



Simulation and implementation of a Conformal Finite Difference Time Domain method

Carlos Julio Ramos Salas



Simulation and implementation of a Conformal Finite Difference Time Domain method

Carlos Julio Ramos Salas

Memoria del **Trabajo Fin de Máster**.
Máster en Física y Matemáticas (FisyMat)
University of Granada (UGR).

Tutored by:

Dr. Luis Manuel Díaz Angulo
Dr. Kenan Tekbaş



UNIVERSIDAD
DE GRANADA

Declaración de originalidad de Trabajo Fin de máster Máster en Física y Matemáticas (FisyMat)

Carlos Julio Ramos Salas, con DNI n.º **E26185963**/Pasaporte n.º **168201985**, autor del trabajo de fin máster con el título *Simulation and implementation of a Conformal Finite Difference in Time Domain method*, declara la originalidad de este trabajo presentado para su defensa y evaluación en el Máster en Física y Matemáticas en el curso 2023/24, en el que se han respetado los derechos de otros autores a ser citados.

Granada, 28 de junio de 2024

Fdo: 

Contents

Acknowledgments	1
English Abstract	2
Resumen en Español	3
1 Introduction and objectives	4
2 Maxwell’s equations and the FDTD method	6
2.1 Introduction to the finite differences	6
2.2 One-dimensional discrete Maxwell equations	7
2.2.1 Boundary conditions	8
2.3 Two-dimensional discrete Maxwell equations	9
2.4 Stability in the FDTD method	10
3 Conformal extension of the FDTD method	11
3.1 Introduction to CFDTD	11
3.2 Stability in the CFDTD method	13
3.3 CFDTD meshing	14

4	CFDTD implementation and results	15
4.1	One-dimensional CFDTD	15
4.2	Two-dimensional mesh creation	19
4.3	Two-dimensional CFDTD algorithm	21
5	Conclusions	27
6	References	29

Acknowledgments

I would like to thank the "Fundación Carolina", this would be impossible without the opportunity they gave to me. Thanks to my best friends Mario Sánchez and Ana Mejía for their unconditional company. I would also like to thank my tutor Luis Díaz for guiding me in this area of research and welcoming me to this country.

Finally and most importantly, I would like to thank my parents Margot Salas and Julio Ramos, I am the person I am today thanks them.

English Abstract

Some differential equations in the literature present arduous work to find the associated solution, even in some cases, the solution to said systems turn out to be impossible to find through analytical methods. In this situation, numerical methods plays an important role since they allow us to solve the system of interest through discrete operations with a low numerical error involved.

Among the numerous existing techniques to solve electromagnetism problems, the Finite Difference in Time Domain method (FDTD) stands out, however, when we consider complicated geometries, it is necessary to refine the method in search of better efficiency, that is where we can introduce the Conformal Finite Difference in Time Domain method (CFDTD), which can be studied as the modification of the FDTD by introducing a Perfect Electric Conductor (PEC) volume into the geometry to consider.

In this work, a simulation and implementation of the CFDTD method is made in both one and two dimensions, in the last one, considering a line or an area of PEC that interrupts the spatial mesh worked. The codes worked out were prepared with test-oriented development in the python language, these can be found in the associated GitHub repository presented in Chapter 4.

Resumen en Español

Algunas ecuaciones diferenciales en la literatura presentan un trabajo arduo para encontrar la solución asociada, incluso en algunos casos, la solución a dicho sistema resulta ser imposible de encontrar a través de métodos analíticos. Ante esta situación los métodos numéricos juegan un papel importante ya que nos permiten resolver el sistema de interés a través de operaciones discretas con un bajo error numérico de por medio.

Entre las diversas técnicas existentes para poder resolver problemas de electromagnetismo destaca el método de diferencias finitas en el dominio del tiempo (FDTD por sus siglas en inglés), sin embargo, al momento de considerar geometrías complicadas, es necesario refinar el método en búsqueda de una mayor eficiencia, allí es donde se puede introducir la técnica conforme de diferencias finitas (CFDTD), la cual puede ser estudiada como la modificación de FDTD al introducir un volumen de conductor eléctrico perfecto (PEC) en la geometría a considerar.

En el presente trabajo se realiza una simulación e implementación del método CFDTD tanto en una como en dos dimensiones, en este último caso, considerando una línea o un área de PEC que interrumpen en el mallado. Los códigos trabajados fueron realizados con desarrollo orientado por tests en el lenguaje python, estos pueden encontrar en el repositorio de GitHub asociado presentado en el capítulo 4.

1 | Introduction and objectives

Mathematical models in science and engineering are usually differential or integral equations, which in many occasions are difficult to find analytical solutions or may even lack them. With the rapid development of computers with high processing speed, attempts have been made to use them to find solutions to these mathematical models in order to optimize and reduce the time spent to solve them. However, a computer is unable to find the exact solution to all mathematical models and in order to approximate the solution it is necessary to implement a numerical method [1].

The most important equations in electromagnetism are the well-known Maxwell's equations, which depending on the geometry of the problem to be solved can also be very complicated to find a solution, for this reason, many numerical methods were developed in order to be applied to this case. In particular, if we focus on the curl equations, the *Finite Difference in Time Domain* method (FDTD) stands out [2], this method has as advantages that the resulting matrices when linearizing the problem are sparse, so it is a fairly optimal method and its stability condition is quite well studied. However, this method has problems when trying to solve very complex geometries that require a spatial mesh with a very small distance between nodes, because in these cases, as the temporal step is restricted by the spatial one, the execution time increases considerably. Additionally, since the FDTD method uses the creation of meshes with orthogonal coordinates to each other for the solution of the maxwell equations, this results in the cells having rectangular shapes. For this reason, when trying to use the method on curved surfaces, an inaccuracy error also appears on these objects [3], due to the inability to adapt most of the curved surfaces to rectangles.

In order to be able to extend the FDTD method when you want to study curved surfaces but still use a rectangular mesh, the *Conformal Finite Difference in Time Domain* method (CFDTD) was introduced by Dey and Mittra [4] as a first approximation, in this method, it is considered that the curved surface is composed of a *Perfect Electric*

Conductor (PEC) and the iterative equation for the evolution of the magnetic field is modified to be able to solve in all the cells of the discretization. This method has the great advantage of eliminating staircasing errors that arise when making inaccurate approximations of the geometry [3] without having to change the simple mesh used by the FDTD method at the cost of having to restrict the time step of the simulation to guarantee numerical stability.

The main objective of this work is the creation and validation of a CFDTD algorithm in both one and two dimensions. For this, it is also necessary in the case of two dimensions to create and construct a meshing algorithm that allows us to recover all the information on the geometry of the problem that the numerical method may later require. Finally, to validate what was done, comparisons will be made both with the FDTD algorithm and with the analytical solution of some problems related to resonance in cavities.

In the next chapters, we will explore the main themes of this work. We will begin with a review of the FDTD method in both one and two dimensions along with its respective stability condition. In the next chapter, we will review the CFDTD method along with the Dey-Mitra algorithm and its modified stability condition. Finally, the implementations carried out will be shown along with the results obtained.

2 | Maxwell's equations and the FDTD method

Let's first start by introducing the Maxwell's equation of electromagnetism and the basic notions of the FDTD algorithm in one and two dimensions in the free space case.

2.1 Introduction to the finite differences

We want to find a function that is the solution to a specific differential equation, however, this is a hard problem in general and only rarely can an analytic formula be found for the solution. A finite difference method proceeds by replacing the derivatives in the differential equation with finite differences approximations [5, 6]. For example, let's consider the Taylor approximation for $f(x + h)$ and $f(x - h)$

$$f(x + h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \mathcal{O}(h^3) = f(x) + hf'(x) + \mathcal{O}(h^2), \quad (2.1)$$

$$f(x - h) = f(x) - hf'(x) + \frac{h^2}{2}f''(x) + \mathcal{O}(h^3) = f(x) - hf'(x) + \mathcal{O}(h^2), \quad (2.2)$$

in both equations it is possible to isolate the derivative, then we obtain:

$$f'(x) = \frac{f(x + h) - f(x - h)}{2h} + \mathcal{O}(h), \quad (2.3)$$

$$f'(x) = \frac{f(x) - f(x - h)}{h} + \mathcal{O}(h). \quad (2.4)$$

If we ignore the order h terms, we obtain the first order approximation for the derivative of the function with an error proportional to h . However, if we want to improve and reduce the error to order h^2 , it's necessary to introduce the central finite difference approximation. If we consider $h = \Delta x/2$ and subtract the equations 2.1 and 2.2

we can obtain the central finite difference as it follows

$$f'(x) = \frac{f\left(x + \frac{\Delta x}{2}\right) - f\left(x - \frac{\Delta x}{2}\right)}{\Delta x} + \mathcal{O}(\Delta x^2), \quad (2.5)$$

we obtain then the approximation searched by ignoring the quadratic order terms. Since the error decreases faster in this case for smaller Δx , the equation will be more efficient to work with, for this reason, this approximation will be used for the discretization of the Maxwell's equations.

