ABSTRACT

In this dissertation, efficient time-domain domain decomposition algorithms are investigated, compared, and further enhanced, and a new domain decomposition method is proposed based on the knowledge of existing ones. First, several explicit domain decomposition methods, including the dual-field domain decomposition (DFDD) method and two versions of the discontinuous Galerkin time-domain (DGTD) method, are investigated and compared in terms of accuracy and efficiency. Furthermore, the hybrid versions of DFDD and DGTD are also compared. Second, the modeling of doubly lossy and dispersive media is incorporated into the DFDD method, which demonstrates the accuracy and efficiency in the comparative study, but can only model non-dispersive media in its original version. The phase error analysis indicates that the enhanced DFDD algorithm maintains the same accuracy level as the original version. Third, a new domain decomposition method named the layered domain decomposition (LADD) method is proposed. Based on the layered geometry of printed circuit board (PCB) structures, the unknowns within each subdomain are eliminated and a global interface problem containing only the unknowns at the via holes is obtained. The interface problem is then solved and the volume unknowns in each subdomain are recovered. This method maintains the unconditional stability of the finite element time-domain (FETD) method and generates results that are identical to FETD. Moreover, the algorithm is highly parallelizable since the computational time is dominated by the solution of subdomain problems which is performed independently for each subdomain. Various numerical examples are presented to compare the existing algorithms and to validate the proposed ones.
To My Parents and Grandparents
ACKNOWLEDGMENTS

I am very grateful to my adviser, Professor Jian-Ming Jin, for the guidance and instructions through the six years, without which this work would not be possible. I have benefited a lot from them, and will continue to benefit in the future.

I would like to thank Professor Andreas C. Cangellaris, Professor Luke Olson, and Professor José E. Schutt-Ainé for serving as members of my doctoral committee and providing valuable comments and suggestions. I would like to thank previous members in Professor Jian-Ming Jin’s research group, Dr. Zheng Lou, Dr. Kaiyu Mao, Dr. Yujia Li, Dr. Shih-Hao Lee, and especially Dr. Rui Wang, for their warm help, useful suggestions, and previous work in this area. I would like to thank the current group members for the helpful discussions and it is my great pleasure to work with them. I also want to thank Dr. Jilin Tan for the industrial knowledge I learned from him which makes my research work more practical.

Last but not least, I am indebted to my family for their dedication and continued support in my life.
TABLE OF CONTENTS

1 INTRODUCTION ......................................................................................................... 1
    1.1 Finite-Element Time-Domain Method................................................................. 1
    1.2 Comparative Study of Three Finite Element–Based Explicit Numerical Schemes .......................................................... 2
    1.3 Modeling of Doubly Lossy and Dispersive Media with the Dual-Field Domain-Decomposition Algorithm ................. 5
    1.4 Time-Domain Modeling of 3D High-Speed Integrated Circuits......................... 6

2 FINITE-ELEMENT TIME-DOMAIN METHOD ........................................................ 9
    2.1 Introduction ...................................................................................................... 9
    2.2 System Equation and Boundary Conditions .................................................... 9
    2.3 Spatial Discretization .................................................................................... 12
    2.4 Temporal Discretization ................................................................................ 14

3 COMPARATIVE STUDY OF THREE FINITE ELEMENT–BASED EXPLICIT NUMERICAL SCHEMES .......................................................... 16
    3.1 Introduction .................................................................................................... 16
    3.2 Formulation ................................................................................................... 16
    3.3 Numerical Examples .................................................................................... 36
    3.4 Figures and Tables ........................................................................................ 41

4 MODELING OF DOUBLY LOSSY AND DISPERersive MEDIA WITH THE DUAL-FIELD DOMAIN-DECOMPOSITION ALGORITHM .......................................................... 50
    4.1 Introduction .................................................................................................. 50
    4.2 Formulation .................................................................................................. 50
    4.3 Analysis of Phase Error ............................................................................... 56
    4.4 Numerical Examples ................................................................................... 59
    4.5 Figures and Tables ....................................................................................... 63

5 TIME-DOMAIN MODELING OF 3D HIGH-SPEED INTEGRATED CIRCUITS .......................................................... 68
    5.1 Introduction .................................................................................................. 68
    5.2 Preliminary Tests .......................................................................................... 68
CHAPTER 1
INTRODUCTION

1.1 Finite-Element Time-Domain Method

Among all algorithms in the area of computational electromagnetics, three are the most important: the method of moments (MoM) [1], [2], the finite-difference time-domain (FDTD) method [3], and the finite element method (FEM) [4], [5]. The MoM is based on Green’s functions and converts Maxwell’s equations into integral equations. This method is perfectly suitable for radiation and scattering problems with metallic surfaces and isotropic, homogeneous or layered homogeneous materials since only a surface discretization is needed and the Sommerfeld radiation condition can be automatically built into Green’s functions. However, this method encounters difficulty in modeling complex, anisotropic and inhomogeneous materials and also in dealing with the full system matrix although the latter problem is largely alleviated by the development of fast solvers. The FDTD method solves Maxwell’s equations directly in the time domain on a Cartesian grid, and gains its popularity due to its simple formulation and the ability to handle material anisotropy and inhomogeneity. Moreover, field unknowns are updated locally and the need to invert a global system matrix is avoided. Nevertheless, the capability of FDTD is challenged when complex geometries are encountered. Due to the staircase approximation in the traditional FDTD, the number of elements becomes extremely large and the time step size becomes quite small when a fine grid is employed, resulting in a high solution cost. Different techniques can be employed to alleviate this problem, but at the cost of sacrificing formulation simplicity or efficiency. The FEM solves Maxwell’s equations or the wave equation on an unstructured grid, thus it has a good geometry modeling capability. Also, anisotropic and inhomogeneous materials can be well handled in FEM. The major limitation of FEM is the need to solve a system equation containing a large number of unknowns resulting from the volume discretization, though this problem is lessened by efficient sparse solvers. (A review of FEM and an extensive list of literature on the subject can be found in [6], [7].)
The FEM can be formulated either in the frequency domain or in the time domain. Compared to the finite-element frequency-domain method (FEFD), the finite-element time-domain method (FETD) is strong in conducting transient analysis, performing broadband characterization, and modeling nonlinear media and devices. According to the equations being solved, FETD can be categorized into two classes. The first class solves two Maxwell’s equations directly for both the electric and magnetic fields and generally works in a leapfrog fashion similar to FDTD, i.e., the electric field is solved at integer time steps and the magnetic field is solved at half-integer time steps [8]. For this class of approaches, the need to solve a global matrix equation can be avoided by applying the mass-lumping technique [9]; however, the time-marching scheme is only conditionally stable and the well-developed FEFD techniques based on the second-order wave equation cannot be adapted straightforwardly to the time-domain scheme. In contrast, the second class solves a second-order wave equation for one field variable and the other one can be recovered through Maxwell’s equations if needed [10]. For the second class of approaches, an unconditionally stable time-marching scheme can be obtained by employing the Newmark-Beta method, so that the time step size can be selected independently of the mesh size. Moreover, the FEFD techniques can be adapted more straightforwardly to the time-domain formulation. The major limitation of this scheme is the need to solve a global matrix equation at each time step. However, due to the above two advantages, we choose to implement the second class.

1.2 Comparative Study of Three Finite Element–Based Explicit Numerical Schemes

As mentioned in the previous section, the FETD has to solve a global matrix equation at every time step. Direct solvers can be used to pre-factorize the system matrix so that the factorization can be reused at each time step to reduce the marching time, but this becomes less practical when the problem size becomes larger, due to the excessive factorization memory and time. Iterative solvers often have to be used for large-scale problems to reduce the memory usage, but the convergence property severely depends on
the problem physics. At a certain point, the problem size becomes so large that even iterative solvers break down.

Various efforts have been made to improve the efficiency of the traditional, fully implicit FETD; important progress was made with the development of the dual-field domain-decomposition (DFDD) method [11], [12]. In DFDD, the electric and magnetic fields are solved from the two second-order vector wave equations in a leapfrog manner, and the tangential field continuities at subdomain interfaces are weakly enforced by exchanging equivalent surface electric and magnetic currents. In this way, the communication cost among processors is minimized. DFDD reduces to the fully implicit FETD when there is only one subdomain, which is the entire computational domain, and to a fully explicit scheme when each finite element is treated as a subdomain, which is named as the dual-field domain-decomposition–element-level decomposition (DFDD-ELD) method. In this fully explicit version, the size of the matrix equations to be solved equals the number of unknowns in one finite element; therefore, the storage and solution of a global system matrix is avoided. Furthermore, the explicit scheme greatly facilitates parallel computation since the computational load can be well balanced among different processors. However, the time step size in the explicit scheme is restricted by the smallest element size throughout the computational domain, which is highly undesirable since fine geometries requiring fine meshes to resolve are usually encountered in realistic problems. To relax the restriction on the time step size in the explicit scheme while keeping the advantage of domain decomposition, a hybrid implicit-explicit scheme has been developed between the two extremes where the smaller elements around fine structures are grouped together and solved using the implicit method and the larger elements elsewhere are handled by using the explicit method [12]. On one hand, the maximum step size is determined by the smallest element size in the explicit region and that on the boundary of the implicit region, and this condition is much looser than that in the fully explicit scheme. On the other hand, the size of the system equation to be solved equals the number of unknowns in one subdomain, which is much smaller than that in the entire computational domain.

Another promising method for solving partial differential equations is the discontinuous Galerkin method which has been applied to the solution of the neutron
transport equation in the last century [13]. This method was introduced into the area of computational electromagnetics to solve time-domain Maxwell’s equations ten years ago [14] and extensive research has been carried out on this topic since then [15]-[28]. The discontinuous Galerkin time-domain (DGTD) method achieves domain decomposition by introducing the numerical fluxes at the element interfaces, where the tangential field components are allowed to be slightly discontinuous. In this way, an explicit scheme is obtained and the matrix equations are solved at the element level like in DFDD-ELD. Since the only communication among processors is the exchange of numerical fluxes, DGTD is also suitable for parallel computation. Similar to DFDD-ELD, the fully explicit DGTD suffers from the time-step restriction problem and hybrid implicit-explicit schemes have been developed to mitigate this problem so that a better efficiency can be achieved [21]-[25].

In contrast to DFDD which solves the two vector wave equations, most DGTD methods solve two Maxwell’s equations directly and they can be categorized into two versions according to the types of fluxes introduced: the upwind flux version (DGTD-Upwind) where the fluxes are obtained by solving a one-dimensional Riemann problem [14], [15], [18], [19] and the central flux version (DGTD-Central) where the fluxes result from taking the average of the tangential field components at the interfaces, or enforcing the energy conservation law [16], [17], [19], [20]. DGTD-Upwind is usually integrated in time by using high-order Runge-Kutta methods, and it has an optimal convergence rate with respect to the spatial discretization but is slightly numerically dissipative. DGTD-Central can be discretized in time using either the leapfrog scheme or Runge-Kutta schemes, and it has a suboptimal convergence rate but conserves a discrete form of electromagnetic energy.

Since DFDD and DGTD share the aforementioned advantages, it is interesting to perform a comparative study. Such studies of the two DGTD methods have been conducted in terms of error convergence rate [19], [26]-[28]. In our work, we perform a more comprehensive study of the three explicit algorithms and compare them in terms of both accuracy and efficiency [29]. The hybrid scheme for DFDD [12] and that for DGTD [24], [25] are also investigated and compared in terms of efficiency.
1.3 Modeling of Doubly Lossy and Dispersive Media with the Dual-Field Domain-Decomposition Algorithm

Although the DFDD algorithm is highly efficient, the frequency dispersion of media has not been considered in its original version, which limits the scope of problems that can be modeled. Different approaches to modeling an electrically dispersive medium have been proposed a few years ago [30]-[32] and the formulation to handle a medium with both electric and magnetic dispersion has also been developed [7], [33], [34]. These approaches assume that the electric and magnetic susceptibility functions take the form of a pole expansion in the frequency domain (typical media are plasma, Debye, and Lorentz media), and thus a sum of exponential functions in the time domain. Then a recursive convolution formula is obtained by making use of the special mathematical property of exponential functions, which allows the fast computation of time convolutions and saves the computational time. To model media with arbitrary susceptibility functions, the well-known vector-fitting technique is usually applied to approximate the susceptibility functions with pole expansions.

A more general approach based on the recursive fast Fourier transform (FFT) algorithm has also been developed, which does not require the aforementioned pole expansion of the susceptibility functions [35]-[40]. The basic idea of this approach is to apply FFT to the field values that have been obtained to pre-calculate part of the convolutions for later time steps. And this idea is then applied in a hierarchical manner to achieve a better efficiency. This approach requires a higher computational cost compared to the recursive convolution approach with only a few poles, but it is very useful when the susceptibility functions cannot be accurately approximated using a small number of poles.

In our work, the recursive convolution approach is employed and extended to the dual-field case for the modeling of doubly lossy and dispersive media in the DFDD algorithm, resulting in a general DFDD algorithm for dealing with large-scale electromagnetic problems involving such media, such as antenna arrays or integrated circuits with dispersive substrates [41]. In contrast to the previous approaches which solve the electric field from the second-order E-equation and the magnetic field from one of Maxwell’s equations, our method directly uses the magnetic field solved from the second-
order H-equation to avoid redundancy. Furthermore, a transformation is performed to remove the instability problem which does not exist in the previous approaches designed for FETD but emerges in DFDD. A quantitative error analysis is performed to estimate the error induced by the modeling of medium dispersion. It should be noted that our method is not limited to the recursive convolution approach; the recursive FFT approach can also be employed in a straightforward manner.

1.4 Time-Domain Modeling of 3D High-Speed Integrated Circuits

Nowadays, three-dimensional (3D) high-speed circuits have gained important applications in a variety of areas. As the operating frequency and integration level increase, some effects which could be safely neglected in the past become significant, and these include increased conductor and substrate losses, frequency-dependent parasitic inductances and capacitances, skin effect, and electromagnetic (EM) coupling among different components. These phenomena may cause signal decay, dispersion, phase delay, and crosstalk, which may adversely affect the circuit performance or even result in a system failure. Therefore, the accurate EM modeling of these effects is critical to the circuit design.

Among various candidates for 3D circuit simulations, the finite element method has become an important one because it can be implemented with unstructured meshes, which allow accurate representation of complicated circuit geometries and can handle conveniently complex, inhomogeneous dielectrics from the board to the chip. In circuit simulations, the transient response is sometimes desired, the broadband impedance and scattering parameters are often required, and nonlinear circuit components are frequently encountered. In these cases, the FETD method would be preferred over its frequency-domain counterpart, and it has been used to simulate 3D circuit structures since a few years ago [42], [43].

As mentioned in the previous sections, the traditional, fully implicit FETD requires the solution of a global matrix equation at each time step, which is computationally intensive, and domain decomposition algorithms such as DFDD and DGTD have been developed to improve the efficiency. Very recently, DGTD methods were applied to
circuit simulations by several research groups [20], [44], [45]. In [20], linear passive lumped elements were incorporated into the explicit DGTD framework through proper modification of the boundary conditions at the element interfaces. This approach is straightforward to implement, and it does not affect the DGTD stability condition. In [44], an efficient hybrid implicit-explicit DGTD scheme was proposed for the modeling of multi-layered circuit structures, where domain decomposition is performed in the direction of the layer stack and implicit and explicit schemes are used for subdomains with dense and coarse meshes, respectively. In [45], a hybrid field-circuit solver is proposed, where the explicit DGTD method which generates the field solution is coupled with SPICE which provides the circuit simulation. This method took advantage of SPICE in simulating complex, linear and nonlinear components and does not require the extra implementation of a circuit solver; thus, it is well suited for industrial applications where SPICE has been used intensively. Despite the aforementioned work, the application of FETD-based algorithms to circuit problems is still quite limited, compared to their vast applications in scattering and radiation problems. In our research, we apply FETD, DFDD, and DGTD to circuit simulations and investigate their efficiency.

Since fine geometries are often encountered in 3D circuits, an unconditionally stable decomposition algorithm is highly desired. It is noticed that an efficient domain decomposition method has been proposed in the frequency domain by exploring the layered geometry of PCBs [46]. In this algorithm, the volume unknowns inside each subdomain are individually eliminated, resulting in a global matrix equation containing only the via hole unknowns at the subdomain interfaces, which can be solved to extract the scattering parameters. Based on this algorithm, a new domain decomposition method in the time domain, named the layered domain decomposition (LADD) method, is proposed, where each subdomain consists of one or more dielectric layers and the subdomains are separated by ground planes. At each time step, the volume unknowns in each subdomain are eliminated and a small global matrix equation is obtained and solved for the via hole unknowns, from which the volume unknowns in each subdomain are recovered. LADD has several advantages: first, it preserves the unconditional stability of FETD since the system solved by the former is completely equivalent to that solved by the latter; second, LADD can achieve a good parallel efficiency since the serial steps
consume little computational time compared to the parallel steps; finally, LADD introduces no extra errors except for rounding errors compared to FETD. Therefore, LADD is likely to gain important applications in 3D circuit simulations, especially in problems where fine geometries are frequently encountered.
CHAPTER 2
FINITE-ELEMENT TIME-DOMAIN METHOD

2.1 Introduction

In this chapter, the basic formulation of the finite-element time-domain (FETD) method will be reviewed briefly. In Section 2.2, the system equation will be derived using the Galerkin testing procedure, and then different types of boundary conditions including the perfect electric conductor (PEC), the first-order absorbing boundary condition (ABC), and the waveguide port boundary condition (WPBC) will be discussed. In Section 2.3, different types of basis functions will be described and a semi-discrete system will be obtained after the spatial discretization. In Section 2.4, the temporal discretization using the unconditionally stable Newmark-Beta scheme will be presented and the fully discrete matrix equation will be obtained.

2.2 System Equation and Boundary Conditions

The FETD system equation can be obtained by applying the Galerkin testing procedure to the wave equation. Assuming that we have a computational domain \( V \) enclosed by the boundary \( S \), the following time-domain Maxwell’s equations are satisfied inside the volume \( V \):

\[ \nabla \times \mathbf{H} = \varepsilon \frac{\partial \mathbf{E}}{\partial t} + \sigma \mathbf{E} + \mathbf{J}_{\text{imp}} \]  \hspace{1cm} (2.1)

\[ \nabla \times \mathbf{E} = -\mu \frac{\partial \mathbf{H}}{\partial t} \]  \hspace{1cm} (2.2)

where \( \varepsilon \), \( \mu \), \( \sigma \), and \( \mathbf{J}_{\text{imp}} \) are the permittivity, permeability, conductivity, and impressed current density, respectively. By taking the time derivative of (2.1) and substituting (2.2) into the resultant equation we obtain the second-order wave equation

\[ \nabla \times \left( \frac{1}{\mu} \nabla \times \mathbf{E} \right) + \varepsilon \frac{\partial^2 \mathbf{E}}{\partial t^2} + \sigma \frac{\partial \mathbf{E}}{\partial t} = -\frac{\partial \mathbf{J}_{\text{imp}}}{\partial t}. \]  \hspace{1cm} (2.3)
Testing the equations above using a vector basis function $T$ yields

$$T \cdot \nabla \times \left( \frac{1}{\mu} \nabla \times E \right) + \varepsilon T \cdot \frac{\partial^2 E}{\partial t^2} + \sigma T \cdot \frac{\partial E}{\partial t} = -T \cdot \frac{\partial J_{\text{imp}}}{\partial t}. \quad (2.4)$$

By integrating this equation in volume $V$ and applying the divergence theorem, we obtain the following system equation:

$$\int \int \int_V \left[ \frac{1}{\mu} (\nabla \times T) \cdot (\nabla \times E) + \varepsilon T \cdot \frac{\partial^2 E}{\partial t^2} + \sigma T \cdot \frac{\partial E}{\partial t} \right] dV - \oint_S T \cdot \left( \hat{n} \times \frac{\partial H}{\partial t} \right) dS = -\int \int \int_V T \cdot \frac{\partial J_{\text{imp}}}{\partial t} dV \quad (2.5)$$

where $\hat{n}$ is the outward unit vector normal to the boundary $S$ and the surface integral term can be used to incorporate proper boundary conditions. Here we consider three types of boundary conditions: PEC, ABC, and WPBC, as described below.

**PEC:** Since we will use the vector basis functions described in the next section, the PEC boundary condition can be enforced by setting the unknowns on PEC to zero, or simply eliminating these unknowns.

