GPU-Accelerated Efficient Implementation of FDTD Methods with Optimum Time-Step Selection

Theodoros T. Zygiridis
Dept. of Informatics and Telecommunications Engineering
Univ. of Western Macedonia
Karamanli and Lygeris, Kozani, 50100, Greece
tzygiridis@uowm.gr

Nikolaos V. Kantartzis, Theodoros D. Tsiboukis
Dept. of Electrical & Computer Engineering
Aristotle Univ. of Thessaloniki
Thessaloniki, 54124, Greece
kant@auth.gr, tsibukis@auth.gr

***Abstract*—The efficient combination of uneven space-time orders in finite-difference time-domain (FDTD) algorithms is the subject of this paper. Operating such schemes close to the stability limit leads to poor performance and low convergence rates. Based on accuracy considerations, we provide an estimate of the optimum time-step size that improves errors in a mean-value sense. To deal with the augmentation of the required iterations, the parallel implementation of the FDTD techniques on graphics processing units is pursued, ensuring faster code executions.**

***Index Terms*—finite-difference time-domain method, high-order algorithms, time stepping, GPU computing.**

I. Introduction

The implementation of finite-difference time-domain (FDTD) schemes [1] with second-order temporal and $2N$-th-order spatial accuracy (hereafter labeled $(2,2N)$ methods) calls for special care, so that performance is not compromised. Unlike Yee's algorithm, only time steps smaller than the stability limit can render temporal accuracy comparable to that in space [2]. A closed-form expression for the proper time-step size in the two-dimensional $(2,4)$ case is given in [3].

In this paper we provide an estimation of the optimum time-step size for $(2,2N)$ FDTD schemes in three-dimensional (3D) formulation. The proposed values are derived by requiring the vanishing of the average discretization error, when the latter is described by the numerical dispersion relation. To deal with the unavoidable increase of the temporal sampling density, the algorithms are also parallelized on graphics processing units (GPUs). Thanks to their many-core architecture, the latter result in drastic reduction of the required computing times [4], [5]. Therefore, combining the optimum time-stepping with GPU implementations that exploit the rich data parallelism, leads to efficient realizations of the $(2,2N)$ techniques.

II. Determination of the Optimum Time Step

$(2,2N)$ FDTD schemes incorporate the second-order leapfrog approach in time, and spatial operators of the form

$$\left.\frac{\partial f}{\partial u}\right|_m \simeq \frac{1}{\Delta u}\sum_{i=1}^{N} C_i^{(N)}\left(f|_{m+\frac{2i-1}{2}} - f|_{m-\frac{2i-1}{2}}\right) \quad (1)$$

where

$$C_i^{(N)} = \frac{(-1)^{i+1}[(2N-1)!!]^2}{4^{N-1}(N+i-1)!\,(N-i)!(2i-1)^2} \quad (2)$$

The dispersion relation in lossless space is described as

$$T^2 = \tilde{S}_x^2 + \tilde{S}_y^2 + \tilde{S}_z^2 \quad (3)$$

where

$$T = \frac{1}{c_0 \Delta t}\sin\left(\frac{\omega \Delta t}{2}\right) \quad (4)$$

$$S_u = \frac{1}{\Delta u}\sum_{i=1}^{N} C_i^{(N)} \sin\left(\frac{2i-1}{2} k \cos\gamma_u \Delta u\right) \quad (5)$$

The tilde over terms S_u, $u = x, y, z$, indicates their dependence on the numerical wavenumber $\tilde{k}$. In addition, $\cos\gamma_x = \sin\theta\cos\phi$, $\cos\gamma_y = \sin\theta\sin\phi$, and $\cos\gamma_z = \cos\theta$ are the directional cosines of the plane-wave propagation direction.

