The Finite-Difference Time-Domain Method: Discretization of Maxwell's Equations in Python

Liam Karl
Department of Physics and Astronomy, University of Southern California, Los Angeles, California 90089, USA

(Dated: April 24, 2024)

In the following replication study, I will be detailing the Finite-Difference Time-Domain Method (FDTD) as devised by Kane S. Yee in 1966 [1]. The FDTD Method is a powerful approach to producing numerical solutions of Maxwell's equations through staggered-grid discretization of space-time. I will first be explicating the theoretical justification for the method and its constituent parts. Then, I will proceed to discuss my Python implementation of Yee's algorithm as it was presented in his original paper. This implementation will be used to simulate electromagnetic scattering from two simple geometries: a one-dimensional conducting medium and the two-dimensional conducting square discussed in Yee's paper [1]. I will conclude with a comparison of my simulation and Yee's original results, elucidating any notable takeaways from the discussion.

Keywords: Finite-Difference Time-Domain Method, Central Difference Approximations, Electromagnetic Waves, Maxwell's Equations

I. INTRODUCTION

The **Finite-Difference Time-Domain (FDTD) Method** is one of the simplest and most common approaches by which numeric solutions to electromagnetic problems can be obtained [2]. Its application have been useful in a variety of problems: scattering from metal objects and dielectrics, antennae, micro-strip circuits, and electromagnetic absorption in the human body exposed to radiation being choice examples [3].

In broad strokes, the FDTD Method can solve **Maxwell's equations** by discretizing the latter with **finite difference approximations** [3]. These approximations, detailed further in Section II B, solve for the derivative of a function at a given point of interest based on the values of the function at surrounding points [4]. Given a continuous field, we can utilize these methods to find the approximate field derivatives in time and space at discrete points.

This spatiotemporal discretization–that is, the making-discrete of continuous fields in both space and time domains–of differential equations had long been utilized in computational fluid dynamics problems [5]. Kane S. Yee's 1966 work broadened the application of finite-difference methods to electromagnetic problems [1]. Its capabilities have been expanded and improved since, but the focus of this replication study is on Yee's original paper.

The computational benefits of Yee's approach are numerous. Due to the discretization of space, field values at each point in space only depend on their neighboring points rather than the entire matrix of field values [5]. Thus, linear algebra is not required in implementation, which saves a lot of computational resources [5]. Additionally, due to the discretization of time, previous field values do not generally need to be stored unless dealing with specific cases as in dispersive media or particular boundary conditions [5]. This reduces computational demands even further.

Though, the FDTD Method is not without drawbacks and limitations. As will be discussed in greater detail in Section II D, the stability of the method relies on small time-steps. For larger wavelengths, as observed in the case of MRI systems, the time-stepping stability criterion creates a constraint in computational feasibility [6].

II. THEORETICAL BACKGROUND

The key principle of the FDTD is replacing the derivatives of the E-field and H-field in Maxwell's equations with approximate forms discretized in space and time. Here, the constituent parts that contribute to this discretization will be explained in detail.

A. Continuous Maxwell Equations

We begin with Maxwell's equations for isotropic media in their continuous form. These are the four coupled partial differential equations that dictate the propagation of electromagnetic waves [5].

$$\frac{\partial \boldsymbol{B}}{\partial t} + \boldsymbol{\nabla} \times \boldsymbol{E} = 0, \tag{1}$$

$$\frac{\partial \boldsymbol{D}}{\partial t} - \boldsymbol{\nabla} \times \boldsymbol{H} = \boldsymbol{J}, \tag{2}$$

$$\boldsymbol{B} = \mu \boldsymbol{H}, \tag{3}$$

$$\boldsymbol{D} = \epsilon \boldsymbol{E}, \tag{4}$$

where $\boldsymbol{J}$, $\boldsymbol{\mu}$, and $\boldsymbol{\epsilon}$ are functions of space and time [1].

