Advances (and Surprises) in Electrodynamics - Time Domain Simulations -

Thorsten Liebig

General and Theoretical Electrical Engineering (ATE)
University of Duisburg-Essen, 47048 Duisburg, Germany

12.06.2012
Outline

1. Motivation
2. Rectangular Waveguide
3. Metamaterial and Plane Waves
Outline

1. Motivation
2. Rectangular Waveguide
3. Metamaterial and Plane Waves
• Visualize electromagnetic fields to get a deeper understanding of the different speeds of light.
• An inspiration to create your own electromagnetic simulations (maybe by using openEMS)
• Just some nice field plots to look at...
Outline

1. Motivation
2. Rectangular Waveguide
3. Metamaterial and Plane Waves
Properties of a rectangular waveguide I

1. TE- and TM-modes propagable
2. Highly dispersive wave propagation \( (\beta = \beta(\omega)) \)

**TE-10 Mode properties**
- Electric field in y-direction only \( (E_y = A \cdot \sin (\frac{\pi x}{a})) \)
- Magnetic fields in x- and z-direction
Properties of a rectangular waveguide I

(a) XY-plane

(b) XZ-plane

Figure: Field distribution (TE-10 Mode)\(^1\)

\(^1\) pictures provided by Fedor Schreiber
Motivation
Rectangular Waveguide
Metamaterial and Plane Waves

Properties
openEMS model
Simulation Results

Dispersion and velocities

Figure: Dispersion diagram for the rectangular waveguide

Figure: Phase- and group-velocity in a rectangular waveguide
Model in openEMS I

Create a rectangular waveguide model using the openEMS-Matlab-Interface:

Initialize & define the FDTD options...

```
1 % initialize the FDTD structure
2 FDTD = InitFDTD();

3 % excite the simulation using a gaussian pulse
4 % center frequency: 200e6 (200 MHz)
5 % 20dB cutoff frequency: 25e6 (25 MHz)
6 FDTD = SetGaussExcite(FDTD, 200e6, 25e6);

7 % define the boundary conditions for x_min, x_max, y_min, y_max, z_min, z_max
8 FDTD = SetBoundaryCond(FDTD, {'PEC', 'PEC', 'PEC', 'PEC', 'PEC', 'PML'});

...done!
```
Model in openEMS II

Setup your FDTD mesh & model...

```
% setup the FDTD mesh
unit=1e-3; % everything is defined in mm
a=1000; b=600; % 1000mm x 600mm waveguide
length=30e3; % 30m long waveguide

CSX = InitCSX();
mesh.x = SmoothMeshLines([0 a], 10);
mesh.y = SmoothMeshLines([0 b], 10);
mesh.z = SmoothMeshLines([0 length], 10);
CSX = DefineRectGrid(CSX, unit,mesh);

% setup your excitation (physical) property + box (geometrical) primitive
CSX = AddExcitation(CSX,'excite',0,[1 1 0]);
weight{1} = func_Ex; % for TE-10 this is '0'
weight{2} = func_Ey; % for TE-10 this is 'sin(0.0031416*x)' 
weight{3} = 0;
CSX = SetExcitationWeight(CSX,'excite',weight);

start=[0 0 0];
stop = [a b 0];
CSX = AddBox(CSX,'excite',0,start,stop);
```

...done.
Setup your dumps...

```matlab
% E-field (type 0) dump on an xz-plane (vtk file format)
CSX = AddDump(CSX, 'Et_xz', 'DumpType', 0, 'FileType', 0, 'DumpMode', 1, 'SubSampling', '2,2,2');
start = [0 b/2 0];
stop = [a b/2 length];
CSX = AddBox(CSX,'Et_xz',0 , start,stop);

% E-field (type 0) dump line along the center-axis of the waveguide (hdf5 file format)
CSX = AddDump(CSX, 'Et_z', 'DumpType', 0, 'FileType', 1, 'DumpMode', 1);
start = [a/2 b/2 0];
stop = [a/2 b/2 length];
CSX = AddBox(CSX,'Et_z',0 , start,stop);

...done.
```
That’s it! Write file and run...

```matlab
WriteOpenEMS([Sim_Path '//' Sim_CSX], FDTD, CSX);
RunOpenEMS(Sim_Path, Sim_CSX)
```

Do your post-processing...

```matlab
% Read the center-axis dump and display time-domain fields in a matlab figure
[Et_field Et_mesh] = ReadHDF5Dump([Sim_Path '//' Et_z.h5]);
mesh_z = Et_mesh.lines{3};
for n = 1: numel(Et_field.TD.values)
    Ey = squeeze(Et_field.TD.values{n}(1,1,:,:)); % reduce dimensions to Ey(z)
    plot(mesh_z, Ey); % plot Ey over z-coordinates
    pause(0.1);
end
```

to be continued ...
Figure: Paraview: Gaussian pulse in a time-domain numerical simulation
Another look at the dispersion and velocities:

**Figure:** Dispersion diagram for the rectangular waveguide

**Figure:** Phase- and group-velocity in a rectangular waveguide
Time domain results II

Figure: Gaussian pulse in a time-domain numerical simulation.
Outline

1. Motivation
2. Rectangular Waveguide
3. Metamaterial and Plane Waves
Metamaterial Properties for Drude type materials

\[ \varepsilon_r = \varepsilon_{r,\infty} \left(1 - \frac{f_{pe}^2}{f^2}\right) \quad \text{and} \quad \mu_r = \mu_{r,\infty} \left(1 - \frac{f_{pm}^2}{f^2}\right) \]

with the plasma frequency \( f_{pe} \) and \( f_{pm} \).

