113EC0199 Sagnik Basu

Problem :

$z=\sin(x)\cos(y)$ where $-\pi/2 \leq x, y \leq \pi/2$

RBF network consist of two layers of computing elements. The first layer is a non-linear layer consisting of centres characterized by its position, an activation function and spread parameter. This is also called the hidden layer. The output layer (the second layer) is a linear combiner.

Solution :

```
clc;

clear all;

% data input

 constant=100;
x=(pi/2)*rand(1,constant)-(pi/2); %training table
y=(pi/2)*rand(1,constant)-(pi/2);

inp=zeros(constant,2);

% for i=1:length(x)
%     inp(i,1)=x(i);
%     inp(i,2)=y(i);
% end
%
% thr=0.000001;

output=sin(pi*x).*cos(pi*y); %desired output


%Assign 9 random centres

centre_x = (pi/2)*rand(1,9)-(pi/2);
centre_y = (pi/2)*rand(1,9)-(pi/2);
centre_x_old=centre_x;
centre_y_old=centre_y;

%Update centre weights
i=1;
j=1;
rate=0.3;
```

```
weights=[-1;-2;-3;-4;-5;-6;-7;-8;-9];
G_train= zeros(constant,9);

for k=1:10
for i =1:constant
    min=abs(centre_x(j)-x(i));
   for j=1:9
      d=abs(centre_x(j)-x(i));
      if(d < min)
         %centre_x(i)=centre_x(i) + rate*(x(j)-centre_x(i));
         nearest=centre_x(j);
         min=d;
         centre_x(j)=centre_x(j)+rate*(x(i)-centre_x(j));
      end
   end

end

for i =1:constant
    min=abs(centre_y(j)-y(i));
   for j=1:9
      d=abs(centre_y(j)-y(i));
      if(d < min)
         %centre_x(i)=centre_x(i) + rate*(x(j)-centre_x(i));
         nearest=centre_y(j);
         min=d;
         centre_y(j)=centre_y(j)+rate*(y(i)-centre_y(j));
      end
   end

end

% RBF Network

inputs = zeros(1,9);
HL= zeros(1,9);
phi = zeros(100,9);
output_train=zeros(1,100);
for i=1:constant
   for j=1:9
      phi(i,j)=exp(-(((x(i)-centre_x(j))^2)+(y(i)-centre_y(j))^2))/2;
      output_train(i)=output_train(i)+weights(j)*exp(-(((x(i)-centre_x(j))^2)+(y(i)-
centre_y(j))^2))/2;
   end
   error=abs(output(i)-output_train(i));
   error_sq=error*error;
   MSE(k)=error_sq/100;
end


% LMS algorithm

A=phi'*(phi);
B=inv(A);
C= B*phi';
weights=C*output';

end
% Testing

test_constant=1600;
x_test=(pi/2)*rand(1,test_constant)-(pi/2); %testing table
```