2.2 One-dimensional discrete Maxwell equations

First let's consider the time-dependent Maxwell's curl equations for vacuum [7, 8]

$$\frac{\partial \mathbf{E}}{\partial t} = \frac{1}{\epsilon_0} \nabla \times \mathbf{H}, \quad (2.6)$$

$$\frac{\partial \mathbf{H}}{\partial t} = -\frac{1}{\mu_0} \nabla \times \mathbf{E}, \quad (2.7)$$

where \mathbf{E} and \mathbf{H} are the electric and magnetic fields with three dimensions, with all the components being functions that depend of the spatial coordinates; and ϵ_0 and μ_0 are the permittivity and permeability of vacuum. For the one-dimensional case we can assume that the only non zero components of \mathbf{E} and \mathbf{H} are E_x and H_y respectively, then, equations 2.6 and 2.7 become

$$\frac{\partial E_x}{\partial t} = -\frac{1}{\epsilon_0} \frac{\partial H_y}{\partial z}, \quad (2.8)$$

$$\frac{\partial H_y}{\partial t} = -\frac{1}{\mu_0} \frac{\partial E_x}{\partial z}. \quad (2.9)$$

These equations represents a plane wave traveling through the z direction. Taking the central difference approximation discussed above for both the temporal and spatial derivatives we obtain [9]

$$\frac{E_x^{n+\frac{1}{2}}(k) - E_x^{n-\frac{1}{2}}(k)}{\Delta t} = -\frac{1}{\epsilon_0} \frac{H_y^n(k + \frac{1}{2}) - H_y^n(k - \frac{1}{2})}{\Delta x}, \quad (2.10)$$

$$\frac{H_y^{n+1}(k + \frac{1}{2}) - H_y^n(k + \frac{1}{2})}{\Delta t} = -\frac{1}{\mu_0} \frac{E_x^{n+\frac{1}{2}}(k + 1) - E_x^{n+\frac{1}{2}}(k)}{\Delta x}. \quad (2.11)$$

In these two equations, the time step is represented by the superscripts (n) while the argument inside functions represent the spatial step (k), so the current time and distance are given by $t = n\Delta t$ and $z = k\Delta x$. Finally, we can rearrange the last equations to obtain the next iterative equations

$$E_x^{n+1/2}(k) = E_x^{n-1/2}(k) - \frac{\Delta t}{\epsilon_0 \Delta x} \left[H_y^n \left(k + \frac{1}{2} \right) - H_y^n \left(k - \frac{1}{2} \right) \right], \quad (2.12)$$

$$H_y^{n+1} \left(k + \frac{1}{2} \right) = H_y^n \left(k + \frac{1}{2} \right) - \frac{\Delta t}{\mu_0 \Delta x} [E_x^{n+1/2}(k+1) - E_x^{n+1/2}(k)]. \quad (2.13)$$

It's important to notice that this formulation assume that the electric and magnetic fields are interleaved in both space and time, this is illustrated in the Figure 2.1.

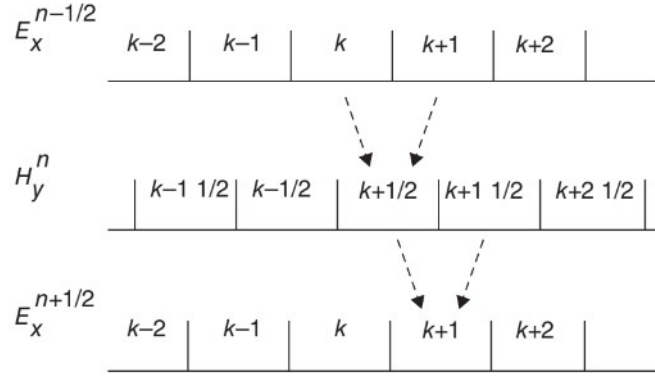


Figure 2.1: Interleaving of the electric and magnetic fields in the FDTD formulation. Image taken from [9].

2.2.1 Boundary conditions

We have observed that to evolve the field at a node, we require the neighboring nodes of the other field. However, some nodes only have one neighboring magnetic field in the discretization, making it impossible to apply the iterative formula of the FDTD algorithm for these nodes. To compute the electric field at these positions, it is necessary to introduce boundary conditions. For example, the perfect electric conductor boundary condition states that the electric field at these nodes is zero, causing the electromagnetic wave to "bounce" when interacting with the edges of the mesh. Another possible solution is the periodic boundary condition, which is characterized as

follows

$$E_x^{n+1/2}(0) = E_x^{n-1/2}(0) - \frac{\Delta t}{\epsilon_0 \Delta x} \left[H_y^n \left(\frac{1}{2} \right) - H_y^n \left(N - \frac{1}{2} \right) \right], \quad (2.14)$$

where N is the number of cells in the mesh and $E_x(N) = E_x(0)$ for all the time steps. A different example of boundary conditions is the Mur's absorption condition, which allows to nullify the fields arriving at the boundary so that there are no bounces. Similarly, there is also the Perfectly Matched Layer (PML) condition, which is also an absorption condition used in scattering problems in order to make it easier to distinguish between the incident wave and the one reflected by the material.

2.3 Two-dimensional discrete Maxwell equations

In two-dimensional problems, the third dimension is invariant [10], for this reason, it's convenient to separate the fields in two groups and only work with one of those; the first one is the transversal magnetic mode (TM), which is composed of H_x , H_y and E_z , and the other group is the transversal electric mode (TE), composed of E_x , E_y and H_z . In this work we only consider the TE mode.

Considering again the time-dependent Maxwell's curl equations for the free space and by introducing the TE mode, we can obtain

$$\frac{\partial H_z}{\partial t} = -\frac{1}{\mu_0} \left(\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} \right), \quad (2.15)$$

$$\frac{\partial E_x}{\partial t} = \frac{1}{\epsilon_0} \frac{\partial H_z}{\partial y}, \quad (2.16)$$

$$\frac{\partial E_y}{\partial t} = -\frac{1}{\epsilon_0} \frac{\partial H_z}{\partial x}. \quad (2.17)$$

We can again use the central difference approximation but now with two spatial steps Δx and Δy to obtain the next iterative equation for the magnetic field

$$\begin{aligned} H_z^{n+1} \left(i + \frac{1}{2}, j + \frac{1}{2} \right) &= H_z^n \left(i + \frac{1}{2}, j + \frac{1}{2} \right) - \frac{\Delta t}{\mu_0 \Delta x} (E_y^{n+1/2}(i+1, j) - E_y^{n+1/2}(i, j)) \\ &\quad + \frac{\Delta t}{\mu_0 \Delta y} (E_x^{n+1/2}(i, j+1) - E_x^{n+1/2}(i, j)), \end{aligned} \quad (2.18)$$

and the next ones for the electric fields

$$E_x^{n+1}(i, j) = E_x^n(i, j) + \frac{\Delta t}{\epsilon_0 \Delta y} \left[H_z^{n+1/2} \left(i, j + \frac{1}{2} \right) - H_z^{n+1/2} \left(i, j - \frac{1}{2} \right) \right], \quad (2.19)$$

$$E_y^{n+1}(i, j) = E_y^n(i, j) + \frac{\Delta t}{\epsilon_0 \Delta x} \left[H_z^{n+1/2} \left(i + \frac{1}{2}, j \right) - H_z^{n+1/2} \left(i - \frac{1}{2}, j \right) \right]. \quad (2.20)$$

As can be seen, the electric field maintains essentially the same iterative formula in one as in two dimensions, while the iterative formula for the magnetic field now requires more arguments.

2.4 Stability in the FDTD method

We have seen that the central difference approximation used converges to the analytic solution with quadratic order, but in order to guarantee the convergence of the numerical solution to the analytical one, it is required to impose certain restrictions on the time and spatial steps.

