Advanced Computation:
Computational Electromagnetics

Implementation of Finite-Difference Frequency-Domain

Outline

• Basic flow of FDFD
• 2× grid technique
• Calculating grid parameters
• Constructing your device on the grid
• Walkthrough of the FDFD algorithm
• Examples for benchmarking
Basic Flow of FDFD

Block Diagram of FDFD Implementation

[Diagram showing the process steps of FDFD implementation, including:
- Dashboard: Source, device, learn, and grid
- Calculate Grid: Ns, Ny, dx, dy, xa, ya
- Build Device on Grid
- Incorporate PML
- Calculate Wave Vector Components: k_xin, k_yin, k_x(m), k_y(m), k_in(m)
- Build Wave Matrix A
- Compute Source: b = (QA - AQ)\(\mathbf{f}_{sc}\)
- Solve: \(\mathbf{Ax} = \mathbf{b}\)
- Post Process: RDE, TDE, REF, TRN, CON]
Detailed FDFD Algorithm

1. Construct FDFD Problem
   a. Define your problem
   b. Calculate the grid parameters
   c. Assign materials to the grid to build ER2 and UR2 arrays

2. Handle PML and Materials
   a. Compute $s_x$ and $s_y$
   b. Incorporate into $\varepsilon_i$ and $\mu_i$
   c. Overlay onto 1× grids

3. Compute Wave Vector Components
   a. Identify materials in reflected and transmitted regions: $\varepsilon_{ref}$, $\varepsilon_{trn}$, $\mu_{ref}$, $\mu_{trn}$, $n_{ref}$, $n_{trn}$
   b. Compute incident wave vector: $k_{inc}$
   c. Compute transverse wave vector expansion: $k_{x,ref}$, $k_{x,trn}$
   d. Compute $k_{y,ref}$ and $k_{y,trn}$

4. Construct $A$
   a. Construct diagonal materials matrices
   b. Compute derivative matrices
   c. Compute $A$

5. Compute Source Vector, $b$
   a. compute source field $f_{src}$
   b. compute $Q$
   c. compute source vector $b$

6. Solve Matrix Problem: $e = A^{-1}b$

7. Post Process Data
   a. Extract $E_{ref}$ and $E_{trn}$
   b. Remove phase tilt
   c. FFT the fields
   d. Compute diffraction efficiencies
   e. Compute reflectance & transmittance
   f. Compute conservation of power

2× Grid Technique
What is the 2× Grid Technique? (1 of 2)

This is the traditional approach for building devices on a Yee grid.

It is very tedious and cumbersome to determine which field components reside in which material.

What is the 2× Grid Technique? (2 of 2)

The 2× grid technique simplifies how devices are built into the permittivity and permeability arrays.
Block Diagram of FDFD With 2x Grid

Dashboard
Source, device, learn, and grid

Calculate Grid
Nx, Ny, dx, dy, xa, ya

Build Device on 2x Grid
ER2
UR2

Incorporate PML on 2x Grid

Parse to 1x Grid

Diagonalize Materials

All hard-coded numbers. No work. No hard-coded numbers. All work.

Calculate Wave Vector Components
k_x, k_y, k_z, k_x (m), k_y (m), k_z (m)

Build Derivate Matrices
D_x^e, D_y^e, D_z^e and D_x^h

Build Wave Matrix A
A_x = D_x^e \mu_x D_x^e + D_y^e \mu_y D_y^e + \varepsilon_x
A_y = D_x^e \varepsilon_x D_x^e + D_y^e \varepsilon_y D_y^e + \mu_y

Compute Source
b = (QA - AQ) f_{src}

Solve Ax = b

Post-Process
ROE, TDE, REF, TRN, CON

Recall the Yee Grid

1D Yee Grid

E_x Mode

2D Yee Grids

E_z Mode

3D Yee Grid

E_z Mode

Hz Mode
The field components are physically positioned at the edges of the cell. The simplified representation shows the fields inside the cells to convey more clearly which cell they are in.
Simplified Representation of H-Field Components

- The field components are physically positioned at the edges of the cell.
- The simplified representation shows the fields inside the cells to convey more clearly which cell they are in.

The 2× Grid

The Conventional 1× Grid

Due to the staggered nature of the Yee grid, we are effectively getting twice the resolution.

