Chapter **2**

Maxwell's Equations

Outline

Some important symmetries, present in the Maxwell Equations, can be made explicit by using a proper substitution for the electromagnetic fields. Using these symmetries, it can be shown that, in a source-free and lossless environment, the time evolution operator[1] is orthogonal: it conserves the total electromagnetic energy density. Ideally, one would like to conserve this property when constructing an algorithm to solve the TDME.

The symmetries of the Maxwell Equations should thus be conserved during the two stages of discretization: the discretization of space and time. The first requirement can be met by using the staggered Yee lattice and the central-difference approximation for the spatial derivatives, to construct a skew-symmetric time evolution matrix, or "Hamiltonian". The time evolution itself, the solution of the TDME, is now given by the matrix exponential of this matrix.

At this point we can adopt three strategies to discretize time, i.e. to approximate the matrix exponential of the Hamiltonian, and obtain three new algorithms to solve the TDME.

The first option is to use the Lie-Trotter-Suzuki method to compute the matrix exponential by decomposition of the Hamiltonian into matrices for which the matrix exponentials can be computed efficiently. Proper application of this procedure ensures that the orthogonality of the matrix exponential is conserved. Consequently, the resulting algorithm will be unconditionally stable. However, to keep the error small, we must use sufficiently small timesteps for the time evolution of the fields.

The other strategy consists of using a Chebyshev expansion to approximate the matrix exponent of the time evolution matrix directly, for a given time instance. Although the orthogonality is not conserved in this case, the approximation is accurate up to machine precision, if enough expansion terms are included in the summation. The resulting algorithm is a 'one-step' algorithm, for the expansion is valid for one time instance only. However, if we interpret the time instance as a timestep, we can also use this algorithm for timestepping to compute e.g. a density of states (DOS).

Finally, we will show that with a small change, the conventional Yee algorithm can be made non staggered-in-time. This can be more convenient for simulations, especially for the preparation of the initial condition, but a second advantage is that in this new formulation, higher order in time algorithms can be easily derived and do not consume additional memory for storage of intermediate steps.

[1] the matrix containing the spatial derivatives and material parameters, as occurring in the first-order differential equation (2.11), will be referred to as 'time evolution matrix', in the course of this thesis. The matrix exponential of this matrix, as occurring in the solution (2.12) of the differential equation (2.11) will be denoted by the 'time evolution operator'.

The chapter will be concluded by the description of some additional tools that are necessary to be able to use the algorithm for measurements.

Units

If we measure distances in the unit λ, time and frequency are expressed in units of λ/c and c/λ, respectively. The TDME take a dimensionless form if we replace ε (μ) by its value relative to ε_0 (μ_0), the values for vacuum, and express $\mathbf{H}$ and $\mathbf{E}$ in units of A/m and V/m respectively. In the course of this thesis, we adopt this dimensionless form, so, from now on, $\mathbf{H}$, $\mathbf{E}$, ε, μ, t and $\mathbf{r}$ are dimensionless quantities.

2.1 Operators and symmetry

Symmetries in equations are of utmost importance in theoretical physics. In general, the existence of a symmetry indicates the conservation of a physical quantity. For example, in quantum-mechanics, the Hamiltonian is an Hermitian operator, and the norm of the wave-function is its related conserved quantity, which makes much sense when it is interpreted as a probability density distribution. At the same time, the measurement of an operator yields an eigenvalue, which is always real if the operator is Hermitian. In electromagnetism, similar symmetries exist, although they maybe less explicit at first sight. After some appropriate substitutions for the fields, the symmetries can be made explicit and the conservation of these symmetries is crucial in the development of our new algorithms.

In differential form, the time-dependent Maxwell Equations read [1, 2]

$$\frac{\partial}{\partial t}\mathbf{H}(t) = -\frac{1}{\mu}\nabla \times \mathbf{E}(t), \tag{2.1a}$$

$$\frac{\partial}{\partial t}\mathbf{E}(t) = \frac{1}{\varepsilon}\left(\nabla \times \mathbf{H}(t) - \mathbf{J}(t)\right), \tag{2.1b}$$

$$\operatorname{div} \varepsilon\mathbf{E}(t) = \rho(t), \tag{2.1c}$$

$$\operatorname{div} \mu\mathbf{H}(t) = 0. \tag{2.1d}$$

Although only the time dependence is written explicitly, all these quantities additionally depend on space, for instance: $\mathbf{E}(t) = (E_x(\mathbf{r}, t), E_y(\mathbf{r}, t), E_z(\mathbf{r}, t))^T$, but for simplicity of notation, we will omit the spatial dependence on $\mathbf{r} = (x, y, z)^T$ unless this leads to ambiguities. For the derivation of the numerical algorithms, we initially assume a source free and lossless environment. Additionally, the boundaries of the system consist of perfectly conducting walls. On the surface of the walls we have

$$\mathbf{n} \times \mathbf{E}(t) = 0, \tag{2.2a}$$

$$\mathbf{n} \cdot \mathbf{H}(t) = 0, \tag{2.2b}$$

with $\mathbf{n}$ denoting the vector normal to a boundary of the surface. The conditions Eq. (2.2) imply that the normal component of the magnetic field and the tangential components of the electric field vanish at the boundary [1]. Later, all these restrictions will be dropped.

The Maxwell Equations now reduce to

$$\frac{\partial}{\partial t}\mathbf{H}(t) = -\frac{1}{\mu}\nabla \times \mathbf{E}(t), \tag{2.3a}$$

$$\frac{\partial}{\partial t}\mathbf{E}(t) = \frac{1}{\varepsilon}\nabla \times \mathbf{H}(t), \tag{2.3b}$$

a set of first-order coupled differential equations.

Some algorithms, which solve the time *in*dependent Maxwell Equations, first decouple the TDME into equations for the fields separately, by substitution

$$\frac{\partial^2}{\partial t^2}\mathbf{H}(t) = -\frac{1}{\mu}\nabla \times \frac{1}{\varepsilon}\nabla \times \mathbf{H}(t), \tag{2.4a}$$

$$\frac{\partial^2}{\partial t^2}\mathbf{E}(t) = \frac{1}{\varepsilon}\nabla \times \frac{1}{\mu}\nabla \times \mathbf{E}(t). \tag{2.4b}$$

If ε and/or μ does not depend on space, these equations reduce to a wave equation. For example, if $\mu = \mu_0$ and $\varepsilon = \varepsilon_0$ the magnetic field obeys

$$\frac{1}{c^2}\frac{\partial^2}{\partial t^2}\mathbf{H}(t) = \nabla^2\mathbf{H}(t), \tag{2.5}$$

where we made use of the vector identity

$$\nabla \times \nabla \times f = \nabla(\nabla \cdot f) - \nabla^2 f \tag{2.6}$$

and the fact that the divergence $\nabla \cdot f$ occurring in this identity is zero for both fields.

It can be shown that the operator $-(1/\mu)\nabla \times (1/\varepsilon)\nabla\times$, occurring in equation (2.4a), is Hermitian only if μ is constant, with respect to the inner product

$$\langle \mathbf{F}|\mathbf{G}\rangle = \int_V \mathbf{F} \cdot \mathbf{G}\, dr, \tag{2.7}$$

where V denotes the volume of the enclosing box. This property is important in constructing an algorithm for solving the eigenvalue problem Eq. (2.4a). This equation is also the starting point of the so-called plane wave expansion method [3], and its applicability is consequently limited to systems with constant μ. For our purposes, Eqs. (2.4a) and (2.4b) are unusable, as they are second-order-in-time partial derivative equations.

Some important symmetries of the Maxwell Equations can be made explicit by introducing the fields

$$\mathbf{X}(t) = \sqrt{\mu}\mathbf{H}(t), \quad \text{and} \quad \mathbf{Y}(t) = \sqrt{\varepsilon}\mathbf{E}(t). \tag{2.8}$$

In terms of the fields $\mathbf{X}(t)$ and $\mathbf{Y}(t)$, the TDME (Eqs. (2.3a) and (2.3b)) become, in matrix notation

$$\frac{\partial}{\partial t}\begin{pmatrix} \mathbf{X}(t) \\ \mathbf{Y}(t) \end{pmatrix} = \begin{pmatrix} 0 & -\frac{1}{\sqrt{\mu}}\nabla \times \frac{1}{\sqrt{\varepsilon}} \\ \frac{1}{\sqrt{\varepsilon}}\nabla \times \frac{1}{\sqrt{\mu}} & 0 \end{pmatrix}\begin{pmatrix} \mathbf{X}(t) \\ \mathbf{Y}(t) \end{pmatrix} \equiv \mathcal{H}\begin{pmatrix} \mathbf{X}(t) \\ \mathbf{Y}(t) \end{pmatrix}. \tag{2.9}$$

It is easy to show that $\mathcal{H}$ is skew-symmetric, i.e. $\mathcal{H}^T = -\mathcal{H}$, with respect to the inner product as defined in Eq. (2.7):

$$\mathcal{H}^T = \begin{pmatrix} 0 & \left(\frac{1}{\sqrt{\varepsilon}}\nabla \times \frac{1}{\sqrt{\mu}}\right)^T \\ -\left(\frac{1}{\sqrt{\mu}}\nabla \times \frac{1}{\sqrt{\varepsilon}}\right)^T & 0 \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{\sqrt{\mu}}\nabla \times \frac{1}{\sqrt{\varepsilon}} \\ -\frac{1}{\sqrt{\varepsilon}}\nabla \times \frac{1}{\sqrt{\mu}} & 0 \end{pmatrix} = -\mathcal{H} \tag{2.10}$$

Writing $\Psi(t) = (\mathbf{X}(t), \mathbf{Y}(t))^T$, Eq. (2.9) becomes

$$\frac{\partial}{\partial t}\Psi(t) = \mathcal{H}\Psi(t). \tag{2.11}$$

The TDME are now reduced to a first-order differential equation, with its formal solution given by

$$\Psi(t) = e^{t\mathcal{H}}\Psi(0) \equiv \mathcal{U}(t)\Psi(0), \tag{2.12}$$

where $\Psi(0)$ represents the initial state of the EM fields and the operator $\mathcal{U}$ determines their time evolution. By construction

$$\|\Psi(t)\|^2 = \langle\Psi(t)|\Psi(t)\rangle = \int_V \left[\varepsilon\mathbf{E}^2(\mathbf{r},t) + \mu\mathbf{H}^2(\mathbf{r},t)\right] d\mathbf{r}, \tag{2.13}$$

relating the length of $\Psi(t)$ to the energy density

$$w(t) \equiv \varepsilon\mathbf{E}^2(t) + \mu\mathbf{H}^2(t), \tag{2.14}$$

of the EM fields. Since $\mathcal{H}$ is skew-symmetric the time evolution operator $\mathcal{U}$ is an orthogonal transformation:

$$\mathcal{U}(t)^T = \mathcal{U}(-t) = \mathcal{U}^{-1}(t) = e^{-t\mathcal{H}}, \tag{2.15}$$

and it follows that

$$\langle\mathcal{U}(t)\Psi(0)|\mathcal{U}(t)\Psi(0)\rangle = \langle\Psi(t)|\Psi(t)\rangle = \langle\Psi(0)|\Psi(0)\rangle. \tag{2.16}$$

Hence, the time evolution operator $\mathcal{U}(t)$ rotates the vector $\Psi(t)$ without changing its length $\|\Psi\|$. In physical terms, this means that the energy density of the electromagnetic (EM) fields does not change with time, as can be expected on physical grounds [1].

