**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет прикладной математики и физики

Кафедра вычислительной математики и программирования

**Лабораторная работа № 4**

по курсу «Нейроинформатика»

Тема: Сети с радиальными базисными элементами.

Студент: Дубинин А. О.

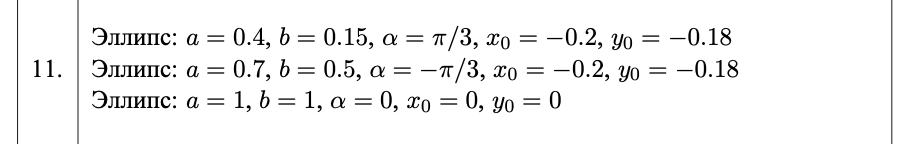
Группа: 80-407Б-17

Преподаватель: Аносова Н.П.

Оценка:

Москва, 2021

Вариант 11





Исследование свойств некоторых видов сетей с радиальными базисными элементами, алгоритмов обучения, а также применение таких сетей в задачах классификации и аппроксимаций функций.

1. Использовать вероятностную нейронную сеть для классификации точек в случае, когда классы не являются линейно разделимыми.
2. Использовать сеть с радиальными базисными элементами для классификации точек в случае, когда классы не являются линейно разделимыми.
3. Использовать обобщено-регрессионную нейронную сеть для аппроксимации функции. Проверить работу сетей с рыхлыми данными.

1. Для трех линейно неразделимых классов из лабораторной работы No 3 решить задачу клас- сификации. Точки, принадлежащие одному классу, лежат на алгебраической линии. Построить вероятностную сеть, которая будет классифицировать точки заданной области.

Обучающий набор {xi,yi}, i = 1,..,N, число классов K = 3. Сеть реализует отображение вида:

f(xi,yi)={(zk)Kk=1=(0,..,1,..0)|zk=K∗ =1при(xi,yi)∈K∗}

* 1. В соответствии с вариантом задания для каждой линии сгенерировать множество точек. Далее для первого класса выбрать из исходного множества случайным образом 60 точек. Для второго и третьего классов 100 и 120 точек соответственно.

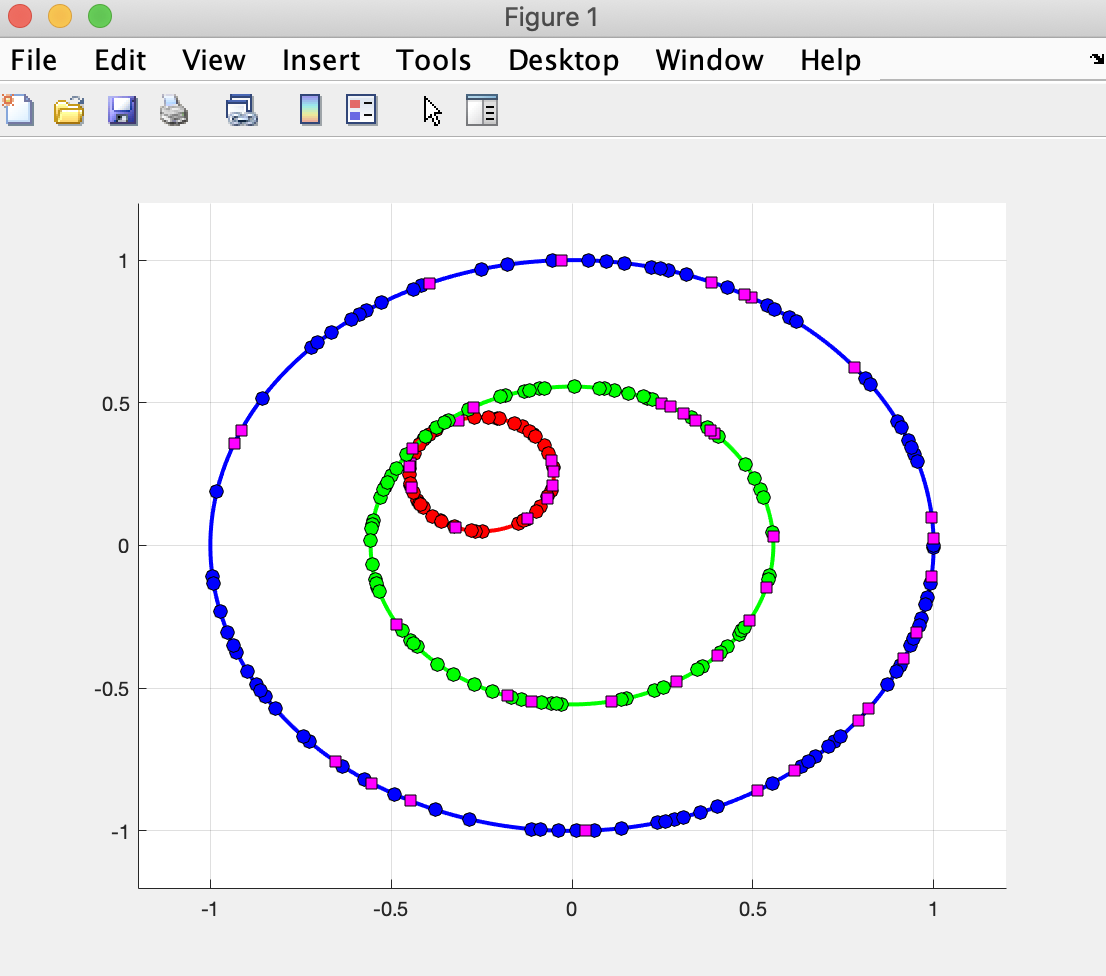
*%1.1 Генерация множества точек для каждой линии*n = length(X1);  
*%Случайные точки 1-го класса (60 шт)*D1 = randperm(n);  
n1 = 60;  
D1 = D1(1:n1);  
K1 = [ones(1, n1); 0\*ones(1, n1); 0\*ones(1, n1)];  
  
*%Случайные точки 2-го класса (100 шт)*D2 = randperm(n);  
n2 = 100;  
D2 = D2(1:n2);  
K2 = [0\*ones(1, n2); ones(1, n2); 0\*ones(1, n2)];  
  
*%Случайные точки 3-го класса (120 шт)*D3 = randperm(n);  
n3 = 120;  
D3 = D3(1:120);  
K3 = [0\*ones(1, n3); 0\*ones(1, n3); ones(1, n3)];

* 1. Множество точек, принадлежащее каждому классу, разделить на обучающее и тестовое подмножества с помощью функции dividerand в отношении 80%-20%.