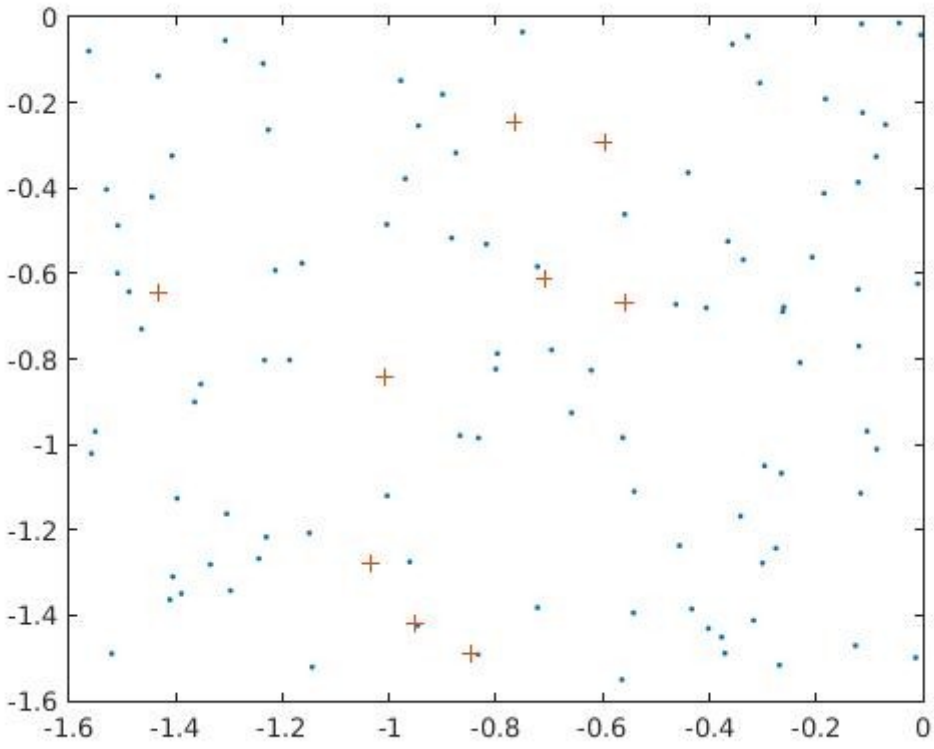
```
y_test=(pi/2)*rand(1,test_constant)-(pi/2);

output_des=sin(pi*x_test).*cos(pi*y_test); %desired output

%Assign 9 random centres

centre_x = (pi/2)*rand(1,9)-(pi/2);
centre_y = (pi/2)*rand(1,9)-(pi/2);
centre_x_old=centre_x;
centre_y_old=centre_y;

%Update centre weights
i=1;
j=1;
rate=0.03;

for i =1:test_constant
     min=abs(centre_x(j)-x_test(i));
    for j=1:9
        d=abs(centre_x(j)-x_test(i));
        if(d < min)
            %centre_x(i)=centre_x(i) + rate*(x(j)-centre_x(i));
            nearest=centre_x(j);
            min=d;
            centre_x(j)=centre_x(j)+rate*(x_test(i)-centre_x(j));
        end
    end

end

for i =1:test_constant
     min=abs(centre_y(j)-y_test(i));
    for j=1:9
        d=abs(centre_y(j)-y_test(i));
        if(d < min)
            %centre_x(i)=centre_x(i) + rate*(x(j)-centre_x(i));
            nearest=centre_y(j);
            min=d;
            centre_y(j)=centre_y(j)+rate*(y_test(i)-centre_y(j));
        end
    end

end


% Output values
output_test = zeros(1,test_constant);
%G= zeros(constant,9);
for i=1:test_constant
    for j=1:9
        output_test(i)=output_test(i)+weights(j)*exp(-(((x_test(i)-centre_x(j))^2)+(y_test(i)-
centre_y(j))^2))/2;
    end
    %output_test(i)=G(i,j);
    %error(
end


% plot values

%  plot(x_test,y_test,'.');
%  hold on;
```

```
%  plot(centre_x,centre_y,'+');
scatter3(x_test,y_test,output_des);
hold on;
scatter3(x_test,y_test,output_test);
```

