально-стовбурової тканини на вищому рівні і вказує на кількість безпосередньо під блоком передаточної функції. Мережа має два входи, так як розмір стовпчика вхідного вікна – рівний двом.

Отже, структура мережі відповідає нашим завданням. Тепер можна закрити вікно попереднього перегляду, натиснувши клавішу "Закрити" (Close), і підтвердити намір створити мережу, натиснувши "Створити" (Create) в вікні створення мережі.

В результаті виконаних операцій в розділі "Мережі" (Networks) головного вікна NNTool з'явиться об'єкт з ім'ям network1.

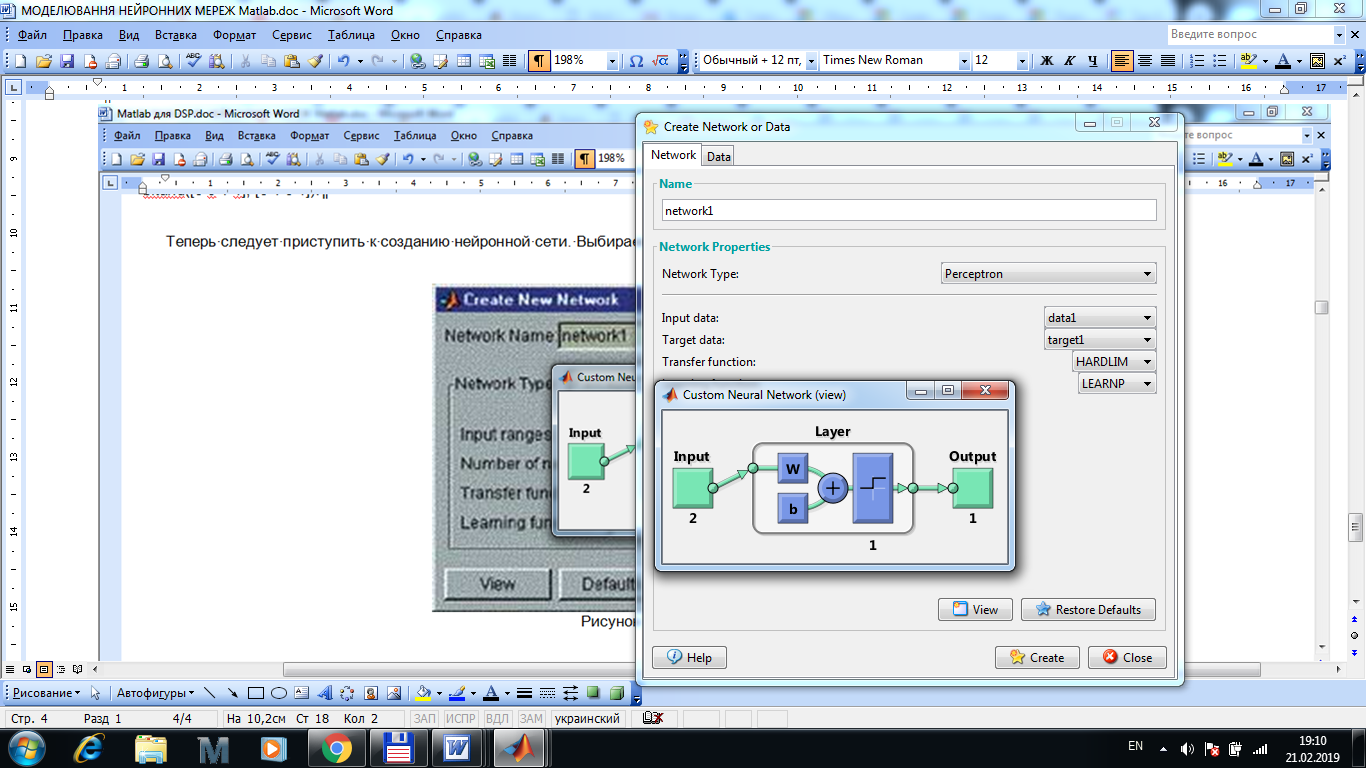


Рис. 5

***Зображення вікна з архітектурою створюваної мережі занесіть у бланк звіту (рис.3)***

**Навчання**

Наша мета - побудувати нейронну мережу, яка виконує функцію логічного "І". Очевидно, не можна розраховувати на те, що відразу після етапу створення мережі остання буде забезпечувати правильний результат (правильне співвідношення "вхід/вихід"). Для досягнення мети мережу необхідно належним чином навчити, тобто підібрати відповідні значення параметрів. В MATLAB реалізовано більшість відомих алгоритмів навчання нейронних мереж, серед яких представлено два для персептронів мереж розглянутого виду. Створюючи мережу, ми вказали LEARNP в якості опції, що реалізує алгоритм навчання (рис. 5).

Повернемося до головного вікна NNTool.

Відзначивши покажчиком миші об'єкт мережі network1, натискаємо два рази на ньому, чим викличемо вікно управління мережею (рис.6).

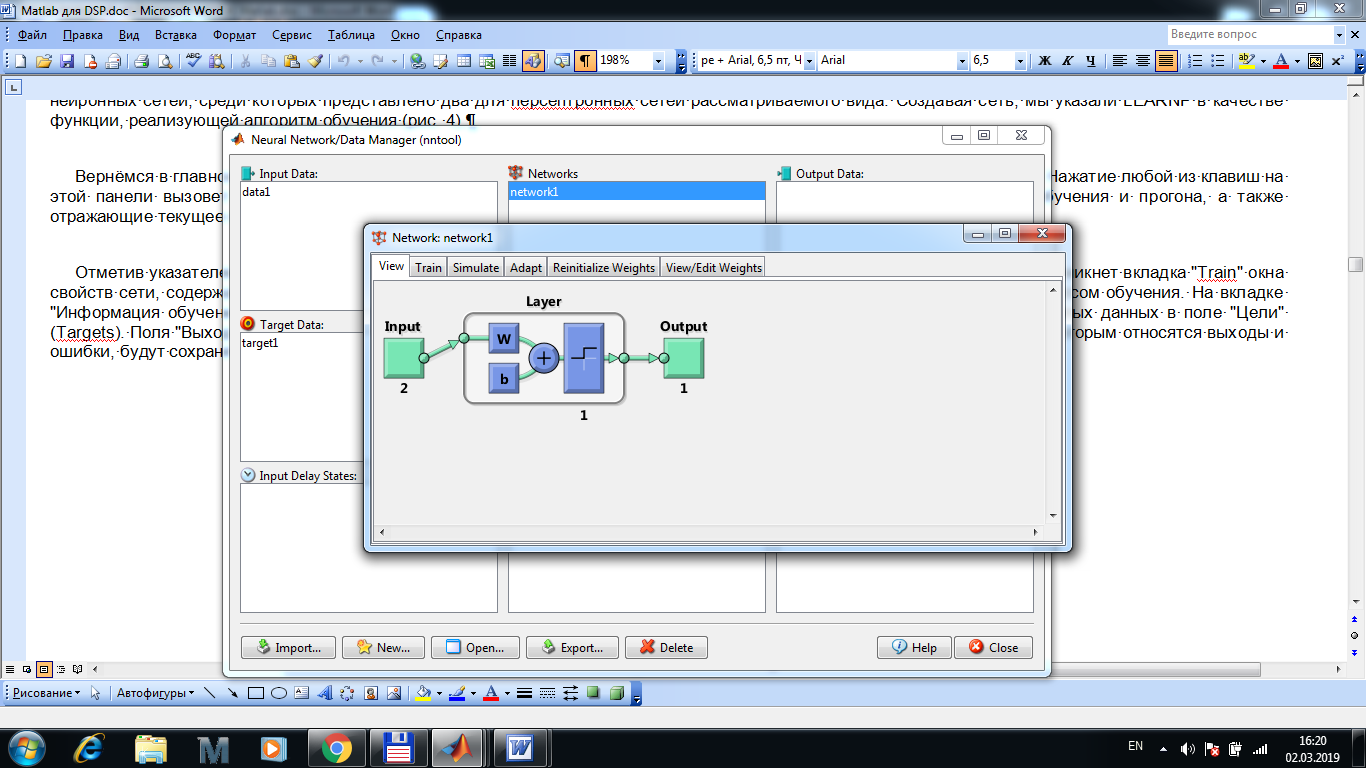


Рис. 6

Натискаємо вкладку Train (рис.7).

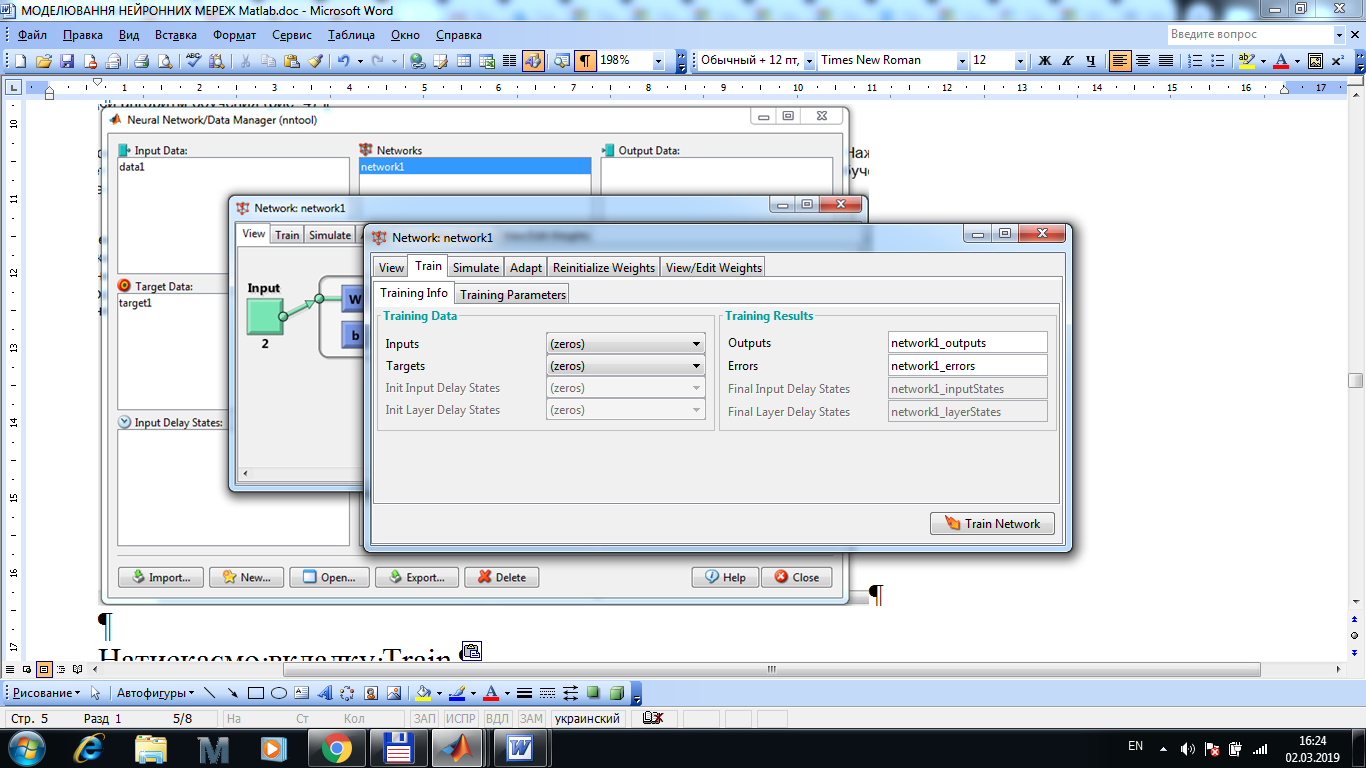


Рис.7

Перед нами виникне вкладка "Train" вікна властивостей мережі, що містить, в свою чергу, ще одну панель вкладок (рис. 7). Їх головне призначення - управління процесом навчання. На вкладці "Інформація навчання" (Training info) потрібно вказати набір навчальних даних в поле "Входи" (Inputs) і набір цільових даних в поле "Цілі" (Targets). Поля "Виходи" (Outputs) і "Помилки" (Errors) NNTool заповнює автоматично. При цьому результати навчання, до яких відносяться виходи і помилки, будуть зберігатися в змінних з зазначеними іменами (рис.8).

