

МОСКОВСКИЙ ИНСТИТУТ ЭЛЕКТРОННОЙ ТЕХНИКИ

Институт системной и программной инженерии
и информационных технологий (Институт СПИНТех)

Лабораторный практикум по курсу "Нейронные сети"

Лабораторная работа 3.

Создание однонаправленной нейронной сети
с помощью нейронно-сетевого инструментария MATLAB.

Цель занятия – продемонстрировать основные этапы реализации нейронно-сетевого подхода для решения тестовой задачи с использованием нейронно-сетевого инструментария пакета MATLAB.

Можно выделить 5 основных этапов:

1. Подготовка данных для тренировки сети.
2. Создание сети.
3. Обучение сети.
4. Тестирование сети
5. Моделирование сети. (Использование сети для решения поставленной задачи).

В качестве примера рассмотрим следующую задачу:

Задан массив, состоящий из нескольких значений функции $y = Ce^{\frac{-(x-A)^2}{S}}$ ($S > 0$) на интервале $(0,1)$. Создать нейронную сеть прямого распространения (англ. feedforward neural network) (многослойный перцептрон), что при вводе значений функции на входы сети на выходах получались бы значения параметров C, A, S .

1. Подготовка данных для обучения сети

В первую очередь необходимо определиться с размерностью входного массива. Выберем количество значений функции равным $N = 21$, т. е. в качестве входных векторов массива используем значения функции y в точках $x = 0.05, \dots, 1.0$.

Указание: Количество значений функции может варьироваться, т. е. Вы можете выбирать это значение произвольно, в зависимости от условий задачи. При этом и значения функции будут, естественно, задаваться в других точках.

Для обучения сети необходимо сформировать массив входных векторов для различных наборов параметров C, A, S . Каждый набор этих параметров является вектором-эталоном для соответствующего входного вектора.

Для подготовки входного и эталонного массивов воспользуемся следующим алгоритмом. Выбираем случайным образом значения компонент вектора – эталона C, A, S и вычисляем компоненты соответствующего входного вектора. Повторяем эту процедуру

M раз и получаем массив входных векторов в виде матрицы размерностью $N \times M$ и массив векторов – эталонов в виде матрицы размерностью в нашем случае $3 \times M$.

Полученные массивы мы можем использовать для обучения сети.

Прежде чем приступать к формированию обучающих массивов, необходимо определиться с некоторыми свойствами массивов.

1. Выберем диапазоны изменения параметров C, A, S равными $(0.1, 1)$. Значения, близкие к 0 и сам 0 исключим в связи с тем, что функция не определена при $S = 0$. Второе ограничение связано с тем, что при использовании типичных передаточных функций желательно, чтобы компоненты входных и выходных векторов не выходили за пределы диапазона $(-1, 1)$. В дальнейшем мы познакомимся с методами нормировки, которые позволяют обойти это ограничение.
2. Количество входных и эталонных векторов выберем равным $M = 100$. Этого достаточно для обучения, а процесс обучения не займет много времени.

Тестовые массивы и эталоны подготовим с помощью программы *mas1*:

```
% Формирование входных массивов (входной массив P) и
(эталонны T)
P=zeros(100,21);
T=zeros(3,100);
x=0:5.e-2:1;
for i=1:100
    c=0.9*rand+0.1;
    a=0.9*rand+0.1;
    s=0.9*rand+0.1;
    T(1,i)=c;
    T(2,i)=a;
    T(3,i)=s;
    P(i,:)=c*exp(-(x-a).^2/s));
end;
P=P';
```

С помощью этой программы формируется матрица P из $M = 100$ входных векторов-столбцов, каждый из которых сформирован из 21 точки исходной функции со случайно выбранными значениями параметров C, A, S , и матрица T эталонов из 100 эталонных векторов-столбцов, каждый из которых сформирован из 3 соответствующих эталонных значений. Матрицы P и T будут использованы при обучении сети. Следует отметить, что при каждом новом запуске этой программы будут формироваться массивы с новыми значениями компонентов векторов, как входных, так и эталонных.

2. Создание сети

Указание: Как отмечалось в лекционном курсе, выбор архитектуры сети для решения конкретной задачи основывается на опыте разработчика. Поэтому предложенная ниже архитектура сети является одним вариантом из множества возможных конфигураций.