As seen in Figure 1, Yee's paper takes the boundary conditions for the electromagnetic wave obeying these equations to be those of the perfect conductor. In this case, all tangential components of our E-field as well as

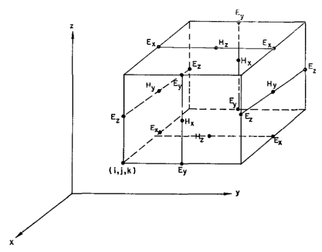


FIG. 1. The conducting surface can be approximated by a collection of surfaces of cubes whose sides are parallel to the coordinate axes. E-components are positioned in the middle of the edges, and H-components in the center of the faces. This figure was taken from Ref. [1].

the normal component of our H-field vanish at the boundary surface [5].

B. Central Difference Approximations [4]

We have established what Maxwell's equations look like in their unaltered, continuous form for an isotropic medium. We can now proceed to introduce finite-difference approximations.

The limit definition of a derivative for some function f at the point x is given as:

$$f'(x) = \lim_{\Delta x \to 0} \frac{f\left(x + \Delta x\right) - f\left(x\right)}{\Delta x}$$

The difference quotient $\frac{f(x+\Delta)-f(x)}{\Delta x}$ is an approximation of the derivative $f'(x)$ which gets better as the limit of Δx is taken to 0. However, this is not the only valid way to approximate $f'(x)$.

If we use Taylor's theorem to expand $f(x + \Delta x)$, we obtain the expansion:

$$f(x + \Delta x) = f(x) + \Delta x f'(x) + \Delta x^2 \frac{f''(\xi_1)}{2!}$$

where $x \leq \xi_1 \leq x + \Delta x$ and $\Delta x > 0$.

Solving for the error of our approximation, we get a linear dependence on Δx:

$$\frac{f(x + \Delta x) - f(x)}{\Delta x} - f'(x) = \Delta x \frac{f''(\xi)}{2}$$

This error is first-order with respect to Δx. Since we are approximating the value of $f'(x)$ from the difference quotient measured between $x + \Delta x$ and x, this term is known as the "forward difference approximation", and is exactly the same difference quotient as mentioned earlier.

We can follow the same logic with Δx instead *subtracted* from x, approximating to first-order the value of $f'(x)$ from the difference quotient measured between x and $x - \Delta x$. This term is known as the "backward difference approximation", and can be characterized by the Taylor series expansion:

$$f(x - \Delta x) = f(x) + \Delta x f'(x) - \Delta x^2 \frac{f''(\xi_2)}{2!}$$

where $x - \Delta x \leq \xi_2 \leq x$ and $\Delta x > 0$.

If we subtract the third-order Taylor series expansions characteristic of the forward and backward difference quotients, we obtain:

$$\begin{aligned} f(x + \Delta x) - f(x - \Delta x) &= 2\Delta x f'(x) \\ &+ \Delta x^3 \frac{(f'''(\xi_1 + f'''(\xi_2))}{6} \end{aligned}$$

Solving for the error of our approximation, we again get a linear dependence on Δx:

$$\frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x} - f'(x) = \Delta x^2 \frac{f'''(\xi_1) + f'''(\xi_2)}{12}$$

This error is second-order with respect to Δx, and thus a more accurate approximation relative to the forward and backward difference approximations. Since we are approximating the value of $f'(x)$ from the difference quotient measured between $x + \Delta x$ and $x - \Delta x$, this term is known as the "central difference approximation".

With this, we can find an approximate value for a derivative at the point of interest x without having any information about the function f at that point. This ability will prove indispensable for calculating field derivatives in a discretized space-time.

There are higher-order approximations that we could ostensibly use for FDTD, but these create complications. Particularly, they need to sample more points that are further away from the point of interest. This can be problematic, especially as we approach a conducting boundary condition [2].

We will see the concrete application of the central difference approximation in discretizing the space-time derivatives of E- and H-fields in Section III.

C. Staggered Grid

To more accurately calculate field derivatives in our discretization of Maxwell's Equations, our space-time grid (hereafter known as the **Yee grid**) needs to conform to both of the following two qualities: 1) Staggered 2) Uncollocated

A coordinate system is staggered if and only if each variable (in our case, E-field and H-field values) is defined in a set of points disjoint from other variables. For the

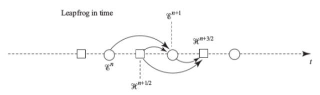


FIG. 2. Visual depiction of the interleaved Leapfrog algorithm. Note the time-staggering of field values. This figure was taken from Ref. [5].

purposes of FDTD, every field will need to be defined at a different set of points in space.