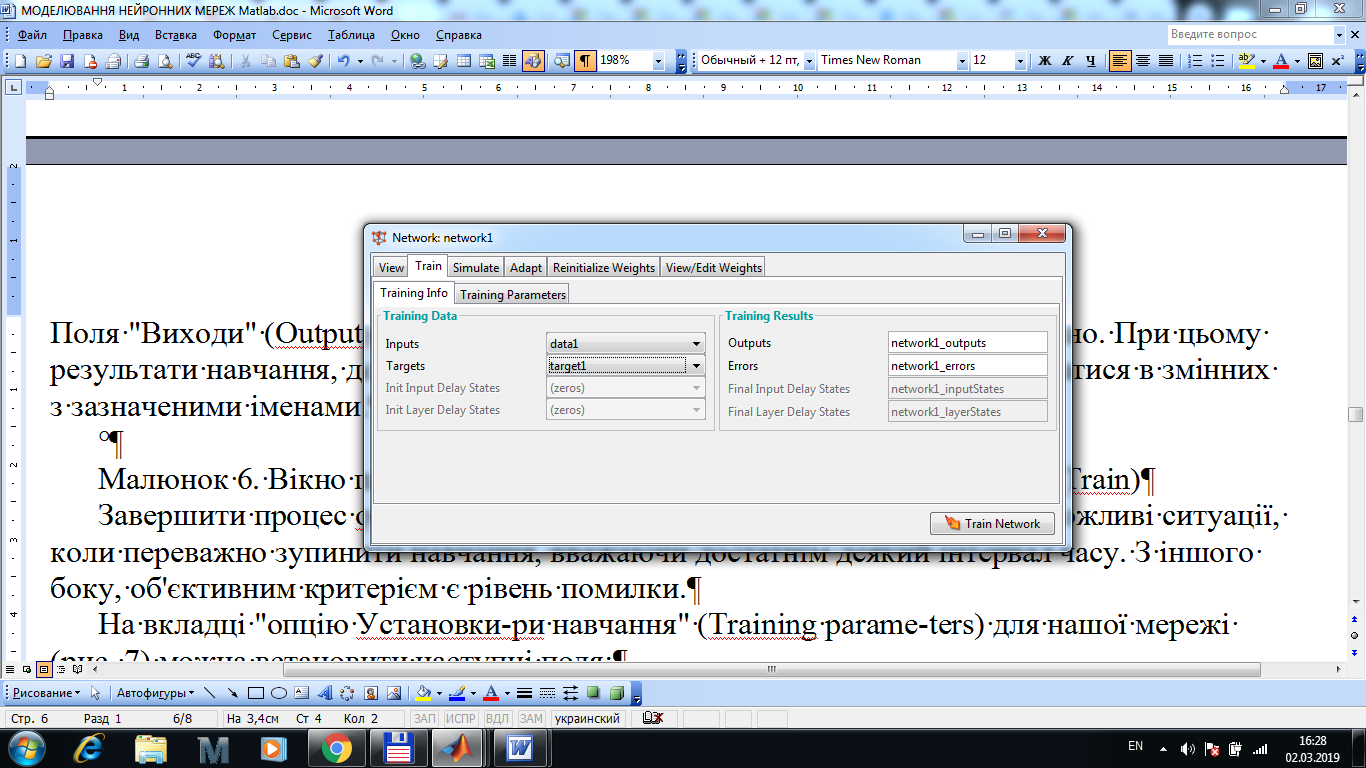


Рис.8.

Завершити процес навчання можна, керуючись різними критеріями. Можливі ситуації, коли бажано зупинити навчання, вважаючи достатнім деякий інтервал часу. З іншого боку, об'єктивним критерієм є рівень помилки.

На вкладці "опції Установки навчання" (Training parameters) для нашої мережі (рис. 9) можна встановити наступні поля:

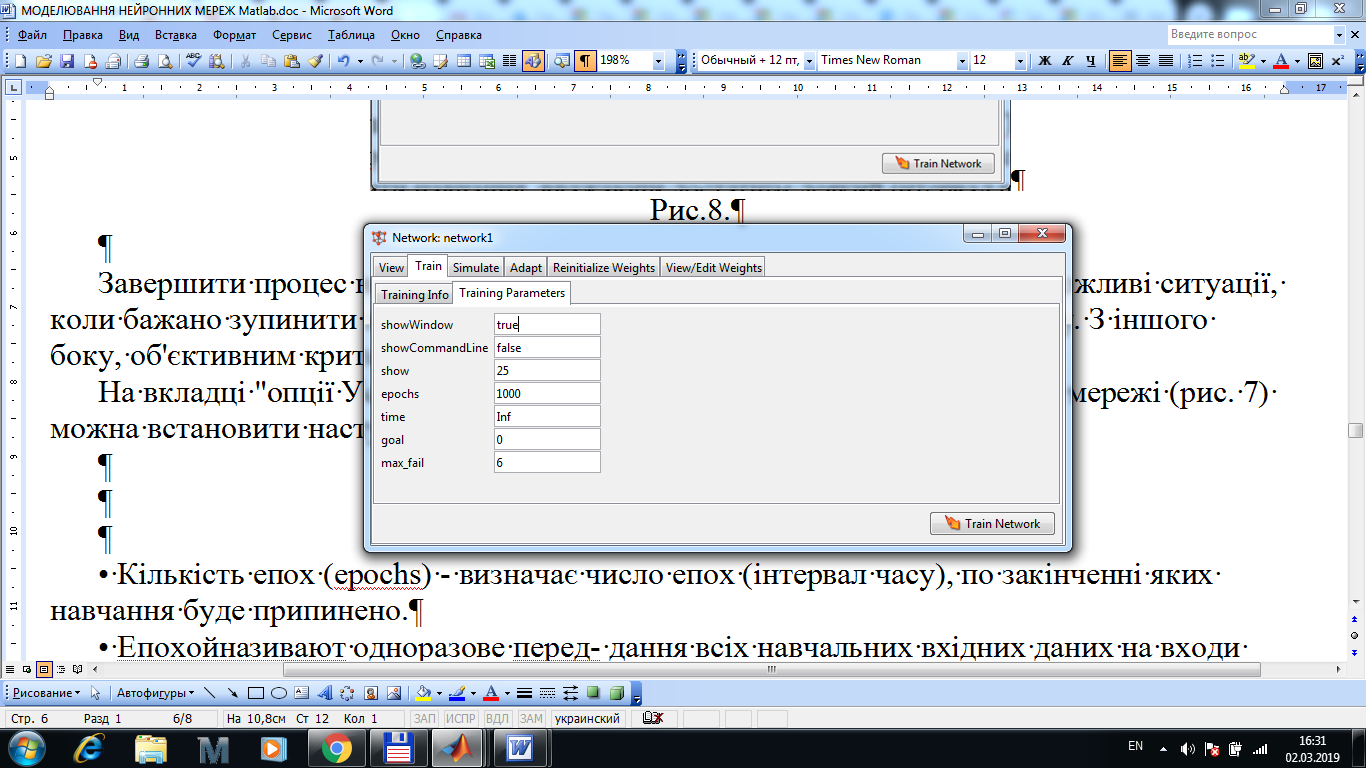


Рис.9.

• Кількість епох (epochs) - визначає число епох (інтервал часу), по закінченні яких навчання буде припинено.

• Епохою називають одноразове передання всіх навчальних вхідних даних на входи мережі.

• Досягнення мети або потрапляння (goal) - тут задається абсолютна величина функції помилки, при якій мета буде вважатися досягнутою.

• Період оновлення (show) - період оновлення графіка кривої навчання, виражений числом епох.

• Час навчання (time) - після закінчення зазначеного в ньому тимчасового інтервалу, вираженого в секундах, навчання припиняється.

Беручи до уваги той факт, що для задач з лінійно роздільними множинами (а наше завдання відноситься до цього класу) завжди існує точне рішення, встановимо поріг досягнення мети, рівний нулю. Значення інших параметрів залишимо за замовчуванням. Зауважимо тільки, що поле часу навчання містить запис Inf, яка визначає нескінченний інтервал часу (від англійського Infinite - нескінченний).

Повернемося до вкладки навчання (Train) - "Інформація навчання" (Training info). Щоб почати навчання, потрібно натиснути кнопку "Навчити мережу" (Train Network).

З’являється вікно навчання мережі (рис.10).

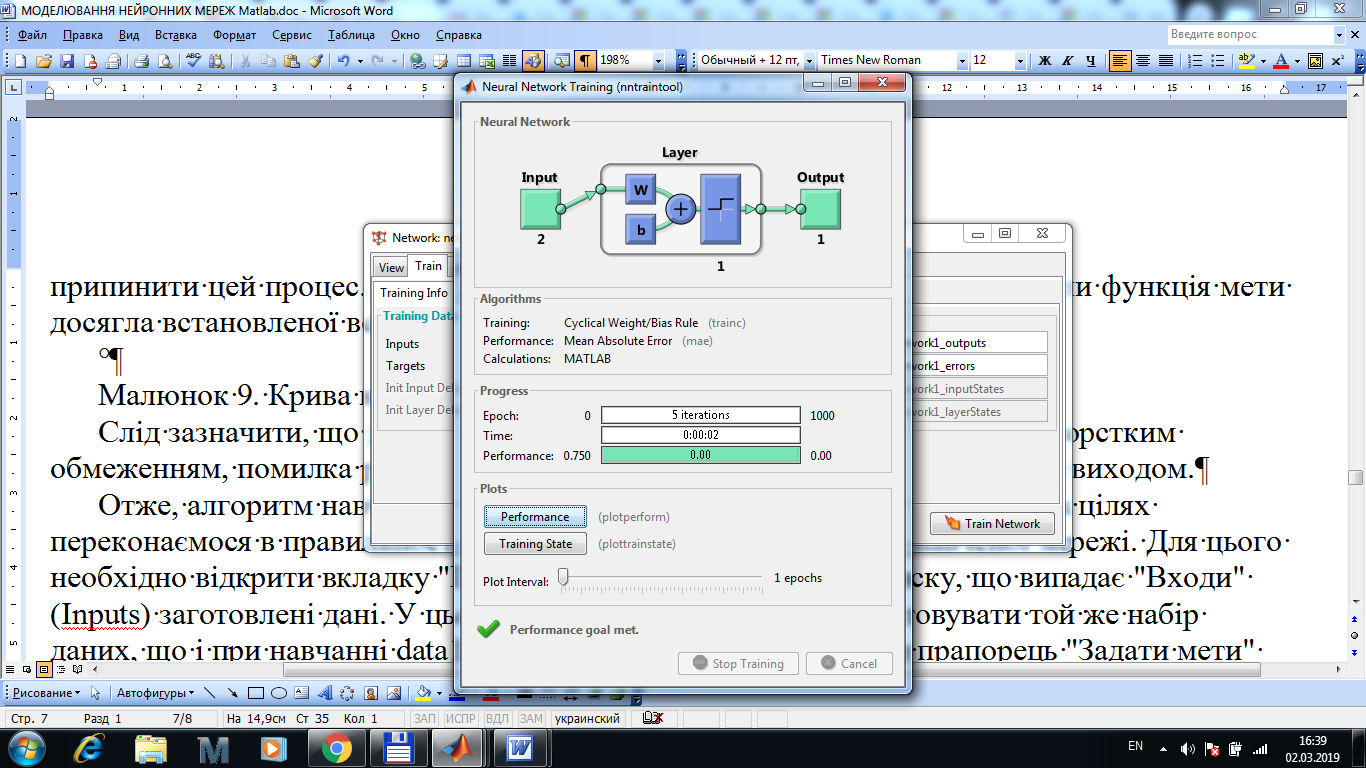


Рис. 10

Після цього, якщо в поточний момент мережа не задовольняє жодному з умов, зазначених в розділі параметрів навчання (Training Parameters), можна визвати вікно, що ілюструє динаміку цільової функції - криву навчання з блоку вікна Plots. За допомогою кнопки "Зупинити навчання" (Stop Training) можна припинити цей процес. З малюнка видно, що навчання було зупинено, коли функція мети досягла встановленої величини (goal = 0).

Слід зазначити, що для персептронів, що мають функцію активації з жорстким обмеженням, помилка розраховується як різниця між ціллю та отриманим виходом.

Отже, алгоритм навчання знайшов точне рішення задачі. У методичних цілях переконаємося в правильності рішення задачі шляхом прогону навченої мережі. Для цього необхідно відкрити вкладку "Прогін" (Simulate - моделювати) і вибрати в списку, що випадає "Входи" (Inputs) заготовлені дані. У цьому завданню природно використовувати той же набір даних, що і при навчанні data1. При бажанні (рис.11) можна встановити прапорець "Задати мету" (Supply Targets). Тоді в результаті прогону додатково будуть розраховані значення помилки.

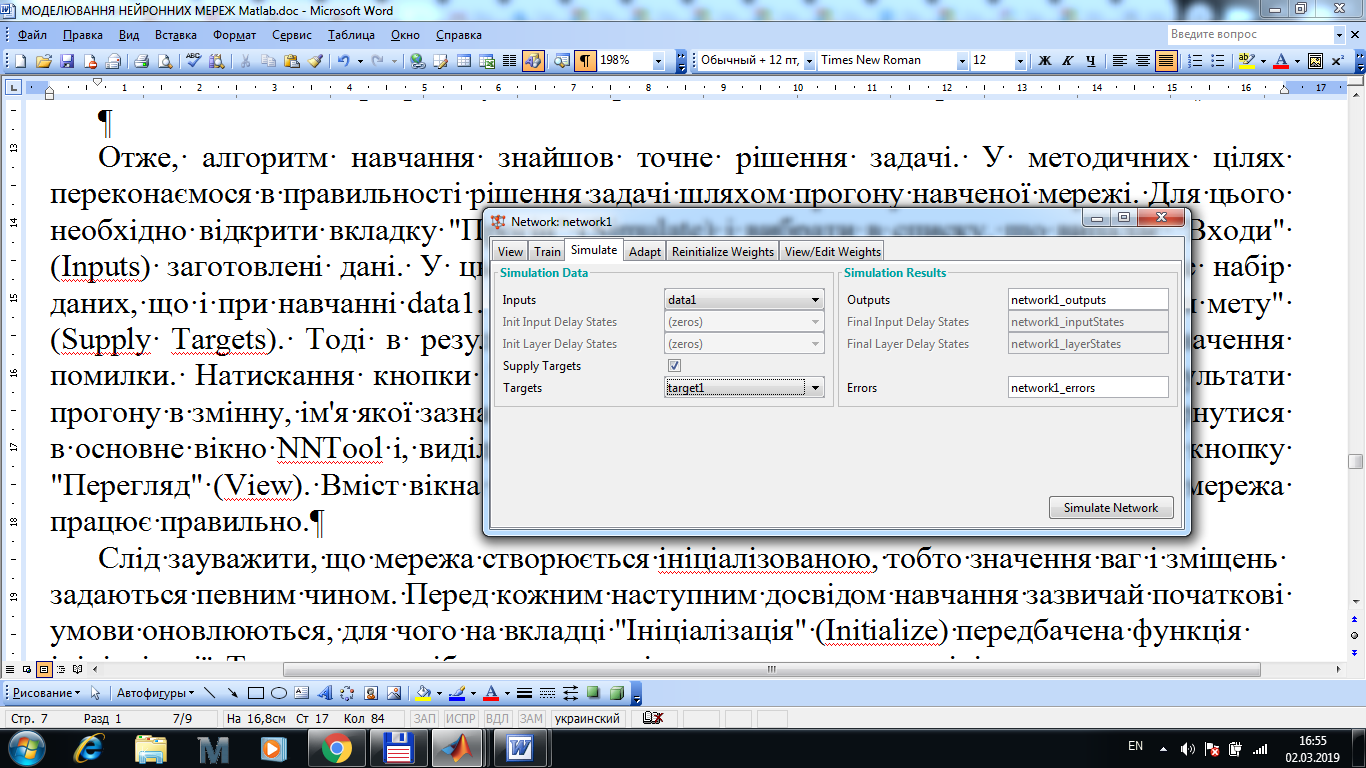


Рис. 11

Натискання кнопки "Прогін мережі" (Simulate Network) запише результати прогону в змінну, ім'я якої зазначено в поле "Виходи" (Outputs). Тепер можна повернутися в основне вікно NNTool і, виділивши мишею вихідну змінну network1, натиснути два рази – з’явиться (рис.12) вікно (View). Вміст вікна перегляду збігається зі значенням вектора цілей - мережа працює правильно.

