FDTD Simulation of 2-D Electromagnetic Wave

Walter L Gordy

I. Introduction

THE goal of this project is to become familiar with the Finite Difference Time Domain (FDTD) algorithm by simulating a 2-D electromagnetic wave with PEC boundaries.

II. Solution Methodology

To simulate the source, two gaussian waves are used and offset from each other. One gaussian is positive and one is negitive. A similar wave could have been generated using a single sine wave, but the gaussian was choosen to keep the code very simple. The following equations are used.

$$Source = e^{-(\frac{n-n_1}{n_{half}})^2} - e^{-(\frac{n-n_2}{n_{half}})^2} \tag{1}$$

The TM_z mode is used to model the system. Since the system simulates a uniform material, C_a, C_b, D_a, D_b are constant.

$$E_z\Big|_{i-1/2,j+1/2}^{n+1/2} = C_a E_z\Big|_{i-1/2,j+1/2}^{n-1/2} + C_b\left(H_y\Big|_{i,j+1/2}^{n} - H_y\Big|_{i-1,j+1/2}^{n} + H_x\Big|_{i-1/2,j}^{n} - H_x\Big|_{i-1/2,j+1}^{n}\right) \tag{2}$$

$$H_x\Big|_{i-1/2,j+1}^{n+1} = D_a H_x\Big|_{i-1/2,j+1}^{n} + D_b\left(E_z\Big|_{i-1/2,j+1/2}^{n+1/2} - E_z\Big|_{i-1/2,j+3/2}^{n+1/2}\right) \tag{3}$$

$$H_y\Big|_{i,j+1/2}^{n+1} = D_a H_y\Big|_{i,j+1/2}^{n} + D_b\left(E_z\Big|_{i+1/2,j+1/2}^{n+1/2} - E_z\Big|_{i-1/2,j+1/2}^{n+1/2}\right) \tag{4}$$

III. Conclusion

The wave did not propogate correctly at first. The wave propogated as a 45 degree rotated square. The problem was found to be a type in code. Just a simple wrong multiplier can make everything not work in the algorithm.

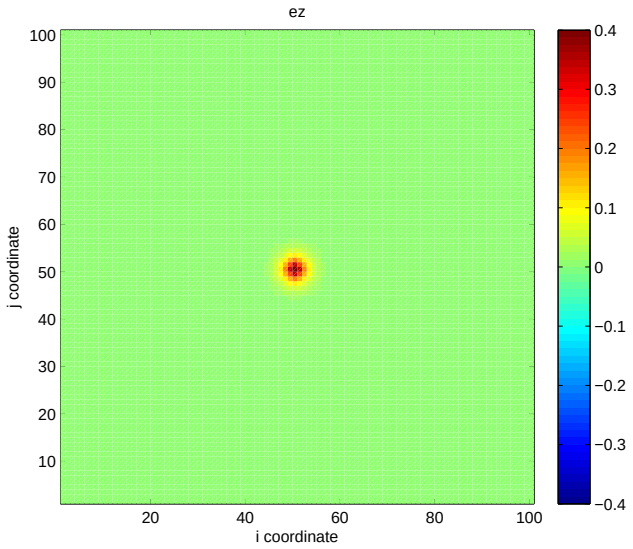

Fig. 1. EZ Field, Uniform Material, Step=10

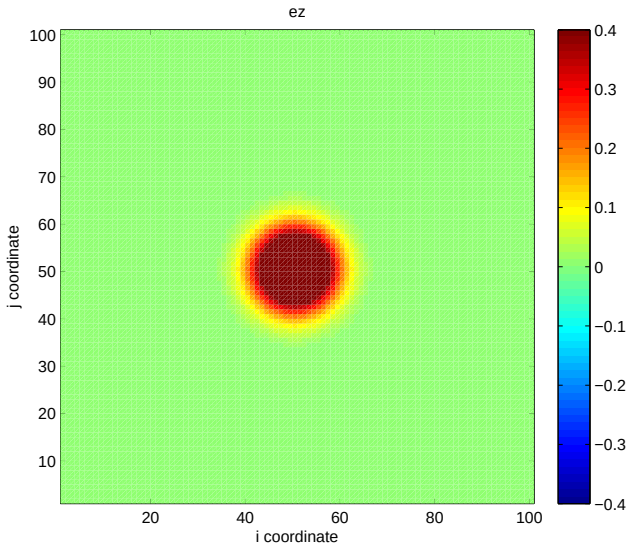

Fig. 2. EZ Field, Uniform Material, Step=25

Walter Gordy is with the Department of Electrical Engineering, University of New Mexico, Albuquerque, NM 87106.