The properties of $\mathcal{H}$ and $\mathcal{U}$ in continuum have been explained now, and we can proceed with the next steps of the scheme to construct new algorithms, which comprise the conservation of the properties of the operators during spatial discretization and time integration. It is convenient to start with the last step, because the spatial discretization is identical for all new algorithms. So, at this point, we assume that the continuum problem described by $\mathcal{H}$ is mapped onto a lattice problem defined by matrix H.

2.2 Time integration

Formally, the time evolution of the EM fields on the lattice is given by

$$\Psi(t+\tau) = e^{\tau H}\Psi(t) = U(\tau)\Psi(t). \tag{2.17}$$

The key issue is to find an approximation $\tilde{U}(\tau)$ for $U(\tau)$ while conserving the orthogonality of the operator. In general, there are many methods that provide approximations to exponential functions. The difficulty to apply these methods to the present problem lies in the restriction to conserve orthogonality. For example, a simple truncation of the Taylor expansion of $U(\tau)$ [16, 17]

$$U(\tau) = e^{\tau H} = \sum_{n=0}^{\infty} \frac{(\tau H)^n}{n!}, \tag{2.18}$$

to first order in τ yields the Euler scheme:

$$\tilde{U}(\tau) = I + \tau H, \tag{2.19}$$

where I denotes the identity operator. As $\tilde{U}(\tau)\tilde{U}(\tau)^T = I - (\tau H)^2 \neq I$ for $\tau \neq 0$, it is clear that the matrix Eq. (2.19) is not orthogonal. Making use of the symmetry of H and the positivity of the inner product, we find that $\langle\Psi(\tau)|\Psi(\tau)\rangle = \langle\tilde{U}(\tau)\Psi(0)|\tilde{U}(\tau)\Psi(0)\rangle = \langle\Psi(0)|\Psi(0)\rangle + \tau^2\langle H\Psi(0)|H\Psi(0)\rangle > \langle\Psi(0)|\Psi(0)\rangle$, implying that the norm of Ψ is not conserved. Hence, the scheme Eq. (2.19) is unstable, a fact which is of course well-known [18]. The same line of reasoning could be used to exclude e.g. the Runga-Kutta method (see also [18]).

As an example of an existing numerical method that is unconditionally stable, we mention the Cranck-Nicholson method [18–20], which is of the form

$$\tilde{U}(\tau) = (I + \tau H/2)(I - \tau H/2)^{-1}. \tag{2.20}$$

It is a popular method to solve for example the diffusion equation. Unfortunately, the method is implicit since it requires the matrix inversion $(I - \tau H/2)^{-1}$.

2.2.1 Yee time integration

The well-known Yee algorithm does not approximate $U(\tau)$ directly, but starts from the partial differential equations themselves. If we denote the value of an arbitrary function $f(\mathbf{r}, t)$, at coordinate $(i\Delta x, j\Delta y, k\Delta z)$ in space and $n\Delta t$ in time by $f^n(i, j, k)$ then the time derivative of $f^n(i, j, k)$ can be written as

$$\frac{\partial}{\partial t} f^n(i, j, k) = \frac{f^{n+1/2}(i, j, k) - f^{n-1/2}(i, j, k)}{\Delta t} + O((\Delta t)^2). \tag{2.21}$$

A closer look at the curl equations (2.1) reveals that the calculation of the time derivative of an $\mathbf{E}$ or $\mathbf{H}$ component only requires knowledge of values of the $\mathbf{H}$ resp. $\mathbf{E}$ fields. This motivates for a leapfrog timestepping approach, where alternatingly the $\mathbf{E}$ field and $\mathbf{H}$ field is updated. One of the fields is then only defined at even time instances (labeled by n), and consequently the other field at the odd time instances. Combining the spatial and time derivatives, the explicit update equations can be easily derived. For instance, for the $\mathbf{E}_x$ component we have

$$\begin{aligned} E_x^{n+1}(i, j, k) \quad = \quad & E_x^n(i, j, k) + \frac{\Delta t}{\varepsilon(i, j, k)} \left[\frac{H_z^{n+1/2}(i, j+1, k) - H_z^{n+1/2}(i, j-1, k)}{\Delta y} \right. \\ & \left. - \frac{H_y^{n+1/2}(i, j, k+1) - H_y^{n+1/2}(i, j, k-1)}{\Delta z} \right]. \end{aligned} \tag{2.22}$$

It is instructive to cast the resulting algorithm into an operator form [14]

$$\begin{pmatrix} \mathbf{H}(t+\tau) \\ \mathbf{E}(t+\tau/2) \end{pmatrix} = (I + \tau A)(I + \tau B) \begin{pmatrix} \mathbf{H}(t) \\ \mathbf{E}(t-\tau/2) \end{pmatrix}, \tag{2.23}$$

where

$$A = \begin{pmatrix} 0 & -\frac{1}{\mu}\nabla\times \\ 0 & 0 \end{pmatrix}, \tag{2.24}$$

and

$$B = \begin{pmatrix} 0 & 0 \\ \frac{1}{\varepsilon}\nabla\times & 0 \end{pmatrix}. \tag{2.25}$$

We observe that

$$\tilde{U}(\tau) = (I + \tau A)(I + \tau B) = \exp(\tau A)\exp(\tau B), \tag{2.26}$$

since $A^2 = B^2 = 0$.

By using the usual substitutions of the fields, we can also write

$$\begin{pmatrix} \mathbf{X}(t+\tau) \\ \mathbf{Y}(t+\tau/2) \end{pmatrix} = (I + \tau C)(I - \tau C^T)\begin{pmatrix} \mathbf{X}(t) \\ \mathbf{Y}(t-\tau/2) \end{pmatrix}, \tag{2.27}$$

with

$$C = \begin{pmatrix} 0 & 0 \\ \frac{1}{\sqrt{\varepsilon}}\nabla\times\frac{1}{\sqrt{\mu}} & 0 \end{pmatrix}. \tag{2.28}$$

And since $C^2 = 0$, the time evolution operator simplifies to

$$\tilde{U}(\tau) = (I + \tau C)(I - \tau C^T) = \exp(\tau C)\exp(-\tau C^T). \tag{2.29}$$

2.2.2 Chebyshev time integration

A well-known alternative for timestepping is to use Chebyshev polynomials to construct approximations to time-evolution operators [21–25]. Applied to the case here, we exploit the fact that the eigenvalues of an anti-symmetric matrix are all imaginary (see Ref. [26]).

For the expansion of a scalar function in Chebyshev polynomials we have [27]

$$f(x) = \frac{1}{2}a_0 T_0(x) + \sum_{n=1}^{\infty} a_n T_n(x), \tag{2.30}$$

where a_n are the expansion coefficients

$$a_n = \frac{2}{\pi}\int_0^{\pi} \cos(n\theta) f(\cos\theta) d\theta. \tag{2.31}$$

The Chebyshev polynomials T_n are given by

$$T_n(x) = \cos(n\cos^{-1}(x)). \tag{2.32}$$

A useful recursion relation to calculate these functions is

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad \text{with } T_0(x) = 1 \text{ and } T_1(x) = x. \tag{2.33}$$

In our case, we want to approximate the operator $U(t) = \exp(tH)$ where H is a real anti-symmetric matrix, and minor modifications must be made to the scalar Chebyshev expansion. First of all, matrix H must be "normalized", since the range of definition for the functions $T_n(x)$ is from -1 to 1 (see Eq. (2.32)). The eigenvalues of skew-symmetric matrix

H are all purely imaginary. Therefore, matrix $A = -iH$ is Hermitian and all its eigenvalues are real and lie in the range $[-\rho(A), \rho(A)]$, where $\rho(A)$ is the spectral radius[2] of A, i.e. the largest absolute eigenvalue of A . The largest eigenvalue of A is hard to find, but for our purposes it suffices to use an upperbound, given by inequality [26]

$$\rho(A) \leq \|A\|_\infty = \max_i \sum_j |A_{ij}|. \tag{2.34}$$

Note that since A is Hermitian, we also have $\|A\|_\infty = \|A\|_1 = \max_j \sum_i |A_{ij}|$. These norms are easy to calculate for any sparse matrix. Thus, the eigenvalues of matrix

$$B \equiv A/\|A\|_1 \tag{2.35}$$

will all lie in the range $[-1, 1]$, by construction. The normalization of H must be compensated for by scaling the time:

$$z = t\|A\|_1. \tag{2.36}$$

Operating on state $\Psi(0)$, the expansion now becomes

$$\Psi(t) = \exp(tH)\Psi(0) = \exp(izB)\Psi(0) = \left[\frac{1}{2}a_0(z)I + \sum_{n=1}^{\infty} a_n(z)T_n(B)\right]\Psi(0), \tag{2.37}$$

where I is the identity matrix and $T_n(B)$ are matrix valued Chebyshev polynomials. The expansion coefficients Eq. (2.31) evaluate to nth order Bessel functions J_n:

$$a_n(z) = \frac{2}{\pi}\int_0^{\pi} \cos(n\theta)\exp(iz\cos\theta)d\theta = 2J_n(z)i^n, \tag{2.38}$$

giving

$$\exp(tH)\Psi(0) = \left[J_0(z)I + 2\sum_{n=1}^{\infty} J_n(z)i^n T_n(B)\right]\Psi(0), \tag{2.39}$$

Finally, the recursion relation Eq. (2.33) becomes, in terms of $\widetilde{T}_n(B) = i^n T_n(B)$,

$$\widetilde{T}_0(B)\Psi(0) = \Psi(0), \tag{2.40a}$$

$$\widetilde{T}_1(B)\Psi(0) = iB\Psi(0), \tag{2.40b}$$

$$\widetilde{T}_{n+1}(B)\Psi(0) = 2iB\widetilde{T}_n(B)\Psi(0) + \widetilde{T}_{n-1}(B)\Psi(0) \quad \text{for } n \geq 1. \tag{2.40c}$$

Due to the fact that matrix B is purely imaginary, it follows from the above recursion relation that $\widetilde{T}_n(B)\Psi$ and thus $\Psi(t)$ will be real valued, as should be the case.

In practice, the summation in Eq. (2.39) will be truncated, and we will only include $m + 1$ polynomials in the approximation:

$$\exp(tH)\Psi(0) \approx \left[J_0(z)I + 2\sum_{n=1}^{m} J_n(z)\widetilde{T}_n(B)\right]\Psi(0). \tag{2.41}$$

[2] if matrix A would not have a complete set of eigenfunctions one could use the more general 2-norm to normalize the matrix, defined by $\|A\|_2 = \sqrt{\rho(A^T A)}$.

The computation of one timestep consequently amounts to carrying out m repetitions of recursion relation Eq. (2.40) to obtain the final state. This is a simple procedure: only the multiplication of a vector with a sparse matrix and the summation of vectors are involved. Note that the intermediate results have no physical meaning.

