***РЕШЕНИЕ ЗАДАЧ АППРОКСИМАЦИИ СРЕДСТВАМИ***

***НЕЙРОСЕТЕВЫХ ТЕХНОЛОГИЙ***

Задачи:

– изучение структуры сетей прямого распространения;

– ознакомление с пакетом *Neural Networks Toolbox* системы (MATLAB);

– рассмотрение способов формирования сетей (изменение числа слоев и числа ИН);

– рассмотрение способов обучения сетей прямого распространения и RBF-сетей;

– оценка способов повышения точности аппроксимации при использовании ИНС;

– исследование эффективности нейросетевого подхода для формирования заданных функций.

1. Аппроксимируем функцию с помощью НС прямого распространения (feedforward)

|  |
| --- |
| Main.m  % Аппроксимация функции  % humps(x) = 1./((x– 0.3)\*0.2+0.01)+1./((x-0.9)\*0.2+0.04) – 6  x = -1:.05:2; % задание входного вектора;  y = humps(x);% задание выходного вектора;  P = x; T = y;  % Создание и тестирование нейронной сети  net = newff([-1 2],[20, 1],{'tansig','purelin'},'trainlm');  % newff– сеть прямого распространения(feedforward);  net.trainParam.show = 400; % результаты выводить через 400 итераций;  net.trainParam.lr = 0.05; % скорость обучения;  net.trainParam.epochs = 1000; % количество циклов обучения;  net.trainParam.goal = 1e-3; % заданная ошибка обучения;  % Обучение сети  net1 = train(net, P, T);  % Тестирование сети  a=sim(net1,P);  % Создание графиков исходного и аппроксимированного сигнала  plot(P,a,'k-'); grid; hold;  xlabel('time(s)'); ylabel('output'); title('humps function')  plot(P,T,'k\*')  gensim(net1) |

***Описание функций***

*Формирование ИНС прямого распространения*.

* имя = newff (PR, [S1 S2… SN ], {TF1 TF2… TFN}, BTF, BLF, PF).

Аргументы функции:

PR – R×2 – матрица минимальных и максимальных значений для R входных элементов;

Si– число нейронов i-го слоя;

TFi – функция активации i-го слоя, по умолчанию «*tansig*» (гиперболический тангенс);

BTF – функция обучения, по умолчанию «*traingd*» (алгоритм обучения обратного распространения ошибки),

BLF – функция настройки весов и смещений, по умолчанию «*learngdm*» (градиентный алгоритм оптимизации с инерционной составляющей),

PF – функция ошибки, по умолчанию «*mse*» (среднеквадратичная ошибка).

*Обучение ИНС*

* [net, tr, Y, E, Pf, Af] = train(NET, P, T, Pi,Ai,VV,TV)

Аргументы функции:

NET – сеть;

P – входы сети;

T – целевые значения, по умолчанию – нули;

Pi – начальные входные задержки, по умолчанию – нули;

Ai – начальные задержки слоев, по умолчанию – нули;

VV – структура векторов верификации, по умолчанию = [];

TV – структура тестовых векторов, по умолчанию = [];

Возвращает:

net – новая сеть (тренированная);

tr – результат тренировки (количество эпох и функция выполнения);

Y – выходы сети;

E – ошибки сети;

Pf – окончательные входные задержки;

Af – окончательные задержки слоев.

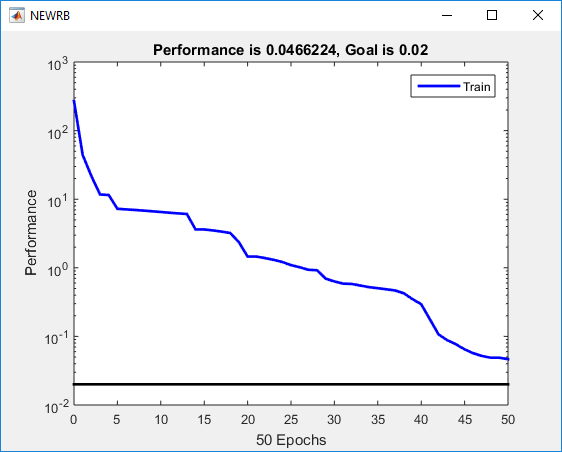


График зависимости ошибки обучения от числа циклов обучения

*Моделирование ИНС*

* [Y, Pf, Af, E, perf] = sim(net,P,Pi,Ai,T)

Аргументы функции:

net – сеть;

P – входы сети;

Pi – начальные задержки входов, по умолчанию – нули;

Ai – начальные задержки слоев, по умолчанию – нули;

T – целевые значения, по умолчанию – нули;

Y – выходы сети;

Pf – конечные входные задержки;

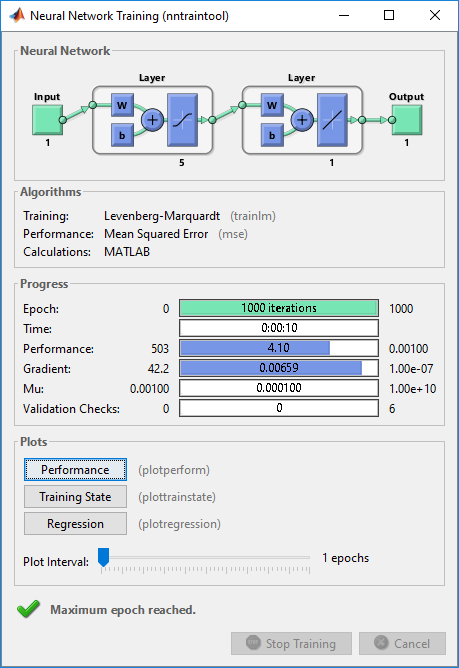
Af – конечные задержки слоев;

E – ошибки сети;

perf – параметр функционирование сети.

Созданная НС:

|  |
| --- |
|  |
| Первый слой |
| Второй слой |



*Увеличение числа слоев*

net = newff([-1 2],[5, 5, 3, 1],{'tansig','tansig','tansig','purelin'},'trainlm');

z=y-a;%ошибка

%mse(z)% mse-среднеквадратичная ошибка;

z=mean(abs(z))%средняя абсолютная ошибка

1. Реализация логических функции «и», «или», «исключающее или» с помощью RBF-сети

|  |
| --- |
|  |
| Рисунок 5.3.2 - Первый слой сети |
|  |

*Проверка программы*

|  |
| --- |
|  |
|  |
|  |
|  |

1. Формирование заданной функции с помощью RBF-сети