Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Physics Communications 156 (2003) 43–61
Computer Physics Communications
www.elsevier.com/locate/cpc

Unified framework for numerical methods to solve the time-dependent Maxwell equations

H. De Raedt *, J.S. Kole, K.F.L. Michielsen, M.T. Figge

Applied Physics–Computational Physics, [1] *Materials Science Centre, University of Groningen, Nijenborgh 4, NL-9747 AG Groningen, The Netherlands*

Received 4 December 2002; accepted 22 July 2003

Abstract

We present a comparative study of numerical algorithms to solve the time-dependent Maxwell equations for systems with spatially varying permittivity and permeability. We show that the Lie–Trotter–Suzuki product-formula approach can be used to construct a family of unconditionally stable algorithms, the conventional Yee algorithm, and two new variants of the Yee algorithm that do not require the use of the staggered-in-time grid. We also consider a one-step algorithm, based on the Chebyshev polynomial expansion, and compare the computational efficiency of the one-step, the Yee-type, the alternating-direction-implicit, and the unconditionally stable algorithms. For applications where the long-time behavior is of main interest, we find that the one-step algorithm may be orders of magnitude more efficient than present multiple time-step, finite-difference time-domain algorithms.

PACS: 02.60.Cb; 03.50.De; 41.20.Jb

1. Introduction

The Maxwell equations describe the evolution of electromagnetic (EM) fields in space and time [1]. They apply to a wide range of different physical situations and play an important role in a large number of engineering applications. In many cases, numerical methods are required to solve Maxwell's equations [2,3]. A well-known class of algorithms is based on a method proposed by Yee [4]. This finite-difference time-domain (FDTD) approach owes its popularity mainly due to its flexibility and speed while at the same time it is easy to implement [2,3].

A limitation of Yee-based FDTD techniques is that their stability is conditional, depending on the mesh size of the spatial discretization and the time step of the time integration [2,3]. Furthermore, in practice, the amount of computational work required to solve the time-dependent Maxwell equations by present FDTD techniques [2,3,5–12] prohibits applications to a class of important fields such as bio-electromagnetics and VLSI design [2,13,14].

* Corresponding author.
E-mail addresses: deraedt@phys.rug.nl (H. De Raedt), j.s.kole@phys.rug.nl (J.S. Kole), kristel@phys.rug.nl (K.F.L. Michielsen), m.t.figge@phys.rug.nl (M.T. Figge).
[1] http://www.compphys.org.

0010-4655/$ – see front matter
doi:10.1016/S0010-4655(03)00414-4

The basic reason for this is that the time step in the FDTD calculation has to be relatively small in order to maintain stability and a reasonable degree of accuracy in the time integration. Thus, the search for new algorithms that solve the Maxwell equations focuses on removing the conditional stability of FDTD methods and on improving the accuracy/efficiency of the algorithms.

A systematic approach to construct unconditionally stable algorithms is to employ a Lie–Trotter–Suzuki product-formula [15] to approximate the time evolution operator [16]. In the case of EM fields, the latter is the matrix exponential of a skew-symmetric matrix and the approximations take the form of products of orthogonal transformations [11,12]. The resulting numerical algorithms are unconditionally stable by construction [16,17].

The spectral-domain split-operator technique proposed in Ref. [10] is one of the many forms that results from the use of the Lie–Trotter–Suzuki product-formulas [15]. This technique makes use of Fast Fourier Transforms to compute the matrix exponentials of the displacement operators. The choice made in Ref. [10] yields an approximation to the time-evolution operator that is no longer orthogonal and hence unconditional stability is not automatically guaranteed [18]. In contrast, the methodology proposed in Refs. [11,12] yields efficient, explicit, unconditionally stable schemes that operate on the EM fields defined on the real space grid only. These algorithms naturally allow for the spatial variations in both the permittivity and the permeability.

The product-formula approach also provides a unified framework to construct and analyze other time stepping algorithms [16,19]. To illustrate this point we show that the conventional Yee algorithm and the alternating-direction-implicit (ADI) time-stepping algorithms [6–9,19] fit into this framework. Furthermore we propose new variants of the Yee algorithm.

Another route to improve upon the accuracy/efficiency of time-integration schemes is to make use of the Chebyshev polynomial expansions of the matrix exponential [20–25]. In this paper we review the one-step algorithm [26,27], based on Chebyshev polynomials, to solve the time-dependent Maxwell equations for (very) large time steps. As the solution obtained by the one-step algorithm is exact to nearly machine precision (for a fixed spatial discretization), we use this algorithm to generate the reference data.

The main purpose of this paper is to review the basic ideas behind the recent developments in numerical algorithms to solve the time-dependent Maxwell equations and to compare the virtues and shortcomings of the different methods. The plan of the paper is as follows: In Section 2 we briefly discuss the basic physical symmetries of the time-dependent Maxwell equations. The general framework to construct time-integration algorithms is laid out in Section 3. We also pay attention to the numerical treatment of the current source term. In Section 4 we use the simplest case of the time-dependent Maxwell equations to illustrate how the various algorithms can be implemented. We explicitly show how the conventional Yee algorithm naturally fits into this framework and, by minor modification, construct second-order and fourth-order time-accurate schemes that do not require the use of staggered-in-time fields, nor extra memory to store intermediate results. Then we recall the steps to construct the unconditionally stable algorithms proposed in Refs. [11,12] and analyze a modification to improve the time-integration accuracy. Finally we discuss the implementation of the ADI and one-step algorithms. A discussion of the results of numerical experiments and our conclusions are given in Sections 5 and 6, respectively.

2. Theory

This section recalls some well-known facts of Maxwell's theory and also serves to introduce the notation used in this paper. We consider EM fields in linear, isotropic, nondispersive and lossless materials. The time evolution of EM fields in these systems is governed by the time-dependent Maxwell equations [1]. Some important physical symmetries of the Maxwell equations can be made explicit by introducing the fields

$$\mathbf{X}(t) \equiv \sqrt{\mu}\,\mathbf{H}(t) \quad \text{and} \quad \mathbf{Y}(t) \equiv \sqrt{\varepsilon}\,\mathbf{E}(t). \tag{1}$$

Here, $\mathbf{H}(t) = (H_x(\mathbf{r},t), H_y(\mathbf{r},t), H_z(\mathbf{r},t))^{\mathrm{T}}$ denotes the magnetic and $\mathbf{E}(t) = (E_x(\mathbf{r},t), E_y(\mathbf{r},t), E_z(\mathbf{r},t))^{\mathrm{T}}$ the electric field vector, while $\mu = \mu(\mathbf{r})$ and $\varepsilon = \varepsilon(\mathbf{r})$ denote, respectively, the permeability and the permittivity. In the

absence of electric charges, Maxwell's curl equations [2] read

$$\frac{\partial}{\partial t}\begin{pmatrix}\mathbf{X}(t)\\ \mathbf{Y}(t)\end{pmatrix}=\mathcal{H}\begin{pmatrix}\mathbf{X}(t)\\ \mathbf{Y}(t)\end{pmatrix}-\frac{1}{\sqrt{\varepsilon}}\begin{pmatrix}0\\ \mathbf{J}(t)\end{pmatrix}, \tag{2}$$

where $\mathbf{J}(t)=(J_x(\mathbf{r},t),J_y(\mathbf{r},t),J_z(\mathbf{r},t))^{\mathrm{T}}$ represents the source of the electric field and $\mathcal{H}$ denotes the operator

$$\mathcal{H}\equiv\begin{pmatrix}0 & -\frac{1}{\sqrt{\mu}}\nabla\times\frac{1}{\sqrt{\varepsilon}}\\ \frac{1}{\sqrt{\varepsilon}}\nabla\times\frac{1}{\sqrt{\mu}} & 0\end{pmatrix}. \tag{3}$$

Writing $\mathbf{Z}(t)=(\mathbf{X}(t),\mathbf{Y}(t))^{\mathrm{T}}$ it is easy to show that $\mathcal{H}$ is skew symmetric, i.e. $\mathcal{H}^{\mathrm{T}}=-\mathcal{H}$, with respect to the inner product $\langle\mathbf{Z}(t)|\mathbf{Z}'(t)\rangle\equiv\int_V\mathbf{Z}^{\mathrm{T}}(t)\cdot\mathbf{Z}'(t)\,\mathrm{d}\mathbf{r}$, where V denotes the system's volume. In addition to Eq. (2), the EM fields also satisfy $\nabla\cdot(\sqrt{\mu}\mathbf{X}(t))=0$ and $\nabla\cdot(\sqrt{\varepsilon}\mathbf{Y}(t))=0$ [1]. Throughout this paper we use dimensionless quantities: We measure distances in units of λ and express time and frequency in units of λ/c and c/λ, respectively.

A numerical algorithm that solves the time-dependent Maxwell equations necessarily involves some discretization procedure of the spatial derivatives in Eq. (2). Ideally, this procedure should not change the basic symmetries of the Maxwell equations. We will not discuss the (important) technicalities of the spatial discretization (we refer the reader to Refs. [2,3]) as this is not essential to the discussion that follows. On a spatial grid Maxwell's curl equations (2) can be written in the compact form [11]

$$\frac{\partial}{\partial t}\boldsymbol{\Psi}(t)=H\boldsymbol{\Psi}(t)-\boldsymbol{\Phi}(t). \tag{4}$$

The vector $\boldsymbol{\Psi}(t)$ is a representation of $\mathbf{Z}(t)$ on the grid. The matrix H is the discrete analogue of the operator (3), and the vector $\boldsymbol{\Phi}(t)$ contains all the information on the current source $\mathbf{J}(t)$. The formal solution of Eq. (4) is given by

$$\boldsymbol{\Psi}(t)=U(t)\boldsymbol{\Psi}(0)-\int_0^t U(t-u)\boldsymbol{\Phi}(u)\,\mathrm{d}u, \tag{5}$$

where

$$U(t)=\mathrm{e}^{tH}, \tag{6}$$

denotes the time-evolution matrix. If the discretization procedure preserves the underlying symmetries of the time-dependent Maxwell equations then the matrix H is real and skew symmetric [11], implying that $U(t)$ is orthogonal [28]. Physically, the orthogonality of $U(t)$ implies conservation of energy [11].