If the coefficients $a_n(z)$ are computed to sufficiently high accuracy, then the error will depend on the number of expansion terms m. It can be fixed by control parameter κ, defined such that

$$|J_n(z)| < \kappa, \quad \text{for all } n > m. \tag{2.42}$$

However, since $|\widetilde{T}_n(B)| \leq 1$ and $|J_n(z)| \leq |z|^n/2^n n!$ for z real [28], the resulting error vanishes exponentially fast, and a value of $\kappa = 10^{-13}$ will ensure that enough expansion terms are included in the approximation, and consequently the error will be very small. In the next chapter, section 3.1, we will examine the error and stability of the Chebyshev algorithm in more detail.

2.2.3 Lie-Trotter-Suzuki time integration

A systematic approach to construct orthogonal approximations to matrix exponentials in general is to make use of the Lie-Trotter-Suzuki formula [29, 30]

$$e^{t(H_1+\ldots+H_p)} = \lim_{m\to\infty}\left(\prod_{i=1}^{p} e^{tH_i/m}\right)^m, \tag{2.43}$$

and generalizations thereof [31, 32]. Applied to the case of interest here, the success of this approach relies on the basic but rather trivial premise that the matrix H can be written as

$$H = \sum_{i=1}^{p} H_i, \tag{2.44}$$

where each of the matrices H_i is real and skew-symmetric.

The expression Eq. (2.43) suggests that

$$U_1(\tau) = e^{\tau H_1}\ldots e^{\tau H_p}, \tag{2.45}$$

might be a good approximation to $U(\tau)$ if τ is sufficiently small. Most importantly, if all the H_i are real and skew-symmetric, $U_1(\tau)$ is orthogonal by construction. Therefore, by construction, a numerical scheme based on Eq. (2.45) will be unconditionally stable. Using the fact that both $U(\tau)$ and $U_1(\tau)$ are orthogonal matrices, it can be shown that [33]

$$\|U(\tau) - U_1(\tau)\| \leq \frac{\tau^2}{2}\sum_{i<j}\|[H_i, H_j]\|, \tag{2.46}$$

where $[H_i, H_j] = H_iH_j - H_jH_i$. From Eq. (2.46) it follows that, in general, the Taylor series of $U(\tau)$ and $U_1(\tau)$ are identical up to first order in τ. We will call $U_1(\tau)$ the first-order approximation to $U(\tau)$.

The product-formula approach provides simple, systematic procedures to improve the accuracy of the approximation to $U(\tau)$ without changing its fundamental symmetries. For example the orthogonal matrix

$$U_2(\tau) = U_1(-\tau/2)^T U_1(\tau/2) = e^{\tau H_p/2}\ldots e^{\tau H_1/2}e^{\tau H_1/2}\ldots e^{\tau H_p/2}, \tag{2.47}$$

is a second-order approximation to $U(\tau)$ [31,32]. Suzuki's fractal decomposition approach [31] gives a general method to construct higher-order approximations based on $U_1(\tau)$ or $U_2(\tau)$. A particularly useful fourth-order approximation is given by [31]

$$U_4(\tau) = U_2(a\tau)U_2(a\tau)U_2((1-4a)\tau)U_2(a\tau)U_2(a\tau), \tag{2.48}$$

where $a = 1/(4 - 4^{1/3})$. The approximations Eqs. (2.45), (2.47), and (2.48) have proven to be very useful in many applications [30, 32–42] and, as we show below, turn out to be equally useful for solving the TDME.

In order to use the Lie-Trotter-Suzuki approach for the temporal integration, we thus have to keep in mind that the spatially discretized matrix H representing the TDME must be real, skew-symmetric and be decomposable into a sum of p real, skew-symmetric matrices H_i. Additionally, the matrix exponentials $\exp(\tau H_1)$, ..., $\exp(\tau H_p)$ should be calculable efficiently. If we meet these requirements, an efficient implementation of the first-order scheme is obtained and the higher-order algorithms Eqs. (2.47) and (2.48) are easily derived.

2.3 Spatial discretization

We will now explain how to conserve the skew-symmetry of operator $\mathcal{H}$, when the fields are mapped onto a lattice and the spatial derivatives are discretized. By adopting the Yee lattice and the central-difference approximation for the spatial derivatives, we will show that the resulting time-evolution matrix H is skew-symmetric and thus can serve as a basis for the time integration schemes of both the Chebyshev and Lie-Trotter-Suzuki algorithms. The latter approach requires an additional decomposition of matrix H, so this will be shown explicitly for one, two and three spatial dimensions.

Initially, we limit ourselves to the discussion of a second-order spatially accurate discretization scheme, but later on some enhancements will be introduced: a fourth-order spatial approximation and a variable mesh. For several realistic problems these enhancements can save a lot of computational resources. Throughout the rest of this thesis, the Lie-Trotter-Suzuki algorithms of order n in time and m in space will be referred to by the short hand notation TnSm.

For pedagogical reasons it is expedient to start by considering the simplest case first: a one-dimensional (1D) system. After the implementation of the Lie-Trotter-Suzuki algorithm, we will show that the staggered Yee grid is the most efficient grid, but not a compulsory one. In this context we will also demonstrate how the conventional Yee algorithm can be modified to a non staggered-in-time algorithm. Finally, we extend the strategy adopted for one dimension to higher spatial dimensions.

2.3.1 Implementation 1D

We consider a 1D system along the x-direction. Accordingly, Maxwell's Equations contain no partial derivatives with respect to y or z and ε and μ do not depend on y or z. Under these conditions, the TDME reduce to two independent sets of first-order differential equations [1],

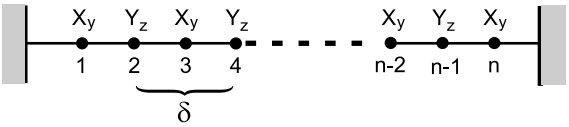

Figure 2-1: Positions of the two TM-mode field components on the one-dimensional grid.

known as the transverse magnetic (TM) mode

$$\frac{\partial}{\partial t}H_y(t) = \frac{1}{\mu}\frac{\partial}{\partial x}E_z(t), \tag{2.49a}$$

$$\frac{\partial}{\partial t}E_z(t) = \frac{1}{\varepsilon}\frac{\partial}{\partial x}H_y(t), \tag{2.49b}$$

and transverse electric (TE) mode

$$\frac{\partial}{\partial t}H_z(t) = -\frac{1}{\mu}\frac{\partial}{\partial x}E_y(t), \tag{2.50a}$$

$$\frac{\partial}{\partial t}E_y(t) = -\frac{1}{\varepsilon}\frac{\partial}{\partial x}H_z(t). \tag{2.50b}$$

As the equations of the TE- and TM-mode only differ by a sign, we can restrict our considerations to the TM-mode and obtain the result for the TE-mode by reversing the time.

From Eq. (2.9) it follows that the magnetic field $H_y(x,t) = X_y(x,t)/\sqrt{\mu(x)}$ and the electric field $E_z(x,t) = Y_z(x,t)/\sqrt{\varepsilon(x)}$ of the TM-mode are solutions of

$$\frac{\partial}{\partial t}X_y(x,t) = \frac{1}{\sqrt{\mu(x)}}\frac{\partial}{\partial x}\left(\frac{Y_z(x,t)}{\sqrt{\varepsilon(x)}}\right), \tag{2.51a}$$

$$\frac{\partial}{\partial t}Y_z(x,t) = \frac{1}{\sqrt{\varepsilon(x)}}\frac{\partial}{\partial x}\left(\frac{X_y(x,t)}{\sqrt{\mu(x)}}\right). \tag{2.51b}$$

Note that in 1D, the divergence of $H_y(x,t)$ and $E_z(x,t)$ are both zero. Hence, Eqs. (2.1c) and (2.1d) are automatically satisfied.

Using the second-order central-difference approximation to the first derivative with respect to x, we obtain the spatially discretized equations

$$\frac{\partial}{\partial t}X_y(i,t) = \frac{1}{\delta\sqrt{\mu_i}}\left(\frac{Y_z(i+1,t)}{\sqrt{\varepsilon_{i+1}}} - \frac{Y_z(i-1,t)}{\sqrt{\varepsilon_{i-1}}}\right), \tag{2.52a}$$

$$\frac{\partial}{\partial t}Y_z(j,t) = \frac{1}{\delta\sqrt{\varepsilon_j}}\left(\frac{X_y(j+1,t)}{\sqrt{\mu_{j+1}}} - \frac{X_y(j-1,t)}{\sqrt{\mu_{j-1}}}\right), \tag{2.52b}$$

where the integer i labels the grid points and δ denotes the distance between two next-nearest neighbor lattice points (hence the absence of a factor two in the nominator). For notational simplicity we will, from now on, specify the spatial coordinates through the lattice index i, e.g. $X_y(i,t)$ stands for $X_y(x = i\delta/2, t)$. Following Yee [5], it is convenient to adopt an *interleaved* grid, i.e. to assign $X_y(i,t)$ and $Y_z(j,t)$ to the odd, respectively, even

numbered lattice site, as shown in figure 2-1 for a grid of n points. Later, we will discuss a non-interleaved grid based approach. Equations (2.52a) and (2.52b) can now be combined into one equation of the form Eq. (2.11) by introducing the n-dimensional vector

$$\Psi(i,t) = \begin{cases} X_y(i,t) = \sqrt{\mu_i}\, H_y(i,t), & i \text{ odd} \\ Y_z(i,t) = \sqrt{\varepsilon_i}\, E_z(i,t), & i \text{ even} \end{cases} . \tag{2.53}$$

The vector $\Psi(t)$ describes both the magnetic and the electric field on the lattice points $i = 1, \ldots, n$. As usual, the i-th element of $\Psi(t)$ is given by the inner product $\Psi(i,t) = \mathbf{e}_i^T \cdot \Psi(t)$ where $\mathbf{e}_i$ denotes the i-th unit vector in the n-dimensional vector space. Using this notation, it is easy to show that Eqs. (2.52a) and (2.52b) reduce to

$$\frac{\partial}{\partial t}\Psi(t) = H\Psi(t), \tag{2.54}$$

where the matrix H is given by

$$H = \sum_{i=1}^{n-2}{}' \left[\beta_{i+1,i}\left(\mathbf{e}_i\, \mathbf{e}_{i+1}^T - \mathbf{e}_{i+1}\mathbf{e}_i^T\right) + \beta_{i+1,i+2}\left(\mathbf{e}_{i+1}\mathbf{e}_{i+2}^T - \mathbf{e}_{i+2}\mathbf{e}_{i+1}^T\right)\right], \tag{2.55}$$

with $\beta_{i,j} = 1/(\delta\sqrt{\varepsilon_i \mu_j})$ and the prime indicates that the sum is over odd integers only. In complete analogy to Eq. (2.12) the time evolution of $\Psi(t)$ is formally given by $\Psi(t) = U(t)\Psi(0)$ with $U(t) = \exp(tH)$.

The notation introduced above will prove most useful for the case of 2D and 3D for which it is rather cumbersome to write down matrix representations. For the 1D case it is not difficult and in fact very instructive to write down the matrix H explicitly. Indeed, we have

$$H = \begin{pmatrix} 0 & \beta_{2,1} & & & \\ -\beta_{2,1} & 0 & \beta_{2,3} & & \\ & \ddots & \ddots & \ddots & \\ & & -\beta_{n-1,n-2} & 0 & \beta_{n-1,n} \\ & & & -\beta_{n-1,n} & 0 \end{pmatrix}, \tag{2.56}$$

and we immediately see that H is skew-symmetric by construction. Furthermore, for n odd we have

$$\frac{\partial}{\partial t}\Psi(1,t) = \beta_{2,1}\Psi(2,t) \quad \text{and} \quad \frac{\partial}{\partial t}\Psi(n,t) = -\beta_{n-1,n}\Psi(n-1,t), \tag{2.57}$$

such that the electric field vanishes at the boundaries ($Y_z(0,t) = Y_z(n+1,t) = 0$, see also figure 2-1), as required by the boundary conditions Eq. (2.2). For this reason we only consider the case of n odd in the sequel.

