

Reducing the Setup Time of a One-Step FDTD Method

Dmitry A. Gorodetsky, *Student Member, IEEE*, and Philip A. Wilsey, *Member, IEEE*
Department of ECECS, University of Cincinnati, Cincinnati, OH 45221-0030 USA

Abstract—As society places greater demands on technology, it becomes increasingly important to be able to simulate that technology. While simulation tools can be very sophisticated and capable of handling very complex devices and structures, the cost paid in computation time can be prohibitive. To reduce this time, a number of one-step schemes have been developed, but these schemes suffer from a setup time that can outweigh the benefits of the scheme. We examine parallel processing as one means to reduce the setup time. By approaching FDTD from the modal analysis perspective we offer an alternative to implement a one-step scheme for FDTD with coarse-grain parallelization.

Keywords: FDTD methods, Design automation, Reduced order systems, Macromodeling, High Performance Computing.

I. INTRODUCTION

The finite-difference time-domain method (FDTD) is a well-known technique to solve Maxwell's equations in the time domain by evolving their discretized counterparts in space and time [1]-[3]. The time-stepping is recursive because the field values are completely dependent on the values at the previous point in time. In many cases simulation proceeds for thousands of iterations until some measure of stability is attained. With the methods proposed in this paper we discuss how to distribute the simulation efficiently on a large number of processors and to overcome the speedup limitations imposed by the communication/computation ratio [4]-[7].

By examining the six FDTD equations we can easily construct a state transition matrix for an enclosed zero-input response region (model) simulated by FDTD [10]. After the matrix is constructed, its modal decomposition is obtained by computing the eigenvalues and the corresponding eigenvectors. The model order reduction of this algorithm is done by discarding the high-frequency eigenmodes that correspond primarily to the numerical artifacts of FDTD. The dominant contribution of this approach is the decoupling of FDTD in time. After the modal decomposition is effected, the response of the algorithm can be obtained for *any point in time* without calculating the response at any previous times. If N is the total number of unknowns in the state transition matrix, the computational complexity is between $O(N)$ and $O(N^2)$ per time step depending on how much order reduction took place. The modal decomposition also gives a clue to the frequency response of the region [10]. The disadvantage of this technique however is the time that it takes to compute the modal decomposition, which can have a complexity of $O(N^3)$ [13],[14]. In many cases this time can be significantly larger than the time taken by FDTD to compute the results of the region. As it stands, this method is only useful in cases where a very large number of iterations is involved. Alternative approaches are discussed in [8] and [9].

In this paper we concentrate on the state transition matrix approach originally suggested by Chen [10]. By utilizing matrix algebra we show how the one-step algorithm can be constructed. The path through which the construction takes place is examined and alternatives are suggested to decrease the computation time with a parallel implementation. Techniques to utilize steady-state and transient excitation with this method are also discussed.

II. MODAL DECOMPOSITION

In order to construct the state transition matrix the conventional FDTD equations can be written in matrix form as follows [10]:

$$\begin{aligned} \mathbf{E}(n) &= \mathbf{D}_1 \mathbf{H}(n-1/2) + \mathbf{G}_1 \mathbf{E}(n-1) \\ \mathbf{H}(n+1/2) &= \mathbf{D}_2 \mathbf{E}(n) + \mathbf{G}_2 \mathbf{H}(n-1/2) \end{aligned} \quad (1)$$

With these equations, the state transition matrix is written as:

$$\mathbf{A} = \begin{bmatrix} \mathbf{G}_1 & \mathbf{D}_1 \\ \mathbf{D}_2 \mathbf{G}_1 & \mathbf{D}_2 \mathbf{D}_1 + \mathbf{G}_2 \end{bmatrix} \quad (2)$$

The \mathbf{D} matrices represent the discrete version of the curl operators on the grid whereas the \mathbf{G} matrices select the field points that are used to find the discrete derivative with respect to time. The \mathbf{D} and \mathbf{G} matrices are very sparse (1-4 entries per row). The state transition matrix that results is very sparse and non-symmetric. The first phase in the analysis involves computing the eigenvalues and eigenvectors represented by the matrices $\mathbf{E} = \text{diag}[\lambda_1, \lambda_2, \dots, \lambda_N]$ and $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N]$, respectively. The time-stepping is done during the second phase, and can take place after the completion of the first phase, as described below.

