Московский авиационный институт
(Национальный исследовательский университет)
Факультет прикладной математики и физики
Кафедра вычислительной математики и программирования

Лабораторная работа № 3

по курсу «Нейроинформатика»

Тема: Многослойные сети. Алгоритм обратного распространения ошибки.

Студент: Ваньков Д. А.
Группа: 8О-407Б-17
Преподаватель: Аносова Н.П.

Москва, 2021

Постановка задачи

Исследование свойств многослойной нейронной сети прямого распространения и алгоритмов её обучения, применения сетей в задачах классификации и аппроксимации функции.

1. Использовать многослойную нейронную сеть для классификации в случае, когда классы не являются линейно разделимыми.
2. Использовать многослойную нейронную сеть для аппроксимации функции. Произвести обучение при помощи одного из методов первого порядка.
3. Использовать многослойную нейронную сеть для аппроксимации функции. Произвести обучение при помощи одного из методов второго порядка.

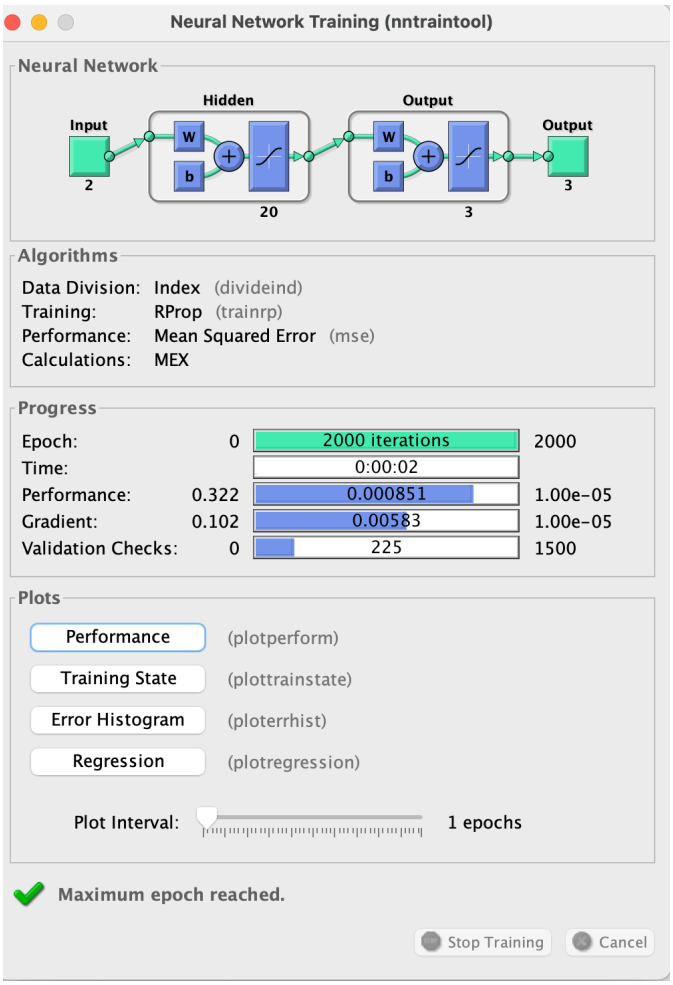
Вариант 6

6.	Эллипс: $a = 0.4$, $b = 0.15$, $\alpha = \pi/6$, $x_0 = 0.1$, $y_0 = -0.15$ Эллипс: $a = 0.7$, $b = 0.5$, $\alpha = -\pi/3$, $x_0 = 0$, $y_0 = 0$ Эллипс: $a = 1$, $b = 1$, $\alpha = 0$, $x_0 = 0$, $y_0 = 0$	$x = \sin\left(0.5t^2 - 5t\right), \quad t \in [0, 2]$, $h = 0.01$	*traincgp, trainlm*

