Лабораторная работа №2.
Нечеткие нейронные сети.

Цель: моделирование нечеткого вывода с использованием нейронной сети. Синтез нечетких моделей с помощью системы нейро-нечеткого вывода ANFIS в среде MatLab.

Вводная часть:

Каждая разновидность систем искусственного интеллекта имеет свои особенности, например, по возможностям обучения, обобщения и выработки выводов, что делает ее наиболее пригодной для решения одного класса задач и менее пригодной — для другого.

Например, нейронные сети хороши для задач распознавания образов, но весьма неудобны для выяснения вопроса, *как они такое распознавание осуществляют.* Они могут автоматически приобретать знания, но процесс их обучения зачастую происходит достаточно медленно, а анализ обученной сети весьма сложен (обученная сеть обычно — черный ящик для пользователя). При этом какую-либо априорную информацию (знания эксперта) для ускорения процесса ее обучения в нейронную сеть ввести невозможно.

Системы с нечеткой логикой, напротив, хороши для объяснения получаемых с их помощью выводов, но они не могут автоматически *приобретать* знания для использования их в механизмах выводов. Необходимость разбиения универсальных множеств на отдельные области, как правило, ограничивает количество входных переменных в таких системах небольшим значением.

Вообще говоря, теоретически, системы с нечеткой логикой и искусственные нейронные сети эквивалентны друг другу, однако, в соответствии с изложенным выше, на практике у них имеются свои собственные достоинства и недостатки. Данное соображение легло в основу аппарата *Нечетких Нейронных Сетей (Fuzzy Neural Network)*, в которых выводы делаются на основе аппарата нечеткой логики, но соответствующие функции принадлежности подстраиваются с использованием алгоритмов обучения нейросетей, например, алгоритма обратного распространения ошибки. Такие системы не только используют априорную информацию, но могут приобретать новые знания и для пользователя являются логически прозрачными.

Теоретический материал:

Алгоритм настройки нечетких нейронных сетей рассмотрим на примере системы, включающей следующие правила (аналог алгоритма Sugeno):

П1: если x есть A_1, тогда z есть B_1,

П2: если x есть A_2, тогда z есть B_2,

………………………………

ПN: если x есть A_N, тогда z есть B_N,

при этом предполагается, что нечеткие понятия A_i имеют сигмоидные функции

$$A_i(x) = \frac{1}{1 + e^{b_i(x - a_i)}}$$

характеризующиеся параметрами a_i и b_i.

Степени истинности правил определяются в данном случае соотношением

$$\alpha_i = A_i(x) = \frac{1}{1 + e^{b_i(x - a_i)}}$$

Функции B_i – заданы прямыми

$$B_i(z) = c_i z + k_i$$

а выход системы - выражением

$$z_0 = \frac{\sum_k \alpha_k z_k}{\sum_k \alpha_k}; \qquad k = \overline{1..N}$$

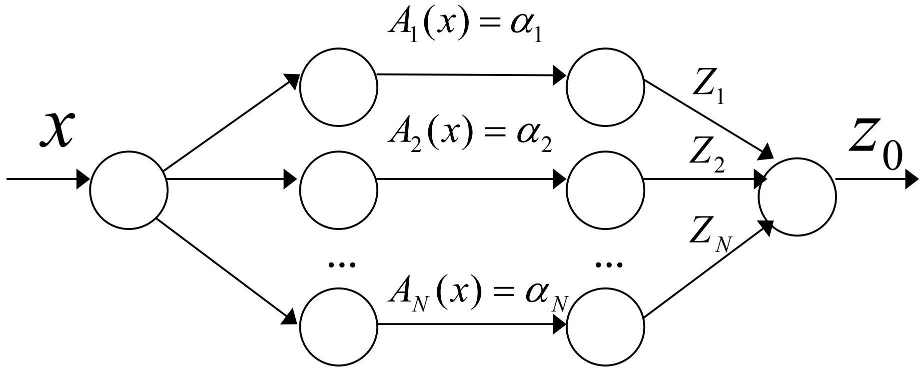
Предположим, что имеется обучающее множество $\{(x^1, z_0^1), \ldots, (x^P, z_0^P)\}$, отображающее неизвестную функцию.

