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

Лабораторная работа №8
по спецкурсу «Нейроинформатика»

Динамические сети

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

Москва, 2020

Цель работы

Целью работы является исследование свойств некоторых динамических нейронных сетей, алгоритмов обучения, а также применение сетей в задачах аппроксимации функций и распознавания динамических образов.

Основные этапы работы:

1. Использовать сеть прямого распространения с запаздыванием для предсказания значений временного ряда и выполнения многошагового прогноза.

2. Использовать сеть сеть прямого распространения с распределенным запаздыванием для распознавания динамических образов.

3. Использовать нелинейную авторегрессионную сеть с внешними входами для аппроксима- ции траектории динамической системы и выполнения многошагового прогноза.

Оборудование

Процессор: 2,4 GHz Intel Core 2 Duo

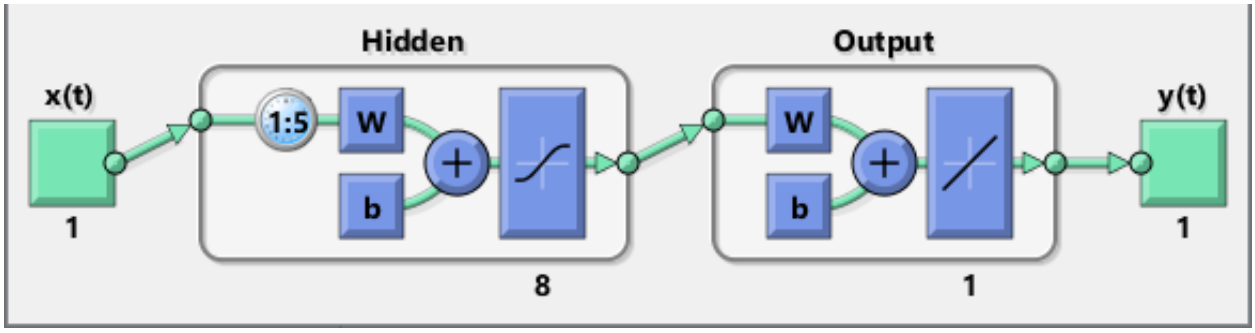
Оперативная память: 8 ГБ 1067 MHz DDR3

Программное обеспечение

Matlab R2020b, 64-bit.

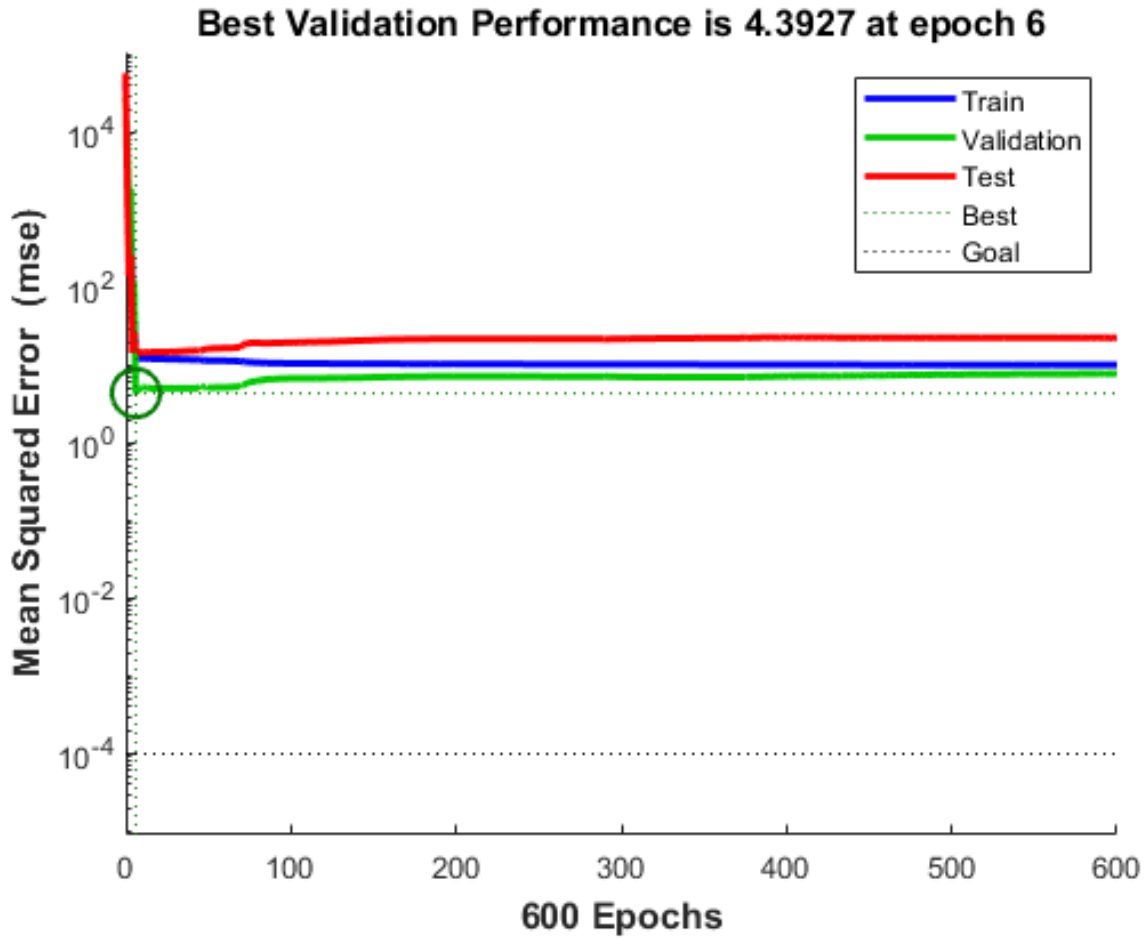
Задание 1.

Структура сети:



Обучение сети:

Проверка качества разбиения:



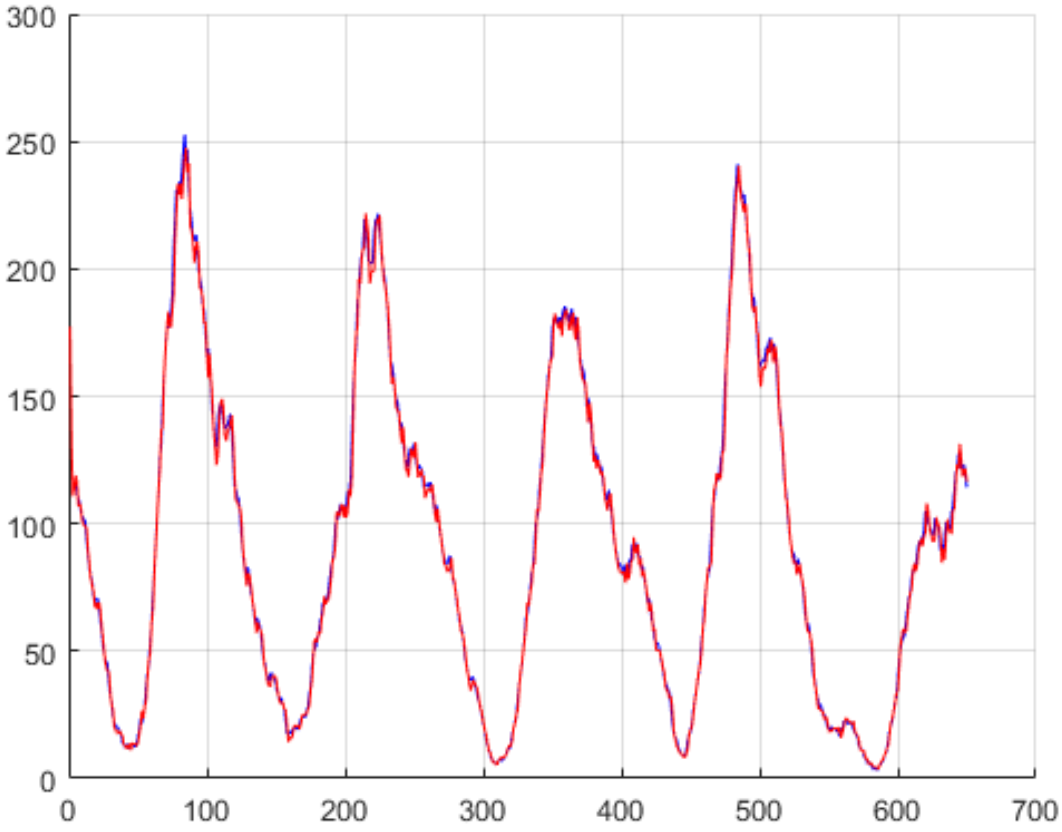
Обучающее множество:

