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

Лабораторная работа №6
по спецкурсу «Нейроинформатика»

Сети Кохонена

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

Москва, 2020

Цель работы
Целью работы является исследование свойств слоя Кохонена, карты Кохонена, а также сетей векторного квантования, обучаемых с учителем, алгоритмов обучения, а также применение сетей в задачах кластеризации и классификации.

Основные этапы работы:

1. Использовать слой Кохонена для выполнения кластеризации множества точек. Проверить качество разбиения.

2. Использовать карту Кохонена для выполнения кластеризации множества точек.

3. Использовать карту Кохонена для нахождения одного из решений задачи коммивояжера.

4. Использовать сеть векторного квантования, обучаемую с учителем, (LVQ-сеть) для клас- сификации точек в случае, когда классы не являются линейно разделимыми.

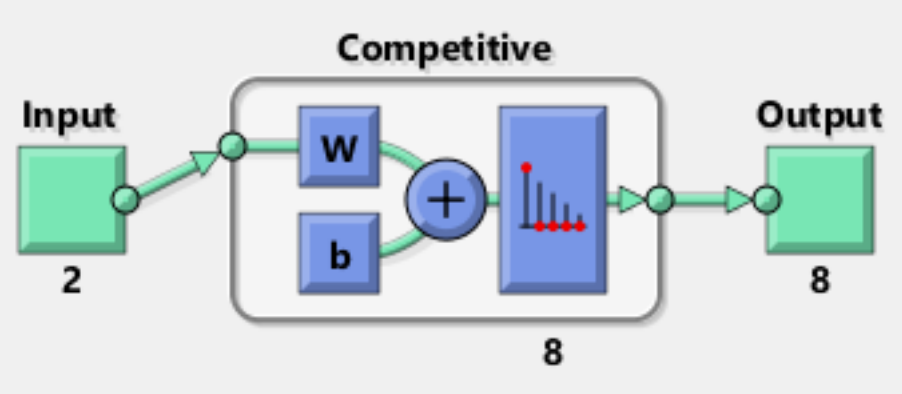
Оборудование
Процессор: 2,4 GHz Intel Core 2 Duo

Оперативная память: 8 ГБ 1067 MHz DDR3

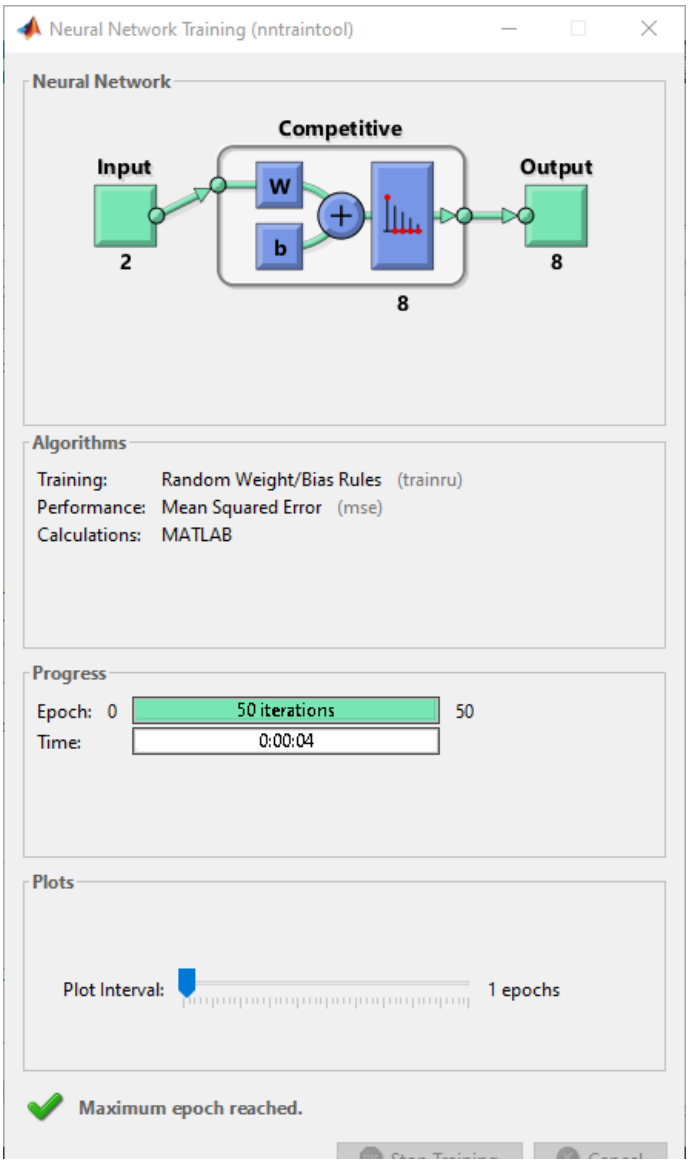
Программное обеспечение
Matlab R2020b, 64-bit.

