САМОСТІЙНА РОБОТА
з дисципліни
“Нейроінформаційні мережі керування біотехнічними обєктами”
(Модуль 2)

Тривалість роботи – 40 годин.

1. Графічний інтерфейс гібридних систем

Графічний інтерфейс гібридних (нечітких) нейронних систем викликається функцією (з режиму командного рядка) **anfisedit.** Виконання функції призводить до появи вікна редактори гібридних систем (**ANFIS Editor** або **ANFIS-редактор**) (рис. 1.1).

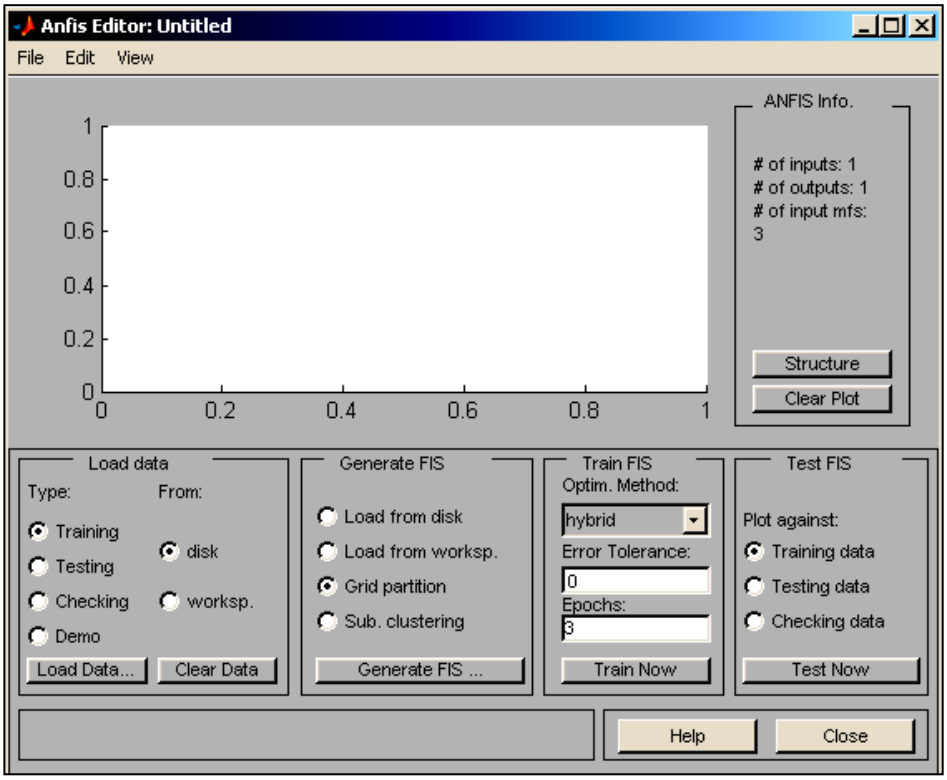

Рис. 1.1. Вікно редактора гібридних (нечітких) нейронних систем

За допомогою даного редактора здійснюється створення або завантаження структури гібридної системи, перегляд структури, настроювання її параметрів, перевірка якості функціонування такої системи. Створення структури, настроювання параметрів і перевірка здійснюються за вибірками (наборами даних) – навчальною (Training data), перевірочною (Checking data) і тестувальною (Testing data), які попередньо повинні бути представлені у вигляді текстових файлів (з розширенням **dat** і розділенням табуляціями). Рекомендується створювати їх у текстовому редакторі “Блокнот”; перші стовпчики відповідають вхідним змінним, а останній (лівий) – єдиній вихідній змінній; кількість рядків у таких файлах дорівнює кількості зразків (прикладів) (рис. 1.2).

Чітких рекомендацій щодо обсягів зазначених вибірок не існує, очевидно, найкраще виходити з принципу «чим більше, тим краще». Навчальні і перевірочна вибірки безпосередньо задіюються в процесі настроювання

параметрів гібридної мережі. Перевірочна – для з'ясування ситуації: чи має місце перенавчання мережі, при якому помилка для навчальної послідовності прямує до нуля, а для перевірочної – зростає; втім, наявність перевірочної вибірки не є строго необхідною, вона лише бажана. Тестувальна вибірка застосовується для перевірки якості функціонування навченої мережі.

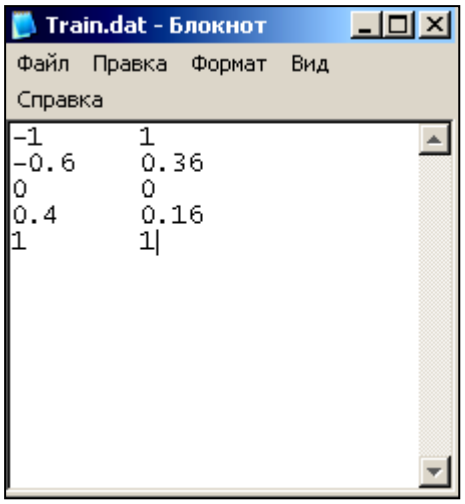

Рис. 1.2. Створення у текстовому редакторі “Блокнот” вибірок для ANFIS-editor

Пояснимо пункти меню й опції редактора. Пункти меню **File** і **View**, у загальному ідентичні аналогічним пунктам **FIS-редактора**, за тим виключенням, що тут робота може відбуватися тільки з алгоритмом нечіткого висновку **Sugeno**. Пункт меню **Edit** містить єдиний підпункт – **Undo** (скасувати виконану дію).

Набір опцій **Load data** (завантажити дані) у нижній лівій частині вікна редактора містить у собі:

– тип (**Type**) завантажуваних даних (для навчання – **Training**, для тестування – **Testing**, для перевірки – **Checking**, демонстраційні – **Demo**);

– місце, звідкіля повинні завантажуватися дані (з диска – **disk** або з робочої області **MATLAB-workspace**).

До даних опцій відносяться дві кнопки, натискання на які призводить до необхідних дій – **Load Data...** (завантажити дані) і **Clear Data** (очистити, тобто стерти введені дані).

