Table of Contents


```
function Abbas_HW7_Yee_with_Liao_ABC_rev1()
```

Hasan Tahir Abbas

ECEN 637 Homework 7: Simulating Yee's Algoirthm with 4th order Liaio Boundary Conditions 10/28/2015

This program generates the results presented in the paper: "Numerical Solution of Initial Boundary Value Problems involving Maxwell's Equations in Isotropic Media", Kane S. Yee (1966) applying Absorbing Boundary Conditions at the edges of the computational Space

```
clear;close all
```

```
% *************************
```

Parameters

****

```
global c xmu eps0 asize
global nx  ny  nt mxst mxnd myst mynd
global dt ds n
global Ez  Hx  Hy; % Create E and H field components.
% global data type enables the global scope of the variables
% within the code. Unlike global variables they can not be accessed
% outside the code
```

```
global mediaEz mediaHx mediaHy; %
global Ca Cb Da Db ; % Define material based coefficients
global c1 c2 c3 c4 c5
global Ez1 Ez2 Ez3 Ez4 Ez5 % For Bubbling of E-fields in Liao ABC
global ABC_order % Order of the Liao ABC


c = 2.99792458e8;
xmu = 4*pi*1e-7;
eps0 = 8.854187817e-12;
asize = 5; % Space Dimension in meters
nx = 80;      %% Number of cells in x-direction
ny = 100;     %% Number of cells in y-direction
nt = 100;     %% Number of time steps
mxst = 17;    %% Start of PEC section in x-direction
mxnd = 49;    %% End of PEC section in x-direction
myst = 33;    %% Start of PEC section in y-direction
mynd = 65;    %% End of PEC section in y-direction
% strip = 50;   %% y-Position where E-field is recorded

% ************************
```

Initialize

```
************************

Ez = zeros(nx,ny); %% z-component of E-field
Hx = zeros(nx,ny); %% x-component of H-field
Hy = zeros(nx,ny); %% y-component of H-field

mediaEz = ones(nx,ny); %% z-component of E-field
mediaHx = ones(nx,ny); %% x-component of H-field
mediaHy = ones(nx,ny); %% x-component of H-field

Ca = zeros(2,1); %% x-component of H-field
Cb = zeros(2,1); %% x-component of H-field
Da = zeros(2,1); %% x-component of H-field
Db = zeros(2,1); %% x-component of H-field

Ez1 = zeros(nx,ny);
Ez2 = zeros(nx,ny);
Ez3 = zeros(nx,ny);
Ez4 = zeros(nx,ny);
Ez5 = zeros(nx,ny);


% % Initialize arrays to create to capture E-field
% % line plots at various time instants.
% Ez_5 = zeros(1,nx);
% Ez_35 = zeros(1,nx);
% Ez_65 = zeros(1,nx);
% Ez_95 = zeros(1,nx);
```

```
ds = asize/(mxnd - mxst - 1); %% Length Increment
% ************************
```

Stability Condition

```
***********************

dt = ds/(c*(2)); % Stability Condition
% ************************
% ************************

% ************************
% ************************
```

Main Program

```
*********************** ***********************
```

PEC

```
iflaga = 2;   %% 1 if Free space; 2 if PEC
define_media(iflaga); % Structre Definition
define_coefficients(); % Create Coefficients for equations
source = 2; % 2 is Gaussian, 1 is sinusoidal source
ABC_order = 4;
define_Liao(); % 4th order LIAO ABC (other options 3 or 5)
%
```

Ez field plotting

```
for n = 1:nt
    adv_Ez(n, source); % Compute E-fields
    Liao_ABC(); % Invoke ABC algorithm
    adv_H(); % Compute H-fields
%     if rem(n,5) == 0 % Plot at every 5th time step
%          figure(1)
%          my_surface_plot(Ez, iflaga);
%                 if rem(n,20) == 0 % Save Plot at every 20th time
 step
% % %                      cleanfigure();
% %
 matlab2tikz('filename',sprintf('ECEN637_HW7_Ez_surf_PEC_space_
%d.tex',n));
%                      saveas(gcf,
[sprintf('ECEN637_HW7_Ez_surf_PEC_space_%d',n),'.png']) % Save
 visualizations
%                 end
%
%     end   % end_if
end
```

```
%
%
%

end
% ************************
% ************************
%
```