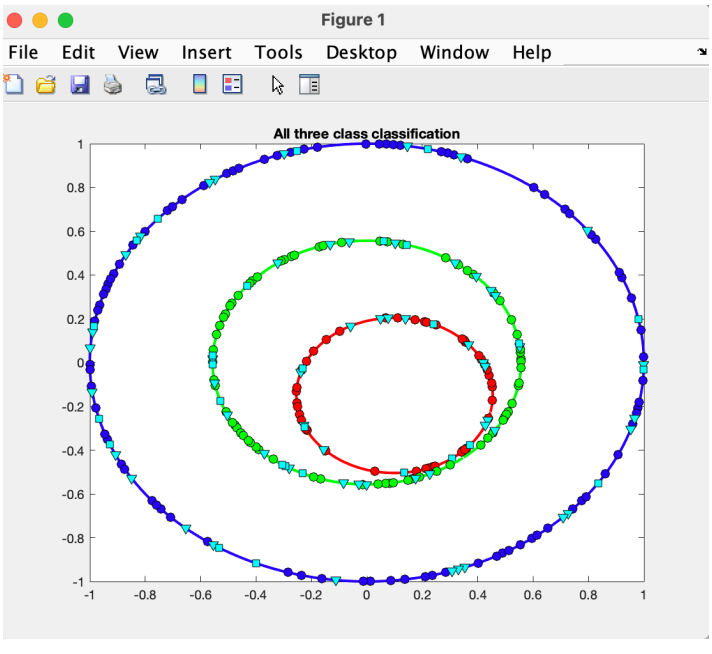
Ход работы

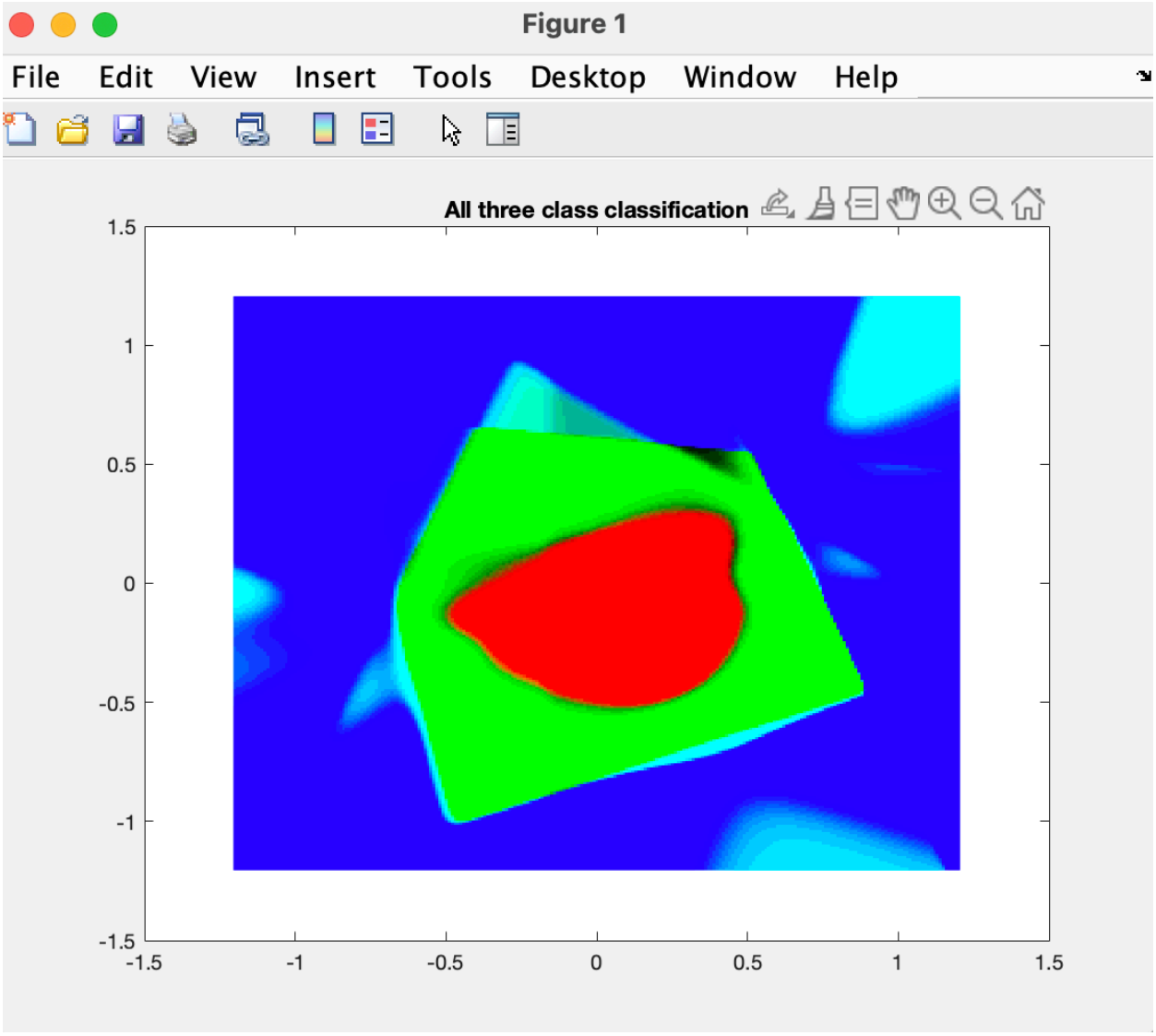
1.

Структура сети



Исходные множества





Результат работы

Training size: 196

Matches: 196

Target size: 56

Matches: 56

Test size: 28

Matches: 26

2.