The final step in the construction of the Lie-Trotter-Suzuki algorithms is to decompose H. Guided by previous work on Schrödinger and diffusion problems [33, 36, 41, 42], we split H into two parts, i.e. $H = H_1 + H_2$, where

$$H_1 = \sum_{i=1}^{n-2}{}' \beta_{i+1,i}\left(\mathbf{e}_i\, \mathbf{e}_{i+1}^T - \mathbf{e}_{i+1}\mathbf{e}_i^T\right), \tag{2.58a}$$

$$H_2 = \sum_{i=1}^{n-2}{}' \beta_{i+1,i+2}\left(\mathbf{e}_{i+1}\mathbf{e}_{i+2}^T - \mathbf{e}_{i+2}\mathbf{e}_{i+1}^T\right), \tag{2.58b}$$

i.e. we divide the lattice into odd and even numbered cells. In matrix notation we have

$$H_1 = \begin{pmatrix} 0 & \beta_{2,1} & & & & & & \\ -\beta_{2,1} & 0 & 0 & & & & & \\ & 0 & 0 & \beta_{4,3} & & & & \\ & & -\beta_{4,3} & 0 & 0 & & & \\ & & & 0 & \ddots & 0 & & \\ & & & & 0 & 0 & \beta_{n-1,n-2} & \\ & & & & & -\beta_{n-1,n-2} & 0 & 0 \\ & & & & & & 0 & 0 \end{pmatrix} \tag{2.59}$$

and

$$H_2 = \begin{pmatrix} 0 & 0 & & & & & & & \\ 0 & 0 & \beta_{2,3} & & & & & & \\ & -\beta_{2,3} & 0 & 0 & & & & & \\ & & 0 & \ddots & 0 & & & & \\ & & & 0 & 0 & \beta_{n-3,n-2} & & & \\ & & & & -\beta_{n-3,n-2} & 0 & 0 & & \\ & & & & & 0 & 0 & \beta_{n-1,n} & \\ & & & & & & -\beta_{n-1,n} & 0 \end{pmatrix}. \tag{2.60}$$

Clearly both H_1 and H_2 are skew-symmetric block-diagonal matrices, containing one 1×1 matrix and $(n-1)/2$ real, 2×2 skew-symmetric matrices.

According to the general theory given above, the first-order algorithm defined by

$$U_1(\tau) = e^{\tau H_1} e^{\tau H_2}. \tag{2.61}$$

is all that is needed to construct unconditionally stable second and higher-order algorithms. As the matrix exponential of a block-diagonal matrix is equal to the block-diagonal matrix of the matrix exponentials of the individual blocks, the numerical calculation of $e^{\tau H_1}$ (or $e^{\tau H_2}$) reduces to the calculation of $(n-1)/2$ matrix exponentials of 2×2 matrices. The matrix exponential of a typical 2×2 matrix appearing in $e^{\tau H_1}$ or $e^{\tau H_2}$ is given by

$$\exp\left[\alpha \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\right] \begin{pmatrix} \Psi(i,t) \\ \Psi(j,t) \end{pmatrix} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} \Psi(i,t) \\ \Psi(j,t) \end{pmatrix}. \tag{2.62}$$

An exact expression for $U_1(\tau)$ in terms of an ordered product of matrix exponentials becomes

$$U_1(\tau) = \left\{ \prod_{i=1}^{n-2}{}' \exp\left[\beta_{i+1,i}\left(\mathbf{e}_i\, \mathbf{e}_{i+1}^T - \mathbf{e}_{i+1}\mathbf{e}_i^T\right)\right] \right\} \cdot \left\{ \prod_{i=1}^{n-2}{}' \exp\left[\beta_{i+1,i+2}\left(\mathbf{e}_{i+1}\mathbf{e}_{i+2}^T - \mathbf{e}_{i+2}\mathbf{e}_{i+1}^T\right)\right] \right\}. \tag{2.63}$$

The ordering of the products in Eq. (2.63) is by no means the only correct implementation possible. In some cases, where for example the physical system being modeled is shaped irregularly, it is more convenient to abandon this ordering. Because if we still want to use a Cartesian coordinate system with fixed dimensions (a rectangle in two dimensions and a box in three dimensions) for such physical structures, then many grid points may not lie inside the system we want to model. An ordering in the form of labeling all pairs of points to be rotated in terms of odd or even becomes unusable. The plane rotations Eq. (2.62)

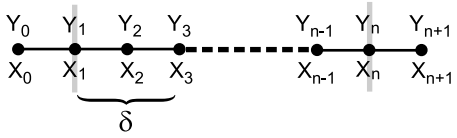

Figure 2-2: Positions of the two TM-mode field components on the one-dimensional non-interleaved grid.

are then performed by simply processing an arbitrarily ordered list S of pairs of EM field vector elements using

$$U_1(\tau) = \prod_S \exp\left(\tau\beta_{i,j}(\mathbf{e}_i\,\mathbf{e}_j^T - \mathbf{e}_j\,\mathbf{e}_i^T)\right), \tag{2.64}$$

instead of the odd-even decomposition Eq. (2.63) for which $S = \{(1,2),(3,4),\ldots,(n-2,n-1),(2,3),(4,5),\ldots,(n-1,n)\}$. The intrinsic possibility of parallelization (the even/odd labeled pairs can all be rotated independently) is lost if the order of iteration is changed in this way. However, later we will see that in practice it is more convenient to use other (read: larger scale) methods of parallelization.

Non-staggered spatial grid

In some cases, it might be preferred not to use an interleaved grid. The method described above to construct the algorithm might give the impression that the use of an interleaved grid is compulsory. This is not the case[3]. We will show that it is indeed possible to use EM fields all defined on identical lattice points, but that this does not open any new possibilities.

The non-interleaved grid requires some extra virtual points outside the boundary to satisfy the boundary conditions, as shown in figure 2-2, which disappear again later. The cavity boundaries are located at lattice index $i = 1$ and $i = n$. For the magnetic field Dirichlet boundary conditions apply: $(\partial/\partial x)X_1 = 0$ which implies $X_0 = X_2$, and $(\partial/\partial x)X_n = 0$ giving $X_{n-1} = X_{n+1}$. For the electric field we have Von Neumann boundary conditions: $Y_1 = Y_n = 0$, and from the partial differential equations (2.52a) and (2.52b) it follows that $Y_0 = -Y_2$ and $Y_{n-1} = -Y_{n+1}$. The virtual points can now be omitted from the matrix equation, since their values are always known. The vector describing the EM fields on the lattice at time t contains now the elements

$$\Psi(t) = [X_1, Y_1, X_2, Y_2, \ldots, X_{n-1}, Y_{n-1}, X_n, Y_n]^T. \tag{2.65}$$

If we again combine equations (2.52a) and (2.52b) into one matrix equation, the result is

[3] this argument was pointed out to us by Robert D. Ryne.

slightly different than the matrix in Eq. (2.56). For example, for $n = 8$ we have

$$H = \begin{pmatrix} 0 & 0 & 0 & 2\beta_{1,2} & & & & \\ 0 & 0 & 0 & 0 & & & & \\ 0 & -\beta_{2,1} & 0 & 0 & 0 & \beta_{2,3} & & \\ -\beta_{1,2} & 0 & 0 & 0 & \beta_{3,2} & 0 & & \\ & & 0 & -\beta_{3,2} & 0 & 0 & 0 & \beta_{4,5} \\ & & -\beta_{2,3} & 0 & 0 & 0 & \beta_{5,4} & 0 \\ & & & & 0 & 0 & 0 & 0 \\ & & & & -2\beta_{4,5} & 0 & 0 & 0 \end{pmatrix}, \tag{2.66}$$

The matrix consists of 4×4 blocks on the diagonal that are all skew-symmetric except for the first and last block. This lack of skew-symmetry does not affect the unconditional stability, however. For the first block, the matrix exponential becomes

$$\exp\begin{pmatrix} 0 & 0 & 0 & 2a \\ 0 & 0 & 0 & 0 \\ 0 & -b & 0 & 0 \\ -a & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} \cos(\sqrt{2}a) & 0 & 0 & \sqrt{2}\sin(\sqrt{2}a) \\ 0 & 1 & 0 & 0 \\ 0 & -b & 1 & 0 \\ -\frac{1}{2}\sin(\sqrt{2}a) & 0 & 0 & \sqrt{2}\cos(\sqrt{2}a) \end{pmatrix}, \tag{2.67}$$

where $a = \beta_{1,2}$ and $b = \beta_{2,1}$. Now, the central sub-matrix combining elements Y_1 and X_2 reduces to the identity matrix if one takes into account that $Y_1(t) = Y_1(0) = 0$, whereas for the outer elements holds that $\frac{1}{2}X_1^2 + Y_2^2 = \text{const}$. For the lower-right matrix analogous arguments apply and there we have that $\frac{1}{2}X_n^2 + Y_{n-1}^2 = \text{const}$. Summarizing, we can say that the resulting algorithm is unconditionally stable and conserves the energy density but with a factor 0.5 multiplying the magnetic fields on the boundary itself.

However, the advantages this new grid setup brings are small. In fact, using the co-located grid one needs twice as many EM field points to achieve the same level of accuracy as an algorithm based on an interleaved grid, due to the fact that the matrix in Eq. (2.66) can be decomposed into two commuting parts. Therefore, the equations of motion decouple into two independent sets of equations, which apply each to one of the two subsets of Ψ, namely $\{X_1, Y_2, \ldots, X_{n-1}, Y_n\}$ and $\{Y_1, X_2, \ldots, Y_{n-1}, X_n\}$. The solution of the time evolution of each subset is as good as the solution of the 'normal' interleaved grid based algorithm, but the solutions cannot be combined to increase the overall accuracy. So, apart from the aesthetically more appealing fact that EM fields are defined on each lattice point, there does not seem to be a compelling reason to use this grid setup.

Non-staggered Yee time integration

In the context of operator splitting techniques, it is convenient to revisit the Yee scheme. In section 2.2.1 we showed that the Yee scheme can be represented, in operator form, by

$$\Psi(t+\tau) = U^{\text{Yee}}(\tau)\Psi(t) = (I + \tau A)(I - \tau A^T)\Psi(t) \tag{2.68}$$

where $H = A - A^T$ and Ψ is a vector containing the spatially discretized fields at staggered-in-time instances. In one dimension, the matrix A is given explicitly by[4]

$$A = \frac{1}{\delta}\sum_{i=2}^{n-1}{}' \left(\beta_{i,i-1}\mathbf{e}_i\,\mathbf{e}_{i-1}^T - \beta_{i,i+1}\mathbf{e}_i\,\mathbf{e}_{i+1}^T\right). \tag{2.69}$$

[4] these arguments generalize to two- and three dimensional systems, although we will not show that explicitly.

