Pattern Recognition and Machine Learning

Assignment - 04

July 21, 2025

Name : Rohan Baghel
Student ID: 202116011

About

In this,we will study regularized least squares kernel regression model with the RBF kernel and implement it using Matlab.
kernel K(x,y) = exp(- ($\|x - y\|2/\sigma$).

kernel values are used to derive weights to predict outputs from given inputs. Steps involved to calculate weights and finally to use them in predicting output variable, y from predictor variable, x is explained in detail in the following sections. Let's start with an example to clearly understand how kernel regression works.

Kernel regression Model

Kernel regression is a non-parametric technique to estimate the conditional expectation of a random variable. The objective is to find a non-linear relation between a pair of random variables X and Y.

Q1

Generate 50 real numbers for the variable X from the uniform distribution U [0,1]

```
clear
clc
%-------------------------------------------------------------------------%
% Q1
X_train =  sort(rand(1,50));

Y_tr = sin(2*pi*X_train);

%-------------------------------------------------------------------------%
```

generating 50 training sets for X

Q2

Construct the training set T = (x 1 ,y 1),(x 2 ,y 2),.....,(x 50 ,y 50) using the relation $Y_i = sin(2(x_i)) + \varepsilon_i \qquad where \varepsilon_i \ N(0, 0.25) and x_i is from R.$

```
%-------------------------------------------------------------------------%
% Q2
%Normal Distribution
ND = makedist('Normal', 'mu',0,'sigma',0.25);
%Noise Vector for training set(50)
noise1 = random(ND,[1,50]);

Y_train = Y_tr + noise1;
%-------------------------------------------------------------------------%
```

Used Sine function to generate points with respect to X training points.

It is the set of data that is used to train and make the model learn the hidden features/patterns in the data.

In each epoch, the same training data is fed to the neural network repeatedly, and the model continues to learn the features of the data.

The training set should have a diversified set of inputs so that the model is trained in all scenarios and can predict any unseen data sample that may appear in the future.

