Advanced Computation:
Computational Electromagnetics

Implementation of Finite-Difference Frequency-Domain

1

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

EMPossible Slide 2

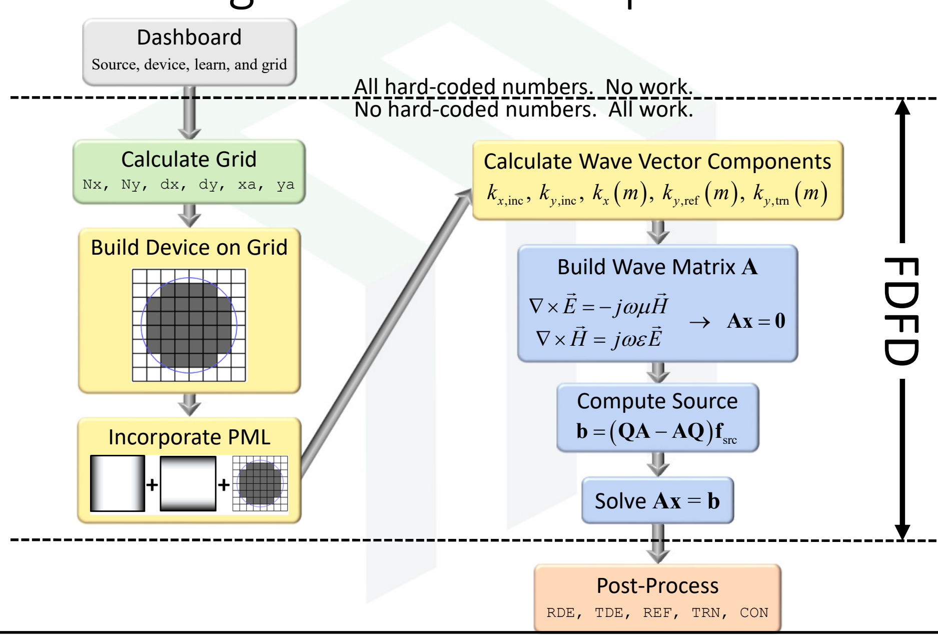
2

Basic Flow of FDFD

Slide 3

3

Block Diagram of FDFD Implementation



EMPossible

Slide 4

4

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 ε_r and μ_r
   c. Overlay onto 1× grids

3. Compute Wave Vector Components
   a. Identify materials in reflected and transmitted regions: ε_{ref}, ε_{trn}, μ_{ref}, μ_{trn}, n_{ref}, n_{trn}
   b. Compute incident wave vector: $\mathbf{k}_{inc}$
   c. Compute transverse wave vector expansion: $k_{x,m}$
   d. Compute $k_{y,\text{ref}}$ and $k_{y,\text{trn}}$

4. Construct $\mathbf{A}$
   a. Construct diagonal materials matrices
   b. Compute derivative matrices
   c. Compute $\mathbf{A}$

5. Compute Source Vector, $\mathbf{b}$
   a. compute source field $\mathbf{f}_{src}$
   b. compute $\mathbf{Q}$
   c. compute source vector $\mathbf{b}$

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

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

EMPossible

Slide 5

5

2× Grid Technique

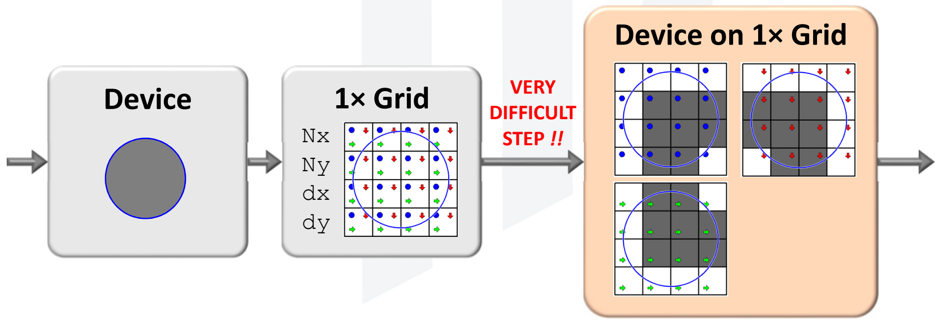
Slide 6

6

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

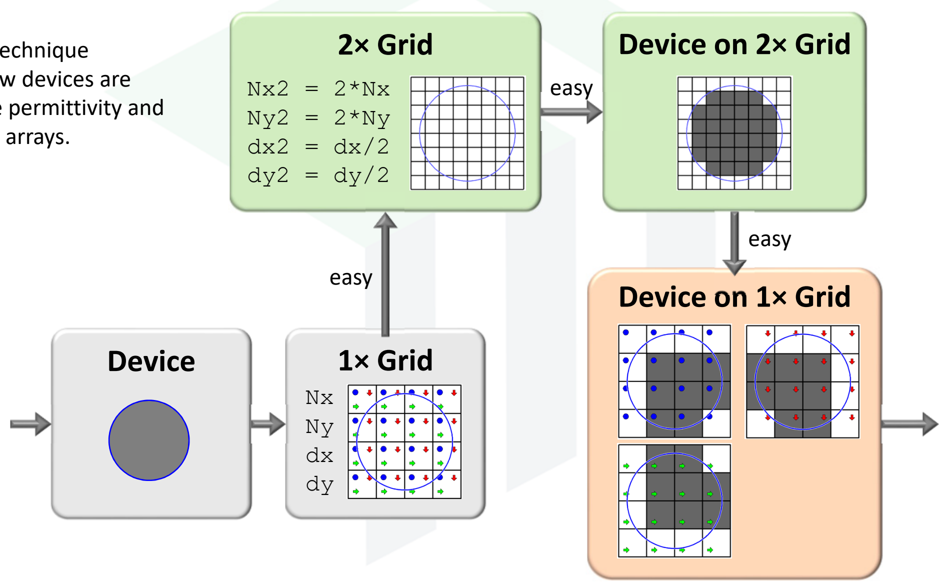
This is the traditional approach for building devices on a Yee grid.

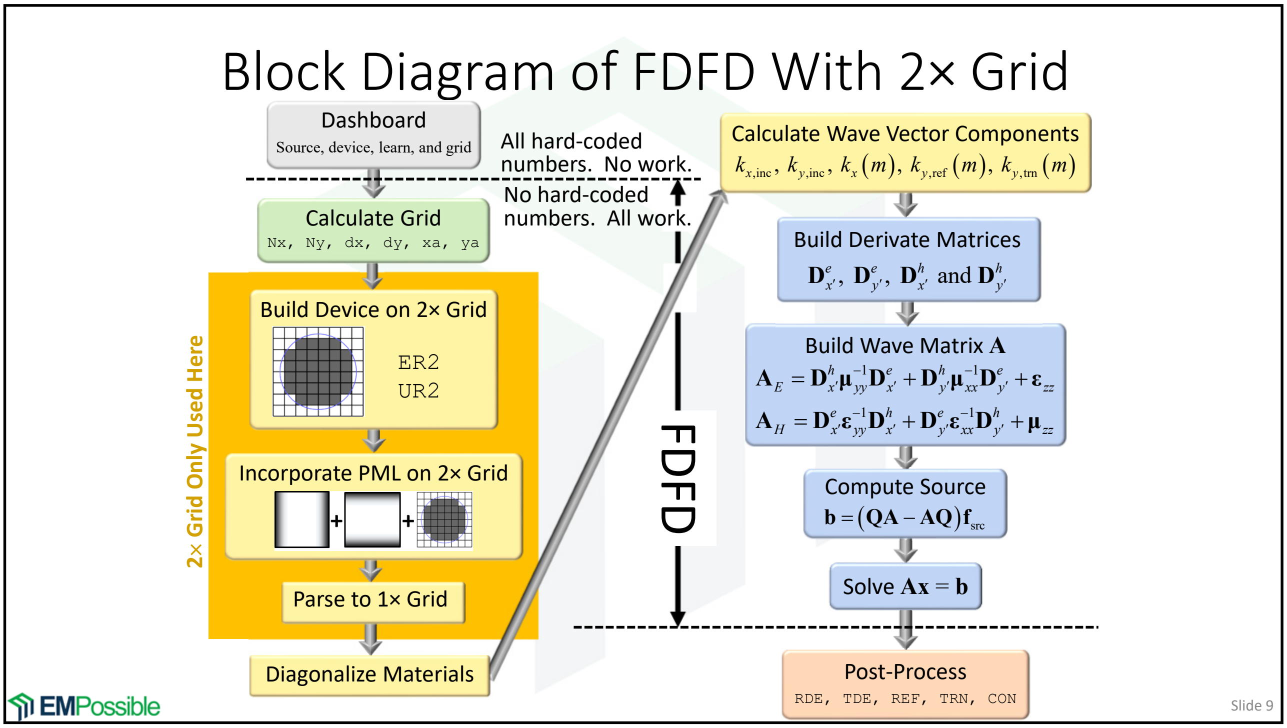
It is very tedious and cumbersome to determine which field components reside in which material.