Структура сети

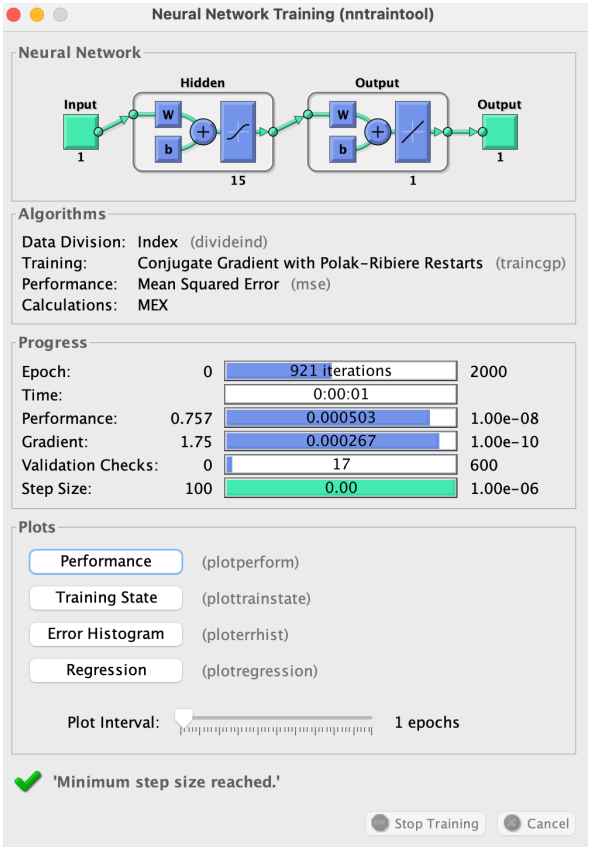


График функции ошибки

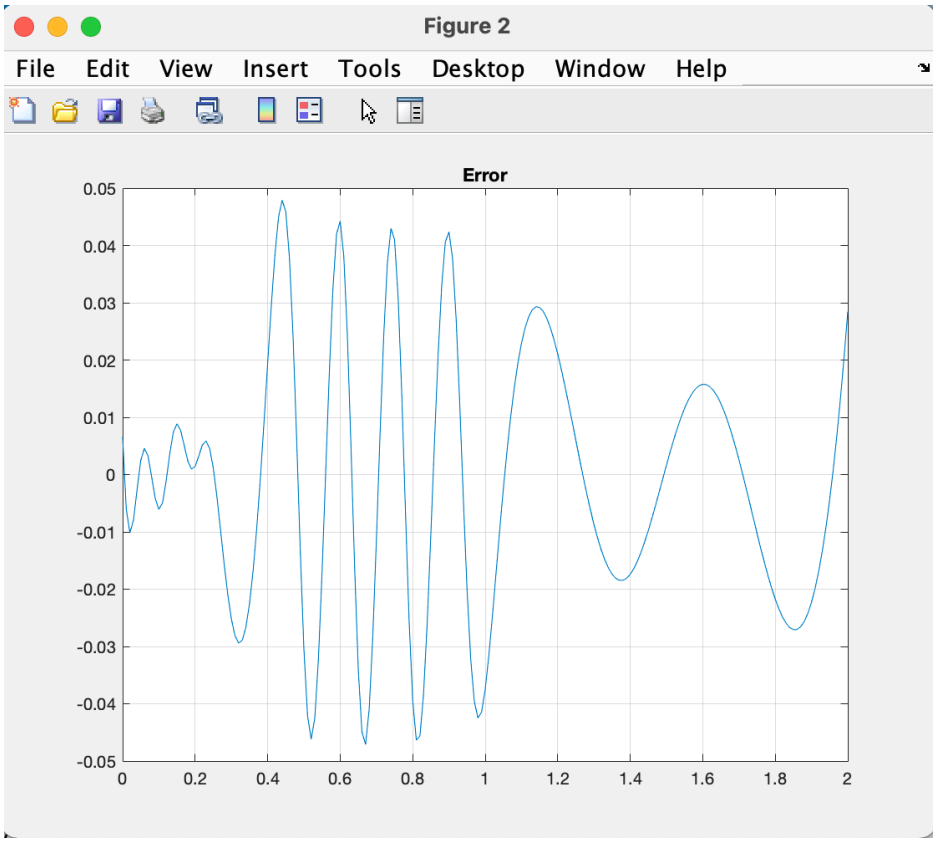
