OE6020 – Meshfree methods applied to hydrodynamics

Assignment No 2 (Coding Tutorial - 1)

Roll No: AM17S015

Name : Yewale Nikhil Janardan

1) MQ-PIM

Consider domain of 25 x 25 nodes regularly and evenly distributed with dx = dy = 0.5 as shown in Figure 1. Using MQ-PIM for following parameters, we get RPIM-MQ shape functions and their derivatives as shown in Figure 2, 3 and 4.Polynomial basis function used is [1 ,x ,y] i.e m = 3

$$\alpha_c = 2, d_c = 0.5, q = \pm 0.5$$

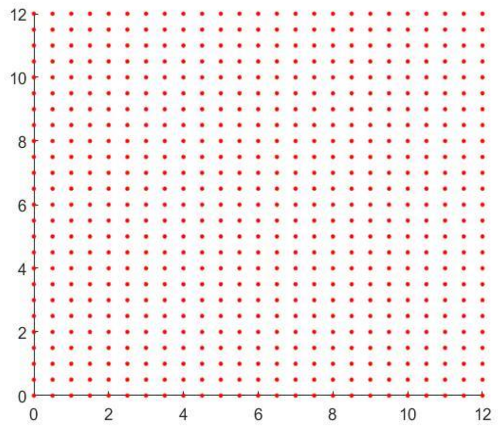


Figure 1

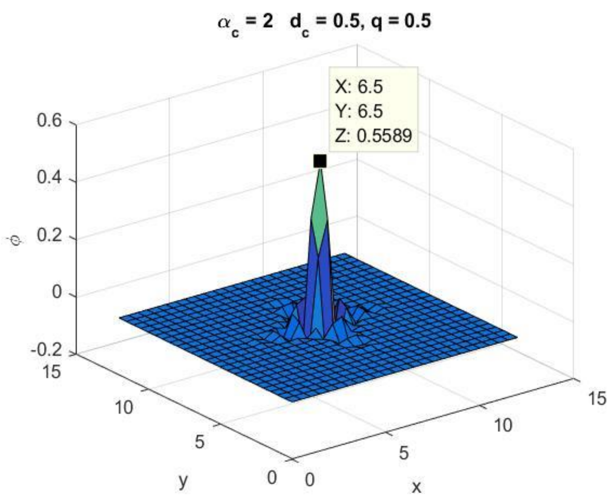


Figure 2

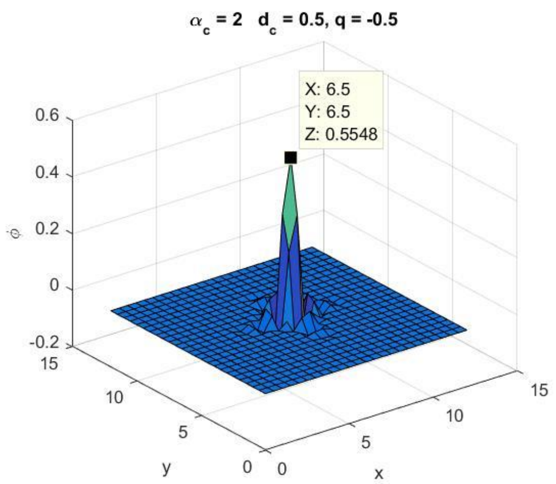


Figure 3

Here we plot shape function in Figure 2 and Figure 3 and it's derivatives for sampling point x = 6.3, y= 6.3. Highest peak can be observed in values of shape function at its nearest nodal point x = 6.5, y = 6.5. There is very little difference in the shapes of the shape function of q = 0.5 and q = -0.5 .Notice the difference in peak value of shape function in Figure 2 and Figure 3.

Similarly, we plot the derivatives of shape function with respect to x and y direction in Figure 4,5 and Figure 6,7 respectively for q = ± 0.5. Positive peaks and negative peaks are observed as shown in figures at x = 6.5, y = 6.5 and x = 6.5, y = 6 respectively. Notice the difference in peak value of derivatives of shape function in for q = ± 0.5.

