MAE 510 – NUMERICAL METHODS FOR MOVING INTERFACES

SPRING 2016 – HW #3

04/22/2016

SUBMITTED BY

SIDDHANT S APHALE

PERSON # 50164327

Introduction:

The Level Set Method is based upon representing an interface as the zero level set of some higher dimensional function ϕ. It is an implicit front tracking technique. Level Set Method is a sharp interface method. In a typical situation, we assume that the level set function is a signed distance function. It is defined as follows for the given curve Γ in the plane

$$\phi(x) = sign\left(-\pi + \int_{\Gamma} \arg(y(s) - x)\, ds\right) \underbrace{min}_{y \in \Gamma} \|x - y\|$$

Here, ϕ is the signed distance function to the interface Γ, where ϕ>0 on one side of the interface and ϕ<0 on the other. At all times, the interface is given by ϕ(x,t)=0.
The very first task is to initialize the level set method. In our case, the initial functions are defined. We are provided with three different functions at the original state. These are the evolution equations of the three different interfaces and are given below
Function 1:

$$\phi_0(x, y) = 0.25x^2 + y^2 - 0.25$$

Function 2:

$$\phi_0(x, y) = 16x^4 - 4x^2 + 16y^4 - 4y^2$$

Function 3:

$$\phi_0(x, y) = (x^2 + y^2)^2 - 1$$

The evolution equation for the level set function ϕ is an equation of the form

$$\phi_t + H(\nabla\phi) = 0$$

This equation is a Hamilton-Jacobi type of equation and H is called the Hamiltonian function. This equation is approximated by numerical Hamiltonian (flux function) as

$$H(\nabla\phi) \approx g(D_{-x}\phi, D_{+x}\phi, D_{-y}\phi, D_{+y}\phi)$$

The boundary conditions used for this problem are found out using Ghost Nodes. We use the ghost points using linear extrapolation. For example, to determine the value of the ghost point $\phi_{-1,j}$, the linear extrapolation is given by

$$\phi_{-1,j} = 2\phi_{0,j} - \phi_{1,j}$$

By using the above mentioned logic, the boundary values of a, b, c and d were found out at every boundary. The original functions given above needs to be reinitialized. Reinitialization was first conceived while trying to solve Plateau's problem using level sets. It was observed that while advancing to steady state, an initially well behaved solution would go unstable. However, if the position of the front were set just before instability and the level sets function were reconstructed to look like an initial condition, then the computation could proceed further. The reinitialization is the process by which ϕ is reconstructed to the signed distance function to the set $\phi^{-1}(0)$. To accomplish this, one method is to consider the following partial differential equation

$$\phi_t + sign(\phi^0)(\|\nabla\phi\| - 1) = 0$$

where ϕ^0 is the original level set function as defined by the three functions above in our case. To evolve this equation, we use technique similar to that used for level set advancement as this is a Hamilton-Jacobi type of equation. We use first-order in time updating scheme

$$\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^{n}}{\Delta\tau} + H(a,b,c,d) = 0$$

where $a = D_{-x}\phi_{i,j}^{n}, b = D_{+x}\phi_{i,j}^{n}, c = D_{-y}\phi_{i,j}^{n}, d = D_{+y}\phi_{i,j}^{n}$

The Hamiltonian *H* is given by numerical approximation as

$$H(a,b,c,d) = s^{+}\left(\sqrt{max((a^{+})^2,(b^{-})^2) + max((c^{+})^2,(d^{-})^2)} - 1\right) + s^{-}\left(\sqrt{max((a^{-})^2,(b^{+})^2) + max((c^{-})^2,(d^{+})^2)} - 1\right)$$

where f^{+}=max(f,0) and f^{-}=min(f,0). Finally the sign function is written as $s=sign(\phi^0)$. To ensure numerical stability, we use a smoothed sign function which can be given as

$$s_{\epsilon}(\phi) = \frac{\phi}{\sqrt{\phi^2 + h^2}}$$

$$s_{\epsilon}(\phi) = \frac{\phi}{\sqrt{\phi^2 + \|\nabla\phi\|^2}}$$

This is a highly diffusive process which can result in a large amount of volume change during the reinitialization process. It is recommended to avoid the use of reinitialization process. However, if the reinitialization process is used we need to use a fix to ensure the spurious movement of the interface. One of the method to ensure that the interface does not undergo spurious movement is the sub-cell fix method. In this method, the nodes next to the interface are treated differently than nodes away from the interface. By using this method, we evolve the level set using the following equations

$$\phi_t = \begin{cases} -\frac{1}{h}\left(sign(\phi^0)|\phi_{i,j}| - D_{i,j}\right) if\ (i,j) \in \Sigma_h \\ H(a,b,c,d) \qquad\qquad otherwise \end{cases}$$

where $D_{i,j}$ is the distance approximated as follows

$$D_{i,j} = \frac{2h\phi_{i,j}^{0}}{\sqrt{(\phi_{i+1,j}^{0} - \phi_{i-1,j}^{0})^2 + (\phi_{i,j+1}^{0} - \phi_{i,j-1}^{0})^2}}$$