3. Time integration algorithms

There are two, closely related, strategies to construct an algorithm for performing the time integration of the time-dependent Maxwell equations defined on the grid [17]. The traditional approach is to discretize (with increasing level of sophistication) the derivative with respect to time [17]. The other is to approximate the formally exact solution, i.e. the matrix exponential $U(t)=\mathrm{e}^{tH}$ by some time evolution matrix $\widetilde{U}(t)$ [16,17]. We adopt the latter approach in this paper as it facilitates the construction of algorithms with specific features, such as unconditional stability [11,16].

If the approximation $\widetilde{U}(t)$ is itself an orthogonal transformation, then $\|\widetilde{U}(t)\|=1$ where $\|X\|$ denotes the 2-norm of a vector or matrix X [28]. In the absence of source terms (i.e. $\boldsymbol{\Phi}(t)=\mathbf{0}$) this implies that $\|\widetilde{\boldsymbol{\Psi}}(t)\|=\|\widetilde{U}(t)\boldsymbol{\Psi}(0)\|=\|\boldsymbol{\Psi}(0)\|$, for an arbitrary initial condition $\boldsymbol{\Psi}(0)$ and for all times t and hence the time integration algorithm defined by $\widetilde{U}(t)$ is unconditionally stable by construction [16,17].

In the presence of current sources, for general $\widetilde{U}(t)$, it follows immediately from Eq. (5) that

$$\left\|\widetilde{\boldsymbol{\Psi}}(t)\right\| \leqslant \left\|\boldsymbol{\Psi}(t)\right\| + \tilde{\epsilon}\left(\left\|\boldsymbol{\Psi}(0)\right\| + \int_0^t \left\|\boldsymbol{\Phi}(u)\right\| \mathrm{d}u\right), \tag{7}$$

where $\|\widetilde{U}(u) - U(u)\| \leqslant \tilde{\epsilon}$ for $0 \leqslant u \leqslant t$ and $\tilde{\epsilon}$ is a measure for the accuracy of the approximation $\widetilde{U}(t)$.

From Eq. (5) it follows that the EM fields $\boldsymbol{\Psi}(t)$ change according to

$$\boldsymbol{\Psi}(t+\tau) = \mathrm{e}^{\tau H}\boldsymbol{\Psi}(t) - \int_t^{t+\tau} \mathrm{e}^{(t+\tau-u)H}\boldsymbol{\Phi}(u)\,\mathrm{d}u. \tag{8}$$

In the time-stepping approach we approximate the source term in Eq. (8) by the standard 3-point Gauss–Legendre quadrature formula [29]

$$\boldsymbol{\Psi}(t+\tau) = \mathrm{e}^{\tau H}\boldsymbol{\Psi}(t) - \frac{\tau}{2}\sum_{i=0}^{2} w_i \mathrm{e}^{(1+x_i)\tau H/2}\boldsymbol{\Phi}\big(t + (1+x_i)\tau/2\big) + \mathcal{O}(\tau^7), \tag{9}$$

where x_0, x_1, x_2 are the zeros of the Legendre polynomial $P_3(x) = x(5x^2 - 3)/2$ and $w_i = 8/(1 - x_i^2)(15x_i^2 - 3)^2$ [29]. In practice we replace $\mathrm{e}^{(1+x_i)\tau H/2}$ in Eq. (9) by an approximation $\widetilde{U}((1+x_i)\tau/2)$.

We now consider three options to construct the approximate time evolution matrix $\widetilde{U}(t)$. We exclude from the discussion the exceptional cases for which the matrix exponential $U(t) = \mathrm{e}^{tH}$ can be calculated directly, as these are usually of little relevance for realistic problems. The first option, which is based on the product-formula approach, yields the conventional Yee algorithm, a higher-order generalization thereof, and the unconditional schemes proposed in Refs. [11,12]. The second option is to use rational approximations to the matrix exponential, yielding the standard ADI methods. Finally, the Chebyshev polynomial approximation to the matrix exponential is used to construct a one-step algorithm [26,27].

3.1. Product-formula approach

As discussed in [11], a systematic approach to construct approximations to matrix exponentials is to make use of the Lie–Trotter–Suzuki product-formula [15,30]

$$\mathrm{e}^{tH} = \mathrm{e}^{t(H_1+\cdots+H_p)} = \lim_{m\to\infty}\left(\prod_{i=1}^{p}\mathrm{e}^{tH_i/m}\right)^m, \tag{10}$$

and generalizations thereof [31,32]. Expression (10) suggests that

$$U_1(\tau) = \mathrm{e}^{\tau H_1}\ldots\mathrm{e}^{\tau H_p}, \tag{11}$$

might be a good approximation to $U(\tau)$ if τ is sufficiently small. Applied to the case of interest here, if all the H_i are real and skew-symmetric $U_1(\tau)$ is orthogonal by construction and a numerical scheme based on Eq. (11) will be unconditionally stable. For orthogonal matrices $U(\tau)$ and $U_1(\tau)$ it can be shown that [16]

$$\left\|U(\tau) - U_1(\tau)\right\| \leqslant \frac{\tau^2}{2}\sum_{i<j}\left\|[H_i, H_j]\right\|, \tag{12}$$

where $[H_i, H_j] = H_iH_j - H_jH_i$ is, in general, non-zero. Relaxing the condition that $U(\tau)$ and $U_1(\tau)$ are orthogonal matrices changes the τ-dependence in Eq. (12) but for small τ the error still vanishes like τ^2 [32]. From Eq. (12) it follows that, in general, the Taylor series of $U(\tau)$ and $U_1(\tau)$ are identical up to first order in τ. We call $U_1(\tau)$ a 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 matrix

$$U_2(\tau) = U_1(-\tau/2)^{\mathrm{T}} U_1(\tau/2) = \mathrm{e}^{\tau H_p/2} \ldots \mathrm{e}^{\tau H_1/2} \mathrm{e}^{\tau H_1/2} \ldots \mathrm{e}^{\tau H_p/2}, \tag{13}$$

is a second-order approximation to $U(\tau)$ [31,32]. If $U_1(\tau)$ is orthogonal, so is $U_2(\tau)$. For orthogonal $U_2(\tau)$ we have [16]

$$\left\| U(t = m\tau) - \left[U_2(\tau)\right]^m \right\| \leqslant c_2 \tau^2 t, \tag{14}$$

where c_2 is a positive constant.

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

$$U_4(\tau) = U_2(a\tau) U_2(a\tau) U_2\big((1-4a)\tau\big) U_2(a\tau) U_2(a\tau), \tag{15}$$

where $a = 1/(4 - 4^{1/3})$. The approximations Eqs. (11) and (13), and (15) have proven to be very useful in many applications [15,16,31,33–41] and, as we show below, turn out to be equally useful for solving the time-dependent Maxwell equations. As before, for orthogonal $U_4(\tau)$ we have [16]

$$\left\| U(t = m\tau) - \left[U_4(\tau)\right]^m \right\| \leqslant c_4 \tau^4 t, \tag{16}$$

where c_4 is a positive constant.

As our numerical results (see below) show, for sufficiently small τ, the numerical error of a time integrator vanishes with τ according to the τ-dependence of the corresponding rigorous bound, e.g., Eqs. (12), (14), or (16). Our experience shows that if this behavior is not observed, there is a fair chance that the computer program contains one or more errors.

In practice an efficient implementation of the first-order scheme is all that is needed to construct the higher-order algorithms (Eqs. (13) and (15)). The crucial step of this approach is to choose the H_i's such that the matrix exponentials $\exp(\tau H_1), \ldots, \exp(\tau H_p)$ can be calculated efficiently. This will turn the formal expressions for $U_2(\tau)$ and $U_4(\tau)$ into efficient algorithms to solve the time-dependent Maxwell equations.

3.2. ADI algorithms

ADI algorithms are usually derived by starting from the differential equation and applying the operator splitting idea [42]. Not surprisingly, the same algorithms can be obtained by approximating the formal solution of the differential equation in terms of a product formula. Instead of hunting for a decomposition that leads to matrix exponentials $\exp(\tau H_1), \ldots, \exp(\tau H_p)$ that are easy to compute, one can opt for an algorithm in which each of these exponentials is calculated approximately. In principle this might be beneficial because there is more flexibility in decomposing H. The standard strategy, preserving the symmetry of $H_1, \ldots, H_p$ is to use rational (Padé) approximations to the exponential [17]. For instance, the approximation $\mathrm{e}^x \approx (1 + x/2)/(1 - x/2)$ with some decomposition $H = H_1 + H_2$ yields the second-order-accurate ADI algorithm [17,19,42]

$$U_2^{\mathrm{ADI}}(\tau) = (I - \tau H_1/2)^{-1} (I + \tau H_2/2)(I - \tau H_2/2)^{-1} (I + \tau H_1/2), \tag{17}$$

where I is the identity matrix. As the subscript indicates, the algorithm (17) is second-order accurate in time. For general skew-symmetric matrices H_1 and H_2, it is easy to show that the algorithm $U_2^{\mathrm{ADI}}(\tau)$ is unconditionally stable. Following Ref. [19] we rearrange factors and obtain

$$\begin{aligned} \left\| \left[U_2^{\mathrm{ADI}}(\tau)\right]^m \right\| &= \left\| (I - \tau H_1/2)^{-1} X_2 X_1 X_2 \ldots X_1 X_2 (I + \tau H_1/2) \right\| \\ &\leqslant \left\| (I - \tau H_1/2)^{-1} \right\| \| X_2 X_1 X_2 \ldots X_1 X_2 \| \left\| (I + \tau H_1/2) \right\| \\ &= \left\| (I - \tau H_1/2)^{-1} \right\| \left\| (I + \tau H_1/2) \right\|. \end{aligned} \tag{18}$$

We used the fact that for skew-symmetric H_i, $X_i \equiv (I - \tau H_i/2)^{-1}(I + \tau H_i/2)$ is orthogonal and that $\|X_2X_1X_2 \ldots X_1X_2\| = 1$. If X is skew-symmetric, it's eigenvalues are pure imaginary and therefore $(I - X)^{-1}$ is non-singular. Hence, for any number of time steps m, $\|[U_2^{\mathrm{ADI}}(\tau)]^m\| \leqslant C$, where C is some finite positive constant, proving that the $U_2^{\mathrm{ADI}}(\tau)$ algorithm is unconditionally stable in the Lax–Richtmyer sense [17].