R^2: 0.997290
MSE: 10.968004
RMSE: 3.311798
Относительная СКО: 1.327858%
MAE: 2.500114
min abs err: 0.000000
max abs err: 14.792498
MAPE: 4.395099
Доля с ошибкой менее 5%: 74.615385%
Доля с ошибкой от 5% до 10%: 17.846154%
Доля с ошибкой от 10% до 20%: 4.153846%
Доля с ошибкой от 20% до 30%: 2.461538%
Доля с ошибкой более 30%: 0.923077%

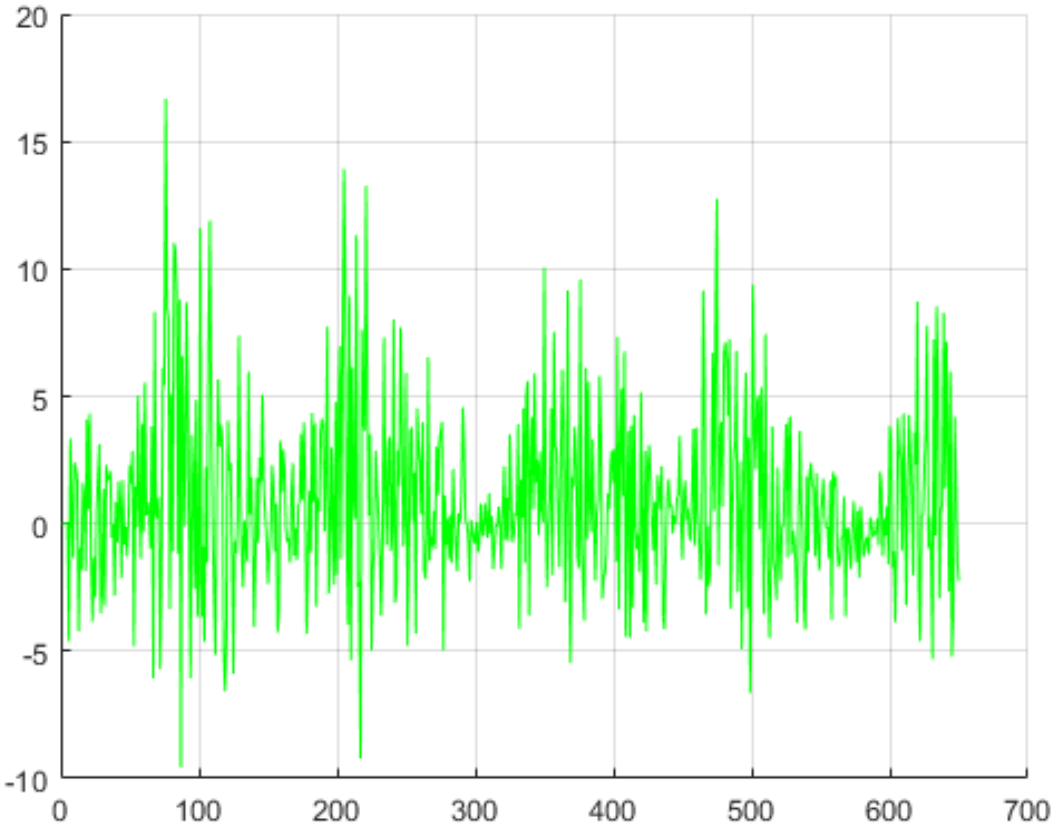
Тестовое множество:

R^2: 0.966939
MSE: 13.646162
RMSE: 3.694071
Относительная СКО: 4.535132%
MAE: 3.088962
min abs err: 0.050764
max abs err: 7.440216
MAPE: 3.416610
Доля с ошибкой менее 5%: 76.000000%
Доля с ошибкой от 5% до 10%: 24.000000%
Доля с ошибкой от 10% до 20%: 0.000000%
Доля с ошибкой от 20% до 30%: 0.000000%
Доля с ошибкой более 30%: 0.000000%

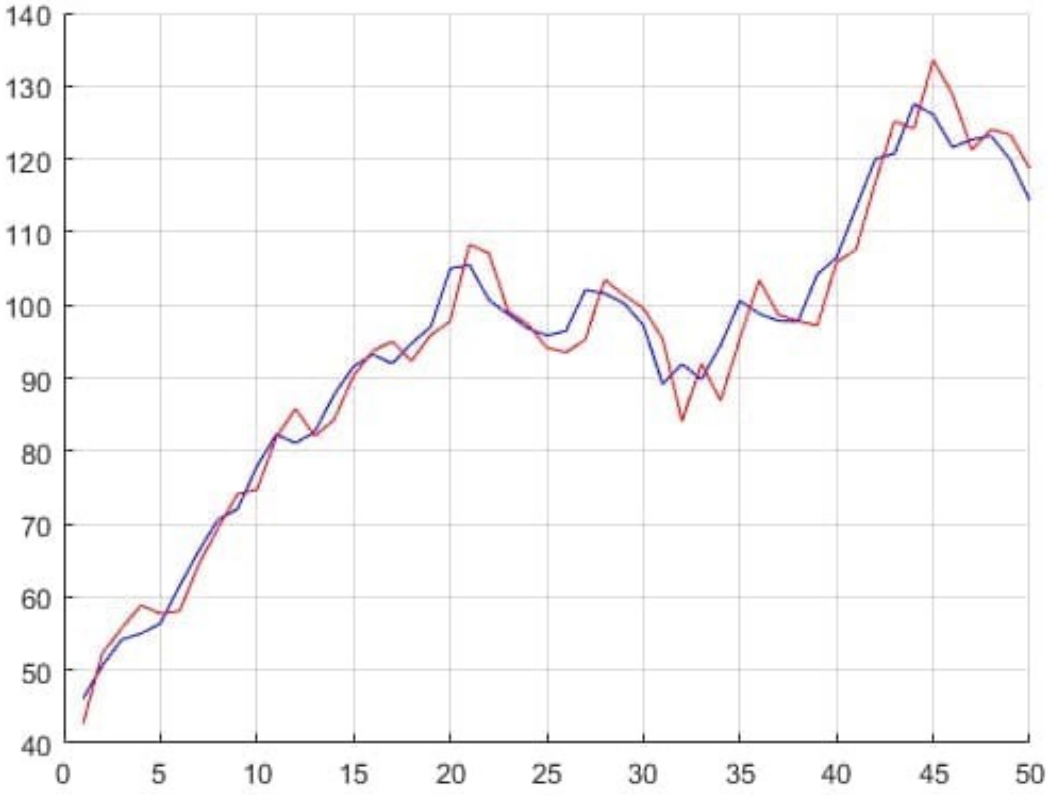
Эталонные и предсказанные значения на обучающем множестве