It now makes sense to talk about a grid that is at twice the resolution, the “2× grid.”

The 2× grid concept is useful because we can create devices (or PMLs) on the 2× grid without having to think about where the different field components are located. In a second step, we can easily pull off the values from the 2× grid where they exist for a particular field component.
2×→1× (1 of 4): Define Grids

Suppose a circle is to be built onto the Yee grid. First, define the standard 1x grid. Second, define a corresponding 2x grid.

Note: the 2x grid represents the same physical space, but with twice the number of points.

2×→1× (2 of 4): Build Device

Third, we construct a cylinder on the 2x grid without having to consider anything about the Yee grid. Fourth, if desired we could perform dielectric averaging on the 2x grid at this point.

Note: This is discussed more in Lecture 14 FDFD Extras.
Recall the relation between the 2× and 1× grids as well as the location of the field components.

2×→1× (3 of 4): Recall Field Staggering

Parse to 1× Grid

2×→1× (4 of 4): Parse to 1× Grid
Extract $\text{ER}_{xx}$ from $\text{ER}_2$

$\text{ER}_{xx} = \text{ER}_2(2:2:Nx2, 1:2:Ny2)$

Extract $\text{ER}_{yy}$ from $\text{ER}_2$

$\text{ER}_{yy} = \text{ER}_2(1:2:Nx2, 2:2:Ny2)$
Extract $\text{ER}_{zz}$ from $\text{ER}_2$

$$\text{ER}_{zz} = \text{ER}_2(1:2:N_x2, 1:2:N_y2)$$

Extract $\text{UR}_{xx}$ from $\text{UR}_2$

$$\text{UR}_{xx} = \text{UR}_2(1:2:N_x2, 2:2:N_y2)$$
Extract $UR_{yy}$ from $UR_2$

$UR_{yy} = UR_2(2:2:Nx2,1:2:Ny2)$

Extract $UR_{zz}$ from $UR_2$

$UR_{zz} = UR_2(2:2:Nx2,2:2:Ny2)$
MATLAB Code for Parsing Onto $1 \times$ Grid

\[
\begin{align*}
ER_{xx} &= ER2(2:2:Nx2,1:2:Ny2,1:2:Nz2); \\
ER_{yy} &= ER2(1:2:Nx2,2:2:Ny2,1:2:Nz2); \\
ER_{zz} &= ER2(1:2:Nx2,1:2:Ny2,2:2:Nz2); \\
UR_{xx} &= UR2(1:2:Nx2,2:2:Ny2,2:2:Nz2); \\
UR_{yy} &= UR2(2:2:Nx2,1:2:Ny2,2:2:Nz2); \\
UR_{zz} &= UR2(2:2:Nx2,2:2:Ny2,1:2:Nz2); \\
\end{align*}
\]

Calculating the Grid Parameters

\[
\begin{align*}
ER_{zz} &= ER2(1:2:Nx2,1:2:Ny2); \\
UR_{xx} &= UR2(1:2:Nx2,2:2:Ny2); \\
UR_{yy} &= UR2(2:2:Nx2,1:2:Ny2); \\
\end{align*}
\]
Consideration #1: Wavelength $\lambda_0$

The grid resolution must be sufficient to resolve the shortest wavelength.

First, determine the smallest wavelength:

$$\lambda_{\text{min}} = \frac{\min[\lambda_0]}{\max[n(x, y)]}$$

$\min[\lambda_0]$ is the shortest wavelength of interest in the simulation. $\max[n(x, y)]$ is the largest refractive index found anywhere in the grid.

Second, resolve the wave with at least $N_\lambda$ cells.

$$\Delta_{\lambda} \approx \frac{\lambda_{\text{min}}}{N_{\lambda}} \quad N_{\lambda} \geq 10$$

<table>
<thead>
<tr>
<th>$N_{\lambda}$</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 to 20</td>
<td>Low contrast dielectrics</td>
</tr>
<tr>
<td>20 to 30</td>
<td>High contrast dielectrics</td>
</tr>
<tr>
<td>40 to 60</td>
<td>Most metallic structures</td>
</tr>
<tr>
<td>100 to 200</td>
<td>Plasmonic devices</td>
</tr>
</tbody>
</table>
Consideration #2: Mechanical Features

The grid resolution must be sufficient to resolve the smallest mechanical features of the device.

