Measuring Diffraction in Tidy3D

Alec Lau, Momchil Minkov, Shashwat Sharma

Abstract

For simulations in which light interacts with diffractive devices, Tidy3D offers an easy way to compute diffraction orders and the amplitudes associated with them. One constructs a simulation of the unit cell of the periodic structure, with periodic or Bloch boundary conditions, and adds a diffraction monitor, which must span the entire simulation domain. Here we present how diffraction orders and their field amplitudes are calculated given the monitors' captured field data.

Process

Suppose we have a simulation injecting a source along the z-axis at angles θ, ϕ with a central frequency f_s, and we're measuring the diffraction of the resulting field data at frequency f_m with a monitor normal to the z-axis.

If the material this wave is traveling through has complex permittivity ϵ at f_s, the source's wavenumber k is $2\pi Re(\sqrt{\epsilon})\frac{f_s}{c}$. Since the injection axis is the z-axis, the wavevector $\mathbf{k}$ is $k(\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$. (If the injection axis and monitor normal were the x-axis, $\mathbf{k}$ would be $k(\cos\theta, \sin\theta\cos\phi, \sin\theta\sin\phi)$, and if they were the y-axis, $\mathbf{k}$ would be $k(\sin\theta\cos\phi, \cos\theta, \sin\theta\sin\phi)$.)

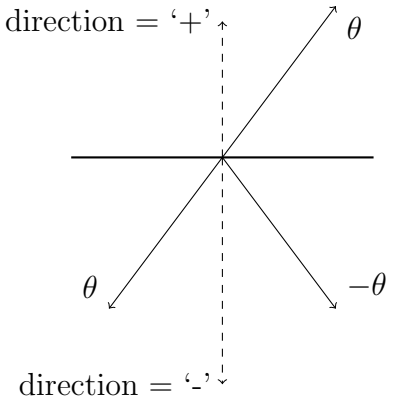
Given the field data that the monitor has captured, we isolate the components of the field tangential to the monitor. We will colocate the field data to a regular grid (since later we will obtain the diffraction orders via a Fourier transform) with

coordinates $\{x_0, ..., x_{N_x}\}, \{y_0, ..., y_{N_y}\}$, where N_x, N_y are the number of x and y data points we have, respectively.

Note on angle definitions

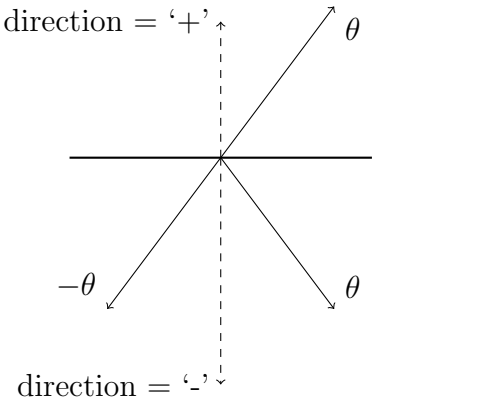
Users may notice that angles from e.g. plane wave sources are defined by the user, but not in diffraction monitors. This is because angles here are determined by the simulation domain size and bloch vector. Looking at the definition of the Bloch vectors, flipping the sign of θ, since $\sin\theta$ is odd, results in the flipping of the Bloch vector ($\sin -\theta = -\sin\theta$) Thus a DiffractionMonitor has the same θ regardless of whether the normal direction is '+' or '-', as in the picture below.

Angle θ conventions for DiffractionMonitors:



This is opposed to the angle definitions in a plane source, where θ can be specified (see below):

Angle θ conventions for PlaneWave sources:



Fields from surface currents

Recall that diffraction monitors span the entire simulation domain in 2D, in this case x and y. Due to the Electromagnetism Uniqueness Theorem and the Surface Equivalence Theorem, we can define electric and magnetic currents on our monitor surface such that the electric and magnetic fields on the source's side of the monitor are 0, and our E and H fields are the same on the other side of the monitor:

$$J = \hat{n} \times H, \; M = -\hat{n} \times E,$$

where $\hat{n}$ is the unit normal vector to our monitor.

The profiles for our diffracted fields in the direction k' can be written in terms of

our currents. Namely,

$$E_\theta = -\frac{1}{4\pi}\left(-L_\phi(k'_x, k'_y) + \eta N_\theta(k'_x, k'_y)\right) \qquad E_\phi = -\frac{1}{4\pi}\left(L_\theta(k'_x, k'_y) - \eta N_\phi(k'_x, k'_y)\right)$$
$$H_\theta = -\frac{1}{\eta}E_\phi \qquad H_\phi = \frac{1}{\eta}E_\theta$$

where $L(k'), N(k')$ are the following surface integrals, written in terms of our currents:

$$L(k'_x, k'_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} M(x', y')e^{-i2\pi(k'_x x' + k'_y y')}dy'dx',$$
$$N(k'_x, k'_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} J(x', y')e^{-i2\pi(k'_x x' + k'_y y')}dy'dx'$$

See the far-field vector potential formulas in our post on far-field projection or Chapter 8 of [1] for more details. Note that in these references, the currents are assumed to be compactly supported within the boundaries of the surface - due to periodicity here, there is no assumption.