A coordinate system is uncollocated if and only if each component of a tensor variable (in our case, the respective Cartesian components of the E-field and H-field) is defined in a set of points disjoint from the other components. For the purposes of FDTD, every component of field will need to be defined at a different set of points in space.

The simplest grid is Cartesian, which is unstaggered and collocated. Thus, all fields and their components are defined on the same set of points. We cannot use a simple Cartesian grid due to the problem of **phase-velocity anisotropy** requiring the choice of very fine discretization. This would imply very long computation times.

To implement the time-staggering necessary for FDTD, the Yee grid utilizes an interleaved leapfrog algorithm. Figure 2 illustrates the algorithm's main logic: calculate spatial derivatives from field values recorded at time step n. Time derivatives in this approach are to be calculated from field values recorded at the surrounding time steps: $n-1$ and $n+1$. This means that the FDTD method models time evolution by computing and storing the values of one field on the basis of the values of the other field previously stored in memory, then repeating at the next time-step with these fields alternated [5].

D. Stability Criterion

I will now explain the upper-bound on the time-step in the FDTD method. This will involve a discussion of the Courant-Friedrichs-Lewy (CFL) condition and how it can be accommodated in this context.

The CFL condition states that a finite difference formula can only be stable if its numerical domain of dependence is at least as large as its mathematical domain of dependence [7]. In other words, information can only travel at a finite speed (in this case, the speed of light c), so only some space-points will be able to affect others in a finite amount of time. If this speed limit is broken by the finite difference formula, it will not produce stable or convergent solutions.

Because the space grid size should be such that the EM-field does not change significantly over one Yee cube, then the linear dimension should be constrained to a fraction of the wavelength of the incident wave. We take our cubes to be of uniform length, width, and height such that:

$$\Delta x = \Delta y = \Delta z$$

[1]

Because wave information cannot travel faster than the max speed of light for a given medium, the CFL condition for a convergent and stable set of solutions to the discretized Maxwell's Equations is as follows:

$$\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} > c_{max} \Delta t$$

[1]
Where c_{max} is the maximum light velocity in the region concerned [1].

Furthermore, the Lax Equivalence Theorem dictates that, for linear problems, convergence is equivalent to stability [7]. As a result, the above condition provides with us not only stable solutions, but convergent ones.

There are important implications of this stability criterion on the FDTD method, particularly on temporal accuracy and computational efficiency. According to this condition: The bigger our grid size, the faster (smaller time-step) this information needs to travel. The larger our time-step, the greater distance this information has to travel per time-step.

This condition usually makes it pointless to try to use higher order accuracy in time than what is used in space. Although doing so would increase the temporal accuracy, stability constraints would prevent this from being utilized to gain computational efficiency through the use of significantly larger time steps.

This can pose significant problems, particularly in the case of required field resolutions that are very high. For instance, The required field resolution in the human body, for example, is of the order of 10^-2m, and the wavelength is in the range of $>= 10^3 m$. We label this as a **high definition problem** [4].

In the high-definition problem, the spatial resolution is a very high number of points per wavelength, on the order of 10^4. Due to the CFL Condition, we end up with such small time-steps that the required number of time-steps is massive, greatly impacting computation times. This is an objective hurdle created by the stability criterion on the FDTD method, but it will be an important one to account for in our code.

III. DISCRETIZATION OF MAXWELL'S EQUATIONS

A. Derivation in One Dimension

I will now show the derivation of the finite-difference equations for the transverse EM wave in one dimension.