In the problem given, we have three different original level set functions. The objective here is to reinitialize the original level set functions and evolve for 512 iterations. We have to compare the results with using sub-cell fix as against without using sub-cell fix. The domain for all the three functions is [-2,2] x [-2,2]. A 64 x 64 grid is used for each domain. For discretization purpose, we have used 1st order for both time and upwind discretization. The time-step used is 0.5*h*. For each function, 4 plots were generated as described below.

First plot is for the contour plot over the domain for the original function. In the second plot, a contour plot for the reinitialized function without the use of sub-cell fix is generated. The third plot is a contour plot for the

reinitialized function with the use of the sub-cell fix. The reinitialized interface with and without sub-cell fix are compared in the fourth plot.

Results:

1. Function 1:

The first function was defined as

$$\phi_0(x, y) = 0.25x^2 + y^2 - 0.25$$

The domain was defined initially and grid was generated over the entire domain. The initial contour plot for the original function 1 is given as in fig. 1.

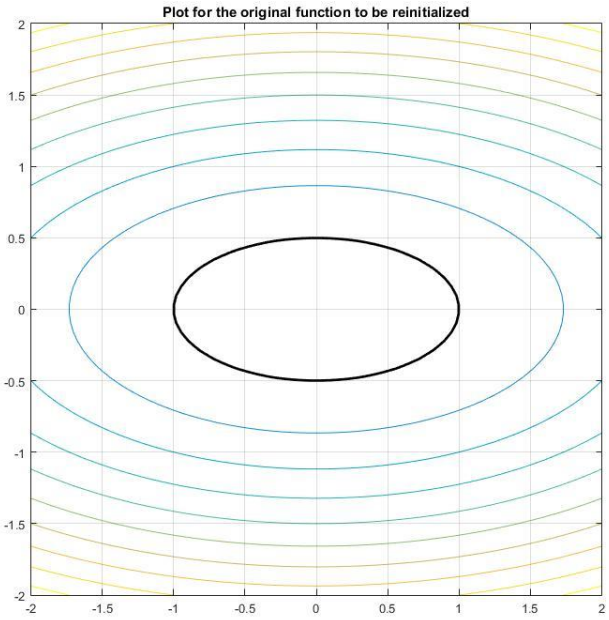

Figure 1 Contour Plot for Original Function

Now, we reinitialize the function as described before. We again plot the contour of the reinitialized function. In this case we do not use the sub-cell fix. Figure 2 depicts the contour plot for the reinitialized function without sub-cell fix. In this plot, we can observe that the evolution of the function is not smooth. This error occurs due to the diffusive process we utilized to reinitialize the original function. We can observe that the initial volume which was between [-1,1] has now drastically changed and now has reduced to approximately [-0.6,0.6] in the X-direction. The Y-direction has not changed. This error can be avoided by using the sub-cell fix.

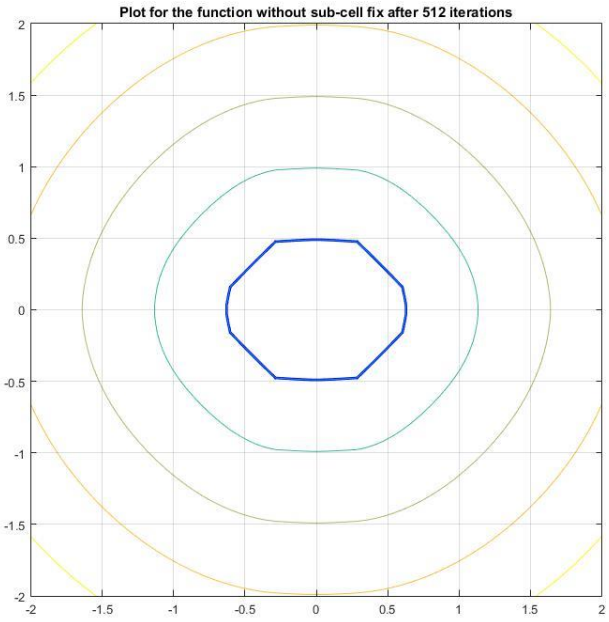

Figure 2 Contour Plot of the reinitialized function without using sub-cell fix

As discussed above, to overcome the volume change problem in the reinitialization process occurring due to the highly diffusive method we used, we employ the sub-cell fix method. The reinitialized function was fixed as mentioned above and the results are as in Fig. 3.