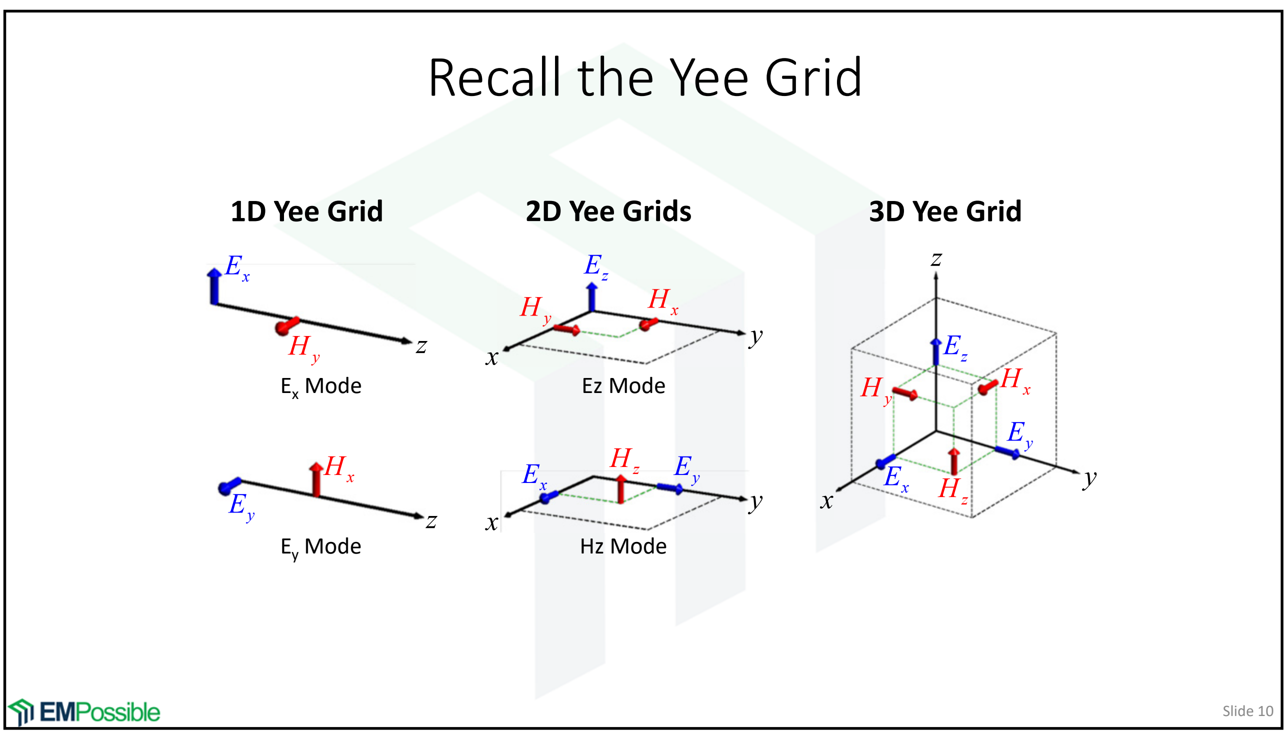
What is the 2× Grid Technique? (2 of 2)

The 2× grid technique simplifies how devices are built into the permittivity and permeability arrays.



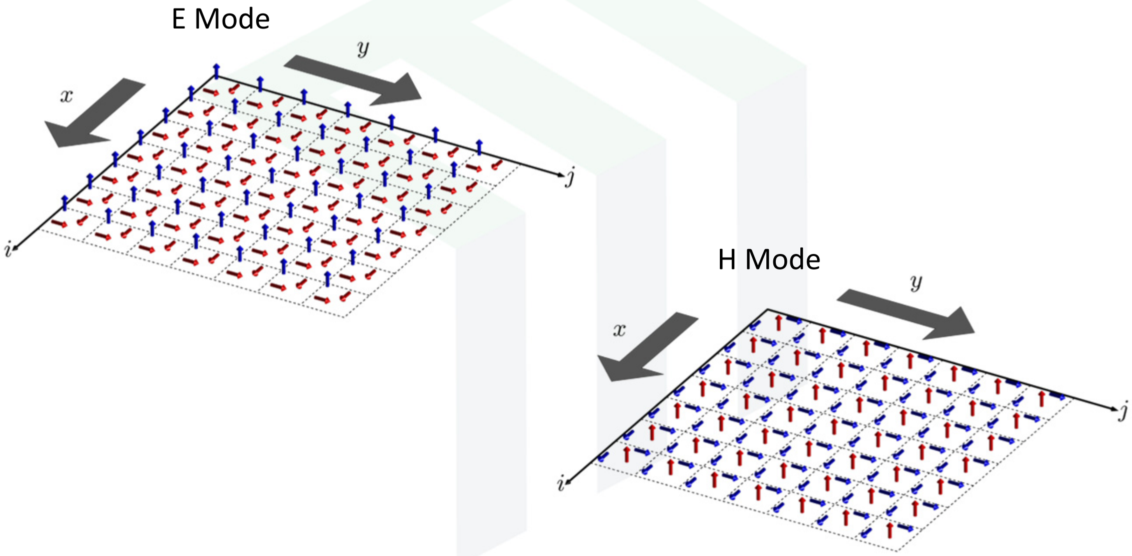

Block Diagram of FDFD With 2× Grid
Dashboard
Source, device, learn, and grid
All hard-coded numbers. No work.
No hard-coded numbers. All work.
Calculate Grid
Nx, Ny, dx, dy, xa, ya
2× Grid Only Used Here
Build Device on 2× Grid
ER2
UR2
Incorporate PML on 2× Grid
Parse to 1× Grid
Diagonalize Materials
FDFD
Calculate Wave Vector Components
$k_{x,\text{inc}},\ k_{y,\text{inc}},\ k_x(m),\ k_{y,\text{ref}}(m),\ k_{y,\text{trn}}(m)$
Build Derivate Matrices
$\mathbf{D}_{x'}^{e},\ \mathbf{D}_{y'}^{e},\ \mathbf{D}_{x'}^{h}$ and $\mathbf{D}_{y'}^{h}$
Build Wave Matrix A
$\mathbf{A}_E = \mathbf{D}_{x'}^{h}\boldsymbol{\mu}_{yy}^{-1}\mathbf{D}_{x'}^{e} + \mathbf{D}_{y'}^{h}\boldsymbol{\mu}_{xx}^{-1}\mathbf{D}_{y'}^{e} + \boldsymbol{\varepsilon}_{zz}$
$\mathbf{A}_H = \mathbf{D}_{x'}^{e}\boldsymbol{\varepsilon}_{yy}^{-1}\mathbf{D}_{x'}^{h} + \mathbf{D}_{y'}^{e}\boldsymbol{\varepsilon}_{xx}^{-1}\mathbf{D}_{y'}^{h} + \boldsymbol{\mu}_{zz}$
Compute Source
$\mathbf{b} = (\mathbf{QA} - \mathbf{AQ})\mathbf{f}_{\text{src}}$
Solve $\mathbf{Ax} = \mathbf{b}$
Post-Process
RDE, TDE, REF, TRN, CON
EMPossible
Slide 9

9

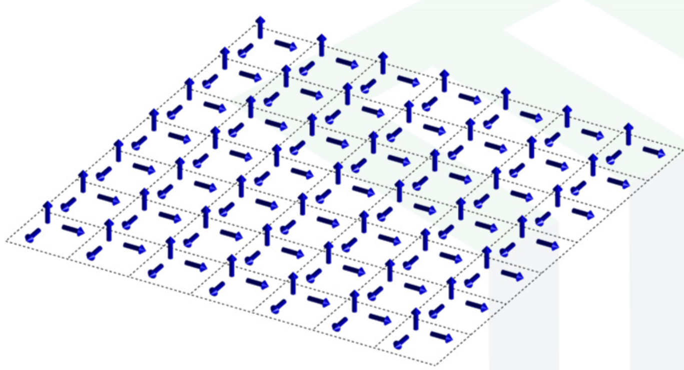

Recall the Yee Grid
1D Yee Grid
E_x
H_y
z
Ex Mode
H_x
E_y
z
Ey Mode
2D Yee Grids
E_z
H_y
H_x
x
y
Ez Mode
E_x
H_z
E_y
x
y
Hz Mode
3D Yee Grid
z
E_z
H_y
H_x
E_y
E_x
H_z
x
y
EMPossible
Slide 10

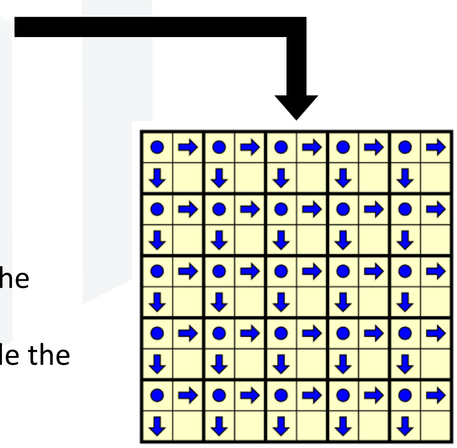
10

4×4 Yee Grids



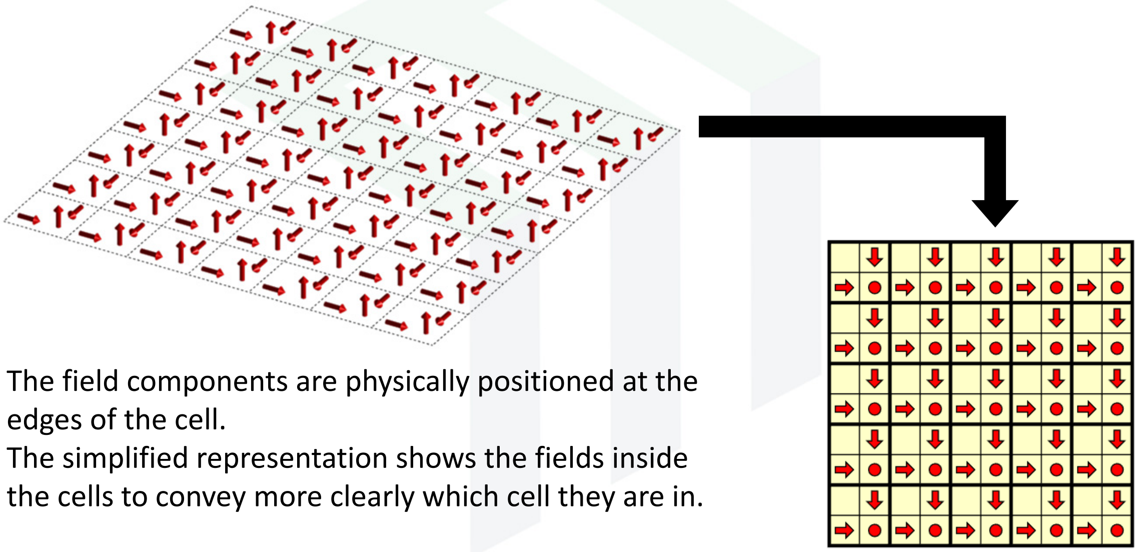
Simplified Representation of E-Field Components





- 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.

