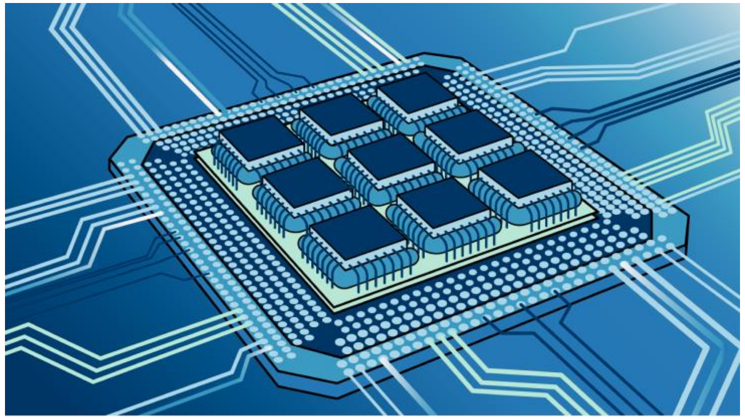
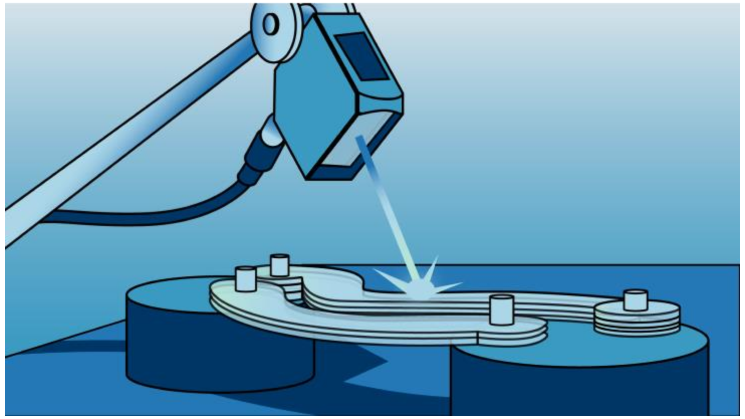
Finite-Difference Time-Domain Method (FDTD)

