```
clear all
clc
close all
format short

% Cálculo numérico para engenharia elétrica com Matlab
% Capítulo 8: equações diferenciais parciais
% FDTD

% Grid Dimension in x (xdim) and y (ydim) directions
xdim=100; ydim=100;
%Total no of time steps
time_tot=350;
%Position of the source (center of the domain)
xsource=50; ysource=25;
%Courant stability factor
S=1/(2^0.5);
% Parameters of free space (permittivity and permeability and speed of
% light) are all not 1 and are given real values
epsilon0=(1/(36*pi))*1e-9; mu0=4*pi*1e-7; c=3e+8;
% Spatial grid step length (spatial grid step= 1 micron and can be
changed)
delta=1e-6;
% Temporal grid step obtained using Courant condition
deltat=S*delta/c;
% Initialization of field matrices
Ez=zeros(xdim,ydim); Hy=zeros(xdim,ydim); Hx=zeros(xdim,ydim);
% Initialization of permittivity and permeability matrices
epsilon=epsilon0*ones(xdim,ydim); mu=mu0*ones(xdim,ydim);
% Initializing electric and magnetic conductivity matrices
sigma=4e-4*ones(xdim,ydim); sigma_star=4e-4*ones(xdim,ydim);
%Choice of nature of source
gaussian=0; sine=0;
% The user can give a frequency of his choice for sinusoidal (if sine=1
above) waves in Hz
frequency=1.5e+13; impulse=0;
%Choose any one as 1 and rest as 0. Default (when all are 0): Unit time
step
%Multiplication factor matrices for H matrix update to avoid being
calculated many times
%in the time update loop so as to increase computation speed
A=((mu-0.5*deltat*sigma_star)./(mu+0.5*deltat*sigma_star));
B=(deltat/delta)./(mu+0.5*deltat*sigma_star);
%Multiplication factor matrices for E matrix update to avoid being
calculated many times
%in the time update loop so as to increase computation speed
C=((epsilon-0.5*deltat*sigma)./(epsilon+0.5*deltat*sigma));
D=(deltat/delta)./(epsilon+0.5*deltat*sigma);
% Update loop begins
for n=1:1:time_tot
    %if source is impulse or unit-time step
    if gaussian==0 && sine==0 && n==1
        Ez(xsource,ysource)=1;
    end
    % Setting time dependent boundaries to update only relevant parts of
the
    % vector where the wave has reached to avoid unnecessary updates.
```

```
    if n<xsource-2       n1=xsource-n-1;  else   n1=1;       end
    if n<xdim-1-xsource  n2=xsource+n;    else   n2=xdim-1;  end
    if n<ysource-2       n11=ysource-n-1; else   n11=1;      end
    if n<ydim-1-ysource  n21=ysource+n;   else   n21=ydim-1; end
    %Vector update instead of for-loop for Hy and Hx fields

Hy(n1:n2,n11:n21)=A(n1:n2,n11:n21).*Hy(n1:n2,n11:n21)+B(n1:n2,n11:n21).*(
Ez(n1+1:n2+1,n11:n21)-Ez(n1:n2,n11:n21));
    Hx(n1:n2,n11:n21)=A(n1:n2,n11:n21).*Hx(n1:n2,n11:n21)-
B(n1:n2,n11:n21).*(Ez(n1:n2,n11+1:n21+1)-Ez(n1:n2,n11:n21));
    %Vector update instead of for-loop for Ez field

Ez(n1+1:n2+1,n11+1:n21+1)=C(n1+1:n2+1,n11+1:n21+1).*Ez(n1+1:n2+1,n11+1:n2
1+1)+D(n1+1:n2+1,n11+1:n21+1).*(Hy(n1+1:n2+1,n11+1:n21+1)-
Hy(n1:n2,n11+1:n21+1)-Hx(n1+1:n2+1,n11+1:n21+1)+Hx(n1+1:n2+1,n11:n21));
    % Perfect Electric Conductor boundary condition
    Ez(1:xdim,1)=0; Ez(1:xdim,ydim)=0; Ez(1,1:ydim)=0; Ez(xdim,1:ydim)=0;
    % Source conditions
    if impulse==0
        % If unit-time step
        if gaussian==0 && sine==0  Ez(xsource,ysource)=1; end
        %if sine
        if sine==1
            tstart=1;  N_lambda=c/(frequency*delta);
            Ez(xsource,ysource)=sin(((2*pi*(c/(delta*N_lambda))*(n-
tstart)*deltat)));
        end
        %if gaussian
        if gaussian==1
            if n<=42   Ez(xsource,ysource)=(10-
15*cos(n*pi/20)+6*cos(2*n*pi/20)-cos(3*n*pi/20))/32;
            else       Ez(xsource,ysource)=0;   end
        end
    else
        %if impulse
        Ez(xsource,ysource)=0;
    end
    %Movie type colour scaled image plot of Ez
    imagesc(delta*(1:1:xdim)*1e+6,(1e+6*delta*(1:1:ydim))',Ez',[-
1,1]);colorbar;colormap(hot);
    title(['\fontsize{16}Colour-scaled image plot of Ez in a spatial
domain with PEC boundary and at time = ',num2str(round(n*deltat*1e+15)),'
fs']);
    xlabel('x (in um)','FontSize',16);  ylabel('y (in
um)','FontSize',16); set(gca,'FontSize',16);
    getframe;
end
```