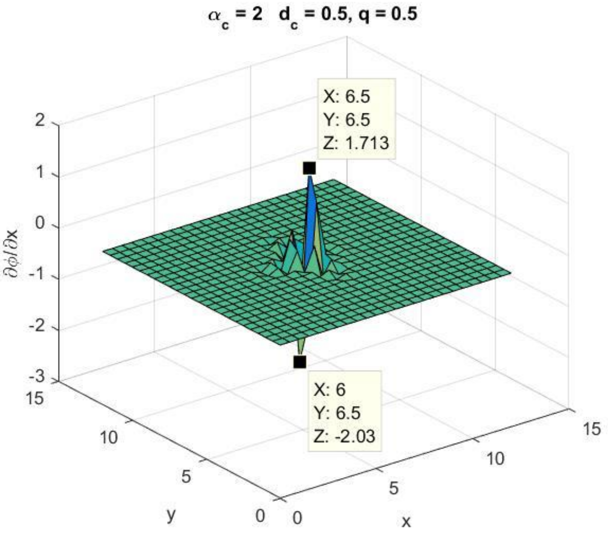


Figure 4

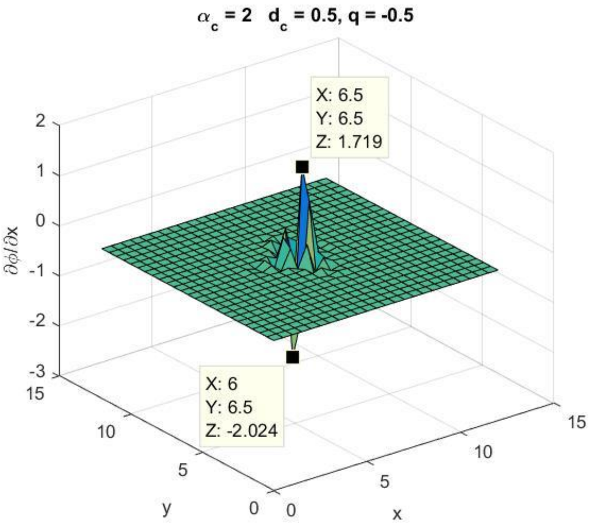


Figure 5

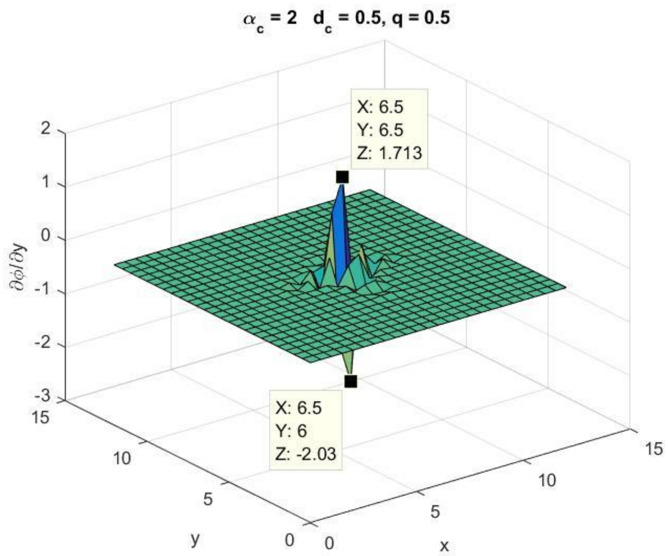


Figure 6

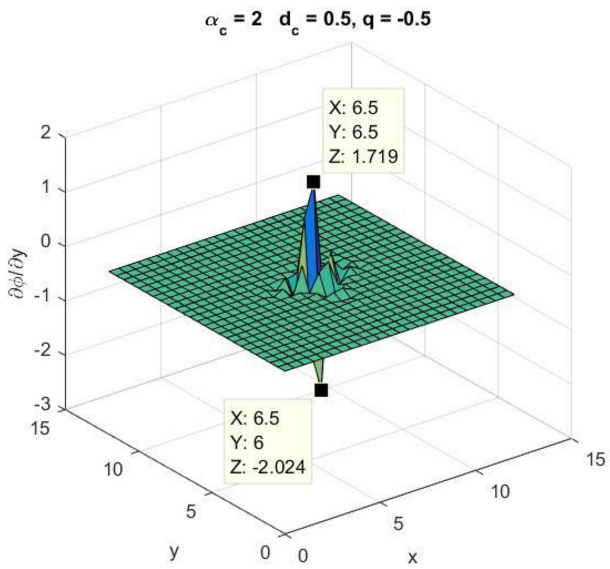
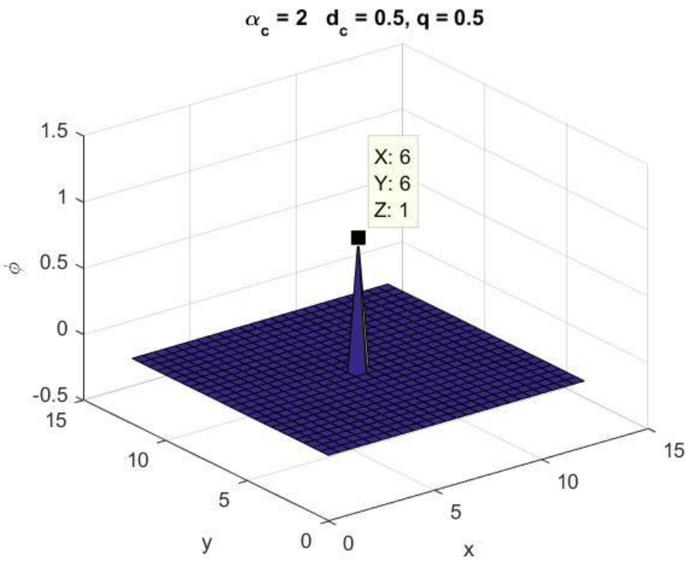


Figure 7

Kronecker-delta property for MQ-PIM: For this we choose sampling point as x = 6, y = 6 (or any nodal point from Figure 1) and plot shape function distribution in Figure 8(for q = 0.5) and Figure 9(for q = -0.5) , we notice $\phi = 1$ at this point and is zeros everywhere else, thus satisfying Kronecker-delta property.