Metasurfaces

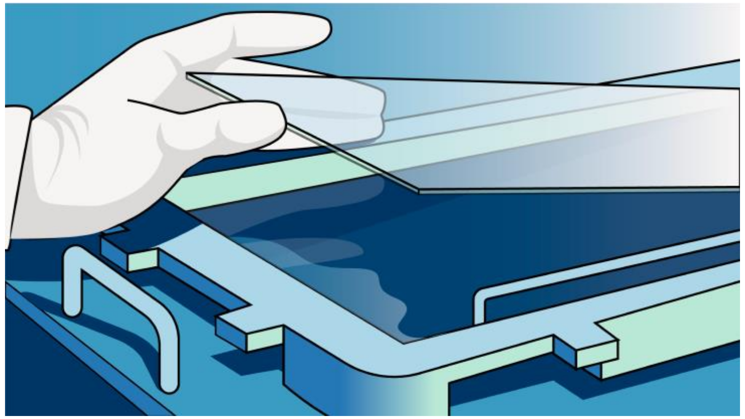


Photonic Integrated Circuits

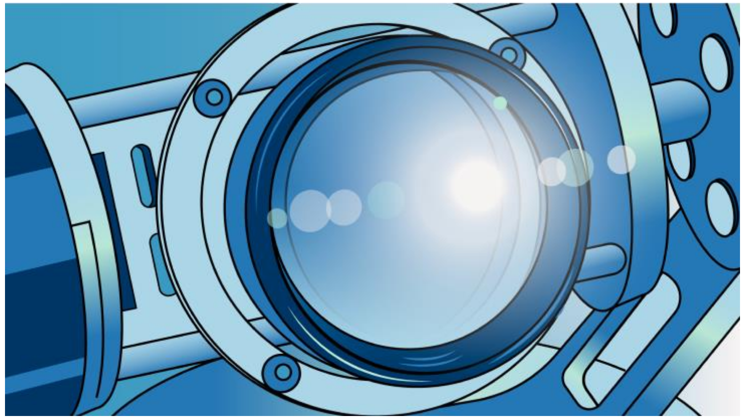
LIDAR/RADAR



Sensors



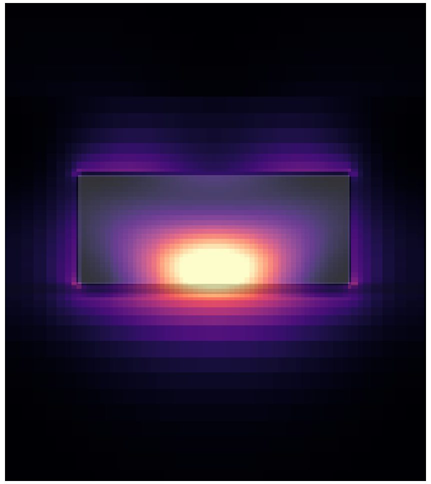
Displays



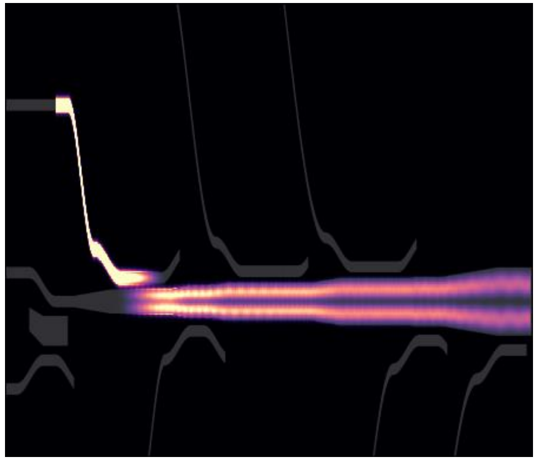
Cameras

What can FDTD simulations do?

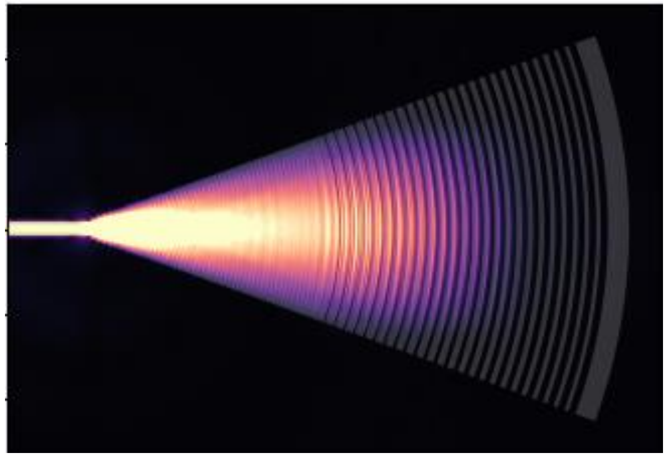
- Transmittance
- Reflectance
- Absorbance
- Coupling Efficiency
- Insertion Losses
- Quality Factor
- Mode Volume
- Far-field radiation
- Scattering Cross-section



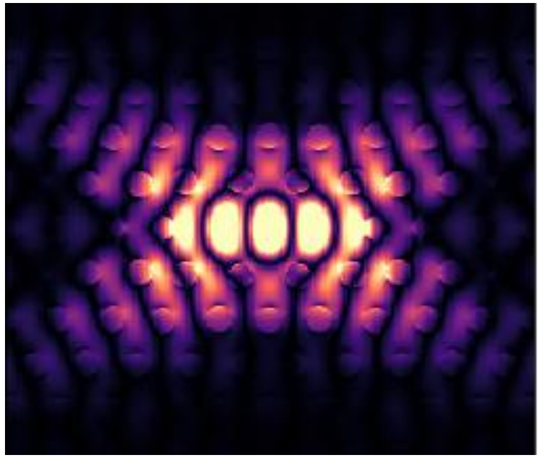
Dielectric Metasurface



Mode and Polarization Demultiplexer

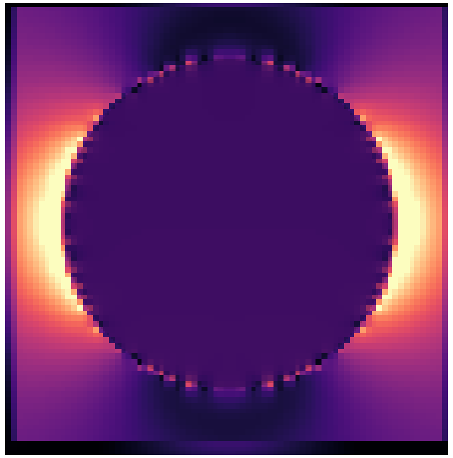


Grating Coupler



PhC Nanocavity

Metalens



Metal nanoparticle

How does an FDTD solver work?