Bloch's Theorem, Fourier series, and Fourier transform

Notice that these surface integrals are Fourier transforms on J and M, and therefore $A(x, y) \in \{E_x, E_y, H_x, H_y\}$. Given the Bloch periodicity of the simulation, we invoke Bloch's Theorem:

$$A(x, y) := \overline{A}(x, y)e^{i2\pi(k_x x + k_y y)}$$

where $\overline{A}$ is periodic in d_x and d_y, and k_x, k_y are the Bloch vectors of A's Bloch state.

The integral becomes

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} A(x',y')e^{-i2\pi(k_x'x'+k_y'y')}dy'dx'$$
$$=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \overline{A}(x',y')e^{-i2\pi(k_x'x'+k_y'y')}e^{i2\pi(k_xx'+k_yy')}dy'dx'$$

Since $\overline{A}$ is periodic, we can write it as a Fourier series and evaluate the resulting integrals:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sum_{n_x,n_y=-\infty}^{\infty} \overline{A}_{n_x,n_y}e^{i2\pi(\frac{n_x}{d_x}x'+\frac{n_y}{d_y}y')}e^{-i2\pi(k_x'x'+k_y'y')}e^{i2\pi(k_xx'+k_yy')}dy'dx'$$
$$=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sum_{n_x,n_y=-\infty}^{\infty} \overline{A}_{n_x,n_y}e^{-i2\pi((k_x'-k_x-\frac{n_x}{d_x})x'+(k_y'-k_y-\frac{n_y}{d_y})y')}dy'dx'$$
$$=\sum_{n_x,n_y=-\infty}^{\infty} \overline{A}_{n_x,n_y}\delta\left(k_x'-k_x-\frac{n_x}{d_x}\right)\delta\left(k_y'-k_y-\frac{n_y}{d_y}\right)$$

where $\overline{A}_{n_x,n_y}=\frac{1}{d_xd_y}\int_0^{d_x}\int_0^{d_y}\overline{A}(x,y)e^{-i2\pi(\frac{n_x}{d_x}x+\frac{n_y}{d_y}y)}dxdy$. This is the continuous version of numpy's 2D Discrete Fourier Transform of $\overline{A}(x,y)=A(x,y)e^{-i2\pi(k_xx+k_yy)}$ multiplied by a factor of 2π due to angular frequency convention.

Allowed Orders

Note that there is a geometric constraint there. The delta functions ensure that our wavevectors k_x', k_y' for L and N must be $k_x \pm \frac{n_x}{d_x}, k_y \pm \frac{n_y}{d_y}$, respectively, for $n = 0, 1, 2...$ Since k_x', k_y' are the components of a Bloch vector, we can write these in the context of a plane wave traveling at some angle θ:

$$k_x' = k_x \pm \frac{n_x}{d_x} := k\sin\theta_{n_xn_y}$$
$$k_y' = k_y \pm \frac{n_y}{d_y} := k\cos\theta_{n_xn_y}$$

for some new Bloch vector k.

We therefore have

$$\theta'_{n_x n_y} = \arcsin \frac{2\pi(k_x \pm \frac{n_x}{d_x})}{k} = \arccos \frac{2\pi(k_y \pm \frac{n_y}{d_y})}{k}$$

Since the domains of arcsine and arccosine must be [-1,1], we have the following constraints on diffraction orders n_x, n_y:

$$\left|\frac{2\pi(k_x \pm \frac{n_x}{d_x})}{k}\right| \leq 1, \quad \left|\frac{2\pi(k_y \pm \frac{n_y}{d_y})}{k}\right| \leq 1$$

Recall that k is also frequency-dependent, so the allowed orders must be filtered from the sum according to this constraint for each frequency that the monitor records.

Normalization

We normalize our ModeSources, PlaneWaves, GaussianBeams, and AstigmaticGaussianBeams to inject 1 Watt of power at every frequency. Our E and H components are thus normalized in terms of $V/\mu m$ and $A/\mu m$, respectively, so that when they are multiplied together and integrated over the area of the simulation, such as when we calculate their Poynting vector, we get (1) Watts. The same must be the case for our L and N components.

When we start with field components in units of $\frac{V}{\mu m}$, the units of the 2D Fourier transform is $\frac{V}{\mu m^2}$. Multiplying this by the $\frac{1}{d_x d_y}$ factor to get the Fourier series coefficients and evaluating the double integral results in units of $\frac{V}{\mu m^2}$. Multiplying this by $\sqrt{d_x d_y}$ allows us, when multiplying with another such field component and integrating this over the simulation area, to reach unity.

Also, since the Fourier transform introduces a factor $\frac{1}{\sqrt{2\pi}}$ for each dimension, we

also multiply by 2π in order to get $\overline{A}$'s Fourier coefficients from numpy's 2D DFT.