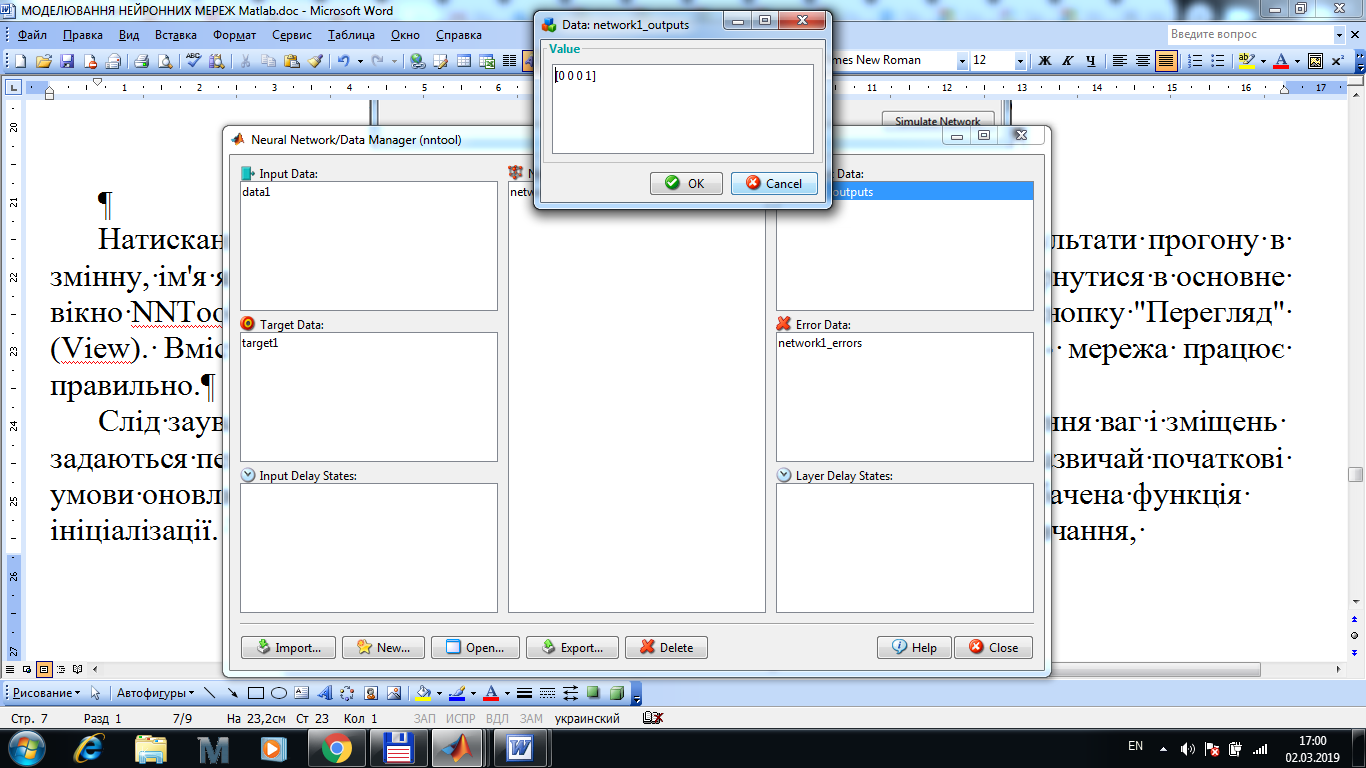


Рис.12

***Зображення вікна з вихідною змінною network1 занесіть у бланк звіту (рис.4)***

Слід зауважити, що мережа створюється ініціалізованою, тобто значення ваг і зміщень задаються певним чином. Перед кожним наступним досвідом навчання зазвичай початкові умови оновлюються, для чого на вкладці "Ініціалізація" (Initialize) передбачена функція ініціалізації. Так, якщо потрібно провести кілька незалежних дослідів навчання, ініціалізація ваг і зміщень перед кожним з них здійснюється натисканням кнопки "Ініціалізувати ваги" (Initialize Weights).

При виборі нейронної мережі для вирішення конкретного завдання важко передбачити її порядок. Якщо вибрати невиправдано великий порядок, мережа може виявитися занадто гнучкою і може подати просту залежність складним чином. Це явище називається перенавчанням. У разі мережі з недостатньою кількістю нейронів, напроти, необхідний рівень помилки ніколи не буде досягнутий. Тут у наявності надмірне узагальнення.

Для попередження перенавчання застосовується наступна техніка. Дані поділяються на дві множини: навчальну (Training Data) і контрольну (Validation Data). Контрольне множина в навчанні не використовується. На початку роботи помилки мережі на навчальному та контрольному множинах будуть однаковими. У міру того, як мережа навчається, помилка навчання убуває, і, поки навчання зменшує дійсну функцію помилки, помилка на контрольному множині також буде спадати. Якщо ж контрольна помилка перестала зменшуватися або навіть стала рости, це вказує на те, що навчання слід закінчити. Зупинка на цьому етапі називається ранньою зупинкою (Early stopping).

Таким чином, необхідно провести серію експериментів з різними мережами, перш ніж буде отримана відповідна. При цьому щоб не бути введеним в оману локальними мінімумами функції помилки, слід кілька разів навчати кожну мережу.

Якщо в результаті послідовних кроків навчання і контролю помилка залишається неприпустимо великою, доцільно змінити модель нейронної мережі (наприклад, ускладнити мережу, збільшивши число нейронів, або використовувати мережу іншого виду). У такій ситуації рекомендується застосовувати ще одну множину - тестову множину спостережень (Test Data), яка представляє собою незалежну вибірку з вхідних даних. Підсумкова модель тестується на цій множині, що дає додаткову можливість переконатися в достовірності отриманих результатів. Очевидно, щоб зіграти свою роль, тестова множина повинна бути використана тільки один раз. Якщо її використовувати для коригування мережі, вона фактично перетвориться на контрольну множину.

Навчання мережі можна проводити в різних режимах. У зв'язку з цим, в NNTool передбачено дві вкладки, що представляють навчальні функції: розглянута раніше вкладка Train і "Адаптація" (Adapt). Adapt (рис.13) вміщує вкладку інформації адаптації (Adaption Info), на якій містяться поля, схожі за своїм призначенням з полями вкладки Training Info і виконують ті ж функції і вкладку параметри адаптації (Adaption Parameters). Остання містить єдине поле "Проходи" (passes). Значення, вказане в цьому полі, визначає, скільки разів все вхідні вектори будуть представлені мережі в процесі навчання.

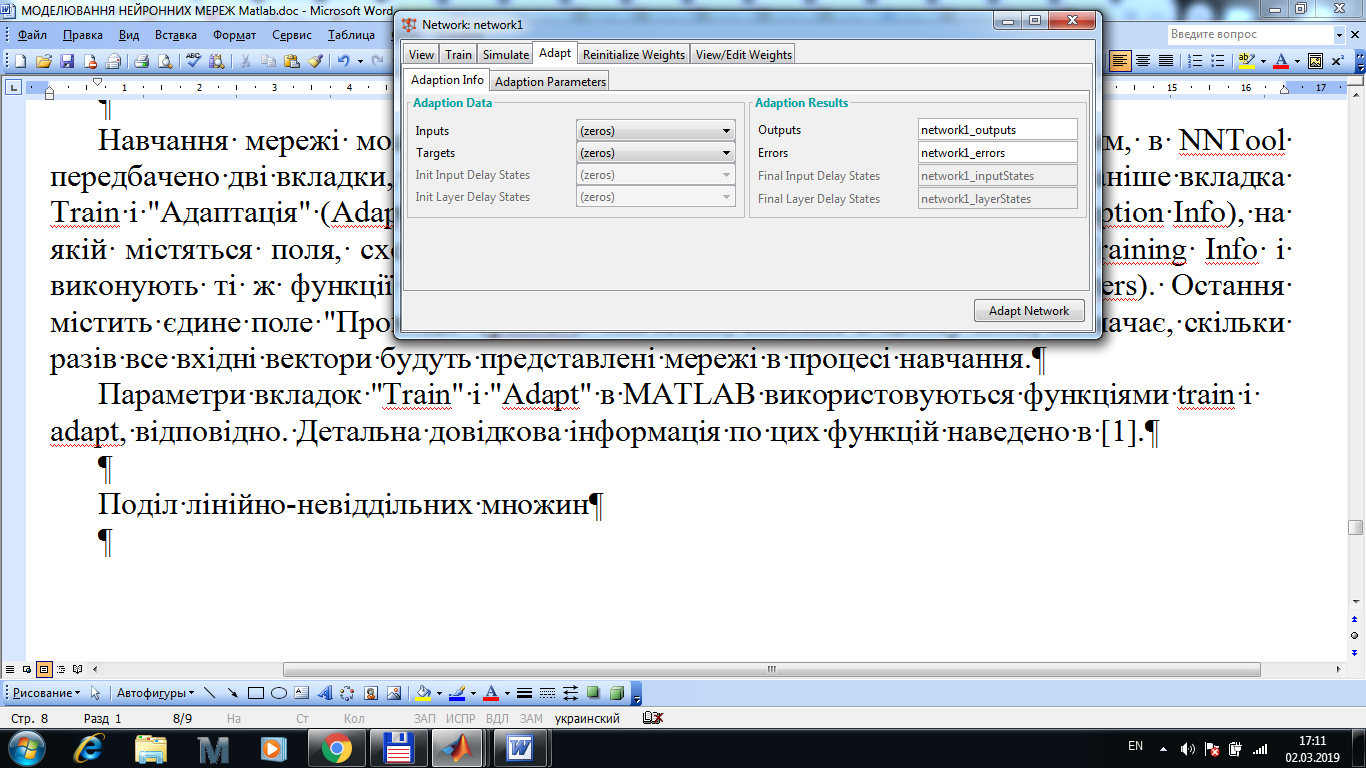
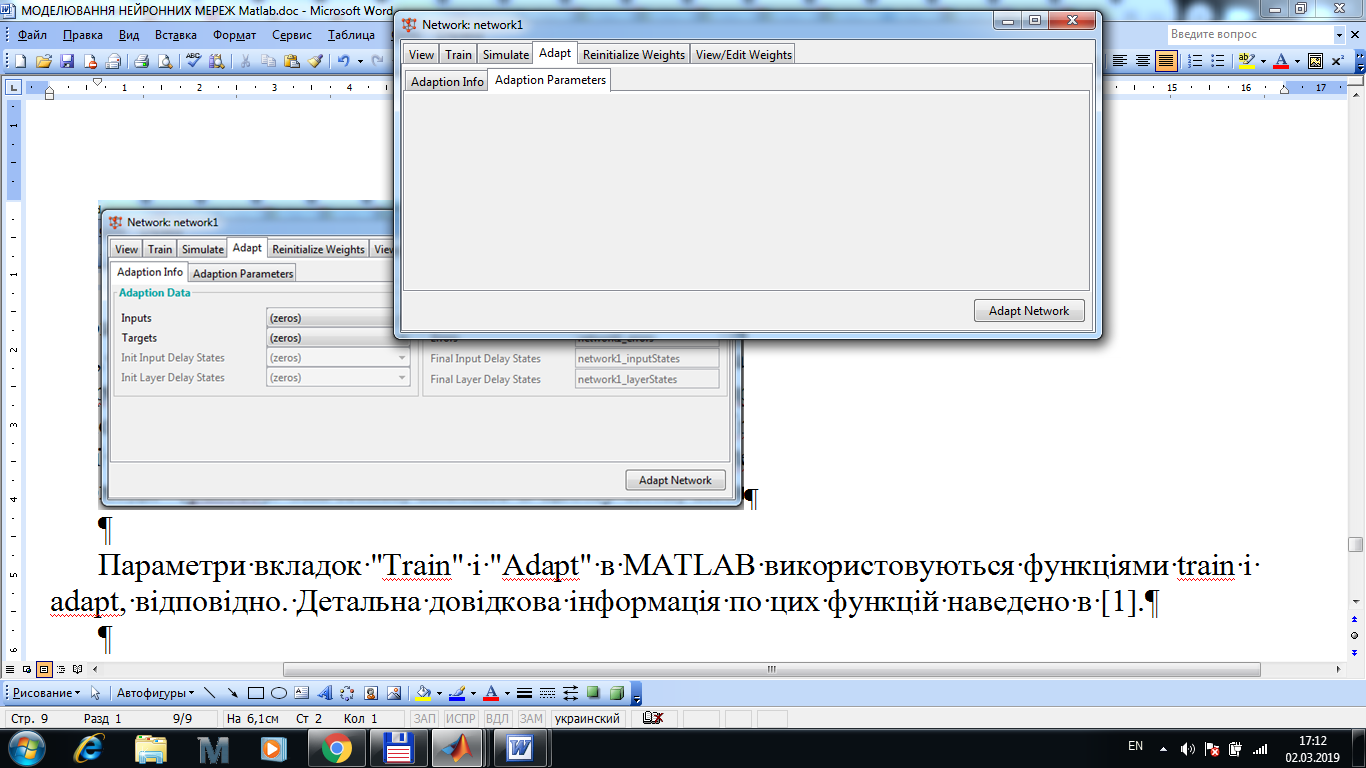
 

Рис.13

Параметри вкладок "Train" і "Adapt" в MATLAB використовуються функціями train і adapt, відповідно. Детальна довідкова інформація по цих функцій наведено в [1].