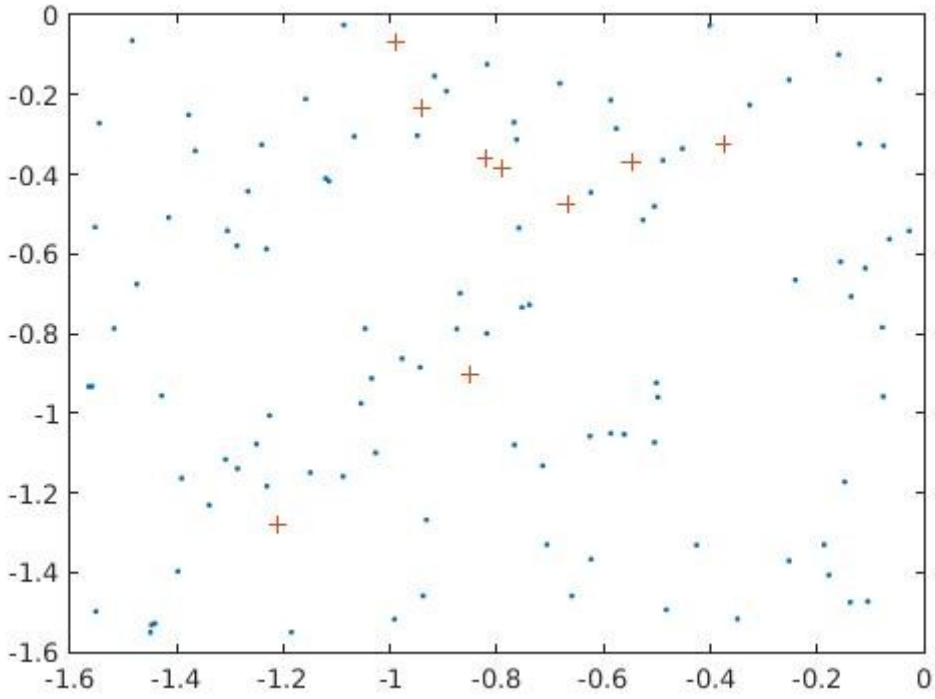
Results :

1) Centre update using KNN algorithm

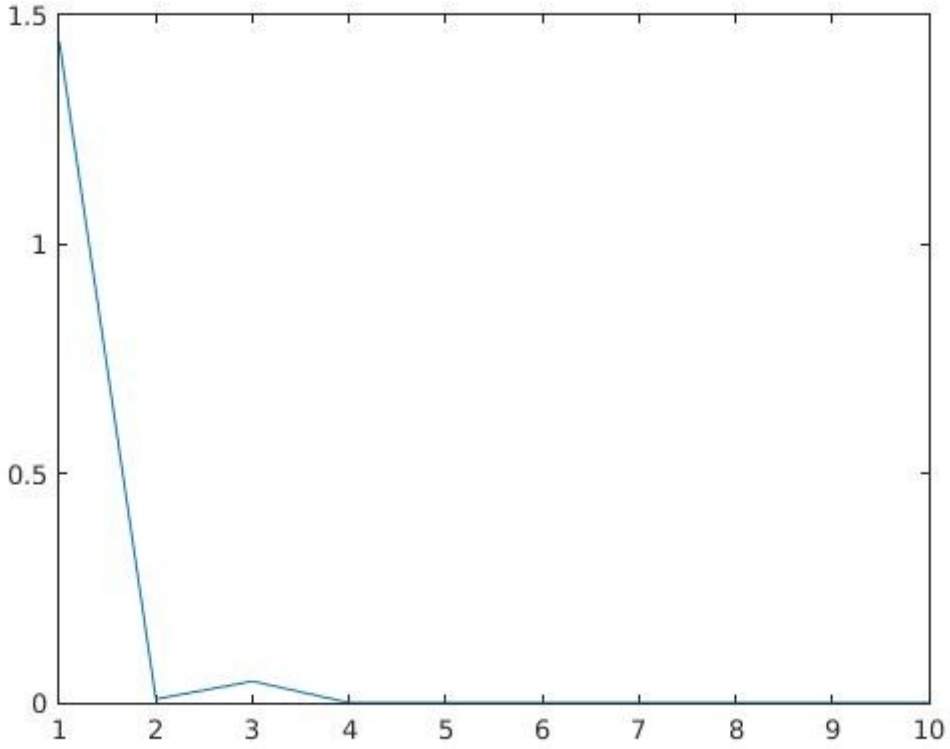
The figure below show random initialisation of weights without using any kind of update algorithm.The '+' symbols in red are the centre points while rest are data points.



After running a online KNN algorithm the centres are updated as given below :



The MSE is calculated in the training data set for 10 iterationsThe MSE is stored in MSE array . Here the x- axis is number of iterations while y-axis is the error :



Finally the RBF network is tested with training data set of 1600 points.

The desired function is given in blue points .

The function from RBF output is given in red points.