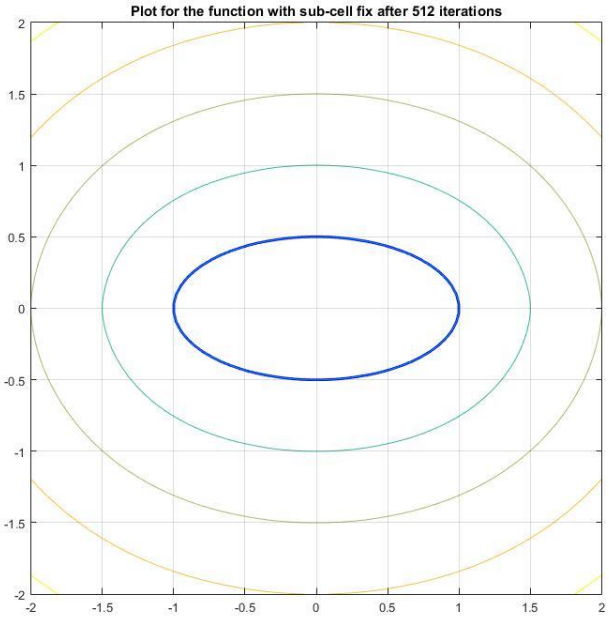

Figure 3 Plot for the reinitialized function with the use of sub-cell fix

It can be observed from the above Fig. 3, the volume change problem is solved and now the interface is evolved in elliptic shape in conformance with the original function. The Fig. 4 shows the comparison of the interfaces

after completing 512 iterations with and without sub-cell fix. We can clearly observe the amount of volume change in this plot.

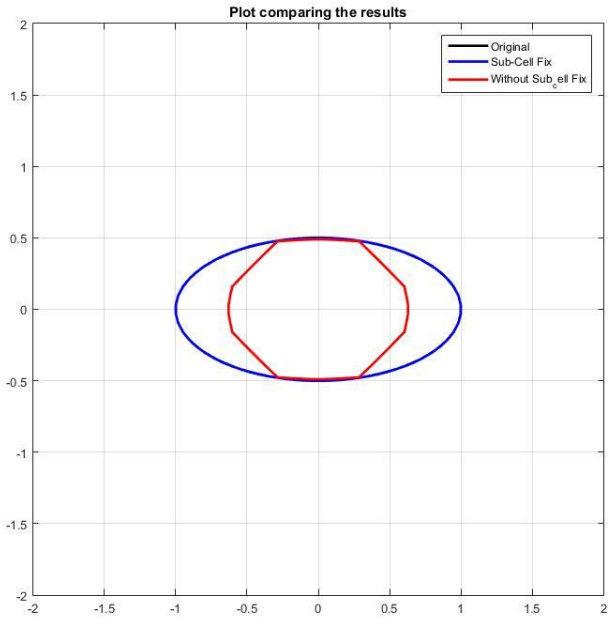

Figure 4 Comparison of the interface after 512 iterations for function 1

2. Function 2:

In this case, our original evolution function for the interface is defined as

$$\phi_0(x, y) = 16x^4 - 4x^2 + 16y^4 - 4y^2$$

We again plot the contour of this original function as in Fig. 5.

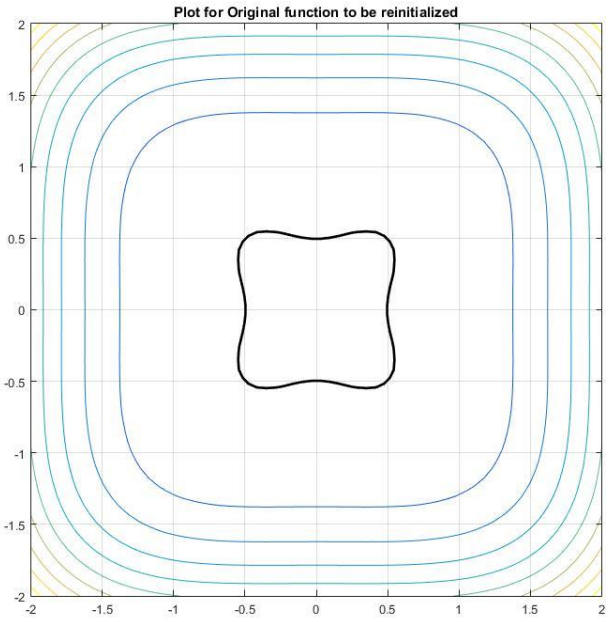

Figure 5 Contour Plot for the Original Function 2

Now we proceed with the reinitialization of the function. The reinitialization without sub-cell fix is obtained and the contour for this reinitializaed function is plotted in Fig. 6.