*%1.2 Разделение точек на обучающее  
%и тестовое подмножества в отношении 80%-20%*[trainInd1, testInd1] = dividerand(length(D1), 0.8, 0.2);  
[trainInd2, testInd2] = dividerand(length(D2), 0.8, 0.2);  
[trainInd3, testInd3] = dividerand(length(D3), 0.8, 0.2);

* 1. Способом, описанным в Л.р. No3, отобразить множества точек для каждого класса, а также соответствующие обучающие и тестовые подмножества.

*%1.3 Отображение исходных данных*figure  
hold on;  
  
*%Отображение 1-го класса*class1 = plot(X1, Y1, **'-r'**, ...  
 **'LineWidth'**, 2);  
  
tr1 = plot(X1(D1(trainInd1)), Y1(D1(trainInd1)), **'or'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'r'**, ...  
 **'MarkerSize'**, 7);  
  
test1 = plot(X1(D1(testInd1)), Y1(D1(testInd1)), **'rs'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'m'**, ...  
 **'MarkerSize'**, 7);  
  
*%Отображение 2-го класса*class2 = plot(X2, Y2, **'-g'**, ...  
 **'LineWidth'**, 2);  
  
tr2 = plot(X2(D2(trainInd2)), Y2(D2(trainInd2)), **'og'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'g'**, ...  
 **'MarkerSize'**, 7);  
  
test2 = plot(X2(D2(testInd2)), Y2(D2(testInd2)), **'gs'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'m'**, ...  
 **'MarkerSize'**, 7);  
  
*%Отображение 3-го класса*class3 = plot(X3, Y3, **'-b'**, ...  
 **'LineWidth'**, 2);  
  
tr3 = plot(X3(D3(trainInd3)), Y3(D3(trainInd3)), **'ob'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'b'**, ...  
 **'MarkerSize'**, 7);  
  
test3 = plot(X3(D3(testInd3)), Y3(D3(testInd3)), **'bs'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'m'**, ...  
 **'MarkerSize'**, 7);  
  
axis([-1.2 1.2 -1.2 1.2]);  
grid on;



* 1. Соответствующие подмножества точек объединить в обучающее и тестовое подмножества обучающей выборки.

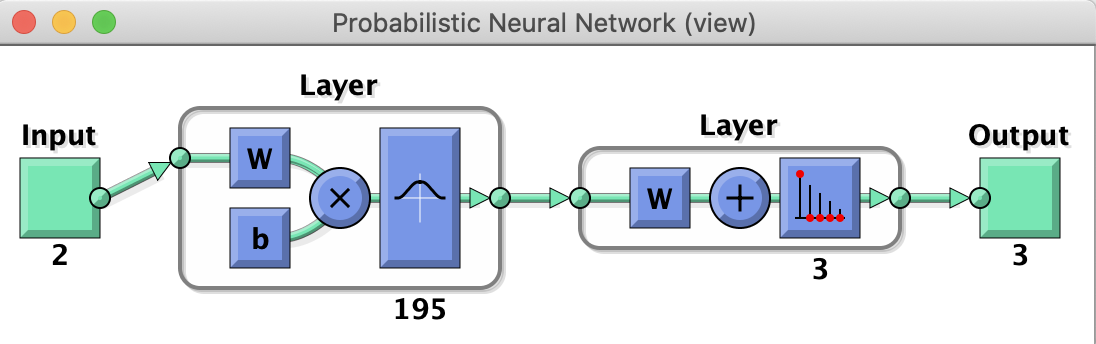
*%1.4 Обьединение соответствующего подмножества точек в обучающее и тестовое*trainset = [trainInd1, trainInd2+60, trainInd3+160];  
testset = [testInd1, testInd2+60, testInd3+160];  
X = [X1, X2, X3];  
Y = [Y1, Y2, Y3];  
D = [D1, D2+length(X1), D3+length(X1)+length(X2)];  
K = [K1, K2, K3];

* 1. Константу SPREAD задать равной 0.3. Создать сеть с помощью функции newpnn. Подать в сеть обучающее подмножество обучающей выборки.

*%1.6 Создание сети*spread = 0.3;  
net = newpnn([X(D(trainset)); Y(D(trainset))], K(:,trainset), spread);

* 1. Отразить структуру сети, заполнив таблицу 1.

*%1.7 Отображение структуры сети*view(net);



1.8 Проверить качество обучения: рассчитать выход сети для обучающего подмножества обу- чающей выборки. Преобразовать выходные значения с помощью функции (vec2ind). Занести в отчет количество правильно классифицированных точек.

1.9 Провести аналогичные расчеты для тестового подмножеств.

*%1.8/1.9 Проверка качества*PK = net([X(D(trainset)); Y(D(trainset))]);  
TK = K(:,trainset);  
  
  
fprintf(**'Размер обучающей выборки: %d\n'**,length(trainset));  
fprintf(**'Количество совпадений: %d\n\n'**,sum(TK(TK == PK)));  
  
PK = net([X(D(testset)); Y(D(testset))]);  
TK = K(:,testset);  
  
fprintf(**'Размер тестовой выборки: %d\n'**,length(testset));  
fprintf(**'Количество совпадений: %d\n\n'**,sum(TK(TK == PK)));