For Testing set

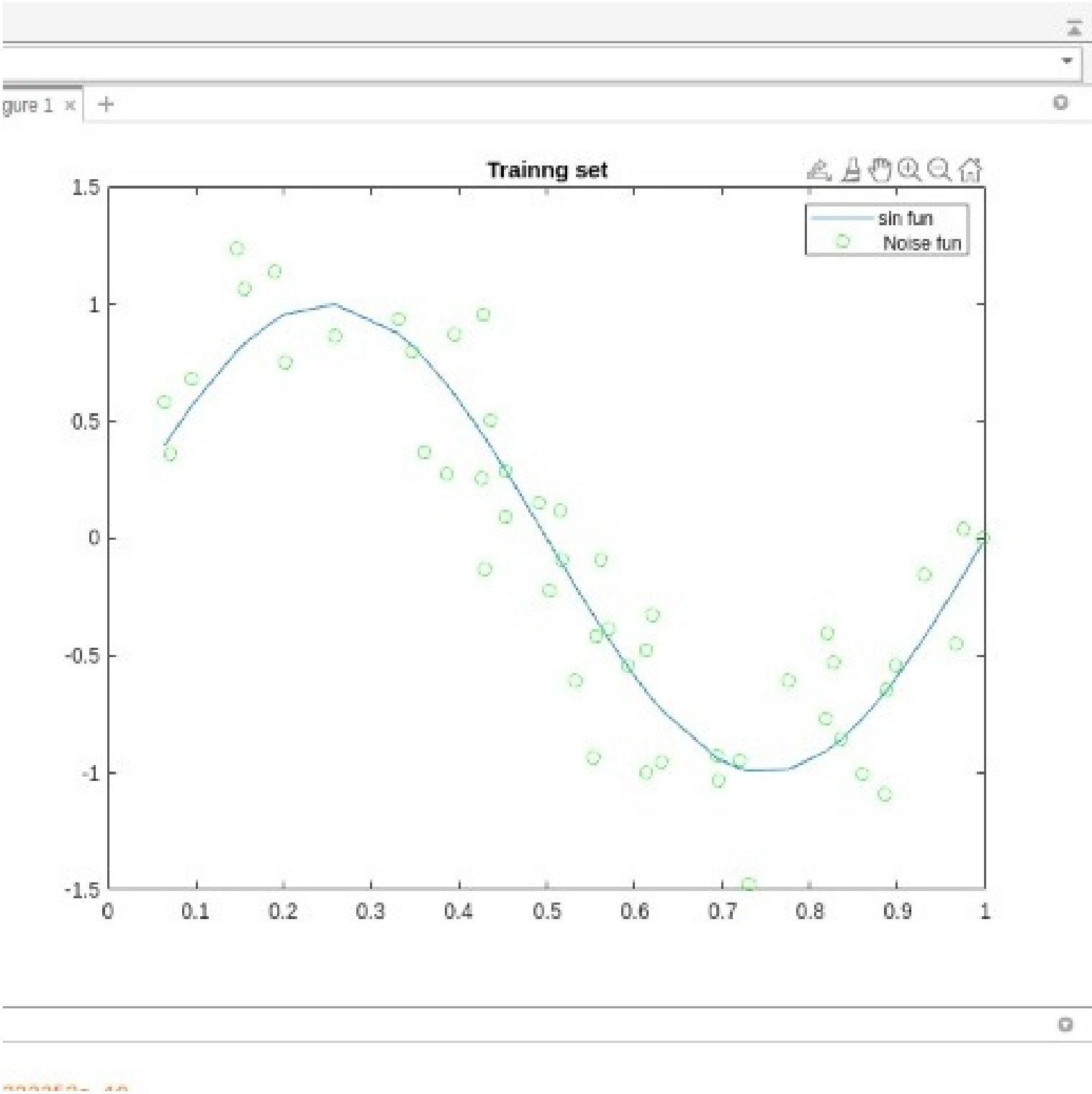

Figure 1: Training set

Graph shows:
line is foe the target function.
Blue dots are the points where we have added the noise on it.
50 points are chosen as training points

Q3

In the similar way construct a testing set of size 500
i,e. Test = (x' 1 ,y' 1),(x' 2 ,y' 2),.....,(x' 50 ,y' 50).

```
%-------------------------------------------------------------------------%
% Q3

X_test =   sort(rand(1,500));

Y_ts = sin(2*pi*X_test);
noise2 = random(ND,[1,500]);

Y_test = Y_ts + noise2;

%plot(X_test,Y_test,'o',X_test,Y_ts)

%-------------------------------------------------------------------------%
```

Generated 500 testing points.

