Лабораторная работа №3
Решение задачи прогнозирования с применением аппарата нечёткой логики

Цель работы: изучить принципы построения нейронных сетей, научиться строить нейронные сети и гибридные нейронные сети для решения задачи прогнозирования в пакете MATLAB.

Продолжительность работы: 4 часа.

Теоретические сведения

Сочетание нечёткой логики и машинного обучения

Нейронные сети и нечеткая логика являются универсальными аппроксиматорами сложных (нелинейных) функциональных зависимостей во многих интеллектуальных задачах кибернетик и прогнозирования, диагностики, распознавания образов и др.

Достоинством нечеткой логики является возможность использования экспертных знаний о структуре объекта в виде лингвистических высказываний: если <входы>, то <выход>. Однако аппарат нечеткой логики не содержит механизмов обучения. Поэтому полученные с его помощью результаты сильно зависят от вида функций принадлежности, которыми формализуются нечеткие термы. Кроме того, эксперту необходимо определить все правила. Такой алгоритм во многом статичен. Необходимость его изменения повлечет за собой довольно трудоемкую экспертную процедуру.

Главной особенностью нейронных сетей является их способность к обучению. Для обучения нейронной сети не требуется никакой априорной информации о структуре искомой функциональной зависимости. Нужна лишь обучающая выборка в виде пар <входы–выход>.

Вместо определения всех функций принадлежности лингвистических переменных и правил необходимо создать обучающую выборку достаточного объема. Одним из критериев "достаточности" служит следующий: число обучающих пар <входы–выход> должно превышать число настраиваемых параметров сети. Преимущество данного метода заключается в возможности самонастройки параметров сети в процессе работы.

Объединение нечеткой логики с нейронными сетями дает принципиально новое качество. Получаемая в результате такого объединения нейронечеткая сеть обладает двумя важнейшими человеческими (интеллектуальными) свойствами:

−лингвистичностью, т.е. использованием знаний на естественном языке;

−обучаемостью в реальном масштабе времени.

В соответствии с анализом, проведенным в ряде работ, для класса задач, к которым применимы нейронечеткие сети, нейронные сети при обучении требуют значительно больше итераций. Поскольку любая итерация занимает определенное машинное время, то и общее время обучения НС больше.

Существует две основных разновидности нейронечетких сетей: Такаги-Сугено-Канга и Ванга-Менделя.

Нейронечёткая сеть Такаги-Сугено-Канга (*TSK*)

Обобщенную схему вывода модели *TSK* при использовании *M* правил и *N* переменных *xj* можно представить в виде:

$$\begin{gathered} IF\left(x_1\ IS\ A_1^{(1)}\right)\ AND\left(x_2\ IS\ A_2^{(1)}\right)\ AND\ \ldots\ AND\left(x_n\ IS\ A_n^{(1)}\right), \\ THEN\ y_1 = p_{10} + \sum_{j=1}^{N} p_{1j}x_j \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ IF\left(x_1\ IS\ A_1^{(M)}\right)\ AND\left(x_2\ IS\ A_2^{(M)}\right)\ AND\ \ldots\ AND\left(x_n\ IS\ A_n^{(M)}\right), \\ THEN\ y_M = p_{M0} + \sum_{j=1}^{N} p_{Mj}x_j \end{gathered} \tag{1}$$

Условие $IF\ (x_i\ IS\ A_i)$ реализуется функцией фазификации, которая представляется обобщенной функцией Гаусса отдельно для каждой переменной x_i:

$$\mu_A(x_i) = \frac{1}{1 + \left(\frac{x_i - c_i}{\sigma_i}\right)^{2b_i}}, \tag{2}$$

где $\mu_A(x_i)$ представляет оператор A_i.

При *M* правилах вывода агрегирование выходного результата сети производится по формуле

$$y(x) = \frac{1}{\sum_{k=1}^{N} w_k}\left(\sum_{k=1}^{M} w_k y_k\right), \tag{3}$$

где$y_k(x) = p_{k0} + \sum_{j=1}^{N} p_{kj}x_j$.

Присутствующие в этом выражении веса w_k интерпретируются как значимость компонентов $\mu_A^{(k)}(x)$. При этом условии формуле выше можно сопоставить многослойную структуру сети (рис.1).

В такой сети выделяется пять слоев:

- первый слой выполняет раздельную фазификацию каждой переменной x_i (i =1,2, …, *N*),определяя для каждого *k*-го правила вывода значение коэффициента принадлежности $\mu_A^{(k)}(x_i)$ в соответствии с применяемой функцией фазификации.

- второй слой (непараметрический) выполняет агрегирование отдельных переменных x_i, определяя результирующее значение коэффициента принадлежности $w_k = \mu_A^{(k)}(x)$ для вектора *x*;

- третий слой представляет собой генератор функции *TSK*, рассчитывающий значения$y_k(x) = p_{k0} + \sum_{j=1}^{N} p_{kj}x_j$. В этом слое также производится умножение сигналов $y_k(x)$ на значения w_k, сформированные на предыдущем слое. Это параметрический слой, в котором адаптации подлежат линейные веса p_{kj} для k = 1, 2, …, *M* и j=1, 2, …, *N*, определяющие функцию следствия модели *TSK*;

- четвертый слой (непараметрический) составляют два нейрона-сумматора, один из которых рассчитывает взвешенную сумму сигналов y_k, а второй определяет сумму весов $\sum_{k=1}^{M} w_k$.

- пятый слой (нормализующий), состоит из единственного выходного нейрона, в котором веса подвергаются нормализации. Выходной сигнал $y(x)$ определяется выражением