В Matlab имеется хорошо разработанный инструментарий для создания, обучения, тестирования нейронных сетей, а также последующего моделирования, а именно Neural Network Toolbox. Для ознакомления выполните команду *nnstart*. Для более углублённого изучения можно порекомендовать книгу [1].

Для решения поставленной задачи сформируем трехслойную сеть прямого распространения, включающую 21 нейрон во входном слое (по числу компонент входного вектора) с передаточной функцией *logsig*, 15 нейронов во втором слое с передаточной функцией *logsig* и 3 нейрона в выходном слое (по числу компонент выходного вектора) с передаточной функцией *purelin*. При этом в качестве обучающего алгоритма выбран алгоритм Левенберга-Марквардта (Levenberg-Marquardt) (*trainlm*). Этот алгоритм обеспечивает быстрое обучение, но требует много ресурсов. В том случае, если для реализации этого алгоритма для выбранной размерности задачи не хватит оперативной памяти, можно использовать другие алгоритмы (*trainbfg*, *trainrp*, *trainscg*, *traincgb*, *traincgf*, *traincgp*, *trainoss*, *traingdx*). По умолчанию используется *trainlm*. Указанная сеть формируется с помощью процедуры:

```
net=newff(minmax(P),[21,15,3],{'logsig','logsig','purelin'},...  
'trainlm');
```

Первый аргумент - матрица $M \times 2$ минимальных и максимальных значений компонент входных векторов вычисляется с помощью процедуры *minmax*.

Указание: Следует правильно употреблять кавычки в приведённой команде, в противном случае будет выдаваться сообщение об ошибках. Для определённости строчка скрипта приводится в формате MS Word, а именно,

```
net=newff(minmax(P),[21,15,3],{'logsig','logsig','purelin'},'trainlm');
```

Кроме того, рекомендуется провести создание архитектуры сети с помощью новой функции *feedforwardnet* (*hiddenSizes,trainFcn*), которая вводится в новых версиях пакета Matlab на смену функции *newff*. Для получения справки по новой функции *feedforwardnet* сделайте вызов *help feedforwardnet*.

Результатом выполнения процедуры *newff* или, соответственно, *feedforwardnet*, является объект – нейронная сеть *net* заданной конфигурации. Сеть можно сохранить на диске в виде *mat*. файла с помощью команды *save* и загрузить снова с помощью команды *load*. Это то самое «ядро», которое после обучения нейронной сети может быть использовано в качестве нейросетевого процессора для выполнения конкретного приложения, в нашем случае для определения коэффициентов функции C, A, S .

3. Обучение сети

Следующий шаг – обучение созданной сети. Перед обучением необходимо задать параметры обучения. Задаем функцию оценки функционирования *sse*.

```
net.performFcn='sse';
```

В этом случае в качестве оценки вычисляется сумма квадратичных отклонений выходов сети от эталонов.

Задаем критерий окончания обучения – значение отклонения, при котором обучение будет считаться законченным:

```
net.trainParam.goal=0.01;
```

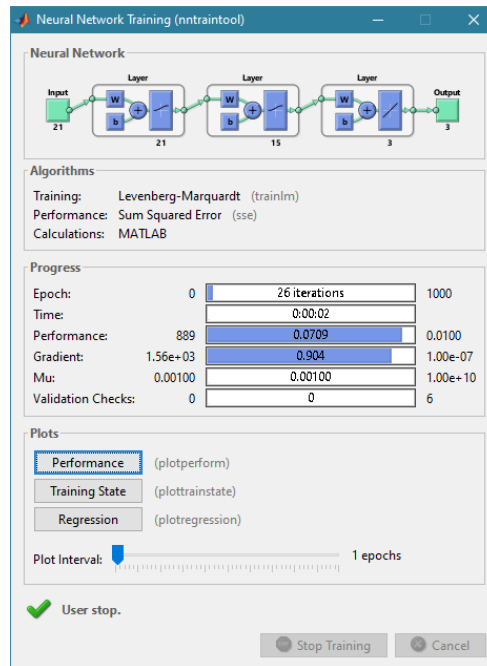
Задаем максимальное количество циклов обучения. После того, как будет выполнено это количество циклов, обучение будет завершено:

```
net.trainParam.epochs=1000;
```

Теперь можно начинать обучение:

```
[net,tr]=train(net,P,T);
```

Архитектура сети (модель), а также процесс обучения, отображаются на графическом интерфейсе нейро-сетевого инструментария.