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.

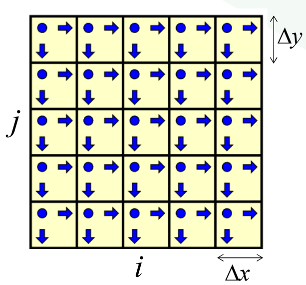
EMPossible

Slide 13

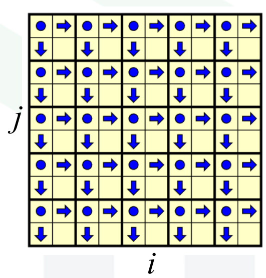
13

The 2× Grid

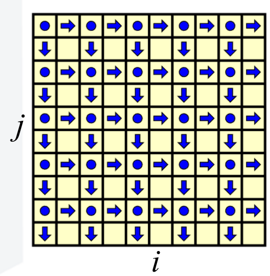
The Conventional 1× Grid



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.

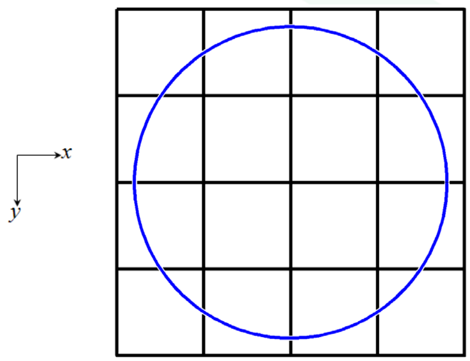
EMPossible

Slide 14

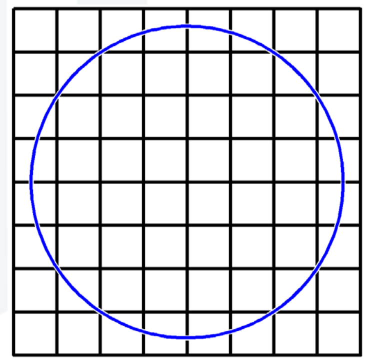
14

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 2× grid.



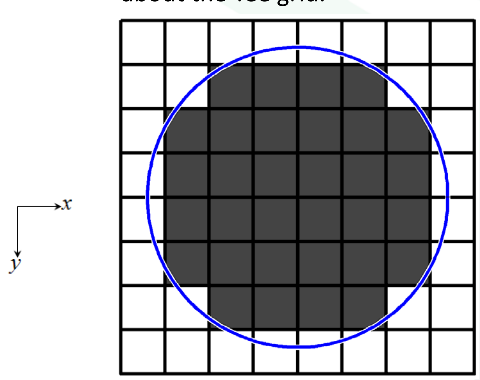
Note: the 2× grid represents the same physical space, but with twice the number of points.

Slide 15

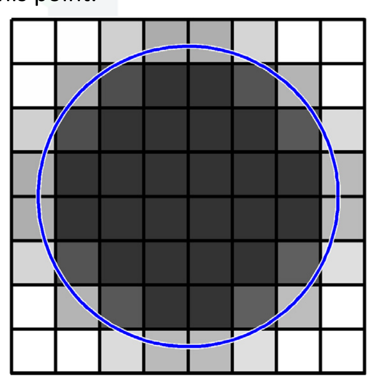
15

2×→1× (2 of 4): *Build Device*

Third, we construct a cylinder on the 2× grid without having to consider anything about the Yee grid.



Fourth, if desired we could perform dielectric averaging on the 2× grid at this point.



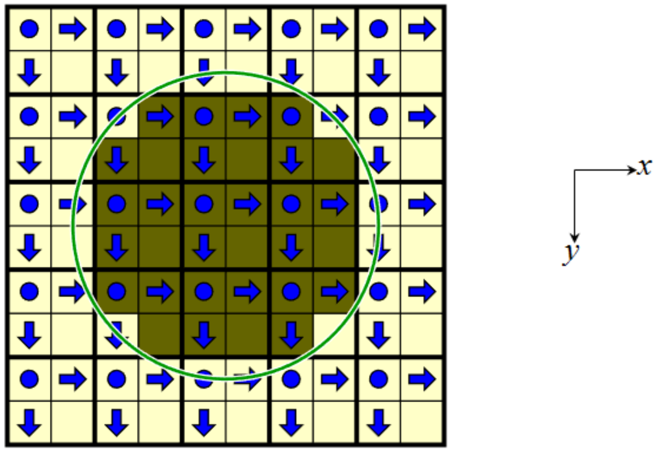
Note: This is discussed more in Lecture 14 FDFD Extras.

Slide 16

16

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

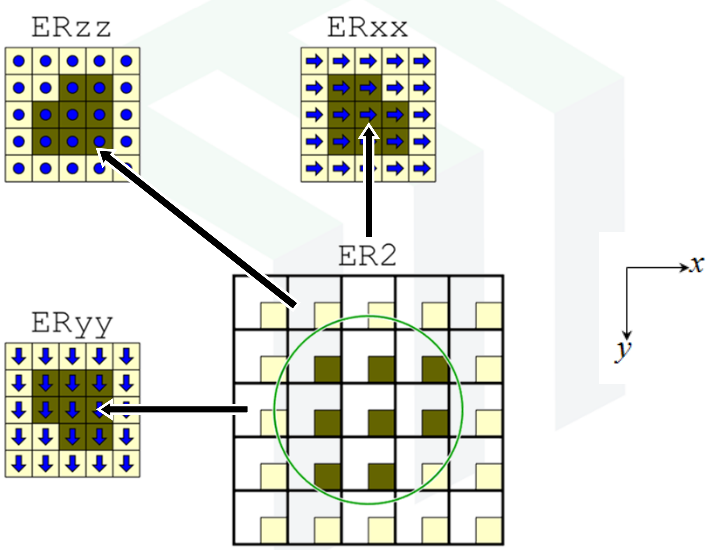
Recall the relation between the 2× and 1× grids as well as the location of the field components.