Our goal is to determine the size of Δt that balances errors emanating from discretizations in space and time. Given that spatial errors are lower than temporal ones when $N > 1$, time steps smaller than the stability limit are required. It is reminded that stability is guaranteed when

$$\Delta t^{(N)} \le \frac{1}{c_0 \sum_{i=1}^{N} (-1)^{i-1} C_i^{(N)} \sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}}} \quad (6)$$

As the numerical wavenumber can not be found analytically for (3), the dispersion relation may serve as an error estimator, provided that we set $\tilde{k} = k = \omega / c_0$. Given the dependence on θ and ϕ, we select that (3) is satisfied in a mean sense:

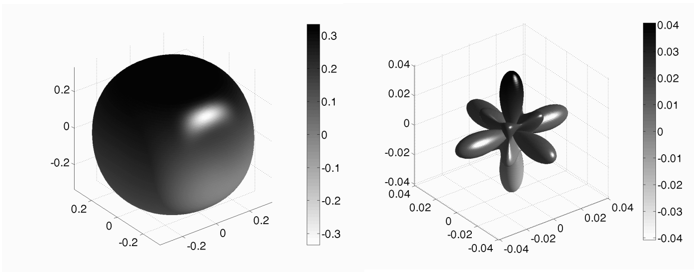
$$\left\langle T^2 \right\rangle = \left\langle S_x^2 \right\rangle + \left\langle S_y^2 \right\rangle + \left\langle S_z^2 \right\rangle \quad (7)$$

where $\langle . \rangle$ denotes the mean value over $\theta \in [0,\pi]$ and $\phi \in [0,2\pi]$ (note that now the tildes are dropped). For instance,

$$\left\langle S_z^2 \right\rangle = \frac{1}{4\pi}\int_0^{2\pi}\int_0^{\pi} S_z^2 \sin\theta \, d\theta \, d\phi \quad (8)$$

Taking into account the following identity,

$$\int_0^{2\pi}\int_0^{\pi} \cos(\alpha\cos\theta)\sin\theta \, d\theta \, d\phi = 4\pi\frac{\sin\alpha}{\alpha} \quad (9)$$



(a) (b)

Figure 1. Error in phase velocity for the (2,4) FDTD scheme when $N_x = N_y = N_z = \lambda/10$: (a) $Q = 6/7$, (b) $Q = 0.23679$ (calculated from (12)).

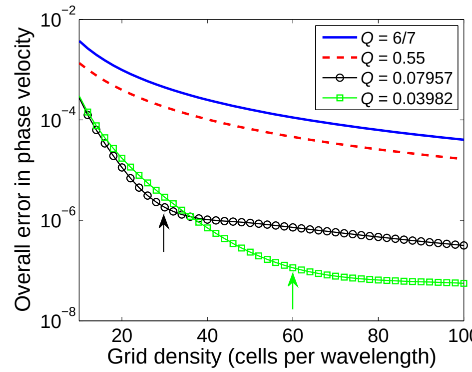
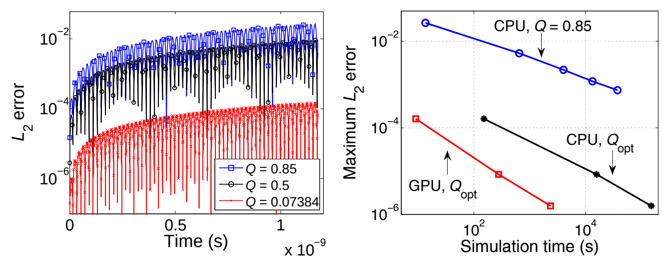


Figure 2. Overall error versus mesh resolution. The two smaller values of Q correspond to optimum choices at 30 and 60 cells per wavelength.



(a) (b)

Figure 3. (a) Evolution of the L_2 error in time for the cavity problem. The smallest time step corresponds to the optimum choice. (b) Maximum L_2 error versus simulation time for CPU and GPU implementations.

we obtain

$$\langle S_z^2 \rangle = \frac{1}{2\Delta z^2} \sum_{i=1}^{N} \left(C_i^{(N)} \right)^2 \left[1 - \frac{\sin\left((2i-1)k\Delta z\right)}{(2i-1)k\Delta z} \right] + \frac{1}{\Delta z^2} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left\{ C_i^{(N)} C_j^{(N)} \times \left[\frac{\sin[(i-j)k\Delta z]}{(i-j)k\Delta z} - \frac{\sin[(i+j-1)k\Delta z]}{(i+j-1)k\Delta z} \right] \right\} \quad (10)$$

Due to symmetry, $\langle S_x^2 \rangle$ and $\langle S_y^2 \rangle$ can be obtained directly from (10). Also, note that T in (3) is constant, hence $\langle T \rangle = T$.