**ABC:** The first-order ABC can be stated as

$$\hat{n} \times \left( \frac{1}{\mu} \nabla \times E \right) + Y \frac{\partial}{\partial t} (\hat{n} \times \nabla \times E) = 0 \quad (2.6)$$

where $Y$ is the characteristic admittance of the medium. The ABC can be incorporated into (2.5) by the following substitution:

$$- \int \int_{S_{\text{inc}}} T \cdot \left( \hat{n} \times \frac{\partial H}{\partial t} \right) dS = \int \int_{S_{\text{inc}}} T \cdot \hat{n} \times \left( \frac{1}{\mu} \nabla \times E \right) dS = -\int \int_{S_{\text{inc}}} T \cdot Y \frac{\partial}{\partial t} (\hat{n} \times \nabla \times E) dS. \quad (2.7)$$

**WPBC:** This boundary condition is developed in order to accurately launch an excitation into a waveguide structure, and to accurately absorb both propagating and evanescent modes coming out from the structure \[47\]. WPBC is a third-kind boundary condition which can be written as

$$\hat{n} \times (\nabla \times E) + P(E) = U^{\text{inc}}. \quad (2.8)$$

By performing a modal expansion for the field as
\[
E = E^{inc} + a_0 e_0^{TEM} + \sum_{m=1}^{\infty} b_m e_m^{TE} + \sum_{m=1}^{\infty} c_m (e_m^{TM} + \hat{e}_m^{TM})
\]  
(2.9)

and making use of the modal orthogonality to obtain the coefficients \(a_0, b_m,\) and \(c_m,\) one can obtain the frequency-domain expressions

\[
P(E) = -\gamma_0 e_0^{TEM} \int_S e_0^{TEM} \cdot E dS - \sum_{m=1}^{\infty} \gamma_m e_m^{TEM} \int_S e_m^{TEM} \cdot E dS + \sum_{m=1}^{\infty} \frac{k^2}{\gamma_m} e_m^{TEM} \int_S e_m^{TEM} \cdot E dS
\]  
(2.10)

\[
U^{inc} = \hat{n} \times (\nabla \times E^{inc}) - \gamma_0 e_0^{TEM} \int_S e_0^{TEM} \cdot E^{inc} dS - \sum_{m=1}^{\infty} \gamma_m e_m^{TEM} \int_S e_m^{TEM} \cdot E^{inc} dS \\
+ \sum_{m=1}^{\infty} \frac{k^2}{\gamma_m} e_m^{TEM} \int_S e_m^{TEM} \cdot E^{inc} dS
\]  
(2.11)

where \(\gamma_0 = jk\) and \(\gamma_m = \sqrt{k^2 - \kappa^2}\) with \(k\) and \(k_m\) being the wave number and cutoff wave number. When transformed into the time domain, (2.10) and (2.11) become

\[
P(E) = -e_0^{TEM} \int_S e_0^{TEM} \cdot \left( \frac{1}{c} \frac{\partial}{\partial t} E \right) dS - \sum_{m=1}^{\infty} e_m^{TEM} \int_S e_m^{TEM} \cdot \left( \frac{1}{c} \frac{\partial}{\partial t} E + h_m(t) \otimes E \right) dS
\]  
(2.12)

\[
U^{inc} = \hat{n} \times (\nabla \times E^{inc}) - e_0^{TEM} \int_S e_0^{TEM} \cdot \left( \frac{1}{c} \frac{\partial}{\partial t} E^{inc} \right) dS \\
- \sum_{m=1}^{\infty} e_m^{TEM} \int_S e_m^{TEM} \cdot \left( \frac{1}{c} \frac{\partial}{\partial t} E^{inc} + h_m(t) \otimes E^{inc} \right) dS \\
- \sum_{m=1}^{\infty} e_m^{TEM} \int_S e_m^{TEM} \cdot \left( \frac{1}{c} \frac{\partial}{\partial t} E^{inc} + g_m(t) \otimes E^{inc} \right) dS
\]  
(2.13)

where \(\otimes\) denotes time convolution, \(c\) is the speed of light, and

\[
h_m(t) = \frac{k_m}{t} J_1(k_m c t) \bar{u}(t)
\]  
(2.14)

\[
g_m(t) = \frac{k_m}{t} J_1(k_m c t) \bar{u}(t) - k_m^2 c J_0(k_m c t) \bar{u}(t)
\]  
(2.15)

with \(J_0, J_1,\) and \(\bar{u}(t)\) being the zeroth-order Bessel function, the first-order Bessel function, and the unit step function. The time-domain WPBC can be incorporated into (2.5) by the following substitution:
\[- \iint_{S_{\text{trac}}} \mathbf{T} \cdot \left( \hat{n} \times \frac{\partial \mathbf{H}}{\partial t} \right) dS = \iint_{S_{\text{trac}}} \mathbf{T} \cdot \hat{n} \times \left( \frac{1}{\mu} \nabla \times \mathbf{E} \right) dS = \iint_{S_{\text{trac}}} \frac{1}{\mu} \mathbf{T} \cdot \left( \mathbf{U}^{\text{inc}} - P(\mathbf{E}) \right) dS. \quad (2.16)\]

2.3 Spatial Discretization

To solve Eq. (2.5), spatial discretization is needed where the electric field is expanded using proper basis functions. Since the traditional nodal basis functions suffer from serious problems such as the occurrence of spurious solutions, the inconvenience of imposing boundary conditions at material interfaces and conducting surfaces, and the difficulty in treating conducting and dielectric corners, vector basis functions are introduced to overcome these problems [48]. In the lowest-order vector basis functions, the degrees of freedom are assigned to the edges instead of nodes; therefore, the tangential field continuity is automatically satisfied at dielectric interfaces and the boundary condition at PEC surfaces can be enforced by simply setting the corresponding unknowns to zero. Besides, the difficulty in handling corners disappears since the field definition at singularity points is avoided. Moreover, the divergence condition is implied by these basis functions which exempt the field solution from spurious modes. Due to these advantages, vector basis functions are widely adopted today in computational EM.

In order to improve the poor convergence rate of the lowest-order edge basis functions, higher-order vector basis functions have been developed [48]. According to the construction procedure, there are two types of higher-order vector basis functions. The first type is the interpolatory basis functions which are defined at a set of interpolatory points on the element, and each basis function vanishes at all points except one. These basis functions have a good linear independence which results in a better-conditioned matrix equation, a clear physical interpretation which makes the enforcement of boundary conditions easier, and a unified expression which significantly simplifies the computer coding [49]. However, the higher-order basis functions are completely different from the lower-order ones, which makes it impossible to use \( p \)-adaptation, i.e. to iteratively increase the basis function order until convergence is achieved. The second type is the hierarchical basis functions [50], where the higher-order basis functions are constructed by adding new basis functions to the lower-order basis functions. This type of basis
functions allows the use of $p$-adaptation, which may significantly improve the computational efficiency. Furthermore, $p$-adaptation can be combined with $h$-adaptation to achieve excellent efficiency. Due to this advantage, we have employed the hierarchical basis functions throughout our work.

After expanding the electric field using vector basis functions as $E = \sum_j N_j e_j$, where $N_j$ is the $j$-th basis function and $e_j$ is the related unknown, and using the same basis functions as testing functions, we obtain the following semi-discrete matrix equation from (2.5):

$$
\frac{1}{c_0^2} [M] \frac{d^2 \{e\}}{dt^2} + \frac{1}{c_0} ([B] + [A] + [P]) \frac{d \{e\}}{dt} + [S] \{e\} + \sum_{m=1}^{\infty} [Q^m] \{u_m\} \\
+ \sum_{m=1}^{\infty} [R^m] \{v_m\} = -\frac{1}{c_0} \frac{d \{f\}}{dt}
$$

(2.17)

where the excitation term due to WPBC has been omitted for simplicity and

$$
M(i, j) = \iiint_V e_i N_j \cdot \nabla \cdot N_j dV
$$

(2.18)

$$
B(i, j) = Z_0 \iiint_V \sigma N_i \cdot N_j dV
$$

(2.19)

$$
A(i, j) = Z_0 \int_{S_e} \nabla \times (\hat{n} \times N_i) \cdot (\hat{n} \times N_j) dS
$$

(2.20)

$$
P(i, j) = \mu_0 \left( \Phi_{i0}^{TEM} \cdot \Phi_{0j}^{TEM} \right) + \sum_{m=1}^{\infty} \Phi_{im}^{TEM} \cdot \Phi_{jm}^{TM} + \sum_{m=1}^{\infty} \Phi_{im}^{TM} \cdot \Phi_{jm}^{TM}
$$

(2.21)

$$
S(i, j) = \iiint_V \frac{1}{\mu_r} (\nabla \times N_i) \cdot (\nabla \times N_j) dV
$$

(2.22)

$$
Q^m(i, j) = Z_0 \Phi_{im}^{TEM} \cdot \Phi_{jm}^{TM}
$$

(2.23)

$$
R^m(i, j) = Z_0 \Phi_{im}^{TM} \cdot \Phi_{jm}^{TM}
$$

(2.24)

$$
\{u_m(t)\} = h_m(t) \otimes \{e(t)\}
$$

(2.25)

$$
\{v_m(t)\} = g_m(t) \otimes \{e(t)\}
$$

(2.26)

$$
f(i) = Z_0 \iiint_V N_i \cdot J_{imp} dV
$$

(2.27)
and \( c_0 \) and \( Z_0 \) denote the speed of light in vacuum and the characteristic impedance of vacuum, \( \varepsilon_r \) and \( \mu_r \) are the relative permittivity and permeability of the medium, and

\[
\Phi_{\text{TEM/TE/TM}} = \int_S (\mu_r \varepsilon) \frac{1}{4} N_j \cdot e_{\text{TEM/TE/TM}} dS.
\]  

(2.28)

### 2.4 Temporal Discretization

After spatial discretization, the problem has been cast into an ordinary differential Eq. (2.17), which needs to be further discretized in time. The Newmark-Beta method is employed here since it is unconditionally stable and second-order accurate:

\[
\begin{align*}
\frac{d^2 y}{dt^2} \bigg|_{t=n\Delta t} &= \frac{y^{n+1} - 2y^n + y^{n-1}}{(\Delta t)^2} \\
\frac{dy}{dt} \bigg|_{t=n\Delta t} &= \frac{y^{n+1} - y^{n-1}}{2\Delta t} \\
y \bigg|_{t=n\Delta t} &= \frac{1}{4} y^{n+1} + \frac{1}{2} y^n + \frac{1}{4} y^{n-1}
\end{align*}
\]  

where \( n \) denotes the current time step and \( \Delta t \) the time step size. Hence, the fully discretized matrix equation can be stated as

\[
[A_0] \{e\}^{n+1} = \{b\}^n - [A_1] \{e\}^n - [A_2] \{e\}^{n-1}
\]  

(2.32)

where

\[
[A_0] = \frac{1}{(c_0\Delta t)^2} [M] + \frac{1}{2c_0\Delta t} ([B] + [A] + [P]) + \frac{1}{4} [S]
\]  

(2.33)

\[
[A_1] = -\frac{2}{(c_0\Delta t)^2} [M] + \frac{1}{2} [S]
\]  

(2.34)

\[
[A_2] = \frac{1}{(c_0\Delta t)^2} [M] - \frac{1}{2c_0\Delta t} ([B] + [A] + [P]) + \frac{1}{4} [S]
\]  

(2.35)

\[
\{b\}^n = -\frac{1}{2c_0\Delta t} \left( \{f\}^{n+1} - \{f\}^{n-1} \right) - \sum_{m=1}^{n} [Q^m] \{u_m\}^n - \sum_{m=1}^{n} [R^m] \{v_m\}^n.
\]  

(2.36)
Equation (2.32) can be marched in time with the initial condition $\{e\}^0 = \{e\}^1 = 0$. If a direct solver is used, the time-independent system matrix $[A]$ can be pre-factorized and stored so that the factorization can be reused at each time step.
CHAPTER 3
COMPARATIVE STUDY OF THREE FINITE ELEMENT-BASED EXPLICIT NUMERICAL SCHEMES

3.1 Introduction

In this chapter, three FETD-based efficient domain decomposition methods will be investigated and compared in terms of accuracy and efficiency. The chapter is organized as follows: first, the formulation for fully-explicit DFDD, DGTD-Upwind, and DGTD-Central will be described in Sections 3.2.1, 3.2.2, and 3.2.3, respectively; then, the formulation for hybrid implicit-explicit DFDD and hybrid DGTD will be presented in Sections 3.2.4 and 3.2.5; finally, different methods are compared with each other in numerical examples in Section 3.3.

3.2 Formulation

3.2.1 Explicit DFDD

The formulation of the fully-explicit DFDD (namely DFDD-ELD) is described in [12] in detail and repeated here for convenience. In the explicit DFDD, each element is treated as a single subdomain and the matrix equations are solved at the element level. The formulation starts from taking the time derivative of Maxwell’s equations to obtain

\[-\mu_0 \nabla \times \frac{\partial \mathbf{H}}{\partial t} + \frac{1}{\varepsilon_0} \varepsilon \frac{\partial^2 \mathbf{E}}{\partial t^2} = 0\]  \hspace{1cm} (3.1)

\[\varepsilon_0 \nabla \times \frac{\partial \mathbf{E}}{\partial t} + \frac{1}{\varepsilon_0} \mu \frac{\partial^2 \mathbf{H}}{\partial t^2} = 0\]  \hspace{1cm} (3.2)
where we have ignored the terms related to conductor loss and impressed currents for the simplicity of presentation. By testing the above equations using a function \( T \) and integrating the equations over the element under consideration, we can obtain

\[
\nabla \cdot \left( \mu \nabla \times \mathbf{T} \right) - \nabla \times \left( \varepsilon \nabla \times \mathbf{E} \right) - \frac{1}{c_0^2} \nabla^2 \mathbf{E} = \mu_0 \sum_{S_i} \mathbf{T} \cdot \left( \hat{n} \times \frac{\partial \mathbf{H}}{\partial t} \right) dS
\]

(3.3)

\[
\nabla \cdot \left( \varepsilon \nabla \times \mathbf{T} \right) - \nabla \times \left( \mu \nabla \times \mathbf{H} \right) - \frac{1}{c_0^2} \nabla^2 \mathbf{H} = -\varepsilon_0 \sum_{S_i} \mathbf{T} \cdot \left( \hat{n} \times \frac{\partial \mathbf{E}}{\partial t} \right) dS
\]

(3.4)

where the divergence theorem has been employed and Maxwell’s equations have been applied to obtain the first terms on the left-hand side in (3.3) and (3.4). Also, \( V_e \) and \( S_e \) denote the volume of the element and its boundary and proper boundary conditions can be incorporated by the surface integrals on the right-hand side. Three types of boundaries are considered: the PEC boundary \( S_{PEC} \), the first-order absorbing boundary \( S_{ABC} \), and the interface between elements, \( S_I \), and thus \( S = S_{PEC} + S_{ABC} + S_I \). First, the PEC boundary condition can be enforced by setting PEC unknowns to zero and then ignoring the surface integrals related to PECs. Second, the ABC reads as

\[
-\hat{n} \times \mathbf{H} + \mathbf{Y} \hat{n} \times \mathbf{E} = 0
\]

(3.5)

\[
\hat{n} \times \mathbf{E} + \mathbf{Z} \hat{n} \times \mathbf{H} = 0.
\]

(3.6)

Although the above two equations seem to be different from Eq. (2.6), they are essentially the same: (2.6) can be obtained by taking the time derivative of (3.5) and applying one of Maxwell’s equations. From (3.5) and (3.6) we can obtain

\[
\mu_0 \sum_{S_{ABC}} \mathbf{T} \cdot \left( \hat{n} \times \frac{\partial \mathbf{H}}{\partial t} \right) dS = -\mu_0 \sum_{S_{ABC}} \mathbf{Y} \hat{n} \times \mathbf{E} \cdot \left( \frac{\partial \mathbf{H}}{\partial t} \right) dS
\]

(3.7)

\[
-\varepsilon_0 \sum_{S_{ABC}} \mathbf{T} \cdot \left( \hat{n} \times \frac{\partial \mathbf{E}}{\partial t} \right) dS = -\varepsilon_0 \sum_{S_{ABC}} \mathbf{Z} \hat{n} \times \mathbf{H} \cdot \left( \frac{\partial \mathbf{H}}{\partial t} \right) dS.
\]

(3.8)

Finally, the interfaces between adjacent elements are considered. Notice that the proper treatment of the element interfaces is the key point in the DFDD algorithm. If the
following equivalent currents are defined at the element interfaces by using the tangential field components

\[ \mathbf{J}_n = \hat{n} \times \mathbf{H} \quad (3.9) \]

\[ \mathbf{M}_n = -\hat{n} \times \mathbf{E}, \quad (3.10) \]

part of the right-hand side in (3.3) and (3.4) can be rewritten as

\[ \mu_0 \int_{S_i} \mathbf{T} \cdot \left( \hat{n} \times \frac{\partial \mathbf{H}}{\partial t} \right) dS = \mu_0 \int_{S_i} (\hat{n} \times \mathbf{J}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{J}}{\partial t} \right) dS \quad (3.11) \]

\[ -\varepsilon_0 \int_{S_i} \mathbf{T} \cdot \left( \hat{n} \times \frac{\partial \mathbf{E}}{\partial t} \right) dS = \varepsilon_0 \int_{S_i} (\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{M}}{\partial t} \right) dS. \quad (3.12) \]

The idea of the DFDD algorithm is to use the fields in the neighboring subdomains to calculate \( \mathbf{J}_n \) and \( \mathbf{M}_n \). By doing this, the tangential field continuities at the boundary are weakly enforced and different subdomains are coupled together. After applying all boundary conditions, (3.3) and (3.4) become

\[ \int_V \left[ \frac{1}{\mu_r} \left( \nabla \times \mathbf{E} \right) \cdot \left( \nabla \times \mathbf{H} \right) + \frac{1}{c_0^2} \varepsilon_r \mathbf{T} \cdot \frac{\partial^2 \mathbf{E}}{\partial t^2} \right] dV + \mu_0 \int_{S_{ac}} Y(\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{E}}{\partial t} \right) dS \]

\[ = \mu_0 \int_{S_i} (\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{J}}{\partial t} \right) dS \quad (3.13) \]

\[ \int_V \left[ \frac{1}{\varepsilon_r} \mathbf{E} \cdot \mathbf{H} + \frac{1}{c_0^2} \mu_r \mathbf{T} \cdot \frac{\partial^2 \mathbf{H}}{\partial t^2} \right] dV + \mu_0 \int_{S_{ac}} Z(\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{H}}{\partial t} \right) dS \]

\[ = \varepsilon_0 \int_{S_i} (\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{M}}{\partial t} \right) dS. \quad (3.14) \]

In DFDD, the electric and magnetic fields are expanded using the same set of basis functions in space: \( \mathbf{E} = \sum_j \mathbf{N}_j e_j \) and \( \mathbf{H} = \sum_j \mathbf{N}_j h_j \), where \( e_j \) and \( h_j \) denote the electric and magnetic field unknowns related to basis function \( \mathbf{N}_j \), respectively. By substituting the expansion into (3.13) and (3.14), the semi-discretized system equations can be obtained as
\[ [S_e]\{e\} + \frac{1}{c_0^2}[M_e]\frac{\partial^2\{e\}}{\partial t^2} + \frac{1}{c_0}[A_e]\frac{\partial\{e\}}{\partial t} = \frac{1}{c_0}\frac{\partial\{j\}}{\partial t} \]  \hspace{1cm} (3.15)

\[ [S_h]\{h\} + \frac{1}{c_0^2}[M_h]\frac{\partial^2\{h\}}{\partial t^2} + \frac{1}{c_0}[A_h]\frac{\partial\{h\}}{\partial t} = \frac{1}{c_0}\frac{\partial\{m\}}{\partial t} \]  \hspace{1cm} (3.16)

where

\[ S_e(i, j) = \iiint_{V_e} \frac{1}{\mu_r} (\nabla \times \mathbf{N}_i) \cdot (\nabla \times \mathbf{N}_j) dV \]  \hspace{1cm} (3.17)

\[ M_e(i, j) = \iiint_{V_e} \varepsilon_r \mathbf{N}_i \cdot \mathbf{N}_j dV \]  \hspace{1cm} (3.18)

\[ A_e(i, j) = \iiint_{S_{vac}} (\hat{n} \times \mathbf{N}_i) \cdot (\hat{n} \times \mathbf{N}_j) dS \]  \hspace{1cm} (3.19)

\[ S_h(i, j) = \iiint_{V_h} \frac{1}{\varepsilon_r} (\nabla \times \mathbf{N}_i) \cdot (\nabla \times \mathbf{N}_j) dV \]  \hspace{1cm} (3.20)

\[ M_h(i, j) = \iiint_{V_h} \mu_r \mathbf{N}_i \cdot \mathbf{N}_j dV \]  \hspace{1cm} (3.21)

\[ A_h(i, j) = \iiint_{S_{vac}} (\hat{n} \times \mathbf{N}_i) \cdot (\hat{n} \times \mathbf{N}_j) dS \]  \hspace{1cm} (3.22)

\[ j(i) = Z_0 \iiint_{S_i} (\hat{n} \times \mathbf{N}_i) \cdot (\hat{n} \times \mathbf{J}_s) dS \]  \hspace{1cm} (3.23)

\[ m(i) = Y_0 \iiint_{S_i} (\hat{n} \times \mathbf{N}_i) \cdot (\hat{n} \times \mathbf{M}_s) dS \]  \hspace{1cm} (3.24)

In the above, the testing functions have been chosen to be the same as the basis functions. For temporal discretization, electric fields are sampled at integer time steps and magnetic field at half-integer time steps, and (3.15) and (3.16) can be discretized in time by using the Newmark-Beta scheme (2.29)-(2.31), resulting in a fully discretized system:

\[
\frac{1}{4}[S_e]\{e\}^{n+1} + 2\{e\}^n + \{e\}^{n-1} + \frac{1}{(c_0\Delta t)^2}[M_e]\{e\}^{n+1} - 2\{e\}^n + \{e\}^{n-1} + \frac{1}{2c_0\Delta t}[A_e]\{e\}^{n+1} - \{e\}^{n-1} = \frac{1}{2c_0\Delta t}\{j\}^{n+1} - \{j\}^{n-1} \\
\frac{1}{4}[S_h]\{h\}^{n+3/2} + 2\{h\}^{n+1/2} + \{h\}^{n-1/2} + \frac{1}{(c_0\Delta t)^2}[M_h]\{h\}^{n+3/2} - 2\{h\}^{n+1/2} + \{h\}^{n-1/2} 
\]  \hspace{1cm} (3.25)
The time marching process can be briefly summarized into four steps: 1. calculate $\textbf{J}^{n+1/2}$ from $\textbf{H}^{n+1/2}_{\text{neighbor}}$; 2. solve for $\textbf{E}^{n+1}$ by using (3.13); 3. calculate $\textbf{M}^{n+1}$ from $\textbf{E}^{n+1}_{\text{neighbor}}$; 4. solve for $\textbf{H}^{n+3/2}$ by using (3.14); where $n$ is the time step index and the subscript “neighbor” denotes that the value is taken from the neighboring subdomain. By repeating these four steps for every time step, the electric and magnetic fields are marched in a leapfrog manner.