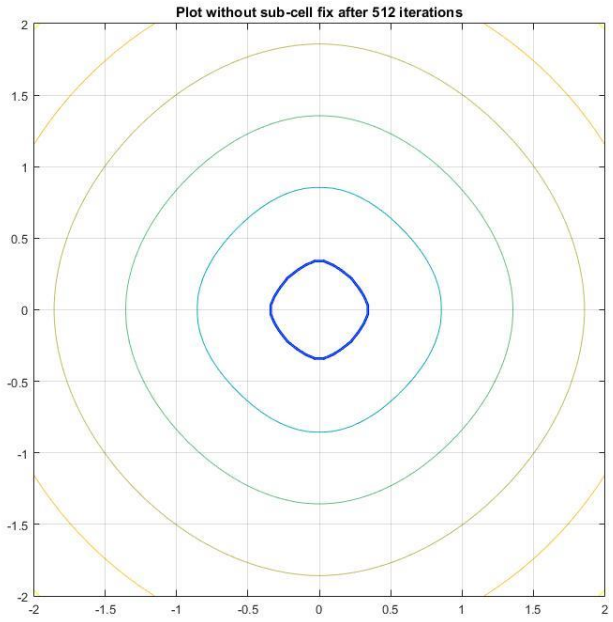

Figure 6 Plot for the reinitialized function without sub-cell fix for function 2

From Fig. 6 we can observe that the volume of the interface has significantly changed. The reduction in the volume of the interface in this case is in both X and Y directions. We can also observe that the evolution of the interface has occurred in the same trend. This error needs to be fixed and sub-cell fix method is utilized to overcome this problem. Figure 7 shows the contour plot for the reinitialized interface function with sub-cell fix.

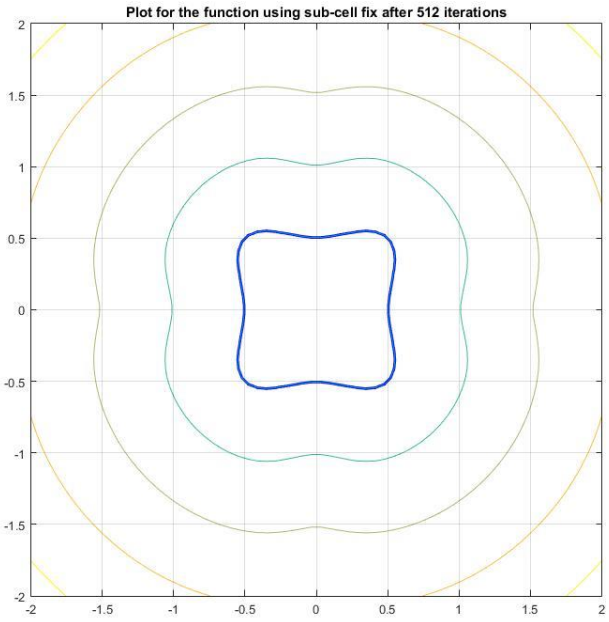

Figure 7 Contour plot for reinitialized function with sub-cell fix-Function 2

We can observe that, on sub-cell fix, the evolution of the interface is in conformance with the original function.

Further, we compare the results after 512 iterations for the reinitialized function with and without sub-cell fix. The results can be observed in Fig. 8.

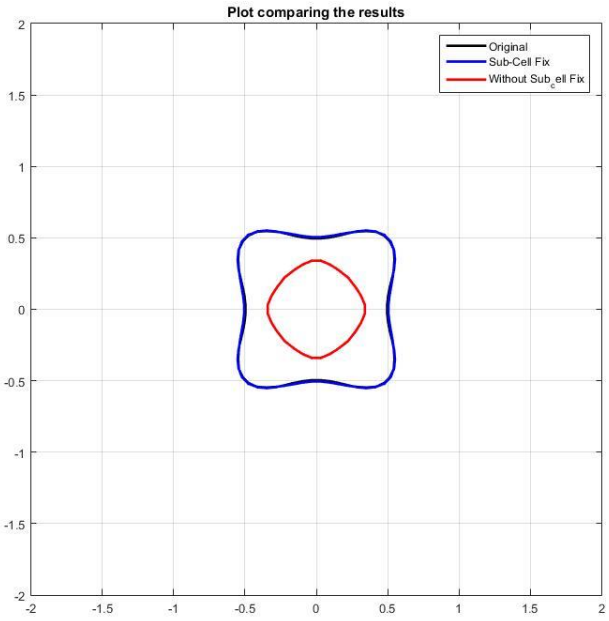

Figure 8 Plot for the comparison of reinitialized function with and without sub-cell fix Function 2

3. Function 3:

In the third case, our original function is defined as

$$\phi_0(x, y) = (x^2 + y^2)^2 - 1$$

We again plot the initial function as in Fig. 9. Further we reinitialize the function without using sub-cell fix and the results obtained are as in Fig. 10. To overcome the change in volume, we use the sub-cell fix method and the results obtained are plotted as in Fig. 11. The results for the evolution of reinitialized interface after 512 iterations with and without sub-cell fix are compared as in Fig. 12.