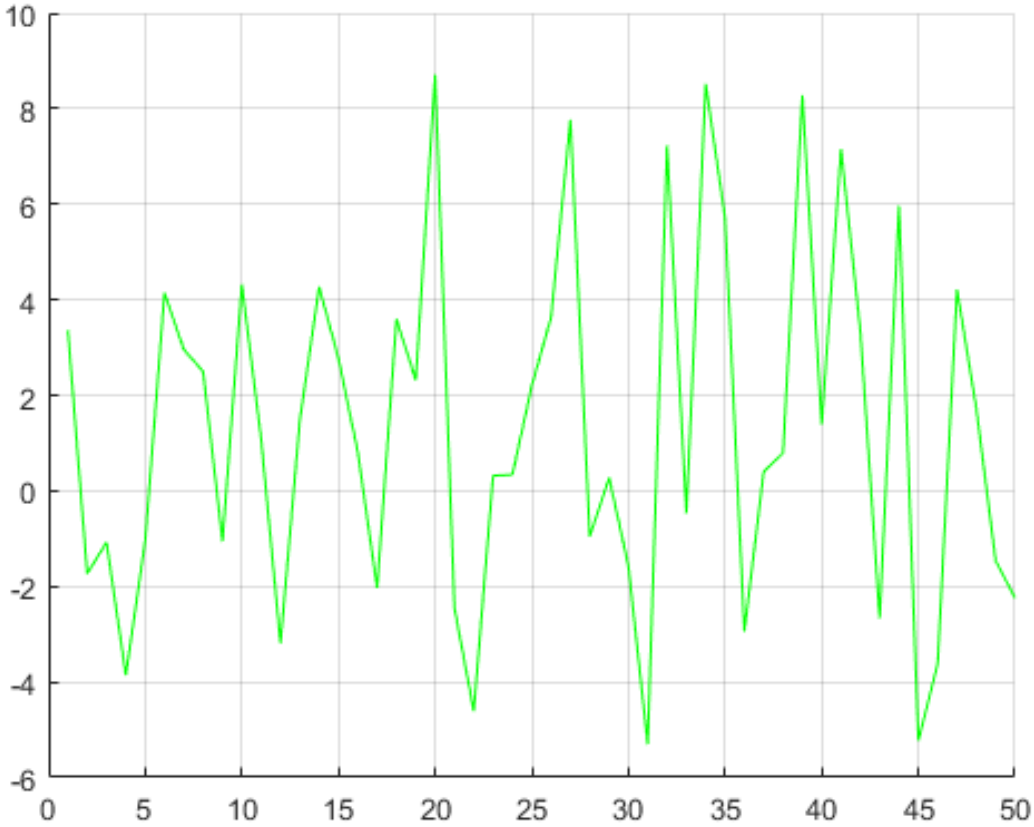
Ошибка на обучающем множестве



Эталонные и предсказанные значения на тестовом множестве

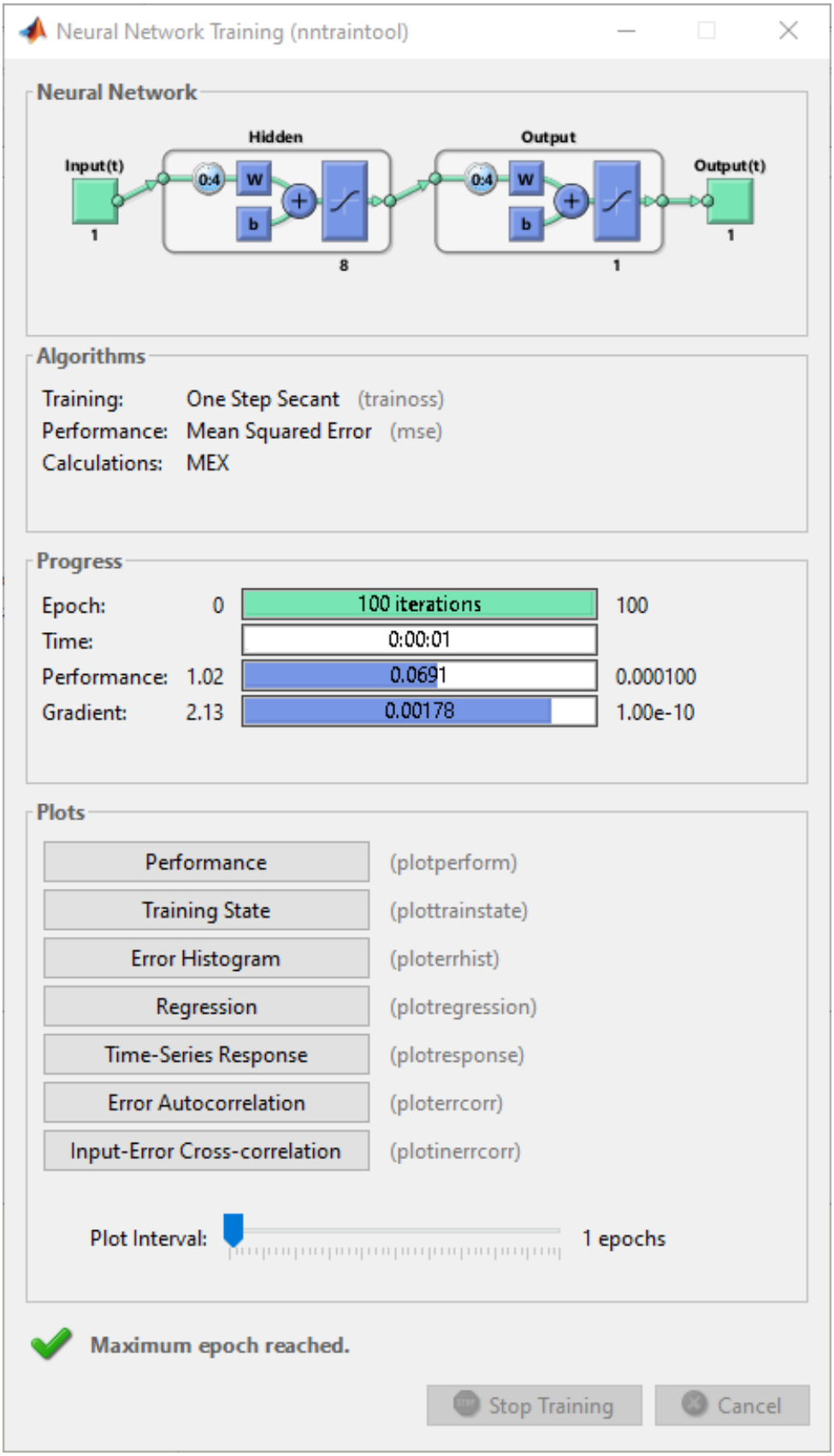


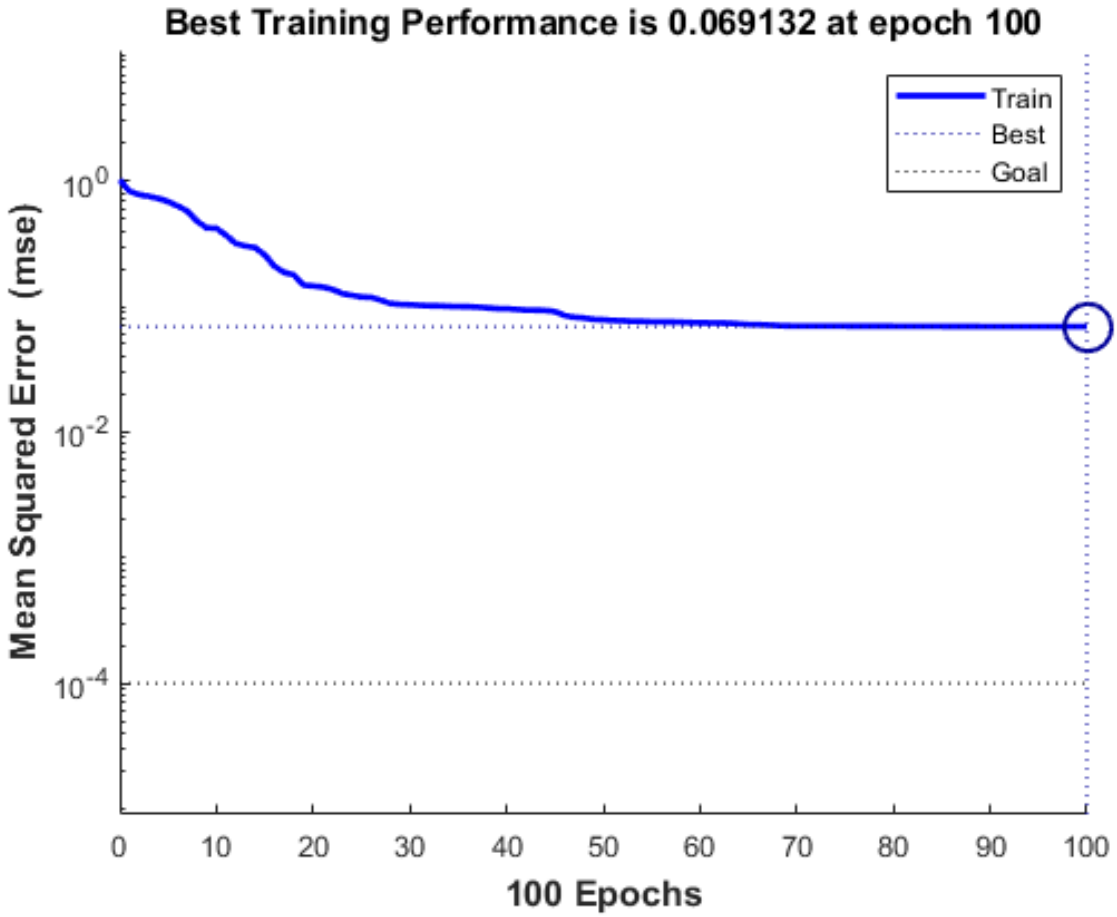
Ошибка на тестовом множестве