***Зробіть висновок про точність та збіжність мережі. Висновок занесіть у звіт.***

**Завдання 2.3. Створити нейронну мережу, що виконує логічну функцію "АБО ".**

Розглянемо таблицю істинності цієї функції (табл. 2).

Таблиця 2. Таблиця істинності логічної функції "АБО"

|  |  |  |
| --- | --- | --- |
| P1 | P2 | A |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

***По аналогії з пунктом завдання 3.1 навчити мережу для нових вхідних та вихідних даних. Результати занести у звіт (рис. 5-8 звіту). Зробіть висновки чи змінилася структура мережі?***

**Завдання 2.4. Створити нейронну мережу, що виконує логічну функцію "виключне АБО ".**

**Поділ лінійно-нероздільних множин**

Розглянута задача синтезу логічного елемента "І" може трактуватися як завдання розпізнавання лінійно роздільних множин. На практиці ж частіше зустрічаються завдання поділу лінійно нероздільних множин, коли застосування персептронів з функцією активації з жорстким обмеженням не дасть рішення. У таких випадках слід використовувати інші функції активації.

Показовим прикладом лінійно невіддільною завдання є створення нейронної мережі, що виконує логічну функцію "виключне АБО".

Розглянемо таблицю істинності цієї функції (табл. 3).

Таблиця 3. Таблиця істинності логічної функції "виключне АБО"

|  |  |  |
| --- | --- | --- |
| P1 | P2 | A |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Що ж мається на увазі під "лінійною нероздільністю" множин? Щоб відповісти на це питання, покажемо множину вихідних значень в просторі входів (рис. 14), слідуючи наступним правилом: поєднання входів P1 і P2, при яких вихід A звертається в нуль, позначаються кружком, а ті, при яких A звертається в одиницю - хрестиком.

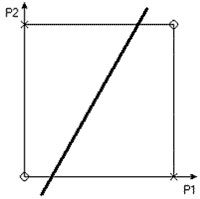


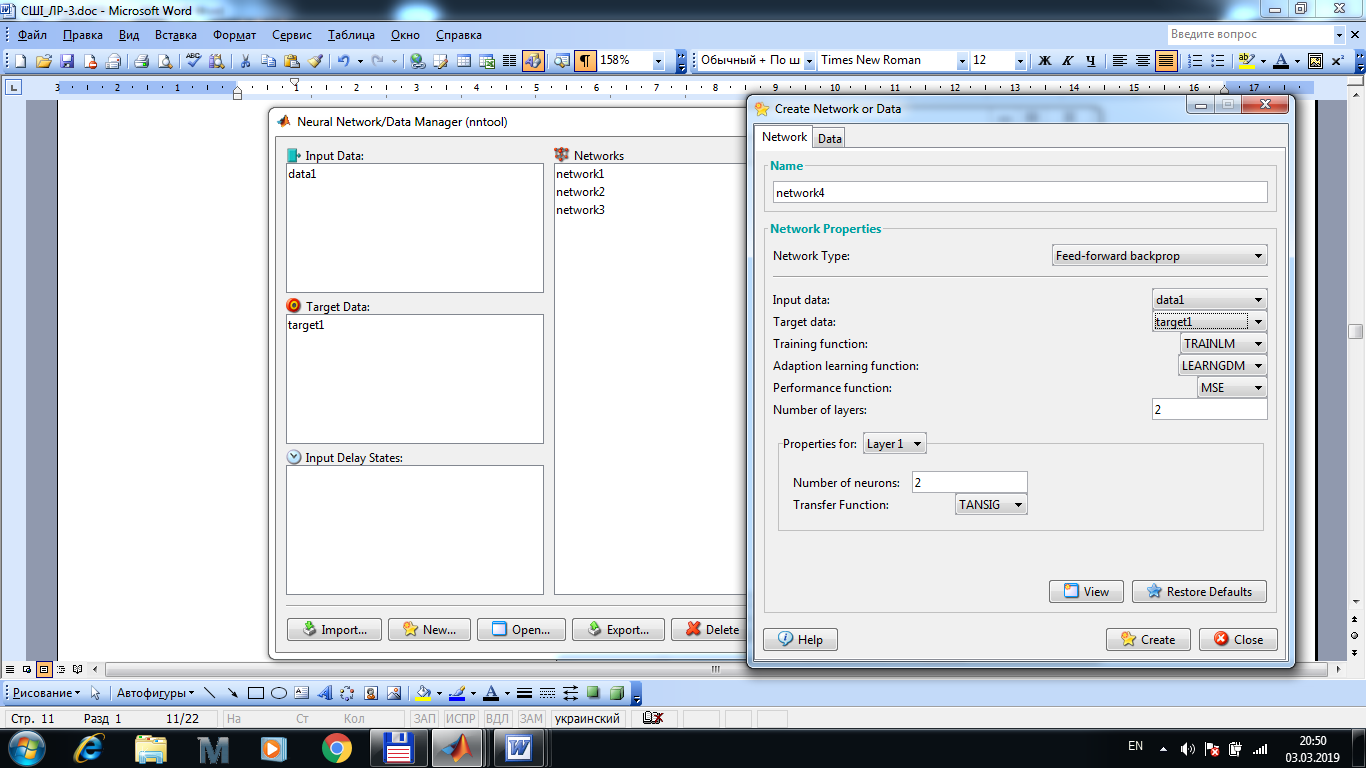
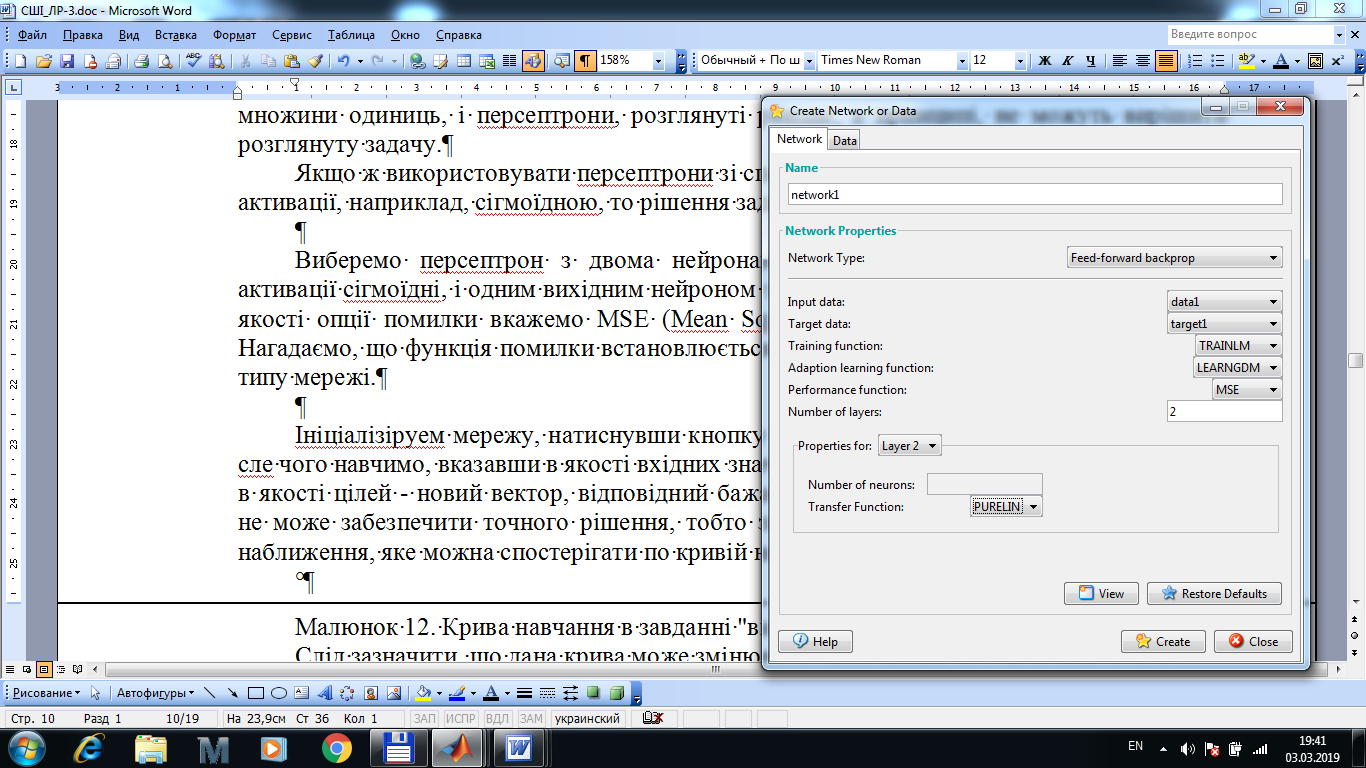
Рис 14. Стани логічного елемента "виключне АБО"

Наша мета - провести межу, що відокремлює множину нулів від множини хрестиків. З побудованої картини на рис. 14 видно, що неможливо провести пряму лінію, яка б відокремила нулі від одиниць. Саме в цьому сенсі множина нулів лінійно невіддільна від множини одиниць, і персептрони, розглянуті раніше, в принципі, не можуть вирішити розглянуту задачу.

Якщо ж використовувати персептрони зі спеціальними нелінійними функціями активації, наприклад, сігмоїдною, то рішення задачі можливо.

Введіть данні та цілі згідно таблиці 3.

Виберемо персептрон з двома нейронами прихованого шару, у яких функції активації сігмоїдні, і одним вихідним нейроном з лінійною функцією активації (рис. 15). В якості опції помилки вкажемо MSE (Mean Square Error - середній квадрат помилки). Нагадаємо, що функція помилки встановлюється у вікні "Створення мережі" після вибору типу мережі.

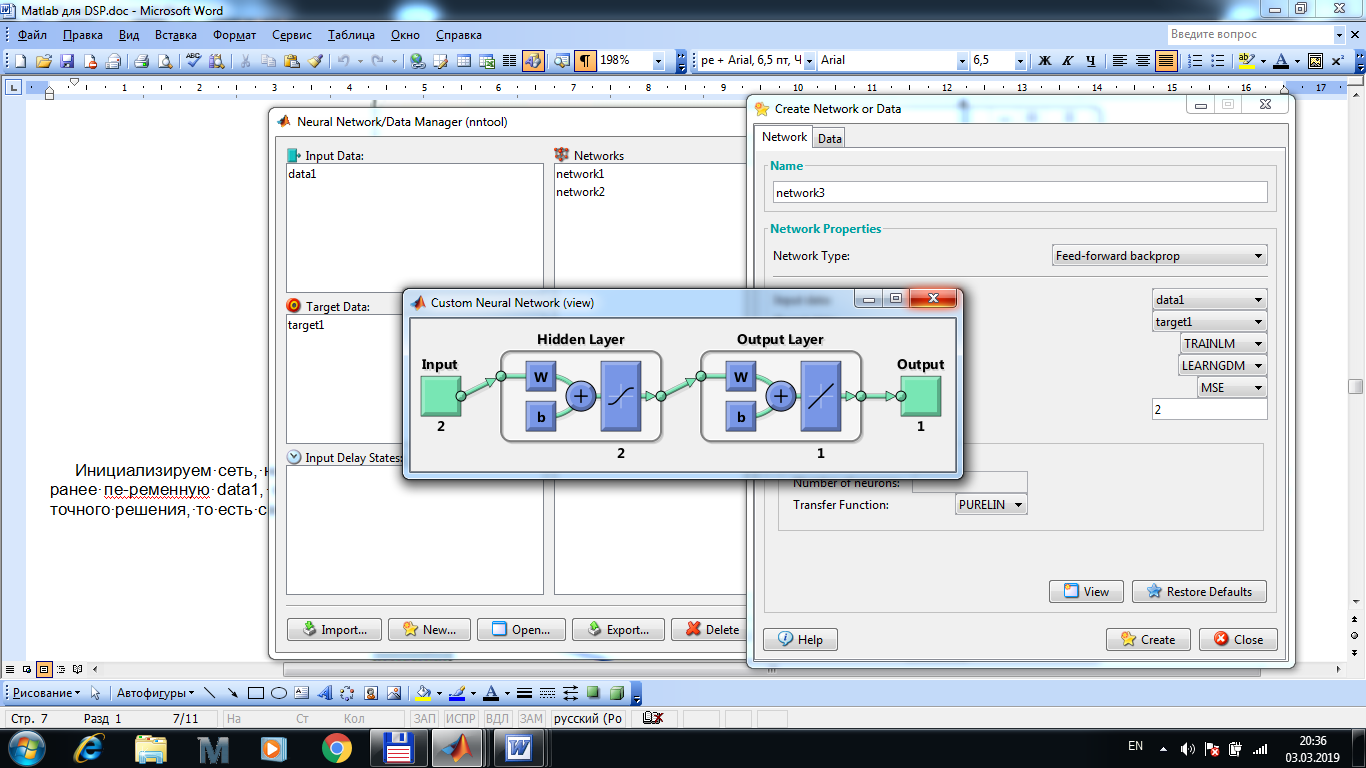


Рис. 15

Ініціалізуйте мережу, натиснувши на ній у основному вікні, після чого навчіть її (рис.16), вказавши в якості вхідних значень сформовану раніше змінну data1, в якості цілей - новий вектор, відповідний бажаним виходам.