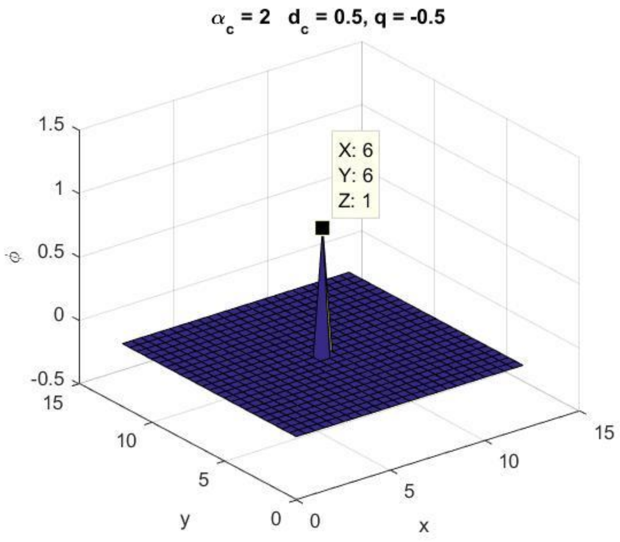
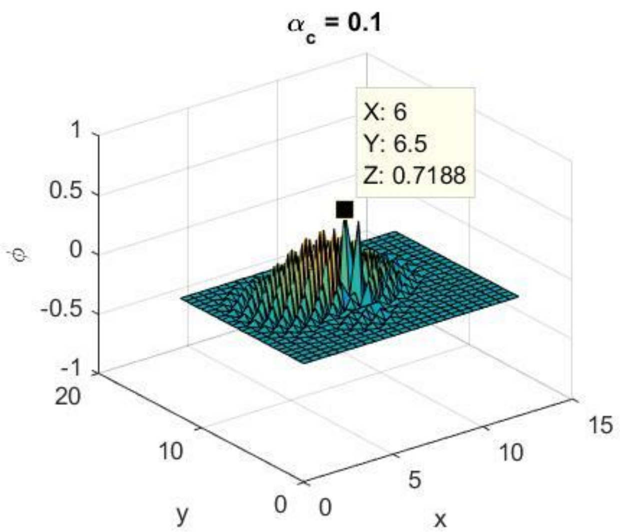


Figure 8

Figure 9

2) EXP – PIM : Using EXP-PIM for following parameters, we get RPIM-EXP shape functions and their derivatives as shown in Figure 10-18. Polynomial basis function used is [1 ,x, y] i.e. m = 3.