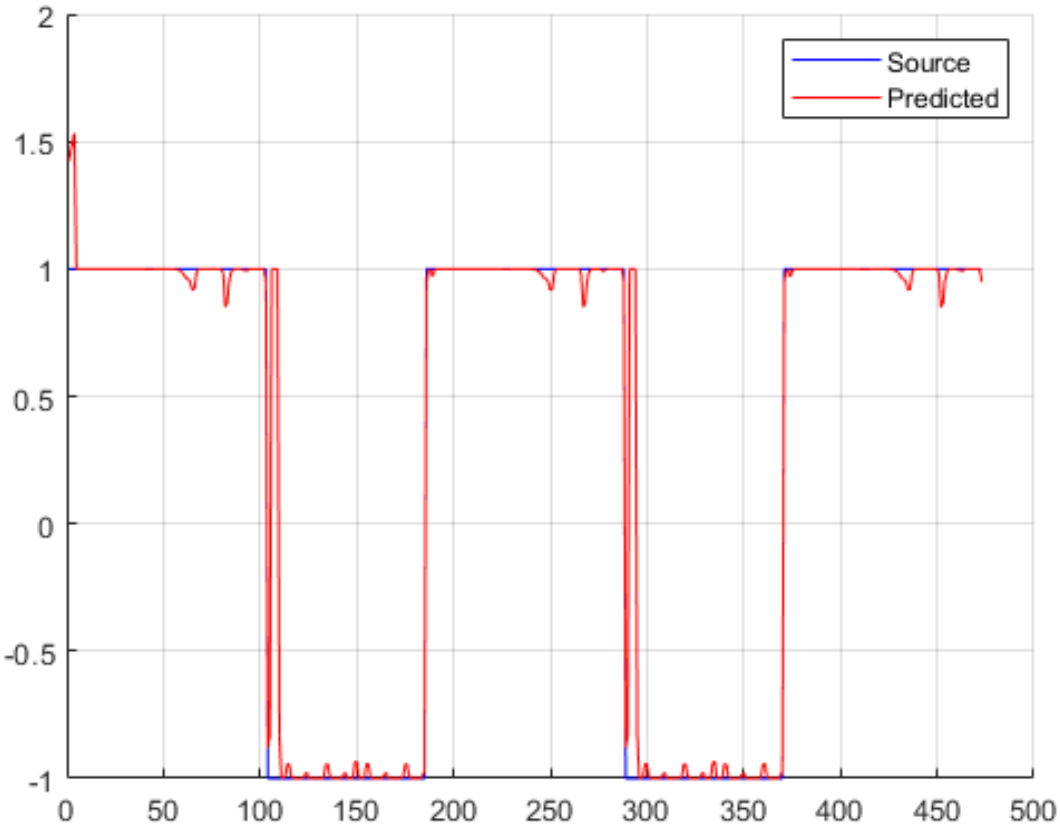
Задание 2.

Обучение сети:



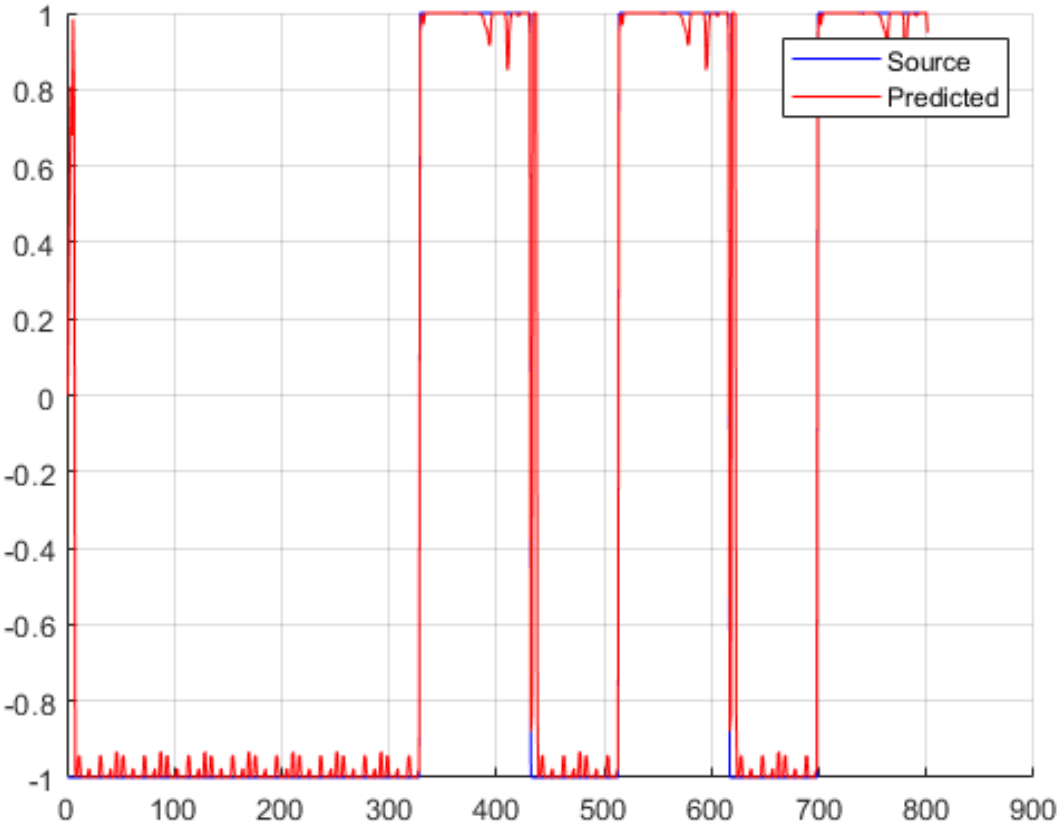

Best Training Performance is 0.069132 at epoch 100
Train
Best
Goal
Mean Squared Error (mse)
10^{0}
10^{-2}
10^{-4}
0
10
20
30
40
50
60
70
80
90
100
100 Epochs