Требуется: осуществить такую настройку параметров системы a_i, b_i, c_i, k_i, при которой обеспечивается наилучшая аппроксимация данной функции.

В данном случае функция ошибки для одного входного образа может быть записана в форме

$$E = \frac{1}{2}(z_0 - e)^2$$

Архитектура нейронной сети для предложенной системы правил:

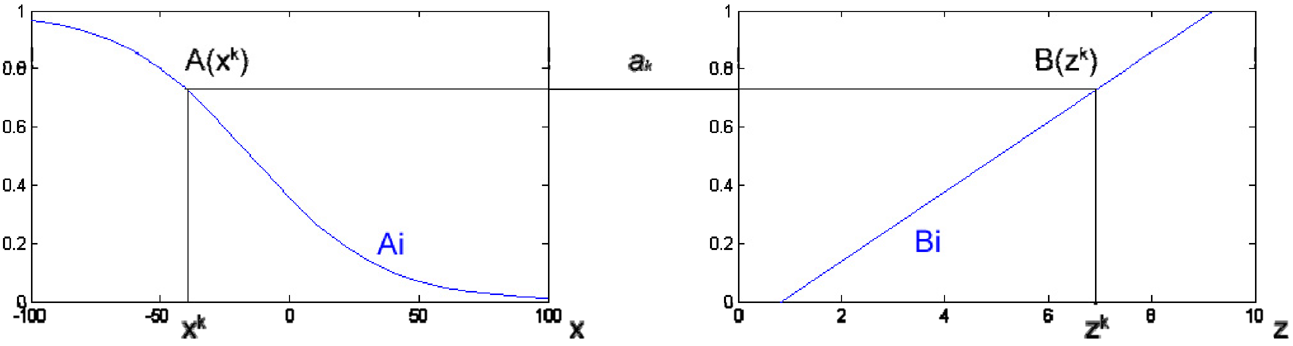


Входной нейрон выполняет распределительные функции. Нейроны первого скрытого слоя вычисляют значения функции принадлежности $a_1, a_2, \ldots, a_N$. Во втором слое выполняется расчет четкого значения заключения $z_1, z_2, \ldots, z_N$ для каждого правила по формуле

$$z_i = B^{-1}(\alpha_i)$$

Последний нейрон определяет общий выход системы.

Такая нечеткая нейронная сеть выполняет следующее преобразование для каждого правила Π_i:



Найдем выражение для вычисления z_i:

$$\alpha_i = c_i z_i + k_i$$

$$z_i = B^{-1}(\alpha_i) = \frac{\alpha_i - k_i}{c_i}$$

Настраиваемые параметры: a_i, b_i, c_i и k_i. Для их вычисления используем метод градиентного спуска:

$$a_i(t+1) = a_i(t) - \gamma \frac{\partial E}{\partial a_i} \qquad b_i(t+1) = b_i(t) - \gamma \frac{\partial E}{\partial b_i}$$

$$c_i(t+1) = c_i(t) - \gamma \frac{\partial E}{\partial c_i} \qquad k_i(t+1) = k_i(t) - \gamma \frac{\partial E}{\partial k_i}$$

Возьмем необходимые производные:

1 – для случая параметров c_i и k_i:

$$\frac{\partial E}{\partial k_i} = \frac{\partial E}{\partial z_0} \frac{\partial z_0}{\partial z_i} \frac{\partial z_i}{\partial c_i} \qquad \frac{\partial E}{\partial k_i} = \frac{\partial E}{\partial z_0} \frac{\partial z_0}{\partial z_i} \frac{\partial z_i}{\partial k_i}$$

$$\frac{\partial E}{\partial z_0} = (z_0 - e)$$

$$\frac{\partial z_0}{\partial z_i} = \frac{\alpha_i}{\sum_k \alpha_k}; \qquad k = \overline{1..N}$$

$$\frac{\partial z_i}{\partial c_i} = -\frac{\alpha_i - k_i}{c_i^2} \qquad\qquad \frac{\partial z_i}{\partial k_i} = -\frac{1}{c_i}$$