$$\alpha_c = 0.1, \alpha_c = 0.35, \alpha_c = 0.5$$



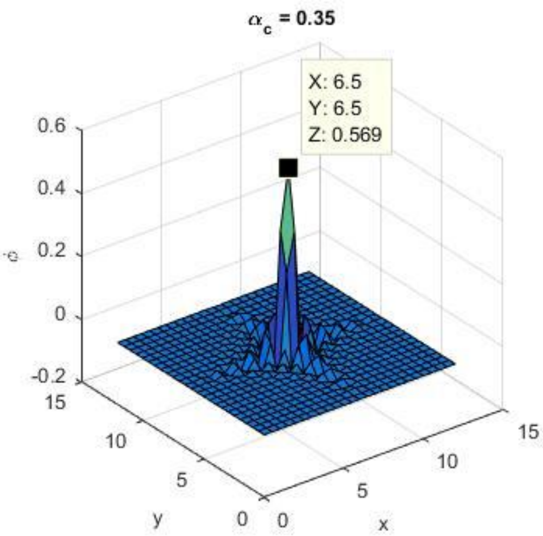


Figure 10

Figure 11

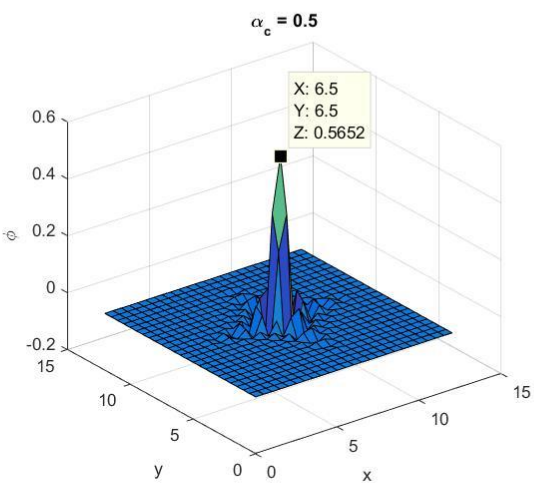


Figure 12

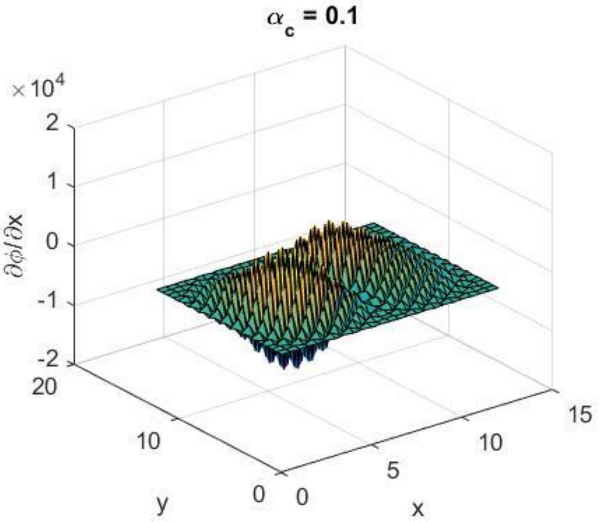


Figure 13

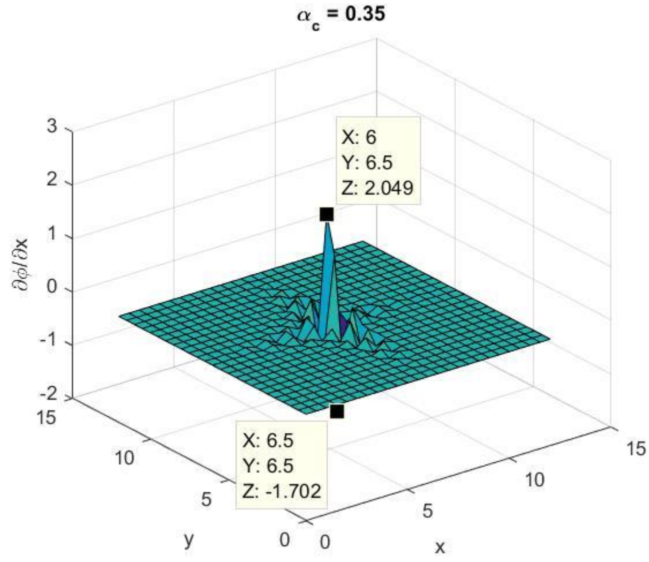


Figure 14

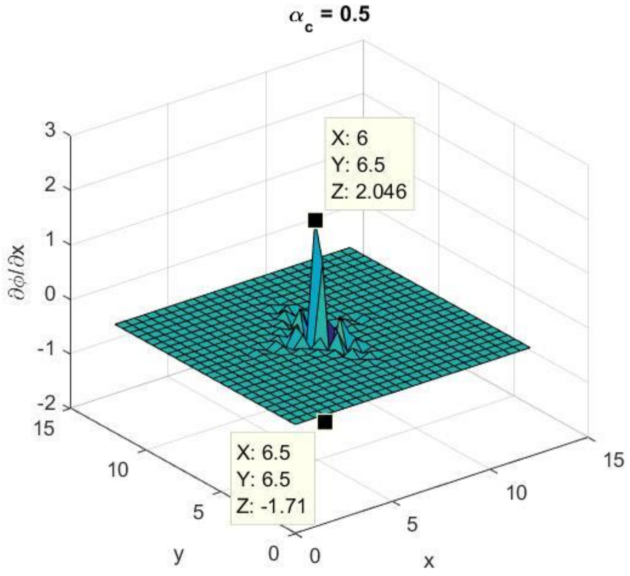


Figure 15

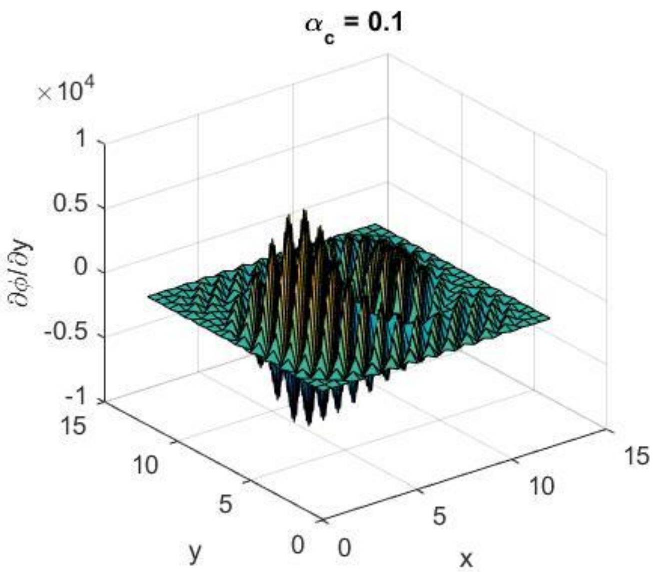


Figure 16

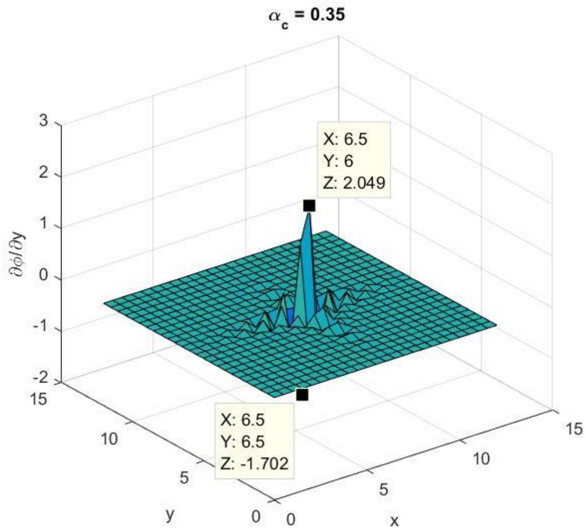


Figure 17

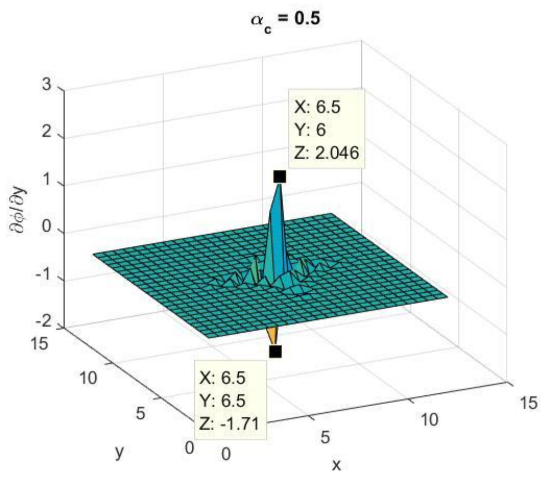


Figure 18

Notice the variation of shape function and it's derivatives in Figure 10,13 and 16 respectively for parameter $\alpha_c = 0.1$. Apart from peak values, shape function and it's derivative show values in neighbourhood of peak ,thus showing less accurate representation. For higher values of α_c however, shape functions and its derivatives tend to be more accurate.