First, determine the smallest feature:

\[ d_{\text{min}} \]

Second, resolve \( d_{\text{min}} \) by at least \( N_{d} \) cells (usually 1 to 4)

\[ \Delta_d \approx \frac{d_{\text{min}}}{N_{d}} \quad N_{d} \geq 1 \]

Calculating the Initial Grid Resolution

1. Must resolve the minimum wavelength.

\[ \Delta_x \leq \frac{\min(\lambda_{0}) + \max[n(x,y)]}{N_{x}} \quad N_{x} \geq 10 \]

Note: If you are performing a parameter sweep over frequency or wavelength, \( \min(\lambda_{0}) \) is the shortest wavelength in the sweep.

2. Must resolve the minimum structural dimension.

\[ \Delta_y \leq \frac{\ell_{\text{min}}}{N_{y}} \quad N_{y} \geq 1 \]

Proceed with the smallest of the above quantities to be our initial grid resolution

\[ \Delta_{\text{init}} = \Delta_{y} = \min[\Delta_{x}, \Delta_{y}] \]
Resolving Critical Dimensions (1 of 3)

The actual dimensions of the device we wish to simulate have not actually yet been considered. This means we likely cannot resolve the exact dimensions of a device.

Suppose we wish to place a device of length \( d \) onto a grid.

\[
\Delta_x \quad \Delta_y \quad d
\]

Not an exact fit. Cannot fill a half of a cell.

Resolving Critical Dimensions (2 of 3)

To fix this, we first calculate how many cells \( N \) are needed to resolve the most important dimension. In this case, let this be \( d \).

\[
N = \frac{d}{\Delta_x}
\]

Second, we determine how many cells we wish to exactly resolve \( d \). We do this by rounding \( N \) up to the nearest integer.

\[
N' = \text{ceil} \left( \frac{d}{\Delta_x} \right)
\]

\[
N' = 11 \text{ cells}
\]
Resolving Critical Dimensions (3 of 3)

Third, we adjust the value of $\Delta_\times$ to represent the dimension $d$ exactly.

$$\Delta'_\times = \frac{d}{N'}$$

This step can be called “snapping” the grid to a critical dimension.

Unfortunately, using a uniform grid, it is only possible to do this for one dimension per axis.

“Snap” Grid to Critical Dimensions

Decide what dimensions along each axis are critical.

- $d_\times$ and $d_\gamma$
  - Typically this is a lattice constant or grating period along $x$
  - Typically this is a film thickness or grating depth along $\gamma$

Compute how many grid cells comprise $d_\times$ and $d_\gamma$ and round UP.

$$M_\times = \text{ceil}(d_\times / \Delta_\times)$$
$$M_\gamma = \text{ceil}(d_\gamma / \Delta_\gamma)$$

Adjust grid resolution to fit these critical dimensions in grid EXACTLY.

$$\Delta'_\times = d_\times / M_\times$$
$$\Delta'_\gamma = d_\gamma / M_\gamma$$

For simplicity, drop the prime symbol ' from $\Delta_\times$ and $\Delta_\gamma$. 
Compute Total Grid Size

- Don’t forget to add cells for PML!
- Must often add “space” between PML and device.

Note: This is particularly important when modeling devices with large evanescent fields.

\[ N_x = \frac{\Lambda_x}{\Delta_x} \]
\[ N_y = \frac{\Lambda_y}{\Delta_y} + 2N_{PML} + 2N_{SPACE} \]
\[ N_{SPACE} \approx \text{ceil} \left( \frac{\lambda_{\text{max}}}{n_{\text{bulk}} \Delta_y} \right) \]

Easiest to make \( N_x \) odd. Reason discussed later.

Compute 2× Grid Parameters

\[ N_{x2} = 2 \times N_x; \quad dx_{2} = dx/2; \]
\[ N_{y2} = 2 \times N_y; \quad dy_{2} = dy/2; \]
Constructing a Device on the Grid

Reducing 3D Problems to 2D

Representation on a Cartesian grid
Building Rectangular Structures

For rectangular structures, considering calculating start and stop indices.

Be very careful with how many points on the grid you fill in!

ny2 = ny1 + round(d/dy2) - 1;

Without subtracting 1 here, filling in ny1 to ny2 would include an extra cell. This can introduce errors into your results.