2 – для случая параметров a_i и b_i:

$$\frac{\partial E}{\partial a_i} = \frac{\partial E}{\partial z_0}\frac{\partial z_0}{\partial \alpha_i}\frac{\partial \alpha_i}{\partial a_i} \qquad\qquad \frac{\partial E}{\partial b_i} = \frac{\partial E}{\partial z_0}\frac{\partial z_0}{\partial \alpha_i}\frac{\partial \alpha_i}{\partial b_i}$$

$$\frac{\partial E}{\partial z_0} = (z_0 - e)$$

$$\frac{\partial z_0}{\partial \alpha_i} = \frac{(\sum_k \alpha_k z_k)'\sum_k \alpha_k - \sum_k \alpha_k z_k (\sum_k \alpha_k)'}{(\sum_k \alpha_k)^2} = \frac{z_i \sum_k \alpha_k - \sum_k \alpha_k z_k}{(\sum_k \alpha_k)^2}; \qquad k = \overline{1..N}$$

Пусть

$$S_i = b_i(x - a_i)$$

Тогда

$$\frac{\partial \alpha_i}{\partial a_i} = \frac{\partial \alpha_i}{\partial S_i}\frac{\partial S_i}{\partial a_i} \qquad\qquad \frac{\partial \alpha_i}{\partial b_i} = \frac{\partial \alpha_i}{\partial S_i}\frac{\partial S_i}{\partial b_i}$$

$$\frac{\partial \alpha_i}{\partial S_i} = -\frac{e^{S_i}}{(1+e^{S_i})^2} = -\frac{1}{1+e^{S_i}}\left(1 - \frac{1}{1+e^{S_i}}\right) = -\alpha_i(1-\alpha_i)$$

$$\frac{\partial S_i}{\partial a_i} = -b_i \qquad\qquad \frac{\partial S_i}{\partial b_i} = (x - a_i)$$

Собрав все рассчитанные производные и подставив их в выражения для $a_i(t+1)$, $b_i(t+1)$, $c_i(t+1)$, $k_i(t+1)$, получим окончательные формулы для вычисления настраиваемых параметров:

$$a_i(t+1) = a_i(t) - \gamma(z_0 - e)\frac{z_i \sum_k \alpha_k - \sum_k \alpha_k z_k}{(\sum_k \alpha_k)^2}\alpha_i(1-\alpha_i)b_i; \qquad k = \overline{1..N}$$

$$b_i(t+1) = b_i(t) + \gamma(z_0 - e)\frac{z_i\sum_k \alpha_k - \sum_k \alpha_k z_k}{(\sum_k \alpha_k)^2}\alpha_i(1-\alpha_i)(x-a_i); \qquad k = \overline{1..N}$$

$$c_i(t+1) = c_i(t) + \gamma(z_0 - e)\frac{\alpha_i}{\sum_k \alpha_k}\frac{(\alpha_i - k_i)}{c_i^2}; \qquad k = \overline{1..N}$$

$$k_i(t+1) = k_i(t) + \gamma(z_0 - e)\frac{\alpha_i}{\sum_k \alpha_k}\frac{1}{c_i}; \qquad k = \overline{1..N}$$