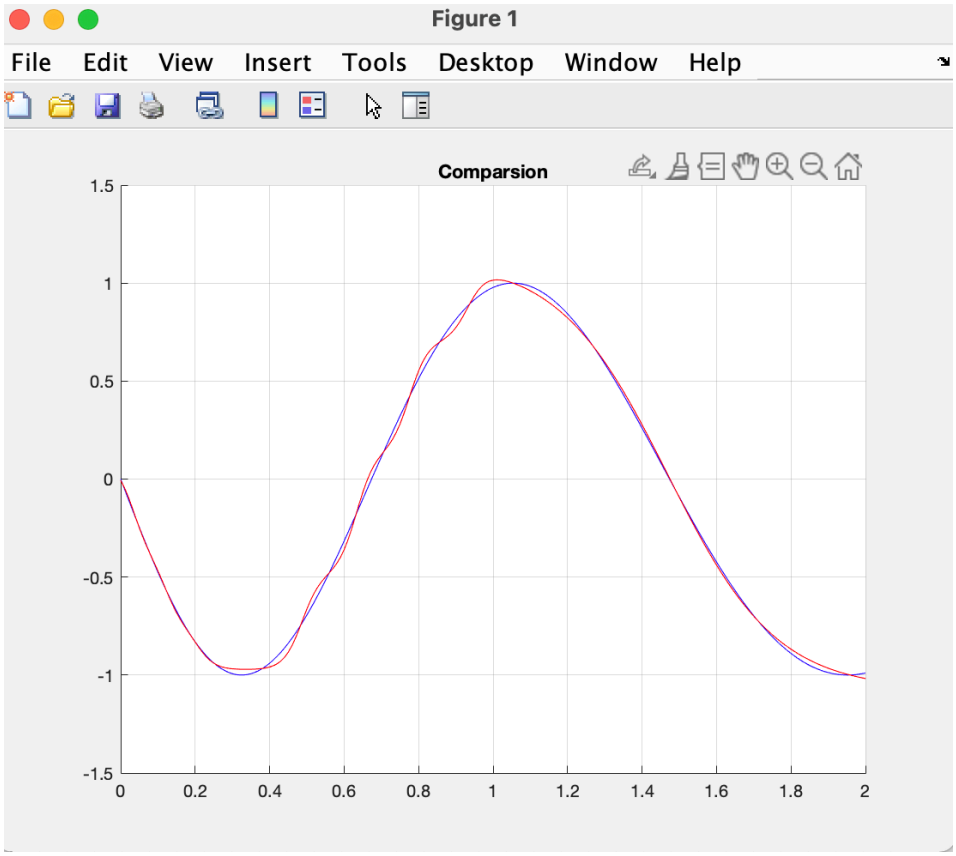
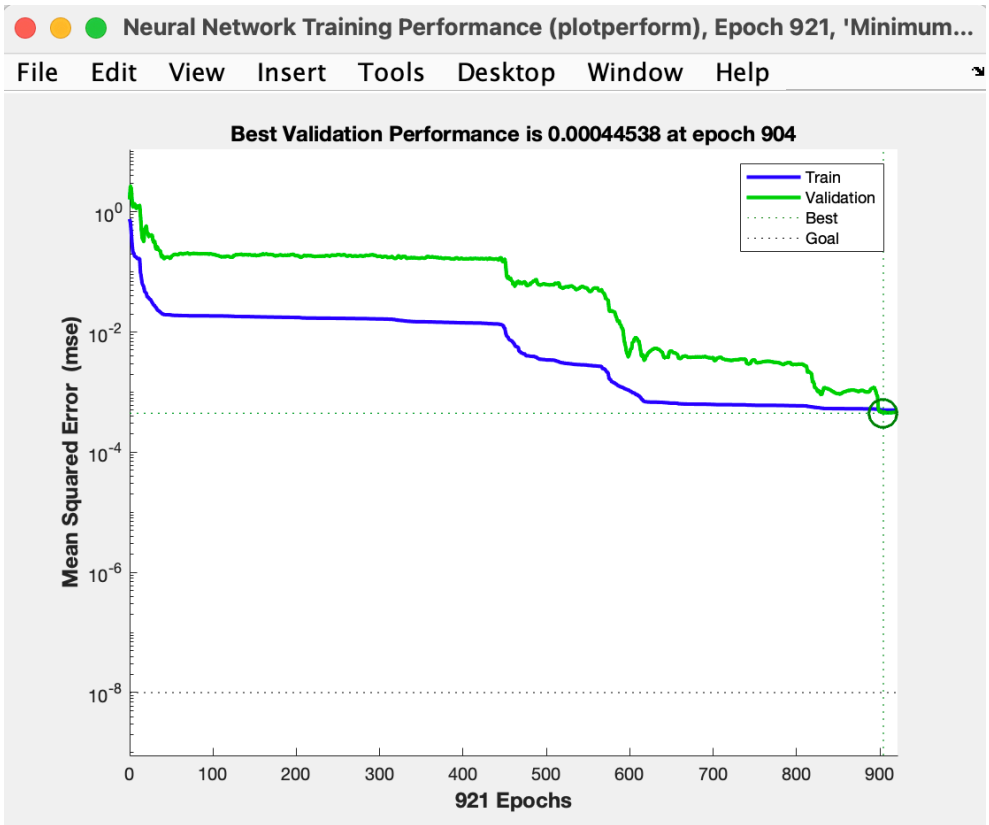


График сравнение эталонных значений с предсказанными



Performance



3.