Phase recentering

In order to set the phase as zero in the monitor center, we multiply the fields at each n_x, n_y by the phase factor $e^{i2\pi(k_x+\frac{n_x}{d_x})x_{center}}e^{i2\pi(k_y+\frac{n_y}{d_y})y_{center}}$, where (x_{center}, y_{center}) is the center of the (regular) discretization on the monitor.

Final Summation to get L and N

Thus, our procedure to get our $L(k'_x, k'_y)$ and $N(k'_x, k'_y)$ is as follows:

1. Take our recorded E and H fields and multiply them by $e^{-2\pi(k_x x+k_y y)}$ to get $\overline{E}$ and $\overline{H}$
2. Normalize by multiplying by $2\pi\sqrt{d_x d_y}$
3. Take the 2D Discrete Fourier transform (and shift the phase so that $n_x, n_y = 0$ is in the middle)
4. Recenter the phase by multiplying this value at each (x, y) by $e^{i2\pi(k_x+\frac{n_x}{d_x})x_{center}}$ and $e^{i2\pi(k_y+\frac{n_y}{d_y})y_{center}}$ for allowed orders n_x, n_y

Field profiles

Following the above procedure, we have

$$L(k'_x, k'_y) = \hat{n} \times 2\pi\sqrt{d_x d_y}\mathcal{FFT}_{2D}\left(E(x,y)e^{-i2\pi(k_x x+k_y y)}\right)e^{i2\pi[(k_x+\frac{n_x}{dx})x_{center}+(k_y+\frac{n_y}{dy})y_{center}]}$$
$$N(k'_x, k'_y) = -\hat{n} \times 2\pi\sqrt{d_x d_y}\mathcal{FFT}_{2D}\left(H(x,y)e^{-i2\pi(k_x x+k_y y)}\right)e^{i2\pi[(k_x+\frac{n_x}{dx})x_{center}+(k_y+\frac{n_y}{dy})y_{center}]}$$

where $\mathcal{FFT}_{2D}$ denotes the recentered 2-dimensional Fourier transform given our allowed orders.

We now convert L and N to spherical coordinates to complete our field profile using the above relationship of E and H to L and N:

$$E_\theta = -\frac{1}{4\pi}(-L_\phi + \eta N_\theta) \qquad E_\phi = -\frac{1}{4\pi}(L_\theta - \eta N_\phi)$$
$$H_\theta = -\frac{1}{\eta}E_\phi \qquad H_\phi = \frac{1}{\eta}E_\theta$$

Diffraction data

We have already calculated the allowed orders and field profiles. From this data, we can calculate further diffraction data:

- Our u_x, u_y vectors are $\frac{2\pi(k_x \pm \frac{n_x}{d_x})}{k}$ and $\frac{2\pi(k_y \pm \frac{n_y}{d_y})}{k}$, respectively, where k is our source's wavenumber in the monitor's medium.

- We can get the (θ, ϕ) angles corresponding to each allowed pair of diffraction orders by converting our u_x, u_y vectors to spherical coordinates; for

$$\theta_l := \arcsin\left(\sqrt{u_x^2 + u_y^2}\right), \qquad \phi_l := \arctan\frac{u_y}{u_x} \rightarrow$$
$$x := \cos(\theta_l), \qquad y := \sin(\theta_l)\cos(\phi_l), \qquad z := \sin(\theta_l)\sin(\phi_l)$$

  we have $\theta_{n_x n_y} = \arccos(z)$, $\theta_{n_x n_y} = \arctan(\frac{y}{x})$.

- To get the complex amplitudes in each order for S and P polarizations, we take the $\theta_{n_x n_y}$ values calculated above for each of the allowed diffraction orders and create a normalization factor $\frac{1}{\sqrt{2\eta\cos\theta_{n_x n_y}}}$ for those $\theta_{n_x n_y}$ that are greater than zero. The reason for this normalization factor is, since we can see above, $|H| = \frac{|E|}{\eta}$, so the

components of the Poynting vector are proportional to $\frac{|E|^2}{\eta}$. The time-averaged Poynting vector is therefore $\frac{E_{rms}^2}{\eta}$, where we connect E_{rms} to our E field data via $E_{rms} = \frac{E}{\sqrt{2}}$. To find the power in the $\theta_{n_x n_y}$ direction, we divide by $\sqrt{\cos\theta_{n_x n_y}}$.

The S polarization amplitudes are therefore $\frac{1}{\sqrt{2\eta\cos\theta_{n_x n_y}}} E_\phi(n_x, n_y)$ and the P polarization amplitudes are $\frac{1}{\sqrt{2\eta\cos\theta_{n_x n_y}}} E_\theta(n_x, n_y)$

Acknowledgements

The Tidy3D diffraction API was written by Momchil Minkov, Ph.D., Director of Photonics at Flexcompute, and Shashwat Sharma, Ph.D., Computational Scientist at Flexcompute.

References

[1] A. Taflove, S.C. Hagness, M. Piket-May. "Computational electromagnetics: the finite-difference time-domain method." The Electrical Engineering Handbook 3 (629-670), 15