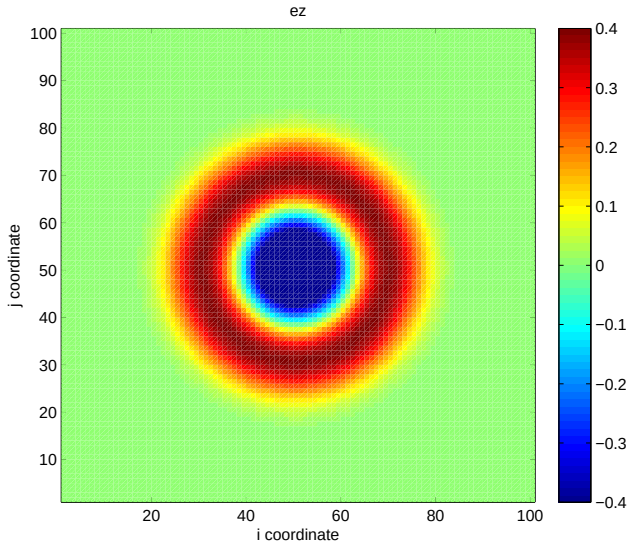

Fig. 3. EZ Field, Uniform Material, Step=50

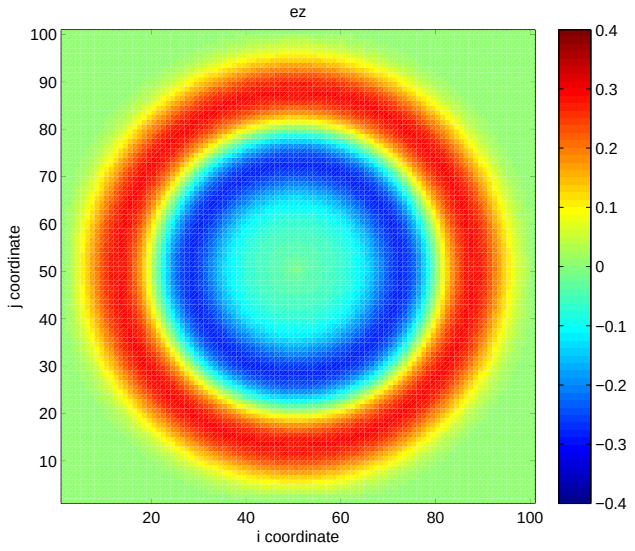

Fig. 4. EZ Field, Uniform Material, Step=75

```
clc; % clear command window
clear; % clear variables

c=2.99792458e8; % speed of light

mu=4.0*pi*1.0e-7; % free space def
eps=1.0/(c*c*mu);

freq=1e8; % frequency
ie = 100; % grids
je = 100;

steps = 75; % number of time steps to simulate

lambda = 1/freq;
dx = lambda / 20;
dy = lambda / 20;

dt = 1 / (c * sqrt(1 / dx^2 + 1 / dy^2));

% updating coefficients
ca = 1;
cb = dt/(eps*dx);
da = 1;
db =  dt/(mu*dx);
% setup arrays
ez = zeros(ie+1,je+1);
hx = zeros(ie+1,je+1);
hy = zeros(ie+1,je+1);

%fdtd algorithm
for n=1:steps
    ez(2:ie,2:je)=ca*ez(2:ie,2:je) + ...
     cb*(hx(2:ie,1:je-1)-hx(2:ie,2:je) + ...
     hy(2:ie,2:je)-hy(1:ie-1,2:je));
    % source
    ez(ie/2,je/2) = exp(-((n-20)/10)^2) - ...
     exp(-((n-40)/10)^2);

    hx(2:ie,1:je)=hx(2:ie,1:je) + ...
     db*(ez(2:ie,1:je)-ez(2:ie,2:je+1));
    hy(1:ie,2:je)=hy(1:ie,2:je) + ...
    db*(ez(2:ie+1,2:je)-ez(1:ie,2:je));
end;
%plot ez at last step
figure('position',[100,100,600,500]);
plot(ez),pcolor(ez);
shading flat;
caxis([-.4 0.4]);
colorbar;
axis image;
title(['ez']);
xlabel('i coordinate');
ylabel('j coordinate');
```

Fig. 5. MATLAB 2009b code to reproduce results.