Структура сети

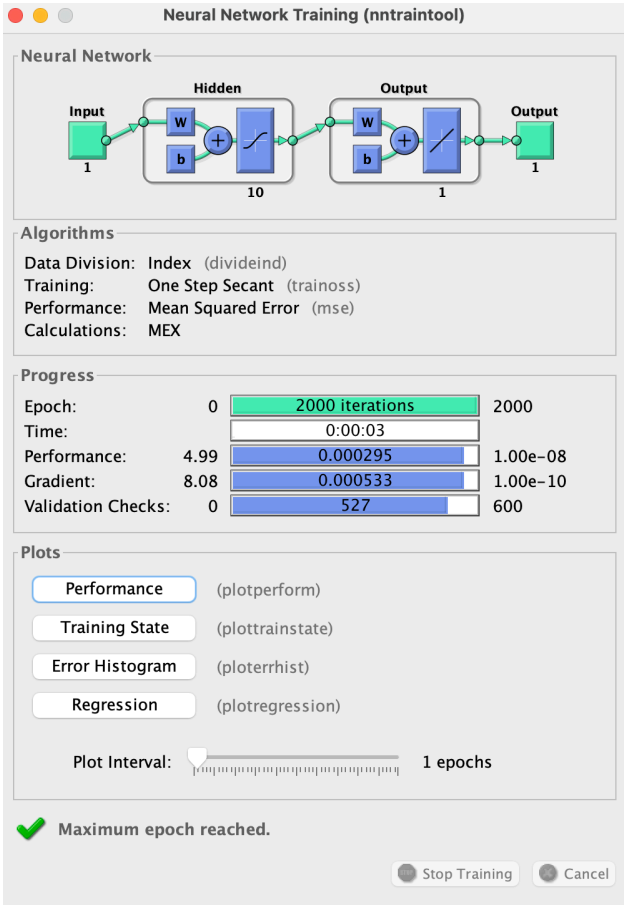


График функции ошибки

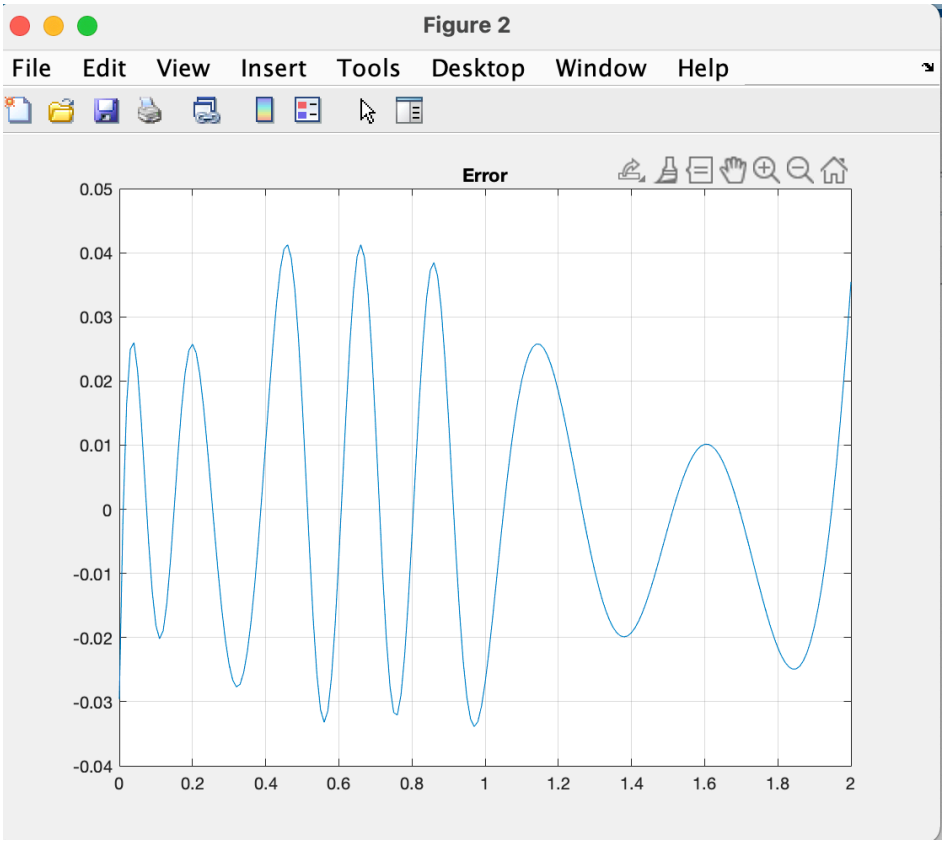