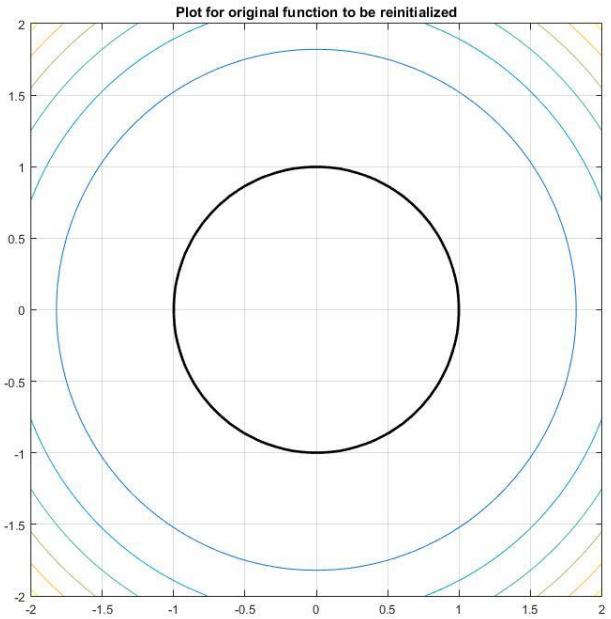

Figure 9 Plot for Original Function-3

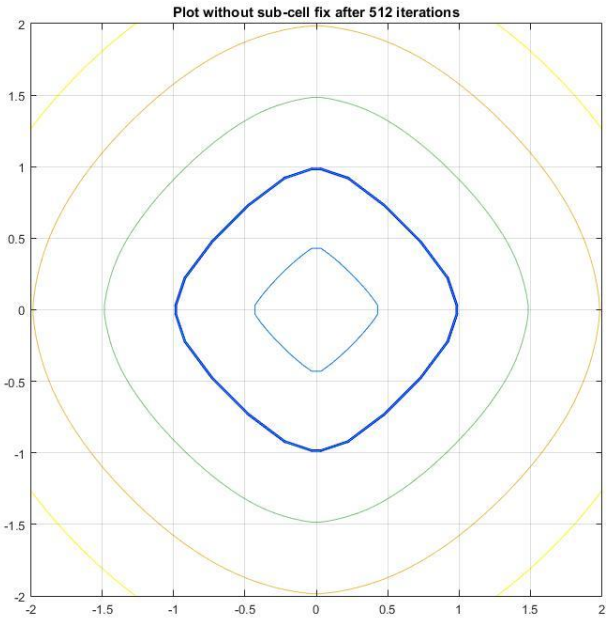

Figure 10 Plot for reinitialized function without sub-cell fix-Function 3

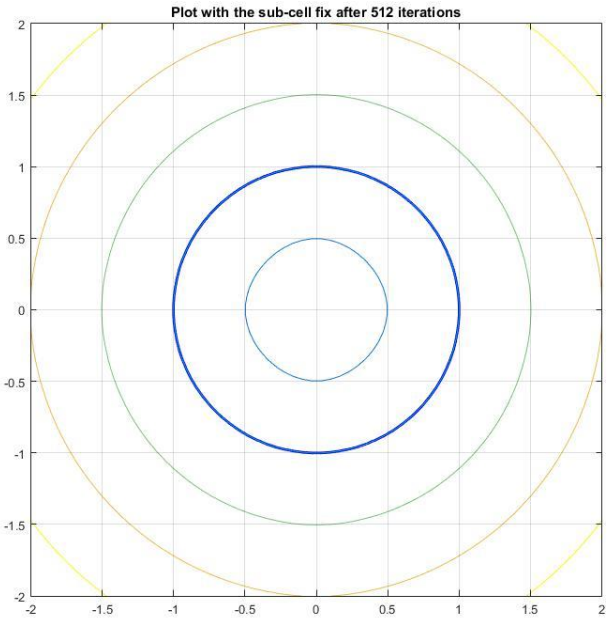

Figure 11 Plot for reinitialized function with sub-cell fix-Function 3

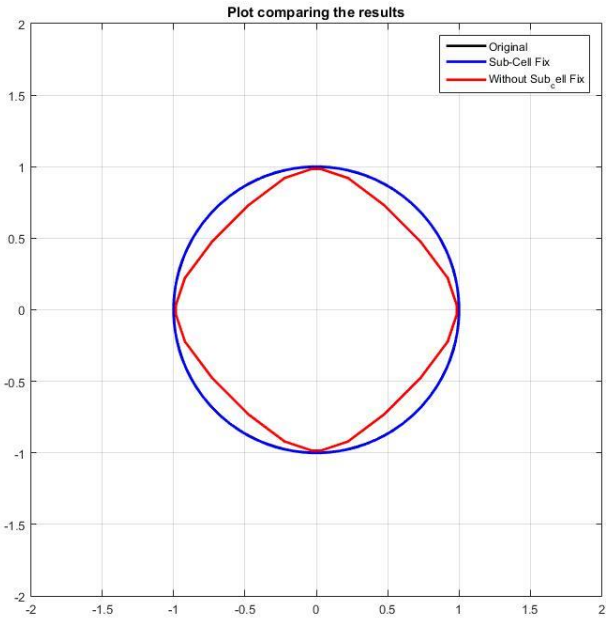

Figure 12 Plot for the comparison of the interfaces after 512 iterations with and without sub-cell fix

Conclusion:

In this assignment we performed the reinitialization of the level set functions. We were given 3 different functions. We observed that by utilizing the reinitializtion of the original function we were implementing a highly diffusive process which is changing the volume of the interface drastically. To fix this problem, we utilized a sub-cell method. On implementation of this method we can observe that there is no change in the volume and the interface evolves in conformance with the original shape.

References:

1. Class Notes by Dr. Salac
2. Printed Notes by Dr. Salac.
3. http://www.mathworks.com/help/matlab/

Appendix:

MATLAB Source Code:

```
close all; clear; clc;
%Domain is a [-2, 2] x [-2, 2] with a 64 x 64 grid
xmax=2; xmin=-2; %Defining the maximum and minimum values of domain in X
ymax=2; ymin=-2; %Defining the maximum and minimum values of domain in y
imax=64; jmax=64; %Defining the grid
%Finding out the grid spacing values
dx=(xmax-xmin)/(imax-1);
dy=(ymax-ymin)/(jmax-1);
dt=0.5*dx; %Defining the time-step

%Finding out the values of X and Y at every grid points
[Y, X] = meshgrid(-2:dx:2, -2:dy:2);
% dlmwrite('X.txt',X);
% dlmwrite('Y.txt',Y);

%Total number of iterations
n=512;

%Initialization
for i=1:imax
    for j=1:jmax
        phi_zero(i,j)=0.25.*(X(i,j).^2)+(Y(i,j).^2)-0.25;
%         phi_zero(i,j)=16*(X(i,j).^4)+16*(Y(i,j).^4)-4*(X(i,j).^2)-4*(Y(i,j).^2);
%         phi_zero(i,j)=((X(i,j).^2)+(Y(i,j).^2)).^2-1;
    end
end
% dlmwrite('Initial.txt',phi_zero);
phi=phi_zero;
for t=1:n

    %Reinitialization of phi
    %Calculating the values of a,b,c and d first

    for i=2:imax-1
        for j=2:jmax-1
            a(i,j)=(phi(i,j)-phi(i-1,j))/dx;
            b(i,j)=(phi(i+1,j)-phi(i,j))/dx;
            c(i,j)=(phi(i,j)-phi(i,j-1))/dy;
            d(i,j)=(phi(i,j+1)-phi(i,j))/dy;
        end
    end

    %Calculation of a,b,c and d at boundaries by using Ghost Nodes

    for j=1:jmax
        a(1,j)=(phi(1,j)-phi(imax-1,j))/dx;
        b(1,j)=(phi(2,j)-phi(1,j))/dx;
        a(imax,j)=(phi(imax,j)-phi(imax-1,j))/dx;
        b(imax,j)=(phi(2,j)-phi(imax,j))/dx;
    end
    for i=1:imax
        c(i,1)=(phi(i,1)-phi(i,jmax-1))/dy;
        c(i,jmax)=(phi(i,jmax)-phi(i,jmax-1))/dy;
```

```
    d(i,1)=(phi(i,2)-phi(i,1))/dy;
    d(i,jmax)=(phi(i,2)-phi(i,jmax))/dy;
end
for i=2:imax-1
    a(i,1)=(phi(i,1)-phi(i-1,1))/dx;
    a(i,jmax)=(phi(i,jmax)-phi(i-1,jmax))/dx;
    b(i,1)=(phi(i+1,1)-phi(i,1))/dx;
    b(i,jmax)=(phi(i+1,jmax)-phi(i,jmax))/dx;
end
for j=2:jmax-1
    c(1,j)=(phi(1,j)-phi(1,j-1))/dy;
    c(imax,j)=(phi(imax,j)-phi(imax,j-1))/dy;
    d(1,j)=(phi(1,j+1)-phi(1,j))/dy;
    d(imax,j)=(phi(imax,j+1)-phi(imax,j))/dy;
end

%Finding out the values of a+, a-, b+, b-, c+, c-, d+, d-, s+, s-

for i=1:imax
    for j=1:jmax
        a_plus(i,j)=max(a(i,j),0);
        a_minus(i,j)=min(0,a(i,j));
        b_plus(i,j)=max(b(i,j),0);
        b_minus(i,j)=min(0,b(i,j));
        c_plus(i,j)=max(c(i,j),0);
        c_minus(i,j)=min(0,c(i,j));
        d_plus(i,j)=max(d(i,j),0);
        d_minus(i,j)=min(0,d(i,j));

        if phi(i,j)>0.0001
            s_plus(i,j)=+1;
            s_minus(i,j)=0;
        elseif phi(i,j)<-0.0001
            s_plus(i,j)=0;
            s_minus(i,j)=-1;
        else
            s_plus(i,j)=phi(i,j)/(sqrt(phi(i,j)^2+dx^2));
            s_minus(i,j)=phi(i,j)/(sqrt(phi(i,j)^2+dx^2));
        end
        if phi(i,j)>0.0001