Since $A^2 = 0$, we have

$$U^{\text{Yee}}(\tau) = \exp(\tau A)\exp(-\tau A^T), \tag{2.70}$$

an expression which holds *exactly*, not approximately. It is well known that the Yee time integration method is second order in time, but this holds if and only if the fields are defined staggered in time. Furthermore, if we interpret the fields not to be staggered-in-time, the Yee time evolution operator becomes a first-order approximation to the matrix exponential:

$$\exp(\tau H) = \exp(\tau(A - A^T)) \approx \exp(\tau A)\exp(-\tau A^T) = U_1^{\text{Yee}}(\tau) \tag{2.71}$$

With this insight we can enhance the non-staggered-in-time Yee algorithm to recover its original second-order accuracy, by using a second-order expansion for the splitting $H = A - A^T$:

$$U_2^{\text{Yee}}(\tau) = \exp(\tau A/2)\exp(-\tau A^T)\exp(\tau A/2). \tag{2.72}$$

Hence, we can eliminate the staggered-in-time nature of the Yee algorithm at a minimal computational cost, namely, by dividing the operation $\exp(\tau A)$ into two operations $\exp(\tau A/2)$, placed at beginning and the end, yielding a time evolution operator with second-order accuracy in time *working on non staggered-in-time fields*. Why? The effect of the operations with factor $\tau/2$ is that one of the fields (non staggered-in-time) is propagated over half a timestep, then the Yee algorithm is applied for the other field, and finally the first field is again propagated over half a timestep. We emphasize that there is no gain in sense of stability; for this new non staggered-in-time Yee algorithm, the same stability criteria hold as for the original one. However, one advantage besides the non staggered-in-time nature, is that higher order in time algorithms are now as easily constructed as for the Lie-Trotter-Suzuki algorithms. For example, a fourth-order algorithm would be (cf. Eq. (2.48))

$$U_4^{\text{Yee}}(\tau) = U_2^{\text{Yee}}(a\tau)U_2^{\text{Yee}}(a\tau)U_2^{\text{Yee}}((1-4a)\tau)U_2^{\text{Yee}}(a\tau)U_2^{\text{Yee}}(a\tau). \tag{2.73}$$

Typically, conventional higher-order Yee algorithms apply Runga-Kutta like schemes, which necessarily must store intermediate results [43].

2.3.2 Implementation 2D

Assuming translational invariance with respect to the z direction, the system effectively becomes 2D and the TDME separate into two sets of equations. The relevant EM fields for the TM-modes in 2D are $\Psi_{TM}(t) = (\mathbf{X}_x(x,y,t), \mathbf{X}_y(x,y,t), \mathbf{Y}_z(x,y,t))^T$, in terms of which TDME Eq.(2.9) reads

$$\frac{\partial}{\partial t}\Psi_{TM}(t) = \mathcal{H}\Psi_{TM}(t) = \begin{pmatrix} 0 & 0 & -\frac{1}{\sqrt{\mu}}\frac{\partial}{\partial y}\frac{1}{\sqrt{\varepsilon}} \\ 0 & 0 & \frac{1}{\sqrt{\mu}}\frac{\partial}{\partial x}\frac{1}{\sqrt{\varepsilon}} \\ -\frac{1}{\sqrt{\varepsilon}}\frac{\partial}{\partial y}\frac{1}{\sqrt{\mu}} & \frac{1}{\sqrt{\varepsilon}}\frac{\partial}{\partial x}\frac{1}{\sqrt{\mu}} & 0 \end{pmatrix}\Psi_{TM}(t)\,. \tag{2.74}$$

Similarly, the TE-mode fields are $\Psi_{TE}(t) = (\mathbf{Y}_x(x,y,t), \mathbf{Y}_y(x,y,t), \mathbf{X}_z(x,y,t))^T$, for which we have the matrix equation

$$\frac{\partial}{\partial t}\Psi_{TE}(t) = \mathcal{H}\Psi_{TE}(t) = \begin{pmatrix} 0 & 0 & \frac{1}{\sqrt{\varepsilon}}\frac{\partial}{\partial y}\frac{1}{\sqrt{\mu}} \\ 0 & 0 & -\frac{1}{\sqrt{\varepsilon}}\frac{\partial}{\partial x}\frac{1}{\sqrt{\mu}} \\ \frac{1}{\sqrt{\mu}}\frac{\partial}{\partial y}\frac{1}{\sqrt{\varepsilon}} & -\frac{1}{\sqrt{\mu}}\frac{\partial}{\partial x}\frac{1}{\sqrt{\varepsilon}} & 0 \end{pmatrix}\Psi_{TE}(t)\,. \tag{2.75}$$

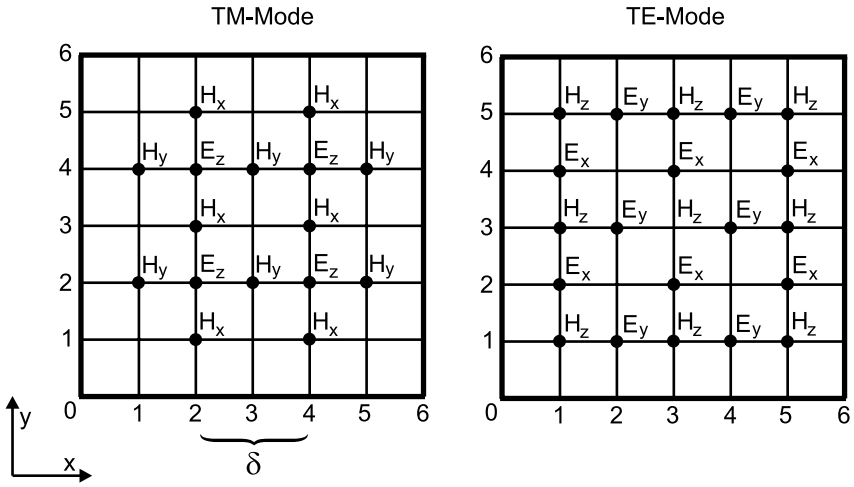

Figure 2-3: Positions of the TM and TE-mode EM field components on the two-dimensional grid for $n_x = 5$ and $n_y = 5$.

We discretize continuum space by simply reusing the one-dimensional lattice introduced above, as exemplified in figure 2-3. This construction automatically takes care of the boundary conditions if n_x and n_y are odd and yields a real skew-symmetric matrix H. In analogy with the 1D case, the elements of $\Psi_{TM}(t)$ and $\Psi_{TE}(t)$ are defined by

$$\Psi_{TM}(i,j,t) = \begin{cases} X_x(i,j,t) = \sqrt{\mu_{i,j}} H_x(i,j,t), & i \text{ even and } j \text{ odd} \\ X_y(i,j,t) = \sqrt{\mu_{i,j}} H_y(i,j,t), & i \text{ odd and } j \text{ even} \\ Y_z(i,j,t) = \sqrt{\varepsilon_{i,j}} E_z(i,j,t), & i \text{ even and } j \text{ even} \end{cases}, \tag{2.76a}$$

$$\Psi_{TE}(i,j,t) = \begin{cases} Y_x(i,j,t) = \sqrt{\varepsilon_{i,j}} E_x(i,j,t), & i \text{ odd and } j \text{ even} \\ Y_y(i,j,t) = \sqrt{\varepsilon_{i,j}} E_y(i,j,t), & i \text{ even and } j \text{ odd} \\ X_z(i,j,t) = \sqrt{\mu_{i,j}} H_z(i,j,t), & i \text{ odd and } j \text{ odd} \end{cases}. \tag{2.76b}$$

In the remainder of this section, we will limit the discussion to the construction of an algorithm for the TM-modes only, as the TE-modes can be treated in exactly the same manner. Discretization of the differential operators that appear in Eq. (2.74) yield expressions that have the same structure as Eq. (2.56), with extra subscripts to account for the second spatial dimension. It follows that on the lattice,

$$\frac{\partial}{\partial t}\Psi_{TM}(t) = H\Psi_{TM}(t) = \sum_{i=1}^{n_x-2}{}' \sum_{j=1}^{n_y-2}{}' \left[H^{(x)}(i,j) + H^{(y)}(i,j)\right] \Psi_{TM}(t), \tag{2.77}$$

where

$$H^{(x)}(i,j) = + \frac{\mathbf{e}_{i,j+1}\mathbf{e}_{i+1,j+1}^T - \mathbf{e}_{i+1,j+1}\mathbf{e}_{i,j+1}^T}{\delta\sqrt{\varepsilon_{i+1,j+1}\mu_{i,j+1}}} + \frac{\mathbf{e}_{i+1,j+1}\mathbf{e}_{i+2,j+1}^T - \mathbf{e}_{i+2,j+1}\mathbf{e}_{i+1,j+1}^T}{\delta\sqrt{\varepsilon_{i+1,j+1}\mu_{i+2,j+1}}}, \tag{2.78a}$$

$$H^{(y)}(i,j) = - \frac{\mathbf{e}_{i+1,j}\mathbf{e}_{i+1,j+1}^T - \mathbf{e}_{i+1,j+1}\mathbf{e}_{i+1,j}^T}{\delta\sqrt{\varepsilon_{i+1,j+1}\mu_{i+1,j}}} - \frac{\mathbf{e}_{i+1,j+1}\mathbf{e}_{i+1,j+2}^T - \mathbf{e}_{i+1,j+2}\mathbf{e}_{i+1,j+1}^T}{\delta\sqrt{\varepsilon_{i+1,j+2}\mu_{i+1,j+1}}}, \tag{2.78b}$$

and the superscripts (x) and (y) refer to the derivative with respect to x and y respectively. Note that we use the pair (i, j) to label the $n_x n_y$ unit vectors $\mathbf{e}_{i,j}$. In complete analogy with the 1D case, we split Eqs. (2.78a) and (2.78b) into two parts to obtain the first-order approximation to $U(\tau)$:

$$U_1(\tau) \quad = \quad e^{\tau H_1^{(x)}} e^{\tau H_2^{(x)}} e^{\tau H_1^{(y)}} e^{\tau H_2^{(y)}}, \tag{2.79}$$

where for instance, in formal analogy to Eq. (2.58b), we have

$$H_2^{(x)} = \sum_{i=1}^{n_x-2}{}' \sum_{j=1}^{n_y-2}{}' \frac{\mathbf{e}_{i+1,j+1}\mathbf{e}_{i+2,j+1}^T - \mathbf{e}_{i+2,j+1}\mathbf{e}_{i+1,j+1}^T}{\delta \sqrt{\varepsilon_{i+1,j+1}\mu_{i+2,j+1}}}. \tag{2.80}$$

It is not difficult to convince oneself that approximation $U_1(\tau)$ and hence also $U_2(\tau)$ and $U_4(\tau)$ do not commute with the (lattice version of) the divergence. Therefore the divergence of the EM fields in 2D is not conserved. Although the initial state (at $t = 0$) of the EM fields satifies Eqs. (2.1c) and (2.1d), time-integration of the TDME using $U_k(\tau)$ yields a solution that will not satisfy Eqs. (2.1c) and (2.1d). However, for algorithm $U_k(\tau)$ the deviations from zero vanish as τ^k so that in practice these errors are under control and can be made sufficiently small if necessary (this is studied in detail in section 3.2.4).