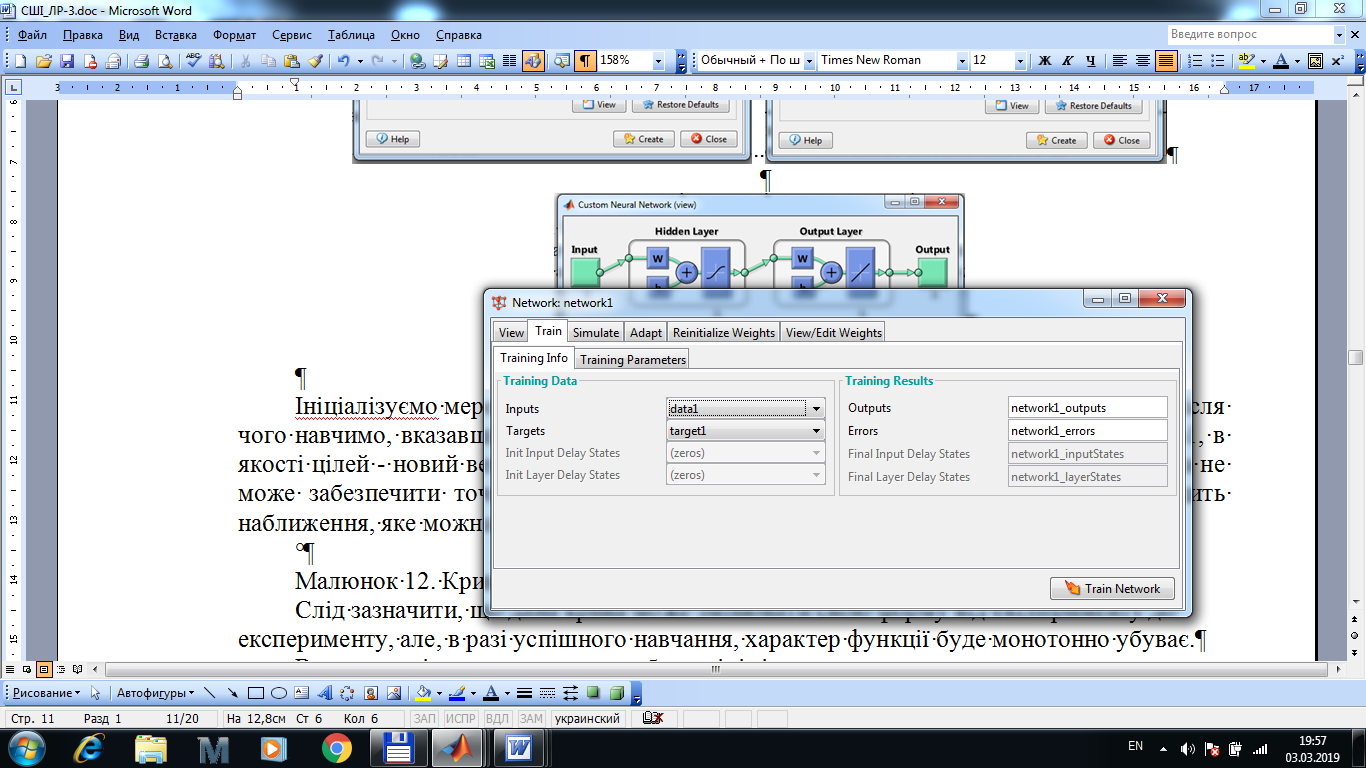


Рис. 16

В процесі навчання мережа не може забезпечити точного рішення, тобто звести помилку до нуля. Однак виходить наближення, яке можна спостерігати по кривій навчання на рис. 17.

Слід зазначити, що дана крива може змінювати свою форму від експерименту до експерименту, але, в разі успішного навчання, характер функції буде монотонно убуває.

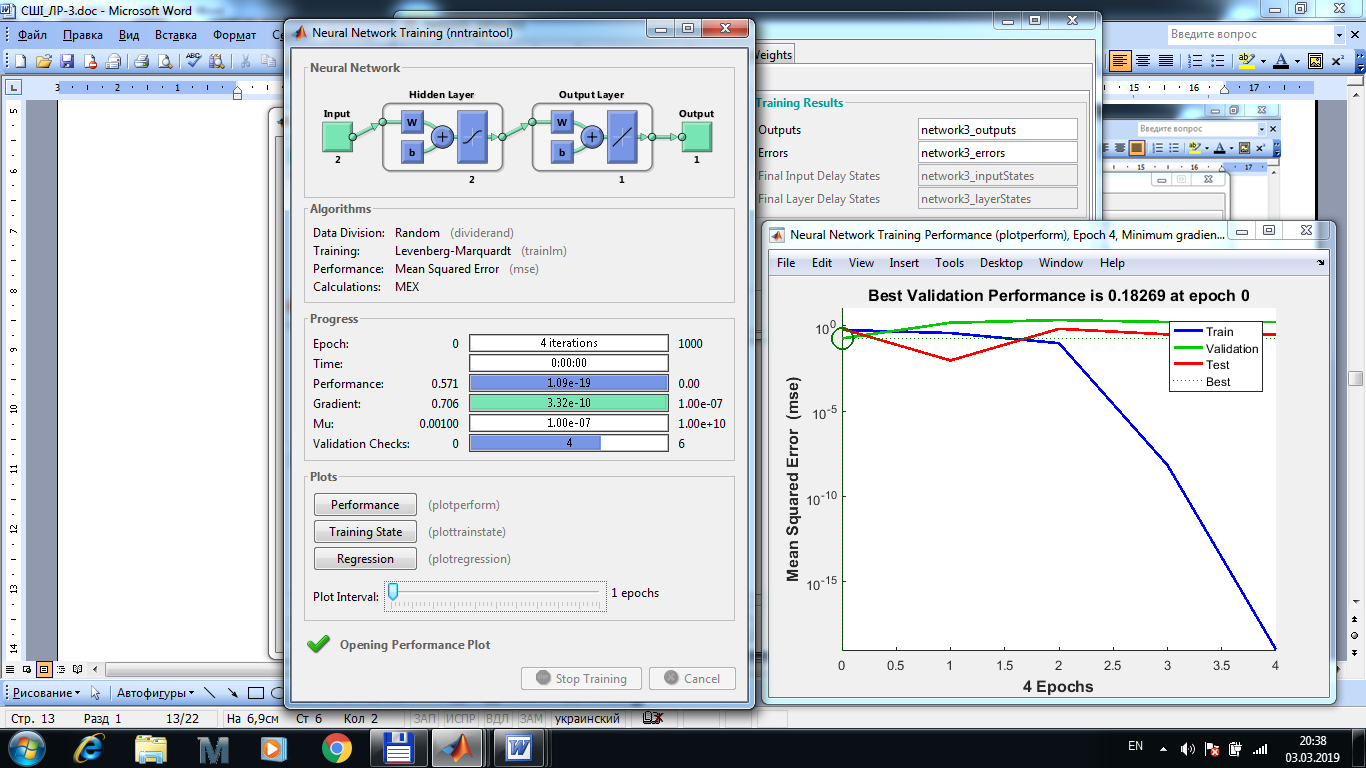
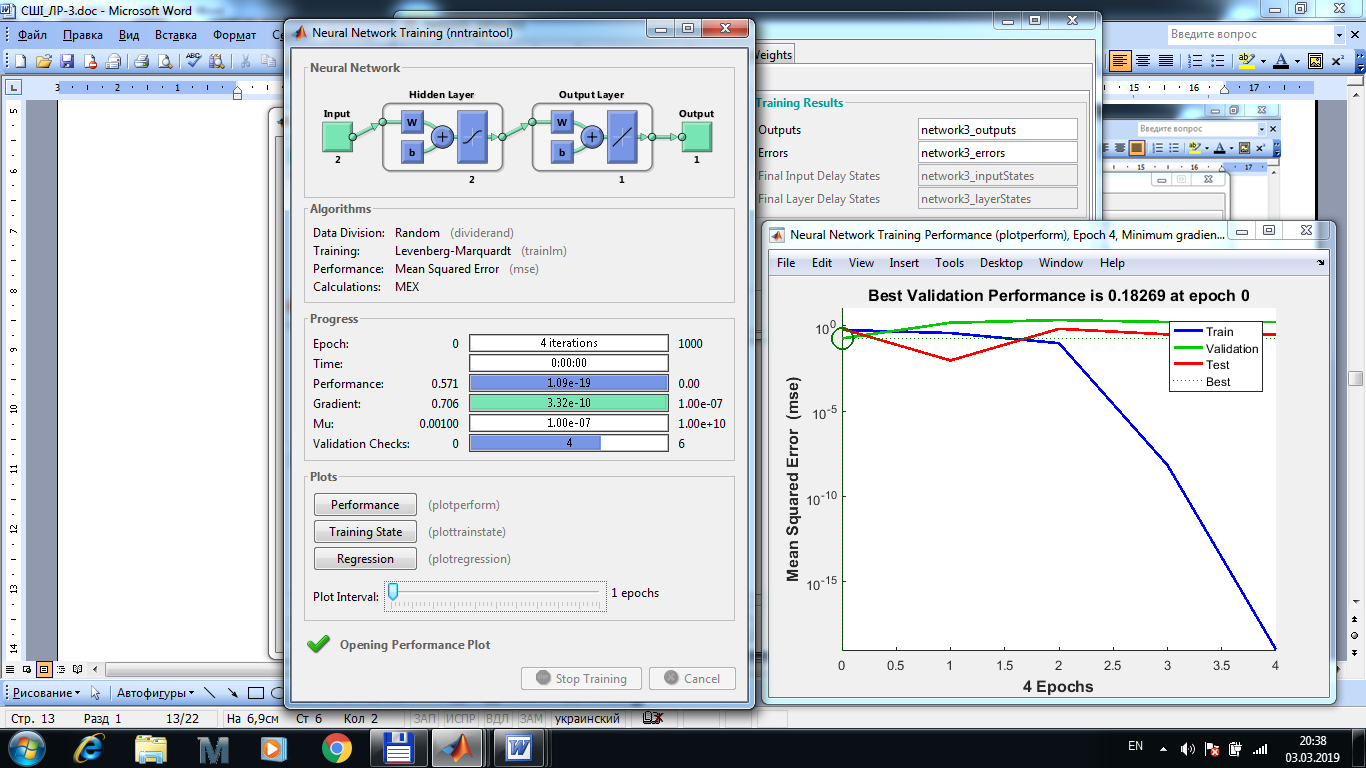
 

Рис. 17

***Зображення вікон навчання та помилок навчання (такі як на рис.17) занесіть у звіт рис. 9 та 10 звіту.***

В результаті навчання, помилка була мінімізована до вельми малого значення, яке практично можна вважати рівним нулю.

Проведемо симуляцію та отримаємо вихідний вектор (рис.18)

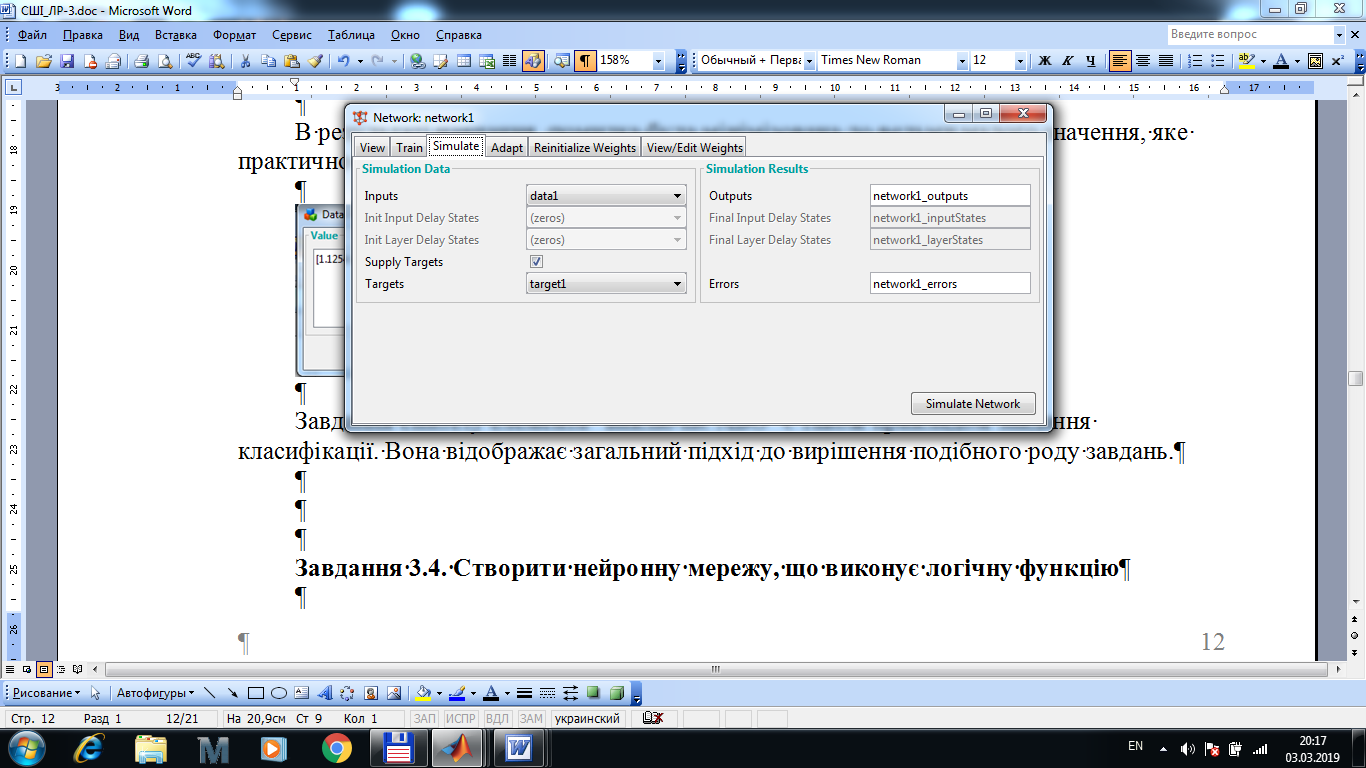


Рис. 18

Перевірте правильність функціонування мережі, для цього два рази натисніть на вектор вихідних даних у вікні Output Data та вектор цілей у вікні Target Data. ***Отримані вікна подібні до рис.19 занесіть у звіт (рис. 11-12 звіту)***