Задание 1.

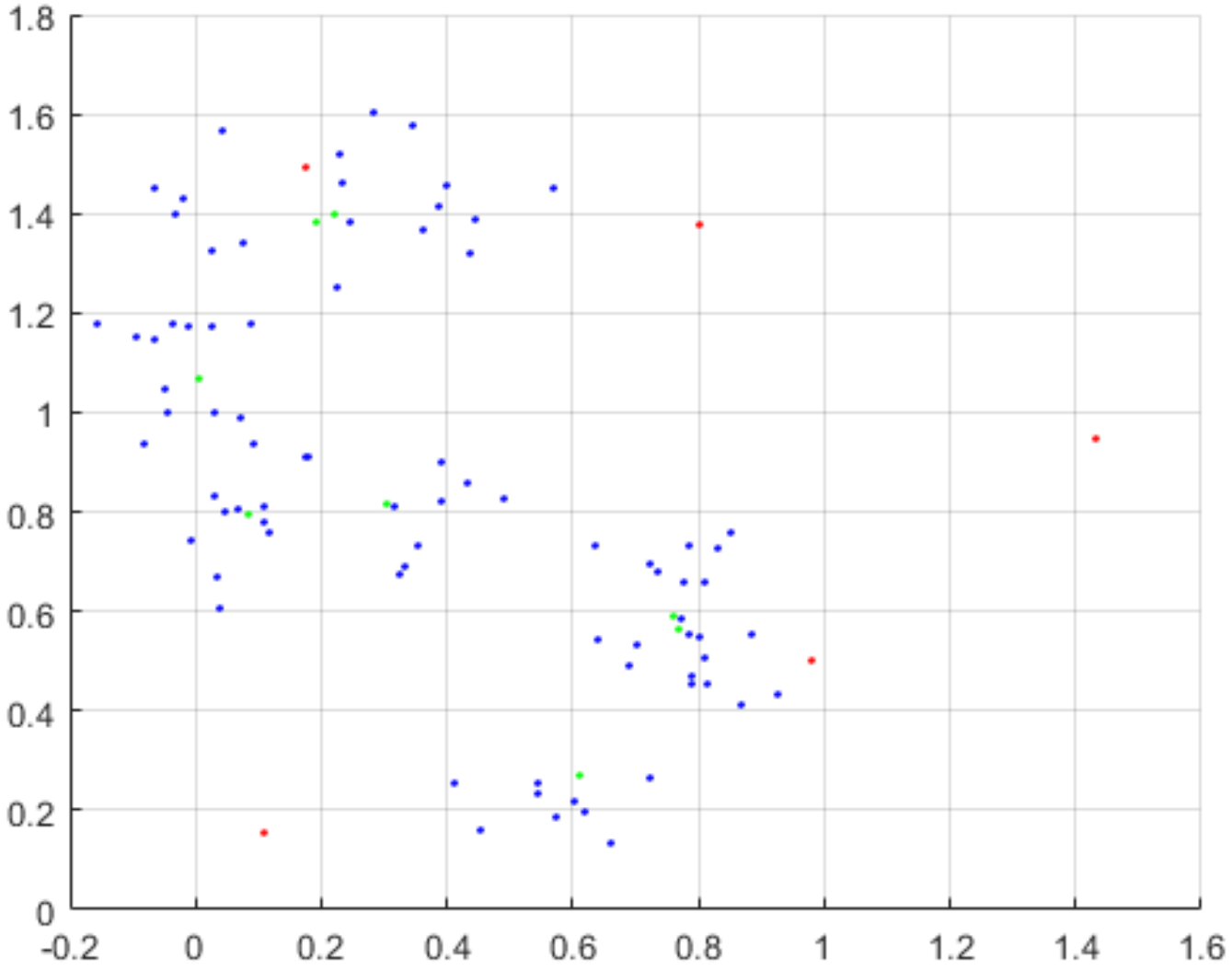
Структура сети:



Обучение сети:

