

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)
Кафедра вычислительной математики и программирования

Лабораторная работа №2
по спецкурсу «Нейроинформатика»

Линейная нейронная сеть. Правило обучения
Уидроу-Хоффа

Выполнил: Днепров И.С.
Группа: М8О-407Б, вариант 10
Преподаватели: Тюменцев Ю.В.

Москва, 2020

Цель работы

Исследование свойств линейной нейронной сети и алгоритмов её обучения, применение сети в задачах аппроксимации и фильтрации.

Основные этапы работы

1. Использовать линейную нейронную сеть с задержками для аппроксимации функции. В качестве метода обучения использовать адаптацию.
2. Использовать линейную нейронную сеть с задержками для аппроксимации функции и выполнения многошагового прогноза.
3. Использовать линейную нейронную сеть в качестве адаптивного фильтра для подавления помех. Для настройки весовых коэффициентов использовать метод наименьших квадратов.

Оборудование

Процессор: 2,4 GHz Intel Core 2 Duo

Оперативная память: 8 ГБ 1067 MHz DDR3

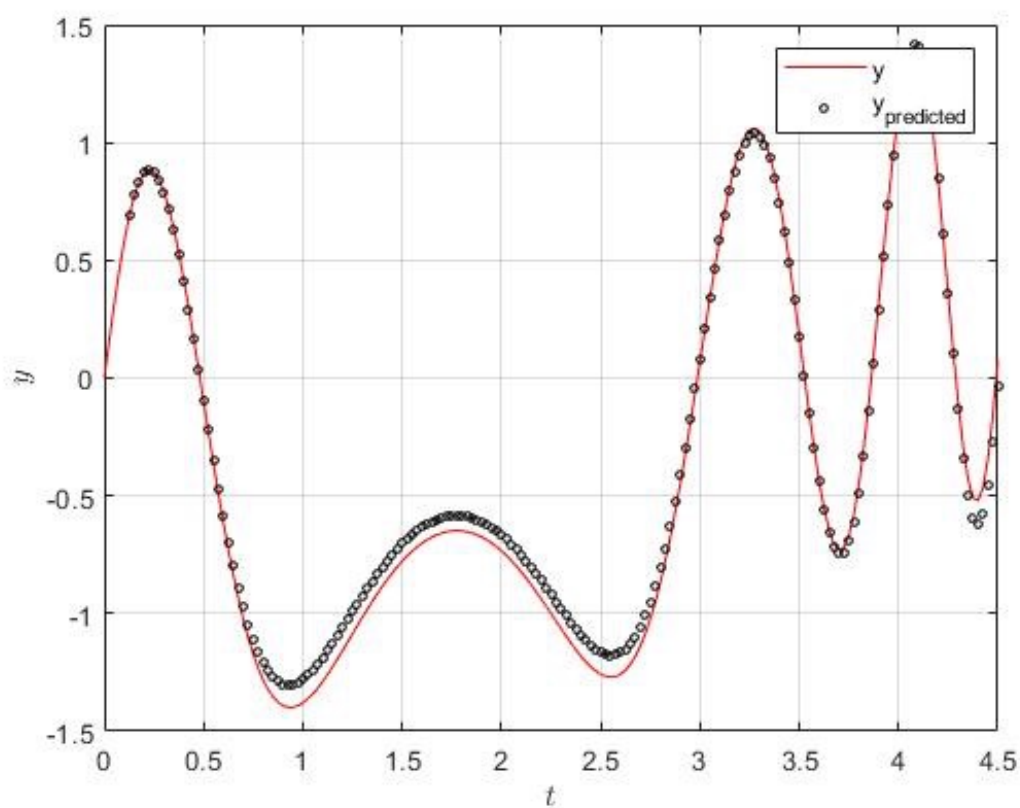
Программное обеспечение

Matlab R2020b, 64-bit.