Kronecker-delta property for EXP-PIM: For this we choose sampling point as x = 6.5, y = 6.5 (or any nodal point from Figure 1) and plot shape function distribution in Figure 19($\alpha_c = 0.1$), Figure 20($\alpha_c = 0.35$),Figure 21($\alpha_c = 0.5$), we notice $\phi = 1$ at this point and is nearly zero everywhere else, thus satisfying Kronecker-delta property.

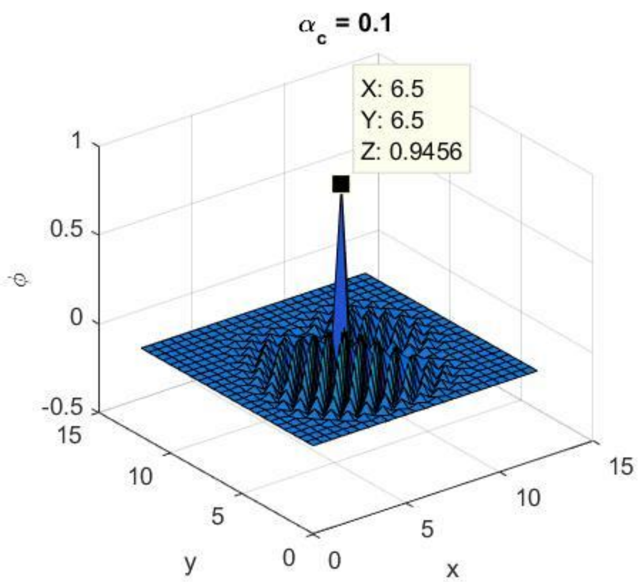


Figure 19

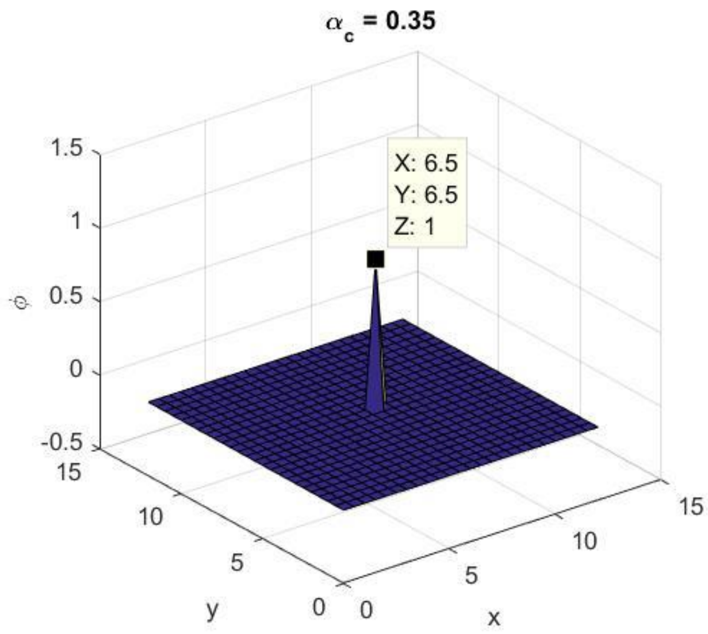


Figure 20

Observe Figure 19 however, for $\alpha_c = 0.1$ doesn't accurately depict Kronecker-delta behaviour. This inaccuracy is observed in lower values of α_c for EXP-PIM method.

NOTE: For partition of unity property of MQ-PIM and EXP-PIM for their various cases , please refer Appendix B.