            s_zero(i,j)=1;
        end
        if phi(i,j)<-0.0001
            s_zero(i,j)=-1;
        end
        if phi(i,j)==0
            s_zero(i,j)=phi(i,j)/(sqrt(phi(i,j)^2+dx^2));
        end
    end
end

%Finding out the new values of phi

for i=1:imax
    for j=1:jmax
        phi_new(i,j)=phi(i,j)-dt*(s_plus(i,j)*...
            ((sqrt(max(a_plus(i,j)^2,b_minus(i,j)^2)+...
            max(c_plus(i,j)^2,d_minus(i,j)^2)))-1))-...
            dt*(s_minus(i,j)*((sqrt(max(a_minus(i,j)^2,b_plus(i,j)^2)+...
            max(c_minus(i,j)^2,d_plus(i,j)^2)))-1));
```

```
    end
end

%Sub-cell Fix
%Calculating signed distance functions
for i=2:imax-1
    for j=2:jmax-1
        if phi(i+1,j)*phi(i,j)<0 ||...
                phi(i,j+1)*phi(i,j)<0 ||...
                phi(i,j)*phi(i-1,j)<0 ||...
                phi(i,j)*phi(i,j-1)<0
            SDF(i,j)=(2*dx*phi_zero(i,j))/...
                (sqrt((phi_zero(i+1,j)-phi_zero(i-1,j)).^2+...
                (phi_zero(i,j+1)-phi_zero(i,j-1)).^2));
        else
            SDF(i,j)=0;
        end
    end
end
for i=2:imax-1
    if phi(i+1,1)*phi(i,1)<0||...
            phi(i,2)*phi(i,1)<0||...
            phi(i,1)*phi(i-1,1)<0||...
            phi(i,1)*phi(i,jmax-1)<0
        SDF(i,1)=(2*dx*phi_zero(i,1))/...
            (sqrt((phi_zero(i+1,1)-phi_zero(i-1,1)).^2+...
            (phi_zero(i,2)-phi_zero(i,jmax-1)).^2));
        phi_sub_fix(i,1)=phi(i,1)-dt*((s_zero(i,1)*...
            abs(phi(i,1)))-SDF(i,1))/dx;
    else
        SDF(i,1)=0;
        phi_sub_fix(i,1)=phi(i,1)-dt*(s_plus(i,1)*...
            ((sqrt(max(a_plus(i,1)^2,b_minus(i,1)^2)+...
            max(c_plus(i,1)^2,d_minus(i,1)^2)))-1))-...
            dt*(s_minus(i,1)*((sqrt(max(a_minus(i,1)^2,...
            b_plus(i,1)^2)+max(c_minus(i,1).^2,d_plus(i,1).^2)))-1));
    end
    if phi(i+1,jmax)*phi(i,jmax)<0||...
            phi(i,2)*phi(i,jmax)<0||...
            phi(i,jmax)*phi(i-1,jmax)<0||...
            phi(i,jmax)*phi(i,jmax-1)<0
        SDF(i,jmax)=(2*dx*phi_zero(i,jmax))/...
            (sqrt((phi_zero(i+1,jmax)-phi_zero(i-1,jmax)).^2+...
            (phi_zero(i,jmax-1)-phi_zero(i,2)).^2));
        phi_sub_fix(i,jmax)=phi(i,jmax)-dt.*((s_zero(i,jmax)*...
            abs(phi(i,jmax)))-SDF(i,jmax))/dx;
    else
        SDF(i,jmax)=0;
        phi_sub_fix(i,jmax)=phi(i,jmax)-dt*(s_plus(i,jmax)*...
            ((sqrt(max(a_plus(i,jmax)^2,b_minus(i,jmax)^2)+...
            max(c_plus(i,jmax)^2,d_minus(i,jmax)^2)))-1))-...
            dt*(s_minus(i,jmax)*((sqrt(max(a_minus(i,jmax)^2,...
            b_plus(i,jmax)^2)+max(c_minus(i,jmax)^2,d_plus(i,jmax)^2)))-1));
    end
end

for j=2:jmax-1
    if phi(2,j)*phi(1,j)<0||...
            phi(1,j+1)*phi(1,j)<0||...
            phi(1,j)*phi(imax-1,j)<0||...
```

```
            phi(1,j)*phi(1,j-1)<0