Create Structure in the computational space

```
********************** **********************

function define_media(iflaga)

global nx  ny mxst mxnd myst mynd;
global mediaEz mediaHx mediaHy; %
if (iflaga == 2)

    for  i = 1:nx
        for j = 1:ny
            if (i >= mxst && i <= mxnd)
                if ( j >= myst && j <= mynd)
                    mediaEz(i,j) = 2;
                end
            end
        end
    end

    for  i = 1:nx
        for j = 1:ny
            if (i >= mxst && i <= mxnd)
                if ( j >= myst && j <= mynd-1)
                    mediaHx(i,j) = 2;
                end
            end
        end
    end

    for  i = 1:nx
        for j = 1:ny
            if (i >= mxst && i <= mxnd-1)
                if ( j >= myst && j <= mynd)
                    mediaHy(i,j) = 2;
                end
            end
        end
    end
end

end


% ************************
```

```
% ************************
%
```

Create Coefficients for the equations

```
*********************** ***********************

function define_coefficients()

global Ca Cb Da Db ; % Define material based coefficients
global xmu eps0 dt ds
% % % % % % % % Field Coefficients
dte = dt/(ds*eps0);
dtm = dt/(ds*xmu);
Da(1) = 1;
Db(1) = dtm;
Ca(1) = 1;
Cb(1) = dte;
Da(2) = 0;
Db(2) = 0;
Ca(2) = 0;
Cb(2) = 0;
end

% ************************
% ************************
%
```

Create Coefficients for LIAO ABC

```
*********************** ***********************

function define_Liao()

global c1 c2 c3 c4 c5 ; % Define material based coefficients
global ABC_order % Order of the Liao ABC
switch ABC_order
    case 5 %% 5th order LIAO ABC Coefficients
        c1=5;
        c2=10;
        c3=10;
        c4=5;
        c5=1;

    case 4 %% 4th order LIAO ABC Coefficients
        c1 = 4;
        c2 = 6;
        c3 = 4;
        c4 = 1;
        c5 = 0;

    case 3 %% 3rd order LIAO ABC Coefficients
        c1 = 3;
```

```
        c2 = 3;
        c3 = 1;
        c4 = 0;
        c5 = 0;

    otherwise
        disp('Error: Wrong Value')
end
end

% ************************
% ************************
%
```

Implement LIAO ABC

*********************** ***********************

```
function Liao_ABC()

global c1 c2 c3 c4 c5 ; % Define material based coefficients
global Ez ; % E and H field components.
global Ez1 Ez2 Ez3 Ez4 Ez5 % For Bubbling of E-fields in Liao ABC
global nx  ny
global ABC_order
% General BC for any order LIAO ABC
for  j = 1:ny
    Ez(1,j) = c1*Ez1(2,j)-c2*Ez2(3,j)+c3*Ez3(4,j)...
        -c4*Ez4(5,j)+c5*Ez5(6,j);  %%%left side
end
for j = 1:ny
    Ez(nx,j) = c1*Ez1(nx-1,j)-c2*Ez2(nx-2,j)+c3*Ez3(nx-3,j) ...
        -c4*Ez4(nx-4,j)+c5*Ez5(nx-5,j);  %%%right side
end
for i = 2:nx-1
    Ez(i,1) = c1*Ez1(i,2)-c2*Ez2(i,3)+c3*Ez3(i,4) ...
        -c4*Ez4(i,5)+c5*Ez5(i,6); %%%bottom
end
for i = 2:nx-1
    Ez(i,ny) = c1*Ez1(i,ny-1)-c2*Ez2(i,ny-2)+c3*Ez3(i,ny-3) ...
        -c4*Ez4(i,ny-4)+c5*Ez5(i,ny-5); %%%top
end

switch ABC_order

    case 5
        Ez5 = Ez4;
        Ez4 = Ez3;
        Ez3 = Ez2;
        Ez2 = Ez1;
        Ez1 = Ez;

    case 4
```

```
        Ez4 = Ez3;
        Ez3 = Ez2;
        Ez2 = Ez1;
        Ez1 = Ez;

    case 3
        Ez3=Ez2;
        Ez2=Ez1;
        Ez1=Ez;

    otherwise
        disp('Error: Wrong Value')
end
end

% ************************
% ************************
%
```