Наступна група опцій (у середині нижньої частини вікна **ANFIS-редактора**) об'єднана під ім'ям **Generate FIS** (створення нечіткої системи висновку). Дана група містить у собі опції:

– завантаження структури системи з диска (**Load from disk**);

– завантаження структури системи з робочої області MATLAB (**Load from worksp.**);

– розбивка (розподіл) областей визначення вхідних змінних (аргументів) на підобласті – незалежно для кожного аргумента (**Grid partition**);

– розбивка всієї області визначення аргументів (вхідних змінних) на підобласті – у комплексі для всіх аргументів (**Subtract clustering** або **Sub.clustering**), а також кнопку **Generate FIS**, натискання якої призводить до процесу створення гібридної системи.

Наступна група опцій – **Train FIS** (навчання нечіткої системи висновку) – дозволяє визначити метод «навчання» (**Optim. Method**) системи (тобто метод настроювання її параметрів) – гібридний (**hybrid**) або зворотного поширення помилки (**backprop**), встановити рівень поточної сумарної (за всіма зразками) помилки навчання (**Error Tolerance**), при досягненні якої процес навчання закінчується і кількість циклів навчання (**Epochs**), тобто кількість «прогонів» всіх зразків (або прикладів) навчальної вибірки; процес навчання, у такий спосіб закінчується або при досягненні визначеного рівня помилки навчання, або при проведенні заданої кількість циклів.

Кнопка **Train Now** (почати навчання) активізує процес настроювання параметрів гібридної мережі.

У правому верхньому куті вікна **ANFIS-редактора** видається інформація (**ANFIS Info.**) щодо спроектованої системи: кількість входів, виходів, функцій належності входів; натискання кнопки **Structure** (структура) дозволяє побачити структуру мережі. Кнопка **Clear** (очистити) дозволяє стерти всі результати.

Опції **Test FIS** у правому нижньому куті вікна дозволяють провести перевірку і тестування створеної і навченої системи з виведенням результатів у вигляді графіків. Відповідні графіки для навчальної вибірки – Training data, тестувальної вибірки – Testing data і перевірочної вибірки – Checking data. Кнопка **Test Now** дозволяє запустити зазначені процеси.

Роботу з редактором розглянемо на прикладі відновлення залежності $y = x^2$ (рис. 1.2). Припустимо, що ці дані збережені у файлі **Train.dat**. Створення і перевірку системи, як і раніш, проведемо поетапно.

1. У вікні **ANFIS-редактора** виберемо тип завантажуваних даних **Training** і натиснемо кнопку **Load data**. У наступному стандартному вікні діалогу вкажемо місце розташування і ім'я файлу. Його відкриття призводить до появи в графічній частині вікна редактора набору точок, що відповідають введеним даним (рис. 1.3).

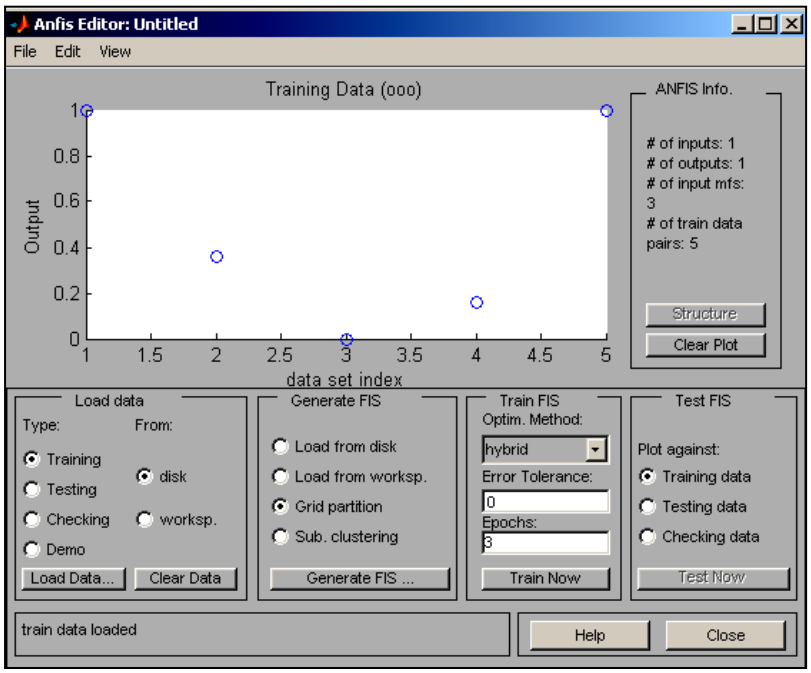

Рис. 1.3. Вікно ANFIS-редактора після завантаження навчальної вибірки

2. У групі опцій **Generate FIS** за замовчуванням активізована опція **Grid partition**. Не будемо її змінювати і натиснемо кнопку **Generate FIS**, після чого з'явиться діалогове вікно (рис. 1.4) для задання числа і типів функцій належності. Збережемо всі установки за замовчуванням, погодившись з ними натисканням кнопки **ОК**. Відбудеться повернення в основне вікно **ANFIS-редактора**. Тепер структура гібридної мережі створена, її графічний вигляд можна переглянути за допомогою кнопки **Structure** (рис. 1.5).

3. Перейдемо до опцій **Train FIS**. Не будемо змінювати налаштування параметрів, які задаються по замовчанням: метод (hybrid – гібридний) і рівень помилки (0), але кількість циклів навчання змінимо на 40, після чого натиснемо кнопку початку процесу навчання (**Train Now**). Результат у вигляді графіка помилки мережі в залежності від числа проведених циклів навчання (з якого випливає, що фактично навчання закінчилося після п'ятого циклу) представлений на рисунку 1.6.

4. Тепер натисканням кнопки **Test Now** можна почати процес тестування навченої мережі, але, оскільки використовувався тільки одна навчальна вибірка, вихід навченої системи практично збігається з точками навчальної вибірки (рис. 1.7).

5. Збережемо розроблену систему у файл на диску з ім'ям **Train** (з розширенням **fis**) і, для дослідження розробленої системи засобами FIS-редактора з командного рядка MATLAB, виконаємо команду **fuzzy,** а потім через пункти меню **File/Open FIS from disk...** відкриємо створений файл. Зі створеною системою можна тепер виконувати всі прийоми редагування (зміна імен змінних тощо) і дослідження. Не важко, до речі, переконатися, що якість апроксимації даних істотно не покращилася – занадто мало даних.