The matrix inversions appearing in Eq. (17) suggest that for practical purposes the implicit method $U_2^{\mathrm{ADI}}(\tau)$ will not be very useful unless $I - \tau H_1/2$ and $I - \tau H_2/2$ take special forms that allow efficient matrix inversion [17,42].

3.3. One-step algorithm

For completeness, we briefly review the theory behind the one-step algorithm described in [26,27]. The basic idea of this approach is to make use of extremely accurate polynomial approximations to the matrix exponential. First we use the Chebyshev polynomial expansion to approximate $U(t)$ and then show how to treat the source term in Eq. (5). We begin by "normalizing" the matrix H. The eigenvalues of the skew-symmetric matrix H are pure imaginary numbers. In practice H is sparse so it is easy to compute $\|H\|_1 \equiv \max_j \sum_i |H_{i,j}|$. Then, by construction, the eigenvalues of $B \equiv -\mathrm{i}H/\|H\|_1$ all lie in the interval $[-1, 1]$ [28]. Expanding the initial value $\boldsymbol{\Psi}(0)$ in the (unknown) eigenvectors $\mathbf{b}_j$ of B, we find from Eq. (5) with $\boldsymbol{\Phi}(t) \equiv 0$:

$$\boldsymbol{\Psi}(t) = \mathrm{e}^{\mathrm{i}zB}\boldsymbol{\Psi}(0) = \sum_j \mathrm{e}^{\mathrm{i}zb_j}\mathbf{b}_j\langle\mathbf{b}_j|\boldsymbol{\Psi}(0)\rangle, \tag{19}$$

where $z = t\|H\|_1$ and the b_j denote the (unknown) eigenvalues of B. There is no need to know the eigenvalues and eigenvectors of B explicitly. We find the Chebyshev polynomial expansion of $U(t)$ by computing the expansion coefficients of each of the functions $\mathrm{e}^{\mathrm{i}zb_j}$ that appear in Eq. (19). In particular, as $-1 \leqslant b_j \leqslant 1$, we can use the expansion [29] $\mathrm{e}^{\mathrm{i}zb_j} = J_0(z) + 2\sum_{k=1}^{\infty} i^k J_k(z)T_k(b_j)$, where $J_k(z)$ is the Bessel function of integer order k, to write Eq. (19) as

$$\boldsymbol{\Psi}(t) = \left[J_0(z)I + 2\sum_{k=1}^{\infty} J_k(z)\widehat{T}_k(B)\right]\boldsymbol{\Psi}(0). \tag{20}$$

Here $\widehat{T}_k(B) = i^k T_k(B)$ is a matrix-valued modified Chebyshev polynomial that is defined by the recursion relations

$$\widehat{T}_0(B)\boldsymbol{\Psi}(0) = \boldsymbol{\Psi}(0), \qquad \widehat{T}_1(B)\boldsymbol{\Psi}(0) = \mathrm{i}B\boldsymbol{\Psi}(0), \tag{21}$$

and

$$\widehat{T}_{k+1}(B)\boldsymbol{\Psi}(0) = 2\mathrm{i}B\widehat{T}_k(B)\boldsymbol{\Psi}(0) + \widehat{T}_{k-1}(B)\boldsymbol{\Psi}(0), \tag{22}$$

for $k \geqslant 1$.

As $\|\widehat{T}_k(B)\| \leqslant 1$ by construction and $|J_k(z)| \leqslant |z|^k/2^k k!$ for z real [29], the kth coefficient in Eq. (20) vanishes exponentially fast for sufficiently large k. Thus, we can obtain an accurate approximation by summing the contributions in Eq. (20) with $k \leqslant K$ only. The number K is fixed by requiring that $|J_k(z)| < \kappa$ for all $k > K$. Here, κ is a control parameter that determines the accuracy of the approximation. For fixed κ, K increases linearly with $z = t\|H\|_1$ (there is no requirement on t being small). From numerical analysis it is known that for fixed K, the Chebyshev polynomial is very nearly the same polynomial as the minimax polynomial [42], i.e. the polynomial of degree K that has the smallest maximum deviation from the true function, and is much more accurate than for instance a Taylor expansion of the same degree K. In Fig. 1 we show a plot of $J_n(z = 200)$ as a function of n to illustrate these points. From Fig. 1 it is clear that the Chebyshev polynomial expansion will only be useful if K lies to the right of the right-most extremum of $J_n(z = 200)$, i.e. K has to be larger than 200 in this example.

We now turn to the treatment of the current source $\mathbf{J}(t)$. The contribution of the source term to the EM field at time t is given by the last term in Eq. (5). For simplicity we only consider the case of a sinusoidal source

$$\mathbf{J}(\mathbf{r}, t) = \Theta(t)\Theta(T - t)\mathbf{s}(\mathbf{r})\sin(\Omega t), \tag{23}$$

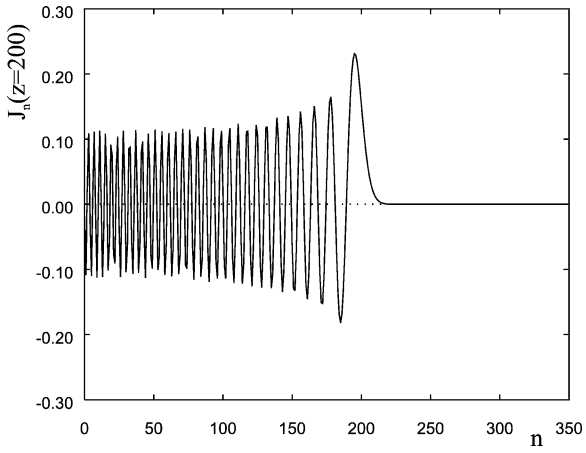

Fig. 1. Dependence of the Bessel function $J_n(z = 200)$ on the order n. The value of z corresponds to the time $t = 100$ used in Figs. 3–6 and Tables 1–3.

where $\mathbf{s}(\mathbf{r})$ specifies the spatial distribution and $\Omega \equiv 2\pi f_s$ the angular frequency of the source. The step functions $\Theta(t)$ and $\Theta(T - t)$ indicate that the source is turned on at $t = 0$ and is switched off at $t = T$. Note that Eq. (23) may be used to compose sources with a more complicated time dependence by a Fourier sine transformation.

The formal expression for the contribution of the sinusoidal source (23) reads

$$\int_0^t \mathrm{e}^{(t-u)H}\boldsymbol{\Phi}(u)\,\mathrm{d}u = \left(\Omega^2 + H^2\right)^{-1}\mathrm{e}^{(t-T')H} \times \left(\Omega\,\mathrm{e}^{T'H} - \Omega\cos\Omega T' - H\sin\Omega T'\right)\boldsymbol{\Xi}$$

$$\equiv f(H, t, T', \Omega)\boldsymbol{\Xi}, \tag{24}$$

where $T' = \min(t, T)$ and $\boldsymbol{\Phi}(u) \equiv \Theta(t)\Theta(T - t)\sin(\Omega t)\boldsymbol{\Xi}$. The vector $\boldsymbol{\Xi}$ represents the spatial (time-independent) distribution $\mathbf{s}(\mathbf{r})$ and has the same dimension as $\boldsymbol{\Psi}(0)$. The coefficients of the Chebyshev polynomial expansion of the formal solution (24) are calculated as follows. First we repeat the scaling procedure described above and substitute in Eq. (24) $H = \mathrm{i}x\|H\|_1$, $t = z/\|H\|_1$, $T' = Z'/\|H\|_1$, and $\Omega = \omega\|H\|_1$. Then, we compute the (Fast) Fourier Transform with respect to x of the function $f(x, z, Z', \omega)$ (which is non-singular on the interval $-1 \leqslant x \leqslant 1$). By construction, the Fourier coefficients $S_k(t\|H\|_1)$ are the coefficients of the Chebyshev polynomial expansion [29].

Taking into account all contributions of the source term with k smaller than K' (determined by a procedure similar to the one for K), the one-step algorithm to compute the EM fields at time t reads

$$\widehat{\boldsymbol{\Psi}}(t) = \left[J_0\left(t\|H\|_1\right)I + 2\sum_{k=1}^{K} J_k\left(t\|H\|_1\right)\widehat{T}_k(B)\right]\boldsymbol{\Psi}(0) - \left[S_0\left(t\|H\|_1\right)I + 2\sum_{k=1}^{K'} S_k\left(t\|H\|_1\right)\widehat{T}_k(B)\right]\boldsymbol{\Xi}. \tag{25}$$

Note that in this one-step approach the time dependence of the source is taken into account exactly, without actually sampling it.

In a strict sense, the one-step method does not yield an orthogonal approximation. However, for practical purposes it can be viewed as an extremely stable time-integration algorithm because it yields an approximation to the exact time evolution operator $U(t) = \mathrm{e}^{tH}$ that is exact to nearly machine precision, i.e. in practice the value of $\tilde{\epsilon}$ in Eq. (7) is very small. This also implies that within the same precision $\nabla \cdot (\mu\mathbf{H}(t)) = \nabla \cdot (\mu\mathbf{H}(t = 0))$ and $\nabla \cdot (\varepsilon\mathbf{E}(t)) = \nabla \cdot (\varepsilon\mathbf{E}(t = 0))$ holds for all times, implying that the numerical scheme will not produce artificial charges during the time integration [2,3].

4. Implementation

The basic steps in the construction of the product-formula and one-step algorithms are best illustrated by considering the simplest case, i.e. the Maxwell equations of a 1D homogeneous problem. From a conceptual point of view nothing is lost by doing this: the extension to 2D and 3D inhomogeneous problems is straightforward, albeit technically non-trivial [11,12,26,27].