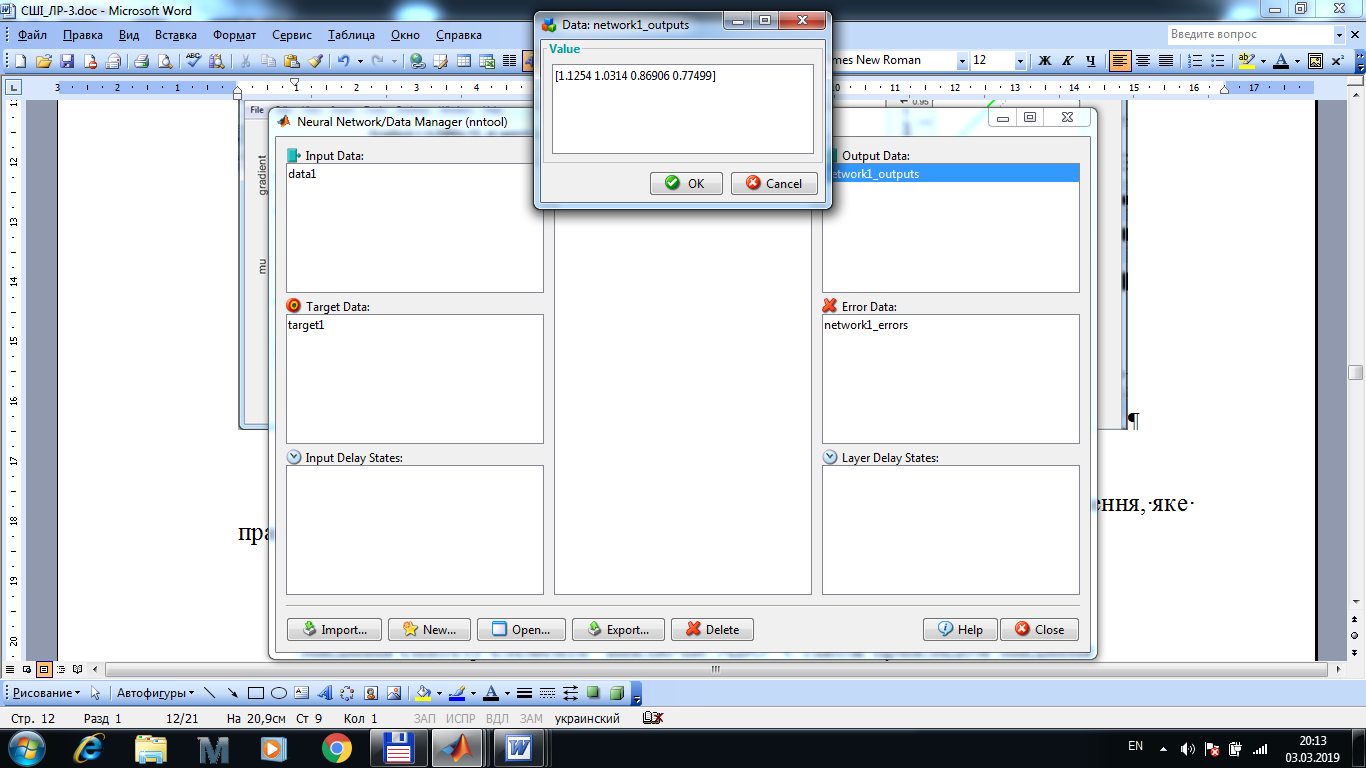


Рис. 18

***Порівняйте значення у цих вікнах та зробіть висновок про правильність (адекватність) та точність функціонування мережі. Висновок запишіть у звіт.***

**Завдання 3.4. Створити нейронну мережу, що виконує функцію**

Створити нейронну мережу, що виконує функцію ***y = x12 + x2***, якщо задані послідовності входу P = [1 0.5 0 1; -2 0 0.5 1] і цілі T = [-1 0.25 0.5 2].

Використовувати нейромережі з параметрами заданими в таблиці 4 для варіантів відповідно до номера студента за журналом

Таблиця 1.2 - Параметри нейромереж для варіантів

|  |  |  |  |
| --- | --- | --- | --- |
| Номер варіанта | Багатошаровий персептрон | | назва функції активації |
| Кількість шарів | Кількості нейронів у шарах |
| 1 | 2 | 3-1 | логістична сигмоїдна |
| 2 | 2 | 2-1 | тангенційцна сигмоїдна |
| 3 | 3 | 3-3-1 | логістична сигмоїдна |
| 4 | 2 | 5-1 | тангенційцна сигмоїдна |
| 5 | 3 | 2-2-1 | логістична сигмоїдна |
| 6 | 2 | 10-1 | тангенційцна сигмоїдна |
| 7 | 2 | 5-1 | логістична сигмоїдна |
| 8 | 3 | 5-5-1 | тангенційцна сигмоїдна |
| 9 | 3 | 3-5-1 | логістична сигмоїдна |
| 10 | 2 | 4-1 | тангенційцна сигмоїдна |
| 11 | 3 | 4-4-1 | логістична сигмоїдна |
| 12 | 3 | 3-4-1 | тангенційцна сигмоїдна |
| 13 | 2 | 6-1 | логістична сигмоїдна |
| 14 | 3 | 6-3-1 | тангенційцна сигмоїдна |
| 15 | 3 | 8-3-1 | логістична сигмоїдна |
| 16 | 2 | 7-1 | тангенційцна сигмоїдна |
| 17 | 3 | 7-7-1 | логістична сигмоїдна |
| 18 | 3 | 7-4-1 | тангенційцна сигмоїдна |
| 19 | 2 | 8-1 | логістична сигмоїдна |
| 20 | 3 | 8-8-1 | тангенційцна сигмоїдна |

**Рекомендації до виконання**

Сформуємо послідовності входів і цілей в робочій області GUI-інтерфейсу NNTool, використовуючи вікно **Create New Data**.

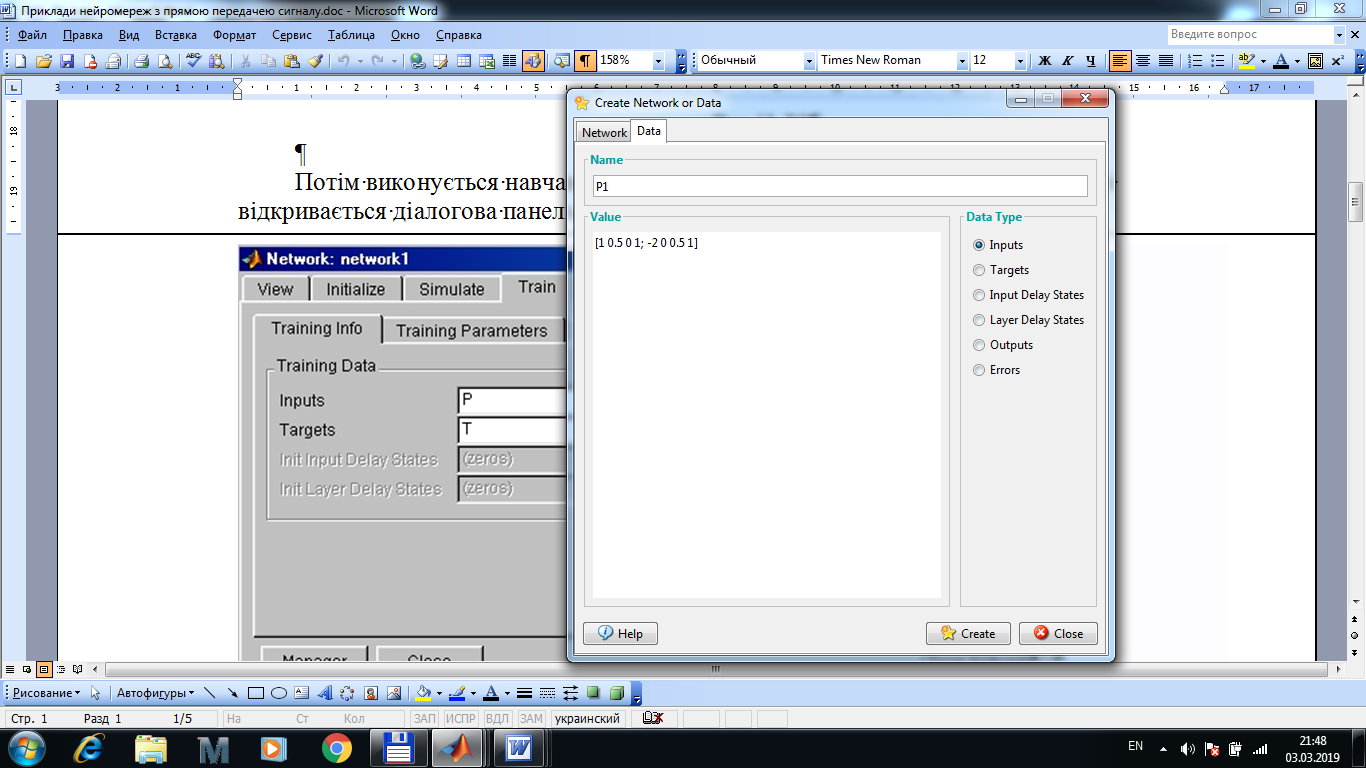
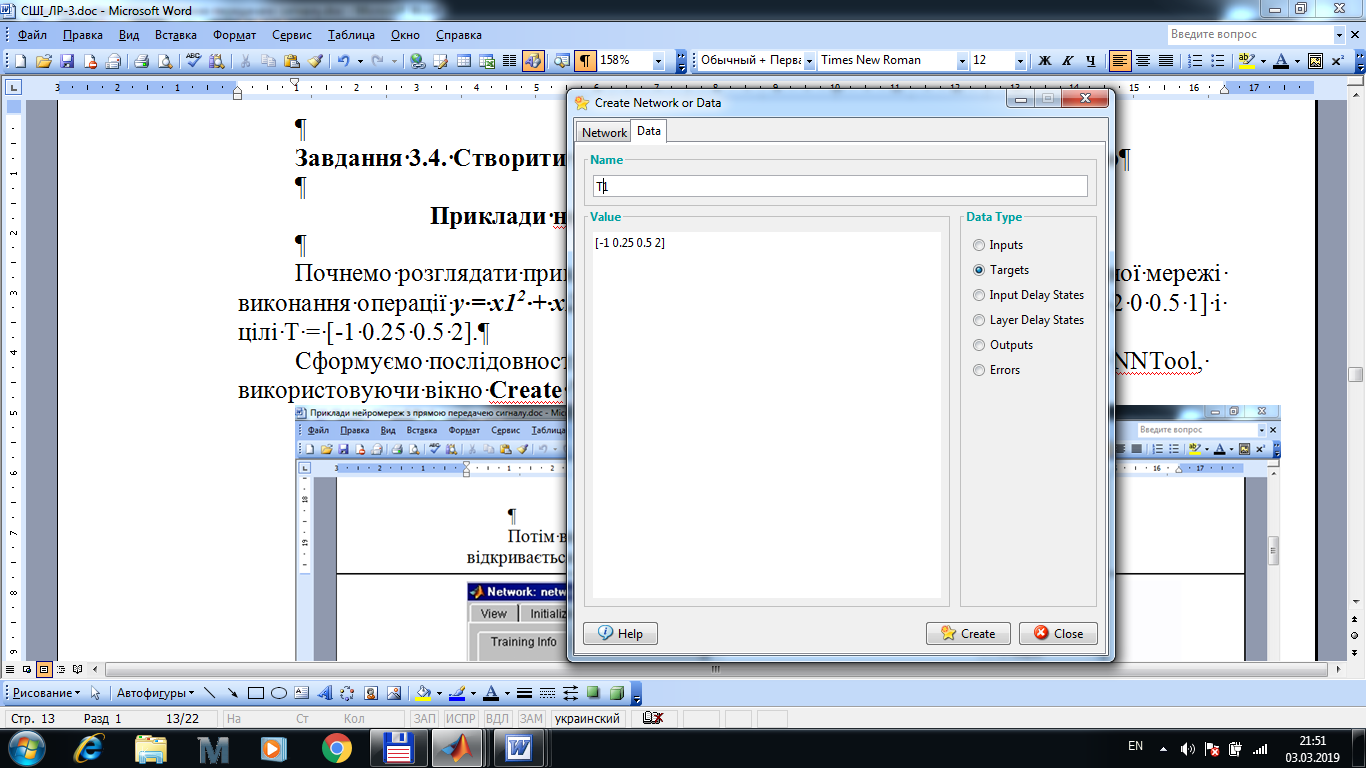
 

Рис. 19

Виберемо нейронну мережу типу **feed-forward backprop** з прямою передачею сигналу і зі зворотним поширенням помилки згідно свого варіанта. Наприклад схема мережі показана на рис. 20.