Процесс обучения иллюстрируется графиком зависимости оценки функционирования от номера цикла обучения.

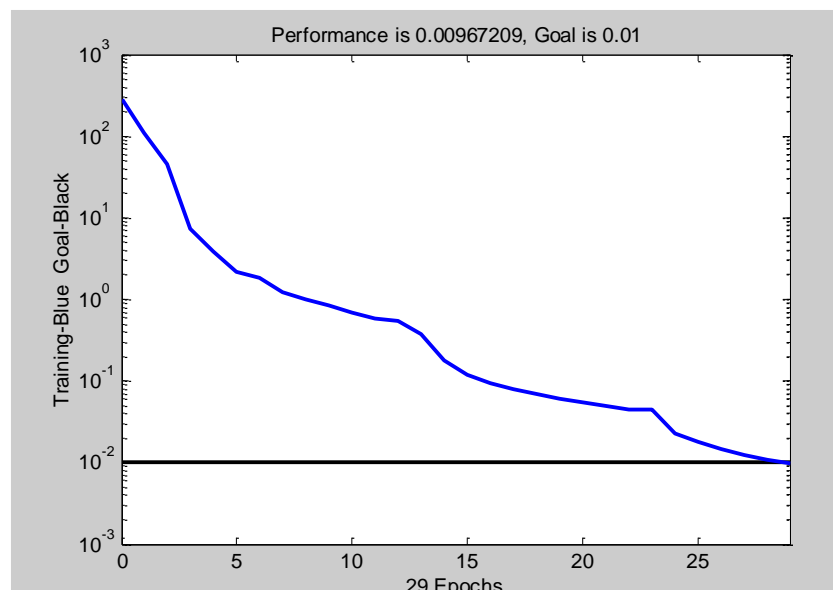


Рис. 1. Обучение нейронной сети

Таким образом, обучение сети окончено. Теперь эту сеть можно сохранить в файле nn1.mat:

```
save nn1 net;
```

4. Тестирование сети

Перед тем, как воспользоваться нейронной сетью, необходимо исследовать степень достоверности результатов вычислений сети на тестовом массиве входных векторов. В качестве тестового массива необходимо использовать массив, компоненты которого отличаются от компонентов массива, использованного для обучения. В нашем случае для получения тестового массива достаточно воспользоваться еще раз программой *mas1*.

Для оценки достоверности результатов работы сети можно использовать результаты регрессионного анализа, полученные в процессе сравнения эталонных значений со значениями, полученными на выходе сети, когда на вход поданы входные векторы тестового массива. В среде MATLAB для этого можно воспользоваться функцией *postreg*. Следующий набор команд иллюстрирует описанную процедуру:

```
mas1                -   создание тестового массива P
y=sim(net,P);       -   обработка тестового массива
[m,b,r]=postreg(y(1,:),T(1,:)) - регрессионный анализ
                                результатов обработки.
```

Сравнение компонентов эталонных векторов с соответствующими компонентами выходных векторов сети. Видно, что все точки легли на прямую линию, что говорит о правильной работе сети на тестовом массиве.