Create the excitation signal

```
*********************** ***********************

function Ezs = Source(n, sources)

% global Ezs
% Creates a half-sinusoidal source between the time increments
% 1 and 10.%
% When source = 1 : Sinusoid
%               2 : Gaussian
%
```

For Gaussian Source

```
if sources == 2

    xndec = 10.0;
    xn0 = 4*xndec;
    Ezs = exp(-((n-xn0)/(xndec))^2);
```

For Sinusoidal Source

```
elseif sources == 1
    if ( n >=1 && n <= 10)
        Ezs = sin(n*pi/10);
    end
end

end


% ************************
```

```
% ************************
```

Algorithm for Computing E-field

```
************************ ***********************

function adv_Ez(n, sources)
% Compute z-component of E-field
global Ez  Hx  Hy
global mediaEz
global Ca Cb
global nx ny

for i = 1 : nx
    for j = 1 : ny
        m  = mediaEz(i,j);
        if (i == 6)   %% Incident Field Source Excitation
            Es = Source(n, sources); % Es is a soft source
        else
            Es = 0;
        end
        if (i >= 2 && j >=2)
            Ez(i,j) = Ez(i,j)*Ca(m) + Cb(m)*(Hy(i,j) - Hy(i-1,j)...
                - (Hx(i,j) - Hx(i,j-1))) + Es;
        elseif (i >= 2 && j == 1) %% Field at the bottom edge of the
 boundary
            Ez(i,j) = Ez(i,j)*Ca(m) + Cb(m)*(Hy(i,j) - Hy(i-1,j)...
                - Hx(i,j)) + Es;
        end
    end
end
end


% ************************
% ************************
```

Algorithm for Computing E-field

```
************************ ***********************

function adv_H()
% Compute z-component of E-field
global Ez  Hx  Hy
global mediaHx mediaHy
global Da Db
global nx ny

% % %     Compute x-component of H-field

for i = 1 : nx
    for j = 1 : ny - 1
        m = mediaHx(i,j);
```

```
        Hx(i,j) = Hx(i,j)*Da(m) - Db(m)*(Ez(i,j+1) - Ez(i,j));
    end
end

% % %     Compute y-component of H-field

for i = 1 : nx - 1
    for j = 1 : ny
        m = mediaHy(i,j);
        Hy(i,j) = Hy(i,j)*Da(m) + Db(m)*(Ez(i+1,j) - Ez(i,j));
    end
end
end


% ************************
% ************************
```

Plot routine for the 3D surface plots

```
*********************** ***********************

function my_surface_plot(field, iflaga)
% This function generates the 3D surface plots for the field in
% the argument of the function
%

%  colormap(viridis)
colormap('jet')
global mxst mxnd myst mynd n nx ny asize ds
xd = 0:asize/(nx-1):asize; % Build the axes in physical dimensions
yd = 0:asize/(ny-1):asize;
[xdg, ydg] = meshgrid(yd, xd);
EzSurf =
 surf(xdg,ydg,field,'EdgeColor','interp','FaceLighting','gouraud');
shading interp % Avoid jittered shading
caxis([-1,1]) % set the colorbar range from -1 to 1
% rectangle('Position',[myst,mxst,mynd-myst,mxnd-
mxst],'FaceColor','r')
set(gcf,'Color','white'); % Set background color to white
title(['3D plot of $E_z$ with PEC box at time step, n =
 ',int2str(n)],'Interpreter','latex')
set (gca,'FontName','times new roman') % Set axes fonts to Times New
 Roman
ax = gca;
ax.XTick = [1 2 3 4 5];
ax.YTick = [1 2 3 4 5];
xlabel('y (meters)','Interpreter','latex'); % X-axis label
h=get(gca,'xlabel');
set(h,'rotation',-6)
ylabel('x (meters)','Interpreter','latex'); % y-axis label
h=get(gca,'ylabel');
set(h,'rotation',60)
```

```
zlabel('Amplitude ($\frac{V}{m}$)','Interpreter','latex') %z-axis
 label
box on %
% grid on
material dull; % Set reflectivity of the surface to dull
% Add three lights below tom improve visuals
% h = light;
%  light('Position',[10 0 1]);
%  light('Position',[-190 -120 1]);
%  light('Position',[0 0 1]);
%  lighting gouraud
axis ([0 asize 0 asize -1 1])
view([-165 45]); % Perspective view
% view([ 90 90 ]) % Top view
M(:,n) = getframe(gcf) ;
end

% ************************
% ************************
```