Задание 1

```
|iteration: 1 | error: 0.629274|
|iteration: 2 | error: 0.037495|
|iteration: 3 | error: 0.036122|
|iteration: 4 | error: 0.035642|
|iteration: 5 | error: 0.035379|
|iteration: 6 | error: 0.035201|
|iteration: 7 | error: 0.035067|
|iteration: 8 | error: 0.034959|
|iteration: 9 | error: 0.034869|
|iteration: 10 | error: 0.034792|
. . . . .
|iteration: 40 | error: 0.033577|
|iteration: 41 | error: 0.033541|
|iteration: 42 | error: 0.033506|
|iteration: 43 | error: 0.033471|
|iteration: 44 | error: 0.033435|
|iteration: 45 | error: 0.033400|
|iteration: 46 | error: 0.033365|
|iteration: 47 | error: 0.033330|
|iteration: 48 | error: 0.033295|
|iteration: 49 | error: 0.033260|
|iteration: 50 | error: 0.033225|
```

эталонные значения и предсказанные сетью



Задание 2

Таблица 1:

Функция создания сети: `newline`

Входной слой: 1

Скрытый слой: нет

Выходной слой: 1

Активационные функции: линейная

Динамика: минимизация `mae`

Функция разделения обучающего множества: задержки

Число примеров в подмножествах: не задавал

Метод обучения: `train`

Параметры обучения:

```
network.trainParam.epochs = 600
```

```
network.trainParam.goal = 1e-6
```

Метод инициализации сети: `rands`

Критерий окончания обучения: `epochs = 600 | goal = 1e-6`

Причина окончания обучения: `epochs = 600`

Число эпох обучения: 600

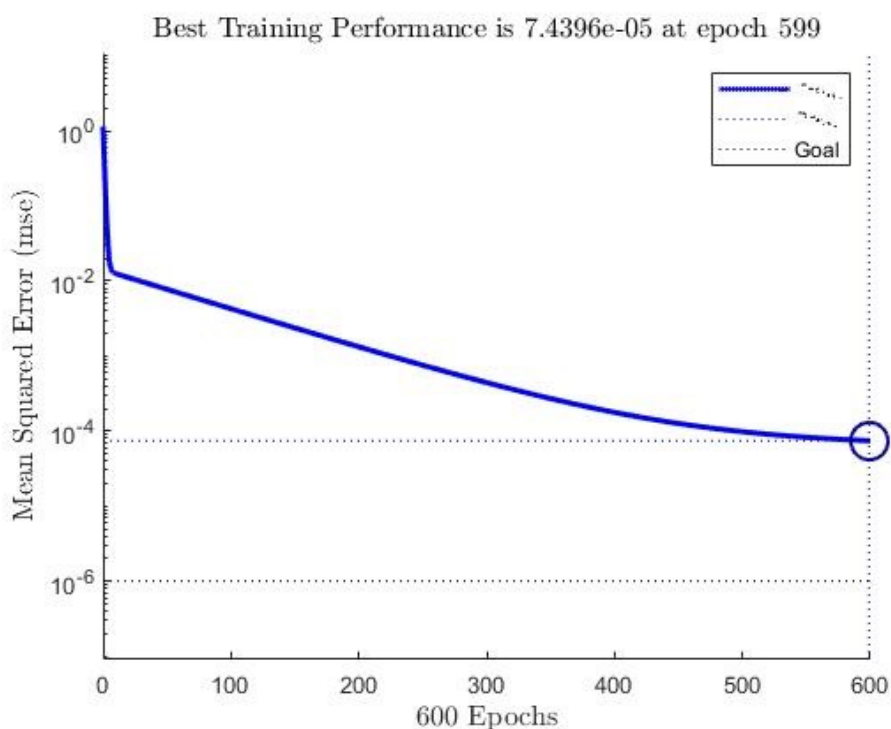
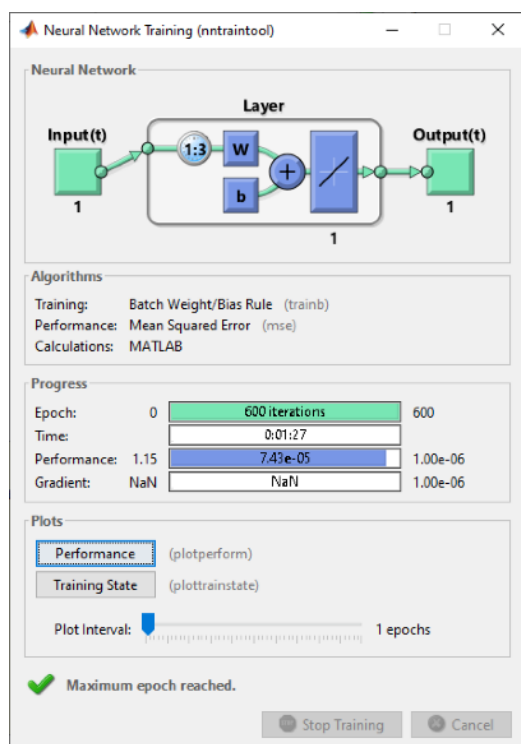