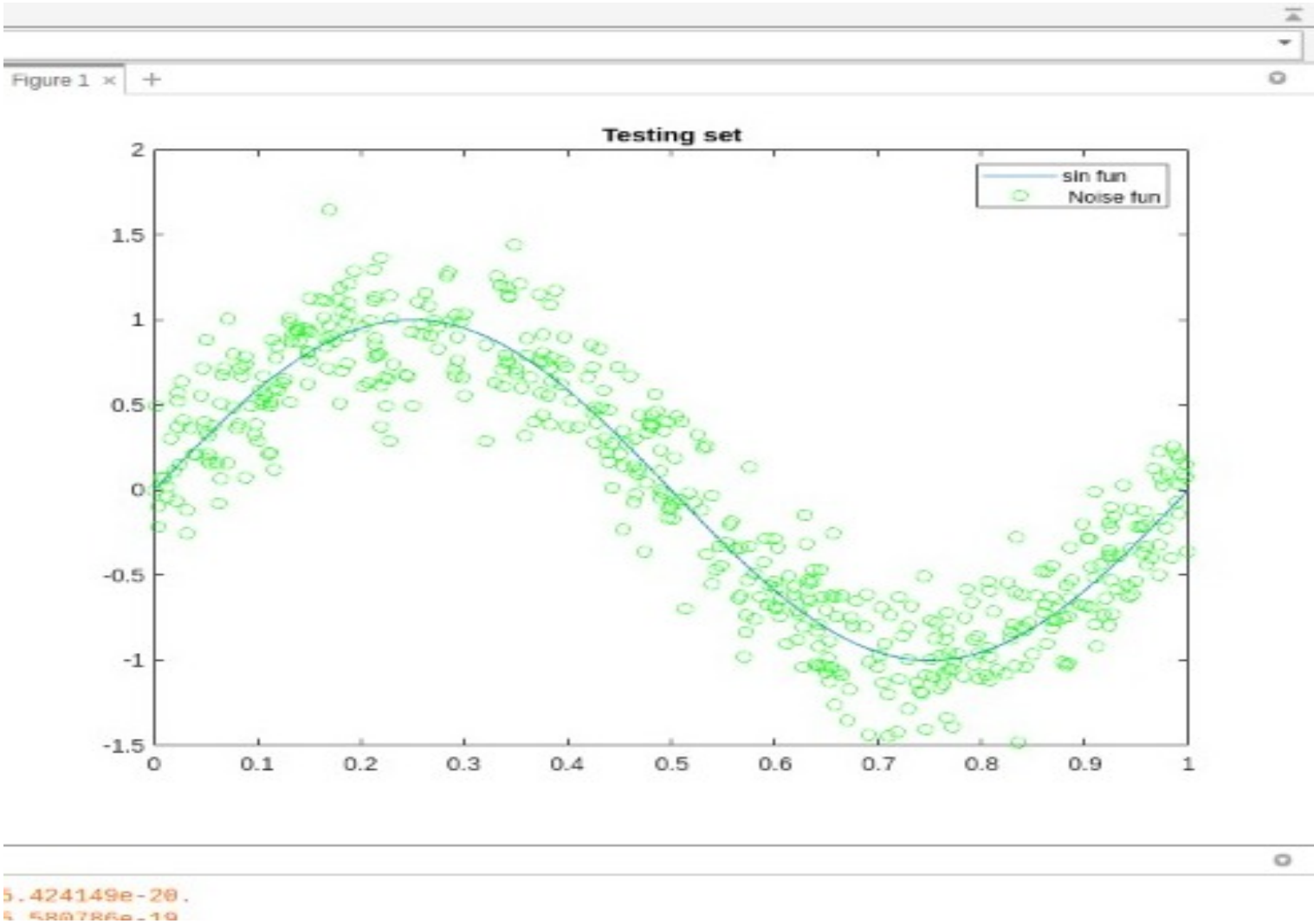

Figure 2: Test set

Q4

Estimate the Regularized Least Square kernel regression model by using the training set T. Plot the training estimate along with the original target function. Also, compute the Test RMSE.

```
%-------------------------------------------------------------------------%
% Q4

%-------------------------------------------------------------------------%
sigma = 0.02;
lambda = eps(-7);
Ker = KernelFun(X_train,X_train,sigma);

[U_train,R_train]= LSPR(Ker,Y_train,lambda);
% disp(size(R))
% disp(size(U))

Y_train_pred = R_train*U_train;
%disp(size(Y_train_pred))


%RMSE of testing
Ker2 = KernelFun(X_test,X_train,sigma);
[U_test,R_test]= LSPR(Ker2,Y_test,lambda);

Y_test_pred = R_test*U_test;


rmse_y_test = RMSE(Y_test',Y_test_pred);
disp("Rmse of test: " + rmse_y_test);

%ploting the graph

%plot(X_train,Y_tr,X_train, Y_train,'o',X_train, Y_train_pred,'go')
legend('sin fun',' Noise fun','Predicted val')
title('Normal set')

%-------------------------------------------------------------------------%
```

In this RBF kernel is used
using the kernel Regularized Least Square kernel regression model is performed in training set
RMSE value of test data is calculate.

```
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND =  2.833075e-19.
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND =  3.485267e-19.
Rmse of test: 0.50093
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND =  2.833075e-19.
```