We consider a system, infinitely large in the y and z direction, for which $\varepsilon = 1$ and $\mu = 1$. Under these conditions, the Maxwell equations reduce to two independent sets of first-order differential equations [1], the transverse electric (TE) mode and the transverse magnetic (TM) mode [1]. As the equations of the TE- and TM-mode differ by a sign we can restrict our considerations to the TM-mode only. The magnetic field $H_y(x,t)$ and the electric field $E_z(x,t)$ of the TM-mode in the 1D cavity of length L are solutions of

$$\frac{\partial}{\partial t}H_y(x,t) = \frac{\partial}{\partial x}E_z(x,t), \tag{26}$$

$$\frac{\partial}{\partial t}E_z(x,t) = \frac{\partial}{\partial x}H_y(x,t) - J_z(x,t), \tag{27}$$

subject to the boundary condition $E_z(0,t) = E_z(L,t) = 0$ [1]. Note that the divergence of both fields is trivially zero.

Following Yee [4], to discretize Eqs. (26) and (27), it is convenient to assign H_y to odd and E_z to even numbered lattice sites, as shown in Fig. 2. Using the second-order central-difference approximation to the first derivative with respect to x, we obtain

$$\frac{\partial}{\partial t}H_y(2i+1,t) = \delta^{-1}\big(E_z(2i+2,t) - E_z(2i,t)\big), \tag{28}$$

$$\frac{\partial}{\partial t}E_z(2i,t) = \delta^{-1}\big(H_y(2i+1,t) - H_y(2i-1,t)\big) - J_z(2i,t), \tag{29}$$

where we have introduced the notation $A(i,t) = A(x = i\delta/2, t)$. The integer i labels the grid points and δ denotes the distance between two next-nearest neighbors on the lattice (hence the absence of a factor two in the nominator). We define the n-dimensional vector $\boldsymbol{\Psi}(t)$ by

$$\boldsymbol{\Psi}(i,t) = \begin{cases} H_y(i,t), & i \text{ odd}, \\ E_z(i,t), & i \text{ even}. \end{cases} \tag{30}$$

The vector $\boldsymbol{\Psi}(t)$ contains both the magnetic and the electric field on the lattice points $i = 1,\ldots,n$. The ith element of $\boldsymbol{\Psi}(t)$ is given by the inner product $\boldsymbol{\Psi}(i,t) = \mathbf{e}_i^{\mathrm{T}} \cdot \boldsymbol{\Psi}(t)$ where $\mathbf{e}_i$ denotes the ith unit vector in the n-dimensional vector space. Using this notation (which proves most useful for the case of 2D and 3D for which it is rather cumbersome to write down explicit matrix representations), it is easy to show that Eqs. (28) and (29) can be

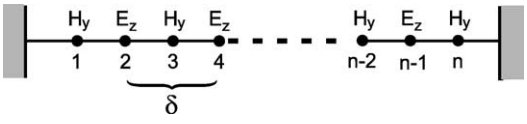

Fig. 2. Positions of the two TM-mode field components on the one-dimensional grid. The distance between two next-nearest neighbors is denoted by δ.

written in the form of Eq. (4) where the matrix H is given by

$$H = \delta^{-1} \sum_{i=1}^{n-1} \left(\mathbf{e}_i \ \mathbf{e}_{i+1}^{\mathrm{T}} - \mathbf{e}_{i+1} \mathbf{e}_i^{\mathrm{T}} \right) = \begin{pmatrix} 0 & \delta^{-1} & & & \\ -\delta^{-1} & 0 & \delta^{-1} & & \\ & \ddots & \ddots & \ddots & \\ & & -\delta^{-1} & 0 & \delta^{-1} \\ & & & -\delta^{-1} & 0 \end{pmatrix}. \tag{31}$$

We immediately see that H is sparse and skew-symmetric by construction.

4.1. Yee-type algorithms

First we demonstrate that the conventional Yee algorithm fits into the product-formula approach. For the 1D model (31) it is easy to see that one time-step with the Yee algorithm corresponds to the operation

$$U_1^{\mathrm{Yee}}(\tau) = (I + \tau A)(I - \tau A^{\mathrm{T}}) = \mathrm{e}^{\tau A} \mathrm{e}^{-\tau A^{\mathrm{T}}}, \tag{32}$$

where

$$A = \delta^{-1} \sum_{i=2}^{n-1}{}' \left(\mathbf{e}_i \ \mathbf{e}_{i-1}^{\mathrm{T}} - \mathbf{e}_i \ \mathbf{e}_{i+1}^{\mathrm{T}} \right) = \begin{pmatrix} 0 & \delta^{-1} & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & -\delta^{-1} & 0 & \delta^{-1} & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & -\delta^{-1} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}, \tag{33}$$

and we used the arrangements of H and E fields as defined by Eq. (30). We use the notation $\sum'$ to indicate that the stride of the summation index is two. Note that since $A^2 = 0$ we have $\mathrm{e}^{\tau A} = 1 + \tau A$ *exactly*. Therefore we recover the time-step operator of the Yee algorithm using the first-order product-formula approximation to $\mathrm{e}^{\tau H}$ and decomposing $H = A - A^{\mathrm{T}}$. However, the conventional Yee algorithm is second-order, not first-order, accurate in time [2,3]. This is due to the use of a staggered grid in time [2,3]. To perform one time step with the conventional Yee algorithm we need to know the values of $E_z(t)$ and $H_y(t + \tau/2)$, not $H_y(t)$. Another method has to supply the H_y-field at a time shifted by $\tau/2$. We will denote the conventional Yee algorithm, i.e. the algorithm that uses the staggered space-and-time grid and is second-order accurate in both space and time (not $U_1^{\mathrm{Yee}}(\tau)$), by "C-Yee" and reserve the use of the upperscript $^{\mathrm{Yee}}$ to algorithms that have been constructed by decomposing $H = A - A^{\mathrm{T}}$ but are not using the staggered-in-time grid.

Within the spirit of this approach, we can easily eliminate the staggered-in-time grid at virtually no extra computational cost or programming effort (if a conventional Yee code is available) by using the second-order product formula

$$U_2^{\mathrm{Yee}}(\tau) = \mathrm{e}^{\tau A/2} \mathrm{e}^{-\tau A^{\mathrm{T}}} \mathrm{e}^{\tau A/2} = (I + \tau A/2)(I - \tau A^{\mathrm{T}})(I + \tau A/2). \tag{34}$$

The effect of the last factor is to propagate the H_y-field by $\tau/2$. The middle factor propagates the E_z-field by τ. The first factor again propagates the H_y-field by $\tau/2$. In this scheme all EM fields are to be taken at the same time. The algorithm defined by $U_2^{\mathrm{Yee}}(\tau)$ is second-order accurate in time by construction [16]. Note that $\mathrm{e}^{\tau A/2}$ is not orthogonal so nothing has been gained in terms of stability. Since $(U_2^{\mathrm{Yee}}(\tau))^m = \mathrm{e}^{-\tau A/2} (U_1^{\mathrm{Yee}}(\tau))^m \mathrm{e}^{+\tau A/2}$, we see that, compared to the conventional Yee algorithm, the extra computational work is proportional to $(1 + 2/m)$, hence negligible if the number of time steps m is large.

According to the general theory outlined in Section 3, the expression

$$U_4^{\mathrm{Yee}}(\tau) = U_2^{\mathrm{Yee}}(a\tau) U_2^{\mathrm{Yee}}(a\tau) U_2^{\mathrm{Yee}}\big((1 - 4a)\tau\big) U_2^{\mathrm{Yee}}(a\tau) U_2^{\mathrm{Yee}}(a\tau), \tag{35}$$

defines a fourth-order accurate Yee-like scheme, the realization of which requires almost no effort once U_2^{Yee} has been implemented. It is easy to see that the above construction of the Yee-like algorithms holds for the much more complicated 2D, and 3D inhomogeneous case as well. Also note that the fourth-order Yee algorithm U_4^{Yee} does not require extra storage to hold field values at intermediate times.

4.2. Unconditionally stable algorithms

Guided by previous work on Schrödinger and diffusion problems we split H into two parts such that $H = H_1 + H_2$, dividing the lattice into odd and even numbered cells [16,35,40]. Following Ref. [11] we have

$$H_1 = \delta^{-1} \sum_{i=1}^{n-1}{}' \left(\mathbf{e}_i\ \mathbf{e}_{i+1}^{\mathrm{T}} - \mathbf{e}_{i+1}\mathbf{e}_i^{\mathrm{T}}\right) = \begin{pmatrix} 0 & \delta^{-1} & 0 & 0 & 0 & \cdots \\ -\delta^{-1} & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & \delta^{-1} & 0 & \cdots \\ 0 & 0 & -\delta^{-1} & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & -\delta^{-1} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}, \tag{36}$$

$$H_2 = \delta^{-1} \sum_{i=1}^{n-2}{}' \left(\mathbf{e}_{i+1}\mathbf{e}_{i+2}^{\mathrm{T}} - \mathbf{e}_{i+2}\mathbf{e}_{i+1}^{\mathrm{T}}\right) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \delta^{-1} & 0 & 0 & \cdots \\ 0 & -\delta^{-1} & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & \delta^{-1} & \cdots \\ 0 & 0 & 0 & -\delta^{-1} & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}. \tag{37}$$

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 is given by

$$\widetilde{U}_1(\tau) = \mathrm{e}^{\tau H_1}\mathrm{e}^{\tau H_2} = \left\{ \prod_{i=1}^{n-1}{}' \exp\left[\tau\delta^{-1}\left(\mathbf{e}_i\ \mathbf{e}_{i+1}^{\mathrm{T}} - \mathbf{e}_{i+1}\mathbf{e}_i^{\mathrm{T}}\right)\right] \right\} \left\{ \prod_{i=1}^{n-2}{}' \exp\left[\tau\delta^{-1}\left(\mathbf{e}_{i+1}\mathbf{e}_{i+2}^{\mathrm{T}} - \mathbf{e}_{i+2}\mathbf{e}_{i+1}^{\mathrm{T}}\right)\right] \right\}. \tag{38}$$