2.3.3 Implementation 3D

In terms of $\Psi(t) = (\mathbf{X}(t), \mathbf{Y}(t))^T$ for a 3D system, Eq. (2.9) reads

$$\frac{\partial}{\partial t}\Psi(t) = \mathcal{H}\Psi(t) = \begin{pmatrix} 0 & h \\ -h^T & 0 \end{pmatrix}\Psi(t), \tag{2.81}$$

where h is given by

$$h = \begin{pmatrix} 0 & \frac{1}{\sqrt{\mu}}\frac{\partial}{\partial z}\frac{1}{\sqrt{\varepsilon}} & -\frac{1}{\sqrt{\mu}}\frac{\partial}{\partial y}\frac{1}{\sqrt{\varepsilon}} \\ -\frac{1}{\sqrt{\mu}}\frac{\partial}{\partial z}\frac{1}{\sqrt{\varepsilon}} & 0 & \frac{1}{\sqrt{\mu}}\frac{\partial}{\partial x}\frac{1}{\sqrt{\varepsilon}} \\ \frac{1}{\sqrt{\mu}}\frac{\partial}{\partial y}\frac{1}{\sqrt{\varepsilon}} & -\frac{1}{\sqrt{\mu}}\frac{\partial}{\partial x}\frac{1}{\sqrt{\varepsilon}} & 0 \end{pmatrix}. \tag{2.82}$$

We discretize the spatial coordinates by adopting the standard Yee grid [5], of which the unit cell is shown in figure 1-1.

In analogy to the systems in 1D and 2D, we assign the EM fields to the lattice points such that the boundary conditions Eq. (2.2) are automatically satisfied. The elements of the vector $\Psi(i, j, k, t)$ are given by

$$\Psi(i,j,k,t) = \begin{cases} X_x(i,j,k,t) = \sqrt{\mu_{i,j,k}}H_x(i,j,k,t), & i \text{ even, } j \text{ odd, } k \text{ odd} \\ X_y(i,j,k,t) = \sqrt{\mu_{i,j,k}}H_y(i,j,k,t), & i \text{ odd, } j \text{ even, } k \text{ odd} \\ X_z(i,j,k,t) = \sqrt{\mu_{i,j,k}}H_z(i,j,k,t), & i \text{ odd, } j \text{ odd, } k \text{ even} \\ Y_x(i,j,k,t) = \sqrt{\varepsilon_{i,j,k}}E_x(i,j,k,t), & i \text{ odd, } j \text{ even, } k \text{ even} \\ Y_y(i,j,k,t) = \sqrt{\varepsilon_{i,j,k}}E_y(i,j,k,t), & i \text{ even, } j \text{ odd, } k \text{ even} \\ Y_z(i,j,k,t) = \sqrt{\varepsilon_{i,j,k}}E_z(i,j,k,t), & i \text{ even, } j \text{ even, } k \text{ odd} \end{cases}, \tag{2.83}$$

for the origin of the coordinate system $(i, j, k) = (0, 0, 0)$ at the center of the unit cell shown in figure 1-1. The number of lattice points in the x, y, and z direction will be denoted by n_x, n_y, and n_z respectively. As before, these numbers are assumed to be odd.

Discretization of the differential operators that appear in Eq. (2.82) yields Eq. (2.9) in the form

$$\frac{\partial}{\partial t}\Psi(t) = H\Psi(t) = \sum_{i=1}^{n_x-2}{}' \sum_{j=1}^{n_y-2}{}' \sum_{k=1}^{n_z-2}{}' \left[H^{(x)}(i,j,k) + H^{(y)}(i,j,k) + H^{(z)}(i,j,k)\right]\Psi(t), \qquad (2.84)$$

where the superscripts (x), (y) and (z) refer to the derivative with respect to x, y and z respectively. For example, we have

$$\begin{aligned} H^{(x)}(i,j,k) = & \\ & + \frac{\mathbf{e}_{i,j+1,k}\mathbf{e}_{i+1,j+1,k}^T - \mathbf{e}_{i+1,j+1,k}\mathbf{e}_{i,j+1,k}^T}{\delta\sqrt{\varepsilon_{i+1,j+1,k}\mu_{i,j+1,k}}} - \frac{\mathbf{e}_{i,j,k+1}\mathbf{e}_{i+1,j,k+1}^T - \mathbf{e}_{i+1,j,k+1}\mathbf{e}_{i,j,k+1}^T}{\delta\sqrt{\varepsilon_{i+1,j,k+1}\mu_{i,j,k+1}}} \\ & + \frac{\mathbf{e}_{i+1,j+1,k}\mathbf{e}_{i+2,j+1,k}^T - \mathbf{e}_{i+2,j+1,k}\mathbf{e}_{i+1,j+1,k}^T}{\delta\sqrt{\varepsilon_{i+1,j+1,k}\mu_{i+2,j+1,k}}} - \frac{\mathbf{e}_{i+1,j,k+1}\mathbf{e}_{i+2,j,k+1}^T - \mathbf{e}_{i+2,j,k+1}\mathbf{e}_{i+1,j,k+1}^T}{\delta\sqrt{\varepsilon_{i+1,j,k+1}\mu_{i+2,j,k+1}}}, \end{aligned} \qquad (2.85)$$

and the expressions for $H^{(y)}(i,j,k)$ and $H^{(z)}(i,j,k)$ follow from Eq. (2.85) by symmetry. Note that we use the triple (i,j,k) to label the $n_x n_y n_z$ unit vectors $\mathbf{e}_{i,j,k}$.

For the Lie-Trotter-Suzuki algorithms, we need to decompose H, and in complete analogy with the 1D and 2D case we split Eq. (2.85) (as well as $H^{(y)}(i,j,k)$ and $H^{(z)}(i,j,k)$) in two parts and obtain for the first-order approximation to $U(\tau)$

$$U_1(\tau) \quad = \quad e^{\tau H_1^{(x)}} e^{\tau H_2^{(x)}} e^{\tau H_1^{(y)}} e^{\tau H_2^{(y)}} e^{\tau H_1^{(z)}} e^{\tau H_2^{(z)}}, \qquad (2.86)$$

where for instance

$$H_1^{(z)} = \sum_{i=1}^{n_x-2}{}' \sum_{j=1}^{n_y-2}{}' \sum_{k=1}^{n_z-2}{}' \left(\frac{\mathbf{e}_{i,j+1,k}\mathbf{e}_{i,j+1,k+1}^T - \mathbf{e}_{i,j+1,k+1}\mathbf{e}_{i,j+1,k}^T}{\delta\sqrt{\varepsilon_{i,j+1,k+1}\mu_{i,j+1,k}}} - \frac{\mathbf{e}_{i+1,j,k}\mathbf{e}_{i+1,j,k+1}^T - \mathbf{e}_{i+1,j,k+1}\mathbf{e}_{i+1,j,k}^T}{\delta\sqrt{\varepsilon_{i+1,j,k+1}\mu_{i+1,j,k}}} \right). \qquad (2.87)$$

Note that each contribution to Eq. (2.87) acts on a different pair of elements of $\Psi(t)$. Hence, each of the matrix exponentials in Eq. (2.86) acts on one quarter of all the lattice points. Performing the timestep operation Eq. (2.86) involves only one and a half sweeps over all lattice points.

By construction, the algorithm defined by Eq. (2.86) is unconditionally stable and so are the higher-order algorithms defined by $U_2(\tau)$ and $U_4(\tau)$. Each contribution to e.g. Eq. (2.87) is of the form Eq. (2.58a) and hence its matrix exponential can be calculated in exactly the same manner as in the 1D case (see Eq. (2.62)). The divergence of the EM fields in 3D is, for the same reason as in the 2D case, not conserved but decreases as τ^k, where k is the order of accuracy of time integration.

2.3.4 A fourth-order spatial approximation

In some cases, the use of a finer mesh to obtain more accurate results may not be as efficient as the use of a higher order spatial derivative. Especially if the algorithm is used for the calculation of spectra, a higher order spatial derivative would be preferred, since a finer mesh leads to the existence of higher frequencies and thus a lower spectral resolution if the same number of samples is taken for the computation of the Fourier spectrum.

Higher order approximations for the spatial derivative are well known [44]. However, some considerations should be taken into account. First of all, the construction of our algorithms demands that skew-symmetric properties are conserved, and secondly, due to the alternating grid setup, one must be careful which points to include at which mesh distances while deriving a higher-order approximation.

In our grid setup (in all dimensions), the distance between two neighboring lattice points is $\delta/2$, thus if we write $f_i = f(i\delta/2, t)$, then f_i denotes either a H or a E field component, and $f_{i\pm1}$ a E respectively H component, due to the alternating grid. Using this notational convention, we can write for the central-difference approximation to the spatial derivative (as used in equations (2.52a) and (2.52b))

$$\frac{\partial}{\partial x} f_i = \frac{f_{i+1} - f_{i-1}}{\delta} + \frac{\delta^2}{6} f_i^{(3)} + O(\delta^4). \tag{2.88}$$

At the same time, if we use the next-nearest neighbor field points (at a distance $3\delta/2$, since at a distance δ the wrong type of field component is defined) to compute the first order spatial derivative, using the same central-difference approximation, we have

$$\frac{\partial}{\partial x} f_i = \frac{f_{i+3} - f_{i-3}}{3\delta} + \frac{9\delta^2}{6} f_i^{(3)} + O(\delta^4). \tag{2.89}$$

Now if we multiply Eq. (2.88) with $9/8$ and subtract Eq. (2.89) multiplied with $1/8$, the error in the third order derivative is cancelled, and we get a fourth-order approximation for the spatial derivative:

$$\frac{\partial}{\partial x} f_i = \frac{9}{8}\left(\frac{f_{i+1} - f_{i-1}}{\delta}\right) - \frac{1}{8}\left(\frac{f_{i+3} - f_{i-3}}{3\delta}\right) + O(\delta^4). \tag{2.90}$$

The actual implementation of an algorithm based on this fourth order approximation is quite simple, since we can reuse the known second-order algorithms, applied with different factors at different meshes. It should be noted though, that for the fourth-order implementation, non-periodic boundary conditions are not as strictly satisfied as for the second-order implementation. For very few elements near the boundary, field points lying outside the cavity (which are implicitly zero), contribute to the computation of the derivative. In fact, at the boundary, the error for this algorithm becomes second order in space, precisely due to this effect.

2.3.5 A variable mesh

A grid with a fixed spatial mesh size may not be the optimal way to discretize the space, because of some physical problems. In many cases, a particular region of space is of more interest than the circumventing vacuum. Consider a piece of material with a high dielectric constant placed in an otherwise empty large cavity. The frequencies occurring inside the material will be much higher than outside the material. A good model for this system would require a fine meshed grid inside the material to account correctly for the high frequencies, and a more coarse grid outside the material. Hence, a variable mesh is required. The use of a variable mesh can save resources while still modeling the problem correctly.

Some caution must be taken into account when constructing unconditionally stable algorithms based on a variable mesh size. The reason is that if one straightforwardly uses

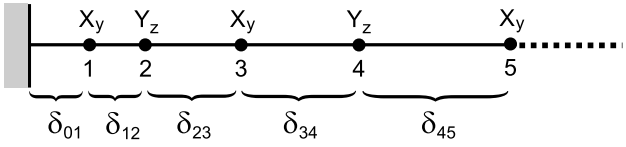

Figure 2-4: One-dimensional system with variable mesh.

varying spatial differences between neighboring field points to compute the spatial derivatives, the necessary skew-symmetric properties of the time evolution matrix would be lost, and the resulting algorithm would be rendered unstable. A small error must be introduced to conserve the anti-symmetry of the matrix.