In free space, the one-dimensional Maxwell Equations can be given by, without loss of generality:

$$\frac{\partial E_x}{\partial t} = -\frac{1}{\epsilon_0}\frac{\partial E_y}{\partial z} \tag{5}$$

$$\frac{\partial H_y}{\partial t} = -\frac{1}{\mu_0}\frac{\partial E_x}{\partial z} \tag{6}$$

[3]

Applying central difference approximations to these equations and shifting both values in space by 1/2 a space-step and in time by 1/2 a time-step will get us the following set of finite-difference equations:

$$\frac{E_x^{n+\frac{1}{2}}(k) - E_x^{n-\frac{1}{2}}(k)}{\Delta t} = -\frac{1}{\epsilon_0}\frac{H_y^n(k) - H_y^n(k)}{\Delta z} \tag{7}$$

$$\frac{H_y^{n+1}(k+\frac{1}{2}) - H_x^n(k+\frac{1}{2})}{\Delta t} = -\frac{1}{\mu_0}\frac{E_x^{n+\frac{1}{2}}(k) - E_x^{n+\frac{1}{2}}(k)}{\Delta z} \tag{8}$$

[3]

The left term of III A states that the derivative of the E-field at time $n\Delta t$ can be expressed as a central difference using E-field values at times $(n+\frac{1}{2})\Delta t$ and $(n-\frac{1}{2})\Delta t$.

This effectively puts the Leapfrog Algorithm discussed in Section 2 into practice. The right term of III A finds the derivative of the H field, approximating at points $k\Delta x$ as a central difference using H-field values at points $(k+\frac{1}{2})\Delta x$ and $(k-\frac{1}{2})\Delta x$.

We follow the same logic for equation III A, but instead applying to different fields defined at different points in time and space.

Rearranging the equations for the respective values $E_x^n + 1/2(k)$ and $H_y^n + 1(k+1/2)$ will put these finite difference equations in explicit form:

$$\begin{aligned} &E_x^{n+\frac{1}{2}}(k) \\ &\quad = E_x^{n-\frac{1}{2}}(k) + \frac{\Delta t}{\epsilon_0 \Delta z}(H_y^n(k-\frac{1}{2}) - H_y^n(k+\frac{1}{2})) \end{aligned} \tag{9}$$

$$\begin{aligned} &H_y^{n+1}(k+\frac{1}{2}) \\ &= H_y^n(k+\frac{1}{2}) + \frac{\Delta t}{\mu_0 \Delta z}(E_x^{n+\frac{1}{2}}(k) - E_x^{n+\frac{1}{2}}(k+1)) \end{aligned} \tag{10}$$

[3]

This can now be implemented into code, albeit with a few alterations that we will discuss in Section IV.

$$\begin{aligned}
E_z{}^{n+1}(i,j) &= E_z{}^n(i,j) \\
&\quad + Z\frac{\Delta\tau}{\Delta x}\left[H_y{}^{n+1/2}(i+\tfrac{1}{2},j) - H_y{}^{n+1/2}(i-\tfrac{1}{2},j)\right] \\
&\quad - Z\frac{\Delta\tau}{\Delta y}\left[H_x{}^{n+1/2}(i,j+\tfrac{1}{2}) - H_x{}^{n+1/2}(i,j-\tfrac{1}{2})\right] \\
H_x{}^{n+1/2}(i,j+\tfrac{1}{2}) &= H_x{}^{n-1/2}(i,j+\tfrac{1}{2}) \\
&\quad - \frac{1}{Z}\frac{\Delta\tau}{\Delta y}\left[E_z{}^n(i,j+1) - E_z{}^n(i,j)\right] \\
H_y{}^{n+1/2}(i+\tfrac{1}{2},j) &= H_y{}^{n-1/2}(i+\tfrac{1}{2},j) \\
&\quad + \frac{1}{Z}\frac{\Delta\tau}{\Delta x}\left[E_z{}^n(i+1,j) - E_z{}^n(i,j)\right].
\end{aligned}$$

FIG. 3. The set of finite difference equations that solves Maxwell's equations for the TM wave. This figure was taken from Ref. [1].

B. Derivation in Two Dimensions

We can generalize the process to more dimensions. We instead just start with Maxwell's Equations from their more general definitions and discretize through central difference approximation and spacetime-staggering in the same way as conducted for the one-dimensional example.

Because in cylindrical coordinates, we can decompose any electromagnetic wave into "transverse electric" (TE) and "transverse magnetic" (TM) fields if ϵ and μ are constant, we can choose to observe the case of the 2-D TM wave, as Yee does.