% BUILD DEVICE
ER2 = er1*ones(Nx2,Ny2);
ER2(nx1:nx2,ny1:ny2) = er;
ER2(nx5:nx6,ny1:ny2) = er;
ER2(:,ny3:ny4) = er;
ER2(:,ny4+1:Ny2) = er2;

%fill everywhere with er1
%add tooth 1
%add tooth 2
%add substrate
%fill transmission region
Critical and Non-Critical Parameters

Position offset is not critical as long as sufficient space remains between device and PMLs.

Device dimensions are critical to get correct!

Oh Yeah, Metals!

Perfect Electric Conductors

\[ \varepsilon_r = -10000 \quad \text{or} \quad \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} E_x \\ E_y \\ E_z \\ f_z \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} \rightarrow E_w = 0 \]

Include Tangential Fields at Boundary (TM modes!)

Bad placement of metals

Good placement of metals
Walkthrough of the FDFD Algorithm

Input to the FDFD Algorithm

The FDFD algorithm requires the following information:

• The materials on the 2× grid: \( ER_2(n_x, n_y) \) and \( UR_2(n_x, n_y) \)
• The grid resolution: \( dx \) and \( dy \)
• The size of the PML on the 1× grid: \( NYLO \) and \( NYHI \)
• The source wavelength, \( \lambda_0 \)
• Angle of incidence, \( \theta \)
• Mode/polarization: ‘E’ or ‘H’
(1) Determine the Material Properties in the Reflected and Transmitted Regions

If these parameters are not provided in the dashboard, or a dashboard does not exist, they can be pulled directly off the grid.

(2) Compute the PML Parameters on 2× Grid

\[ s_x(x, y) \]
\[ s_y(x, y) \]
(3) Incorporate the PML

We can incorporate the PML parameters into $[\mu]$ and $[\varepsilon]$ as follows:

$$\nabla \times \vec{E} = k_0 [\mu_s'] \vec{H}$$
$$\nabla \times \vec{H} = k_0 [\varepsilon_s'] \vec{E}$$

For 2D simulations, $s_z = 1$ and we have:

$$\varepsilon'_{sx} = \frac{s_x}{s_s} \varepsilon_r$$
$$\mu'_{sx} = \frac{s_x}{s_s} \mu_r$$
$$\varepsilon'_{sy} = \frac{s_y}{s_s} \varepsilon_r$$
$$\mu'_{sy} = \frac{s_y}{s_s} \mu_r$$
$$\varepsilon'_{zz} = s_x s_y \varepsilon_r$$
$$\mu'_{zz} = s_x s_y \mu_r$$

Note: the PML is incorporated into the 2x grid.

% INCORPORATE PML
URxx = UR2.*sx.*sy;
URyy = UR2.*sx./sy;
URzz = UR2.*sx.*sy;
ERxx = ER2./sx.*sy;
ERyy = ER2.*sx.*sy;
ERzz = ER2.*sx.*sy;

(4) Overlay Materials Onto 1x Grids

Field and materials assignments

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>II</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>$E_x$</td>
<td>$E_z$</td>
</tr>
<tr>
<td>$\varepsilon_r$</td>
<td>$\varepsilon_{sx}$</td>
<td>$\varepsilon_{sz}$</td>
</tr>
<tr>
<td></td>
<td>$H_x,E_y$</td>
<td>$H_y,E_z$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>III</th>
<th>IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_x,E_y$</td>
<td>$H_z$</td>
</tr>
<tr>
<td>$\mu_{sx}$, $\varepsilon_{sx}$</td>
<td>$\mu_{sz}$</td>
</tr>
</tbody>
</table>

% OVERLAY MATERIALS ONTO 1X GRID
URxx = URxx(1:2:Nx2,2:2:Ny2);
URyy = URyy(1:2:Nx2,1:2:Ny2);
URzz = URzz(1:2:Nx2,1:2:Ny2);
ERxx = ERxx(1:2:Nx2,2:2:Ny2);
ERyy = ERyy(1:2:Nx2,1:2:Ny2);
ERzz = ERzz(1:2:Nx2,1:2:Ny2);
(5) Compute the Wave Vector Terms

\[ k_0 = \frac{2\pi}{\lambda_0} \]

\[ \vec{k}_{\text{inc}} = k_0 n_{\text{ref}} \begin{bmatrix} \sin \theta \\ \cos \theta \end{bmatrix} \]

This is a vector quantity

\[ k_x (m) = k_{x,\text{inc}} - m \frac{2\pi}{\Lambda_x} \]

\[ m = \ldots, -2, -1, 0, 1, 2, \ldots \]

\[ k_y,\text{ref} (m) = \sqrt{(k_0 n_{\text{ref}})^2 - k_x^2 (m)} \]

\[ k_y,\text{trn} (m) = \sqrt{(k_0 n_{\text{trn}})^2 - k_x^2 (m)} \]

These equations come from the dispersion equation for the reflected and transmitted regions.