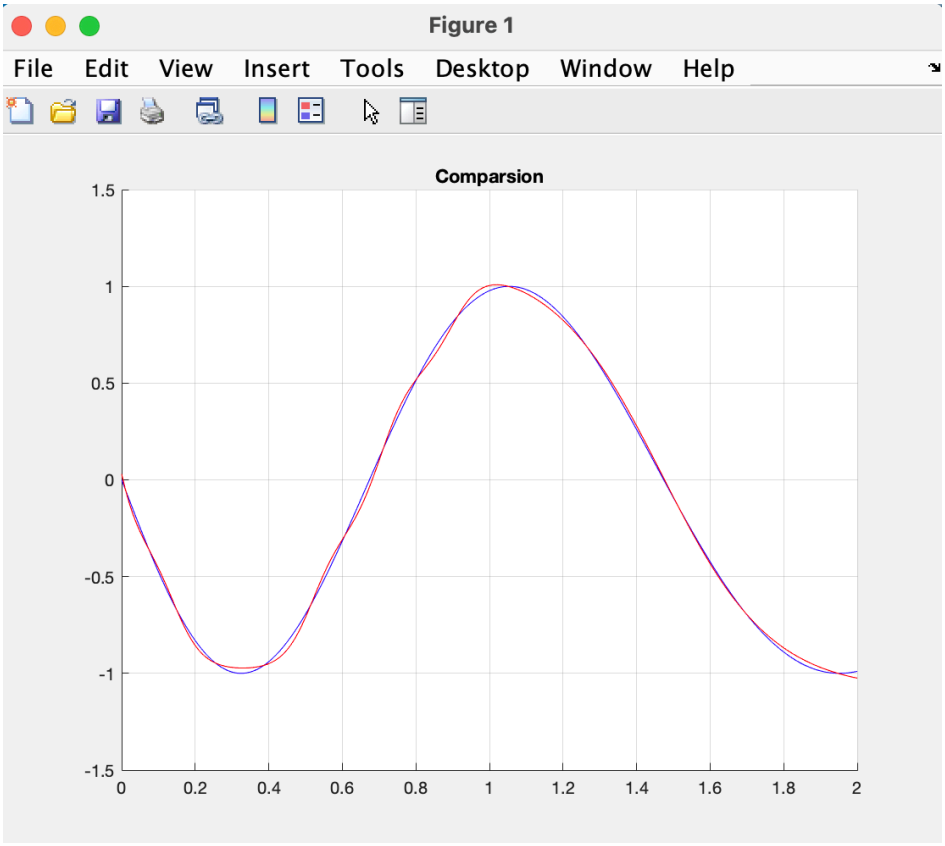
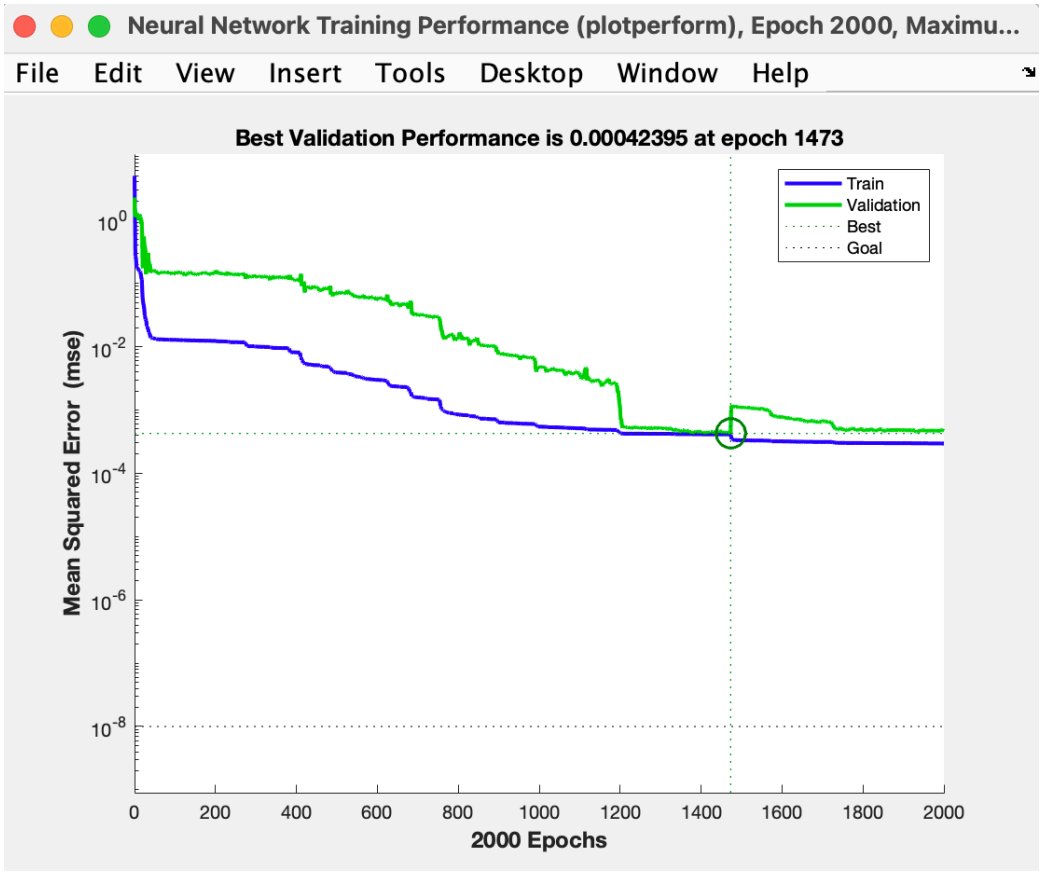