$$y(x) = f(x) = \frac{f_1}{f_2}. \quad (4)$$

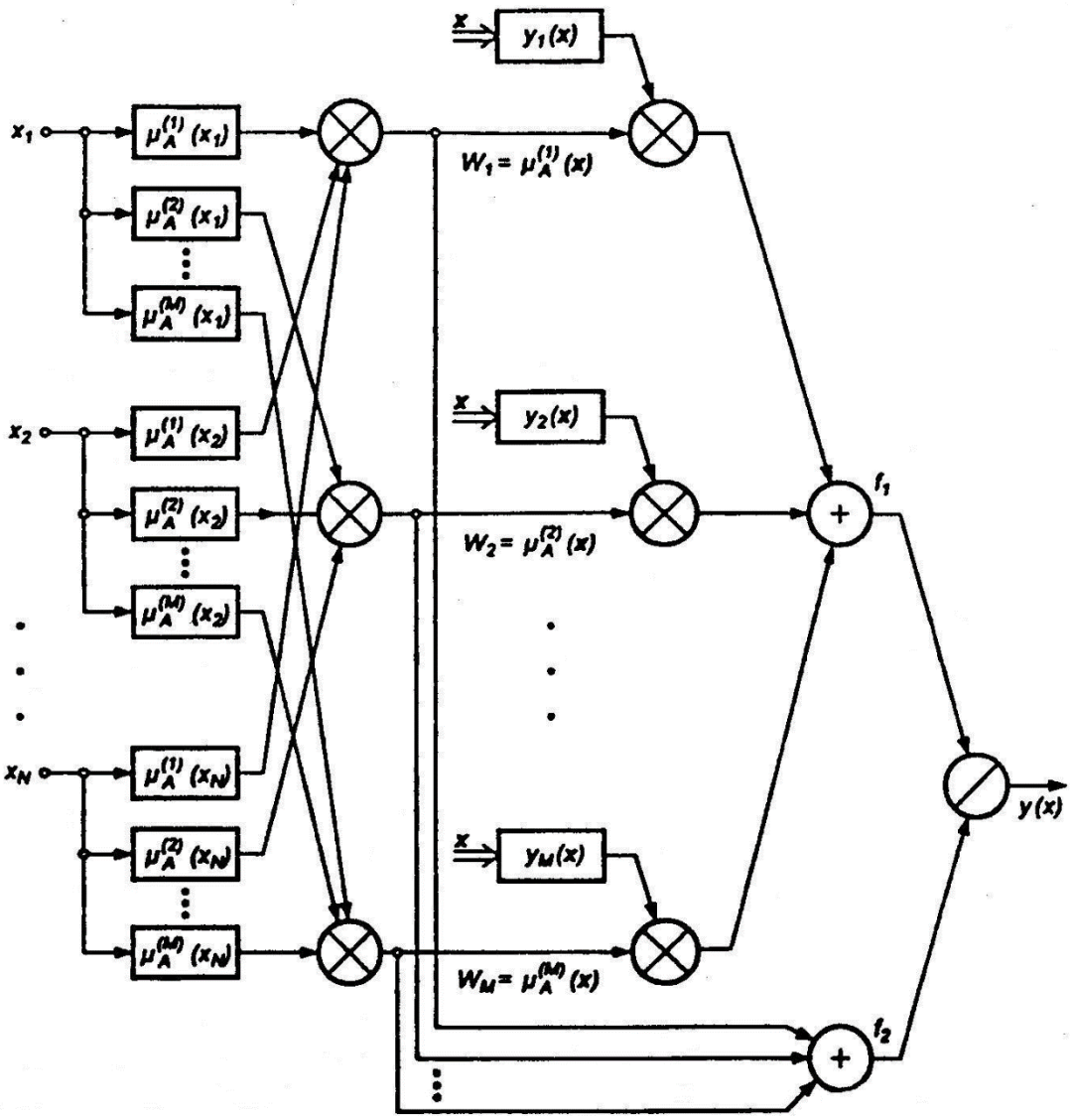

Рис. 1. Структура нечёткой нейронной сети *TSK*

Нейронечёткая сеть Ванга-Менделя

Структура сети Ванга-Менделя приведена на рис. 2.

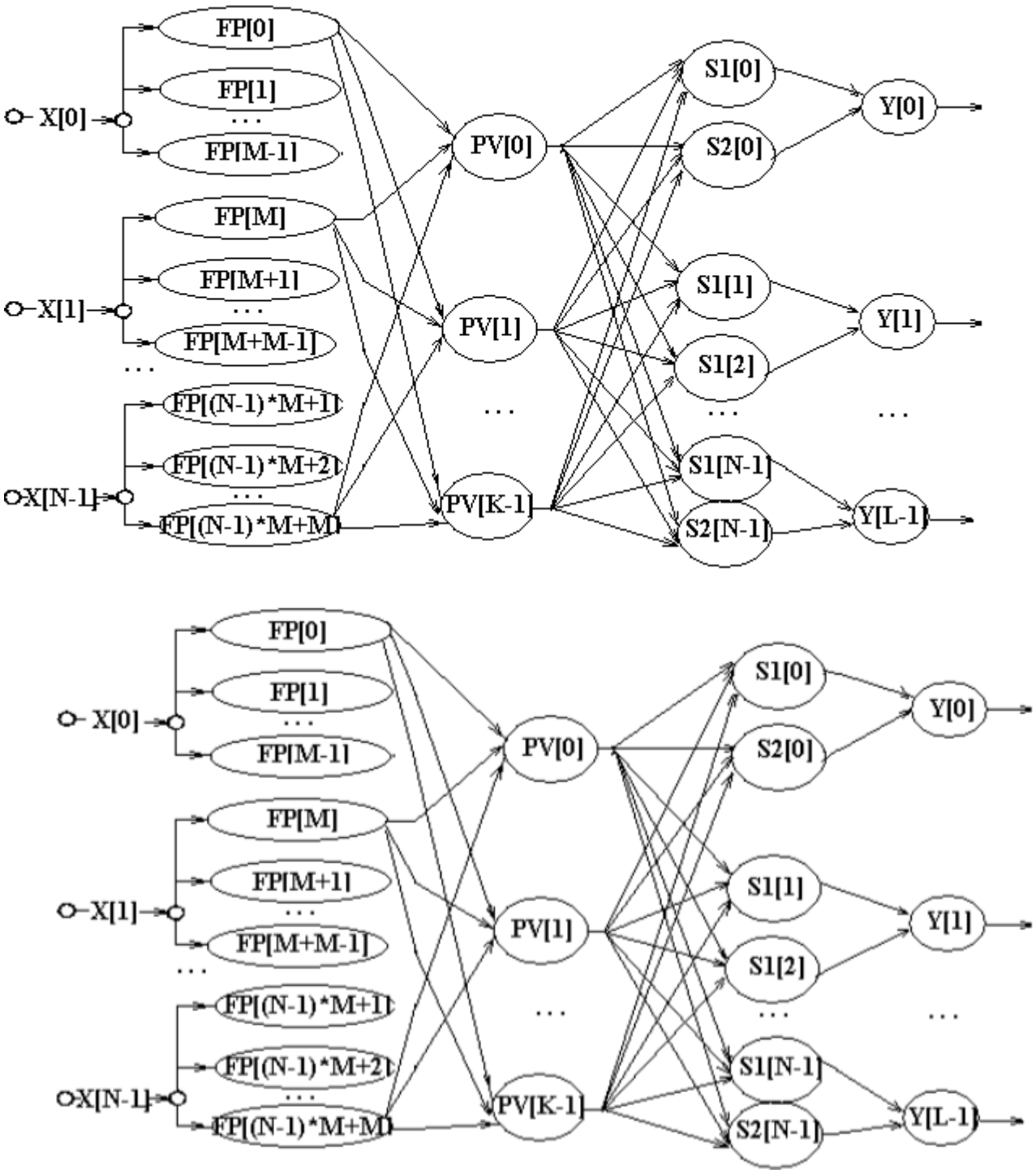

Рис. 2. Структура нечёткой нейронной сети Ванга-Менделя

Слой I состоит из нейронов-фазификаторов и выполняет раздельную фазификацию каждой переменной по функциям принадлежности (FP). Выходы данного слоя рассчитываются с помощью обобщенной функции Гаусса:

$$FP[iM+j] = exp\left[\left(-\frac{(x - C[iM+j])^2}{2D[iM+j]^2}\right)^B\right], \quad (5)$$

представленной в рациональной форме:

$$FP[iM+j] = \frac{1}{1+\left(\frac{X[j]-C[iM+j]}{D[iM+j]}\right)^{2B[iM+j]}}, \quad (6)$$

где $i = \overline{0, N-1}$, $j = \overline{0, M-1}$; M – количество функций принадлежности для каждой переменной; N – число входных переменных; $C[iM+j]$ (смещение); $D[iM+j]$(масштаб);$B[iM+j]$(форма) – это нелинейные параметры сети, описывающие функцию Гаусса (подлежат настройке при обучении).