- Proposed in 1966 by Kane Yee.

Maxwell's equations

$$\nabla \times \vec{E} = -\mu \frac{\partial \vec{H}}{\partial t} \qquad \nabla \times \vec{H} = \varepsilon \frac{\partial \vec{E}}{\partial t}$$

Finite-difference approximations

$$-\frac{1}{\mu} \nabla \times \vec{E}(t) = \frac{\vec{H}\left(t + \frac{\Delta t}{2}\right) - \vec{H}\left(t - \frac{\Delta t}{2}\right)}{\Delta t} \qquad \frac{1}{\varepsilon} \nabla \times \vec{H}\left(t + \frac{\Delta t}{2}\right) = \frac{\vec{E}(t + \Delta t) - \vec{E}(t)}{\Delta t}$$

Time loop

Finished?

yes

Results

no

$$\vec{H}\left(t + \frac{\Delta t}{2}\right) = \vec{H}\left(t - \frac{\Delta t}{2}\right) - \frac{\Delta t}{\mu} \nabla \times \vec{E}(t)$$

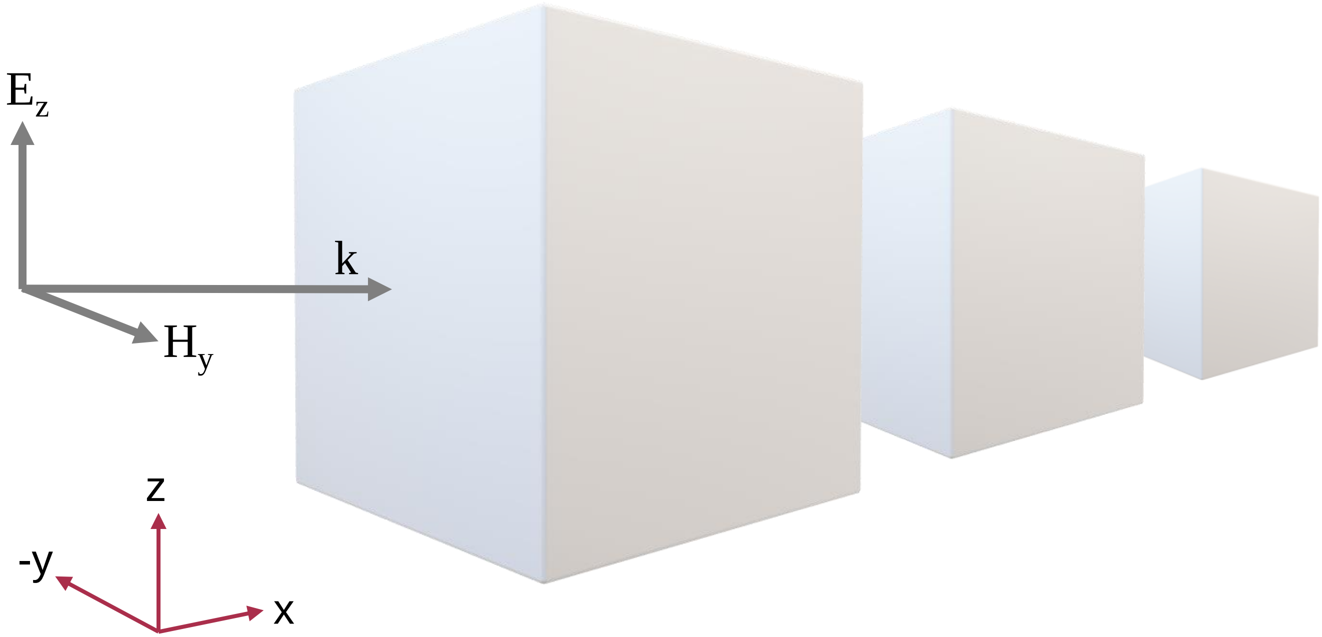
Update the $\vec{H}$ field from $\vec{E}$

$$\vec{E}(t + \Delta t) = \vec{E}(t) + \frac{\Delta t}{\varepsilon} \nabla \times \vec{H}\left(t + \frac{\Delta t}{2}\right)$$

Update the $\vec{E}$ field from $\vec{H}$

How does an FDTD solver work?