График сравнение эталонных значений с предсказанными



Performance



Исходный код

```
% 1.1
% Генерация точек из варианта
% Эллипс:a=0.4, b=0.15, α=π/6, x0=0.1, y0=-0.15
% Эллипс:a=0.7, b=0.5, α=-π/3, x0=0, y0=0
% Эллипс:a=1, b=1, α=0, x0=0, y0=0
[X1, Y1] = arangePoints(0.4, 0.15, 0.1, -0.15, -pi/6, 0.025);
[X2, Y2] = arangePoints(0.7, 0.5, 0, 0, -pi/3, 0.025);
[X3, Y3] = arangePoints(1, 1, 0, 0, 0, 0.025);

n = length(X1);

% Генарция точек для первого класса
D1 = randperm(n);
n1 = 60;
D1 = D1(1:n1);
display(D1);
K1 = [ones(1, n1); 0 * ones(1, n1); 0 * ones(1, n1)];

% Генарция точек для вторго класса
D2 = randperm(n);
n2 = 100;
D2 = D2(1:n2);
display(D2);
K2 = [0 * ones(1, n2); ones(1, n2); 0 * ones(1, n2)];

% Генарция точек для третьего класса
D3 = randperm(n);
n3 = 120;
D3 = D3(1:120);
display(D3);
K3 = [0 * ones(1, n3); 0 * ones(1, n3); ones(1, n3)];

% 1.2
% test train split Разделение на обучающее и контрольное
[trainInd1, valInd1, testInd1] = dividerand(length(D1), 0.7, 0.2, 0.1);
[trainInd2, valInd2, testInd2] = dividerand(length(D2), 0.7, 0.2, 0.1);
[trainInd3, valInd3, testInd3] = dividerand(length(D3), 0.7, 0.2, 0.1);

% 1.3
figure
```

```
% Резудлтаты для первого класса
class1 = plot(X1, Y1, '-r', ...
    'LineWidth', 2);

hold on;

tr1 = plot(X1(D1(trainInd1)), Y1(D1(trainInd1)), 'or', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'r', ...
    'MarkerSize', 7);

hold on;

val1 = plot(X1(D1(valInd1)), Y1(D1(valInd1)), 'rV', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'c', ...
    'MarkerSize', 7);

hold on;

test1 = plot(X1(D1(testInd1)), Y1(D1(testInd1)), 'rs', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'c', ...
    'MarkerSize', 7);
hold on;

% Результаты для первого класса
class2 = plot(X2, Y2, '-g', ...
    'LineWidth', 2);

hold on;

tr2 = plot(X2(D2(trainInd2)), Y2(D2(trainInd2)), 'og', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'g', ...
    'MarkerSize', 7);

hold on;

val2 = plot(X2(D2(valInd2)), Y2(D2(valInd2)), 'gV', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'c', ...
    'MarkerSize', 7);

hold on;
```

```
test2 = plot(X2(D2(testInd2)), Y2(D2(testInd2)), 'gs', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'c', ...
    'MarkerSize', 7);

hold on;

% Резудлтаты для третьего класса
class3 = plot(X3, Y3, '-b', ...
    'LineWidth', 2);

hold on;

tr3 = plot(X3(D3(trainInd3)), Y3(D3(trainInd3)), 'ob', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'b', ...
    'MarkerSize', 7);

hold on;

val3 = plot(X3(D3(valInd3)), Y3(D3(valInd3)), 'bV', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'c', ...
    'MarkerSize', 7);

hold on;

test3 = plot(X3(D3(testInd3)), Y3(D3(testInd3)), 'bs', ...
    'MarkerEdgeColor', 'k', ...
    'MarkerFaceColor', 'c', ...
    'MarkerSize', 7);

title("All three class classification");
hold on;

% 1.4
% Обучающая выборка
trainset = [trainInd1, trainInd2+60, trainInd3+160,...
    valInd1, valInd2+60, valInd3+160, ...
    testInd1, testInd2+60, testInd3+160];
```

```
% Инициализация множества
net = feedforwardnet(20);
net = configure(net, [-1.2 1.2; 0 1]);
net.layers{:}.transferFcn = 'tansig';
net.trainFcn = 'trainrp';

% 1.6
% Разделение на подмножества
net.divideFcn = 'divideind';

trnInd = length(trainInd1) + length(trainInd2) + length(trainInd3);
tstInd = length(valInd1) + length(valInd2) + length(valInd3);
proInd = length(testInd1) + length(testInd2) + length(testInd3);

net.divideParam.trainInd = 1:trnInd;
net.divideParam.valInd = (1:tstInd) + trnInd;
net.divideParam.testInd = (1:proInd) + (tstInd + trnInd);

% 1.7
% Инициализация сети
net = init(net);
% 1.8
% Задание параметров обучения
net.trainParam.epochs = 2000;
net.trainParam.max_fail = 1500;
net.trainParam.goal = 0.00001;

% 1.9
% Обучение
D = [D1, D2 + length(X1), D3 + length(X1) + length(X2)];
X = [X1, X2, X3];
Y = [Y1, Y2, Y3];
K = [K1, K2, K3];

[net, tr] = train(net, [X(D(trainset)); Y(D(trainset))], K(:, trainset));

% 1.10
% Вывод
display(net)

% 1.11
% Выход сети для множеств
trainInd = [trainInd1, trainInd2+60, trainInd3+160];
valInd = [valInd1, valInd2+60, valInd3+160];
testInd = [testInd1, testInd2+60, testInd3+160];
```

```
% Обучающее
A = net([X(D(trainInd)); Y(D(trainInd))]);
nA = A >= 0.5;

fprintf('Training size: %d\n', length(trainInd));
fprintf('Matches: %d\n\n', sum((sum(K(:,trainInd) == nA)) == 3));

% Контрольное
A = net([X(D(valInd)); Y(D(valInd))]);
nA = A >= 0.5;

fprintf('Target size: %d\n', length(valInd));
fprintf('Matches: %d\n\n', sum((sum(K(:,valInd) == nA)) == 3));

% Тестовая
A = net([X(D(testInd)); Y(D(testInd))]);
nA = A >= 0.5;

fprintf('Test size: %d\n', length(testInd));
fprintf('Matches: %d\n\n', sum((sum(K(:,testInd) == nA)) == 3));

% 1.13 Классификация точек области [-1.2,1.2]x[-1.2,1.2]
M = -1.2:0.010:1.2;
[gX, gY] = meshgrid(M);

A = net([gX(:)';gY(:)']);

% 1.14
mA = (round(10 * A) / 10)';
cmap = unique(mA,'rows');
out = [];

for i=1:length(mA)
    out = [out, find(ismember(cmap, mA(i,:), 'rows') == 1)];
end

out = reshape(out,length(gX), length(gY));
image(gX(:), gY(:), out);
colormap(cmap);
```