First, let's start with the restriction of the time step Δt . The maximum value this parameter can have is determined by the *Courant-Friedrichs-Lewy (CFL) condition*. The physical meaning of this condition is that the electromagnetic wave must not travel through more than one cell in a single time step [11]. This condition makes mathematical sense because the central difference approximation only considers the nearest neighbors to estimate the evolution of the cell. In general, for a 3D rectangular grid, we have

$$c\Delta t \leq \left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2} + \frac{1}{(\Delta z)^2} \right)^{-1/2} \equiv d, \quad (2.21)$$

where c represents the speed of the electromagnetic wave (speed of light in the free space case). We can convert the inequality into an equality by multiplying by a constant k less than 1 on the right side, so we finally have

$$\Delta t = \frac{kd}{c}. \quad (2.22)$$

Finally, the spatial step can not be selected at random either. The fundamental restriction is that the cell size must be smaller than the smallest wavelength of the electromagnetic wave [11]. A frequently used rule states that we need to have at least 10 cells per wavelength.

3 | Conformal extension of the FDTD method

Using the FDTD method with rectangular grid to analyze objects with curved metallic surfaces not only introduces errors due to inaccurate approximation of the geometry [12]. If we want to extend the FDTD method to solve curved surfaces or irregularities composed of perfect electric conductor, it's necessary to introduce the CFDTD method. In this chapter we introduce the CFDTD basics.

3.1 Introduction to CFDTD

The CFDTD algorithm was first introduced to make a FDTD analysis of a curved two-dimensional PEC using a locally conformal grid [4]. For two-dimensional problems we can consider, for example, a quadrant of the cross section of a circular resonator shown in Figure 3.1. We can see that the magnetic fields over the cells that don't intersect with the curve can be solved with the regular FDTD method. However, we can't use the same algorithm for the other cells because some neighboring nodes of the electric field discretization are no longer able to interact due to the presence of the PEC surface between them. For this reason, we need to introduce another formulation for the FDTD that can be generalized and used for the CFDTD case.

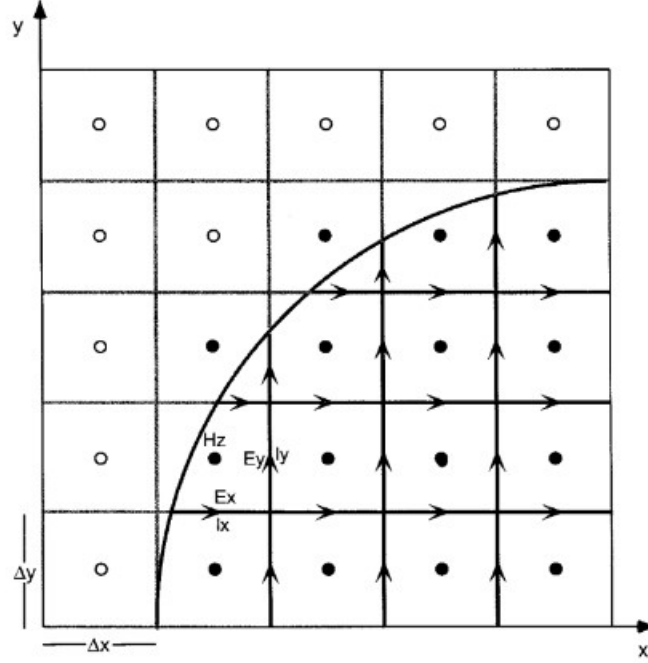


Figure 3.1: Example of a rectangular mesh on a curved PEC surface. Image taken from [3].

Let's consider a two-dimensional rectangular grid discretization same as the FDTD case. Now, let's consider the Faraday equation in the integral form for the free space given by

$$\oint_{\partial S} \mathbf{E} \cdot d\mathbf{l} = -\frac{d}{dt} \int_S \mu_0 \mathbf{H} \cdot d\mathbf{s}. \quad (3.1)$$

If we assume that the electric fields are constant along the edges between the nodes, that the magnetic fields are also constant in the worked cell and if we are working on the TE mode, we can obtain from the equation 3.1 the following relation

$$\begin{aligned} -\mu_0 A(i, j) \frac{\partial}{\partial t} H_z \left(i + \frac{1}{2}, j + \frac{1}{2} \right) = & E_x(i, j) l_x(i, j) - E_x(i, j + 1) l_x(i, j + 1) \\ & + E_y(i + 1, j) l_y(i + 1, j) - E_y(i, j) l_y(i, j), \end{aligned} \quad (3.2)$$

where $l_x(i, j)$ and $l_y(i, j)$ represents the length of the edge where is assumed the electric node $E_x(i, j)$ and $E_y(i, j)$ respectively, and $A(i, j)$ the area where the magnetic field $H_z(i + 1/2, j + 1/2)$ is placed. We also assumed that the area is constant in the time so it can separated from the time derivative. Finally we take the central difference

approximation for the time derivative and rearranging we obtain

$$\begin{aligned}
 H_z^{n+1} \left(i + \frac{1}{2}, j + \frac{1}{2} \right) &= H_z^n \left(i + \frac{1}{2}, j + \frac{1}{2} \right) \\
 &- \frac{\Delta t}{\mu_0 A(i, j)} \left[E_x^{n+1/2}(i, j) l_x(i, j) - E_x^{n+1/2}(i, j+1) l_x(i, j+1) \right. \\
 &\quad \left. + E_y^{n+1/2}(i+1, j) l_y(i+1, j) - E_y^{n+1/2}(i, j) l_y(i, j) \right].
 \end{aligned} \tag{3.3}$$

In the uniform rectangular grid case, if there is no curved surface in the mesh, we have $l_x = \Delta x$, $l_y = \Delta y$ and $A = \Delta x \Delta y$ for any pair i, j , then, by replacing these values into the equation 3.3, we can obtain again the magnetic field iterative formula presented in the equation 2.18.

The great advantage of this formulation is that we can use it to solve curved PEC surfaces with rectangular grids, as shown in Figure 3.1. Indeed, as mentioned before, the cells that don't intersect with the curved surface can be solved using the standard method. On the other hand, in cells with non-empty intersections, we need to consider another component in the path integral of the electric field corresponding to the value of this field over the part of the curved surface inside the cell. However, since the field over this path is always zero, the construction of the iterative formula for the magnetic field remains valid. It must be noted, though, that the lengths and areas in these cells do not have the usual values. This formulation is known as the Dey-Mitra algorithm in two dimensions [4].

3.2 Stability in the CFDTD method

As we have seen, the CFDTD method follows a different scheme than the FDTD method in the cells where the curved PEC surface intersects the mesh, since the area and the lengths of some cells change. We need to find a stability condition that depends on these variables, then, we need a more restrictive time step stability condition. For example, the condition reported in [13] establishes that

$$\Delta t_k = \sqrt{3} \sqrt{\frac{A_k^{\text{ratio}}}{\max(l_k^{\text{ratio}})}} \Delta t_{\text{CFL}} = h_{0,k} \Delta t_{\text{CFL}}, \tag{3.4}$$

with $h_{0,k} = \sqrt{3} \sqrt{A_k^{\text{ratio}} / \max(l_k^{\text{ratio}})} \leq 1$. In this equation k represents one of the possible remaining areas when we consider the cell cut by the curve, A^{ratio} represent

the quotient between the remaining area from the usual area of the cell and l^{ratio} the quotient between the cut length of the edge from the usual length (considering all the four possible lengths per cell). This time step is similar to the time step reported in [14], the only difference is the factor $\sqrt{3}$, however, both of these criteria are made for 3D problems. In case we want to work with 2D problems, we need to change the previously factor $\sqrt{2}$.

Returning to the equation 3.4, there is a problem with this formulation. To guarantee the stability of the scheme, we need to look over all the cells of the discretization, making the algorithm inefficient or slow for large problems with short spatial step or problems with complex geometries that require very detailed refinement of the mesh. For this reason, the study of local stability conditions that require only specific information about the problem has been a research objective for both CFDTD and FDTD [15].

3.3 CFDTD meshing

Although the numerical method of the CFDTD method is described by the iterative equations given by the Dey-Mitra algorithm 3.3, an important part of the formula are the length and area variables that can be extracted from the mesh for a specific cell.

Several researches have been done on meshing techniques for the CFDTD method as can be seen in [16] and [17]. In general, an appropriate meshing is very important because it is directly related to the simulation accuracy of the numerical method. Additionally, it is important to optimize the creation or interpretation of the mesh to work with, since in the real world, some objects to be studied, such as airplanes, require a very high number of cells or patches.