Слой II состоит из нейронов-правил и определяет уровни активации правил вывода (*PV*). Выходы данного слоя рассчитываются по формуле:

$$PV[i] = \prod_{j=0}^{N-1} FP[j], \quad (7)$$

где $i = \overline{0, K-1}$, а $K = M^N$.

Слой III состоит из двух типов нейронов-сумматоров, один из которых (S_1) производит агрегирование правил вывода, а второй (S_2) – агрегирование взвешенных правил вывода. Выходы нейронов рассчитываются по формулам:

$$S_1[j] = \sum_{i=0}^{K-1} W[j,i] * PV[i], \quad S_2[j] = \sum_{i=0}^{K-1} PV[i], \quad (8)$$

где $j = \overline{0, L-1}$, L – количество выходов сети; $W[j,i]$ – линейные параметры сети, определяющие весовые коэффициенты применяемого правила (подлежат настройке при обучении).

Слой IV содержит нейроны-нормализаторы и нормализует выходные переменные сети *Y*. Его выход рассчитывается по формуле:

$$Y[i] = {S_1[i]}/{S_2[i]}, \quad (9)$$

где $i = \overline{0, K-1}$.

Данная сеть содержит два параметрических слоя – первый и третий. В первом слое каждый нейрон хранит три нелинейных параметра сети *C*, *D* и *B*, которые определяют функцию принадлежности Гаусса. В третьем слое хранятся линейные параметры сети *W*, определяющие весовые коэффициенты правил и значения выходных переменных.

Построение нейронных сетей и гибридных нейронных сетей для прогнозирования в пакете MATLAB

Одной из основных возможностей нейронных сетей (НС) является способность к экстраполяции. В процессе обучения на имеющейся статистике (например, о некотором временном ряде) НС способна находить зависимости (в т.ч. скрытые даже для опытного эксперта) между данными и ситуациями, которые в последующем обученной НС позволяют строить прогноз поведения некоторого функционала.

В частности, НС довольно широко применяются для прогнозирования курса валют на

определенный период. Например, для прогнозирования курса на пятый банковский день по имеющимся данным о четырех предшествующих или прогноз на месяц по данным за полгода. Для этих целей возможно применение как обычных НС прямого распространения, так и нечетких нейронных сетей.

Построим **нейронную сеть**, которая по данным о курсе валюты за четыре банковских дня предсказывает курс на пятый день (экстраполирует). Используются данные с 01.01.2012 по 10.03.2012 г. о курсе доллара США. Обучающие данные – с 01.01.2012 по 01.03.2012 г.

Создадим матрицу обучающих данных:

```
>>obych=[ 31.5673 31.6053 31.6117 31.6113 31.6123
        31.6053 31.6117 31.6113 31.6123 31.6268
        31.6117 31.6113 31.6123 31.6268 31.6302
        31.6113 31.6123 31.6268 31.6302 31.6409
        31.8382 31.8423 31.8445 31.8424 31.844];
```

Создадим матрицу входных значений:

```
>>vxod=obych(:,1:4);
```

Создадим матрицу ожидаемых выходных значений:

```
>>vixod=obych(:,5);
```

Создадим нейронную сеть для прогнозирования:

```
>>prognoz=newff(minmax(vxod'),[151],{'logsig' 'purelin'});
```

Обучим нейронную сеть:

```
>>prognoz=train(prognoz,vxod',vixod');
```

В результате обучения нейронной сети получим окно, приведенное на рис. 3.

Создадим тестовую выборку для проверки результатов обучения:

```
>>test =[31.8400 31.8400 31.8424 31.8547 31.8584
        31.8400 31.8424 31.8547 31.8584 31.8596
        31.8424 31.8547 31.8584 31.8596 31.8578
        31.8547 31.8584 31.8596 31.8578 31.8600];
>>test_vxod=test(:,1:4);
>>test_vixod=test(:,5);
```