Considering material variations only in the x-direction: $\frac{\partial}{\partial y} = 0 \qquad \frac{\partial}{\partial z} = 0$



Faraday's law

$$\nabla \times \vec{E} = -\mu \frac{\partial \vec{H}}{\partial t}$$

$$-\mu \frac{\partial \vec{H}}{\partial t} = \nabla \times \vec{E} = \begin{bmatrix} \hat{a}_x & \hat{a}_y & \hat{a}_z \\ \frac{\partial}{\partial x} & 0 & 0 \\ 0 & 0 & E_z \end{bmatrix} = -\hat{a}_y \frac{\partial E_z}{\partial x}$$

Ampere's law

$$\nabla \times \vec{H} = \varepsilon \frac{\partial \vec{E}}{\partial t}$$

$$\varepsilon \frac{\partial \vec{E}}{\partial t} = \nabla \times \vec{H} = \begin{bmatrix} \hat{a}_x & \hat{a}_y & \hat{a}_z \\ \frac{\partial}{\partial x} & 0 & 0 \\ 0 & H_y & 0 \end{bmatrix} = \hat{a}_z \frac{\partial H_y}{\partial x}$$

Maxwell's scalar equations

$$\mu \frac{\partial H_y}{\partial t} = \frac{\partial E_z}{\partial x} \qquad \varepsilon \frac{\partial E_z}{\partial t} = \frac{\partial H_y}{\partial x}$$

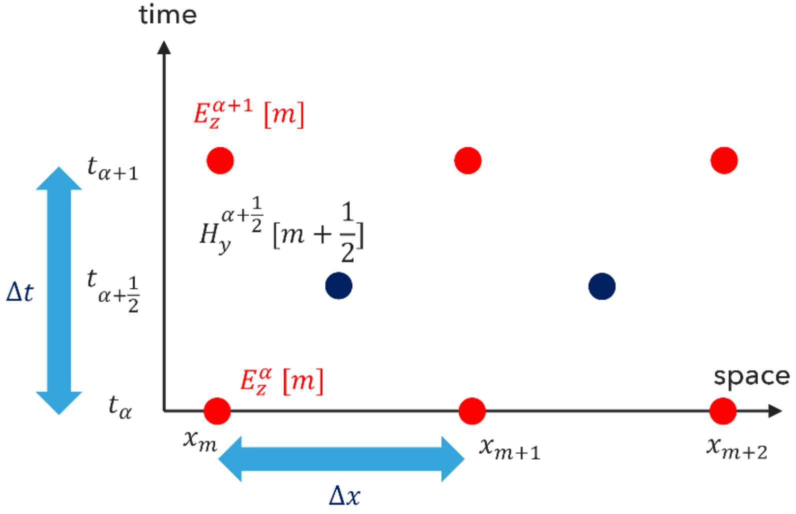
How does an FDTD solver work?

Maxwell's scalar equations

$$\mu \frac{\partial H_y}{\partial t} = \frac{\partial E_z}{\partial x} \qquad \varepsilon \frac{\partial E_z}{\partial t} = \frac{\partial H_y}{\partial x}$$

Yee Grid

- Spatial step size: Δx
- Temporal step size: Δt
- Staggered **E** and **H** in space and time



Convert differential operator to difference operator, for example:

$$\frac{\partial E_z}{\partial t} \rightarrow \frac{E_z(t+\Delta t) - E_z(t)}{\Delta t}$$

Finite-difference equations

$$\mu \frac{H_y\left(t+\frac{\Delta t}{2}\right) - H_y\left(t-\frac{\Delta t}{2}\right)}{\Delta t} = \frac{E_z(x+\Delta x) - E_z(x)}{\Delta x}$$

$$\varepsilon \frac{E_z(t+\Delta t) - E_z(t)}{\Delta t} = \frac{H_y\left(x+\frac{\Delta x}{2}\right) - H_y\left(x-\frac{\Delta x}{2}\right)}{\Delta x}$$