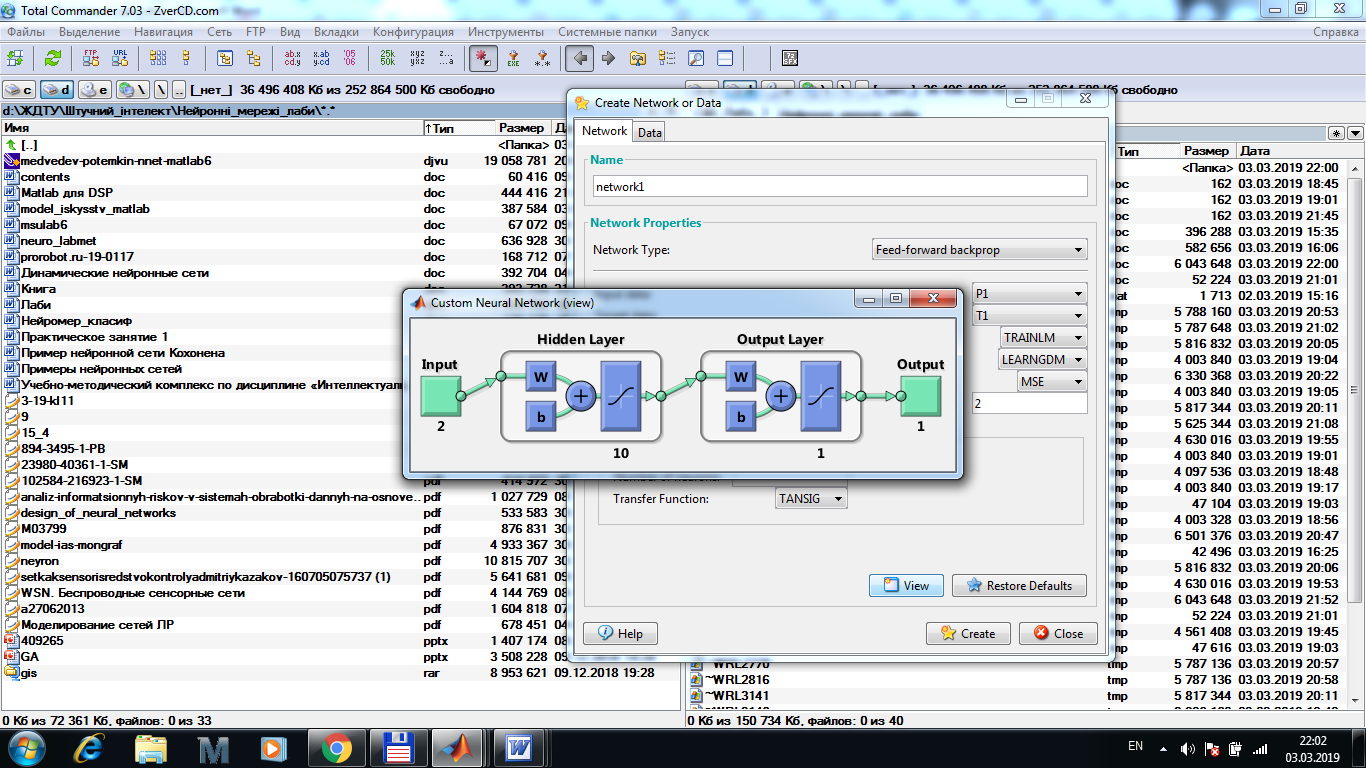


Рис.20

Виконаємо ініціалізацію мережі, для чого виберемо закладку Initialize, відкриється діалогова панель, показана на рис. 1.20. Діапазони значень вихідних даних виберемо по входах з спадаючого меню **Get from input**. Для введення встановлених діапазонів і ініціалізації ваг треба скористатися кнопками **Set Ranges** (Встановити діапазони) і **Initialize Weights** (Ініціалізувати ваги). Якщо потрібно повернутися до колишніх діапазонах, то слід вибрати кнопки **Revert Ranges** (Повернути діапазони) і **Revert Weights** (Повернути ваги).

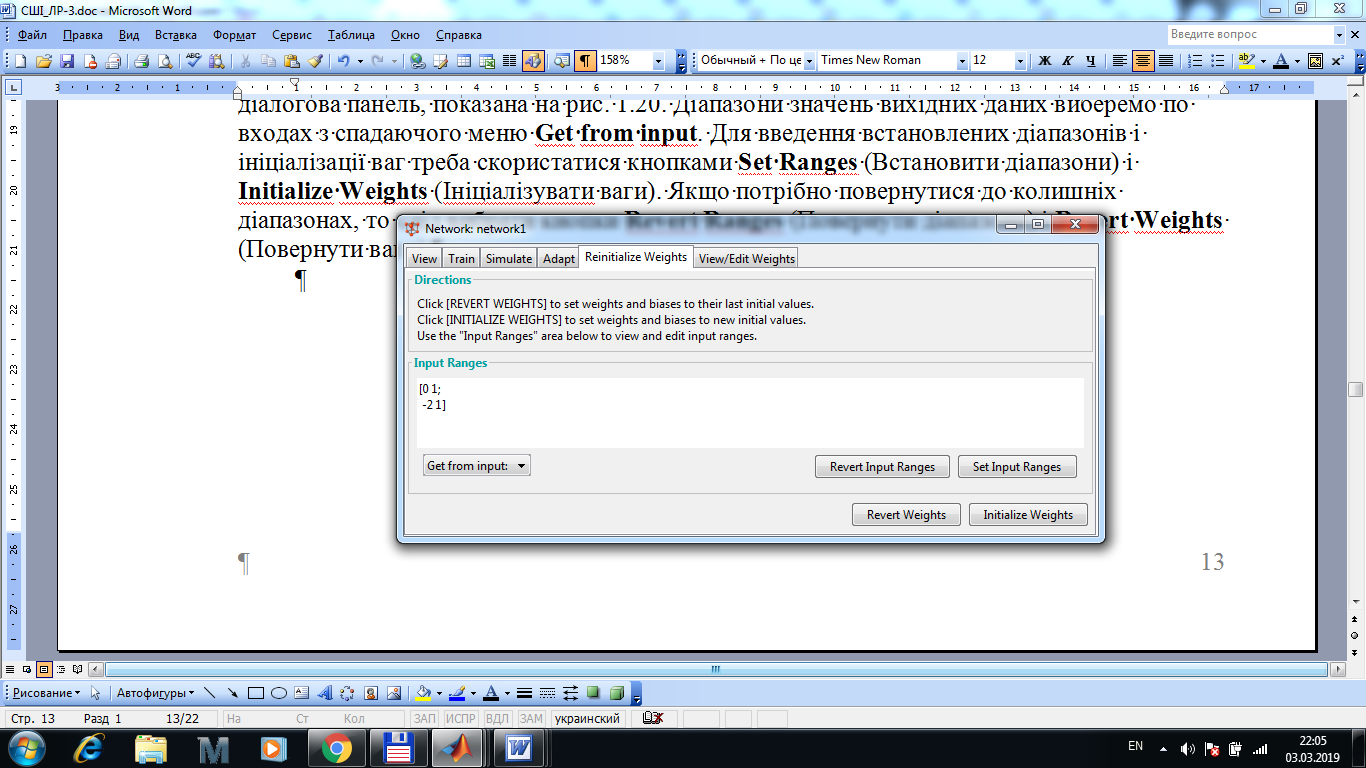


Рис. 1.20

Потім виконується навчання мережі, для чого вибирається закладка **Train** і відкривається діалогова панель.

Панель має дві закладки:

**Training Info** (Інформація про навчальні послідовності);

**Training Parametrs** (Параметри навчання);

Застосовуючи ці закладки, можна встановити імена послідовностей входу і цілі, а також параметрів процедури навчання.

Тепер можна приступити до навчання мережі (кнопка **Train Network**).

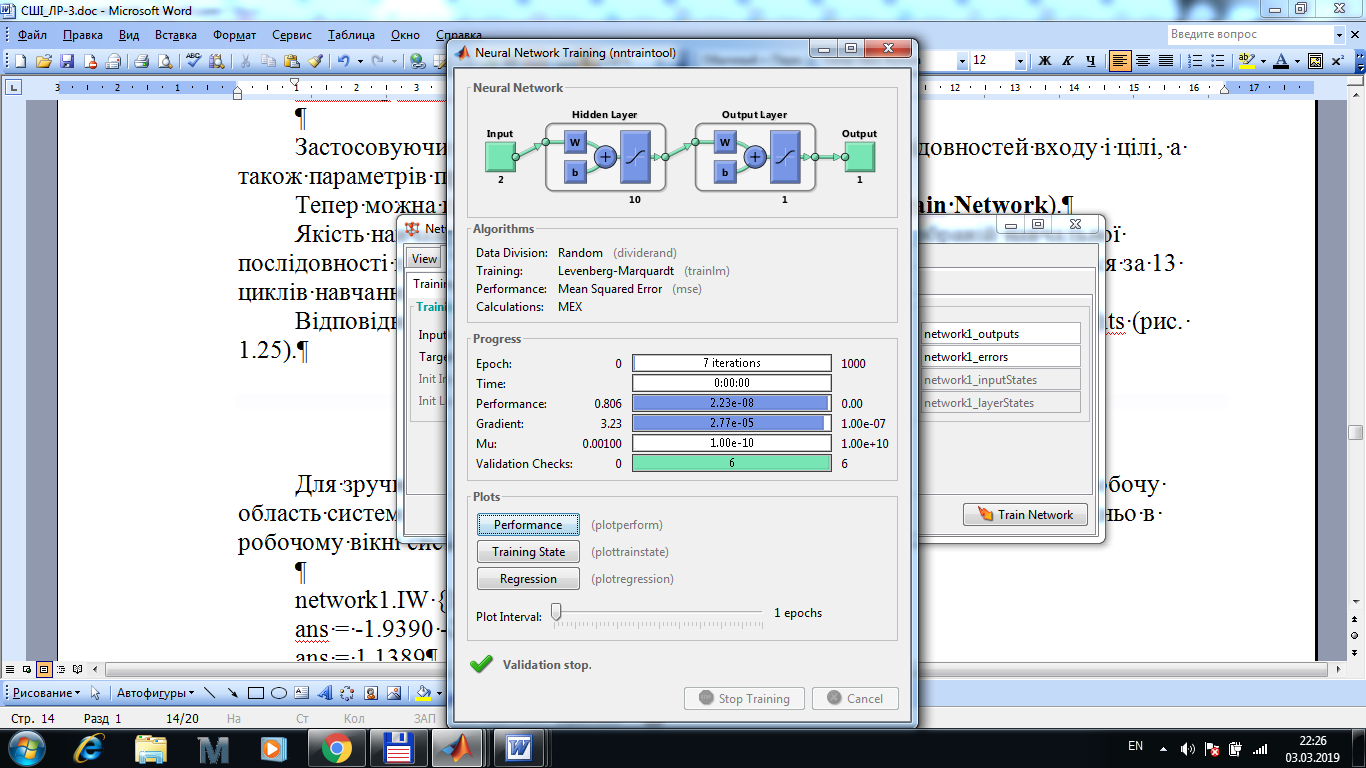


Рис.21

Якість навчання мережі з прямою передачею сигналу на обраній навчальної послідовності пояснюється на рис. 1.22. Практично нульова точність досягається за 7 циклів навчання.