For Maxwell's Equations in 2-D, for a TM wave, the FDTD method solves out to the following set of equations, given in Fig. 3.

IV. IMPLEMENTATION

In this section, I will be detailing my implementation of the FDTD Method in code.

Arrays do not support $\frac{1}{2}$-integer grids. Thus, a mapping has to be applied that shifts the indices in a consistent way and should not create holes or lead to double occupation. To support this, we have to implement our code to stagger the grid along integer-values. The mapping is given as follows, and pictured in Fig. **??**

In the update equations of the E-field:

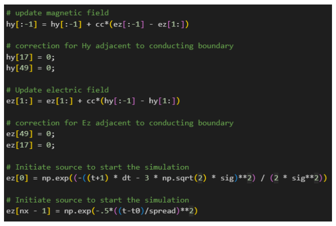
- The indices of the E-field stay the same.
- For the indices of the H-field, $x+\frac{1}{2}$ is mapped to x and $x-\frac{1}{2}$ is mapped to $x-1$.

In the update equations of the H-field:

- The indices of the H field stay the same.
- For the indices of the E-field, $x+\frac{1}{2}$ is mapped to $x+1$ and $x-\frac{1}{2}$ is mapped to x.

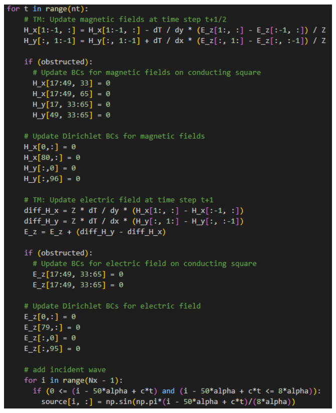
To apply this to the one-dimensional case, we used the code pictured in Fig. 4:

To apply this to the two-dimensional case, we used the code pictured in Fig. 5



```
# update magnetic field
hy[:-1] = hy[:-1] + cc*(ez[:-1] - ez[1:])

# correction for Hy adjacent to conducting boundary
hy[17] = 0;
hy[49] = 0;

# Update electric field
ez[1:] = ez[1:] + cc*(hy[:-1] - hy[1:])

# correction for Ez adjacent to conducting boundary
ez[49] = 0;
ez[17] = 0;

# Initiate source to start the simulation
ez[0] = np.exp((-((t+1) * dt - 3 * np.sqrt(2) * sig)**2) / (2 * sig**2))

# Initiate source to start the simulation
ez[nx - 1] = np.exp(-.5*((t-t0)/spread)**2)
```

FIG. 4. Code for the 1-D FDTD simulation.



```
for t in range(nt):
    # TM: Update magnetic fields at time step t+1/2
    H_x[1:-1, :] = H_x[1:-1, :] - dT / dy * (E_z[1:, :] - E_z[:-1, :]) / Z
    H_y[:, 1:-1] = H_y[:, 1:-1] + dT / dx * (E_z[:, 1:] - E_z[:, :-1]) / Z

    if (obstructed):
      # Update BCs for magnetic fields on conducting square
      H_x[17:49, 33] = 0
      H_x[17:49, 65] = 0
      H_y[17, 33:65] = 0
      H_y[49, 33:65] = 0

    # Update Dirichlet BCs for magnetic fields
    H_x[0,:] = 0
    H_x[80,:] = 0
    H_y[:,0] = 0
    H_y[:,96] = 0

    # TM: Update electric field at time step t+1
    diff_H_x = Z * dT / dy * (H_x[1:, :] - H_x[:-1, :])
    diff_H_y = Z * dT / dx * (H_y[:, 1:] - H_y[:, :-1])
    E_z = E_z + (diff_H_y - diff_H_x)

    if (obstructed):
      # Update BCs for electric field on conducting square
      E_z[17:49, 33:65] = 0
      E_z[17:49, 33:65] = 0

    # Update Dirichlet BCs for electric field
    E_z[0,:] = 0
    E_z[79,:] = 0
    E_z[:,0] = 0
    E_z[:,95] = 0

    # add incident wave
    for i in range(Nx - 1):
      if (0 <= (i - 50*alpha + c*t) and (i - 50*alpha + c*t <= 8*alpha)):
        source[i, :] = np.sin(np.pi*(i - 50*alpha + c*t)/(8*alpha))
```

FIG. 5. Code for the 2-D FDTD simulation.