Plot routine for the 2D Line Plot of Ez

```
*********************** ***********************

function my_line_plot(Ez_5, Ez_35, Ez_65, Ez_95,iflaga, source)
% Plots the line plots for E-field
figure('Name', sprintf('Line Plots for E-field_%d',iflaga))
set(gcf,'Color','white');
title('E-field at different Time Steps');

subplot(4,1,1)
plot(Ez_5,'Color','black','LineWidth',1.2)
a = gca;
a.YMinorGrid = 'on'; % Display Minor Grid lines along the x-axis
a.XTickLabel = {}; % Hide x-labels
a.FontName = 'times new roman';
legend('n=5')
legend('boxoff')

subplot(4,1,2)
plot(Ez_35,'Color','black','LineWidth',1.2)
a = gca;
a.YMinorGrid = 'on';
a.XTickLabel = {};
a.FontName = 'times new roman';
legend('n=35')
legend('boxoff')

subplot(4,1,3)
plot(Ez_65,'Color','black','LineWidth',1.2)
a = gca;
a.YMinorGrid = 'on';
a.XTickLabel = {}; % Hide x-labels
a.FontName = 'times new roman';
```

```
legend('n=65')
legend('boxoff')

subplot(4,1,4)
plot(Ez_95,'Color','black','LineWidth',1.2)
a = gca; % Set current axis handle
a.YMinorGrid = 'on';
a.FontName = 'times new roman';
legend('n=95')
legend('boxoff')
if iflaga == 1 && source == 1 % This part creates a title for the
 subplot which can not be done by default
    set(gcf,'NextPlot','add');
    axes;
    h = title('Line Plots for E-field with no obstacle for a sinusoid
 source');
    set(gca,'Visible','off');
    set(h,'Visible','on');
elseif iflaga == 2 && source == 1
    set(gcf,'NextPlot','add');
    axes;
    h = title('Line Plots for E-field with a PEC box for a sinusoid
 source');
    set(gca,'Visible','off');
    set(h,'Visible','on');
elseif source == 2
    set(gcf,'NextPlot','add');
    axes;
    h = title('Line Plots for E-field with no obstacle for a Gaussian
 source');
    set(gca,'Visible','off');
    set(h,'Visible','on');
end
xlabel('Cells in x-direction','FontSize',11,...
    'Interpreter','latex','FontName','times new roman')
% matlab2tikz('filename',sprintf('ECEN637_HW6_a_%d_
%d.tex',iflaga,source))
saveas(gcf,[sprintf('ECEN637_HW6_a_%d',iflaga),'.eps'],'epsc') % Save
 visualizations

end
```

Routine to zero out the global variables

```
*********************** ***********************

function zeroing()
% Clears but retains the variables in memory

global nx  ny
global Ez  Hx  Hy; % Create E and H field components.
% global data type enables the global scope of the variables
% within the code. Unlike global variables they can not be accessed
% outside the code
```

```
global mediaEz mediaHx mediaHy; %
global Ca Cb Da Db ; % Define material based coefficients
global Ez1 Ez2 Ez3 Ez4 Ez5

Ez = zeros(nx,ny); %% z-component of E-field
Hx = zeros(nx,ny); %% x-component of H-field
Hy = zeros(nx,ny); %% y-component of H-field

Ez1 = zeros(nx,ny);
Ez2 = zeros(nx,ny);
Ez3 = zeros(nx,ny);
Ez4 = zeros(nx,ny);
Ez5 = zeros(nx,ny);


mediaEz = ones(nx,ny); %% z-component of E-field
mediaHx = ones(nx,ny); %% x-component of H-field
mediaHy = ones(nx,ny); %% x-component of H-field

Ca = zeros(2,1); %% x-component of H-field
Cb = zeros(2,1); %% x-component of H-field
Da = zeros(2,1); %% x-component of H-field
Db = zeros(2,1); %% x-component of H-field

end
```

Published with MATLAB® R2015b