This DFDD-ELD method breaks the original problem into smaller element-level problems, avoiding the need to factorize and solve a global matrix equation. Since the
only communication among elements is the exchange of surface currents, the communication cost among different processors is minimal in parallel computations, yielding a high parallel efficiency. The major limitation of this scheme is that it is conditionally stable and the time step size is limited by the smallest element size throughout the computational domain.

3.2.2 Explicit DGTD with Upwind Fluxes

In the explicit version of DGTD, Maxwell’s equations (2.1) and (2.2) are solved at the element level. In the following derivation we will again omit the terms related to sources and conductor losses for simplicity. By applying the Galerkin testing procedure to Maxwell’s equations within one element, we obtain

\[
\iiint_{V_e} \mathbf{T} \cdot \left( \epsilon \frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{H} \right) dV = 0 \\
\iiint_{V_e} \mathbf{T} \cdot \left( \mu \frac{\partial \mathbf{H}}{\partial t} + \nabla \times \mathbf{E} \right) dV = 0.
\]

(3.37)  
(3.38)

Applying the vector identity \( \mathbf{T} \cdot (\nabla \times \mathbf{A}) = \nabla \cdot (\mathbf{A} \times \mathbf{T}) + \mathbf{A} \cdot (\nabla \times \mathbf{T}) \) and the divergence theorem, the above two equations can be rewritten as

\[
\iiint_{V_e} \left[ \epsilon \mathbf{T} \cdot \frac{\partial \mathbf{E}}{\partial t} - (\nabla \times \mathbf{T}) \cdot \mathbf{H} \right] dV = \oint_{S_e} \mathbf{T} \cdot (\hat{n} \times \mathbf{H}) dS \\
\iiint_{V_e} \left[ \mu \mathbf{T} \cdot \frac{\partial \mathbf{H}}{\partial t} + (\nabla \times \mathbf{T}) \cdot \mathbf{E} \right] dV = -\oint_{S_e} \mathbf{T} \cdot (\hat{n} \times \mathbf{E}) dS.
\]

(3.39)  
(3.40)

Part of the surface integrals on the right-hand sides above will be replaced by the field values from the neighboring element, so that the tangential field continuities at the element interface can be weakly enforced. To be more specific, a 1D Riemann problem is solved in the direction normal to the elemental interface [15], [52]. Without loss of generality, we assume that the element interface lies in the y-z plane and the local element lies in the region \( x \leq 0 \) and the neighboring element in \( x \geq 0 \). Assume that there are two incident waves: the one propagating in the local element (the \( x \leq 0 \) region) in the +x direction characterized by
\[ E_{\text{inc}} = \hat{y}E_{\text{inc}} e^{-jkx}, \quad H_{\text{inc}} = \hat{z}E_{\text{inc}} e^{-jkx} / Z \]  

(3.41)

and that propagating in the neighboring element (the \(x \geq 0\) region) in the \(-x\) direction characterized by

\[ E_{\text{inc}}^* = \hat{y}E_{\text{inc}}^* e^{jkx^*}, \quad H_{\text{inc}}^* = -\hat{z}E_{\text{inc}}^* e^{jkx^*} / Z^*. \]  

(3.42)

The key step in the Riemann problem is to solve for the total fields in the local element, \(E\) and \(H\), in terms of the above two incident waves. It is observed that the wave in the local element can be decomposed into three waves: the incident wave in (3.41), the reflected wave \((E_{\text{ref}}, H_{\text{ref}})\) due to this incident wave, and the transmitted wave \((E_{\text{tr}}, H_{\text{tr}})\) due to the incident wave in (3.42). The last two waves can be solved by enforcing the tangential field continuities at the element interface \(x = 0\):

\[ \hat{j}k x_{\text{ref inc}} Z E e Z_{\text{y}} + - = + E, \quad \hat{j}k x_{\text{inc ref}} E Z z_{\text{e}} Z Z + - = - + H. \]  

(3.43)

\[ \hat{j}k x_{\text{tr inc}} Z E e Z + + = + E, \quad \hat{j}k x_{\text{inc tr}} E Z z_{\text{e}} Z Z + + = - + H. \]  

(3.44)

Therefore, the total field at the element interface can be written as

\[ E(x = 0^-) = E_{\text{inc}}(x = 0^-) + E_{\text{ref}}(x = 0^-) + E_{\text{tr}}(x = 0^-) \]

\[ = \hat{j} \left( \frac{Z^* - Z}{Z^* + Z} E_{\text{inc}} e^{jkx} + \frac{2Z}{Z^* + Z} E_{\text{tr}} \right) \]  

(3.45)

\[ H(x = 0^-) = H_{\text{inc}}(x = 0^-) + H_{\text{ref}}(x = 0^-) + H_{\text{tr}}(x = 0^-) \]

\[ = \hat{z} \left( \frac{2}{Z^* + Z} E_{\text{inc}} - \frac{2Z}{Z^* + Z} E_{\text{tr}} \right). \]  

(3.46)

The next step is to replace the incident fields with the total fields. By multiplying (3.46) with \(Z\), taking the cross-product with \(\hat{x}\), and adding the resultant equation to (3.45), we can obtain

\[ \hat{j} 2 E_{\text{inc}} = E + Z H \times \hat{x} \]  

(3.47)

where \((x = 0^-)\) has been omitted for simplicity. In this case, the unit vector normal to the element interface is simply \(\hat{n} = \hat{x}\), therefore

\[ \hat{j} 2 E_{\text{inc}} = E - \hat{n} \times Z H. \]  

(3.48)
By decomposing the waves in the neighboring element using the procedure described above, we can obtain

\[ \dot{\mathbf{2} E_{inc}^*} = \mathbf{E}^\ast - \mathbf{n}^\ast \times Z^\ast \mathbf{H}^\ast = \mathbf{E}^\ast + \mathbf{n} \times Z \mathbf{H}^* . \]  

(3.49)

Substituting (3.48) and (3.49) into (3.45) and (3.46) yields

\[ \mathbf{E} = \frac{1}{Z^* + Z} \left[ Z^* (\mathbf{E} - \mathbf{n} \times Z \mathbf{H}) + Z (\mathbf{E}^\ast + \mathbf{n} \times Z^\ast \mathbf{H}^\ast) \right] \]

\[ = \mathbf{E} + \frac{Z}{Z^* + Z} \left[ (\mathbf{E}^* - \mathbf{E}) + Z^* \mathbf{n} \times (\mathbf{H}^* - \mathbf{H}) \right] \]

\[ = \mathbf{E} + \frac{1}{Y^* + Y} \left[ Y^* (\mathbf{E}^* - \mathbf{E}) + \mathbf{n} \times (\mathbf{H}^* - \mathbf{H}) \right] \]  

(3.50)

\[ \mathbf{H} = \mathbf{n} \times \frac{1}{Z^* + Z} \left[ (\mathbf{E} - \mathbf{n} \times Z \mathbf{H}) - (\mathbf{E}^* + \mathbf{n} \times Z^\ast \mathbf{H}^\ast) \right] \]

\[ = \frac{1}{Z^* + Z} \left[ (\mathbf{n} \times \mathbf{E} + Z \mathbf{H}) - (\mathbf{n} \times \mathbf{E}^* - Z^\ast \mathbf{H}^\ast) \right] \]

\[ = \mathbf{H} + \frac{1}{Z^* + Z} \left[ -\mathbf{n} \times (\mathbf{E}^* - \mathbf{E}) + Z^* (\mathbf{H}^* - \mathbf{H}) \right] . \]  

(3.51)

By plugging the above two equations into the right-hand sides in (3.39) and (3.40) and applying the divergence theorem and the vector identity in the reverse order, we have

\[ \iint \limits_{V_r} \mathbf{T} \cdot \left( \epsilon \frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{H} \right) dV = \iint \limits_{S_r} \mathbf{T} \cdot \left\{ \mathbf{n} \times \frac{1}{Z^* + Z} \left[ -\mathbf{n} \times (\mathbf{E}^* - \mathbf{E}) + Z^* (\mathbf{H}^* - \mathbf{H}) \right] \right\} dS \]

\[ = \iint \limits_{S_r} \mathbf{T} \cdot \left( Z^* \left[ \mathbf{H} \right] - \mathbf{n} \times \left[ \mathbf{E} \right] \right) dS \]  

(3.52)

\[ \iint \limits_{V_r} \mathbf{T} \cdot \left( \mu \frac{\partial \mathbf{H}}{\partial t} + \nabla \times \mathbf{E} \right) dV = \iint \limits_{S_r} \mathbf{T} \cdot \left\{ \mathbf{n} \times \frac{1}{Y^* + Y} \left[ Y^* (\mathbf{E}^* - \mathbf{E}) + \mathbf{n} \times (\mathbf{H}^* - \mathbf{H}) \right] \right\} dS \]

\[ = \iint \limits_{S_r} \frac{1}{Y^* + Y} \mathbf{T} \cdot \left( Y^* \left[ \mathbf{E} \right] + \mathbf{n} \times \left[ \mathbf{H} \right] \right) dS \]  

(3.53)

where \( Z \) and \( Y \) denote the characteristic impedance and admittance of the material in the local element, \( Z^* \), \( Y^* \) are the corresponding values in the neighboring element, and the field jumps are defined as

\[ [\mathbf{E}] = \mathbf{n} \times (\mathbf{E}^* - \mathbf{E}) \]  

(3.54)

\[ [\mathbf{H}] = \mathbf{n} \times (\mathbf{H}^* - \mathbf{H}) . \]  

(3.55)
Again, let us consider three types of boundaries: the PEC surface $S_{\text{PEC}}$, the ABC surface $S_{\text{ABC}}$, and the elemental interface $S_{i}$. Notice that in the above equations we have considered elemental interfaces but not the other two types. The boundary condition at PEC surface can be strictly enforced by setting the electric field unknowns related to PEC to zero. It can also be weakly enforced by setting $Z^+ = 0$, $Y^+ = \infty$, and $E^+ = 0$, which physically means a short circuit. There is no need to worry about $H^+$ since all terms related to $H^+$ are zero. For an ABC surface we have

$$-\hat{n} \times H^+ + Y^+ \hat{n} \times \hat{n} \times E^+ = 0$$

(3.56)

$$\hat{n} \times E^+ + Z^+ \hat{n} \times \hat{n} \times H^+ = 0.$$  

(3.57)

Therefore, the surface integral related to $E^+$ exactly cancels that related to $H^+$ on the right-hand sides of (3.52) and (3.53) and we can simply set $Z^+ = Z$, $Y^+ = Y$, $E^+ = 0$, and $H^+ = 0$.

After expanding the electric and magnetic fields using basis functions as $E = \sum_j N_j e_j$ and $H = \sum_j N_j h_j$, we obtain the following matrix equations:

$$[M_e] \frac{\partial \{e\}}{\partial t} + [S_e] \{h\} = [F_{eh}] \{h^--h\} + [F_{ee}] \{e^+-e\}$$

(3.58)

$$[M_h] \frac{\partial \{h\}}{\partial t} + [S_h] \{e\} = [F_{he}] \{e^+-e\} + [F_{hh}] \{h^+-h\}$$

(3.59)

where $e$ and $h$ are the surface electric and magnetic field unknowns from the local element and $e^+$ and $h^+$ are the unknowns from the neighboring element. The matrix entries are given by

$$M_e(i,j) = \iiint_{V_i} \epsilon N_i \cdot N_j dV$$

(3.60)

$$S_e(i,j) = -\iiint_{V_i} N_i \cdot (\nabla \times N_j) dV$$

(3.61)

$$M_h(i,j) = \iiint_{V_i} \mu N_i \cdot N_j dV$$

(3.62)

$$S_h(i,j) = \iiint_{V_i} N_i \cdot (\nabla \times N_j) dV$$

(3.63)
To describe the temporal integration scheme, Eqs. (3.58) and (3.59) are first rewritten as

\[
\frac{\partial \{e\}}{\partial t} = \{\text{rhsE}\} = [M_e]^{-1} \left( -[S_e] \{h\} + [F_{ee}] \{h^r - h\} + [F_{eh}] \{e^r - e\} \right) \quad (3.68)
\]

\[
\frac{\partial \{h\}}{\partial t} = \{\text{rhsH}\} = [M_h]^{-1} \left( -[S_h] \{e\} + [F_{he}] \{e^r - e\} + [F_{hh}] \{h^r - h\} \right). \quad (3.69)
\]

By defining \( q = \begin{pmatrix} \{e\} \\ \{h\} \end{pmatrix} \) and \( F = \begin{pmatrix} \{\text{rhsE}\} \\ \{\text{rhsH}\} \end{pmatrix} \), the above can be cast into a compact form:

\[
\frac{dq}{dt} = F(t, q) \quad (3.70)
\]

which can be integrated in time using an \( s \)-stage Runge-Kutta method:

\[
q^{(i)} = q^n + \Delta t \sum_{j=1}^{s} a_j F \left( t^n + c_j \Delta t, q^{(i)} \right), \quad 1 \leq i \leq s
\]

\[
q^{n+1} = q^n + \Delta t \sum_{i=1}^{s} b_i F \left( t^n + c_i \Delta t, q^{(i)} \right) \quad (3.72)
\]

where the coefficients \( a_j \), \( b_j \), and \( c_j \) determine the accuracy and stability properties and can be written into the Butcher tableau [53]

<table>
<thead>
<tr>
<th>( c_j )</th>
<th>( a_j )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( b_j )</td>
<td></td>
</tr>
</tbody>
</table>

Fully explicit Runge-Kutta schemes (ERK schemes) have zeros on and above the main diagonal of the \( a_j \) matrix, or \( a_j = 0 \) for \( j \geq i \). It is observed that the \( s \)-stage Runge-Kutta
method generally requires the storage of $s$ intermediate $q^{(i)}$ vectors and one $q^n$ vector. In order to reduce the memory cost while maintaining a high accuracy, a low-storage five-stage fourth-order ERK scheme has been developed which only requires the storage of two vectors [54], [55]:

$$
\begin{align*}
    w_i &= \alpha_i w_{i-1} + \Delta t F(t_{i-1}, q^{(i-1)}), \\
    q^{(i)} &= q^{(i-1)} + \beta_i w_i \quad i = 1, 2, \ldots, 5
\end{align*}
$$

(3.73)

where $q^{(0)} = q^n$, $q^{n+1} = q^{(5)}$ and $t_{i-1} = t^n + \gamma_i \Delta t$. The values of $\alpha_i$, $\beta_i$ and $\gamma_i$ are given as:

- $\alpha_1 = 0.0000000000, \quad \beta_1 = 0.1496590219993, \quad \gamma_1 = 0.000000000000000,
- $\alpha_2 = -0.4178904745, \quad \beta_2 = 0.3792103129999, \quad \gamma_2 = 0.1496590219993,
- $\alpha_3 = -1.92151694643, \quad \beta_3 = 0.822950293869, \quad \gamma_3 = 0.370409573644,
- $\alpha_4 = -1.697784692471, \quad \beta_4 = 0.6994504559488, \quad \gamma_4 = 0.622557631345,
- $\alpha_5 = -1.514183444257, \quad \beta_5 = 0.1530572479681, \quad \gamma_5 = 0.9582821306748.$

Note that $\alpha_1 = 0$ and $\gamma_1 = 0$ so that the algorithm is self-starting.

Since the explicit DGTD scheme described above solves the system equations at the element level in a way similar to the explicit DFDD, the DGTD also avoids the need to invert a global system matrix and thus has a high computational efficiency. It is also perfectly suitable for parallel computation since it is easy to balance the computational load among different processors.

On the other hand, DFDD and DGTD are different from each other in the sense that DFDD solves two second-order wave equations and information is exchanged among elements through equivalent surface currents, while DGTD solves two first-order Maxwell’s equations and elements are coupled together by numerical fluxes. These differences result in different performances, as will be shown by numerical examples.

### 3.2.3 Explicit DGTD with Central Fluxes

In contrast to DGTD-Upwind, DGTD-Central uses the central fluxes and can be derived in two ways: the first approach is to take the average of tangential field components at the element interfaces [17] and the second approach is to enforce the
energy conservation law [51], as described below.

First approach: To weakly enforce the tangential field continuity \( \hat{n} \times \mathbf{H} = \hat{n} \times \mathbf{H}' \), an average of the tangential field is used on the right-hand side of (3.39):

\[
\hat{n} \times \mathbf{H} = \frac{1}{2} (\hat{n} \times \mathbf{H} + \hat{n} \times \mathbf{H}') = \frac{1}{2} (\hat{n} \times \mathbf{H} + \hat{n} \times \mathbf{H}')
\]  

(3.74)

and thus (3.39) becomes

\[
\iint_{V} \left[ \varepsilon \nabla \cdot \nabla \mathbf{E} - (\nabla \times \mathbf{T}) \cdot \mathbf{H} \right] dV = \frac{1}{2} \oint_{S} \mathbf{T} \cdot (\hat{n} \times \mathbf{H} + \hat{n} \times \mathbf{H}') dS.
\]

(3.75)

By applying the same vector identity and the divergence theorem in the reverse order, the above equation becomes

\[
\iint_{V} \mathbf{T} \cdot \left( \varepsilon \frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{H} \right) dV = \frac{1}{2} \oint_{S} \mathbf{T} \cdot \hat{n} \times (\mathbf{H}' - \mathbf{H}) dS = \frac{1}{2} \oint_{S} \mathbf{T} \cdot \mathbf{H} dS.
\]

(3.76)

A similar procedure can be applied to (3.40) to obtain the other system equation

\[
\iint_{V} \mathbf{T} \cdot \left( \mu \frac{\partial \mathbf{H}}{\partial t} + \nabla \times \mathbf{E} \right) dV = -\frac{1}{2} \oint_{S} \mathbf{T} \cdot \hat{n} \times (\mathbf{E}' - \mathbf{E}) dS = -\frac{1}{2} \oint_{S} \mathbf{T} \cdot \mathbf{E} dS
\]

(3.77)

which can be discretized in space and time.