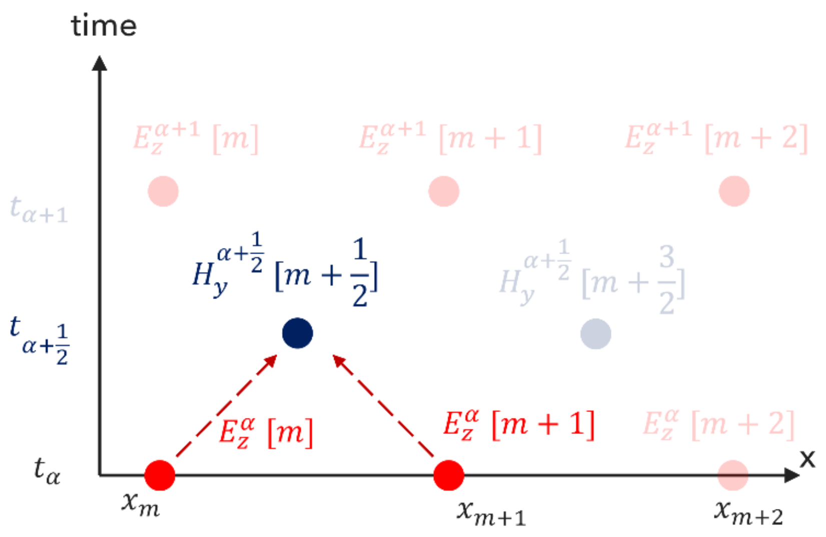
Update equations

$$H_y\left(t+\frac{\Delta t}{2}\right) = H_y\left(t-\frac{\Delta t}{2}\right) + \frac{\Delta t}{\mu \Delta x}\left[E_z(x+\Delta x) - E_z(x)\right]$$

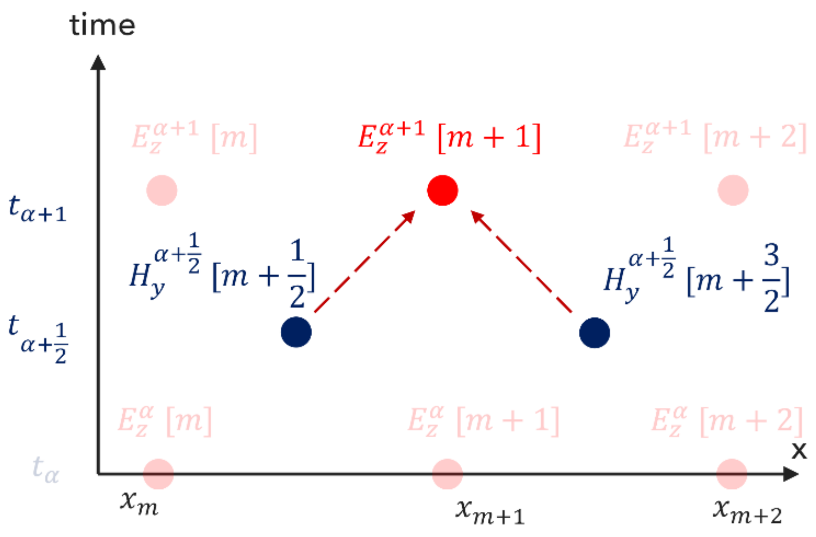
$$E_z(t+\Delta t) = E_z(t) + \frac{\Delta t}{\varepsilon \Delta x}\left[H_y\left(x+\frac{\Delta x}{2}\right) - H_y\left(x-\frac{\Delta x}{2}\right)\right]$$

How does an FDTD solver work?

Yee Grid at time step $\alpha + \frac{1}{2}$



Yee Grid at time step $\alpha + 1$



Update all $H_y^{\alpha+\frac{1}{2}}$

$$H_y^{\alpha+\frac{1}{2}}\left[m+\frac{1}{2}\right] = H_y^{\alpha-\frac{1}{2}}\left[m+\frac{1}{2}\right] + \frac{\Delta t}{\mu \Delta x}\left(E_z^{\alpha}[m+1] - E_z^{\alpha}[m]\right)$$

Update $E_z^{\alpha+1}$

$$E_z^{\alpha+1}[m+1] = E_z^{\alpha}[m+1] + \frac{\Delta t}{\varepsilon \Delta x}\left(H_y^{\alpha+\frac{1}{2}}\left[m+\frac{3}{2}\right] - H_y^{\alpha+\frac{1}{2}}\left[m+\frac{1}{2}\right]\right)$$

How does an FDTD solver work?