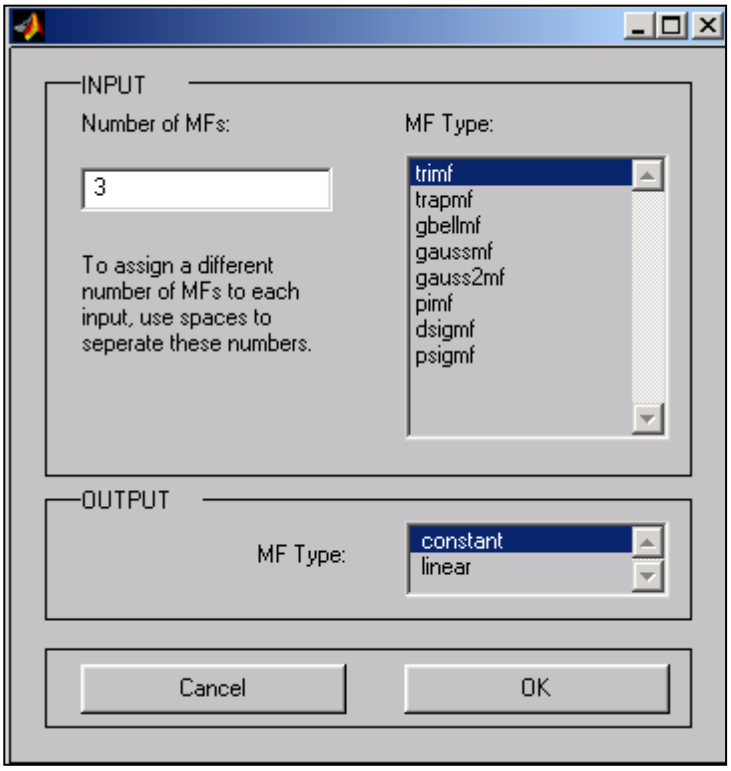

Рис. 1.4. Вікно задання функцій належності

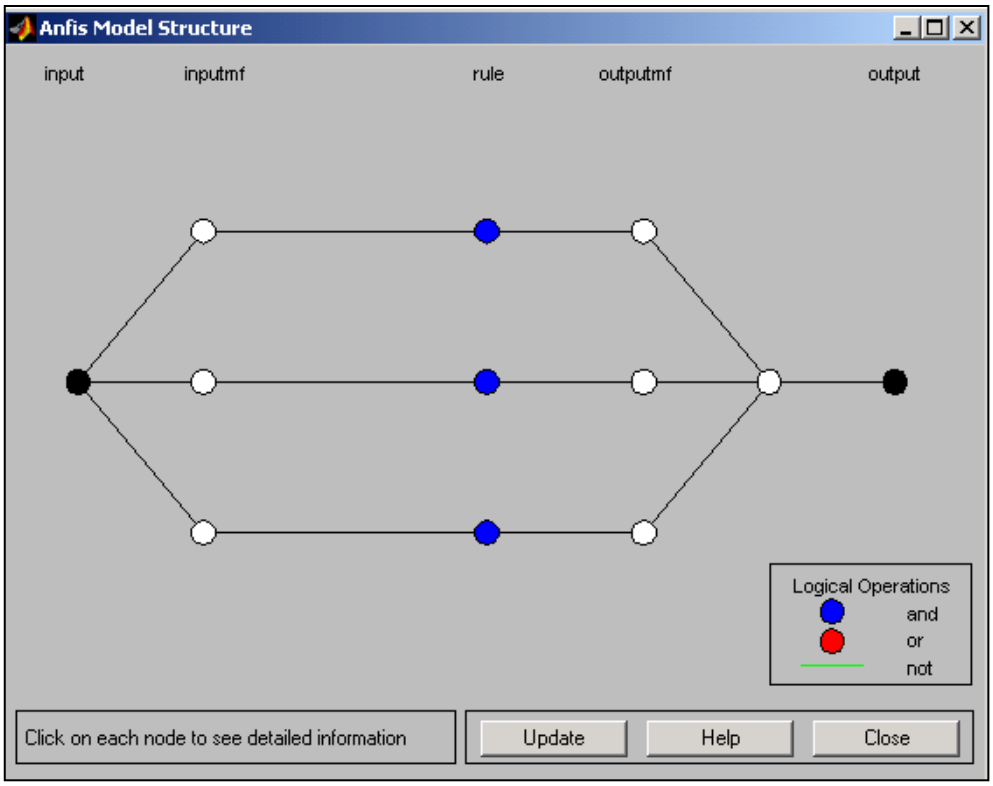

Рис. 1.5. Структура створеної гібридної мережі

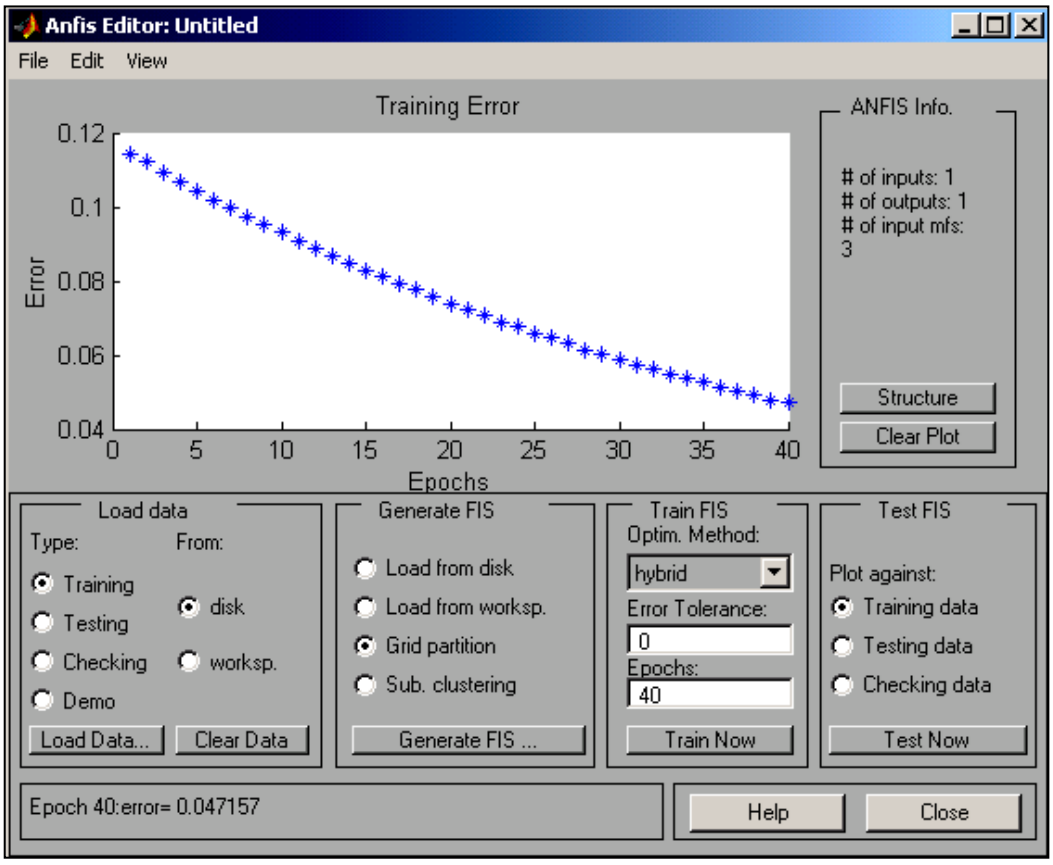

Рис. 1.6. Результат навчання мережі