EMPossible

17

2×→1× (4 of 4): *Parse to 1× Grid*

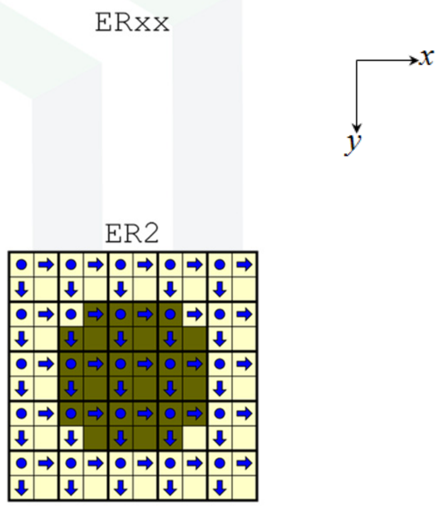


EMPossible

18

Extract ERxx from ER2

```
ERxx = ER2(2:2:Nx2,1:2:Ny2)
```



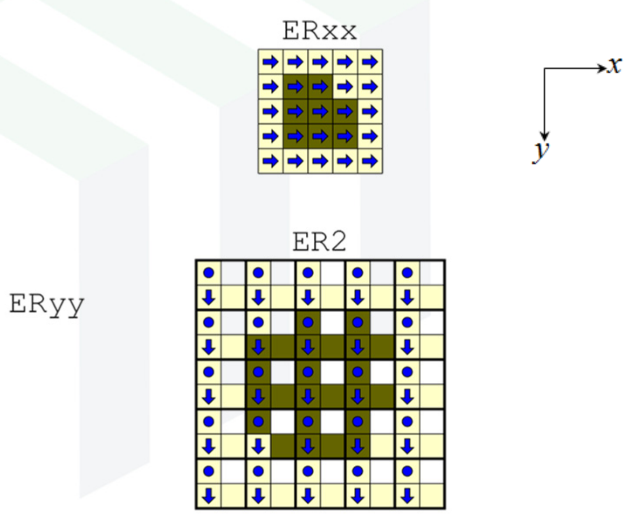
EMPossible

Slide 19

19

Extract ERyy from ER2

```
ERyy = ER2(1:2:Nx2,2:2:Ny2)
```



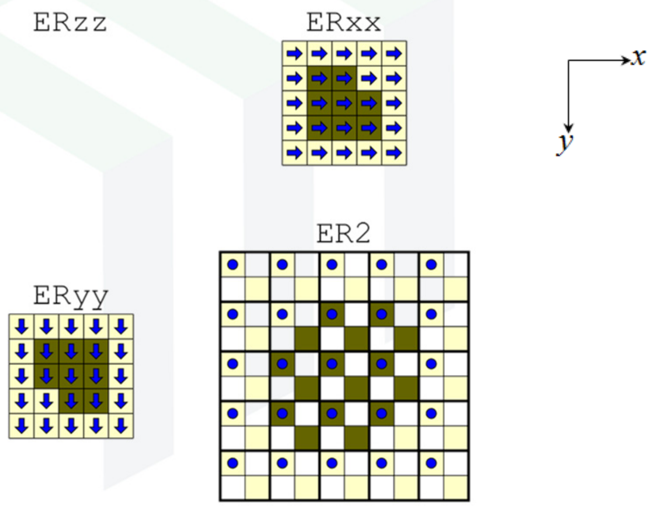
EMPossible

Slide 20

20

Extract `ERzz` from `ER2`

```
ERzz = ER2(1:2:Nx2,1:2:Ny2)
```



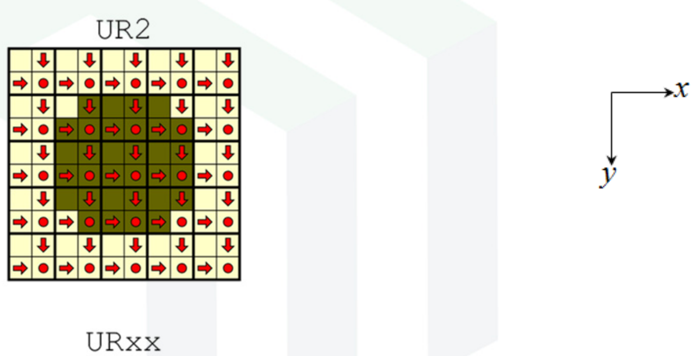
EMPossible

Slide 21

21

Extract `URxx` from `UR2`

```
URxx = UR2(1:2:Nx2,2:2:Ny2)
```



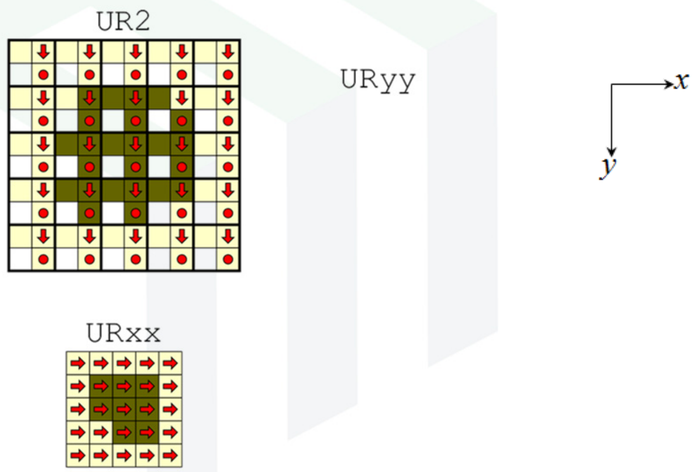
EMPossible

Slide 22

22

Extract URyy from UR2

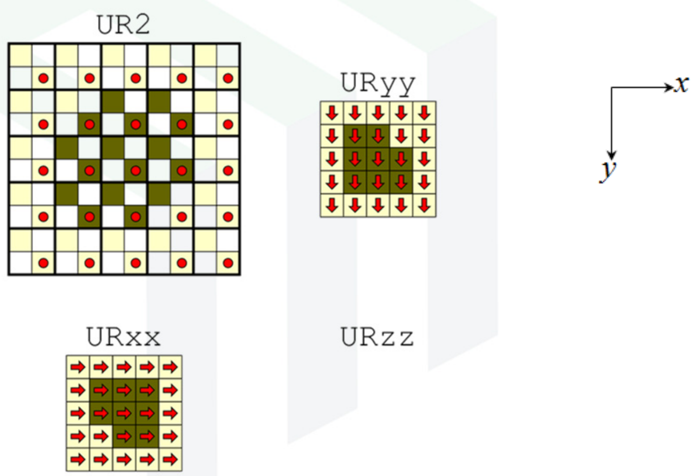
```
URyy = UR2(2:2:Nx2,1:2:Ny2)
```



23

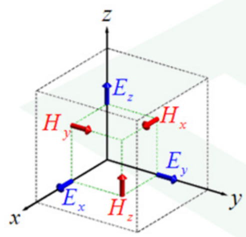
Extract URzz from UR2

```
URzz = UR2(2:2:Nx2,2:2:Ny2)
```

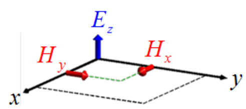


24

MATLAB Code for Parsing Onto 1× Grid

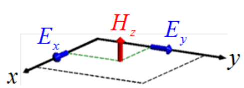



```
ERxx = ER2(2:2:Nx2,1:2:Ny2,1:2:Nz2);
ERyy = ER2(1:2:Nx2,2:2:Ny2,1:2:Nz2);
ERzz = ER2(1:2:Nx2,1:2:Ny2,2:2:Nz2);

URxx = UR2(1:2:Nx2,2:2:Ny2,2:2:Nz2);
URyy = UR2(2:2:Nx2,1:2:Ny2,2:2:Nz2);
URzz = UR2(2:2:Nx2,2:2:Ny2,1:2:Nz2);
```



E_z Mode

```
ERzz = ER2(1:2:Nx2,1:2:Ny2);

URxx = UR2(1:2:Nx2,2:2:Ny2);
URyy = UR2(2:2:Nx2,1:2:Ny2);
```



H_z Mode

```
ERxx = ER2(2:2:Nx2,1:2:Ny2);
ERyy = ER2(1:2:Nx2,2:2:Ny2);

URzz = UR2(2:2:Nx2,2:2:Ny2);
```

EMPossible

Slide 25

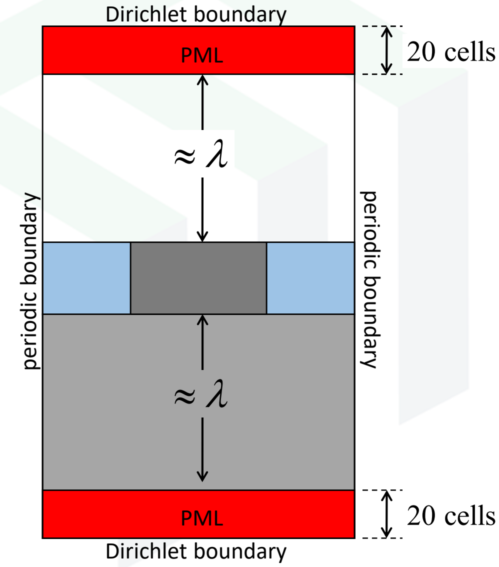
25

Calculating the Grid Parameters

Slide 26

26

Grid Strategy



Consideration #1: Wavelength λ_0

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

First, determine the smallest wavelength:

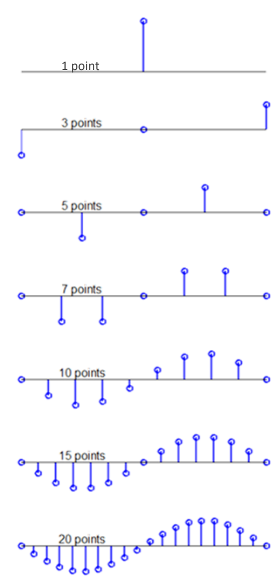
$$\lambda_{\min} = \frac{\min\left[\lambda_0\right]}{\max\left[n(x,y)\right]}$$

$\min[\lambda_0]$ 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_λ cells.

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

N_λ	Comments
10 to 20	Low contrast dielectrics
20 to 30	High contrast dielectrics
40 to 60	Most metallic structures
100 to 200	Plasmonic devices



Consideration #2: Mechanical Features

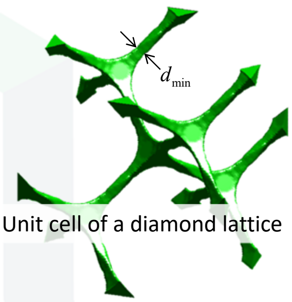
The grid resolution must be sufficient to resolve the smallest mechanical features of the device.

First, determine the smallest feature:

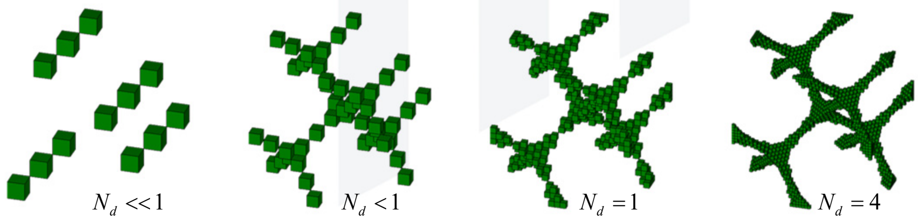
$$d_{\min}$$

Second, resolve $d_{\min}$ by at least N_d cells (usually 1 to 4)

$$\Delta_d \approx \frac{d_{\min}}{N_d} \qquad N_d \geq 1$$



Unit cell of a diamond lattice



Calculating the Initial Grid Resolution

1. Must resolve the minimum wavelength.

$$\Delta_\lambda \leq \frac{\min(\lambda_0) \div \max\left[n(x,y)\right]}{N_\lambda} \qquad N_\lambda \geq 10$$

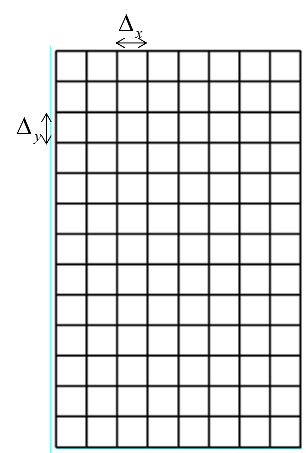
Note: If you are performing a parameter sweep over frequency or wavelength, $\min(\lambda_0)$ is the shortest wavelength in the sweep.