2.

```
t0 = 0;
tn = 2;
h = 0.01;
n = (tn - t0) / h + 1;

X = t0:h:tn;
Y = sin(0.5 * (X.^2) - 5 * X);

% 2.1
% Создание сети и конфигурация под множества
% Метод сопряженных градиентов Полака-Рибейры
net = feedforwardnet(15, 'traincgp');
net = configure(net, X, Y);

% 2.2
% Train test split (обучающая и контрольная)
trainInd = 1 : floor(n * 0.9);
valInd = floor(n * 0.9) + 1 : n;
net.divideFcn = 'divideind';
net.divideParam.trainInd = trainInd;
net.divideParam.valInd = valInd;
net.divideParam.testInd = [];

% 2.3
net = init(net);
```

```
% 2.4
net.trainParam.epochs = 2000;
net.trainParam.max_fail = 600;
net.trainParam.goal = 1.0e-8;

% 2.5
% Обучение сети
net = train(net, X, Y);

% 2.7
R = sim(net, X);
sqrt(mse(Y(trainInd) - R(trainInd)))
sqrt(mse(Y(valInd) - R(valInd)))

figure;
hold on;
% Эталонные значения
plot(X, Y, '-b');
title("Comparsion")
% Предсказанные значения
plot(X, R, '-r');
grid on;

figure;
% Ошибка
plot(X, Y - R);
title("Error")
grid on;
```

3.

```
t0 = 0;
tn = 2;
h = 0.01;
n = (tn - t0) / h + 1;

X = t0:h:tn;
Y = sin(0.5 * (X.^2) - 5 * X);

% 3.1
% Создание сети и конфигурация под множества
% Одношаговый метод секущих
net = feedforwardnet(10, 'trainoss');
net = configure(net, X, Y);

% 3.2
% Train test split (обучающая и контрольная)
trainInd = 1 : floor(n * 0.9);
valInd = floor(n * 0.9) + 1 : n;
net.divideFcn = 'divideind';
net.divideParam.trainInd = trainInd;
net.divideParam.valInd = valInd;
net.divideParam.testInd = [];

% 3.3
net = init(net);
```

```
% 3.4
net.trainParam.epochs = 2000;
net.trainParam.max_fail = 600;
net.trainParam.goal = 1.0e-8;

% 3.5
% Обучение сети
net = train(net, X, Y);

% 3.7
R = sim(net, X);
sqrt(mse(Y(trainInd) - R(trainInd)))
sqrt(mse(Y(valInd) - R(valInd)))

figure;
hold on;
% Эталонные значения
plot(X, Y, '-b');
title("Comparsion")
% Предсказанные значения
plot(X, R, '-r');
grid on;

figure;
% Ошибка
plot(X, Y - R);
title("Error")
grid on;
```