Figure 3: RMSE of test

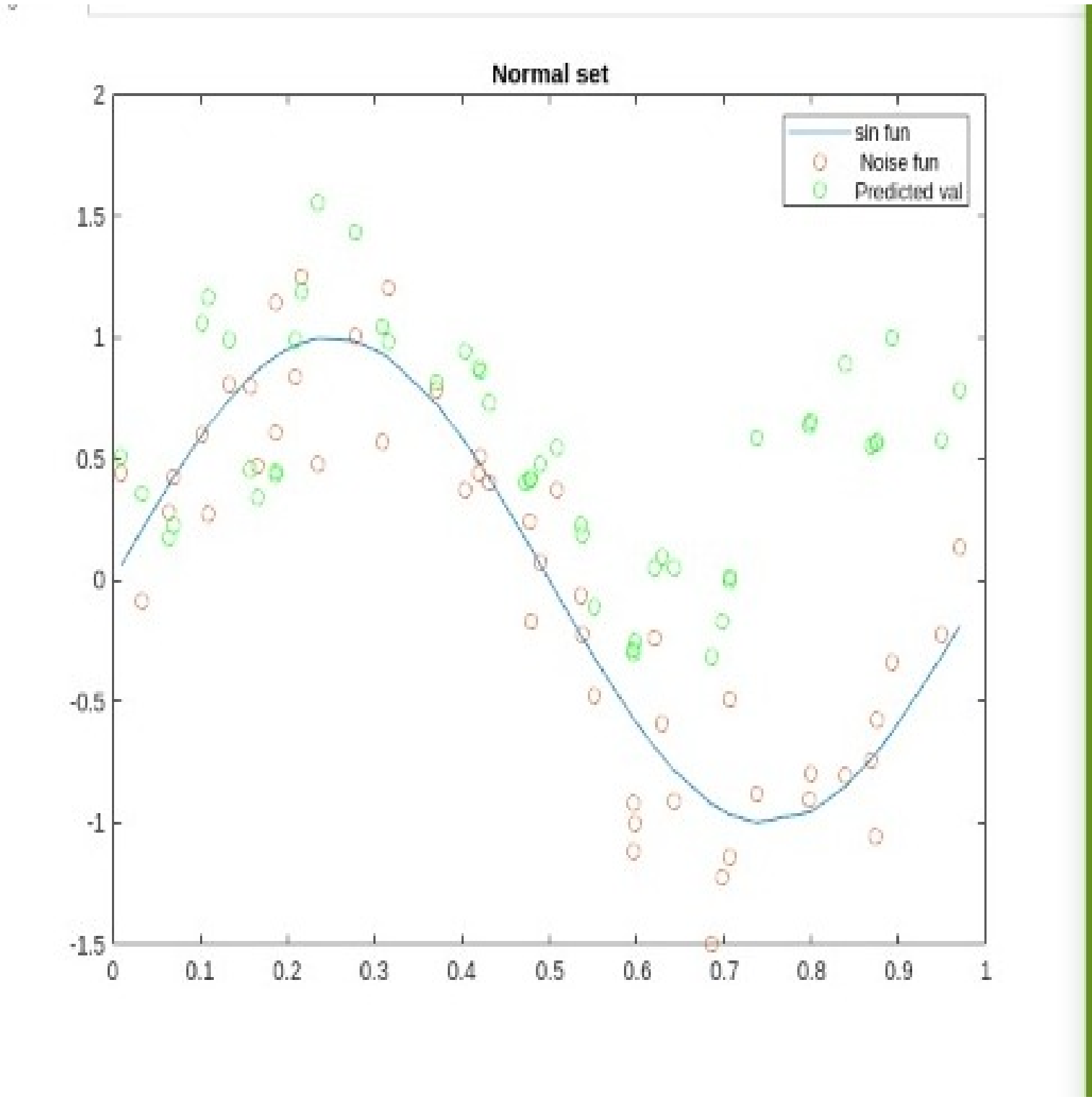

Figure 4: Test set

Predicted Value is bit aligned to the target value.
noise function is bit same to the predicted values.