Consider a one dimensional system with variable mesh spacing, see figure 2-4. In a straightforward, non skew-symmetric, implementation of the variable grid, we would replace the constant next-nearest neighbor distance δ in Eqs. (2.52a) and (2.52b) of the fixed-mesh implementation by the corresponding variable distance. It is convenient to write this substitution in the form

$$\delta \;\rightarrow\; \Delta_{i,i+1}\left[1 \;+\; \frac{\delta_{i-1,i} - \delta_{i+1,i+2}}{2\Delta_{i,i+1}}\right], \tag{2.91}$$

where $\delta_{i,j}$ is the distance between grid points i and j (see figure 2-4) and

$$\Delta_{i,i+1} \;\equiv\; \frac{1}{2}(\delta_{i-1,i} \;+\; 2\delta_{i,i+1} \;+\; \delta_{i+1,i+2}) \tag{2.92}$$

is the averaged next-nearest neighbor distance. It can be easily checked that an implementation of the variable grid that relies on the replacement Eq. (2.91) would destroy the skew-symmetry property of the corresponding matrix H. This is unphysical: the original form of the Maxwell equations do have this property. However, a variable grid implementation that does preserve the underlying symmetry of Maxwell's Equations can be constructed for a sufficiently smooth, variable grid. In this case, the second term in the brackets of Eq. (2.91) may be neglected and the replacement

$$\delta \;\rightarrow\; \Delta_{i,i+1} \;=\; \Delta_{i+1,i} \tag{2.93}$$

may yield a reasonable approximation of Eqs. (2.52a) and (2.52b) for the variable grid implementation:

$$\frac{\partial}{\partial t}X_y(i,t) = \frac{1}{\sqrt{\mu_i}}\left(\frac{Y_z(i+1,t)}{\Delta_{i,i+1}\sqrt{\varepsilon_{i+1}}} - \frac{Y_z(i-1,t)}{\Delta_{i,i-1}\sqrt{\varepsilon_{i-1}}}\right), \tag{2.94a}$$

$$\frac{\partial}{\partial t}Y_z(i+1,t) = \frac{1}{\sqrt{\varepsilon_{i+1}}}\left(\frac{X_y(i+2,t)}{\Delta_{i+1,i+2}\sqrt{\mu_{i+2}}} - \frac{X_y(i,t)}{\Delta_{i+1,i}\sqrt{\mu_i}}\right). \tag{2.94b}$$

The corresponding matrix H is seen to be skew-symmetric,

$$H = \sum_{i=1}^{n}{}' \left[\frac{\mathbf{e}_i\,\mathbf{e}_{i+1}^T - \mathbf{e}_{i+1}\mathbf{e}_i^T}{\Delta_{i,i+1}\sqrt{\varepsilon_{i+1}\mu_i}} + \frac{\mathbf{e}_{i+1}\mathbf{e}_{i+2}^T - \mathbf{e}_{i+2}\mathbf{e}_{i+1}^T}{\Delta_{i+1,i+2}\sqrt{\varepsilon_{i+1}\mu_{i+2}}}\right], \tag{2.95}$$

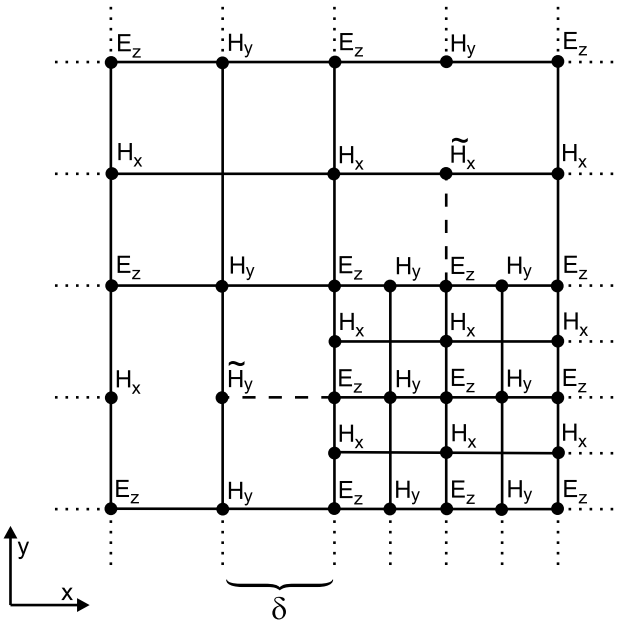

Figure 2-5: Two-dimensional system with variable mesh and phantom grid-points.

and, for the Lie-Trotter-Suzuki algorithms, may again be separated into an odd and an even part of which the exponents can be easily calculated following the same steps as given above in the fixed-mesh implementation.

Unfortunately, the extension of this approach to construct algorithms using a variable mesh in higher dimensions is not as straightforward as it seems. Ideally, one would like to have a grid with a small mesh size only at the locations where that is required, for example near a sharp edge in a cavity system. However, the continuity of the fields must be conserved at the transition (interface) of the large mesh size to the small mesh size. One way to accomplish this is to solve the Maxwell Equations in their integral formulation locally at the interface, and this can be done for FDTD type algorithms [45]. However, for our class of algorithms it is not trivial to increase the grid point density locally. The use of another method to update a subset of all gridpoints almost certainly leads to loss of the orthogonality of the entire algorithm.

We demonstrate this for an algorithm solving the two dimensional TM-mode Maxwell Equations for an interface where the grid point density is doubled. In order to maintain the specific alternating (staggered) grid setup, some extra grid points must be inserted that have only one neighboring point. A typical grid generated according to this procedure is shown in figure 2-5. The spatial derivatives at each of these points can only be computed by using a 'phantom' gridpoint of which the field value is obtained by averaging nearby field values. The implementation of this construction requires a lot of bookkeeping, but more importantly, leads to unstable algorithms. The instability can cause the total energy density to increase exponentially, as can be seen in figure 2-6.

One solution to this problem is to increase the grid density uniformly for all dimensions

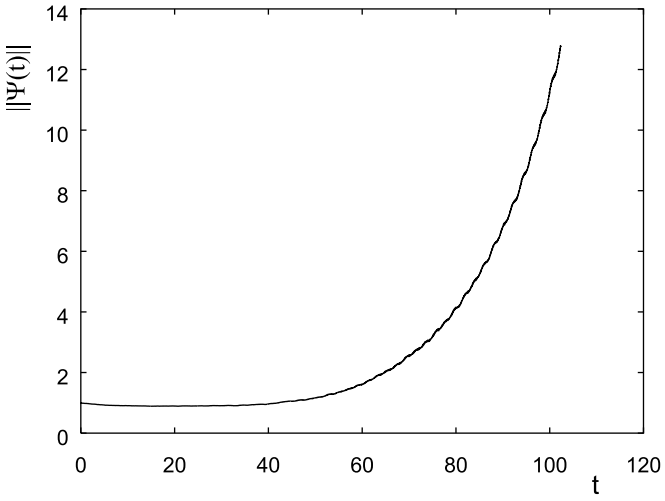

Figure 2-6: Energy density as a function of time for the unstable phantom points implementation, for a typical 2D system.

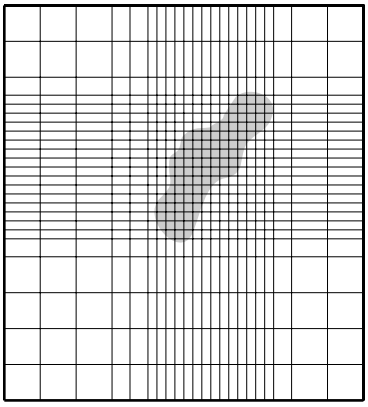

Figure 2-7: A variable mesh implementation in two dimensions. Near the material the grid density is increased gradually with a factor 4 in each dimension separately, not locally.

separately. Thus $\Delta x = \Delta x(x)$, $\Delta y = \Delta y(y)$ and $\Delta z = \Delta z(z)$, instead of for example $\Delta x = \Delta x(x, y, z)$. In two dimensions, this typically generates grid spacings as shown in figure 2-7.

2.4 Sources, losses and absorbing boundary conditions

In the previous sections, we explained the details of a set of new numerical algorithms to solve the TDME. However, a realistic time-dependent simulation of any physical system requires more than just an algorithm to perform the time evolution.

The space dependency EM fields must be specified either at the initial time instance or continuously, by means of a source. For example, if one wishes to measure the density of states, a random initial condition must be used, as explained in more detail in appendix B. But for the simulation of scattering phenomena, one typically requires a localized EM field moving in a specific direction with a particular energy. The construction of such wavepackets is discussed in appendix A.

To account for a current source in a physically correct way[5], demands a slight change in the algorithms themselves. In the next few sections, we show how to incorporate a source for the Lie-Trotter-Suzuki and the Chebyshev algorithms.

Scattering phenomena typically involve a particular region of interest, for example an interface. In this case, it is preferred to use open boundaries, in other words, minimal reflections should occur at the boundary. Up till now, we only assumed perfectly reflecting boundaries, so in the final section we explain a simple way to incorporate absorbing boundary conditions.

2.4.1 Adding a source and dissipative materials for TnSm algorithms

If there exists a current source and/or lossy materials in a physical system, then instead of Eq. (2.11), we must return to the general Maxwell Equations

$$\frac{\partial}{\partial t}\Psi(t) = \begin{pmatrix} 0 & -\frac{1}{\sqrt{\mu}}\nabla\times\frac{1}{\sqrt{\varepsilon}} \\ \frac{1}{\sqrt{\varepsilon}}\nabla\times\frac{1}{\sqrt{\mu}} & -\sigma/\sqrt{\varepsilon} \end{pmatrix}\Psi(t) - \begin{pmatrix} 0 \\ \mathbf{J}(t)/\sqrt{\varepsilon} \end{pmatrix}, \tag{2.96}$$

where the current source may depend on space and time, $\mathbf{J}(t) = \mathbf{J}(\mathbf{r}, t)$, and the conductivity $\sigma = \sigma(\mathbf{r})$ is assumed to be a scalar quantity, independent of time. The simplest way to solve this equation, is to re-use the (known) solution to the equation $(\partial/\partial t)\Psi(t) = \mathcal{H}\Psi(t)$. Writing equation (2.96) as

$$\frac{\partial}{\partial t}\Psi(t) = \mathcal{H}'\Psi(t) - \mathcal{J}(t), \tag{2.97}$$

where $\mathcal{J}(t) = (0, \mathbf{J}(t)/\sqrt{\varepsilon})^T$ represents the source term, and

$$\mathcal{H}' = \mathcal{H} + \begin{pmatrix} 0 & 0 \\ 0 & -\sigma/\sqrt{\varepsilon} \end{pmatrix} = \mathcal{H} + \mathcal{S}, \tag{2.98}$$

the formal solution of Eq. (2.97) is given by

$$\Psi(t) = e^{t\mathcal{H}'}\Psi(0) - \int_0^t e^{(t-u)\mathcal{H}'}\mathcal{J}(u)du. \tag{2.99}$$

[5] for the Yee algorithm, the current source is usually implemented by means of a 'hard' source [6], in which case the values of the source simply overwrite the EM fields at the spatial location of the source. Although this might work in practice in some cases, it is quite unphysical.