2. Must resolve the minimum structural dimension.

$$\Delta_d \leq \frac{\ell_{\min}}{N_d} \qquad N_d \geq 1$$

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

$$\Delta_x = \Delta_y = \min\left[\Delta_\lambda, \Delta_d\right]$$

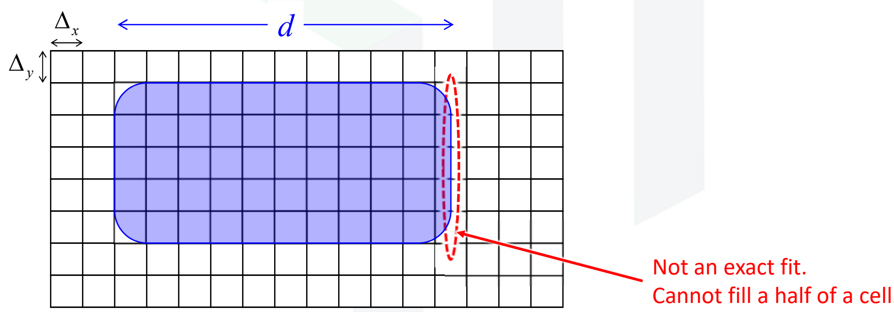


Resolving Critical Dimensions (1 of 3)

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.

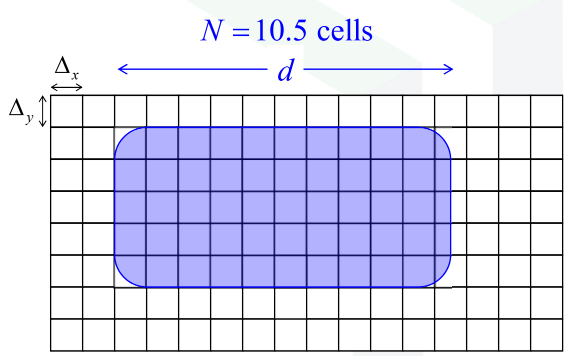
Suppose we wish to place a device of length d onto a grid.



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.

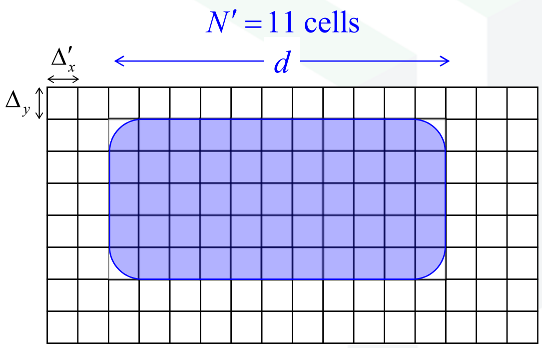
$$N' = \text{ceil}\left[\frac{d}{\Delta_x}\right]$$

$N' = 11$ cells

Resolving Critical Dimensions (3 of 3)

Third, we adjust the value of Δ_x to represent the dimension d exactly.

$$\Delta'_x = \frac{d}{N'}$$



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.

$$d_x \quad \text{and} \quad d_y$$

→ Typically this is a lattice constant or grating period along x
→ Typically this is a film thickness or grating depth along y

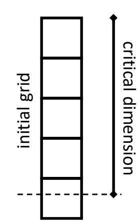
Compute how many grid cells comprise d_x and d_y and round UP.

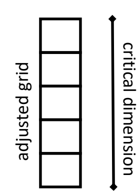
$$M_x = \text{ceil}\left(d_x / \Delta_x\right)$$
$$M_y = \text{ceil}\left(d_y / \Delta_y\right)$$

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

$$\Delta'_x = d_x / M_x$$
$$\Delta'_y = d_y / M_y$$

For simplicity, drop the prime symbol ' from Δ_x and Δ_y.





Compute Total Grid Size

- Don't forget to add cells for PML!
- Must often add "space" between PML and device.

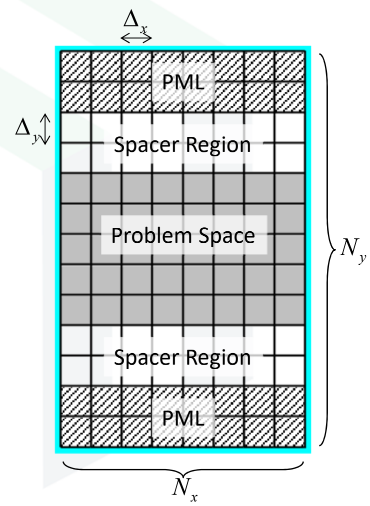
Note: This is particularly important when modeling devices with large evanescent fields.

$$N_x = \frac{\Lambda_x}{\Delta_x}$$

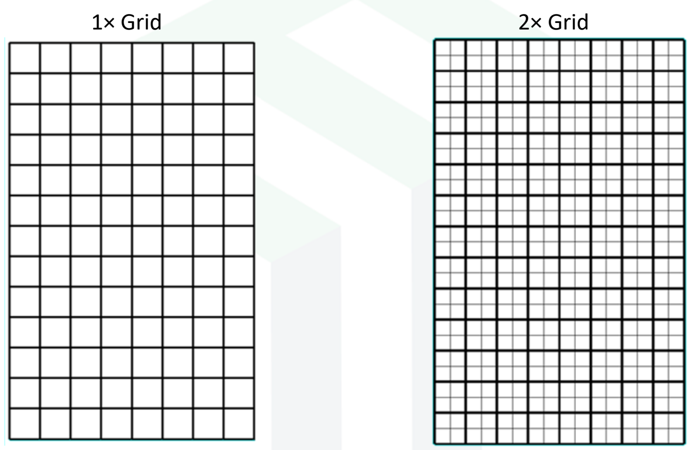
Easiest to make N_x odd. Reason discussed later.

$$N_y = \frac{\Lambda_y}{\Delta_y} + 2N_{\text{PML}} + 2N_{\text{SPACE}}$$

$$N_{\text{SPACE}} \cong \text{ceil}\left(\frac{\lambda_{\max}}{n_{\text{buff}}\Delta_y}\right)$$



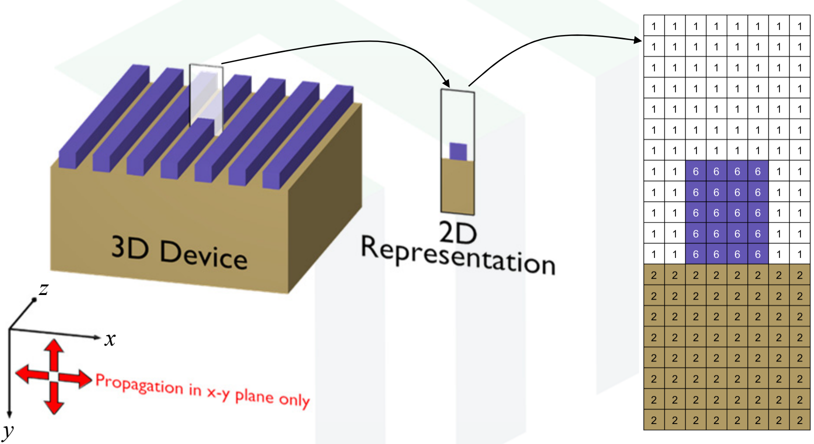
Compute 2× Grid Parameters




```
Nx2 = 2*Nx;   dx2 = dx/2;
Ny2 = 2*Ny;   dy2 = dy/2;
```

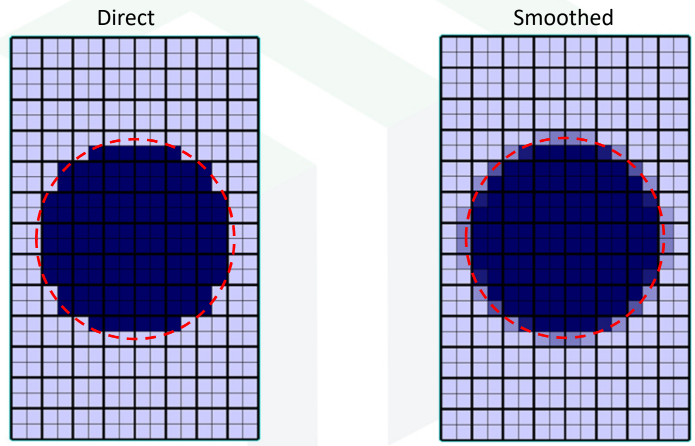
Constructing a Device on the Grid

Reducing 3D Problems to 2D



Representation on a Cartesian grid

Averaging At the Edges



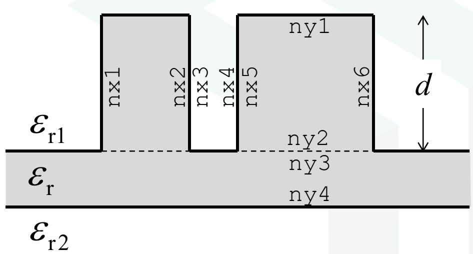
EMPossible

Slide 39

39

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 error into your results.

