EM Fields and Waves I Group Project

Omar Elsayed 202100597
Aser Osama 202101266
Saifelden Mohamed 202100432
Mahmoud Mohamed 202100678

January 8, 2024

Communications and Information Engineering
Zewail City of Science and Technology

January 8, 2024

Introduction

The study of electromagnetic waves and their behavior is a crucial part of modern wireless communication systems. This behavior can be articulated using mathematics and simulated using software. We will briefly cover the derivation of the electromagnetic wave equations and their solution in one and two dimensions as well as go over a brief user manual for the Matlab Simulation Software we have created.

EM Wave Equation

Derivation

The vector wave equation can be derived by combining Maxwell's equations. Starting with Maxwell's equations:

$$\text{Gauss's Law for Electricity:}\quad \nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0} \tag{1}$$

$$\text{Gauss's Law for Magnetism:}\quad \nabla \cdot \mathbf{B} = 0 \tag{2}$$

$$\text{Faraday's Law of Induction:}\quad \nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{3}$$

$$\text{Ampère's Law with Maxwell's Addition:}\quad \nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t} \tag{4}$$

Here, $\mathbf{E}$ is the electric field, $\mathbf{B}$ is the magnetic field, $\mathbf{H}$ is the magnetic field auxiliary, $\mathbf{D}$ is the electric displacement field, ρ is the charge density, ε_0 is the permittivity of free space, and $\mathbf{J}$ is the current density.

Now, let's derive the vector wave equation. Start by substituting Faraday's Law into Ampère's Law:

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t} \tag{5}$$

Using the constitutive relation $\mathbf{D} = \varepsilon \mathbf{E}$, where ε is the permittivity, we have:

$$\nabla \times \mathbf{H} = \mathbf{J} + \varepsilon \frac{\partial \mathbf{E}}{\partial t} \tag{6}$$

Now, apply the curl identity $\nabla \times (\nabla \times \mathbf{A}) = \nabla(\nabla \cdot \mathbf{A}) - \nabla^2 \mathbf{A}$ to the left-hand side:

$$\nabla(\nabla \cdot \mathbf{H}) - \nabla^2 \mathbf{H} = \mathbf{J} + \varepsilon \frac{\partial \mathbf{E}}{\partial t} \tag{7}$$

Assuming the medium is linear and isotropic, we can use $\mathbf{H}$ as $\mathbf{H} = \frac{1}{\mu}\mathbf{B}$, where μ is the permeability. Substituting this into the equation and applying $\nabla \cdot \mathbf{B} = 0$, we get:

$$\nabla^2 \mathbf{E} - \mu\varepsilon \frac{\partial^2 \mathbf{E}}{\partial t^2} = \nabla(\nabla \cdot \mathbf{H}) - \frac{1}{\mu}\nabla^2 \mathbf{B} \tag{8}$$

Now, recognizing that $\nabla(\nabla \cdot \mathbf{H}) - \frac{1}{\mu}\nabla^2 \mathbf{B} = -\mu\varepsilon \frac{\partial^2 \mathbf{B}}{\partial t^2}$ (from Faraday's Law), we can simplify the equation to obtain the vector wave equation:

$$\nabla^2 \mathbf{E} - \mu\varepsilon \frac{\partial^2 \mathbf{E}}{\partial t^2} = -\mu\varepsilon \frac{\partial^2 \mathbf{B}}{\partial t^2} \tag{9}$$

This is the vector wave equation describing the propagation of electromagnetic waves in a linear and isotropic medium.

In a 1D system and two coordinates (1 spatial dimension x and 1 time dimension t), the wave equation above is simplified into the normal 1D standard wave equation:

$$\frac{\partial^2 u}{\partial t^2} = C^2 \frac{\partial^2 u}{\partial x^2} \tag{10}$$

As the $u(x,t)$ function is synonymous with the Electric field $\mathbf{E}$ in (x,t), and c^2 is equivalent to $1/\mu\varepsilon$.

In a 2D-Wave equation, the function u would be synonymous with $\mathbf{E}(x,y,t)$, therefore when applying the Laplacian, the following equation would be presented:

$$\frac{\partial^2 u}{\partial t^2} = C^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{11}$$

The boundary conditions of the upper Partial Differential Equation (PDE) are declared as Dirichlet conditions. In the case of wave interference by reflection at the boundary, the end points would have Dirichlet boundary conditions given by:

$$\delta(0,y,t) = 0 \quad \text{and} \quad \delta(L_x,y,t) = 0 \tag{12}$$

$$\delta(x,0,t) = 0 \quad \text{and} \quad \delta(0,L_y,t) = 0 \tag{13}$$

Here, δ represents the wave function, and $(0,y,t)$, (L_x,y,t), $(x,0,t)$, and $(0,L_y,t)$ are the coordinates at the boundaries of the square plate.

