I. Simulasi Gelombang Elektromagnetik 1 Dimensi

➢ Algoritma

1. Masukkan nilai domain komputasi/*Grid Points* (Domain Ruang).
2. Masukkan nilai atau lokasi gelombang EM sumber.
3. Masukkan nilai *Time Step* maksimum.
4. Masukkan parameter $c = 3\ x\ 10^8 m/s$, lebar pias untuk domain ruang dx, dan lebar pias untuk domain waktu dt.
5. Menginisialisasi vektor untuk menghindari *error* saat kompilasi.
6. Memasukkan parameter gelombang elektromagnetik sumber (disini menggunakan pulsa gaussian)
7. Memulai Loop untuk t=1,t=nilai step maksimum, t++
   a. Untuk k=1 ; hingga nilai domain ruang maksimum dikurangi satu; k++, melakukan perhitungan untuk *Governing Equation* (bagian medan E)*:*

$$E_x^{n+1/2}(k) = E_x^{n-1/2}(k) + \frac{1}{\sqrt{\mu_0 \varepsilon_0}} \frac{\Delta t}{\Delta z} \left(H_y^n(k-1/2) - H_y^n(k+1/2)\right)$$

   b. Untuk suatu posisi tertentu, tempatkan sejumlah gelombang elektromagnetik sumber.
   c. Untuk k=1 ; hingga nilai domain ruang maksimum dikurangi satu; k++, melakukan perhitungan untuk *Governing Equation* (bagian medan H)*:*

$$H_y^{n+1}(k+1/2) = H_y^n(k+1/2) + \frac{1}{\sqrt{\mu_0 \varepsilon_0}} \frac{\Delta t}{\Delta z} \left(E_x^{n+1/2}(k) - E_x^{n+1/2}(k+1)\right).$$

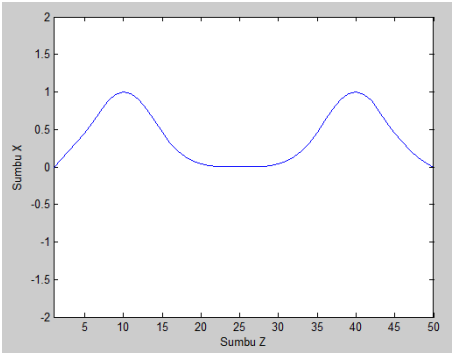
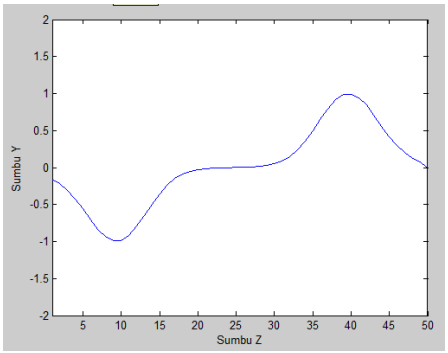
8. Plot gelombang E atau Gelombang H.

➢ Listing Program

```
clear;
ke=50;
%Posisi Gelombang Sumber
ks=ke/2;
%nilai time step maksimum
nsteps=200;
```

```
%Parameter dan juga lebar pias domain ruang dan waktu
c0=3.e8;
dx=0.01;
dt=dx/(2.*c0);
%penyederhanaan konstanta
cc=c0*dt/dx;
%inisialisasi vektor awal
ex=zeros(1,ke);
hy=zeros(1,ke);
%parameter pulsa gaussian
t0=20;
spread=8;
%memulai loop untuk domain waktu
M=moviein(nsteps);
for t=1:nsteps
% looping untuk governing equation medan E
for k=2:ke-1
ex(k)=ex(k)+cc*(hy(k-1)-hy(k));
end
%gelombang EM sumber
ex(ks)=exp(-.5*((t-t0)/spread)^2);
% looping untuk governing equation medan H
for k=1:ke-1
hy(k)=hy(k)+cc*(ex(k)-ex(k+1));
end
%Plotting gelombang Ex atau Hy
plot(ex);axis([1 ke -2 2]); %Plotting Ex
M(:,t) = getframe ;
end
```

- **Tampilan**

Simulasi Gelombang E_x	**Simulasi Gelombang H_y**
	

Analisa:

Dari program yang telah dilakukan diatas, dapat dilihat bahwa program diawali dengan memasukkan nilai yang dibutuhkan dalam melakukan komputasi seperti memasukkan nilai grid point untuk waktu dan ruang. Kemudian, program dilanjutkan dengan mensimulasikan *Governing Equation* untuk gelombang E. Lalu, tak lupa untuk menyediakan gelombang sumber yang berguna untuk meramalkan propagasi dari gelombang sumber tersebut. lalu

kemudian menyelesaikan *Governing Equation* untuk gelombang H. Dalam simulasi gelombang EM 1 Dimensi ini dihasilkan sejumlah gelombang gelombang EM yang hanya disimulasikan dalam satu arah saja. Namun, tidak menutup kemungkinan untuk mensimulasikan dalam satu plotting yang sama.

I. Simulasi Gelombang Elektromagnetik 3 Dimensi (Penerapan dalam Bahan)

➢ Algoritma

1. Memasukkan semua konstanta dan parameter. Seperti konstanta dielektrik relatif dari medium dll.
2. Memasukkan lebar pias untuk sumbu x dan y dan juga panjang dari sumbu x dan sumbu y.
3. Memasukkan posisi dari sumber gelombang.
4. Memasukkan parameter *perfectly matched layer*
5. Menginisialisasi vektor untuk menghindari *error* saat kompilasi.
6. Menghitung nilai-nilai PML yang berguna untuk perhitungan Hx, Hy, Dz, dan Ez.
7. Mengkalkulasikan Hx, Hy, Dz, dan Ez.
8. Membuat sumber gelombang berbentuk gelombang pulsa sinus.
9. Plotting gelombang Ez.