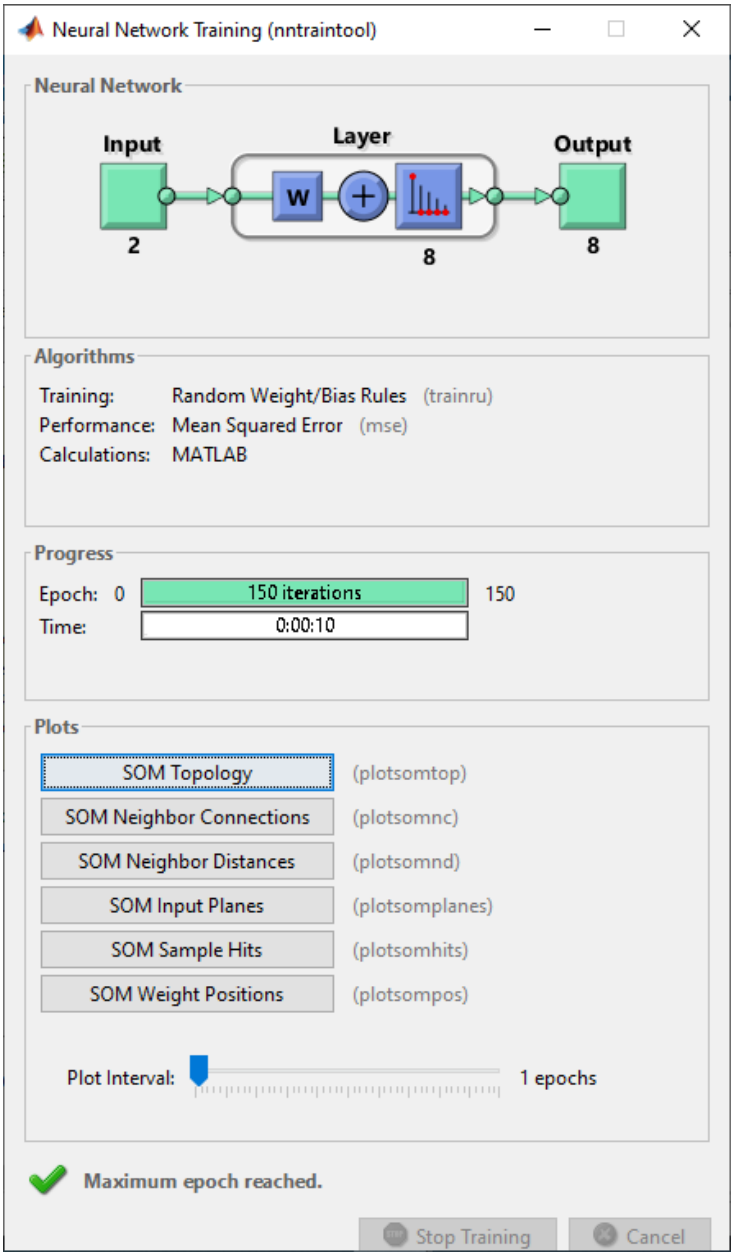


Проверка качества разбиения:

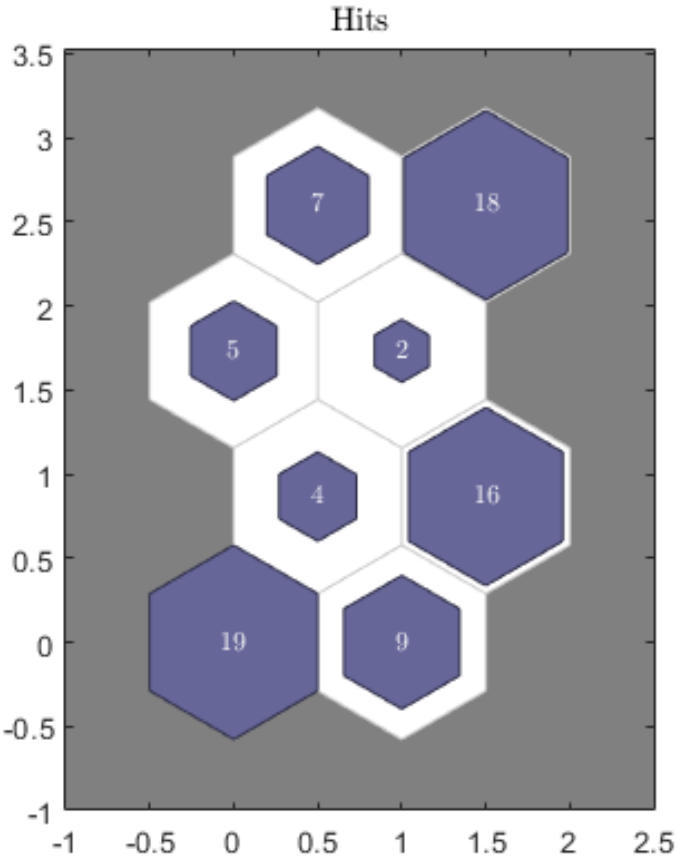


Задание 2.