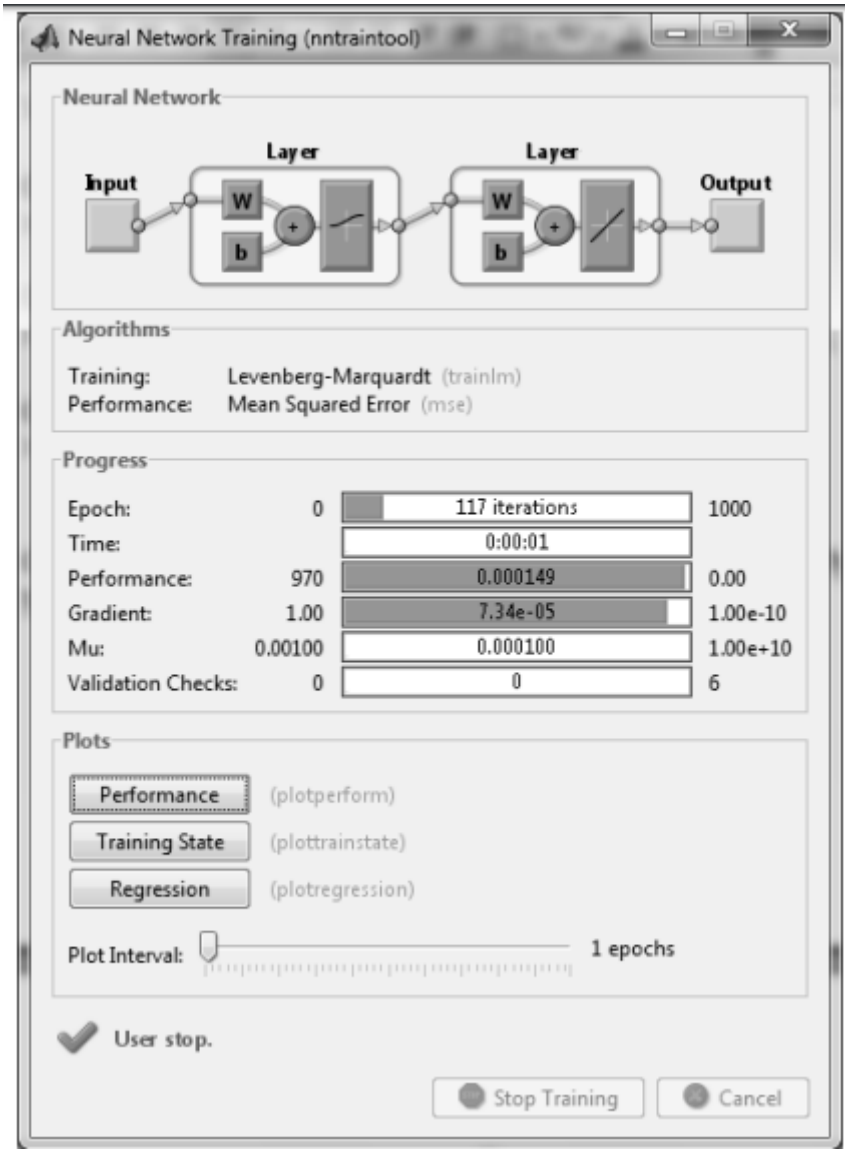

Рис. 3. Результаты обучения НС

Проведем моделирование обученной сети на тестовых данных:

```
>>sim(prognoz,test_vxod)

ans =

31.8694   31.8573   31.9216   31.9450

>>test_vixod

test_vixod =

   31.8584
   31.8596
   31.8578
   31.8600
```

Результаты моделирования и истинные значения курса валют отличаются в достаточной степени.

Построим **гибридную нейронную сеть**, которая по данным о курсе валюты за четыре банковских дня предсказывает курс на пятый день (экстраполирует). Используются данные с 01.01.2012 по 10.03.2012 г. о курсе доллара США. Обучающие данные – с 01.01.2012 по 01.03.2012 г., тестирующие – с 1.01.2012 по 9.01.2012 г., проверочные – на10.01.2012 г.

Создадим файлы данных **.dat: training.dat, testing.dat, cheking.dat.** Эти данные можно также хранить в переменных рабочего пространства (workspace) MATLAB. Импорт в редактор **Anfis Editor** возможен для обоих форматов. В файлах будут содержаться данные, необходимые для обучения и проверки нейронной сети. Данные в них – матрицы, в каждой из которых по 5 столбцов – 4 банковских дня (вход) и 1 день (выход). В первом файле – 40 строк, во втором – 4, в третьем – 1.

Файл **Training.dat**

31.5673 31.6053 31.6117 31.6113 31.6123
31.6053 31.6117 31.6113 31.6123 31.6268
31.6117 31.6113 31.6123 31.6268 31.6302
31.6113 31.6123 31.6268 31.6302 31.6409
31.8382 31.8423 31.8445 31.8424 31.8424

Файл **Testing.dat**

31.8422 31.8445 31.8424 31.8547 31.8584
31.8423 31.8424 31.8547 31.8584 31.8596
31.8424 31.8547 31.8584 31.8596 31.8578
31.8547 31.8584 31.8596 31.8578 31.8634

Файл **Cheking.dat**

31.858431.859631.857831.862331.8597

Для построения нечетких нейронных сетей в MATLAB используется редактор **Anfis Editor** (рис. 4).

Запустим редактор командой **anfisedit**.

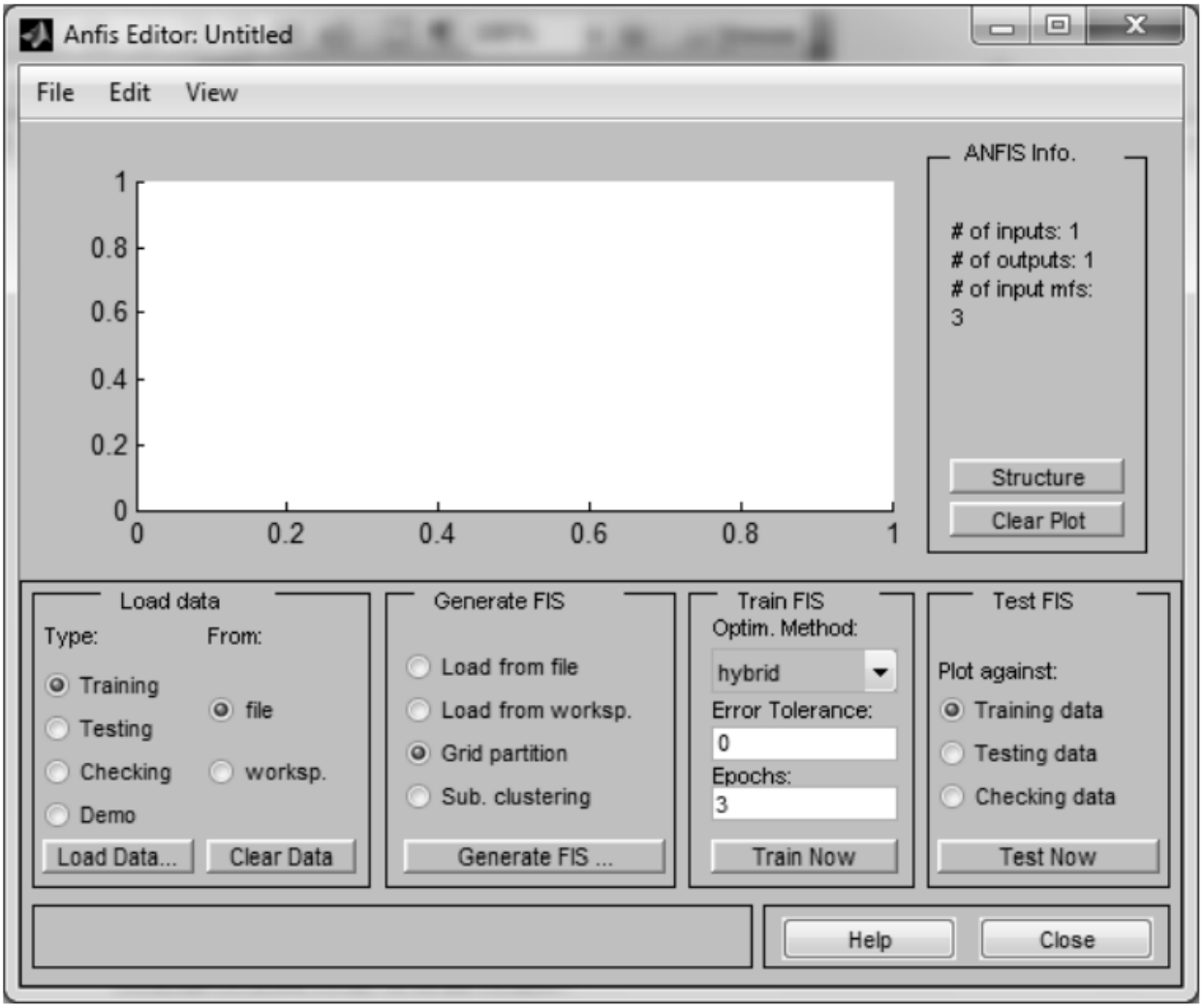

Рис. 4. Главное окно Anfis editor

В меню **Load data** следует выбрать **Training** и **From disk**, нажать кнопку **load data**. В появившемся окне следует выбрать созданный ранее **training.dat**.