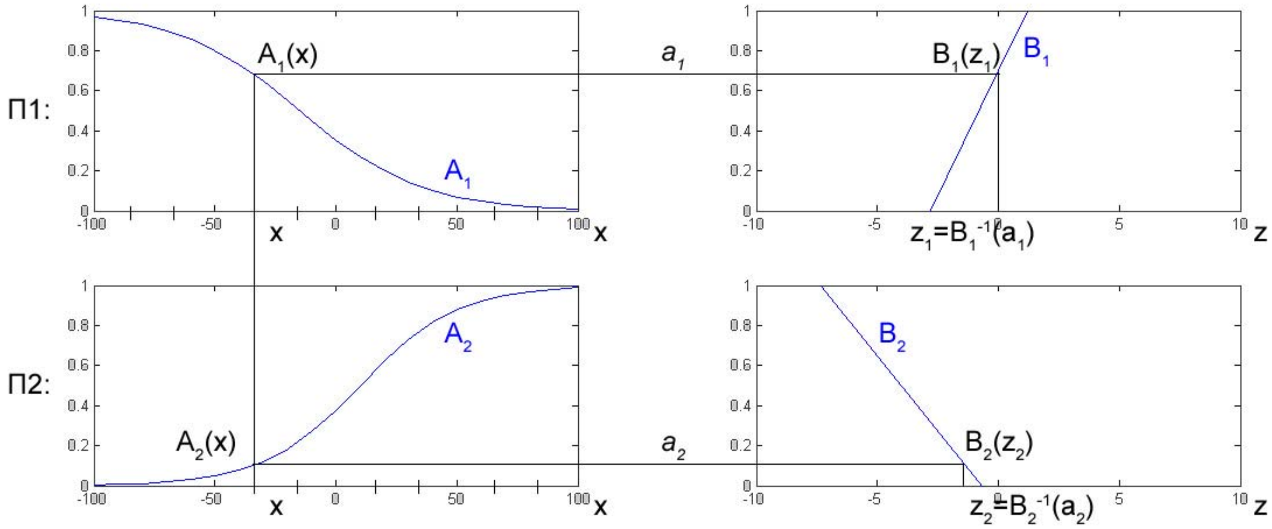
Задание для выполнения:

1) Используя эталонную нейро-нечеткую сеть из двух правил, описанную в теоретической части,

П1: если x есть A_1, тогда z есть B_1,
П2: если x есть A_2, тогда z есть B_2,

$$z_0 = \frac{\alpha_1 z_1 + \alpha_2 z_2}{\alpha_1 + \alpha_2}$$

получить обучающую выборку: множество значений $\{(x^1, z_0^1), \ldots, (x^P, z_0^P)\}$ в диапазоне *[-100..100]*, где x – значение входной переменной, а e – четкое значение выхода нечеткой системы.



Неизвестные параметры сигмоидных функции A_1 и A_2, а так же линейных функции B_1 и B_2 взять из варианта задания.

2) Используя сформированную выборку в модуле ANFIS системы MatLab *(см. файл anfis_forms.doc)* выполнить построение и обучение (т.е. настройку параметров a_i, b_i, c_i и k_i) нечеткой нейронной сети, рассмотренной в теоретической

части. Функции принадлежности для входных и выходной переменных подобрать самостоятельно. Установить оптимальный метод обучения (*backpropa* или *hybrid*).

Привести график промежуточных результатов обучения, а также информацию о числе эпох и значении ошибки в форме:

Число эпох	
Значение ошибки	

3) Протестировать нечеткую нейронную сеть. Сверить результаты работы обученной нейронной сети с исходными данными (из пункта 1). Результат представить графически в виде следующей таблицы:

x	*Результаты обучения*		
	e	z_0	$\Delta = \lvert z_0 - e \rvert$

4) Кроме того, для просмотра результатов работы системы воспользоваться графическими возможностями пакета Fuzzy Logic Toolbox: модуль Rule Viewer и модуль Surface Viewer .

5) Увеличить количество правил в исходной нейронной сети и повторить пункты 2-4.

6) Сделать выводы по выполненной работе и оформить отчет.

Варианты:

№	Сигмоидная функция Ai		Линейная функция Bi	
1	a1=15; a2=-10	b1=0.04; b2=-0.05	c1=0.05; c2=-0.03	k1=0.7; k2=-0.1
2	a1=12; a2=-20	b1=-0.03; b2=0.06	c1=0.02; c2=-0.02	k1=0.5; k2=0.5
3	a1=40; a2=0	b1=-0.07; b2=0.05	c1=0.02; c2=-0.05	k1=1; k2=-0.2
4	a1=40; a2=-40	b1=0.07; b2=0.08	c1=0.05; c2=-0.03	k1=-0.9; k2=1.5
5	a1=35; a2=-45	b1=0.05; b2=-0.09	c1=-0.05; c2=0.02	k1=1.5; k2=0
6	a1=-20; a2=0	b1=-0.1; b2=0.06	c1=0.02; c2=-0.02	k1=1; k2=1
7	a1=25; a2=-20	b1=-0.05; b2=0.09	c1=-0.04; c2=0.06	k1=0.2; k2=0