Размер обучающей выборки: 195

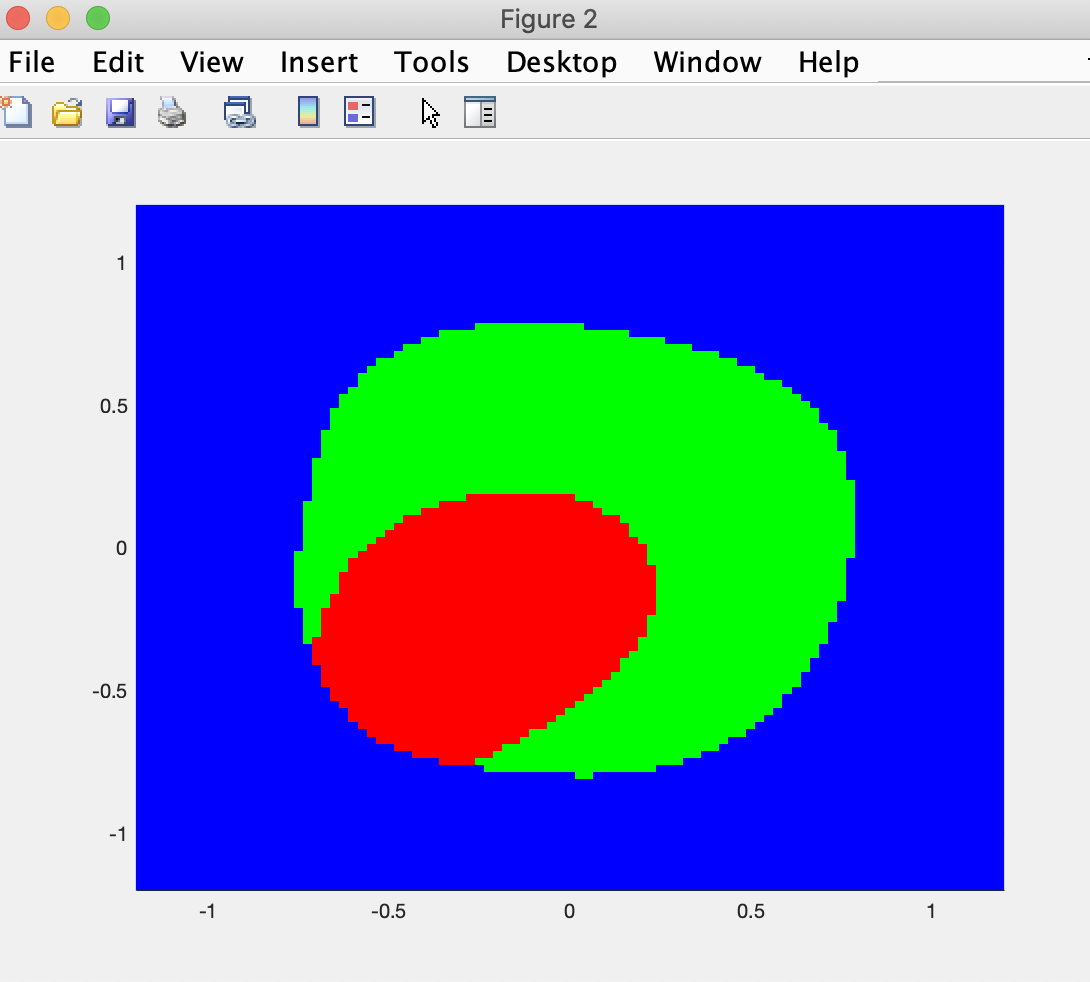
Количество совпадений: 177

Размер тестовой выборки: 48

Количество совпадений: 43

1.10 Произвести классификацию точек области [−1.2, 1.2] × [−1.2, 1.2]. Закодировать принад- лежности к классам различными цветами и занести полученное изображение в отчет. Для этого использовать методику, описанную в лабораторной работе No 3.

*%1.10 Классификация точек области*[gX, gY] = meshgrid(-1.2:0.025:1.2, 1.2:-0.025:-1.2);  
  
A = net([gX(:)';gY(:)']);  
n = length(gX);  
A = max(0, min(1, A));  
A = round(A \* 10) \* 0.1;  
  
ctable = unique(A', **'rows'**);  
cmap = zeros(n, n);  
  
**for** i = 1 : size(ctable, 1)  
 cmap(ismember(A', ctable(i, :), **'rows'**)) = i;   
**end**image([-1.2, 1.2], [-1.2, 1.2], cmap);   
colormap(ctable);  
  
figure  
hold on;  
axis([-1.2 1.2 -1.2 1.2]);  
grid on;

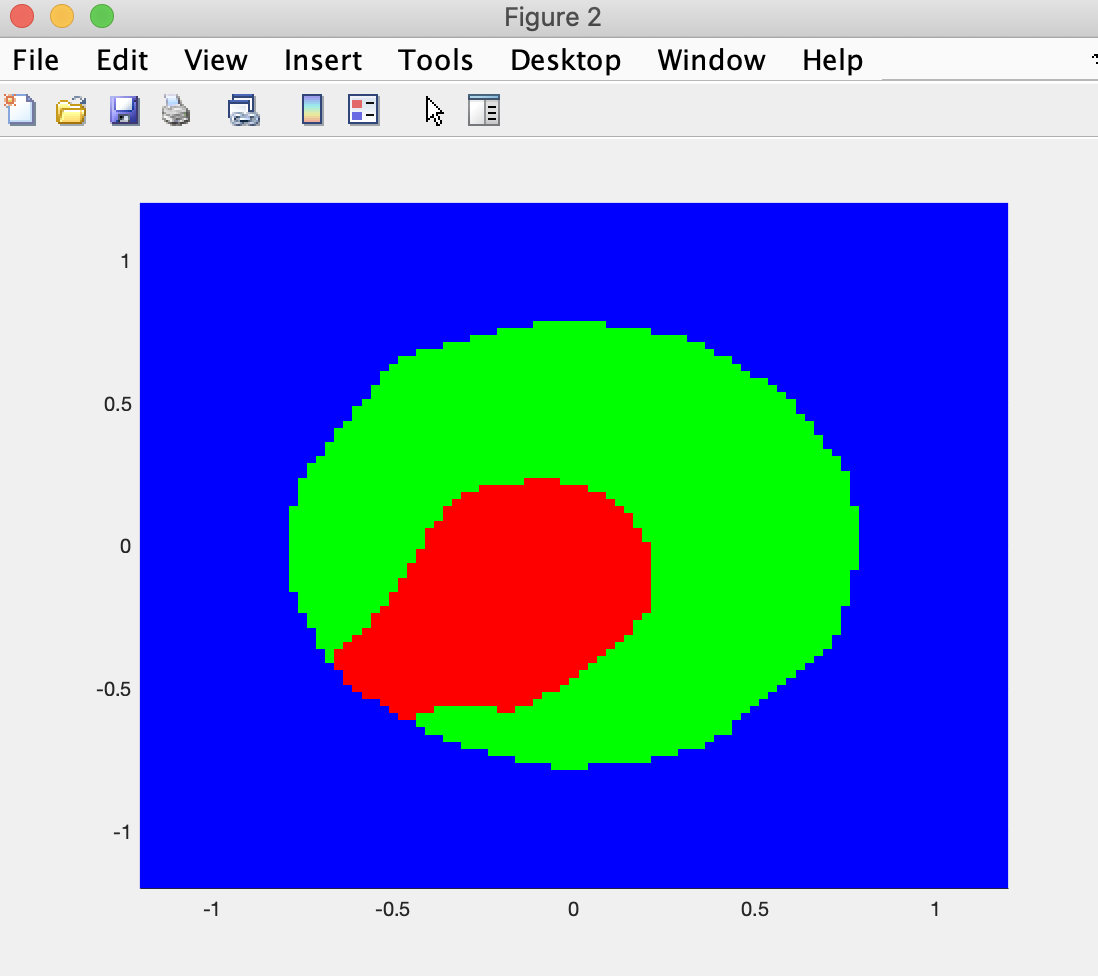


1.11 Константу SPREAD задать равной 0.1. Создать сеть с помощью функции newpnn.