Finally, as we could observe in the adapted stability condition in equation 3.4. We see that it is necessary to access the mesh before the numerical method in order to guarantee the stability of the latter, and given that in some cases it might be necessary to access many cells to guarantee the correct choice of the time step. This shows again the importance of creating an efficient and well adapted mesh so that the algorithm is also optimal.

4 | CFDTD implementation and results

In this section, we discuss the CFDTD method implementation along some results that we compare with the literature. All developed codes are publicly available and can be found in the GitHub repository with the following url <https://github.com/ashybabashyba/CFDTD>.

4.1 One-dimensional CFDTD

Let's start with the formalism in one dimension. In this case, it is impossible to consider a curved PEC surface because of the dimensions of the mesh. Before thinking about the iterative formula, it is necessary to see that the equivalent of the curved surface in this case is equal to a PEC sheet between two nodes of the mesh as we can see in the Figure 4.1. Following this, we can separate the nodes of the mesh in two groups, the first one being those nodes n_i in such a way that the PEC sheet it's not contained in the intervals (n_{i-1}, n_i) or (n_i, n_{i+1}) ; the second group being those nodes that are "neighbors" of the PEC sheet.

Example of an electric field mesh



Figure 4.1: Example of an uniform mesh for the electric field in one dimension (blue lines) with a PEC sheet between two nodes (red line).

In the first group of nodes (the non-frontier nodes), for both the electric and magnetic fields, we can again use the usual FDTD iterative formula because the distance between neighboring nodes remains equal to Δx . However, for the neighboring nodes of the PEC sheet, we need to change the iterative equations due to the presence of the irregularity. To formulate a possible solution, we can treat the position of the PEC sheet as another node in the electric field mesh, adding one degree of freedom to the magnetic and electric field solution (keeping in mind that the electric field must be zero at all times at this extra node). With this in mind, we can use linear interpolation to formulate a valid iterative formula. If k_p represents the PEC sheet node in the electric field mesh, we have the following equations

$$H_y^{n+1} \left(k_p - \frac{1}{2} \right) = H_y^n \left(k_p - \frac{1}{2} \right) + \frac{\Delta t}{\mu_0 \Delta x_l} E_x^{n+1/2}(k_p - 1), \quad (4.1)$$

$$H_y^{n+1} \left(k_p + \frac{1}{2} \right) = H_y^n \left(k_p + \frac{1}{2} \right) - \frac{\Delta t}{\mu_0 \Delta x_r} E_x^{n+1/2}(k_p + 1), \quad (4.2)$$

$$E_x^{n+1/2}(k_p - 1) = E_x^{n-1/2}(k_p - 1) - \frac{\Delta t}{\epsilon_0 \Delta x_l} \left[H_y^n \left(k_p + \frac{1}{2} \right) - H_y^n \left(k_p - \frac{1}{2} \right) \right], \quad (4.3)$$

$$E_x^{n+1/2}(k_p + 1) = E_x^{n-1/2}(k_p + 1) - \frac{\Delta t}{\epsilon_0 \Delta x_r} \left[H_y^n \left(k_p + \frac{3}{2} \right) - H_y^n \left(k_p + \frac{1}{2} \right) \right]. \quad (4.4)$$

Where $\Delta x_l + \Delta x_r = \Delta x$ represents the existing distance between the PEC sheet node to the left and right node respectively. With these equations already formulated, we can start with the study and validation of our implementation; the electromagnetic wave to study is a Gaussian pulse defined by the initial pulse

$$E_0(k_0) = \Delta t \exp \left[-0.5 \left(\frac{t_0 - n\Delta t}{s} \right)^2 \right], \quad (4.5)$$

$$H_0(k_0) = \Delta t \exp \left[-0.5 \left(\frac{t_0 - n\Delta t - \frac{\Delta t}{2} - \frac{\Delta x}{2c}}{s} \right)^2 \right], \quad (4.6)$$

where s is the spread of the wave, t_0 the initial time and n the current step of the simulation; in order to simplify the algorithm, we also work with natural units ($c = 1$). The python script used to recreate the CFDTD algorithm in one dimension and the tests associated can be found in the attached GitHub repository in the files `cf dtd/CFDTD1D.py` and `test/test_1D.py` respectively. The first result that is expected to be observed is

the instability of the solution with some Courant numbers (the constant k in the equation 2.22) when the PEC sheet node doesn't match with any other node of the mesh. Indeed, we can consider a mesh of length equals to 200 m and spatial step $\Delta x = 1.0$ m and a PEC sheet positioned at 150.5 m; if the behavior of the CFDTD algorithm were the same as that of the FDTD, we expect the solution of the fields to be stable at all time steps. However, this doesn't happen as we can see in the Figure 4.2.

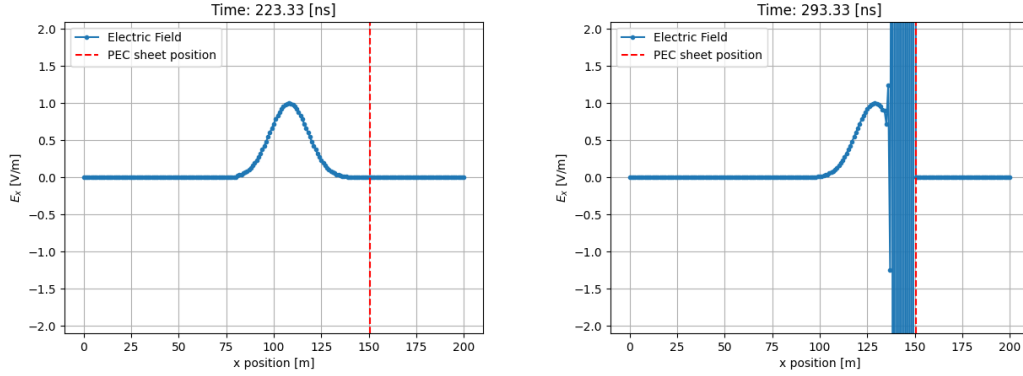


Figure 4.2: Propagation of the Gaussian electromagnetic wave with PEC boundary conditions and Courant number equals to 1.0 in different time steps.

We can also make a comparison between a FDTD solution with PEC boundary conditions and a CFDTD solution with also PEC boundary conditions but with the PEC sheet position located 0.5 m before the right end of the mesh. We expect to see a delay of the non conformal solution with respect to the conformal solution just after bouncing the PEC sheet, this delay should increase depending on the number of times the wave has interacted with the PEC surface, as seen in the Figure 4.3. In particular, we can corroborate that the conformal solution after two bounces is the same as the non conformal one but moved two nodes as can be found in the respective test.