Second approach: Penalty terms are directly added to the right-hand sides of (3.37) and (3.38) to weakly enforce the tangential field continuities, yielding

\[
\iint_{V} \mathbf{T} \cdot \left( \varepsilon \frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{H} \right) dV = a \oint_{S} \mathbf{T} \cdot \mathbf{H} dS + b \oint_{S} \mathbf{T} \cdot \hat{n} \times \mathbf{E} dS
\]

(3.78)

\[
\iint_{V} \mathbf{T} \cdot \left( \mu \frac{\partial \mathbf{H}}{\partial t} + \nabla \times \mathbf{E} \right) dV = c \oint_{S} \mathbf{T} \cdot \mathbf{E} dS + d \oint_{S} \mathbf{T} \cdot \hat{n} \times \mathbf{H} dS
\]

(3.79)

where \( a, b, c, \) and \( d \) are coefficients to be determined. If \( \mathbf{E} \) and \( \mathbf{H} \) are used as testing functions in (3.78) and (3.79), respectively, the two equations become

\[
\iint_{V} \mathbf{E} \cdot \left( \varepsilon \frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{H} \right) dV = a \oint_{S} \mathbf{E} \cdot \mathbf{H} dS + b \oint_{S} \mathbf{E} \cdot \hat{n} \times \mathbf{E} dS
\]

(3.80)
\[
\iint_{V'} \left( \varepsilon \frac{\partial \mathbf{E}}{\partial t} + \frac{\partial \mathbf{H}}{\partial t} \right) \, dV + \iint_{S'} \varepsilon \mathbf{E} \cdot \hat{n} \, dS = \iint_{V'} (\mu \frac{\partial \mathbf{H}}{\partial t} + \nabla \times \mathbf{E}) \, dV + \iint_{S'} \mu \mathbf{H} \cdot \hat{n} \times \mathbf{H} \, dS.
\]

(3.81)

By adding the above two equations over all elements in the computational domain \( \Omega \) and integrating by parts, we obtain

\[
\sum_{\epsilon \in \Omega} \iint_{V'} \left( \varepsilon \mathbf{E} \cdot \frac{\partial \mathbf{E}}{\partial t} + \mu \mathbf{H} \cdot \frac{\partial \mathbf{H}}{\partial t} \right) \, dV
\]

\[
= \sum_{\epsilon \in \Omega} \iint_{V'} (1-a+c)\mathbf{E} \cdot (\hat{n} \times \mathbf{H}) \, dV + \sum_{S_{\epsilon \subset \partial \Omega}} \iint_{S_{\epsilon \subset \partial \Omega}} (a+c) \left[ \mathbf{E} \cdot (\hat{n} \times \mathbf{H}^+) + (\hat{n} \times \mathbf{E}^+) \cdot \mathbf{H} \right] \, dS
\]

\[
+ \sum_{S_{\epsilon \subset \partial \Omega}} \iint_{S_{\epsilon \subset \partial \Omega}} b \left[ (\hat{n} \times (\mathbf{E}^+ - \mathbf{E})) \cdot (\hat{n} \times (\mathbf{E}^+ - \mathbf{E})) \right] \, dS + \sum_{S_{\epsilon \subset \partial \Omega}} \iint_{S_{\epsilon \subset \partial \Omega}} d \left[ (\hat{n} \times (\mathbf{H}^+ - \mathbf{H}) \cdot (\hat{n} \times (\mathbf{H}^+ - \mathbf{H})) \right] \, dS
\]

(3.82)

where \( \partial \Omega \) represents the boundary of the computational domain. Note that the left-hand side in the above is the time derivative of the electromagnetic energy

\[
\frac{1}{2} \sum_{\epsilon \in \Omega} \iint_{V'} (\varepsilon \mathbf{E} \cdot \mathbf{E} + \mu \mathbf{H} \cdot \mathbf{H}) \, dV.
\]

To enforce the energy conservation law, the left-hand side must be zero all the time; therefore, the right-hand side also has to be zero all the time, which requires \( 1-a+c = 0 \), \( b = d = 0 \), and \( a+c = 0 \), or \( a = \frac{1}{2} \), \( b = 0 \), \( c = -\frac{1}{2} \), and \( d = 0 \). Therefore, Eqs. (3.78) and (3.79) are reduced to Eqs. (3.76) and (3.77).

Notice that in the derivations above only the boundary condition at element interfaces has been considered. The PEC boundary condition can be either strictly enforced by setting the electric unknowns related to PEC to zero, or weakly enforced by setting

\[
\hat{n} \times \mathbf{E}^+ = -\hat{n} \times \mathbf{E}
\]

(3.83)

\[
\hat{n} \times \mathbf{H}^+ = \hat{n} \times \mathbf{H}
\]

(3.84)

which physically represents the image theory. The ABC can be enforced by setting

\[
\hat{n} \times \mathbf{H}^+ = Y\hat{n} \times \hat{n} \times \mathbf{E}
\]

(3.85)

\[
\hat{n} \times \mathbf{E}^+ = -Z\hat{n} \times \hat{n} \times \mathbf{H}.
\]

(3.86)

Therefore, the surface integrals related to ABC become
After applying the ABC and performing the spatial discretization, the following matrix
equations are obtained:

\[
\frac{1}{2} \iint_{S_{abc}} T \cdot [H] dS = \frac{1}{2} \iint_{S_{abc}} T \cdot \hat{n} \times (H - H) dS
\]

\[
= \frac{1}{2} \iint_{S_{abc}} T \cdot Y \hat{n} \times \hat{E} dS - \frac{1}{2} \iint_{S_{abc}} T \cdot \hat{n} \times H dS
\]

\[
= - \frac{Y}{2} \iint_{S_{abc}} \hat{E} \cdot \hat{n} \times \hat{E} dS - \frac{1}{2} \iint_{S_{abc}} T \cdot \hat{n} \times H dS
\]

(3.87)

\[- \frac{1}{2} \iint_{S_{abc}} T \cdot [E] dS = - \frac{1}{2} \iint_{S_{abc}} T \cdot \hat{n} \times (E - E) dS
\]

\[
= \frac{1}{2} \iint_{S_{abc}} T \cdot Z \hat{n} \times H dS + \frac{1}{2} \iint_{S_{abc}} T \cdot \hat{n} \times E dS
\]

\[
= - \frac{Z}{2} \iint_{S_{abc}} \hat{E} \cdot \hat{n} \times \hat{E} dS + \frac{1}{2} \iint_{S_{abc}} T \cdot \hat{n} \times E dS
\]

(3.88)

where \([M_e], [S_e], [M_h], \) and \([S_h]\) are the same as those in DGTD-Upwind and are
given in (3.60)-(3.63) and the other matrices are given by

\[
[M_e] \frac{\partial \{e\}}{\partial t} + [S_e] \{h\} = [F_{eh}] \{h^* - h\} + [A_{eh}] \{h\} + [A_{ee}] \{e\}
\]

(3.89)

\[
[M_h] \frac{\partial \{h\}}{\partial t} + [S_h] \{e\} = [F_{he}] \{e^* - e\} + [A_{he}] \{e\} + [A_{eh}] \{h\}
\]

(3.90)

where \([M_e], [S_e], [M_h], \) and \([S_h]\) are the same as those in DGTD-Upwind and are
given in (3.60)-(3.63) and the other matrices are given by

\[
F_{eh} (i, j) = \frac{1}{2} \iint_{S_i} (\hat{n} \times N_i) \cdot (\hat{n} \times \hat{n} \times N_j) dS
\]

(3.91)

\[
A_{eh} (i, j) = - \frac{1}{2} \iint_{S_{abc}} (\hat{n} \times N_i) \cdot (\hat{n} \times \hat{n} \times N_j) dS
\]

(3.92)

\[
A_{ee} (i, j) = - \frac{Y}{2} \iint_{S_{abc}} (\hat{n} \times N_i) \cdot (\hat{n} \times \hat{n} \times N_j) dS
\]

(3.93)

\[
F_{he} (i, j) = - \frac{1}{2} \iint_{S_i} (\hat{n} \times N_i) \cdot (\hat{n} \times \hat{n} \times N_j) dS
\]

(3.94)

\[
A_{he} (i, j) = \frac{1}{2} \iint_{S_{abc}} (\hat{n} \times N_i) \cdot (\hat{n} \times \hat{n} \times N_j) dS
\]

(3.95)
\[ A_{\text{th}}(i, j) = -\frac{Z}{2} \iint_{s_{\text{acc}}} (\hat{n} \times \mathbf{N}_i) \cdot (\hat{n} \times \mathbf{N}_j) dS . \quad (3.96) \]

For the temporal discretization of (3.89), the central difference is used for \( \frac{\partial \{\mathbf{e}\}}{\partial t} \) and an average is used for \( \{\mathbf{e}\} \), yielding

\[
\left[ M_e \right] \frac{\{\mathbf{e}\}^{n+1} - \{\mathbf{e}\}^n}{\Delta t} + \left[ S_e \right] \{h\}^{n+1/2} = \left[ F_{eh} \right] \{h^* - h\}^{n+1/2} + \left[ A_{eh} \right] \{h\}^{n+1/2} + \left[ A_{ve} \right] \frac{\{\mathbf{e}\}^{n+1} + \{\mathbf{e}\}^n}{2} \quad (3.97)
\]

or

\[
\left( \frac{1}{\Delta t} [M_e] - \frac{1}{2} [A_{ve}] \right) \{\mathbf{e}\}^{n+1} = \left( [A_{eh}] - [S_e] - [F_{eh}] \right) \{h\}^{n+1/2} + \left[ F_{eh} \right] \{h^* \}^{n+1/2} + \left( \frac{1}{\Delta t} [M_e] + \frac{1}{2} [A_{ve}] \right) \{\mathbf{e}\}^n. \quad (3.98)
\]

Similarly, the temporal discretization of (3.90) yields

\[
\left[ M_h \right] \frac{\{h\}^{n+3/2} - \{h\}^{n+1/2}}{\Delta t} + \left[ S_h \right] \{\mathbf{e}\}^{n+1} = \left[ F_{he} \right] \{e^* - e\}^{n+1} + \left[ A_{he} \right] \{e\}^{n+1} + \left[ A_{ha} \right] \frac{\{h\}^{n+3/2} + \{h\}^{n+1/2}}{2} \quad (3.99)
\]

or

\[
\left( \frac{1}{\Delta t} [M_h] - \frac{1}{2} [A_{ha}] \right) \{h\}^{n+3/2} = \left( [A_{he}] - [S_h] - [F_{he}] \right) \{e\}^{n+1} + \left[ F_{he} \right] \{e^* \}^{n+1} + \left( \frac{1}{\Delta t} [M_h] + \frac{1}{2} [A_{ha}] \right) \{h\}^{n+1/2}. \quad (3.100)
\]

Equations (3.98) and (3.100) are updated using the leapfrog time marching:

\( \{\mathbf{e}\}^n \rightarrow \{h\}^{n+1/2} \rightarrow \{\mathbf{e}\}^{n+1} \rightarrow \{h\}^{n+3/2} \).

In contrast to the DGTD-Upwind scheme, the DGTD-Central scheme employs central fluxes instead of upwind fluxes. The DGTD-Upwind scheme is advantageous in the sense that it has an optimal convergence rate, but it is numerically dissipative. In contrast, the DGTD-Central scheme conserves a discrete form of the electromagnetic energy, but the convergence rate is suboptimal. This comparison will become more
obvious with the numerical examples.

### 3.2.4 Hybrid Implicit-Explicit DFDD

The construction of a hybrid implicit-explicit DFDD is fairly straightforward; it is actually already implied in [11] and realized in [12]. To form a hybrid scheme, one simply needs to group the implicit elements into one or a few subdomains and treat each explicit element as a single subdomain. The implicit and explicit regions communicate with each other through exchanging equivalent surface currents. The equations to be solved for the implicit subdomains are

\[
\iiint_{V_i} \left[ \frac{1}{\mu_r} (\nabla \times \mathbf{T}) \cdot (\nabla \times \mathbf{E}) + \frac{1}{c_0^2} \varepsilon_r \mathbf{T} \cdot \frac{\partial^2 \mathbf{E}}{\partial t^2} \right] dV + \mu_0 \iiint_{S_{\text{inc}}} Y(\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{E}}{\partial t} \right) dS = \mu_0 \iiint_{S_i} (\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{J}}{\partial t} \right) dS 
\]

(3.101)

\[
\iiint_{V_i} \left[ \frac{1}{\varepsilon_r} (\nabla \times \mathbf{T}) \cdot (\nabla \times \mathbf{H}) + \frac{1}{c_0^2} \mu_r \mathbf{T} \cdot \frac{\partial^2 \mathbf{H}}{\partial t^2} \right] dV + \mu_0 \iiint_{S_{\text{inc}}} Z(\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{H}}{\partial t} \right) dS = \varepsilon_0 \iiint_{S_i} (\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{M}}{\partial t} \right) dS
\]

(3.102)

and there are only two changes from (3.13) and (3.14) to (3.101) and (3.102): one is that the volume to perform integration is changed from the element volume \(V_e\) to the subdomain volume \(V_s\), and the other is that the surface currents are exchanged at the subdomain interfaces instead of at the element interfaces.

### 3.2.5 Hybrid Implicit-Explicit DGTD

The construction of a hybrid DGTD is more involved than that of the hybrid DFDD. There are basically two hybrid DGTD schemes: one using an explicit singly diagonally implicit Runge-Kutta (ESDIRK) method for the implicit region and an explicit Runge-Kutta (ERK) method for the explicit region [21]-[23] and the other using the Crank-Nicolson method for the implicit region and a modified leapfrog algorithm (named the Verlet method) for the explicit region [24], [25]. The latter is implemented and
investigated in this work because it has a proven energy-conservation property and it employs the leapfrog time marching which is similar to that in the DFDD algorithm. The formulation starts from rewriting (3.76) and (3.77) as

\[
\iiint_{V} T \cdot \varepsilon \frac{\partial E}{\partial t} dV = \frac{1}{2} \iiint_{V} (T \cdot \nabla \times H + \nabla \times T \cdot H) dV + \frac{1}{2} \iiint_{S} T \cdot \hat{n} \times H^* dS \tag{3.103}
\]

\[
\iiint_{V} T \cdot \mu \frac{\partial H}{\partial t} dV = -\frac{1}{2} \iiint_{V} (T \cdot \nabla \times E + \nabla \times T \cdot E) dV - \frac{1}{2} \iiint_{S} T \cdot \hat{n} \times E^* dS. \tag{3.104}
\]

By expanding the electric and magnetic fields using vector basis functions as \( E = \sum_{j} N_j e_j \) and \( H = \sum_{j} N_j h_j \), the two equations above can be cast into the semi-discrete form:

\[
[M_e] \frac{\partial \{ e \}}{\partial t} = [S_e] \{ h \} + [F_{eh}] \{ h^* \} \tag{3.105}
\]

\[
[M_h] \frac{\partial \{ h \}}{\partial t} = [S_h] \{ e \} + [F_{eh}] \{ e^* \} \tag{3.106}
\]

where we have omitted the ABC for simplicity and

\[
M_e(i, j) = \iiint_{V} e N_i \cdot N_j dV \tag{3.107}
\]

\[
S_e(i, j) = \frac{1}{2} \iiint_{V} \left[ N_j \cdot (\nabla \times N_j) + (\nabla \times N_j) \cdot N_j \right] dV \tag{3.108}
\]

\[
F_{eh}(i, j) = \frac{1}{2} \iiint_{S} (\hat{n} \times N_j) \cdot (\hat{n} \times \hat{n} \times N_j) dS \tag{3.109}
\]

\[
M_h(i, j) = \iiint_{V} \mu N_i \cdot N_j dV \tag{3.110}
\]

\[
S_h(i, j) = -\frac{1}{2} \iiint_{V} \left[ N_j \cdot (\nabla \times N_j) + (\nabla \times N_j) \cdot N_j \right] dV \tag{3.111}
\]

\[
F_{eh}(i, j) = -\frac{1}{2} \iiint_{S} (\hat{n} \times N_j) \cdot (\hat{n} \times \hat{n} \times N_j) dS. \tag{3.112}
\]
Note that (3.105) and (3.106) are obtained at the element level. To construct the hybrid implicit-explicit algorithm, these two equations from each element are assembled into two global matrix equations:

\[
\begin{align*}
\left[ M^e \right] \frac{d\{e\}}{dt} &= [S]\{h\} \\
\left[ M^\mu \right] \frac{d\{h\}}{dt} &= -[S]\{e\}
\end{align*}
\]

where \( \left[ M^e \right] \), \( \left[ M^\mu \right] \), and \( [S] \) have the size \( n_{tot} \times n_{tot} \), and the length of vectors \( \{e\} \) and \( \{h\} \) have been augmented to \( n_{tot} \) (although their notations remain unchanged), where \( n_{tot} \) denotes the total number of degrees of freedom in the entire computational domain. If the same basis function order is used everywhere, we have \( n_{tot} = n_{ele} n_{loc} \) with \( n_{ele} \) being the total number of finite elements and \( n_{loc} \) being the number of degrees of freedom in one element. Besides, \( \left[ M^e \right] \) and \( \left[ M^\mu \right] \) are symmetric block-diagonal matrices with a block size \( n_{loc} \). Notice that the \( [S] \) in (3.114) is changed to its transpose in [25] due to the weak enforcement of the PEC boundary condition, while in our implementation there is no such change since we enforce the PEC boundary condition in the strict way. Equations (3.113) and (3.114) are still the general DGTD formulas and can be applied to both the explicit and implicit regions.

For the fully explicit DGTD, the leapfrog time marching scheme can be used to discretize (3.113) and (3.114) in time, yielding

\[
\begin{align*}
\left[ M^e \right]\{e\}^{n+1} - \{e\}^{n} &= [S]\{h\}^{n+1/2} \\
\left[ M^\mu \right]\{h\}^{n+3/2} - \{h\}^{n+1/2} &= -[S]\{e\}^{n+1}.
\end{align*}
\]

For the fully implicit DGTD, a second-order Crank-Nicolson scheme can be used to obtain an unconditionally stable system:

\[
\begin{align*}
\left[ M^e \right]\{e\}^{n+1} - \{e\}^{n} &= [S]\frac{\{h\}^{n+1} + \{h\}^{n}}{2} \\
\left[ M^\mu \right]\{h\}^{n+1} - \{h\}^{n} &= -[S]\frac{\{e\}^{n+1} + \{e\}^{n}}{2}.
\end{align*}
\]
\[
\left[ M^\mu \right] \frac{\{h^+\}_{n+1} - \{h^-\}_n}{\Delta t} = -\left[ S \right] \frac{\{e^+\}_{n+1} + \{e^-\}_n}{2}.
\] (3.118)

To construct the hybrid implicit-explicit DGTD, a coupling term has to be added to the right-hand sides in (3.113) and (3.114) to include the interaction between the explicit and implicit regions, yielding

\[
\left[ M^\mu \right] \frac{d\{e\}_{ex}}{dt} = \left[ S \right] \{e\}_{ex} - \left[ A_{ex,im} \right] \{h\}_{im} \] (3.119)
\[
\left[ M^\mu \right] \frac{d\{h\}_{ex}}{dt} = -\left[ S \right] \{e\}_{ex} + \left[ A_{ex,im} \right] \{e\}_{im} \] (3.120)
\[
\left[ M^\mu \right] \frac{d\{e\}_{im}}{dt} = \left[ S \right] \{h\}_{im} - \left[ A_{im,ex} \right] \{e\}_{ex} \] (3.121)
\[
\left[ M^\mu \right] \frac{d\{h\}_{im}}{dt} = -\left[ S \right] \{e\}_{im} + \left[ A_{im,ex} \right] \{e\}_{ex} \] (3.122)

where the subscripts “ex” and “im” denote the explicit and implicit regions, respectively, and the coupling matrix \( \left[ A_{ex,im} \right] \) can be computed as

\[
A_{ex,im}(j,k) = -\frac{1}{2} \int_{S_e} (\hat{n} \times N_{ex,j}) \cdot (\hat{n} \times \hat{n} \times N_{im,k}) dS
\] (3.123)

where \( N_{ex,j} \) is the testing function in the explicit region and \( N_{im,k} \) is the basis function expanding the fields in the implicit region. In the implementation, (3.123) is first computed for each element and then assembled to the \((j, k)\)-th place of \( \left[ A_{ex,im} \right] \). If the same basis function order is used throughout the computational domain, \( j \) and \( k \) can be determined as \( j = (idx_{ele} - 1)n_{loc} + j_{loc} \) and \( k = (idx_{ele} - 1)n_{loc} + k_{loc} \), where \( idx_{ele} \) is the index of the element under consideration, \( j_{loc} \) is the local index of the testing function \( N_{ex,j} \), and \( k_{loc} \) is the local index of the basis function that coincides with \( N_{im,k} \) at the element interface. Similarly, the coupling matrix \( \left[ A_{im,ex} \right] \) can be calculated as

\[
A_{im,ex}(j,k) = -\frac{1}{2} \int_{S_e} (\hat{n} \times N_{im,k}) \cdot (\hat{n} \times \hat{n} \times N_{ex,j}) dS.
\] (3.124)
For temporal discretization, the aforementioned Crank-Nicolson scheme is used for the implicit region and the Verlet method is used for the explicit region. In the resultant hybrid implicit-explicit algorithm, the following equations are solved to march the fields from time step \( n \) to \( n+1 \):

\[
\begin{align*}
\left[ M^\mu_{ex} \right] & \left\{ \frac{h_{ex}^{n+1/2} - h_{ex}^n}{\Delta t / 2} \right\} = -\left[ S_{ex} \right] \left\{ e_{ex}^n \right\} + \left[ A_{ex,im} \right] \left\{ e_{im}^n \right\} \\
\left[ M^e_{ex} \right] & \left\{ \frac{e_{ex}^{n+1/2} - e_{ex}^n}{\Delta t / 2} \right\} = \left[ S_{ex} \right] \left\{ h_{ex}^{n+1/2} \right\} - \left[ A_{ex,im} \right] \left\{ h_{im}^n \right\} \\
\left[ M^e_{im} \right] & \left\{ \frac{e_{im}^{n+1} - e_{im}^n}{\Delta t} \right\} = \frac{1}{2} \left[ S_{im} \right] \left\{ h_{im}^{n+1} \right\} + \left[ h_{im}^n \right] - \left[ A_{im,ex} \right] \left\{ h_{ex}^{n+1/2} \right\} \\
\left[ M^e_{im} \right] & \left\{ \frac{h_{im}^{n+1} - h_{im}^n}{\Delta t} \right\} = -\frac{1}{2} \left[ S_{im} \right] \left\{ e_{im}^{n+1} \right\} + \left[ e_{im}^n \right] + \left[ A_{im,ex} \right] \left\{ e_{ex}^{n+1/2} \right\} \\
\left[ M^e_{ex} \right] & \left\{ \frac{e_{ex}^{n+1} - e_{ex}^n}{\Delta t / 2} \right\} = \left[ S_{ex} \right] \left\{ h_{ex}^{n+1/2} \right\} - \left[ A_{ex,im} \right] \left\{ h_{im}^{n+1} \right\} \\
\left[ M^e_{ex} \right] & \left\{ \frac{h_{ex}^{n+1} - h_{ex}^n}{\Delta t / 2} \right\} = -\left[ S_{ex} \right] \left\{ e_{ex}^{n+1} \right\} + \left[ A_{ex,im} \right] \left\{ e_{im}^{n+1} \right\}.
\end{align*}
\]