V. PHYSICAL EXAMPLES

In this section, I will explicate two examples of boundary-value problems that can demonstrate the FDTD Method.

In both subsections, I will begin by illustrating the physical example in words and diagrams. Then, I will be detailing the physical constraints and boundary conditions that each scenario places on all relevant observables.

A. The One-Dimensional Conducting Line

The first example I hope to discuss will be a simpler case of a one-dimensional conductor reflecting an incident EM wave. We are treating the simple case of reflection at two points along a line, with conducting points providing Dirichlet boundary conditions at x-coordinates 17 and 49.

B. The Two-Dimensional Conducting Square.

The second of these will be a more complex, though not arduous, example. Specifically, I will be covering the example given in Yee's paper: the two-dimensional, conducting square.

This square provides boundary conditions along the x-coordinate range of 17 to 49, and along the y-coordinate range of 33 to 65. Fig. 6 depicts the geometry of this situation well.

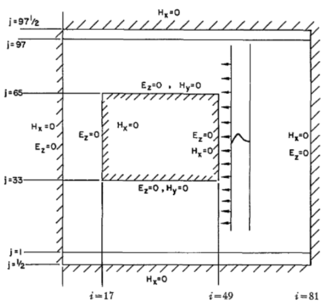


FIG. 6. Visual depiction of 2-D conducting square. This figure was taken from Ref. [1].

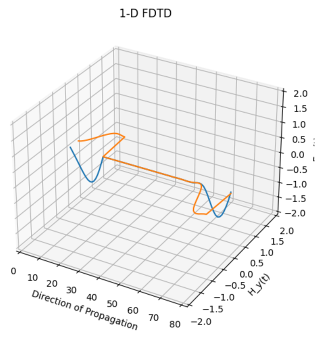


FIG. 7. Results from the one-dimensional simulation coded in Python.

VI. RESULTS

I will be highlighting the results of my coded simulations here, making sure to include graphs that can be compared to the original data in Yee's paper.

For the one-dimensional FDTD method, Fig. 7 is specifically a snapshot of an animation we created from the simulation, with the E_z and H_y values graphed against the direction of wave propagation.

For the two-dimensional simulations, we graphed the amplitude of the z-component of the E-field at different time-steps (described by n). As Yee did, we are holding specific y-values fixed to observe the wave propagation given differing interactions with different parts of the square. The results are captured in Fig. 8.

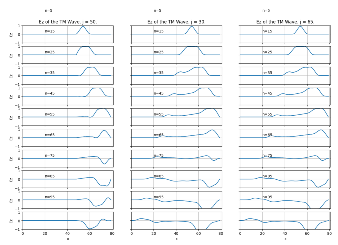


FIG. 8. Results from the case of the two-dimensional simulation wherein the conducting square was present, coded in Python.

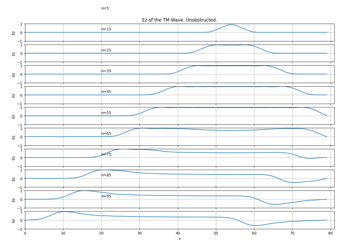


FIG. 9. Results from the unobstructed case of the two-dimensional simulation coded in Python.

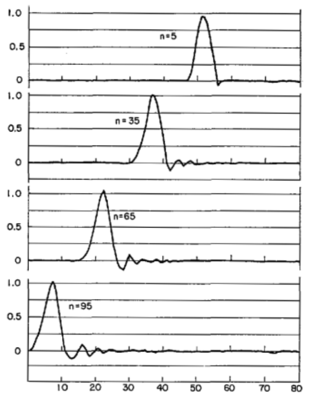


FIG. 10. Yee's results for the unobstructed wave simulation. Taken from Ref. [1].

We also chose to create a control simulation where the conducting square is not present so as to see how the FDTD method calculates the propagation of a wave in empty space. The results are captured in Fig. 9.