To derive Eq. (38) we used the block-diagonal structure of H_1 and H_2 (see Eqs. (36) and (37)) and obtained an exact expression for $\widetilde{U}_1(\tau)$ in terms of an ordered product of matrix exponentials: the order of the matrix exponentials between each pair of curly brackets is irrelevant as these matrices commute with each other. Each of these matrix exponentials only operates on a pair of elements of $\boldsymbol{\Psi}(t)$ and leaves other elements intact. The indices of each of these pairs are given by the subscripts of $\mathbf{e}$ and $\mathbf{e}^{\mathrm{T}}$. From Eq. (38) it is clear what a program should do: Make loops over i with stride 2. For each i pick a pair of elements from $\Psi(t)$ according to the subscripts of $\mathbf{e}$ and $\mathbf{e}^{\mathrm{T}}$, compute (or fetch from memory) the elements of the plane rotation (see Eq. (39)), perform the plane rotation, i.e. multiply the 2×2 matrices and the vectors of length two, and overwrite the same two elements. 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 $\mathrm{e}^{\tau H_1}$ (or $\mathrm{e}^{\tau H_2}$) 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 $\mathrm{e}^{\tau H_1}$ or $\mathrm{e}^{\tau H_2}$ is given by

$$\exp\left[\alpha \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\right] \begin{pmatrix} \Psi(i,t) \\ \Psi(j,t) \end{pmatrix} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} \Psi(i,t) \\ \Psi(j,t) \end{pmatrix}. \tag{39}$$

Using the algorithm to compute Eq. (38), it is easy to construct the unconditionally stable, higher-order algorithms $\widetilde{U}_2(\tau)$ and $\widetilde{U}_4(\tau)$, see Eqs. (13) and (15).

Obviously, the decomposition into H_1 Eq. (36) and H_2 Eq. (37) yields the most simple real-space algorithm. It is not difficult to imagine that a better but slightly more complicated algorithm can be constructed by using blocks of 3×3 instead of 2×2 matrices. Thus we are lead to consider the decomposition

$$H_3 = \delta^{-1} \sum_{i=1}^{n-2}{}'' \left(\mathbf{e}_i \, \mathbf{e}_{i+1}^{\mathrm{T}} + \mathbf{e}_{i+1}\mathbf{e}_{i+2}^{\mathrm{T}} - \mathbf{e}_{i+1}\mathbf{e}_i^{\mathrm{T}} - \mathbf{e}_{i+2}\mathbf{e}_{i+1}^{\mathrm{T}} \right) = \begin{pmatrix} 0 & \delta^{-1} & 0 & 0 & 0 & \cdots \\ -\delta^{-1} & 0 & \delta^{-1} & 0 & 0 & \cdots \\ 0 & -\delta^{-1} & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & -\delta^{-1} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}, \tag{40}$$

$$H_4 = \delta^{-1} \sum_{i=1}^{n-4}{}'' \left(\mathbf{e}_{i+2}\mathbf{e}_{i+3}^{\mathrm{T}} + \mathbf{e}_{i+3}\mathbf{e}_{i+4}^{\mathrm{T}} - \mathbf{e}_{i+3}\mathbf{e}_{i+2}^{\mathrm{T}} - \mathbf{e}_{i+4}\mathbf{e}_{i+3}^{\mathrm{T}} \right) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & \delta^{-1} & 0 & \cdots \\ 0 & 0 & -\delta^{-1} & 0 & \delta^{-1} & \cdots \\ 0 & 0 & 0 & -\delta^{-1} & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}. \tag{41}$$

where the double prime indicates that the stride of the index i is three. Obviously both H_3 and H_4 are skew-symmetric block-diagonal matrices, built from the 3×3 skew-symmetric matrix

$$B = \begin{pmatrix} 0 & \delta^{-1} & 0 \\ -\delta^{-1} & 0 & \delta^{-1} \\ 0 & -\delta^{-1} & 0 \end{pmatrix}. \tag{42}$$

As $B^3 = -2B$ we have

$$\mathrm{e}^{\tau B} = 1 + sB + cB^2 = \begin{pmatrix} 1-c & s & c \\ -s & 1-2c & s \\ -c & -s & 1-c \end{pmatrix}, \tag{43}$$

where $s = \sin(\sqrt{2}\,\tau)$ and $c = \sin^2(\tau/\sqrt{2})$. In practice, using the 3×3 instead of the 2×2 decomposition is marginally more difficult. We will denote the corresponding second- and fourth-order algorithm by $\widetilde{U}_2^{3\times3}$ and $\widetilde{U}_4^{3\times3}$, respectively.

4.3. ADI algorithm

For the tri-diagonal matrix (31), the ADI algorithm reduces to the Cranck–Nicholson method [42]. The tri-diagonal structure of the matrix H permits the calculation of $(I - \tau H/2)^{-1}\boldsymbol{\Psi}$ in $\mathcal{O}(n)$ operations by standard linear algebra methods [42].

4.4. One-step algorithm

The one-step algorithm is based on the recursion (see Eqs. (21) and (22))

$$\boldsymbol{\Psi}_{k+1} = \frac{2H}{\|H\|_1}\boldsymbol{\Psi}_k + \boldsymbol{\Psi}_{k-1}. \tag{44}$$

Thus, the explicit form Eq. (31) is all we need to implement the matrix–vector operation (i.e. $\Psi' \leftarrow H\Psi$) that enters Eq. (44).

The coefficients $J_k(z)$ and $S_k(z)$ (see Eq. (25)) should be calculated to high precision. Using the recursion relation of the Bessel functions, all K coefficients can be obtained with $\mathcal{O}(K)$ arithmetic operations [42]. The numbers $S_k(z)$ can be calculated in $\mathcal{O}(K \log K)$ by standard Fast Fourier transformation techniques. Clearly both computations are a negligible fraction of the total computational cost for solving the Maxwell equations.

Performing one time step amounts to repeatedly using recursion (22) to obtain $\widehat{T}_k(B)\Psi(0)$ for $k = 2, \ldots, K$, multiply the elements of this vector by $J_k(z)$ (or $S_k(z)$) and add all contributions. This procedure requires storage for two vectors of the same length as $\Psi(0)$ and some code to multiply such a vector by the sparse matrix H. The result of performing one time step yields the solution at time t, hence the name one-step algorithm. In contrast to what Eqs. (21) and (22) might suggest, the algorithm does not require the use of complex arithmetic.

5. Numerical experiments

Except for the conventional Yee algorithm, all algorithms discussed in this paper operate on the vector of fields defined at the same time t. We use the one-step algorithm (with a time step $\tau/2$) to compute $E_z(\tau/2)$ and $H_y(\tau/2)$. Then we use $E_z(0)$ and $H_y(\tau/2)$ as the initial values for the conventional Yee algorithm. In the presence of a current source, there are some ambiguities with this procedure as it is not obvious how to treat the source term in Eq. (9). In order to compare of the final result of the conventional Yee algorithm with those of the other methods, we use the one-step algorithm once more to shift the time of the H_y field by $-\tau/2$. This procedure to prepare the initial and to analyze the final state of the Yee algorithm does in fact make the results of the conventional Yee algorithm look a little more accurate than they would be if the exact data of the $\tau/2$-shifted fields are not available. Thus, the results on the errors of the conventional Yee algorithm presented in this paper give a too optimistic view on the accuracy of this algorithm but we nevertheless adopt the above procedure to make a quantitative comparison between the various algorithms.

We define the error of the solution $\widetilde{\Psi}(t)$ for the wave form by $\|\widetilde{\Psi}(t) - \widehat{\Psi}(t)\|/\|\widehat{\Psi}(t)\|$. Here and in the sequel, the caret $\widehat{\ }$ on top of a symbol indicates that the results have been obtained by means of the one-step algorithm. Thus, $\widehat{\Psi}(t)$ is the vector of EM fields obtained by the one-step algorithm. In our definition of the error we have already assumed that the one-step algorithm yields the exact (within numerical precision) results but this has to be demonstrated of course. A comparison of the results of an unconditionally stable algorithm, e.g., $\widetilde{U}_4$ with those of the one-step algorithm is sufficient to show that within rounding errors the latter yields the exact answer. Using the triangle inequality

$$\left\|\Psi(t) - \widehat{\Psi}(t)\right\| \leqslant \left\|\Psi(t) - \widetilde{\Psi}(t)\right\| + \left\|\widetilde{\Psi}(t) - \widehat{\Psi}(t)\right\|, \tag{45}$$

and the rigorous bound

$$\left\|\Psi(t) - \widetilde{\Psi}(t)\right\| \leqslant c_4\tau^4 t\left(\left\|\Psi(0)\right\| + \int_0^t \left\|\mathbf{J}(u)\right\| \mathrm{d}u\right), \tag{46}$$

we can be confident that the one-step algorithm yields the numerically exact answer if (i) Eq. (46) is not violated and (ii) if $\|\widetilde{\Psi}(t) - \widehat{\Psi}(t)\|$ vanishes like τ^4.

If an algorithm conserves the norm of the normalized vector $\Psi(t)$ (as in the case of the unconditionally stable algorithms discussed in this paper), in the absence of current sources, the difference between the exact and approximate solution can never exceed 2 ($\|\widetilde{\Psi}(t) - \widehat{\Psi}(t)\| \leqslant \|\widetilde{\Psi}(t)\| + \|\widehat{\Psi}(t)\| = 2$). Therefore, the behavior of the error as a function of τ has no significance if the error is of the order of one.

In Fig. 3 we show a typical result of a one-step calculation on a grid of $n = 5001$ sites with $\delta = 0.1$ (corresponding to a physical length of 250.05), and a current source placed at $i = 2500$ to eliminate possible

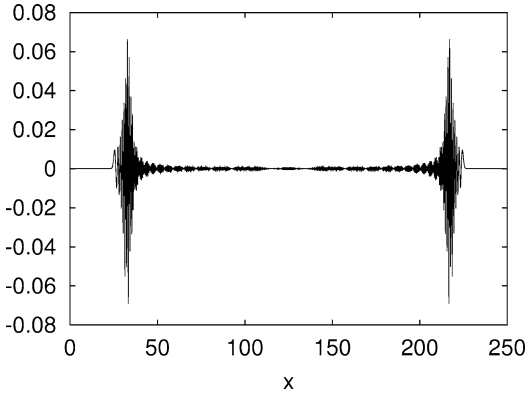

Fig. 3. The field $H_y(x, t = 100)$ generated by a current source at $x = 125$ that oscillates at frequency $f_s = 2\pi$ during the interval $0 \leqslant t \leqslant 6$, as obtained by the one-step algorithm with $K' = 2103$ ($K = 0$ in this case).