➢ Listing Program

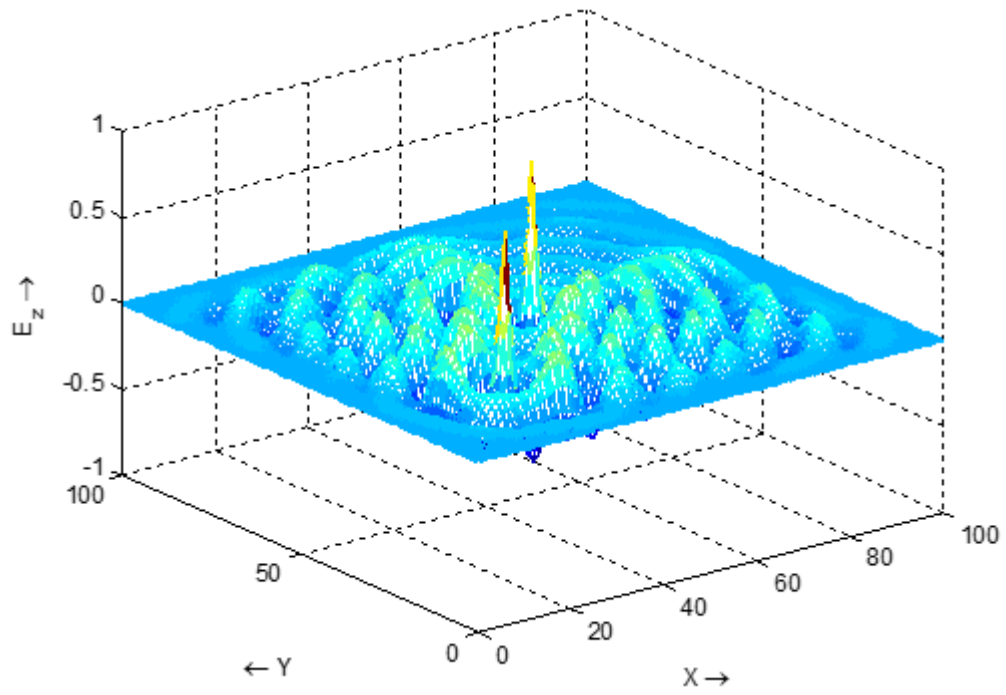
```
close all;
clear all;
clc;

%Grid Dimension in x (xdim) direction
c = 3e8;%1; % speed of light
freq_in=3e7;% frquency in Hz
eps_r = 1; % relative dielectric constant of medium

lamda = (c/freq_in)/sqrt(eps_r);
xdim = 100;
dx = lamda/10; % x-position step
x = 0:dx:xdim ;
xsteps = length(x);

%Total time
time_tot=400;
Rx = 0.5; % courant constant of stability R < 1 => dispersion, R > 1 =>
unstability
c = 3e8; % speed of light
dt = Rx*dx/c;
% t = 0:dt:time_tot;
tsteps = time_tot;

%Grid Dimension in y (ydim) direction
ydim = 100;
Ry = 0.5;
dy = c*dt/Ry;
y = 0:dy:ydim ;
ysteps = length(y);
```

```
% position of source
xsource=floor(xsteps/2);
ysource=floor(ysteps/2);
xsource2=floor(xsteps/4);
ysource2=floor(ysteps/4);
% Defining PML parameters
length_pml_x = xsteps/5; % width of PML along x-direction
length_pml_y = ysteps/5; % width of PML along y-direction

xn = zeros(1,xsteps);
yn = zeros(1,ysteps);
fi1 = zeros(1,xsteps);
fi2 = ones(1,xsteps);
fi3 = ones(1,xsteps);
gi2 = ones(1,xsteps);
gi3 = ones(1,xsteps);
fj1 = zeros(1,ysteps);
fj2 = ones(1,ysteps);
fj3 = ones(1,ysteps);
gj2 = ones(1,ysteps);
gj3 = ones(1,ysteps);

gaz = ones(ysteps,xsteps);
gbz = zeros(ysteps,xsteps);

for ii = 1:length_pml_x
    xnum = length_pml_x-ii ; % making the transition at problem space
smooth, i.e. making zero conductivity at boundary of PML and problem space
    xn(ii)=0.333*((xnum/length_pml_x)^3 );
    fi1(ii)=xn(ii); % for 1-side of PML in x-direction
    fi1(xsteps-ii)=xn(ii);  % for 2nd-side of PML in x-direction
    fi2(ii)=1/(1+xn(ii));
    fi2(xsteps-ii)=1/(1+xn(ii));
    fi3(ii)=(1-xn(ii))/(1+xn(ii));
    fi3(xsteps-ii)=(1-xn(ii))/(1+xn(ii));
    gi2(ii)=1/(1+xn(ii));
    gi2(xsteps-ii)=1/(1+xn(ii));
    gi3(ii)=(1-xn(ii))/(1+xn(ii));
    gi3(xsteps-ii)=(1-xn(ii))/(1+xn(ii));
    gaz(:,ii)=1/(1+2*xn(ii));
    gaz(:,xsteps-ii)=1/(1+2*xn(ii));
    gbz(:,ii)=2*xn(ii);
    gbz(:,xsteps-ii)=2*xn(ii);

end

for jj = 1:length_pml_y
    ynum = length_pml_y-jj ; % making the transition at problem space
smooth, i.e. making zero conductivity at boundary of PML and problem space
    yn(jj)=0.333*((ynum/length_pml_y)^3 );
    fj1(jj)=yn(jj); % for 1-side of PML in y-direction
    fj1(ysteps-jj)=yn(jj);  % for 2nd-side of PML in y-direction
    fj2(jj)=1/(1+yn(jj));
    fj2(ysteps-jj)=1/(1+yn(jj));
    fj3(jj)=(1-yn(jj))/(1+yn(jj));
    fj3(ysteps-jj)=(1-yn(jj))/(1+yn(jj));
    gj2(jj)=1/(1+yn(jj));
    gj2(ysteps-jj)=1/(1+yn(jj));
    gj3(jj)=(1-yn(jj))/(1+yn(jj));
```

```
    gj3(ysteps-jj)=(1-yn(jj))/(1+yn(jj));
    gaz(jj,:)=1/(1+2*yn(jj));
    gaz(ysteps-jj,:)=1/(1+2*yn(jj));
    gbz(jj,:)=2*yn(jj);
    gbz(ysteps-jj,:)=2*yn(jj);
end

% Initialization of field vectors
Dz=zeros(ysteps,xsteps);
Ez=zeros(ysteps,xsteps);
Hx=zeros(ysteps,xsteps);
Hy=zeros(ysteps,xsteps);

IHx = zeros(ysteps,xsteps);
IHy = zeros(ysteps,xsteps);