Define the permeability (μ) and permittivity (ε) needed for our medium. We may test the simulation in the following cases and observe simulation differences:

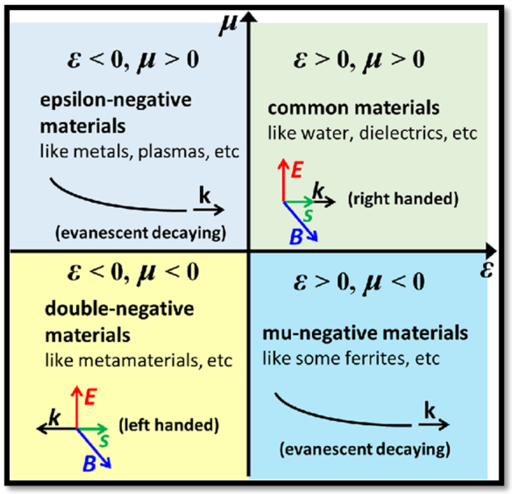


Figure 1: Electric permittivity and Magnetic permeability materials spectrum.

Define the medium of propagation as a square plane of side length (a), and we would experiment with different side lengths:

- Trial 1: $a = 0.1$ m
- Trial 2: $a = 1$ m
- Trial 3: $a = 10$ m

Once the Electric permittivity and Magnetic permeability constants, as well as the square dimensions, are all set, we start the simulation and change the frequency to its harmonics to observe constant nodal points upon interference of incident and reflected waves at the boundary of the square plate. We spot these frequencies on the Electromagnetic spectrum and investigate if these values change when we alter the materials.

PDE Discretization

Consider the one-dimensional electromagnetic wave equation:

$$\frac{\partial^2 E}{\partial t^2} - c^2 \frac{\partial^2 E}{\partial x^2} = 0 \tag{14}$$

We discretize this equation using a finite difference scheme. Let E_i^n represent the numerical solution at spatial point x_i and time level n. The finite difference scheme yields:

$$\frac{E_i^{n+1} - 2E_i^n + E_i^{n-1}}{\Delta t^2} - c^2 \frac{E_{i+1}^n - 2E_i^n + E_{i-1}^n}{\Delta x^2} = 0 \tag{15}$$

Rearranging, we get:

$$E_i^{n+1} = 2E_i^n - E_i^{n-1} + \frac{c^2 \Delta t^2}{\Delta x^2}(E_{i+1}^n - 2E_i^n + E_{i-1}^n) \tag{16}$$

This discretized equation represents the one-dimensional electromagnetic wave equation in discrete form. Numerical solutions can be obtained by iterating through time steps.

Now, let's express the discretized electromagnetic wave equation in matrix-vector form and explore a numerical example.

$$E_i^{n+1} = 2E_i^n - E_i^{n-1} + \frac{c^2 \Delta t^2}{\Delta x^2}(E_{i+1}^n - 2E_i^n + E_{i-1}^n) \tag{17}$$

We can rewrite this equation in a matrix-vector form as follows:

$$\mathbf{E}^{n+1} = \mathbf{A}\mathbf{E}^n - \mathbf{E}^{n-1} \tag{18}$$

where:

1. $\mathbf{E}^{n+1}$ is the column vector representing the numerical solution at the next time step,
2. $\mathbf{E}^n$ is the column vector representing the numerical solution at the current time step,
3. $\mathbf{E}^{n-1}$ is the column vector representing the numerical solution at the previous time step,

4. **A** is the tridiagonal matrix representing the coefficients of the discretized equation.

   For a one-dimensional spatial domain with N grid points, the matrix **A** is defined as:

$$\mathbf{A} = \begin{bmatrix} 2+\frac{c^2\Delta t^2}{\Delta x^2} & -1 & 0 & \cdots & 0 \\ -1 & 2+\frac{c^2\Delta t^2}{\Delta x^2} & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 2+\frac{c^2\Delta t^2}{\Delta x^2} & -1 \end{bmatrix}$$

The vectors $\mathbf{E}^{n+1}$, $\mathbf{E}^{n}$, and $\mathbf{E}^{n-1}$ contain the numerical solution values at each spatial grid point.

This system of linear equations may be solved efficiently using an algorithm, and the resulting solution provides an approximation to the original electromagnetic wave equation. Similar equation is used for two dimensions.