В меню **Load data** следует выбрать **Testing** и **Fromdisk**, нажать кнопку **load data**. В появившемся окне следует выбрать созданный ранее **testing.dat**.

В меню **Load data** следует выбрать **Cheking** и **From disk**, нажать кнопку **load data**. В появившемся окне следует выбрать созданный ранее **cheking.dat**.

Данные для обучения и проверки загружены (рис. 5).

Далее, установив переключатель меню **Generate FIS** в положение **Grid partition**, следует нажать кнопку **Generate FIS** (рис. 6).

В данном случае в модели 4 входных переменных, каждой из которых соответствуют по 3 терма типа **gaussmf**. Выходная переменная задается линейной функцией.

Нажав на кнопку **Structure** меню **Anfis info**, можно увидеть структуру сети (рис. 7).

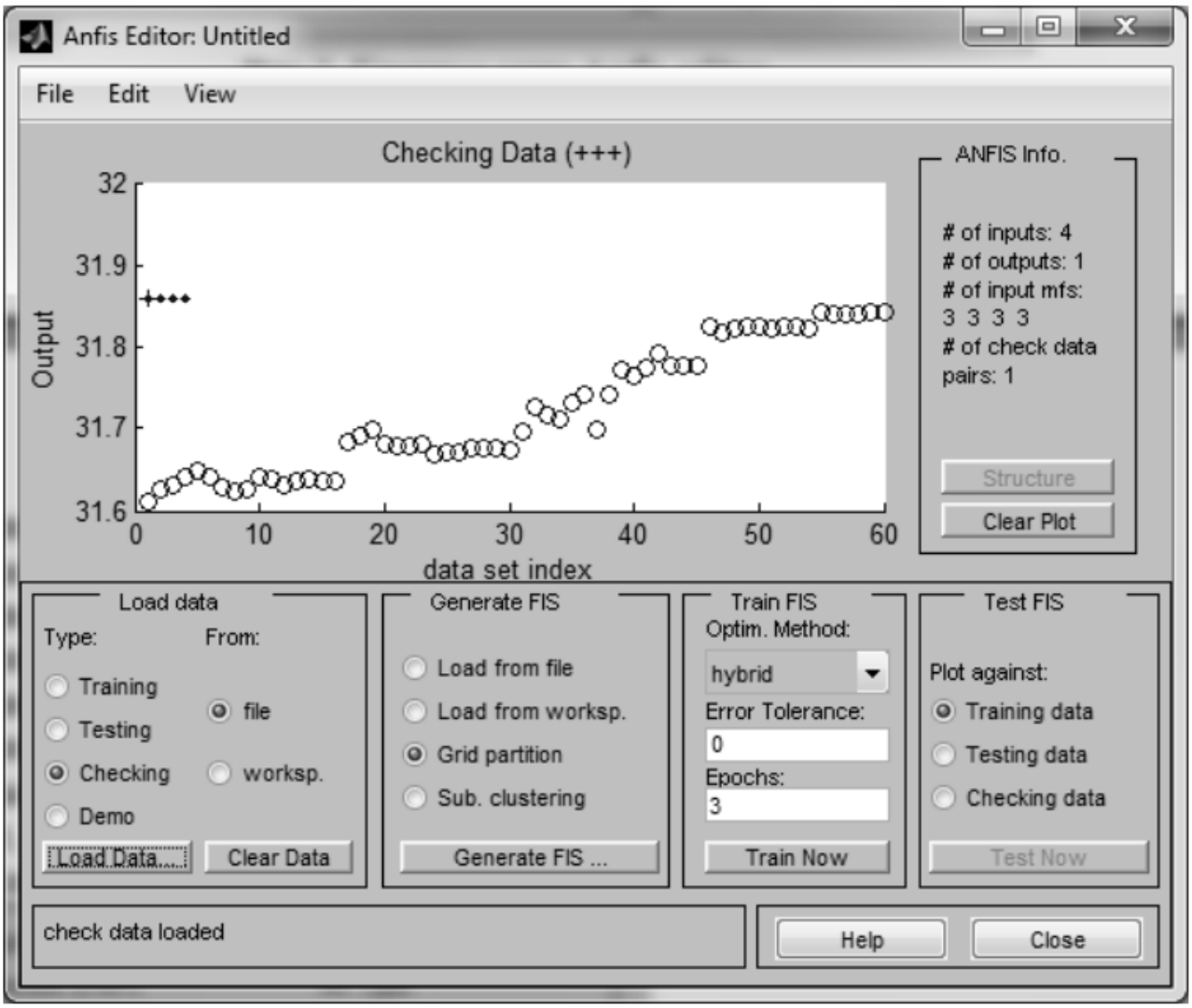

Рис. 5. Вид главного окна редактора после загрузки обучающих данных

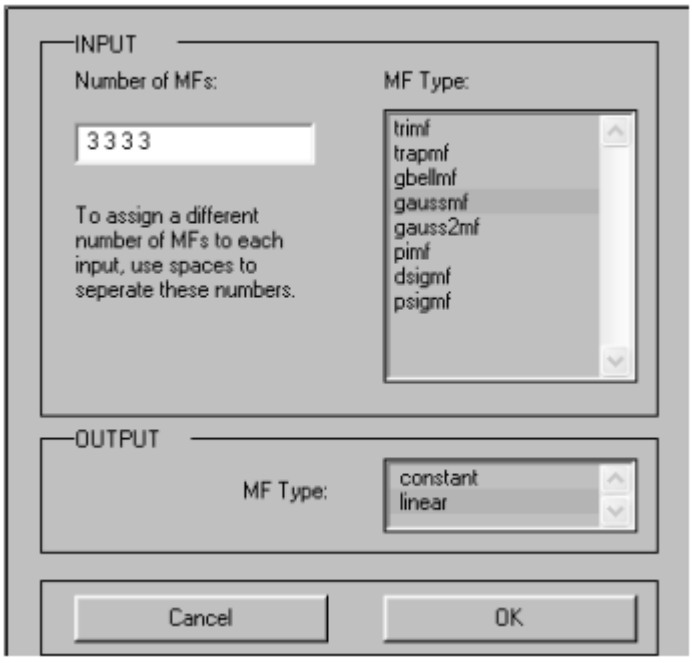

Рис. 6. Параметры генерируемой сети

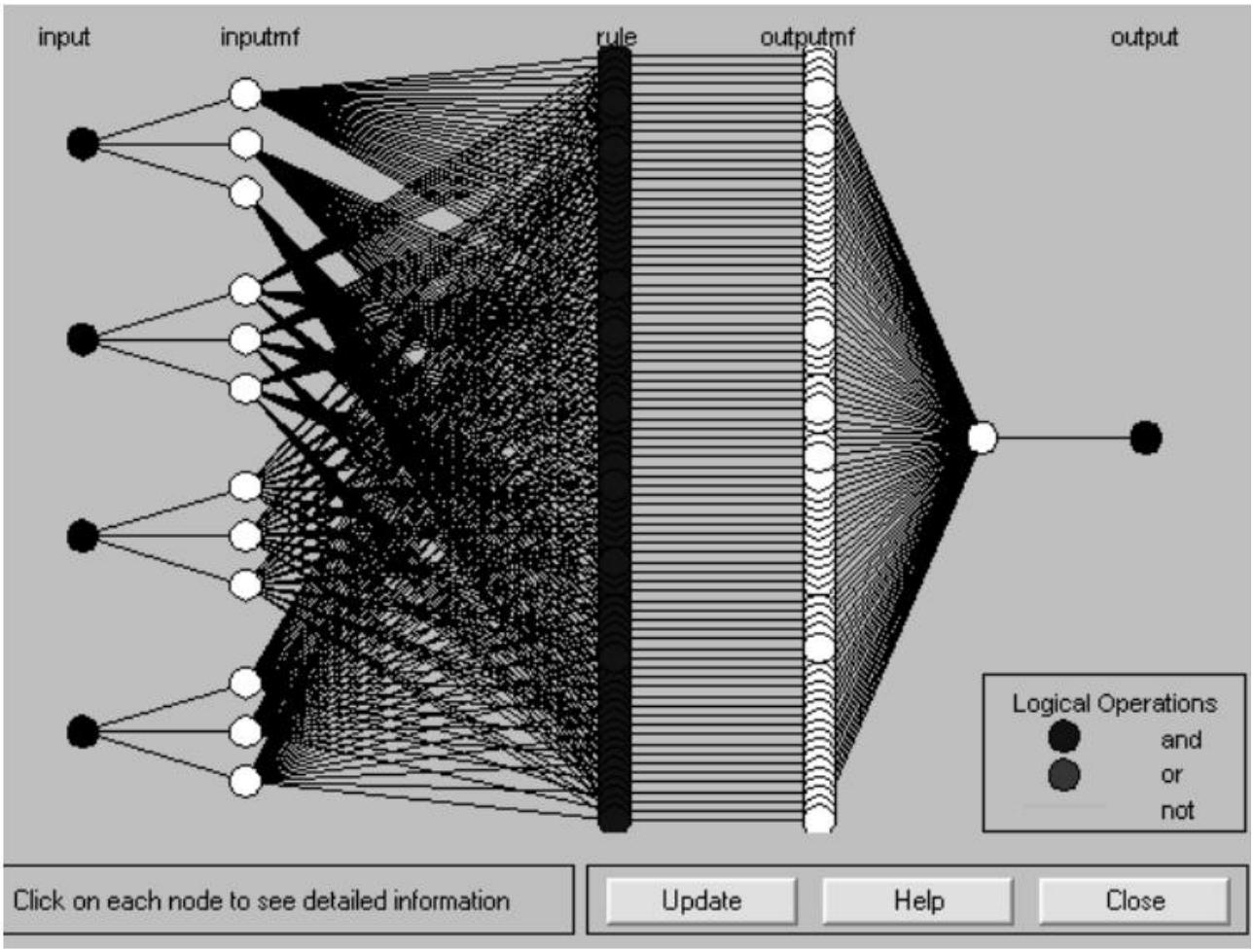

Рис. 7. Структура нечёткой нейронной сети

Далее следует выбрать гибридный метод обучения, требуемую ошибку обучения – 0 и количество циклов обучения – 10, как показано на рис. 8. Затем следует нажать кнопку **Train Now**.

Ошибка обучения установилась на уровне 0.00010501.

Проведем тестирование сети. Для этого в разделе **Test FIS** главного окна следует выбрать соответствующую выборку и нажать кнопку **Nest Now**. В результате для обучающих данных ошибка установилась на уровне 0.00014247 (рис. 9), для тестирующих данных ошибка установилась на уровне 0.020429 (рис. 10), а для проверочных (контрольных) данных ошибка установилась на уровне 0.002094 (рис.11).

С помощью команды **FIS Properties** меню **Edit** возможно просмотреть полученную нечёткую нейронную сеть как систему нечеткого логического вывода.

Далее экспортируем результаты в рабочую область **Export -> To workspace**.

Воспользуемся командой **evalfis** для точного определения значения прогноза:

```
>>out=evalfis([31.858431.8596 31.8578 31.86],anfis)
out=31.8568
```

Действительное же значение прогноза равно 31.8597.