A method to express the state $\mathbf{Q}(n)$ as a superposition of eigenvectors and to solve for the zero-input response was discussed in [10]. The initial state is written as a sum of eigenvectors \mathbf{u}_m modulated by the coefficients a_m :

$$\mathbf{Q}(0) = \sum_{m=1}^N a_m \mathbf{u}_m \quad (3)$$

Using (3), the evolution in time can be expressed as:

$$\mathbf{Q}(n) = \sum_{m=1}^N (\lambda_m)^n a_m \mathbf{u}_m \quad (4)$$

The advantage of (4) is that now time-stepping is de-coupled and we can obtain the solution at any time-step without any knowledge of the history of the solution states. If no eigenmodes are discarded, the result of (4) is completely identical to the result of the conventional FDTD algorithm because the state \mathbf{Q} is always represented with no loss of data. Discrepancy may occur due to numerical noise that occurs as a result of round-off error, but in our experiments this numerical noise did not appear to be significant. Parallelization of (4) is straightforward and good results can be achieved because no communication is involved. Below we discuss how various excitation scenarios can be adapted to work with the modal decomposition method. That is followed by a discussion on how to parallelize the less complex algorithm. The discussion on order reduction can be found in [10].

A. TRANSIENT EXCITATION

In Yee's seminal paper that gave rise to the FDTD method, he describes the propagation and scattering of a single pulse [1]. In order to begin the propagation, it is enough to specify the state of the electromagnetic field at a single snapshot in time. The propagation is then automatically handled by the FDTD algorithm. If the pulse is located on the boundary of a model, then it can be treated as just another input. However, if the pulse is initially located somewhere *within* the model, then we have a *non-zero* state system that must be converted to one with *zero* state. This can be done if we assume that the system (model) is *completely controllable*. In this case the initial state can be expressed as:

$$\mathbf{Q}(0) = \sum_{m=1}^N a_m \mathbf{u}_m \quad (5)$$

We end up with a set of non-zero coefficients a_m , which can be obtained from:

$$\mathbf{a}_m = \mathbf{U}^{-1} \mathbf{Q}(0) \quad (6)$$

Time stepping can then begin as follows:

$$\mathbf{Q}(n) = \sum_{m=1}^N (\lambda_m)^n a_m \mathbf{u}_m \quad (7)$$

B. STEADY STATE EXCITATION

According to Taflove, the excitation method suggested by Yee suffers from excessive elongation of the computational domain for a long-duration source [11]. The improvement is to assign the desired time function (sinusoidal, Gaussian, etc) to Electric field components in the FDTD domain. For three-dimensional grids with plane wave excitation these field components are the same across an entire plane.

To use this approach with our method, assume the excitation is in the form:

$$E_z(i = i_s, j, k, n\Delta t) = E_0 e^{j\omega n\Delta t} \quad (8)$$

The first step is to solve (3) with every component of \mathbf{Q} equal to E_0 . This is the situation for $n=0$. We will end up with a set of non-zero coefficients a_m . Taking the real part of (6), every successive input will be equal to the multiplication of $\sin(\omega n\Delta t)$ by E_0 . Since E_0 is written as a superposition of eigenvectors, each of the a_m must be multiplied by the sinusoid as follows:

$$\mathbf{Q}(n) = \sum_{m=1}^N (\lambda_m)^n a_m \mathbf{u}_m \operatorname{Re}(e^{j\omega n\Delta t}) \quad (9)$$

With this approach, the number of multiplications is reduced because (3) only has to be solved once.

C. PARALLELIZATION

Here we discuss the parallelization of the time-stepping part of the algorithm, represented by (5) and (7). One way to distribute the computations is to split up the vector \mathbf{U} so that each processor is assigned a specific region of the FDTD domain. In this case each processor can find the solution in its region at any point in time, independent of the states of the surrounding processors. If the solution is computed in consecutive time steps then this algorithm performs slower than conventional 2nd order FDTD even for small values of P. However, as mentioned earlier, parallelizing FDTD involves frequent communication which significantly limits the number of processors that can be used. In the proposed algorithm

however, this limitation is non-existent and its parallel version can exhibit near-unity speedup efficiencies for a large number of processors.