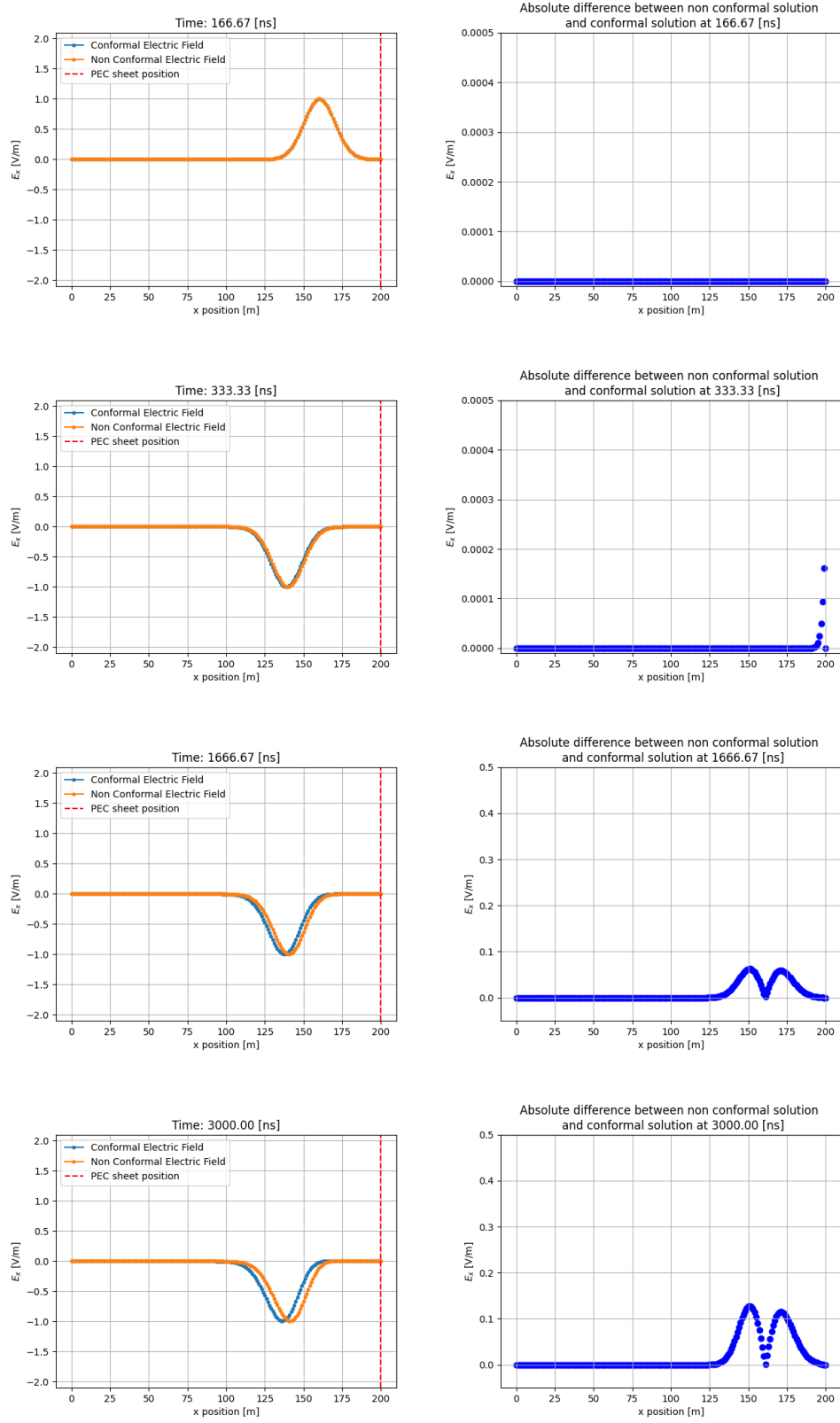


Figure 4.3: Comparison between conformal and non conformal solutions of the Gaussian electromagnetic wave in different time steps. On the left is the evolution of the wave using both FDTD and CFDTD, on the right is the absolute difference between both solutions.

4.2 Two-dimensional mesh creation

Now, unlike the previous case, the CFDTD algorithm in more than one dimension becomes more complicated because there is more freedom in the election of the PEC surface inside the mesh. For this and other reasons, it is necessary to carry out a study on the geometry of the problem before entering the CFDTD-2D algorithm.

Returning to the formulation of the Dey-Mittra algorithm, the difference between the CFDTD and FDTD method lies in the areas and lengths of the cells, so for example, if the curved surface intersects one cell, we need the coordinates of the intersection points to estimate the value of the respective length. For this reason, the coordinates of all existing intersections play an important role in the development of the algorithm.

To simplify the problem worked on, we restrict the PEC surface to be a line or a polygon that separates a maximum of one cell into two cells. The intersection points of a cell must be zero or two depending on whether the intersection between the cell and the PEC polygon is empty or not. An example of the 2D mesh with a PEC polygon or PEC line is presented in the Figure 4.4. With this in mind, if we work with a PEC line we can separate the cells in two groups depending on the number of intersection points: the conformal cells with a number of intersections equal to 2 and the non conformal cells with empty intersection. In a similar way, for the PEC polygon we can separate the cells in three groups: the conformal cells and the outside or inside non conformal cells, the latter depending on whether they are inside or outside the PEC polygon.

In order to keep the mesh as a simple object and not to increase its mathematical complexity, instead of using a `meshgrid` or `array` function [18], we propose another way of working with the mesh using polygons and the python library `shapely` [19]. Instead of working with all the nodes of the 2D `meshgrid` with all the intersection points, we characterize the mesh with a total of $(N_x - 1) \times (N_y - 1)$ (where N_x and N_y are the number of nodes in the 1D spatial discretization for both spatial steps) polygons that represent all the cells contained in the mesh and another polygon used to represent the PEC surface. In this way, it is possible to calculate the lengths and areas of each cell using Boolean operations.

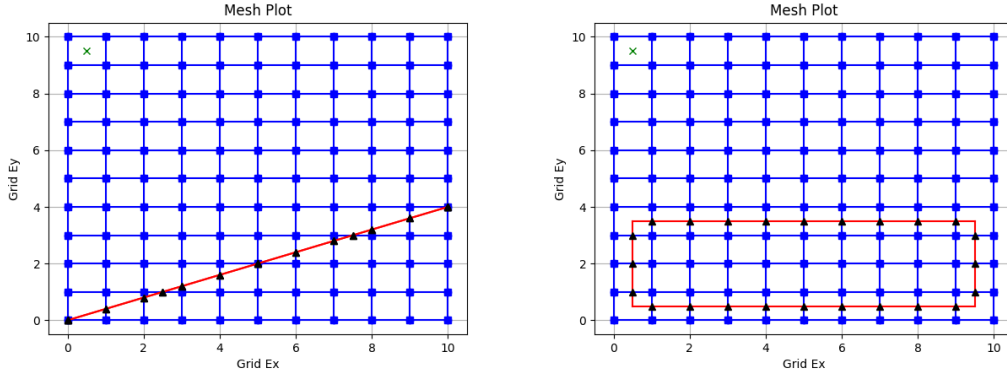


Figure 4.4: Two-dimensional mesh (blue dots) considering a PEC line (left) or a PEC polygon (right) represented by the red line, the intersection points are identified as the black triangular marker, the green cross marker represents a reference cell for the initial wave.

The mesh creation algorithm can be found in the file `mesh/MESH2D.py`, this python class includes several routines for creating and accessing the mesh information. It takes as input arguments the size of the box, the spatial step Δx and Δy , a cell reference that has the purpose of determining whether we solve inside or outside the PEC surface and the nodes that characterize the PEC polygon enclosed inside the mesh. The first of these routines corresponds to the detection of the number of unique intersection points between the cell and the PEC polygon through the interpretation of the polygon as a set of lines. The next routine corresponds to the calculation of area using the Boolean difference between the cell and the PEC polygon (or its complement if we want to solve inside the surface). Finally, there is a routine that allows measuring the lengths of the cells by means of intersections of sets.

To finish, a separation is made so that the mesh knows whether to interpret the lengths or areas depending on whether the wave to be solved is inside or outside the PEC polygon (the option of solving both regions at the same time is excluded) using the previous three routines. To guarantee the correct functioning of the algorithm, various tests are carried out such as: that the groups in which the cells are separated are mutually exclusive and the sum of its elements is equal to the total number of cells, that the area of non conformal cells are equal to $\Delta x \Delta y$, that the lengths of non conformal cells are equal to Δx or Δy and that the sum of lengths of the two cells into which a conformal cell is divided also meets the same previous equality, etc; all of these tests can be found in the file `test/test_Mesh_2D.py`.

The great advantage of the constructed algorithm is its ease of being adapted to any curved PEC surface; it is only necessary to introduce as an argument a list of points corresponding to the intersection points of the curve with the mesh for the construction of the PEC polygon. The difference between working with the curved surface and the polygon is that the areas of the cell can be different, however, for a sufficiently dense mesh, this difference is minimal.

4.3 Two-dimensional CFDTD algorithm

Once the mesh is built, adapting it to the Dey-Mitra algorithm is relatively easy, it is only necessary to pass the mesh as an argument and when performing the time steps of the electromagnetic fields, call the area and length calculation routine defined in the mesh script for the worked cell.

For the validation of the method, we consider three different possible initial pulses for the electromagnetic wave, these being the two-dimensional Gaussian, the Gaussian restricted to one dimension and the resonant rectangular cavity in the mode 44; all of these can be found in the python script `initial_pulse/INITIAL_PULSE2D.py`, this class needs as arguments the spread of the wave, the mesh worked, the same reference cell mentioned in the previous subsection and the type of the pulse to work. Once we have defined our mesh and initial pulse, we can start our solver which needs as arguments the two previously mentioned classes and the Courant number. The solver class can be found in the file `cfddt/CFDTD2D.py`.

We begin our validation, seeing at the solution obtained from a Gaussian magnetic wave that depends only on the x coordinate as follows

$$H_{0z}(x, y) = \exp \left[-\frac{(x - x_0)^2}{\sqrt{2}s} \right], \quad (4.7)$$

where x_0 is the first coordinate of the reference cell and s is the spread of the wave. We work with a square mesh of dimensions $100 \text{ m} \times 100 \text{ m}$ and spatial steps $\Delta x = \Delta y = 1.0 \text{ m}$, inside this mesh we put a rectangular PEC polygon defined by the nodes $(5.5, 40.5)$, $(94.5, 40.5)$, $(94.5, 60.5)$ and $(5.5, 60.5)$, we take $(50, 50)$ as the reference cell and $s = 20$; then, we put the initial condition defined in equation 4.7 inside the PEC polygon and start the solver with a Courant number equals to 0.3. With these parameters, we expect the evolution of the wave to have a similar behavior as if it were in one dimension, i.e., it only propagates in one direction. The result of our simulation is represented in figure 4.5 where the expected behavior is confirmed.