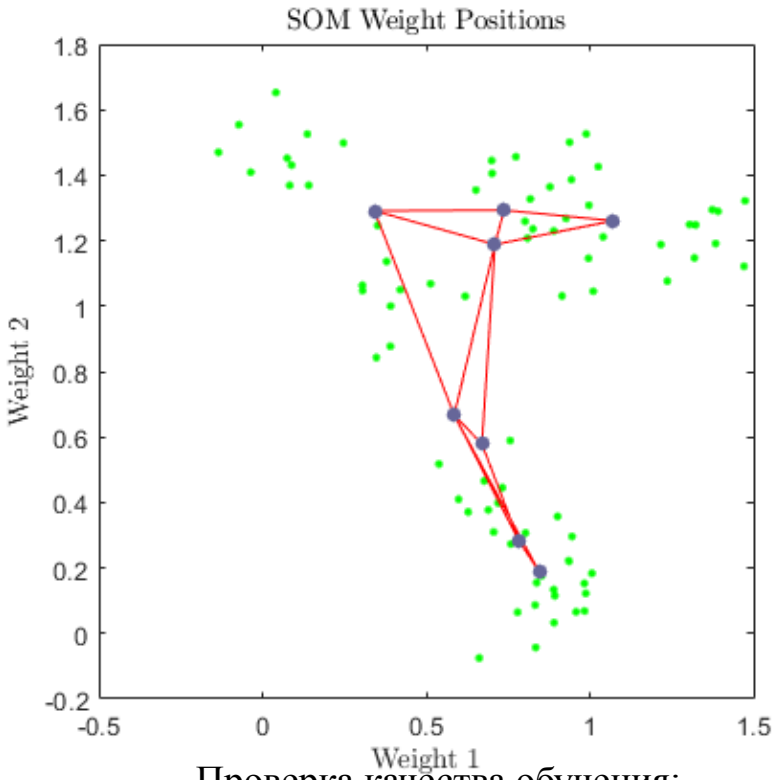
Обучение сети:



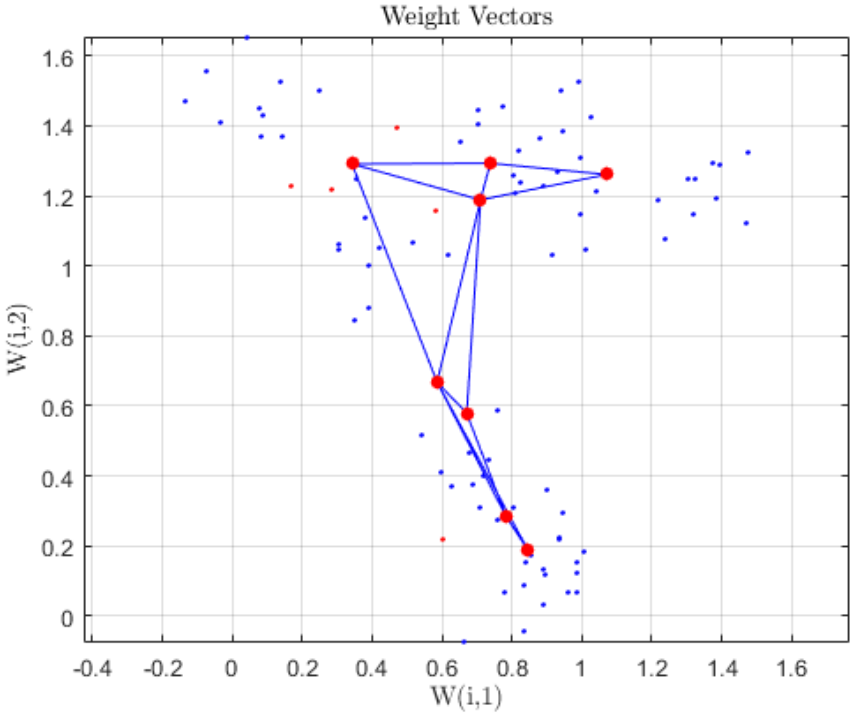
Обучение сети:



SOM Weight Positions:

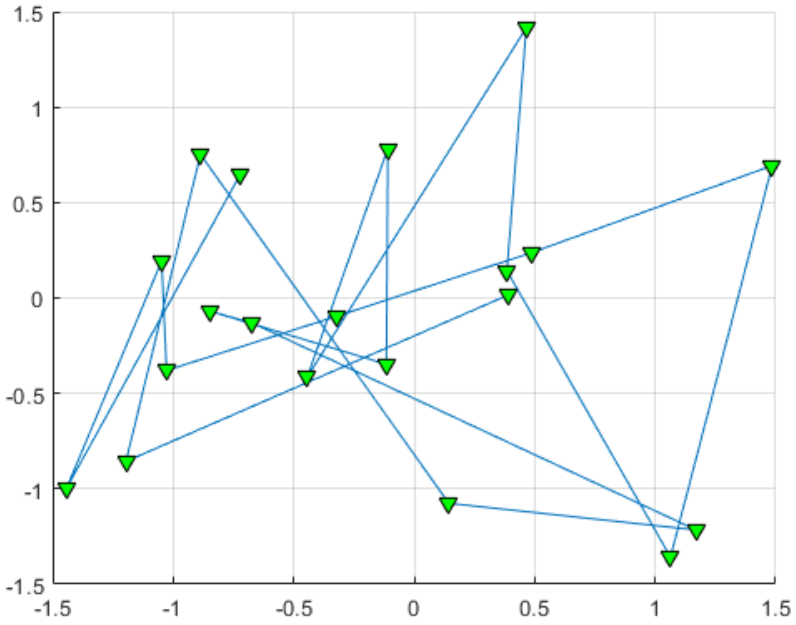


Проверка качества обучения:

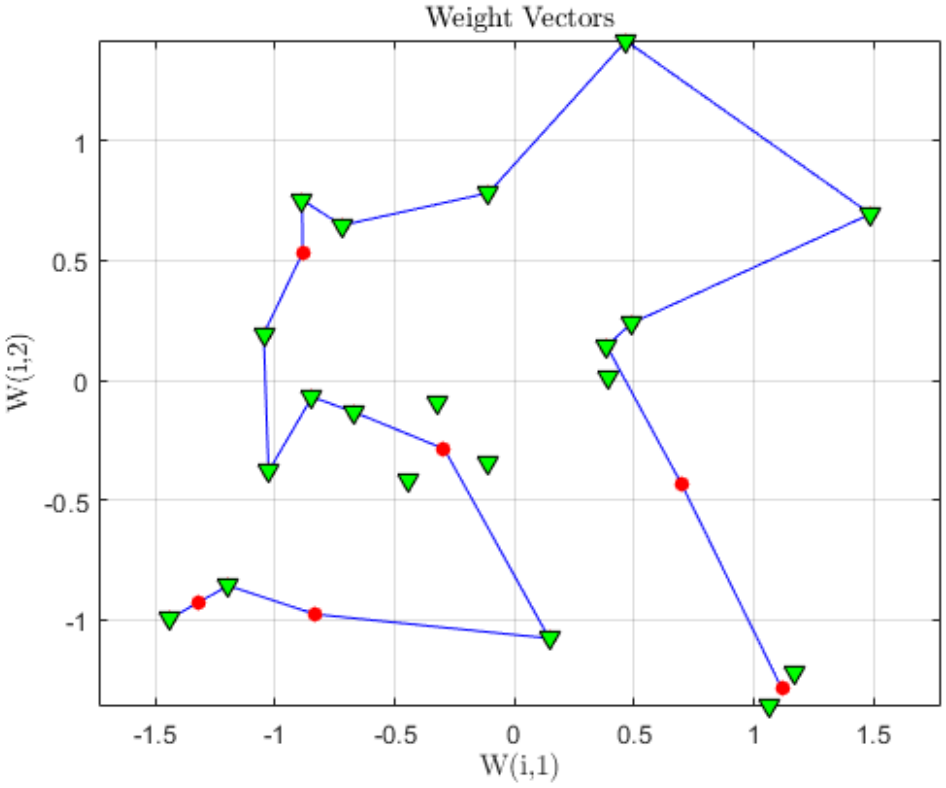


Задание 3.