        SDF(1,j)=(2*dx*phi_zero(1,j))/...
            (sqrt((phi_zero(2,j)-phi_zero(imax-1,j)).^2+...
            ((phi-zero(1,j+1)-phi_zero(1,j-1)).^2)));
        phi_sub_fix(1,j)=phi(1,j)-dt*((s_zero(1,j)*...
            abs(phi(1,j)))-SDF(1,j))/dx;
    else
        SDF(1,j)=0;
        phi_sub_fix(1,j)=phi(1,j)-dt*(s_plus(1,j)*...
            ((sqrt(max(a_plus(1,j)^2,b_minus(1,j)^2)+...
            max(c_plus(1,j)^2,d_minus(1,j)^2)))-1))-...
            dt*(s_minus(1,j)*((sqrt(max(a_minus(1,j)^2,...
            b_plus(1,j)^2)+max(c_minus(1,j)^2,d_plus(1,j)^2)))-1));
    end
    if phi(2,j)*phi(imax,j)<0||...
            phi(imax,j+1)*phi(imax,j)<0||...
            phi(imax,j)*phi(imax-1,j)<0||...
            phi(imax,j)*phi(imax,j-1)<0
        SDF(imax,j)=(2*dx*phi_zero(imax,j))/...
            (sqrt((phi_zero(2,j)-phi_zero(imax-1,j)).^2+...
            (phi_zero(imax,j+1)-phi_zero(imax,j-1)).^2));
        phi_sub_fix(imax,j)=phi(imax,j)-dt*((s_zero(imax,j)*...
            abs(phi(imax,j)))-SDF(imax,j))/dx;
    else
        SDF(imax,j)=0;
        phi_sub_fix(imax,j)=phi(imax,j)-dt*(s_plus(imax,j)*...
            ((sqrt(max(a_plus(imax,j)^2,b_minus(imax,j)^2)+...
            max(c_plus(imax,j)^2,d_minus(imax,j)^2)))-1))-...
            dt*(s_minus(imax,j)*((sqrt(max(a_minus(imax,j)^2,...
            b_plus(imax,j)^2)+max(c_minus(imax,j)^2,d_plus(imax,j)^2)))-1));
    end
end

%Calculating the value of phi after sub-cell fix by marching in time
for i=2:imax-1
    for j=2:jmax-1
        if phi(i+1,j)*phi(i,j)<0||...
                phi(i,j+1)*phi(i,j)<0||...
                phi(i,j)*phi(i-1,j)<0||...
                phi(i,j)*phi(i,j-1)<0
            phi_sub_fix(i,j)=phi(i,j)-dt*((s_zero(i,j)*...
                abs(phi(i,j)))-SDF(i,j))/dx;
        else
            phi_sub_fix(i,j)=phi(i,j)-dt*(s_plus(i,j)*...
                ((sqrt(max(a_plus(i,j)^2,b_minus(i,j)^2)+...
                max(c_plus(i,j)^2,d_minus(i,j)^2)))-1))-dt*...
                (s_minus(i,j)*((sqrt(max(a_minus(i,j)^2,...
                b_plus(i,j)^2)+max(c_minus(i,j)^2,d_plus(i,j)^2)))-1));
        end
    end
end
phi_sub_fix(1,1)=phi_sub_fix(2,1);
phi_sub_fix(imax,jmax)=phi_sub_fix(imax-1,jmax);
phi_sub_fix(1,jmax)=phi_sub_fix(2,jmax);
phi_sub_fix(imax,1)=phi_sub_fix(imax-1,1);

%For sub-cell fix un-comment this
%    phi=phi_sub_fix;
%For without sub-cell fix un-comment this
phi=phi_new;
```

```
end
% dlmwrite('withoutsub.txt',phi);
%Plotting the Initial Condition
figure(1)
contour(X,Y,phi_zero)
grid on
axis square
hold on;
contour(X,Y,phi_zero,[0,0],'-k','LineWidth',2)

%Plotting the Reinitialization
figure(2)
contour(X,Y,phi,[0,0],'-b','LineWidth',2)
grid on
axis square
hold on;
% contour(X,Y,phi_zero,[0,0],'-r','LineWidth',2)
contour(X,Y,phi)
```