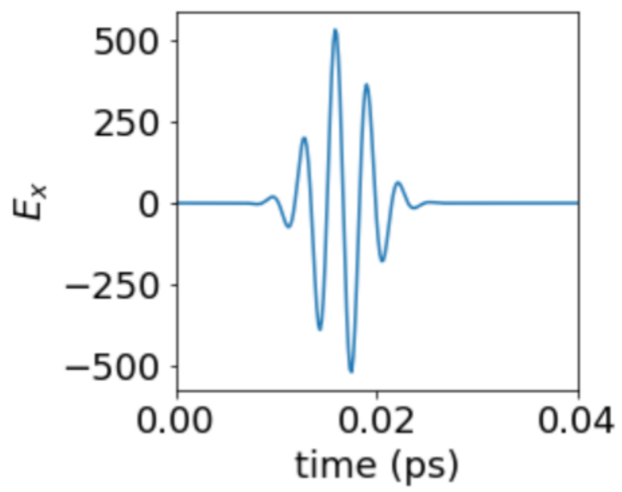
Physical light speed		Numerical light speed
c	$\leq$	$\dfrac{\Delta x}{\Delta t}$

Courant number:

$$C = \frac{c\Delta t}{\Delta x} \leq 1$$

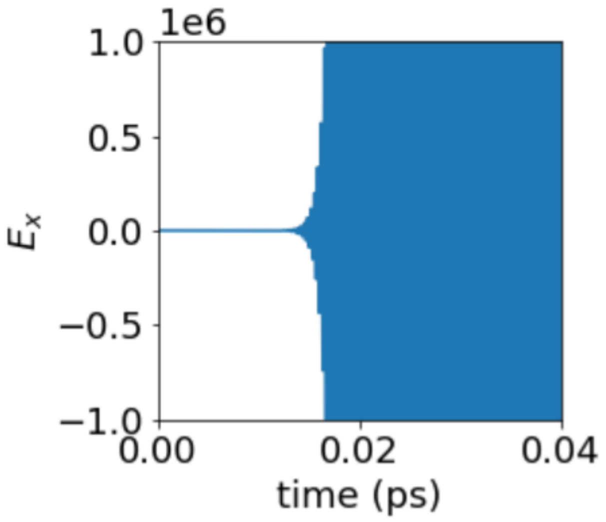
The minimum time step is limited by the spatial grid size.