Recall that there used to be a negative sign here. We are able to drop it as long as we also drop the negative sign when calculating diffraction efficiency.

(6) Construct Diagonal Materials Matrices

\[ \varepsilon_{xx} = \begin{bmatrix} \varepsilon_{xx} (1) & 0 & \cdots & 0 \\ 0 & \varepsilon_{xx} (2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \varepsilon_{xx} (N) \end{bmatrix} \]

\[ \mu_{xx} = \begin{bmatrix} \mu_{xx} (1) & 0 & \cdots & 0 \\ 0 & \mu_{xx} (2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_{xx} (N) \end{bmatrix} \]

\[ \varepsilon_{yy} = \begin{bmatrix} \varepsilon_{yy} (1) & 0 & \cdots & 0 \\ 0 & \varepsilon_{yy} (2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \varepsilon_{yy} (N) \end{bmatrix} \]

\[ \mu_{yy} = \begin{bmatrix} \mu_{yy} (1) & 0 & \cdots & 0 \\ 0 & \mu_{yy} (2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_{yy} (N) \end{bmatrix} \]

\[ \varepsilon_{zz} = \begin{bmatrix} \varepsilon_{zz} (1) & 0 & \cdots & 0 \\ 0 & \varepsilon_{zz} (2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \varepsilon_{zz} (N) \end{bmatrix} \]

\[ \mu_{zz} = \begin{bmatrix} \mu_{zz} (1) & 0 & \cdots & 0 \\ 0 & \mu_{zz} (2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_{zz} (N) \end{bmatrix} \]

\[ \text{ERxx} = \text{diag} \left( \text{sparse}(\text{ERxx}(::)) \right) \]
(7) Construct the Derivative Matrices

\[ D_x', D_y', D_x, \text{ and } D_y' \]

\[ [D_X, D_Y, D_X, D_Y] = yeeder(NGRID, k0*RES, BC, kinc/k0); \]

Be sure this function uses sparse matrices from the very beginning.

\[ A = \text{sparse}(M,M) \quad \text{creates a sparse } M\times M \text{ matrix of zeros.} \]

\[ A = \text{spdiags}(b,d,A) \quad \text{Add array } b \text{ to diagonal } d \text{ in matrix } A. \]

Frequency (or wavelength) information is incorporated into FDFD here.

(8) Compute the Wave Matrix A

**E Mode**

\[ A_E = D_{x'}^h \mu_{yy}^{-1} D_{x'}^e + D_{y'}^h \mu_{xx}^{-1} D_{y'}^e + \epsilon_{zz} \]

**H Mode**

\[ A_H = D_{x'}^e \epsilon_{yy}^{-1} D_{x'}^h + D_{y'}^e \epsilon_{xx}^{-1} D_{y'}^h + \mu_{zz} \]
(9) Compute the Source Field

\[ f_{\text{src}}(x, y) = \exp\left(j k_{\text{inc}} \cdot \vec{r}\right) \]

\[ = \exp\left[j \left(k_{x,\text{inc}} x + k_{y,\text{inc}} y\right)\right] \]

The source has an amplitude of 1.0

Don’t forget to make \( f_{\text{src}} \) a column vector.

(10) Compute the Scattered-Field Masking Matrix, \( Q \)

\[
Q = \text{diag}\left(\text{sparse}\{Q(:)\}\right) ;
\]

It is good practice to make the scattered-field region at least one cell larger than the \( y \)-low PML.
(11) Compute the source vector, \( b \)

\[
Af = b = (QA - AQ) f_{src}
\]

(12) Compute the Field \( f \)

\[
f = A^{-1} b
\]

Aside

In MATLAB, \( f = A \backslash b \) employs a direct LU decomposition to calculate \( f \). This is very stable and robust, but a half-full matrix is created so memory can explode for large problems.