Table 1
The error $\|\widetilde{\Psi}(t) - \widehat{\Psi}(t)\|/\|\widehat{\Psi}(t)\|$ at time $t = 100$ as a function of the time step τ for eight different FDTD algorithms. The current source is positioned at the center of the system (see Fig. 3), and oscillates at frequency $f_s = 2\pi$ during the interval $0 \leqslant t \leqslant 6$, $\widehat{\Psi}(t)$ is the vector obtained by the one-step algorithm with $\kappa = 10^{-9}$, using $K' = 2103$ matrix–vector operations $\Psi' \leftarrow M\Psi$. The largest time step ($\tau = 0.1$) corresponds to the upperbound on the Courant number (τ/δ) for which the conventional Yee algorithm is stable [2]. C-Yee: $\widetilde{\Psi}(t)$ obtained by the conventional Yee algorithm [2–4]; Other columns: $\widetilde{\Psi}(t)$ obtained by the algorithms indicated

τ	C-Yee	U_2^{Yee}	U_2^{ADI}	$\widetilde{U}_2$	$\widetilde{U}_2^{3\times3}$	U_4^{Yee}	$\widetilde{U}_4$	$\widetilde{U}_4^{3\times3}$
0.10000E+0	0.16E+1	0.15E+1	0.15E+1	0.15E+1	0.14E+1	0.15E+0	0.37E+0	0.27E+0
0.50000E−1	0.18E+1	0.18E+1	0.13E+1	0.16E+1	0.13E+1	0.36E−1	0.33E−1	0.23E−1
0.25000E−1	0.68E+0	0.65E+0	0.12E+1	0.72E+0	0.12E+1	0.25E−2	0.22E−2	0.15E−2
0.12500E−1	0.21E+0	0.19E+0	0.35E+0	0.13E+1	0.31E+0	0.16E−3	0.14E−3	0.96E−4
0.62500E−2	0.63E−1	0.55E−1	0.10E+0	0.35E+0	0.78E−1	0.99E−5	0.87E−5	0.60E−5
0.31250E−2	0.19E−1	0.14E−1	0.28E−1	0.88E−1	0.20E−1	0.62E−6	0.55E−6	0.38E−6
0.15625E−2	0.61E−2	0.35E−2	0.69E−2	0.22E−1	0.49E−2	0.39E−7	0.34E−7	0.24E−7
0.78125E−3	0.23E−2	0.86E−2	0.17E−2	0.55E−2	0.12E−2	0.24E−8	0.21E−8	0.15E−8
0.39063E−3	0.10E−2	0.22E−3	0.43E−3	0.14E−2	0.31E−3	0.24E−9	0.24E−9	0.22E−9

artifacts of the boundaries. The frequency of the source is set to one ($f_s = 2\pi$) and the number of periods the source radiates is set to six (i.e. $T = 6$). In Table 1 (Fig. 4) we present results for the errors, as obtained by repeating the simulation shown in Fig. 3 using eight different FDTD methods. In Tables 2 and 3 (Figs. 5 and 6, respectively) we show similar results but instead of using a current source, a random wave form (Table 2) and Gaussian wave packet (Table 3) was taken as the initial condition.

From the data in Tables 1–3 we conclude that the error of algorithm $\widetilde{U}_4$ vanishes like τ^4, demonstrating that the one-step algorithm yields the numerically exact result (see Eqs. (45) and (46)). The results presented in Tables 2 and 3 have been obtained by using a vector of initial values that is normalized to one, i.e. $\|\Psi(0)\| = 1$. As $\|\widehat{\Psi}(t)\| = 1$ for $0 \leqslant t \leqslant 100$ to at least 10 digits, $\|\widetilde{\Psi}(t) - \widehat{\Psi}(t)\|/\|\widehat{\Psi}(t)\| = \|\widetilde{\Psi}(t) - \widehat{\Psi}(t)\|$ for all entries in Tables 2 and 3. The high precision of the one-step algorithm also allows us to use it for genuine time stepping with arbitrarily large time steps, this in spite of the fact that strictly speaking, the one-step algorithm is not unconditionally stable.

The data in Tables 1–3 suggests that there does not seem to be a significant difference between the conventional Yee algorithm and its variant U_2^{Yee} but in fact there is. The time evolution matrix corresponding to the Yee and the U_2^{Yee} algorithm is not orthogonal. Therefore the energy of the electromagnetic field ($W = \|\Psi(t)\|^2$) is not conserved. Furthermore, in the conventional Yee algorithm, the E- and H-fields are time-shifted by $\tau/2$ with respect to each other. Obviously, by construction U_2^{Yee} does not rely on staggered-in-time E- and H-fields and it

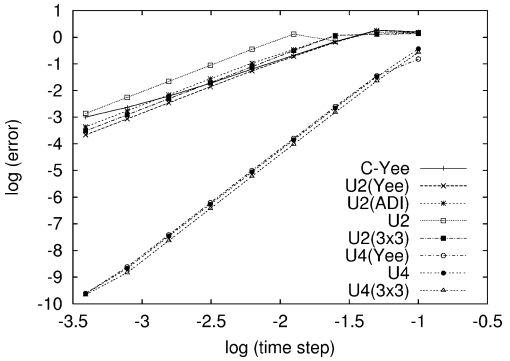

Fig. 4. The data presented in Table 1 plotted on a double logarithmic scale. C-Yee: data obtained by the conventional Yee algorithm [2–4]. The largest time step ($\tau = 0.1$) corresponds to the upperbound on the Courant number (τ/δ) for which the conventional Yee algorithm is stable [2]. Lines are guides to the eye.

Table 2
The error $\|\widetilde{\boldsymbol{\Psi}}(t) - \widehat{\boldsymbol{\Psi}}(t)\|/\|\widehat{\boldsymbol{\Psi}}(t)\|$ at time $t = 100$ as a function of the time step τ for eight different FDTD algorithms. The system is the same as in Fig. 3 and Table 1. The initial values of the EM fields are random, distributed uniformly over the interval $[-1, 1]$. $\widehat{\boldsymbol{\Psi}}(t)$ is the vector obtained by the one-step algorithm $\kappa = 10^{-9}$, using $K = 2080$ matrix–vector operations $\boldsymbol{\Psi}' \leftarrow M\boldsymbol{\Psi}$. The largest time step ($\tau = 0.1$) corresponds to the upperbound on the Courant number (τ/δ) for which the conventional Yee algorithm is stable [2]. C-Yee: $\widetilde{\boldsymbol{\Psi}}(t)$ obtained by the conventional Yee algorithm [2–4]; Other columns: $\widetilde{\boldsymbol{\Psi}}(t)$ obtained by the algorithms indicated

τ	C-Yee	U_2^{Yee}	U_2^{ADI}	$\widetilde{U}_2$	$\widetilde{U}_2^{3\times3}$	U_4^{Yee}	$\widetilde{U}_4$	$\widetilde{U}_4^{3\times3}$
0.10000E+0	0.99E+1	0.11E+2	0.14E+1	0.15E+1	0.17E+1	0.11E+1	0.13E+1	0.13E+1
0.50000E−1	0.13E+1	0.13E+1	0.13E+1	0.13E+1	0.14E+1	0.78E+0	0.16E+0	0.16E+0
0.25000E−1	0.13E+1	0.13E+1	0.13E+1	0.13E+1	0.12E+1	0.57E−2	0.11E−1	0.11E−1
0.12500E−1	0.12E+1	0.12E+1	0.14E+1	0.12E+1	0.63E+0	0.36E−2	0.71E−3	0.71E−3
0.62500E−2	0.70E+0	0.70E+0	0.12E+1	0.32E+0	0.16E+0	0.22E−3	0.45E−4	0.45E−4
0.31250E−2	0.18E+0	0.18E+0	0.36E+0	0.82E−1	0.41E−1	0.14E−4	0.28E−5	0.28E−5
0.15625E−2	0.46E−1	0.46E−1	0.92E−1	0.20E−1	0.10E−1	0.89E−6	0.17E−6	0.17E−6
0.78125E−3	0.11E−1	0.11E−1	0.23E−1	0.51E−2	0.26E−2	0.56E−7	0.11E−7	0.28E−8
0.39063E−3	0.29E−2	0.29E−2	0.57E−2	0.13E−2	0.64E−3	0.35E−8	0.68E−9	0.18E−9

Table 3
The error $\|\widetilde{\boldsymbol{\Psi}}(t) - \widehat{\boldsymbol{\Psi}}(t)\|/\|\widehat{\boldsymbol{\Psi}}(t)\|$ at time $t = 100$ as a function of the time step τ for eight different FDTD algorithms. The system is the same as in Fig. 3 and Table 1. The initial state of the EM fields is a Gaussian wave packet ($E_z(t) = \exp(-(x - x_0 - t)^2/\sigma^2)$) with a width $\sigma = 4$, and its center $x_0 = 125$ positioned at the middle of the system (see Fig. 3). $\widehat{\boldsymbol{\Psi}}(t)$ is the vector obtained by the one-step algorithm with $\kappa = 10^{-9}$, using $K = 2080$ matrix–vector operations $\boldsymbol{\Psi}' \leftarrow M\boldsymbol{\Psi}$. The largest time step ($\tau = 0.1$) corresponds to the upperbound on the Courant number (τ/δ) for which the conventional Yee algorithm is stable [2]. C-Yee: $\widetilde{\boldsymbol{\Psi}}(t)$ obtained by the conventional Yee algorithm [2–4]; Other columns: $\widetilde{\boldsymbol{\Psi}}(t)$ obtained by the algorithms indicated