Эталонные и предсказанные значения на обучающем множестве



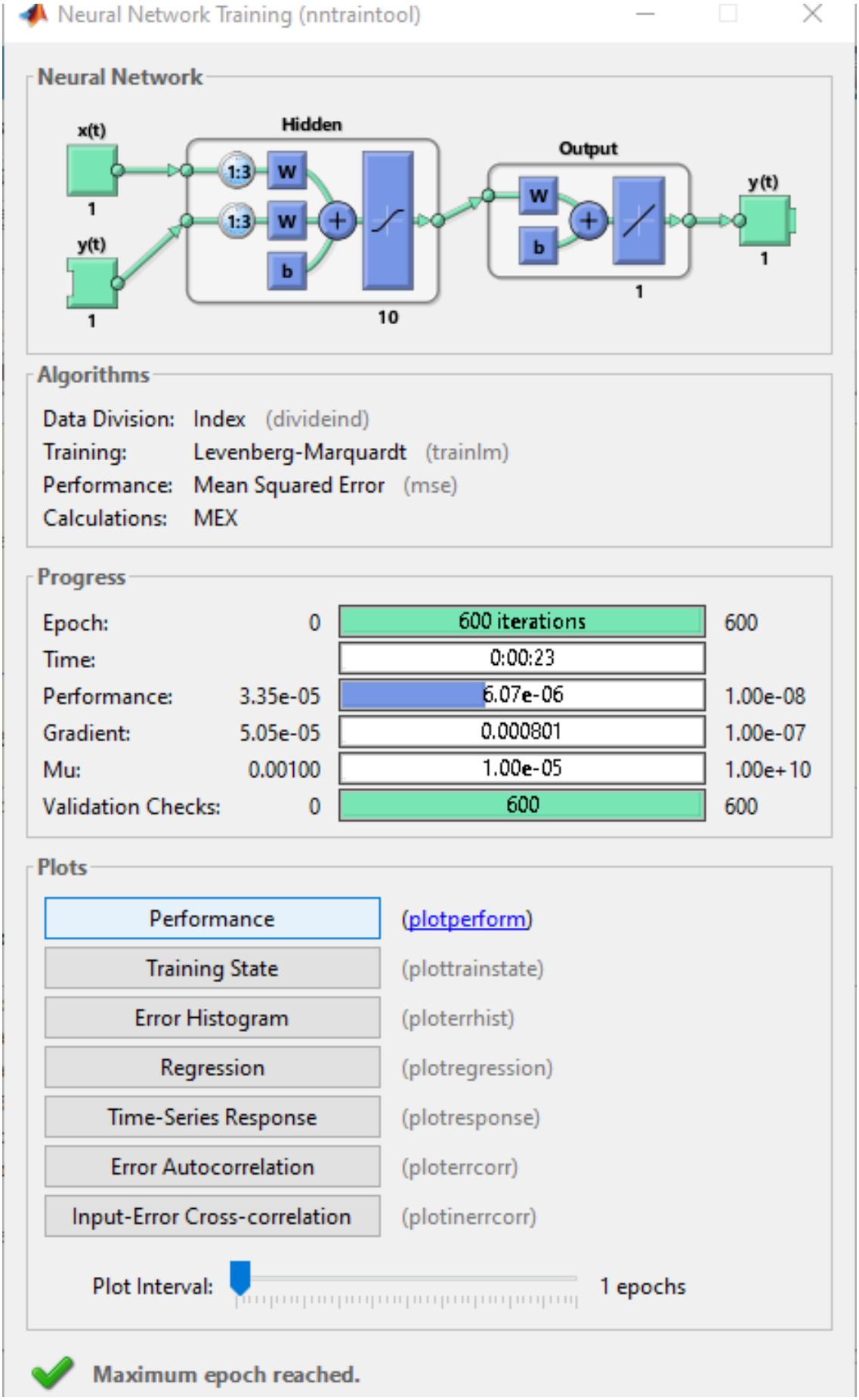
Эталонные и предсказанные значения на тестовом множестве

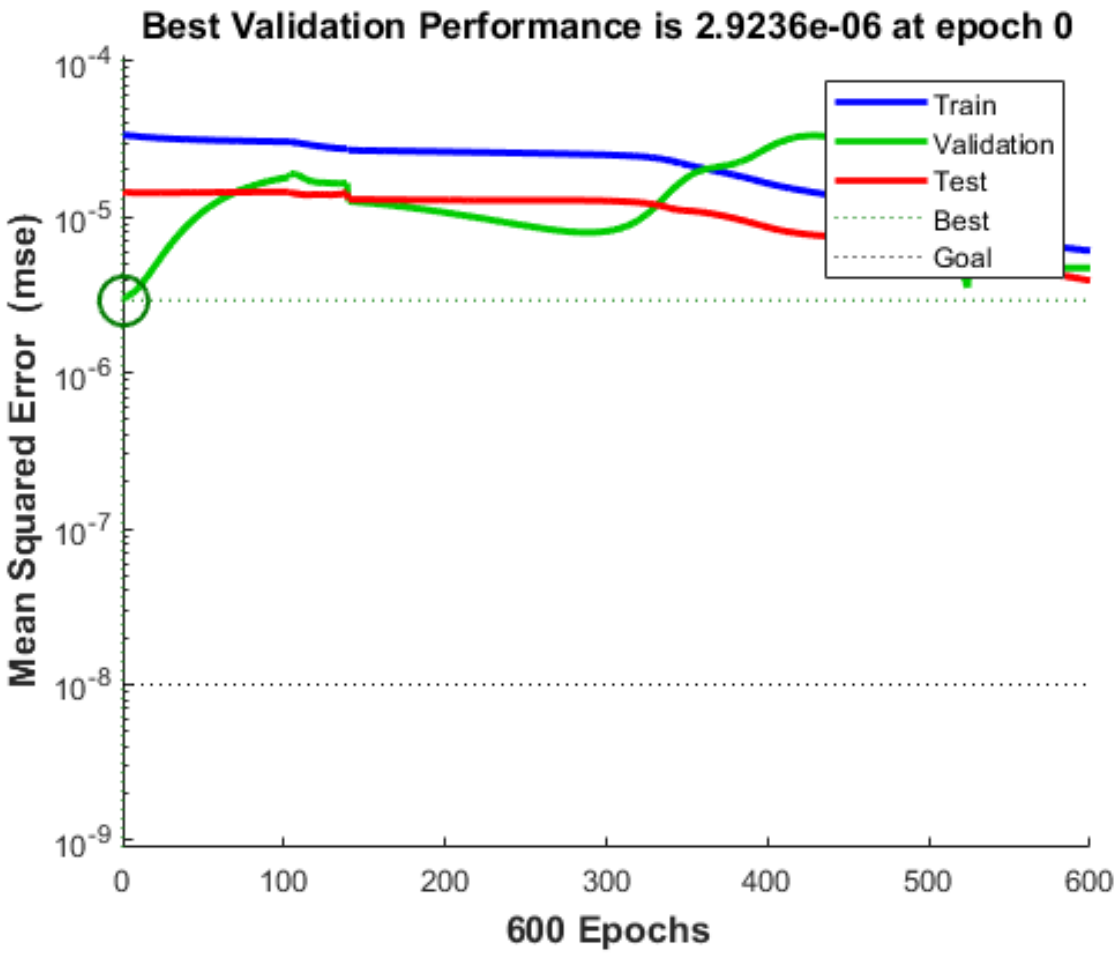
95.7717%



96.5044%

Задание 3.