*%1.11 Меняем spread*spread = 0.1;  
net = newpnn([X(D(trainset)); Y(D(trainset))], K(:,trainset), spread);  
view(net);

1.12 Произвести классификацию точек области [−1.2, 1.2] × [−1.2, 1.2]. Закодировать принад- лежности к классам различными цветами и занести полученное изображение в отчет. Использо- вать методику, описанную в лабораторной работе No 3.

*%1.12 Классификация точек области*A = net([gX(:)';gY(:)']);  
n = length(gX);  
A = max(0, min(1, A));  
A = round(A \* 10) \* 0.1;  
  
ctable = unique(A', **'rows'**);  
cmap = zeros(n, n);  
  
**for** i = 1 : size(ctable, 1)  
 cmap(ismember(A', ctable(i, :), **'rows'**)) = i;   
**end**image([-1.2, 1.2], [-1.2, 1.2], cmap);   
colormap(ctable);



2. Для трех линейно неразделимых классов из лабораторной работы No 3 решить задачу клас- сификации. Точки, принадлежащие одному классу, лежат на алгебраической линии. Построить сеть с радиальными базисными элементами, которая будет классифицировать точки заданной области.

2.1 В соответствии с вариантом задания для каждой линии сгенерировать множество точек. Далее для первого класса выбрать из исходного множества случайным образом 60 точек. Для второго и третьего классов 100 и 120 точек соответственно.

*%2.1 Генерация множества точек для каждой линии*[X1, Y1] = arrangeEllipse(-0.25, 0.25, 0.2, 0.2, 0, 0.025);  
[X2, Y2] = arrangeEllipse(0, 0, 0.7, 0.5, -pi/3, 0.025);  
[X3, Y3] = arrangeEllipse(0, 0, 1, 1, 0, 0.025);  
  
n = length(X1);  
*%Случайные точки 1-го класса (60 шт)*D1 = randperm(n);  
n1 = 60;  
D1 = D1(1:n1);  
K1 = [ones(1, n1); 0\*ones(1, n1); 0\*ones(1, n1)];  
  
*%Случайные точки 2-го класса (100 шт)*D2 = randperm(n);  
n2 = 100;  
D2 = D2(1:n2);  
K2 = [0\*ones(1, n2); ones(1, n2); 0\*ones(1, n2)];  
  
*%Случайные точки 3-го класса (120 шт)*D3 = randperm(n);  
n3 = 120;  
D3 = D3(1:120);  
K3 = [0\*ones(1, n3); 0\*ones(1, n3); ones(1, n3)];

2.2 Множество точек, принадлежащее каждому классу, разделить на обучающее и тестовое подмножества с помощью функции dividerand в отношении 80%-20%.

*%2.2 Разделение точек на обучающее  
%и тестовое подмножества в отношении 80%-20%*[trainInd1, testInd1] = dividerand(length(D1), 0.8, 0.2);  
[trainInd2, testInd2] = dividerand(length(D2), 0.8, 0.2);  
[trainInd3, testInd3] = dividerand(length(D3), 0.8, 0.2);

2.3 Способом, описанным в Л.р. No 3, отобразить множества точек для каждого класса, а также соответствующие обучающие и тестовые подмножества.

*%2.3 Отображение исходных данных*figure  
hold on;  
  
*%Отображение 1-го класса*class1 = plot(X1, Y1, **'-r'**, ...  
 **'LineWidth'**, 2);  
  
tr1 = plot(X1(D1(trainInd1)), Y1(D1(trainInd1)), **'or'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'r'**, ...  
 **'MarkerSize'**, 7);  
  
test1 = plot(X1(D1(testInd1)), Y1(D1(testInd1)), **'rs'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'r'**, ...  
 **'MarkerSize'**, 7);  
  
*%Отображение 2-го класса*class2 = plot(X2, Y2, **'-g'**, ...  
 **'LineWidth'**, 2);  
  
tr2 = plot(X2(D2(trainInd2)), Y2(D2(trainInd2)), **'og'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'g'**, ...  
 **'MarkerSize'**, 7);  
  
test2 = plot(X2(D2(testInd2)), Y2(D2(testInd2)), **'gs'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'g'**, ...  
 **'MarkerSize'**, 7);  
  
*%Отображение 3-го класса*class3 = plot(X3, Y3, **'-b'**, ...  
 **'LineWidth'**, 2);  
  
tr3 = plot(X3(D3(trainInd3)), Y3(D3(trainInd3)), **'ob'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'b'**, ...  
 **'MarkerSize'**, 7);  
  
test3 = plot(X3(D3(testInd3)), Y3(D3(testInd3)), **'bs'**, ...  
 **'MarkerEdgeColor'**, **'k'**, ...  
 **'MarkerFaceColor'**, **'b'**, ...  
 **'MarkerSize'**, 7);  
  
legend(**'Исходное множество 1'**, **'Обучающее подмножество 1'**, **'Тестовое подмножество 1'**,...  
 **'Исходное множество 2'**, **'Обучающее подмножество 2'**, **'Тестовое подмножество 2'**,...  
 **'Исходное множество 3'**, **'Обучающее подмножество 3'**, **'Тестовое подмножество 3'**);

2.4 Соответствующие подмножества точек объединить в обучающее и тестовое подмножества обучающей выборки.

*%2.4 Обьединение соответствующега подмножества точек в обучающее и тестовое*trainset = [trainInd1, trainInd2+60, trainInd3+160];  
testset = [testInd1, testInd2+60, testInd3+160];  
X = [X1, X2, X3];  
Y = [Y1, Y2, Y3];  
D = [D1, D2+length(X1), D3+length(X1)+length(X2)];  
K = [K1, K2, K3];

2.5 Создать сеть с помощью newrb, задав следующие параметры: предельное значение кри- терия обучения (goal) — 10−5, SPREAD — 0.3, размер обучающей выборки — число элементов в обучающем подмножестве. В сеть подается обучающее подмножество обучающей выборки.

*%2.5 Создание сети*spread = 0.3;  
goal = 1.0e-5;  
net = newrb([X(D(trainset)); Y(D(trainset))], K(:,trainset), goal, spread);

2.6 Занести в отчет окно Training with newrb. Отразить структуру сети, заполнив таблицу 1. Указать число радиальных базисных нейронов.

*%2.6 Отображение структуры сети*view(net);

2.7 Проверить качество обучения: рассчитать выход сети для обучающего подмножества обу- чающей выборки. Занести в отчет количество правильно классифицированных точек.