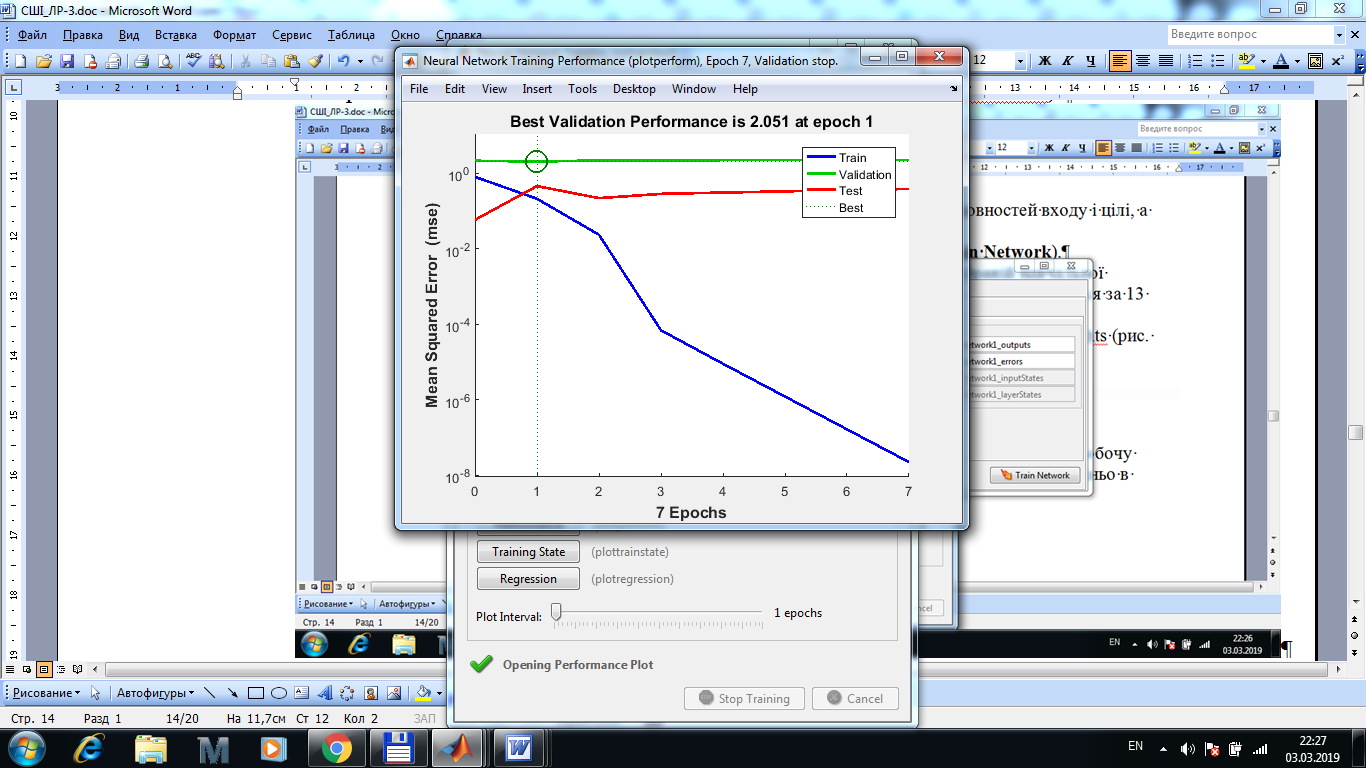


Рис.22

***Зображення вікон навчання та помилок навчання (такі як на рис.21-22) занесіть у звіт рис. 13 та 14 звіту.***

Відповідні ваги і зміщення можна побачити, якщо вибрати закладку Weights

Для зручності роботи можна експортувати створену нейронну мережу в робочу область системи MATLAB і отримати інформацію про ваги і зсувах безпосередньо в робочому вікні системи:

network1.IW {1, 1}, network1.b {1}

ans = -1.9390 -2.2747

ans = 1.1389

network1.LW {2, 1}, network1.b {2}

ans = -1.5040

ans = 0.5024

Перевірте правильність функціонування мережі, для цього два рази натисніть на вектор вихідних даних у вікні Output Data та вектор цілей у вікні Target Data. ***Отримані вікна подібні до рис.23 занесіть у звіт (рис. 15-16 звіту)***

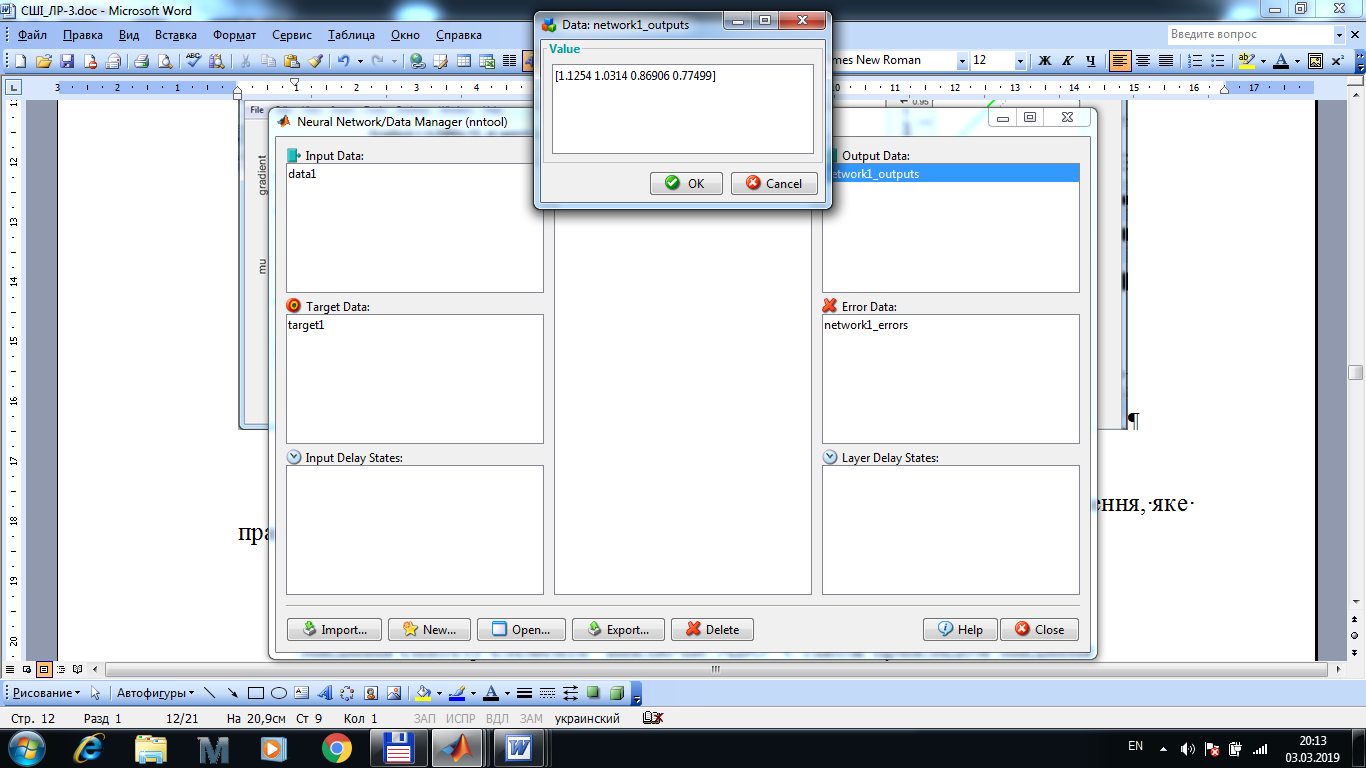


Рис. 23

***Порівняйте значення у цих вікнах та зробіть висновок про правильність (адекватність) та точність функціонування мережі. Висновок запишіть у звіт.***

**Завдання 2.6. Програмування feedforward нейронної мережі, у командному режимі без використання графічного інтерфейсу MATLAB**

**Складіть та виконайте програму по класифікації даних**

% Задамо першу множину значень кластера №1.

% Центром цього кластера є точка (2; 0) з розкидом від центру rand (1,50) і кількістю відліків 50.

% Т.n x1 і y1 буде масивом розмірність 1х50

x1=2+rand(1,50)

y1=0+rand(1,50)

plot(x1,y1,'or')

% Також задамо другу множину кластера №2 з центром (-2;0)

x2=-2+rand(1,50)

y2=0+rand(1,50)

plot(x2,y2,'ob')

% Задамо третю множину значень кластера №3 з центром (0;2)

x3=0+rand(1,50)

y3=2+rand(1,50)

plot(x3,y3,'oy')

% Задамо четверту множину значень кластера №4 з центром (0;2)

x4=0+rand(1,50)

y4=-2+rand(1,50)

plot(x4,y4,'og')

% Побудуємо графік figure (1)

figure (1)

hold on

plot(x1,y1,'or')

plot(x2,y2,'ob')

plot(x3,y3,'oy')

plot(x4,y4,'og')

grid on

hold off

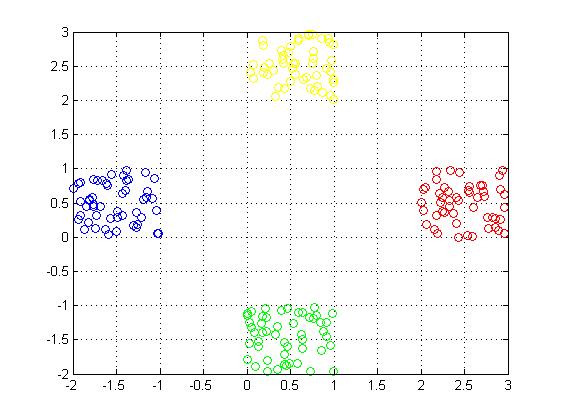


Рис.24

***Отриманий графік подібний до рис.24 занесіть у звіт (рис. 17 звіту)***

% Задамо цілі чотирьох кластерів, тобто яке значення буде видавати

% Нейронна мережа при симуляції

T1(1:50)=1;

T2(1:50)=2;

T3(1:50)=3;

T4(1:50)=4;

% Поєднаємо всі цілі кластерів

T(1:50)=T1;

T(51:100)=T2;

T(101:150)=T3;

T(151:200)=T4;

% З'єднаємо всю базу знань для нейронної системи в одну матрицю,

% А саме поєднаємо x

x(1:50)=x1;

x(51:100)=x2;

x(101:150)=x3;

x(151:200)=x4;

% поєднаємо y

y(1:50)=y1;

y(51:100)=y2;

y(101:150)=y3;

y(151:200)=y4;

% і все це з'єднаємо в z

z(1,1:200)=x

z(2,1:200)=y

% перевіримо отримані розміри

size(T)

size(z)

% Створимо сруктуру feedforward нейронної мережі, а саме створимо

% трьохслойну нейронну систему з 10-ю нейронами на першому шарі, 2-мя

% нейронами на другому шарі і 1-ним нейроном на третьому шарі.

% z - база знань спектра

% T - ціль мережі

% logsig, tansig, purelin - вид активації шару

% 1-ый слой logsig, 2-ой слой logsig, 3-ий слой purelin

net = newff(z,T,[10,2],{'logsig','logsig'})

% Навчимо створену мережу

net = train(net,z,T);

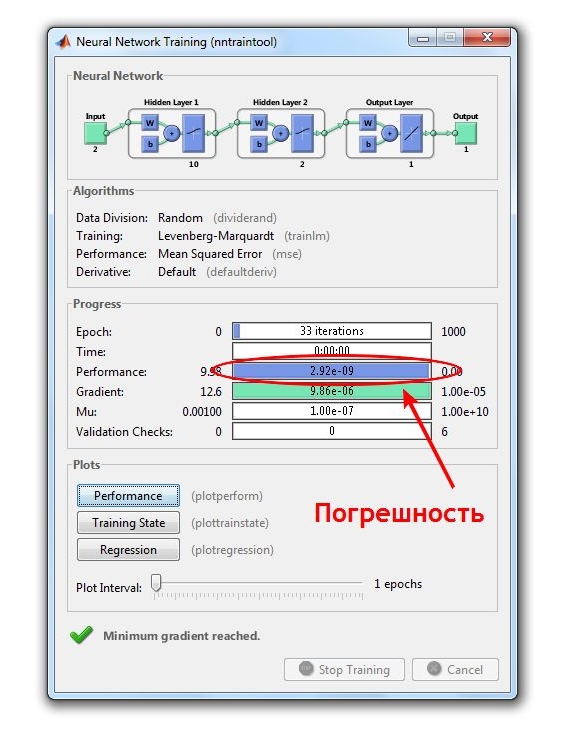


Рис.25

***Отримане вікно подібне до рис.25 занесіть у звіт (рис. 18 звіту)***

% І перевіримо мережу на спектрах за допомогою яких мы навчали мережу

% Перевірка першого кластера

a=sim(net,z(:,1:50))

100% результат

% Перевірка другого кластера

a=sim(net,z(:,51:100))

100% результат

% Перевірка третього кластера

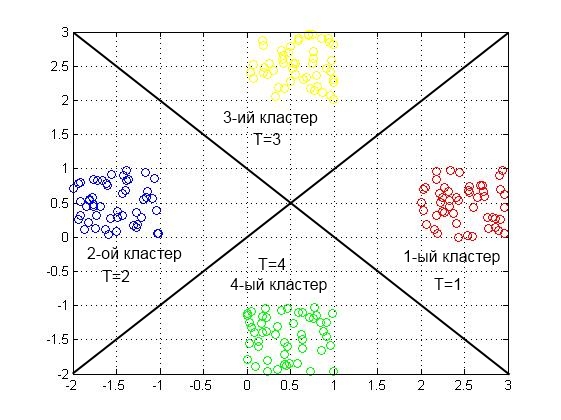
a=sim(net,z(:,101:150))

100% результат

% Перевірка четвертого кластера

a=sim(net,z(:,151:200))

100% результат



Максимальна помилка 0.0002

Також можете перевірити систему на інших реалізаціях, результат системи буде залежати від відстані між вектором, який ви подаєте і центром кластера.

***Текст програми занесіть у звіт***

***Зробіть висновки по роботі у яких укажіть:***

- призначення функції активації;

- про можливості нейронних мереж прямого поширення.

***Назвіть бланк звіту ШІ-КБ-ЛР-2-NNN-XXXXX.doc***

***де NNN – номер групи***

***XXXXX – позначення прізвища студента.***

***Переконвертуйте файл звіту в ШІ-КБ-ЛР-2-NNN-XXXXX.pdf***

**3. ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ**

1. Поняття: нейрон, нейромережа, нейрокомп'ютер, нейроінформатика.

2. Класифiкацiя та види моделей нейромереж.

3. Властивостi штучних нейромереж.

4. Загальне уявлення про навчання нейромереж.

5. Загальна характеристика та принципи побудови нейромереж.

6. Характеристики процесу навчання нейромереж.

7. Лінійна роздільність і лінійна нерозділеність класів.

8. Порiвняння моделей та алгоритмiв навчання нейромереж прямого поширення.

9. Нейроннi мережi у пакетi MATLAB. Модуль Neural Network Toolbox.