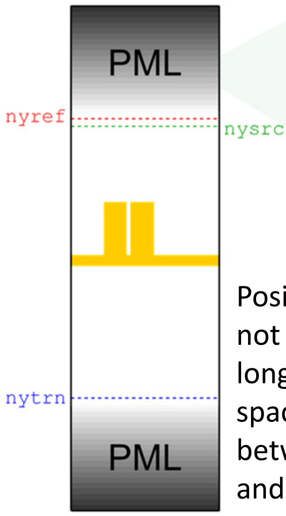
```
% BUILD DEVICE
ER2 = er1*ones(Nx2,Ny2);          %fill everywhere with er1
ER2(nx1:nx2,ny1:ny2) = er;        %add tooth 1
ER2(nx5:nx6,ny1:ny2) = er;        %add tooth 2
ER2(:,ny3:ny4)       = er;        %add substrate
ER2(:,ny4+1:Ny2)     = er2;       %fill transmission region
```

EMPossible

Slide 40

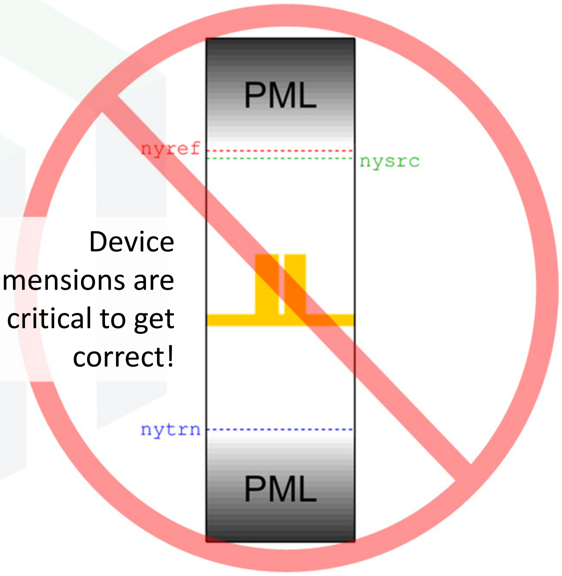
40

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!



EMPossible

Slide 41

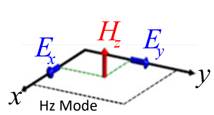
41

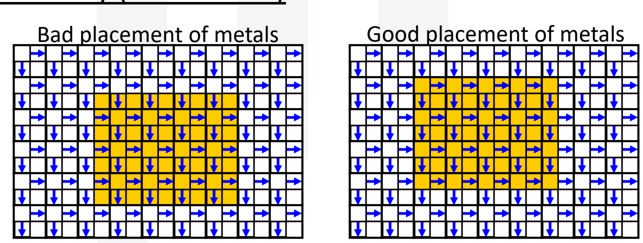
Oh Yeah, Metals!

Perfect Electric Conductors

$$\varepsilon_{\mathrm{r}} = -10000 \qquad \text{or} \qquad \begin{bmatrix} \ddots & & & & \\ 0 & 0 & 1 & 0 & 0 \\ & & & \ddots & \end{bmatrix} \begin{bmatrix} E_1 \\ \vdots \\ E_m \\ \vdots \\ E_M \end{bmatrix} = \begin{bmatrix} f_1 \\ \\ 0 \\ \\ f_M \end{bmatrix} \longrightarrow E_m = 0$$

Include Tangential Fields at Boundary (TM modes!)





EMPossible

Slide 42

42

Walkthrough of the FDFD Algorithm

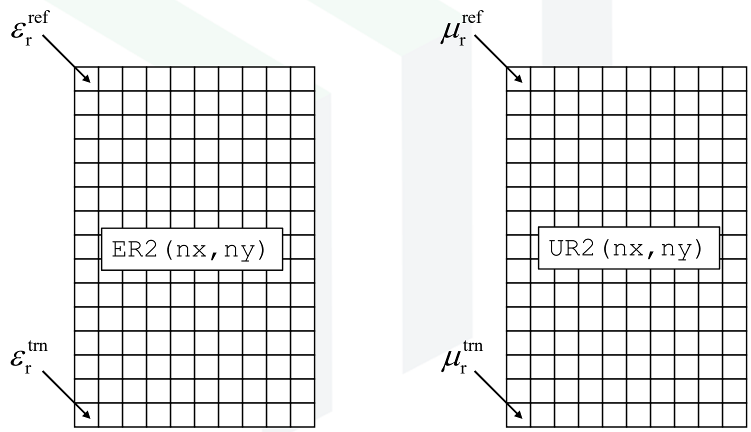
Input to the FDFD Algorithm

The FDFD algorithm requires the following information:

- The materials on the 2× grid: `ER2(nx,ny)` and `UR2(nx,ny)`
- The grid resolution: `dx` and `dy`
- The size of the PML on the 1× grid: `NYLO` and `NYHI`
- The source wavelength, λ_0
- Angle of incidence, θ
- 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 of the grid.

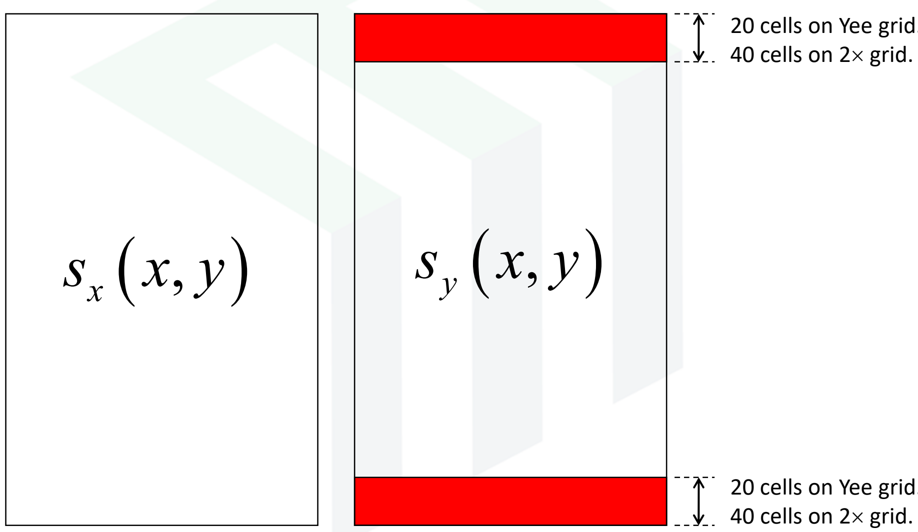


EMPossible

Slide 45

45

(2) Compute the PML Parameters on 2× Grid



EMPossible

Slide 46

46

(3) Incorporate the PML

We can incorporate the PML parameters into [μ] and [ε] as follows

$$\nabla \times \vec{E} = k_0 [\mu'_r] \vec{\tilde{H}}$$
$$\nabla \times \vec{\tilde{H}} = k_0 [\varepsilon'_r] \vec{E}$$

$$[\varepsilon'_r] = \begin{bmatrix} \frac{s_y s_z}{s_x}\varepsilon_{xx} & 0 & 0 \\ 0 & \frac{s_x s_z}{s_y}\varepsilon_{yy} & 0 \\ 0 & 0 & \frac{s_x s_y}{s_z}\varepsilon_{zz} \end{bmatrix} \qquad [\mu'_r] = \begin{bmatrix} \frac{s_y s_z}{s_x}\mu_{xx} & 0 & 0 \\ 0 & \frac{s_x s_z}{s_y}\mu_{yy} & 0 \\ 0 & 0 & \frac{s_x s_y}{s_z}\mu_{zz} \end{bmatrix}$$

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

$$\varepsilon'_{xx} = \frac{s_y}{s_x}\varepsilon_r \qquad \mu'_{xx} = \frac{s_y}{s_x}\mu_r$$
$$\varepsilon'_{yy} = \frac{s_x}{s_y}\varepsilon_r \qquad \mu'_{yy} = \frac{s_x}{s_y}\mu_r$$
$$\varepsilon'_{zz} = s_x s_y \varepsilon_r \qquad \mu'_{zz} = s_x s_y \mu_r$$

Note: the PML is incorporated into the 2× 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;
```

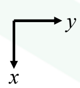
EMPossible Slide 47

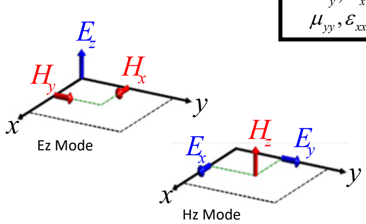
47

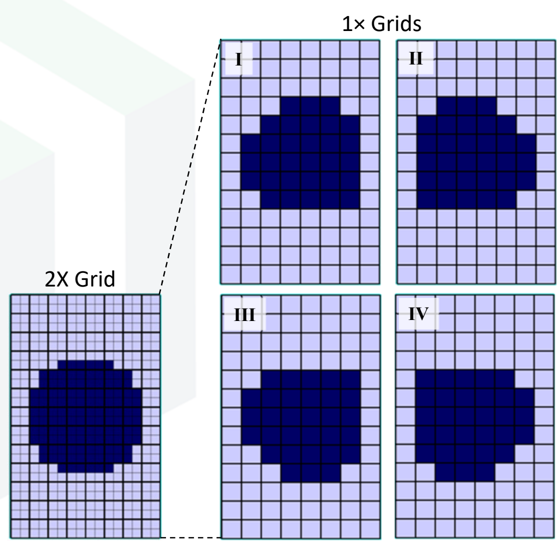
(4) Overlay Materials Onto 1× Grids

Field and materials assignments

I	II
E_z, ε_{zz}	H_x, E_y, $\mu_{xx}, \varepsilon_{yy}$
III	**IV**
H_y, E_x, $\mu_{yy}, \varepsilon_{xx}$	H_z, μ_{zz}








```
% OVERLAY MATERIALS ONTO 1X GRID
URxx = URxx(1:2:Nx2,2:2:Ny2);
URyy = URyy(2:2:Nx2,1:2:Ny2);
URzz = URzz(2:2:Nx2,2:2:Ny2);
ERxx = ERxx(2:2:Nx2,1:2:Ny2);
ERyy = ERyy(1:2:Nx2,2:2:Ny2);
ERzz = ERzz(1:2:Nx2,1:2:Ny2);
```

EMPossible Slide 48

48

(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

$$k_x(m) = k_{x,\text{inc}} - m\frac{2\pi}{\Lambda_x} \qquad m = \cdots, -2, -1, 0, 1, 2, \cdots$$

N_x points total.
For proper symmetry, N_x should be odd.