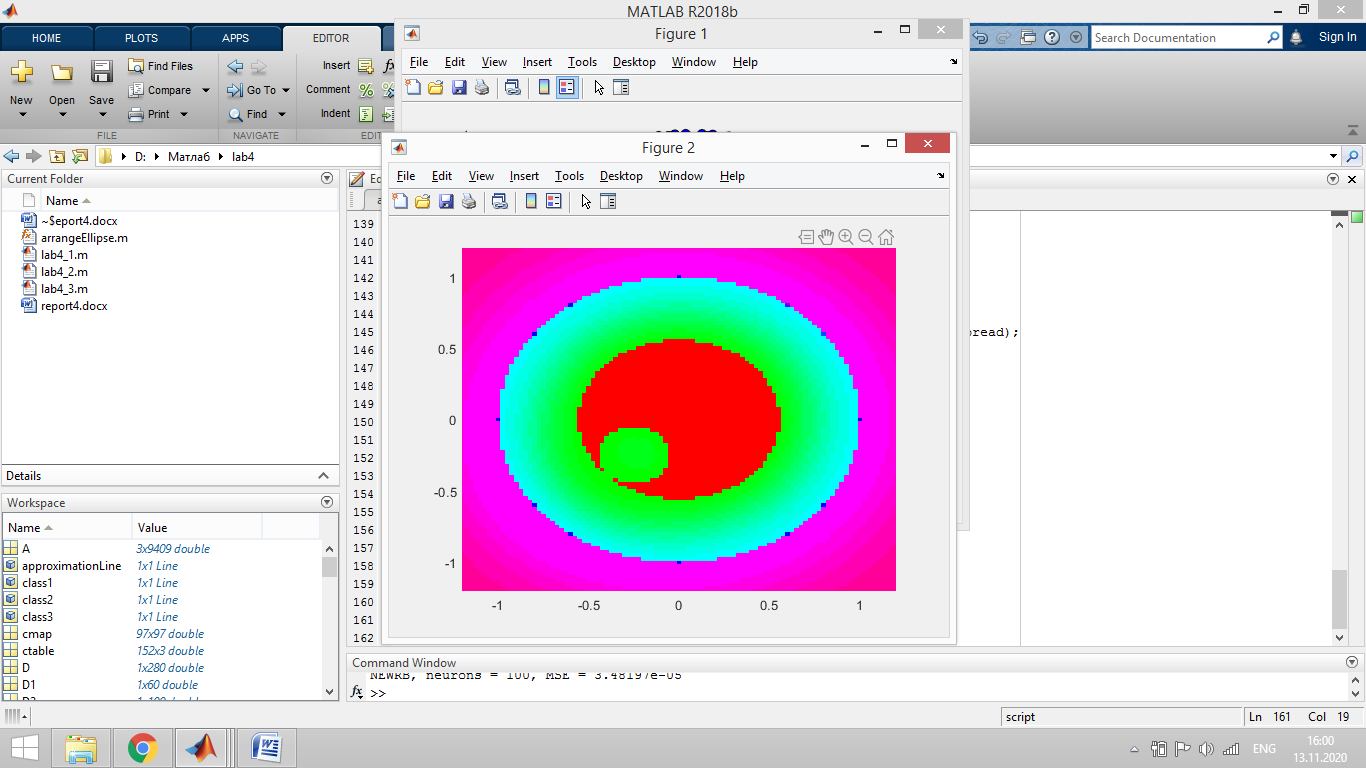
2.8 Провести аналогичные расчеты для тестового подмножеств.

*%2.7/2.8 Проверка качества*PK = net([X(D(trainset)); Y(D(trainset))]);  
TK = K(:,trainset);  
  
fprintf(**'Размер обучающей выборки: %d\n'**,length(trainset));  
fprintf(**'Количество совпадений: %d\n\n'**,sum(TK(TK == PK)));  
  
PK = net([X(D(testset)); Y(D(testset))]);  
TK = K(:,testset);  
  
fprintf(**'Размер тестовой выборки: %d\n'**,length(testset));  
fprintf(**'Количество совпадений: %d\n\n'**,sum(TK(TK == PK)));

2.9 Произвести классификацию точек области [−1.2, 1.2] × [−1.2, 1.2]. Закодировать принад- лежности к классам различными цветами и занести полученное изображение в отчет. Для этого использовать методику, описанную в лабораторной работе No 3.

*%2.9 Классификация точек области*figure  
hold on;  
axis([-1.2 1.2 -1.2 1.2]);  
grid on  
  
[gX, gY] = meshgrid(-1.2:0.025:1.2, 1.2:-0.025:-1.2);  
  
A = net([gX(:)';gY(:)']);  
n = length(gX);  
A = max(0, min(1, A));  
A = round(A \* 10) \* 0.1;  
  
ctable = unique(A', **'rows'**);  
cmap = zeros(n, n);  
  
**for** i = 1 : size(ctable, 1)  
 cmap(ismember(A', ctable(i, :), **'rows'**)) = i;   
**end**image([-1.2, 1.2], [-1.2, 1.2], cmap);   
colormap(ctable);  
  
  
hold on;  
axis([-1.2 1.2 -1.2 1.2]);  
grid on

Spread = 0.3



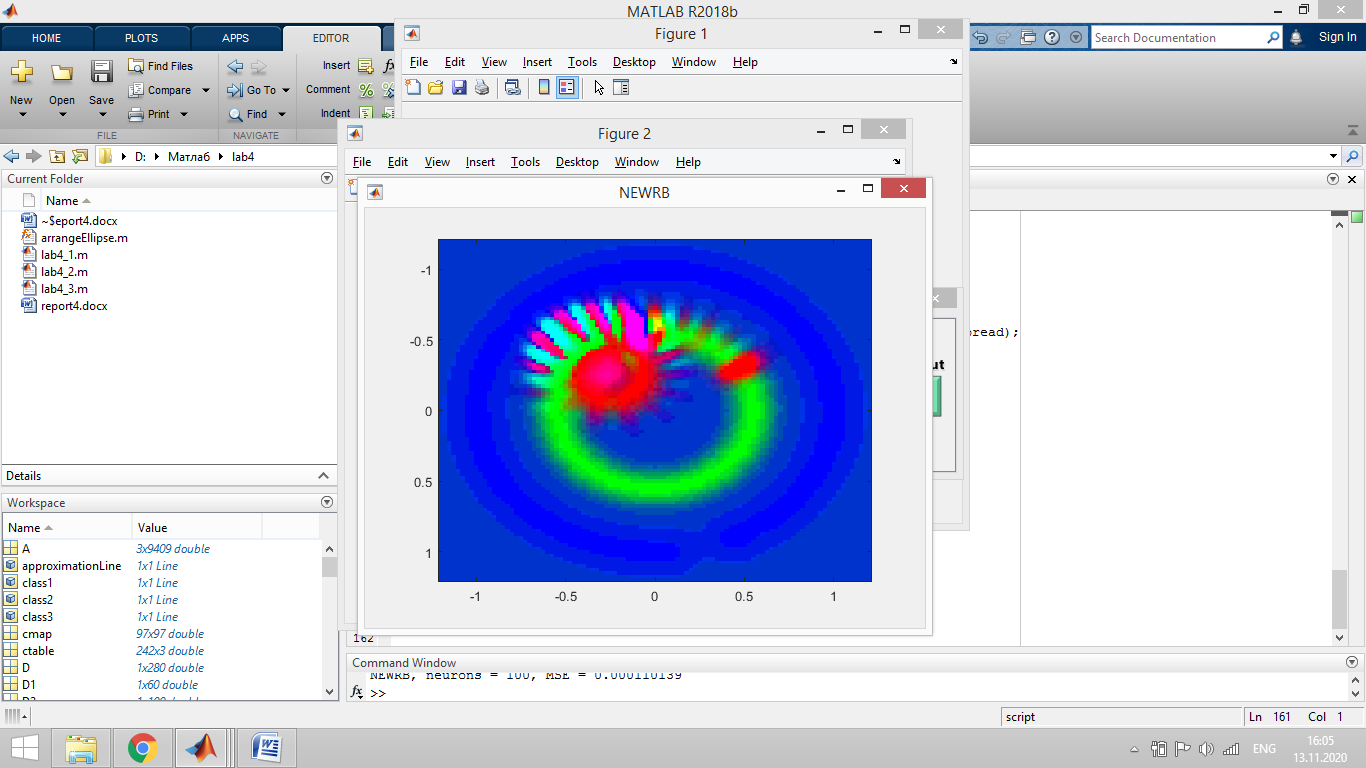
2.10 Константу SPREAD задать равной 0.1. Создать сеть с помощью функции newrb.