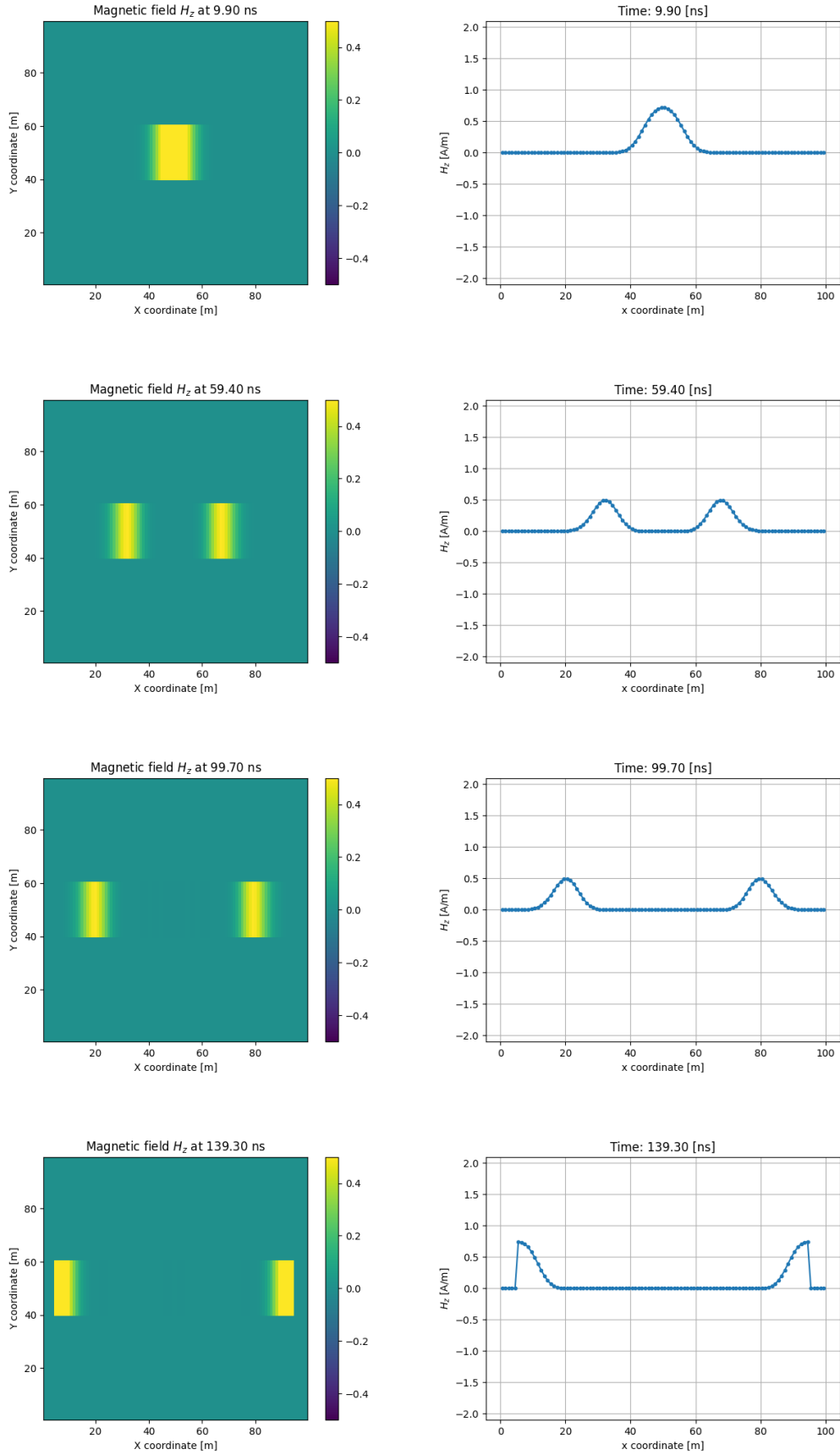


Figure 4.5: Evolution of a Gaussian pulse defined by equation 4.7 inside a rectangular PEC polygon with CFDTD. On the left is the evolution of the 2D mesh, on the right the evolution for a fixed y -coordinate.

Next we want to study the stability of a initial Gaussian magnetic wave defined by

$$H_{0z}(x, y) = \exp \left[-\frac{(x - x_0)^2 + (y - y_0)^2}{\sqrt{2}s} \right], \quad (4.8)$$

if we work with a mesh similar to the previous one used but now with $\Delta x = \Delta y = 1.0$ m, and a rectangular PEC polygon defined by the nodes (25.5, 10.5), (75.5, 10.5), (75.5, 95.5) and (25.5, 95.5). In this case, we have two types of conformal cells, the cells that have $A^{\text{ratio}} = 1/4$ and $\max(l^{\text{ratio}}) = 1/2$ (the corner conformal cells); and the cells that have $A^{\text{ratio}} = 1/2$ and $\max(l^{\text{ratio}}) = 1$, following the equation 3.4, we observe in both cases that $\Delta t = \Delta t_{\text{CFL}}$, so we expect not to observe any instability for any Courant number less or equal to 1 as seen in the Figure 4.7. On the other hand, if we replace 10.5 with 10.6 and 95.5 with 95.4 in the nodes of the PEC polygon, we obtain $A^{\text{ratio}} = 1/5$ and $\max(l^{\text{ratio}}) = 1/2$ for the corner conformal cells, then, for this cells we have $\Delta t \neq \Delta t_{\text{CFL}}$ and we expect to have numerical instability for Courant Number Equal to 1 as shown in Figure 4.6.

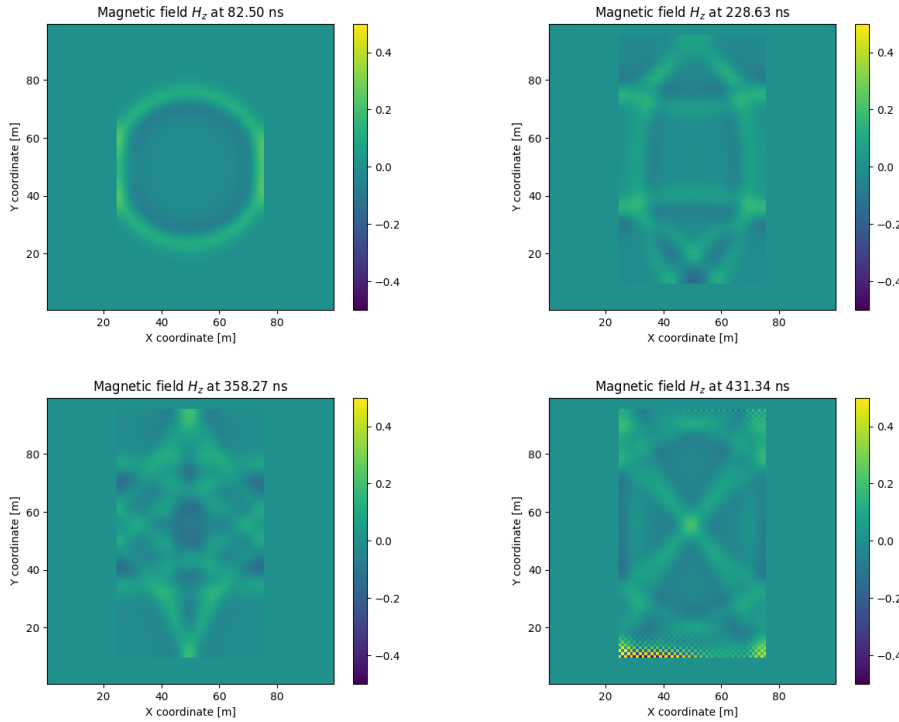


Figure 4.6: Evolution of a Gaussian 2D magnetic pulse defined by equation 4.8 inside a rectangular PEC polygon with $\Delta t \neq \Delta t_{\text{CFL}}$ and Courant number equal to 1.0. Instability starts to appear around 130 [s].