τ	C-Yee	U_2^{Yee}	U_2^{ADI}	$\widetilde{U}_2$	$\widetilde{U}_2^{3\times3}$	U_4^{Yee}	$\widetilde{U}_4$	$\widetilde{U}_4^{3\times3}$
0.10000E+0	0.25E−2	0.25E−2	0.50E−2	0.14E+1	0.79E+0	0.28E−6	0.15E−1	0.17E−1
0.50000E−1	0.63E−3	0.63E−3	0.13E−2	0.90E+0	0.25E+0	0.17E−7	0.95E−3	0.15E−3
0.25000E−1	0.16E−3	0.16E−3	0.32E−3	0.26E+0	0.65E−1	0.11E−8	0.60E−4	0.97E−4
0.12500E−1	0.40E−4	0.39E−4	0.79E−4	0.65E−1	0.16E−1	0.69E−10	0.37E−5	0.61E−5
0.62500E−2	0.99E−5	0.98E−5	0.20E−4	0.16E−1	0.41E−2	0.12E−10	0.23E−6	0.38E−6
0.31250E−2	0.25E−5	0.25E−5	0.49E−5	0.41E−2	0.10E−2	0.12E−10	0.15E−7	0.24E−7
0.15625E−2	0.63E−6	0.61E−6	0.12E−5	0.10E−2	0.26E−3	0.12E−10	0.91E−9	0.15E−8
0.78125E−3	0.16E−6	0.15E−6	0.31E−6	0.25E−3	0.64E−4	0.12E−10	0.55E−10	0.10E−9
0.39063E−3	0.41E−7	0.38E−7	0.77E−7	0.64E−4	0.16E−4	0.12E−10	0.43E−10	0.46E−10

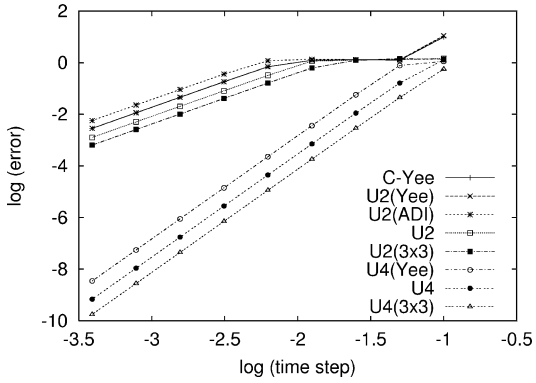

Fig. 5. The data presented in Table 2 plotted on a double logarithmic scale. C-Yee: data obtained by the conventional Yee algorithm [2–4]. The largest time step ($\tau = 0.1$) corresponds to the upperbound on the Courant number (τ/δ) for which the conventional Yee algorithm is stable [2]. Lines are guides to the eye.

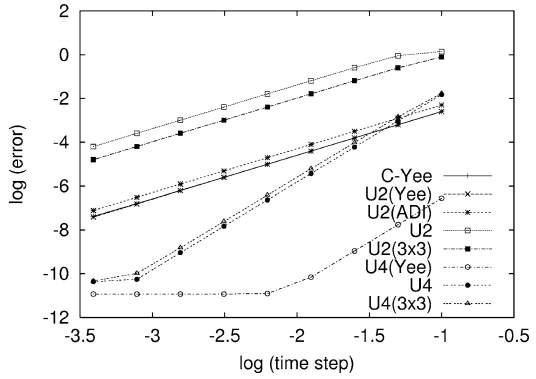

Fig. 6. The data presented in Table 3 plotted on a double logarithmic scale. C-Yee: data obtained by the conventional Yee algorithm [2–4]. The largest time step ($\tau = 0.1$) corresponds to the upperbound on the Courant number (τ/δ) for which the conventional Yee algorithm is stable [2]. Lines are guides to the eye.

also performs better than the conventional Yee algorithm with respect to energy conservation. In Fig. 7 we show results of the time evolution of the total energy of the EM field, for a system of $n = 97$ sites, a mesh size $\delta = 0.1$ and a time step of $\tau = 0.01$. A normalized ($\|\boldsymbol{\Psi}(0)\|^2 = 1$) random initial condition was used.

For the conventional Yee algorithm, the fluctuations of the energy are a factor of ten larger than in the case of the U_2^{Yee} algorithm. As expected on theoretical grounds, the $\widetilde{U}_4$ algorithm (dotted, horizontal line) exactly conserves the energy. The fact that U_2^{Yee} conserves EM-field energy much better than the conventional Yee algorithm also has a considerable impact on the quality of the eigenmode distribution. The latter is obtained by Fourier transformation of $\boldsymbol{\Psi}^{\text{T}}(t) \cdot \boldsymbol{\Psi}(0)$ (see Ref. [11] for more details). In Fig. 8 we show the low-frequency part of the eigenmode distribution of the same system as the one of Fig. 7. It is obvious that there is a significant improvement in the quality of the spectrum if we use U_2^{Yee} instead of the conventional Yee algorithm but for this application $\widetilde{U}_4$ performs much better than the Yee-type algorithms.

From Tables 1–3 it is clear that all time-stepping algorithms analyzed in this paper yield inaccurate results if the time step approaches the stability limit $\tau/\delta = 1$ of the conventional Yee algorithm. However this is no longer the case if the dimensionality of the system increases [26,27]. In particular, using the $\widetilde{U}_4$ algorithm it is possible to compute the density of states of three-dimensional photonic bandgap systems, employing a time step that exceeds the value of the stability limit of the conventional Yee algorithm [26,27].

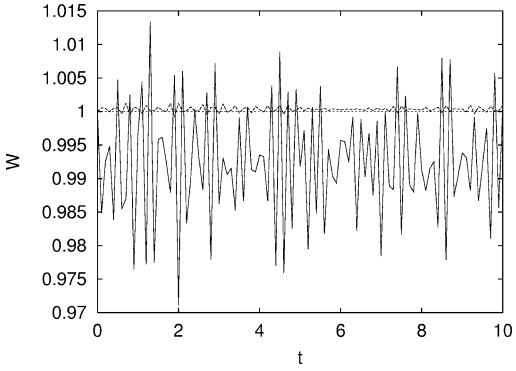

Fig. 7. The energy $W = \boldsymbol{\Psi}^{\mathrm{T}}(t) \cdot \boldsymbol{\Psi}(t)$ of the EM fields as a function of time as obtained by the Yee (solid line), U_2^{Yee} (dashed line), and $\widetilde{U}_4$ (dotted line) algorithm for a 1D cavity of size 48.05 ($n = 97$ mesh points), a mesh size $\delta = 0.1$ and a time step $\tau = 0.01$.

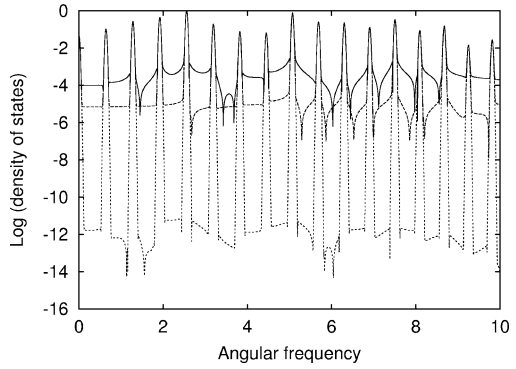

Fig. 8. The eigenvalue distribution (density of states) of the matrix H, as obtained by Fourier transformation of $\boldsymbol{\Psi}^{\mathrm{T}}(t) \cdot \boldsymbol{\Psi}(0)$, for the same system as in Fig. 7. The function $\boldsymbol{\Psi}^{\mathrm{T}}(t) \cdot \boldsymbol{\Psi}(0)$ is sampled at time intervals of 0.1, the total number of samples being 4096. Solid line: conventional Yee algorithm; dashed line: U_2^{Yee} algorithm; dotted line: energy conserving algorithm $\widetilde{U}_4$.

Table 1 suggests that $\widetilde{U}_2$ is the least efficient of the eight FDTD methods: It uses more arithmetic operations than the conventional Yee algorithm and yields errors that are larger than those of the Yee algorithm. However, this conclusion is biased by the choice of the model problem and does not generalize. If the initial EM field distribution is random then, for sufficiently small τ, algorithm $\widetilde{U}_2$ is more accurate than the two second-order accurate Yee-like algorithms, as is clear from the data in Table 2 [43]. Also in this case, for the largest τ in Table 2, the Yee and U_2^{Yee} algorithm are operating at the point of instability, signaled by the fact that the norm of $\boldsymbol{\Psi}(t)$ grows rapidly, resulting in errors that are very large. From Tables 1 and 2 one might conclude that the decomposition that generates Yee-type algorithms yields the least accurate approximations to the time evolution operator, although the difference is not really significant, but, as Table 3 shows, this conclusion would be wrong. If the initial state is a Gaussian wave packet that is fairly broad, the Yee-type algorithms are much more accurate than the unconditionally stable algorithms employed in this paper. From the data in Tables 1–3 we conclude that there is no good reason to use the ADI algorithm (even disregarding the fact that it is slower than the other second-order methods). In general the $\widetilde{U}_2^{3\times3}$ ($\widetilde{U}_4^{3\times3}$) algorithm performs a little better than $\widetilde{U}_2$ ($\widetilde{U}_4$) but the gain is marginal. In contrast to the numerical data presented in Ref. [19], for all algorithms the data of Tables 1–3 clearly agree with the theoretically expected behavior of the error as a function of τ if τ is small enough [44].