*%2.10 Меняем spread*spread = 0.1;  
goal = 1.0e-5;  
net = newrb([X(D(trainset)); Y(D(trainset))], K(:,trainset), goal, spread);  
view(net);

2.11 Произвести классификацию точек области [−1.2, 1.2] × [−1.2, 1.2]. Закодировать принад- лежности к классам различными цветами и занести полученное изображение в отчет. Использо- вать методику, описанную в лабораторной работе No 3.

*%2.11 Классификация точек области*A = net([gX(:)';gY(:)']);  
n = length(gX);  
A = max(0, min(1, A));  
A = round(A \* 10) \* 0.1;  
  
ctable = unique(A', **'rows'**);  
cmap = zeros(n, n);  
  
**for** i = 1 : size(ctable, 1)  
 cmap(ismember(A', ctable(i, :), **'rows'**)) = i;   
**end**image([-1.2, 1.2], [-1.2, 1.2], cmap);   
colormap(ctable);

Spread = 0.1



3. Задан обучающий набор {x(i), y(i)}. Построить обобщенно-регрессионную нейронную сеть, которая будет выполнять аппроксимацию функции

yˆ(i) = f[x(i)]  
Функцию, соответствующего варианта, взять из лабораторной работы No 3.

3.1 Создать сеть с помощью функции *newgrnn(P1,T1,SPREAD)*. Константу *SPREAD* задать равной h, где h — величина шага для заданной функции.

3.2 Произвести разделение обучающей выборки на обучающее и тестовое подмножества. Индексы обучающего подмножества использовать для создания сети.

P 1 = P (trainInd); T 1 = T (trainInd);

Выделить с конца временной последовательности 10% отсчетов на тестовое подмножество.

3.3 Если результаты неудовлетворительные, то изменить значение *SPREAD* и создать новую сеть.

3.4 Отразить структуру сети и проведенное обучение в отчете, заполнив таблицу 1.

*%Использование обобщенно-регрессионной нейронной сети для апроксимации  
%функции*t0 = 0;  
tn = 3.5;  
h = 0.01;  
n = (tn - t0) / h + 1;  
  
t = t0:h:tn;  
x = sin(-2\*(t.\*t) + 7\*t);  
  
*%3.1-3.4 Создание сети*spread = h;  
  
[trainInd, testInd] = dividerand(n, .8, .2);  
net = newgrnn(t, x, spread);  
view(net);  
  
y = sim(net, t);  
fprintf(**'Train error: %d\n'**,sqrt(mse(x(trainInd) - y(trainInd))));  
fprintf(**'Test error: %d\n'**,sqrt(mse(x(testInd) - y(testInd))));

3.5 Рассчитать выход сети (*sim*) для обучающего подмножества. Сравнить выход сети с соот- ветствующим эталонным подмножеством: рассчитать показатели качества обучения и заполнить таблицу 2. Отобразить на графике эталонные значения и предсказанные сетью. Отобразить на отдельном графике ошибку обучения. Графики занести в отчет.

3.6 Получить апостериорную оценку качества работы сети: проделать аналогичные действия для тестового подмножества.

3.7 Сформировать обучающее множество с рыхлыми данными. Для этого произвести раз- деление обучающей выборки на обучающее и тестовое подмножества. с помощью функции (*dividerand*) в соотношении 80% и 20%.

3.8 Рассчитать выход сети (*sim*) для обучающего подмножества. Сравнить выход сети с соот- ветствующим эталонным подмножеством: рассчитать показатели качества обучения и заполнить таблицу 2. Отобразить на графике эталонные значения и предсказанные сетью, а также ошибку обучения. Графики занести в отчет.

*%3.5-3.8 Графики*figure  
referenceLine = plot(t, x, **'r'**);  
  
hold on;  
approximationLine = plot(t, y, **'-b'**);  
  
grid on;  
legend([referenceLine,approximationLine],**'reference line'**, **'approximation line'**);  
figure;  
plot(t, x - y, **'r'**);  
grid on;  
  
net = newgrnn(t(trainInd), x(trainInd), spread);  
y = sim(net, t);  
figure  
referenceLine = plot(t, x, **'r'**);  
set(referenceLine, **'linewidth'**, 4);  
  
hold on;  
approximationLine = plot(t, y, **'--b'**);  
set(approximationLine, **'linewidth'**, 4);  
  
grid on;  
legend([referenceLine,approximationLine],**'reference line'**, **'approximation line'**);  
figure;  
plot(t, x - y, **'r'**);  
grid on;