Q5

Randomly select 5 training points to say x' 1 , x' 2 , x' 3 , x' 4 , x' 5 and scale(multiply) their corresponding y' 1 , y' 2 , y' 3 , y' 4 , y' 5 by 4. So these 5 points are to be considered as outliers. Estimate the Regularized Least Square kernel regression model by using the modified training set T. Plot the training estimate along with the original target function. Also, compute the RMSE.

```
%-------------------------------------------------------------------------%
% Q5
% selecyt random 5 points from traising set;

msize = numel(X_train);
X_modified = X_train(randperm(msize, 5));
%disp(ran)



% y respective to random points selected and multiplied with 4;
Y_rn = sin(2*pi*X_train);
noise3 = random(ND,[1,50]);
Y_modified = Y_rn + noise3;


lenT = size(X_train,2);
for i = 1:(lenT-1)
       for k = 1: (size(X_modified,2)-1)
           if (X_train(i) == X_modified(k))
               Y_modified(i) =  Y_modified(i)*4;
           end
       end
end

%Regularized Least Square kernel regression model by using the modified training...
    set T
Ker_mod = KernelFun(X_train,X_train,sigma);
[U_mod,R_mod]= LSPR(Ker_mod,Y_modified,lambda);

%RMSE Value
Y_modified_predict = R_mod*U_mod;
rmse_y_mod = RMSE(Y_modified',Y_modified_predict);
disp(rmse_y_mod)

%Ploting the Graph

%plot(X_train,Y_tr,X_train, Y_modified,'o',X_train, Y_modified_predict,'go')
%legend('sin fun',' Noise fun','Predicted val')
%title('Modified set')


%-------------------------------------------------------------------------%
```

```
irning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND =  4.113157e-19.
   0.9895
```