Таблица 2 и график для обучающего множества:

R^2 : 0.432275

MSE: 0.023011

RMSE: 0.151695

Относительная СК0: 24.418777%

MAE: 0.124227

min abs err: 0.010371

max abs err: 0.273873

MAPE: 17.053237

Доля с ошибкой менее 5%: 10.000000%

Доля с ошибкой от 5% до 10%: 20.000000%

Доля с ошибкой от 10% до 20%: 30.000000%

Доля с ошибкой от 20% до 30%: 30.000000%

Доля с ошибкой более 30%: 10.000000%

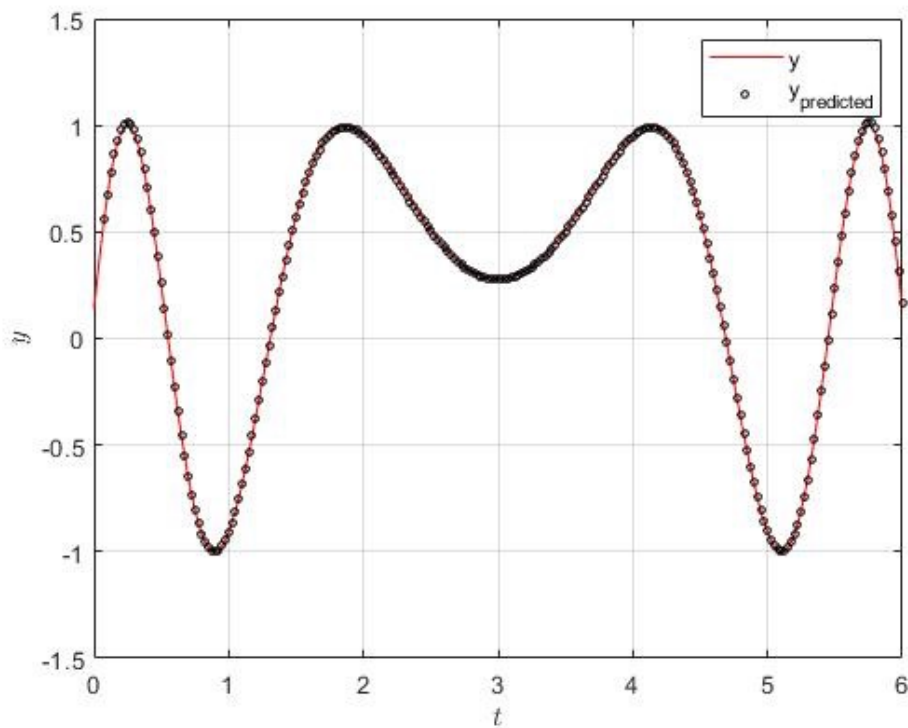


Таблица 2 и график для тестового множества:

R^2 : 0.432275

MSE: 0.023011

RMSE: 0.151695

Относительная СК0: 24.418777%

MAE: 0.124227

min abs err: 0.010371

max abs err: 0.273873

MAPE: 17.053237

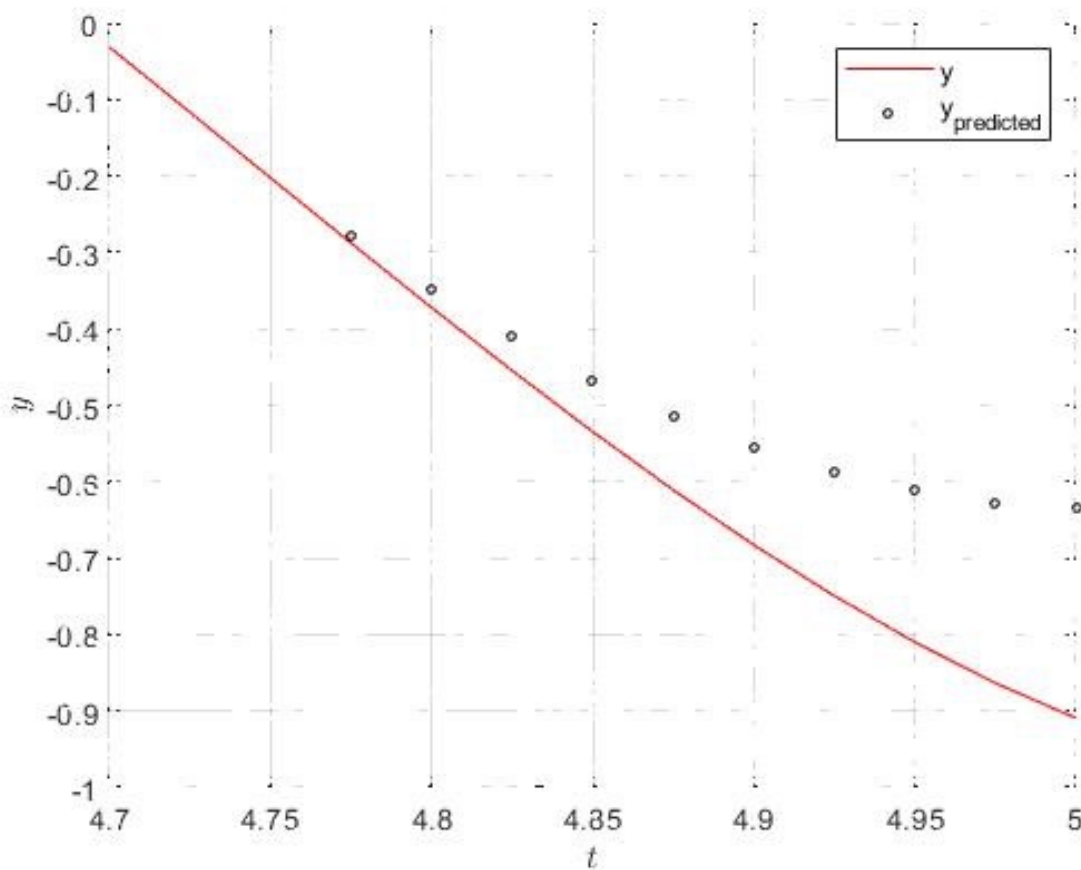
Доля с ошибкой менее 5%: 10.000000%

Доля с ошибкой от 5% до 10%: 20.000000%

Доля с ошибкой от 10% до 20%: 30.000000%

Доля с ошибкой от 20% до 30%: 30.000000%

Доля с ошибкой более 30%: 10.000000%



Задание 3

Весовые коэффициенты и смещение

$W = \begin{bmatrix} -1.23349539043816 & 0.996735826822112 \\ 0.99723889333458 & -1.0481119270546 \end{bmatrix}$