Сравнение эталонных значений с предсказанными

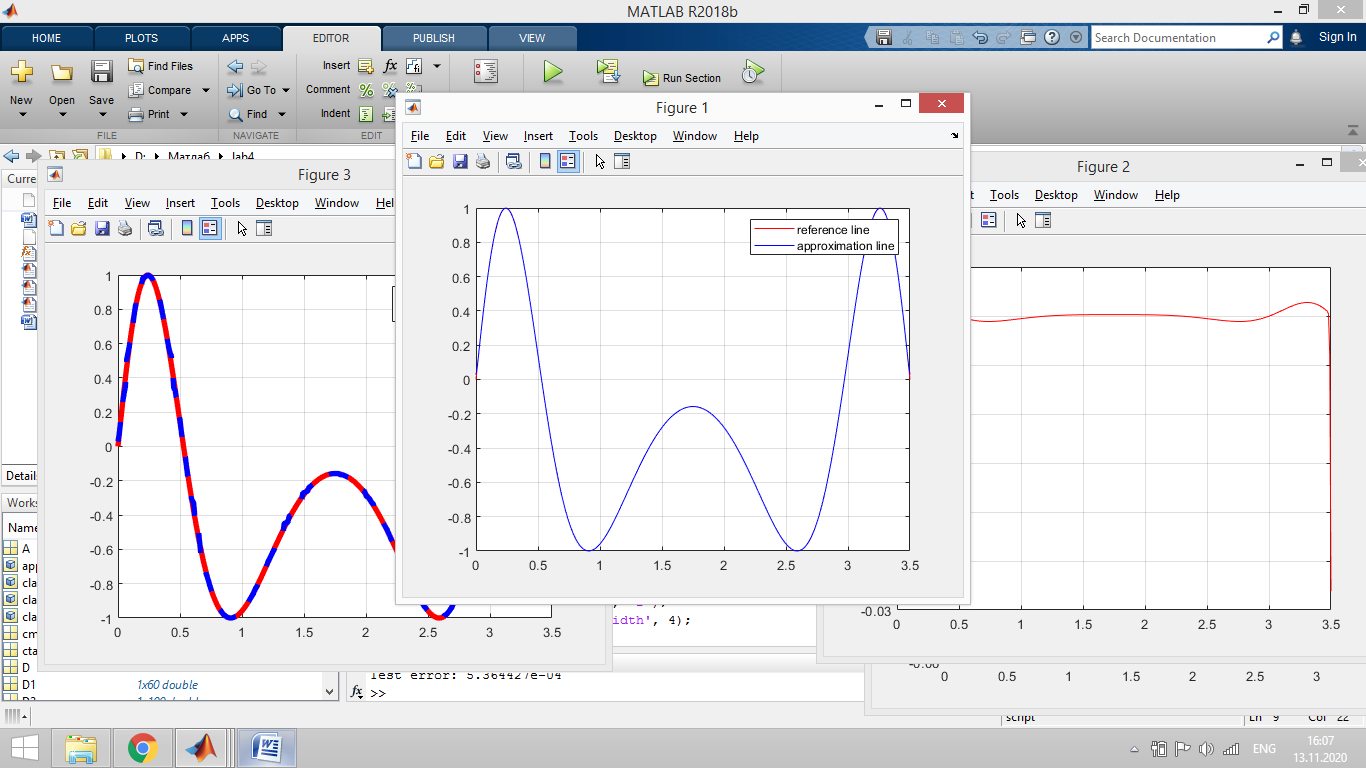
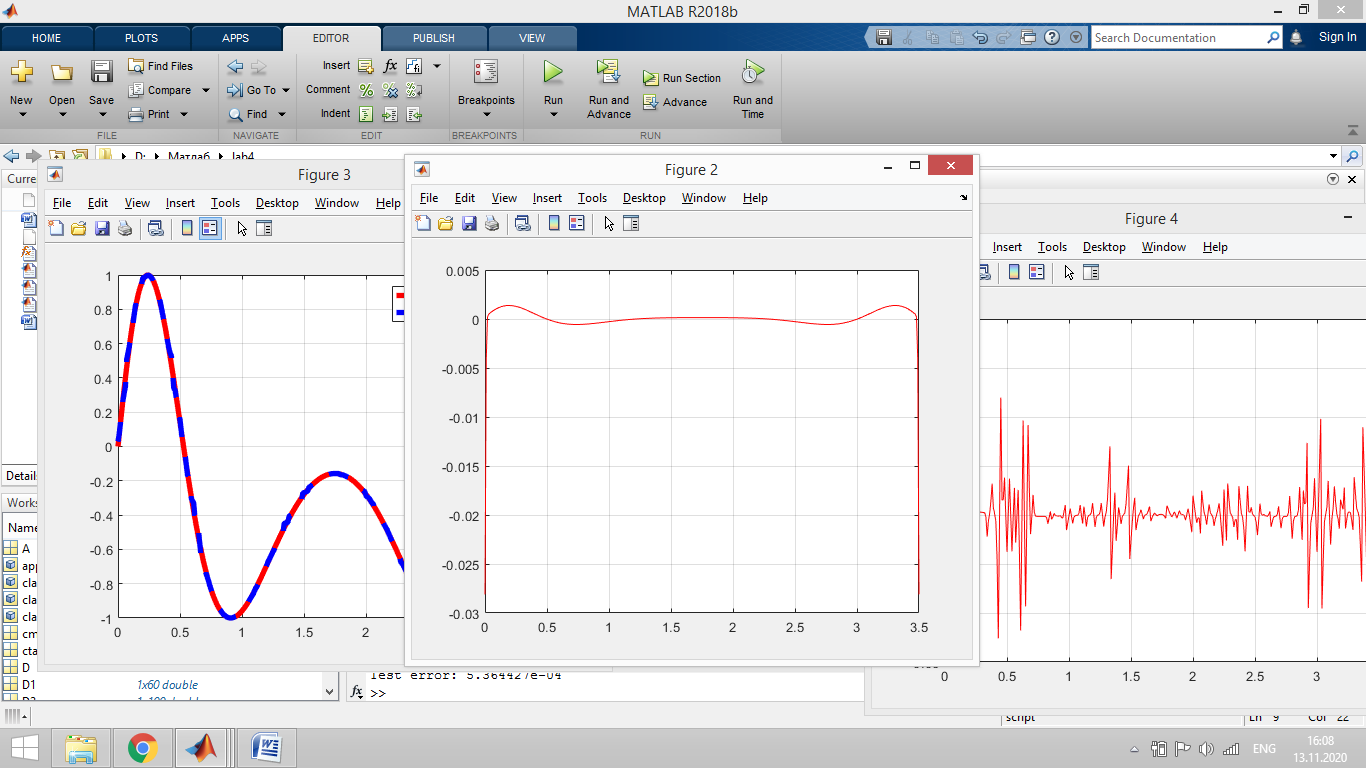
****