$\Delta t = 0.99\Delta x/c$



Courant number: $C = 0.99$

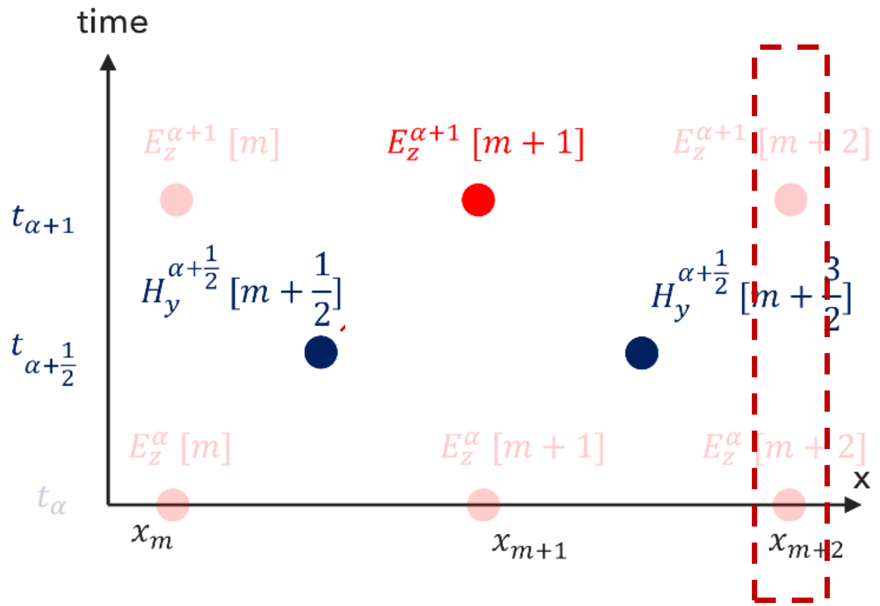
$\Delta t = 1.01\Delta x/c$



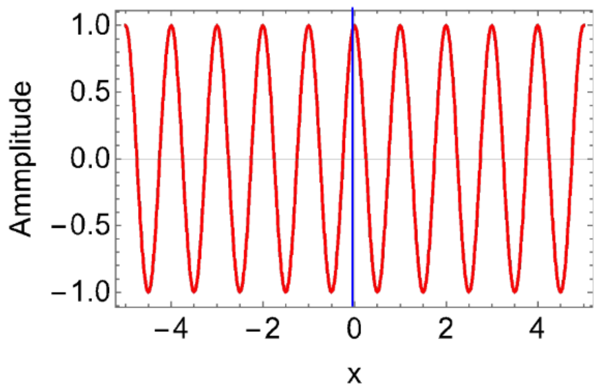
Courant number: $C = 1.01$

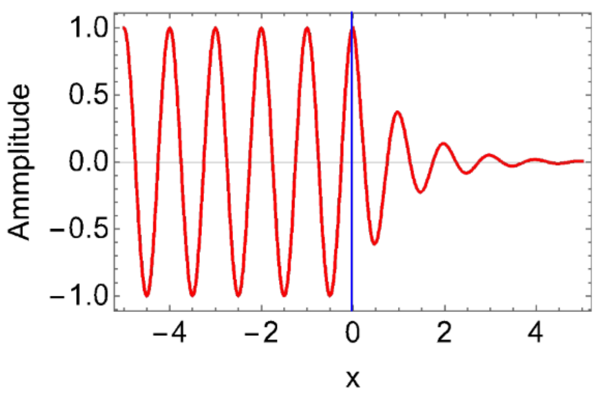
How does an FDTD solver work?

Boundary Conditions:



Absorbing Boundary Condition





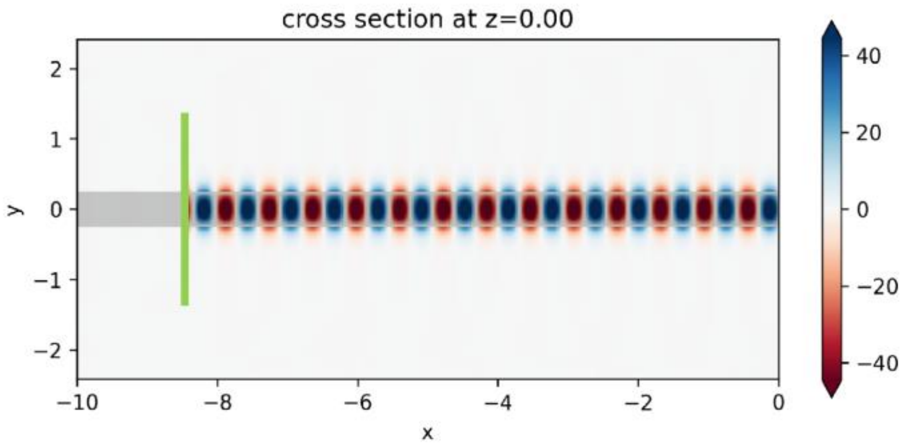
?

$$E_z^{\alpha+1}[m+2] = E_z^{\alpha}[m+2] + \frac{\Delta t}{\varepsilon \Delta x}\left(H_y^{\alpha+\frac{1}{2}}\left[m+\frac{5}{2}\right] - H_y^{\alpha+\frac{1}{2}}\left[m+\frac{3}{2}\right]\right)$$

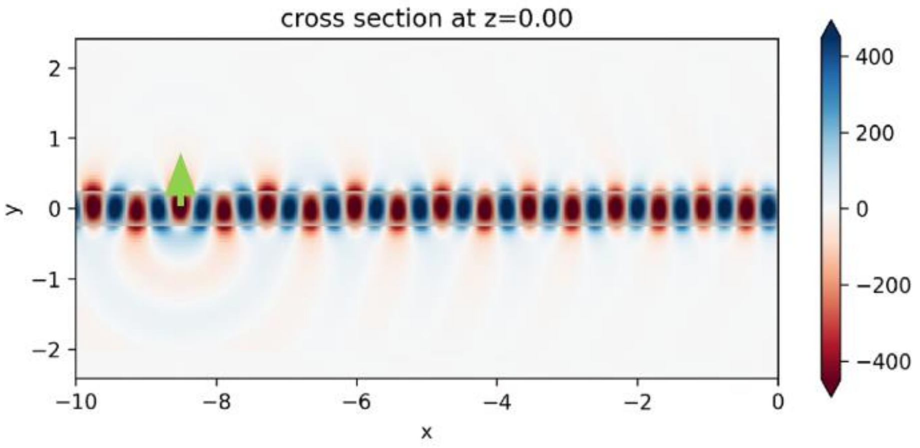
How does an FDTD solver work?