To solve (7) approximately, the temporal quantity is replaced by the first terms of its Maclaurin series. Specifically,

$$\frac{1}{k^2} T^2 \simeq \frac{1}{4} - \frac{1}{48} \left(Q \frac{2\pi/(N_x N_y N_z)}{\sqrt{\sum_{(u,v)} 1/(N_u N_v)^2}} \right)^2 \quad (11)$$

where $N_u = \lambda/\Delta u$ denotes grid density along u-axis, and $Q = c_0 \Delta t (\Delta x^{-2} + \Delta y^{-2} + \Delta z^{-2})^{1/2}$ corresponds to the Courant number. Hence, (7) is replaced by an easily solvable equation, from which the optimum time-step is determined by

$$Q = \frac{\sqrt{\sum_{(u,v)} 1/(N_u N_v)^2}}{2\pi/(N_x N_y N_z)} \sqrt{12 - 48\Lambda_{xyz}^2} \quad (12)$$

where $\Lambda_{xyz}^2 = k^{-2}[\langle S_x^2 \rangle + \langle S_y^2 \rangle + \langle S_z^2 \rangle]$.

Fig. 1 displays the phase-velocity error in the case of the $(2, 4)$ scheme. Time-differentiation errors dominate when the stability limit ($Q = 6/7$) is selected, whereas significant improvement is accomplished with the proposed time-step size. Furthermore, the effect of smaller time steps over a wide frequency band is shown in Fig. 2, for two design frequencies.

III. GPU- Implemented Numerical Test

We simulate a $2 \times 1 \times 1$ cm^3 air-filled cavity with perfectly conducting walls, where the $(1, 1, 1)$ mode is excited at 25.963 GHz. Starting from an initial $63 \times 31 \times 31$ grid and duration of 2000 time steps, we perform simulations with refined mesh densities (we have set $N = 2$). Apart from the serialized CPU code, a GPU implementation is also pursued. The latter features two main kernels, one for electric- and one for magnetic-field updating, and two additional kernels that apply symmetric/antisymmetric conditions at the domain's ends. Threads are organized in $32 \times 4 \times 4$ blocks, which form a simple one-dimensional grid (the available Tesla C1060 card does not support 3D grids). Fig. 3(a) displays the L_2 error (with respect to E_x) for the coarse mesh. As seen, the proposed time step causes error reduction by more than 160 times, compared to the case $Q = 0.85$. Fig. 3(b) verifies that: a) the choice of the optimum time step not only corrects errors, but also improves the efficiency, despite the larger number of iterations, and b) the GPU implementation reduces simulation times decisively. It is verified that acceleration by more than 50 times is gained for the $127 \times 63 \times 63$ grid. Note that our initial code is not optimized (e.g. the fast shared memory is not exploited). Despite that, even the mere use of the slower global memory renders the proposed approach highly efficient.

References

[1] A. Taflove and S. C. Hagness, *Computational electrodynamics: The finite-difference time-domain method*, Third Edition, Artech House, Norwood MA, 2005.

[2] K. L. Shlager and J. B. Schneider, "Comparison of the dispersion properties of several low-dispersion finite-difference time-domain algorithms," *IEEE Trans. Antennas Propag.*, vol. 51, no. 3, pp. 642-653, Mar. 2003.

[3] T. T. Zygiridis, "Optimum time-step size for 2D (2, 4) FDTD method," *Electron. Lett.*, vol. 47, no. 5, pp. 317-319, Mar. 2011.

[4] N. Godel, S. Schomann, T. Warburton, and M. Clemens, "GPU accelerated Adams-Bashforth multirate discontinuous Galerkin FEM simulation of high-frequency electromagnetic fields," *IEEE Trans. Magn.*, vol. 46, no. 8, pp. 2735-2738, Aug. 2010.

[5] S. Nakata, Yu Takeda, N. Fujita, and S. Ikuno, "Parallel algorithm for meshfree radial point interpolation method on graphics hardware," *IEEE Trans. Magn.*, vol. 47, no. 5, pp. 1206-1209, May 2011.