The solutions are performed in the following order: (1) Equation (3.125) is solved for \( \left\{ h_{ex}^{n+1/2} \right\} \); (2) Equation (3.126) is solved for \( \left\{ e_{ex}^{n+1/2} \right\} \); (3) Equations (3.127) and (3.128) are solved together for \( \left\{ e_{im}^{n+1} \right\} \) and \( \left\{ h_{im}^{n+1} \right\} \); (4) Equation (3.129) is solved for \( \left\{ e_{ex}^{n+1} \right\} \); (5) Equation (3.130) is solved for \( \left\{ h_{ex}^{n+1} \right\} \). The above five solutions are repeated for every time step. Note that \( \left[ M^\mu_{ex} \right] \) and \( \left[ M^e_{ex} \right] \) are both block-diagonal matrices and the unknowns in Eqs. (3.125), (3.126), (3.129), and (3.130) only exist on the left-hand side; therefore, each of these four global matrix equations can be decomposed into \( n_{ele} \) decoupled local matrix equations of the size \( n_{loc} \times n_{loc} \) and solved independently for each element in the explicit region. For Eqs. (3.127) and (3.128), the unknowns \( \left\{ e_{im}^{n+1} \right\} \) and \( \left\{ h_{im}^{n+1} \right\} \) exist at both sides; since \( \left[ S_{im} \right] \) is not block-diagonal due to the existence of the numerical flux term, the two equations can no longer be decomposed into a number of
decoupled smaller equations, which means a global matrix equation of size $2n_{tot} \times 2n_{tot}$ has to be solved to obtain the global unknown vector $\left[ e_{im} \right]_{m+1}$.

### 3.3 Numerical Examples

In this section, several numerical examples are presented to validate our implementation of the methods described in Section 3.2 and to perform a comparative study of them. High-order hierarchical basis functions are used and all examples are calculated on an SGI Altix 350 system using an Intel Itanium II 1.5 GHz processor.

#### 3.3.1 Parallel-Plate Waveguide

To compare the accuracy of DFDD-ELD, DGTD-Upwind, and DGTD-Central, an empty parallel-plate waveguide is considered. The computational domain is a 1 m $\times$ 1 m $\times$ 1 m square box with PEC walls on the top and the bottom and PMC walls on the left and the right. The exact WPBC is applied to the remaining two surfaces. The numerical electric field $E$ is compared with the analytical solution $E_a$ and the RMS errors are computed for the three methods as a function of the mesh density and the order of basis functions. The convergence curves using mixed-order basis functions are shown in Fig. 3.1(a), where $\lambda_{\text{min}}$ denotes the wavelength corresponding to the highest frequency of interest and $h_{\mu}$ equals the average element edge length $h$ divided by the order of basis functions (1, 2, or 3). It is observed that the accuracy of DFDD-ELD and DGTD-Central are on the same order. On the other hand, DGTD-Upwind is obviously more accurate than the other two methods. The FETD result is also plotted in the Fig. 3.1(a) as a reference. Another convergence test is performed using full-order basis functions and the result is shown in Fig. 3.1(b). This time, DFDD-ELD is more accurate than DGTD-Central, and DGTD-Upwind still outperforms the other two methods. From this example, it can be concluded that the accuracy of the three explicit methods orders as DGTD-Upwind $>$ DFDD-ELD $\geq$ DGTD-Central. In Figs. 3.1(a) and 3.1(b), the second-order and third-order results have been shifted downward by 10 dB and 20 dB, respectively.
To perform the comparison in a more systematic way, the convergence rates of the three methods are fitted to the form $O(h^p)$ and $q$ is compared to the order of basis functions, $p$, in Table 3.1. Here only full-order basis functions are used in order to compare with some literature employing nodal basis functions where no mixed orders are defined. It is observed that the convergence rates of DFDD-ELD, DGTD-Upwind, and DGTD-Central are approximately $O(h^p)$, $O(h^{p+1})$, and $O(h^p)$. The convergence rates of DGTD-Upwind and DGTD-Central agree with the results in [14]-[20].

To compare the efficiency of the three explicit methods, the CPU time is plotted against the basis function orders in Fig. 3.1(c) where two different meshes have been used: $h = 0.25$ m and $h = 0.125$ m. It is observed that the CPU time for DFDD-ELD and DGTD-Central are almost the same, while DGTD-Upwind consumes two times more time than the other two methods. Note that for DGTD-Upwind a higher-order time marching scheme is employed while for the other two methods the simple leapfrog time marching is used. DGTD-Upwind may also employ a second-order time integration scheme such as the second-order explicit Runge-Kutta (ERK) scheme, but then the time step size has to be reduced in order to guarantee stability because the stability region of lower-order ERK schemes is smaller than that of the higher-order ones, according to [22].

Since DGTD-Upwind yields a smaller error by consuming more time, it would be interesting to explore which method is more accurate under a specified CPU time. For this purpose, the error versus the CPU time is shown in Figs. 3.1(d)-(f). Figs. 3.1(d) and 3.1(e) illustrate the case where the $h$-adaptation is performed (gradually refine the mesh until convergence) and along each line the three data points represent $h = 0.5$, 0.25, and 0.125 m, respectively. Notice that the mixed and full second-order results are shifted downward by 5 dB and the mixed and full third-order results are shifted downward by 20 dB. It can be observed that DFDD-ELD performs better than DGTD-Central, and DGTD-Upwind shows the best performance. Fig. 3.1(f) shows the case where the $p$-adaptation is used (gradually increase the basis function order until convergence) and along each line the six data points represent $p = 0.5$, 1.0, 1.5, 2.0, 2.5, and 3.0, respectively. Two meshes of different resolutions are tested and along each line the order of basis functions are
increased. The comparison shows that the three algorithms have about the same performance.

### 3.3.2 Monopole Antenna

To perform a more complete comparison of the three methods, a monopole antenna radiating above an infinitely large ground plane is simulated. The structure is excited using a short segment of coaxial waveguide whose inner and outer radii are 0.02 m and 0.05 m. The inner conductor is extended by a length of 0.3 m above the ground plane to form the monopole and the outer conductor is connected to the ground plane. The WPBC in [47] is used to terminate the coaxial port and the first-order ABC is used to truncate the radiation boundary. The mesh density around the port is 0.01 m and that on the ABC surface is 0.1 m and mixed second-order basis functions are employed, resulting in totally 177,534 unknowns. The three algorithms are compared with the traditional, fully implicit FETD. The total CPU times for FETD, explicit DFDD, explicit DGTD-Upwind, and explicit DGTD-Central are $2.11 \times 10^3$ s, $1.51 \times 10^4$ s, $4.08 \times 10^4$ s and $1.02 \times 10^4$ s while the memory costs are 2.4 GB, 1.0 GB, 1.4 GB and 1.3 GB, respectively. For each method, the maximum allowable time step has been used. The computational time of the three explicit methods is much longer than that of FETD because they are conditionally stable and the time step size is limited by the fine mesh on the port, while FETD is unconditionally stable due to the Newmark-Beta time integration. On the other hand, the explicit methods save memory compared to FETD since they do not require the factorization of a global system matrix and the storage of its LU decomposition. The magnitude of the reflection coefficient extracted on the port is plotted in Fig. 3.2(a) and perfect agreement among the four methods is observed, demonstrating the correctness of our implementation of the three explicit methods. The far field pattern is shown in Fig. 3.2(b) for both $E_\theta$ polarization and $E_\phi$ polarization. For $E_\theta$ polarization the result of the four methods overlay on each other. It is noticed that the far field of $E_\phi$ polarization is negligible compared to that of $E_\theta$ polarization since in this specific case the electric far field is indeed in the $\theta$ direction and has no $\phi$ components.
3.3.3 Dielectric Ring in PEC Cavity

To investigate the energy-conservation property of the three explicit methods, a cavity structure described in [56] and depicted in Fig. 3.3(a) is modeled and simulated. This structure consists of a dielectric ring in a PEC cavity, and an infinitesimal dipole is used as the excitation. The voltage on the dipole against time is plotted in Fig. 3.3(b) and it is clearly observed that DGTD-Upwind has a numerical loss while DFDD-ELD and DGTD-Central do not. On the other hand, the numerical loss in DGTD-Upwind can be reduced by employing higher-order basis functions, as is shown in Fig. 3.3(c). In the calculation generating Fig. 3.3(a), a total of 1,262 tetrahedrons are used with mixed second-order basis functions. The total number of unknowns in FETD is 9,242 and those in the three explicit methods are 25,240 for the same mesh. The resonant wave numbers of the first four modes named \( k_{01-k04} \) are listed in Table 3.2 where our results are compared with those in [56] and a good agreement is observed. In the calculation of resonant wave numbers, the same meshes with full second-order basis functions are used and the total number of global unknowns in FETD is 13,863 while those in the three explicit methods are 37,860.

3.3.4 Microstrip Patch Array

To further compare the three explicit algorithms, a 2×2 microstrip patch array in [11] is modeled and simulated. The structure is depicted in Fig. 3.4(a) and is excited by coaxial waveguide ports which are terminated by the WPBC. The open region is truncated by the first-order ABC. The element size on the port is 0.3 mm and that on the ABC is 10 mm. There are four ports in total and the magnitudes of scattering parameters are plotted in Fig. 3.4(b) where the reference solution is provided by a finite-element frequency-domain (FEFD) code and in all three methods, mixed second-order basis functions are used for the most of the computational domain while full-first order basis functions are used for the port regions where a denser mesh is employed. There are 42,965 tetrahedrons and 258,778 unknowns in total for FETD, and 784,588 unknowns for the three explicit algorithms. The total CPU times for DFDD-ELD, DGTD-Upwind, and DGTD-Central are \( 8.02 \times 10^4 \) s, \( 2.75 \times 10^5 \) s, and \( 8.22 \times 10^4 \) s, respectively; and the maximum allowable time
step has been used in each calculation. The memory costs for the three methods are 1.8 GB, 2.0 GB, and 1.8 GB, respectively. It is observed that for the same spatial discretization, DFDD-ELD and DGTD-Central consume a similar amount of time while DGTD-Upwind consumes two times more time than the other two methods. Note that the three methods have about the same memory efficiency.

3.3.5 Microstrip Patch Antenna

This example is created to illustrate the advantage of the hybrid implicit-explicit schemes and to compare the hybrid DFDD with the hybrid DGTD. The geometry is simply a patch antenna taken from the previous example and the patch size, the substrate thickness, and the substrate dielectric constant all remain the same. The difference is that the inner and outer radii of the feeding coaxial line are reduced by a factor of 5 to 0.096 mm and 0.3 mm, respectively. There are 49,907 edges in total and mixed first-order basis functions are used. A mesh size of 0.05 mm is used around the port to resolve the fine feeding structure, yielding a highly non-uniform mesh. In the hybrid methods, the small elements around the coaxial port formed an implicit region and the element size on the implicit-explicit interface is uniformly 1 mm. The results from the explicit DFDD (DFDD-ELD), explicit DGTD, hybrid DFDD, and hybrid DGTD are compared in Fig. 3.5 where the four curves overlay on each other, demonstrating the accuracy of the hybrid methods. The performances of the four schemes are compared in Table 3.3, which shows that the speeds of DFDD and DGTD are improved by a factor of 25 and 18 by employing the hybrid algorithms, at the cost of a larger memory usage. The explicit DFDD consumes slightly more CPU time and slightly less memory compared to the explicit DGTD, while the hybrid DFDD outperforms the hybrid DGTD in terms of both CPU time and memory, which is expected since DGTD yields more unknowns than DFDD in the implicit region due to the fact that one edge yields only one unknown for DFDD while it yields several unknowns (one in each neighboring element) for DGTD.
3.4 Figures and Tables

Fig. 3.1: (a) Error versus spatial discretization using mixed-order basis functions. The second-order and third-order results have been shifted downward by 10 dB and 20 dB, respectively. (b) Error versus spatial discretization using full-order basis functions. The second-order and third-order results have been shifted downward by 10 dB and 20 dB, respectively. (c) CPU time versus the order of basis functions. (d) Error versus CPU time for the $h$-adaptation using mixed orders ($h = 0.5, 0.25, \text{and} 0.125 \text{ m}$). The second-order and third-order results have been shifted downward by 5 dB and 20 dB, respectively. (e) Error versus CPU time for the $h$-adaptation using full orders. The second-order and third-order results have been shifted downward by 5 dB and 20 dB, respectively. (f) Error versus CPU time for the $p$-adaptation ($p = 0.5, 1.0, 1.5, 2.0, 2.5, \text{and} 3.0$).
Fig. 3.1: Continued.
Fig. 3.1: Continued.
Fig. 3.2: (a) Reflection coefficient of the monopole. (b) Radiation pattern.
Fig. 3.3: (a) Geometry of the dielectric ring structure, where $a = 324$, $b = 121$, $c = 43$, $h = 39$, $w_1 = 207.25$, $w_2 = 116.75$, $r_1 = 16.65$, $r_2 = 26.75$, all units are mm. (b) Dipole voltages by the three methods. (c) Dipole voltages by DGTD-Upwind using different orders of basis functions.
Fig. 3.3: Continued.
Fig. 3.4: (a) Geometry of the patch array. (b) Scattering parameters.
Fig. 3.5: Reflection coefficient of the single patch antenna.

Table 3.1: Convergence orders of the three algorithms.

<table>
<thead>
<tr>
<th>p</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>q (DFDD-ELD)</td>
<td>1.35</td>
<td>1.98</td>
<td>3.18</td>
</tr>
<tr>
<td>q (DGTD-Upwind)</td>
<td>1.96</td>
<td>3.01</td>
<td>3.83</td>
</tr>
<tr>
<td>q (DGTD-Central)</td>
<td>0.91</td>
<td>2.10</td>
<td>2.75</td>
</tr>
</tbody>
</table>

Table 3.2: Resonant wave numbers of the first four modes in the PEC cavity.

<table>
<thead>
<tr>
<th>Mode</th>
<th>FETD (Gedney)</th>
<th>DGTD-Central (Gedney)</th>
<th>DFDD-ELD (this work)</th>
<th>DGTD-Upwind (this work)</th>
<th>DGTD-Central (this work)</th>
</tr>
</thead>
<tbody>
<tr>
<td>k01</td>
<td>0.952</td>
<td>0.952</td>
<td>0.958</td>
<td>0.956</td>
<td>0.956</td>
</tr>
<tr>
<td>k02</td>
<td>1.420</td>
<td>1.415</td>
<td>1.422</td>
<td>1.421</td>
<td>1.421</td>
</tr>
<tr>
<td>k03</td>
<td>1.615</td>
<td>1.611</td>
<td>1.619</td>
<td>1.617</td>
<td>1.618</td>
</tr>
<tr>
<td>k04</td>
<td>2.026</td>
<td>2.026</td>
<td>2.027</td>
<td>2.025</td>
<td>2.025</td>
</tr>
</tbody>
</table>
Table 3.3: Performances of the four algorithms.

<table>
<thead>
<tr>
<th></th>
<th>Explicit DFDD</th>
<th>Explicit DGTD</th>
<th>Hybrid DFDD</th>
<th>Hybrid DGTD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Max. Time Step Size (ns)</strong></td>
<td>0.055</td>
<td>0.066</td>
<td>1.40</td>
<td>1.76</td>
</tr>
<tr>
<td><strong>Num. Time Steps</strong></td>
<td>254550</td>
<td>212121</td>
<td>10000</td>
<td>7955</td>
</tr>
<tr>
<td><strong>CPU Time (min)</strong></td>
<td>1443</td>
<td>1309</td>
<td>57</td>
<td>73</td>
</tr>
<tr>
<td><strong>Memory (MB)</strong></td>
<td>339</td>
<td>401</td>
<td>354</td>
<td>727</td>
</tr>
</tbody>
</table>
CHAPTER 4
MODELING OF DOUBLY LOSSY AND
DISpersive Media WITH THE DUAL-FIELD
DOMAIN-DECOMPOSITION ALGORITHM

4.1 Introduction
In this chapter, the modeling of doubly lossy and dispersive media will be incorporated into the DFDD scheme to improve the modeling capability of the algorithm. In Section 4.2, the modeling strategy will be presented, then the boundary conditions will be incorporated, and after that the computation of convolutions will be described. In Section 4.3, a brief analysis of phase error will be performed to identify the accuracy of the modeling strategy. In Section 4.4 some numerical examples will be presented to validate the formulation and to investigate the performance. In Section 4.5 the chapter will be summarized.