Случайные точки:



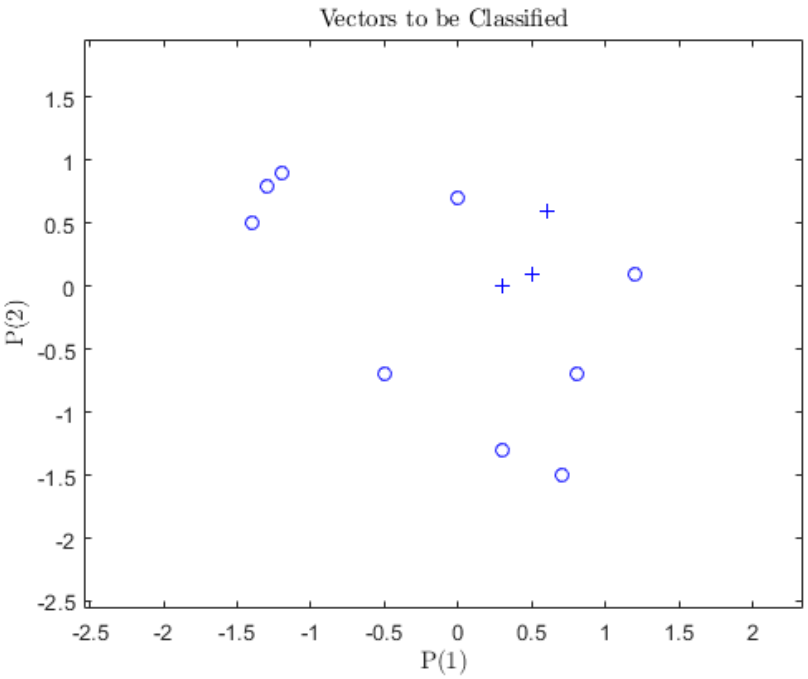
Координаты городов и центры кластеров, сгенерированные сетью :

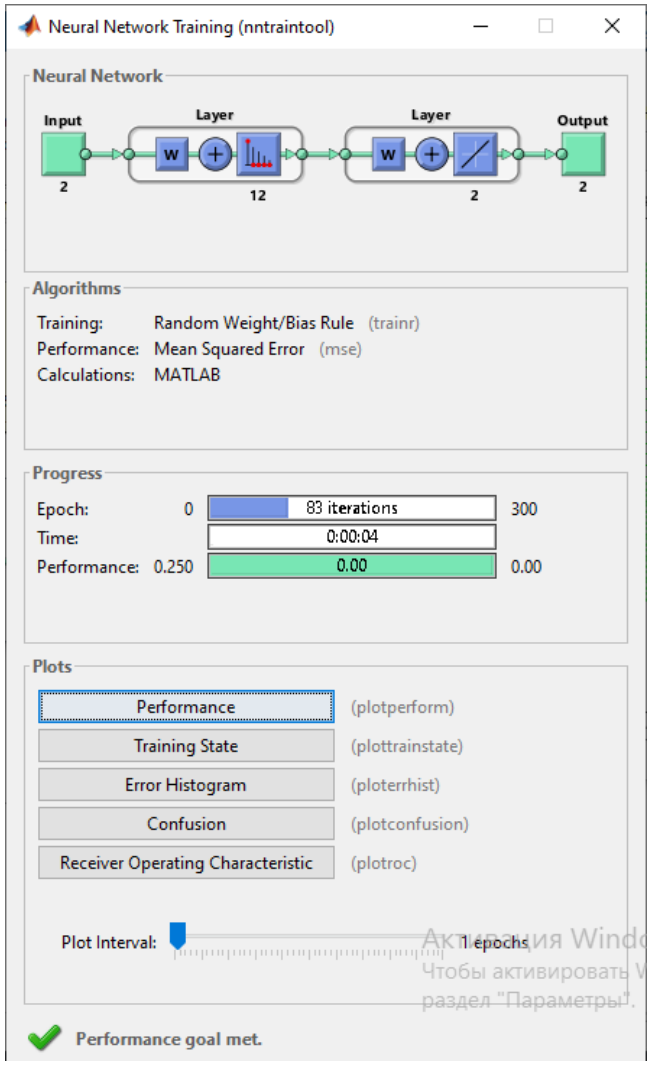


Задание 4.