$b = -0.0233789785203087$

Сравнение выхода сети с эталонным множеством

$R^2: 0.599160$

MSE: 0.020762

RMSE: 0.144090

Относительная СК0: 21.613455%

MAE: 0.129449

min abs err: 0.000368

max abs err: 0.364117

MAPE: 325.783706

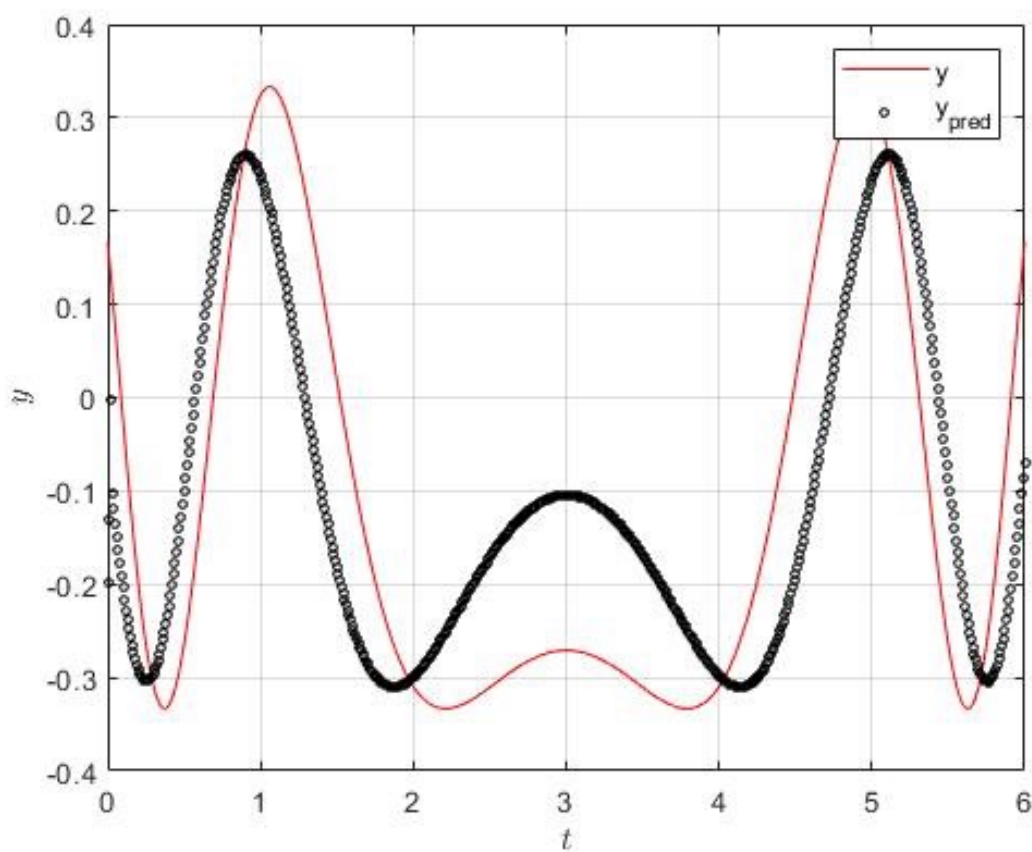
Доля с ошибкой менее 5%: 4.326123%

Доля с ошибкой от 5% до 10%: 4.326123%

Доля с ошибкой от 10% до 20%: 8.652246%

Доля с ошибкой от 20% до 30%: 9.317804%

Доля с ошибкой более 30%: 73.377704%



Код программы

Lab2.m

```
set(0, 'DefaultTextInterpreter', 'latex');

% Задание 1

% Входные данные

signalFunction = @(t) sin(-2 .* t .^ 2 + 7 .* t) - 0.5 .* sin(t);

t = 0:0.025:4.5;

signal = signalFunction(t);

D = 1:5; % задержки

x = con2seq(signal(D(end)+1:end)); % образцы
y = con2seq(signal(D(end)+1:end)); % цели

% Создаем сеть и инициализируем ее случайными значениями

network = newlin([-1 1], 1, D, 0.01);
network.inputweights{1}.initFcn = 'rands';
network.biases{1}.initFcn = 'rands';
network = init(network);

% Обучаем ее с помощью adapt

for i = 1:50
    [network, ~, err, ~] = adapt(network, x, y, con2seq(signal(D)));
    fprintf('| iteration: %d | error: %f |\n', i, sqrt(mse(err)));
end

%% Результаты обучения

predictedSignal = cell2mat(network([con2seq(signal(D)) x]));
```

```

    p = plot(t, signal, t(D(end)+1:end), predictedSignal(D(end)+1:end),
'o');

    p(1).Color = [1 0 0];
    p(2).MarkerSize = 3;
    p(2).Color = [0 0 0];

    grid on

    xlabel('$t$');
    ylabel('$y$');
    legend('y', 'y_{predicted}');

%% Задание 2

% Входные данные

signalFunction = @(t) sin(t.^ 2 - 6 .* t + 3);
t = 0:0.025:6;
signal = signalFunction(t);
D = 1:3; % задержки
x = con2seq(signal(D(end)+1:end)); % образцы
y = con2seq(signal(D(end)+1:end)); % цели

% Создаем сеть и инициализируем ее случайными значениями
network = newlin([-1 1], 1, D, maxlinlr(cell2mat(x), 'bias'));
network.inputweights{1}.initFcn = 'rands';
network.biases{1}.initFcn = 'rands';
network = init(network);

% Обучаем ее с помощью train
network.trainParam.epochs = 600;
network.trainParam.goal = 1e-6;
network = train(network, x, y, con2seq(signal(D))));

```

```

%% Результаты обучения

predictedSignal = cell2mat(network([con2seq(signal(D)) x]));

p = plot(t, signal, t(D(end)+1:end), predictedSignal(D(end)+1:end),
'o');

p(1).Color = [1 0 0];
p(2).MarkerSize = 3;
p(2).Color = [0 0 0];

grid on
xlabel('$t$');
ylabel('$y$');
legend('y', 'y_{predicted}');

% Данные для таблицы

display(dataForTable(cell2mat(y), predictedSignal(D(end)+1:end)));

%% Вычисляем прогноз

t = 4.7:0.025:5;

signal = signalFunction(t); %% len = 13
predictedSignal = [signal(1:3) zeros(1, 10)];

for i = 4:13
    tmp = cell2mat(network(con2seq(predictedSignal(i-3:i))));
    predictedSignal(i) = tmp(end);
end

% Строим график

p = plot(t, signal, t(4:end), predictedSignal(4:end), 'o');

p(1).Color = [1 0 0];
p(2).MarkerSize = 3;
p(2).Color = [0 0 0];