Source

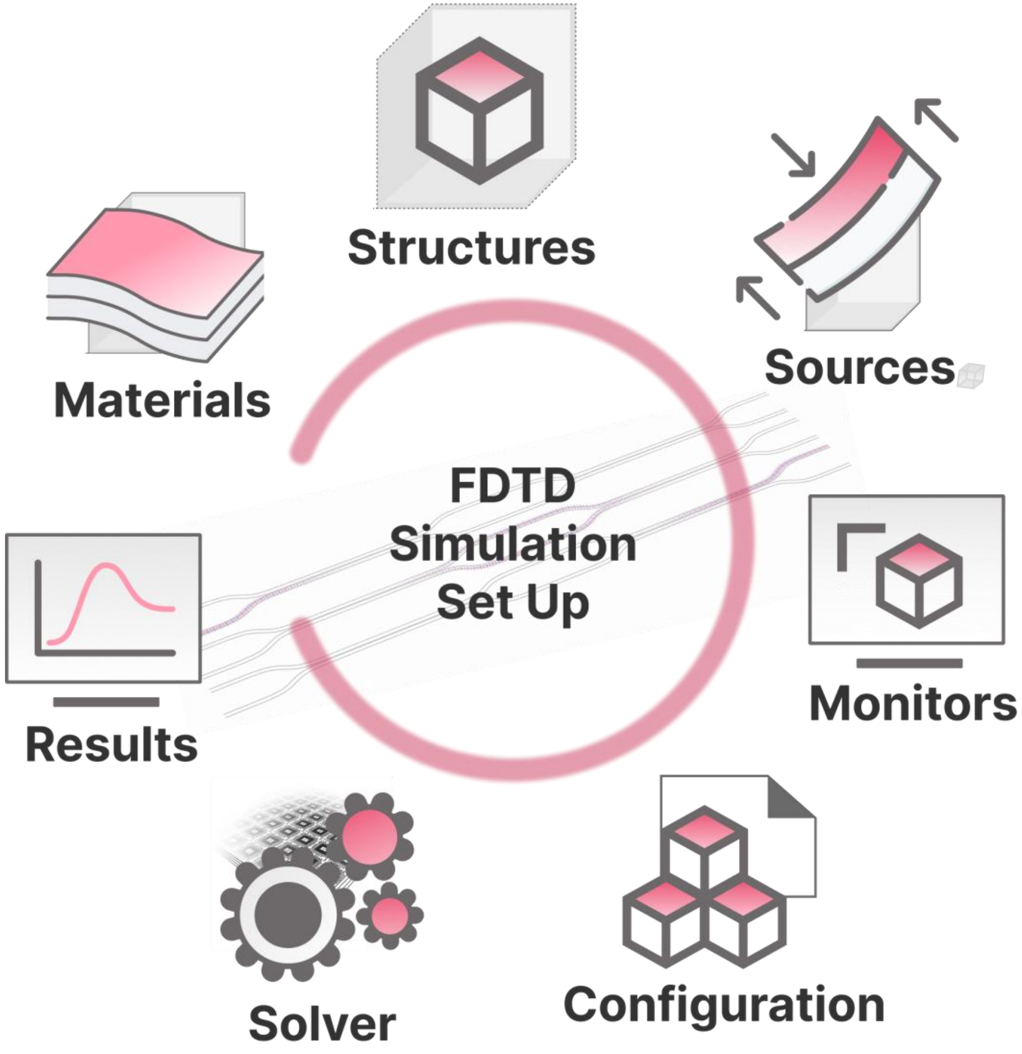
Model source to excite the 1st TE mode



Point dipole source excitation



How to set up a FDTD simulation



Conclusion

- ❑ Maxwell's equations are approximated by finite-differences.
- ❑ FDTD can obtain the response for multiple frequencies in a single simulation.
- ❑ The FDTD algorithm should include:
- ✓ Structures and Materials
- ✓ Discretization
- ✓ Update equations
- ✓ Boundary Conditions
- ✓ Sources
- ✓ Monitors

Tidy3D

Sign up for Free

Cloud-based FDTD solver

100–1000X larger

10–100X faster

Python-based User Interface

Graphical User Interface

FDTD 101

Learn FDTD, from basics to advanced topics.

Examples Library

Explore various FDTD simulation examples, such as photonic integrated circuits, metasurfaces, photonic crystals, metalenses, inverse designs, and more.

Inverse Design

Learn the adjoint-based inverse design method.

Tidy3D GUI

Step-by-step tutorials on using Tidy3D graphical user interface (GUI). Practical examples included.

Tidy3D Python

Complete documentation to Python-based Tidy3D user interface.