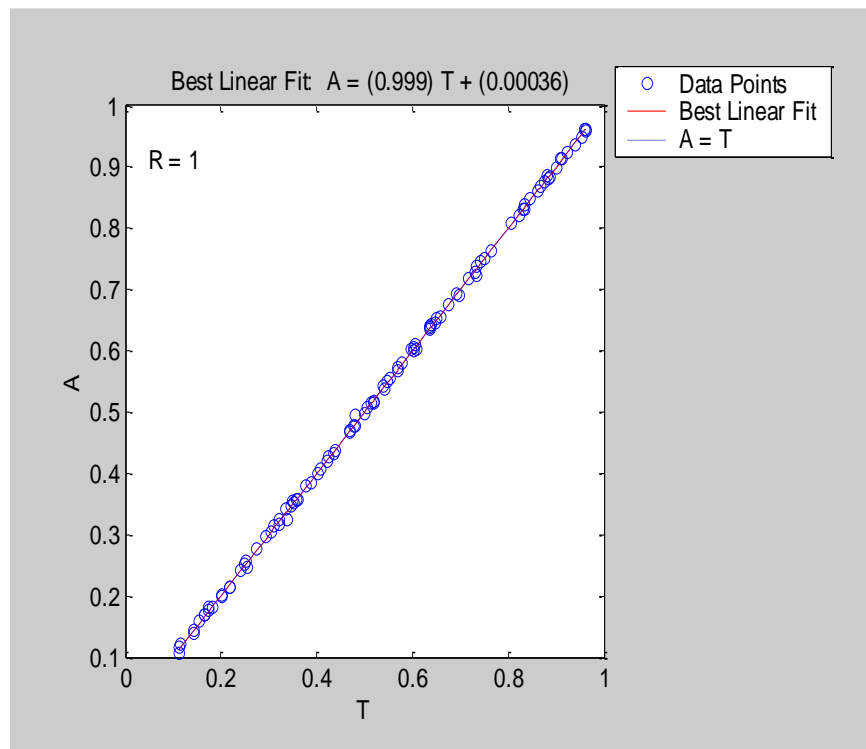


Рис. 2. Тестирование нейронной сети относительно параметра *C*

```
[m,b,r]=postreg(y(2,:),T(2,:));
```

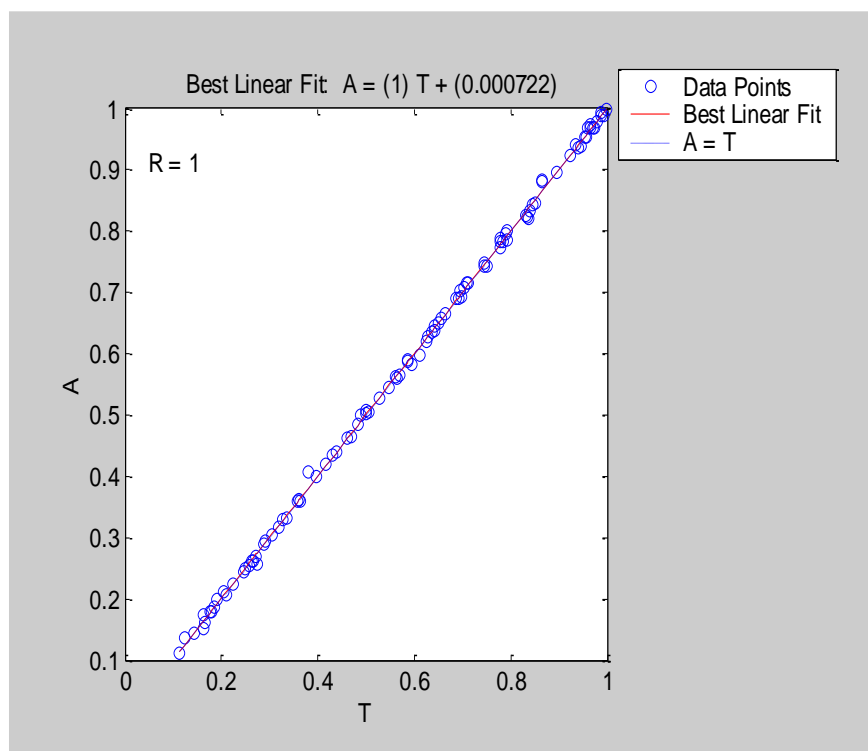


Рис. 3. Тестирование нейронной сети относительно параметра A

```
[m,b,r]=postreg(y(3,:),T(3,:));
```