- Both \( \varepsilon_r \) and \( \mu_r \) are frequency dependent and can be negative for frequencies below their respective plasma frequencies.
- In case of \( \varepsilon_r < 0 \) and \( \mu_r < 0 \) the refractive index is negative as well.
Material Dispersion

Figure: Dispersion diagram

Figure: Material constants and refractive index $n$
Phase and group velocity

Figure: Phase and group velocity
Simulate a plane wave in openEMS:

```matlab
1  % initialize the FDTD structure and set the max number of timesteps to "numTS"
2  % and the end criteria to 1e-5
3  FDTD = InitFDTD(numTS, 1e-5);

4  % excite the simulation using a gaussian pulse
5  FDTD = SetGaussExcite(FDTD, f0, f0/sqrt(2));

6  % define the boundary conditions for x_min, x_max, y_min, y_max, z_min, z_max
7  FDTD = SetBoundaryCond(FDTD, {'PMC', 'PMC', 'PEC', 'PEC', 'MUR', 'MUR'});

...done!
```
Setup your FDTD mesh & model:

```matlab
% setup the FDTD mesh
CSX = InitCSX();
mesh.x = -width/2 : mesh_res : width/2;
mesh.y = -height/2 : mesh_res : height/2;
mesh.z = -length/2 : mesh_res : length/2;
CSX = DefineRectGrid(CSX, 1e-3,mesh);

% setup your excitation
CSX = AddExcitation(CSX, 'excite', 0, [0 1 0]); % excite E_y
start = [-width/2 -height/2 mesh.z(3)];
stop = [ width/2 height/2 mesh.z(3)];
CSX = AddBox(CSX, 'excite', 0, start, stop);

% apply drude material
CSX = AddLorentzMaterial(CSX, 'drude');
CSX = SetMaterialProperty(CSX, 'drude', 'Epsilon', 2, 'Mue', 2);
CSX = SetMaterialProperty(CSX, 'drude', 'EpsilonPlasmaFrequency', MTM.f0);
CSX = SetMaterialProperty(CSX, 'drude', 'MuePlasmaFrequency', MTM.f0);
start = [mesh.x(1) mesh.y(1) -MTM.length/2];
stop = [mesh.x(end) mesh.y(end) MTM.length/2];
CSX = AddBox(CSX, 'drude', 10, start, stop);
```
Setup your dumps:

29  \% E-field (type 0) dump on an xz-plane (vtk file format)
30  CSX = AddDump(CSX, 'Et_xz', 'SubSampling', '2,2,2');
31  start = [mesh.x(1) 0 mesh.z(1)];
32  stop = [mesh.x(end) 0 mesh.z(end)];
33  CSX = AddBox(CSX,'Et_xz',0 , start,stop);

35  \% E-field (type 0) dump line along the center-axis (x=y=0) (hdf5 file format)
36  CSX = AddDump(CSX,'Et_z', 'FileType', 1);
37  start = [0 0 mesh.z(1)];
38  stop = [0 0 mesh.z(end)];
39  CSX = AddBox(CSX,'Et_z',0 , start,stop);
That’s it! Write file and run...

```matlab
WriteOpenEMS([Sim_Path '/ Sim_CSX], FDTD, CSX);
RunOpenEMS(Sim_Path, Sim_CSX)
```

Do your post-processing...

```matlab
% Read the center-axis dump and display frequency-domain fields in a matlab figure
[Et_field Et_mesh] = ReadHDF5Dump([Sim_Path '/Et_z.h5'], 'Frequency', [f_m f_0 f_p]);
mesh_z = Et_mesh.lines{3};
phase = linspace(0, 360, 51);
phase = phase(1:end-1);

for n=1:numel(Et_field.FD.values) % loop through all frequencies
    Ey = squeeze(Et_field.FD.values{n}(1,1,:),2)); %reduce dimensions to Ey(z)
    for p = phase %loop through phase
        plot(mesh_z, real(Ey * exp(1i*p*pi/180)));
        pause(0.1);
    end
end
```
Time domain results

Figure: Gaussian pulse in a time-domain numerical simulation
Material Dispersion

A closer look at the dispersion and materials constants:

Figure: Dispersion diagram

Figure: Material constants and refractive index.
The perfect lens - The wave front...

Figure: The wave front travels through the lens. Source: www.trnmag.com/Pendry-perfect-lens-diagram.gif
The perfect lens - Refocus Effect?

Figure: The perfect lens effect? Source: www.trnmag.com/Pendry-perfect-lens-diagram.gif
For further information:

www.ate.uni-due.de

http://openEMS.de

Thank you for your attention!