4.2 Formulation
4.2.1 Modeling of Doubly Lossy and Dispersive Media
The constitutive relations for a general isotropic, lossy, and dispersive medium are given by

\[ \mathbf{D}(t) = \varepsilon_0 \varepsilon_\infty \mathbf{E}(t) + \varepsilon_0 \chi_e(t) \otimes \mathbf{E}(t) \] (4.1)

\[ \mathbf{B}(t) = \mu_0 \mu_\infty \mathbf{H}(t) + \mu_0 \chi_m(t) \otimes \mathbf{H}(t) \] (4.2)

where \( \varepsilon_\infty \) and \( \mu_\infty \) are the relative permittivity and permeability at infinite frequencies, \( \chi_e(t) \) and \( \chi_m(t) \) are the time-dependent electric and magnetic susceptibility functions,
and symbol “\( \otimes \)” denotes the time convolution. By taking the time derivative of Maxwell’s equations

\[
\nabla \times H = \frac{\partial D}{\partial t} + \sigma_e E + J_{\text{imp}}
\]

\( (4.3) \)

\[
\nabla \times E = -\frac{\partial B}{\partial t} - \sigma_m H - M_{\text{imp}}
\]

\( (4.4) \)

and substituting (4.1) and (4.2) into the resultant equations, we obtain the following second-order wave equations:

\[
-\nabla \times \frac{\partial H}{\partial t} + \varepsilon_0 \varepsilon_\infty \frac{\partial^2 E}{\partial t^2} + \sigma_e \frac{\partial E}{\partial t} + \varepsilon_0 \chi_\varepsilon \otimes \frac{\partial^2 E}{\partial t^2} = -\frac{\partial J_{\text{imp}}}{\partial t}
\]

\( (4.5) \)

\[
\nabla \times \frac{\partial E}{\partial t} + \mu_0 \mu_\infty \frac{\partial^2 H}{\partial t^2} + \sigma_m \frac{\partial H}{\partial t} + \mu_0 \chi_m \otimes \frac{\partial^2 H}{\partial t^2} = -\frac{\partial M_{\text{imp}}}{\partial t}
\]

\( (4.6) \)

where \( \sigma_e \) and \( \sigma_m \) are the electric and magnetic conductivities, and \( J_{\text{imp}} \) and \( M_{\text{imp}} \) are the impressed electric and magnetic current densities, respectively. Testing the equations above using a vector basis function \( \mathbf{T} \) and performing the integration in the subdomain under consideration yields the weak-form solutions

\[
\iiint_{V_s} \left[ \frac{1}{\mu_0 \mu_\infty} (\nabla \times \mathbf{T}) \cdot (\nabla \times \mathbf{E}) - (\nabla \times \mathbf{T}) \cdot \mathbf{Q} + \varepsilon_0 \varepsilon_\infty \mathbf{T} \cdot \frac{\partial^2 \mathbf{E}}{\partial t^2} + \sigma_e \mathbf{T} \cdot \frac{\partial \mathbf{E}}{\partial t} + \varepsilon_0 \chi_\varepsilon \otimes \frac{\partial^2 \mathbf{E}}{\partial t^2} \right] dV - \iint_{S_s} \left( \hat{n} \times \frac{\partial \mathbf{H}}{\partial t} \right) dS = -\iiint_{V_s} \mathbf{T} \cdot \frac{\partial \mathbf{J}_{\text{imp}}}{\partial t} dV
\]

\( (4.7) \)

\[
\iiint_{V_s} \left[ \frac{1}{\varepsilon_0 \varepsilon_\infty} (\nabla \times \mathbf{T}) \cdot (\nabla \times \mathbf{H}) - (\nabla \times \mathbf{T}) \cdot \mathbf{R} + \mu_0 \mu_\infty \mathbf{T} \cdot \frac{\partial^2 \mathbf{H}}{\partial t^2} + \sigma_m \mathbf{T} \cdot \frac{\partial \mathbf{H}}{\partial t} + \mu_0 \chi_m \otimes \frac{\partial^2 \mathbf{H}}{\partial t^2} \right] dV + \iint_{S_s} \left( \hat{n} \times \frac{\partial \mathbf{E}}{\partial t} \right) dS = -\iiint_{V_s} \mathbf{T} \cdot \frac{\partial \mathbf{M}_{\text{imp}}}{\partial t} dV
\]

\( (4.8) \)

where \( V_s \) and \( S_s \) are the volume and boundary of the subdomain, \( \hat{n} \) is the outward unit vector normal to \( S_s \), and

\[
\mathbf{Q} = -\frac{1}{\mu_0 \mu_\infty} \left( \mu_0 \chi_m \otimes \frac{\partial \mathbf{H}}{\partial t} + \sigma_m \mathbf{H} + \mathbf{M}_{\text{imp}} \right)
\]

\( (4.9) \)
\[ R = \frac{1}{\varepsilon_0\varepsilon_w} \left( \varepsilon_0\varepsilon_w \otimes \frac{\partial \mathbf{E}}{\partial t} + \sigma_s \mathbf{E} + \mathbf{J}_{\text{imp}} \right), \]  

(4.10)

Note that the above two equations are theoretically identical to

\[ Q = \frac{\partial \mathbf{H}}{\partial t} + \frac{1}{\mu_0\mu_w} \nabla \times \mathbf{E} \]  

(4.11)

\[ R = -\frac{\partial \mathbf{E}}{\partial t} + \frac{1}{\varepsilon_0\varepsilon_w} \nabla \times \mathbf{H} \]  

(4.12)

which is the formulation used in the literature for FETD. However, the identity does not hold numerically for DFDD, since the electric and magnetic fields are solved from the vector wave equations separately, instead of directly related to each other via Maxwell’s equations. When (4.11) and (4.12) are used, it is observed that \( Q \) and \( R \), which represent the contribution from media dispersion, are nonzero numerically even if the dispersion does not exist. For this reason, the time marching becomes unstable. This problem is solved by transforming (4.11) and (4.12) into (4.9) and (4.10), where \( Q \) and \( R \) are exactly zero when the medium is lossless and non-dispersive.

### 4.2.2 Incorporation of Boundary Conditions

The incorporation of boundary conditions such as PEC, ABC, and subdomain interfaces is exactly the same as that in Section 3.2.1 and thus is not repeated here. The resultant system equations can be stated as

\[
\int \int \int_{V, j} \left[ \frac{1}{\mu_0} (\nabla \times \mathbf{T}) \cdot (\nabla \times \mathbf{E}) - \frac{1}{c_0^2} Z_0 (\nabla \times \mathbf{T}) \cdot \mathbf{Q} + \frac{1}{c_0^2} \varepsilon_0 \mathbf{T} \cdot \frac{\partial^2 \mathbf{E}}{\partial t^2} + \frac{1}{c_0} Z_0 \sigma_s \mathbf{T} \cdot \frac{\partial \mathbf{E}}{\partial t} 
\right. 
\left. + \frac{1}{c_0^2} \mathbf{T} \cdot \chi_e \otimes \frac{\partial^2 \mathbf{E}}{\partial t^2} \right] dV + \frac{1}{c_0} Z_0 \int \int \int_{S_{\text{sc}}} Y (\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{E}}{\partial t} \right) dS 
\]

\[
= \frac{1}{c_0} Z_0 \int \int_{S_i} (\hat{n} \times \mathbf{T}) \cdot \left( \hat{n} \times \frac{\partial \mathbf{J}_{\text{imp}}}{\partial t} \right) dS - \frac{1}{c_0} Z_0 \int \int \int_{V, j} \mathbf{T} \cdot \frac{\partial \mathbf{J}_{\text{imp}}}{\partial t} dV 
\]  

(4.13)
\[
\iint_{V_c} \left[ \frac{1}{\varepsilon_\infty} (\nabla \times T) \cdot (\nabla \times H) - \frac{1}{c_0} Y_0 (\nabla \times T) \cdot R + \frac{1}{c_0^2} \mu_n T \cdot \frac{\partial^2 H}{\partial t^2} + \frac{1}{c_0} Y_0 \sigma_m T \cdot \frac{\partial H}{\partial t} \right. \\
+ \frac{1}{c_0} T \cdot \varepsilon_n \otimes \frac{\partial^2 H}{\partial t^2} \int dV + \frac{1}{c_0} Y_0 \iint_{S_{\mathrm{inc}}} Z (\hat{n} \times T) \cdot \left( \hat{n} \times \frac{\partial H}{\partial t} \right) dS \\
= \frac{1}{c_0} Y_0 \iint_{S_{\mathrm{inc}}} (\hat{n} \times T) \cdot \left( \hat{n} \times \frac{\partial M_{\mathrm{imp}}}{\partial t} \right) dS - \frac{1}{c_0} Y_0 \iint_{V_c} T \cdot \frac{\partial M_{\mathrm{imp}}}{\partial t} dV.
\]

4.2.3 Computation of Convolutions and Construction of Matrix Equations

In total, there are four time convolutions to be calculated: the two shown explicitly in (4.13) and (4.14), and the other two buried implicitly in \( Q \) and \( R \) which have to be calculated using (4.9) and (4.10). For simplicity, we assume that the susceptibility functions follow the Debye or Lorentz model (although the formulation is not limited to these models); therefore, their time-domain expressions can be written as

\[
\chi_e(t) = \Re(a_e e^{-b_e t})u(t)
\]

\[
\chi_m(t) = \Re(a_m e^{-b_m t})u(t)
\]

where \( u(t) \) is the unit step function. To compute the explicit convolution in (4.13), we follow the approach in [32] and obtain

\[
\chi_e(t) \otimes \frac{\partial^2 \tilde{E}(t)}{\partial t^2} \approx \left[ \int_{0}^{\Delta t/2} \chi_e(\tau) d\tau \right] \tilde{E}^n + \sum_{m=0}^{n-1} \left[ \int_{(m+1/2)\Delta t}^{(m+3/2)\Delta t} \chi_e(\tau) d\tau \right] \tilde{E}^{n-m-1} \\
\approx \Re(a_e c_e) \tilde{E}^n + \Re(a_e d_e) \sum_{m=0}^{n-1} e^{-b_e (m+1/2) \Delta t} \tilde{E}^{n-m-1}
\]

where the second time derivative of the electric field has been denoted as \( \tilde{E} \) and we have assumed that \( \tilde{E}(t) = 0 \) for \( t < 0 \) and have used the \( \tilde{E} \) at the midpoint of each time interval for its approximation. The time-independent coefficients \( c_e \) and \( d_e \) can be calculated as

\[
c_e = \frac{1}{b_e} (1 - e^{-b_e \Delta t/2})
\]
\[
d_{e} = \frac{1}{b_{e}}(1 - e^{-b_{e}A_{e}})e^{-b_{e}A_{e}/2}.
\] (4.19)

Note that the first term in (4.17) should be treated as an unknown and remain on the left-hand side of (4.13). After expanding the electric field using vector basis functions as \( \mathbf{E} = \sum_{j} \mathbf{N}_{j} e_{j} \), Eq. (4.13) can be discretized as

\[
\begin{align*}
\left[ S_{e} \right] e + \frac{1}{c_{0}^{2}} \left[ M_{e} \right] \frac{d^{2} \{ e \}}{dt^{2}} + \frac{1}{c_{0}} \left[ B_{e} \right] \frac{d \{ e \}}{dt} + \frac{1}{c_{0}} \left[ A_{e} \right] \frac{d \{ e \}}{dt} &= -\frac{1}{c_{0}} \left\{ f_{e} \right\} + \frac{1}{c_{0}} \left\{ j \right\} - \frac{1}{c_{0}^{2}} \left\{ q \right\} - \frac{1}{c_{0}^{2}} \left\{ g_{e} \right\} \\
&= -\frac{1}{c_{0}} \left\{ f_{e} \right\} + \frac{1}{c_{0}} \left\{ j \right\} - \frac{1}{c_{0}^{2}} \left\{ q \right\} - \frac{1}{c_{0}^{2}} \left\{ g_{e} \right\} \\
\end{align*}
\] (4.20)

where

\[
S_{e}(i, j) = \iiint_{V_{i}} \frac{1}{\mu_{\infty}} (\nabla \times \mathbf{N}_{i}) \cdot (\nabla \times \mathbf{N}_{j}) dV
\] (4.21)

\[
M_{e}(i, j) = \iiint_{V_{i}} \left[ e_{\infty} + \text{Re}(a_{e}c_{e}) \right] \mathbf{N}_{i} \cdot \mathbf{N}_{j} dV
\] (4.22)

\[
B_{e}(i, j) = Z_{0} \iiint_{V_{i}} \sigma_{e} \mathbf{N}_{j} \cdot \mathbf{N}_{j} dV
\] (4.23)

\[
A_{e}(i, j) = Z_{0} \iiint_{S_{\text{arc}}} Y (\hat{n} \times \mathbf{N}_{i}) \cdot (\hat{n} \times \mathbf{N}_{j}) dS
\] (4.24)

\[
f_{e}(i) = Z_{0} \iiint_{V_{i}} \mathbf{N}_{i} \cdot \mathbf{J}_{\text{imp}} dV
\] (4.25)

\[
j(i) = Z_{0} \iiint_{S_{i}} (\hat{n} \times \mathbf{N}_{i}) \cdot (\hat{n} \times \mathbf{J}_{i}) dS
\] (4.26)

\[
q(i) = Z_{0} \iiint_{V_{i}} (\nabla \times \mathbf{N}_{i}) \cdot \mathbf{Q} dV
\] (4.27)

\[
g_{e}(i) = \sum_{j} \iiint_{V_{i}} \text{Re}(a_{e} d_{e} \psi_{ej}^{n}) \mathbf{N}_{i} \cdot \mathbf{N}_{j} dV
\] (4.28)

In (4.28),

\[
\psi_{ej}^{n} = \sum_{m=0}^{n-1} e^{-b_{m}A_{e}} \epsilon_{ej}^{n-m-1}
\] (4.29)

which can be updated using the recursive convolution algorithm [7], [30]-[33]:
\[ \psi_{ej}^n = e^{-h_j \Delta t} \psi_{ej}^{n-1} + \ddot{e}_j^{n-1} \approx e^{-h_j \Delta t} \psi_{ej}^{n-1} + \frac{e_j^n - 2e_j^{n-1} + e_j^{n-2}}{\Delta t^2} \]  
(4.30)

where \( \ddot{e}_j \) represents the second time derivative of the electric field unknown \( e_j \). To calculate the right-hand side component given by (4.27), the convolution in (4.9) has to be evaluated. The same procedure as the one shown in (4.17)-(4.19) and (4.30) can be used to obtain

\[ \chi_m \otimes \frac{\partial \mathbf{H}}{\partial t} \bigg|_{t=n\Delta t} = \sum_{j=1}^{N_j} N_j \left[ \text{Re}(a_m c_m) \dot{h}_j^n + \text{Re}(a_m d_m \varphi_{mj}^n) \right] \]  
(4.31)

where

\[ \varphi_{mj}^n = \sum_{m=1}^{n-1} e^{-h_m \Delta t} \dot{h}_j^{n-m-1} \approx e^{-h_m \Delta t} \varphi_{mj}^{n-1} + \frac{h_j^{n-1/2} - h_j^{n-3/2}}{\Delta t} \]  
(4.32)

and \( \dot{h}_j \) is the first time derivative of the magnetic field unknown in the expansion

\[ \mathbf{H} = \sum_j N_j h_j. \]

Equation (4.14) can be treated in the same way as (4.13) and the semi-discrete form can be obtained as

\[ 
\left[S_n\right]\{h\} + \frac{1}{c_0} \left[M_n\right] \frac{d^2\{h\}}{dt^2} + \frac{1}{c_0} \left[B_n\right] \frac{d\{h\}}{dt} + \frac{1}{c_0} \left[A_n\right] \frac{d\{h\}}{dt} \\
= -\frac{1}{c_0} \frac{d\{f_h\}}{dt} + \frac{1}{c_0} \frac{d\{m\}}{dt} + \frac{1}{c_0} \{r\} - \frac{1}{c_0^{3/2}} \{g_h\} 
\]  
(4.33)

where

\[ S_h(i, j) = \iint_{V_i} \frac{1}{\epsilon_\infty} \nabla \times \mathbf{E}_i \cdot (\nabla \times \mathbf{E}_j) dV \]  
(4.34)

\[ M_h(i, j) = \iint_{V_i} [\mu_\infty + \text{Re}(a_m c_m)] N_i \cdot N_j dV \]  
(4.35)

\[ B_h(i, j) = Y_0 \iint_{V_i} \sigma_m N_i \cdot N_j dV \]  
(4.36)

\[ A_h(i, j) = Y_0 \iint_{S_{\text{vac}}} Z(\hat{n} \times \mathbf{N}_i) \cdot (\hat{n} \times \mathbf{N}_j) dS \]  
(4.37)
\[ f_s(i) = Y_e \iint_{V_s} \mathbf{N}_i \cdot \mathbf{M}_{imp} dV \]  

(4.38)

\[ m(i) = Y_e \iiint_{S_s} (\hat{n} \times \mathbf{N}_i) \cdot (\hat{n} \times \mathbf{M}_s) dS \]  

(4.39)

\[ r(i) = Y_e \iiint_{V_s} (\nabla \times \mathbf{N}_i) \cdot \mathbf{R} dV \]  

(4.40)

\[ g_s(i) = \sum_j \iiint_{V_s} \text{Re}(a_m d_m \psi_{mj}^{n+1/2}) \mathbf{N}_i \cdot \mathbf{N}_j dV \]  

(4.41)

and the convolution term

\[ \psi_{mj}^{n+1/2} = \sum_{m=0}^{n-1} e^{-h_m \Delta t} \tilde{h}_j^{n+1/2-m-1} \]  

(4.42)

can be updated as

\[ \psi_{mj}^{n+1/2} \approx e^{-h_{t,m\Delta t}} \psi_{mj}^{n-1/2} + \frac{h_j^{n+1/2} - 2h_j^{n-1/2} + h_j^{n+3/2}}{\Delta t^2}. \]  

(4.43)

In order to evaluate \( R \) in (4.40), the convolution term in (4.10) is calculated as

\[ \chi_e \otimes \frac{\partial \mathbf{E}}{\partial t} \bigg|_{t=(n+1/2)\Delta t} = \sum_j \mathbf{N}_j \left[ \text{Re}(a_c \psi_j) \tilde{e}_j^{n+1/2} + \text{Re}(a_e d_e \varphi_{ej}^{n+1/2}) \right] \]  

(4.44)

where

\[ \varphi_{ej}^{n+1/2} = \sum_{m=0}^{n-1} e^{-h_{t,m\Delta t}} \tilde{e}_j^{n+1/2-m-1} \approx e^{-h_{t,m\Delta t}} \varphi_{ej}^{n-1/2} + \frac{e_j^n - e_j^{n-1}}{\Delta t}. \]  

(4.45)

In the above, \( \tilde{h}_j \) and \( \tilde{e}_j \) denote the second time derivative of the magnetic field unknown and the first time derivative of the electric field unknown, respectively.

### 4.3 Analysis of Phase Error

In this section, we perform a quantitative error analysis in order to estimate the error induced by the modeling of medium dispersions. We follow the dispersion analysis in [11] based on the one-dimensional wave propagation problem with two finite elements (as shown in Fig. 4.1) and the numerical propagation constant \( k \) is obtained and compared
with the analytical propagation constant $k$. All losses and impressed currents are assumed to be zero and the error due to domain decomposition is not considered here since it is already analyzed in [11]. Without loss of generality, it is further assumed that the wave is propagating in the positive $x$-direction.

4.3.1 Modeling of Electric Dispersion

In the case where only the electric dispersion exists, Eq. (4.13) can be reduced to

$$\iiint \left[ \frac{1}{\mu_0} (\nabla \times \mathbf{T}) \cdot (\nabla \times \mathbf{E}) + \frac{1}{c_0^2} \varepsilon_\infty \mathbf{T} \cdot \frac{\partial^2 \mathbf{E}}{\partial t^2} + \frac{1}{c_0^2} \mathbf{T} \cdot \chi_e \otimes \frac{\partial^2 \mathbf{E}}{\partial t^2} \right] dV = 0. \quad (4.46)$$

For the one-dimensional case shown in Fig. 4.1, $\mathbf{T} = N_j$, $\mathbf{E} = \sum_{j=1}^{N} N_j \mathbf{e}_j$, and $N_j = \hat{y}N_j$, where $N_j$ is the one-dimensional nodal basis function. If $\varepsilon_\infty = \mu_\infty = 1$, Eq. (4.46) can be further simplified to

$$\sum_{j=1}^{N} \int \left( \frac{dN_j}{dx} \frac{dN_j}{dx} \mathbf{e}_j + \frac{1}{c_0^2} N_iN_j \frac{d^2 \mathbf{e}_j}{dt^2} + \frac{1}{c_0^2} N_iN_j \chi_e \otimes \frac{d^2 \mathbf{e}_j}{dt^2} \right) dx = 0. \quad (4.47)$$

In this analysis, it is assumed that the wave is a uniform plane wave which is periodic in the propagation direction; therefore, $e_j^\omega = e^{(i\omega t - \tilde{k}h)}$ where $j$ is the unit of imaginary numbers, $h$ is the element length, and $\tilde{k}$ is the numerical wave number. By employing the first-order basis functions and a Debye model $\chi_e(\omega) = \frac{a_e}{\omega} + b_e$, the following expression can be obtained:

$$\sin^2 \tilde{k}h = \frac{h^2(1 + \alpha) \sin^2 \frac{\omega \Delta t}{2}}{c_0^2 \Delta t^2 \cos^2 \frac{\omega \Delta t}{2} + \frac{2}{3} h^2(1 + \alpha) \sin^2 \frac{\omega \Delta t}{2}} \quad (4.48)$$

where

$$\alpha = a_e \Delta t \left[ \sum_{m=0}^{N} e^{-(j\omega+b_e)m\Delta t} - \frac{1}{2} \right] \approx a_e \Delta t \left[ \frac{1}{1 - e^{-(j\omega+b_e)\Delta t}} - \frac{1}{2} \right] \quad (4.49)$$
and we have assumed that \( n \) is sufficiently large. Since \( \omega \Delta t \) is usually much smaller than one, the above can be approximated as

\[
\alpha \approx \chi_e(\omega) \left[ 1 + \frac{1}{12} (j \omega + b_e)^2 \Delta t^2 \right]. \tag{4.50}
\]

First, it is obvious that \( \tilde{k} \) approaches \( k = \omega / c_0 \) when both \( \Delta t \) and \( h \) approach zero. Second, when \( h \gg c_0 \Delta t \) or when the spatial discretization error dominates the total error, the relative error is given by

\[
\frac{\tilde{k} - k}{k} \approx -\frac{1}{24} k^2 h^2. \tag{4.51}
\]

Compared to the spatial discretization error of the standard FETD analyzed in [11], the error here takes the same form except that the \( k_0 \) in [11] is now replaced by \( k \). Finally, when \( h \ll c_0 \Delta t \), that is, when the temporal discretization error dominates the total error,

\[
\frac{\tilde{k} - k}{k} \approx \frac{1}{12} \omega^2 \Delta t^2 + \frac{1}{24} \frac{\chi_e(\omega)}{1 + \chi_e(\omega)} (j \omega + b_e)^2 \Delta t^2 \tag{4.52}
\]

where the first term is exactly the temporal discretization error of the standard FETD and the second term is the error purely induced by the modeling of the electric dispersion. Since the error is second order with respect to both \( h \) and \( \Delta t \), it can be concluded that the accuracy of the dispersion modeling is on the same order as that of the standard FETD.