III. SETUP COSTS

In MATLAB, the modal decomposition is performed with the following command:

$$[U, E] = \text{eig}(A) \quad (10)$$

The `eig` command calls a high-level LAPACK driver routine `DGEEV`, that computes all the eigenvalues and eigenvectors of A via several lower-level computational routines. The routine `DGEEV` follows the standard two-phase approach for eigenmodal decomposition [13]. First the matrix A is reduced to a condensed form, upper Hessenberg in this case because it is non-symmetric. An optional balancing step is performed as well [16]. The reduction is performed via a sequence of Householder transformations [13]. At the end of the process, we end up with the matrix H , which is upper Hessenberg:

$$H = P_{n-2} P_{n-1} \dots P_2 P_1 A P_1 P_2 \dots P_{n-1} P_{n-2}, \quad (11)$$

where P represents Householder transformations. The reduction of A to upper Hessenberg form is in general an $O(N^3)$ process, but it exhibits a high-degree of parallelism and can be expressed as a block algorithm [14].

In the second phase the upper Hessenberg matrix computed above is reduced to Schur form T via the QR iteration method. This is expressed as [15]:

$$H = STS^{\text{TR}} \quad (12)$$

The eigenvalues are contained on the diagonal of T . The QR iteration method belongs to a group of iterative methods with $O(N^2)$ complexity that are not easy to parallelize. Other competitive methods for eigenvalue computation are reduction to nonsymmetric tridiagonal form, Jacobi's method, Hessenberg divide-and-conquer, and spectral divide and conquer [13].

The spectral divide- and-conquer is very useful for FDTD because it has the capability to find only those eigenvalues in which the user is interested without wasting time with those that will be discarded. Furthermore, this method is easier to parallelize than the QR approach. Spectral divide-and-conquer is an alternative mechanism for the standard two-phase approach. One of the dominant contributors to the computation time with the spectral divide-and-conquer method is the sign function. Computation of the sign function directly is usually avoided and various iterative schemes such as Newton's method with scaling are used [17]. When computing the sign function, the real coordinate of the complex region beyond which eigenvalues are to be calculated must be specified. Specifying this coordinate is a non-trivial task because it may affect the convergence rate of the Newton's method. Slow convergence can result if eigenvalues lie too close to the boundary of the region [19]. In general, the spectral divide-and-conquer requires kN^3 flops, with k proportional to the number of eigenmodes in the specified region.

IV. RESULTS

A parallel plate waveguide structure of varying x -dimensions and constant y,z dimensions of 20,9 cells respectively was simulated with the modal decomposition algorithm. The eigenvectors were eliminated according to the $p=0.1$ criterion presented in [10]. The waveguide was excited with a constant plane wave source of 10 GHz. We were concerned with the characteristics of the eigenvalues and how those characteristics vary with the model size.

We investigated the dependence of the number of remaining modes on the x -dimension of the modeled region and the results are shown in Fig. 1. These results indicate that as the model size is increased, the proportion of the number of distinct modes required to represent the propagating field within the waveguide to the total number of modes decreases. Consequentially, when the spectral divide-and-conquer algorithm is utilized, the k in its kN^3 dependence will be reduced with increasing model size.

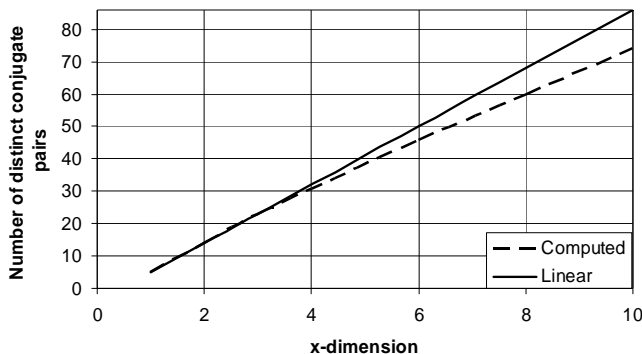


Fig. 1. The dependence of the mode quantity on the x -dimension of the model. It can be seen that the number of non-discarded modes begins to saturate as the x -dimension is increased.

V. CONCLUSION

In this paper we discussed several approaches to parallelizing the one-step version of the FDTD algorithm. We showed how the algorithm can be used in step-by-step mode and how to parallelize that version of the algorithm as well. After overcoming the expensive setup step, we gain a lot of information that affords us a lot of flexibility in calculating the solution at any time step. One approach to parallelizing the modal decomposition is based on calculating only those modes that will not be discarded later. It was shown that the performance of this approach will improve with increasing model size. Another fruitful method is to take advantage of the sparsity of the state transition matrix. In general, the improvement of the modal method over the conventional FDTD will depend on the total number of time steps performed with the FDTD algorithm multiplied by the total number of processors over which the modal method is distributed. Future work will involve a study of the desirable characteristics of the one-step algorithm that will allow it to be parallelized with the greatest possible savings of simulation time.