Usually if a current source is present we have $\boldsymbol{\Psi}(0) = \mathbf{0}$. Then the one-step algorithm requires K' (sparse) matrix–vector operations ($\boldsymbol{\Psi}' \leftarrow M\boldsymbol{\Psi}$) to compute $\boldsymbol{\Psi}(t)$. For a 1D system the standard Yee, U_2^{Yee} and U_4^{Yee}, $\widetilde{U}_2$, and $\widetilde{U}_4$ algorithms perform (in worst case, without additional optimization), respectively, 1, 3/2, 6, 3/2, and 6 $M\boldsymbol{\Psi}$-operations per time step. The one-step algorithm carries out $K' = 2103$ matrix–vector operations $\boldsymbol{\Psi}' \leftarrow M\boldsymbol{\Psi}$ to complete this simulation. This implies that for all $\tau < t/K'$, the FDTD algorithms will perform more $\boldsymbol{\Psi}' \leftarrow M\boldsymbol{\Psi}$ operations than the one-step algorithm. For the data presented in this paper, this is the case if $\tau < 0.05$ for the conventional Yee algorithm and is always the case for $\widetilde{U}_4$ because the latter uses a factor of 6 more $\boldsymbol{\Psi}' \leftarrow M\boldsymbol{\Psi}$ operations than the conventional Yee algorithm.

6. Conclusion

The answer to the question which of the algorithms is the most efficient one crucially depends on the accuracy that one finds acceptable. Taking the data of Table 1 as an example we see that if one is satisfied with an error of more than 2%, one could use the conventional Yee algorithm. With $\tau = 0.05$ it needs 2000 time steps to find the solution at $t = 100$, close to the $K' = 2103$. Nevertheless we recommend to use the one-step algorithm because then the time-integration error is negligible. The conventional Yee algorithm is no competition for $\widetilde{U}_4$ if one requires an error of less than 1% but then $\widetilde{U}_4$ is not nearly as efficient (by a factor of about 6) as the one-step algorithm. Increasing the dimensionality of the problem favors the one-step algorithm [26,27]. These conclusions seem to be quite general and are in concert with numerical experiments on 1D, 2D and 3D systems [27]. A simple theoretical analysis of the τ-dependence of the error shows that the one-step algorithm is more efficient than any other FDTD method if we are interested in the EM fields at a particular (large) time only [26,27]. This may open possibilities to solve problems in computational electrodynamics that are currently intractable. The Yee-like algorithms do not conserve the energy of the EM fields and therefore they are less suited for the calculation of the eigenvalue distributions (density of states), a problem for which the $\widetilde{\widetilde{U}}_4$ algorithm may be the most efficient of all the algorithms covered in the paper. The main limitation of the one-step algorithm is directly linked to its mathematical justification. The Chebyshev approach requires that H is diagonalizable and that its eigenvalues are real or pure imaginary. The effect of relaxing these conditions on the applicability of the Chebyshev approach is left for future research.

In this paper we have focused entirely on the accuracy of time integration algorithms, using the most simple discretization of the spatial derivatives. In practice it is straightforward, though technically non-trivial, to treat more sophisticated discretization schemes [2,12] by the methodology reviewed in this paper. We also did not touch the subject of different boundary conditions, a very important aspect in real-life applications [2,3]. All results presented in this paper have been obtained for systems that have perfectly reflecting boundaries. For most applications it is necessary to adopt boundary conditions that perfectly absorb, not reflect, all waves that move towards the simulation box boundary [2]. As demonstrated above, the conventional Yee and the unconditionally stable algorithms are particular implementations of the same product formula. Therefore one would expect that Berenger's perfectly matched layer approach [45] should work for the latter class of algorithms as well. In fact, it is easy to see that it will by considering the equivalent, perfectly-matched unaxial-medium formulation [2,46]. In this approach the perfectly-matched material enters through frequency-dependent anisotropic electric permittivity and magnetic permeability tensors. In the Yee-algorithm implementation, the frequency dependence of these tensors is taken into account by introducing auxiliary field variables [2]. The same procedure can be used for the unconditionally stable algorithms as well. The practical implementation of this scheme is left for future research. Extending the one-step algorithm to account for absorbing boundaries also requires further research: The appearance of eigenvalues of H with a non-zero real part changes the rate of convergence of expansion (20) and hence also its computational efficiency.

Acknowledgements

H.D.R. and K.M. are grateful to T. Iitaka for drawing our attention to the potential of the Chebyshev method and for illuminating discussions. This work is partially supported by the 'Nederlandse Stichting voor Nationale Computer Faciliteiten (NCF)', and EC IST project CANVAD.

References

[1] M. Born, E. Wolf, Principles of Optics, Pergamon, Oxford, 1964.
[2] A. Taflove, S.C. Hagness, Computational Electrodynamics—The Finite-Difference Time-Domain Method, Artech House, Boston, 2000.
[3] K.S. Kunz, R.J. Luebbers, Finite-Difference Time-Domain Method for Electromagnetics, CRC Press, 1993.
[4] K.S. Yee, IEEE Trans. Antennas Propagation 14 (1966) 302.
[5] See http://www.fdtd.org.
[6] F. Zheng, Z. Chen, J. Zhang, IEEE Trans. Microwave Theory Techn. 48 (2000) 1550.
[7] T. Namiki, IEEE Trans. Microwave Theory Techn. 48 (2001) 1743.
[8] F. Zheng, Z. Chen, IEEE Trans. Microwave Theory Techn. 49 (2001) 1006.
[9] S.G. Garcia, T.W. Lee, S.C. Hagness, IEEE Trans. Wireless Prop. Lett. 1 (2002) 31.
[10] W. Harshawardhan, Q. Su, R. Grobe, Phys. Rev. E 62 (2000) 8705.
[11] J.S. Kole, M.T. Figge, H. De Raedt, Phys. Rev. E 64 (2001) 066705.
[12] J.S. Kole, M.T. Figge, H. De Raedt, Phys. Rev. E 65 (2002) 066705.
[13] O.P. Gandi, in: A. Taflove (Ed.), Advances in Computational Electrodynamics—The Finite-Difference Time-Domain Method, Artech House, Boston, 1998.
[14] B. Houshmand, T. Itoh, M. Piket-May, in: A. Taflove (Ed.), Advances in Computational Electrodynamics—The Finite-Difference Time-Domain Method, Artech House, Boston, 1998.
[15] M. Suzuki, S. Miyashita, A. Kuroda, Prog. Theor. Phys. 58 (1977) 1377.
[16] H. De Raedt, Comp. Phys. Rep. 7 (1987) 1.
[17] G.D. Smith, Numerical Solution of Partial Differential Equations, Clarendon Press, Oxford, 1985.
[18] This is due to the specific split-up adopted in [10] and not an intrinsic property of the spectral-domain approach.
[19] R. Horváth, I. Faragó, W.H.A. Schilders, Preprint, 2002, http://www.win.tue.nl/analysis/preprints/2002.html.
[20] H. Tal-Ezer, SIAM J. Numer. Anal. 23 (1986) 11.
[21] H. Tal-Ezer, R. Kosloff, J. Chem. Phys. 81 (1984) 3967.
[22] C. Leforestier, R.H. Bisseling, C. Cerjan, M.D. Feit, R. Friesner, A. Guldberg, A. Hammerich, G. Jolicard, W. Karrlein, H.-D. Meyer, N. Lipkin, O. Roncero, R. Kosloff, J. Comp. Phys. 94 (1991) 59.
[23] T. Iitaka, S. Nomura, H. Hirayama, X. Zhao, Y. Aoyagi, T. Sugano, Phys. Rev. E 56 (1997) 1222.
[24] R.N. Silver, H. Röder, Phys. Rev. E 56 (1997) 4822.
[25] Y.L. Loh, S.N. Taraskin, S.R. Elliot, Phys. Rev. Lett. 84 (2000) 2290; Phys. Rev. Lett. 84 (2000) 5028.
[26] H. De Raedt, K. Michielsen, J.S. Kole, M.T. Figge, Chebyshev method to solve the time-dependent Maxwell equations, in: D.P. Landau, et al. (Eds.), Computer Simulation Studies in Condensed-Matter Physics XV, in: Springer Proceedings in Physics, Vol. 90, Springer, Berlin, 2003, pp. 211–215.
[27] H. De Raedt, K. Michielsen, J.S. Kole, M.T. Figge, Phys. Rev. E 67 (2003) 056706; IEEE Trans. Antennas Propagation (in press), http://arxiv.org/abs/physics/0208060.
[28] J.H. Wilkinson, The Algebraic Eigenvalue Problem, Clarendon Press, Oxford, 1965.
[29] M. Abramowitz, I. Stegun, Handbook of Mathematical Functions, Dover, New York, 1964.
[30] H.F. Trotter, Proc. Am. Math. Soc. 10 (1959) 545.
[31] H. De Raedt, B. De Raedt, Phys. Rev. A 28 (1983) 3575.
[32] M. Suzuki, J. Math. Phys. 26 (1985) 601; J. Math. Phys. 32 (1991) 400.
[33] A.J. Chorin, T.J.R. Hughes, M.F. McCracken, J.E. Marsden, Comm. Pure Appl. Math. XXXI (1978) 205.
[34] H. Kobayashi, N. Hatano, M. Suzuki, Physica A 211 (1994) 234.
[35] H. De Raedt, K. Michielsen, Comp. in Phys. 8 (1994) 600.
[36] A. Rouhi, J. Wright, Comp. in Phys. 9 (1995) 554.
[37] B.A. Shadwick, W.F. Buell, Phys. Rev. Lett. 79 (1997) 5189.
[38] M. Krech, A. Bunker, D.P. Landau, Comp. Phys. Comm. 111 (1998) 1.
[39] P. Tran, Phys. Rev. E 58 (1998) 8049.
[40] K. Michielsen, H. De Raedt, J. Przeslawski, N. Garcia, Phys. Rep. 304 (1998) 89.
[41] H. De Raedt, A.H. Hams, K. Michielsen, K. De Raedt, Comp. Phys. Comm. 132 (2000) 1.

[42] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, Numerical Recipes, Cambridge, New York, 1986.
[43] This also explains why the unconditionally stable algorithms $\widetilde{U}_2$ and $\widetilde{U}_4$ yield more accurate eigenvalue distributions than the conventional Yee algorithm [11].
[44] The data for small τ, obtained from the Yee algorithm for the case of a current source is an exception. This is due to the ambiguities mentioned in Section 5.
[45] J.P. Berenger, J. Comp. Phys. 114 (1994) 185.
[46] Z.S. Sacks, D.M. Kingsland, R. Lee, J.F. Lee, IEEE Trans. Antennas Propagation 43 (1995) 1460.