Iterative solutions can be faster and require much less memory, but they are less stable and may never converge to a solution.

Correcting these problems requires significant modification to the FDFD algorithm taught here.

Don’t forget to \texttt{reshape()} \( f \) from a column vector to a 2D grid after the calculation.
(13) Extract Transmitted and Reflected Fields

The reflected field is extracted from inside the scattered-field, but outside the PML.

The transmitted field is extracted from the grid after the device, but outside the PML.

(14) Remove the Phase Tilt

Recall Bloch’s theorem,
\[ \vec{E}(\vec{r}) = \frac{\bar{A}}{k \rho} (\vec{r}) \cdot e^{ikx} \]

This implies the transmitted and reflected fields have the following form
\[ E_{\text{ref}}(x) = A_{\text{ref}}(x) e^{jkx} \]
\[ E_{\text{trn}}(x) = A_{\text{trn}}(x) e^{jkx} \]

We remove the phase tilt to calculate the amplitude terms.
\[ A_{\text{ref}}(x) = E_{\text{ref}}(x) e^{-jkx} \]
\[ A_{\text{trn}}(x) = E_{\text{trn}}(x) e^{-jkx} \]
(15) Calculate the Complex Amplitudes of the Spatial Harmonics

Recall that the plane wave spectrum is the Fourier transform of the field.

We calculate the FFT of the field amplitude arrays.

\[
S_{\text{ref}}(m) = \text{FFT}\left\{ A_{\text{ref}}(x) \right\}
\]

\[
S_{\text{tm}}(m) = \text{FFT}\left\{ A_{\text{tm}}(x) \right\}
\]

Some FFT algorithms (like MATLAB) require that you divide by the number of points and shift after calculation.

\[
S_{\text{ref}} = \text{flipud(fftshift(fft(Aref))))/Nx;}
\]

\[
S_{\text{tm}} = \text{flipud(fftshift(fft(Atrn))))/Nx;}
\]

(16) Calculate Diffraction Efficiencies

The source wave was given unit amplitude so

\[|S_{\text{inc}}|^2 = 1\]

The diffraction efficiencies of the reflected modes are then

\[
R(m) = |S_{\text{ref}}(m)|^2 \frac{\text{Re}\left[ k_{\text{inc}}^m(m)/\mu_{\text{inc}} \right]}{\text{Re}\left[ k_{\text{inc}}^m/m_{\text{inc}} \right]} \quad \text{E Mode}
\]

\[
R(m) = |U_{\text{ref}}(m)|^2 \frac{\text{Re}\left[ k_{\text{inc}}^m(m)/\epsilon_{\text{inc}} \right]}{\text{Re}\left[ k_{\text{inc}}^m/\epsilon_{\text{inc}} \right]} \quad \text{H Mode}
\]

The diffraction efficiencies of the transmitted modes are then

\[
T(m) = |S_{\text{tm}}(m)|^2 \frac{\text{Re}\left[ k_{\text{inc}}^m/m_{\text{inc}} \right]}{\text{Re}\left[ k_{\text{inc}}^m/\mu_{\text{inc}} \right]} \quad \text{E Mode}
\]

\[
T(m) = |U_{\text{tm}}(m)|^2 \frac{\text{Re}\left[ k_{\text{inc}}^m/\epsilon_{\text{inc}} \right]}{\text{Re}\left[ k_{\text{inc}}^m/\epsilon_{\text{inc}} \right]} \quad \text{H Mode}
\]

Note: these equations assume that \[|S| = 1\].
(17) Calculate Reflectance, Transmittance, and Conservation of Power

The overall reflectance is
\[ \text{REF} = \sum_{m} R(m) \]

The overall transmittance is
\[ \text{TRN} = \sum_{m} T(m) \]

Conservation of power is computed as
\[ \text{REF} + \text{TRN} + \text{ABS} = 100\% \]

If no loss or gain is incorporated, then \( \text{ABS} = 0 \) and we will have

- \( \text{REF} + \text{TRN} < 100\% \) loss
- \( \text{REF} + \text{TRN} = 100\% \) no loss or gain
- \( \text{REF} + \text{TRN} > 100\% \) gain

Remember the Third Dimension!

We grabbed a unit cell of a 3D device.

We represented it on a 2D grid.