График ошибки

****

Сравнение эталонных значений с предсказанными

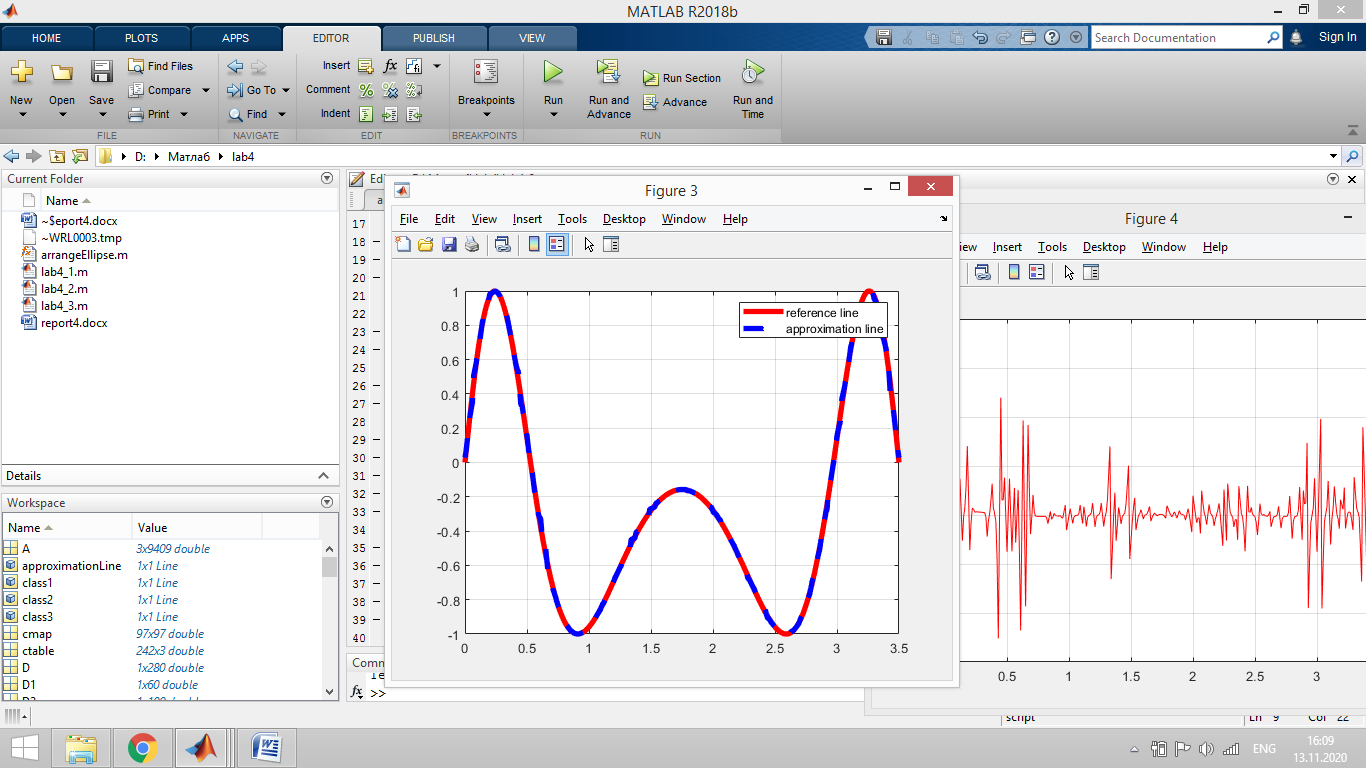
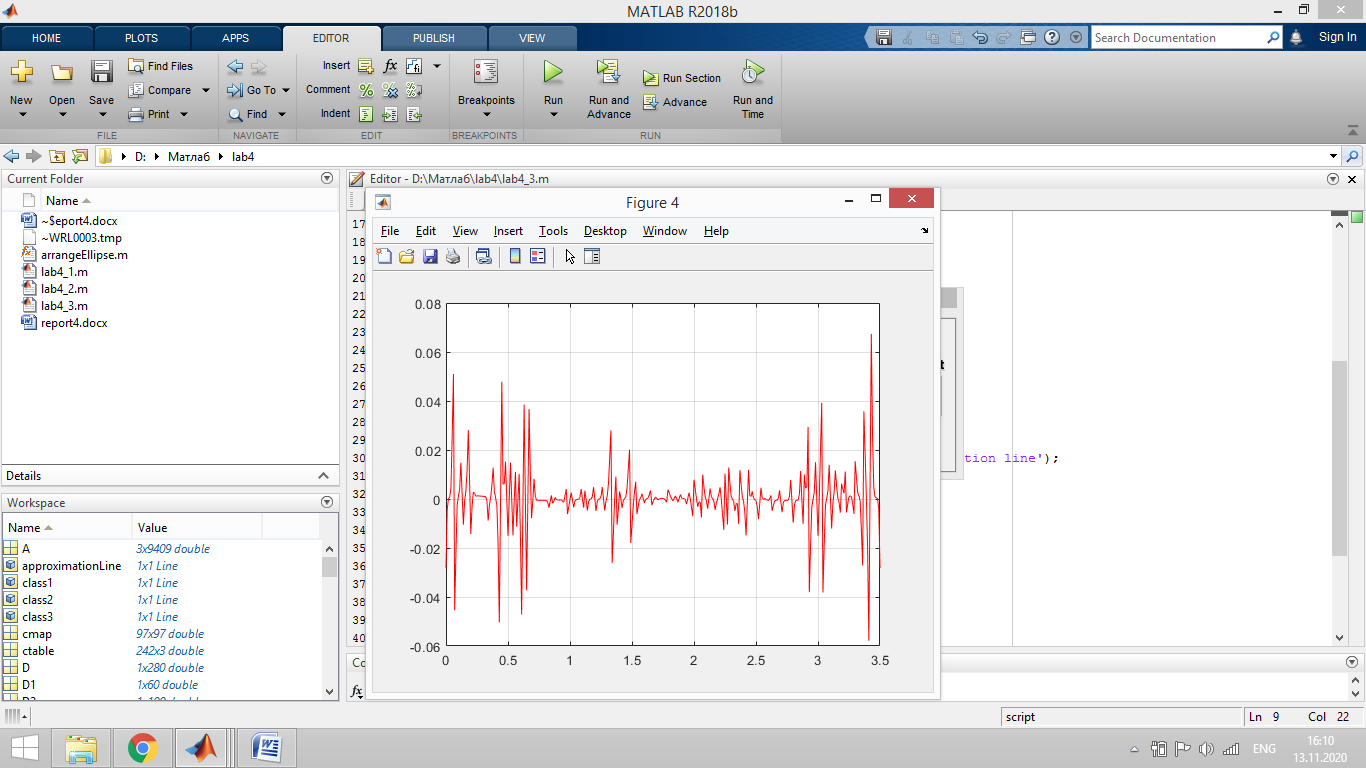
****

График ошибки

****