```

```

grid on
xlabel('$t$');
ylabel('$y$');
legend('y', 'y_{predicted}');

% Данные для таблицы
display(dataForTable(signal(4:end), predictedSignal(4:end)));

%% Задание 3

% Входные данные
t = 0:0.01:6;
x = sin(t.^2 - 6 * t + 3);
y = 1 / 3 * sin(t.^2 - 6 * t + pi / 6);
%x = cos(-5 * t.^2 + 10 * t - 5);           % входной сигнал
%y = 1 / 8 * cos(-5 * t.^2 + 10 * t);       % выходной сигнал

D = 4;           % глубина
Q = length(x); % число образцов

P = zeros(D, length(x));
for i = 1:D
    P(i,i:Q) = x(1:Q-i+1);
end

%% Создание сети + график + характеристики

network = newlind(P, y);

fprintf('W = %s\nb = %s\n\n', mat2str(network.IW{1}),
mat2str(network.b{1}));

```

```
display(dataForTable(y, network(P)));
```

```
p = plot(t, y, t, network(P), 'o');
```

```
p(1).Color = [1 0 0];
```

```
p(2).MarkerSize = 3;
```

```
p(2).Color = [0 0 0];
```

```
grid on
```

```
xlabel('$t$');
```

```
ylabel('$y$');
```

```
legend('y', 'y_{pred}');
```

dataForTable.m

```
function res = dataForTable(y, yp)
```

```
    R2 = 1 - sum((y - yp) .^ 2)/sum((y - mean(y)) .^ 2);
```

```
    MSE = mse(y - yp);
```

```
    RMSE = sqrt(MSE);
```

```
    CKO = RMSE / (max(y) - min(y)) * 100;
```

```
    MAE = mae(y - yp);
```

```
    MinAbsErr = min(abs(y - yp));
```

```
    MaxAbsErr = max(abs(y - yp));
```

```
    MAPE = mean(abs((y - yp) ./ y)) * 100;
```

```
    errors = abs((y - yp) ./ y) * 100;
```

```
    Under5PersetPortion = sum(errors < 5) / length(y) * 100;
```

```
    Under10PersetPortion = sum(5 <= errors & errors < 10) / length(y)  
    * 100;
```

```
    Under20PersetPortion = sum(10 <= errors & errors < 20) /  
length(y) * 100;
```

```

        Under30PersentPortion = sum(20 <= errors & errors < 30) /
length(y) * 100;

        Over30PersentPortion = sum(errors >= 30) / length(y) * 100;

res = sprintf(['R^2: %f\n' ...
              'MSE: %f\n' ...
              'RMSE: %f\n' ...
              'Относительная CKO: %f%%\n' ...
              'MAE: %f\n' ...
              'min abs err: %f\n' ...
              'max abs err: %f\n' ...
              'MAPE: %f\n' ...
              'Доля с ошибкой менее 5%%: %f%%\n' ...
              'Доля с ошибкой от 5%% до 10%%: %f%%\n' ...
              'Доля с ошибкой от 10%% до 20%%: %f%%\n' ...
              'Доля с ошибкой от 20%% до 30%%: %f%%\n' ...
              'Доля с ошибкой более 30%%: %f%%\n'], ...

              R2, MSE, RMSE, CKO, MAE, MinAbsErr, MaxAbsErr,
MAPE, Under5PersentPortion, Under10PersentPortion, Under20PersentPortion,
Under30PersentPortion, Over30PersentPortion);

end

```

Выводы

В данной лабораторной работе использована линейные нейронные сети с задержками для аппроксимации функции, многошаговой и адаптивной фильтрации. Я имел возможность сравнить применить различные методы обучения линейных нейросетей и сравнить результаты. Лабораторная работа оказалась интересна тем, что в ней решалось сразу несколько задач нейроинформатики при помощи линейных нейросетей.