Neural Network Training (nntraintool)
Neural Network
x(t)
1
y(t)
1
Hidden
1:3
W
1:3
W
b
10
Output
W
b
1
y(t)
1
Algorithms
Data Division: Index (divideind)
Training: Levenberg-Marquardt (trainlm)
Performance: Mean Squared Error (mse)
Calculations: MEX
Progress
Epoch: 0 600 iterations 600
Time: 0:00:23
Performance: 3.35e-05 6.07e-06 1.00e-08
Gradient: 5.05e-05 0.000801 1.00e-07
Mu: 0.00100 1.00e-05 1.00e+10
Validation Checks: 0 600 600
Plots
Performance (plotperform)
Training State (plottrainstate)
Error Histogram (ploterrhist)
Regression (plotregression)
Time-Series Response (plotresponse)
Error Autocorrelation (ploterrcorr)
Input-Error Cross-correlation (plotinerrcorr)
Plot Interval: 1 epochs
Maximum epoch reached.



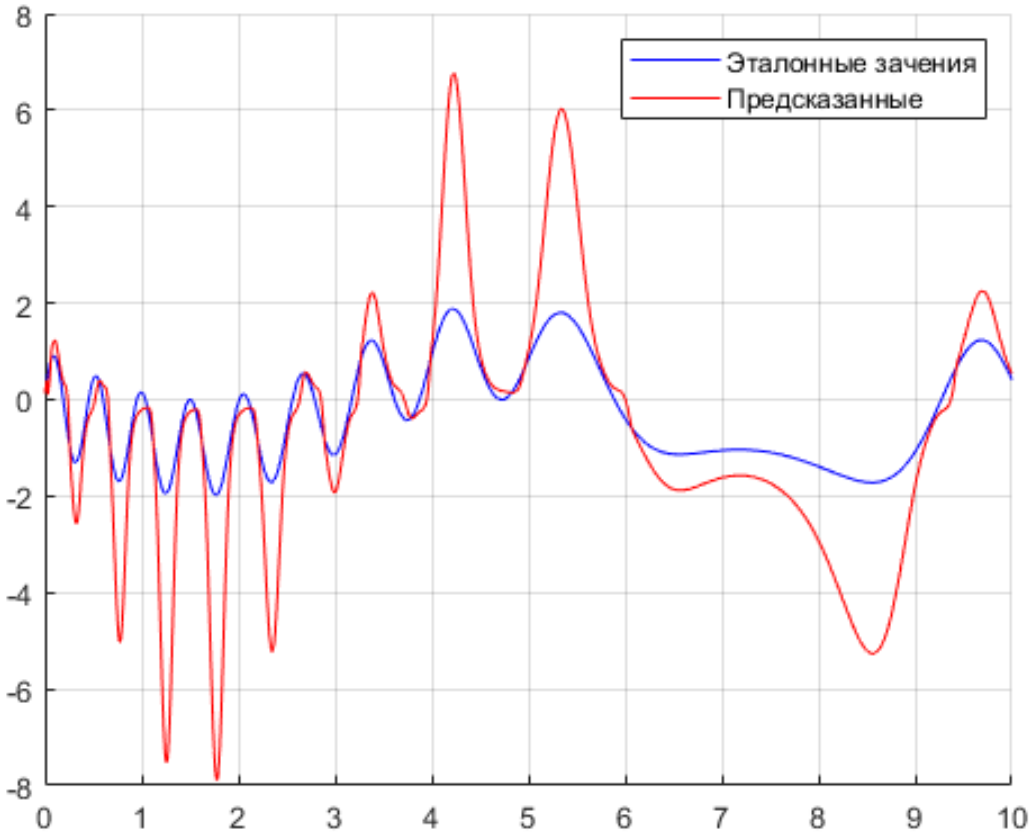
Обучающее множество:

R^2: -1.552743
MSE: 2.787986
RMSE: 1.669726
Относительная СКО: 43.277130%
MAE: 1.093578
min abs err: 0.000000
max abs err: 5.918433
MAPE: 442.354549
Доля с ошибкой менее 5%: 1.498501%
Доля с ошибкой от 5% до 10%: 1.298701%
Доля с ошибкой от 10% до 20%: 2.497502%
Доля с ошибкой от 20% до 30%: 6.393606%
Доля с ошибкой более 30%: 88.311688%

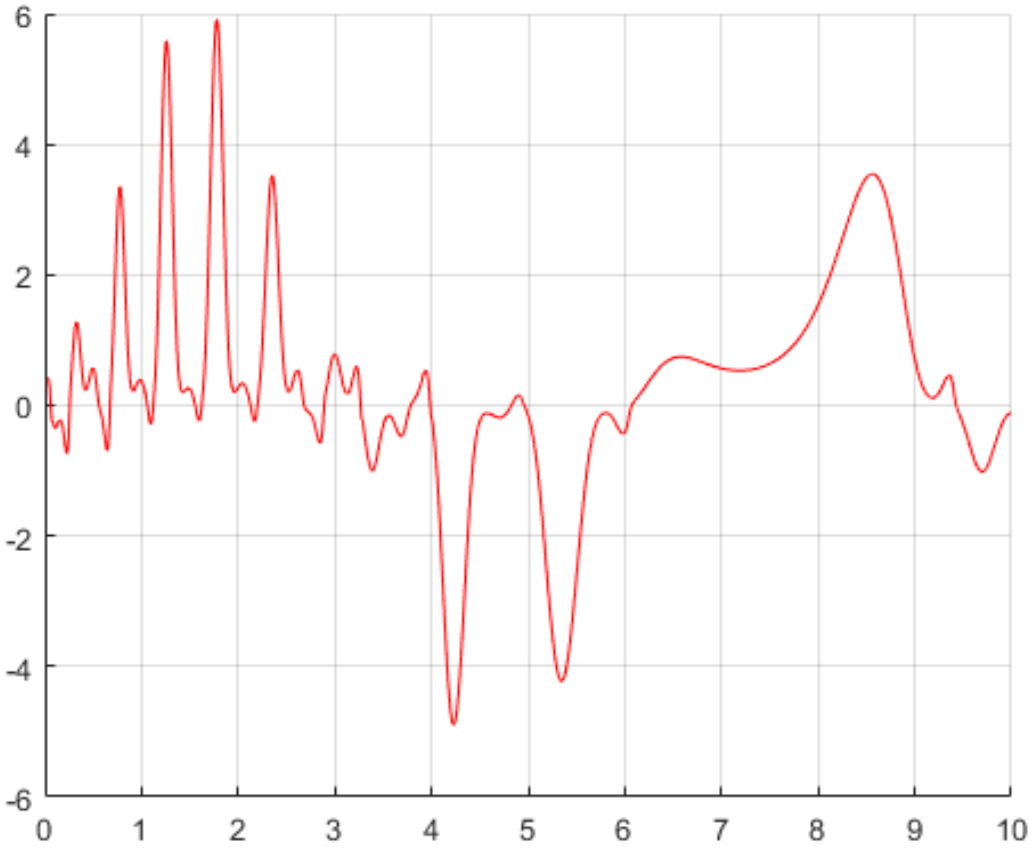
Тестовое множество:

R^2: 0.969641
MSE: 0.016341
RMSE: 0.127832
Относительная СКО: 5.520439%
MAE: 0.053563
min abs err: 0.000356
max abs err: 0.672342
MAPE: 9.681135
Доля с ошибкой менее 5%: 72.164948%
Доля с ошибкой от 5% до 10%: 9.278351%
Доля с ошибкой от 10% до 20%: 7.216495%
Доля с ошибкой от 20% до 30%: 2.061856%
Доля с ошибкой более 30%: 9.278351%