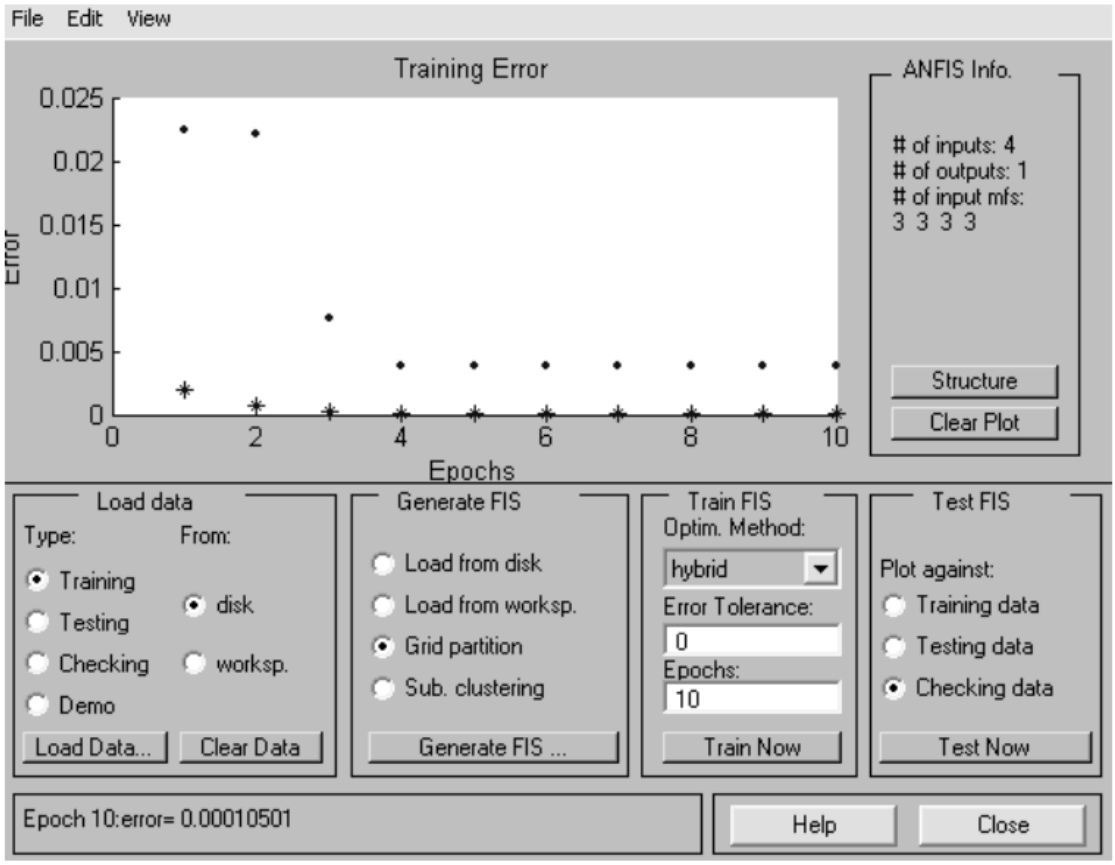

Рис. 8. Результаты обучения ННС

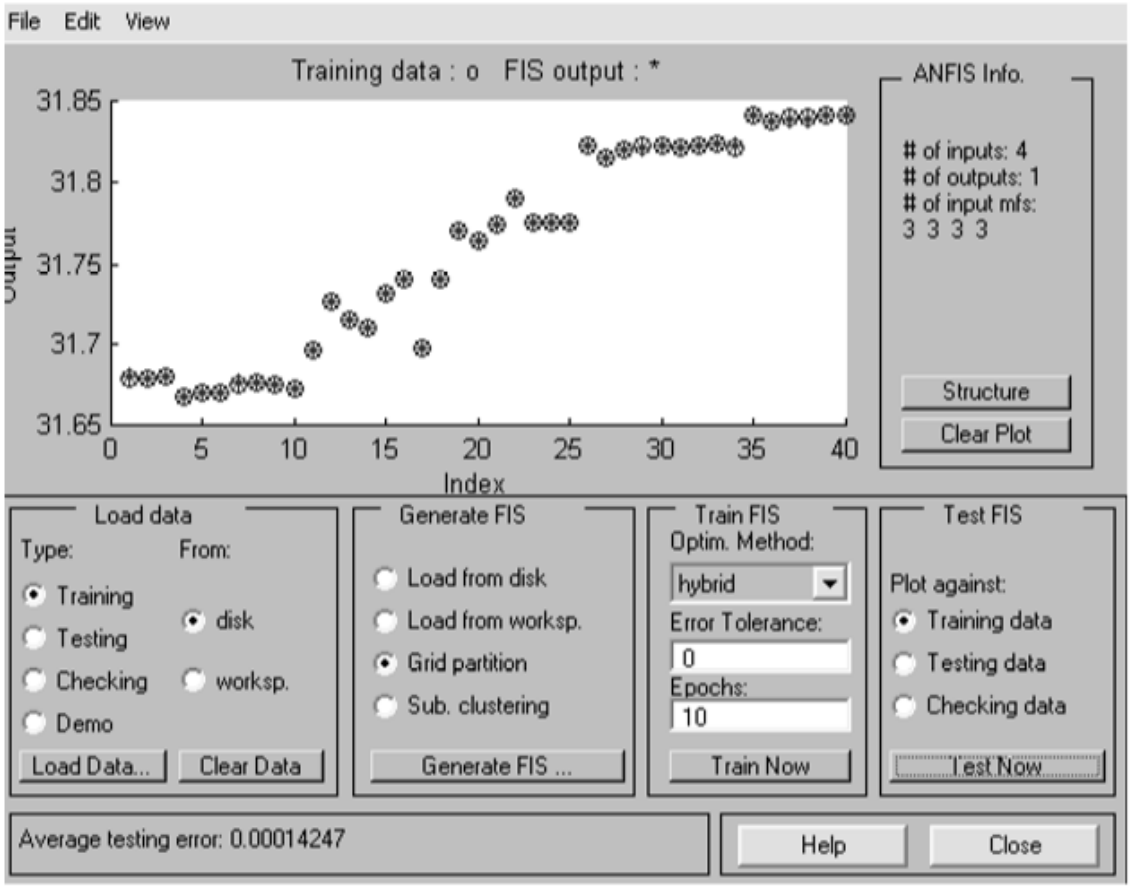

Рис. 9. Результаты моделирования ННС для обучающих данных

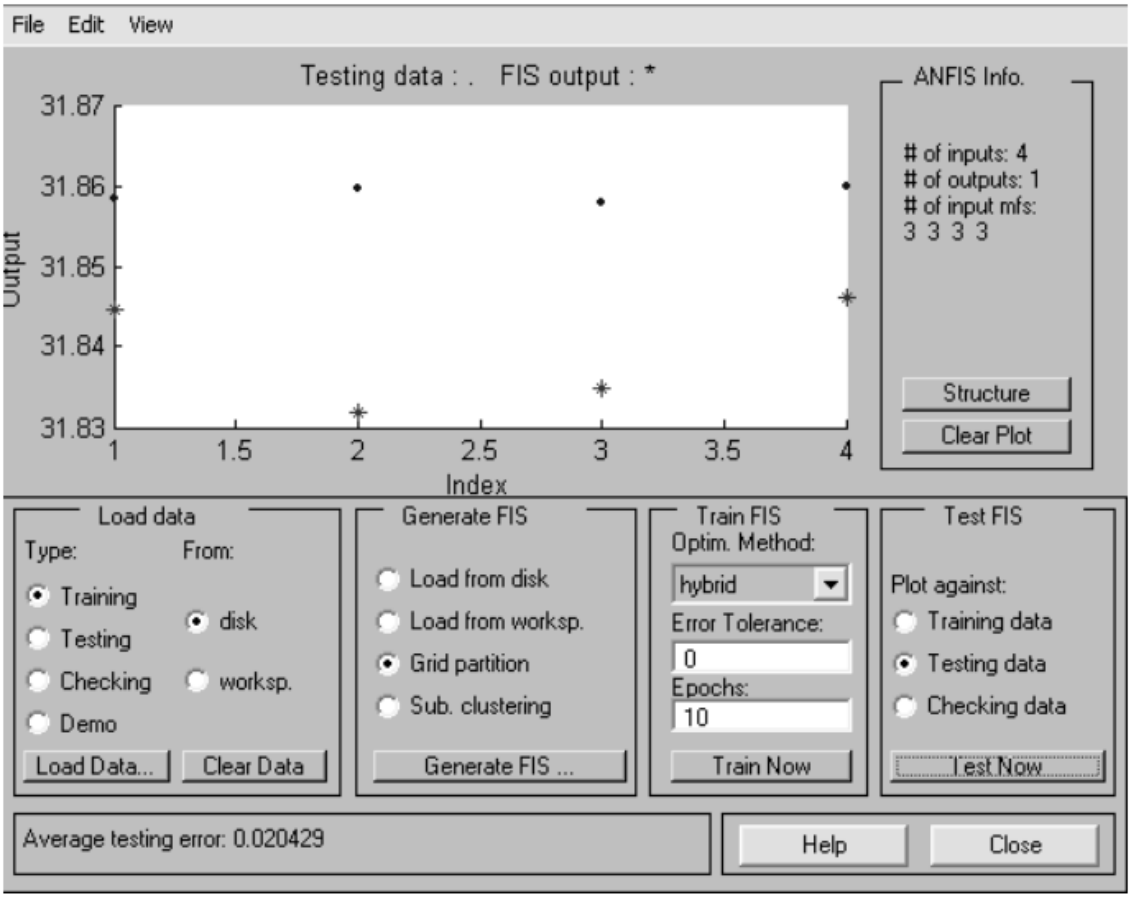

Рис. 10. Результаты моделирования ННС для тестирующих данных

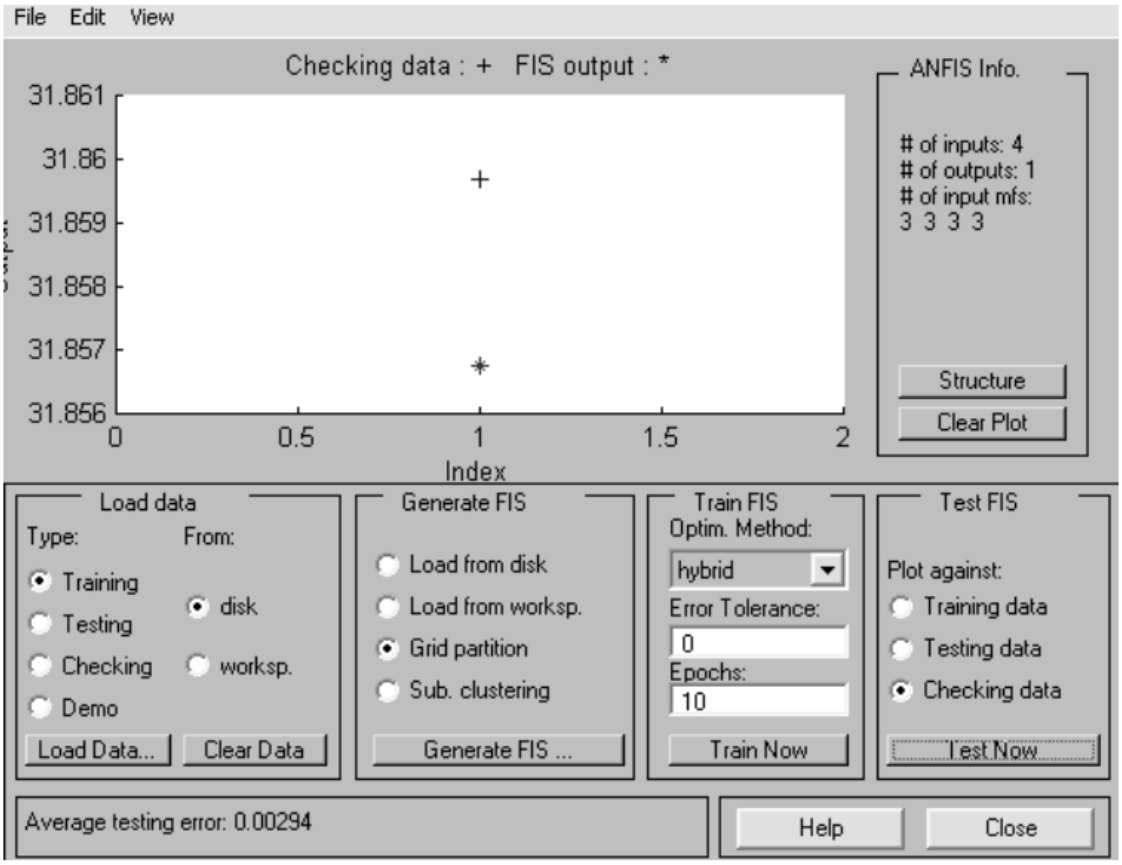

Рис. 11. Результаты моделирования ННС для контрольных данных

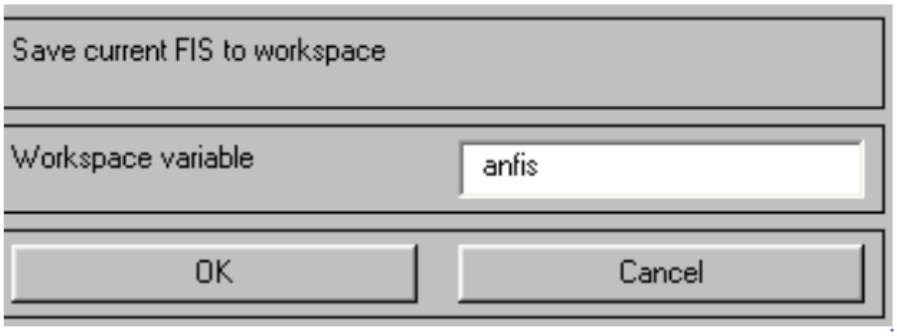

Рис. 12. Экспорт результатов

В данном случае, по сравнению с обычной нейронной сетью прямого распространения, полученный результат более близок к реальному.

Задание

1. Сформулировать задачу прогнозирования из области информатики и вычислительной техники, для решения которой было бы обосновано применение НС и ННС.
2. Сформировать обучающую выборку для НС и ННС.
3. Построить НС и, экспериментируя с количеством нейронов во входных и скрытых слоях, функциями активации, методами обучения, добиться наилучшего результата по прогнозированию.
4. Визуализировать полученную структуру НС в системе MATLAB.
5. Построить ННС и, экспериментируя с методами обучения, количеством функций принадлежности во входном слое, добиться наилучшего результата прогнозирования. Количество входов взять таким же, как и в п. 3 индивидуального задания.
6. Визуализировать полученную структуру ННС в системе MATLAB. Построить систему нечеткого логического вывода для полученной ННС.
7. Сравнить полученные с помощью НС и ННС результаты (численно) и сделать выводы.

Рекомендации

