Computational complexity of artificial boundary conditions for Maxwell's equations in the FDTD method

Mikhail Osintcev[1,*], **Semyon Tsynkov**[1]
[1]Department of Mathematics, North Carolina State University, Raleigh, USA
*Email: mishaosintsev@gmail.com

Abstract

We compare several approaches for handling the artificial outer boundaries that can be implemented with the standard FDTD method in 3D. Our goal is to obtain the asymptotic estimates of computational complexity for each class of methods and corroborate those with numerical results so as to show the advantages and disadvantages of the various methodologies.

Keywords: computational electromagnetics, unbounded domains, numerical efficiency.

1 Local artificial boundary conditions

There is a number of established artificial boundary conditions (ABCs) for Maxwell's equations that are derived using asymptotic considerations. The low-order ABCs (e.g., Sommerfeld, Higdon, Betz-Mittra, or Mur) are widely used in FDTD methods since 1970's [1]. These ABCs are easy to implement (independently at each boundary node), but their accuracy is not satisfactory in most cases. Furthermore, these ABCs may suffer from a deterioration of performance in long-time computations [2]. The overall complexity is

$$C_0(N) = N^3 a, \tag{1}$$

where N is the grid dimension in one direction and $a = \text{const}$ depends on the implementation.

A number of more accurate local high-order ABCs (e.g., Givoli-Neta or Hagstrom-Warburton [3]) have also been proposed, but in most cases the treatment of corners is difficult and no 3D implementation has been described by the authors. The exception is the recent development of double absorbing boundaries (DAB) by LaGrone and Hagstrom [4]. It provides a nearly uniform theoretical accuracy over long time intervals, yet in our experiments we observed a rapid growth of the error for the magnetic field when the solution was driven by a non-solenoidal current. The algorithm of DAB is rather sophisticated, but we assume that the computational complexity can still be estimated in the form similar to (1).

2 Perfectly matched layer

Another group of popular and efficient approaches for truncating the unbounded regions in EM simulations are perfectly matched layers (PMLs). A PML is an absorbing layer that surrounds the computational domain. The PML and the computational domain are usually discretized on the same grid, hence the computational complexity of a FDTD/PML implementation is as follows:

$$C_{\text{PML}}(N) = (1+2\nu)^3 N^3 a, \quad \nu = P/N, \tag{2}$$

where P is the number of nodes in the PML. The value of ν may be relatively small for large grid dimensions N. The PMLs usually provide a low level of spurious reflections and their implementation is straightforward in FDTD. Yet they may be prone to error growth in long-time simulations [5, 6]. In particular, our computations show that the performance depends on the type of the source (antenna current) that drives the EM field. For solenoidal currents the solution with a PML is usually stable yet for non-solenoidal currents it may deteriorate rapidly.

3 Lacunae-based time marching

The lacunae-based time marching can be used to mitigate the long-time deterioration of the PML [6, 7]. This approach is based on the following property of Maxwell's equations in vacuum (the Huygens' principle): provided that the currents are compactly supported in both space and time, the propagating electromagnetic waves have sharp aft fronts. The original problem is decomposed into a series of partial subproblems driven by compactly supported partial currents. The latter are obtained by a smooth partition of the original currents. The computational complexity of this algorithm is

$$C_{\text{Lac-PML}} = \frac{(1+2\nu)^3}{(1-\mu)} N^3 a, \tag{3}$$

where $0 < \mu < 1$ is the overlap between the consecutive partial currents. If the overlap is small, then the computational complexity (3) tends to (2). The original lacunae-based time marching requires divergence-free sources [6]. Its extension based on quasi-lacunae [7] removes this limitation yet keeps the asymptotic complexity (3) unchanged. The most recent work [8] reduces quasi-lacunae to classical lacunae, which improves the performance and enables the proof of a temporally uniform error bound.

The lacunae-based time marching can also be used as a standalone closure at the artificial outer boundary [9]. The idea is to take a sufficiently large auxiliary domain beyond the actual computational domain so that the reflections off the outer boundary of this auxiliary domain won't reach the computational domain by the time the latter falls into the lacuna. This algorithm is considerably more expensive than the lacunae-based algorithm with PML. Its complexity is given by the same expression (3) but with the quantity $(1+2\nu)^3$ replaced with a large constant, on the order of 150. The advantage of this algorithm, however, is that it does not require any special treatment of the artificial outer boundary per se. The outgoing waves propagate into the auxiliary region and then get canceled there once the main computational domain falls inside the lacuna. As such, this algorithm is provably free from any error associated with the domain truncation.

4 Computational results

We use our serial FORTRAN FDTD code to compare the computational complexity of the various boundary conditions numerically. Consider a cubic domain with side $l = 10$ and the propagation speed $c = 1$. Take a $100 \times 100 \times 100$ grid for the FDTD scheme, with the spatial size $h = 0.1$ and the time step $\Delta t = 0.03(3)$. We compute on a 16-core Linux server, Intel Xeon CPU E5-2698 v3 with 2.30 GHz. Table 1 shows the time T for an update of one time step, the value of a, and the maximum relative error ε for each ABC. Note that for more accurate ABCs the value of ε is dominated by the discretization error rather than reflections. Finally, we include the data for the original implementation of the DAB [4], which is written in C++ and therefore does not allow for a direct comparison.

ABC	T (sec)	a	ε
1	0.657	6.565×10^{-7}	17.08×10^{-2}
2	0.667	6.665×10^{-7}	5.44×10^{-2}
3	0.805	4.656×10^{-7}	3.56×10^{-2}
4	1.301	1.506×10^{-7}	3.56×10^{-2}
5	26.709	3.459×10^{-8}	3.56×10^{-2}
6	0.6669	8.551×10^{-7}	3.56×10^{-2}

Table 1: Comparison of the various boundary conditions in a serial implementation. The ABC types: 1 - Sommerfeld ABC, 2 - Higdon ABC, 3 - Uniaxial 10-point PML, 4 - Lacunae with Uniaxial 10-points PML, 5 - Lacunae without PML, 6 - DAB with 4 recursions.

Acknowledgment

Work supported by US ARO, grants W911NF-16-1-0115 and W911NF-14-C-0161, and US–Israel BSF, grant 2014048.

References

[1] S. D. Gendey, *Introduction to the finite-difference time-domain (FDTD) method for electromagnetics*. Morgan & Claypool Publishers, San Rafael, CA, 2011.

[2] O. M. Ramahi, *IEEE Trans. Antennas Propagat.*, 47:593-599, 1999.

[3] D. Givoli, *Wave Motion*, 39:319-326, 2004.

[4] J. LaGrone and T. Hagstrom, *J. Comput. Phys.*, 326:650-665, 2016.

[5] S. Abarbanel, D. Gottlieb, and J. Hesthaven, *J. Sci. Comput.*, 17(14):405-422, 2002.

[6] H. Qasimov and S. Tsynkov, *J. Comput. Phys.*, 227:7322-7345, 2008.

[7] S. V. Petropavlovsky and S. V. Tsynkov, *J. Comput. Phys.*, 231:558-585, 2012.

[8] S. V. Petropavlovsky and S. V. Tsynkov, *J. Comput. Phys.*, 336C:1-35, 2017.

[9] S. V. Tsynkov *J. Comput. Phys.*, 199(1):126-149, 2004.