```
m = [-floor(Nx/2):floor(Nx/2)]';
```

$$k_{y,\text{ref}}(m) = \sqrt{(k_0 n_{\text{ref}})^2 - k_x^2(m)}$$

$$k_{y,\text{trn}}(m) = \sqrt{(k_0 n_{\text{trn}})^2 - k_x^2(m)}$$

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.

EMPossible Slide 49

49

(6) Construct Diagonal Materials Matrices

$$\boldsymbol{\varepsilon}_{xx} = \begin{bmatrix} \varepsilon_{xx}(1) & & & 0 \\ & \varepsilon_{xx}(2) & & \\ & & \ddots & \\ 0 & & & \varepsilon_{xx}(N) \end{bmatrix} \qquad \boldsymbol{\mu}_{xx} = \begin{bmatrix} \mu_{xx}(1) & & & 0 \\ & \mu_{xx}(2) & & \\ & & \ddots & \\ 0 & & & \mu_{xx}(N) \end{bmatrix}$$

$$\boldsymbol{\varepsilon}_{yy} = \begin{bmatrix} \varepsilon_{yy}(1) & & & 0 \\ & \varepsilon_{yy}(2) & & \\ & & \ddots & \\ 0 & & & \varepsilon_{yy}(N) \end{bmatrix} \qquad \boldsymbol{\mu}_{yy} = \begin{bmatrix} \mu_{yy}(1) & & & 0 \\ & \mu_{yy}(2) & & \\ & & \ddots & \\ 0 & & & \mu_{yy}(N) \end{bmatrix}$$

$$\boldsymbol{\varepsilon}_{zz} = \begin{bmatrix} \varepsilon_{zz}(1) & & & 0 \\ & \varepsilon_{zz}(2) & & \\ & & \ddots & \\ 0 & & & \varepsilon_{zz}(N) \end{bmatrix} \qquad \boldsymbol{\mu}_{zz} = \begin{bmatrix} \mu_{zz}(1) & & & 0 \\ & \mu_{zz}(2) & & \\ & & \ddots & \\ 0 & & & \mu_{zz}(N) \end{bmatrix}$$

```
ERxx = diag(sparse(ERxx(:)));
```

EMPossible Slide 50

50

(7) Construct the Derivative Matrices

$$\mathbf{D}_{x'}^{\mathrm{e}},\ \mathbf{D}_{y'}^{\mathrm{e}},\ \mathbf{D}_{x'}^{\mathrm{h}},\ \text{and}\ \mathbf{D}_{y'}^{\mathrm{h}}$$

```
[DEX,DEY,DHX,DHY] = yeeder(NGRID,k0*RES,BC,kinc/k0);
```

Frequency (or wavelength) information is incorporated into FDFD here

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

`A = sparse(M,M)` – creates a sparse M×M matrix of zeros.
`A = spdiags(b,d,A)` – Adds array `b` to diagonal `d` in matrix `A`.

EMPossible Slide 51

51

(8) Compute the Wave Matrix A

E Mode

$$\mathbf{A}_{\mathrm{E}} = \mathbf{D}_{x'}^{\mathrm{h}}\boldsymbol{\mu}_{yy}^{-1}\mathbf{D}_{x'}^{\mathrm{e}} + \mathbf{D}_{y'}^{\mathrm{h}}\boldsymbol{\mu}_{xx}^{-1}\mathbf{D}_{y'}^{\mathrm{e}} + \boldsymbol{\varepsilon}_{zz}$$

H Mode

$$\mathbf{A}_{\mathrm{H}} = \mathbf{D}_{x'}^{\mathrm{e}}\boldsymbol{\varepsilon}_{yy}^{-1}\mathbf{D}_{x'}^{\mathrm{h}} + \mathbf{D}_{y'}^{\mathrm{e}}\boldsymbol{\varepsilon}_{xx}^{-1}\mathbf{D}_{y'}^{\mathrm{h}} + \boldsymbol{\mu}_{zz}$$

EMPossible Slide 52

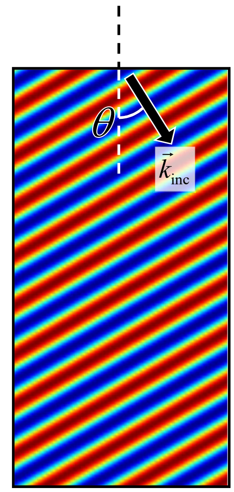
52

(9) Compute the Source Field

The source has an amplitude of 1.0

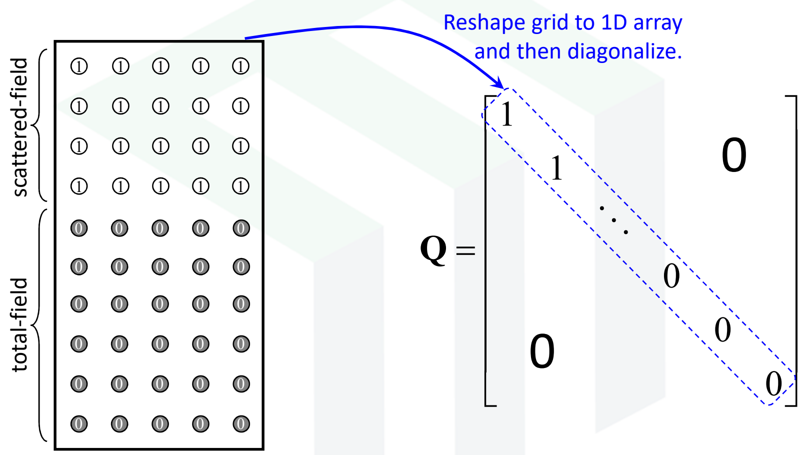
$$f_{\text{src}}(x,y) = \exp\left(j\vec{k}_{\text{inc}} \bullet \vec{r}\right)$$
$$= \exp\left[j\left(k_{x,\text{inc}}x + k_{y,\text{inc}}y\right)\right]$$

Don't forget to make $\mathbf{f}_{\text{src}}$ a column vector.



1× grid

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




```
Q = diag(sparse(Q(:)));
```

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

$$\mathbf{Af} = \mathbf{b} \qquad \mathbf{b} = (\mathbf{QA} - \mathbf{AQ})\mathbf{f}_{\text{src}}$$

EMPossible

Slide 55

55

(12) Compute the Field f

$$\mathbf{f} = \mathbf{A}^{-1}\mathbf{b}$$

Aside

In MATLAB, `f = A\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.

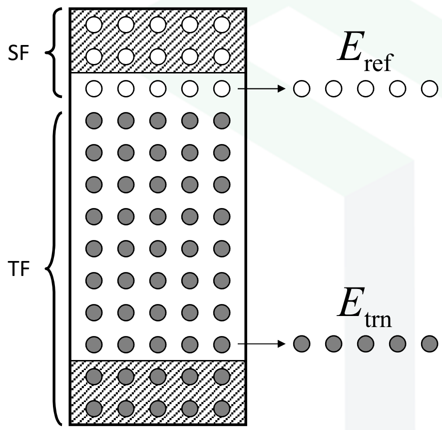
Don't forget to `reshape()` **f** from a column vector to a 2D grid after the calculation.

EMPossible

Slide 56

56

(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.

EMPossible

Slide 57

57

(14) Remove the Phase Tilt

Recall Bloch's theorem,

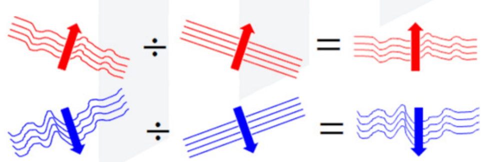
$$\vec{E}(\vec{r}) = \vec{A}_{\vec{k}_{\text{inc}}}(\vec{r}) \cdot e^{j\vec{k}_{\text{inc}} \bullet \vec{r}}$$

This implies the transmitted and reflected fields have the following form

$$E_{\text{ref}}(x) = A_{\text{ref}}(x) e^{jk_{x,\text{inc}}x}$$
$$E_{\text{trn}}(x) = A_{\text{trn}}(x) e^{jk_{x,\text{inc}}x}$$

We remove the phase tilt to calculate the amplitude terms.

$$A_{\text{ref}}(x) = E_{\text{ref}}(x) e^{-jk_{x,\text{inc}}x}$$
$$A_{\text{trn}}(x) = E_{\text{trn}}(x) e^{-jk_{x,\text{inc}}x}$$



EMPossible

Slide 58

58

(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.

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

$$S_{\text{trn}}(m) = \text{FFT}\left\{A_{\text{trn}}(x)\right\}$$

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