### 4.3.2 Modeling of Magnetic Dispersion

In the DFDD scheme, the numerical wave number is assumed to be the same for the electric and magnetic fields. Therefore, the error induced by the modeling of the magnetic dispersion can be obtained directly by replacing \( b_e \) and \( \chi_e \) in (4.52) with \( b_m \) and \( \chi_m \), respectively:

\[
\frac{\tilde{k} - k}{k} \approx \frac{1}{12} \omega^2 \Delta t^2 + \frac{1}{24} \frac{\chi_m(\omega)}{1 + \chi_m(\omega)} (j \omega + b_m)^2 \Delta t^2. \tag{4.53}
\]
Since the approach for modeling the magnetic dispersion in this work is different from that in [7], [33] which can also be adopted by DFDD, it would be interesting to compare the error of the two approaches and find out which one is more accurate. The analysis procedure is similar to that described in Section 4.3.1 and is thus omitted here. The final result can be stated as

$$\frac{\tilde{k} - k}{k} \approx \frac{1}{12} \omega^2 \Delta t^2 + \frac{1}{8} \chi_m(\omega) \omega^2 \Delta t^2 + \frac{1}{24} \frac{\chi_m(\omega)}{1 + \chi_m(\omega)} (j \omega + b_m)^2 \Delta t^2$$  \hspace{1cm} (4.54)

where the first term is the temporal discretization error of the standard FETD, and the remaining terms represent the error induced by the modeling of the magnetic dispersion. Compared to (4.53), it is observed that there is an extra error term $$\frac{1}{8} \chi_m(\omega) \omega^2 \Delta t^2$$; therefore, the conclusion is that the magnetic dispersion modeling in this work is slightly more accurate than that in [7], [33].

### 4.4 Numerical Examples

In this section, several numerical examples are presented to validate the formulation described in Section 4.3 and to investigate the performance of the algorithm. All examples are calculated on an SGI Altix 350 system using an Intel Itanium II 1.5 GHz processor.

#### 4.4.1 Parallel-Plate Waveguide

In the first example, a simple parallel-plate waveguide structure is simulated to explore the stability condition of our algorithm and to validate our approach in the case where a general electrically and magnetically dispersive medium is involved. The computational domain is a 1 m $\times$ 1 m $\times$ 2 m rectangular box with PEC walls on the top and the bottom and PMC walls on the left and the right. The WPBC is applied to the remaining two surfaces. The computational domain is partitioned into two 1 m $\times$ 1 m $\times$ 1 m subdomains by an interface perpendicular to the wave propagation direction and the two subdomains are filled with the same material.
(a) Stability condition: Two dispersive media are considered here, and the first one is a magnetic Lorentz medium that will be used in the second example (Section 4.4.2) whose relative permittivity is 1.0 and the relative permeability is given by

\[
\mu_r(\omega) = \mu_\infty + \chi_m(\omega) = \mu_\infty + \frac{(\mu_s - \mu_\infty)\omega_0^2}{\omega_0^2 + j2\omega\delta - \omega^2}
\]  

(4.55)

where the nominal values are \( \mu_s = 2.0 \), \( \mu_\infty = 1.0 \), \( \omega_0 = 2\pi \times 0.2 \times 10^9 \text{ rad/s} \), and \( \delta = 0.1\omega_0 \). Three parameters, \( \mu_s - \mu_\infty \), \( \omega_0 \), and \( \delta \) are varied independently while the other two parameters remain the same as the nominal values, and the maximum allowable time step \( \Delta t_{\text{max}} \) is recorded Table 4.1. Notice that the medium dispersion disappears when \( \mu_s - \mu_\infty = 0 \). The value of \( \Delta t_{\text{max}} \) changes little although the dispersion parameters vary substantially. The second medium is an electric Debye medium that will be used in the third example (Section 4.4.3) whose relative permittivity is given by

\[
\varepsilon_r(\omega) = \varepsilon_\infty + \chi_e(\omega) = \varepsilon_\infty + \frac{\varepsilon_s - \varepsilon_\infty}{1 + j\omega\tau_e}
\]  

(4.56)

where \( \varepsilon_s = 4.18 \), \( \varepsilon_\infty = 3.64 \), and \( \tau_e = 11.40 \ \text{ps} \), and the relative permeability is 1.0. Two parameters, \( \varepsilon_s - \varepsilon_\infty \) and \( \tau_e \), are varied and the results are shown in Table 4.2. Again, \( \Delta t_{\text{max}} \) remains the same for different dispersive cases as that for the non-dispersive case.

(b) Modeling of a doubly dispersive medium: The waveguide is filled with an electrically and magnetically dispersive medium whose permittivity is given by (4.56) with \( \varepsilon_s = 1.5 \), \( \varepsilon_\infty = 1.0 \), and \( \tau_e = 1.06 \ \text{ns} \) and whose relative permeability is given by

\[
\mu_r(\omega) = \mu_\infty + \chi_m(\omega) = \mu_\infty + \frac{\mu_s - \mu_\infty}{1 + j\omega\tau_m}
\]  

(4.57)

where \( \mu_s = 3.0 \), \( \mu_\infty = 2.0 \), and \( \tau_m = 2.12 \ \text{ns} \). The frequency profiles of \( \varepsilon_r \) and \( \mu_r \) are plotted in Fig. 4.2(a). The total number of unknowns is 15,280 and the numerical solution by DFDD is compared with the analytical solution in Fig. 4.2(b) where the two curves overlap on each other, demonstrating the algorithm’s capability of modeling a general, doubly dispersive medium.
4.4.2 Rectangular Waveguide

The second example is a segment of a metallic rectangular waveguide with a cross section of 2.0 m by 0.3 m. The wave propagation direction is the positive \( z \)-direction and the waveguide occupies the region \( 0 < z < 0.7 \) m. A magnetic Lorentz medium is filled in the region \( 0.1 \) m \( < z < 0.6 \) m and the permeability of this material is given by (4.55) where \( \mu_s = 2.0 \), \( \mu_r = 1.0 \), \( \omega_0 = 2\pi \times 0.2 \times 10^9 \) rad/s, and \( \delta = 0.1\omega_0 \). The frequency profile of \( \mu_r \) is plotted in Fig. 4.3(a) and the \( S \) parameters are computed for the one-domain and six-subdomain cases and the results are compared with the analytical solution in Fig. 4.3(b). In the six-subdomain case, the region holding the dispersive material is partitioned into four subdomains by two planes perpendicular to the \( x \) and \( y \) axes and the other two subdomains are constituted of the two air regions \( 0 < z < 0.1 \) m and \( 0.6 \) m \( < z < 0.7 \) m. The finite element discretization with mixed second-order hierarchal vector basis functions yields 49,462 unknowns for the one-domain case and totally 84,790 unknowns for the six-subdomain case. (The number of unknowns for the six-subdomain case is significantly larger because a large portion of unknowns reside on the subdomain interfaces, which is not often the case.) The WPBC is employed for the two waveguide ports. Fig. 4.3(b) shows that both the one-domain and the six-subdomain results overlay on the analytical solution.

4.4.3 Parallel-Plate Transmission Line Structure

The third example is a parallel-plate transmission-line structure described in Fig. 3 in [57] and the geometry is depicted in Fig. 4.4(a). The substrate consists of an electric Debye material whose relative permittivity is given by (4.56) with the nominal values given in the first example and a conductor loss of \( \sigma_e = 6.29 \times 10^{-3} \) S/m. The frequency profile of the substrate permittivity is plotted in Fig. 4.4(b). The feeding coaxial lines have the inner and outer radii of 0.63 mm and 2.0 mm, respectively, and are filled with a lossless and non-dispersive material with \( \varepsilon_r = 2.1 \) and \( \mu_r = 1.0 \). Simulation is performed for the one-domain and two-subdomain cases. In the two-subdomain case, the subdomain
interface is perpendicular to the wave propagation direction and it partitions the structure exactly in the middle. The total number of unknowns is 57,497 for the one-domain case and 57,882 for the two-subdomain case. In Fig. 4.4(c) the one-domain and two-subdomain results are compared with the measurement result from [57] and a good agreement is observed. Note that the S parameters are plotted in dB to comply with the reference result.

4.4.4 Microstrip Patch Array

Recently, magnetic Lorentz materials have drawn significant attention in antenna design. It has been shown that by properly arranging this kind of material the antenna resonance can be achieved at a lower frequency with a relatively large bandwidth. In this example, the $2 \times 2$ rectangular patch antenna array described in Fig. 11 in [11] is simulated again and the geometry can also be found in Fig 3.4(a). In our simulation, the substrate has been changed to a magnetic Lorentz material with the following parameters:

\[ \varepsilon_r = 2.67, \quad \mu_r = 1.5, \quad \mu_e = 1.0, \quad \omega_0 = 2\pi \times 0.8 \times 10^9 \text{ rad/s}, \quad \text{and} \quad \delta = 0.5 \times 10^9 \text{ rad/s}. \]

The frequency profile of the substrate permeability is plotted in Fig. 4.5(a). The feeding coaxial lines have the inner and outer radii of 0.48 mm and 1.5 mm, respectively, and are filled with a lossless and non-dispersive material with $\varepsilon_r = 1.86$ and $\mu_r = 1.0$. A dense mesh with mixed second-order basis functions has been used and the total number of unknowns is 929,096 for the one-domain case and 937,808 for the four-subdomain case. The results from the two cases are compared with that from the finite-element frequency-domain (FEFD) method in Fig. 4.5(b) and the agreement is good. Two resonant frequencies are correctly predicted: the one around 0.8 GHz due to the resonance of the Lorentz material, and the other around 2.3 GHz which is the resonance frequency of the antenna array without medium dispersion.
4.5 Figures and Tables

\[ e_{i-1} \quad e_i \quad e_{i+1} \]

Element 1 \quad Element 2

Fig. 4.1: Two line elements.

![Graph showing relative permittivity and permeability of a doubly dispersive medium.](image)

(a)

![Graph showing scattering parameters for the parallel-plate waveguide.](image)

(b)

Fig. 4.2: (a) Relative permittivity and permeability of a doubly dispersive medium. (b) Scattering parameters for the parallel-plate waveguide.
Fig. 4.3: (a) Relative permeability of a magnetic Lorentz medium. (b) Scattering parameters for the rectangular waveguide.
Fig. 4.4: (a) Geometry of the parallel-plate transmission-line structure. (b) Relative permittivity of an electric Debye medium. (c) Scattering parameters for the parallel-plate transmission-line structure.
Fig. 4.5: (a) Relative permeability of a magnetic Lorentz medium. (b) Scattering parameters for the patch array.
Table 4.1: Maximum allowable time steps for the parallel-plate waveguide filled with a magnetic Lorentz medium.

(a) $\Delta t_{\text{max}}$ vs $\mu_s - \mu_\infty$.

<table>
<thead>
<tr>
<th>$\mu_s - \mu_\infty$</th>
<th>$\Delta t_{\text{max}}$ (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.36</td>
</tr>
<tr>
<td>1.00</td>
<td>0.36</td>
</tr>
<tr>
<td>2.00</td>
<td>0.36</td>
</tr>
</tbody>
</table>

(b) $\Delta t_{\text{max}}$ vs $\omega_b$.

<table>
<thead>
<tr>
<th>$\omega_b$ (rad/s)</th>
<th>$\Delta t_{\text{max}}$ (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2\pi \times 0.1 \times 10^4$</td>
<td>0.36</td>
</tr>
<tr>
<td>$2\pi \times 0.2 \times 10^4$</td>
<td>0.36</td>
</tr>
<tr>
<td>$2\pi \times 0.3 \times 10^4$</td>
<td>0.35</td>
</tr>
</tbody>
</table>

(c) $\Delta t_{\text{max}}$ vs $\delta$.

<table>
<thead>
<tr>
<th>$\delta$</th>
<th>$\Delta t_{\text{max}}$ (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0.05\omega_b$</td>
<td>0.36</td>
</tr>
<tr>
<td>$0.10\omega_b$</td>
<td>0.36</td>
</tr>
<tr>
<td>$0.20\omega_b$</td>
<td>0.36</td>
</tr>
</tbody>
</table>

Table 4.2: Maximum allowable time steps for the parallel-plate waveguide filled with an electric Debye medium.

(a) $\Delta t_{\text{max}}$ vs $\varepsilon_s - \varepsilon_\infty$.

<table>
<thead>
<tr>
<th>$\varepsilon_s - \varepsilon_\infty$</th>
<th>$\Delta t_{\text{max}}$ (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.38</td>
</tr>
<tr>
<td>0.54</td>
<td>0.38</td>
</tr>
<tr>
<td>1.08</td>
<td>0.38</td>
</tr>
</tbody>
</table>

(b) $\Delta t_{\text{max}}$ vs $\tau_\varepsilon$.

<table>
<thead>
<tr>
<th>$\tau_\varepsilon$ (ps)</th>
<th>$\Delta t_{\text{max}}$ (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.70</td>
<td>0.38</td>
</tr>
<tr>
<td>11.40</td>
<td>0.38</td>
</tr>
<tr>
<td>22.80</td>
<td>0.38</td>
</tr>
</tbody>
</table>
CHAPTER 5
TIME-DOMAIN MODELING OF 3D HIGH-SPEED
INTEGRATED CIRCUITS

5.1 Introduction

In this chapter, the modeling of 3D high-speed integrated circuits using the existing FETD-based methods is explored, and the new layered domain decomposition (LADD) method which is unconditionally stable is proposed. This chapter is organized as follows. First, some preliminary tests using FETD, DFDD, and DGTD are presented and the efficiency issue is analyzed in Section 5.2. Then the LADD method is proposed and the formulation is described in Section 5.3. Finally, several numerical examples are presented to validate the accuracy and efficiency of the LADD method in Section 5.4.

5.2 Preliminary Tests

In this section, two numerical examples are simulated to test the accuracy and efficiency of FETD, DFDD, and DGTD. All examples are calculated on an SGI Altix 350 system using an Intel Itanium II 1.5 GHz processor.

5.2.1 Single Via through a Single-Layer Printed-Circuit Board (PCB)

In the first example, a simple structure containing one via through a single-layer PCB is simulated, whose detailed geometry is described in the second and third paragraphs in Section IV in [58] and sketched in Fig. 5.1(a). Three methods, FETD, DFDD, and DGTD are tested. To relax the time step restriction due to the small mesh size around the via, hybrid implicit-explicit methods are used for both DFDD and DGTD, and the central flux version is used for DGTD due to its energy conservation property. The same meshes and full-first-order basis functions are used for all three algorithms and the total number of unknowns is 74,184 for FETD, 172,396 for DFDD, and 308,460 for DGTD. The number of unknowns for DFDD is much larger than that of FETD because in
the explicit region the unknowns at the element interfaces and corners are shared by the neighboring elements in FETD whereas they are assigned to each neighboring element in DFDD. The number of unknowns is even larger in DGTD because even in the implicit region unknowns are assigned to all neighboring elements. The simulated scattering parameters are plotted in Fig. 5.1(b), where the result from the FEFD method is used as the reference, and good agreement is observed.

5.2.2 Two Pairs of Differential Vias in a Seven-Layer PCB

To test the modeling capability of the algorithms, a more realistic example is considered here. This structure contains two pairs of differential vias connected through traces in a seven-layer PCB. The detailed geometry is given in part B, Section IV in [59] and in Section 5.5.4 in [60], and is shown in Fig. 5.2(a) for convenience. For the simplicity of implementation, the dielectric loss has been approximated by a constant conductivity throughout the frequency band. To reduce the computational time, a perfect conducting ground plane is inserted at the symmetric plane of the structure and only half of the structure (two vias connected by one trace) is simulated. Due to the tiny mesh size on the strip connecting the via pairs and on the thin ground plane, the hybrid implicit-explicit DFDD instead of the fully explicit DFDD is used. The DGTD described in Section 3.2.5 is also applied here, but the time step size becomes vanishingly small when the fully explicit scheme is used, and the number of unknowns in the implicit region becomes prohibitively large if a reasonable time step is to be achieved when the hybrid scheme is used. To obtain an accurate result at a reasonable computational cost, more efficient methods such as the hybrid FETD-DGTD have to be used, and this course is not investigated in this work. For this reason, the DGTD result is not generated for this example. The results from FETD and DFDD are compared with those from the measurement and FEFD. The magnitudes of scattering parameters are shown in Figs. 5.2(b) and 5.2(c) and a good agreement among the four curves is observed. The total number of unknowns is 1,141,936 for FETD and 1,644,460 for DFDD and the total CPU time is 9.4 hours for FETD and 53.0 hours for DFDD. (In order to reduce the long simulation time, the Prony method has been used to extrapolate the late-time response.)
It is observed that the CPU time for FETD is significantly shorter than that of DFDD. This is because the unconditionally stable FETD scheme allows a time step of 2.0 ps and requires only 2,050 time steps, while the conditionally stable DFDD scheme requires a step size of 0.22 ps and 17,035 steps, although the solution time per step for FETD is 16.5 s, versus 11.2 s for DFDD. The maximum allowable time step size for DFDD is small because the fine structures like vias and the strip connecting the vias requires an excessively dense mesh to resolve the fast-varying field around them, and this imposes a serious restriction on the time step size even if the hybrid implicit-explicit method is used. Notice that this example is only a middle-size problem; it is expected that when the problem size becomes large, FETD will eventually break down, while DFDD can still solve the problem by performing domain decomposition but the CPU time would become prohibitively long due to the restriction on the time step size.

5.3 LADD Formulation

The preliminary tests shows that when domain decomposition methods for general purposes like DFDD and DGTD are applied to 3D circuits, the long CPU time caused by the restricted time step size becomes a serious problem. Therefore, a domain decomposition method that is unconditionally stable is highly desired. It is noticed that an efficient domain decomposition method has been proposed in the frequency domain for PCB simulations by exploring the layered geometry [46]. In this algorithm, the volume unknowns inside each subdomain are individually eliminated, resulting in a global matrix equation containing only the via hole unknowns at the subdomain interfaces, which can be solved to extract the scattering parameters. Based on this algorithm, a new domain decomposition method in the time domain, LADD, is proposed, which preserves the unconditional stability of FETD. The formulation of LADD starts from the FETD matrix equation (2.32) which can be rewritten as

\[
[A]\{e\}^{n+1} = \{r\}^n
\]  

where

\[
\{r\}^n = \{b\}^n - [B]\{e\}^n - [C]\{e\}^{n-1}
\]  

(5.2)
and we have replaced matrices $A_0$, $A_1$, and $A_2$ with $A$, $B$, and $C$, for the simplicity of presentation.