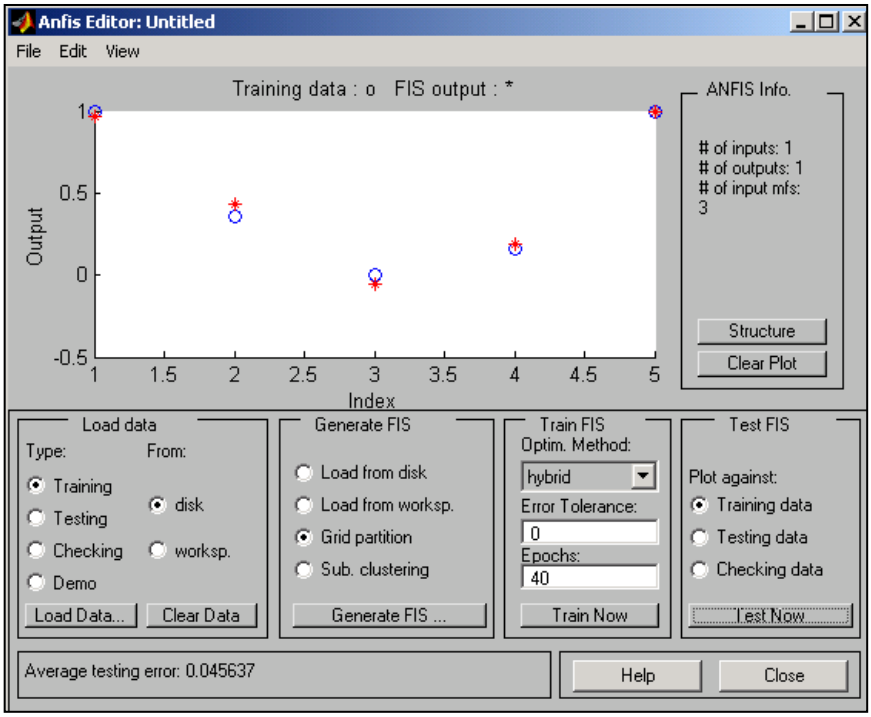

Рис. 1.7. Результат тестування навченої системи

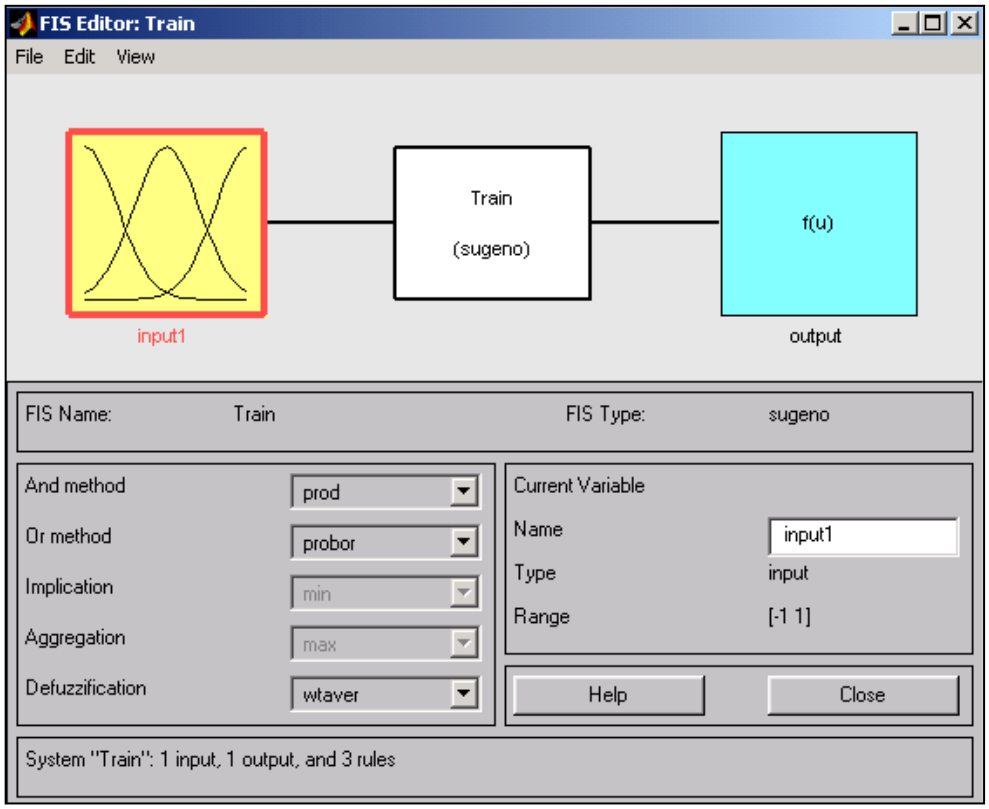

Рис. 1.7. Завантаження створеного файлу **Train** командою **fuzzy** з командного рядка

Завдання

1. Реалізувати наведений приклад.