```
Sref = flipud(fftshift(fft(Aref)))/Nx;
Strn = flipud(fftshift(fft(Atrn)))/Nx;
```

EMPossible Slide 59

59

(16) Calculate Diffraction Efficiencies

The source wave was given unit amplitude so

$$\left|S_{\text{inc}}\right|^2 = 1$$

The diffraction efficiencies of the reflected modes are then

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

$$R(m) = \left|\vec{U}_{\text{ref}}(m)\right|^2 \frac{\text{Re}\left[k_y^{\text{ref}}(m)/\varepsilon_{\text{r,inc}}\right]}{\text{Re}\left[k_y^{\text{inc}}/\varepsilon_{\text{r,inc}}\right]} \quad \text{H Mode}$$

Recall that there used to be a negative sign here. We dropped it because we also dropped the sign when calculating $k_{y,\text{ref}}(m)$.

The diffraction efficiencies of the transmitted modes are then

$$T(m) = \left|S_{\text{trn}}(m)\right|^2 \frac{\text{Re}\left[k_y^{\text{trn}}/\mu_{\text{r,trn}}\right]}{\text{Re}\left[k_y^{\text{inc}}/\mu_{\text{r,inc}}\right]}$$

(E Mode)

$$T(m) = \left|U_{\text{trn}}(m)\right|^2 \frac{\text{Re}\left[k_y^{\text{trn}}/\varepsilon_{\text{r,trn}}\right]}{\text{Re}\left[k_y^{\text{inc}}/\varepsilon_{\text{r,inc}}\right]}$$

(H Mode)

$S(m) \equiv$ amplitudes of E mode spatial harmonics

$U(m) \equiv$ amplitudes of H mode spatial harmonics

Note: these equations assume that $\left|\vec{S}_{\text{inc}}\right| = 1$.

EMPossible Slide 60

60

(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 ABS = 0 and we will have

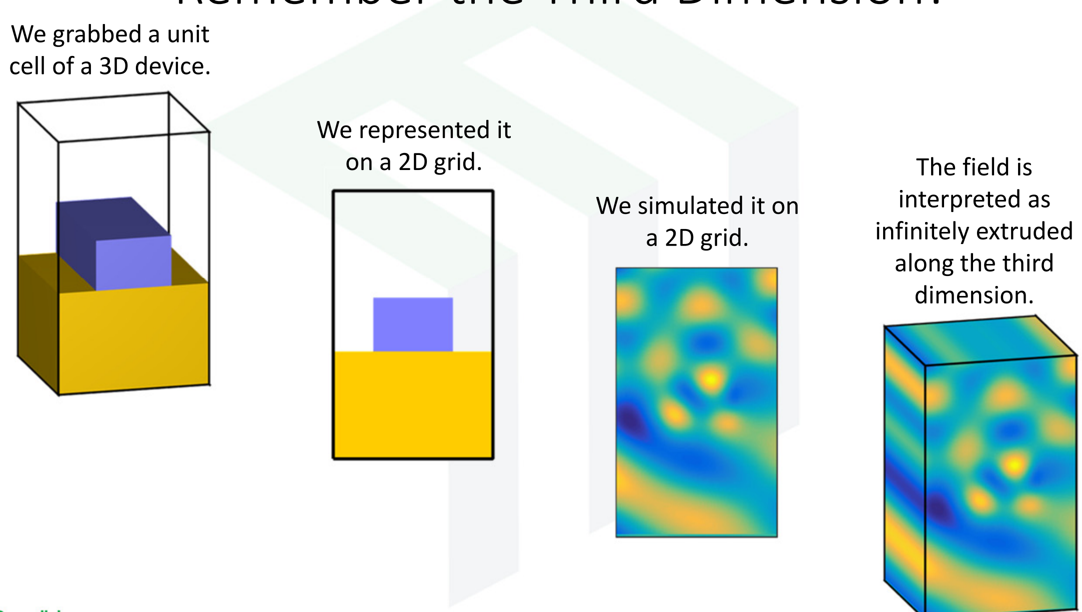
$$\text{REF} + \text{TRN} = 100\%$$

$$\begin{aligned} &\text{REF} + \text{TRN} < 100\% && \text{loss} \\ &\text{REF} + \text{TRN} = 100\% && \text{no loss or gain} \\ &\text{REF} + \text{TRN} > 100\% && \text{gain} \end{aligned}$$

EMPossible Slide 61

61

Remember the Third Dimension!



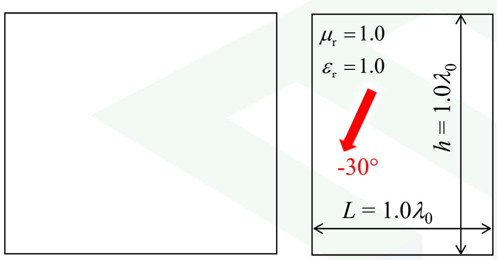
EMPossible Slide 62

62

Examples for Benchmarking

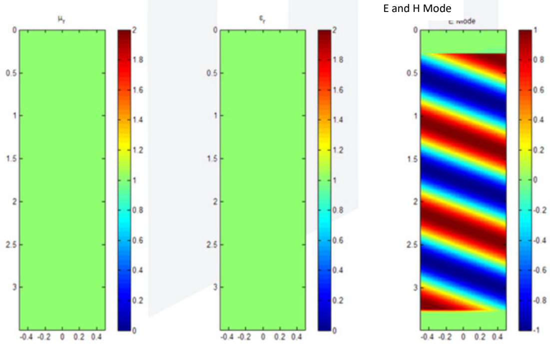
63

Air Simulation



	E Mode		*H* Mode	
m	*R*	*T*	*R*	*T*
-4	0%	0%	0%	0%
-3	0%	0%	0%	0%
-2	0%	0%	0%	0%
-1	0%	0%	0%	0%
0	0%	100%	0%	100%
1	0%	0%	0%	0%
2	0%	0%	0%	0%
3	0%	0%	0%	0%
4	0%	0%	0%	0%

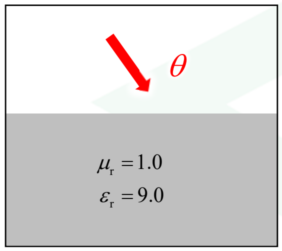
E and H Mode



EMPossible

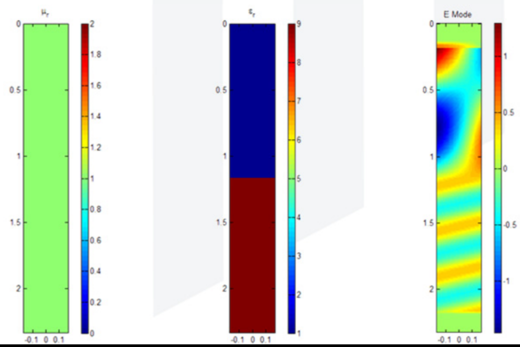
64

Dielectric Slab Grating



	E Mode		H Mode	
θ	R	T	R	T
-60°	49.5%	50.6%	5.0%	95.4%
-25°	28.5%	71.7%	21.8%	78.4%
0°	25.1%	75.0%	25.1%	75.1%
15°	26.3%	73.8%	24.0%	76.3%
45°	37.2%	62.9%	13.9%	86.4%
80°	77.7%	21.8%	8.3%	91.9%

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

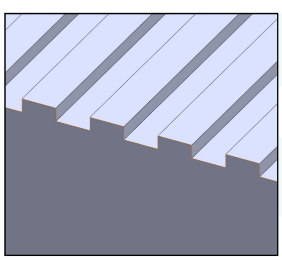


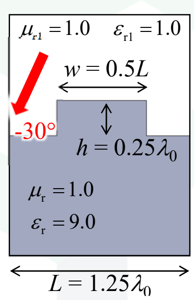
$\theta = 45°$

Slide 65

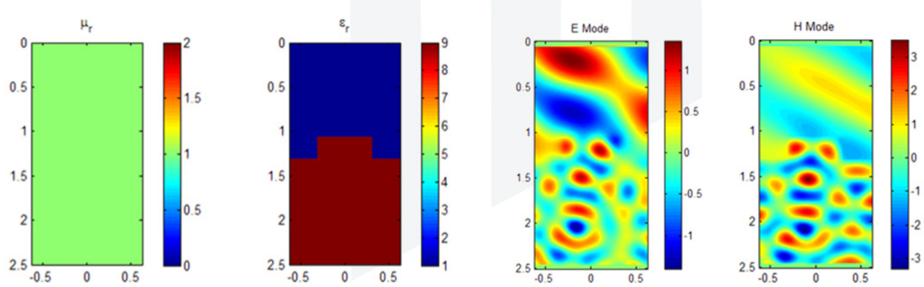
65

Binary Diffraction Grating





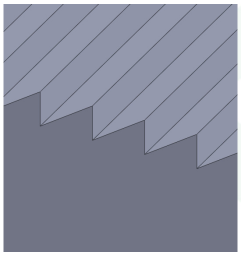
	E Mode		H Mode	
m	R	T	R	T
-4	0%	4.6%	0%	2.4%
-3	0%	5.3%	0%	7.5%
-2	0%	2.6%	0%	3.1%
-1	6.9%	22.1%	14.1%	38.8%
0	12.7%	7.8%	0.9%	5.4%
1	0%	17.7%	0%	17.7%
2	0%	16.4%	0%	7.6%
3	0%	3.9%	0%	2.4%
4	0%	0%	0%	0%

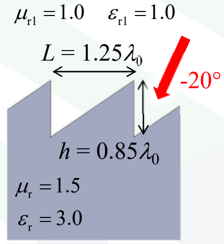


Slide 66

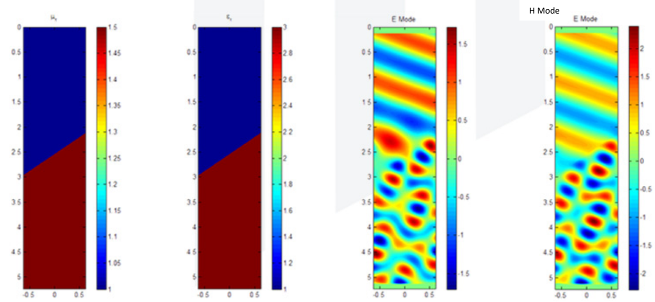
66

Sawtooth Diffraction Grating





	E Mode		*H* Mode	
<u>*m*</u>	<u>*R*</u>	<u>*T*</u>	<u>*R*</u>	<u>*T*</u>
-4	0%	0%	0%	0%
-3	0%	0%	0%	0%
-2	0%	5.6%	0%	0.4%
-1	0%	39.2%	0%	53.1%
0	0%	0.8%	0%	3.4%
1	0%	44.2%	0%	29.5%
2	0%	10.1%	0%	13.6%
3	0%	0%	0%	0%
4	0%	0%	0%	0%



EMPossible

67