Figure 5: RMSE of modified set

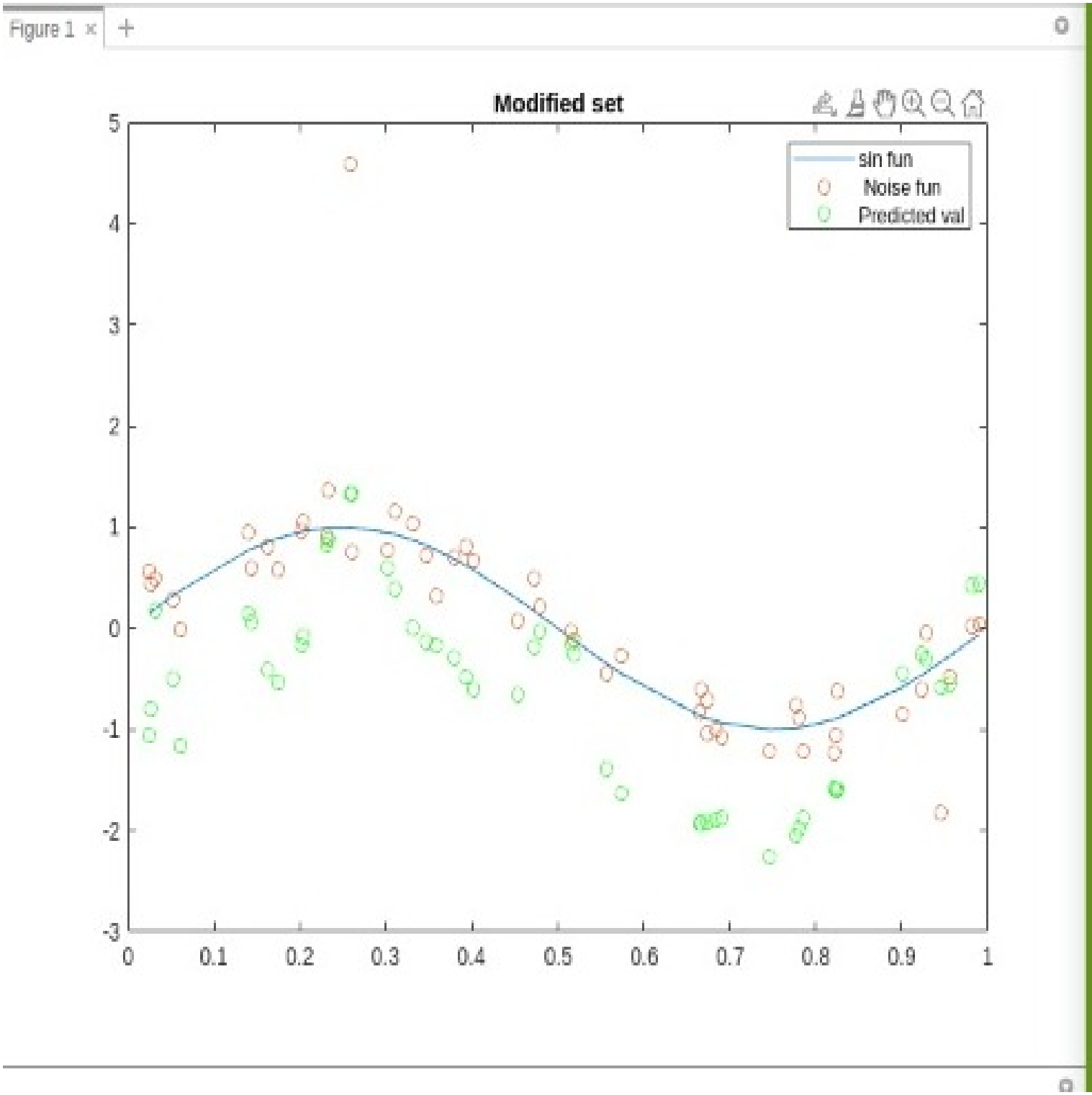

Figure 6: Modified training set

Function used

RBF kernel Function used.

```
%-------------------------------------------------------------------------%
function Ker = KernelFun(X_1,X_2,sigma)
        N1 = size(X_1,2);
        %disp(N1)
        N2 = size(X_2,2);
        %disp(N2)
        Ker= zeros(N1,N2);
        i=1;
        while i≤N1
            j=1;
            while j≤N2
                X_i = X_1(:,i);
                X_j = X_2(:,j);
                Norm2_sqr = sum((X_i-X_j).^2);
                Ker(i,j) = exp(-(Norm2_sqr)/sigma);
                j=j+1;
            end
            i=i+1;
        end
end
%-------------------------------------------------------------------------%
```

To calculate RMSE value Function used.

```
%-------------------------------------------------------------------------%
function rmse = RMSE(y,y_pred)
    N = size(y,1);
    rmse = sqrt(mean((y_pred-y).^2));
end
%-------------------------------------------------------------------------%
```

Least square regression model Function used.

```
%-------------------------------------------------------------------------%
function [U,R]= LSPR(K,Y_,lam)
    R =  [K ones(size(K,1),1)];
    %disp(size(Y_))
    %disp(size(R))
    ide = eye(size(R,2));
    %disp(size(ide))
    U = inv((R'*R)+(lam*ide))*((R')*(Y_'));
end
%-------------------------------------------------------------------------%
```

Q6

Observation of 1-5

We have observed that:

When we use simple least square kernel model estimators of parameters have always turned out to be the best linear unbiased estimators.
However if we use the outliers, not much but it affect the least square kernel regression model.

As when we are using the noraml least square kernel regression model than
RMSE vlaue is near to 0.50093

And when when we are using outliers in least square kernel regression model than
RMSE value is near to 0.9095

AS the RMSE value of normal least square regression model is less than least square kernel model using outliers. hence normal Least square performs well.

Q7

Repeat the experiment (1-5) for the L1-norm loss kernel regression model. For obtaining the solution of the L1-norm loss kernel regression model, you need to use the gradient descent or stochastic gradient descent algorithm. Write your observation regarding the performance of L1- norm kernel Regression model in presence of outliers.

```
clear
clc

X_train =  sort(rand(1,50));
norm_x1 =sqrt(X_train.^2+X_train.^2);
Y_tr = sin(2*pi*norm_x1);

ND = makedist('Normal', 'mu',0,'sigma',0.25);
%Noise Vector for training set(50)
noise1 = random(ND,[1,50]);

Y_train = Y_tr + noise1;
X_test =   sort(rand(1,500));

norm_x2 =sqrt(X_test.^2+X_test.^2);
Y_ts = sin(2*pi*norm_x2);
noise2 = random(ND,[1,500]);

Y_test = Y_ts + noise2;

%plot(X_test,Y_test,'o',X_test,Y_ts)

%-------------------------------------------------------------------------%
% Q4

sigma = 0.02;
lambda = eps(-7);
Ker = KernelFun(X_train,X_train,sigma);
R =  [Ker ones(size(Ker,1),1)];
L1 = Norm_of_L1(lambda, U, R, Y_train);
Grad_vac1 = GD_Algo(l1,U,A,Y);


Y_train_pred = R*Grad_vac;
%disp(size(Y_train_pred))


%RMSE of testing
Ker2 = KernelFun(X_test,X_train,sigma);
R =  [Ker ones(size(Ker,1),1)];
L1 = Norm_of_L1(lambda, U, R, Y_test);
Grad_vec2 = GD_Algo(l1,U,A,Y_test);

Y_test_pred = R_test*Grad_vec;


rmse_y_test = RMSE(Y_test',Y_test_pred);
disp("Rmse of test: " + rmse_y_test);