To illustrate the LADD strategy, let us consider a four-subdomain problem shown in Fig. 5.3, where the four subdomains are denoted as $V_0$, $V_1$, $V_2$, and $V_3$, and the three interfaces between subdomains are denoted as $S_0$, $S_1$, and $S_2$. Each of the subdomains consists of one or more dielectric layers, and the neighboring subdomains are separated by ground planes and only connected through small via holes. For each subdomain $V_m$, the unknowns are reordered and classified into three types: those on the top interface of the subdomain (referred to as “top unknowns” and denoted by subscript “t” in the following), those on the bottom interface (“bottom unknowns” or “b”), and the remaining ones (“volume unknowns” or “v”). Thereafter, Eq. (5.1) can be partitioned as

$$
\begin{bmatrix}
A_{vv}^0 & A_{vb}^0 \\
A_{bv}^0 & A_{bb}^0 + A_{vb}^1 & A_{tb}^1 \\
A_{vb}^1 & A_{bb}^1 + A_{vb}^2 & A_{vb}^2 & A_{tt}^2 \\
A_{vb}^2 & A_{bb}^2 & A_{vb}^3 & A_{vb}^3 \\
A_{vb}^3 & A_{bb}^3 + A_{vb}^4 & A_{tb}^4 & A_{tt}^4
\end{bmatrix}
\begin{bmatrix}
e_0^{n+1} \\
e_{01}^1 \\
e_1^1 \\
e_{12}^2 \\
e_2^2 \\
e_{23}^3 \\
e_3^3
\end{bmatrix}
= \begin{bmatrix}
r_0^n \\
r_{01}^0 + r_{01}^1 \\
r_{11}^1 \\
r_{12}^1 + r_{12}^2 \\
r_{22}^2 \\
r_{23}^2 + r_{23}^3 \\
r_{33}^3
\end{bmatrix} \quad (5.3)
$$

where $A_{kl}^m$ ($m = 0, 1, 2, 3$ and $k, l = t, v, b$) represents the interaction between unknown group $k$ and unknown group $l$ in subdomain $V_m$. $A_{kl}^m$ is calculated by using (2.33) except that now the volume integrals are performed within $V_m$ instead of in the entire computational domain, and the testing and basis functions in the integrand belong to unknown group $k$ and $l$, respectively. Vectors $e^{m-1,m}$, $e^m$, and $e^{m,m+1}$ contain the top, volume, and bottom unknowns in $V_m$. Vector $r_k^m$ is the excitation vector corresponding to unknown group $k$ in $V_m$ and can be calculated as

$$
\begin{bmatrix}
r_{t}^m \\
r_{v}^m \\
r_{b}^m
\end{bmatrix}^n = \begin{bmatrix}
b_{t}^m \\
b_{v}^m \\
b_{b}^m
\end{bmatrix}^n - \begin{bmatrix}
B_{tt}^m & B_{tv}^m & B_{tb}^m \\
B_{vt}^m & B_{vv}^m & B_{vb}^m \\
B_{bt}^m & B_{bv}^m & B_{bb}^m
\end{bmatrix} \begin{bmatrix}
e_{m-1,m}^n \\
e_{m}^m \\
e_{m,m+1}^m
\end{bmatrix} - \begin{bmatrix}
C_{tt}^m & C_{tv}^m & C_{tb}^m \\
C_{vt}^m & C_{vv}^m & C_{vb}^m \\
C_{bt}^m & C_{bv}^m & C_{bb}^m
\end{bmatrix} \begin{bmatrix}
e_{m-1,m}^n \\
e_{m}^m \\
e_{m,m+1}^m
\end{bmatrix} \quad (5.4)
$$
where \( B^m_{kl} \) and \( C^m_{kl} \) are submatrices from the partitions of \( B \) and \( C \):

\[
B = \begin{bmatrix}
B^0_{vv} & B^0_{vb} \\
B^0_{bv} & B^0_{bb} + B^1_{bb} & B^1_{bv} \\
B^1_{vt} & B^1_{vb} & B^1_{bb} + B^2_{bb} & B^2_{bv} \\
B^2_{vt} & B^2_{vb} & B^2_{bb} + B^3_{bb} & B^3_{bv} \\
B^3_{vt} & B^3_{vb} & B^3_{bb} & B^3_{vb}
\end{bmatrix}
\]  
(5.5)

\[
C = \begin{bmatrix}
C^0_{vv} & C^0_{vb} \\
C^0_{bv} & C^0_{bb} + C^1_{bb} & C^1_{bv} \\
C^1_{vt} & C^1_{vb} & C^1_{bb} + C^2_{bb} & C^2_{bv} \\
C^2_{vt} & C^2_{vb} & C^2_{bb} + C^3_{bb} & C^3_{bv} \\
C^3_{vt} & C^3_{vb} & C^3_{bb} & C^3_{vb}
\end{bmatrix}
\]  
(5.6)

From (5.3), volume unknowns in each subdomain can be individually eliminated, yielding a global interface problem:

\[
\begin{bmatrix}
\tilde{A}^0_{bb} + \tilde{A}^1_{bb} & \tilde{A}^1_{vb} & \tilde{A}^1_{bb} + \tilde{A}^2_{bb} & \tilde{A}^2_{vb} \\
\tilde{A}^1_{vt} & \tilde{A}^1_{vb} & \tilde{A}^1_{bb} + \tilde{A}^3_{bb} & \tilde{A}^3_{vb} \\
\tilde{A}^2_{vt} & \tilde{A}^2_{vb} & \tilde{A}^2_{bb} + \tilde{A}^3_{bb} & \tilde{A}^3_{vb} \\
\tilde{A}^3_{vt} & \tilde{A}^3_{vb} & \tilde{A}^3_{bb} & \tilde{A}^3_{vb}
\end{bmatrix}
\begin{bmatrix}
\varepsilon^{01} \\
\varepsilon^{12} \\
\varepsilon^{23}
\end{bmatrix}^{n+1}
= \begin{bmatrix}
r_b^0 + \tilde{r}_b^1 \\
r_b^2 + \tilde{r}_b^3 \\
r_r^2 + \tilde{r}_l^3
\end{bmatrix}
\]  
(5.7)

where

\[
\tilde{A}^m_{kl} = A^m_{kl} - A^m_{kv} \left( A^m_{vv} \right)^{-1} A^m_{vl} \quad m = 0, 1, 2, 3 \quad \text{and} \quad k, l = t, b
\]  
(5.8)

\[
\tilde{r}^m_k = r^m_k - A^m_{kv} \left( A^m_{vv} \right)^{-1} r^m_v \quad m = 0, 1, 2, 3 \quad \text{and} \quad k = t, b.
\]  
(5.9)

Note that the size of (5.7) is small since it only contains the unknowns on the subdomain interfaces which are typically small via holes. After (5.7) is solved, the volume unknowns are recovered in each subdomain individually by using part of (5.3), or specifically,

\[
\begin{bmatrix}
A^m_{vv} \{e^m\} = \{r^m_v\} - \left[ A^m_{vt} \right] \{e^{m-1,v}\} - \left[ A^m_{vb} \right] \{e^{m-1,b}\}.
\]  
(5.10)

The entire LADD flow can be summarized as follows.
1. Assemble each subdomain matrix $A_{kl}^m$ in (5.3).

2. Factorize each $A_{vv}^m$.

3. Calculate each $\tilde{A}_{kl}^m$ by using (5.8).

4. Assemble and factorize the entire left-hand-side matrix in (5.7).

5. For a new time step, calculate each $r_k^m$ by using (5.4), and then $\tilde{r}_k^m$ by using (5.9).

6. For the current time step, solve (5.7) for interface unknowns.

7. For the current time step, recover volume unknowns by using (5.10). If time marching is not finished yet, go to step 5 for the next time step.

From the above formulation it is obvious that the numerical system solved in LADD is completely equivalent to that in FETD; therefore, LADD produces the same solution as FETD (except for a slight difference due to rounding errors) and has the same stability condition, which means LADD is also unconditionally stable. This property is critically important when modeling 3D circuits with fine geometries.

In the above seven steps, the most time-consuming part is time marching, or specifically, steps 5 and 7, since they involve the solution of a subdomain-level matrix equation at every time step. These two steps can be parallelized without any communication among processors since they require no information from other subdomains. The second most time-consuming part is the handling of subdomain-level matrices before time marching, or steps 1-3, which are also performed independently for each subdomain and can be parallelized. The two remaining steps, steps 4 and 6, are serial since they involve the handling of a global matrix equation; however, they consume little computational time compared to other steps due to the small size of Eq. (5.7). Therefore, the LADD algorithm can take the advantage of multiple CPUs/cores to achieve a shorter computational time.

### 5.4 Numerical Examples

In this section, several examples are tested to explore the accuracy and efficiency of the LADD method. It will be shown that LADD produces identical result to FETD while the computational time is shortened due to parallelization. The memory usage is also
reduced for large problems where the sparse solver has entered the nonlinear region where the solution time per time step is higher than $O(N)$. All examples are calculated on an SGI Altix 350 system using one or more Intel Itanium II 1.5 GHz processors.

### 5.4.1 Single Via through a Four-Layer PCB

The first example is a single via through four dielectric layers from Fig. 4.12 in [46] which is also sketched in Fig. 5.4(a). In the LADD simulation, each layer is treated as one subdomain, and thus there are totally four subdomains. The LADD result is compared with the FEFD result from [46] and the FETD result in Fig. 5.4(b), where the three results agree well and the LADD result is identical to the FETD result, demonstrating the accuracy of the LADD method.

Since one major advantage of LADD is the option of parallel computation, the parallel performance is investigated. The total number of unknowns for this example is 350,802 and the CPU time and memory usage of FETD and LADD using different numbers of CPUs are given in Table 5.1. The LADD parallel efficiency is obtained by comparing the time using different numbers of CPUs versus that using a single CPU in Fig. 5.4(c), where the speedup using four CPUs is 3.69, corresponding to a parallel efficiency of 92.3%. The efficiency of LADD is also compared with FETD in Fig. 5.4(d), where the speedup is defined as the FETD computational time using one CPU divided by the LADD computational time. Note that the speedup is significantly smaller than the number of CPUs, which is due to the extra time spent on step 7, the recovery of volume unknowns, which is not needed in FETD because it directly solves for all the unknowns. This is the price paid to perform the domain decomposition. It should also be pointed out that the scalability remains good up to four CPUs, and the speedup will further increase if more CPUs are available. The memory usage is also monitored and shown in Fig. 5.4(e), where “normalized memory usage” is defined as the LADD memory usage divided by the FETD memory usage. The memory usage of LADD using a single CPU is significantly less than that of FETD due to the nonlinear scaling of the sparse solver. The memory usage increases with the number of CPUs due to the initialization memory of the solver; however, LADD consumes no more memory than FETD up to four processors.
5.4.2 Coupled Vias through a Four-Layer PCB

The second example is coupled vias through four dielectric layers from Fig. 4.13 in [46] which is depicted in Fig. 5.5(a). Again, each layer is treated as one subdomain in LADD, resulting in totally four subdomains. The LADD result is compared with the FEFD result from [46] and the FETD result in Fig. 5.5(b), where the three results agree with each other and the FETD and LADD results are identical again.

The total number of unknowns for this example is 456,982 and the performance data are recorded in Table 5.2. The LADD speedup by using multiple CPUs over a single one is plotted in Fig. 5.5(c), where the speedup using four CPUs is 3.74, corresponding to a parallel efficiency of 93.5%. The speedup of LADD over FETD is shown in Fig. 5.5(d), where a speedup of 2.47 is achieved by using four CPUs. The memory usage of LADD versus FETD is shown in Fig. 5.5(e), where LADD consumes about the same amount of memory as FETD.

5.4.3 Three Vias in a Three-Layer PCB

The third example is a three-layer PCB structure with three vias: one signal via running through the dielectric layers and two return path vias connecting the top and bottom ground planes and passing the two middle PEC planes through voids. The geometry is sketched in Fig. 5.6(a), where two coaxial waveguide ports are used to terminate the two ends of the signal via and the structure is decomposed into three subdomains when LADD is used. The scattering parameters by using FEFD, FETD, and LADD are plotted against each other in Fig. 5.6(b), where agreement is observed and FETD and LADD results are identical.

There are totally 866,374 unknowns for this structure and the performance data of FETD and LADD using one and three CPUs are reported in Table 5.3. The speedup of LADD with three CPUs versus that with one CPU is 2.47, corresponding to a parallel efficiency of 82.3%, and the speedup of LADD with three CPUs versus FETD is 1.4. It is noticed that the LADD parallel efficiency is lower than the previous two examples, which is largely due to the load imbalance caused by the different thickness (and thus different numbers of unknowns) of the three dielectric layers. With respect to the memory usage, a
20% reduction is achieved by employing LADD with three CPUs, due to the nonlinear scaling of the sparse solver.

### 5.4.4 Two Pairs of Differential Vias in a Seven-Layer PCB

In order to test LADD on a larger problem, the differential via pair structure in Section 5.2.2 is simulated again. Notice that the bottom of the structure is open, where one air layer is added to model the free space. Counting the seven dielectric layers, there are totally eight layers and every two adjacent layers are grouped into one subdomain, resulting in four subdomains in LADD. The result from LADD is compared with those from FEFD, measurement, and FETD in Figs. 5.7(a) and (b), where all results agree well and the FETD and LADD results are identical.

The total number of unknowns for this example is 1,105,196 and the performance data are recorded in Table 5.4. The LADD speedup by using multiple CPUs over a single one is plotted in Fig. 5.7(c), where the speedup using four CPUs is 3.56, corresponding to a parallel efficiency of 88.9%, which is lower than the first two examples since the number of unknowns is not evenly distributed among subdomains in this example due to geometrical differences. The speedup of LADD over FETD is shown in Fig. 5.7(d), and the memory usage of is shown in Fig. 5.7(e). Due to the larger problem size and nonlinear scaling of the sparse solver, memory is significantly saved by performing domain decomposition, and a speedup of 2.42 and a memory reduction of 18.2% are achieved simultaneously by using LADD with four CPUs.
5.5 Figures and Tables

Fig. 5.1: (a) Geometry of the single via structure. (b) Scattering parameters.
Fig. 5.2: (a) Geometry of the differential via pair structure. (b) $|S_{11}|$.  (c) $|S_{21}|$. 

78
Fig. 5.2: Continued.

\[ |S_{21}| \text{(dB)} \]

---

\[ V_0 \]

\[ \cdots S_0 \]

\[ V_1 \]

\[ \cdots S_1 \]

\[ V_2 \]

\[ \cdots S_2 \]

\[ V_3 \]

Fig. 5.3: Geometry of a four-subdomain problem.
Fig. 5.4: (a) Geometry of the single-via structure. (b) Scattering parameters. The FETD and LADD results overlap. (c) Speedup of LADD using multiple processors versus using one processor. (d) Speedup of LADD versus FETD. (e) Memory usage of LADD versus FETD.
Fig. 5.4: Continued.
Fig. 5.4: Continued.
Fig. 5.5: (a) Geometry of the coupled-via structure. (b) Scattering parameters. The FETD and LADD results overlap. (c) Speedup of LADD using multiple processors versus using one processor. (d) Speedup of LADD versus FETD. (e) Memory usage of LADD versus FETD.
Fig. 5.5: Continued.
Fig. 5.5: Continued.
Fig. 5.6: (a) Geometry of the via structure. (b) Scattering parameters. The FETD and LADD results overlap.
Fig. 5.7: (a) $|S_{11}|$ of the differential via pair structure. The FETD and LADD results overlap. (b) $|S_{21}|$ of the structure. The FETD and LADD results overlap. (c) Speedup of LADD using multiple processors versus using one processor. (d) Speedup of LADD versus FETD. (e) Memory usage of LADD versus FETD.
Fig. 5.7: Continued.
Table 5.1: Performances of FETD and LADD for the single via structure.

<table>
<thead>
<tr>
<th></th>
<th>FETD</th>
<th>LADD (1 CPU)</th>
<th>LADD (2 CPU)</th>
<th>LADD (4 CPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. Time Steps</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>Total CPU Time (s)</td>
<td>1384</td>
<td>2002</td>
<td>1079</td>
<td>542</td>
</tr>
<tr>
<td>Memory (MB)</td>
<td>1676</td>
<td>1382</td>
<td>1453</td>
<td>1592</td>
</tr>
</tbody>
</table>

Fig. 5.7: Continued.
Table 5.2: Performances of FETD and LADD for the coupled via structure.

<table>
<thead>
<tr>
<th></th>
<th>FETD</th>
<th>LADD (1 CPU)</th>
<th>LADD (2 CPU)</th>
<th>LADD (4 CPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. Time Steps</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>Total CPU Time (s)</td>
<td>1905</td>
<td>2881</td>
<td>1510</td>
<td>770</td>
</tr>
<tr>
<td>Memory (GB)</td>
<td>2.12</td>
<td>1.99</td>
<td>2.06</td>
<td>2.17</td>
</tr>
</tbody>
</table>

Table 5.3: Performances of FETD and LADD for the three via structure.

<table>
<thead>
<tr>
<th></th>
<th>FETD</th>
<th>LADD (1 CPU)</th>
<th>LADD (3 CPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. Time Steps</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>Total CPU Time (s)</td>
<td>3266</td>
<td>5754</td>
<td>2327</td>
</tr>
<tr>
<td>Memory (GB)</td>
<td>5.21</td>
<td>3.94</td>
<td>4.18</td>
</tr>
</tbody>
</table>

Table 5.4: Performances of FETD and LADD for the differential via pair structure.

<table>
<thead>
<tr>
<th></th>
<th>FETD</th>
<th>LADD (1 CPU)</th>
<th>LADD (2 CPU)</th>
<th>LADD (4 CPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. Time Steps</td>
<td>2200</td>
<td>2200</td>
<td>2200</td>
<td>2200</td>
</tr>
<tr>
<td>Total CPU Time (s)</td>
<td>7159</td>
<td>10515</td>
<td>5909</td>
<td>2955</td>
</tr>
<tr>
<td>Memory (GB)</td>
<td>6.93</td>
<td>4.99</td>
<td>5.24</td>
<td>5.67</td>
</tr>
</tbody>
</table>
CHAPTER 6
CONCLUSIONS AND FUTURE RESEARCH

This dissertation is devoted to the investigation and development of FETD-based domain decomposition algorithms for electromagnetic analysis of dispersive media and high-speed circuits. Different algorithms are studied, compared, enhanced, and a new algorithm is proposed. This chapter concludes the current work and discusses the possible future research.

Three FETD–based explicit numerical algorithms for solving Maxwell’s equations are investigated in terms of accuracy and efficiency. Numerical examples show that all three algorithms can yield accurate and efficient solutions to Maxwell’s equations. Among the three, DGTD-Upwind is the most accurate and has the best error convergence; however, it is numerically dissipative and consumes about three times the computational time (when the fourth-order ERK is used). DFDD is slightly more accurate than DGTD-Central and they have a similar efficiency in terms of computational time and memory usage. Hybrid implicit-explicit schemes for DFDD and DGTD-Central are also investigated and the study shows that the hybrid DFDD outperforms the hybrid DGTD-Central implemented in this dissertation in terms of time and memory.

The comparative study shows that the DFDD method is accurate and highly suited for an efficient, easy-to-implement hybrid scheme. To further improve its modeling capability, a numerical scheme for modeling doubly lossy and dispersive materials is merged into the DFDD method. In the proposed approach, the existing algorithms for modeling medium dispersion are extended to the dual-field case and the enhanced DFDD algorithm is capable of modeling general complex media. A dispersion analysis is performed and the error due to the modeling of medium dispersion is estimated. The formulation is validated by comparing the numerical results with the analytical solution, the measurement result, and the FEFD result. The formulation is likely to find its application in the simulation of large-scale electromagnetic problems involving dispersive media, such as antenna arrays or integrated circuits with dispersive substrates.
Due to the wide and increasing applications of 3D high-speed circuits, the domain decomposition methods that have been investigated (DFDD and DGTD) are applied to this kind of problem. To solve the efficiency problem in DFDD and DGTD due to the restriction on the time step size, the unconditionally stable LADD method is proposed where each subdomain consists of one or more dielectric layers and subdomains are separated by ground planes. Based on the layered geometry, the volume unknowns in each subdomain are eliminated independently, resulting in a global matrix equation containing only the via hole unknowns at the subdomain interfaces. The global matrix equation can be solved in a negligible amount of time due to its small size and the volume unknowns in each subdomain can be recovered independently by using the solved via hole unknowns. Since the only steps that have to be performed in serial are those dealing with the small global system, the LADD method can be parallelized with good efficiency. Moreover, LADD generates the same solution as FETD if the rounding errors are neglected. Due to these advantages, LADD may find important applications in 3D circuit simulations, especially in cases where fine geometries are often encountered.

In order to further enhance the modeling capability of LADD, one possible research work is to reduce the time consumed by the serial steps and perform parallel computation using many cores for large-scale multi-layer structures. Currently, the serial steps consume little time compared to the parallel steps; however, the situation may change when many cores are used. Consider a PCB structure consisting of many layers: To fully explore the capability of many cores, the structure needs to be partitioned into more subdomains so that at least one subdomain can be assigned to a core, in which case the number of unknowns per subdomain decreases but that in the final global system increases. Therefore, the time for solving each subdomain is reduced but that for solving the final global system is increased. At a certain point, the latter may become a significant portion of the total computational time, impeding the further scaling of the LADD algorithm. If the computational time for solving the final system can be reduced, the efficiency of LADD would be improved and its speed would continue to scale up with the number of processors.

Another interesting topic closely related to LADD is the decomposition within one dielectric layer. The current LADD algorithm relies on the ground planes to separate
adjacent subdomains so that the subdomains only share a small number of unknowns at the via holes, which ensures the small size of the global system. If there are more processors than the number of dielectric layers, or the number of unknowns in each layer is highly unbalanced, it would be desirable to further decompose a layer. The major difficulty is that, when one layer is decomposed into several subdomains, the number of unknowns on the subdomain interfaces would be large, no matter where the interfaces lie, and this will result in a large global system and ruin the efficiency of LADD. If this problem can be solved, LADD would be more versatile in handling different types of 3D circuit structures and would gain more applications.
REFERENCES