В качестве обучающей выборки (входных данных для НС, и данные в training.dat для ГНС) необходимо выбирать реальные значения для прогнозирования либо, если таковых не имеется, сгенерировать свои данные. ВАЖНО: при генерации своих данных используйте малые границы в генераторе случайных чисел. Отсортируйте данные таким образом, чтобы на одном участке наблюдалось снижение вашей наблюдаемой величины, а на другом повышение. Это обеспечит корректный вывод при обучении вашей НС или ГНС.

Для записи файлов .dat воспользуйтесь в качестве примера следующим кодом:

```
a = 1;                              % нижняя граница диапазона
b = 10;                             % верхняя граница диапазона
% создаем матрицу 2x50
A = a + (b-a)*rand(2,50);
fid = fopen('my_file.dat', 'wb');   % открытие файла на запись
if fid == -1                        % проверка корректности открытия
        error('File is not opened');
end
fwrite(fid, A, 'double');           % запись матрицы в файл (40 байт)
fclose(fid);                        % закрытие файла
```

MATLAB автоматически создает файл my_file.dat.

Контрольные вопросы

1. Дайте определение нейронечеткой сети.
2. Каково предназначение сетей нейронечеткого вывода?
3. В чем преимущества использования нейронечетких сетей?
4. Охарактеризуйте структуру нейронечеткой сети.
5. Опишите процесс разработки нейронечеткой сети в среде MATLAB.
6. Как проверить адекватность построенной нейронечеткой сети?
7. Какие возможности по визуализации результатов моделирования предоставляет система MATLAB?