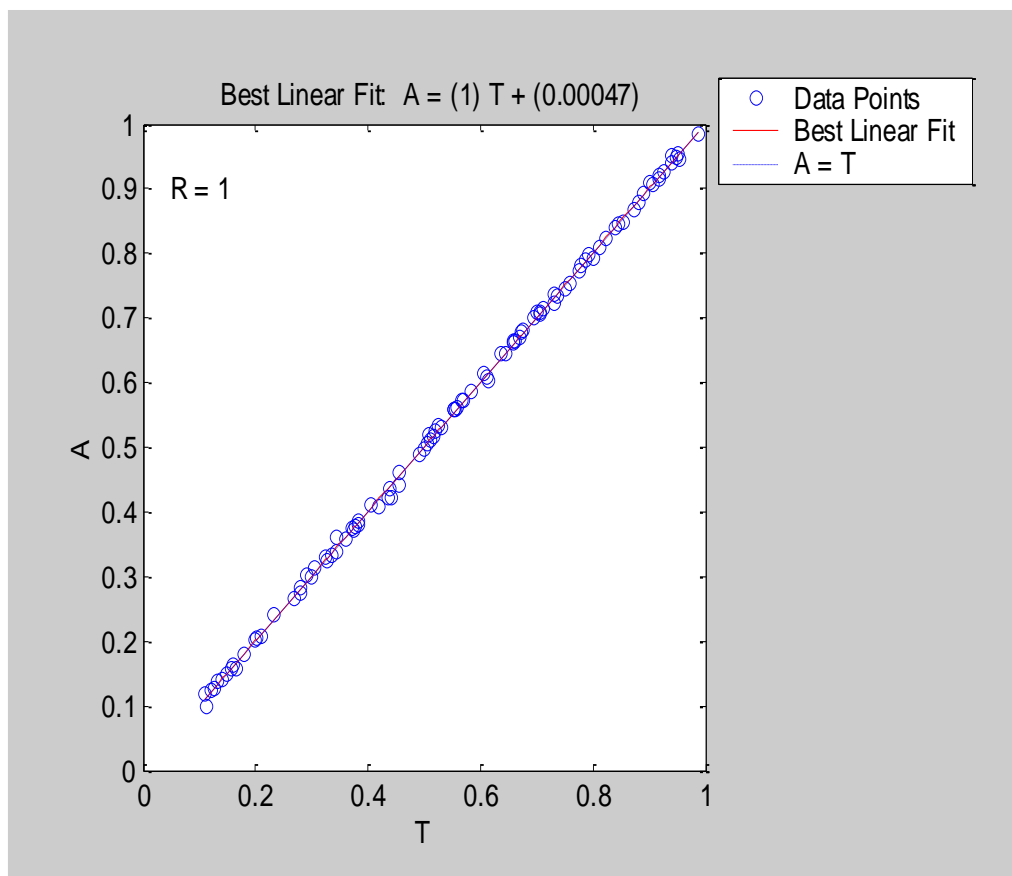


Рис. 4. Тестирование нейронной сети относительно параметра S

Как видно из рисунков, созданная нейронная сеть отлично решает поставленную задачу для всех трех выходных параметров. Сохраним обученную сеть `net` на диске в файл `nn1.mat`

5. Моделирование сети (использование сети для решения поставленной задачи)

Для того, чтобы применить обученную сеть для обработки данных, необходимо воспользоваться функцией `sim`:

```
Y=sim(net,p);
```

где p – набор входных векторов, Y – результат анализа в виде набора выходных векторов.

Например, пусть $C=0.2$, $A=0.8$, $S=0.7$, тогда

```
p=0.2*exp(-(x-0.8).^2/0.7));
```

Подставив этот входной вектор в качестве аргумента функции `sim` :

```
Y=sim(net,p')
```

Получим:

```
Y =  
    0.2048    (C)  
    0.8150    (A)  
    0.7048    (S)
```

Что весьма близко к правильному результату (0.2; 0.8; 0.7).

Задание.

1. Исследовать влияние шума в исходных данных на результаты обучения нейронной сети. Для этого к исходному массиву данных прибавить случайные числа из диапазонов (0 – 0.01; 0 – 0.05; 0 – 0.1; 0 -0.2). Провести процедуру обучения и протестировать сеть.
2. Сформировать исходный массив и массив эталонов из случайных чисел и провести обучение сети. Прокомментировать результаты.

Дополнительная литература.

1. Медведев В.С. Нейронные сети. MATLAB 6 / В.С. Медведев, В.Г. Потемкин. – М.: ДИАЛОГ-МИФИ, 2002. – 496 с.