2. Згідно завдання викладача створити ANFIS.

2. Створення і дослідження нечітких нейронних мереж із командного рядка

Функція, яка створює і/або навчає гібридні нейронні мережі з архітектурою **ANFIS**, записується **anfis**.

Значення можливих аргументів функції:

– **trnData** – матриця даних для навчання мережі (навчальна вибірка); останній стовпець відповідає вихідній змінній, інші стовпці – вхідним змінним;

– **ім'я** – ідентифікатор створюваної гібридної мережі; якщо структура системи нечіткого висновку з таким ідентифікатором уже створена, то вона буде використана для настроювання числових параметрів, у противному випадку структура буде створена під час виконання функції;

– **trnOpt** – вектор опцій навчання з елементами: **trnOpt(l)**: кількість циклів навчання (за замовчуванням 10), **trnOpt(2)**: цільовий рівень помилки навчання (за замовчуванням 0), **trnOpt(3)**: початковий крок алгоритму навчання (за замовчуванням 0,01), **trnOpt(4)**: коефіцієнт зменшення кроку (за замовчуванням 0,9), **trnOpt(5)**: коефіцієнт збільшення кроку (за замовчуванням 1,1);

– **dispOpt** – вектор опцій вигляду виведеної інформації (за замовчуванням всі елементи – одиничні з елементами: **dispOpt(l)**: ANFIS-інформація, така, як число входів, функції належності тощо, **dispOpt(2)**: помилка, **dispOpt(S)**: крок відновлення (коректування) по кожному параметрові, **dispOpt(4)**: кінцеві результати.

Процес навчання мережі закінчується, коли виконано задане число циклів навчання або помилка навчання зменшилася до заданого рівня. При відсутності деяких аргументів їхні значення приймаються за замовчуванням.

Вихідні параметри функції:

– **помилка_об** – масив (вектор) значень помилок для навчальної вибірки;

– **помилка п** – масив (вектор) значень помилок для перевірочної вибірки;

– **крок** – масив значень кроку в алгоритмі навчання;

– **ім'я!** – ідентифікатор (ім'я) мережі, створеної виходячи з мінімальної помилки навчання;

– **ім'я2** – ідентифікатор (ім'я) мережі, створеної виходячи з мінімальної помилки для перевірочної вибірки.

Приклад

```
>> x=(0:0.1:10)';
>> y=sin(2*x)./exp(x/5);
>> trnData=[x y];
>> a=anfis(trnData);
ANFIS info:
    Number of nodes: 12
    Number of linear parameters: 4
    Number of nonlinear parameters: 6
    Total number of parameters: 10
    Number of training data pairs: 101
```

```
    Number of checking data pairs: 0
    Number of fuzzy rules: 2
Start training ANFIS ...
   1        0.313942
   2        0.31388
   3        0.313817
   4        0.313753
   5        0.313689
Step size increases to 0.011000 after epoch 5.
   6        0.313624
   7        0.313552
   8        0.313479
   9        0.313406
Step size increases to 0.012100 after epoch 9.
  10        0.313332
Designated epoch number reached --> ANFIS training completed at epoch 10.
>> plot(x,y,x,evalfis(x,a),'-.');
>> legend('Real','ANFIS')
```
(рис. 2.1)

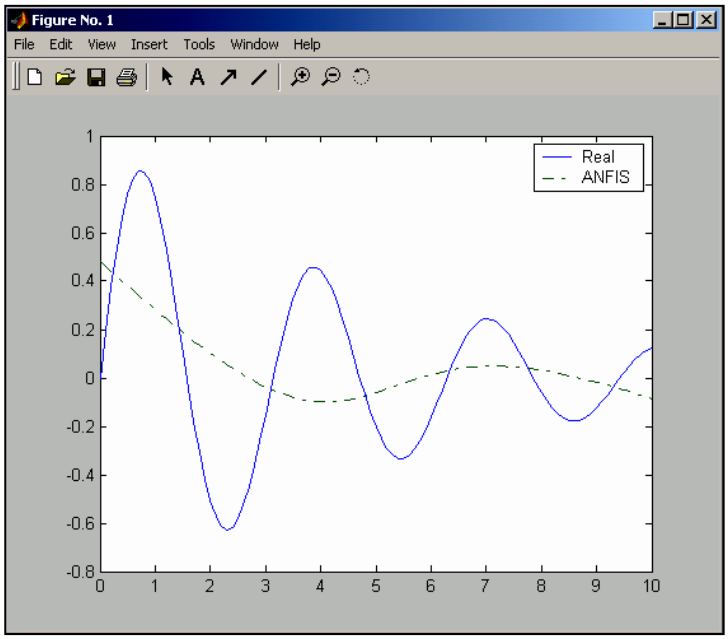

Рис. 2.1. Навчальна вибірка і вихід системи

Приклад

```
>> plot(x,y,x,evalfis(x,a),'-.');
>> legend('ANFIS')
>> legend('ANFIS','Real')
>> legend('Real','ANFIS')
>> x=(0:0.1:10)';
>> y=sin(2*x)./exp(x/5);
>> trnData=[x y];
>> numMFs=5;
>>
>> mfType='gbellmf';
>> epoch_n=20;
>> in_fismat=genfis1(trnData, numMFs, mfType);
>> out_fismat=anfis(trnData, in_fismat, 20);
```

Start training ANFIS ...

1	*0.0694086*
2	*0.0680259*
3	*0.066663*
4	*0.0653198*
5	*0.0639961*

Step size increases to 0.011000 after epoch 5.

6	*0.0626917*
7	*0.0612787*
8	*0.0598881*
9	*0.0585193*

Step size increases to 0.012100 after epoch 9.

10	*0.0571712*
11	*0.0557113*
12	*0.0542741*
13	*0.052858*

Step size increases to 0.013310 after epoch 13.

14	*0.0514612*
15	*0.0499449*
16	*0.0484475*
17	*0.0469667*

Step size increases to 0.014641 after epoch 17.

18	*0.0455003*
19	*0.0439022*
20	*0.0423184*

Designated epoch number reached --> ANFIS training completed at epoch 20.