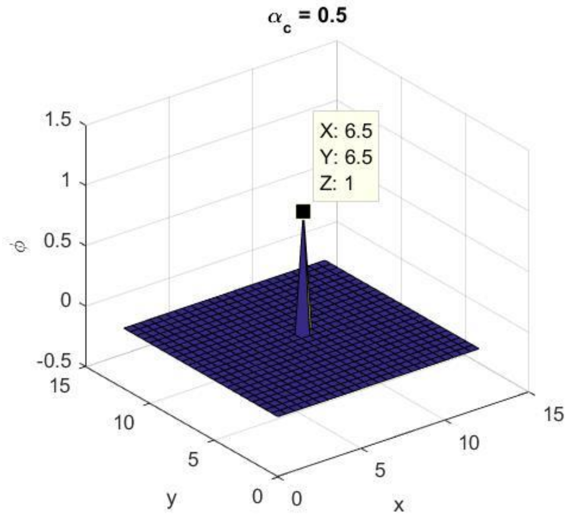


Figure 21

2D plots: Variation of shape function and its derivatives along y = 6.5 line in the domain, when shape function and its derivatives are calculated at all nodal points for a sampling point x = 6.5 and y = 6.5. Following Figures (22, 23 and 24) depict shape function and its derivatives variation along y = 6.5 using EXP-PIM for $\alpha_c = 0.35$ *and* $\alpha_c = 0.5$.

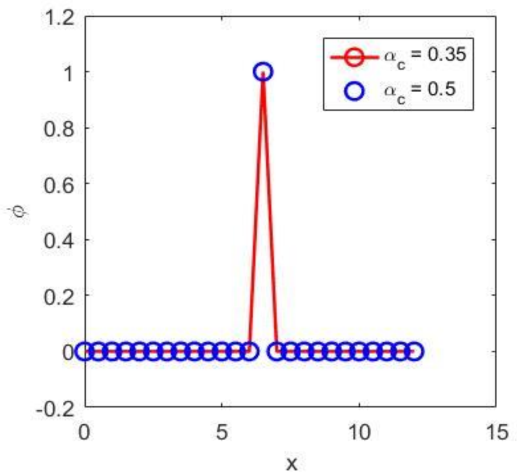


Figure 22

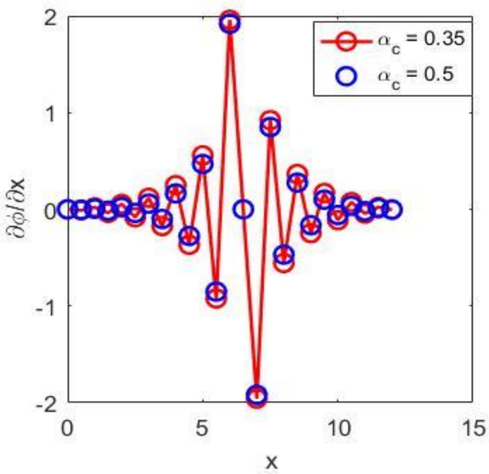


Figure 23

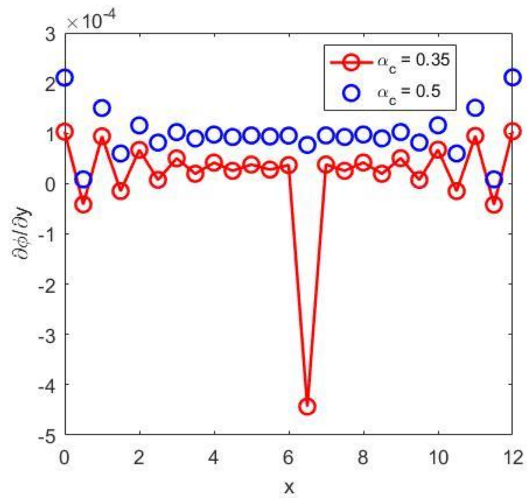


Figure 24

Values of shape function and its derivatives agree closely for $\alpha_c = 0.35$ *and* $\alpha_c = 0.5$ .Accuracy of EXP-PIM increases for increase in α_c. Figures (25, 26 and 27) depict shape function and its derivatives variation along y = 6.5 using MQ-PIM for $\alpha_c = 2, d_c = 0.5, q = \pm 0.5$