We simulated it on a 2D grid.

The field is interpreted as infinitely extruded along the third dimension.
Examples for Benchmarking

Air Simulation

$\mu = 1.0$

$\varepsilon_t = 1.0$

$L = 1.0\lambda_0$

-30°

$E$ and $H$ Mode

<table>
<thead>
<tr>
<th>$m$</th>
<th>$R$</th>
<th>$T$</th>
<th>$R$</th>
<th>$T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>-3</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>-2</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>-1</td>
<td>0%</td>
<td>100%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>0</td>
<td>0%</td>
<td>100%</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>2</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>3</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>4</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>
Dielectric Slab Grating

\[ \theta \]

\[ \mu_r = 1.0 \]
\[ \varepsilon_r = 9.0 \]

<table>
<thead>
<tr>
<th>( \theta )</th>
<th>( R )</th>
<th>( T )</th>
<th>( R )</th>
<th>( T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>-60°</td>
<td>49.5%</td>
<td>50.6%</td>
<td>5.0%</td>
<td>95.4%</td>
</tr>
<tr>
<td>-25°</td>
<td>28.6%</td>
<td>71.4%</td>
<td>21.8%</td>
<td>78.4%</td>
</tr>
<tr>
<td>0°</td>
<td>25.1%</td>
<td>75.0%</td>
<td>25.1%</td>
<td>75.1%</td>
</tr>
<tr>
<td>15°</td>
<td>26.3%</td>
<td>73.8%</td>
<td>24.0%</td>
<td>76.3%</td>
</tr>
<tr>
<td>45°</td>
<td>37.2%</td>
<td>62.9%</td>
<td>13.9%</td>
<td>86.4%</td>
</tr>
<tr>
<td>80°</td>
<td>77.7%</td>
<td>21.8%</td>
<td>8.3%</td>
<td>91.9%</td>
</tr>
</tbody>
</table>

Note: you can come up with your own benchmarking examples using the transfer matrix method!

\[ \theta = 45° \]

Binary Diffraction Grating

\[ \mu_{11} = 1.0 \]
\[ \varepsilon_{11} = 1.0 \]

\[ w = 0.5L \]
\[ \delta = 0.25\lambda_0 \]

<table>
<thead>
<tr>
<th>( m )</th>
<th>( \mu_{11} )</th>
<th>( \varepsilon_{11} )</th>
<th>( R )</th>
<th>( T )</th>
<th>( R )</th>
<th>( T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3</td>
<td>0%</td>
<td>5.3%</td>
<td>0%</td>
<td>7.5%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>0%</td>
<td>2.6%</td>
<td>0%</td>
<td>3.1%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td>0%</td>
<td>22.1%</td>
<td>14.1%</td>
<td>38.8%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>12.7%</td>
<td>7.8%</td>
<td>0.9%</td>
<td>5.4%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0%</td>
<td>17.7%</td>
<td>0%</td>
<td>17.7%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0%</td>
<td>16.4%</td>
<td>0%</td>
<td>7.6%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0%</td>
<td>3.9%</td>
<td>0%</td>
<td>2.4%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ L = 1.25\lambda_0 \]
Sawtooth Diffraction Grating

\[ \mu = 1.5 \quad \varepsilon = 3.0 \]

\[ L = 1.25 \lambda_0 \]

\[ h = 0.85 \lambda_0 \]

<table>
<thead>
<tr>
<th>( m )</th>
<th>( R )</th>
<th>( T )</th>
<th>( R )</th>
<th>( T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>-3</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>-2</td>
<td>0%</td>
<td>5.6%</td>
<td>0%</td>
<td>0.4%</td>
</tr>
<tr>
<td>-1</td>
<td>0%</td>
<td>39.2%</td>
<td>0%</td>
<td>53.1%</td>
</tr>
<tr>
<td>0</td>
<td>0%</td>
<td>0.8%</td>
<td>0%</td>
<td>3.4%</td>
</tr>
<tr>
<td>1</td>
<td>0%</td>
<td>44.2%</td>
<td>0%</td>
<td>29.5%</td>
</tr>
<tr>
<td>2</td>
<td>0%</td>
<td>10.1%</td>
<td>0%</td>
<td>13.6%</td>
</tr>
<tr>
<td>3</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>4</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>

Mode: E Mode

Mode: H Mode