VII. COMPARISON OF RESULTS

Yee's original results are pictured in the following figures:

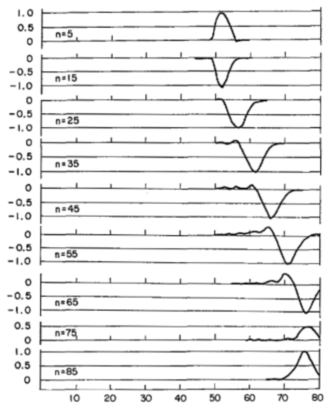


FIG. 11. Yee's results for the wave simulation at y=50. Taken from Ref. [1].

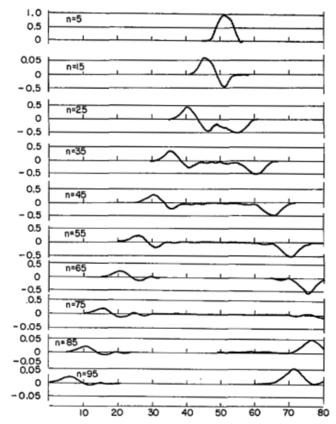


FIG. 12. Yee's results for the wave simulation at y=30. Taken from Ref. [1].

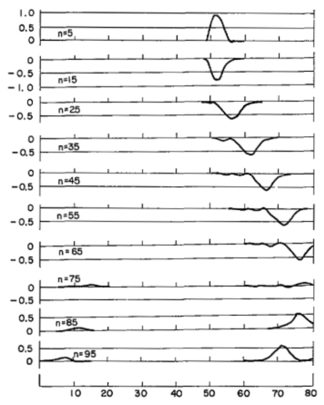


FIG. 13. Yee's results for the wave simulation at y=65. Taken from Ref. [1].

The most apparent differences come from the unobstructed case, seen in Fig. 10. For an unknown reason, it looks as if the wave in my simulation, unlike in Yee's propagated outwards in both directions. In Yee's original unobstructed figure, the wave propagates to the left exclusively.

The most general difference comes from the obstructed cases, seen in Figures 11, 12, and 13. In these cases, Yee's incident wave propagates leftward at the start of the simulations, being in the three instances either totally or partially reflected by the conducting boundary of the square. When it is reflected, its amplitude is inverted and it heads back in the opposite direction (i.e. to the right).

The distinction with my results is that, unlike in Yee's results, my wave propagates to the right, reflects off of the far boundary of the overall range of the simulation (dictated by Dirichlet conditions), with possibly a part of the wave propagating to the left at the start. This is unusual, as it causes a distinction in what is in Yee's case a left-moving wave, and in our case a mostly-right moving wave with some movement to the left.

I am unsure as to why this is the case, but I imagine that the FDTD I programmed has issues with the incident wave source, possible distinguishing itself from the source that Yee used.

I did try to replicate its behavior as much as possible, including using the original definition as Yee describes in his paper, but this was to little avail. The closer I made the source pulse to the definition given by Yee, the less clear the simulation's relation to Yee's became.

In future work, this would be a point to revise: to find a correct incident wave source that matches Yee's results more correctly.

CODE AVAILABILITY

Code is available at `https://github.com/liamkarl/PHYS-408-Term-Project`.

ACKNOWLEDGMENTS

I greatly appreciate the guidance and understanding of Dr. Abram throughout the semester.

[1] K. Yee, Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media, IEEE Transactions on Antennas and Propagation **14**, 302 (1966).

[2] J. B. Schneider, Understanding the Finite-Difference Time-Domain Method (2010).

[3] The Finite-Difference Time-Domain Method (FDTD).

[4] A. Yew, Numerical differentiation: finite differences (2011).

[5] G. Ohner, Two Dimensional Finite Difference Time Domain Computation of Electromagnetic Fields in Python (2018).

[6] H. Zhao, S. Crozier, and F. Liu, A high definition, finite difference time domain method, Applied Mathematical Modelling **27**, 409 (2003).

[7] L. N. Trefethen, Finite difference and spectral methods for ordinary and partial differential equations (1996) Chap. 4.