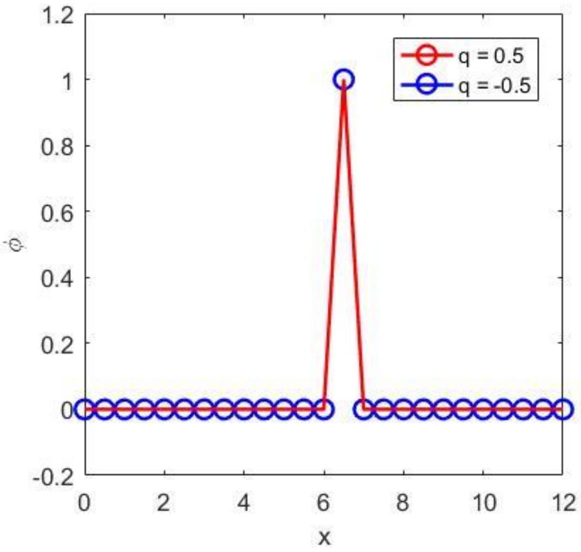


Figure 25

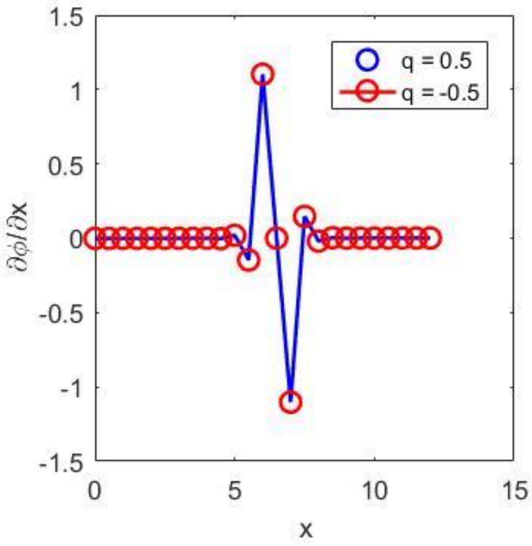


Figure 26

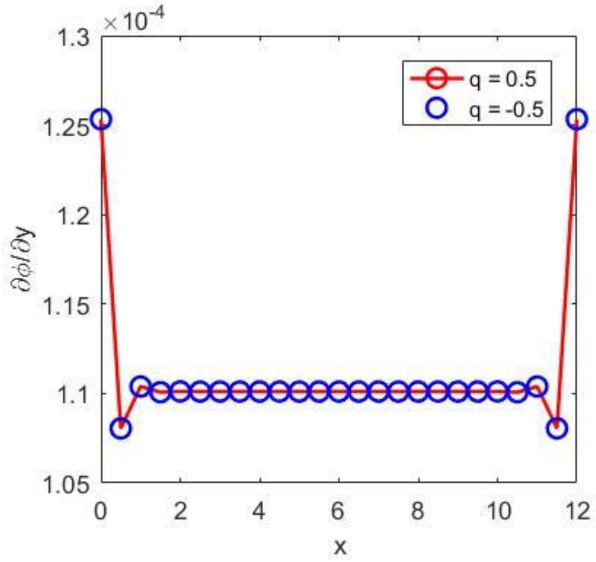


Figure 27

Appendix A – MATLAB Code for Radial- PIM Method

Rpim.m is main file

```
clc;

Nx=25; Ny=25;  % number of nodal cooridnates in x and y direction
dx =0.5; dy =0.5;    Lx = dx*(Nx-1) ; Ly = dy*(Ny-1);

% parameters for MQ-PIM/EXP-PIM----------
dc = 0.5;
alphac = 2;
q = 0.5;
%-----------------------------------------------
Flag = 1;     % flag for radial basis functions,1-MQ  2-exp ,3-Thin
plate spline
```

```
n = Nx*Ny;  % number of nodes
x_pt = 6.3; y_pt = 6.3;  % point about which interpolation is to be
done

X = zeros(Ny,Nx);    % assigning coordinates to zeros
Y = zeros(Ny,Nx);

% creating coordinates in matrix form
for i = 1:Nx-1  % over column
    X(1,i+1) = X(1,i) + dx;
    for j =1:Ny-1    %over rows
        X(j+1,:) = X(j,:);
        Y(j+1,1) = Y(j,1) + dy;
    end
    Y(:,i+1)=Y(:,i);
end

m = 3;   % number of monomials for polynomial interpolation
N = n+m;    % need it if using RPIM !

R = zeros(n,n);  % allocating R matrix ..related matrix to radial
basis function
P = ones(n,m);  % allocating polynomial moment matrix

[R,P,Rt_Pt,Rt_Ptx,Rt_Pty] =
moment_matrix(R,P,n,X,Y,x_pt,y_pt,Flag,m,alphac,dc,q);
if m==0
    G = R;
else
    G = [R P;P' zeros(m,m)];   % need not invert this matrix
end
[phi,phi_dx,phi_dy] = gauss_elisolver(G,Rt_Pt,Rt_Ptx,Rt_Pty);  %
gauss elimination solver


phi = phi(1:end-m);  % final shape function and it's derivatives in
vector form

% Proving Partition of unity property------------------

fprintf('Summation of shape functions at all the nodes is :\n ');
disp(sum(phi))
% ---------------------------------------

phi_dx = phi_dx(1:end-m);
phi_dy = phi_dy(1:end-m);

% ----------------------Post-processing-----------------------------
---
PHI = zeros(Ny,Nx);       DPHIDX =  zeros(Ny,Nx);     DPHIDY =
zeros(Ny,Nx);
```

```
 % assign shape function matrix and it's derivatives to zeros(same
size as that of X,Y)
for i = 1:n      % matlab is column major.. i guess
    PHI(i) = phi(i);      DPHIDX(i) = phi_dx(i);      DPHIDY(i) =
phi_dy(i);
end
figure(1)
surf(X,Y,PHI)
grid on
xlabel('x'); ylabel('y');zlabel('\phi');

figure(2)
surf(X,Y,DPHIDX)
grid on
xlabel('x'); ylabel('y');zlabel('\partial\phi/\partialx');


figure(3)
surf(X,Y,DPHIDY)
grid on
xlabel('x'); ylabel('y');zlabel('\partial\phi/\partialy');
```