*>> **plot(x,y,x,evalfis(x,a),'-.');***
*>> **legend('Real','ANFIS')*** (рис. 2.2)

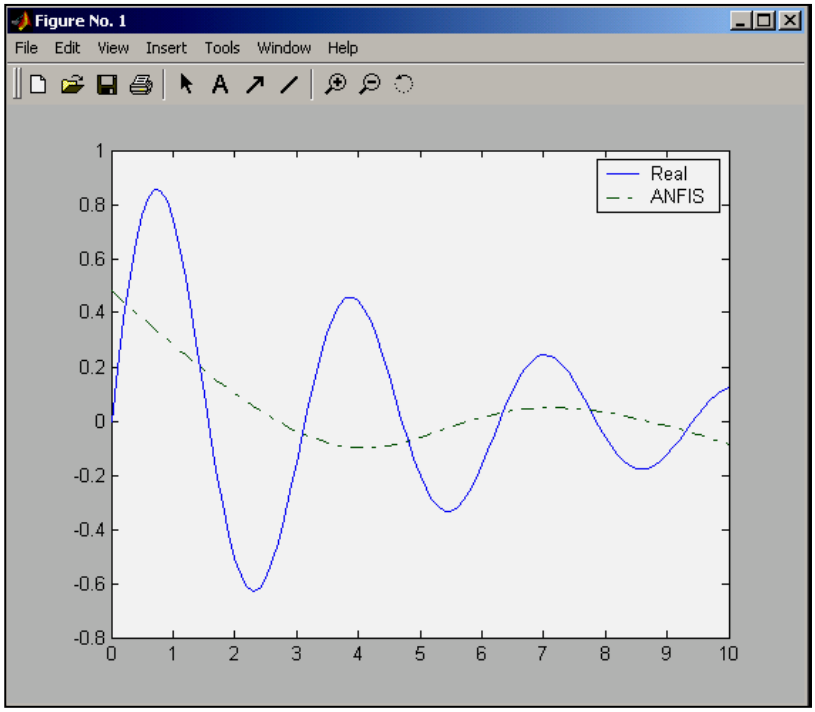

Рис. 2.2. Навчальна вибірка і вихід системи **flsinat**

У другому прикладі при проектуванні гібридної системи заздалегідь були задані деякі значення (кількість і тип функцій належності, кількість циклів навчання), у першому прикладі для тієї ж навчальної вибірки всі налаштування при проектуванні системи обрані за замовчуванням. Відмінність функціонування двох систем наочно ілюструються на рисунках 2.1 та 2.2.

Функція **genfisl** генерує FIS-структуру за експериментальним даними без проведення їх кластеризації:

ім'я = genfisl (data)

ім'я = genfisl (data,numMFs,inmftype, outmftype)

Опис. Функцією генерується структура системи нечіткого висновку типу **Sugeno**.

Аргументи функції:

Data – матриця даних (навчальна вибірка), останній стовпець якої відповідає вихідній змінній, а решта – вхідним змінним; число рядків дорівнює числу наборів експериментальних даних (зразків).

NumMFs – вектор, елементи якого визначають число функцій належності, що задаються для кожного входу; якщо для усіх входів задається одне і те число таких функцій, такий аргумент задається як скаляр (один елемент).

Inmftype – рядковий масив, елементи якого – типи функцій належності, які задаються для вхідних змінних.

Outmftype – рядкова змінна (можливі тільки значення **'linear'** або **'constant')**.

Число функцій належності вихідної змінної дорівнює числу нечітких правил, генерованих **genfisl**, і залежить від елементів вектора **numMFs**. За замовчуванням **numMFs = 2**, **inmftype = = 'gbellmf**. Значення за замовчуванням встановлюються, якщо функція застосовується без трьох останніх аргументів.

Застосування функції обмежене розмірністю задачі: так, при N вхідних змінних з мінімальною розбивкою області визначення кожної змінної на дві підобласті буде генеруватися 2^N правил, що при числі входів більше 5-6 робить аналіз цих правил практично неможливим. Більш раціональним в цьому плані є функція **genfis2**.

Приклад

```
data = [rand(10,1) 10*rand(10,1)-5 rand(10,1)];
fis = genfis1(data,[3 7],char('pimf','trimf'));
[x,mf] = plotmf(fis,'input',1);

subplot(2,1,1), plot(x,mf);
xlabel('input 1 (pimf)');
[x,mf] = plotmf(fis,'input',2);

subplot(2,1,2), plot(x,mf);
xlabel('input 2 (trimf)');
```

(рис. 2.3)