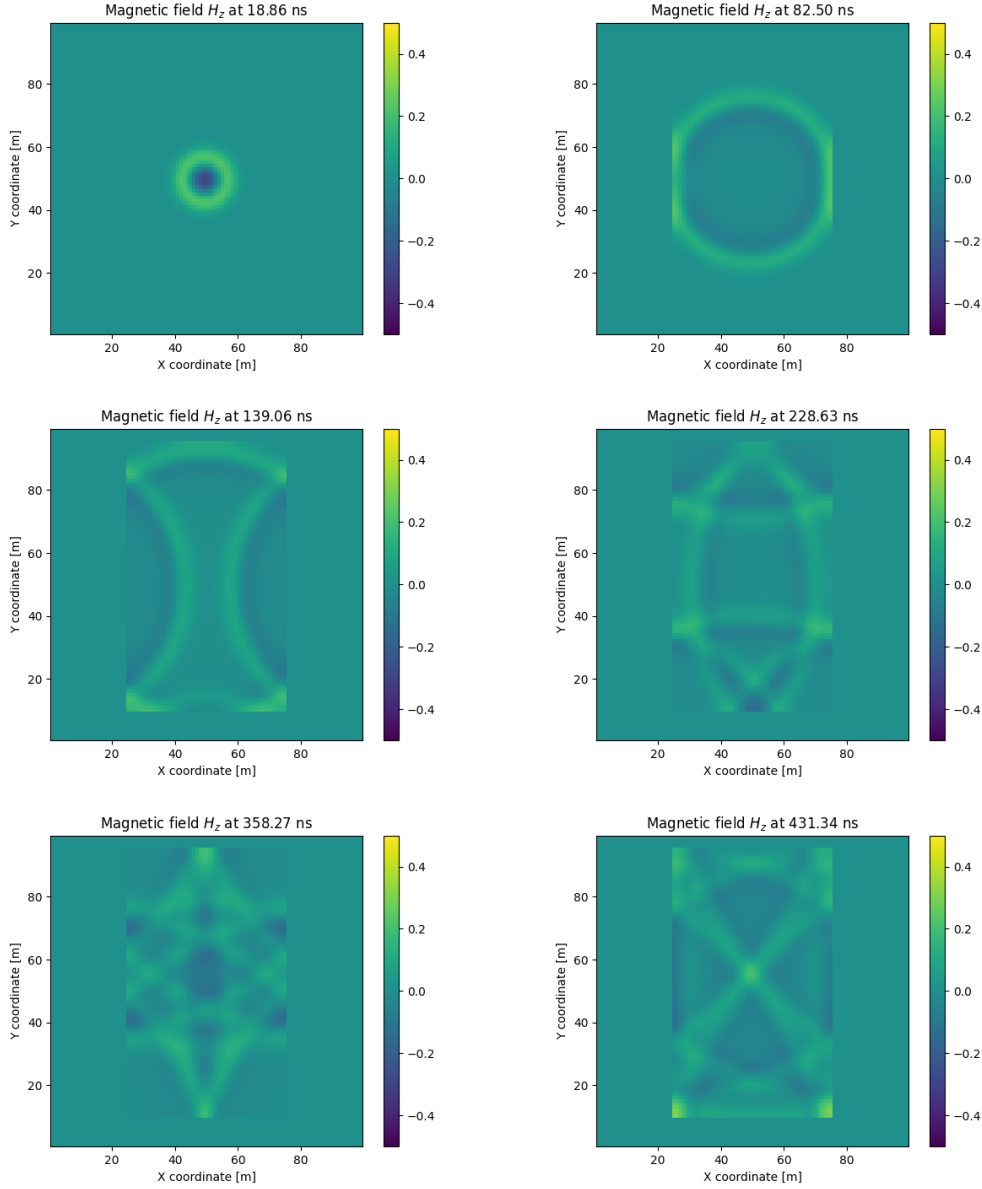


Figure 4.7: Evolution of a Gaussian 2D magnetic pulse defined by equation 4.8 inside a rectangular PEC polygon with $\Delta t = \Delta t_{\text{CFL}}$ and Courant number equal to 1.0.

Finally, we can also study the evolution of an electromagnetic wave inside a rectangular resonant cavity whose initial condition for the TE mode is given by [20]

$$H_z(x, y) = A_{mn} \cos \frac{m\pi(x - x_0)}{a} \cos \frac{n\pi(y - y_0)}{b}, \quad (4.9)$$

where a and b are the lengths of the rectangular box, (x_0, y_0) is the lower left coordinate of the box, m, n are positive integers and A_{mn} is the amplitude of the wave. For the simulation, we again construct a mesh equal to the case of the 2D Gaussian, and a rectangular PEC polygon defined by the nodes $(10.5, 10.5)$, $(95.5, 10.5)$, $(95.5, 95.5)$ and $(10.5, 95.5)$, we again define the initial wave inside the PEC polygon. In particular, the wave we simulate is the mode $m = n = 4$, we expect to observe a resonance and periodic behavior in all the electromagnetic fields, and in fact, it is shown in the Figures 4.8 and 4.9. In addition, the period of oscillation of the electromagnetic wave can be calculated through a routine and this can be compared to the theoretical period which is obtained as the inverse of the resonant frequency given by [20]

$$f_{mn} = \frac{c}{2\pi\sqrt{\mu_r\epsilon_r}} \sqrt{\left(\frac{m\pi}{a}\right)^2 + \left(\frac{n\pi}{b}\right)^2}, \quad (4.10)$$

where ϵ_r and μ_r are the relative permittivity and permeability respectively, the comparison is presented in the Table 4.1 for various modes and using both CFDTD and FDTD method, it can be observed that as the values of m and n increase, the period of oscillation decreases as expected, although the relative error between the simulated and the theoretical values also increases. It can also be observed that in this case, since the PEC box has integer dimensions, the period results between both methods are practically the same, however, if the cavity were to have rational dimensions, the periods would be slightly different since the box could not be adjusted to the FDTD mesh.

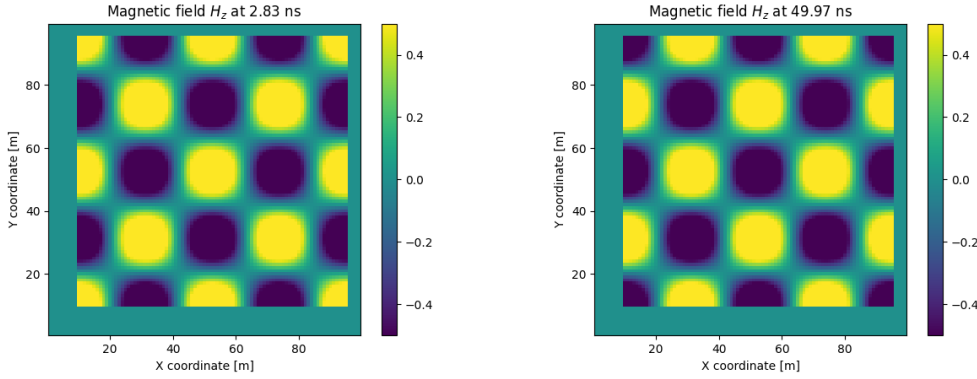


Figure 4.8: Evolution of the magnetic wave defined by the equation 4.9 inside a square PEC using CFDTD. The evolution of the wave consists of transitions between the left state and the right state.

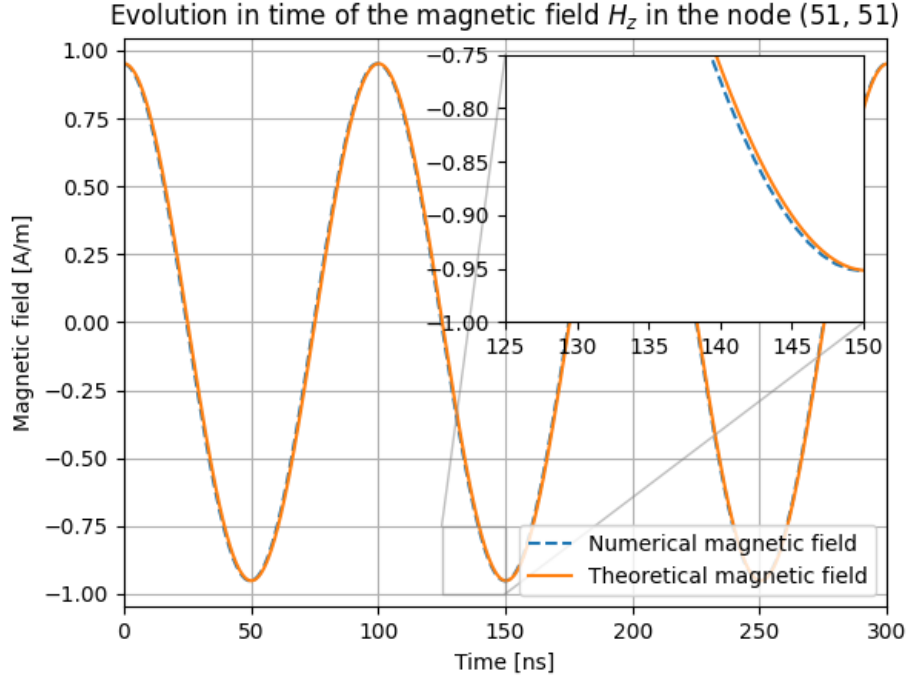


Figure 4.9: Comparison between the numerical solution of the magnetic field and the analytical solution in a specific node.