Iz = zeros(ysteps,xsteps);

for n = 1:tsteps
%% Implementing 2D-FDTD
 % Calculating Hx
for l = 1: xsteps
    for m = 1:ysteps-1
        curl_ex = Ez(m,l)-Ez(m+1,l);
        IHx(m,l) = IHx(m,l)+fi1(l)*curl_ex;
   Hx(m,l) = fj3(m)*Hx(m,l)+fj2(m)*Ry*(curl_ex+IHx(m,l));
    end
end

 % Calculating Hy
for m1 = 1: ysteps
    for l1 = 1:xsteps-1
   curl_ey = Ez(m1,l1+1)-Ez(m1,l1);
        IHy(m1,l1) = IHy(m1,l1)+fj1(m1)*curl_ey; %** check
   Hy(m1,l1) = fi3(l1)*Hy(m1,l1)+fi2(l1)*Rx*(curl_ey+IHy(m1,l1));
    end
end

 % Calculating Dz
for m2 = 2: ysteps
    for l2 = 2:xsteps
   Dz(m2,l2) = gi3(l2)*gj3(m2)*Dz(m2,l2)+gi2(l2)*gj2(m2)*(Rx*(Hy(m2,l2)-
Hy(m2,l2-1))-Ry*(Hx(m2,l2)-Hx(m2-1,l2))); % can make separte Rx,Ry
    end
end

% Calculating Ez
Ez = gaz.*(Dz-Iz);
Iz = Iz + gbz.*Ez;

%% creating sine source

    pulse=sin(((2*pi*(freq_in)*n*dt)));

    % % for a single Gaussian source
%     if n<=42
%         pulse =(10-15*cos(n*pi/20)+6*cos(2*n*pi/20)-cos(3*n*pi/20))/32;
%     end
%
% for peroidic Gaussian wave source
```

```
        % pulse =(10-15*cos(n*pi/20)+6*cos(2*n*pi/20)-cos(3*n*pi/20))/32;

% Another way to implement Gaussian source
% t0=40.0;
% spread=15.0;
% pulse=exp(-1.*((t0-n)./spread)^2);
%% Assigning source
    Ez(ysource,xsource) = pulse; % Hard source acts as Metal wall
    Ez(ysource2,xsource2) = pulse;  %matrix
%             Ez(ysource,xsource) =  Ez(ysource,xsource)+pulse; % soft
source passes the wave through it

%         Ez(ysource+ysource/2,xsource+xsource/2) = pulse;
%         Ez(ysource-ysource/2,xsource-xsource/2) = -1*pulse;

%         Hy(xsource) = pulse;

%% Plotting Ez-wave
    mesh(x,y,Ez,'linewidth',2);
    xlabel('X \rightarrow');
    ylabel('\leftarrow Y');
    zlabel('E_z \rightarrow');
    titlestring=['\fontsize{20}Plot of E_z vs X & Y for 2D FDTD of
sinusoidal excitation with PML at time step = ',num2str(n)];
    title(titlestring,'color','k');
    axis([0 xdim 0 ydim -1 1]);
    %goodplot()

    getframe;

end
```

- **Tampilan**



- **Analisa**

Dari program diatas, dapat dilihat bahwa program ini merupakan bentuk penerapan propagasi Gelombang EM 3 Dimensi. Program diawali dengan memasukkan terlebih dahulu konstanta dan parameter yang akan digunakan dalam perhitungan komputasi. Kemudian, program akan menghitung parameter PML untuk selanjutnya digunakan dalam iterasi perhitungan Hx, Hy, Dz, dan Ez. Dari perhitungan atau komputasi inilah kemudian didapatkan nilai Ez yang menjadi basis data akhir yang kita dapatkan yang selanjutnya kita dapat plot ke dalam grafik 3 Dimensi untuk kita lihat bentuk propagasi gelombang elektromagnetik dalam bahan yang memiliki suatu nilai konstanta dielektrik relatif tertentu. Program ini memiliki batasan yaitu pergerakan medan magnet berada pada mode TMz.

DAFTAR PUSTAKA

_____. 2014. The Finite-Difference Time-Domain Method (FDTD). http://www.ece.utah.edu/~ece6340/LECTURES/lecture%2014/FDTD.pdf (Diakses 25 Mei 2015).