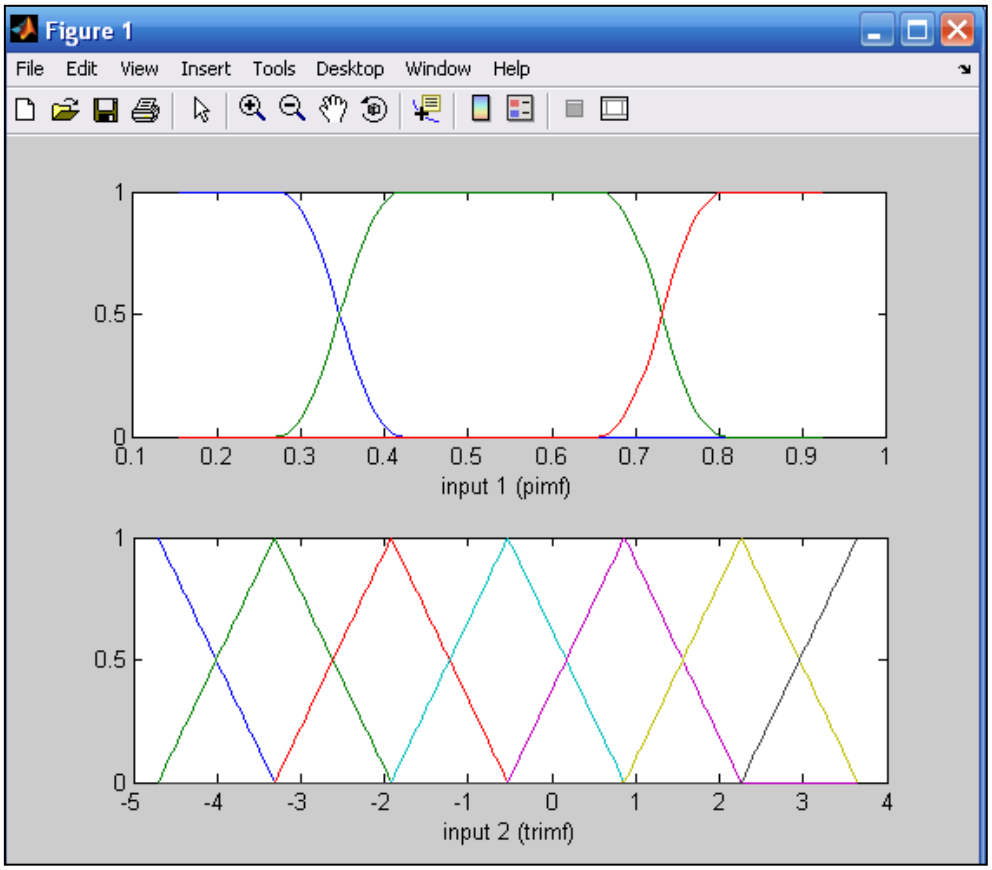

Рис. 2.3. Функції належності, визначені **genflsl**

Функція **genfis2** генерує структуру системи нечіткого висновку (типу **Sugeno**):

ім'я = genfis2(Xin,Xout, radii)

ім'я = genfis2(Xin,Xout,radii,xBounds)

ім'я = genfis2(Xin,Xout,radii,xBounds,options)

Аргументами розглянутої функції є:

Xin – матриця (масив) вхідних даних навчальної вибірки, стовпці якої асоційовані з вхідними змінними, а кожен рядок – з окремим дослідом.

Xout – матриця вихідних змінних навчальної вибірки, стовпці якої представляють значення даних змінних, а число рядків дорівнює числу рядків матриці Xin.

Radii (радіуси) – вектор, що визначає «області впливу» центрів кластерів за кожною вхідною змінною з ненегативними елементами, меншими одиниці; нехай, наприклад, число входів – 3, тоді **radii** = [0,5 0,4 0,3] означає, що за першою змінною «область впливу» складає 0,5 від діапазону її зміни, а за другою і третьою – відповідно 0,4 і 0,3 (ці діапазони визначаються за елементах стовпців матриці Xin). Якщо розглянутий аргумент заданий як скаляр, то за всіма змінними встановлюється той самий розмір «області впливу» кластерів. Параметр має важливе значення для проведення розбивки області визначення входів на підобласті з наступним встановленням числа нечітких правил.

XBounds – матриця з 2 рядками і *N* стовпцями *(N* – число стовпців у матриці Xin, тобто число вхідних змінних), що встановлює границі областей визначення вхідних змінних. Дані цієї матриці використовуються для перетворення (масштабування) входів до діапазону [-1, 1]. Наприклад, **Xbounds = [**-10 0 -1; 10 50 1] задає діапазон [-10, 10] для першої змінної, [0, 50] – для другої і [-1, 1] – для третьої. Якщо розглянутий аргумент опущено, границі областей визначаються як максимальні і мінімальні елементи за стовпцями матриці Xin.

Options – вектор опцій, що встановлює параметри алгоритму кластеризації.

Функція **genfis2**, так само як **genfisl**, може застосовуватися в комплекті з функцією **anfis** (для настроювання параметрів системи нечіткого висновку), але, на відміну від останньої, **genfis2** повертає вже цілком визначену, готову до використання систему, що не вимагає додаткової оптимізації.

Функція **genfis2** робить раціональнішу, ніж **genfisl**, розбивку простору входів на підобласті.

Завдання

1. Реалізувати наведені приклади.
2. Згідно завдання викладача створити ANFIS.

3. Кластеризація з командного рядка

Функція **fcm** здійснює кластеризацію з використанням алгоритму **Fuzzy c-means**. Вона записується у вигляді:

```
[center,U,obj_fcn] = fcm(data,cluster_n)
[center,U,obj_fcn] = fcm(data,cluster_n,options)
```

Опис. Аргументи функції:

Data – матриця даних, кожен стовпець якої відповідає одній змінній, а кожен рядок – одному із дослідів.

Cluster_n – число кластерів, які задаються, (повинно бути більше 1).

Options – вектор опцій функції з елементами:

options(l): показник для матриці U (за замовчуванням 2,0),

options(2): максимальне число ітерацій при визначенні центрів кластерів (за замовчуванням 100),

options(S): мінімальна сума (за замовчуванням 1е-5),

options(4): виведення інформації в процесі ітерацій (за замовчуванням 1).

Вихідні параметри функції:

Center – матриця, елементи якої (по стовпцях) є координатами знайдених центрів кластерів, число рядків дорівнює числу центрів;

U – матриця значень належності даних до виявлених кластерів;

Obj_fcn – значення цільової функції в процесі ітерацій.

Приклад

```
>> load clusterdemo.dat;
>> data=clusterdemo;
>> [center, U, obj_fcn]=fcm(data, 3);

Iteration count = 1, obj. fcn = 56.554232
Iteration count = 2, obj. fcn = 42.996494
Iteration count = 3, obj. fcn = 42.931708
Iteration count = 4, obj. fcn = 42.639644
Iteration count = 5, obj. fcn = 41.191412
Iteration count = 6, obj. fcn = 34.682033
Iteration count = 7, obj. fcn = 21.215631
Iteration count = 8, obj. fcn = 15.681175
Iteration count = 9, obj. fcn = 15.434686
Iteration count = 10, obj. fcn = 15.430601
Iteration count = 11, obj. fcn = 15.430530
Iteration count = 12, obj. fcn = 15.430528

>> center

center =

   0.2051    0.5960    0.8090
   0.5963    0.1923    0.5057
   0.7939    0.7892    0.1968
```

Інша функція **subclust** повертає центри кластерів.

Запис: [C,S] **= subclust(X,radii,xBounds,options)**

Функцією визначаються центри кластерів набору експериментальних даних, представлених матрицею X (стовпці матриці відповідають змінним, рядки – експериментальним «точкам» або дослідам) на основі реалізації алгоритму кластеризації (**Subtractive clustering**). У його основі лежить гіпотеза, що кожна експериментальна точка може бути центром кластера, при цьому спочатку для кожної точки обчислюється ступінь правдоподібності даного припущення («потенціал точки»). Подальші обчислення відбуваються ітеративно:

- точка з найбільшим потенціалом являється центром першого кластера;
- з відзначеної околиці цієї точки віддаляються решта точок;
- із точок, які залишилися, вибирається центр наступного кластера, поки не будуть розглянуті (виключені або оголошені центрами кластерів) всі точки.