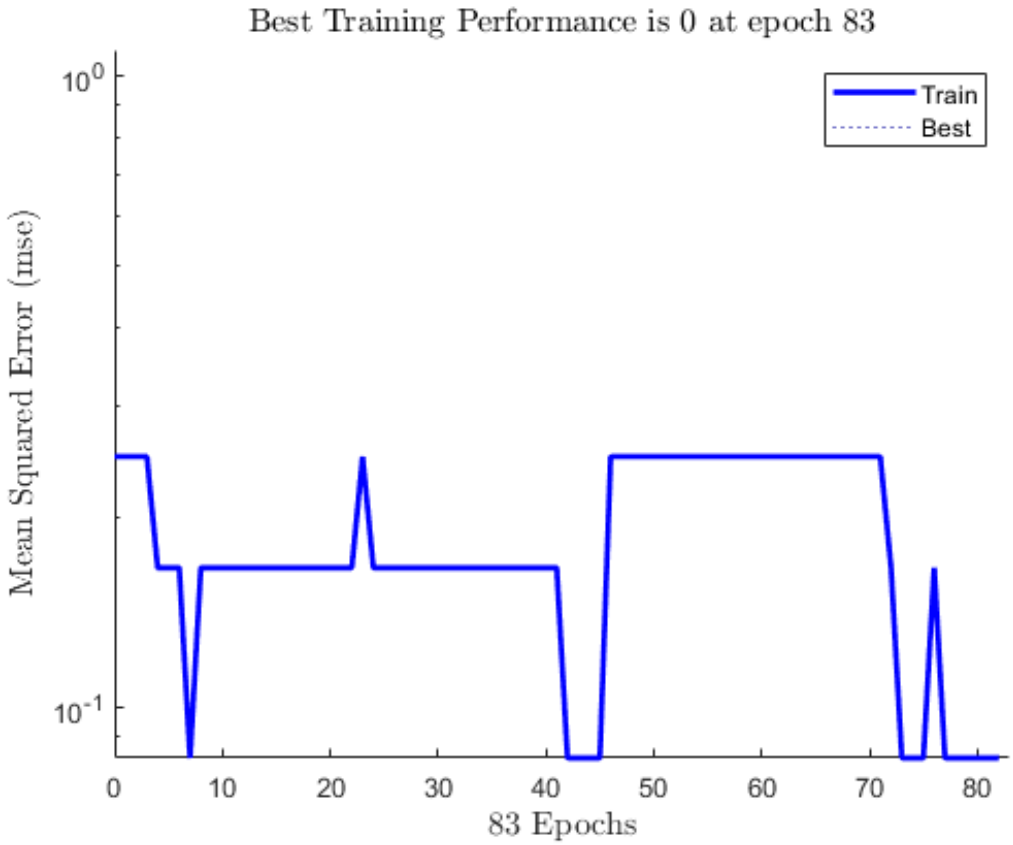
Обучение сети:

Входное множество:

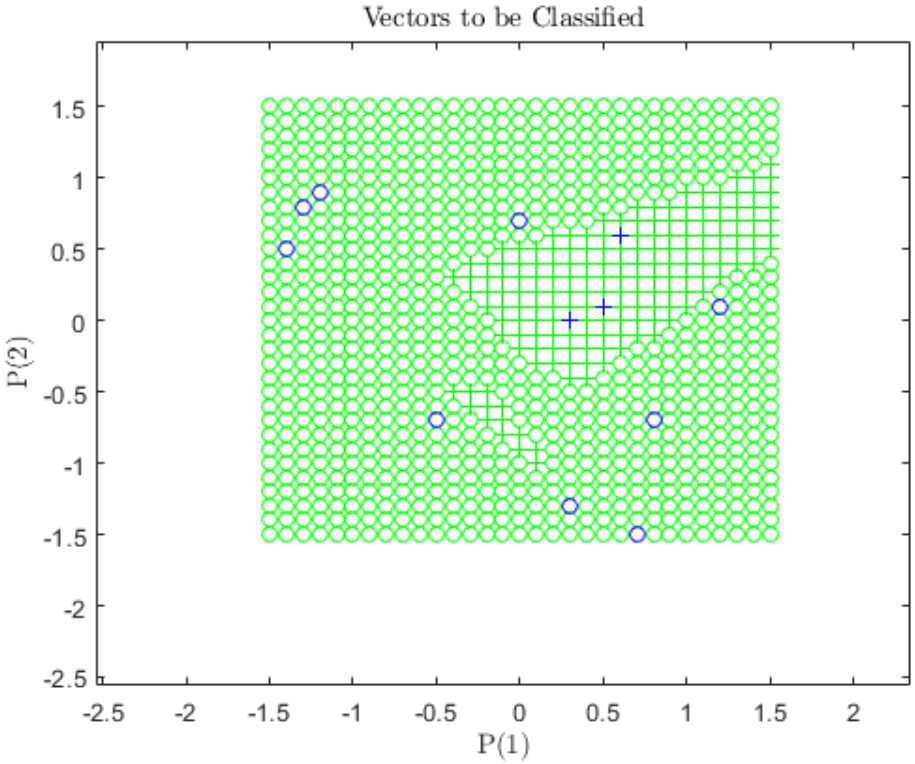




Perfomance:



Проверка качества обучения:



Код программы

Lab6.m

```
set(0, 'DefaultTextInterpreter', 'latex');
% Задание 1
% Формируем множество случайных точек
X = [0, 1.5; 0, 1.5];
clusters = 8;
points = 10;
deviation = 0.1;
randomDots = nngenc(X, clusters, points, deviation);

% создаем сеть
network = competlayer(8);
network = configure(network, randomDots);
view(network);

%% Обучаем её
network.trainParam.epochs = 50;
network = train(network, randomDots);

%% Проверяем качество обучения
dotsForCheck = 1.5 * rand(2, 5);
result = vec2ind(sim(network, dotsForCheck));

% Смотрим на результат
figure;
hold on;
grid on;
scatter(randomDots(1, :), randomDots(2, :), 5, [0 0 1], 'filled');
scatter(network.IW{1}(:, 1), network.IW{1}(:, 2), 5, [0 1 0], 'filled');
scatter(dotsForCheck(1, :), dotsForCheck(2, :), 5, [1 0 0], 'filled');

%% Задание 2
% Формируем множество случайных точек
X = [0, 1.5; 0, 1.5];
clusters = 8;
points = 10;
deviation = 0.1;
randomDots = nngenc(X, clusters, points, deviation);

% создаем сеть
network = newsom(X, [2 4]);
network = configure(network, X);
% Обучаем её
network.trainParam.epochs = 150;
%network.inputWeights{1,1}.learnParam.init_neighborhood = 3;
%network.inputWeights{1,1}.learnParam.steps = 100;
network = train(network, randomDots);

%% Проверяем качество обучения
dotsForCheck = 1.5 * rand(2, 5);
result = vec2ind(sim(network, dotsForCheck));
```

```
% Смотрим на результат
figure;
hold on;
grid on;
scatter(randomDots(1, :), randomDots(2, :), 5, [0 0 1], 'filled');
scatter(network.IW{1}(:, 1), network.IW{1}(:, 2), 5, [0 1 0], 'filled');
scatter(dotsForCheck(1, :), dotsForCheck(2, :), 5, [1 0 0], 'filled');
plotsom(network.IW{1, 1}, network.layers{1}.distances);

%% Задание 3
% Формируем множество случайных точек
T = -1.5 * ones(2, 20) + 3 * rand(2, 20);

figure;
hold on;
grid on;
plot(T(1,:), T(2,:), '-V', 'MarkerEdgeColor', 'k', 'MarkerFaceColor', 'g',
'MarkerSize', 7);

%% Создаем сеть
network = newsom(T, 20);
network = configure(network, T);

% Обучаем её
network.trainParam.epochs = 600;
network = train(network, T);

% Координаты городов и центры кластеров сгенерированные сетью
figure;
hold on;
plotsom(network.IW{1,1}, network.layers{1}.distances);
plot(T(1,:), T(2,:), 'V', 'MarkerEdgeColor', 'k', 'MarkerFaceColor', 'g',
'MarkerSize', 7);
grid on;

%% Задание 4
% Инициализируем входное множество и распределение по классам
P = [0    0.3 -1.3 1.2 -1.2 -0.5  0.7 -1.4 0.3 0.6  0.8 0.5;
     0.7 -1.3  0.8 0.1  0.9 -0.7 -1.5  0.5  0  0.6 -0.7 0.1];
T = [-1   -1   -1  -1   -1   -1   -1   -1   1   1   -1   1];

% Отображаем его
plotpv(P, max(0, T));

% Строим вектор индексов классов
Ti = T;
Ti(Ti == 1) = 2;
Ti(Ti == -1) = 1;
Ti = ind2vec(Ti);

%% Создаем нейросеть
portion = [nnz(T(T == -1)) nnz(T(T == 1))] / numel(T);
network = lvqnet(12, 0.1);
```

```
network = configure(network, P, Ti);

%network.IW{1,1}
%network.LW{2,1}

% Обучем нейросеть
network.trainParam.epochs = 300;
network = train(network, P, Ti);

%% Проверяем качество обучения
% Задаем сетку
[X,Y] = meshgrid([-1.5 : 0.1 : 1.5], [-1.5 : 0.1 : 1.5]);

% Получаем выход сети
result = sim(network, [X(:)'; Y(:)']);
result = vec2ind(result) - 1;

% Граффическая демонстрация
plotpv([X(:)'; Y(:)'], result);
point = findobj(gca,'type','line');
set(point,'Color','g');
hold on;
plotpv(P, max(0, T));
```

Вывод

Для применения сетей Кохогена необходимо знать количество кластеров, что существенно сужает количество решаемых задач. Карты Кохогена — довольно интересный инструмент для визуальной оценки данных, благодаря которому можно на плоскости оценивать точки принадлежащие многомерным пространствам.