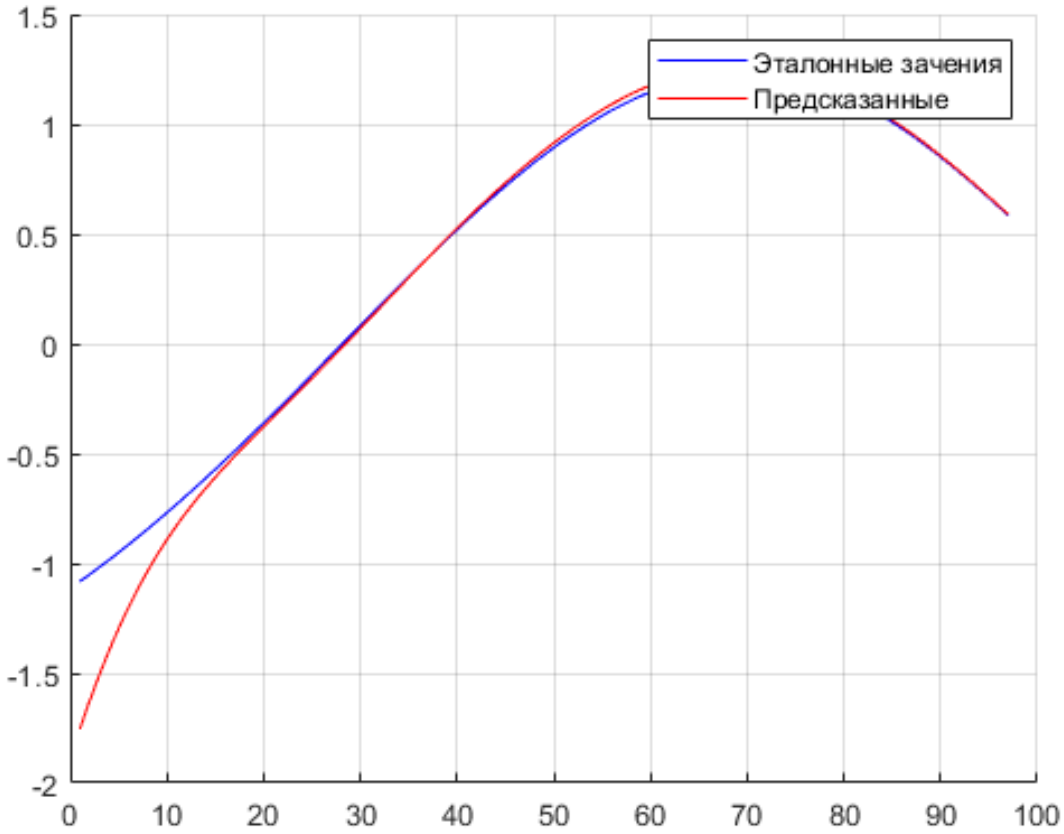
Эталонные и предсказанные значения на обучающем множестве