CFL condition

The Courant-Friedrichs-Lewy (CFL) condition is a numerical stability criterion commonly used in finite difference methods for solving partial differential equations (PDEs). It ensures that the time step size is appropriately chosen to capture the fastest wave or signal in the system.

Mathematically, the CFL condition is given by:

$$\text{CFL} = \frac{c \cdot \Delta t}{h} \leq 1$$

The condition involves three key variables:

- **Wave Propagation Speed (c):** Higher c makes the CFL condition more restrictive, necessitating smaller time steps and grid spacings to capture fast-moving waves accurately.
- **Time Step Size (Δt):** Smaller Δt relaxes the CFL condition, allowing for less frequent updates, but a balance must be struck to avoid instability.
- **Grid Spacing (h):** Finer grid spacing (smaller h) increases CFL restrictiveness, demanding smaller time steps for stability, especially with high wave speeds.

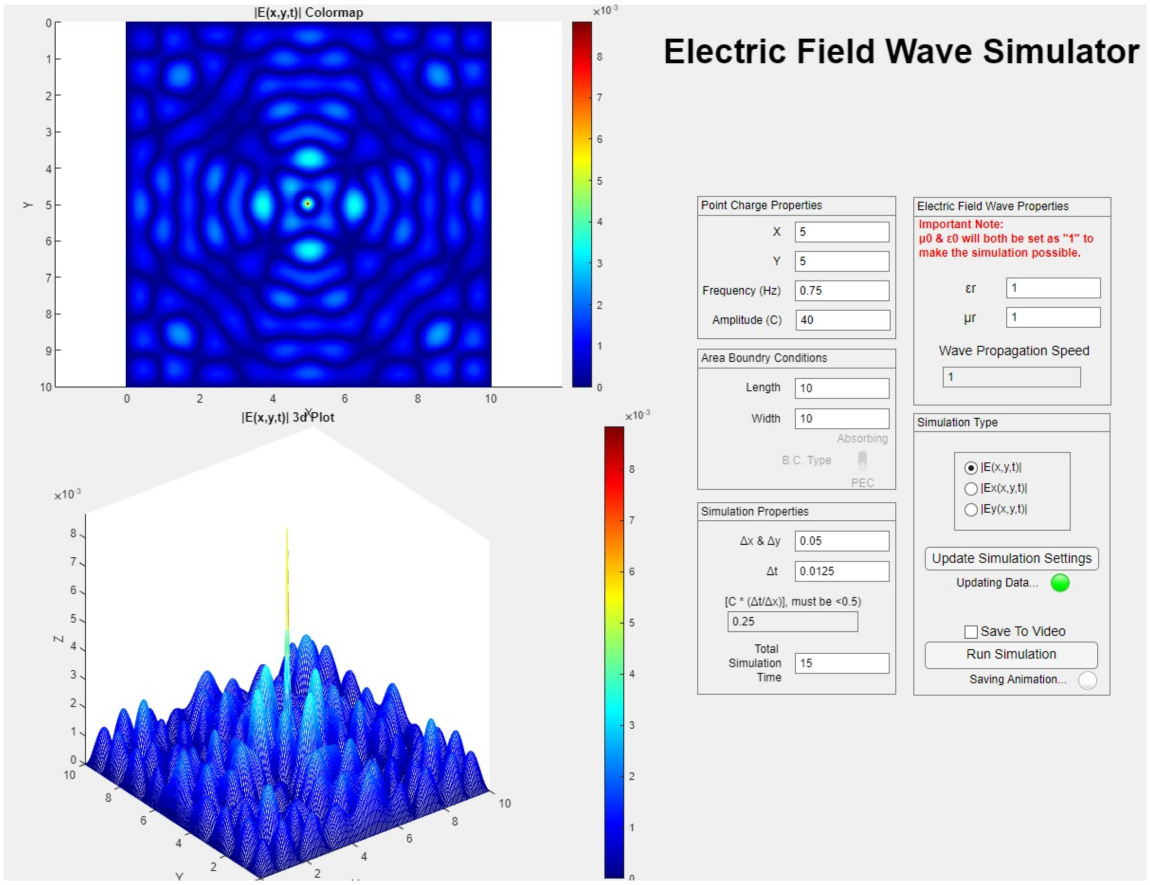
The CFL condition guarantees stability in numerical simulations, preventing issues such as oscillations or divergence in the solution. When the CFL condition is satisfied, the time step is small enough to prevent the loss of important details in the simulation. Just as in signal processing where undersampling can lead to distorted signals, violating the CFL condition in numerical simulations can lead to inaccurate representation of wave propagation. In our case, the condition will be adjusted to

$$\text{CFL} = \frac{c \cdot \Delta t}{h} \leq 0.5$$

in order to satisfy the condition for two dimensions

Program User Manual: Electromagnetic Interference Simulation

Welcome to the Electromagnetic Interference Simulation program! This MATLAB GUI simulates electromagnetic interference using partial differential equation (PDE) discretization. This user manual provides a detailed guide on using and understanding the program.