```

```
%ploting the graph

% plot(X_train,Y_tr,X_train, Y_train,'ro',X_train, Y_train_pred,'go')
% legend('Sin Fun','Sin fum with Noise','Predict_Sin')
% title('Normal Set')

%-------------------------------------------------------------------------%
% Q5
% selecyt random 5 points from traising set;
msize = numel(X_train);
X_modified = X_train(randperm(msize, 5));
%disp(ran)

norm_x3 =sqrt(X_modified.^2+X_modified.^2);
Y_rn = sin(2*pi*norm_x3);
noise3 = random(ND,[1,5]);
Y_modified = Y_rn + noise3;


Ker3 = KernelFun(X_modified,X_modified,sigma);
R_modified = [Ker3 ones(size(Ker3,1),1)];
L1 = Norm_of_L1(lambda, U, R_modified, Y_modified);
Grad_vec2 = GD_Algo(l1,U,A,Y_modified);


Y_modified_predict = R_mod*U_mod;
rmse_y_mod = RMSE(Y_modified',Y_modified_predict);
disp(rmse_y_mod)

%Ploting the Graph

% plot(X_train,Y_tr,X_modified, Y_modified,'ro',X_modified, Y_modified_pred,'go...
    ')
% legend('Sin Fun','Sin fum with Noise','Predict_Sin')
% title('Normal Set')

%-------------------------------------------------------------------------%
```

In this L1-norm loss kernel regression model is used as.
and other thing are same as previous.

Function used

RBF kernel Function used.

```
%-------------------------------------------------------------------------%
function Ker = KernelFun(X_1,X_2,sigma)
        N1 = size(X_1,2);
        %disp(N1)
        N2 = size(X_2,2);
        %disp(N2)
        Ker= zeros(N1,N2);
        i=1;
        while i≤N1
            j=1;
            while j≤N2
                X_i = X_1(:,i);
                X_j = X_2(:,j);
                Norm2_sqr = sum((X_i-X_j).^2);
                Ker(i,j) = exp(-(Norm2_sqr)/sigma);
                j=j+1;
            end
            i=i+1;
        end
end

%-------------------------------------------------------------------------%
```

RMSE value calculate Function used.

```
%-------------------------------------------------------------------------%
function rmse = RMSE(y,y_pred)
    N = size(y,1);
    rmse = sqrt(mean((y_pred-y).^2));
end
%-------------------------------------------------------------------------%
```

Used Gradient Decent and L1 Norm Function used.

```
%-------------------------------------------------------------------------%
function grad_vac = GD_Algo(l1,U,A,Y)
         grad_vac = (l1*U)-(A'*(Y-A*U));
end
%-------------------------------------------------------------------------%
function L1_Norm = Norm_of_L1(lambda, U, K, Y)
    L1_Norm = (lambda/2)*(U'*U) + Norm(Y-K*U);
end
%-------------------------------------------------------------------------%
```

Observed

RMSE value is : 0.5016 testing
RMSE value is : 0.974 training

L1 Norm is the sum of the magnitudes of the vectors in a space. It is the most natural way of measure distance between vectors, that is the sum of absolute difference of the components of the vectors. In this norm, all the components of the vector are weighted equally.

AS when we use L1-norm loss kernel regression model. some changes is been observed.

We can observe that RMSE value for the testing is approx same when we use least square kernel regression mode.
But for the training set RMSE value is high as we used outliers and outliers can affect the outcomes.