Розміри околиць задаються елементами вектора **radii,** що (як і аргумент **xBounds)** аналогічний розглянутому при описі функції **Genfis2**.

Функція повертає центри кластерів у виді матриці; кожен рядок цієї матриці містить координати одного зі знайдених центрів.

Вектор S містить елементи, що визначають діапазони впливу центрів кластерів за кожною змінною (для всіх центрів ці діапазони однакові).

Завдання

1. Реалізувати наведений приклад.
2. Згідно завдання викладача встановити центри кластерів.

4. Графічний інтерфейс програми кластеризації Clustering

У пакет **Fuzzy Logic Toolbox** входить ще одна програма, яка дозволяє роботу в режимі графічного інтерфейсу – програма **Clustering** (кластеризація) виявлення центрів кластерів, тобто точок у багатомірному просторі даних, біля яких групуються експериментальні дані. Виявлення подібних центрів є важливим етапом при попередній обробці даних, оскільки дозволяє співставити з цими центрами функції належності змінних при наступному проектуванні системи нечіткого висновку.

Запуск програми **Clustering** здійснюється командою (функцією) **findcluster**. У вікні програми, що з'являється, мається (вгорі) головне меню, яке містить досить стандартний набір пунктів (**File**, **Edit**, **Window**, **Help**) і набір керуючих кнопок і опцій (праворуч). До цих кнопок відносяться:

– кнопка завантаження файла даних **Load Data**,

– кнопка вибору алгоритму кластеризації **Method**,

– чотири розташовані нижче кнопки опцій алгоритму (їхні назви змінюються в залежності від обраного алгоритму),

– кнопка початку ітеративного процесу знаходження центрів кластерів (кластеризації) **– Start**,

– кнопка збереження результатів кластеризації **SaveCenter**,

– кнопка очищення (стирання) графіків **Clear Plot**,

– кнопка довідкової інформації **Info**,

– кнопка завершення роботи з програмою **Close**.

У програмі використовуються два алгоритми виявлення центрів кластерів: **Fuzzy c-means** (який можна перевести як «Алгоритм нечітких центрів») і **Subtractive clustering**.

Якщо не вдаватися в їх детальний теоретичний виклад, а обмежитися виявленням розходжень на рівні користувача, то можна відзначити, що алгоритм **Fuzzy c-means** точніший (якщо поняття точності взагалі можна використовувати у такому випадку), але для своєї роботи вимагає задання таких опцій, як число кластерів (кнопка **Cluster num**) і число ітерацій (кнопка **Max Iteration**). Якщо число ітерацій ще можна задати апріорно, то помилка в заданні числа кластерів може призвести до негативних наслідків.

Алгоритм **Subtractive clustering** менш точний, але і менш вимогливий до апріорної інформації; при роботі з ним можна зберегти опції, задані в програмі за замовчуванням.

На рисунку 4.1 приведено приклад використання програми для файла даних **clusterdemo.dat** з директорії **Matlab/toolbox/fuzzy/fuzdemos/** при використанні алгоритму **Subtractive clustering**. Відмітимо, що виводиться тільки двовимірне поле розсіювання, але змінюючи змінні у відповідних полях (**X-axis** і **Y-axis**), можна «переглянути» весь багатомірний простір змінних.

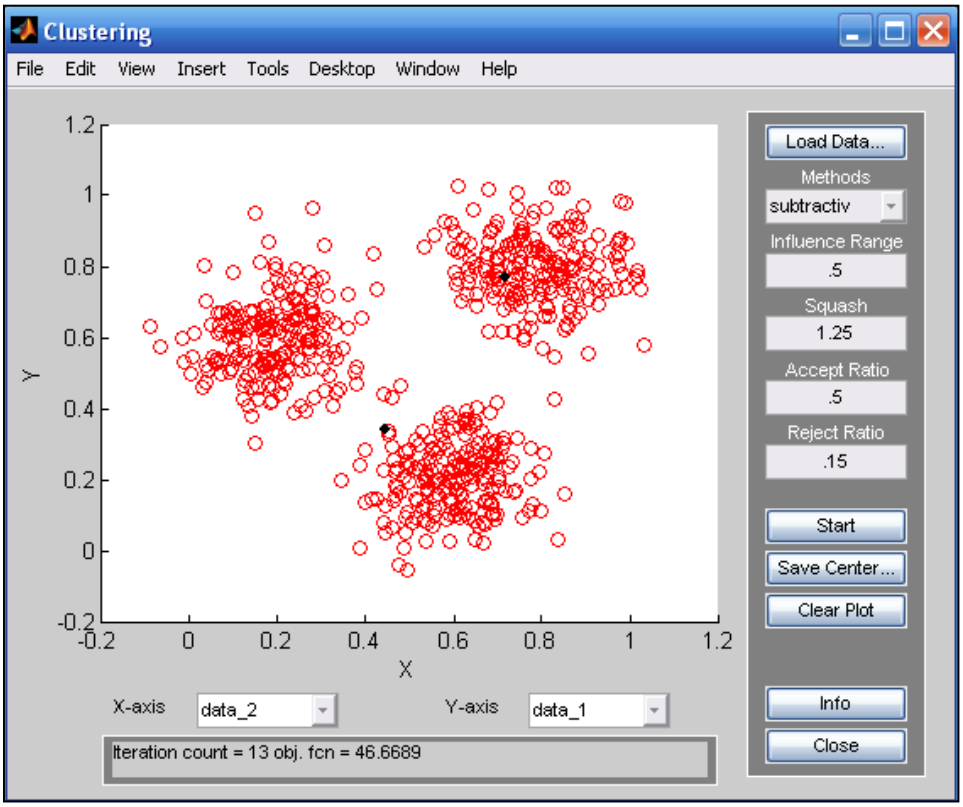

Рис. 4.1. Приклад реалізації програми **Clustering**

Завдання

1. Реалізувати наведений приклад.
2. Згідно завдання викладача встановити центри кластерів за допомогою програми **Clustering**.