The matrix exponent $e^{t\mathcal{H}'}$ can be solved by decomposition, for example

$$e^{t\mathcal{H}'}\Psi(0) = e^{t\mathcal{H}+t\mathcal{S}}\Psi(0) \approx e^{t\mathcal{S}/2}e^{t\mathcal{H}}e^{t\mathcal{S}/2}\Psi(0). \tag{2.100}$$

Obviously, matrix exponent $e^{t\mathcal{H}}$ can be computed by one of the algorithms discussed in the previous sections, and matrix $\mathcal{S}$ is diagonal, so

$$e^{t\mathcal{S}}\Psi(t) = \begin{pmatrix} X(t) \\ \mathrm{diag}(\exp(-\sigma t/\sqrt{\varepsilon}))Y(t) \end{pmatrix}. \tag{2.101}$$

In practice, for a timestep τ, we update $\Psi(t)$ according to

$$\Psi(t+\tau) = e^{\tau\mathcal{H}'}\Psi(t) - \int_t^{t+\tau} e^{(t+\tau-u)\mathcal{H}'}\mathcal{J}(u)\,du. \tag{2.102}$$

A standard quadrature formula can be used to compute the integral over u, for example the fourth-order accurate Simpson rule [27]

$$\int_t^{t+\tau} e^{(t+\tau-u)\mathcal{H}'}\mathcal{J}(u)\,du \approx \frac{\tau}{6}\left(e^{\tau H'}\mathcal{J}(t) + 4e^{\tau H'/2}\mathcal{J}(t+\tau/2) + \mathcal{J}(t+\tau)\right). \tag{2.103}$$

2.4.2 Adding a source for the Chebyshev algorithm

Although the Chebyshev one-step algorithm can be used to calculate terms like e^{tH}, with H being a skew-symmetric matrix, the above scheme to handle dissipative materials and current sources does not readily extend for use with the algorithm. More specifically, the decomposition Eq. (2.100) and the computation of the integral in Eq. (2.103) converge to the exact values only in the limit of small time intervals τ. However, one wishes to use the Chebyshev algorithm for arbitrarily large times t, for which it becomes more efficient in comparison with the other algorithms.

As it stands, there are two separate problems that must be overcome in order to use a Chebyshev expansion: the presence of lossy materials and that of a current source. The former problem is the hardest one to solve, because the matrix decomposition of $\mathcal{H}'$ is not going to help. On the other hand, the direct computation of $\exp(t\mathcal{H}')$ by means of a Chebyshev expansion is hard to justify mathematically, since, although matrix $\mathcal{H}'$ still has a complete set of eigenvalues, some of these may have a nonzero real part. In that case, the convergence of the series expansion cannot be guaranteed rigorously.

However, the incorporation of a current source can be achieved quite elegantly. It will turn out that there is no need to sample the time-dependence of the current source. First, we assume that the source has a sinusoidal time dependence (later, this assumption may be dropped)

$$\mathbf{J}(\mathbf{r},t) = \Theta(T-t)\mathbf{s}(\mathbf{r})\sin(\omega_s t), \tag{2.104}$$

where $\mathbf{s}(\mathbf{r})$ specifies the spatial distribution of the source and ω_s is the source angular frequency. As indicated by the presence of the step function $\Theta(T-t)$ in Eq.(2.104), the source is turned on at $t = 0$ and is switched off at $t = T$. It is most convenient to suppress artifacts that result from the discontinuity at $t = T$ by choosing T such that $\omega_s T$ is an integer number.

The contribution of the source term to the EM field at time t is given by the last term of Eq.(2.99). For the sinusoidal source described by Eq.(2.104), the integral evaluates to

$$\begin{aligned}
&\int_0^t e^{(t-u)H}\mathcal{J}(u)\,du \\
&\quad = (\omega_s^2 + H^2)^{-1}e^{(t-T')H}(\omega_s e^{T'H} - \omega_s\cos\omega_s T' - H\sin\omega_s T')\;\boldsymbol{\Xi} \\
&\quad \equiv f(H,t,T',\omega_s)\;\boldsymbol{\Xi},
\end{aligned} \tag{2.105}$$

where $\boldsymbol{\Xi}$ denotes the vector representing the spatial distribution $\mathbf{s}(\mathbf{r})$ of the source and $T' = \min(t,T)$. The expansion coefficients of the Chebyshev polynomial approximation of the time evolution operator in Eq.(2.105) may be calculated as follows. First we repeat the scaling procedure described in section 2.2.2, then we substitute in Eq.(2.105) $H = ix\|H\|_1$, $t = z/\|H\|_1$, $T' = Z'/\|H\|_1$ and $\omega_s = \omega\|H\|_1$ and finally we compute the Fourier expansion coefficients (i.e. the coefficients of the Chebyshev polynomial expansion) of the function

$$\begin{aligned}
f(x,z,Z',\omega) = \frac{Z'}{2\|H\|_1}e^{ix(z-Z')}\Bigg\{&\omega Z'\frac{\sin(\omega+x)Z'/2}{(\omega+x)Z'/2}\frac{\sin(\omega-x)Z'/2}{(\omega-x)Z'/2} \\
&+i\left[\frac{\sin(\omega+x)Z'/2}{(\omega+x)Z'/2}\cos(\omega-x)Z'/2 - \frac{\sin(\omega-x)Z'/2}{(\omega-x)Z'/2}\cos(\omega+x)Z'/2\right]\Bigg\},
\end{aligned} \tag{2.106}$$

where $Z' = \min(z,Z)$. We have written Eq.(2.106) in a form that emphasizes that $f(x,z,Z',\omega)$ has no singularities as a function of $-1 \le x \le 1$. Note that sources with a more complicated time dependence $G(t)$ can be synthesized by computing $\int_{-\infty}^{\infty} g(\omega)f(x,z,Z',\omega)d\omega$, where $g(\omega)$ is the Fourier sine transform of $G(t)$. Therefore, in general, the one-step algorithm to compute the EM field at time t is given by

$$\begin{aligned}
\Psi(t) \approx &\left[J_0(t\|H\|_1)I + 2\sum_{k=1}^{K} J_k(t\|H\|_1)\widetilde{T}_k(B)\right]\Psi(0) \\
&+\left[S_0(t\|H\|_1)I + 2\sum_{k=1}^{K'} S_k(t\|H\|_1)\widetilde{T}_k(B)\right]\boldsymbol{\Xi},
\end{aligned} \tag{2.107}$$

where $S_k(t\|H\|_1)$ denotes the k-th expansion coefficient of the (time-dependent) current source. Similar to the source-less Chebyshev expansion terms, the polynomial expansion of the current source also has to be truncated at some K'. To determine K', a similar procedure as the one used for K can be followed. Furthermore, the numerical procedure to compute the contribution of both terms in Eq.(2.107) is the same. The expansion coefficients of $f(x,z,Z',\omega)$ are most easily obtained by performing a Fast Fourier Transform (FFT) with respect to x [46]. In particular, for the sinusoidal source described by Eq.(2.104) we have

$$S_k(t\|H\|_1) = i^{-k}\sum_{n=0}^{N-1} e^{2\pi ink/N} f(\cos\frac{2\pi n}{N}, z, Z', \omega), \tag{2.108}$$

where N is the number of points in the FFT. For the Chebyshev approach to work well, it is necessary that the coefficients are calculated to high precision. In other words, N has to be large enough; in practice, we use $N \ge 2^{20}$ to perform the FFT in (2.108).

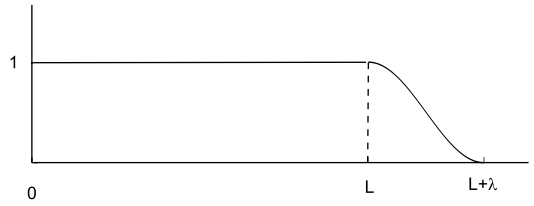

Figure 2-8: Function to implement absorbing boundary conditions at $x = L$ for a one dimension system. All EM fields are multiplied with this function at time intervals Δt.

2.4.3 Absorbing boundary conditions

Scattering processes typically involve systems with open boundaries and instead of the cavities that we considered up till now. Absorbing boundary conditions (ABC) can account for open boundaries, preferably with minimal reflection. For the Yee algorithm, Berenger developed a quite successful method to implement ABC with a low degree of reflection, the so-called perfectly matched layer (PML) method [47–49]. Unfortunately, it does not preserve the symmetries of the Maxwell Equations, due to the introduction of extra, unphysical field points. It is therefore unclear how to extend the PML method to the matrix exponential approach, crucial for the Lie-Trotter-Suzuki and Chebyshev algorithms. Alternatively, a crude implementation of ABC can be accomplished by extending the system with a buffer region, wherein all EM fields are multiplied by a smooth function at regular time-intervals. A suitable function for this purpose is for example, in one dimension and for the end region (see figure 2-8)

$$f(x) = 1 + \Theta(x - L)(\cos^2(\pi(x - L)/2\lambda) - 1), \tag{2.109}$$

where $\Theta(x)$ is the step function.

The degree of reflection for this implementation obviously depends on the length of the buffer region λ and the time interval Δt at which the function is applied, but it also not entirely frequency independent. We define the reflection as the ratio of the norms of the energy density of the reflected as compared with the initial wave. In figure 2-9 we show the reflection as function of Δt for a buffer regions placed at the end with length $\lambda = 1, 2, 3$. A time interval of $\Delta t = 0.3$ seems to give the best results, while a buffer region of $\lambda = 2$ suffices, giving a reflection of approximately $1.6 \cdot 10^{-4}$. The frequency dependency of the reflection is very weak; only very high frequency components ($\omega \geq 10$) are somewhat more reflected (results not shown).

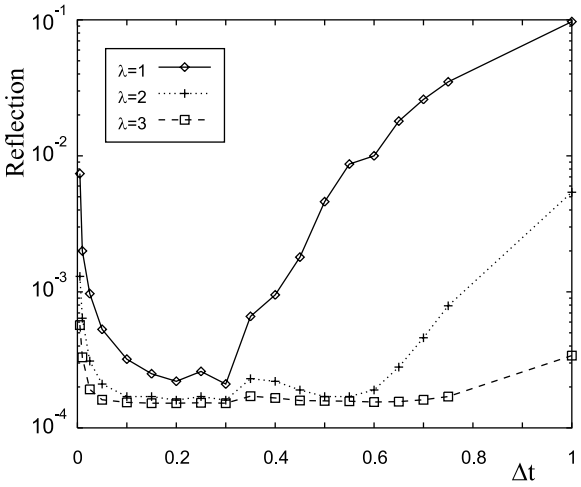

Figure 2-9: Reflection as a function of time interval Δt, for buffer lengths $\lambda = 1, 2, 3$ in a one dimensional system with absorbing boundary conditions at $x = L$. The system measured $L = 30$, with $\delta = 0.1$ and was simulated with the T4S2 algorithm. In this case, an oscillating Gaussian wavepacket (generated according to the procedure explained in appendix A) with $k = 6$, $\sigma = 4$ and $x_0 = 8$ is used as initial condition, and the reflection is measured at $t = 38.4$.