Table 4.1: Comparison between the numerical and theoretical period of diverse modes of a square resonant cavity defined by the nodes (10.5, 10.5), (95.5, 10.5), (95.5, 95.5) and (10.5, 95.5) for the CFDTD method and (11, 11), (96, 11), (96, 96) and (11, 96) for the FDTD method, both using a Courant number equal to 0.4.

Mode	CFDTD period [ns]	FDTD period [ns]	Analytical period [ns]	Relative error %
(1, 1)	399.751	399.751	400.694	0.235
(2, 2)	199.876	199.876	200.347	0.235
(3, 3)	132.936	132.936	133.565	0.471
(4, 4)	98.995	98.995	100.173	1.176

5 | Conclusions

In this work, a review and implementation of conformal finite differences in the time domain in both one and two dimensions was carried out, proposing for the first case an iterative formula based on an interpolation. It is verified that the conformal solution of a Gaussian presents a delay with respect to the non conformal solution when the PEC sheet is 0.5 units before the end of the mesh, finding in particular that after two bounces, the conformal wave is practically equal to the non conformal solution but with a two time steps difference.

For the two-dimensional case, a Python class was built for meshing where instead of interpreting the grid by nodes, it is interpreted through polygons in order to reduce the complexity of the problem for problems with large geometries. In order to guarantee the correct operation of the routines of this class, several tests were developed to check and verify the proper behavior in particular cases.

The Dey-Mitra algorithm was implemented in order to adapt the two-dimensional solver to solve problems requiring CFDTD, the evolution of a "1.5 "D Gaussian pulse was studied to observe behaviors similar to the evolution of a 1D Gaussian wave. The stability behavior according to the PEC polygon using a 2D Gaussian wave was observed, seeing that in the case where the conformal cells are divided into two groups characterized by $A^{\text{ratio}} = 1/4$ and $\max(l^{\text{ratio}}) = 1/2$, and $A^{\text{ratio}} = 1/2$ and $\max(l^{\text{ratio}}) = 1$, the FDTD stability condition is recovered. Finally, the evolution of a wave inside a rectangular resonant cavity is studied, observing for the TE mode the expected periodicity effects for all fields, it is found that the numerical period differs a maximum of 2% with respect to the theoretical period.

Once the CFDTD method has been implemented, it is possible to use this code in order to extend it to various applications where the FDTD algorithm lacked precision due to the presence of curved geometries. It is possible to perform a local stability study taking advantage of the tests developed to guarantee the correct operation of the

numerical method, in this way, the mathematical results can be easily verified through information extracted from the defined classes. Finally, a possible future work with respect to the codes could be the implementation of the algorithm to three dimensions, where instead of working with polygons in the mesh, it would be necessary to work with polyhedrons.

6 | References

1. Johnson, C. *Numerical Solution of Partial Differential Equations by the Finite Element Method* 1st ed. (Cambridge University Press, Cambridge, 1987).
2. Taflove, A. & Hagness, S. *Computational Electrodynamics: The Finite-Difference Time-Domain Method* 3rd (Artech House, Boston, MA, 2005).
3. Dey, S. & Mittra, R. A locally conformal finite-difference time-domain (FDTD) algorithm for modeling three-dimensional perfectly conducting objects. *IEEE Microwave and Guided Wave Letters* **7**, 273–275 (1997).
4. Dey, S., Mittra, R. & Chebolu, S. A technique for implementing the FDTD algorithm on a nonorthogonal grid. *Microwave and Optical Technology Letters* **14**, 213–215 (1997).
5. LeVeque, R. J. *Finite Difference Methods for Ordinary and Partial Differential Equations* (Society for Industrial and Applied Mathematics, 2007).
6. Burden, R. L. *Numerical analysis / Richard L. Burden, J. Douglas Faires, Annette M. Burden*. Tenth edition. eng. ISBN: 9781305253667 (Cengage Learning, 2016).
7. Jackson, J. D. *Classical electrodynamics* (John Wiley & Sons, 1999).
8. Griffiths, D. J. *Introduction to electrodynamics* (Pearson, 2013).
9. Sullivan, D. M. & Houle, J. E. *Electromagnetic Simulation Using the FDTD Method with Python, 3rd Edition* 224. ISBN: 978-1-119-56580-2 (Wiley-IEEE Press, 2020).
10. Peterson, A., Ray, S. L. & Mittra, R. *Computational methods for electromagnetics* (Wiley-IEEE Press, 1997).
11. Kunz, K. S. & Luebbers, R. J. *The Finite Difference Time Domain Method for Electromagnetics* 1st. ISBN: 9780849386576 (CRC Press, Boca Raton, FL, 1993).

12. Cangellaris, A. & Wright, D. Analysis of the numerical error caused by the stair-stepped approximation of a conducting boundary in FDTD simulations of electromagnetic phenomena. *IEEE Transactions on Antennas and Propagation* **39**, 1518–1525 (1991).
13. Cabello, M. R. *et al.* A New Efficient and Stable 3D Conformal FDTD. *IEEE Microwave and Wireless Components Letters* **26**, 553–555 (2016).
14. Benkler, S., Chavannes, N. & Kuster, N. A new 3-D conformal PEC FDTD scheme with user-defined geometric precision and derived stability criterion. *IEEE Transactions on Antennas and Propagation* **54**, 1843–1849 (2006).
15. Bekmambetova, F., Zhang, X. & Triverio, P. A Dissipative Systems Theory for FDTD With Application to Stability Analysis and Subgridding. *IEEE Transactions on Antennas and Propagation* **65**, 751–762 (2017).
16. Benkler, S., Chavannes, N. & Kuster, N. Mastering Conformal Meshing for Complex CAD-Based C-FDTD Simulations. *IEEE Antennas and Propagation Magazine* **50**, 45–57 (2008).
17. Guo, Y., Wang, X. & Hu, J. A new parallel meshing technique integrated into the conformal FDTD method for solving complex electromagnetic problems. *Journal of Zhejiang University - Science C* **15**, 1087–1097 (2014).
18. Harris, C. R. *et al.* Array programming with NumPy. *Nature* **585**, 357–362. <https://doi.org/10.1038/s41586-020-2649-2> (Sept. 2020).
19. Gillies, S. *et al.* Shapely version 2.0.4. Apr. 2024. <https://github.com/shapely/shapely>.
20. Pozar, D. M. *Microwave Engineering* 4th. ISBN: 978-0470631553 (John Wiley & Sons, 2012).
21. Angulo, L. D., Amador, F. M. & Manterola, A. M. *PyDG1D* Jan. 2024. <https://github.com/lmdiazangulo/PyDG1D>.
22. Junkin, G. Conformal FDTD Modeling of Imperfect Conductors at Millimeter Wave Bands. *IEEE Transactions on Antennas and Propagation* **59**, 199–205 (2011).
23. Tekbaş, K. *Subcell FDTD Method for Ultrawideband Applications* PhD thesis (The University of Manchester, 2017).
24. Rao, S. M. *Time Domain Electromagnetics* (Elsevier Science, 1999).
25. Davidson, D. B. *Computational Electromagnetics for RF and Microwave Engineering* (Cambridge University Press, 2005).

26. Yee, K. S. Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. *IEEE Trans. Antennas Propag.* **AP-14** (1966).
27. Courant, R., Friedrichs, K. & Lewy, H. On the partial difference equations of mathematical physics. *IBM Journal of Research and Development* **11**, 215–234 (Mar. 1967).
28. Inan, U. S. & Marshall, R. A. *Numerical Electromagnetics: The FDTD Method* (Cambridge University Press, 2011).