VI. REFERENCES

- [1] K. S. Yee, "Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media," *IEEE Transactions on Antennas and Propagation*, vol. 14, no. 3, pp. 302-307, 1966.
- [2] C. A. de Moura, "Non-Sequential numerical algorithms for time marching models," *RT06/99, IC-UFF*, July 1999.
- [3] A. C. Cangellaris, "Time-Domain finite methods for electromagnetic wave propagation and scattering," *IEEE Tran. Magn.*, vol. 27, no. 5, pp. 3780-3785, 1991.
- [4] U. Andersson, *Time-Domain Methods for the Maxwell Equations*, Ph.D. Dissertation, Department of Numerical Analysis and Computer Science, Royal Institute of Technology, Stockholm, Sweden, February 2001.
- [5] J. Wang, O. Fujiwara, S. Watanabe; Y. Yamanaka, "Computation with a parallel FDTD system of human-body effect on electromagnetic absorption for portable telephones," *Microwave Theory and Techniques, IEEE Transactions on*, Vol.52, Iss.1, pp. 53-58, Jan. 2004.
- [6] Qing-Xin Chu, Ka-Fai Chan, Chi-Hou Chan, "Parallel FDTD analysis of active integrated antenna array," *IEEE Antennas and Propagation Society International Symposium*, vol.1, pp. 628-631, 2002.
- [7] D. A. Gorodetsky and P. A. Wilsey, "Innovative approaches to parallelizing finite-difference time-domain computations," *IEEE Workshop on Direct and Inverse Problems in Electrodynamics*, 2005.
- [8] A. Fijany, M. A. Jensen, Y. Rahmart-Samii, and J. Barhen, "A massively parallel computation strategy for FDTD: time and space parallelism applied to electromagnetic problems," *IEEE Trans. on Antennas and Propagation*, vol. 43, no. 12, pp. 1441-1449, 1995.
- [9] H. De Raedt, M. Michielsens, J. S. Kole, and M. T. Figge, "One step finite-difference time-domain algorithm to solve the Maxwell equations," *Physical Review*, vol. 67, no. 5, pp. 056706-1-12, 2003.
- [10] Z. Chen and P. P. Silvester, "Analytic solutions for the finite-difference time-domain and transmission-line-matrix methods," *Microwave and Optical Tech. Lett.*, vol. 7, no.1, pp. 5-8, 1994.
- [11] A. Taflove, *Computation of the Electromagnetic Fields and Induced Temperatures within a Model of the Microwave-Irradiated Human Eye*, Ph.D. Dissertation, Department of Electrical Engineering, Northwestern University, Evanston, IL, 1975.
- [12] K. S. Kunz, R. J. Luebbers, *The Finite Difference Time Domain Method for Electromagnetics*, Boca Raton: CRC Press, 1993.
- [13] J. W. Demmel, M. T. Heath, and H. A. van der Vortst, *Parallel numerical linear algebra*, in Acta Numerica 1993, Cambridge, 1993, Cambridge University Press, pp. 111-197.
- [14] J. J. Dongarra, S. J. Hammarling, and D. C. Sorensen, "Block Reduction of Matrices to Condensed Forms for Eigenvalue Computations," *Journal of Computational and Applied Mathematics* vol. 27, no. 1-2, pp.215-227, September 1989.
- [15] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK User's Guide* (http://www.netlib.org/lapack/lug/lapack_lug.html), Third Edition, SIAM, Philadelphia, 1999.
- [16] The MathWorks Inc., *MATLAB Function Reference*, vol. 1: A-E (http://www.mathworks.com/access/helpdesk/help/pdf_doc/matlab/refbook.pdf), 2004.
- [17] Z. Bai and J. Demmel, "Design of a Parallel Nonsymmetric Eigenroutine Toolbox, Part I," Computer Science Division Technical Report, UCB/CSD-92-718, University of California, Berkeley, California, 1992.
- [18] Z. Bai, "Progress in the numerical solution of the nonsymmetric eigenvalue problem," *Journal of Numerical Linear Algebra with Applications*, vol. 2, pp. 219-234, 1995.
- [19] Z. Bai, J. Demmel, and M. Gu, "An inverse free parallel spectral divide and conquer algorithm for nonsymmetric eigenproblems," *Numer. Math.*, vol. 76, pp. 279-308, 1997.