Functions/Subroutines called – 1)moment_matrix.m 2) gauss.elisolver.m

```
function [R,P,Rt_Pt,Rt_Ptx,Rt_Pty] =
moment_matrix(R,P,n,X,Y,x_pt,y_pt,Flag,m,alphac,dc,q)

if m==0
    P = [];
else
    % polynomial interpolation matrix
    for i = 1:n
        P(i,:) = [1 X(i) Y(i)]; % change as per number of monomials

    end
end

switch(Flag)
    case 1    % Multiquadratics(MQ-PIM)

        %-------------filling R matrix as per MQT --------%
        for i = 1:n
            Rt(i,1) = ( (x_pt-X(i))^2 + (y_pt-Y(i))^2  +
(alphac*dc)^2   )^q;
            Rt_x(i,1)  = 2*q*(x_pt-X(i))*( (x_pt-X(i))^2 + (y_pt-
Y(i))^2  + (alphac*dc)^2 )^(q-1) ;
            Rt_y(i,1)  = 2*q*(y_pt-Y(i))*( (x_pt-X(i))^2 + (y_pt-
Y(i))^2  + (alphac*dc)^2 )^(q-1) ;
            for j = 1:n
```

```
                R(i,j) = ( (X(i)-X(j))^2 + (Y(i)-Y(j))^2  +
(alphac*dc)^2   )^q;
            end
        end


    case 2   % Gaussian (exp-PIM)

        %-------------filling R matrix as per gaussian --------%
        for i = 1:n
            Rt(i,1) = exp( (-alphac/(dc^2))* ((x_pt-X(i))^2 +
(y_pt-Y(i))^2)    );
            Rt_x(i,1) = -2*(-alphac/(dc^2))*(x_pt-X(i))*Rt(i,1);
            Rt_y(i,1) = -2*(-alphac/(dc^2))*(y_pt-Y(i))*Rt(i,1);
            for j = 1:n
                R(i,j) = exp( (-alphac/dc^2)* ((X(i)-X(j))^2 +
(Y(i)-Y(j))^2)    );
            end
        end

    case 3    % Thin plate spline(TPS-PIM)
        eta= 0.3;
        for i = 1:n
            Rt(i,1) =( ((x_pt-X(i))^2 + (y_pt-Y(i))^2) )^eta;
            Rt_x(i,1) = 2*eta*(x_pt-X(i))*( ((x_pt-X(i))^2 + (y_pt-
Y(i))^2) )^(eta-1) ;
            Rt_y(i,1) = 2*eta*(y_pt-Y(i))*( ((x_pt-X(i))^2 + (y_pt-
Y(i))^2) )^(eta-1);
            for j = 1:n
                R(i,j) =( ((X(i)-X(j))^2 + (Y(i)-Y(j))^2) )^eta;
            end
        end


end

if m ==0
    Rt_Pt = Rt; Rt_Ptx = Rt_x; Rt_Pty = Rt_y;
else
    Rt_Pt = [Rt;1;x_pt;y_pt];Rt_Ptx = [Rt_x;0;1;0];Rt_Pty =
[Rt_y;0;0;1];
end

end


----------------------------------------------------------
function [phi,dphidx,dphidy] =
gauss_elisolver(G,Rt_Pt,Rt_Ptx,Rt_Pty)

% elimination step
n=length(Rt_Pt);
```

```
phi = zeros(n,1);   dphidx = zeros(n,1);    dphidy = zeros(n,1);
for i=1:n
    lam = -1*(G(i+1:n,i)/G(i,i));
    G(i+1:n,:) = G(i+1:n,:) + lam*G(i,:);
    Rt_Pt(i+1:n,:) = Rt_Pt(i+1:n,:) + lam*Rt_Pt(i,:);
    Rt_Ptx(i+1:n,:) = Rt_Ptx(i+1:n,:) + lam*Rt_Ptx(i,:);
    Rt_Pty(i+1:n,:) = Rt_Pty(i+1:n,:) + lam*Rt_Pty(i,:);
end
phi(n,:) = Rt_Pt(n,:)/G(n,n);
dphidx(n,:) = Rt_Ptx(n,:)/G(n,n);
dphidy(n,:) = Rt_Pty(n,:)/G(n,n);

%Backsubstitution
for i = n-1:-1:1
    phi(i,:) = (Rt_Pt(i,:) - G(i,i+1:n)*phi(i+1:n,:))/G(i,i);
    dphidx(i,:) = (Rt_Ptx(i,:) -
G(i,i+1:n)*dphidx(i+1:n,:))/G(i,i);
    dphidy(i,:) = (Rt_Pty(i,:) -
G(i,i+1:n)*dphidy(i+1:n,:))/G(i,i);
end
end
```

Appendix B

Partition of unity property:

$$\sum_{i=1}^{N} \phi_i = 1, \text{ in this case N = 25 x 25}$$

The following code of line in Rpim.m proves the partition of unity for MQ-PIM and EXP-PIM shape functions

```
% Proving Partition of unity property------------------

fprintf('Summation of shape functions at all the nodes is :\n ');
disp(sum(phi))
% ---------------------------------------
```

Following table obtained by running code in Appendix A for following cases, shows summation of shape function follows partition of unity property

Method	$\sum_{i=1}^{N} \phi_i$
MQ-PIM q = 0.5	1.0000
MQ-PIM q = -0.5	1.0000
EXP-PIM $\alpha_c = 0.1$	1.0000
EXP-PIM $\alpha_c = 0.35$	1.0000
EXP-PIM $\alpha_c = 0.5$	1.0000