Point Charge Properties

X	5
Y	5
Frequency (Hz)	0.75
Amplitude (C)	40

Area Boundry Conditions

Length	10
Width	10
B.C. Type	Absorbing / PEC

Simulation Properties

Δx & Δy	0.05
Δt	0.0125
[C * (Δt/Δx)], must be <0.5)	0.25
Total Simulation Time	15

Electric Field Wave Properties

Important Note:
μ0 & ε0 will both be set as "1" to make the simulation possible.

εr	1
μr	1
Wave Propagation Speed	1

Simulation Type

● |E(x,y,t)|
○ |Ex(x,y,t)|
○ |Ey(x,y,t)|

Updating Data... ●

☐ Save To Video

Saving Animation... ○

Figure 2: Screenshot of the Electromagnetic Interference Simulation Program

Overview of Graph Panels

The program interface consists of five panels, each serving a specific purpose:

Point Charge Properties Panel

This panel deals with the properties of the point charge, representing the initial condition of the simulation. It includes modifications for:

- Amplitude
- Frequency
- Position (X, Y) of the point charge.

Area Boundary Condition Panel

This panel defines the shape of the simulation region. Parameters include:

- Length (X-axis)
- Width (Y-axis).

Simulation Properties Panel

This panel contains modifiable variables that influence the simulation display:

- Duration of the simulation
- Number of points simulated.

Electric Field Properties Panel

This panel covers constants influencing the electric field waves:

- μ_r: Relative permeability
- ϵ_r: Relative permittivity.
- their product is C, which is the speed of light

Simulation Type Panel

This panel determines the type of simulation to be performed. You can select:

- Magnitude of E
- Magnitude of E_x
- Magnitude of E_y.

Graphs

The program generates two types of graphs for visualization:

- Magnitude of E Colormap - This graph displays the magnitude of the electric field using a colormap.
- 3D Plot - This graph provides a 3D plot representing the electromagnetic interference simulation.

Getting Started

To run the simulation, follow these steps:

1. Set the parameters in each panel according to your preferences.
2. Click the `Update Simulation Settings` button to apply changes.
3. Choose the visualization mode using the radio buttons in the `Simulation Type` panel.
4. If saving to video is desired, select `Save as Video`.
5. Click the `Run Simulation` button to start the simulation.
6. Wait until the simulation completes, and observe the results.

Important Code Snippets

The generation of E_x field (similar to E_y)

```

            for tIndex= 1 : app.numSamplesTime - 1
                app.U_current(1,:) = 0;
                app.U_current(end,:) = 0;

                % Saving N+1 and N-1 Values
                app.U_minus1 = app.U_current;
                app.U_current = app.U_plus1;

                % Value Due to Source
                app.U_current(app.Point_x, app.Point_y) = SoureOutput(
                    tIndex);

                for xIndex= 2 : app.numSamplesX- 1
                    for yIndex= 2 : app.numSamplesY- 1
                        app.U_plus1(xIndex, yIndex) = ...
                            2 * app.U_current (xIndex, yIndex) - ...
                            app.U_minus1(xIndex, yIndex) + ...
                            app.CFL ^ 2 * ...
                            ( app.U_current(xIndex+1, yIndex) + app.
                                U_current(xIndex, yIndex+1) + ...
                            app.U_current(xIndex-1, yIndex) + app.U_current
                                (xIndex, yIndex-1) - ...
                            4 * app.U_current(xIndex, yIndex) );
                    end
                end

                app.E_X(:,:,tIndex) = app.U_current;
            end
```

generation of the total E_{total}

```

            for tIndex= 1 : app.numSamplesTime - 1
                app.E_t(:,:,tIndex) = (app.E_Y(:,:,tIndex) .^ 2 + app.E_X
                    (:,:,tIndex) .^ 2) .^ 0.5;
            end
```

Notes

- Once the simulation starts, it cannot be stopped until completion.
- Video recording may impact the simulation speed.

Disclaimer

This program is provided as-is. The user is responsible for understanding and modifying the code for specific needs.

References

[1] Huang, H. (n.d.). Numerical methods for PDE (two quick examples). Retrieved from Arizona State University: `https://www.public.asu.edu/~hhuang38/pde.pdf`