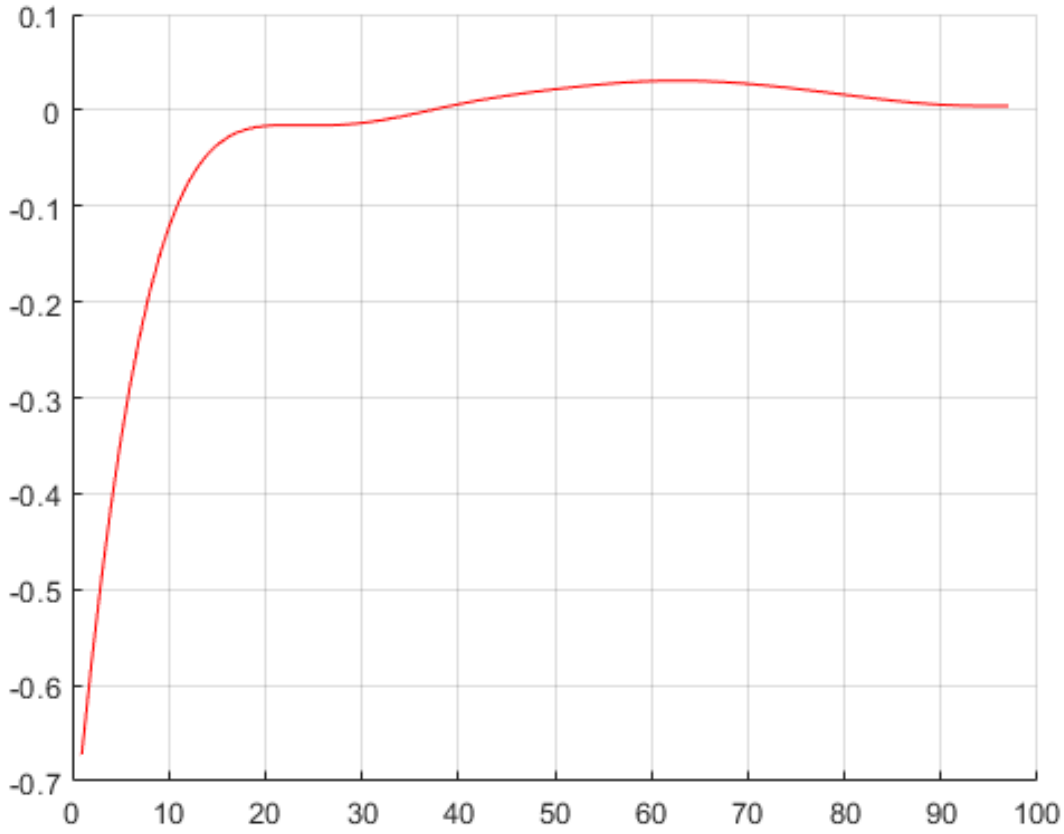
Ошибка на обучающем множестве



Эталонные и предсказанные значения на тестовом множестве



Ошибка на тестовом множестве



Код программы

Lab8.m

```
% Задание 1
% Выгружаем данные из файла
data = load('100years.txt');
nums = data(:, 4);
% Сглаживание траектории усредяющим фильтром с шириной окна 12
x = smooth(nums, 12);
% Получаем матрицу строк
x = con2seq(x(1:650)');

% Создаем сеть

D = 5;
network = timedelaynet(1:D, 8, 'trainlm');
network.layers{1}.transferFcn = 'tansig';
network.layers{2}.transferFcn = 'purelin';

% Разбиваем данные на подмножества
network.divideFcn = 'divideind';
network.divideParam.trainInd = 1 : 500;
network.divideParam.valInd = 501 : 600;
network.divideParam.testInd = 601 : 650;

% Конфигурируем сеть и инициализируем весовые коэффициенты
network = configure(network, x, x);
network = init(network);
view(network);

%% Обучаем ее
network.trainParam.epochs = 600;
network.trainParam.max_fail = 600;
network.trainParam.goal = 10e-5;

%%Подготовка входных и целевых временных серий данных для моделирования или обучения
% Xs Сдвинутые входы
% Xi Начальные состояния задержки ввода
% Ai Состояния задержки начального слоя
% Ts Сдвинутые цели
[Xs, Xi, Ai, Ts] = preparets(network, x, x);

% Получаем значения предсказанные сетью
network = train(network, Xs, Ts, Xi, Ai);
Y = sim(network, Xs, Xi);

%% Эталонное и предсказанное значение на обучающей выборке
figure;
hold on;
grid on;
plot(cell2mat(x), '-b');
plot([cell2mat(Xi), cell2mat(Y)], '-r');
dataForTable(cell2mat(x), [cell2mat(Xi), cell2mat(Y)])
```

```
%% Ошибка на обучающей выборке
figure;
hold on;
grid on;
error = cell2mat(x)- [cell2mat(Xi), cell2mat(Y)];
plot(error, '-g');
%% Эталонное и предсказанное значение на тестовой выборке
xCheck = cell2mat(x);
yCheck = cell2mat(Y);

figure;
hold on;
grid on;
plot(xCheck(601 : 650), '-b');
plot(yCheck(601 - D : 650 - D), '-r');
dataForTable(xCheck(601 : 650), yCheck(601 - D : 650 - D))

%% Ошибка на тестовой выборке
figure;
hold on;
grid on;
error = xCheck(601 : 650)- yCheck(601 - D : 650 - D);
plot(error, '-g');

%% Задание 2
% Генерируем обучающее множесво
k = 0 : 0.025 : 1;
p1 = sin(4 * pi * k);
t1 = -ones(size(k));

k = 0.46 : 0.025 : 3.01;
g = @(k) cos( -3 * k .^ 2 + 5 * k + 10) + 0.8;
p2 = g(k);
t2 = ones(size(k));

R = {0; 2; 2};
P = [repmat(p1, 1, R{1}), p2, repmat(p1, 1, R{2}), p2, repmat(p1, 1, R{3}), p2];
T = [repmat(t1, 1, R{1}), t2, repmat(t1, 1, R{2}), t2, repmat(t1, 1, R{3}), t2];

PPrepared = con2seq(P);
TPrepared = con2seq(T);

% Создаем сеть
network = distdelaynet({0 : 4, 0 : 4}, 8, 'trainoss');
network.layers{1}.transferFcn = 'tansig';
network.layers{2}.transferFcn = 'tansig';
network.divideFcn = '';
network = configure(network, PPrepared, TPrepared);
view(network);

%% Обучаем ее
network.trainParam.epochs = 100;
network.trainParam.goal = 10e-5;
```

```
[Xs, Xi, Ai, Ts] = preparets(network, PPrepared, TPrepared);
network = train(network, Xs, Ts, Xi, Ai);

%% Рассчитываем выход сети и отображаем результаты на графике
Y = sim(network, Xs, Xi);
figure;
hold on;
grid on;
plot(cell2mat(TPrepared), '-b');
plot([cell2mat(Xi) cell2mat(Y)], '-r');
legend('Source', 'Predicted');

%% Преобразовываем значения
Yc = zeros(1, numel(Xi) + numel(Y));
for i = 1 : numel(Xi)
    if Xi{i} >= 0
        Yc(i) = 1;
    else
        Yc(i) = -1;
    end
end
for i = numel(Xi) + 1 : numel(Y)
    if Y{i} >= 0
        Yc(i) = 1;
    else
        Yc(i) = -1;
    end
end

% Сравнивам полученные значения с эталонными
disp((nnz(Yc == cell2mat(TPrepared)) / length(TPrepared) * 100));

R = {8; 2; 2};
P = [repmat(p1, 1, R{1}), p2, repmat(p1, 1, R{2}), p2, repmat(p1, 1, R{3}), p2];
T = [repmat(t1, 1, R{1}), t2, repmat(t1, 1, R{2}), t2, repmat(t1, 1, R{3}), t2];

PPrepared = con2seq(P);
TPrepared = con2seq(T);

[Xs, Xi, Ai, Ts] = preparets(network, PPrepared, TPrepared);

Y = sim(network, Xs, Xi);

figure;
hold on;
grid on;
plot(cell2mat(TPrepared), '-b');
plot([cell2mat(Xi) cell2mat(Y)], '-r');
legend('Source', 'Predicted');

Yc = zeros(1, numel(Xi) + numel(Y));
for i = 1 : numel(Xi)
    if Xi{i} >= 0
        Yc(i) = 1;
    else
```

```
        Yc(i) = -1;
    end
end
for i = numel(Xi) + 1 : numel(Y)
    if Y{i} >= 0
        Yc(i) = 1;
    else
        Yc(i) = -1;
    end
end

disp( nnz(Yc == cell2mat(TPrepared)) / length(TPrepared) * 100 );

%% Задание 3
% Задаем функции
uFunction = @(k) sin(k.^2 - 15 .* k + 3) - sin(k);
yNextFunction = @(y, u) y ./ (1 + y.^2) + u.^3;

% Инициализируем константы
tBegin = 0;
tEnd = 10;
h = 0.01;
n = 1 + (tEnd - tBegin) / h;
u = zeros(1, n);

% Заполняем массивы
y = zeros(1, n);
u(1) = uFunction(0);
for i = 2 : n
    t = tBegin + (i - 1) * h;
    y(i) = yNextFunction(y(i - 1), u(i - 1));
    u(i) = uFunction(t);
end

% Задаем глубину погружения и выделяем подмножества
x = u;
D = 3;
trainInd = 1 : 700;
valInd = 701 : 900;
testInd = 901 : 997;

% Создаем сеть
network = narxnet(1 : 3, 1 : 3, 10);
network.trainFcn = 'trainlm';
network.layers{1}.transferFcn = 'tansig';
network.layers{2}.transferFcn = 'purelin';
network.divideFcn = 'divideind';
network.divideParam.trainInd = trainInd;
network.divideParam.valInd = valInd;
network.divideParam.testInd = testInd;
network = init(network);
view(network);

%% Обучаем ее
```

```
network.trainParam.epochs = 600;
network.trainParam.max_fail = 600;
network.trainParam.goal = 1.0e-8;
[Xs, Xi, Ai, Ts] = preparets(network, con2seq(x), {}, con2seq(y));
network = train(network, Xs, Ts, Xi, Ai);

%% Вычисляем выход сети
Y = sim(network, Xs, Xi, Ai);

% Эталонные и предсказанные занчения на обучающем множестве
figure;
hold on;
grid on;
plot(tBegin : h : tEnd, x, '-b', tBegin : h : tEnd, [x(1:D) cell2mat(Y)], '-r')
legend('Эталонные зачения', 'Предсказанные');
dataForTable(x, [x(1:D) cell2mat(Y)])

%% Ошибка на обучающем множестве
figure;
hold on;
grid on;
plot(tBegin+D*h : h : tEnd, x(D+1:end) - cell2mat(Y), '-r')

%% Вычисляем пошаговый прогноз на тестовом множестве
xValid = x( valInd( length(valInd) - (D - 1) : length(valInd) ) );
uValid = u( valInd( length(valInd) - (D - 1) : length(valInd) ) );
inpuT   = [con2seq( u(testInd)  ); con2seq( x(testInd) )];
delay = [con2seq( uValid ); con2seq(xValid)];
predictTest = sim(network, inpuT, delay, Ai);

% Эталонные и предсказанные занчения на тестовом множестве
figure;
hold on;
grid on;
plot(x(testInd), '-b')
plot(cell2mat(predictTest), '-r');
legend('Эталонные зачения', 'Предсказанные');
dataForTable(x(testInd), cell2mat(predictTest))

%% Ошибка на тестовом множестве
figure;
hold on;
grid on;
error = cell2mat(predictTest) - x(testInd);
plot(error, '-r');
```

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;
    Under5PersentPortion = sum(errors < 5) / length(y) * 100;
    Under10PersentPortion = sum(5 <= errors & errors < 10) / length(y) * 100;
    Under20PersentPortion = 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 СКО: %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
```

Вывод

В этой лабораторной работе я научился использовать динамические сети для многошагового прогноза временного ряда, распознавания динамических образов и аппроксимации траектории динамической системы. Особенно интересным для меня оказалось задание по аппроксимации динамической системы, так как сеть на обучающей выборке показывала отвратительные результаты, в не зависимости от количества прогонов обучающей функции, но на тестовой выборке результаты сильно улучшились, что свидетельствует о том, что даже если на обучающей выборке результат не заметен, это не значит, что весовые коэффициенты в результате такого обучения могут оказаться значительно точнее на другой выборке.